A Machine Learning approach to determine Stellar Atmospheric Parameters using Spectral Data

Joana Catarina Santos Leite
Mestrado em Ciência de Computadores
Departamento de Ciência de Computadores
2022/2023

Orientador
Luís Lopes, Professor Associado, Faculdade de Ciências da Universidade do Porto

Coorientador
Eduardo Marques, Professor Auxiliar, Faculdade de Ciências da Universidade do Porto


U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Abstract

Ongoing (and upcoming) astrophysical surveys are planning on observing tens of millions of stars. Spectroscopy, among other techniques, is one of the gold-standards when it comes to obtaining physical and chemical information about an astronomical system like a star. Each absorption line in a star's spectrum stores and reveals unique information about its physical parameters, as well as its chemical composition. Deriving stellar atmospheric parameters is *per se* utterly important in astrophysics. The chemical composition of a star, as well as its effective temperature and surface gravity, from which other quantities, including the mass and the age, can be further derived, create windows into the history of that star and the galaxy to which it belongs to. This becomes even more interesting from the exoplanet research perspective, as the composition of an exoplanet is related with the composition of the host star and its characteristics. Therefore, high precision and accuracy must be ensured when deriving atmospheric parameters and chemical elements' abundances of a star.

Unconventional data-driven techniques hold the promise of efficiently dealing with these vast collections of data while still rendering results of astrophysical value.

Here, a machine learning approach to derive three stellar atmospheric parameters simultaneously is proposed. A tabular working dataset has been constructed using a sample of 449 F-G-K stars from the Gaia-ESO Survey for a target space composed of three response variables: Effective Temperature Teff, surface gravity log g and metallicity [Fe/H]. The predictor variables consisted of equivalent widths of a total of 520 spectral lines obtained from running ARES, a previously developed software to automatically extract equivalent widths from stellar spectra.

Several machine learning algorithms have been evaluated, including simpler models such as Linear Regression and its regularized variants (Ridge, Lasso and ElasticNet) or the distance-based Nearest-Neighbors, more complex kernel-based models such as the Support Vector Regressor (SVR), as well as Decision Trees and their bagging and boosting variants, Random Forests and Gradient Boosting Trees, respectively. All of these yielded significantly good performances against a dummy model that always predicts the median. 80% of the star sample (359 stars) was used for training the models, whereas the remaining 20% (90 stars) was kept for testing as a *hold-out*. Hyperparameter tuning has been conducted in the train set to find the best combination of hyperparameters for each model.

Even though a wrapper multi-output approach is expected to yield the same result as three

separate single-models, one for each of the target variables, in this case the results slightly differ because a wrapper multi-output setting implies the same model and respective hyperparameters are used to predict the three variables, so tuned hyperparameters in this setting may differ from the ones optimized for single-output models for each target. Choosing between a single- or a multi-output model will ultimately depend on a trade-off between predictive power and efficiency. A multi-output approach is highly advisable due to higher efficiency, however the choice depends on the error tolerance compared to a single-output setting. The effective temperature was best predicted by a Ridge regression, with a mean absolute error of 27.2 K, whereas a SVR model performed the best at predicting the surface gravity and the metallicity, with mean absolute errors of 0.026 dex and 0.017 dex, respectively. In a multi-output setting, the Ridge regression and the SVR yielded the best predictions for the three targets, recovering Teff, log g and [Fe/H] with typical mean absolute errors of 27.2 K *versus* 32.4 K, 0.033 dex *versus* 0.032 dex and 0.023 dex *versus* 0.018 dex, respectively, in the *hold-out* test set. However, these scores were not different than the ones obtained for other regularized approaches, such as the Lasso or the Elastic Net. Therefore, any of these models is expected to work well in this problem. A detailed analysis of important features and the effects of dimensionality reduction is provided. It is shown that, in general, using only 50 spectral lines from the initial 520 set suffices to produce satisfactory results. However, the best performing models change. If feature selection is conducted prior to modelling, regularized linear regressions like the Ridge or the Lasso significantly loose predictive power and are clearly outperformed by a multi-output SVR, with respective mean absolute errors of 38.8 K, 0.033 dex and 0.027 dex, or Gradient Boosting, with errors of 39.4 K, 0.031 dex and 0.031 dex. Therefore, these methods seem to be more reliable due to their robustness to the decrease in the number of predictors.

Resumo

Sondagens astrofísicas actuais e futuras planeiam observar dezenas de milhões de estrelas nos próximos anos. A espetroscopia, entre outras técnicas, é o “gold standard” quando se trata de obter informações físicas e químicas sobre um sistema astronómico como uma estrela. Cada linha de absorção no espectro dessa estrela contém informação única sobre os seus parâmetros físicos, bem como da sua composição química. A determinação de parâmetros atmosféricos de estrelas é, por si só, de importância crítica em astrofísica. A composição química de uma estrela, bem como a sua temperatura efectiva e gravidade superficial, a partir das quais outras quantidades, incluindo a massa e a idade, podem ser posteriormente derivadas, criam janelas para a história dessa estrela e da galáxia a que pertence. Isto torna-se ainda mais interessante do ponto de vista da investigação de exoplanetas, pois a composição de um exoplaneta está intimamente relacionada com a composição da estrela-hospedeira e das suas características. Por conseguinte, é necessário garantir elevada precisão e exactidão ao derivar os parâmetros estelares e abundâncias dos elementos químicos das estrelas. Técnicas não convencionais baseadas em dados são hoje a grande promessa no que toca a lidar eficazmente com estas vastas colecções de dados, por forma a produzir resultados com relevância do ponto de vista da astrofísica.

80was used for training the models, whereas the remaining 20as a hold-out. Hyperparameter tuning has been conducted in the train set to find the best combination of hyperparameters for each mo

Nesta tese, uma abordagem de “aprendizagem de máquina” para derivar três importantes parâmetros atmosféricos de estrelas simultaneamente, é proposta. Foi compilado e construído um conjunto de dados tabular utilizando uma amostra de 449 estrelas F-G-K da sondagem Gaia-ESO para um espaço-alvo de três variáveis dependentes: temperatura efectiva Teff, gravidade superficial log g e metalicidade [Fe/H]. As variáveis de previsão (ou independentes) consistiram nas larguras equivalentes de um total de 520 linhas espectrais obtidas a partir da execução do programa ARES para cada estrela, previamente desenvolvido para extrair automaticamente larguras equivalentes de espectros.

Foram testados vários algoritmos de "aprendizagem de máquina", incluindo modelos mais simples como a Regressão Linear e variantes com regularização (“Ridge", "Lasso" e "ElasticNet") ou algoritmos baseados em distância como os “vizinhos mais próximos", estratégias baseadas em "kernel", como "Support Vetor Regressor" (daqui em diante, SVR), e ainda Árvores de Decisão

e as suas variantes de "bagging" e “boosting”, Random Forests e “Gradient Boosting Trees”, respectivamente. Todos os modelos demonstraram desempenhos relativamente satisfatórios em comparação com um modelo de base que prevê sempre a mediana. 80% dos dados (359 estrelas) foram usados para treinar os modelos, enquanto os restantes 20% foram mantidos para teste (90 estrelas). Foi ainda efectuada a optimização dos hiperparâmetros utilizando o conjunto de treino para encontrar a melhor combinação para cada tipo de modelo.

Choosing between a single- or a multi-output model will ultimately depend on a trade-off between predictive power and efficiency. A multi-output approach is highly advisable due to higher efficiency, however the choice depends on the error tolerance compared to a single-output setting.

Embora fosse esperado que uma abordagem de resposta múltipla calculasse o mesmo resultado que três modelos de resposta única separados, um para cada uma das variáveis alvo, neste caso os resultados diferem ligeiramente porque a aborgem de resposta múltipla exige que o mesmo modelo e hiperparâmetros sejam usados para prever as três variáveis em conjunto, pelo que os hiperparâmetros nesta experiência diferiram dos optimizados para a abordagem de repsosta única. Escolher entre resposta múltipla ou individual irá, em última instância, depender de um equilíbrio entre eficiência e poder de previsão. Uma abordagem de resposta múltipla é altamente recomendável devido a ser mais eficiente, no entanto isso dependerá da tolerância face ao erro comparado com a abordagem de resposta individual.

A temperature efectiva foi melhor estimada por uma regressão "ridge", com um erro médio absoluto de 27.2 K, enquanto que um modelo SVR se revelou melhor para previsão da gravidade superficial e da metalicidade, com erros médios absolutos de 0.026 dex e 0.017 dex, respectivamente. Numa abordagem de resposta múltipla, os modelos Ridge e SVR resultaram nas melhores previsões para as três variáveis alvo, recuperando Teff, log g and [Fe/H] com erros médios absolutos de 27.2 K *versus* 32.4 K, 0.033 dex *versus* 0.032 dex e 0.023 dex *versus* 0.018 dex, respectivamente, no conjunto de teste. No entanto, estes erros não são diferentes dos obtidos para outros modelos regularizados, como Lasso ou Elastic Net. Assim, é esperado que qualquer um destes modelos funcione bem neste problema. É feita uma análise detalhada das variáveis de previsão importantes, bem como dos efeitos da redução de dimensionalidade. É demonstrado que, em geral, usar apenas 50 riscas espectrais das 520 iniciais é suficiente para resultar em resultados staisfatórios. No entanto, os modelos que se comportam melhor mudam. Se a selecção de previsores for feita antes do treino dos modelos, regressões lineares regularizadas como Ridge e Lasso perdem significativamente poder de previsão e são claramente ultrapassadas por um modelo SVR de resposta múltipla, com erros médios absolutos de 38.8 K, 0.033 dex e 0.027 dex, ou por um modelo Gradient Boosting, com erros de 39.4 K, 0.031 dex e 0.031 dex. Assim, estes métodos parecem ser mais confiáveis devido à sua robustez à redução do número de previsores.

Acknowledgements

There is a strong possibility that this thesis would not have existed without my supervisors, Luís and Eduardo. I do not mean it in terms of scientific support, at least not here. It is rather obvious that the scientific support of a mentor will always be crucial for a project to come into fruition.

For many reasons, the last few years of my life have been filled with a mixture of manic, obsessive and frequently, depressive episodes. Eventually, the sentiment of delusion with the world became unbearable and the thoughts in my head were racing at speeds hard for my body to catch up with.

To my supervisors, for their incredible support and for giving me the time and space I needed to get me back on my feet.

To our collaborators, Sérgio and Nuno, from CAUP, for always finding time for our scientific discussions, despite their sometimes busy schedules.

To Hervé, my PhD co-supervisor, who despite not being involved in this thesis directly, has always understood my reasons for enrolling on a second degree simultaneously and believed in my abilities even when I did not believe them myself.

To Ivo, who, I believe, might as well be the love of my life.

To Nuno, my forever soulmate and best friend, a frequent companionship for every overthinking process and miserable moments in life.

To my very scarce, but precious friends in the lab, whose identity must be protected at all costs, that made me omnipresent whenever needed, even though we all known I do not possess that ability of being in two places at the same time.

To my family, for always finding a way to support my stupid decisions, even when they do not completely understand them.

To my father, who is, I'm afraid, and always will be, my greatest inspiration in life.

Dedicated to my father, from whom, for the better or for the worse, I inherited the ability to never stop wondering

Contents

List of Tables

List of Figures

Listings

Acronyms

ANN - Artificial Neural Network
DNN - Deep Neural Network
EPE - Error
GANN - Generative Artificial Neural Network
HARPS - High Accuracy Radial velocity Planet Searcher
IQR - Inter-Quartile Range
kNN - k-Nearest Neighbours
LTE - Local Thermal Equilibrium
MAE - Mean Absolute Error
MATISSE - MATrix Inversion for Spectral SynthEsis
MI - Mutual Information
MSE - Mean Squared Error
OLS - Ordinary Least Squares
RBF - Radial Basis Function
RMSE - Root Mean Squared Error
RSS - Residual Sum of Squares
S/N - Signal-to-noise
SVD - Singular Value Decomposition
SVM - Support Vector Machine
SVR - Support Vector Regressor

Chapter 1

Introduction

Astronomy is one of the very few branches of science where it is not possible to interact with the system being studied. Therefore, scientists must rely on interpreting light and other forms of radiation emitted by regions or objects of interest from those systems. Every material emits, absorbs and scatters light and stellar spectra, which are measurements of the amount of light received at different wavelengths, are rich in information about the composition of stars.

The history of spectroscopy in astronomy began in 1802, when the English scientist William Wollaston passed a beam of sunlight through a thin slit and a prism, and observed that the slit provided a sharp and high-resolution view of a 'rainbow' spectrum, with a range of non-overlapping different colours. Wollaston noticed that, when seen this way, the Sun's spectrum was marked by many narrow, black lines of varying intensities that stayed at exactly the same places in the colorful band from day to day and year to year. These lines were later measured and cataloged by Josef von Fraunhofer, to whom they own their designation as 'Fraunhofer lines' [1].

Roughly around the same time, similar experiments in the laboratory were revealing the existence of spectral lines from different materials. Also using a slit and prism, physicists discovered that when a solid, liquid, or dense gas is heated to glow, it emits a smooth spectrum of light with no lines: a continuum. A rarefied hot gas, on the other hand, glows only in certain colors, or wavelengths: bright, narrow emission lines instead of a 'rainbow' continuous band. If a cooler sample of the same gas is placed in front of a glowing object emitting a continuum, then dark absorption lines appear at the wavelengths where the emission lines would appear if the gas were hot [4].

Around 1859, the picture on spectroscopy became pretty clear: the Sun's hot surface could be seen through a cooler solar atmosphere that imposed the dark spectral lines. Every element, every chemical compound, shows its own set of spectral lines, much like a fingerprint. These reveal not only which atoms and molecules are present, but also many other physical conditions, including the Temperature. When astronomers put the same slit-and-prism device - a spectroscope - on a telescope, they could observe spectral lines in the light coming from stars. This was likely the

19th century greatest astronomical discovery.

After that point, finding the composition of the Sun and other stars was just a matter of comparing spectral lines seen in a telescope with those seen in a laboratory, which turned spectroscopy into one of the pillars of modern astronomy. Using spectra, the light from distant objects measured on Earth can be related back to the physical and chemical conditions of the astronomical matter and thus physical parameters that describe those conditions can be derived.

Almost a century after, catalogues to classify stars started to appear. The MK spectral classification system, proposed by Morgan Keenan in 1943 with the publication of An Atlas of Stellar Spectra was the first astronomical atlas of photographic stellar spectra [5], despite smaller-scale versions existed before [2]. The MK system is a powerful tool for the study of stellar populations and it has proved useful for many decades to astronomers who need to infer basic quantitative information from a stellar spectrum. In fact, it is still used these days. Based on this system, stars can be classified into different types depending on their temperatures - O, B, A, F, G, K, and M - where type O stars are the hottest and type M stars are the coolest. Each of these types is divided further into sub-divisions from 0 to 9 in order to distinguish between slight differences in each star's spectral patterns, which depend on the star's temperature.

However, whilst spectra hide a treasure of astronomical information, their analysis is often complex and complicated by further technical problems and/or gaps in our theoretical knowledge. The vast catalogue of spectroscopic stellar surveys - general scans that provide a map or image of a region of the sky (or of the whole sky) with no specific target - of recent years, including SEGUE [6], RAVE (RAdial Velocity Experiment) [7, 8], LAMOST (Large Sky Area Multi-Object Fiber Spectroscopic Telescope [9], APOGEE ([10, 11], Gaia-ESO ([12, 13] and HERMES/GALAH (GALactic Archaeology with HERMES)[14, 15] hold gigantic potential, however, with great potential often come greater challenges. One of these challenges lies in consistently and accurately determining stellar atmospheric parameters and chemical elements' abundances. The parameters are usually determined from comparison of the observed data with synthetic model spectra, with approaches often customized specifically to the particular wavelength region of a given survey [8, 16–20]. Spectra obtained from this large-scale surveys are often used to derive essential stellar atmospheric parameters, such as the effective temperature of the star's photosphere, the surface gravity and the metallicity, as well as chemical compositions of certain elements.

Synthetic model spectra use theoretical physical models to generate the spectra, whose physical assumptions are usually incomplete and oversimplified. For example, 1D stellar photosphere models are almost always adopted for large surveys, often assumed to be in local thermal equilibrium, which is a rather strong assumption that may not be entirely true. In many cases, the model spectra also do not account for variables such as molecular opacities, convection, stellar winds, and the chromosphere itself. As a consequence, the results obtained for the same stars show discrepancies between different research groups, which are a consequence from analysis across different wavelength regions and different input assumptions and methods used to generate these models [21–23].

A small range of automatic procedures to extract these parameters from spectra have been proposed over the years. However, some of them rely on the abovementioned synthetic models for comparison, and others, which use line ratios, require the user to pick up the lines that should be used to determine the ratios [24]. Machine and deep learning approaches create an opportunity not only to deal with vast amounts of data, but also to optimize the prediction of stellar parameters and chemical abundances in a data-driven, physical-model free manner. Some studies have already tried to address these problems, using either gold-standard machine learning algorithms [25–31], or deep-learning approaches using with neural networks [32–42], all of them with satisfactory results.

However, the majority of these methods have not been tested using other stellar samples apart from the one used in their respective studies. In addition, even the neural network approaches, that require less spectra preprocessing techniques to extract features that can work as predictors in the model, still require continuum detection and normalization, which is a long-lasting issue in astronomical spectroscopy, even using standard techniques.

An ideal automatic tool to predict stellar parameters can be envisioned as one able to deal with the whole wavelength range (including even the ultraviolet and infrared regions), any class of star, as well as different resolutions and S/N rations - perhaps even dealing with data originating from different source instruments - and still provide reliable results under these conditions. The existent studies are quite preliminary in this front, as stellar samples tend to be small. This may happen for two reasons. First, because the machine learning strategies applied in this field use supervised learning - a type of learning that requires the existence of target labels derived by independent and well-established methods to work as a "ground-truth" to which the obtained labels are compared to - which are not easy to obtain for an extremely large dataset, as the traditional methods to analyze stellar spectra and derive atmospheric parameters, despite being now more automatic, still require an expert's eye to select good spectra, the best spectral lines for the analysis or to catch potential discrepancies and degeneracies. Second, because even if the ultimate goal is to have a universal tool that works with data coming from different sources, the first studies should privilege consistency over quantity for interpretability. With effect, if the spectra come from different sources, a user may stress where does a potentially bad performance of the model might come from - the model itself or simply from the lack of consistency in the data acquisition.

Altogether, machine learning holds great promise in stellar spectroscopy, but the next few years will be critical to define its true potential at predicting stellar parameters and chemical abundances on yet unlabeled data provided by recent large-scale sky surveys. The answer will probably rely on waiting for these data to be labelled using well-established physics' tools or, alternatively, on the development of reliable semi-supervised or unsupervised machine learning approaches which do not require new data to be previously labelled.

1.1 Motivation

The relevance of this preliminary work and hence, its main motivation, lies in two crucial aspects. First, from a purely astrophysics perspective, it is intrinsically interesting to derive stellar properties. As mentioned above, stellar physical parameters such as the effective temperature, surface gravity and chemical composition, reveal information about the history of stars and galaxies. This becomes particularly relevant in exoplanet research. With effect, stellar properties such as surface temperature and size determine the region around a star in which water can remain in liquid state. This is regarded as a fundamental prerequisite for life, to which this region owns its designation as the 'habitable zone' [43].

Second, there are currently no efficient methods to deal with the large amount of data being generated from recent astronomical surveys, which hold the potential to provide information about millions of celestial objects and thus, new data-driven techniques will become utterly important in the years to come.

Deriving stellar parameters from stellar spectra has a long history and several approaches were already proposed over the years, from manual inspection and comparison with previously known spectra and corresponding stellar parameters, to automatic tools based on physics' theoretical principles. However, the latter still often rely on the choice of spectral line ratios and/or comparison with synthetic atmospheric models from which those parameters are known.

Besides the possibility of processing and analyzing huge amounts of data, data-driven approaches, such as machine and deep learning, may in the future accurately predict stellar atmospheric parameters independently of physical models, which will be useful to validate previous conclusions or, hopefully, even creating an array of new questions. Altogether, there is a promise for, not only more, but also incredibly faster knowledge.

1.2 Contributions

The work developed in this thesis is a preliminary attempt at using machine learning approaches to obtain information about astronomical systems. Several machine learning models were employed to evaluate their performance in predicting stellar atmospheric parameters - effective temperature, surface gravity and metallicity. The stellar parameters range is quite small and limited in range, corresponding in the MK Classification system only to F-G-K stars, and the spectra S/N ratio is quite high, which makes the task easier to solve. Deep learning approaches were not yet evaluated due to the small sample size and because of their lack of interpretability, which was privileged in this work. Due to the large number of spectral lines compared with the number of observations, there is a high dimensional feature space. Thus, besides finding an algorithm that reliably predicts the stellar parameters, understanding whether some of those lines were more important than others for the algorithm's performance is also important for dimensionality reduction and raise questions on their physical meaning. A multi-output framework was used

to estimate the three target variables simultaneously with satisfactory errors compared to their respective ranges. The framework allows the input of different sets of features for each of the target variables in the same model, which provides a way to maximize predictive power while maintaining the efficiency of using a multi-output setting. This work also provides a strong basis for future goals, which include the estimation of chemical abundances and estimation of these same parameters for M-type stars, which most of the existent tools in the literature fail to address. This will likely include deep learning approaches, not directly for prediction, but to create more efficient ways to segment the spectra and perhaps learn to estimate a continuum with data-driven strategies, which would be completely novel in the field.

1.3 Organization

The thesis is organized as follows. Chapter 2 is dedicated to the theoretical background and foundations relevant to the research approach later developed. This includes both astrophysical scientific concepts and an extensive review on the mathematical aspects behind the machine learning algorithms used. Chapter 3 capitalizes on these foundations and provides a revision on published work in machine learning applications in astrophysics. Finally, Chapters 4 and 5 present, respectively, the methodology to address the research question proposed, the results obtained and a relatively thorough discussion on those results that feeds back onto the mathematical foundations formerly described on Chapter 2. In these chapters, we introduce machine learning approaches for extracting the chemical content from stellar spectra which do not rely on manual spectral inspection or physics-based theoretical models. This removes the burden of building faithful future models of stellar spectroscopy in order to precisely extract the chemical properties of stars. The impact or importance of certain predictors (spectral lines) over others on each respective algorithm performance is discussed in an attempt to extract some physical meaning. Finally, the multi-output setting, in which several target variables are predicted simultaneously, is also questioned in terms of . Chapter 6 concludes with the main lessons and scientific contributions of this preliminary work, as well as future short- and long-term perspectives nor only for improvement, but for new ideas implementation.

Chapter 2

Background

In this Chapter, some fundamental concepts on astrophysics, more specifically related to spectroscopy (Section 2.1), as well as some detailed theoretical foundations of machine learning algorithms (Section 2.2), are reviewed. The intention is to provide the reader with a full experience on the problem. We try to answer some basic questions that may come to mind: what is a spectrum? What is the absorption spectrum of a star and how is it formed? Why do we have several spectral lines for the same element, at different wavelengths? Why do the constituent elements of a star change with its physical properties? Even though domain knowledge may seem rather excessive for the computer scientist, it becomes oftentimes necessary when emergent behavior arises in the models, when data analysis and algorithms have to be slightly adapted to include some peculiarities that are not transversal among fields or because of interpretability and new transferable knowledge.

Still, this may seem rather dispensable for some. As such, the experienced reader is welcomed to skip any of these subsections, respectively. For the otherwise inexperienced and curious, we hope this provides a clearer picture to interpret the results of this work discussed on Section 5.

2.1 Astrophysics and spectroscopy

Astronomy is one of the very few branches of science where it is not possible to interact with the system because that is completely beyond our reach. Therefore, scientists must rely on interpreting light and other forms of radiation emitted by regions or objects of interest from those systems. Spectroscopy measures the electromagnetic spectra resulting from the interaction between electromagnetic radiation and matter as a function of the wavelength of the radiation [44]. The radiative transfer theory, a physical theory describing the transfer of electromagnetic radiation through matter is the essential connection for relating spectra obtained on Earth back to the physical and chemical conditions of the astronomical regions being observed. Within this framework, the radiative transfer equation is used to parameterize the change in intensity I_ν of a beam of light at a wavelength ν, along a path ds through a material

$$\frac{dI_\nu}{ds} = j_\nu - \alpha_\nu I_\nu. \tag{2.1}$$

where j_ν is the emission coefficient and α_ν is the absorption coefficient. Together, the values of j_ν and α_ν capture emission, absorption and scattering processes occurring in the beam at the given frequency ν. Their value will depend on the wavelength ν and on the chemical composition of the material [45]. Spectral lines are incorporated into this framework of radiative transfer through the absorption and emission coefficients.

Recall that, as mentioned in Chapter 1, a spectrum can be continuous, meaning that the dense gas in question emits light in all wavelengths, an emission spectrum which is represented by a dark background with colored lines corresponding to the light emitted by the gas or, an absorption spectrum with dark absorption lines in a continuous rainbow-like background, that represent the wavelengths of light which the gas absorbs [46]. For a visual representation of these concepts, refer to Figure 2.1.

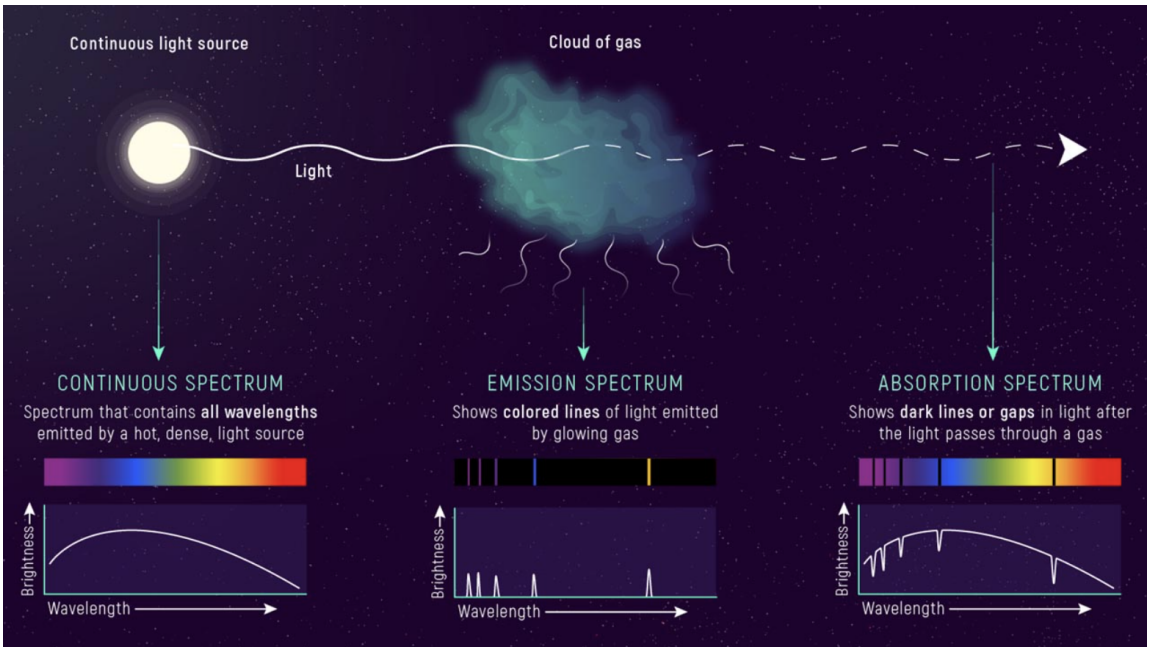


Figure 2.1: The relationship between the continuous spectrum of a star whose light is shining on gas, as well as the emission and absorption spectra of that gas. (Top half) A light wave passing through a cloud of gas; (Bottom half) Plots illustrating the three types of spectra formed. (From left to right) Continuous Spectrum - continuous rainbow and represented by smooth hump-shaped line on a plot of brightness versus wavelength; Emission Spectrum - a series of thin colored lines separated by black spaces, and a series of sharp peaks on a plot of brightness versus wavelength; Absorption Spectrum - rainbow background with a series of thin black lines replacing some of the colors, and a smooth curve with a series of sharp valleys on a plot of brightness *versus* wavelength. [1]

These spectra emerge because, at the particle level, electrons in atoms and molecules reside in

[1] https://webbtelescope.org/contents/media/images/

discrete energy levels, and so they emit and absorb photons with specific energies corresponding to the differences between energy levels, as illustrated in Figure 2.2 for the Hydrogen atom. The reader should keep in mind that the number of energy levels for chemical species with higher atomic number is bigger, hydrogen is the simplest species and works here merely as an example. More general information on atomic energy levels and electronic transitions can be found in Appendix A. Such emission and absorption phenomena lead to those narrow spectral features known as emission and absorption lines in the spectra. These spectral lines are thus unique to every chemical species and so can be used to work out the chemical composition of astronomical objects of interest. The absorption of photons resulting from electronic transitions to higher levels in the hydrogen atom (see Figure 2.2 left and top right), produce the absorption spectra observed in the bottom right of the same Figure. As a side note, the plot observed in the bottom right is the kind of signal that astronomers usually work with to analyze spectra - smooth curve with a series of sharp valleys on a plot of brightness versus wavelength.

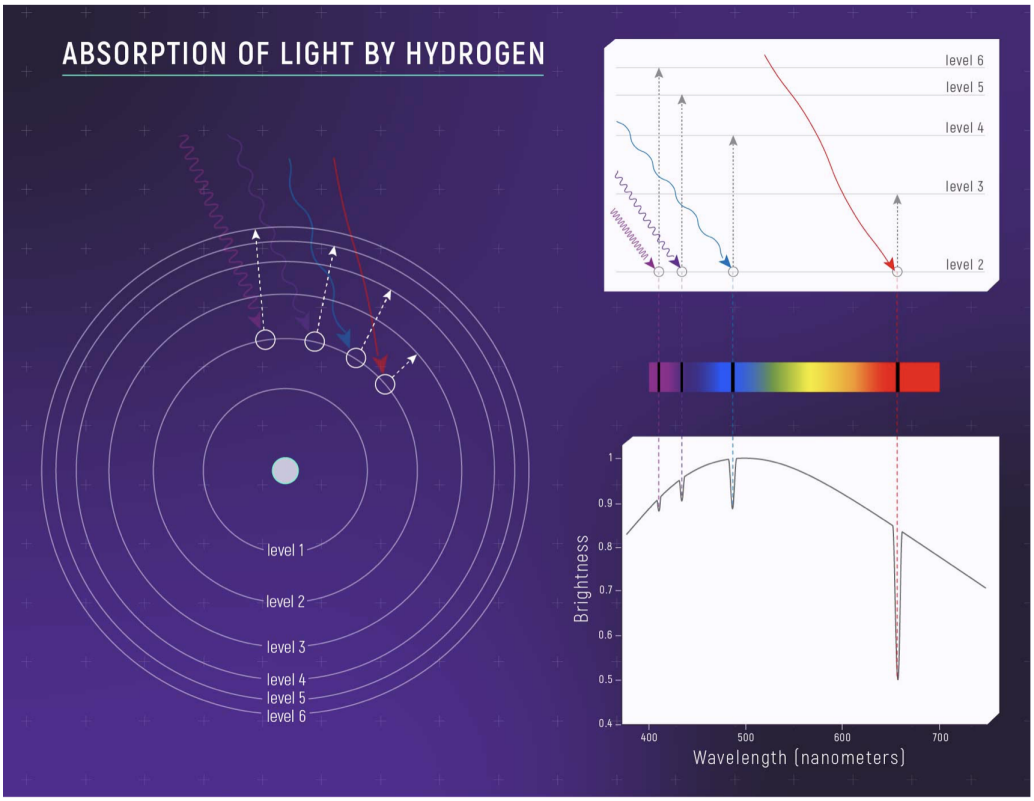


Figure 2.2: The relationship between a hydrogen atom and its absorption spectrum. (Left) A simple model of a hydrogen atom showing four of the many possible energy levels the electron could jump to when it absorbs light. (Right) The relationship between those electron jumps and the specific wavelengths of light that the atom absorbs. An electron jumps from an energy level to another only when it absorbs a very specific wavelength of light, that is when it absorbs a photon with a specific energy. The shorter the wavelength, the higher the energy, and consequently, the higher the jump. This illustration shows a set of jumps that correspond to absorption of visible wavelengths (the Balmer Series). [2]

[2] https://webbtelescope.org/contents/media/images/

2.1.1 Measure the strength of spectral lines

Figure 2.2 in the previous section illustrated clearly how an adsorption spectra looks like. To access the presence of certain chemical species in the spectra and quantify their abundances, astronomers can measure the strength of specific spectral lines.

A spectral line can be defined by three main parts: the continuum, the wing and the core, as depicted in Figure 2.3.

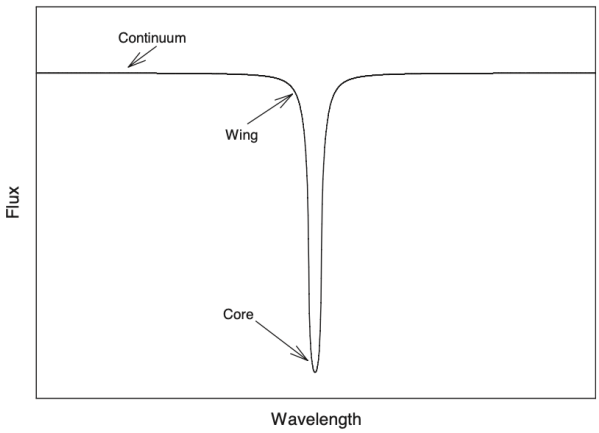


Figure 2.3: The parts of a spectral line. *Adapted from* Gray (2009) [1].

These parts are related to the temperature gradient of the stellar photosphere: the top or outer layer is considerably cooler than the inner layer. Therefore, the core of an absorption line is formed in the cool upper layers of the stellar photosphere, the surrounding continuum is formed at deep layers, and the wings of the absorption line in intermediate layers. This implies that if the continuous opacity is relatively high, the spectral line will be formed only over a limited range in the atmosphere, and thus will be weaker than if the continuous opacity was low [1].

The strength of spectral lines can be measured using different quantities based on the shape of the spectral line.

Peak intensity The so-called line depth, directly measured in a non-normalised profile, corresponds to the distance from the level of the continuum until the core of the line. However, one should be cautious as this is not exactly a measure of the intensity. The peak intensities can only be compared to each other considering a relation to the local continuum level.

Full Width at Half Maximum Height (FWHM) The FWHM value is the line width in angstroms (Å) at half height of the maximum intensity. It can be correctly measured even in non-normalised spectral profiles. The width of a spectral line depends on the temperature, pressure, density, and turbulence effects in stellar atmospheres. Therefore, it yields important conclusions and is often used as a variable in equations, for example to determine the rotational velocity of stars.

Equivalent Width The equivalent width of a spectral line is a combination of its width and intensity. It measures the area of the line on a plot of intensity *versus* wavelength in relation to the underlying continuum level. It is found by forming a rectangle with a height equal to that of the continuum, and finding the width such that the area of the rectangle is equal to the area of the spectral line [47]. This is depicted in Figure 2.4.

Formally, the equivalent width is given by the equation

$$W_\lambda = \int \frac{F_c - F_s}{F_c} d\lambda = \int 1 - \frac{F_s}{F_c} d\lambda, \tag{2.2}$$

where $F_c(\lambda)$ represents the underlying continuum intensity and $F_s(\lambda)$ represents the intensity of the actual spectrum (line and continuum).

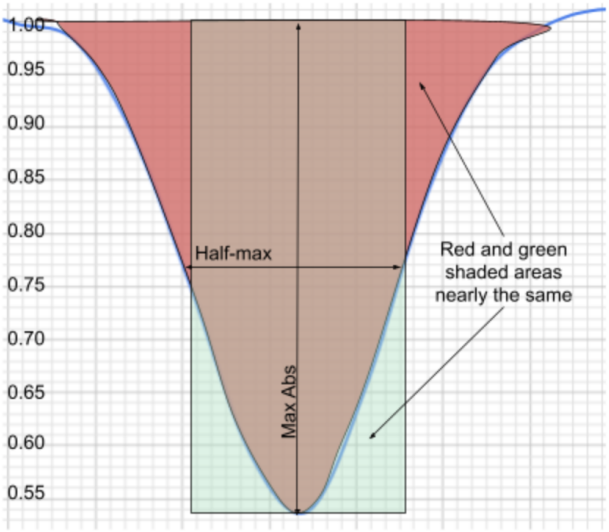


Figure 2.4: Illustration depicting how to measure equivalent width from spectral lines. The top level is called the continuum and the equivalent width corresponds to the integral of the area beneath the curve delimited by the intersection of the spectral line curve at its half-height extended to the curve depth form the continuum. [3]

The equivalent width is a convenient choice to measure the line strengths because the shapes of spectral features can vary depending upon the configuration of the system which is producing the lines [48]. For instance, the line may experience Doppler broadening due to motions of the gas emitting the photons. The photons will be shifted away from the line center, thus rendering the height of the emission line a poor measure of its overall strength. The equivalent width, on the other hand, measures the 'fraction of energy' removed from the spectrum by the line, regardless of the broadening intrinsic to the line or of the resolution of the detector. Thus, it can be extrapolated that, under certain conditions, the equivalent width can indicate the number of absorbing or emitting atoms [49].

[3] Source: https://www.jimmynewland.com/wp/

2.1.2 Stellar atmospheric parameters

Before moving on to the definition of each stellar atmospheric parameter and their possible impact on spectral lines, it seems pertinent to look at stellar types' classification systems, even if from a rather superficial perspective. Although these will not be explored directly in this thesis, some terminology used in this or other upcoming Sections may benefit from some context regarding these classification systems.

The Harvard scheme of stellar spectral classification divides stellar spectra into seven main classes, based on a temperature sequence: O, B, A, F, G, K, and M, where O-type (with effective temperatures above 30 000K) corresponds to the hottest and M-type to the coolest (2200–3700 K) stars. Each main class is then followed by 10 subclasses ranging from 0 to 9 indicating decreasing temperature within that class. An addition to the above classification scheme was proposed by Morgan and Keenan - the MK classification system - where each spectrum can also be classified into six primary luminosity classes: Ia, Ib, II, III, IV, and V. Luminosity classes are appended to the main class and subclass, so first should come the temperature class, followed by the subclass and, finally, the luminosity class. These luminosity classes are indicators of the stellar surface gravity, with 'Ia' denoting luminous giants (having smaller values of log g) and 'V' denoting dwarfs (having the maximum value of surface gravity). Therefore, they add another physical property of the star to the classification system. For a visual representation, a correspondence between temperature and luminosity classes is depicted in Figure 2.5, in a Hertzsprung-Russell Diagram.

A very detailed description of the stellar spectral classes can be found in Gray (2009) [1].

The MK spectral classification system was proposed by Morgan Keenan in 1943 with the publication of An Atlas of Stellar Spectra, the first astronomical atlas of photographic stellar spectra, which is still widely used [5].

The stellar atmospheric parameters, including the effective temperature Teff, surface gravity log g and metallicity, are of fundamental importance in astrophysics. They are not only essential prerequisites to any detailed subsequent abundance analysis, but also to the definition of the physical conditions in the stellar atmosphere as they are directly related to physical properties of the star, including the mass M, the radius (R), luminosity (L) and the age [50, 51]. Furthermore, stellar properties such as surface temperature and size determine the region around a star in which water can remain in its liquid state, which is a prerequisite for life [43]. Most of the currently known planet hosts have properties resembling those of the Sun – but this can simply be because most planet searches chose solar-like stars as their targets.

[5]Source: https://en.wikipedia.org/wiki/Stellar_classification

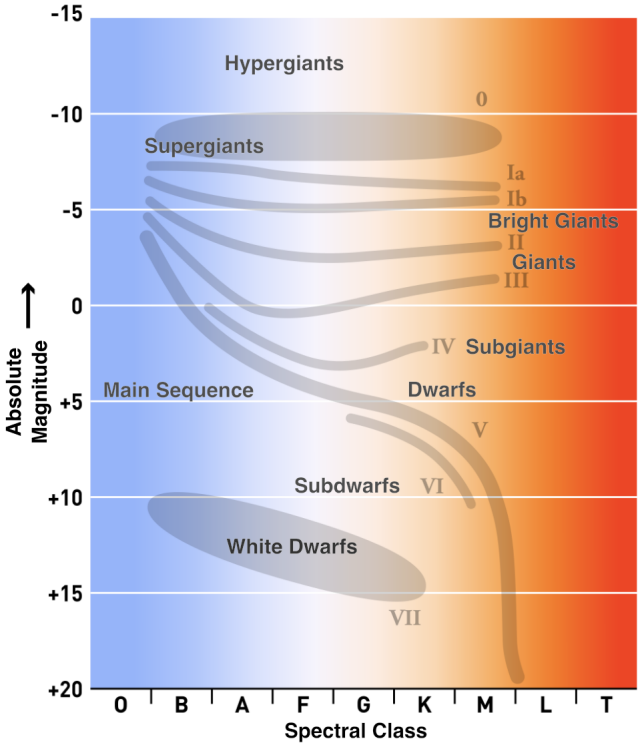


Figure 2.5: Hertzsprung-Russell (HR) Diagram. The luminosity is the measure of the star's brightness. The spectral type on the lower horizontal axis is based on the Temperature. The diagram also describes the phases of a star over its lifetime.[5]

2.1.2.1 Effective Temperature

The effective temperature of a star, Teff, is defined as the temperature of a black body radiator that would produce the same luminosity as the star does. If stars are considered black body radiators, their luminosity L is related to their effective temperature by the Stefan-Boltzmann equation

$$L_* = 4\pi R_* \sigma T_{eff}^4 \tag{2.3}$$

where Teff is the effective temperature in the stellar photosphere, R_* is the radius and σ is the Stefan-Boltzmann constant.

The temperature is the parameter that has the strongest effect on the placement of the continuum and absorption line strength, as conveyed by Figure 2.6 and the description on Table 2.1.

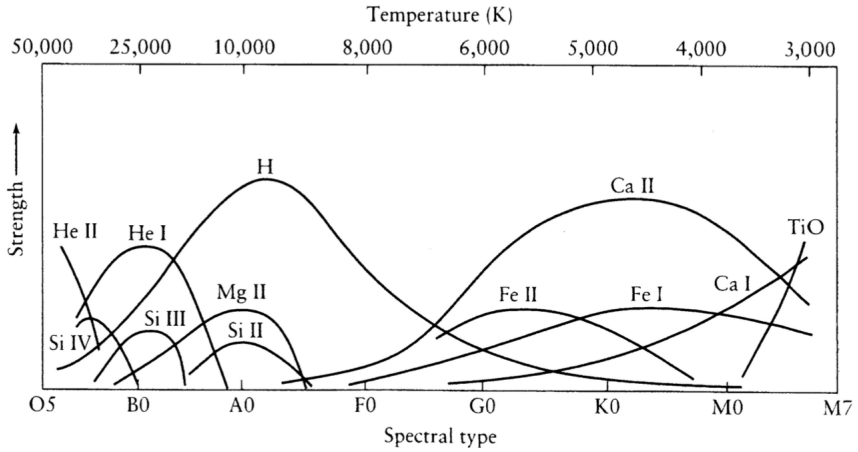


Figure 2.6: Spectral line strength as a function of stars' temperatures. Adapted from Gray(2005) [2].

MK Classification	**Spectral Features**
O	Strong ionized He lines but weak H and Ca lines
B	Strong neutral He lines with some subtypes with strong ionized He lines
A	Strong H lines, some ionized metals like Ca
F	Strong Ca lines with weak H lines
G	Mostly strong Ca and weak H lines, with some other metals showing up in fine lines in violet and blue regions
K	Strongest Ca lines, other neutral metals like Na
M	Lots of absorption lines, cool enough for lines by molecules like TiO

Table 2.1: Star types according to the MK classification system, their temperature ranges and spectral features.[6]

This sensitivity stems from the exponential and power dependencies with temperature and the radiative transfer equations regulating excitation-ionization equilibrium, as well as the continuum opacity. An increase of equivalent width goes hand in hand with that of temperature because of an increase in excitation. After reaching its maximum in strength **??**, the line starts weakening as the continuous opacity of the negative hydrogen ion (H^-) starts increasing due to high electron pressures. A decrease in equivalent width can also be due to the frequent ionisation of absorbent species at high temperatures, specially in the optical region where the majority of lines come from neutral species [2].

Measurement uncertainties in Teff are typically found to be around 200 - 300 K using standard methods. If different methods are available, the errors can be brought down to 50 K. It is also

[6]Source: Sloan Digital Sky Survey: http://cas.sdss.org/dr7/en/proj/basic/spectraltypes/

reported that a change of 100 K affects abundances obtained from neutral lines up to 0.1 dex, yet those derived from ionised lines change very little [52].

2.1.2.2 Surface gravity

The surface gravity, commonly found in the literature as log g, is the Newtonian gravitational acceleration of the star at its surface assuming it is perfectly spherical, The surface gravity, g of a star is directly given by the stellar mass and radius (in solar units):

$$g = G\frac{M_*}{R_*^2}, \tag{2.4}$$

where M_* is the mass enclosed by R_*, the stellar radius, and G is the gravitational constant.

Or, in logarithmic scale:

$$\log g = \log M - 2\log R + \log G \tag{2.5}$$

Since stars are astrophysical systems in equilibrium, the gravitational force that binds the stellar atmosphere with the star is balanced out by the pressure gradient and therefore log g is directly related to the pressure structure of the stellar atmosphere.

Spectral lines are not as sensitive to log g as they are to Teff, so it is more challenging to measure it using spectroscopy. There are three ways to visualise pressure effects. The first comes from the abundance of absorbers with respect to the continuous opacity. The second is via the damping coefficient of strong lines and the third is due to pressure dependencies of the Stark effect. For the purpose of this thesis, only the first one is explained (refer to Gray(2005) [2] for additional details). The key factor is the ionisation state of the elements involved. Not only does the number of absorbers increase with pressure, but also the number of free electrons provided by the ionised species further affecting the continuous opacity (H^- free-free transitions). If an ion or a neutral atom's lines are in the same or higher ionisation levels compared to the majority of the elements of the same species, their lines strengthen when the pressure, and thus the continuous opacity decreases.

Reported errors of log g are usually not better than 0.1 dex, though asteroseismic surveys may be able improve these uncertainties by a factor of 10 [52]. As expected, from the explanation above on ionization states, surface gravity influences the chemical abundances derived from weak ionised lines more than it does with neutral lines. However, generally speaking, high surface gravity tends to widen spectral lines, while lower surface gravity narrows them [50]. There are multiple approaches to obtain log g, such as the ionisation balance of the FeII to FeI line ratio,

through the wings of strong lines or with asteroseismology [2].

2.1.2.3 Metallicity, [Fe/H]

Metallicity is simply given by the chemical abundance ratio between iron (Fe) and hydrogen scaled with respect to the Sun:

$$[Fe/H] = log\left(\frac{N_{Fe}}{N_H}\right)_* - log\left(\frac{N_{Fe}}{N_H}\right)_\odot \tag{2.6}$$

Iron is used as a proxy to represent metals in general, as the number of iron lines present in the optical are much more abundant than any other element. However, abundances of other elements do not necessarily have to correlate with Fe. The importance of metallicity lies in how it shapes the overall structure of the stellar atmosphere, influencing the strength of absorption lines and continuum opacities. High metallicity implies larger abundances of metallic electron donors resulting in a change in the continuous absorption through H ions [2, 52].

The relationship between the several stellar parameters, particularly Teff and log g, and specific spectral lines, does not seem to be completely well-established as to whether looking at the strength of certain lines is enough to estimate reliably a certain parameter or not, and if so, which lines. The intention here is not to claim that this topic has not been given any thought, as there is a modest body of literature addressing it. In addition, Figure 2.7 demonstrates exactly that the decision process in the past did not use many features to reach a decision. Yet, this knowledge seems to be quite disperse and it feels hard to compile a set of consistent spectral lines that suffice to estimate Teff, log g or even metallicity. One of the reasons could be related with spectral lines being simultaneously affected by several atmospheric parameters, among other phenomena, which makes the decoupling a difficult task.

What is known exactly on spectral lines or spectral features that are specifically temperature- or pressure-dependent? In 1996, Gray had already asserted that some spectral lines are more sensitive to temperature than others [53]. By comparing the behavior of certain spectral lines between hotter and cooler stars, he observed that whilst some lines hardly changed, such as SiI, or changed only slightly like FeI, others showed pronounced changes, including VI, or even a reversal, in the case of FeII. However, since alterations in surface gravity or metallicity can also affect line strengths, he introduced the idea that the best approach would be to rely on differences of one line compared to another rather than directly on absolute line strengths. In 1994, he came up with 10 line-depth ratios using only weak lines, as stronger lines might be dependent on metallicity and affect the results. These included TiI-6220.48Å/NiI-6223.99Å, TiI-6221.34Å/NiI-6223.99Å, VI-6224.51Å/NiI-6223.99Å, VI-6224.51Å/FeI-6226.74Å, VI-6233.20Å/FeI-6229.23Å, VI-6242.84Å/SiI-6243.83Å, VI-6242.84Å/FeII-6247.56Å, VI-6251.83Å/FeII-6247.56Å, VI-6251.83Å/FeI-6253.83Å and VI-6251.83Å/FeI-6252.57Å [54]. Other similar studies followed [24, 55]. More

recently, 5 FeI–FeII and 4 CaI–CaII line pairs together with 13 FeI–FeI pairs were selected for estimating Teff for the spectra of 42 F-G-K stars [56]. Even though these details are not mentioned, the reader should keep in mind that oftentimes, the wavelength band used is not the same among different studies.

It is also well-known that strong lines with pronounced wings can be good tracers of the surface gravity [2]. Mg Ib lines λ5172 and λ5183 have been appointed as pressure-dependent and were used to estimate the surface gravity of a G8III subtype star [57]. In addition, Mg Ib triplet lines have been advocated as one of the best criteria to estimate surface gravity in late type stars [58]. The surface gravity of several star subtypes has also been determined based on the pressure-broadened CaI λ6162 line [59]. Fuhrmann (1997) proposed to employ the pressure-broadened Mg I b lines to derive the gravity parameter for F and G stars. These lines were believed to be more robust and reliable compared to the ionization equilibrium of FeI/FeII, which is susceptible to over-ionization effects and uncertainties in the temperature structure of the model atmosphere [60]. Finally, Bruntt (2010) proposed a method for the determination of effective temperature and surface gravity based on FeI and FeII lines. In this study, broad lines are used for determination of surface gravity as these are extremely sensitive to variations in surface gravity, as previously mentioned above on the definition of this stellar parameter [61].

In summary, although some common conclusions can be found in the literature, there is yet no magic formula when it comes to select a number of spectral lines that can estimate these parameters reliably for every class of stars.

2.1.3 Methods to derive stellar parameters and chemical abundances

In the previous section, essential definitions of each stellar atmospheric parameters were provided, as well as a general commentary on how each of them may affect lines in stellar spectra. Nevertheless, how can this information about the stellar parameters be obtained from a spectrum?

For many years, and still today, at least as a classical physics approach, the first step in stellar classification or determination of quantitative parameters was generally to identify the spectral lines whose strengths should serve as criterion. Then, the decision process followed through as a sequence of binary decisions, as illustrated on the schematic of Figure 2.7. These lines could be identified by their position in a comparison spectrum or to some few prominent features of that spectrum itself. For example, iron and titanium were frequently used in comparison spectra. Further details on the specific bands and line ratios present in the figure will not be provided in this thesis, but extensive descriptions can be found in Gray (2005) [2].

Now, a rather interesting observation is that this rudimentary decision process in stellar classification based on spectral lines resembles a decision-tree algorithm (to be discussed on Subsection 2.2.3), which might suggest that tree-based algorithms are well-suited for this kind of problems. This is addressed and discussed in Section 5.

There are several methods to derive these parameters, however, their results often show

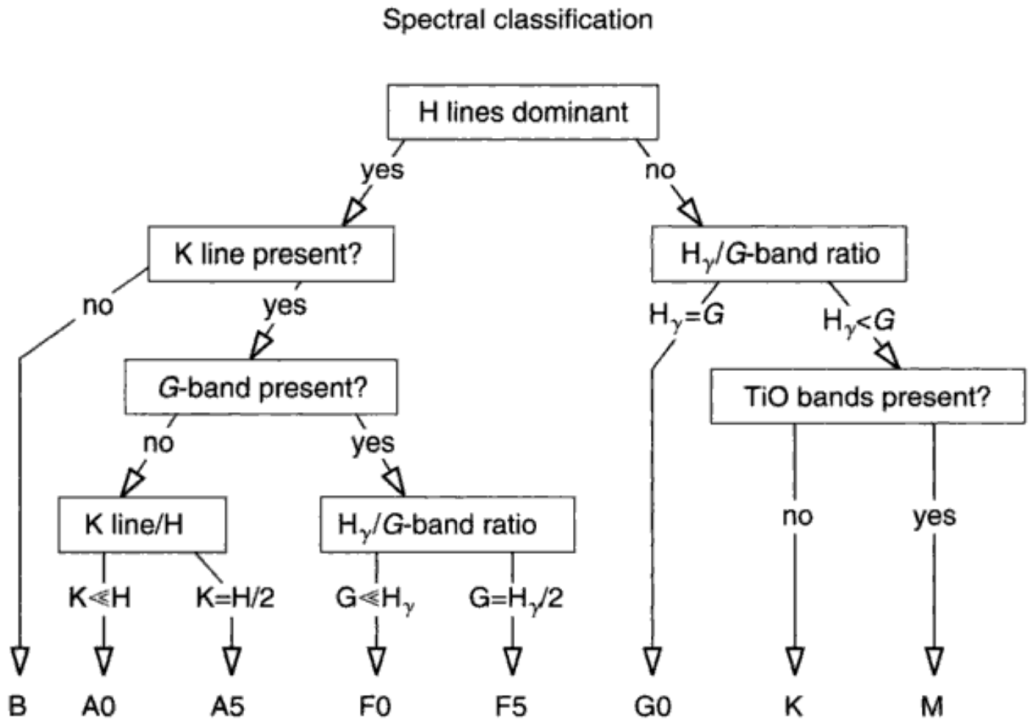


Figure 2.7: Decision workflow of stellar type classification. The rationale resembles a decision tree that starts with H lines and includes other bands known to be associated with certain stellar types by previous experience or theoretical knowledge. *Adapted from:* Gray (2005) [2].

considerable discrepancies, even in the restricted group of solar-type F-G-K dwarfs. From an astrophysical perspective, these methodologies can be divided in some main categories [50]:

- **Balmer profiles:** used to predict effective temperatures for stars cooler than about 8000 K due to their virtually null gravity dependence [2]. For stars hotter than 8000 K the profiles are sensitive to both temperature and gravity. As such, for these stars, the Balmer lines can be used to obtain values of log g, provided that Teff can be determined using a different method.

- **Spectral line ratios:** Spectral lines are sensitive to temperature variations within the line-forming regions. Thus, line strength ratios can be used as temperature diagnostics, similar to their use in spectral classification [54, 55]. Line depth ratios have been used to determine stellar effective temperatures with a precision of ± 10 K. While this method can yield very precise relative temperatures, the absolute calibration on the Teff scale is not so well determined [54]. This method is ideal for investigating stellar temperature variations [62]. This strategy is still used today [24, 63]. Sousa *et al.* derived the effective temperature Teff for solar type stars using line equivalent width ratios. The calibration was done for 433 line equivalent width ratios built from 171 spectral lines of different chemical elements. The approach consisted in using the ratio of the equivalent widths of two lines that have different sensitivities for measuring the temperature [24]. Contrarily to previous studies that also used line-ratios [54], in this study the equivalent widths were employed instead of

the line depths. This method allows to reach a precision for the temperature down to a few Kelvins in the best cases. However, despite being entirely automatic, the procedure requires laborious preprocessing, including outliers removal and the selection of lines obeying to specific criteria, as well as the best line ratios that must be then calibrated. 934 line ratios that should be sensitive to temperature were derived, which is quite an extensive list. The ratios between lines CrII-4592.05 Å/CrI-4626.18 Å, NiI-6130.14 Å/VI-6135.36 Å and FeI-6705.11 Å/FeI-6710.32 Å corresponded to the best, the average, and the worst cases, respectively, from the final list of calibrated line ratios. For more information on other line ratios, refer to the original publication [24].

- **Metal Line Diagnostics:** the equivalent width of many spectral lines can be used to determine the atmospheric parameters via metal line diagnostics, which involves the ionization balance and the excitation potential. The former means the abundances obtained from differing ionization stages of the same element must agree and usually gives a line in a Teff – log g diagram. The latter states that abundances from the same element and ionization stage should agree for all excitation potentials.

- **Global spectra fitting using model atmospheres:** instead of analysing individual spectral lines, the entire observed spectrum is used to find the best-fitting synthetic spectrum. The procedure involves taking a rather large multi-dimensional grid of synthetic spectra computed with various combinations of Teff, log g, ζturb, metallicity, among other parameters, and find the best solution through, for example, least squares techniques. Naturally, the final parameters are derived by comparison, thus model dependent, and their quality is limited to the model atmospheres used.

In the abovementioned methods, the procedure to derive stellar atmospheric parameters from absorption spectra was still manual for the most part, or it required human supervision for comparison with model atmospheres. Despite computer programs were already used to automate parts of the procedure, those still required the user to manually find and draw the line and to mark the position of the continuum as well as the position of possible additional lines in the case of blending effects.

Some automatic tools to facilitate the determination of stellar parameters have been developed over the years. The majority of those tools focused on spectra treatment and preprocessing for further analysis and may include, in general, the following steps:

- **Continuum normalization:** the continuum points of a spectrum are found by applying a median and a maximum filter with different window sizes to reduce noise and ignore deeper fluxes that belong to absorption lines, respectively. A polynomial or group of splines can be used to model the continuum, and finally the spectrum is normalized by dividing all the fluxes by the model.

- **Resolution degradation:** the spectral resolution can be degraded by convolving the fluxes with a Gaussian of a given full width at half maximum.

- **Telluric lines identification:** telluric lines from Earth's atmosphere contaminate the observed spectra which can affect the parameters' determination. The position of these lines in the spectra can be determined by cross-correlating with a telluric mask built from a synthetic spectrum.
- **Re-sampling:** spectra can be re-sampled by using linear (two points) or Bessel (four points) interpolation.
- **Equivalent width measurement:** equivalent widths are determined by fitting a Gaussian profile in each absorption line and integrating its area, as explained in 2.1.1.

Some well-known software that automatically perform some of all of this tasks, include ARES [48], DAOSPEC [64] or DOOp [65]. For example, ARES is an automatic procedure to determine equivalent widths in stellar spectra. It receives as input, among others, the 1D-fits file (IRAF format), the line-list containing the lines to be found, the wavelength domain to be used, the minimum interval to be considered between successive lines, the wavelength interval to be considered around each line, the minimum accepted equivalent width of a line, as well as calibration parameters for either the continuum estimation and noise smoothing. Yet, it assumes the input spectrum has already been corrected for the Doppler shift. The computation of the equivalent width is performed locally around each line, much like the old manual procedure [48].

In summary, despite providing estimations with high precision, the automatic procedures based on physics principles still require an extensive amount of human intervention. Other alternatives, using artificial intelligence and data-driven approaches hold great promise in completely optimizing this procedures in the near-future, as discussed below on Chapter 3.

2.2 Machine Learning

This section is intended to provide an overview of important machine learning concepts and gold-standard algorithms. For an expert in the field, the mathematical details might seem unnecessary and dispensable and could certainly be replaced by simpler descriptions accompanied by the proper reference. Yet, these details might become important from an astrophysics perspective or for the general computer scientist not completely familiarised with machine learning. Regardless of the target audience, these details might oftentimes feel quite exasperating. However, their understanding turns critical when moving to variants of the models here explained, such as the multi-output setting in which more than one response has to be predicted, model stacking or hybrid models. Otherwise stated, most of the information on this section follows very closely the book [3].

2.2.1 Statistical models and Supervised Learning

In Supervised Learning, the goal is to find a useful approximation $\hat{f}(x)$ to the function $f(x)$ that defines the predictive relationship between the inputs and outputs. In a theoretical setting the squared error loss leads to the regression function $f(x) = E(Y|X = x)$ for a quantitative response - called a regression problem.

Supposing, for simplicity, that the errors are additive and that the statistical model $Y = f(X) + \epsilon$ is a good assumption, Supervised Learning consists in learning the function f by example. One observes the system under study, both the inputs and outputs, and assembles a training set of observations $T = (x_i, y_i)$, $i = 1, ..., N$. The observed input values x_i are also fed into the system - the learning algorithm (usually a computer program) - which also produces outputs $\hat{f}(x_i)$ in response to the inputs. The learning algorithm has the property that it can modify its input-output relationship f in response to differences $y_i - \hat{f}(x_i)$ between the real and the generated outputs. When the learning process is finished, the predicted and real outputs should be close enough to be able to generalize for all sets of inputs likely to be encountered in practice.

2.2.2 Variance-Bias Tradeoff

A statistical model or a machine learning algorithm is said to underfit when a model is too simple to capture data complexities. It represents the inability of the model to learn the training data effectively, resulting in poor performance both on the training and testing data. This model is said to have high bias and low variance. A model is said to be overfitted when it has a good performance on the training data, but it does not make accurate predictions on testing data. When a model is to complex or gets trained with too much data, it starts learning from the noise and inaccurate data entries. Thus, when testing on unseen observations, the results will show high variance.

To better understand these concepts, let's try to express them mathematically.

Let the response variable be Y and other predictor variable be X. It is assumed to exist a relationship between them, such that

$$Y = f(X) + e, \tag{2.7}$$

where e is the error term which normally distributed with a mean of 0.

The expected squared error at a point x is

$$Err(x) = E\left[(Y - \hat{f}(x))^2\right] \tag{2.8}$$

which can be further decomposed as

$$Err(x) = \left(E[\hat{f(x)}] - f(x)\right)^2 + E\left[\left(\hat{f}(x) - E[\hat{f}(x)]\right)\right] + \sigma_e^2 \tag{2.9}$$

Figure 2.8 shows the behaviour of the prediction error for the train and test sets with varying model complexity.

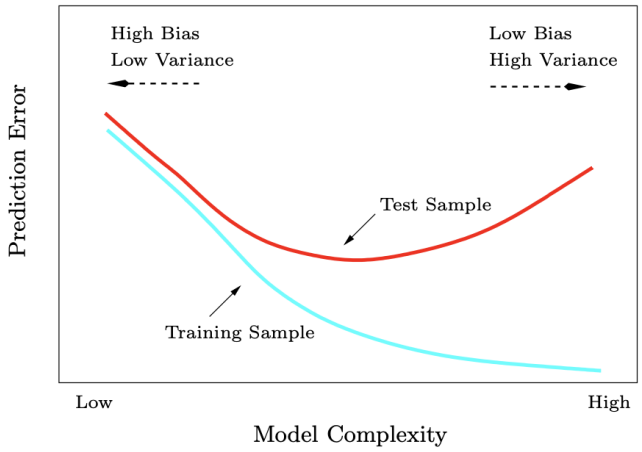


Figure 2.8: The Bias-Variance Tradeoff: Test and training error as a function of model complexity. *Adapted from:* Hastie, Tibshirani Friedman (2009) [3].

The training error tends to decrease whenever the model complexity increases, that is, whenever we fit the data harder. However with too much fitting, the model adapts itself too closely to the training data, and will not generalize well (i.e., have large test error). In that case the predictions $\hat{f}(x_0)$ will have large variance, as reflected in the last term of expression (2.46). In contrast, if the model is not complex enough, it will underfit and may have large bias, again resulting in poor generalization.

As the model becomes more and more complex, it uses the training data more and is able to adapt to more complicated underlying structures. Hence there is a decrease in bias but an increase in variance. There is some intermediate model complexity that gives minimum expected test error. Unfortunately training error is not a good estimate of the test error. Training error consistently decreases with model complexity, typically dropping to zero if we increase the model complexity enough. However, a model with zero training error is overfit to the training data and will typically generalize poorly.

2.2.3 Algorithms

In this subsection, a mathematical revision on several gold-standard machine learning algorithms is provided. The focus is on the versions of those models applied to regression tasks. The recap starts with the simplest models, such as linear regression (also known as ordinary least squares) and nearest-neighbors (a distance-based algorithm), passes through kernel-based methods and finishes with decision trees and its bagging and boosting ensemble variants.

Ordinary Least Squares (hereafter, OLS) is one of the oldest and most simple algorithms used for regression, yet it remains a very important tool in several problems, as it can often outperform more sophisticated models. OLS is particularly useful when there is not a huge amount of observations, or when the inputs reliably predict the response (low S/N ratio). However, the linear model makes very strong assumptions on the structure of the data, thus it yields stable but possibly also inaccurate predictions.

They are simple and often provide an adequate and interpretable description of how the inputs affect the output. For prediction purposes they can sometimes outperform fancier nonlinear models, especially in situations with small numbers of training cases, low signal-to-noise ratio or sparse data. Finally, linear methods can be applied to transformations of the inputs and this considerably expands their scope. These generalizations are sometimes called basis-function methods

2.2.3.1 Linear regression

Let $\mathcal{D}$ be an $n \times d$ data matrix whose ith data point (row) is the d-dimensional input feature vector X, and the corresponding response variable is y. Given the vector of inputs $X^T = (X_1, X_2, ..., X_p)$, the output y is predicted according to

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j, \tag{2.10}$$

where $\hat{\beta}_0$ is the intercept, best known as the *bias* in a machine learning setting, and $\hat{\beta}_j$ is the vector of coefficients corresponding to each input feature X_j. It is often convenient, from a linear algebra practical perspective, to include the constant term 1 in the vector of inputs X and the intercept $\hat{\beta}_0$ in the vector of coefficients $\hat{\beta}$, so the model can be expressed as the inner product in vector notation

$$\hat{Y} = X^T \hat{\beta}, \tag{2.11}$$

where X^T denotes either a vector, X being a column vector on a single linear regression, or matrix transpose in multiple linear regression. In equations 2.10 and 2.11 above, a single output $\hat{Y}$ is mapped by a set of inputs X, thus $\hat{Y}$ is a scalar. However, $\hat{Y}$ can be also be a k–dimensional vector, in which case the coefficients $\hat{\beta}$ would be a $p \times k$ matrix - this is called a multi-output

problem, which is discussed in the upcoming subsections.

In the $(p+1)$-dimensional input–output space, $(X, \hat{Y})$ represents a hyperplane. If the constant is included in X, then the hyperplane includes the origin and is a subspace; if not, it is an affine set cutting the Y-axis at the point $(0, \hat{\beta}_0)$. From now on let's assume that the intercept is included in $\hat{\beta}$. Viewed as a function over the p-dimensional input space, $f(X) = X^T\beta$ is linear, and the gradient $f'(X) = \beta$ is a vector in the input space that points in the steepest uphill direction.

There are several methods to fit a linear model to a dataset, but the most widely used is the OLS. In this approach, the coefficients β are picked such that they minimize the residual sum of squares RSS

$$RSS(\beta) = \sum_{i=1}^{N}(y_i - x_i^T\beta)^2. \tag{2.12}$$

Figure 2.9 shows the geometry of OLS fitting in the $\mathbb{R}^{p+1}$-dimensional space occupied by the the (X, Y) pairs.

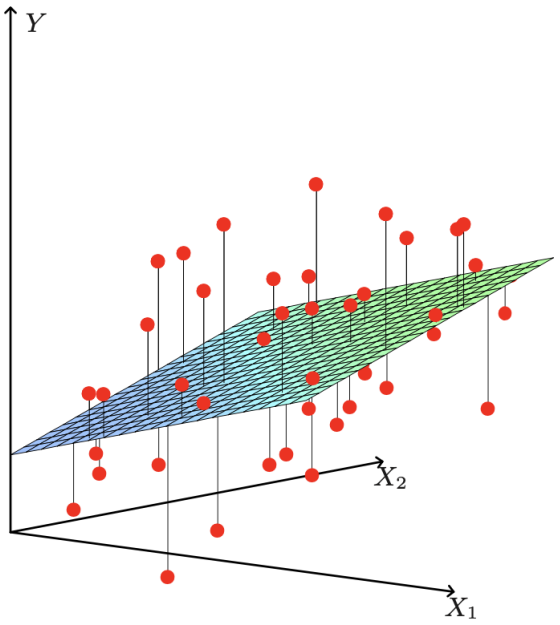


Figure 2.9: Linear least squares fitting with $X \in \mathbb{R}^2$. We seek the linear function of X that minimizes the sum of squared residuals from Y. *Adapted from:* Hastie, Tibshirani and Friedman (2009) [3].

Note that the RSS is a quadratic function, therefore its minimum will always exist. Yet, it may not be unique. Again, if the expression in 2.12 is written in matrix notation

$$RSS(\beta) = (y - X\beta)^T(y - X\beta), \tag{2.13}$$

where X is a $N \times p$ matrix in which each row is an input vector, and y is a $N-$vector of the

outputs.

Differentiating w.r.t. β we get the normal equations

$$\frac{\partial RSS}{\partial \beta} = -2X^T(y - X\beta) \tag{2.14}$$

$$\frac{\partial^2 RSS}{\partial \beta \partial \beta^T} = 2X^T X \tag{2.15}$$

Assuming that X has full column rank, and hence $X^T X$ is positive definite, we set the first derivative to zero

$$X^T(y - X\beta) = 0. \tag{2.16}$$

Now, if $X^T X$ is non-singular, the unique solution is given by

$$\hat{\beta} = (X^T X)^{-1} X^T y, \tag{2.17}$$

The predicted values at an input vector x_0 are given by $f(x_0) = (1 : x_0)\beta$ and the fitted values at the training inputs are

$$\hat{y} = X\hat{\beta} = X(X^T X)^{-1} X^T y, \tag{2.18}$$

where $y_i = f(x_i)$.

We denote the column vectors of X by $x_0, x_1, ..., x_p$. These vectors span a subspace of $\mathbb{R}^N$ - the column space of X. We minimize $RSS(\beta) = ||y - X\beta||^2$ by choosing $\hat{\beta}$ so that the residuals vector $y\hat{y}$ is orthogonal to this subspace. This orthogonality is expressed in 2.16, and the resulting estimate $\hat{y}$ is called the orthogonal projection of y onto this subspace. The hat matrix $H = X(X^T X)^{-1} X^T$ computes the orthogonal projection.

It might happen that the columns of X are not linearly independent, so that X is not full-rank. This would occur, for example, if two of the inputs were perfectly correlated - multicollinearity (refer to Appendix B). Then $X^T X$ is singular and the least squares coefficients $\hat{\beta}$ are not uniquely defined. However, the fitted values $\hat{y} = X\hat{\beta}$ are still the projection of y onto the column space of X, there is just more than one way to express that projection in terms of the column vectors of X. There is usually a natural way to resolve the non-unique representation, by recoding and/or dropping redundant columns in X with regularization, as we will see in subsection 2.2.3.2.

Multiple linear regression The linear model in 2.10, but with $p > 1$ inputs, is called the multiple linear regression model. Considering the univariate model (with no intercept), the least

squares estimate $\hat{\beta}$ and residuals r_i are given by

$$\hat{\beta} = \frac{\sum_1^N x_i y_i}{\sum_1 N x_i^2} \tag{2.19}$$

$$r_i = y_i - xi\hat{\beta} \tag{2.20}$$

In vector notation, we have $y = (y_1, ..., y_N)^T$, $x = (x_1, ..., x_N)^T$ and define the inner product between $\mathbf{x}$ and $\mathbf{y}$ as

$$\langle x, y \rangle = \sum_{i=1}^{N} x_i y_i = \mathbf{x}^T \mathbf{y}. \tag{2.21}$$

Rewritting $\hat{\beta}$ and r

$$\hat{\beta} = \frac{\langle x, y \rangle}{\langle x, x \rangle,} \tag{2.22}$$

$$\mathbf{r} = \mathbf{y} - \mathbf{x}\hat{\beta} \tag{2.23}$$

Suppose now that the inputs $x_1, x_2, ..., x_p$ are orthogonal, that is $\langle x_j, x_k \rangle = 0$ for all $j \neq k$. Therefore, the multiple least squares estimates $\hat{\beta}_j$ are equal to $\langle x_j, y \rangle / \langle x_j, x_j \rangle$ — the univariate estimates. In other words, when the inputs are orthogonal, they have no effect on each other's parameter estimates in the model.

Multiple-outputs Suppose there are multiple outputs $Y_1, Y_2, ..., Y_K$ to be predicted from the inputs $X_0, X_1, X_2, ..., X_p$. Assuming a linear model for each output

$$Y_k = \beta_{0k} + \sum_{j=1}^{p} X_j \beta_{jk} + \varepsilon_k \tag{2.24}$$

With N training cases, the model can be written in matrix notation as

$$Y = XB + E \tag{2.25}$$

Here $\mathbf{Y}$ is the $N \times K$ response matrix, with ik entry y_ik, $\mathbf{X}$ is the $N \times (p+1)$ input matrix, $\mathbf{B}$ is the $(p+1) \times K$ matrix of parameters and $\mathbf{E}$ is the $N \times K$ matrix of errors.

The multi-output loss function is a generalization of **??** in the single-output case

$$RSS(B) = \sum_{k=1}^{K} \sum_{i=1}^{N} (y_{ik} - f_k(x_i))^2 = tr[(Y - XB)^T (Y - XB)] \tag{2.26}$$

where tr is the *trace* of the matrix. Recall that the *trace* of a square matrix is the sum of elements on the main diagonal. Recall also that the *trace* of a square matrix which is the product of two real matrices can be rewritten as the sum of entry-wise products of their elements, that is, the sum of all elements of their Hadamard product, hence 2.26.

The estimates have the same form as for the single-ouput case

$$\hat{B} = (X^T X)^{-1} X^T Y. \tag{2.27}$$

Therefore, the coefficients for the *kth* outcome are simply the least squares estimates in the regression of y_k on $x_0, x_1, ..., x_p$. From here, it follows a very important property: multiple outputs do not affect one another's least squares estimates. This explains some results in Chapter 5 when multi-output models are discussed.

2.2.3.2 Regularized Linear Regression

Prediction accuracy in linear regression can, in some cases, be improved by using regularization methods - shrinking or setting some coefficients to zero. This strategy introduces a bit of bias to reduce the variance of the predicted values, and hence may improve the overall prediction accuracy. Any method that shrinks or sets to zero some of the least squares coefficients may result in a biased estimate. Picking the right model amounts to creating the right balance between bias and variance.

Ridge Regression or L_2-regularization

Ridge regression shrinks the regression coefficients by imposing a penalty on their size [66]. The ridge coefficients minimize a penalized RSS, which can be written in a *Lagrangian* form

$$\hat{\beta}^{ridge} = \underset{\beta}{\operatorname{argmin}} \{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \}. \tag{2.28}$$

where $\lambda \geq 0$ is a parameter that controls the amount of shrinkage of the coefficients β: the larger the value of λ, the greater the amount of shrinkage. The coefficients are shrunk toward zero (and each other).

Equation 2.28 can also be written as

$$\hat{\beta}^{ridge} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2, \tag{2.29}$$

$$\text{subject to the following constraint } \sum_{j=1}^{p} \beta_j^2 \leq t. \tag{2.30}$$

which makes the size constraint on the parameters more explicit. Each λ on 2.28 corresponds to one t on equation 2.30.

When there are many correlated variables in a linear regression model, their coefficients can become poorly determined and exhibit high variance. A very large positive coefficient on one variable can be, for example, canceled by a similarly large negative coefficient on another correlated variable. This is exactly the kind of effect the size constraint on the coefficients

expressed on equation 2.30 intends to mitigate.

From the equations above, it becomes clear that the solutions are not equivariant under scaling of the inputs. Therefore, the inputs should be standardized. It may be noticed that the intercept β_0 has not been included in the penalty term. This is because the penalization of the intercept would make the whole procedure depend on the origin chosen for Y, meaning that adding a constant c to each of the targets y_i would not simply result in a shift of the predictions by the same amount c.

The criterion in equation 2.28 can be written in matrix form as

$$RSS(\lambda) = (y - X\beta)^T(y - X\beta) + \lambda\beta^T\beta, \tag{2.31}$$

and the ridge solutions are given by

$$\hat{\beta}^{ridge} = (X^TX + \lambda I)^{-1}X^Ty, \tag{2.32}$$

where I is the $p \times p$ identity matrix. Notice that with the choice of quadratic penalty $\beta^T\beta$, the ridge regression solution is again a linear function of y. The solution adds a positive constant to the diagonal of X^TX before inversion. This makes the problem nonsingular, even if X^TX is not of full rank. This was the main motivation for ridge regression when it was first proposed [66].

To get a better perception of the effect of the ridge penalties in a linear regression coefficients, the *singular value decomposition* (SVD) of the centered input matrix $\mathbf{X}$ can be used. The SVD of the $N \times p$ matrix X is given by

$$X = UDV^T, \tag{2.33}$$

where U and V are $N \times p$ and $p \times p$ orthogonal matrices, in which the columns of U span the column space of X, and the columns of V span the row space of X. D is a $p \times p$ diagonal matrix, with diagonal entries $d_1 \geq d_2 \geq ... \geq d_p \geq 0$ called the singular values of X. If one or more values $d_j = 0$, then X is singular. Recall that a singular matrix is a square matrix that is not invertible, because its determinant is 0.

After some simplification, the least squares fitted vector can now be written as

$$X\hat{\beta} = X(X^TX)^{-1}X^Ty = UU^Ty, \tag{2.34}$$

Note that U^Ty are the coordinates of y with respect to the orthonormal basis U. Q and U are generally different orthogonal bases for the column space of X.

The ridge solutions can be rewritten as

$$X\hat{\beta}^{ridge} = X(X^TX + \lambda I)^{-1}X^Ty = UD(D^2 + \lambda I)^{-1}DU^Ty = \sum_{j=1}^{p} u_j \frac{d_j^2}{d_j^2 + \lambda} \tag{2.35}$$

where u_j are the columns of U. Since $\lambda \geq 0$, then $d_j^2/(d_j^2 + \lambda) \leq 1$. Like linear regression, ridge regression computes the coordinates of y with respect to the orthonormal basis U. It then shrinks these coordinates by the factors $d_j^2/(d_j^2 + \lambda)$. This means that a greater amount of shrinkage is applied to the coordinates of basis vectors with smaller d_j^2 (refer to Appendix B for an explanation on how ridge regression solves the problem of multicollinearity). To better understand what small d_j^2 means, the SVD of the centered matrix X is another way of expressing the principal components of the variables in X. Using the sample covariance matrix $S = X^T X/N$, and the original equation 2.33, we get

$$X^T X = VD^2V\hat{}T, \tag{2.36}$$

which corresponds to the *eigen decomposition* of $X^T X$. The eigenvectors v_j, which are the columns of V, are also called the principal components directions of X. The first principal component direction v_1 has the property that $z_1 = Xv_1$. has the largest sample variance amongst all normalized linear combinations of the columns of X. This sample variance is given by

$$Var(z_1) = Var(Xv_1) = \frac{d_1^2}{N} \tag{2.37}$$

with $z_1 = Xv_1 = u_1 d_1$. $z1$ is called the first principal component of X and u_1 is the normalized first principal component. Subsequent principal components, z_j, will have maximum variance d_j^2/N, as a consequence of being orthogonal to the principal ones. Conversely, the last principal component has minimum variance. Going back to the claim above that the small singular values d_j are the directions that ridge regression shrinks the most, that is because they correspond to directions in the column space of X having small variance.

Lasso Regression or L_1-regularization

The Lasso algorithm is another regularization method proposed by Tibshirani (1996) [67], and stands for Least Absolute Shrinkage and Selection Operator. The differences between Lasso and Ridge are subtle, but important.

The Lasso estimate is defined by

$$\hat{\beta}^{lasso} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2, \tag{2.38}$$

$$\text{subject to the constraint } \sum_{j=1}^{p} |\beta_j| \leq t. \tag{2.39}$$

and can also be written in the equivalent *Lagrangian form*

$$\hat{\beta}^{lasso} = \underset{\beta}{\operatorname{argmin}} \{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \}. \tag{2.40}$$

Notice the similarity to the ridge regression problem. The $L2$ ridge penalty $\sum_1^p \beta_j^2$ is replaced by the $L1$ lasso penalty $\sum_1^p |\beta_j|$. This latter constraint makes the solutions nonlinear in the y_i, and there is no closed form expression as in ridge regression. Computing the lasso solution is a quadratic programming problem.

Because of the nature of the constraint, making t sufficiently small will cause some of the coefficients to be exactly zero. Thus the lasso does a kind of continuous subset selection. If t is chosen larger than $t_0 = \sum_1 p|\hat{\beta}_j|$ (where $\hat{\beta}_j = \hat{\beta}_j^{OLS}$), then lasso estimates are the $\hat{\beta}_j$'s.

The lasso and ridge regressions are depicted on Figure 2.10, on the left and on the right, respectively, for two estimators (β_1 and β_2).

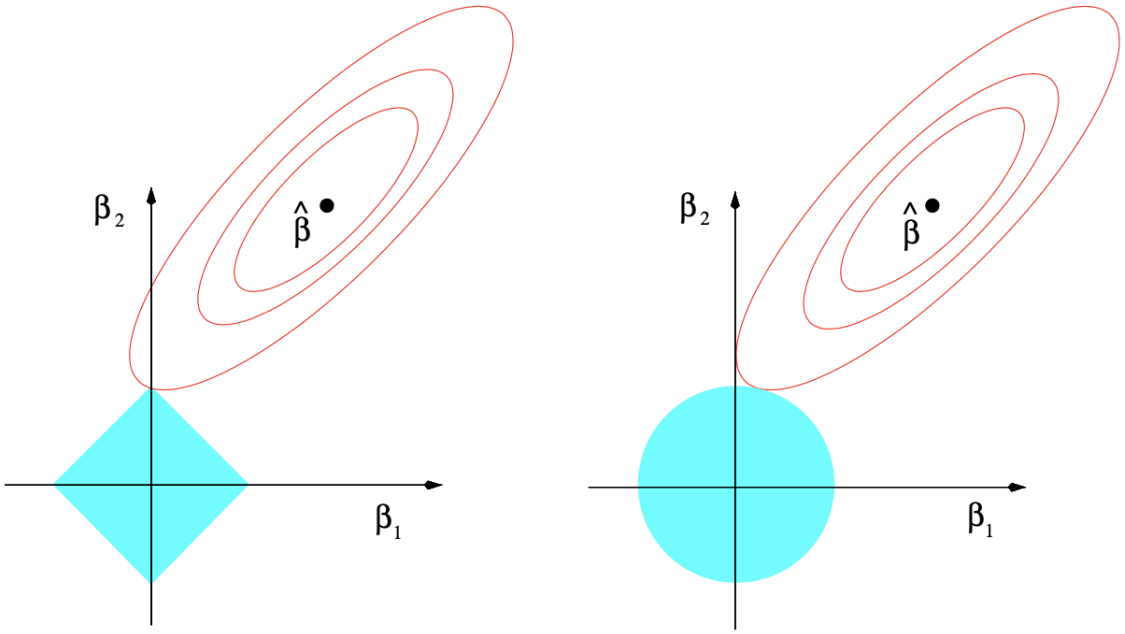


Figure 2.10: Estimations for the lasso (left) and ridge regression (right). The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t^2$, respectively, while the red ellipses are the contours of the least squares error function. *Adapted from:* Hastie, Tibshirani and Friedman (2009) [3].

The residual sum of squares has elliptical contours (shown in red), centered at the full least squares estimate $\hat{\beta}$. The constraint region for ridge regression is defined by the circle $\beta_1^2 + \beta_2^2 \leq t^2$, whereas that for lasso is a diamond-shape defined by $|\beta_1| + |\beta_2| \leq t$. The objective in both methods is to find the first point where the elliptical contours touch the constraint region. Unlike the circle, the diamond-shape has corners, thus, if the solution occurs at a corner, that coefficient β_j is equal to zero.

When the number of predictors is larger than 2, the diamond-shape becomes a rhomboid, which has many corners, flat edges and faces. Therefore, there are many more opportunities for the estimated parameters to be zero.

Elastic-Net

For computational tractability, Zou and Hastie (2005) introduced the elastic-net penalty [68].

The elastic-net selects variables like the lasso, and shrinks together the coefficients of correlated predictors like the ridge. It also has considerable computational advantages over the the L_q penalties.

Multiple output linear regression with regularization

As seen above, the least squares estimates in a multiple-output linear setting are simply given by the individual least squares estimates for each of the outputs (refer to equation 2.26).

Because of this property, when applying regularization methods in the multiple output case, one could either apply a univariate technique individually to each outcome or simultaneously to all outcomes, as the result will not change.

For example, in ridge regression, we could apply 2.32 to each of the K columns of the outcome matrix Y, using possibly different parameters λ, or apply it to all columns using the same value of λ. The former strategy would allow different amounts of regularization to be applied to different outcomes but require estimation of k separate regularization parameters $\lambda_1, ..., \lambda_k$, while the latter would allows all k outputs to be used in estimating the sole regularization parameter λ.

2.2.3.3 Support Vector Machines for Regression

Here, we start with the mathematical foundations for Support Vector Machines in classification problems. The concept is more easily illustrated for discrete target variables and the extension to the regression case can thus be better understood.

Support vector machines (hereafter, SVMs), also called support vector networks, map training examples to points in space as a way to maximise the width of the distance between two categories.

Depending on the relationships between the predictors and the target variables, it may be or not possible to separate the categories linearly. That is where the real power of SVMs resides: if the data is not separable in the original feature space, because, for example, there is significant overlap between the data points, then it is possible to map the inputs into a high- or even an infinite-dimensional feature space where those points can be linearly separable. This is called the *kernel trick*. The algorithm was first proposed by Vladimir Vapnik [69] and is based on a theory of statistical learning frameworks developed by himself and Alexey Chervonenkis between the 60s and the 90s [70].

Constructing the best hyperplanes The linear decision boundaries that explicitly separate the data into different classes are constructed by setting up an hyperplane.

A good guess for the best hyperplane is usually the one that defines the largest distance between the classes to separate, more specifically, the distance from the hyperplane to the nearest observation on each side should be maximized - that is called a *maximum-margin hyperplane* and defines a *maximum-margin linear classifier*. In general, the larger the *margin*, the better the generalization ability of the classifier, which in turn reduces the chances of overfitting (recall the Bias-Variance tradeoff in Subsection 2.2.2).

The learning algorithm consists in trying to find the best separating hyperplane by minimizing the distance of misclassified points to the decision boundary. If a target variable $y_i = 1$ is misclassified, then $x_i^T\beta + \beta_0 < 0$, and the opposite for a misclassified target with $y_i = 1$. The goal is to minimize

$$D(\beta, \beta_0) = -\sum_{i \in \mathcal{M}} y_i(x_i^T\beta + \beta_0), \tag{2.41}$$

where M represents the set of misclassified observations. The result is proportional to the distance of the misclassified observations to the decision boundary defined by $\beta^T x + \beta_0 = 0$. To solve the minimization problem, the gradients w.r.t β and β_0 are respectively given by

$$\frac{\partial D(\beta, \beta_0)}{\partial \beta} = -\sum_{i \in \mathcal{M}} y_i x_i, \tag{2.42}$$

and

$$\frac{\partial D(\beta, \beta_0)}{\partial \beta_0} = -\sum_{i \in \mathcal{M}} y_i. \tag{2.43}$$

If the classes are linearly separable, then it can be demonstrated that the algorithm converges to a separating hyperplane in a finite number of steps.

However, the 'finite' number of steps can be very large: the smaller the gap, the longer the time to find it. This problem can be solved by seeking a hyperplane that is not in the original space, but in a much enlarged space obtained by creating many basis-function transformations of the original variables.

Considering the following optimization problem

$$\max_{\beta, \beta_0, ||\beta||=1} M \text{ subject to } y_i(x_i^T\beta + \beta_0) \geq M, i = 1, ...N. \tag{2.44}$$

The constraint ensures that all the points are at least at a signed distance M from the decision boundary, which is defined by β and β_0. The objective is to find the largest M that satisfies that condition and its associated parameters.

To get rid of the $||\beta|| = 1$ constraint, the conditions can be replaced with

$$\frac{1}{||\beta||} y_i(x_i^T\beta + \beta_0) \geq M \tag{2.45}$$

or, equivalently,

$$y_i(x_i^T\beta + \beta_0) \geq M||\beta||. \tag{2.46}$$

Since for any β and β_0 satisfying these inequalities, any positively scaled multiple of those satisfies them as well, so $||\beta||$ can be set arbitrarily to $1/M$, for mathematical convenience. Therefore, equation 2.44 is equivalent to

$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 \text{ subject to } y_i(x_i^T\beta + \beta_0) \geq 1, i = 1, ..., N. \tag{2.47}$$

The constraints define an empty margin around the linear decision boundary of thickness $1/||\beta||$, so β and β_0 should be chosen to maximize it. This is a convex optimization problem (quadratic criterion with linear inequality constraints). The Lagrange function, to be minimized w.r.t. β and β_0 is

$$L_P = \frac{1}{2}||\beta||^2 - \sum_{i=1}^{N} \alpha_i[y_i(x_i^T\beta + \beta_0) - 1]. \tag{2.48}$$

where α_i are the Lagrange multipliers.

Setting the derivatives to zero gives

$$\beta = \sum_{i=1}^{N} \alpha_i y_i xi, \tag{2.49}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i. \tag{2.50}$$

Substituting these in equation 2.48, the following is obtained

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha k y_i y_k x_i^T x_k, \text{ subject to } \alpha_i \geq 0. \tag{2.51}$$

The solution can be obtained by maximizing L_D, a simpler convex optimization problem than solving for the primal, which besides satisfying the conditions in 2.49, 2.50 and 2.51, must also satisfy

$$\alpha_i[y_i(x_i^T\beta + \beta_0) - 1] = 0 \forall i. \tag{2.52}$$

Altogether, these constitute the so-called Karush–Kuhn–Tucker conditions.

If the lagrange multiplier $\alpha_i > 0$,then $y_i(x_i^T\beta + \beta_0) = 1$, meaning that x_i is on the boundary. If $y_i(x_i^T\beta + \beta_0) > 1$, x_i is not on the boundary, and $\alpha_i = 0$.

From 2.49, the solution vector β is defined in terms of a linear combination of the *support points* x_i — the points on the boundary via $\alpha_i > 0$.

The optimal separating hyperplane produces a function $\hat{f}(x) = \hat{x}^T\hat{\beta} + \hat{\beta}_0$ for classifying new observations

$$\hat{G}(x) = \hat{f}(x). \tag{2.53}$$

The intuition is that a large margin on the training data will lead to good separation on the test data. $f(x)$ gives the signed distance from a point x to the hyperplane $f(x) = x^T\beta + \beta_0 = 0$. Since the classes are separable, we can find a function f(x) = xT + 0 with yif(xi) > 0 i. Hence we are able to find the hyperplane that creates the biggest margin between the training points for both classes. This corresponds to the optimization problem

$$\max_{\beta,\beta_0,||\beta||=1} M, \text{ subject to } y_i(x_i^T\beta + \beta_0) \geq M, i = 1, ..., N, \tag{2.54}$$

where M is called the *margin*, represented in Figure 2.11 by a band which is M units away from the hyperplane on either side, hence $2M$ in total.

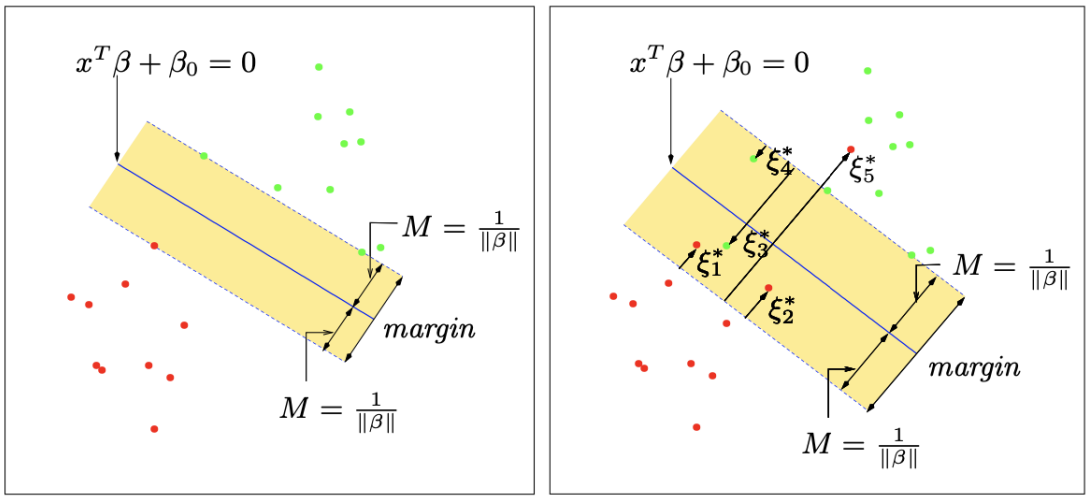


Figure 2.11: Support vector classifiers. (Left) separable case. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width $2M = 2/||\beta||$. (Right) nonseparable/overlapping case. The points labeled ζ_j^* are on the wrong side of their margin by an amount $\zeta_* = M\zeta_j$, whereas the points on the correct side have $\zeta_* = 0$. The margin is maximized subject to a total budget $\sum \zeta_i \leq constant$. Hence $\sum \zeta_j^*$ is the total distance of points on the wrong side of their margin. *Adapted from* Hastie, Tibshirani and Friedman (2009) [3].

Non-separable/overlapping case In the case of classes overlapping in the feature space, the objective is still to maximize M, but allow for some points to be on the wrong side of the *margin*.

Defining the slack variables $\zeta = (\zeta_1, \zeta_2, ..., \zeta_N)$, the constraint in equation 2.54 can be modified

$$y_i(x_i^T\beta + \beta_0) \geq 1(M - \zeta_i), \tag{2.55}$$

$\forall i, \zeta_i > 0, \sum_{i=0}^{N} \zeta_i \leq constant$. The value ζ_i in the constraint 2.55 is the proportional amount

by which the prediction $f(x_i) = x_i^T\beta + \beta_0$ is on the wrong side of its *margin*. Therefore, by bounding the sum $\sum \zeta_i$, we bound the total proportional amount by which predictions fall on the wrong side.

Dropping the norm constraint on β, and defining $M = 1/||\beta||$, the support vector classifier for the non-separable case can be written as

$$\min ||\beta|| \text{ subject to } \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \zeta_i \forall i, \\ \zeta_i \geq 0, \sum \zeta_i \leq constant. \end{cases} \tag{2.56}$$

The right panel of Figure 2.11 illustrates this non-separable case.

The criterion in equation 2.56 implicitly shows that points that are well inside their class boundary do not have a strong effect in shaping that boundary.

Computing the support vector classifier The problem defined by 2.56 is quadratic with linear inequality constraints, hence it is a convex optimization problem. Such as in the separable case, a quadratic programming solution using Lagrange multipliers is briefly described.

Computationally it is convenient to re-express 2.56 in the equivalent form

$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 + C\sum_i^N \zeta_i, \tag{2.57}$$

$$\text{subject to } \zeta_i \geq 0, y_i(x_i^T\beta + \beta_0) \geq 1 - \zeta_i \forall i, \tag{2.58}$$

where the parameter C replaces the *constant* from the second case in equation 2.56. For linearly separable classes, $C = \infty$.

The Lagrange (primal) function is given by

$$L_P = \frac{1}{2}||\beta||^2 + C\sum_{i=1}^N \zeta_i - \sum_{i=1}^N \alpha_i[y_i(x_i^T\beta + \beta_0) - (1 - \zeta_i)] - \sum_{i=1}^N \mu_i\zeta_i, \tag{2.59}$$

which must be minimized w.r.t β, β_0 and ζ_i. Setting the respective derivatives to zero gives

$$\beta = \sum_{i=1}^N \alpha_i y_i x_i, \tag{2.60}$$

$$0 = \sum_{i=1}^N \alpha_i y_i, \tag{2.61}$$

$$\alpha_i = C - \mu_i, \forall i \tag{2.62}$$

with α_i, μ_i, $\zeta_i \geq 0, \forall i$. α_i are Lagrange multipliers. Substituting 2.60, 2.61 and 2.62 in 2.59, the

Lagrange dual objective function is obtained

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'}, \tag{2.63}$$

which gives a lower bound on the objective function 2.58 for any point. L_D is maximized, subject to $0 < \alpha_i < C$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$. In addition to 2.60-2.62, the Karush–Kuhn–Tucker conditions include the following constraints

$$\alpha_i[y_i(x_i^T \beta + \beta_0)(1\zeta_i)] = 0, \tag{2.64}$$

$$\mu_i \zeta_i = 0, \tag{2.65}$$

$$y_i(x_i^T \beta + \beta_0)(1\zeta_i) \geq 0, \tag{2.66}$$

for $i = 1, 2, ..., N$. Altogether, equations 2.60-2.66 uniquely characterize the solution to the primal and dual problem.

From equation 2.60, the solution for β has the form

$$\hat{\beta} = \sum_{i=1} N \hat{\alpha}_i y_i x_i, \tag{2.67}$$

with nonzero coefficients $\hat{\alpha}_i$ only for the observations i for which the constraints in 2.66 are exactly met due to 2.64.

These observations are called the *support vectors*, since β_0 is represented in terms of them alone. Among these support points, some will lie on the edge of the margin ($\hat{\zeta} = 0$), and hence from 2.65 and 2.62 will be characterized by $0 < \hat{\alpha}_i < C$; the remainder ($\hat{\zeta}_i > 0$) have $\hat{\alpha}_i = C$. From 2.64 we can see that any of these margin points ($0 < \hat{\alpha}_i, \hat{\zeta}_i = 0$) can be used to solve for β_0, and we typically use an average of all the solutions for numerical stability.

Formulating the primal problem as the dual problem is used to implement the *kernel trick*, when the classes are not linearly separable in the original input space. The Lagrangian dual problem is obtained by forming the Lagrangian of a minimization problem by using nonnegative Lagrange multipliers to add the constraints to the objective function, and then solving for the primal variable values that minimize the original objective function.

Given the solutions $\hat{\beta}_0$ and $\hat{\beta}$, the decision function can be written as

$$\hat{G}(x) = [\hat{f}(x)] = [x^T \hat{\beta} + \hat{\beta}_0] \tag{2.68}$$

The tuning parameter of this procedure is the cost parameter C. A large value of C will discourage any positive i, and lead to an overfit wiggly boundary in the original feature space; a small value of C will encourage a small value of , which in turn causes f(x) and hence the boundary to be smoother. The regularization parameter for the SVM classifier is the cost parameter C, or its inverse λ ($C = 1/\lambda$). Common usage is to set C high, leading often to somewhat overfitted classifiers.

The optimization algorithm to generate the weights β proceeds in such a way that only the support vectors determine the weights and thus the boundary.

For computational efficiency reasons, the primal optimization problem 2.59 and its solution can be represented using inner products of the input features transformed feature vectors $h(x_i)$. The Lagrange dual problem 2.63 adopts the form

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \langle h(x_i), h(x_{i'}) \rangle. \tag{2.69}$$

The solution function 2.60 can also be re-written as

$$f(x) = h(x)^T \beta + \beta_0 = \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i) \rangle + \beta_0. \tag{2.70}$$

Given α_i, β_0 can be determined by solving $y_i f(x_i) = 1$ in 2.70, for any x_i for which $0 < \alpha_i < C$.

Both 2.69 and 2.70 involve $h(x)$ only through inner products. In fact, we need not specify the transformation $h(x)$ at all, but require only knowledge of the kernel function K

$$K(x, x') = \langle h(x), h(x') \rangle \tag{2.71}$$

Well-known choices for K are

$$\textbf{dth-Degree polynomial: } K(x, x') = (1 + \langle x, x' \rangle)^d, \tag{2.72}$$

$$\textbf{Radial basis function: } K(x, x') = exp(-\gamma ||x - x'||^2), \tag{2.73}$$

$$\textbf{Sigmoid (neural networks): } K(x, x') = tanh(\kappa_1 \langle x, x' \rangle + \kappa_2) \tag{2.74}$$

Support vector regression The SVM algorithm can be adapted for regression by inheriting some of the properties of the SVM classifier. This was first proposed in 1996, again by Vapnik and co-workers ([71]). Recalling the linear regression model

$$f(x) = x^T \beta + \beta_0, \tag{2.75}$$

this can be generalized to non-linear cases. β can be estimated by minimizing

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V(y_i - f(x_i)) + \frac{\lambda}{2} ||\beta||^2, \tag{2.76}$$

where

$$V_\epsilon(r) = \begin{cases} 0, & \text{if } |r| < \epsilon, \\ |r| - \epsilon, & \text{otherwise.} \end{cases} \tag{2.77}$$

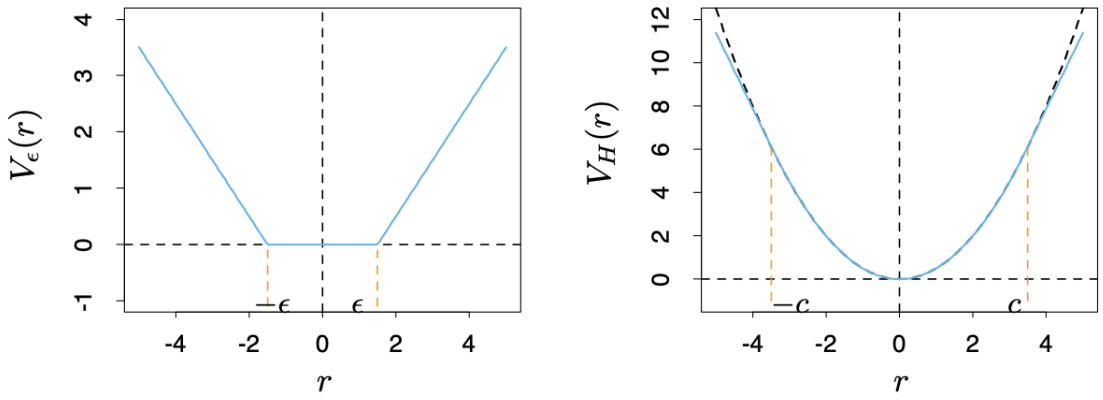


Figure 2.12: Examples of error measures used by the SVR machine. (Left) ϵ-insensitive error function. (Right) Error function used in Huber's robust regression. Beyond $|c|$, the function changes from quadratic to linear. *Adapted from* Hastie, Tibshirani and Friedman (2009) [3].

$V_\epsilon(r)$ is called an ϵ-insensitive error measure, which essentially ignores errors of size less than ϵ, as depicted in the left panel of Figure 2.12. Whereas in classification the points on the correct side of the decision boundary and far away from it are ignored in the optimization, in regression, these points are the ones with small residuals. It is interesting to contrast this with error measures used in robust regression in statistics. The most popular, due to Huber (1964), has the form

$$V_H = \begin{cases} \frac{r^2}{2}, & \text{if} |r| \leq c, \\ c|r| - \frac{r^2}{2}, & |r| > c, \end{cases} \tag{2.78}$$

shown in the right panel of Figure 2.12. This function reduces from quadratic to linear the contributions of observations with residuals with absolute values greater than a constant c. This makes the fitting less sensitive to outliers. The support vector error measure 2.77 also has linear tails (beyond ϵ), but in addition it flattens the contributions of those cases with small residuals.

If $\hat{\beta}$, $\hat{\beta}_0$ are the minimizers of H, the solution function has the form

$$\hat{\beta} = \sum_{i=1}^{N} (\hat{\alpha}_i{}^* - \hat{\alpha}_i x_i), \tag{2.79}$$

$$\hat{f}(x) = \sum_{i=1}^{N} (\hat{\alpha}_i{}^* - \hat{\alpha}_i)\langle x, x_i \rangle + \beta_0, \tag{2.80}$$

where $\hat{\alpha}_i{}^*$ and $\hat{\alpha}_i$ are positive and solve the quadratic programming problem

$$\min_{\alpha i,\, \alpha_i^* \epsilon} \sum_{i=1}^{N} (\alpha_i^* + \alpha_i) - \sum_{i=1}^{N} y_i(\alpha_i^* - \alpha_i) + \frac{1}{2} \sum_{i,i'=1} (\alpha_i^* - \alpha_i)(\alpha_{i'}^* - \alpha_{i'})\langle x_i, x_{i'} \rangle. \tag{2.81}$$

subject to the following constraints

$$0 \leq \alpha_i, \alpha_i^* \leq \frac{1}{\lambda}, \tag{2.82}$$

$$\sum_{i=1}^{N}(\alpha_i^* - \alpha_i) = 0, \tag{2.83}$$

$$\alpha_i \alpha_i^* = 0. \tag{2.84}$$

Due to the nature of these constraints, typically only a subset of the solution values $(\hat{\alpha_i}^* - \alpha_i)$ are nonzero, and the associated data values are called the support vectors. As was the case in the classification setting, the solution depends on the input values only through the inner products $\langle x_i, x_i \rangle$. Thus we can generalize the methods to richer spaces by defining an appropriate inner product, for example, one of those defined in 2.72, 2.73 or 2.74. Note that there are parameters, η and λ, associated with the criterion (11.36). These seem to play different roles. ϵ is a parameter of the loss function V_ϵ, just like c is for V_H. Note that both V_ϵ and V_H depend on the scale of y and thus, on r.If the target is re-scaled to $V_H(r/\sigma)$ and $V_\epsilon(r/\sigma)$ instead, then preset values for c and ϵ can be used.

2.2.3.4 Nearest-neighbors

k-Nearest Neighbors (kNN) is a supervised algorithm in which the basic idea is to find K nearest data points in the training space to the new data point and then 'classify' the new data point based on the majority class (classification) or the average (regression) among the k nearest data points. It is a lazy learner as it does not require the fitting of a model, and instead tries to match the test instance with k training examples.

The kNN regressor is quite different from the classifier. As in a regressor the dependent variable is continuous, it is scattered throughout the coordinate plane. When there is a new data point, the number of neighbors k is found by any of the distance metrics. After finding the neighbors, the predicted value of the new data point is the average of all the neighbor's values combined, as shown on the right of Figure 2.13. Typical distance metrics include the Euclidean, Manhattan or Minkowski distances, as well as the cosine similarity.

Unlike linear regression methods, k-nearest-neighbours' algorithms do not rely on any strong assumptions about the underlying data. However, any subregion of the decision boundary depends on a collection of input points and their specific positions, and thus, it is wiggly and unstable — high variance and low bias. The choice of k value depends on the dataset and the problem. A smaller k value can lead to overfitting, while a larger value of k can lead to underfitting.

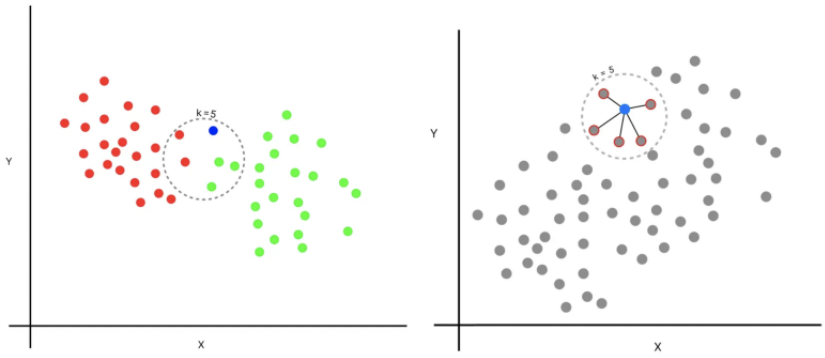


Figure 2.13: Nearest neighbors algorithm. k is the number of neighbors considered. (Left) Classification. (Right) Regression.

2.2.3.5 Decision Trees

A decision tree is a non-parametric supervised learning algorithm, which can be used for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes.

Tree-based methods partition the feature space into a set of rectangular regions, and then fit a simple model on each one (i.e. a constant). Despite being conceptually simple, they are in general powerful and extremely interpretable compared to other algorithms. Interpretability is the key advantage of decision trees.

Consider a regression problem with continuous target variable Y and inputs X_1 and X_2. The feature space is partitioned by lines that are parallel to the coordinate axes (in this case, X_1 and X_2.) In each partition, the target Y can be modelled with a different constant, for example. To simplify the explanation, recursive binary partitions will be assumed, such as the one depicted in the top right panel of Figure 2.14, where all regions are easy to describe. First, the space is split into two regions, and the target variable is obtained by the mean of Y in each region, provided the feature X_1 or X_2 and the split-point to achieve the best fit. Then, one or both of these regions are split into two other regions, and the process continues until some stopping criteria is applied. For example, in the left panel of Figure 2.14, the first split uses $X_1 \leq t_1$. Then the region $X_1 \leq t_1$ is split at $X_2 t_2$ and the region $X_1 > t_1$ is split at $X1 \leq t3$. Finally,the region $X_1 > t_3$ is split at $X_2 = t_4$. The final result is a partition into five regions $R_1, R2, ..., R_5$. The corresponding regression model predicts Y with a constant c_m in region R_m,

$$\hat{f}(X) = \sum_{m=1}^{5} c_m I(X_1, X_2) \in R_m \tag{2.85}$$

where m is the maximum number of partitions allowed.

This same model can be illustrated by the binary tree in the top left panel of Figure 2.14. At the top of the tree, there is the whole dataset. Then, for each level, observations that satisfy the condition at each node are assigned to the left side, and the others to the right side. The leaves

(terminal nodes) correspond to the regions $R_1, ..., R5$. The bottom panel on that same Figure is simply a 3-dimensional perspective of the regression surface from this model. For more than two inputs, representations are difficult to draw, but the binary tree representation follows exactly the same procedure.

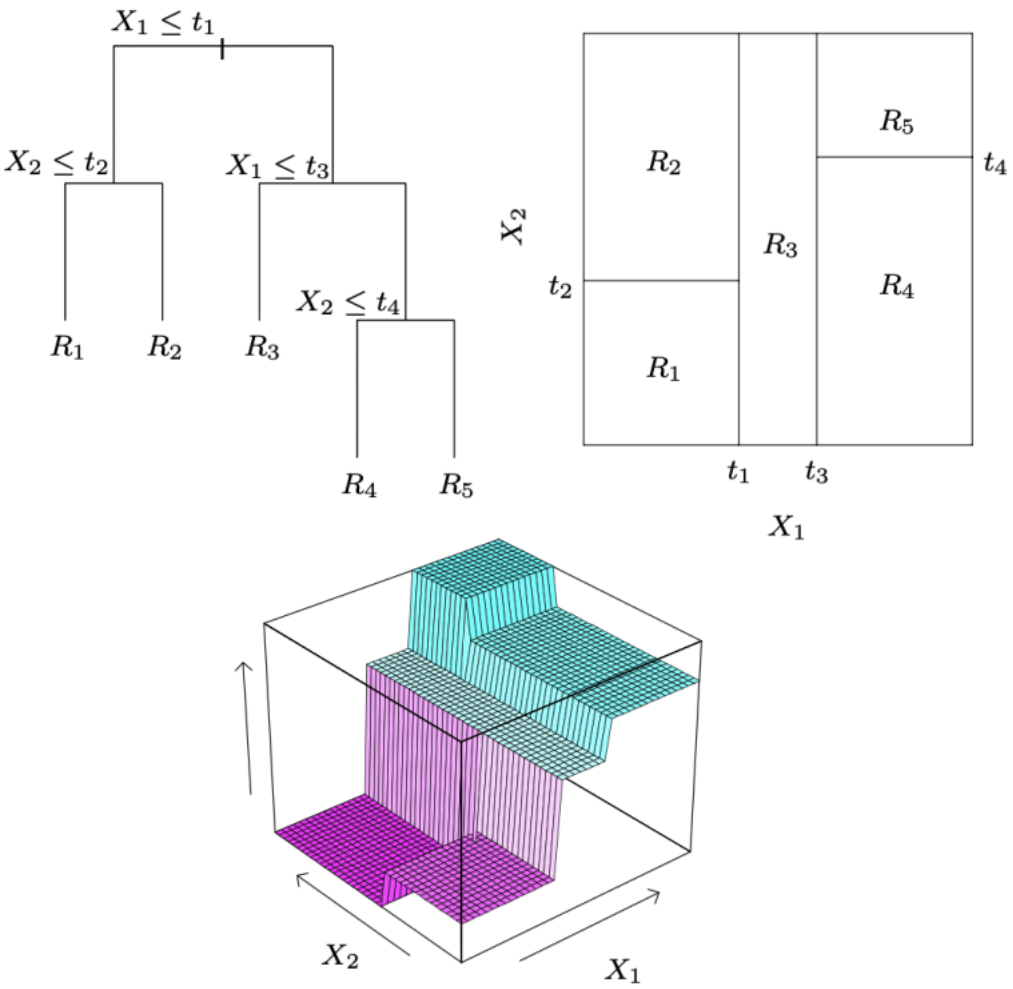


Figure 2.14: Partitions and Decision Trees. Top left panel shows the tree corresponding to the partition in the top right panel. Top right panel shows a partition of a two-dimensional feature space by recursive binary splitting. Bottom panel is a perspective plot of the prediction surface.Adapted from: Hastie, Tibshirani and Friedman (2009) [3].

Advantages of decision trees include their high interpretability because of their hierarchical nature, flexibility due to being insensitive to underlying relationships (correlations) between attributes (if two variables are highly correlated, the algorithm will only choose one of the features to split on) and little data preparation.

Regression Trees Consider a set of p inputs and a target variable for each of N observations: (x_i, y_i) for $i = 1, 2, ..., N$, with $x_i = (x_{i1}, x_{i2}, ..., x_{ip})$. The algorithm must automatically decide on the splitting variables and split points and, consequently, on the topology of the tree as well.

Supposing a partition into M regions $R_1, R_2, ..., R_M$, the response can be modelled as a constant c_m in each region:

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m). \tag{2.86}$$

If the chosen minimization criterion is the sum of squares $\sum(y_i f(x_i))^2$, it is easy to see that the best $\hat{c}_m$ is just the average of y_i in region R_m:

$$\hat{c}_m = \text{avg}(y_i | x_i \in R_m). \tag{2.87}$$

However, finding the best partition in terms of minimum sum of squares is generally computationally infeasible and a greedy algorithm is generally used. It starts with all the data, and the choice of a splitting variable j and a split point s. With this, a pair of half-planes is defined

$$R_1(j, s) = X | X_j \leq s \text{ and } R_2(j, s) = X | X_j > s. \tag{2.88}$$

This is followed by the choice of the splitting variable j and split point s that solve the following minimization

$$\min_{j,s}[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2]. \tag{2.89}$$

For any j and s, the inner minimization above is solved by

$$\hat{c_1} = \text{avg}(y_i | x_i \in R_1(j, s)) \text{ and } \hat{c_2} = \text{avg}(y_i | x_i \in R_2(j, s)) \tag{2.90}$$

For each splitting variable j, the determination of the split point s is relatively fast. Having found the best split, the data is partitioned into the two resulting regions and the splitting process is repeated on each of the two regions and successively for all of the resulting regions. A legitimate question is when to stop this process and hence, to stop growing the tree. Whilst a very large tree may overfit due to adjusting excessively to the characteristics of the training data, a small one runs the risk of not even capturing the basic structure of the data. Therefore, the tree size is an important model hyperparameter that must be tuned from the data. A generally accepted strategy relies on growing a large tree T_0 first and stop the splitting process only when some minimum node size is reached. Then, this large tree can be pruned. The preferred strategy is to grow a large tree T_0, stopping the splitting process only when some minimum node size is reached. Then this large tree is pruned, that is, collapsing any number of its internal nodes.

A subtree $T \subset T_0$ is defined as any tree that can be obtained by pruning T_0. We index terminal nodes by m, with node m representing region R_m. Let $|T|$ denote the number of terminal

nodes in T. Letting

$$N_m = \#\{x_i \in R_m\}, \tag{2.91}$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i, \tag{2.92}$$

$$Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2, \tag{2.93}$$

the cost-complexity criterion is defined as

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T|. \tag{2.94}$$

The idea is to find, for each α, the subtree $T_{alpha} \subseteq T_0$ to minimize $C_\alpha(T)$. The tuning parameter $\alpha \geq 0$ controls the trade-off between tree size and its goodness of fit to the data. Large values of α result in smaller trees, and the opposite for smaller values of α As the notation suggests, with $\alpha = 0$ the solution corresponds to the full tree T_0.

The major problem with trees is their high variance. Often a small change in the data can result in a very different series of splits.The main reason for this instability is the hierarchical nature of the process: the effect of an error in the top split is propagated down to all of the splits below it. Bagging and Boosting approaches in section 2.2.3.6 below employ many trees to reduce this high variance.

2.2.3.6 Model ensembles

The idea of ensemble learning is to build a prediction model by combining the strengths of a collection of simpler base models. They are among the most powerful techniques in machine learning, often outperforming other methods. This comes at the cost of increased algorithmic and model complexity.

Random Forests

Bagging or bootstrap aggregation is a technique for reducing the variance of an estimated prediction function. Bagging seems to work especially well for high-variance, low-bias procedures, such as trees. For regression, we simply fit the same regression tree many times to bootstrap-sampled versions of the training data, and average the result, as shown in Figure 2.15. For classification, each of the trees casts a vote for the predicted class. Random forests is a substantial modification of bagging that builds a large collection of decorrelated trees, and then averages them [72, 73].

Since trees are notoriously noisy, they benefit greatly from the averaging. Moreover, since each tree generated in bagging is identically distributed, the expectation of an average of B such trees is the same as the expectation of any one of them. This means the bias of bagged trees is

the same as that of the individual trees, and the only hope of improvement is through variance reduction. This is in contrast to boosting, where the trees are grown in an adaptive way to remove bias, and hence are not identically distributed.

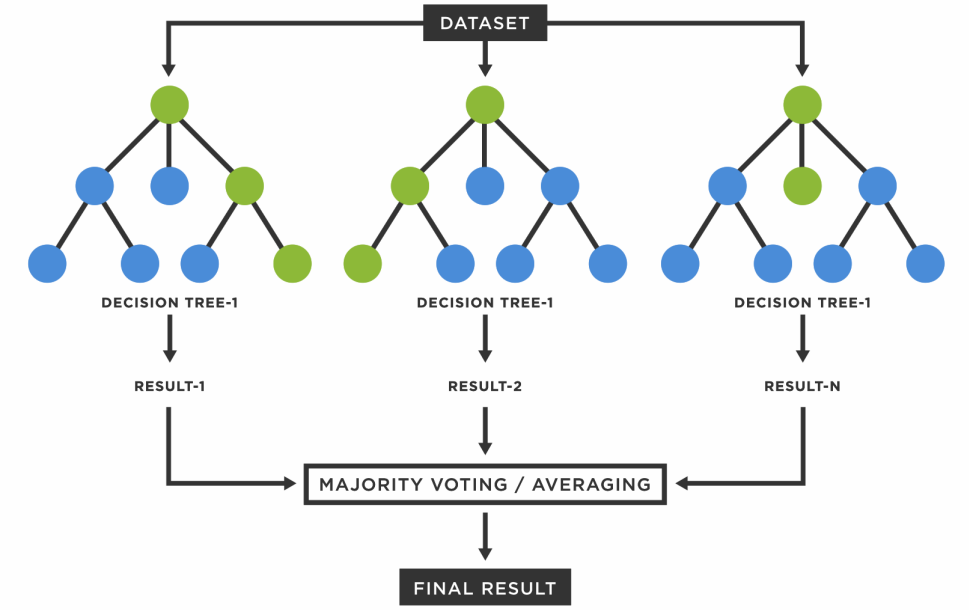


Figure 2.15: Schematic showing the workflow of the Random Forest Algorithm.[7]

The idea in random forests is to improve the variance reduction of bagging by reducing the correlation between the trees, without increasing the variance too much. This is achieved in the tree-growing process through random selection of the input variables. The algorithm 1 shows the process of growing a tree on a bootstrapped dataset.

[7]Source: https://www.tibco.com/reference-center/what-is-a-random-forest

Algorithm 1 Random Forest - Regression

Input: $\{x_i, y_i\}_{i=1}^{N}$
Output:

for $b = 1$ to B trees **do**
    bootstrap a sample Z^* of size N from the training data
    grow a tree T_b with the bootstrapped data
    while minimum node size n_{min} not reached **do**
        for each terminal node **do**
            select m of the p variables at random
            choose best variable/split-point among m (maximizes information gain or minimizes the error)
            split node into two nodes
        end for
    end while
end for
output the ensemble of trees $\{T_b\}_1^B$

Predict new point x:
Regression: $\hat{f}_{rf}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$
Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth tree. Then $\hat{C}_{rf}^B(x) = majorityvote\{\hat{C}_b(x)\}_1^B$

Gradient Boosting Trees

Despite both algorithms using decision trees as base models, there are two main differences between random forests and gradient boosting. Whereas the former constructs trees independently of one another, the latter is trained sequentially, one tree at a time, each to correct the errors of the previous ones, as illustrared in Figure 2.16. For this reason, an important conclusion is that random forests can be trained in parallel, but not the gradient-boosted trees. The other main difference is in how each model output decisions. Since the trees in a random forest are independent, their outputs can be determined in any order. The individual predictions are then aggregated into a collective one: the majority class in classification problems or the average value in regression. On the other hand, the gradient boosting trees run iteratively in a fixed order, and that sequence cannot change. For that reason, they admit only sequential evaluation.

The description provided here follows closely the section on gradient boosting trees from Hastie, Tibshirani and Friedman (2009) [3] and Friedman's paper [74], where Gradient Boosting Machines have been initially proposed.

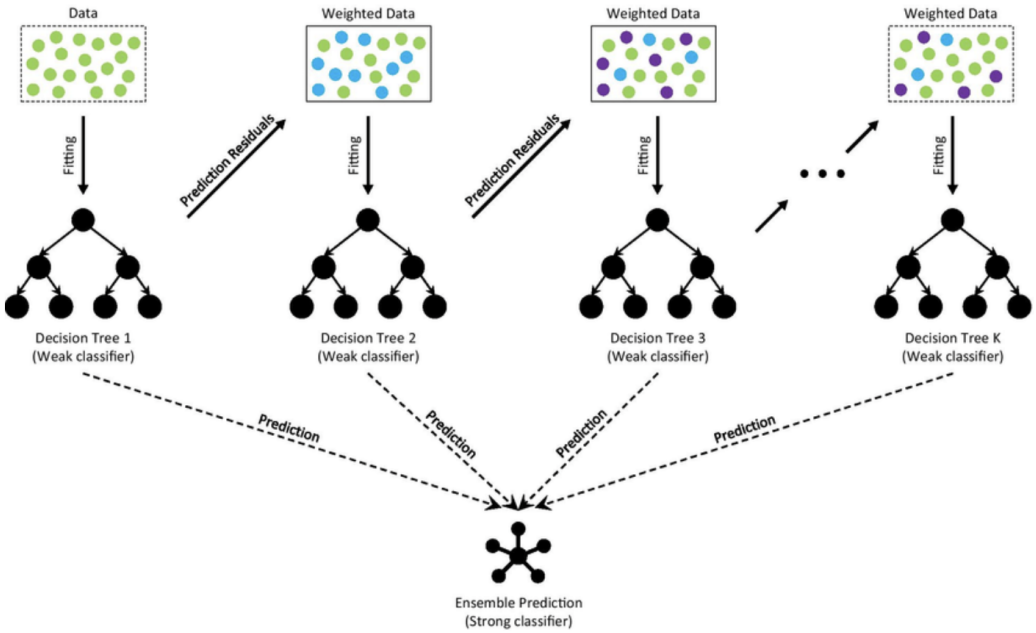


Figure 2.16: Wotkflow of the Gradient Boosting Algorithm.

Algorithm In many supervised learning problems there is an output variable y and a vector of input variables x, related to each other with some probabilistic distribution. The goal is to find some function $\hat{F}(x)$ that best approximates the output variable from the values of input variables. This is formalized by introducing some loss function $L(y, F(x))$ and minimizing it in expectation

$$\hat{F}(x) = \underset{F}{\operatorname{argmin}}\, \mathbb{E}_{x,y}[L(y, F(x))]. \tag{2.95}$$

The gradient boosting method assumes a real-valued y. It seeks an approximation $\hat{F}(x)$ in the form of a weighted sum of M functions $h_m(x)$ from some class H $\mathcal{H}$, called base (or weak) learners:

$$\hat{F}(x) = \sum_{m=1}^{M} \gamma_m h_m(x) + const. \tag{2.96}$$

We are usually given a training set $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ of known sample values of x and corresponding values of y. In accordance with the empirical risk minimization principle, the method tries to find an approximation $\hat{F}(x)$ that minimizes the average value of the loss function on the training set, i.e., minimizes the empirical risk. It does so by starting with a model, consisting of a constant function $F_0(x)$, and incrementally expands it in a greedy fashion, as shown in algorithm 2.

$$F_0(x) = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, \gamma), \tag{2.97}$$

$$F_m(x) = F_{m-1}(x) + (\underset{hm \in \mathcal{H}}{\text{argmin}}[\sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + h_m(x_i))])(x). \tag{2.98}$$

Algorithm 2 Gradient Regression Tree Boosting

Input: $\{x_i, y_i\}_{i=1}^{N}; \mathcal{L} : lossfunction$
Output: $f_M = f_0 + \sum_{m=1}^{M} \gamma_m h_m$ ▷ Sum of residuals of every weaker learner/tree

Initialize $F_0(x) = \text{argmin}_\gamma \sum_{i=1}^{N} \mathcal{L}(y_i, \gamma)$
for $m = 1$ to M **do**
  for $i = 1, ..., N$ **do**
    $r_{im} = -[\frac{\partial L(y_i, f(x_i}{\partial f(x_i)}]_{f=f_{m-1}}$ ▷ calculate the pseudo-residuals
  end for
  Fit a regression tree to the targets r_{im}giving terminal regions $R_{jm}, j = 1, ...J_m$
  for $j = 1, ..., J_m$ **do**
    $\gamma_{jm} = \text{argmin}_\gamma \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i + \gamma.$
  end for
  Update $f_m(x) = f_{m-1}(x) \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$
end for
Output $\hat{f}(x) = f_M(x)$

Two basic tuning parameters are the number of iterations M and the sizes of each of the constituent trees $J_m, m = 1, ..., M$. A frequent problem is that trees tend to be too large, mainly during the early iterations, which on top of degrading performance, also significantly increases computation time. The simplest strategy to avoid this is to restrict all trees to be the same size, $J_m = J \forall m$. At each iteration a J terminal node regression tree is induced and thus it becomes a hyper-parameter of the entire boosting procedure, to be adjusted to maximize estimated performance. This suggests that the value chosen for J should reflect the level of dominant interactions of $\eta(x)$. Even though this is not generally known, it is straightforward to conclude that in most situations it will tend to be low. Experience indicates that $4 \leq J \leq 8$ works well in the context of boosting, with results being fairly insensitive to values within this range [3].

Regularization Besides the size of the trees (J), another important parameter of gradient boosting is the number of boosting iterations M. Each iteration usually reduces the training risk $L(fM)$, so that for large enough M this risk can be made arbitrarily small. However, fitting the training data too well can lead to overfitting, which degrades the risk on future predictions. Thus, there is an optimal number M^* minimizing future risk that depends on the application. A convenient way to estimate M^* is to monitor prediction risk as a function of M on a validation sample. The value of M that minimizes this risk is taken to be an estimate of M^*.

Shrinkage Besides controlling the value of M, another way to apply regularization is through shrinkage techniques, as it is often done with ridge regression (see section 2.2.3.2) or neural networks. In the context of booting, this is normally implemented by scaling the

contribution of each tree by a factor ν (with $0 < \nu < 1$), which is often regarded as the learning rate. That consists in replacing line 10 of Algorithm 2 to

$$f_m(x) = f_{m-1}(x) + \nu . \sum_{j=1}^{J} \gamma_{jm} I(x \in R_{jm}). \tag{2.99}$$

Smaller values of ν, and thus more shrinkage, lead to an increased training risk for the same number of iterations M. Therefore, both ν and M control prediction risk on the training data. However, these parameters are apparently dependent on each other: smaller values of ν lead to larger values of M for the same training risk, revealing there is a trade-off between them. Empirically it has been found that smaller values of ν tend to result in smaller test errors, but require correspondingly larger values of M [74]. The best strategy appears to be to set ν to a very small value ($\nu < 0.1$) and then choose M by early stopping, that is, when the score on the validation set does not improve further after a certain number of iterations. This supposedly yields dramatic improvements over no shrinkage ($\nu = 1$) in regression problems.

Subsampling With stochastic gradient boosting , at each iteration we sample a fraction η of the training observations (without replacement), and grow the next tree using that subsample [75]. The rest of the algorithm is identical. A typical value for η can be 1, although for large N, it can be substantially smaller. Not only does the sampling reduce the computing time by the same fraction η, but in many cases it actually produces a more accurate model.

Predictor variable importances The input predictor variables are seldom equally relevant. Often only a few of them have substantial influence on the response, and some of them, or even the vast majority, are irrelevant. For that reason, it is often very useful to learn the relative importance or contribution of each input variable in predicting the response, especially when the dimensionality of features is too big. For a single decision tree T, Breiman (1984) proposed a measure of the squared importance of a feature variable as

$$\mathcal{I}_l^2(T) = \sum_{t=1}^{J-1} \hat{\imath}_t^2 I(v(t) = l) \tag{2.100}$$

as a measure of relevance for each predictor variable X_l [76]. The sum in equation 2.100 is over the $J1$ internal nodes of the tree. At each node t, one of the input variables $X_{v(t)}$ is used to partition the region associated with that node into two subregions. Within each of those subregions a separate constant is fit to the target variables. The particular predictor variable chosen is the one that gives maximal estimated improvement $\hat{\imath}_t^2$ in squared error over that for a constant fit over the entire region. The squared relative importance of variable X_l is the sum of these squared error improvements over all internal nodes for which that variable has been chosen as the splitting variable.

The predictor variable importance $\mathcal{I}_l$ can be easily extended to additive tree expansions by a

simple averaging procedure over the trees, given by

$$\mathcal{I}_l^2 = \frac{1}{M} \sum_{m=1} M\mathcal{I}_l^2(T_m). \tag{2.101}$$

Averaging provides a stabilizing effect, and as such this measure becomes reliable than its counterpart in equation 2.100 for a single tree. Also, because of shrinkage, the masking of important variables by others with which they are highly correlated is much less of a problem. As a side note, both equations 2.100 and 2.101 refer to squared importance and so, the actual importance is given by its square root.

Chapter 3

State of the Art

Large-scale sky surveys are observing massive amounts of stellar spectra. As a consequence, astronomy has been experiencing a rapid growth in data size and complexity over the last decade.

As seen above in this manuscript (see Section 2.1), the extraction of useful physical information from these large spectral databases, mainly stars' atmospheric parameters such as effective temperatures, surface gravities, metallicities and elemental abundance ratios, can be rigorously extracted one star at a time using well-established spectral analysis techniques. Needless to say, this also requires a small "army of experts" with loads of spare time and even greater loads of patience.

After describing some foundations of spectroscopy and the classical physics techniques used to analyze stellar spectra on Section 2.1, as well as reviewing several important machine learning algorithms on Section 2.2, this chapter is mostly concerned with studies that addressed the task of estimating stellar atmospheric parameters using machine or deep learning approaches. As abovementioned, classical methods in astronomy to derive these parameters have already been discussed on Section 2.

Machine learning, especially its deep learning branch, is an increasingly popular tool and is used in several fields of science. This kind of learning is generally data-driven and has proven accurate in predicting outcomes without the need for the user to explicitly create a method to solve the problem at hand. The algorithms in machine learning receive input data and by applying statistical analysis, they predict an output value within a reasonable range (recall Section 2.2).

The interest for machine learning algorithms and automatic processes in astronomy emerged naturally from the increasing volume of the data. Despite being an apparently appealing area for machine learning and data science in general, the application of these methodologies to astronomical data is relatively recent and has only kicked-off for good in the past decade. This did not however prevent the incredibly fast-growing tendency observed in the field, which already sums up a reasonable catalogue of studies that tried to address the problem in multiple ways.

In fact, from an historical perspective, some authors have pursued the development of methods for obtaining estimates of atmospheric parameters in the past that closely resemble techniques today encompassed in the machine learning category. For example, Jones (1966), in a pioneering effort, performed a principal component analysis of visual estimates of 10 line ratios and six line strengths for a uniform set of photographic spectra obtained with the Palomar 200 inch telescope [77, 78]. The three largest principal components were then calibrated with temperature, luminosity, and metal abundance. Principal component analysis is today considered a gold-standard in machine learning, not necessarily as a predictive modelling approach, but as a dimensionality reduction algorithm. Thévenin and Foy (1983) also compared measured equivalent widths for several prominent spectral lines in low-resolution spectra with grids of theoretical equivalent widths obtained from model atmospheres in what mimics the comparison with a ground-truth value today in supervised machine learning [79].

Currently, the type of algorithms found in the literature varies according to the type of input data, essentially whether that is based on quite extensive preprocessed data extracted from spectra using other prior and complementary methods or, alternatively, from raw spectra with little to no prior treatment. Based on this, they can probably be divided in two major categories: network-dependent and network-independent models. Whereas the former use neural networks, either feed-forward, deep or convolutional, the latter use a variety of other machine learning techniques that will be better described below. Note that in this thesis, neural networks were not employed as a first attempt to solve the problem, thus these were neither explained nor even mentioned on Subsection 2.2.3. Comprehensive details on neural networks can be found in the literature [3, 80].

Additionally, the type of problem to solve also varies, and both regression, to predict continuous numerical outcomes such as Teff, log g and [Fe/H], and classification strategies, to predict discrete non-numerical targets such as the stellar type and subclass, have been employed.

Regardless of the algorithmic choice, the automatic fit of synthetic spectra to get the correct parameters of an observed signal can be rather complex. Achieving a good predictive performance will depend on the number and type of parameters used to create the models, the number of points the models generate for every spectrum, and very importantly, the quality of the observed signal itself. Moreover, one of the biggest issues in spectral analysis is the continuum estimation, as briefly discussed on Section 2.1.3. As a consequence, methods that rely on an accurate determination of the continuum only work for some types of stars, specifically metal-poor and earliest types of M dwarfs [81].

3.1 Machine Learning

Starting with machine learning algorithms that do not use neural networks, multiple methods have been proposed to accomplish this goal using synthetic and/or observed spectra, including [25–31, 82].

One of the most well-known automatic platforms to predict stellar parameters and chemical abundances is probably *The Cannon* [82]. *The Cannon* uses very little spectra pre-processing, but still works under the assumption that all spectra have been continuum-normalized in a consistent manner and sampled on a similar wavelength grid. The method is essentially based on *label transfer* - a type of generative model (with approximately 80,000 parameters) that describes a probability density function for the flux at every pixel in a continuum normalized spectrum as a function of the labels - after which the labels are transferred to the observed stellar objects from a survey. Naturally, this assumes that the generative model holds for the objects in the survey. Altogether, despite not using any stellar physical models or line-list, *The Cannon* still requires a subset of objects within a survey - reference objects - for which the stellar labels are known and cover enough label space. This platform was initially trained on only 542 stars as reference objects, with Teff, log g and [Fe/H] as labels, and then validated on the spectra of 55 000 stars from the *APOGEE SDSS Data Release 10 (DR10)* [83]. APOGEE, which is part of the SDSS-III7, is a high resolution ($R \sim 22500$), high S/N (~ 100), spectroscopic survey consisting primarily of red giant stars spanning the bulge, disk, and halo of the Milky Way [84]. APOGEEs ASPCAP pipeline provided the stellar labels for these stars, which include stellar parameters and multiple elemental abundances, and is based on χ^2 fitting of the data to 1D LTE this may need to be briefly explained in the theoretical background I think models for seven labels. The root mean squared error differences between the ASPCAP and *The Cannon* are 95 K in Teff, 0.24 dex in log g and 0.08 dex in [Fe/H], with ranges spanning 3500-5300 K, 0.0-5.0 dex and -2.5-0.45 dex for each one of those labels, respectively. Since its release in 2015, *The Cannon* has been tested and improved further with both new target labels and new data [85].

Blanco and colleagues proposed the MATISSE (MATrix Inversion for Spectral SynthEsis) algorithm, which determines a basis function $B_\theta(\lambda)$ allowing the derivation of a particular stellar parameter θ [25]. This $B_\theta(\lambda)$ function is determined from an optimal linear combination of theoretical spectra and it essentially relates the variations in the spectrum flux with variations in the parameter θ. The algorithm is able to derive Teff, log g, metallicity, as well as some individual chemical abundances from the spectra. Depending on the S/N ratio, this method yielded errors in the range of 200-700 K for Teff, 0.1-0.6 dex for log g and 0.1-0.6 dex for metallicity, for sample of 1858 stars with temperatures between 4000 and 8000 K, log g between -1.0 and 5.0 dex and metallicities between -5.0-1.0 dex. However, MATISSE uses a large grid of synthetic spectra and is therefore as limited as other physics-based methods in that regard.

Li *et al.* used a combined Lasso-SVR approach to estimate the same target variables as the MATISSE. First, the Lasso algorithm extracts features from the spectra, which are then feeded onto a SVR model to estimate the parameters. In a stellar subsample of 20 000 spectra from the Sloan Digital Sky Survey (SDSS) with reference parameters provided by the SDSS/SEGUE Spectroscopic Parameter Pipeline, spanning the ranges of 4088-9749 K for Teff, 1.0 - 5.0 dex for log g and -3.5 - 0.27 for [Fe/H], this approach yielded mean absolute errors of 0.0074 dex for log Teff (equivalent to 102 K for Teff), 0.19 dex for log g, and 0.182 dex for [Fe/H] [27]. An important advantage worth mentioning is the interpretability of this strategy. With effect, of the 3821 fluxes present on each spectra, the LASSO detected 10, 19, and 14 typical wavelength

positions, for Teff, log g and [Fe/H] respectively, which are indeed related to typical lines of stellar spectra. This highlights the potential ability of a machine learning algorithm to capture features with physical meaning that can be further interpreted and used to validate previously obtained parameters or, perhaps, even instigate and propel new research questions.

In 2018, Fierro-Santillán *et al.* launched FITspec. The validation of the algorithm was performed by analyzing the spectra of a sample of eight O-type stars taken from the spectroscopic survey of Northern Galactic OB stars. The spectral lines used for the adjustment of the analyzed stars are reproduced with good accuracy. In particular, the effective temperatures calculated with the FITspec are in good agreement with those derived from spectral type and other calibrations for the same stars [26]. The stellar luminosities and projected rotational velocities are also in good agreement with previous quantitative spectroscopic analyses in the literature.

More recently, Karnavas *et al.* released ODUSSEAS (Observing Dwarfs Using Stellar Spectroscopic Energy-Absorption Shapes), an automatic platform based on the measurement of the pseudo-equivalent widths for more than 4000 stellar absorption lines that uses linear models and their variants, Ridge, Lasso and Elastic Net to derive Teff and [Fe/H] of M dwarfs using spectra obtained from different spectographs with different resolutions. The estimated results were around 30 K for Teff and 0.04 dex for [Fe/H] and were consistent for spectra with resolutions between 48 000 and 115 000 and a S/N ratio above 20 [30].

This year, Anders and co-workers, capitalizing on the *Gaia Data Release 3 (DR3)* that came out in mid-2022, proposed to release a stellar-parameter catalogue for more than 300 million Gaia stars using the Extreme Gradient Boosting algorithm using the new Gaia XP spectra, which are very suitable to build a tabular dataset [31].

3.2 Deep Learning

On the other hand, the majority of studies to derive quantitative stellar parameters or stellar classes use neural networks [32–42].

Neural networks have the advantage of inherently handling multi-output responses (that is, predicting multiple target variables simultaneously), depending on the number of neurons in the last layer. Regarding other machine learning algorithms, there is a limited number that inherently supports a real multi-output setting, including only linear regression and variants, as well as decision-trees and tree-based Random Forests. Some solutions to implement a multi-output setting for other algorithms are still under investigation. Other advantages include less requirement for spectra preprocessing to extract features for building a tabular dataset and, consequently, the possibility of testing spectra with reduced resolution and/or lower S/N ratio. It is challenging to work with low quality spectra in pipelines that use tabular datasets as those tend to rely on preprocessing tecnhiques that struggle themselves to work and extract information from those spectra.

Neural networks may however seem less appealing against other machine learning algorithms when the question of interpretability is considered relevant. Neural networks are implicitly known for their "black-box" nature, which, despite existing in other machine learning models, is certainly more prominent in the former. Once information starts being propagated across layers with a varying number of neurons, it becomes almost impossible to track where the information relative to each input observation is or how it is distributed. Therefore, if a neural network fails to predict the target variables or if there is a chance for improvement, it becomes difficult to know why or exactly what has to be changed to boost performance.

For example, Snider et al. used artificial neural networks (ANNs) for the simultaneous estimation of atmospheric parameters Teff, log g, and [Fe/H] in F- and G-type stars, within the following ranges: $4250 \leq$ Teff ≤ 6500 K, $1.0 \leq \log g \leq 5.0$ dex and $4.0 \leq$ [Fe/H] ≤ 0.3 [39]. Their approach yielded errors of about 135-150 K for Teff, 0.25-0.30 dex for log g and 0.15-0.20 dex for [Fe/H] on a sample of medium-resolution spectra, with S/N ratios as low as 13 of 279 stars (a subset of calibration stars used in previous high-resolution determinations of metallicity and external estimates of Teff and log g). Li and colleagues used instead deep neural networks (DNNs) to predict the same parameters, which differ from the simple ANNs in the number of hidden layers they construct, with the latter having more layers [35]. The methodology requires the spectra to have similar resolution and be aligned to the same wavelength scale (3850-4450 Å) and wavelength binning. Both observed and theoretical spectra, from the SDSS survey and from Kurucz's models, respectively, were analyzed, with reported mean absolute errors in the observed spectra of 0.1477, 0.0048 and 0.1129 dex for log g, log Teff (64.85 K for Teff)and [Fe/H], within ranges 1.015 - 4.998 dex, 4088 -9740 K, and –3.497, 0.268. Lou *et al.* (2018) trained a more complex type of neural network, a Convolutional neural network (CNN) model, to derive the same atmospheric parameters on 3129 observed spectra, in the wavelength region of 6095-6185 Å of supergiants to main sequence dwarfs of M to F-type stars from the PolarBase and Elodie databases, obtaining mean errors of 85 K within a range of 3182 K to 7500 K for Teff, 0.11 dex for log g between 0 and 5 dex, 0.07 dex within an interval of -0.7 to 0.7 dex for [Fe/H] [86].

Along with the abovementioned network-based models, Dafonte and co-authors introduced the novel architecture of generational neural networks (GANNs). The architecture of a GANN follows the same scheme as a traditional ANN, but the inputs and outputs are inverted, so the first step is to train the network with the set of atmospheric parameters (Teff, logg, [Fe/H] and [α/ Fe]), obtaining the stellar spectra for such inputs, where the goal is to minimize the residuals between the spectra in the grid and the estimated spectra [33].

Neural networks have also been employed to solve classification tasks. This includes the work from Navarro *et al.* (2012), who used several ANN architectures to predict stellar types. Their stellar database included a total of 1500 stars after all spectral types from O5 to M7 and all luminosity classes [36]. Other examples of classification tasks include STARMIND [87] and MKCLASS [88]. STARMIND mimics the human reasoning for stellar classification by using knowledge-based spectral features computed from template-spectra applied logic for classifying spectra with an overall accuracy of 82.7% at the main-class level using the MK classification

system. On the same line, MKCLASS classifies stellar spectra (in the region of 3800-5600 Å) based on a comparison between the input spectrum and MK standard spectra with errors of around 0.6 for the temperature sequence and 0.5 for the luminosity class on spectra with S/N ratio ≥ 100.

An attempt at comparing the results obtained with neural networks *versus* other machine learning models reveals that, in general, the accuracies or errors are within similar ranges for estimating quantitative stellar parameters. It seems however unwise to compare the vast majority of these studies, although some of the authors do that themselves. Regardless of using similar algorithms or not, the stellar sample is almost always different in range and size, and the instruments used to obtain the spectra, which may affect resolution and S/N ratio, and the spectra preprocessing techniques, including the wavelength region selected for the analysis, continuum subtraction, noise smoothing, among others, also differ. Therefore, there is no direct proof as to whether an algorithm or a strategy proposed in a study would perform better or worse on the stellar sample used in another study.

There is, however, a common characteristic shared by almost all the abovementioned models - the training and validation catalogs were built with spectra with a S/N of about 100 or higher. The same condition is required for the spectra to be classified, but typical spectra obtained in many observational surveys do not reach this quality. For example, in Navarro *et al.* (2012, 75% of the spectra has a S/N ratio < 40, and just around 5% of the sample has S/N ratio > 100 [36]. Needless to say, this reduces substantially the amount of data a machine learning algorithm can work with.

Methods to derive stellar parameters specifically in spectra with low S/N ratio are rather scarce. Indeed, these spectra are masked by the noise background and thus, it becomes difficult to extract any line characteristics from them. This may be, nevertheless, one of the most important challenges to tackle, mainly because these spectra are usually discarded and if a specialized method that could extract some information from them with relatively satisfactory results existed, the waste of data could be significantly reduced.

Minglei and colleagues proposed a 1D convolutional network that works with both synthetic and observed raw spectra where preprocessing is almost absent [34]. Their results showed an improvement when compared with previous methods based on linear [89] and non-linear [90] approaches using the same datasets. Their strategy divided both synthetic and observed total spectra in three groups of decreasing resolution. The worst results were obtained for the lowest resolution observed spectra and were within 160 K for the Teff, 0.3 dex for the log g and 0.2 dex for the [Fe/H], but those still surpassed the performance of the abovementioned methods.

Even though the discussed studies were all relatively successful in their respective tasks, it should be clear by now that all of them, either neural network-dependent or independent, share the requirement for a prior continuum normalization. Despite being one of the more important steps in spectra preprocessing, continuum estimation and normalization is still an open question from an astrophysics perspective, and new methods continue being investigated today. Last year,

Rózański *et al.* proposed SUPPNet, the first deep learning-based system for automatic continuum estimation and normalisation, a problem never addressed before from the artificial intelligence perspective. SUPPNet uses DNNs to train architectures that can satisfy a pseudo-continuum prediction task [91]. This data-driven normalisation method was tested with high-resolution spectra of stars with spectral types from O to G, and gives a root mean squared error of 0.0128 in the spectral range from 3900 - 7000 Å.

Ultimately, to be able to deal with the large-amounts of data resulting from the recent new releases of large-scale sky surveys in an efficient manner, it is very likely that future developments will follow the trend of SUPPNet and focus on the investigation of data-driven and artificial intelligence strategies to process and extract spectral features rather than predictive modelling, which should continue to evolve but is currently limited by the lack of consistent labelled data.

Chapter 4

Methodology

The main goal of this preliminary study was to estimate three stellar atmospheric parameters - the effective temperature Teff, the surface gravity log g and the metallicity [Fe/H] - using machine learning approaches on data extracted from stellar spectra.

In this section, the methodology adopted to address this problem is described. First, the star sample and the data sources to construct the working dataset are described. Then, a data exploratory analysis is performed to extract potentially relevant information from the data. That is followed by a feature selection procedure not only to attempt at reducing dimensionality, but also to find the most meaningful or informative features. Finally, several machine learning models are employed to investigate their potential to accurately predict the three target variables. Single- and multi-output strategies are investigated and analyzed in detail, including strategies with and without prior feature selection.

4.1 Data collection and dataset construction

To avoid external sources of variability that could interfere with the models' performances, it was important to find a star sample that was consistent in the ground-truth labels, that is, using the same method of acquisition, at least for the same target variable, obtained from the same instrument and where all the spectra had similar wavelength ranges and resolution. We are aware that more often than not, real data is not so clean and consistent and may come from different sources and that is exactly where a machine learning algorithm thrives or strives. On the other hand, to build a reliable methodology, it might be advantageous to start with consistent data as a first approach and move on to more complex or raw data afterwards, as it might be easier to uncouple the sources of possible predictive performance degeneracies in the models.

4.1.1 Stellar Sample

The stellar sample used in this work is practically the same as in Sousa *et al.* (2008) [92], for the sake of establishing a reliable comparison. The same sample was used by Tsantaki *et al.* (2013) [93], from which the ground-truth values for our target variables were obtained.

The dataset contains 449 stars (despite there are 451 in the dataset from Sousa *et al.* [92]) consisting mainly of dwarf F-G-K stars (revisit Sections 1 and {2.1 to recall stars' classification system) selected from a volume-limited sample of the CORALIE survey [94]. This is part of the High Accuracy Radial velocity Planet Searcher (HARPS) high-precision GTO program at the ESO La Silla 3.6 m telescope with the objective to detect low-mass extra-solar planets with high radial velocity accuracy [95]. Planet hosts from the southern hemisphere were also added to this sample [92]. These stars are slowly-rotating, non-evolved, and low-activity stars, with apparent magnitudes that range from 3.5 to 10.2 and have distances of less than 56 parsec. The spectra have a resolution of R $\sim$ 110 000 and 90% of the combined spectra have S/N higher than 200.

An important note refers to the origin of the target variables to be predicted. Unlike the usual 'real-world' applications of data science, in this case these variables could not be directly measured. It is trivial to imagine why the effective temperature of a star, as well as other parameters, cannot be directly measured or observed. For this reason, they have to be estimated by other methods and physical models. This may raise a legitimate concern as to whether the values adopted as ground-truth may cause the trained model to mimic the method used to obtain those values, rather than the real and 'natural' relationships between the predictors and the target variables. This can become a bigger concern if, for example, linear relationships have been assumed somewhere in those physical models, in which case a linear model would fit the data very well, not because there are 'natural' linear relationships in the data, but because the ground-truth values have been obtained from models that assume linear relationships between them and the predictors. This concern can however be put aside as the Teff, log g and [Fe/H] values used do not assume any linear relationships between spectral lines and any of those atmospheric parameters. Those parameters were obtained from spectroscopic analysis based on the iron excitation and ionization balance, assuming Kurucz model atmospheres in local thermal equilibrium (LTE) [93].

The main parameters that define these model atmospheres, namely effective temperature, surface gravity, microturbulent velocity and the overall metallicity, among others, are obtained as explained below, including the ground-truth values for this work [93]. This information is very-well summarized in [96].

- **Temperature:** The best value of Teff is derived by imposing the excitation equilibrium, which requires that there is no correlation between the abundance and the excitation potential χ of the neutral iron lines. In other words, the abundance of an element should be independent of the excitation potential of the individual lines because all the lines of a given element should give the same abundance for the same star. This means that the number of electrons populating each energy level is essentially a function of Teff according

to the Boltzmann equation. In practice, there is a (small) scatter around an average value, in which the x-axis represents increasing excitation potentials (in electronvolts, eV) and the y-axis a specific spectral line. If the value of Teff is incorrectly determined, it will affect the weak excitation potentials more than the strong potentials or vice-versa. For example, a value of Teff that is too large will lead to underpopulated lower energy levels, thus the predicted line profile for low χ transitions will be too shallow and a higher abundance will be needed to match the line profile. The opposite reasoning follows for when the calculated Teff is lower than the true one.

- **Surface gravity:** The best value of log g is generally derived with the ionization equilibrium method, requiring that for a given species, the same abundance (within the uncertainties) has been obtained from lines of two ionization states, typically neutral and single ionized lines. Because gravity is a direct measure of the pressure of the photosphere, variations of log g lead to variations in the ionized lines - which are very sensitive to electronic pressure - whereas the neutral lines are essentially insensitive to this parameter, as mentioned in Section 2.1.2. This method implicitly assumes that the energy levels of a given species are populated according to the Boltzmann and Saha equations, thus under the abovementioned LTE conditions.

  In this work, the trigonometric log g values were used instead of the spectrocopic ones. The former are based on *Gaia DR2* photometry and parallax [97, 98]. The trigonometric surface gravity is estimated using the following expression derived from the luminosity-radius-temperature relation for stars

$$\log \frac{g}{g_\odot} = \log \frac{M}{M_\odot} + 4 \log \frac{Teff}{Teff_\odot} - \log \frac{L}{L_\odot} \tag{4.1}$$

  where g is the stellar surface gravity, M is the mass, Teff is the effective temperature, and L is the luminosity. The stellar luminosities were directly obtained from precise data in Gaia eDR3 [98]. The precise calculation of surface gravity from Gaia parallax is an iterative process to converge simultaneously for the best estimates of the stellar mass and the trigonometric surface gravity. It requires knowing the stellar mass with respective calibrations from previous works, as well as other parameters derived from spectroscopy, such as Teff, [Fe/H] or even surface gravity itself. This will not be further detailed here, but more precise information and references can be found in [98].

  Log g values obtained with this method are generally more reliable compared to the spectroscopic one, because the abovementioned ionized lines which are sensitive and useful to infer the pressure are in fact quite scarce compared with the contrasting high number of insensitive neutral lines. They are also more difficult to work with.

- **Metallicity:** the best value is chosen according to the average iron content of a star, assuming [Fe/H] as a proxy for the overall metallicity, as indicated by the definition in 2.1.2.3. Despite [Fe/H] being often adopted as a good proxy of the metallicity because of

its large number of available lines, it does not necessarily indicate the overall metallicity of the star being studied. Elements such as C, N, and O, which are more abundant in stars than iron would be the best indicators of stellar metallicity, however those are often difficult to measure.

Both Boltzmann and Saha equations can be easily consulted elsewhere and will not be explained here, as they fall out of the main scope. Hopefully, the explanations above suffice to convince the reader that, although the target variables used as ground-truth in this work have been derived from model atmospheres and not directly measured, they do not assume any prior relationship between the target variables and the spectral lines (the predictors). Therefore, the results of the modelling strategies presented below should not depend on the assumptions of those physical models.

4.1.2 Data sources

The data for this star sample was obtained from three sources:

- raw spectra consisting in .fits files with all the properties of raw spectra.
- a collection of .ares files resulting from running a previously developed software [48] for each star .fits file (from the HARPS database) containing, among other data, the equivalent width of each spectral line (and an associated error). The dataset was composed of 455 stars' spectra. In short, the ARES software tries to emulate the 'manual' procedure of measuring the equivalent widths of absorption lines in spectra. The underlying approach is to instruct the computer on how to estimate the continuum position and to guess the number of lines (in case of blending effects) necessary for the best fit to the normalized spectrum around the line we want to measure. The procedure is to take a one-dimensional spectrum, a list of the spectral lines to measure and a file that contains the parameters necessary for the computation. Then the code selects a region around each line, where it finds the continuum position. After this, it identifies the number and the position of the lines needed to fit the local spectrum. The fitting parameters are used to calculate the equivalent width of each line. The result is produced in a file defined by the user. For more details regarding theory and implementation, refer to Sousa *et al.* (2007) [48] and https://github.com/sousasag/ARES_v2.c
- a line list file containing wavelengths and chemical elements' identifications of lines commonly found in stars' spectra analysis. The available file had a total of 520 different lines, which were used as predictors or independent variables of the working dataset. Note that this does not mean this is the maximum possible number of spectral lines that could appear, but rather lines known to be present in spectra according to prior domain knowledge.
- a table containing the target or dependent variables corresponding to stellar atmospheric parameters - the effective temperature Teff, in Kelvin K; the surface gravity log g, in *dex*;

and the metallicity [Fe/H], in *dex* - as well as associated errors of those measurements. This table is available from Tsantaki *et al.* (2013) [93].

There were two possibilities to handle these data, which are explained below.

- use raw spectra and perform segmentation to find the position of spectral lines.

  Whilst deep neural networks are undeniably the hottest topic in Artificial Intelligence, in some cases it is justifiable not to use them as a first approach. DNNs are very powerful at detecting particular features and/or patterns in the data that are often imperceptible to humans. This means they are exceptionally useful to object detection or segmentation. However, when the objective is the predictive modelling of a single or multiple target variable(s), one must be aware of some important details, such as the lack of interpretability. Indeed, neural network-based models are often referred as 'black-boxes' because as the number of layers and neurons on each layer increases, it becomes difficult to track what is happening at the layer or neuron level. It is known that some neurons will keep being activated whereas others will be shut-down among epochs depending on the result of the backpropagation algorithm of the previous step. Nonetheless, it is difficult to translate this into which features are considered relevant or being discarded.

- use a software to extract equivalent widths of spectral lines directly, build a tabular dataset and employ gold-standard machine learning algorithms. The objects in this star sample were observed and obtained with the same instrumental setup, which is a very important detail. Even though the ideal ultimate scenario would be to train a model that could work with any spectra coming from any instrument, in a preliminary study it is important to keep the sources of variability to a minimum, especially when a first model is being tuned and optimized.

  This dataset was indeed too small to justify using a neural network. Moreover, the available data sources above, especially the existence of a line list with the wavelengths of possible lines, allowed the construction of a tabular dataset, following the procedure illustrated in Figure 4.1.

Essentially, the strategy in Figure 4.1 consisted in running the ARES software [48] to extract the equivalent widths from raw spectra. This generated a file for each star, including, among other parameters, equivalent widths for each spectral line the software could find in the selected wavelength range, as well as associated uncertainties of those measurements. This was joined with the line list file, which contained a list of 520 possible lines and their respective wavelength. After some simple transformations, this resulted in a tabular dataset in which the columns are possible spectral lines - the predictors - and the 455 rows are the number of observations (stars). Therefore, each entry corresponds to the value of the equivalent width for a specific spectral line in a particular star's spectrum. Finally, this has been joined with a last table containing the target variables Teff, log g and [Fe/H] for each star on the first table to construct the final dataset. Note that the initial collection contained 455 spectra, however, when merging that data

with the targets table, the final number of stars was reduced to 449. The final dataset of 449 rows $\times$ 520 columns contained some NAs (no values) for some columns. This was expected because some spectral lines do not appear in every star's spectrum. These NAs were replaced with zeros. This strategy was the most logical given that the predictors represent values of equivalent widths for each line and a value of zero means the absence of that line.

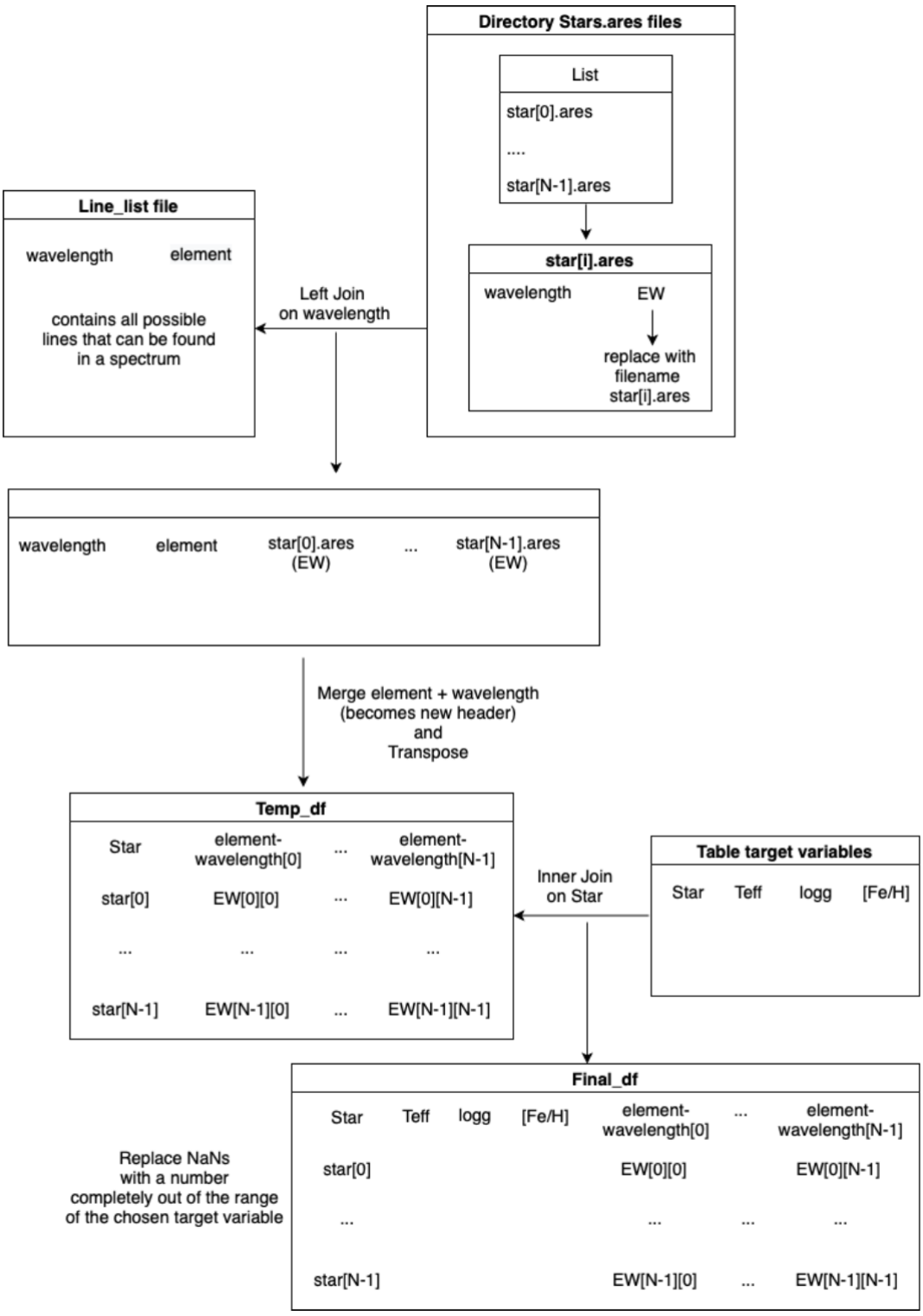


Figure 4.1: Workflow to construct the final working dataset from the available data. In this dataset, the number of rows corresponds to the number of spectra and the number of columns to the number of spectral lines to be assessed. Each occurrence is the value of the equivalent width (EW) for a specific spectral line in a star spectrum.

The variables, their type and domain/range in the dataset are shown in Table

Table 4.1: Pre-processed data variables in the working dataset.

Variables	Type	**Domain**
Teff	*int*	$4400 \leq ... \leq 6431$
logg	*float*	$3.6 \leq ... \leq 4.82$
Fe_H	*float*	$-0.83 \leq ... \leq 0.36$
FeI-5247.06 (EW)	*float*	$26.89 \leq ... \leq 125.59$
...	*float*	...
EuII-6645.13 (EW)	*float*	$2.11 \leq ... \leq 18.9$

4.2 Data exploratory analysis

The first step in any machine learning setting is to get familiarized with the data. Here, the methodologies used for descriptive statistics and assessment of correlations between variables are described.

Descriptive statistics The distribution of the target variables was accessed via histograms. Relationships between each of the target variables was also analyzed by plotting them one against each other. The range and distribution of each of the predictors - the spectral lines - were analyzed with boxplots. These show the range of the variables and whether there is a lot of variability within that range, the presence of outliers, among other details. It is also useful to get a sense of the variables that might have higher or lower variance.

Estimation of correlations between predictors and target variables Two types of correlations were investigated between predictors, between predictors and targets and between the targets: the Pearson's and the Spearman's correlation coefficients.

The Pearson's correlation coefficient is a correlation coefficient that measures linear correlation between variables. It consists in the ratio between the covariance of two variables and the product of their standard deviations, thus, it is roughly a normalized measurement of the covariance, whose result has a value between 1 and 1, with 1 meaning a perfect positive correlation and -1 a perfect negative correlation. It should be reinforced that this coefficient can only measure the strength of a linear correlation of variables, ignoring many other types of relationships or correlations.

Mathematically, the pearson coefficient r can be written as

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}, \tag{4.2}$$

where x_i represents the i-th sample of variable x, y_i the i-th sample of variable y, and $\bar{x}$ and $\bar{y}$ the respective mean of values of those variables.

The Spearman's rank correlation coefficient is a nonparametric statistical dependence between the rankings of two variables. That means it assesses how well the relationship between two variables can be described using a monotonic function (not necessarily linear).

The Spearman correlation can also vary between +1 or 1, meaning that each of the variables is a perfect (positive or negative, respectively) monotone function of the other. A value of 0 means no correlation.

Mathematically, the Spearman's coefficient ρ can be written as

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1}), \tag{4.3}$$

where d_i is the difference between the ranks of each two coordinates for each point (x_i, y_i) and n is the total number of observations.

Matrices of correlations based on either coefficient have been constructed for the entire dataset, including the correlations predictor-predictor, predictor-target and target-target variables. Unfortunately, due to the prohibitive size of the matrix (520×520), it is difficult to visualize the labels of the spectral lines.

4.3 Feature Selection and dimensionality reduction

Feature selection approaches can be generally divided in two categories:

- **model-dependent**, in which the most important features are chosen after training or performing cross-validation using a specific model; or
- **model-independent**, which does not involve any modelling steps, focusing instead on relationships between the features. These could include correlations such as the Pearson's or Spearman's, or Information Theory-based approaches.

Several strategies were employed to perform feature selection and those are compared with the respective model they were extracted from containing all features, in case of model-dependent strategies, or compared with the best performing model using all *versus* selected features, when the approach was model-independent.

R-regression Computes Pearson's correlation coefficient r between each feature and the target variable. The Pearson's correlation coefficient has been explained in subsection 4.2 and its mathematical expression can be recalled from equation 4.2.

Mutual Information Mutual Information (hereafter, MI) has an information theory background [99] and shares close ties with Shannon entropy [100]. In contrast to the linear (or non-linear) correlation coefficients, it is also sensitive to dependencies which are not considered in the covariance. MI is zero if and only if the two random variables are strictly independent.

Considering a set of N bivariate measurements, $z_i = (x_i, y_i), i = 1, ..., N$, which are assumed to be iid (independent identically distributed) realizations of a random variable $Z = (X, Y)$ with density $\mu(x, y)$, and that the marginal densities of X and Y are defined as $\mu_x(x) = \int dy \mu(x, y)$ and $\mu_y(y) = \int dx \mu(x, y)$, then the MI can be defined as

$$MI(X, Y) = \int \int dx dy \mu(x, y) log \frac{\mu(x, y)}{\mu_x(x) - \mu_y(y)} \tag{4.4}$$

where the base of the logarithm determines the units in which information is measured. For example, base 2 leads to information measured in bits [100].

Informally, MI measures the information that two variables X and Y share, thus how much knowing one of those variables reduces the uncertainty about the other. For instance, if X and Y are independent, then knowing X does not give any information about Y (and vice-versa) and thus, their MI is zero. On the other hand,

Although the oldest and most widely-known method to calculate MI consists in partitioning the supports of X and Y into bins of finite size, new approaches have been proposed from k-nearest neighbor statistics [100–102]. The latter is the method implemented in `scikit-learn`. For each data point i, it finds the kth-closest neighbor to point i among those N_{x_i} data points whose value of the discrete variable equals x_i, using some distance metric. Defining d as the distance to this k-th neighbor, the number of neighbors m_i in the full data set that lie within distance d to point i (including the k-th neighbor itself) is counted. Based on N_{x_i} and m_i, we compute

$$I_i = \psi(N) - \psi(N_{x_i}) + \psi(k) - \psi(m_i), \tag{4.5}$$

where $\psi(.)$ is known as the digamma function. Finally, to estimate the MI from the whole dataset, I_i is averaged over all data points

$$I(X, Y) = \langle I_i \rangle = \psi(N) - \langle \psi(N_x) \rangle + \psi(k) - \langle \psi(m) \rangle \tag{4.6}$$

The algorithm only requires a fixed (small) integer k that defines the number of neighbors to consider.

Best subset A third approach was implemented from scratch, based on the paper from Hall (2001) [103] and adapted for continuous features and targets. This method is somewhat similar to the **Forward Sequential Feature Selection** approach from `scikit-learn` in its greedy nature, but on the contrary, it does not select the features from fitting a model. Essentially, it

tries to find the best subset of features that are strongly correlated with the target variable and, at the same time, less correlated with each other, to maximize the gain of information and avoid multicollinearity, consequently reducing high variance. It starts from an empty subset and the first feature to be added is the one that has the strongest correlation with the target variable (either Pearson's or Spearman's correlation coefficients can be employed, the mathematical expression can be adapted to handle one or the other). Next, it searches for the next best feature to add to that subset. This feature must maximize the correlation with the target variable, but minimize the correlation with the feature that is already in the subset, according to the following metric

$$Score = \frac{kr_{feature-target}}{\sqrt{k + k(k-1)r_{feature-feature}}} \tag{4.7}$$

This is implemented using a **priority queue**. Initially, the best subset, which is composed of only one feature with the highest correlation with the target variable, is pushed onto the queue. The rest of the features is also pushed according with decreasing priorities. Then, until the queue is empty or a certain number of 'backtracks' is reached (stopping criterion), we get the element with the highest score. It iterates through all features and looks for one that increases the score of the subset according to 4.7: if it does, that feature is added to the subset; otherwise, it backtracks to the previous subset without that feature. If a maximum number of backtracks is defined, the algorithm stops when it reaches it.

The results obtained with this method are not shown, as they did not produce scores as good as the **mutual information** method. It can be speculated that this is due to selecting very few features, as in the absence of further improvement in the $score_{feature}$, the algorithm stops adding new features after a certain number of attempts. Overall, the algorithm selected only a total of 9 features for Teff, 6 for log g and 10 for [Fe/H]. Indeed, Figure 5.13 shows that for the other feature selection methods tested, a number of features below 20 leads to a decrease in predictive power for any model. This method should be improved in the future, using a stopping criterion that allows addition of a higher number of features. The features selected with this method can be however consulted in Appendix C.

4.4 Predictive modelling

4.4.1 Data splitting

Prior to any modelling, the dataset, composed of a total of 449 observations, has been divided into a train and a test sets. We chose a 80%-20% train-test split, getting a train and test sets with 359 and 90 observations, respectively. Note that all the experiments involving optimization used only the train set. The test set was kept as a *hold-out* sample that was only used to access

the final performances.

Also worth keeping in mind, the majority of these experiments was performed using a single-output model for each of the three target variables, including the ones that used all or a pre-selected subset of features. Only after running all the experiments, a multi-output strategy has been leveraged using the best performing strategies. This is described in detail in the upcoming sections.

4.4.2 Data pre-processing

The information collected about the features in Section 5.1, particularly their distribution, revealed that some spectral lines have very different ranges. Despite the equivalent width varying between 0-400 in general, some lines had equivalent widths very close to zero, whereas others had equivalent widths . A careful observation of the boxplots in Figure 5.3 shows that some spectral lines have equivalent widths in the order of hundreds, despite having a high dispersion. The plot shows some points that are considered outliers, however, the majority of them does not seem to be completely out-of-range.

Because of the differences in equivalent width ranges between some spectral lines and also due to the high dispersion of some of them, two well-known scaling strategies - MinMax and Z-score - were used to pre-process the data in the modelling pipeline, before training any estimator. These scaling strategies are explained below.

MinMax Scaling The MinMax scaling transforms the features by scaling each feature to a given range. It scales and translates each feature individually such that it is in the given range on the training set, generally between zero and one.

$$x_{min-max} = \frac{x - x_{min}}{x_{max} - x_{min}} \times (max - min) + min, \tag{4.8}$$

where x_{min} - x_{max} are the minimum and the maximum values in the feature range.

The MinMax scaler does not reduce the effect of outliers, but it linearly scales them down into a fixed range. It is generally more useful when the different features on the dataset have different ranges.

Standard or Z-score Scaling The scaler standardizes the features by removing the mean and scaling to unit variance. It is generally used to ensure that the features' distributions have mean 0 and standard-deviation 1.

The standard score z of a sample x is given by

$$z = \frac{x - \mu}{\sigma}, \tag{4.9}$$

where μ is the mean of the training samples and σ their standard-deviation.

4.4.3 Single-output Regression

Even though the ultimate goal was to have one model that could estimate the three target variables simultaneously, single-output models for each target were initially performed, not only as a control, but also to verify whether the best performing model was the same for the three quantities. If not, it would be necessary to decide whether precision could be slightly sacrificed for a gain in efficiency by keeping the multi-output setting or, alternatively, build separate models for each one of them.

A range of gold-standard machine learning models for regression have been tested, including:

- linear regression and its regularized versions, Ridge, Lasso and ElasticNet.
- distance-based and kernel-based approaches: Nearest Neighbors and Support vector machines for regression (SVR), respectively.
- Tree-based algorithms, including model ensembles: Decision tree, Random Forest, Extra tress and Gradient Boosted Trees.

On the one hand, as previously mentioned on subsection 2.1.3, the old process to classify stars based on the presence/absence of certain spectral lines resembled a decision tree (recall Figure 2.7). Therefore, it was expected that, in theory, this kind of algorithm could work well in this task. Since simple decision trees are prone to overfitting ([104]), tree-based model ensembles were expected to improve the predictions in comparison. In addition, model ensembles such as Random Forests or Gradient Boosting tend to be more powerful than any other machine learning models. On the other hand, due to the presence of strong multicollinearity in the data, as addressed on 4.2 and shown on 5.1, which renders trained models to high variance, it was also possible that methods that use regularization, either linear or non-linear, could show a good performance. For further mathematical details on these algorithms and concepts, recall Section 2.2 from Chapter 2.

The first step was to search for the best model among the indicated alternatives, and for each model, the best values for each hyperparameter. This was performed on single-output models, one for each of the target variables.

4.4.3.1 Hyperparameter tuning

Depending on the model, different hyperparameters were optimized using the GridSearchCV algorithm. The GridsearchCV pipeline has been adjusted so different models and their respective hyperparameters could be tested on a single GridSearch.

The GridSearch algorithm uses a K-fold cross-validation procedure internally, which evaluates how good a trained estimator performs and, most importantly, whether overfitting is occurring. The standard methodology to evaluate an estimator is simply splitting the data in a train and a test sets, and if overfitting has occurred, then the performance will be significantly lower on the test than on the train set. However, even if the test instances are chosen randomly from the data, every once in a while we may get lucky, if most of the test instances are similar to training instances – or perhaps unlucky, if the test instances are very noisy or tend to deviate from the train set. K-fold cross-validation performs this train–test split several times (in this case k). The procedure works as follows: the data is randomly divided in k parts of equal size, from which $k-1$ parts are used for training and 1 one part for testing. This is done $k-1$ times, until each fold has been used once for testing [105]. At the end, the average test set performance is computed (and usually also its standard deviation), which is useful to determine whether small differences in average performance of different learning algorithms are meaningful or not. This process is illustrated in Figure 4.2.

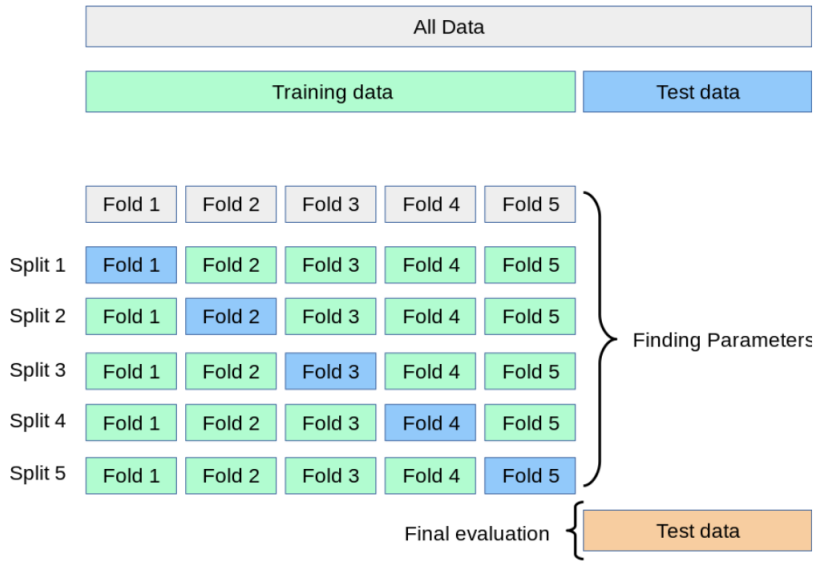


Figure 4.2: K-Fold cross-validation procedure. The full dataset is divided into a train and a test set. The test set is completely kept as a *hold-out*. The train set is then divided into several different combinations of train and validation sets to evaluate performance. The final performance, after hyperparameter tuning, is obtained on the *hold-out* test set. [8]

Note that, in the Figure 4.2, the test data shown in orange is the $hold-out$ test set, which is never seen during any GridSearch or Cross-validation experiment.

[8] https://scikit-learn.org/stable/modules/cross_validation

Relevant tuned hyperparameters

Here, the hyperparameters tuned for each of the algorithms employed in the GridSearchCV are described in terms of their meaning. According to the `scikit-learn` documentation, many other parameters exist that can be optimized. Others have already a default value that is generally the best independently of the data. We decided to optimize only the ones we considered as possibly having a stronger effect on the performance and depending on the data. As GridSearch is a brute-force algorithm that searches for all combinations of parameters, it should be used with care regarding the number of combinations when resources are limited.

Ridge Regression

- **alpha** - the strength of the regularization or the L2 penalty.
- **solver** - the mathematical optimization algorithm to solve the algebraic problem. Some are faster, and others are slower but more accurate.

Lasso regression

- **alpha** - the strength of the regularization or the L1 penalty.
- **selection** - The default is to loop over features sequentially by default ('cyclic'). There is the possibility to set this to 'random', in which a random coefficient is updated every iteration rather than looping over features sequentially by default. 'Random' selection often leads to significantly faster convergence.

Elastic Net

- **alpha** - Constant that multiplies the penalty terms (it uses both L1 and L2 penalties).
- **L1 ratio** - mixing parameter between l1 and l2 penalties. For l1 ratio = 0, the penalty is an L2 penalty (Lasso). For L1 ratio = 1, it is an L1 penalty (Ridge). For a number between 0 and 1, the penalty is a ratio between L1 and L2.
- **selection** - similar to Lasso regression.

Support Vector Regression

- **Kernel** - Specifies the kernel type to be used in the algorithm. If none is given. It can be linear, polynomial with different degrees, sigmoid or a radial basis function.

- **Regularization parameter (C)** - The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.
- **Epsilon** - It specifies the epsilon-tube within which no penalty is associated in the training loss function with points predicted within a distance epsilon from the actual value. Must be non-negative.

Random Forests

- **Number of estimators** - this is the number of trees that will be running in parallel.
- **Maximum depth** - maximum depth that any tree can reach. This resembles a stopping criteria.
- **Maximum features** - the number of features to consider when looking for the best split. There so the possibility to use all features, or, for example, $\sqrt{allfeatures}$ or $\log_2(allfeatures)$.
- **Minimum samples split** - The minimum number of samples required to split an internal node.

Gradient Boosting

- **Number of estimators** - the number of boosting stages to perform. Gradient boosting is fairly robust to over-fitting so a large number usually results in better performance.
- **Maximum depth** - maximum depth that any tree can reach. This resembles a stopping criteria.
- **Maximum features** - the number of features to consider when looking for the best split. There so the possibility to use all features, or, for example, $\sqrt{allfeatures}$ or $\log_2(allfeatures)$.
- **Minimum samples split** - The minimum number of samples required to split an internal node.
- **Learning rate** - shrinks the contribution of each tree by this quantity. There is a trade-off between this parameter and the number of estimators.
- **Subsample** - the fraction of samples to be used for fitting the individual base learners. If smaller than 1, it results in Stochastic Gradient Boosting. It has a relatioship with the number of estimators, and generally choosing subsample < 1 leads to a reduction of variance and an increase in bias.

4.4.4 Multi-output Regression

As mentioned in the subsection above 4.4.3, the main goal would be to have a multi-output model that could predict the three target variables simultaneously. When should a multi-output approach be preferred over separate single-output ones? Having a single pipeline to train is more efficient that running m different models separately, which requires specifying parameters for each of them, as well as storing m models in memory. Furthermore, a single-output model is optimized for each target separately instead of all targets together, which means they do not consider possible relationships between the different targets. This may provide an advantage, but can also become a pitfall, it really depends whether targets are related or not. Also, for example, if the inputs predict one output very badly, this output cannot provide support to the other outputs. And if the inputs predict one output very well, this output does not require support from the other outputs, which constitutes an additional complication.

Not all regression algorithms support multi-output regression. Linear regression and variants, including Ridge, Lasso and Elastic Net, as well as distance-based algorithms like nearest neigbours or finally, Decision trees and Random Forests are said to be inherently multi-output, since the same regressor object is used, but instead of receiving a vector of $Y = (n\_samples,)$, they are able to receive a $Y(n\_samples, n\_outputs)$ and are all included in the gold-standard machine learning `scikit-learn` and were already discussed in the subsection above about single-output models.

There is also a special workaround that can be used to wrap and use algorithms that do not natively support multiple outputs, such as SVR and Gradient Boosting Machines. This workaround consists in dividing the multi-output regression problem into multiple sub-problems and merging the results at the end into a vector of outputs, thus it works as a wrapper.

The schematic in Figure 4.3 illustrates very well the three kinds of settings that can be leveraged to predict multiple target variables.

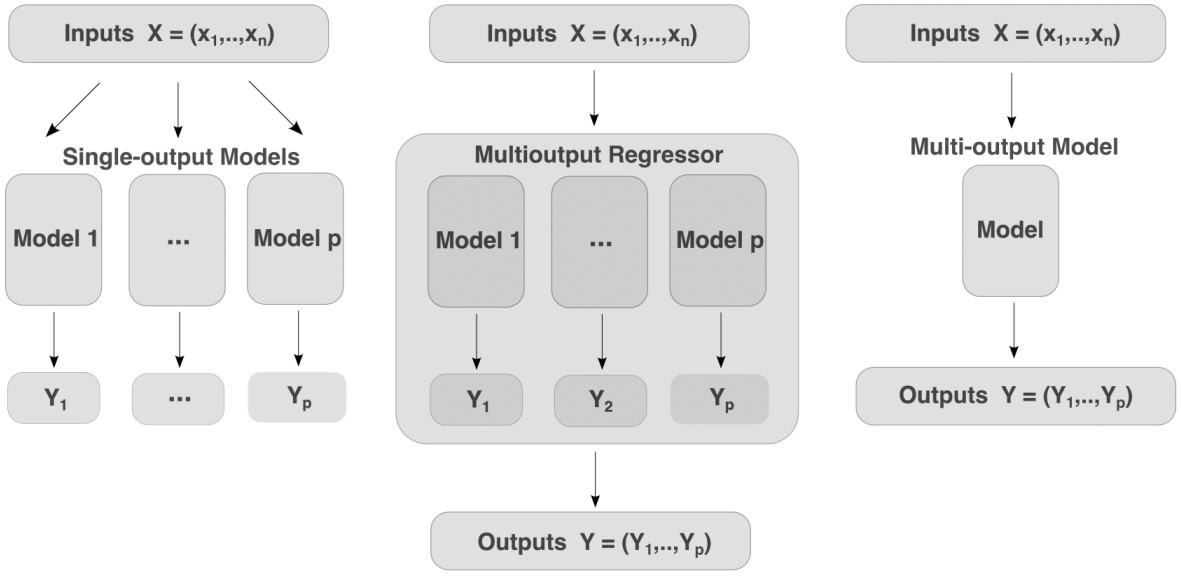


Figure 4.3: Schematic illustrating the different settings to handle the three targets variables. (Left) Single-output: a model is built for each target separately. (Middle) Multioutput Regressor: a `scikit-learn` built-in multi-output configuration that treats all the targets simultaneously, but builds a separate model for each one internally. (Right) Multi-output: builds a model that inherently supports simultaneous handling of several target variables.

The single-output approach is shown on the Left and it consists in building a model for each target variable separately, as seen in 4.4.3. The algorithms that natively support a multi-output setting belong to the scheme on the Right. A single model is built and it predicts a matrix of outputs instead of a vector.

The schematic in the middle illustrates the wrapper multi-output regressor. The regression problem is divided into a separate problem for each target variable to be predicted internally. This approach is supported by the **MultiOutputRegressor** class from , which creates one instance of the chosen model for each target variable. For example, if a multi-output regression problem required the prediction of three values y_1, y_2 and y_3 given an input X, then this could be partitioned into three single-output regression problems. This of course assumes that the outputs are independent of each other, which might not be a correct assumption. For example, it can be that the outputs for a problem may, in fact, be mostly independent, if not completely independent, in which case this approach would be preferred even for algorithms that natively support multi-output predictions. The approach on the right shows the true multi-output setting, in which the algorithms natively support a 2-dimensional array of response variables.

Here, we employed the wrapper multi-output approach using the **MultioutputRegressor** for `scikit-learn` for the same models in section 4.4.3. We conducted another GridSearchCV to optimize the hyperparameters *de novo*, since the same model and hyperparameters must be used for the three target variables in this setting. Despite the regressor being forked internally into three, one for each target, the score in GridSearch is calculated across all target variables.

Therefore, the target variables were also MinMax normalized (recall definition 4.8), so none of them dominated the score.

Moreover, as in the single-output case, experiments with and without feature selection were also compared. To perform predictive modelling with feature selection in a multi-output setting, the **MultiOutputRegressor** class from `scikit-learn` has been slightly altered to receive as input a different set of features for each target - the **MultioutputTargetSelectRegressor** class. The code can be found in https://github.com/joanaleite92/stars_ML. In short, the features are passed to the regressor in a dictionary in which the number of keys is the number of targets and the values are arrays spanning the indexes of the columns corresponding to the features selected for each target. Then, internally, inside the wrapper, each array in the dictionary is mapped to the estimator corresponding to each target.

The native multi-output setting was not tested for two reasons: 1) our target variables did not show strong correlations with each other (see section 5.1); 2) only the linear regression and variants, kNN and decision trees/Random Forests support this approach. The result for the linear regressions would remain the same as in the wrapper method or the single-outputs, because the coefficients for the kth outcome are simply the least squares estimates in the regression of y_k on $x_0, x_1, ..., x_p$, so the outputs do not affect each others' estimates, as we have seen in Section 2.2.3.1 from Chapter 2. The kNN and decision trees have been shown to not perform very well and we would be left with only Random Forests, without any term of comparison. Given the lack of correlation between the targets, there would be no advantage in testing this.

4.4.5 Predictive modelling with equivalent width uncertainties

Intrinsic uncertainties exist both in the predictor and in the target variables. The derivations of Teff, log g and [Fe/H] reference values of the HARPS dataset have associated average uncertainties in the order of 34 K, 0.017 dex and 0.021 dex, respectively, because of their initial photometric derivations [93].

Additionally, the estimation of equivalent widths of spectral lines using the ARES software also has an error associated. Since these parameters are used as the training values for the machine learning process, these uncertainties have been injected by perturbing equivalent widths values accordingly in order to see how the final results of the predictions vary.

With this in mind, 100 Gaussian distributions were applied to each row value of the HARPS dataset, including both the train and the test sets, increasing the dispersion of distribution of the reference parameters. Essentially, after this procedure, each star has 100 different sets of equivalent widths inside a range whose margins match the error mentioned above. After running the model, the data is again grouped by star and, for each of the initial observations, an average of the predicted values for each target variable can be predicted together with a standard-deviation.

However, it is intuitive to see that this trasnformation adds different training values to

the machine learning algorithm. In machine learning, there is a rather similar process called data augmentation. Even though data augmentation has its own well-established strategies, this process of injecting uncertainties augments the data as a consequence. For that reason, four different tests were run to evaluate the real effects of this procedure from a data science perspective. This is illustrated in Figure 4.4. The Model A1 is the typical machine learning approach, where the original dataset is used. The Model A2 uses the original training set and injects uncertainties only on the test set. This could be valid, since the interest was to have an uncertainty in the final test data and not necessarily during training. However, if these uncertainties are not also learned during training, it can also be anticipated that this approach will not provide good results. The Model B1 and B2 would be the real analogy to data augmentation, particularly Model B1, as data augmentation is generally only performed on the training data and not on the test set. Finally, Model B2 has uncertainties injected in the test set as well. This may combine data augmentation in the training set and still allow the possibility to have a mean predicted value for each star, with an associated uncertainty.

The uncertainties obtained via the machine learning method were then compared with the uncertainties of each target variables obtained using astrophysics' methods.

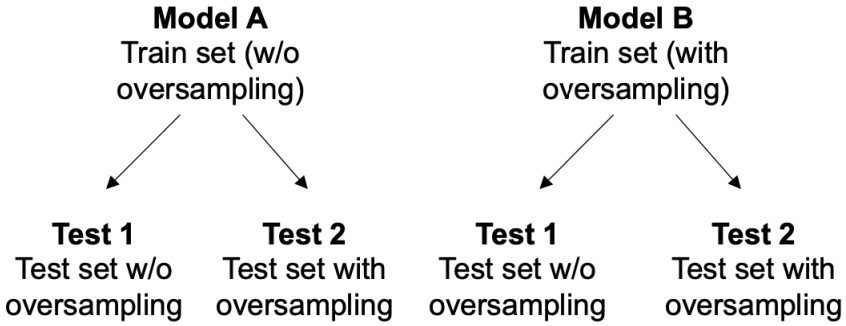


Figure 4.4: Strategies to handle the insertion of equivalent width uncertainties associated with ARES procedure in the machine learning models. Data augmentation may be a collateral effect of this approach.

4.4.6 Evaluation metrics

To evaluate the different models on each experiment, evaluation metrics adequate to regression problems have been used. These are described below.

Mean absolute Error The mean absolute error (MAE) simply finds the absolute value of the difference between the observed values and the predicted values and is written as

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{4.10}$$

where N is the number of data points, y_i are the observed/true values and $\hat{y}$ the predicted values for the target variable, respectively.

Mean Squared Error The mean squared error (MSE) represents the difference between the original and predicted values extracted by squaring the average difference over the data set and can be written as

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2 \tag{4.11}$$

Rooted Mean Squared Error The Root Mean Square Error (RMSE) measures the standard deviation of the predicted errors from the model. It essentially measures how spread out the residuals are in the dataset - that means it is actually the standard-deviation of the residuals.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2} \tag{4.12}$$

Coefficient of Determination R^2 The coefficient of determination R^2 represents how well the values fit compared to the original values. The value from 0 to 1 interpreted as percentages. The higher the value is, the better the model is.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}, \tag{4.13}$$

where y_i and $\hat{y}_i$ are the same as equations above and $\bar{y}$ is the mean value of y.

All experiments used MAE as a metric, including the GridSearch algorithm or simple cross-validations, as well as the final evaluation on the $hold-out$. In some cases, RMSE and R^2 were also calculated.

Chapter 5

Results and Discussion

Here, the results obtained using the methodology from Chapter 4 are described and discussed. The first step consisted in a data exploratory analysis to find meaningful insights about the data distribution and possible correlations among predictors, as well as between predictors and target variables. Due to the high dimensionality of the dataset, feature selection is conducted using different methods, not only to identify possible highly informative features both from the machine learning and physics perspective, but also to reduce the feature space and possibly develop simpler and more explainable models. The final stage is the predictive modelling, in which several machine learning models are fine-tuned and evaluated with respect to their ability to predict the target variables based on features extracted from the spectra. Strategies involving all features and, alternatively, with feature selection, are investigated and compared, as well as single- and multi-output predictive modelling approaches.

5.1 Data exploratory analysis

Data exploratory analysis generally involves descriptive statistics to evaluate the distribution and range of both predictor and target variables, outlier detection and identification of possible domain imbalances, as well as assessment and visualization of potential correlations (linear or, more generally, monotonic) between each predictor and target, between predictors and/or between targets (if more than one exists).

Distribution and relationships between target variables

Figure 5.1 shows the distribution of each target variable - Teff, log g and [Fe/H] in the HARPS dataset.

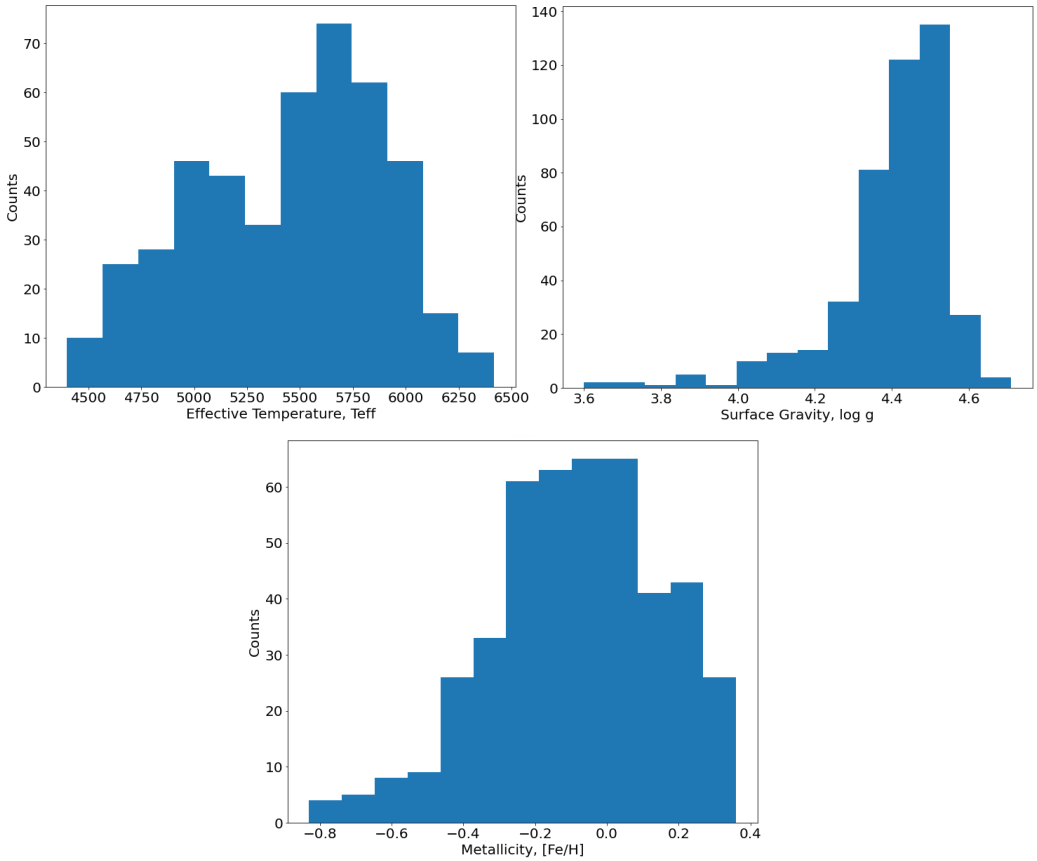


Figure 5.1: Histograms showing the distribution of each of the targets variables: (Top Left) Effective Temperature, Teff; (Top Right) Surface Gravity, log$g$; (Bottom) Metallicity, [Fe/H].

It becomes immediately apparent that the effective temperature Teff distribution is quite uniform over its entire range, whilst the surface gravity log g is not normally distributed and displays instead a rather strong right-skewness, with larger values being significantly more represented than the smaller ones. A slight right-skewness is also observed for [Fe/H], albeit less prominent than that of log g. In the case of log g, the vast majority of points falls within the range $\approx$ 4.3 - 4.5 dex, whereas the dispersion in the case of [Fe/H] is broader in its respective range.

Even though oversampling techniques could be employed to increase the representation of the minority range, such as SMOTER ([106]), an adaptation of the well-known SMOTE for regression, this procedure must be experimented with some caution, given that there are three target variables, which do not all suffer from domain imbalance and, for that reason, it has been decided not to use that approach in a first stage.

Given the differences in distribution of each target variable - some were clearly skewed whereas others followed a roughly normal distribution - these were plotted against each other in Figure 5.2 to infer possible relationships between them.

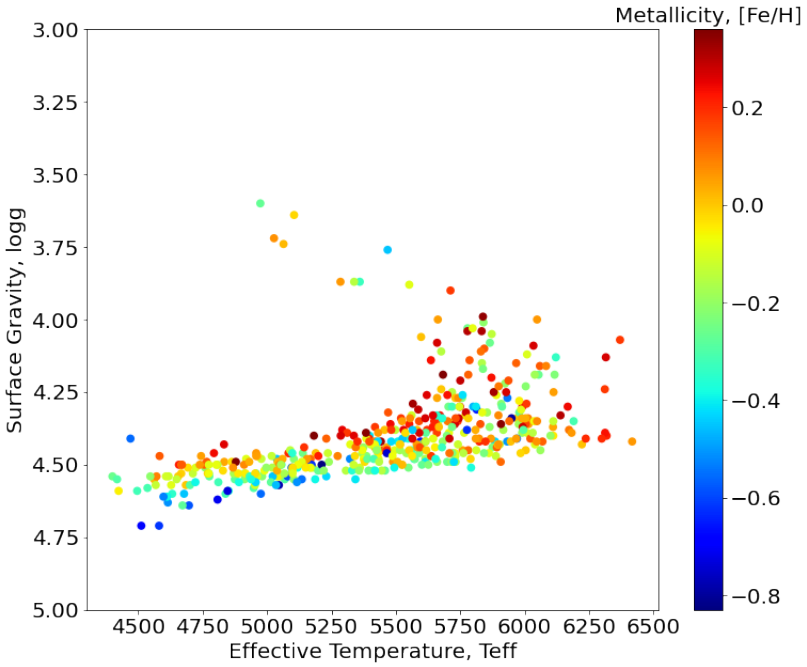


Figure 5.2: Inspection of target variables relationships and possible dependencies. The x-axis represents the Effective Temperature Teff and the y-axis the Surface Gravity, log$g$. The Metallicity [Fe/H] is represented by the colours of the points according to the colorbar on the side.

For higher values of log g ($\approx$ 4.3 - 4.7 dex), its relationship with Teff could almost be fitted by a constant line (see Figure 5.2). This suggested that stars with very different effective temperatures, for example Teff = 4500 K or Teff = 6400 K, could have similar surface gravities, although a very modest tendency for log g to slightly decrease with Teff is observed. On the other hand, there is a positive linear dependence on Teff for lower log g values, although that is rather farfetched given the misrepresentation in that range of log g. This is in agreement with the absence of a linear correlation between them (Pearson's correlation coefficient is -0.50), and presence of a negative monotonic one (Spearman's coefficient is -0.74). It is hard to see a trend between log g and the [Fe/H], with Pearson's and Spearman's coefficients of -0.29 and -0.41. Moreover, there is no clear relationship between Teff and [Fe/H] (the Pearson's and Spearman's coefficients are 0.26 and 0.24, respectively). This may seem contradictory given that cooler stars tend to have higher metallicity, as explained on Chapter 2, Subsection 2.1.2. It should be noted, nonetheless, that the range of temperatures in this dataset is rather small and only covers the F-G-K classes in the MK classification system (recall ranges from Table 4.1), according to which the maximum temperatures may reach more than 50 000 K for O-type stars. Thus, it is possible that a similar plot would yield different information if the whole range of stars' classes had been covered.

Altogether, this initial inspection on the three target variables suggests that for the range of effective temperatures covered in this dataset, there are no strong linear correlations between that and the other two target variables.

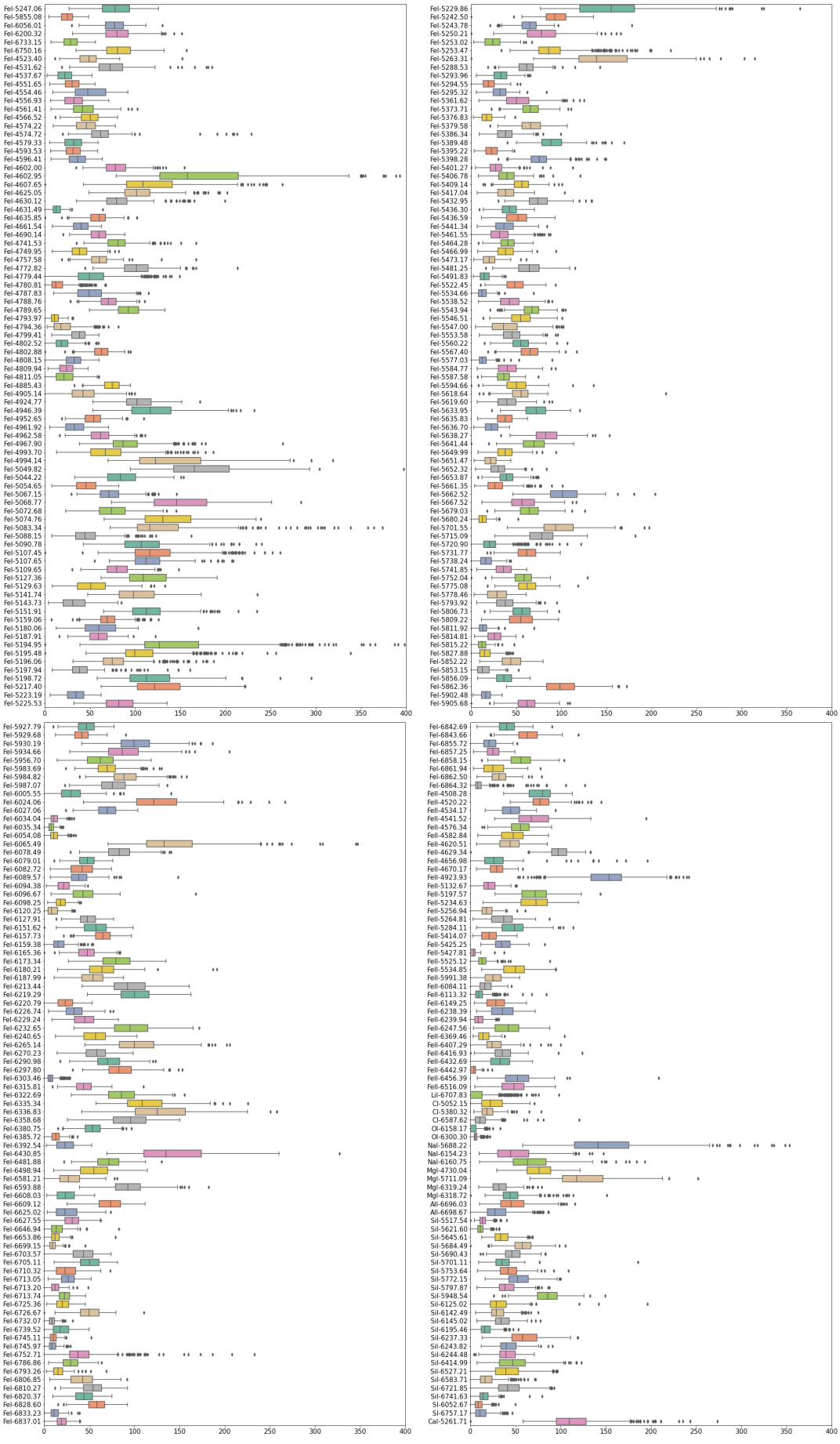
Distribution and correlations between predictor variables

Perhaps more important than the inspection of the target variables is the analysis of the predictors. Indeed, the former is only crucial to investigate whether different target variables being predicted simultaneously are allowed to interact with each other, as in model chains or ensembles. In that case, predictions may be significantly improved or not, depending on the relationships between targets.

On the other hand, the distribution of predictors and correlations between them are known to easily boost or compromise a model's performance, as the majority of machine learning models suffer from the *curse of dimensionality*. In addition, the ratio between the number of observations and the number of predictors, as well as the predictors' variance may also impact a model predictive power. For example, it is intuitive to reason that features that do not vary much, (low variance) may not be very good predictors. Moreover, in the presence of a high dimensional space, which is known as the $p >> N$ problem, where N is the number of observations and p is the number of predictors [3], there is a strong possibility that models struggle to generalize well and high variance and overfitting become a major concern, which means the models' predictive power may vary significantly with the data. As a result, simple and highly regularized models often become the methods of choice.

In this dataset, there are 520 spectral lines representing the predictors p, which is *per se* a high-dimensional feature space. In contrast, the dataset is composed of only 449 stars, thus the number of observations N is relatively small. These two characteristics combined may be, in some cases, sufficient to degrade the quality of a machine learning model, as mentioned above.

The distributions of all predictors are shown in Figure 5.3. This is not easy to inspect due to the large amount of features. However, there are some general characteristics that pop up and can provide interesting insights for data preprocessing or interpretation of models' results.


FeI-5247.06
FeI-5855.08
FeI-6056.01
FeI-6200.32
FeI-6733.15
FeI-6750.16
FeI-4523.40
FeI-4531.62
FeI-4537.67
FeI-4551.65
FeI-4554.46
FeI-4556.93
FeI-4561.41
FeI-4566.52
FeI-4574.22
FeI-4574.72
FeI-4579.33
FeI-4593.53
FeI-4596.41
FeI-4602.00
FeI-4602.95
FeI-4607.65
FeI-4625.05
FeI-4630.12
FeI-4631.49
FeI-4635.85
FeI-4661.54
FeI-4690.14
FeI-4741.53
FeI-4749.95
FeI-4757.58
FeI-4772.82
FeI-4779.44
FeI-4780.81
FeI-4787.83
FeI-4788.76
FeI-4789.65
FeI-4793.97
FeI-4794.36
FeI-4799.41
FeI-4802.52
FeI-4802.88
FeI-4808.15
FeI-4809.94
FeI-4811.05
FeI-4885.43
FeI-4905.14
FeI-4924.77
FeI-4946.39
FeI-4952.65
FeI-4961.92
FeI-4962.58
FeI-4967.90
FeI-4993.70
FeI-4994.14
FeI-5049.82
FeI-5044.22
FeI-5054.65
FeI-5067.15
FeI-5068.77
FeI-5072.68
FeI-5074.76
FeI-5083.34
FeI-5088.15
FeI-5090.78
FeI-5107.45
FeI-5107.65
FeI-5109.65
FeI-5127.36
FeI-5129.63
FeI-5141.74
FeI-5143.73
FeI-5151.91
FeI-5159.06
FeI-5180.06
FeI-5187.91
FeI-5194.95
FeI-5195.48
FeI-5196.06
FeI-5197.94
FeI-5198.72
FeI-5217.40
FeI-5223.19
FeI-5225.53
0
50
100
150
200
250
300
350
400
FeI-5229.86
FeI-5242.50
FeI-5243.78
FeI-5250.21
FeI-5253.02
FeI-5253.47
FeI-5263.31
FeI-5288.53
FeI-5293.96
FeI-5294.55
FeI-5295.32
FeI-5361.62
FeI-5373.71
FeI-5376.83
FeI-5379.58
FeI-5386.34
FeI-5389.48
FeI-5395.22
FeI-5398.28
FeI-5401.27
FeI-5406.78
FeI-5409.14
FeI-5417.04
FeI-5432.95
FeI-5436.30
FeI-5436.59
FeI-5441.34
FeI-5461.55
FeI-5464.28
FeI-5466.99
FeI-5473.17
FeI-5481.25
FeI-5491.83
FeI-5522.45
FeI-5534.66
FeI-5538.52
FeI-5543.94
FeI-5546.51
FeI-5547.00
FeI-5553.58
FeI-5560.22
FeI-5567.40
FeI-5577.03
FeI-5584.77
FeI-5587.58
FeI-5594.66
FeI-5618.64
FeI-5619.60
FeI-5633.95
FeI-5635.83
FeI-5636.70
FeI-5638.27
FeI-5641.44
FeI-5649.99
FeI-5651.47
FeI-5652.32
FeI-5653.87
FeI-5661.35
FeI-5662.52
FeI-5667.52
FeI-5679.03
FeI-5680.24
FeI-5701.55
FeI-5715.09
FeI-5720.90
FeI-5731.77
FeI-5738.24
FeI-5741.85
FeI-5752.04
FeI-5775.08
FeI-5778.46
FeI-5793.92
FeI-5806.73
FeI-5809.22
FeI-5811.92
FeI-5814.81
FeI-5815.22
FeI-5827.88
FeI-5852.22
FeI-5853.15
FeI-5856.09
FeI-5862.36
FeI-5902.48
FeI-5905.68
0
50
100
150
200
250
300
350
400
FeI-5927.79
FeI-5929.68
FeI-5930.19
FeI-5934.66
FeI-5956.70
FeI-5983.69
FeI-5984.82
FeI-5987.07
FeI-6005.55
FeI-6024.06
FeI-6027.06
FeI-6034.04
FeI-6035.34
FeI-6054.08
FeI-6065.49
FeI-6078.49
FeI-6079.01
FeI-6082.72
FeI-6089.57
FeI-6094.38
FeI-6096.67
FeI-6098.25
FeI-6120.25
FeI-6127.91
FeI-6151.62
FeI-6157.73
FeI-6159.38
FeI-6165.36
FeI-6173.34
FeI-6180.21
FeI-6187.99
FeI-6213.44
FeI-6219.29
FeI-6220.79
FeI-6226.74
FeI-6229.24
FeI-6232.65
FeI-6240.65
FeI-6265.14
FeI-6270.23
FeI-6290.98
FeI-6297.80
FeI-6303.46
FeI-6315.81
FeI-6322.69
FeI-6335.34
FeI-6336.83
FeI-6358.68
FeI-6380.75
FeI-6385.72
FeI-6392.54
FeI-6430.85
FeI-6481.88
FeI-6498.94
FeI-6581.21
FeI-6593.88
FeI-6608.03
FeI-6609.12
FeI-6625.02
FeI-6627.55
FeI-6646.94
FeI-6653.86
FeI-6699.15
FeI-6703.57
FeI-6705.11
FeI-6710.32
FeI-6713.05
FeI-6713.20
FeI-6713.74
FeI-6725.36
FeI-6726.67
FeI-6732.07
FeI-6739.52
FeI-6745.11
FeI-6745.97
FeI-6752.71
FeI-6786.86
FeI-6793.26
FeI-6806.85
FeI-6810.27
FeI-6820.37
FeI-6828.60
FeI-6833.23
FeI-6837.01
0
50
100
150
200
250
300
350
400
FeI-6842.69
FeI-6843.66
FeI-6855.72
FeI-6857.25
FeI-6858.15
FeI-6861.94
FeI-6862.50
FeI-6864.32
FeII-4508.28
FeII-4520.22
FeII-4534.17
FeII-4541.52
FeII-4576.34
FeII-4582.84
FeII-4620.51
FeII-4629.34
FeII-4656.98
FeII-4670.17
FeII-4923.93
FeII-5132.67
FeII-5197.57
FeII-5234.63
FeII-5256.94
FeII-5264.81
FeII-5284.11
FeII-5414.07
FeII-5425.25
FeII-5427.81
FeII-5525.12
FeII-5534.85
FeII-5991.38
FeII-6084.11
FeII-6113.32
FeII-6149.25
FeII-6238.39
FeII-6239.94
FeII-6247.56
FeII-6369.46
FeII-6407.29
FeII-6416.93
FeII-6432.69
FeII-6442.97
FeII-6456.39
FeII-6516.09
LiI-6707.83
CI-5052.15
CI-5380.32
CI-6587.62
OI-6158.17
OI-6300.30
NaI-5688.22
NaI-6154.23
NaI-6160.75
MgI-4730.04
MgI-5711.09
MgI-6319.24
MgI-6318.72
AlI-6696.03
AlI-6698.67
SiI-5517.54
SiI-5621.60
SiI-5645.61
SiI-5684.49
SiI-5690.43
SiI-5701.11
SiI-5753.64
SiI-5772.15
SiI-5797.87
SiI-5948.54
SiI-6125.02
SiI-6142.49
SiI-6145.02
SiI-6195.46
SiI-6237.33
SiI-6243.82
SiI-6244.48
SiI-6414.99
SiI-6527.21
SiI-6583.71
SiI-6721.85
SiI-6741.63
SI-6052.67
SI-6757.17
CaI-5261.71
0
50
100
150
200
250
300
350
400

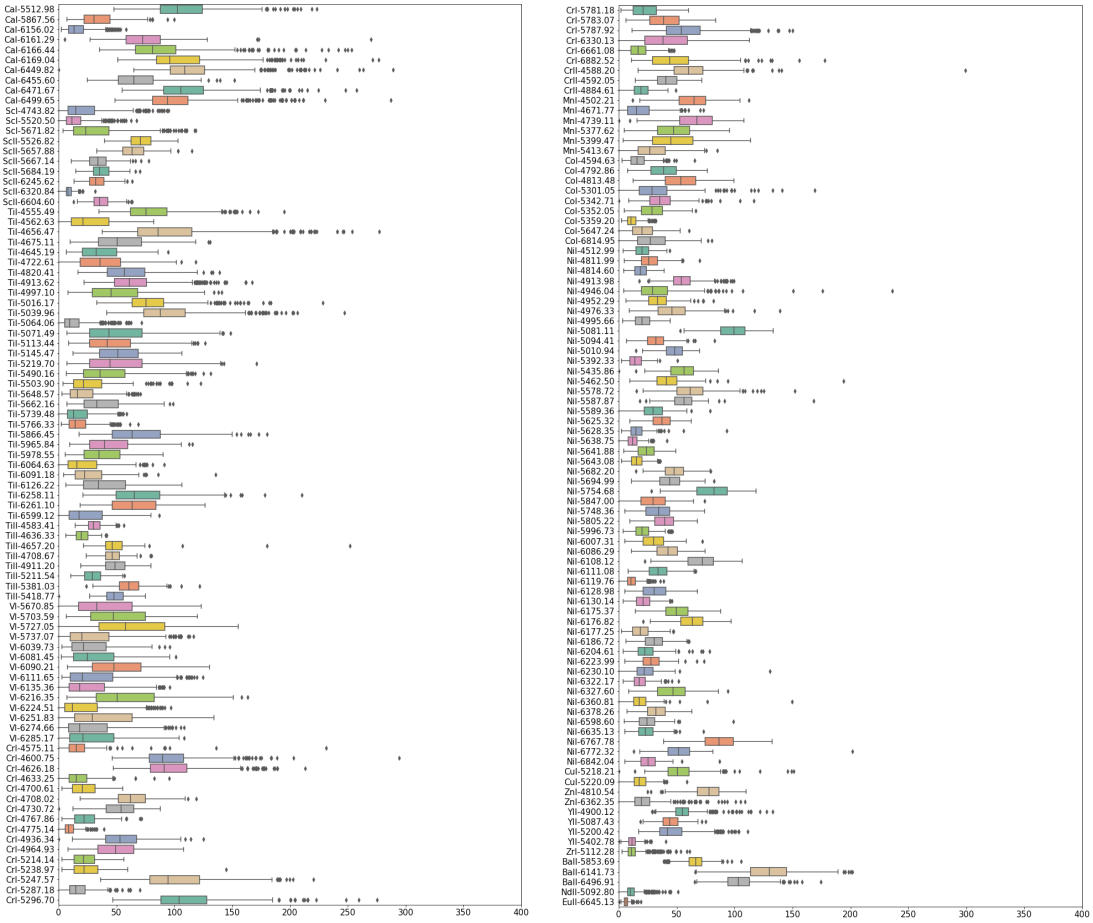


Figure 5.3: Boxplots showing the distributions of predictor variables, which correspond to equivalent widths of specific spectral lines. The 520 features were divided across six subplots. In each box, the central line is the median, the boundaries of the box are the 25th (Q1) and 75th (Q3) percentiles and define the interquartile range (IQR); the whiskers extend to include the maximum (Q3 - 1.5xIQR) and minimum (Q1 - 1.5xIQR) values; outliers are defined as any value reaching past 1.5 times the IQR from either the lower or upper quartile.

Even though the equivalent width of a spectral line can vary from 0 to 400 in some cases, it is interesting to observe that the median or the entire range for some of them is usually higher than it is for others. For example, the FeI-4602.00 line has a higher median value and a larger dispersion towards higher values than all of the spectral lines that appear before it (Top Left). The FeI-5194.95 line follows the same trend, although higher values are more likely to be outliers than actual dispersion of the variable. The FeI-5229.86 and FeI-5263.31 lines clearly stand out compared to all the others on the same plot for having larger equivalent widths, as well as larger dispersions (Top Right). The FeI-6065.49 and FeI-6430.85 lines (Middle Left) or the FeII-4923.93 and NaI-5688.22 lines (Middle Right) show a similar behavior. On the other hand, there are also plenty of spectral lines that have very low equivalent widths, and consistently small dispersions, thus low variance. It is known that a feature with zero variance will have no predictive ability in a model. Features with higher variance have, in principle, the potential for more predictive ability than these features with lower variance, however that is not guaranteed and depends on the specific problem at hand. The predictive ability of features can only be exactly known after fitting a model. Furthermore, a higher variability of a feature can have a physical meaning or be simply derived from the ARES software used to extract the equivalent widths or from the spectra acquisition itself. The same applies to the points indicated as outliers. These were not eliminated prior to further analysis as there are many of them, so even if the boxplot categorizes them as outliers, their nature should be addressed before removal. This analysis could be continued, however, for the sake of simplicity, it is probably more interesting to come back to these data after reporting conclusions from feature selection and predictive modelling.

Another interesting aspect to investigate is the correlation between features. Given the high-dimensional space in this dataset, it is very likely that some features are redundant, that is, highly correlated with each other. This is called multicollinearity (recall from Section 2.2 and Appendix B). Correlations between every predictor variable with the other predictors and with each of the target variables have been evaluated and the results are shown in Figure 5.4. Both Pearson's and Spearman's correlation coefficients were determined, as the former measures strictly linear relationships between variables whereas the later measures the existence of a relationship, being that linear or not. Figure 5.2 shows the correlation matrices for both coefficients. Note that due to the extremely high number of features, it was not possible to fit an entire correlation matrix with visible labels in this format, as it gives a total of 520×520 combinations.

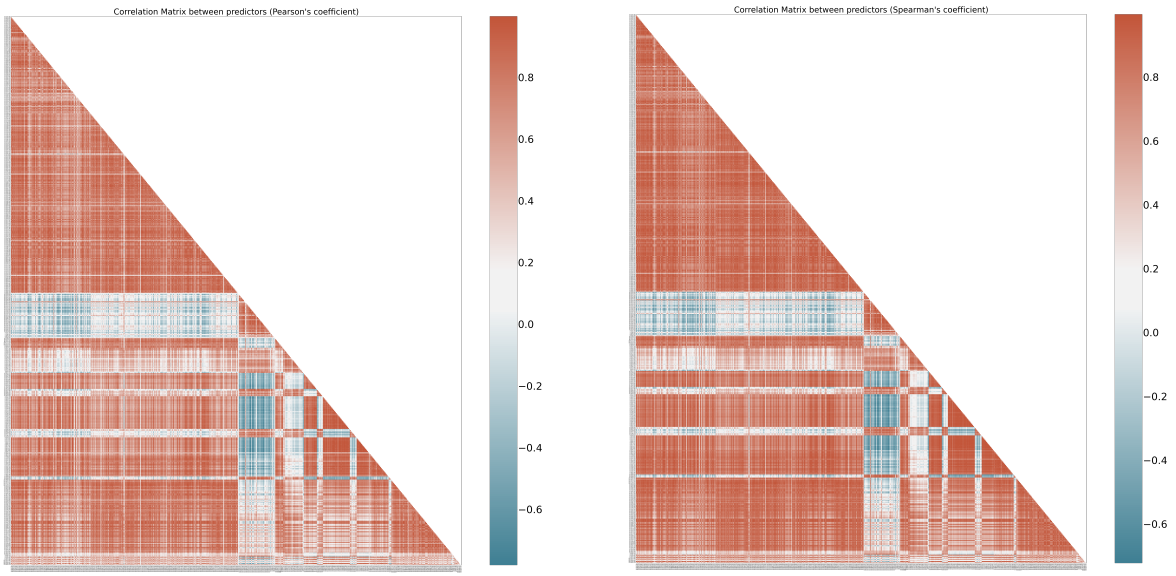


Figure 5.4: Correlation matrices between features. (Left) Pearson's coefficient. (Right) Spearman's coefficent. The colorbar indicates the strength of correlation.

The point of Figure 5.4 is only to highlight the high degree of correlation between features in this dataset. The vast majority of features is linearly correlated by more than 0.7 (Left). The matrices are very similar, but, as expected, the Spearman's correlation matrix (Right) is slightly darker, which essentially means either that besides the linearly correlated features, there are additional features that are correlated but not in a linear fashion, or that some features may not have a perfectly linear relationship, thus their Spearman's coefficients are higher that the Pearson's. The Pearson's coefficient provides important information, as it directly measures the degree of multicollinearity in the data.

However, one may not forget the meaning of the predictor variables from an astrophysics perspective and, despite the removal of collinear features being strongly advised in machine learning, it may be wise to keep comparisons of models with all features *versus* selected features.

A set of Pearson's correlations sorted in descending order can be consulted in Table C.2, Appendix C. The table only shows the first 500 most correlated features, as the total list would be too extensive.

5.2 Feature selection and dimensionality reduction

Different approaches were used to perform feature selection and dimensionality reduction. As previously mentioned, this is interesting from the machine learning perspective, not only to yield more efficient model pipelines (in principle), but also to avoid correlations between predictors that could degrade the models' predictive power.

Additionally, there was an inherent curiosity to try to understand whether a data-driven

approach would select as best features some of the spectral lines that have been already pointed out as important or correlated with each of the three target variables by astronomy standard techniques. Recall the studies referenced on Subsections 2.1.2 and 2.1.3.

In this Section only the results of model-independent methods are presented, mainly because that allows a direct comparison between models. At a first glance, it is intuitive to think that having fewer features will result in lower training times. However, depending on which algorithm is being used, this may not be the case. During training, an objective function is being minimized by the algorithm. Taking the case of the SVR, if the set of features changes, especially depending on whether they are informative or not, the feature space over which the optimization occurs changes as well and possibly the minima of the function will be harder to find, resulting in more steps and longer training in order to satisfy the convergence criteria (data not shown).

Nevertheless, the main objective with these experiments was not to reduce the computation time, despite that being a critical point to address in the future, but rather to build less complex models in terms of dimensionality, and to uncover some new information regarding which and how many lines are necessary to make a reasonable prediction of stellar atmospheric parameters. As mentioned in Subsection 2.1.2, different astrophysical studies sometimes refer different spectral lines as being important to estimate the stellar atmospheric parameters, but that information is very dispersed and hard to compile. Moreover, these experiments work as a test to the problem of multicollineary in this dataset. As previously discussed, some machine learning models are better equipped to deal with this problem, whereas others are not (refer to Section 4.2). Therefore, the question was whether the best performing models remained the same with and without prior features selection. It was expected that models that handle multicollinearity perform the best with no feature selection, but the same may not hold if that multicollinearity is attenuated prior to the modelling stage.

Two different model-independent methods were employed to select features *a priori*, that is, before predictive modelling. These included a univariate feature selection procedure using the `scikit-learn` **SelectKBest** class, with two different methods: **mutual information** and **r-regression**. The results of both methods are shown in Figures 5.5 for Teff, 5.6 for log g and 5.7 for [Fe/H]. Four groups of best features were obtained for each method, containing an increasing number of features: 20, 50, 100 and 200. Only the set containing the best 50 features is displayed in the Figures.

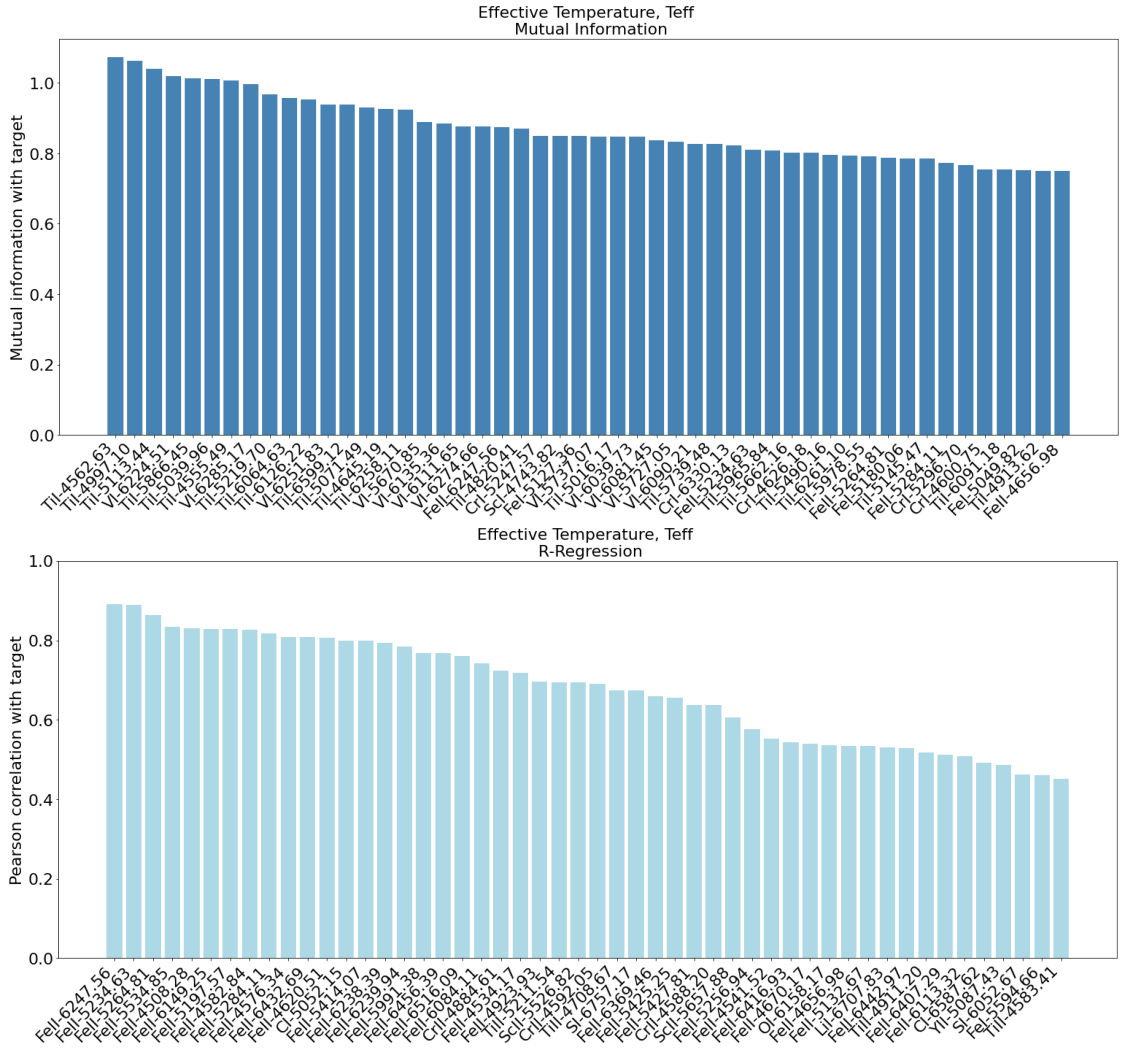


Figure 5.5: Best 50 features for Teff selected by the `scikit-learn` **SelectK** class and their respective scores. (Top) Mutual Information method; (Bottom) R-regression method.

The two methods choose different spectral lines. This observation is consistent for all target variables. This is not entirely surprising, as the methods are completely different from each other in their premise, but it is intriguing which method will perform better in the predictive modelling stage. In the **mutual information** method in Figure 5.5, it is kind of striking that approximately the first 20 features are all from TiI and VI, despite spanning different wavelength regions of the spectra. Spectral lines from these chemical elements have been previously identified as important [53], however, no other studies reported anything specific on the importance of these same lines. Other elements appear as well, including FeI, CrI, ScI, or even ionized elements such as FeII. These are nonetheless much less predominant than the former. If this method works well with the models in the upcoming Section 5.3, then perhaps it can be speculated that TiI and VI, mainly specific lines of these elements at certain wavelengths, may constitute good indicators of the effective temperature of a star. If this is not already widely known from the astronomy perspective, then it might be interesting to explore these spectral lines. On

the contrary, when the best features are obtained by **R-regression**, where a simple Pearson's correlation is calculated between the target variable and each predictor, the result completely changes. In this case, the majority of the selected lines are from ionized elements, predominantly FeII, which is not expected at first, because to the best of our knowledge, these lines are not usually related with the effective temperature Teff.

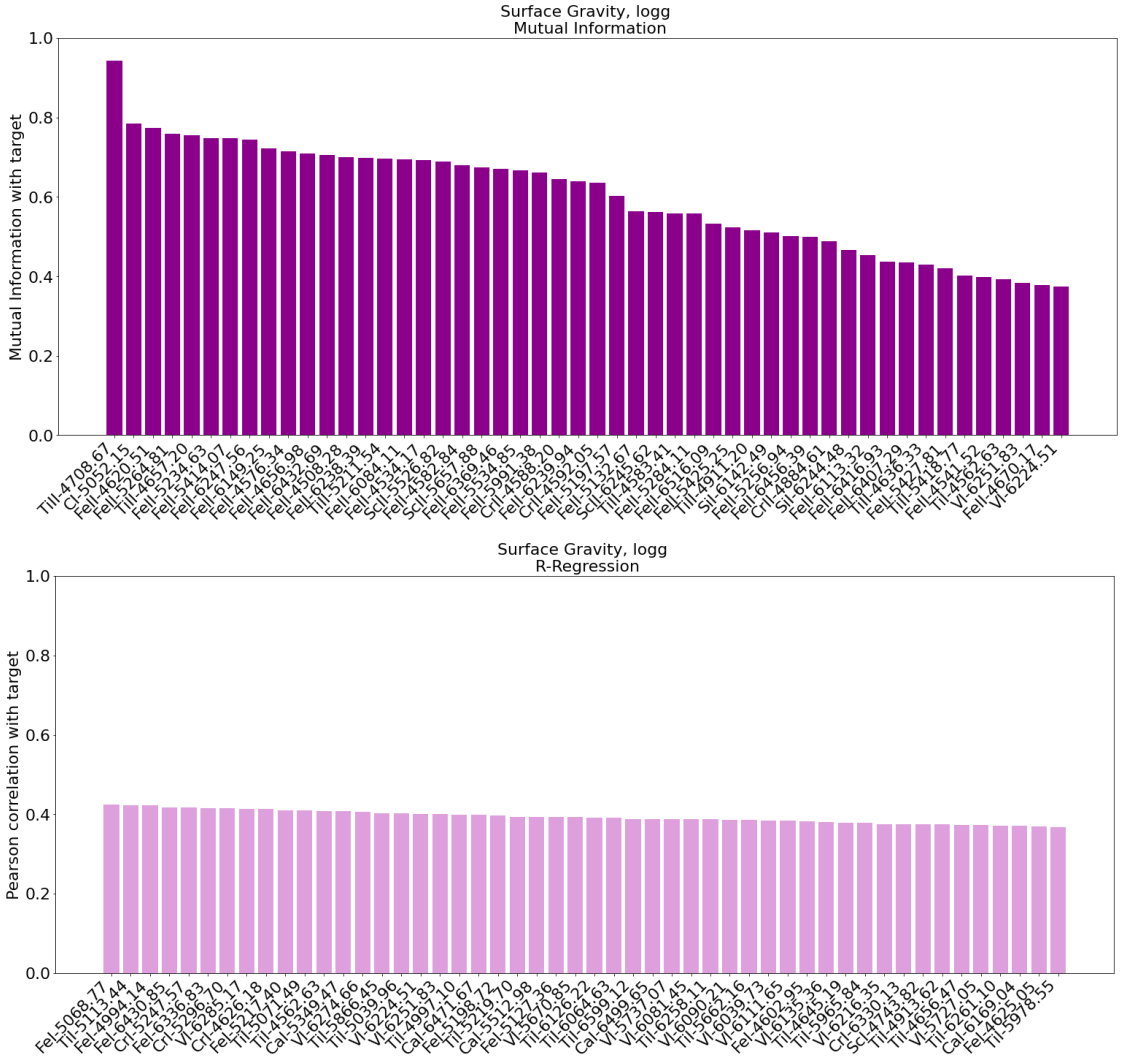


Figure 5.6: Best 50 features for log g selected by the `scikit-learn` **SelectK** class and their respective scores. (Top) Mutual Information method; (Bottom) R-regression method.

Regarding the Surface Gravity log g in Figure 5.6, all spectral lines extracted with the mutual information method corresponded to ionized elements, with a predominance of FeII, but also TiII, ScII, CrII, among others. This is in line with previous studies that reported ionized lines to be more sensitive to surface gravity than neutral lines (recall sections 2.1.2 and 4.1.1), which suggests that the mutual information method may provide reliable results from the physics perspective as well. An interesting observation to be further investigated is the presence of the CI-5052.15 line as the second most informative, which corresponds to the chemical element Carbon. To the best of our knowledge, this has never been mentioned in the literature. Regarding the R-regression

method, the results are again very different, consistent with what was observed for the effective temperature. The absence of ionized elements is notorious in this case.

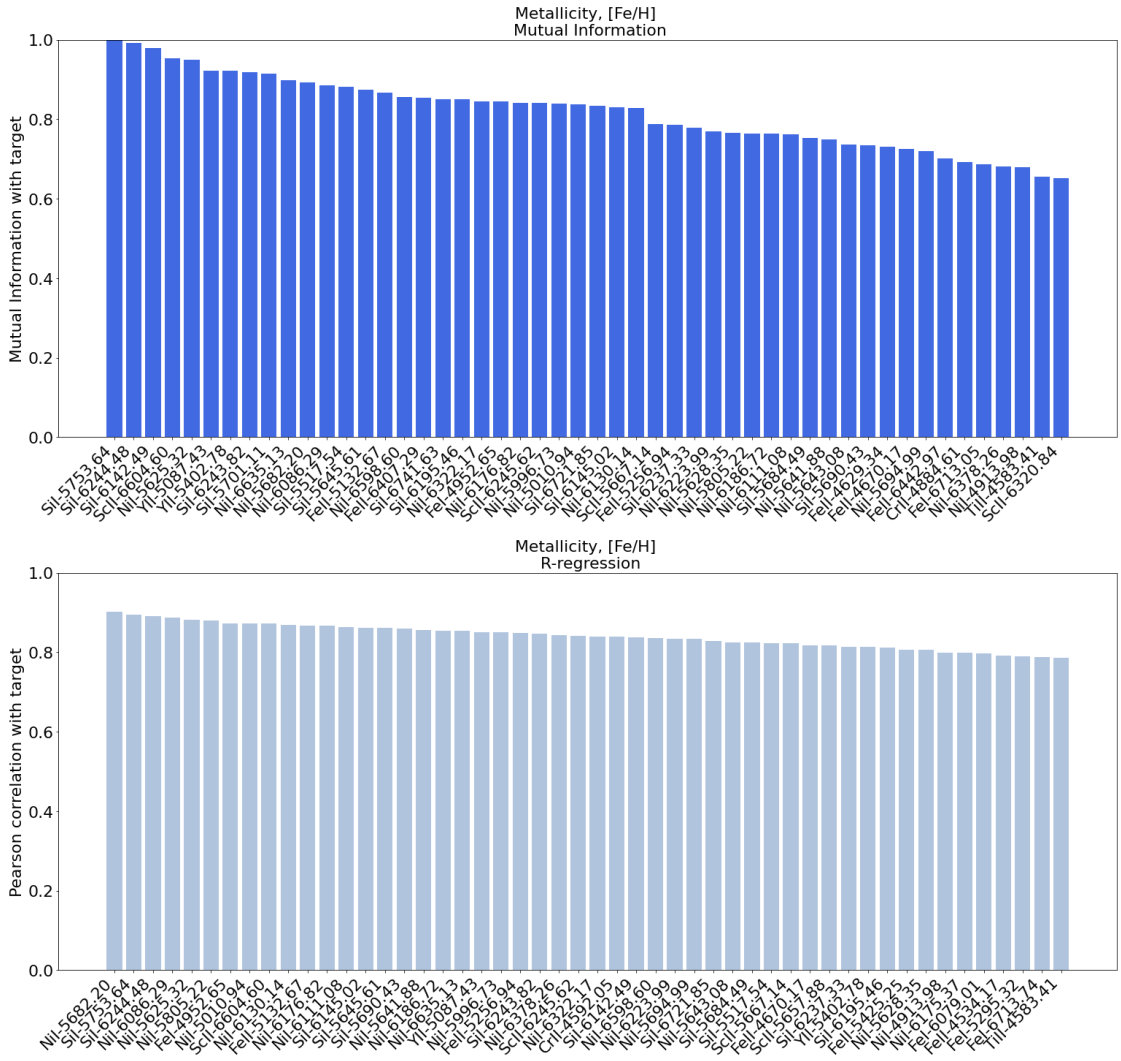


Figure 5.7: Best 50 features for [Fe/H] selected by the `scikit-learn` **SelectK** class and their respective scores. (Top) Mutual Information method; (Bottom) R-regression method.

Finally, even though iron (Fe) is used as a proxy to estimate the metallicity of a star (refer to the definition in Section 2.1.2), it is noticeable in Figure 5.7 that most of the lines are not from Fe, but from other metals, mainly SiI, NiI in neutral state, but also ionized elements such as YII, FeII, ScII or CrII. This is the only stellar parameter for which both methods did not yield very different results. This could be an indication that there is an approximately linear relationship between these lines and the metallicity, since R-regression is based on the Pearson's coefficient that measures linear relationships. In that specific case, mutual information may provide similar results.

It is important to clarify, nonetheless, that these observations are simply intuitive guesses regarding the importance of these spectral lines from an astrophysics perspective. A future

comprehensive analysis will be needed to clarify the relevance of these findings.

5.3 Predictive modelling using Machine Learning

After the exploratory data analysis and feature selection, at least at an informative level, several regression models were employed and compared in order to choose the one that best estimated the target variables. Depending on the algorithm, different hyperparameters were optimized using GridSearchCV.

Of note, three GridSearch procedures were leveraged, one for each target variable separately. Despite the objective being a multiple-output setting that could predict the three targets simultaneously using just one model, it was necessary to understand whether the same model and the same tuned hyperparameters would perform well for all target variables. Moreover, in Section 5.2, best feature selection has been performed separately for each target. Therefore, this section begins with a single-output framework. Critical topics such as model comparison and hyperparameter optimization are carried out for each target variable using GridSearch. The possibility of high bias/variance is addressed based on the cross-validation scores. Experiments with or without prior feature selection are also employed, not only to understand how simple the models could become without significantly loosing predictive power, but also to clarify whether the best model and parameters for these data would remain the same in both scenarios. Finally, multi-output versions of the same algorithms are compared to the single-output ones. As mentioned on Section 4, 80% of data has been used for training the models and 20% for testing. Recall that the GridSearch procedure only uses the training set (which is internally split into several validation sets for cross-validation), so the test set is kept as a *hold-out* sample that is never seen during model optimization to completely avoid data leakage.

5.3.1 Single-output models

Evaluating single-output models was a crucial preliminary experiment to conclude whether the best performing one and its tuned hyperparameters were similar for each target variable or not and if a good compromise could be found in finding a common model to predict the three simultaneously.

5.3.1.1 No feature selection

The first approach consisted in performing a GridSearch to compare several the models and a respective range of hyperparameters for each of them, using all features. This procedure was performed as explained in Section 4.4.3.1, Figure 4.2. The boxplots in Figure 5.8 show the distributions of the MAE for each of the evaluated models with the best tuned hyperparameters for a 10-fold cross-validation. For visual simplicity, these optimized hyperparamaters are not

indicated in the plot, but can be found on Table 5.1, where performance is evaluated in the *hold-out* test set.

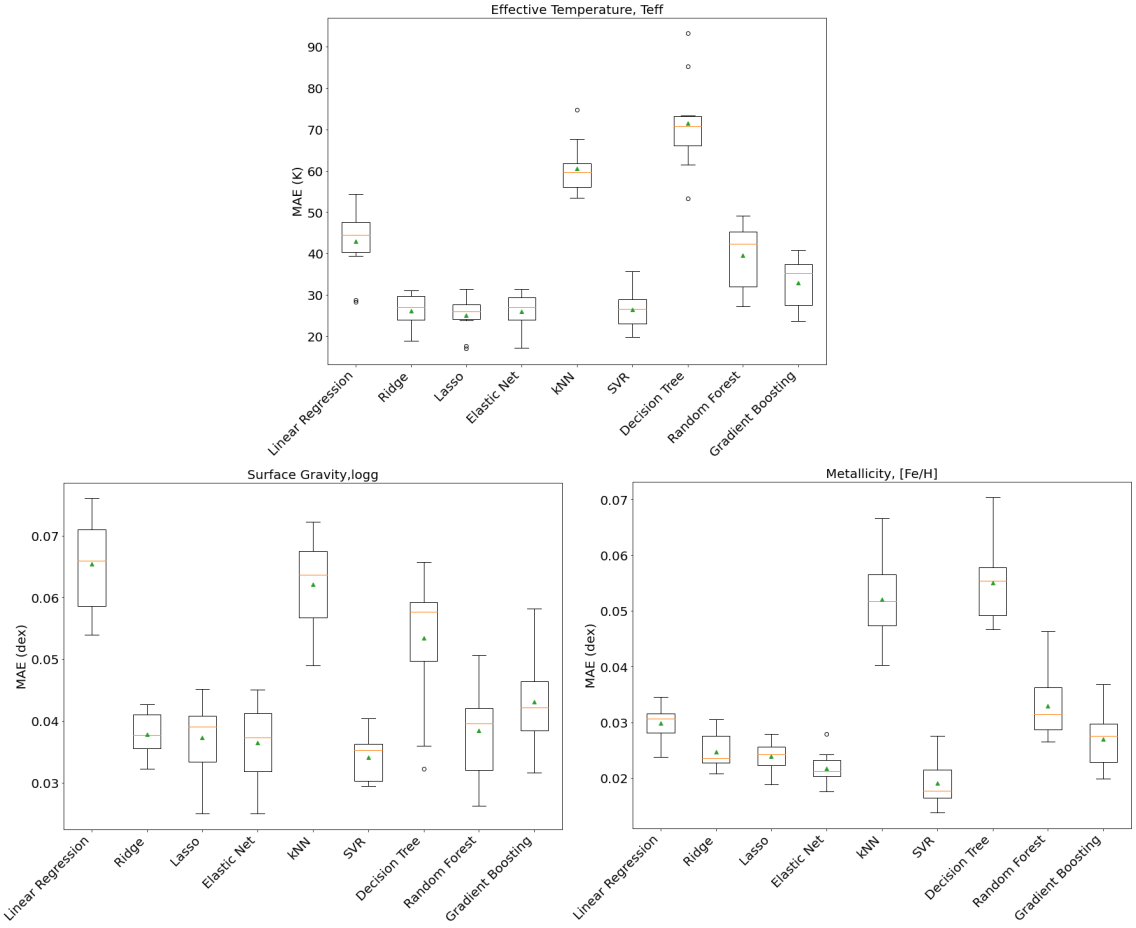


Figure 5.8: MAE score from 10-fold cross-validation for the indicated models and target variables. The scores were calculated on the validation sets of each of the 10 folds. In each box, the central red line is the median, the boundaries of the box are the 25th (Q1) and 75th (Q3) percentiles and define the interquartile range (IQR); the whiskers extend to include the maximum (Q3 - 1.5xIQR) and minimum (Q1 - 1.5xIQR) values; outlier data points are defined as any value reaching past 1.5 times the IQR from either the lower or upper quartile. The green triangles are the means. (Top Left) Effective Temperature, Teff; (Top Right) Surface gravity, log g; (Bottom) Metallicity, [Fe/H].

A visual inspection of Figure 5.8 reveals that, apart from the kNN and Decision trees (which are known to suffer from overfitting problems), all the other algorithms performed reasonably well at making predictions for the three target variables in a 10-fold cross-validation, with respect to the scale of each target. Linear regressions with regularization, either L1 (Lasso), L2 (Ridge) or both (ElasticNet) outperformed powerful ensemble algorithms such as Random Forests and Gradient Boosting Trees. This could seem quite surprising, however, given the high dimensional feature space and additionally, the high correlation between several predictor variables, it cannot be ruled out that this group of algorithms works well because they possess inherent mechanisms to deal with this problem by significantly attenuating the influence or even eliminating some

some of the features via the introduction of penalization terms in the cost function. This reduces multicollinearity and prevents the model from overfitting the training data and result in high variance (revisit section 2.2.2). Random Forests and Gradient Boosting Trees do not have mechanisms to deal with multicollinearity and, despite being powerful, may struggle in cases where the number of observations N is equal or lower than the number of features p ($N << p$ problem) [3].

On the other hand, it is still possible that there is in fact a linear relationship between the predictors and the target variables, especially for the effective temperature or even more pronounced in the case of metallicity, since a simple linear regression with no regularization does not perform so bad itself. The same does not hold for the surface gravity, and so it also becomes clear that this target variable is the most difficult to predict. This is not completely unexpected, as it was mentioned previously on this thesis that this parameter, unlike the temperature, is not so sensitive to spectral lines and the lines to which it is mostly sensitive to (ionized chemical species) are often difficult to measure with high accuracy, thus it is possible that they carry larger equivalent width measurement errors that pose additional struggles to the learning algorithms. The inter-quartile ranges are also larger for this target variable, which means the scores between different cross-validation folds vary more.

Apart from these aspects, which are all interesting to explore further, the SVR model proved to be quite robust at predicting the three atmospheric parameters. It reaches scores very similar to the regularized linear regressions in the case of the effective temperature and it was the best performing model for both surface gravity and metallicity. Contrarily to Ridge, Lasso or Elastic Net, the SVR model does not raise so many concerns as to the good performance being simply due to the multicollinearity between the features. Therefore, this model was a strong candidate for a multi-output strategy to predict the three targets with the same model.

A second experiment that might seem unnecessary given the main scope of the work, but is in fact informative, consisted in observing the scores of each fold of the cross-validation algorithm both for the respective train and validation splits. This is shown on Figure 5.9. To simplify, only three models are shown for each target variable: a regularized linear regression, a SVR and an ensemble (Gradient Boosting). The main conclusion from this experiment is that every algorithm, even the ones that seem to perform well and yield very small errors considering the ranges of each target variable, are not completely free from a slight overfitting. Notice the difference between the train and validation curves: all the models, regardless of the target variable, perform well on the training data, which is expected. However, the curves for the validation data show that the scores vary from one fold to another, sometimes reaching almost 2-fold differences. This means that the models predictions are sensitive to the data - sometimes they reach a score very close to the training one, but in other folds the performance decays slightly. Interestingly, the models that perform better, the Ridge and the SVR, have a slight worse performance than the Gradient Boosting on the train set, especially for Teff and [Fe/H], but generalize better on the test set, likely because regularization introduces some bias that attenuates overfitting.

Since this behavior is common to every model, which has been fine-tuned, and that the data have been normalized and pre-processed, this is probably due to the small size of the dataset, which seems to be a limiting factor that may not allow us to go beyond a certain margin of error. Despite this, the amount of overfitting is rather low given the size of the dataset, with promising evaluation metrics on the $hold-out$ test set for the majority of the algorithms, as shown on Table 5.1.

This table displays the best tuned hyperparameters, as well as the MAE, RMSE and R^2 as evaluation metrics. In terms of the best performing models, the conclusions are essentially the same as already stated for the cross-validation procedure, with the effective temperature Teff being best estimated by the regularized linear regressions followed by the SVR, and both surface gravity and metallicity being best predicted by an SVR.

Table 5.1: Evaluation metrics for several regression models to predict Teff, log g and [Fe/H], using all features. Data has been Min-Max normalized, except for SVR in the case of Teff, Lasso and ElasticNet in case of log g and the three of them in the case of [Fe/H].

Regression model	**Hyperparameters**	**Target variable**	**MAE**	**RMSE**	R^2
Dummy/median	median	Teff	386.2	467.9	-0.037
		Log g	0.089	0.14	-0.03
		[Fe/H]	0.19	0.24	-0.015
Linear Regression	—	Teff	45.4	63.1	0.98
		Log g	0.055	0.072	0.72
		[Fe/H]	0.034	0.046	0.95
Ridge	alpha=1 solver='saga'	Teff	27.2	40.3	0.99
	alpha=0.5 solver='saga'	Log g	0.033	0.052	0.85
	alpha=0.5 solver='lsqr'	[Fe/H]	0.024	0.035	0.97
Lasso	alpha=0.1 selection='random'	Teff	28.9	42.4	0.99
	alpha=0.01 selection='random'	Log g	0.034	0.051	0.86
	alpha=0.01 selection='random'	[Fe/H]	0.024	0.044	0.96
ElasticNet	alpha=0.01 l1_ratio=0.75 selection='random'	Teff	27.0	40.3	0.99

to be continued ...

Regression model	**Hyperparameters**	**Target variable**	**MAE**	**RMSE**	R^2
	alpha=0.01 l1_ratio=0.75 selection='random'	Log g	0.034	0.051	0.86
	alpha=0.01 l1_ratio=0.25 selection='random'	[Fe/H]	0.027	0.044	0.96
kNN	k=3	Teff	68.8	100.7	0.74
	k=5	Log g	0.068	0.11	0.54
	k=3	[Fe/H]	0.042	0.063	0.89
SVR	C=5000 epsilon=10 kernel:'rbf'	Teff	30.8	47.8	0.99
	C=5 epsilon=0.01 kernel:'rbf'	Log g	0.026	0.036	0.93
	C=1 epsilon=0.01 kernel:'rbf'	[Fe/H]	0.017	0.023	0.99
Decision tree	max_depth=15 max_features=all min_samples_split=8	Teff	74.9	96.2	0.96
	max_depth=5 max_features=all min_samples_split=8	Log g	0.060	0.094	0.52
	max_depth=20 max_features=all min_samples_split=8	[Fe/H]	0.053	0.067	0.90
Random Forest	n_estimators=250 max_depth=20 max_features='sqrt' min_samples_split=4	Teff	48.9	76.2	0.98
	n_estimators=100 max_depth=10 max_features=all min_samples_split=4	Log g	0.038	0.051	0.81
	n_estimators=250 max_depth=10 max_features='sqrt' min_samples_split=2	[Fe/H]	0.028	0.036	0.97

to be continued ...

Regression model	**Hyperparameters**	**Target variable**	**MAE**	**RMSE**	R^2
Gradient Boosting	n_estimators=500 max_depth = 10 learning rate = 0.01 subsample = 0.5 max_features = 'sqrt' min_samples_split=8	Teff	40.8	57.7	0.99
	n_estimators=500 max_depth = 10 learning rate = 0.01 subsample = 0.5 max_features = 'log2' min_samples_split=8	Log g	0.032	0.050	0.87
	n_estimators=500 max_depth = 10 learning rate = 0.01 subsample = 0.5 max_features = 'log2' min_samples_split=8	[Fe/H]	0.027	0.037	0.97

The best model performance for each target variable is also visually depicted on Figure 5.10, that illustrates the predicted *versus* the true values for each target variable, as well as the model residuals, which, in an optimal scenario, should distribute around zero with a normal distribution. For the three target variables, the test cases practically overlap the training cases on the $y = x$ line, with the highest dispersion being observed for log g, which, as mentioned above, is the hardest stellar atmospheric parameter to predict. Still, the distribution onto the $y = x$ with an $R^2 = 0.93$ is an excellent approximation to the ground-true values. The residuals also show a reasonably acceptable normal pattern for all the three targets, even though they are slightly more dispersed for log g and [Fe/H]. However, since the same is observed for the train set, that might be noise or inherent to the data distribution.

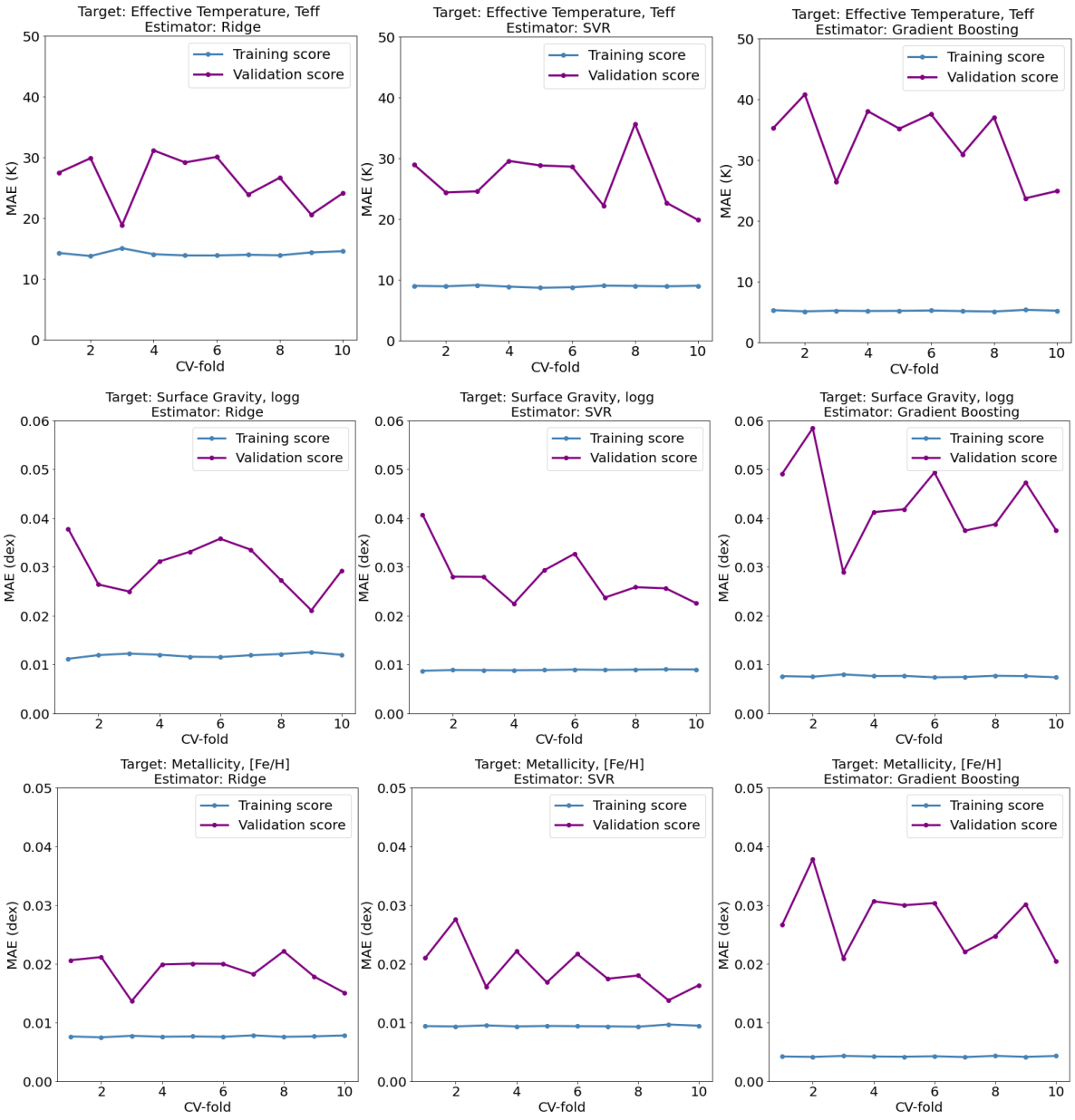


Figure 5.9: MAE from 10-fold cross-validation for the indicated models for the different target variables. The scores were calculated for both the training (blue) and validation (purple) sets of the 10 folds. (Top) Effective Temperature, Teff; (Middle) Surface gravity, log g; (Bottom) Metallicity, [Fe/H].

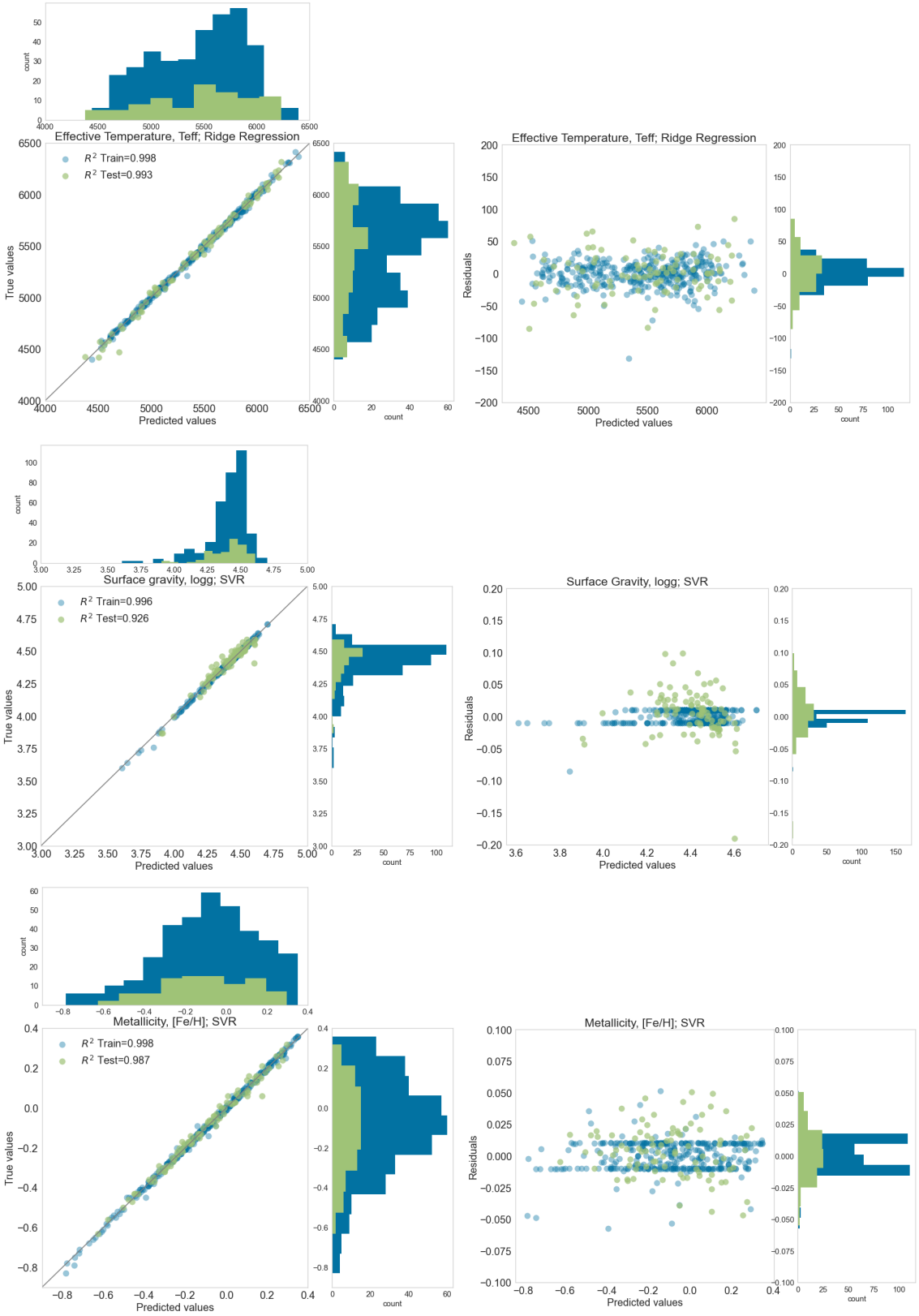


Figure 5.10: True *versus* predicted values and residuals predicted by the best performing model, according to Table 5.1 with the indicated tuned hyperparameters, for each target variable. (Left) From Top to Bottom: Effective Temperature, Teff; Surface Gravity, logg; and Metallicity, [Fe/H]. (Right) Residuals of the respective models on the left side. The plotted data corresponds to the train set in blue, comprised of 351 stars, and the *hold* − *out* test set in green, comprised of 91 stars. The histograms on the side show the respective distributions of the true and predicted values, and the distribution of the residuals, for both the train and the test sets.

5.3.1.2 Model-dependent feature selection

In Sections 4.3 and 5.2, strategies to select features independently of any model were discussed. Those were based either on correlations between the features and targets or on information theory methods.

However, some of the models trained on this section also allow the possibility of extracting the most important features, or the ones that have more weight on the algorithm decisions, including the linear regression and variants, as well as Random Forests and Gradient Boosting. However, the definition of feature importance in these algorithms differs. Whereas the feature importance in decision trees is directly inferred from the number of times that feature has been used to split a node, because it leads to information gain or a decrease in entropy from one level to the next (note that in regression problems, it consists in the minimization of the error, generally the MSE), in linear regressions that is reflected on the absolute value (weight) of the coefficient associated with that feature.

Feature importances and coefficient values were extracted after model training, for tree-based and linear regression algorithms, respectively, and the sorted results for the 50 best features are shown in Figures 5.11 and 5.12.

Starting with the feature importances for the Random Forest and Gradient Boosting algorithms (Figure 5.11), from the 50 features selected, Random Forest and Mutual Information (see Subsection 5.2) share 38, 37 and 44 of them, for Teff, log g and [Fe/H], respectively, whereas Gradient Boosting shares 38, 32 and 35 for the same order of the targets. Further investigation will be necessary to understand why these model-dependent algorithms choose almost the same 50 features as the most informative as an algorithm that does not depend on any model. There is the possibility that this is related with Random Forests and Gradient Boosting trying to minimize entropy by minimizing the MSE, which, as explained on Section 4.3, also has a relation with the mutual information approach. The details must be clarified mathematically, especially how MSE relates with entropy. Nevertheless, it is an interesting finding that different methods of feature selection consider the same features as being the most informative.

Regarding the linear regression and its regularized versions (Ridge and Lasso), Figure 5.12 shows the coefficient values for the effective temperature (Top), surface gravity (Middle) and metallicity (Bottom), sorted by their absolute value (despite being displayed with their true values).

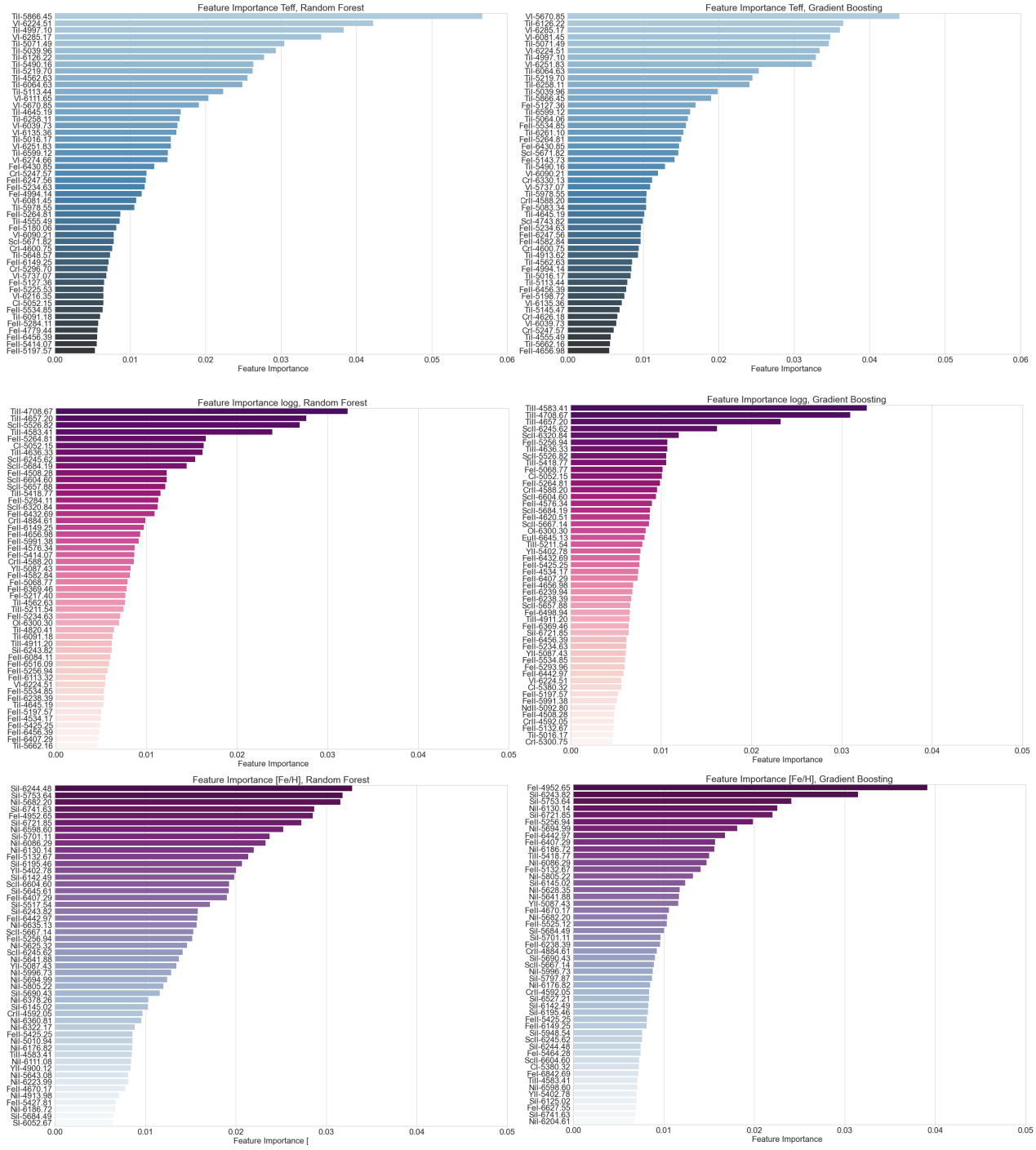


Figure 5.11: Feature importance for single-output Random Forest and Gradient Boosting models. The best 50 features are displayed. From Top to Bottom Left, important features for Random Forests for each target variable: Teff , log g and [Fe/H], respectively. From Top to Bottom Right, import features for Gradient Boosting, for the same target variables.

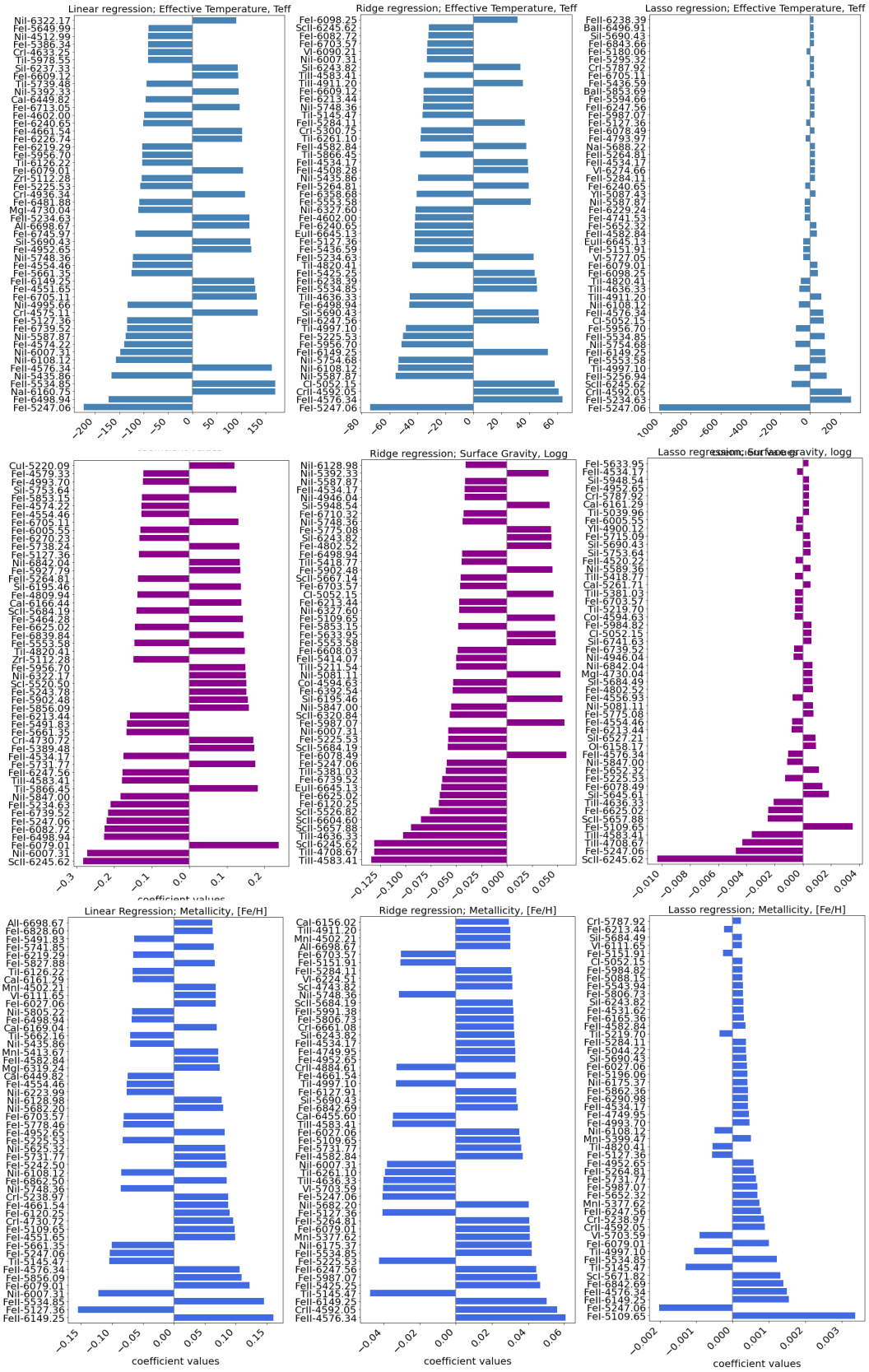


Figure 5.12: Feature selection based on the 50 highest coefficients obtained for the indicated models. From Top to Bottom Left, Linear regression for each target variable: Teff, log g and [Fe/H]. In the Middle, Ridge regression, and on the Right, Lasso regression, for the same target variables.

It is possible to observe that the highest coefficients differ between these algorithms. This was expected, since Ridge and Lasso regressions tend to penalize very high coefficients to prevent the model from overfitting, so the important features may change when compared to a simple linear regression. However, whereas Ridge penalizes high coefficients by bringing them close to zero, Lasso can eliminate features by setting their weight to exactly zero. Thus, it would not be expected their coefficients to have the same weight, but it would be expected that they at least share some similar highest coefficients, if those really correspond to informative features. Despite the different coefficient values, there are some 'important features' shared between Ridge and Lasso, including the FeI-5247.06, CrII-4592.05 or the NiI-5754.68 lines for Teff, the TiII-4583.41, TiII-4708.67 and ScII-6245.62 lines for log g and the FeII-4576.34, the FeII-6149 and TiI-5145.47 lines for [Fe/H]. There are other common lines, but they rank very different places between the two models.

5.3.1.3 With Feature selection

As described on Subsection 4.3, additional experiments were necessary with *a priori* feature selection. This was mainly related with the concern of the high dimensional and highly correlated feature space in this dataset, as some models are better equipped to deal with this problem than others. Whereas ridge and lasso regressions, as well as kernel-based approaches with regularization, can eliminate or weight-down the coefficients that represent the influence of a predictor on the model and avoid problems such as multicollinearity, Random Forests, on the other hand, may struggle with this kind of data, despite being a very powerful algorithm [107].

Therefore, we wondered whether the best models would remain the same if feature selection was performed first and the models trained with a reduced set of more informative features. In a first instance, the models from Table 5.1 were retrained using 10-fold cross-validation with the 20, 50, 100 and 200 best features obtained from the **mutual information** and the **r-regression methods**, which can be recalled from Figures 5.5, 5.6 and 5.7 for 50 features.

Some models were excluded from this analysis because they were not relevant to answer the question at hand. Only the best performing models (often, SVR), the Ridge and Lasso regressions that deal well with high dimensional features spaces, and powerful model ensembles that do not handle highly correlated features spaces so well were kept for this experiment, to access how the predictive power of each type of algorithm varies with decreasing number of features.

The results for decreasing number of selected best features by each method are displayed on Figure 5.13. Note that these models and hyperparameters were not optimized for each number of features in this figure. The models are exactly the ones with the same parameters tuned for all features. This means the same exact model is compared among different numbers of features.

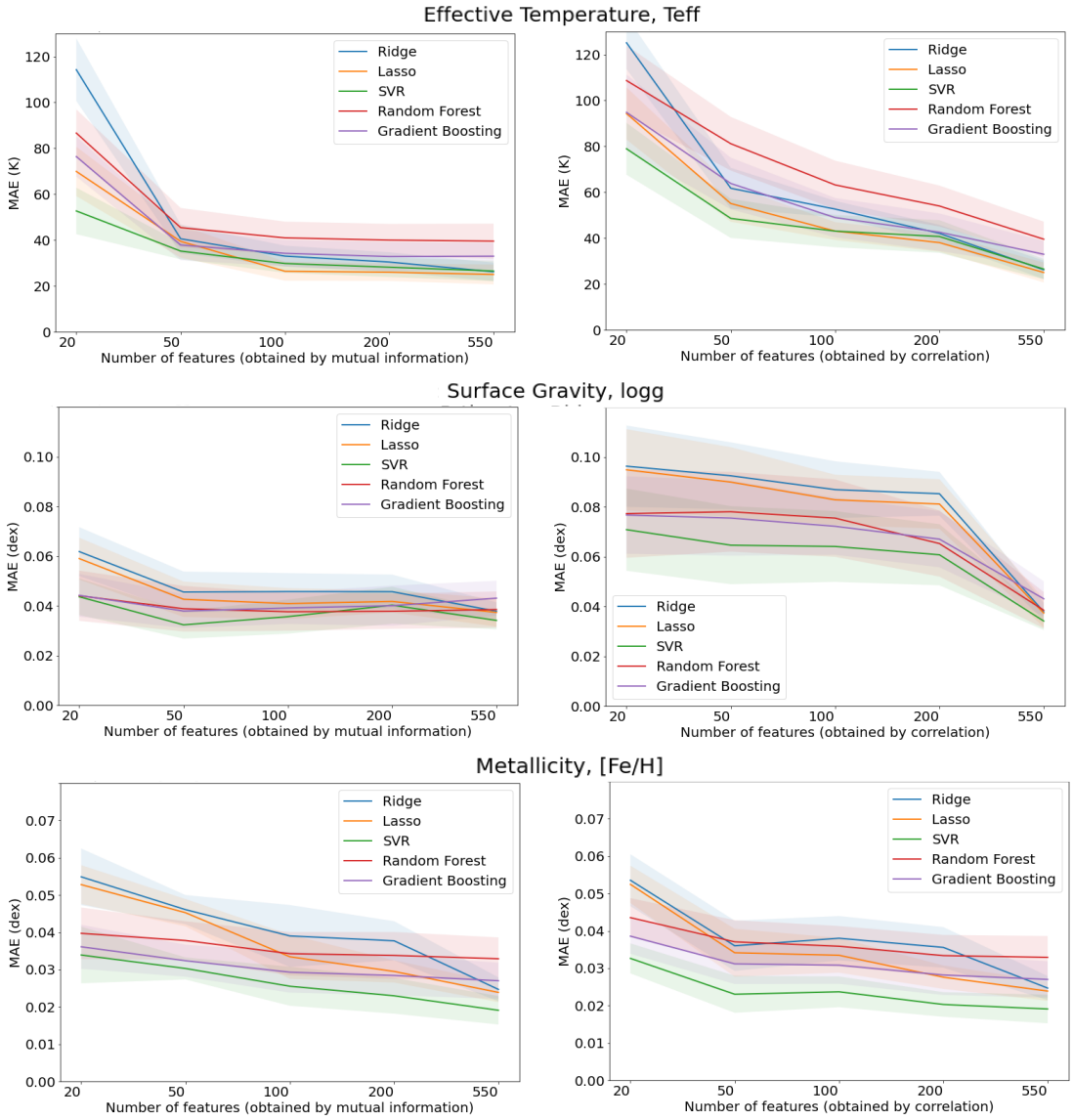


Figure 5.13: 10-fold cross-validation scores (MAE) of the best performing tuned models from Table 5.1 for each of the target variables, using two different methods of feature selection: (Left) Mutual Information; (Right) R-regression (Pearson's correlation). The plots from Top to Bottom refer to the different target variables as indicated. The solid lines are the mean of the 10-fold validation scores for each number of features and the shaded intervals correspond to the standard-deviation.

The first interesting observation is that performance does not seem to decay significantly when only using 50 features, selected by the **mutual information** method. This is a consistent observation for every target variable, with some models to predict log g even reaching similar performances as the same model with no feature selection, for example the SVR, or even slightly surpassing it, such as in the Gradient Boosting case. However, this does not hold when the method of feature selection is based on direct Pearson's correlations between the target variable and each of the predictors. In this case, the predictive power of every model decayed consistently

for all target variables with decreasing number of features. Two possible reasons to explain this are the fact that the Pearson's coefficient (as explained in subsection 4.2) only considers linear relationships between variables, in this case, between each predictor and the target variables. It is more or less intuitive to see how that may not be helpful, as although the selected predictors are the more correlated with each target, they can still, and most probably are, correlated with each other as well. This means this method does not abolish the problem of multicollinearity, which has been shown known to exist in this dataset. Consequently, models that could be struggling to handle that problem will still struggle, but now they also have less information, so performance degrades even more. The Ridge and the Lasso regressions should not struggle since they handle the problem of multicollineary. It can be nonetheless speculated that their performance also degrades because they perform their own feature selection or weighting internally, so adding a previous one may result in an amplified loss of information.

The second interesting point is related with how much each model's performance degrades with decreasing number of features. Notice that, in general, the Ridge and Lasso regressions tend to degrade more than the other three models with a decreasing number of features and become more unstable, both for the **mutual information** and the **r-regression** methods, especially the Ridge. This somehow supports the point of these experiments, which was whether the Ridge and Lasso regressions performed very well in this dataset because of the high dimensionality and highly correlated feature space (indeed, they were not always the best model for every target variable, but were always the second or third best). Especially in the log g and [Fe/H] cases, this implies that if 50 or 100 best features are selected prior to model fitting instead of using all features, then using a SVR or Gradient Boosting may be advantageous compared to Ridge or Lasso Regressions. That difference is even more pronounced if only 20 best features were available, as all the models handle it better than Ridge. However, that would never be a realistic choice, as all the models have worse performances for 20 features compared to using more features.

Finally, it is worth highlighting that SVR is indeed a very promising model, as it seems to be more resistant to the reduction in the number of informative features than any of the other models and, additionally, shows the best performance for both log g and [Fe/H].

Note also that, despite this conclusion being valid, these are probably not the best models that can be achieved for a decreased number of features, as hyperparameters were not fined-tuned again and are the same used to tune models using all the features. In terms of consistency, that is a correct approach.

However, if one is aiming at the best approach to predict the target variable(s), then it is not known whether this same model and parameters are still the best performing for the new feature subset or whether another model could now show a better predictive power. That requires a new GridSearch optimization. Following this rationale, new GridSearches were leveraged for the same models and hyperparameter space, but for a reduced subset with only the best 50 features for each target variable selected with different methods described in Subsection 4.3. The results of these re-tuned models are shown on Table 5.2, evaluated on the $hold-out$ test set. Only a

model of each type is displayed for simplicity as this suffices to support the conclusions.

Table 5.2: Evaluation metrics for several regression models to predict Teff, log g and [Fe/H], using only the best 50 features selected for each target variable using the Mutual Information method. Data has been Min-Max normalized, except for SVR in the case of Teff, Lasso and ElasticNet in the case of log g and the three of them in the case of [Fe/H]. The scores reported are from the $hold-out$ test set.

Regression model	**Hyperparameters**	**Target variable**	**MAE**	**RMSE**	R^2
SVR	C=1000 epsilon=1 kernel:'rbf'	Teff	38.8	62.1	0.98
	C=1 epsilon=0.01 kernel:'rbf'	Log g	0.033	0.051	0.89
	C=10 epsilon=0.01 kernel:'rbf'	[Fe/H]	0.027	0.039	0.97
Lasso	alpha=0.1 selection='random'	Teff	44.6	57.7	0.99
	alpha=0.01 selection='random'	Log g	0.044	0.062	0.79
	alpha=0.01 selection='random'	[Fe/H]	0.046	0.060	0.91
Gradient Boosting	n_estimators=500 max_depth = 10 learning rate = 0.01 subsample = 0.5 max_features =all min_samples_split=8	Teff	39.4	55.4	0.99
	n_estimators=500 max_depth = 5 learning rate = 0.01 subsample = 0.5 max_features = all min_samples_split=4	Log g	0.031	0.044	0.89
	n_estimators=500 max_depth = 5 learning rate = 0.01 subsample = 0.5 max_features = all min_samples_split=8	[Fe/H]	0.031	0.042	0.96

Compared with the results shown on Table 5.1, the same algorithms show similar scores for all the three target variables, with Gradient Boosting showing the minor performance degradation and even outperforming the Lasso regressor. Although SVR shows some performance degradation in the order of 1 - 1.5-fold, Lasso shows higher degradations.

More importantly, this experiment indicates that using only the best 50 features for each of

the target variables suffices to reach reasonable predictive powers.

5.3.2 Multiple-output models

A multiple-output problem, also called multivariate, occurs when the response consists not only of one variable, but of $d > 1$ output variables, with $Y \in \mathbb{R}^d$. The interest is then finding a relationship between each output Y and some feature variables $X \in \mathbb{R}^p$, thus a multi-output multiple regression analysis. Unlike single-output multiple regression (with $d = 1$), which includes multiple features $X \in \mathbb{R}^p$ but only one target variable Y, multi-output regression wants to find the relationship of several response variables with all the features X simultaneously.

In the literature, it is often claimed that the interest in performing multi-output estimations lies in exploring possible dependencies between the different target variables, in case those exist, thus enhancing the predictive power of a model. However, it should be noted that this is only expected if there is a strong intuition, or a certainty, that there are correlations between the different target variables.

Here, we used the the **MultioutputRegressor** from `scikit-learn` to provide a multi-output setting. This strategy works as a wrapper, thus it internally runs a model for each target in parallel and outputs the final results for the three targets simultaneously (refer to Section 4.4.4 for details). Since the target variables do not interact in this setting, the results were not expected to change compared to the respective single-output models. However, a new GridSearch was carried out to optimize the hyperparameters, which in this case must be the same for the three targets. The results of the 10-fold cross-validation and the final scores in the *hold-out* test set are displayed in Figure 5.14 and Table 5.3. As expected, the best performing models were the same as in a single-output framework, including the SVR, as well as the regularized linear regressions, followed by Gradient Boosting. Note that the results are not exactly the same as in single-output models because the hyperparameters are slightly different as they have been optimized again. The new hyperparameters can be consulted on Table 5.3, as well as the scores in the *hold-out* test set, for each of the targets separately. These models used all the features.

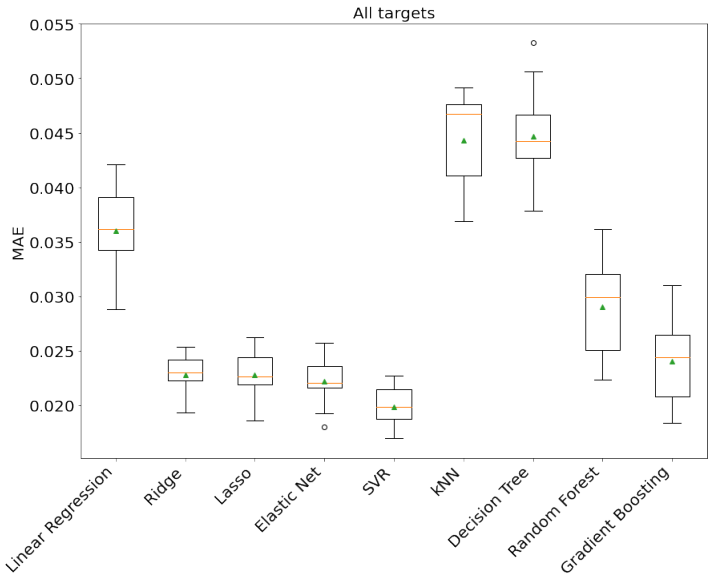


Figure 5.14: MAE from 10-fold cross-validation for the indicated multi-output models - the total score is calculated over the three target variables, which have been Min-Max normalized. The scores were calculated on the validation sets of each of the 10 folds. In each box, the central red line is the median, the boundaries of the box are the 25th (Q1) and 75th (Q3) percentiles and define the interquartile range (IQR); the whiskers extend to include the maximum (Q3 - 1.5xIQR) and minimum (Q1 - 1.5xIQR) values; outliers are defined as any value reaching past 1.5 times the IQR from either the lower or upper quartile. The green triangles are the means.

Table 5.3: Evaluation metrics for different regression models to predict Teff (K), log g and [Fe/H] simultaneously, using the **Multioutput Regressor**. These metrics refer to the final prediction score on the *hold-out* test set, after selection of the best models using the training set. Data has been Min-Max normalized, except the Lasso and ElasticNet estimators, for which no normalization yielded the best result. The target variables have also been Min-Max normalized.

Regression model	**Hyperparameters**	**Target variable**	**MAE**	**RMSE**	R^2
Dummy(median)	median	Teff	386.2	467.9	-0.037
		Log g	0.089	0.14	-0.03
		[Fe/H]	0.19	0.24	-0.015
Linear Regression	—	Teff	45.4	63.1	0.98
		Log g	0.055	0.072	0.72
		[Fe/H]	0.034	0.046	0.95
Ridge	alpha=0.5 solver= 'saga'	Teff	27.1	40.6	0.99
		Log g	0.033	0.052	0.85
		[Fe/H]	0.023	0.035	0.97
Lasso	alpha=0.01 selection='random'	Teff	29.4	43.9	0.99
		Log g	0.035	0.051	0.87
		[Fe/H]	0.024	0.043	0.96
ElasticNet	alpha=0.01 l1_ratio= 0.75 selection='random'	Teff	29.3	40.7	0.99
		Log g	0.034	0.051	0.86
		[Fe/H]	0.025	0.045	0.96
kNN	k=3	Teff	68.8	100.7	0.74
		Log g	0.068	0.11	0.54
		[Fe/H]	0.042	0.063	0.89
SVR	C=5 epsilon=0.01 kernel:'rbf'	Teff	32.4	49.7	0.99
		Log g	0.032	0.045	0.89
		[Fe/H]	0.018	0.024	0.99
Decision Tree	max_depth=15 max_features=all min_samples_split=8	Teff	72.4	95.3	0.96
		Log g	0.037	0.057	0.82
		[Fe/H]	0.033	0.043	0.96
Random Forest	n_estimators=250 max_depth=20 max_features='sqrt' min_samples_split=4	Teff	46.9	65.8	0.98
		Log g	0.037	0.057	0.82
		[Fe/H]	0.033	0.043	0.96
Gradient Boosting	n_estimators=500 max_depth = 10 learning rate = 0.01 subsample = 0.5 max_features =all min_samples_split=8	Teff	39.4	55.4	0.99
		Log g	0.031	0.044	0.89
		[Fe/H]	0.031	0.042	0.96

We also compared multi-output models with feature selection. The **MultioutputTargetSelectRegressor**, a new class implemented based on the **MultioutputRegressor** that allows different subsets of features to be provided to the internal regressor for each target variable. The results are shown on Table 5.4. Only SVR and Gradient Boosting were compared, since the predictive power of regularized linear regressions has been shown to decay with a decreasing number of features for the single-output setting.

Table 5.4: Evaluation metrics for different regression models to predict Teff (K), log g and [Fe/H] simultaneously, using the **MultioutputTargetSelectRegressor** regressor in the *hold-out* test set. Data has been Min-Max normalized, except the Lasso and ElasticNet estimators, for which no normalization yielded the best result. The target variables have also been Min-Max normalized.

Regression model	**Hyperparameters**	**Target variable**	**MAE**	**RMSE**	R^2
SVR	C=5	Teff	40.5	78.5	0.97
	epsilon=0.01	Log g	0.029	0.040	0.90
	kernel:'rbf'	[Fe/H]	0.028	0.038	0.97
Gradient Boosting	n_estimators=500	Teff	39.8	55.1	0.99
	max_depth = 10				
	learning rate = 0.01	Log g	0.031	0.045	0.89
	subsample = 0.5				
	max_features =all	[Fe/H]	0.029	0.040	0.96
	min_samples_split=8				

In summary, the SVR and Gradient Boosting with feature selection using the **mutual information** method show very similar results between them for the three targets. However, the SVR shows a slight decay in performance when compared to the version using all features, especially for Teff and [Fe/H], whilst the Gradient Boosting algorithm maintains predictive power.

Overall, these results show that a multi-output model can be used to predict the three target variables without significantly degrading predictive power.

5.3.3 Predictive modelling with equivalent width uncertainties

As previously mentioned, analyzing and measuring properties of astronomical systems greatly differs from the usual quantities or categories used in other data science challenges. Humans cannot see or measure them directly, the only way to do it is via complex instrumentation and the analysis often lies in extracting parameters and derive quantities through theoretical physical models.

The target variables have some uncertainty associated as they frequently lie within a confidence range [93]. However, the equivalent widths (predictors), which are derived using the ARES software (see Section 4.1), also have an associated error intrinsic to the technique. The targets' errors could not be directly included in the models because, theoretically, the same exact set

of independent variables' values cannot produce a different result for the dependent variable(s). Therefore, only equivalent widths' associated errors have been included through the injection of a distribution of equivalent widths for each spectral line and each star using Monte Carlo random sampling. It is intuitive to reason that, as a consequence, this procedure could produce a sort of 'data augmentation' effect in the data. To access the impact of injecting more data, four different combinations of experiments were performed (Figure 4.4). The results are displayed on Table 5.5.

Table 5.5: Evaluation metrics (MAE) for the indicated multi-output models to predict each of the three targets variables: Teff, log g and [Fe/H] according to the experiments indicated in Figure 4.4 from Section 4.4.5. Data has been Min-Max normalized, including the target variables. The best conditions for each type of model were used: all features for the Ridge ans SVR and only the best 50 features selected by the **mutual information** for the Gradient Boosting.

Experiment	**Model**	**Parameters**	**Teff**	**log g**	**[Fe/H]**
A1	Ridge	alpha=0.5 solver='saga'	27.2	0.033	0.023
A2			43.8	0.052	0.036
B1			28.4	0.036	0.022
B2			34.1	0.034	0.025
A1	SVR	C = 5 epsilon = 0.01 kernel: 'rbf'	32.4	0.032	0.018
A2			46.7	0.041	0.032
B1			38.8	0.033	0.021
B2			39.6	0.032	0.022
A1	Gradient Boosting	n_estimators=500 max_depth = 10 learning rate = 0.01 subsample = 0.5 max_features =all min_samples_split=8	39.4	0.031	0.031
A2			52.8	0.054	0.044
B1			42.3	0.031	0.03
B2			45.8	0.034	0.037

The model A1, where uncertainties have not been introduced, is the control, and shows the best results. The model B2, where uncertainties have been injected both in the train and test sets, reproduces the desired effect in terms of evaluating how much the model performance deviates from the average for every star in the test set. Model B1 emulates a data augmentation procedure typically used in machine learning. Generally, performing data augmentation is known to have a positive effect in performance, however, since this procedure was not a true data augmentation, it as possible that it could have no effect or even worsen the performance. It is easy to see that model A2 would not perform so well as the train sample is not 'augmented', whereas the test set is. This increases the probability of the model not being trained well enough to handle the extra perturbation injected into the test set.

In summary, data augmentation by oversampling both the train and the test sets did not have a significant impact in predictive power, as the result remained approximately the same as with no augmentation. However, when oversampling is also performed on the test set to produce the plots shown in Figure 5.15 (model B2), there are some points where the mean of the predicted

values does not deviate significantly from the ground-truth, but the standard-deviations are quite large. It should be clarified that these large standard-deviations are slightly related to the random samplings performed to generate the additional 100 new points from each original observation, which ultimately depend on the errors associated with equivalent widths' calculation by the ARES software. Both SVR and Gradient Boosting seem to be less affected by this perturbation than the Ridge, which suffers from more instabilities, especially when estimating log g.

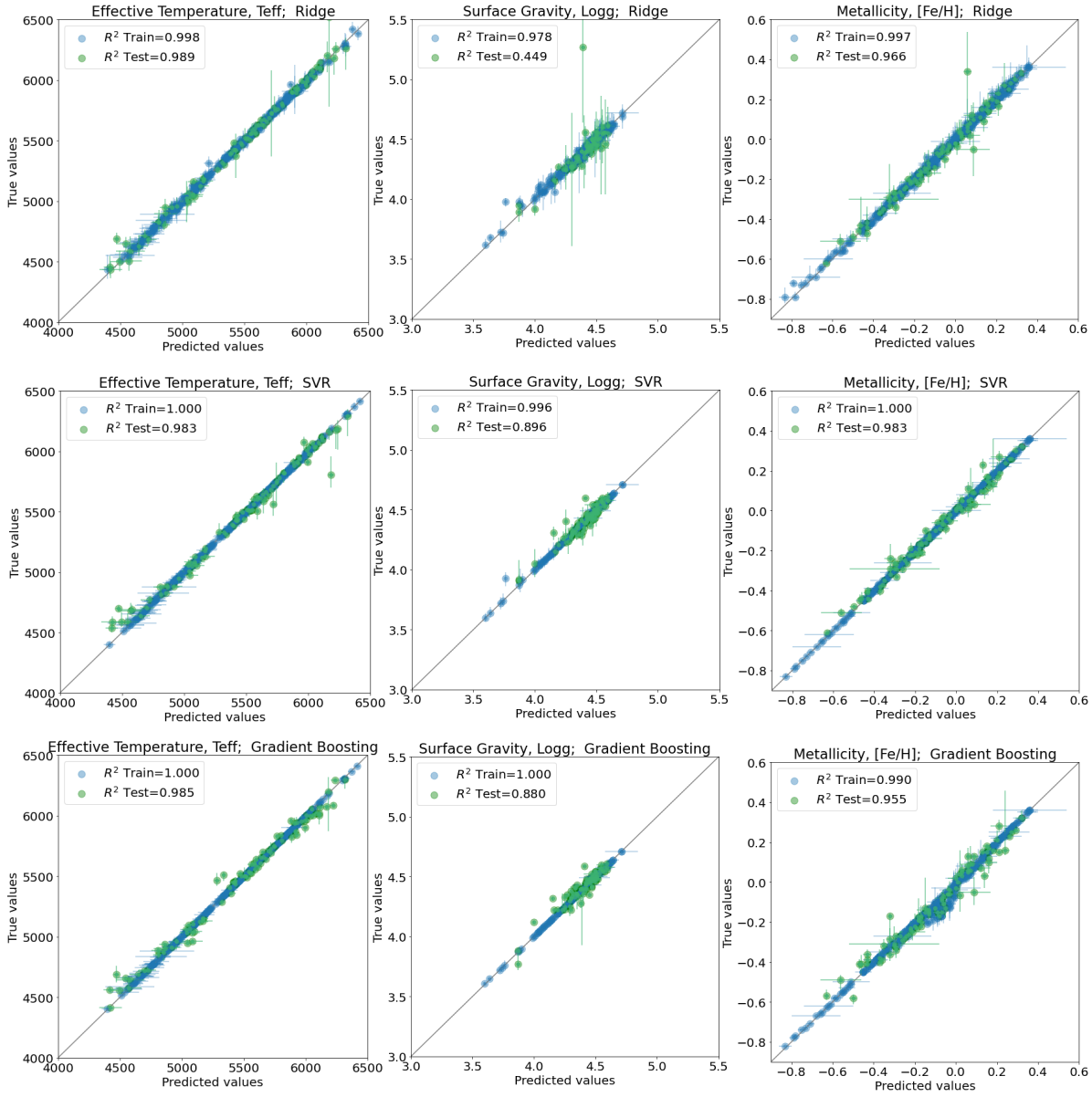


Figure 5.15: True *versus* predicted values of each target variable using the indicated multi-output models (Ridge, SVR and Gradient Boosting) with hyperparameters summarized in Table 5.5, for the model B2. Data has been Min-Max normalized, including the target variables. The data corresponds to the 'augmented' dataset, which is composed of 100 different values generated with a Monte-Carlo sampling technique for each of the stars. Each point on the graph represents the mean error for a star and the vertical error bars represent the standard-deviation from the gaussian distribution generated in turn of each observation of the original dataset. The horizontal error bars correspond to the error associated with the ground-truth values of the targets.

Chapter 6

Conclusions and Future Perspectives

6.1 Conclusions

The recent new releases of large-scale astronomical surveys are generating more data than ever, increasing the demand for new accurate and automatic methods to categorize these data and extract information without the need of persistent human intervention.

In this thesis, a preliminary machine-learning strategy to predict three important stellar atmospheric parameters - effective temperature, surface gravity and metallicity - based on the stars' spectra, is explored. Although there is already some literature that handles the same problem, there is still a constant need for refined results.

This method is built upon the ARES software, a previously developed software that automatically extracts the equivalent width of spectral lines from the spectra, which are then used as predictors in the machine-learning pipeline to estimate the three response variables.

An extensive and comprehensive study on the predictive behavior of different types of machine learning models, including simpler ones, such as linear regressions and variants (Ridge, Lasso and Elastic Net) or distance-based as kNN, passing through Kernel methods such as SVR and finally, tree-based algorithms, including decision trees and the respective bagging and boosting ensembles Random Forests and Gradient Boosting, respectively, is provided.

Both single and multi-output strategies were employed and the preference for one or for the other might depend on a tradeoff between error tolerance and efficiency. If the objective is to minimize the error as much as possible, then single-models might be justifiable, even though they required training and storing three different models, possibly taking longer times. This is not a big problem if the stellar sample is small, but once it starts increasing in size, the computational efficiency may start to degenerate rapidly. In a multi-output setting, since most of the models do not support this inherently, and therefore a wrapper solution must be used, the same hyperparamaters must be used for the three target variables, when in reality those are not always the best for each one of them, as our experiments demonstrate. Nevertheless,

more often than not, the errors for the best models are of the same order of magnitude and do not vary more than some units for the effective temperatures and hundredths of the unit for the surface gravity and the metallicity, so if those can be considered negligible, a multi-output strategy might be highly preferred.

In summary, a multi-output Ridge regression model trained on 359 examples with no prior feature selection provided estimates for the effective temperature Teff, the surface gravity log g and the metallicity [Fe/H] with MAE and RMSE of 27.1 and 40.6 K, 0.033 and 0.052 dex, and 0.023 and 0.035 dex, respectively, on a *hold-out* test set composed of 90 stars, initially sampled from the same population as the train set. All the stars in this sample belonged to the F-G-K classes according to the MK classification system, with effective temperatures Teff, surface gravities log g and metallicities [Fe/H] spanning the ranges 4400 - 6431 K, 3.6 - 4.82 dex and -0.83 - 0.36 dex, respectively.

However, our several feature selection experiments reveal, as expected, that algorithms such as Ridge and Lasso regression work extremely well on this data because of its high dimensional space and the existence of multicollinearity. Indeed, if feature selection based on Mutual Information theory (which is independent of any fitted model) is conducted before the predictive modelling stage, the performance of those models degrades easily, being outperformed by SVR and Gradient Boosting. SVR yielded MAEs and RMSEs of 38.8 and 61.1 K for the effective temperature, 0.033 and 0.051 dex for the surface gravity, and 0.027 and 0.039 dex for the metallicy. Similarly, Gradient Boosting, which shows an improvement in performance with feature selection, showed MAEs and RMSEs of 39.4 and 55.4 for the effective temperature, 0.031 and 0.044 dex for surface gravity, and 0.031 and 0.042 dex for the metallicity. Feature selection is also possible using a multi-output strategy. That required changing the source code of the **Multioutput Regressor** class from `sklearn`, which we slightly altered to accept a different set of features for each target as input.

Finally, experiments with feature selection show some results that would be interesting to investigate further from the astrophysics perspective. Calculating mutual information, it has been found that several spectral lines of TiI and VI are very informative to predict effective temperatures, while CI and ionized lines of several elements, mainly FeII, are informative to predict surface gravity and several metal neutral and ionized lines from SiI, NiI and FeII to estimate metallicity. Some of these findings are in line with astrophysics literature, especially the fact that ionized spectral lines are good predictors of surface gravity, but which ones is not completely known. Note that the wavelength at where these lines appear is also critical, and not mentioned specifically here for the sake of simplicity. Interestingly enough, tree-based algorithms such as Random Forest and Gradient Boosting select similar spectral lines to the mutual information method as being the most important, despite being obtained with a different method.

6.2 Future Perspectives

Despite the promising results, this study still suffers from countless weaknesses. First, a two-step approach is still needed to perform the analysis, which is not ideal. Equivalent widths must be initially extracted from the spectra with the ARES software so the final working dataset can be constructed. For obvious reasons, it means the proposed approach will not work if ARES or a similar software to calculate the equivalent widths is not previously employed. The second weakness stems directly from the first: because the methodology depends on ARES, it also suffers from the same limitations. For example, currently ARES cannot reliably extract equivalent widths for M-dwarfs' spectra due to blending effects and the existence of many spectral lines in close proximity that do not allow the software to precisely separate single lines, frequently merging them together. M dwarfs are, nonetheless, the most interesting stellar target from the exoplanet research perspective, as these are particularly suitable targets to detect low-mass rocky planets, essentially because these stars have lower masses compared to those of solar-like stars which facilitates the detection of lower mass planets and also because their low luminosity means the habitable zone is generally located closer than that of hotter and more massive stars.

Another major problem in this study is the dataset size. Our dataset consists of only 449 F-G-K stars that must be further split in a train and a test sets. From the machine learning perspective, this may constitute a problem in the ability of the models to generalize and some specialists may even argue that such a small dataset size does not justify the use of machine learning. This point is addressed in this thesis, and indeed it is observed that the cross-validation scores oscillate depending on the cross-validation fold, which means the models' predictions might be slightly sensitive to the specific samples constituting a validation/test set.

The limitations of this work propel us to its future directions. In terms of improving the work already developed, it would be interesting to increase the star sample's size, even if the data has lower S/N ratio or comes from different instruments. That will ultimately reveal the true robustness of this method. An alternative is to maintain data from the same instrument, but try to find a larger dataset to access whether the results can be further improved.

Also, using the current developed methodology, we are already running experiments to estimate not only the three atmospheric parameters, but also 10 chemical abundances of specific elements.

The main goal for the future is probably related with the utter need to have a reliable method to estimate these parameters in M dwarfs. As mentioned above, a new method will have to be developed from scratch. This will probably require the use of convolutional neural networks (possibly 1D), as others have used in the literature. However, the idea here would be slightly different than what has been done in other works. Instead of being used for predictive modelling, these networks can be used for object detection and segmentation in images and signals, for example. In short, the first goal would be to precisely detect and segment the spectral lines, which may also be employed to pseudo-normalize the continuum. If this succeeds, the result of

the segmentation may be used to extract features from the spectral lines and fed onto a more interpretable machine learning model, such as the ones presented here, for example through a model stacking pipeline.

Ultimately, it would be interesting to attempt a semi-supervised learning strategy, in which parts of the data are labelled and others are unlabelled, instead of the supervised approaches being currently used. This is related with the huge amounts of data being generated by the new releases of the large sky surveys, which remain largely unlabelled and labelling them using standard approaches may take years. A semi-supervised learning approach may thus succeed at labelling these data much faster and accelerate new discoveries in astrophysics.

Bibliography

[1] Richard O. Gray, Christopher J. Corbally, Adam J. Burgasser, Margaret M. Hanson, J. Davy Kirkpatrick, and Nolan R. Walborn. *Stellar Spectral Classification*, volume 15. Princeton University Press, 2009. ISBN: 9780691125107.

[2] David F. Gray. *The observation and analysis of stellar photospheres*. Cambridge books online. Cambridge University Press, Cambridge, 3rd ed. edition, 2005. ISBN: ISBN: 9780521851862 (hbk.).

[3] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[4] D. H. Menzel. The history of astronomical spectroscopy II. Quantitative chemical analysis and the structure of the solar atmosphere. *Annals of the New York Academy of Sciences*, 198(1):235–244, January 1972. doi:10.1111/j.1749-6632.1972.tb12727.x.

[5] William Wilson Morgan, Philip Childs Keenan, and Edith Kellman. *An atlas of stellar spectra, with an outline of spectral classification*. 1943.

[6] T. C. Beers, Y. Lee, T. Sivarani, C. Allende Prieto, R. Wilhelm, P. Re Fiorentin, and C. Bailer-Jones. The SDSS-I Value Added Catalog of Stellar Parameters and the SEGUE Pipeline. In *IAU Joint Discussion*, volume 26 of *IAU Joint Discussion*, page 26, August 2006.

[7] Matthias Steinmetz, Tomaž Zwitter, Arnaud Siebert, Fred G Watson, Kenneth C Freeman, Ulisse Munari, Rachel Campbell, Mary Williams, George M Seabroke, Rosemary FG Wyse, et al. The radial velocity experiment (rave): first data release. *The Astronomical Journal*, 132(4):1645, 2006.

[8] C. Boeche, A. Siebert, M. Williams, R. S. de Jong, M. Steinmetz, J. P. Fulbright, G. R. Ruchti, O. Bienaymé, J. Bland-Hawthorn, R. Campbell, K. C. Freeman, B. K. Gibson, G. Gilmore, E. K. Grebel, A. Helmi, U. Munari, J. F. Navarro, Q. A. Parker, W. Reid, G. M. Seabroke, A. Siviero, F. G. Watson, R. F. G. Wyse, and T. Zwitter. The rave catalog of stellar elemental abundances: First data release. *The Astronomical Journal*, 142(6):193, nov 2011. doi:10.1088/0004-6256/142/6/193.

[9] Gang Zhao, Yongheng Zhao, Yaoquan Chu, Yipeng Jing, and Licai Deng. LAMOST Spectral Survey. 6 2012.

[10] Steven R. Majewski. APOGEE – SDSS-III's Other Milky Way Experiment. In *American Astronomical Society Meeting Abstracts #219*, volume 219 of *American Astronomical Society Meeting Abstracts*, page 205.06, January 2012.

[11] Steven R. Majewski et al. The apache point observatory galactic evolution experiment (apogee). *The Astronomical Journal*, 154(3):94, aug 2017. doi:10.3847/1538-3881/aa784d.

[12] G. Gilmore, S. Randich, M. Asplund, J. Binney, P. Bonifacio, J. Drew, S. Feltzing, A. Ferguson, R. Jeffries, G. Micela, I. Negueruela, T. Prusti, H. -W. Rix, A. Vallenari, E. Alfaro, C. Allende-Prieto, C. Babusiaux, T. Bensby, R. Blomme, A. Bragaglia, E. Flaccomio, P. François, M. Irwin, S. Koposov, A. Korn, A. Lanzafame, E. Pancino, E. Paunzen, M. Collins, K. Eriksson, G. Fiorentino, J. Gonzalez Hernandez, M. Groenewegen, C. Hansen, A. Helmi, R. de Jong, P. Jonker, A. Koch, F. van Leeuwen, B. Lemasle, H. Ludwig, S. Messina, M. Meyer, J. De Ridder, A. Robin, S. Sharma, M. Smith, M. Steffen, E. Tolstoy, and J. Vink. The gaia-eso public spectroscopic survey. *The Messenger*, 147:25–31, March 2012. ISSN: 0722-6691.

[13] S. Randich, G. Gilmore, and Gaia-ESO Consortium. The Gaia-ESO Large Public Spectroscopic Survey. *The Messenger*, 154:47–49, December 2013.

[14] Kenneth C. Freeman. The hermes project: Reconstructing galaxy formation. In David L. Block, Kenneth C. Freeman, and Ivânio Puerari, editors, *Galaxies and their Masks*, pages 319–326, New York, NY, 2010. Springer New York. ISBN: 978-1-4419-7317-7.

[15] G. M. De Silva, K. C. Freeman, J. Bland-Hawthorn, S. Martell, E. Wylie de Boer, M. Asplund, S. Keller, S. Sharma, D. B. Zucker, T. Zwitter, B. Anguiano, C. Bacigalupo, D. Bayliss, M. A. Beavis, M. Bergemann, S. Campbell, R. Cannon, D. Carollo, L. Casagrande, A. R. Casey, G. Da Costa, V. D'Orazi, A. Dotter, L. Duong, A. Heger, M. J. Ireland, P. R. Kafle, J. Kos, J. Lattanzio, G. F. Lewis, J. Lin, K. Lind, U. Munari, D. M. Nataf, S. O'Toole, Q. Parker, W. Reid, K. J. Schlesinger, A. Sheinis, J. D. Simpson, D. Stello, Y. S. Ting, G. Traven, F. Watson, R. Wittenmyer, D. Yong, and M. Žerjal. The GALAH survey: scientific motivation. , 449(3):2604–2617, May 2015. doi:10.1093/mnras/stv327.

[16] Re Fiorentin, P., Bailer-Jones, C. A. L., Lee, Y. S., Beers, T. C., Sivarani, T., Wilhelm, R., Allende Prieto, C., and Norris, J. E. Estimation of stellar atmospheric parameters from sdss/segue spectra. *A&A*, 467(3):1373–1387, 2007. doi:10.1051/0004-6361:20077334.

[17] CAL Bailer-Jones, R Andrae, B Arcay, T Astraatmadja, I Bellas-Velidis, A Berihuete, A Bijaoui, C Carrión, C Dafonte, Yassine Damerdji, et al. The gaia astrophysical parameters inference system (apsis)-pre-launch description. *Astronomy & Astrophysics*, 559:A74, 2013.

[18] Sz Meszaros, J Holtzman, AE García Pérez, C Allende Prieto, RP Schiavon, S Basu, D Bizyaev, WJ Chaplin, SD Chojnowski, K Cunha, et al. Calibrations of atmospheric

parameters obtained from the first year of sdss-iii apogee observations. *The Astronomical Journal*, 146(5):133, 2013.

[19] Chao Liu, Li-Cai Deng, Jeffrey L Carlin, Martin C Smith, Jing Li, Heidi Jo Newberg, Shuang Gao, Fan Yang, Xiang-Xiang Xue, Yan Xu, et al. The k giant stars from the lamost survey data. i. identification, metallicity, and distance. *The Astrophysical Journal*, 790(2): 110, 2014.

[20] Rodolfo Smiljanic, Andreas J Korn, Maria Bergemann, Antonio Frasca, Laura Magrini, Thomas Masseron, ELENA Pancino, Gregory Ruchti, I San Roman, L Sbordone, et al. The gaia-eso survey: The analysis of high-resolution uves spectra of fgk-type stars. *Astronomy & Astrophysics*, 570:A122, 2014.

[21] Carlos Allende Prieto, Ramón J. García López, David L. Lambert, and Bengt Gustafsson. A consistency test of spectroscopic gravities for late-type stars. *The Astrophysical Journal*, 527(2):879, dec 1999. doi:10.1086/308096.

[22] Natalie R. Hinkel, F.X. Timmes, Patrick A. Young, Michael D. Pagano, and Margaret C. Turnbull. Stellar abundances in the solar neighborhood: The hypatia catalog. *The Astronomical Journal*, 148(3):54, aug 2014. doi:10.1088/0004-6256/148/3/54.

[23] Jofré, P., Heiter, U., Soubiran, C., Blanco-Cuaresma, S., Worley, C. C., Pancino, E., Cantat-Gaudin, T., Magrini, L., Bergemann, M., González Hernández, J. I., Hill, V., Lardo, C., de Laverny, P., Lind, K., Masseron, T., Montes, D., Mucciarelli, A., Nordlander, T., Recio Blanco, A., Sobeck, J., Sordo, R., Sousa, S. G., Tabernero, H., Vallenari, A., and Van Eck, S. Gaia fgk benchmark stars: Metallicity. *A&A*, 564:A133, 2014. doi:10.1051/0004-6361/201322440.

[24] Sousa, S. G., Alapini, A., Israelian, G., and Santos, N. C. An effective temperature calibration for solar type stars using equivalent width ratios* - a fast and easy spectroscopic temperature estimation. *A&A*, 512:A13, 2010. doi:10.1051/0004-6361/200913388.

[25] A. Recio-Blanco, A. Bijaoui, and P. De Laverny. Automated derivation of stellar atmospheric parameters and chemical abundances: the MATISSE algorithm. *Monthly Notices of the Royal Astronomical Society*, 370(1):141–150, 07 2006. ISSN: 0035-8711. doi:10.1111/j.1365-2966.2006.10455.x.

[26] Celia R. Fierro-Santillán, Janos Zsargó, Jaime Klapp, Santiago A. Díaz-Azuara, Anabel Arrieta, Lorena Arias, and Leonardo Di G. Sigalotti. Fitspec: A new algorithm for the automated fit of synthetic stellar spectra for ob stars. *The Astrophysical Journal Supplement Series*, 236(2):38, jun 2018. doi:10.3847/1538-4365/aabd3a.

[27] Xiangru Li, Q. M. Jonathan Wu, Ali Luo, Yongheng Zhao, Yu Lu, Fang Zuo, Tan Yang, and Yongjun Wang. Sdss/segue spectral feature analysis for stellar atmospheric parameter estimation. *The Astrophysical Journal*, 790(2):105, jul 2014. doi:10.1088/0004-637X/790/2/105.

[28] Celia R. Fierro-Santillán, Jaime Klapp, Leonardo Di G. Sigalotti, Janos Zsargó, and Markus Hareter. Analysis of spectral lines in large databases of synthetic spectra for massive stars. *The Astronomical Journal*, 161(3):121, feb 2021. doi:10.3847/1538-3881/abd950.

[29] Kaushal Sharma, Harinder P Singh, Ranjan Gupta, Ajit Kembhavi, Kaustubh Vaghmare, Jianrong Shi, Yongheng Zhao, Jiannan Zhang, and Yue Wu. Stellar spectral interpolation using machine learning. *Monthly Notices of the Royal Astronomical Society*, 496(4):5002–5016, 06 2020. ISSN: 0035-8711. doi:10.1093/mnras/staa1809.

[30] A. Antoniadis-Karnavas, S. G. Sousa, E. Delgado-Mena, N. C. Santos, G. D. C. Teixeira, and V. Neves. ODUSSEAS: a machine learning tool to derive effective temperature and metallicity for m dwarf stars. *Astronomy &amp Astrophysics*, 636:A9, apr 2020. doi:10.1051/0004-6361/201937194.

[31] F. Anders, A. Khalatyan, A. B. A. Queiroz, S. Nepal, and C. Chiappini. Parameters for > 300 million gaia stars: Bayesian inference vs. machine learning, 2023.

[32] H. Teimoorinia. Spectral classification of galaxies at 0.5 z 1 in the cdfs: The artificial neural network approach. *The Astronomical Journal*, 144(6):172, nov 2012. doi:10.1088/0004-6256/144/6/172.

[33] Dafonte, C., Fustes, D., Manteiga, M., Garabato, D., Álvarez, M. A., Ulla, A., and Allende Prieto, C. On the estimation of stellar parameters with uncertainty prediction from generative artificial neural networks: application to gaia rvs simulated spectra. *A&A*, 594: A68, 2016. doi:10.1051/0004-6361/201527045.

[34] Wu Minglei, Pan Jingchang, Yi Zhenping, Kong Xiaoming, and Bu Yude. Atmospheric parameter measurement of low-s/n stellar spectra based on deep learning. *Optik*, 218: 165004, 2020. ISSN: 0030-4026. doi:https://doi.org/10.1016/j.ijleo.2020.165004.

[35] Xiang-Ru Li, Ru-Yang Pan, and Fu-Qing Duan. Parameterizing stellar spectra using deep neural networks. *Research in Astronomy and Astrophysics*, 17(4):036, mar 2017. doi:10.1088/1674-4527/17/4/36.

[36] Navarro, S. G., Corradi, R. L. M., and Mampaso, A. Automatic spectral classification of stellar spectra with low signal-to-noise ratio using artificial neural networks. *A&A*, 538: A76, 2012. doi:10.1051/0004-6361/201016422.

[37] Edgar Villavicencio-Arcadia, Silvana G. Navarro, and Luis J. Corra. Application of Artificial Neural Networks for the Automatic Stellar Spectral Classification. In Pascal Ballester, Jorge Ibsen, Mauricio Solar, and Keith Shortridge, editors, *Astronomical Data Analysis Software and Systems XXVII*, volume 522 of *Astronomical Society of the Pacific Conference Series*, page 401, April 2020.

[38] Kaushal Sharma, Ajit Kembhavi, Aniruddha Kembhavi, T Sivarani, Sheelu Abraham, and Kaustubh Vaghmare. Application of convolutional neural networks for stellar spectral

classification. *Monthly Notices of the Royal Astronomical Society*, 491(2):2280–2300, 11 2019. ISSN: 0035-8711. doi:10.1093/mnras/stz3100.

[39] Shawn Snider, Carlos Allende Prieto, Ted von Hippel, Timothy C. Beers, Christopher Sneden, Yuan Qu, and Silvia Rossi. Three-dimensional spectral classification of low-metallicity stars using artificial neural networks. *The Astrophysical Journal*, 562(1):528, nov 2001. doi:10.1086/323428.

[40] S Fabbro, K A Venn, T O'Briain, S Bialek, C L Kielty, F Jahandar, and S Monty. An application of deep learning in the analysis of stellar spectra. *Monthly Notices of the Royal Astronomical Society*, 475(3):2978–2993, 12 2017. ISSN: 0035-8711. doi:10.1093/mnras/stx3298.

[41] Bin Jiang, Donglai Wei, Jiazhen Liu, Shuting Wang, Liyun Cheng, Zihao Wang, and Meixia Qu. Automated classification of massive spectra based on enhanced multi-scale coded convolutional neural network. *Universe*, 6(4), 2020. ISSN: 2218-1997. doi:10.3390/universe6040060.

[42] Marwan Gebran, Kathleen Connick, Hikmat Farhat, Frédéric Paletou, and Ian Bentley. Deep learning application for stellar parameters determination: I-constraining the hyperparameters. *Open Astronomy*, 31(1):38–57, 2022. doi:doi:10.1515/astro-2022-0007 [visited 2023-09-02].

[43] S. Seager. *Exoplanets*. 2010.

[44] S. Duckett and B. Gilbert. *Foundations of Spectroscopy*. Oxford chemistry primers. Oxford University Press, 2000. ISBN: 9780198503354.

[45] Annamaneni Peraiah. *The equation of radiative transfer*. Cambridge University Press, 2001. doi:10.1017/CBO9781139164474.003.

[46] P. Atkins and L. Jones. *Chemical Principles*. W. H. Freeman, 2008. ISBN: 9781429209656.

[47] S.W. Stahler and F. Palla. *The Formation of Stars*. Wiley, 2005. ISBN: 9783527405596.

[48] Sousa, S. G., Santos, N. C., Israelian, G., Mayor, M., and Monteiro, M. J. P. F. G. A new code for automatic determination of equivalent widths: Automatic routine for line equivalent widths in stellar spectra (ares) ***. *A&A*, 469(2):783–791, 2007. doi:10.1051/0004-6361:20077288.

[49] L. Spitzer. *Physical Processes in the Interstellar Medium*. A Wiley-Interscience publication. Wiley, 1978. ISBN: 9780471022329.

[50] Barry Smalley. T(eff) and log g determinations. *Mem. Soc. Astron. Ital. Suppl.*, 8:130, 2005.

[51] Girardi, L., Bressan, A., Bertelli, G., and Chiosi, C. Evolutionary tracks and isochrones for low- and intermediate-mass stars: From 0.15 to 7 $m_{sun}, and from z = 0.0004 to 0.03$. *Astron. Astrophys. Suppl. Ser.*, 141(3):371–383, 2000. doi:10.1051/aas:2000126.

[52] Paula Jofré, Ulrike Heiter, and Caroline Soubiran. Accuracy and Precision of Industrial Stellar Abundances. , 57:571–616, August 2019. doi:10.1146/annurev-astro-091918-104509.

[53] David F. Gray. The determination of temperature from spectral lines. 1996.

[54] David F. Gray. Spectral line-depth ratios as temperature indicators for cool stars. *Publications of the Astronomical Society of the Pacific*, 106(706):1248, dec 1994. doi:10.1086/133502.

[55] David F. Gray and Kevin Brown. Line-depth ratios: Temperature indices for giant stars. *Publications of the Astronomical Society of the Pacific*, 113(784):723, jun 2001. doi:10.1086/320811.

[56] Noriyuki Matsunaga, Mingjie Jian, Daisuke Taniguchi, and Scarlet S Elgueta. Line-depth ratios as indicators of effective temperature and surface gravity. *Monthly Notices of the Royal Astronomical Society*, 506(1):1031–1044, 06 2021. ISSN: 0035-8711. doi:10.1093/mnras/stab1770.

[57] Giusa Cayrel and Roger Cayrel. A Detailed Analysis of the Spectrum of Epsilon Virginis. , 137:431, February 1963. doi:10.1086/147521.

[58] Gingerich, Owen. *Theory and observation of normal stellar atmospheres*. January 1969.

[59] G. Smith, B. Edvardsson, and U. Frisk. Non-resonance lines of neutral calcium in the spectra of the alpha Centauri binary system. , 165:126–134, September 1986.

[60] K. Fuhrmann, M. Pfeiffer, C. Frank, J. Reetz, and T. Gehren. The surface gravities of cool dwarf stars revisited. , 323:909–922, July 1997.

[61] H. Bruntt, M. Deleuil, M. Fridlund, R. Alonso, F. Bouchy, A. Hatzes, M. Mayor, C. Moutou, and D. Queloz. Improved stellar parameters of CoRoT-7. A star hosting two super Earths. , 519:A51, September 2010. doi:10.1051/0004-6361/201014143.

[62] David F. Gray and William C. Livingston. Monitoring the solar temperature: Spectroscopic temperature variations of the sun. *The Astrophysical Journal*, 474(2):802, jan 1997. doi:10.1086/303489.

[63] Melike Afşar, Zeynep Bozkurt, Gamze Böcek Topcu, Sergen Özdemir, Christopher Sneden, Gregory N. Mace, Daniel T. Jaffe, and Ricardo López-Valdivia. Effective temperature estimations from line depth ratios in the h- and k-band spectra of igrins. *The Astrophysical Journal*, 949(2):86, jun 2023. doi:10.3847/1538-4357/acc946.

[64] Peter B. Stetson and Elena Pancino. Daospec: An automatic code for measuring equivalent widths in high-resolution stellar spectra1. *Publications of the Astronomical Society of the Pacific*, 120(874):1332, nov 2008. doi:10.1086/596126.

[65] Cantat-Gaudin, Tristan, Donati, Paolo, Pancino, Elena, Bragaglia, Angela, Vallenari, Antonella, Friel, Eileen D., Sordo, Rosanna, Jacobson, Heather R., and Magrini, Laura.

Doop, an automated wrapper for daospec. *A&A*, 562:A10, 2014. doi:10.1051/0004-6361/201322533.

[66] Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. doi:10.1080/00401706.1970.10488634.

[67] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. ISSN: 00359246.

[68] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320, 2005.

[69] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3): 273–297, sep 1995. ISSN: 0885-6125. doi:10.1023/A:1022627411411.

[70] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995. ISBN: 0-387-94559-8.

[71] Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. 9, 1996.

[72] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282 vol.1, 1995. doi:10.1109/ICDAR.1995.598994.

[73] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[74] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232, 2001. doi:10.1214/aos/1013203451.

[75] Jerome H. Friedman. Stochastic gradient boosting. *Computational Statistics Data Analysis*, 38(4):367–378, 2002. ISSN: 0167-9473. Nonlinear Methods and Data Mining. doi:https://doi.org/10.1016/S0167-9473(01)00065-2.

[76] L. Breiman, J. Friedman, C.J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Taylor & Francis, 1984. ISBN: 9780412048418.

[77] D. H. P. Jones. Multivariate Analysis as a Tool in Spectral Classification. In Kerstin Loden, L. O. Loden, and Ulf Sinnerstad, editors, *Spectral Classification and Multicolour Photometry*, volume 24, page 37, January 1966.

[78] Derek H. P. Jones. *Spectrophotometric observations of cool stars*. PhD thesis, University of London, UK, January 1966.

[79] F. Thevenin and R. Foy. Determination of the atmospheric parameters of late-type stars from low resolution spectra. , 122:261–266, June 1983.

[80] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1999.

[81] Vincent M. Woolf and George Wallerstein. Calibrating m dwarf metallicities using molecular indices. *Publications of the Astronomical Society of the Pacific*, 118(840):218, jan 2006. doi:10.1086/498459.

[82] M. Ness, David W. Hogg, H. W. Rix, Anna. Y. Q. Ho, and G. Zasowski. The Cannon: A data-driven approach to Stellar Label Determination. , 808(1):16, July 2015. doi:10.1088/0004-637X/808/1/16.

[83] Christopher P. et al. Ahn. , 211(2):17, April 2014. doi:10.1088/0067-0049/211/2/17, [link].

[84] G. Zasowski, Jennifer A. Johnson, P. M. Frinchaboy, S. R. Majewski, D. L. Nidever, H. J. Rocha Pinto, L. Girardi, B. Andrews, S. D. Chojnowski, K. M. Cudworth, K. Jackson, J. Munn, M. F. Skrutskie, R. L. Beaton, C. H. Blake, K. Covey, R. Deshpande, C. Epstein, D. Fabbian, S. W. Fleming, D. A. Garcia Hernandez, A. Herrero, S. Mahadevan, Sz. Mészáros, M. Schultheis, K. Sellgren, R. Terrien, J. van Saders, C. Allende Prieto, D. Bizyaev, A. Burton, K. Cunha, L. N. da Costa, S. Hasselquist, F. Hearty, J. Holtzman, A. E. García Pérez, M. A. G. Maia, R. W. O'Connell, C. O'Donnell, M. Pinsonneault, B. X. Santiago, R. P. Schiavon, M. Shetrone, V. Smith, and J. C. Wilson. Target Selection for the Apache Point Observatory Galactic Evolution Experiment (APOGEE). , 146(4):81, October 2013. doi:10.1088/0004-6256/146/4/81.

[85] Andrew R. Casey, David W. Hogg, Melissa Ness, Hans-Walter Rix, Anna Q Y Ho, and Gerry Gilmore. The Cannon 2: A data-driven model of stellar spectra for detailed chemical abundance analyses. *arXiv e-prints*, art. arXiv:1603.03040, March 2016. doi:10.48550/arXiv.1603.03040.

[86] R. Kou, P. Petit, F. Paletou, L. Kulenthirarajah, and J. M. Glorian. Deep learning determination of stellar atmospheric fundamental parameters. In P. Di Matteo, F. Billebaud, F. Herpin, N. Lagarde, J. B. Marquette, A. Robin, and O. Venot, editors, *SF2A-2018: Proceedings of the Annual meeting of the French Society of Astronomy and Astrophysics*, page Di, December 2018.

[87] M. Manteiga, I. Carricajo, A. Rodríguez, C. Dafonte, and B. Arcay. Starmind: A Fuzzy Logic Knowledge-Based System for the Automated Classification of Stars in the MK System. , 137(2):3245–3253, February 2009. doi:10.1088/0004-6256/137/2/3245.

[88] R. O. Gray and C. J. Corbally. An expert computer program for classifying stars on the mk spectral classification system. *The Astronomical Journal*, 147(4):80, mar 2014. doi:10.1088/0004-6256/147/4/80.

[89] Tan Xin, Pan Jing-chang, Wang Jie, Luo A-li, and Tu Liang-ping. Stellar spectrum parameter measurement based on line index by linear regression. *SPECTROSCOPY AND SPECTRAL ANALYSIS*, 33(05):1397, 2013. doi:10.3964/j.issn.1000-0593(2013)05-1397-04.

[90] M. Manteiga, D. Ordóñez, C. Dafonte, and B. Arcay. Anns and wavelets: A strategy for gaia rvs low s/n stellar spectra parameterization. *Publications of the Astronomical Society of the Pacific*, 122(891):608–617, 2010. ISSN: 00046280, 15383873.

[91] Róza´nski, T., Niemczura, E., Lemiesz, J., Posilek, N., and Róza´nski, P. Suppnet: Neural network for stellar spectrum normalisation. *A&A*, 659:A199, 2022. doi:10.1051/0004-6361/202141480.

[92] S. G. Sousa, N. C. Santos, M. Mayor, S. Udry, L. Casagrande, G. Israelian, F. Pepe, D. Queloz, and M. J. P. F. G. Monteiro. Spectroscopic parameters for 451 stars in the HARPS GTO planet search program. Stellar [Fe/H] and the frequency of exo-Neptunes. , 487(1):373–381, August 2008. doi:10.1051/0004-6361:200809698.

[93] Tsantaki, M., Sousa, S. G., Adibekyan, V. Zh., Santos, N. C., Mortier, A., and Israelian, G. Deriving precise parameters for cool solar-type stars - optimizing the iron line list. *A&A*, 555:A150, 2013. doi:10.1051/0004-6361/201321103.

[94] S. Udry, M. Mayor, D. Naef, F. Pepe, D. Queloz, N. C. Santos, M. Burnet, B. Confino, and C. Melo. The CORALIE survey for southern extra-solar planets. II. The short-period planetary companions to <ASTROBJ>HD 75289</ASTROBJ> and <ASTROBJ>HD 130322</ASTROBJ>. , 356:590–598, April 2000.

[95] M. Mayor, F. Pepe, D. Queloz, F. Bouchy, G. Rupprecht, G. Lo Curto, G. Avila, W. Benz, J. L. Bertaux, X. Bonfils, Th. Dall, H. Dekker, B. Delabre, W. Eckert, M. Fleury, A. Gilliotte, D. Gojak, J. C. Guzman, D. Kohler, J. L. Lizon, A. Longinotti, C. Lovis, D. Megevand, L. Pasquini, J. Reyes, J. P. Sivan, D. Sosnowska, R. Soto, S. Udry, A. van Kesteren, L. Weber, and U. Weilenmann. Setting New Standards with HARPS. *The Messenger*, 114: 20–24, December 2003.

[96] Alessio Mucciarelli, Elena Pancino, Loredana Lovisi, Francesco R. Ferraro, and Emilio Lapenna. Gala: An automatic tool for the abundance analysis of stellar spectra*. *The Astrophysical Journal*, 766(2):78, mar 2013. doi:10.1088/0004-637X/766/2/78.

[97] Anna Brucalassi, Maria Tsantaki, Laura Magrini, Sergio Sousa, Camilla Danielski, Katia Biazzo, Giada Casali, Mathieu Van der Swaelmen, Monica Rainer, Vardan Adibekyan, et al. Determination of stellar parameters for ariel targets: a comparison analysis between different spectroscopic methods. *Experimental Astronomy*, pages 1–22, 2021.

[98] Sousa, S. G., Adibekyan, V., Delgado-Mena, E., Santos, N. C., Rojas-Ayala, B., Soares, B. M. T. B., Legoinha, H., Ulmer-Moll, S., Camacho, J. D., Barros, S. C. C., Demangeon, O. D. S., Hoyer, S., Israelian, G., Mortier, A., Tsantaki, M., and Monteiro, M. A. Sweet-cat 2.0: The cat just got sweeter - higher quality spectra and precise parallaxes from gaia edr3. *A&A*, 656:A53, 2021. doi:10.1051/0004-6361/202141584.

[99] *Entropy, Relative Entropy, and Mutual Information*, chapter 2, pages 13–55. John Wiley Sons, Ltd, 2005. ISBN: 9780471748823. doi:https://doi.org/10.1002/047174882X.ch2.

[100] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Phys. Rev. E*, 69:066138, Jun 2004. doi:10.1103/PhysRevE.69.066138.

[101] Brian C. Ross. Mutual information between discrete and continuous data sets. *PLOS ONE*, 9:1–5, 02 2014. doi:10.1371/journal.pone.0087357.

[102] Lyudmyla F Kozachenko and Nikolai N Leonenko. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii*, 23(2):9–16, 1987.

[103] Mark Hall. Correlation-based feature selection for machine learning. *Department of Computer Science*, 19, 06 2000.

[104] Paul R. Cohen and David Jensen. Overfitting explained. In David Madigan and Padhraic Smyth, editors, *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*, volume R1 of *Proceedings of Machine Learning Research*, pages 115–122. PMLR, 04–07 Jan 1997.

[105] Peter Flach. *Machine Learning: The Art and Science of Algorithms That Make Sense of Data*. Cambridge University Press, USA, 2012. ISBN: 1107422221.

[106] Luís Torgo, Rita P Ribeiro, Bernhard Pfahringer, and Paula Branco. Smote for regression. In *Portuguese conference on artificial intelligence*, pages 378–389. Springer, 2013.

[107] Louis Capitaine, Robin Genuer, and Rodolphe Thiébaut. Random forests for high-dimensional longitudinal data, 2019.

Appendix A

Physical principles underlying the formation of stellar spectra

A.1 The atom and energetic electronic transitions

Otherwise stated, this section follows explanations found in [46] and [44].

As proposed by Niels Bohr in the early 20th century for the hydrogen atom, several quantized energy levels can be devised for its only electron. These discrete states, represented by energy level diagrams, such as the one in Figure A.1 are identified by positive integers n (with $n = 1$ being the ground state) and described as perfectly defined orbits around the nucleus.

Although this concept was later replaced by that of the atomic orbitals from quantum mechanics, the energetic states are still used to this day to describe electronic transitions. As such, the electron can jump between these levels whenever there is an absorption or emission of electromagnetic radiation, a photon, with each transition corresponding to a certain wavelength. This property, and the fact that electrons can only take discrete amounts of energy to move between states, allows for each element to have its particular absorption and emission spectrum, which can then be used to determine the composition of a star. The energy required for the electron to transition from $n = 1$ to $n = \infty$ is called ionization energy.

Electrons can move to excited states. Emissions occur when an electron drops from an excited state to a lower energetic level, emitting a photon with a specific amount of energy equal to the difference in energy between the two states. These transitions can be described by spectral series, such as the Lyman, Balmer and Paschen, according to the final state of the electron. Each is associated with emission in a certain wavelength: for instance, the Lyman series corresponds to a drop to $n = 1$ from a higher level, and an emission of ultraviolet radiation. Emission spectra show the collection of transitions and corresponding wavelengths allowed for the electrons in an atom. These are specific for each element of the periodic table, and as such can be used to determine stars' composition through astronomical spectroscopy. Similarly, an absorption spectrum is also

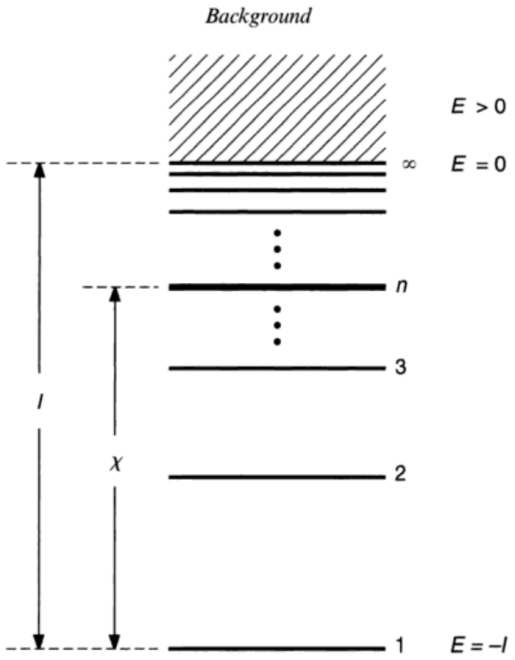


Figure A.1: One-dimensional schematic of an energy-level diagram showing the excitation potential $\mathcal{X}$ for an arbitrary energy level n and the corresponding ionization potential I. $n = 1$ is the ground level and subsequent energy levels are number upwards. The energy E of the electron is negative for all bound states (see bottom right), positive in the continuum (top shaded area) and zero at the ionization limit $n = \infty$. *Adapted from:* [2]

a fingerprint of the atom, with black lines corresponding to the fraction of absorbed radiation as photons by the element and to the difference in energy between two quantum states.

A.2 The stellar photosphere

It should be understood that the stellar spectrum is formed only in the surface layers of a star - the stellar photosphere. In cool stars the chromosphere and the corona can also contribute to the emergent spectrum, especially in the far and extreme ultraviolet. The stellar photosphere plays such an important role in the formation of the spectrum because it is there that photons undergo their last few, and thus most critical, interactions with matter before escaping to free space. Energy, in the form of photons, which is produced in the core of the star by nuclear reactions, reaches the surface only after several interactions with the material in the interior of the star, which is composed of a nearly completely ionized plasma (a gas consisting of electrons and ions). This plasma interacts with the radiation field in the interior through the physical processes of electron scattering and free–free and bound–free (photoionization) absorption and emission by ions. Free–free absorption occurs when a free electron becomes able to absorb a photon when in the vicinity of an ion.

Thus a gamma-ray photon produced in the core of the star random walks its way to the surface and, in the process, is degraded into hundreds of lower energy photons (mostly in the UV, optical,

and infrared parts of the spectrum). In the interior, because of the high densities, the mean free path between interactions is very short—on the order of a centimeter in main-sequence stars. The high densities also mean that in the interior, collisions are extremely effective in coupling the radiation field with the thermal state of the gas. In this condition, the interior is very nearly in a perfect state of thermodynamic equilibrium, and, as a consequence, the material in the interior of the star radiates at the local temperature as a black-body. As we move toward the surface and encounter lower densities, the mean free path of a typical photon becomes much longer. In the stellar photosphere, the mean free path may measure in the kilometers; eventually a level in the stellar photosphere is reached at which the photon will likely escape without further interaction. Since at each interaction the energy (wavelength) of the photon may be altered, it is clear that the last few interactions are most important in the formation of the emergent spectrum [1], [2].

Appendix B

Theoretical foundations on Machine Learning Algorithms

Theorem B.0.1 ***Singular matrix***
A square matrix that does not have a matrix inverse. A matrix is singular iff its determinant is 0.

B.1 Ridge Regression and Multicollinearity

Let's consider a matrix X of shape $n \times p$. Its columns $X_1, ..., X_p \in \mathbb{R}^n$ are said to be linearly independent when $\sum \alpha_i X_i = 0$ iff $\alpha_i = 0$ for $i = 1, ..., p$. Intuitively, none of the columns in X can be written as a weighted sum of the others. The other way around, if it's not the case for some columns, they are called linearly dependent. Assume $rank(X) = r$, then $p - r$ columns of X are linearly dependent.

Multicollinearity occurs when a predictor variable in a multiple regression model can be linearly predicted from the others, or, in other words, is approximately a linear combination of the others. An hypothetically perfect multicollinearity indicates linear dependency in the feature matrix. Intuitively, it implies redundancy in the predictors and, as a consequence, some of them fail to provide unique and/or independent information to the regression, because, as mentioned above, the assumption of independence is not satisfied.

For that reason, the presence of multicollinearity is an important aspect both theoretically and in practice, as it can compromise a model's performance due to the coefficient estimates changing erratically in response to small changes in the model or the data.

The demonstration of the problem of multicollinearity in the realm of OLS and how ridge regularization can solve uses SVD, which is recalled below.

Theorem B.1.1 *For a matrix X, there always exist matrices U,Σ and V, such that:*

$$X = U_{nxn}\Sigma_{nxp}V^T_{pxp}, \tag{B.1}$$

where both U and V are orthogonal: $U^TU = UU^T = I$ and $V^TV = VV^T = I$.

OLS with Multicollinearity Recalling that for the feature matrix X and the target variable y, least squares attempts to approximate the solution of the linear system $y = X\beta$ by minimizing the sum of squares of the residuals $||y - X\beta||^2$. The weights vector can be written as

$$\hat{\beta}^{OLS} = (X^TX)^{-1}X^Ty \tag{B.2}$$

Note that X^TX is invertible if and only if $n \geq p$ and $rank(X) = p$. Now it becomes easier to see why perfect multicollinearity is a major problem for least squares: it implies that the feature matrix is not full-rank, thus one cannot find a proper set of coefficients that minimize the sum of squared residuals.

But why is multicollinearity a problem, even if it is only partial?

We know that $y = X\beta + \epsilon$, where ϵ is some negligible error. The equation B.1 above can be written as

$$\hat{\beta}^{OLS} = (X^TX)^{-1}X^Ty = (X^TX)^{-1}X^T(X\beta + \epsilon) = \beta + (X^TX)^{-1}X^T\epsilon. \tag{B.3}$$

One can see the least squares coefficients deviate from the true weights by ϵ multiplied by some inflation term.

If all the columns of X are linearly independent, we still have p singular values and $d_1 \geq d_2 \geq ... \geq d_p > 0$. However, with the presence of multicollinearity, some of those values will be close to zero. Then the diagonal element $1/d_p$ will be huge, leading to a really large inflation term and therefore a great deviation in the least squares weights from the true weights. Intuitively, multicollinearity can compromise least squares as it leads to small singular values. The estimation errors of the coefficients are inflated by the reciprocals of those singular values and therefore become too large to be neglected. Ridge regression solves this problem.

Solving multicollinearity with Ridge regression Ridge regression capitalizes on least squares by adding a regularization term in the cost function. using the same notation as above, that is $||y - X\beta||^2 + \lambda||\beta||^2$, where λ is the strength of regularization. We can write the cost function $f(\beta)$ as

$$y(\beta) = y^T y - 2\beta^T X^T y + \beta^T X^T X\beta + \lambda\beta^T\beta. \tag{B.4}$$

Computing the gradient and setting it to zero,

$$\nabla^\beta f = 0 - 2X^T y + 2X^T X\beta + 2\lambda\beta = 0, \tag{B.5}$$

we get the following solution

$$\hat{\beta}^{ridge} = (X^T X + \lambda I)^{-1} X^T y = V(\Sigma). \tag{B.6}$$

$$(\Sigma^T\Sigma + \lambda I)^{-1}\Sigma^T = \begin{bmatrix} \frac{d_1}{d_1^2+\lambda} & & & & \cdots \\ & \frac{d_2}{d_2^2+\lambda} & & & \cdots \\ & & \ddots & & \cdots \\ & & & \frac{d_p}{d_p^p+\lambda} & \cdots \end{bmatrix}$$

Consider $d_p \approx 0$, this time with $d_p/(d_p^2 + \lambda) \approx 0$ iff $\lambda \neq 0$. Therefore, the coefficients of unimportant features will be close to zero (but not exactly zero unless there is perfect multicollinearity) and the error term would not increase indefinitely to explosion. This explains why ridge regularizarion solves the problem of multicollinearity. Note that when there is no regularization, $\lambda = 0$, this resumes to OLS. Also, since in the majority of cases we have $d_p >> \lambda$, $d_p/(d_p^2 + \lambda) \approx 1/p$, just like in OLS.

Appendix C

Data exploratory analysis

C.1 Best features selected by the Best Subset method

Table C.1: Best features selected by the greedy method **Best Subset**.

Target	**Selected features**
Teff	TiI-4562.63
	FeII-6247.56
	FeI-5247.06
	FeII-5534.85
	FeI-5956.70
	FeII-5234.63
	TiI-4820.41
	CI-5052.15
	TiI-5145.47
log g	TiII-4708.67
	TiII-4583.41
	FeI-5068.77
	TiII-4636.33
	FeI-5229.86
	ScII-6245.62
[Fe_H]	SiI-5753.64
	YII-5087.43
	FeI-4952.65
	CrII-4592.05
	FeI-5638.27
	FeII-5425.25
	FeI-6810.27
	FeII-4534.17
	FeI-6842.69
	FeII-5256.94

C.2 Correlations between predictor variables

Table C.2: Pearson's correlation coefficients between predictors of the HARPS dataset sorted in decreasing order. Only the 500 highest correlations are displayed.

Var1	Var2	correlation
VI-6135.36	VI-6111.65	1.0
VI-6081.45	VI-5670.85	1.0
VI-6224.51	VI-6111.65	1.0
VI-6135.36	VI-6081.45	1.0
VI-6251.83	TiI-4562.63	1.0
VI-6251.83	VI-5670.85	1.0
VI-6251.83	VI-6081.45	0.99
VI-6251.83	VI-6135.36	0.99
TiI-6126.22	TiI-4645.19	0.99
VI-6224.51	VI-6135.36	0.99
VI-6111.65	VI-5737.07	0.99
VI-6111.65	ScI-5671.82	0.99
VI-6135.36	ScI-5671.82	0.99
VI-6135.36	VI-5670.85	0.99
VI-6135.36	VI-5737.07	0.99
VI-6111.65	ScI-4743.82	0.99
VI-6251.83	VI-6111.65	0.99
VI-6285.17	VI-6135.36	0.99
VI-6251.83	TiI-6599.12	0.99
TiI-6126.22	TiI-4562.63	0.99
VI-6285.17	VI-6251.83	0.99
VI-6224.51	VI-5737.07	0.99
VI-6251.83	VI-6039.73	0.99
ScI-5671.82	ScI-4743.82	0.99
VI-5737.07	ScI-5671.82	0.99
FeI-4596.41	FeI-4593.53	0.99
VI-5670.85	TiI-4562.63	0.99
VI-6135.36	TiI-6064.63	0.99
VI-6224.51	ScI-4743.82	0.99
VI-6111.65	TiI-6064.63	0.99
VI-6111.65	VI-6081.45	0.99
VI-6081.45	VI-6039.73	0.99

to be continued ...

Var1	Var2	correlation
VI-6285.17	VI-6081.45	0.99
FeI-4596.41	FeI-4551.65	0.99
VI-6135.36	VI-6039.73	0.99
VI-6285.17	VI-5670.85	0.99
TiI-4645.19	TiI-4562.63	0.99
VI-6039.73	VI-5670.85	0.99
VI-6285.17	VI-6224.51	0.99
VI-6285.17	VI-6111.65	0.99
VI-6111.65	TiI-5648.57	0.99
VI-6224.51	TiI-6064.63	0.99
TiI-6126.22	TiI-5662.16	0.99
VI-6135.36	TiI-5648.57	0.99
FeI-4661.54	FeI-4596.41	0.99
VI-6251.83	TiI-6126.22	0.99
VI-6251.83	TiI-6064.63	0.99
VI-5670.85	TiI-6126.22	0.99
VI-6224.51	ScI-5671.82	0.99
VI-6081.45	TiI-4562.63	0.99
VI-6081.45	TiI-6064.63	0.99
VI-6081.45	VI-5737.07	0.99
TiI-5662.16	TiI-4645.19	0.99
VI-6285.17	TiI-6064.63	0.99
FeI-6857.25	FeI-4593.53	0.99
TiI-5648.57	ScI-5671.82	0.99
VI-6111.65	VI-6039.73	0.99
VI-6285.17	TiI-4562.63	0.99
VI-6135.36	TiI-6599.12	0.99
FeI-6358.68	FeI-6082.72	0.99
FeI-4596.41	FeI-4566.52	0.99
TiI-6599.12	TiI-4562.63	0.99
FeI-4593.53	FeI-4551.65	0.99
VI-6081.45	ScI-5671.82	0.99
VI-6111.65	VI-5670.85	0.99
VI-6111.65	TiI-6599.12	0.99
VI-5670.85	TiI-5978.55	0.99
VI-6081.45	TiI-6599.12	0.99
VI-6135.36	ScI-4743.82	0.99
VI-6251.83	VI-6224.51	0.99
FeI-6608.03	FeI-6392.54	0.99

to be continued ...

Var1	Var2	correlation
VI-6251.83	TiI-5071.49	0.99
VI-6216.35	VI-6081.45	0.99
VI-6081.45	TiI-5648.57	0.99
VI-5670.85	TiI-6064.63	0.99
TiI-5978.55	TiI-4645.19	0.99
VI-6216.35	TiI-5662.16	0.99
FeI-5225.53	FeI-5247.06	0.99
VI-6251.83	VI-6216.35	0.99
TiI-6126.22	TiI-5978.55	0.99
VI-6251.83	TiI-5648.57	0.99
FeII-5264.81	FeII-5234.63	0.99
VI-5670.85	TiI-4645.19	0.99
VI-6285.17	VI-5737.07	0.99
VI-6081.45	TiI-6126.22	0.99
VI-6081.45	TiI-5662.16	0.99
VI-6081.45	TiI-5978.55	0.99
VI-5737.07	TiI-6064.63	0.99
VI-6251.83	VI-5737.07	0.99
MnI-5413.67	MnI-5399.47	0.99
VI-6251.83	TiI-4645.19	0.99
VI-6251.83	TiI-5662.16	0.99
TiI-6064.63	ScI-5671.82	0.99
FeI-4661.54	FeI-4551.65	0.99
VI-6216.35	VI-5670.85	0.99
VI-6039.73	TiI-5662.16	0.99
TiI-6064.63	TiI-5648.57	0.99
VI-6285.17	VI-6039.73	0.99
VI-5670.85	TiI-5662.16	0.99
VI-6039.73	TiI-6126.22	0.99
VI-6224.51	VI-6081.45	0.99
VI-6135.36	TiI-4562.63	0.99
VI-5737.07	ScI-4743.82	0.99
VI-6039.73	TiI-4562.63	0.99
CrI-4730.72	FeI-6358.68	0.99
VI-6216.35	VI-6039.73	0.99
VI-5737.07	TiI-5648.57	0.99
TiI-6064.63	TiI-4562.63	0.99
VI-6039.73	TiI-5648.57	0.99
FeI-4661.54	FeI-4593.53	0.99

to be continued ...

Var1	Var2	correlation
VI-5670.85	TiI-6599.12	0.99
TiI-6599.12	TiI-5648.57	0.99
VI-6285.17	TiI-6599.12	0.99
TiI-6126.22	TiI-5071.49	0.99
TiI-5648.57	ScI-4743.82	0.99
VI-6039.73	TiI-6599.12	0.99
TiI-6599.12	TiI-6064.63	0.99
TiI-6064.63	ScI-4743.82	0.99
VI-6081.45	TiI-4645.19	0.99
CrI-5247.57	CrI-4626.18	0.99
TiI-5071.49	TiI-4562.63	0.99
VI-5670.85	TiI-5071.49	0.99
VI-6216.35	TiI-5071.49	0.99
VI-6224.51	TiI-6599.12	0.99
VI-6039.73	TiI-6064.63	0.99
FeI-4554.46	FeI-5247.06	0.99
FeI-6857.25	FeI-4596.41	0.99
VI-5737.07	TiI-6599.12	0.99
FeII-6432.69	FeII-6149.25	0.99
VI-6216.35	VI-6135.36	0.99
VI-5737.07	VI-5670.85	0.99
TiI-5978.55	TiI-5662.16	0.99
FeI-6739.52	FeI-6625.02	0.99
VI-6216.35	TiI-6126.22	0.99
ScI-5671.82	CaI-6156.02	0.99
VI-6039.73	ScI-5671.82	0.98
TiI-5662.16	TiI-5648.57	0.98
VI-5670.85	TiI-5648.57	0.98
VI-6090.21	TiI-4645.19	0.98
CrI-5781.18	CrI-5214.14	0.98
TiI-5662.16	TiI-4562.63	0.98
VI-6224.51	TiI-5648.57	0.98
VI-6039.73	VI-5737.07	0.98
TiI-6064.63	TiI-5071.49	0.98
VI-6039.73	TiI-4645.19	0.98
FeI-5814.81	FeI-5376.83	0.98
VI-6090.21	TiI-6126.22	0.98
VI-6081.45	TiI-5071.49	0.98
FeII-6432.69	FeII-5414.07	0.98

to be continued ...

Var1	Var2	correlation
VI-5670.85	ScI-5671.82	0.98
VI-6216.35	FeI-5143.73	0.98
FeI-5815.22	FeI-5376.83	0.98
TiI-5662.16	TiI-5071.49	0.98
VI-6285.17	TiI-5071.49	0.98
FeI-4661.54	FeI-4566.52	0.98
VI-5703.59	TiI-5978.55	0.98
VI-6216.35	TiI-5648.57	0.98
VI-6285.17	TiI-5648.57	0.98
FeI-6857.25	FeI-4551.65	0.98
VI-6251.83	ScI-5671.82	0.98
MnI-5413.67	FeI-5738.24	0.98
CrI-4700.61	FeI-4554.46	0.98
VI-6224.51	VI-5670.85	0.98
VI-6039.73	TiI-5071.49	0.98
FeI-6392.54	FeI-5778.46	0.98
FeI-6739.52	FeI-6710.32	0.98
VI-5703.59	TiI-4645.19	0.98
FeI-6739.52	FeI-6120.25	0.98
VI-6135.36	TiI-5071.49	0.98
VI-6285.17	ScI-5671.82	0.98
TiI-6126.22	TiI-5866.45	0.98
VI-5703.59	TiI-6126.22	0.98
FeI-4566.52	FeI-4551.65	0.98
VI-6111.65	TiI-4562.63	0.98
FeI-5814.81	FeI-5436.30	0.98
TiI-5866.45	TiI-4645.19	0.98
VI-6216.35	TiI-4645.19	0.98
VI-6224.51	VI-6039.73	0.98
VI-6216.35	VI-6111.65	0.98
TiI-5978.55	TiI-4562.63	0.98
VI-6111.65	TiI-5071.49	0.98
TiI-5071.49	TiI-4645.19	0.98
TiI-6599.12	TiI-5662.16	0.98
NiI-5748.36	NiI-5847.00	0.98
FeI-6358.68	FeI-6240.65	0.98
FeI-6857.25	FeI-4661.54	0.98
FeI-6498.94	FeI-5247.06	0.98
VI-6135.36	TiI-5662.16	0.98

to be continued ...

Var1	Var2	correlation
FeII-6238.39	FeII-5414.07	0.98
CrI-4730.72	FeI-6082.72	0.98
VI-6081.45	TiI-5739.48	0.98
TiI-6599.12	TiI-6126.22	0.98
FeII-6247.56	FeII-5264.81	0.98
VI-6285.17	TiI-6126.22	0.98
VI-6090.21	TiI-5662.16	0.98
VI-6251.83	TiI-5978.55	0.98
CrI-4700.61	VI-5703.59	0.98
FeII-6247.56	FeII-5234.63	0.98
TiI-6599.12	TiI-5071.49	0.98
VI-6135.36	TiI-6126.22	0.98
FeI-4593.53	FeI-4566.52	0.98
VI-6216.35	ScI-5671.82	0.98
VI-5703.59	TiI-5662.16	0.98
VI-6090.21	TiI-5978.55	0.98
FeI-5956.70	FeI-5247.06	0.98
NiI-6086.29	NiI-5010.94	0.98
FeI-6082.72	FeI-5778.46	0.98
VI-6090.21	VI-5703.59	0.98
VI-6251.83	ScI-4743.82	0.98
FeI-6857.25	FeI-6733.15	0.98
NiI-5996.73	NiI-5643.08	0.98
FeI-5636.70	FeI-4596.41	0.98
FeI-6220.79	FeI-4809.94	0.98
TiI-6126.22	TiI-6064.63	0.98
VI-6216.35	TiI-4562.63	0.98
TiI-6599.12	TiI-4645.19	0.98
VI-6285.17	ScI-4743.82	0.98
TiI-5648.57	TiI-5071.49	0.98
TiI-6599.12	ScI-5671.82	0.98
VI-5703.59	FeI-4554.46	0.98
VI-6216.35	TiI-6599.12	0.98
MnI-5399.47	FeI-6220.79	0.98
TiI-5648.57	TiI-4562.63	0.98
VI-6216.35	TiI-5978.55	0.98
VI-5670.85	FeI-5143.73	0.98
FeI-6710.32	FeI-6625.02	0.98
CrI-4730.72	FeI-4809.94	0.98

to be continued ...

Var1	Var2	correlation
FeI-5223.19	FeI-4809.94	0.98
FeII-5414.07	FeII-5264.81	0.98
FeI-4809.94	FeI-4537.67	0.98
VI-6224.51	TiI-4562.63	0.98
TiI-6064.63	TiI-5662.16	0.98
TiI-6126.22	TiI-5648.57	0.98
VI-6039.73	TiI-5978.55	0.98
FeI-6739.52	FeI-6392.54	0.98
VI-6224.51	TiI-5071.49	0.98
TiI-5219.70	TiI-4645.19	0.98
FeI-5636.70	FeI-4593.53	0.98
TiI-5071.49	FeI-5143.73	0.98
VI-6135.36	TiI-5739.48	0.98
FeI-6625.02	FeI-6120.25	0.98
TiI-5866.45	TiI-5662.16	0.98
FeI-6392.54	FeI-6358.68	0.98
TiI-5064.06	ScI-5671.82	0.98
CrI-5247.57	TiI-5071.49	0.98
FeI-6699.15	FeI-4631.49	0.98
TiI-6599.12	ScI-4743.82	0.98
TiI-5739.48	ScI-5671.82	0.98
FeII-6149.25	FeII-5414.07	0.98
VI-6111.65	TiI-5662.16	0.98
NiI-6130.14	NiI-6111.08	0.98
TiI-6126.22	TiI-5219.70	0.98
CrI-5247.57	VI-6216.35	0.98
TiI-6126.22	FeI-5143.73	0.98
FeI-6240.65	FeI-6082.72	0.98
VI-6216.35	VI-5737.07	0.98
VI-6090.21	VI-5670.85	0.98
VI-6039.73	ScI-4743.82	0.98
FeI-6392.54	FeI-6082.72	0.98
FeI-6713.74	FeI-6733.15	0.98
VI-5737.07	TiI-5071.49	0.98
FeI-6857.25	FeI-6725.36	0.98
TiI-5648.57	CaI-6156.02	0.98
VI-6081.45	ScI-4743.82	0.98
VI-6285.17	VI-6216.35	0.98
VI-6081.45	TiI-5219.70	0.98

to be continued . . .

Var1	Var2	correlation
VI-6285.17	TiI-5662.16	0.98
TiI-5978.55	TiI-5219.70	0.98
FeI-4593.53	FeI-6733.15	0.98
FeI-4799.41	FeI-4661.54	0.98
VI-5670.85	TiI-5219.70	0.98
VI-6285.17	TiI-4645.19	0.98
TiI-6064.63	TiI-4645.19	0.98
TiI-5866.45	TiI-5219.70	0.98
VI-6216.35	TiI-6064.63	0.98
TiI-5648.57	TiI-4645.19	0.98
VI-6135.36	TiI-4645.19	0.98
TiI-5662.16	FeI-5143.73	0.98
NiI-6186.72	NiI-6130.14	0.98
VI-6216.35	VI-5703.59	0.98
VI-5670.85	TiI-5866.45	0.98
VI-6081.45	FeI-5143.73	0.98
FeII-6432.69	FeII-6238.39	0.98
NiI-6130.14	NiI-6086.29	0.98
CrI-5783.07	CrI-5781.18	0.98
FeI-6608.03	FeI-5778.46	0.98
CrI-5247.57	TiI-5662.16	0.98
NiI-6128.98	NiI-5748.36	0.98
FeI-6358.68	FeI-5778.46	0.98
FeI-6725.36	FeI-4593.53	0.98
TiI-5978.55	TiI-5866.45	0.98
VI-6135.36	TiI-5978.55	0.98
TiI-5866.45	TiI-5071.49	0.98
VI-6274.66	VI-6224.51	0.98
NiI-5847.00	FeI-6392.54	0.98
VI-6274.66	VI-6111.65	0.98
VI-6090.21	TiI-4562.63	0.98
TiI-4645.19	FeI-5143.73	0.98
NiI-6007.31	FeI-5814.81	0.98
FeI-5225.53	FeI-4554.46	0.98
TiI-5071.49	ScI-5671.82	0.98
FeII-6149.25	FeII-5534.85	0.98
VI-5737.07	TiI-4562.63	0.98
MgI-4730.04	FeI-4596.41	0.98
TiI-5662.16	ScI-5671.82	0.98

to be continued ...

Var1	Var2	correlation
NiI-6007.31	FeI-5376.83	0.98
VI-6111.65	TiI-6126.22	0.98
CrI-6330.13	VI-5703.59	0.98
CrI-5247.57	VI-6081.45	0.98
NiI-6176.82	NiI-5010.94	0.98
FeI-6358.68	FeI-6270.23	0.98
FeI-4961.92	FeI-4809.94	0.98
TiI-5219.70	TiI-4562.63	0.98
CrI-4700.61	FeI-6739.52	0.98
VI-5703.59	VI-5670.85	0.98
FeI-4799.41	FeI-4596.41	0.98
CoI-4792.86	FeI-4809.94	0.98
CrI-5247.57	FeI-5143.73	0.98
VI-5670.85	TiI-5739.48	0.98
TiI-5071.49	ScI-4743.82	0.98
VI-6081.45	TiI-5866.45	0.98
TiI-5866.45	TiI-4562.63	0.98
VI-6216.35	VI-6090.21	0.98
CrI-5783.07	CrI-5214.14	0.98
FeI-6220.79	FeI-5223.19	0.98
FeI-6498.94	FeI-4554.46	0.98
VI-5737.07	TiI-5739.48	0.98
TiI-6126.22	TiI-5113.44	0.98
FeI-4799.41	FeI-4566.52	0.98
FeI-6392.54	FeI-4537.67	0.98
FeI-5636.70	FeI-4551.65	0.98
TiI-6126.22	TiI-4997.10	0.98
VI-6251.83	VI-6090.21	0.98
NiI-5847.00	FeI-5778.46	0.98
VI-5737.07	TiI-5662.16	0.98
CrI-4700.61	FeI-6120.25	0.98
NiI-5847.00	FeI-4537.67	0.98
TiI-5866.45	TiI-5113.44	0.98
VI-6090.21	TiI-5866.45	0.98
NiI-6128.98	NiI-5847.00	0.98
VI-6274.66	VI-6135.36	0.98
NiI-6086.29	NiI-5625.32	0.98
TiI-5978.55	FeI-5143.73	0.98
VI-6111.65	CaI-6156.02	0.98

to be continued ...

Var1	Var2	correlation
FeI-5436.30	FeI-4661.54	0.98
CrI-5247.57	VI-6039.73	0.98
VI-5703.59	FeI-5247.06	0.98
NiI-5847.00	FeI-4809.94	0.98
VI-6090.21	VI-6081.45	0.98
FeI-6857.25	FeI-5636.70	0.98
FeI-4799.41	FeI-4593.53	0.98
NiI-6111.08	NiI-6086.29	0.98
VI-6285.17	TiI-5978.55	0.98
NiI-5748.36	FeI-5778.46	0.98
VI-6039.73	FeI-5143.73	0.98
VI-5703.59	TiI-4675.11	0.98
CrI-5247.57	TiI-6126.22	0.98
NiI-4814.60	NiI-4512.99	0.98
VI-6135.36	CaI-6156.02	0.98
VI-6251.83	TiI-5219.70	0.98
FeI-6392.54	FeI-4809.94	0.98
NiI-5847.00	CoI-5647.24	0.98
CrI-4730.72	FeI-6392.54	0.97
TiI-4997.10	TiI-4645.19	0.97
FeI-5436.30	FeI-5855.08	0.97
NiI-6108.12	NiI-5754.68	0.97
FeI-6857.25	FeI-5436.30	0.97
FeI-5815.22	FeI-5811.92	0.97
TiI-6258.11	TiI-6126.22	0.97
CrI-5247.57	VI-5670.85	0.97
VI-6090.21	VI-6039.73	0.97
CrI-4730.72	FeI-6270.23	0.97
TiI-5662.16	TiI-5219.70	0.97
FeI-5956.70	FeI-5225.53	0.97
FeI-6703.57	FeI-6358.68	0.97
VI-5670.85	ScI-4743.82	0.97
VI-5703.59	FeI-5143.73	0.97
TiI-5113.44	TiI-4645.19	0.97
FeI-4799.41	FeI-4551.65	0.97
TiI-5978.55	TiI-5071.49	0.97
TiI-5064.06	ScI-4743.82	0.97
VI-5737.07	CaI-6156.02	0.97
CrI-5247.57	VI-6251.83	0.97

to be continued ...

Var1	Var2	correlation
FeI-6229.24	FeI-4566.52	0.97
FeI-6608.03	FeI-6358.68	0.97
CrI-4730.72	FeI-4961.92	0.97
FeII-6238.39	FeII-6149.25	0.97
FeI-6739.52	FeI-6608.03	0.97
CrI-4730.72	FeI-6240.65	0.97
VI-6251.83	TiI-5866.45	0.97
VI-6090.21	TiI-5071.49	0.97
CrI-6330.13	TiI-6126.22	0.97
VI-6251.83	FeI-5143.73	0.97
MgI-4730.04	FeI-4593.53	0.97
FeII-6084.11	FeII-5414.07	0.97
CoI-6814.95	CoI-5647.24	0.97
VI-6285.17	VI-6274.66	0.97
TiI-4997.10	TiI-4562.63	0.97
TiI-5978.55	TiI-5648.57	0.97
CrI-6330.13	TiI-5978.55	0.97
NiI-5847.00	FeI-6220.79	0.97
NiI-6186.72	NiI-6111.08	0.97
VI-6224.51	VI-6216.35	0.97
TiI-5113.44	TiI-4562.63	0.97
TiI-5866.45	FeI-5143.73	0.97
FeI-6627.55	FeI-4593.53	0.97
TiI-6064.63	TiI-5866.45	0.97
VI-6081.45	VI-5703.59	0.97
VI-6090.21	FeI-5143.73	0.97
NiI-6111.08	NiI-5010.94	0.97
VI-5670.85	TiI-5113.44	0.97
FeI-6627.55	FeI-6733.15	0.97
VI-6274.66	VI-6251.83	0.97
TiI-5866.45	TiI-4997.10	0.97
NiI-6635.13	NiI-6130.14	0.97
FeI-6229.24	FeI-4596.41	0.97
CrI-6330.13	TiI-4645.19	0.97
NiI-5754.68	FeI-6358.68	0.97
FeI-6857.25	FeI-4566.52	0.97
VI-6039.73	TiI-5866.45	0.97
CrI-4700.61	VI-6216.35	0.97
FeI-5856.09	FeI-5436.30	0.97

to be continued . . .

Var1	Var2	correlation
VI-6111.65	TiI-5739.48	0.97
NiI-6635.13	NiI-5996.73	0.97
FeI-6220.79	FeI-5778.46	0.97
FeI-6739.52	FeI-5778.46	0.97
TiI-6258.11	TiI-5662.16	0.97
VI-6111.65	TiI-4645.19	0.97
CrI-5247.57	VI-6135.36	0.97
CrI-5247.57	TiI-4645.19	0.97
FeI-5054.65	FeI-4961.92	0.97
FeI-5223.19	FeI-5054.65	0.97
VI-6216.35	TiI-5866.45	0.97
CrI-4964.93	FeI-5247.06	0.97
FeI-6392.54	FeI-6240.65	0.97
VI-6274.66	TiI-6064.63	0.97
FeI-6625.02	FeI-6498.94	0.97
NiI-5754.68	FeI-4809.94	0.97
VI-6216.35	ScI-4743.82	0.97
NiI-6378.26	NiI-5010.94	0.97
VI-6039.73	VI-5703.59	0.97
CrI-4730.72	FeI-5054.65	0.97
TiI-5113.44	TiI-5071.49	0.97
CrI-6330.13	TiI-5662.16	0.97
CrI-6330.13	FeI-4554.46	0.97
NiI-5754.68	CrI-4730.72	0.97
VI-5737.07	TiI-6126.22	0.97
FeII-6247.56	FeII-6149.25	0.97
VI-6251.83	VI-5703.59	0.97
FeI-6732.07	FeI-6699.15	0.97
FeI-6498.94	FeI-6358.68	0.97
FeI-6703.57	FeI-6082.72	0.97
FeI-5814.81	FeI-5738.24	0.97
VI-6081.45	CaI-6156.02	0.97
CrI-5247.57	TiI-5866.45	0.97
FeI-6082.72	FeI-5956.70	0.97
NiI-5625.32	NiI-5010.94	0.97
TiI-6599.12	TiI-5978.55	0.97
NiI-6186.72	NiI-5996.73	0.97
TiI-4820.41	TiI-4645.19	0.97
FeI-6226.74	FeI-4596.41	0.97

to be continued ...

Var1	Var2	correlation
VI-5703.59	TiI-4562.63	0.97
NiI-6186.72	NiI-5010.94	0.97
NiI-6635.13	NiI-6598.60	0.97
FeI-6392.54	FeI-4579.33	0.97
CrI-4700.61	TiI-5978.55	0.97
VI-6135.36	TiI-5219.70	0.97
FeI-6082.72	FeI-5223.19	0.97
NiI-5748.36	NiI-5754.68	0.97
FeI-5778.46	FeI-4537.67	0.97
CrI-4626.18	TiI-5071.49	0.97
TiI-6258.11	TiI-4645.19	0.97
NiI-6130.14	NiI-5996.73	0.97
VI-6274.66	VI-5737.07	0.97
VI-6039.73	TiI-5219.70	0.97
FeI-5956.70	FeI-4554.46	0.97
VI-6285.17	TiI-5866.45	0.97
FeI-6270.23	FeI-6082.72	0.97
CrI-5247.57	VI-6285.17	0.97
CrI-4730.72	FeI-5778.46	0.97
NiI-5754.68	FeI-5223.19	0.97
FeI-6082.72	FeI-5247.06	0.97
FeI-5636.70	FeI-4661.54	0.97
VI-6135.36	FeI-5143.73	0.97
CrI-5247.57	VI-6111.65	0.97
FeI-5587.58	FeI-5436.30	0.97
FeI-6703.57	FeI-6392.54	0.97
FeI-6725.36	FeI-4596.41	0.97
FeI-5636.70	FeI-4566.52	0.97
TiI-6064.63	TiI-5978.55	0.97
CrI-5247.57	TiI-5648.57	0.97
FeI-5778.46	FeI-4809.94	0.97
FeI-6857.25	FeI-5491.83	0.97
MnI-5413.67	FeI-6220.79	0.97
TiI-6258.11	TiI-5866.45	0.97
VI-6216.35	CaI-6156.02	0.97
NiI-6130.14	NiI-5010.94	0.97
VI-5703.59	TiI-5145.47	0.97
ScI-4743.82	CaI-6156.02	0.97
FeI-6725.36	FeI-4551.65	0.97