

Model aggregation methods for distributed machine learning

Ângelo Gabriel Almeida Gomes

Dissertação de Mestrado apresentada à
Faculdade de Ciências da Universidade do Porto em
Master's degree in Network and Information Systems Engineering
Computer Science Department
2022

Orientador

João Vinagre, Professor Auxiliar Convidado, Faculdade de
Ciências da Universidade do Porto



Sworn Statement

I, Ângelo Gabriel Almeida Gomes, enrolled in the Master Degree Engenharia de Redes e Sistemas Informáticos at the Faculty of Sciences of the University of Porto hereby declare, in accordance with the provisions of paragraph a) of Article 14 of the Code of Ethical Conduct of the University of Porto, that the content of this dissertation reflects perspectives, research work and my own interpretations at the time of its submission.

By submitting this dissertation, I also declare that it contains the results of my own research work and contributions that have not been previously submitted to this or any other institution.

I further declare that all references to other authors fully comply with the rules of attribution and are referenced in the text by citation and identified in the bibliographic references section. This dissertation does not include any content whose reproduction is protected by copyright laws.

I am aware that the practice of plagiarism and self-plagiarism constitute a form of academic offense.

Ângelo Gomes,

18-12-2022.

Acknowledgments

This dissertation marks the end of a five-year journey of extensive learning and personal development, which I had the pleasure of living in FCUP. As such, I would like to thank initially all the faculty staff that allowed this to become an enriching experience.

This journey would be immeasurably more complicated without the support and guidance of several people who have stood alongside me in this challenge.

Initially, I would like to thank my classmates and friends for all the moments they provided me with. I would like to mention with special emphasis Simão, Sónia, Eduardo, and all the members of the "herdeiros", for the constant support throughout these 5 years and for all the memories. I would also like to thank my colleague Maria João Lavoura, for the important help that her work represented in an early stage of this dissertation.

I would also like to take this opportunity to thank my mother, my father, my grandmother, my stepmother, and my sister for their education and support, that allowed me to endure and achieve all the goals I have achieved to this date, and in great part made me the man I am. To my girlfriend, I would like to thank for the encouragement and belief in me.

Finally, I want to thank Prof. João Vinagre for the guidance provided not only throughout the dissertation, but also during this last academic year. His advice and knowledge directly influenced this dissertation's outcome.

Abstract

In recent years, the amount of data and information has allowed machine learning to be increasingly present in our daily lives. Whether through systems that recommend several services and items to each user, models that allow us to estimate how long a trip will take, or translators that allow us to write a text in a language we don't know, machine learning allows a lifestyle that countless people have already become accustomed to. The huge amount of existing data, and the forecast that this amount will increase exponentially over the following years, raises several challenges to the use of machine learning techniques.

Besides the increased need for processing power that this increase in data entails, issues related to security and user privacy are also raised. Distributed machine learning has the ability to respond to these needs, since it ensures that data and processing power are distributed over numerous systems.

This dissertation analyses the effectiveness of using distributed machine learning methods, comparing them with centralized machine learning methods. We focus on the use of an interpretable method of distributed machine learning, more specifically the use of rules through the AMRules algorithm. We tested several distinct strategies, either in the splitting phase of the original dataset, or in the aggregation phase of the rules resulting from each model.

The results show that distributed machine learning has the ability to be as or even more effective than centralized machine learning. Either in regression problems as in classification problems, distributed machine learning presented very satisfactory results during our tests. Through the analysis of these results it was possible to realize in which occasions distributed machine learning is an effective and viable option.

Resumo

Nos últimos anos, a quantidade de dados e informação existente tem permitido que machine learning esteja cada vez mais presente no nosso dia-a-dia. Seja através de sistemas que recomendam diversos serviços e itens a cada utilizador, de modelos que permitem estimar o tempo que determinada viagem irá demorar, ou de tradutores que nos permitem escrever um texto numa língua que desconhecemos, machine learning permite o estilo de vida que inúmeras pessoas já se habituaram. A enorme quantidade de dados existentes, e a previsão que essa quantidade vai aumentar exponencialmente ao longo dos próximos anos, levanta diversos desafios à utilização de técnicas de machine learning.

Além do aumento das necessidades de poder de processamento que esse aumento de dados acarreta, questões relacionadas com a segurança e privacidade dos utilizadores também são levantadas. Distributed machine learning tem a capacidade de responder a essas necessidades, uma vez que garante que os dados e o poder de processamento estejam distribuídos por inúmeros sistemas.

Esta dissertação analisa a eficiência da utilização de métodos de distributed machine learning, comparando-os com métodos de machine learning centralizados. Focamo-nos na utilização de um método interpretável de distributed machine learning, mais concretamente o uso de regras, através do algoritmo AMRules. Testamos diversas estratégias distintas, seja na fase de divisão da dataset original, seja na fase de agregação das regras resultantes de cada modelo.

Os resultados mostram que distributed machine learning tem a capacidade de ser tão, e até mais, eficiente que machine learning centralizada. Seja em problemas de regressão como de classificação, distributed machine learning apresentou resultados bastante satisfatórios durante os nossos testes. Através da análise desses resultados foi possível perceber em que ocasiões distributed machine learning é uma opção eficiente e viável.

**“Without data,
you’re just another person with an opinion.”
W. Edwards Deming**

Contents

Acknowledgments	i
Abstract	iii
Resumo	v
Contents	ix
List of Tables	xi
List of Figures	xix
Acronyms	xxi
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	2
2 Background	5
2.1 Machine Learning	5
2.1.1 Supervised Learning	6
2.1.2 Unsupervised Learning	7
2.1.3 Reinforcement learning	8
2.2 Distributed Machine Learning	8
2.2.1 Federated Learning	8

2.3	Evaluating	12
2.3.1	Evaluation Paradigms	12
2.3.2	Design in offline evaluation	13
2.4	Summary	14
3	State of the art	15
3.1	Federated learning: challenges and answers	15
3.2	AMRules algorithm	17
3.2.1	Expansion of a rule	18
3.2.2	Prediction Strategies	18
3.3	Contributions of our study	19
3.4	Summary	19
4	Development	21
4.1	Aggregation methods	21
4.2	Tuning Parameters	22
4.2.1	Results	23
4.3	Summary	25
5	Experiments and results	27
5.1	Experimental methodology	27
5.2	Evaluation methodology	28
5.3	Regression results	29
5.4	Classification results	35
5.5	Summary	39
6	Conclusions	41
6.1	Future work	42
A	Exploratory Data Analysis	45

A.1	Regression datasets	45
A.1.1	Diamonds	45
A.1.2	Traffic deaths	49
A.1.3	Used cars	52
A.2	Classification datasets	56
A.2.1	Wine	56
A.2.2	Salary	59
A.2.3	Heart disease	64
B	Regression results charts	69
C	Classification results charts	83
	Bibliography	99

List of Tables

5.1	Case specification depending on the result obtained (regression).	31
5.2	Case specification depending on the result obtained (classification).	36

List of Figures

1.1	Black box models compared with white box models	2
2.1	Bias-Variance trade-off	7
2.2	Horizontal Federated Learning axis, adapted from [41]	9
2.3	Horizontal Federated Learning example, adapted from [41]	10
2.4	Vertical Federated Learning axis, adapted from [41]	10
2.5	Vertical Federated Learning example, adapted from [40]	11
2.6	Transfer Federated Learning axis, adapted from [41]	11
2.7	Transfer Federated Learning example, adapted from [41]	11
2.8	Confusion matrix	14
4.1	Ordered vs unordered results	23
4.2	Drift detection vs no drift detection results	23
4.3	2 models merged results	23
4.4	4 models merged results	24
4.5	8 models merged results	24
4.6	16 models merged results	24
4.7	32 models merged results	24
5.1	Regression results overview	29
5.2	Regression results divided by strategy	30
5.3	Regression results divided by dataset	30

5.4	Comparison between results obtained in Random Split	31
5.5	Targets distribution	32
5.6	Traffic deaths splitted through Entity and Pop_bin	32
5.7	Diamonds splitted through X and Clarity	33
5.8	Traffic deaths splitted through Pop_bin	33
5.9	Traffic deaths splitted through Entity	34
5.10	Used cars splitted through Region and State	34
5.11	Used cars splitted through Transmisssion	34
5.12	Classifications results overview	35
5.13	Classifications results divided by strategy	36
5.14	Classifications results divided by dataset	36
5.15	Target distribution affect on target split	37
5.16	Random split vs target split on Wine dataset	37
5.17	Salary splitted through Occupation	38
5.18	Random split: Wine and Salary	39
5.19	BMI_bin affect on Heart disease attribute split	39
A.1	Uncleaned diamonds data	45
A.2	Cleaned diamonds data	46
A.3	Diamonds price distribution	46
A.4	Diamonds correlation	47
A.5	X and carat correlation with price	47
A.6	Average price for each value of clarity	48
A.7	Average price for each value of color	48
A.8	Final state of the dataset	49
A.9	Uncleaned traffic deaths data	49
A.10	Cleaned traffic deaths data	50
A.11	Correlation between Historical_Population, Sidedness and Deaths	50

A.12 Average Deaths for each value of Sidedness	51
A.13 Average Deaths for each Entity	51
A.14 Final state of the dataset	51
A.15 Uncleaned used cars data	52
A.16 Cleaned used cars data	53
A.17 Cars price distribution	53
A.18 Used cars correlation	54
A.19 Correlation between transmission and car price	54
A.20 Correlation between year of the car and car price	55
A.21 Correlation between odometer and car price	55
A.22 Correlation between state and car price	56
A.23 Correlation between region and car price	56
A.24 Uncleaned wine dataset	56
A.25 Cleaned wine dataset	57
A.26 Wine quality distribution	57
A.27 Mean values for each wine quality	57
A.28 Wine quality correlation	58
A.29 Relation between alcohol and quality	58
A.30 Relation between volatile acidity and quality	59
A.31 Final state of the dataset	59
A.32 Original salary data	60
A.33 Salary distribution	60
A.34 Relation between education and salary	61
A.35 Relation between native country and salary	61
A.36 Relation between occupation and salary	62
A.37 Relation between relationship and salary	62
A.38 Relation between education num and salary	63
A.39 Final state of the dataset	63

A.40	Original heart disease dataset	64
A.41	HeartDisease distribution	65
A.42	Relation between GenHeath and HeartDisease	65
A.43	Cleaned heart disease dataset	66
A.44	Relation between Diabetic and HeartDisease	66
A.45	Relation between AgeCategory and HeartDisease	67
A.46	Relation between BMI and HeartDisease	67
B.1	Random split on Diamonds dataset	69
B.2	Random split on Traffic deaths dataset	69
B.3	Random split on Used cars dataset	70
B.4	Target split on Diamonds dataset	70
B.5	Target split on Traffic deaths dataset	70
B.6	Target split on Used cars dataset	71
B.7	Attribute split on Diamonds dataset using carat with 66% probability	71
B.8	Attribute split on Diamonds dataset using carat with 80% probability	71
B.9	Attribute split on Diamonds dataset using carat with 100% probability	72
B.10	Attribute split on Diamonds dataset using clarity with 66% probability	72
B.11	Attribute split on Diamonds dataset using clarity with 80% probability	72
B.12	Attribute split on Diamonds dataset using clarity with 100% probability	73
B.13	Attribute split on Diamonds dataset using clarity with 100% probability	73
B.14	Attribute split on Diamonds dataset using color with 66% probability	73
B.15	Attribute split on Diamonds dataset using color with 80% probability	74
B.16	Attribute split on Diamonds dataset using color with 100% probability	74
B.17	Attribute split on Diamonds dataset using x with 66% probability	74
B.18	Attribute split on Diamonds dataset using x with 80% probability	75
B.19	Attribute split on Diamonds dataset using x with 100% probability	75
B.20	Attribute split on Traffic deaths dataset using Entity with 66% probability . . .	75

B.21 Attribute split on Traffic deaths dataset using Entity with 80% probability . . .	76
B.22 Attribute split on Traffic deaths dataset using Entity with 100% probability . . .	76
B.23 Attribute split on Traffic deaths dataset using pop_bin with 66% probability . .	76
B.24 Attribute split on Traffic deaths dataset using pop_bin with 80% probability . .	77
B.25 Attribute split on Traffic deaths dataset using pop_bin with 100% probability .	77
B.26 Attribute split on Traffic deaths dataset using Sidedness with 66% probability . .	77
B.27 Attribute split on Traffic deaths dataset using Sidedness with 80% probability . .	78
B.28 Attribute split on Traffic deaths dataset using Sidedness with 100% probability .	78
B.29 Attribute split on Used cars dataset using region with 66% probability	78
B.30 Attribute split on Used cars dataset using region with 80% probability	79
B.31 Attribute split on Used cars dataset using region with 100% probability	79
B.32 Attribute split on Used cars dataset using state with 66% probability	79
B.33 Attribute split on Used cars dataset using state with 80% probability	80
B.34 Attribute split on Used cars dataset using state with 100% probability	80
B.35 Attribute split on Used cars dataset using transmission with 66% probability . .	80
B.36 Attribute split on Used cars dataset using transmission with 80% probability . .	81
B.37 Attribute split on Used cars dataset using transmission with 100% probability . .	81
B.38 Attribute split on Used cars dataset using year with 66% probability	81
B.39 Attribute split on Used cars dataset using year with 80% probability	82
B.40 Attribute split on Used cars dataset using year with 100% probability	82
C.1 Random split on Wine dataset	83
C.2 Random split on Salary dataset	83
C.3 Random split on Heart disease dataset	84
C.4 Target split on Wine dataset	84
C.5 Target split on Salary dataset	84
C.6 Target split on Heart disease dataset	85
C.7 Attribute split on Wine dataset using acidity with 66% probability	85

C.8	Attribute split on Wine dataset using acidity with 80% probability	85
C.9	Attribute split on Wine dataset using acidity with 100% probability	86
C.10	Attribute split on Wine dataset using alcohol with 66% probability	86
C.11	Attribute split on Wine dataset using alcohol with 80% probability	86
C.12	Attribute split on Wine dataset using alcohol with 100% probability	87
C.13	Attribute split on Salary dataset using education with 66% probability	87
C.14	Attribute split on Salary dataset using education with 80% probability	88
C.15	Attribute split on Salary dataset using education with 100% probability	88
C.16	Attribute split on Salary dataset using education num with 66% probability . . .	89
C.17	Attribute split on Salary dataset using education num with 80% probability . . .	89
C.18	Attribute split on Salary dataset using education num with 100% probability . .	90
C.19	Attribute split on Salary dataset using native country with 66% probability . . .	90
C.20	Attribute split on Salary dataset using native country with 80% probability . . .	91
C.21	Attribute split on Salary dataset using native country with 100% probability . .	91
C.22	Attribute split on Salary dataset using occupation with 66% probability	92
C.23	Attribute split on Salary dataset using occupation with 80% probability	92
C.24	Attribute split on Salary dataset using occupation with 100% probability	93
C.25	Attribute split on Salary dataset using relationship with 66% probability	93
C.26	Attribute split on Salary dataset using relationship with 80% probability	94
C.27	Attribute split on Salary dataset using relationship with 100% probability	94
C.28	Attribute split on Heart disease dataset using age with 66% probability	95
C.29	Attribute split on Heart disease dataset using age with 80% probability	95
C.30	Attribute split on Heart disease dataset using age with 100% probability	95
C.31	Attribute split on Heart disease dataset using bmi_bin with 66% probability . . .	96
C.32	Attribute split on Heart disease dataset using bmi_bin with 80% probability . . .	96
C.33	Attribute split on Heart disease dataset using bmi_bin with 100% probability . .	96
C.34	Attribute split on Heart disease dataset using diabetic with 66% probability . . .	97
C.35	Attribute split on Heart disease dataset using diabetic with 80% probability . . .	97

C.36 Attribute split on Heart disease dataset using diabetic with 100% probability . .	97
C.37 Attribute split on Heart disease dataset using genHealth with 66% probability .	98
C.38 Attribute split on Heart disease dataset using genHealth with 80% probability .	98
C.39 Attribute split on Heart disease dataset using genHealth with 100% probability .	98

Acronyms

AMRules Adaptive Model Rules from
High-Speed Data Streams

DCC Departamento de Ciência de
Computadores

EDA Exploratory Data Analyses

FCUP Faculdade de Ciências da
Universidade do Porto

MAE Mean Absolute Error

ML Machine Learning

MSE Mean Squared Error

Non-IID Non-Independent and Identically
Distributed

RMSE Root Mean Squared Error

USA United States of America

Chapter 1

Introduction

Machine learning is the ability to adapt to new data and make conclusions and suggestions based on calculations and analyses. It's accomplished by giving computers the ability to learn from the data they are given. Human intervention is minimal in the system's learning and decision-making. Machines, in theory, boost efficiency while eliminating (or considerably reducing) the risk of human error [4].

The near-infinite amount of data available, affordable data storage, and the development of less expensive and more powerful processing power have all boosted the evolution of machine learning. Because of the tools that machine learning provides, organizations can more easily detect lucrative opportunities and potential dangers, which is why more organizations are adopting machine learning models as C-level executives become more aware of the advantages of machine learning investments. Several global companies, such as Netflix [24] and Spotify[32], use machine learning in various areas of their business. This use allows them, among other things, to gain new customers and keep current ones satisfied, allowing the company to be profitable and successful.

Multinode machine learning methods and systems are referred to as distributed machine learning. They are designed to allow the potentialities given by centralized machine learning, overcoming problems related to large-scale learning where algorithm complexity and memory limitation are always obstacles, while allows more privacy to the users and enables the achievement of higher efficiency, mainly on a large-size domain [27].

1.1 Motivation

The total amount of data created, recorded, copied, and consumed around the world is expected to expand fast, reaching more than 180 zettabytes by 2025. The rise was greater than predicted owing to increasing demand induced by the COVID-19 pandemic, as more individuals worked and learned from home and used home entertainment sources more often [33]. The fact that we are witnessing an unprecedented increase in available data opens up new horizons and challenges for the application of distributed machine learning in our daily lives. More information, on the

one hand, is frequently associated with more efficient models; on the other hand, it requires more processing power and computing costs. Hereupon distributed machine learning will increasingly become a necessity rather than a mere possibility.

Aside from all of the processing power and computing costs mentioned in the preceding paragraph, it's also worth noting that the increase in data raises some serious concerns about user privacy. "Privacy is one of the major problems in this new electronic age", stated Andrew S. Grove, co-founder and former CEO of Intel Corporation, and it is a serious issue that users of online devices are suffering now more than ever[6].

No one can claim that obtaining information about a specific individual is just harmful to the user in question. Without the knowledge that the information provides, the existence of personalized tools that we now take advantage of, it would be nothing but a myth. We might, however, be satisfied with a product or system while yet believing that "there is room for improvement".

Distributed machine learning is a method that has the ability to provide solutions to these concerns and, as a consequence, have a positive influence on our day-to-day lives since it can address the issue of privacy without jeopardizing the ML model's efficiency. These factors make distributed machine learning a research topic with increasing relevance. However, a huge percentage of the research's discoveries produced "black box" models, such as deep neural networks, whose behavior cannot be understood even when the model's structure can be seen[39].

This lack of clarity makes it harder to believe that the model's forecasts are accurate. This is especially significant when machine learning algorithms are used in decision support systems in sensitive domains, such as jail sentences or health prognoses. Transparency, obtained by using comprehensible models (or "white-box" models), assists in properly understanding and predicting a model's behavior, which is crucial to minimize costly errors such as judgments based on personal or discriminating data (for example, gender or race should not be considered in a model responsible for defining whether or not a certain person is entitled to a bank loan) [35].

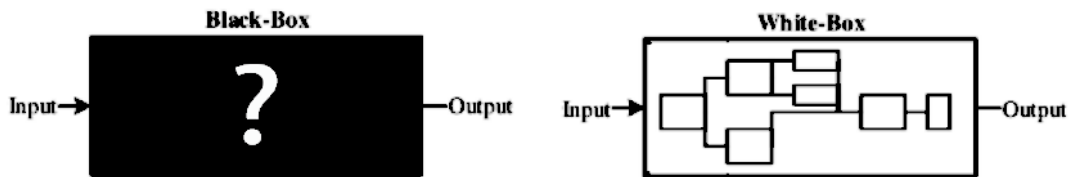


Figure 1.1: Black box models compared with white box models

1.2 Objectives

With this dissertation, we hope to demonstrate the state of the art in machine learning model aggregation, and then provide solutions to the following research questions:

-
1. Distributed machine learning works with interpretable models?
 2. Rule-based models in a distributed context perform as well, or better, than centralized models?
 3. Which criteria in a dataset indicate whether or not distributed machine learning is a good option?

Having that in mind, for this work the main goal is to compare different methods of model aggregation in distributed machine learning, applying and evaluating those methods in a federated learning environment.

Chapter 2

Background

With the growing quantity of data produced in our daily lives, the privacy of personal data is a concern that directly impacts our society. Using machine learning, big data, and deep learning approaches, service providers (such as social media, mobile applications, gaming, communications, and so on) may now gather huge amounts of information and extract important insights. This data is critical for businesses since they seek to build their business models on behavioral insights in order to optimise revenues. As a result, in today's rapidly evolving digital world, user privacy has emerged as a critical problem that must be addressed. This is a serious problem that must be addressed for the benefit of our well-being[13].

One of the solutions is the transition of centralized machine learning to distributed machine learning.

In this chapter, we will give an overview of machine learning, distributed machine learning and how to evaluate models.

2.1 Machine Learning

A machine learns when it modifies its structure, program, or data in such a way that its predicted future performance improves (based on its inputs or in reaction to external information). For example, when the performance of a natural disaster prediction model increases after seeing numerous samples of information about prior disasters, we may confidently declare that the computer has learned [25].

Exists a large number of methods that can be used to solve all kinds of real-world problems. These methods can be divided into three approaches:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

2.1.1 Supervised Learning

In supervised learning, the input and desired output are both contained in the data. In other words, the data provides instances to the learning algorithm that include both the input and the result. The purpose of supervised learning is to train a function $f : X \rightarrow Y$ that as correctly as possible describes the relationship between the input, X , and the output, Y . When approximating f , take into account that our model must be able to handle data that has never been seen before, i.e. data that has not been utilized in the training process [11]. As a result, we want to avoid overfitting our model (which performs well on training data but not well enough on unseen observations). On the other end of the spectrum, we are underfitting our model when it is too simple to even capture patterns in the data [5].

2.1.1.1 Classification and regression

Classification and regression, are among the most popular tasks for which machine learning models are used. Both of which fall under the supervised learning category.

Each example has a matching category label or class in classification. There is a difference between binary and multi-class classification. Each class in a binary context may only take one of two categorical values, commonly referred to as a negative and positive class, such as "yes" or "no". In a multi-class environment the problem's complexity is increased since the target variable can take one of three or more categorical values. However, any multi-class classification issue may be converted into a set of binary classification problem[18].

A regression task is structured similarly to a classification problem. The main difference is that the target variable now contains numerical values. The implicit ranking between observations is another distinctive feature of regression. Because the goal is a number, an order is generated that may be utilized to distinguish between larger and smaller numbers. As a result, many regression algorithms are built to work in a ranking situation, where the output is an ordered ranking of the "best" or "most suited" values rather than a single value[18].

2.1.1.2 Bias-variance trade-off

We must be wary of our model overfitting or underfitting our data, as previously indicated. It must be able to adapt well to the issue setting by learning from the training data, but not to the point where it overfits by modeling noise present in the data. The bias-variance trade-off is a characteristic that indicates that as a model's complexity grows, so does its variance, while the bias decreases. The difficulty of finding the best balance between variance and bias, with the least amount of mistake, is known as the bias-variance trade-off. This is a significant issue in supervised learning. To summarize, a large variance indicates that our model is most likely modeling random noise in the training data, implying that it is overfitting, whereas a high bias indicates that our model is unable to effectively predict f , implying that it is underfitting [8].

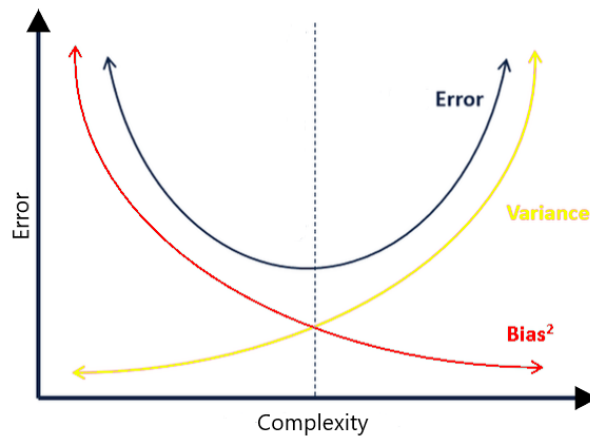


Figure 2.1: Bias-Variance trade-off

2.1.2 Unsupervised Learning

When there is no output or labeling of the input and the algorithm is expected to infer some structure from the data. The collection of training instances is now in the form of $x_1, \dots, x_N$, and the learning algorithm's purpose is to find an underlying structure or information in the training set[15].

2.1.2.1 Clustering

Clustering is the most well-known method for unsupervised learning. A clustering method seeks to group data that are "similar." The clustering process is usually guided by a distance rule that determines the distance between two points (or observations) in the feature space. The closer the two points are, the more similar the two observations are[30].

2.1.2.2 Dimensionality reduction

A modification that reduces the number of variables in a dataset is known as dimensionality reduction. Although it is common sense to believe that "more data" equals "better outcomes", we must be cautious since "more data" also implies "higher computing costs"[38],

Several unsupervised learning approaches, such as Principal Components Analysis, are available to do this task (PCA). This approach aims to create a low-dimensional representation of a dataset by selecting a collection of linear combinations of characteristics with the highest variance and are unrelated to one another [38].

2.1.3 Reinforcement learning

Reinforcement learning is the process of determining how to translate events into behaviours in order to optimize a reward signal. The learner is not informed which action to do; instead, it must pick the action that will provide the greatest reward. The two most significant characteristics of reinforcement learning are the trial-and-error search and the reward aspect. Reinforcement learning differs from supervised learning in that it learns from interaction, making it ideal for dealing with interactive problems where learning from experience is required and it is often impossible to obtain examples of desired behavior that are both correct and representative of how the learning agent should act. It also differs from unsupervised since the learner, instead of trying to find some hidden structure in the data, is trying to maximize a certain cumulative reward [34].

2.2 Distributed Machine Learning

Since the amount of existing data, and the processing power it entails, has exceeded the computational performance that exists today, the transfer of the machine learning load over numerous machines is now a necessity, transforming the centralized system into a distributed system. Distributed machine learning is gradually becoming the rule, and single-machine solutions the exception[39].

The majority of distributed learning methods are based on ensemble learning. Ensemble learning creates a group of classifiers to improve the efficiency of a single classifier. The most typical strategy involves training each classifier on a distinct sample of data. Following that, the classifiers are merged in the manner specified by the ensemble algorithm [27].

The Distributed Machine Learning ecosystem is growing, with a wide range of solutions that varies in design, methodology, and efficiency [39].

2.2.1 Federated Learning

The majority of Computer Learning research and applications focus on building a model for a specific job, such as forecasting, utilising training data stored on a single machine or in a data centre. Nowadays, training data occurs in multiple (isolated) places in many businesses and sectors. In order to ensure data privacy and security, it is not feasible to collect all of the training data in one area and undertake centralised training of machine learning models[17].

Federated Learning is a machine learning approach in which the objective is to build a high-quality model that has been trained across various clients (such as mobile devices) or data centres without ever passing training data[17].

Given the increasing computational capabilities of edge nodes (devices), Federated Learning

decentralises the centralised machine learning process and allocates the decomposed computing responsibilities to the edge nodes, where raw data is stored and learnt. Each edge node broadcasts its learnt model parameter to the central server after a certain number of iterations[22].

Based on the different distribution strategies of data sample space and feature space, there are 3 different types of Federated Learning:

- Horizontal Federated Learning
- Vertical Federated Learning
- Transfer Federated Learning

2.2.1.1 Horizontal Federated Learning

Horizontal federated learning is appropriate when the user characteristics of the two datasets coincide significantly but the users intersect little. Horizontal federated learning involves splitting datasets by example dimension and then removing the data from which example attributes are similar but examples are not precisely the same for training. To put it another way, data in various rows share the same data properties (aligned by example features). As a result, horizontal federated learning can expand the size of the sample available.[43].

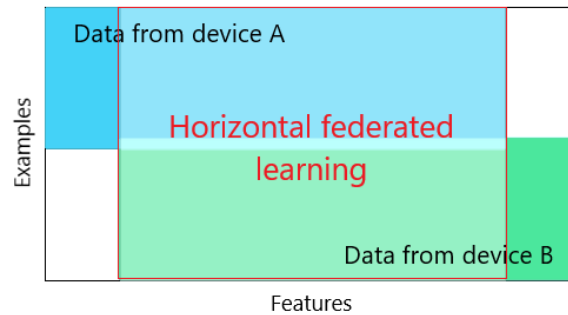


Figure 2.2: Horizontal Federated Learning axis, adapted from [41]

One example of a case where horizontal Federated Learning can be used is when the user wants to enjoy a personalized movie recommendation service provided by a movie recommender but does not want their private data to be collected. This is made feasible by preferring to have the training data distributed on local devices, which protects each user's data privacy. Each user device is considered as a party that has the rating information for all items for one individual user. These devices can work together to create a Recommender that meets both personalization and privacy standards.

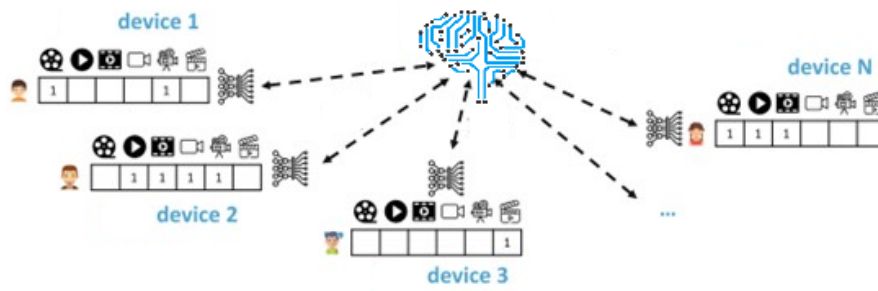


Figure 2.3: Horizontal Federated Learning example, adapted from [41]

2.2.1.2 Vertical Federated Learning

Vertical federated learning is conceivable when the example characteristics of the two datasets overlap slightly, but the examples overlap significantly. Vertical federated learning divides datasets by example characteristic dimension, then removes data from which examples are similar but example features are not identical for training. That is to say, data from different columns all belong to the same example. Vertical federated learning can therefore raise the feature dimension of training data[43].

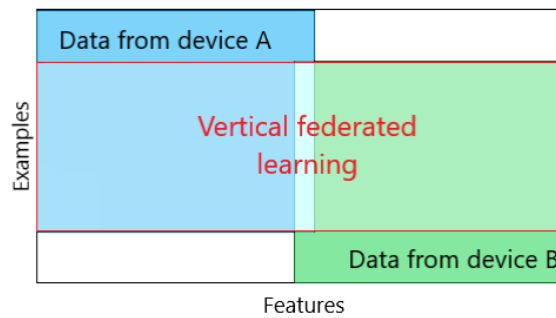


Figure 2.4: Vertical Federated Learning axis, adapted from [41]

One scenario in which Vertical Federated Learning may be used is when a bank and a Fintech firm want to collaborate on the development of a machine learning model for evaluating credit card applications. The bank has some limited data on the customers (for example, account balances), and the Fintech firm provides more data (for example, the users' online transactions). In this case, Vertical Federated Learning might produce a better model and contribute to better decisions about the bank's credit policy[40].

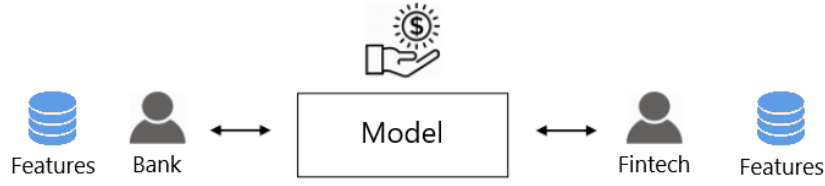


Figure 2.5: Vertical Federated Learning example, adapted from [40]

2.2.1.3 Transfer Federated Learning

When the examples and example features of the two datasets hardly overlap, we do not segment the data but instead apply transfer learning to compensate for the absence of data or labels[43].

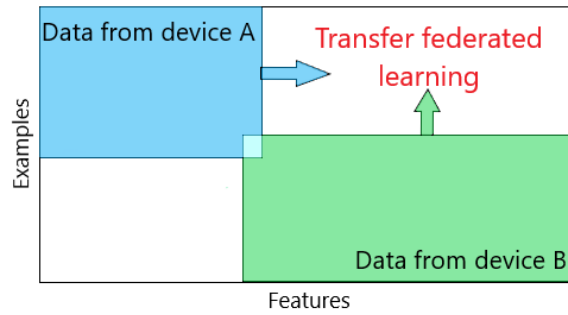


Figure 2.6: Transfer Federated Learning axis, adapted from [41]

The goal of transfer federated learning is to train a model by transferring knowledge from the source-domain party to the target-domain party while maintaining user privacy. When a popular book recommender system in region A wants to assist a movie recommender system in region B in learning a movie recommendation model cooperatively, we may utilize Transfer Federated Learning. Both the users and the objects of the two parties are different in this situation.

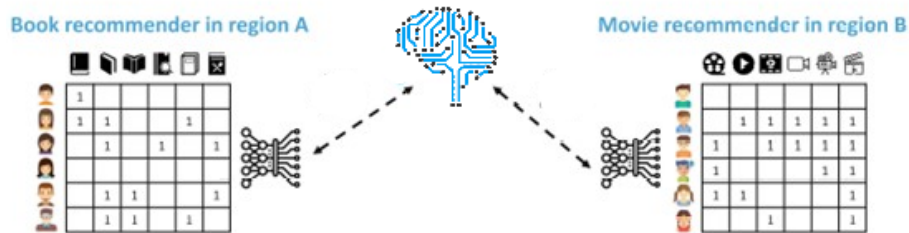


Figure 2.7: Transfer Federated Learning example, adapted from [41]

2.3 Evaluating

Evaluating is the process of determining how well a model performs. This is an important stage since, without it, we won't be able to understand the variations between algorithms, as well as the benefits and drawbacks that each one offers.

The evaluation is frequently multifaceted, and a single criterion cannot adequately reflect many of the designer's objectives. An inappropriate experimental assessment design might result in either a severe underestimating or an overestimation of a given algorithm's or model's genuine accuracy. There are two ways to proceed with the evaluation: utilizing online methods or using offline methods. User reactions are usually monitored through online methods. As a result, in online approaches, many times, user interaction is critical. Offline approaches, on the other hand, are by far the most popular methods for evaluating from a research and practice aspect[1].

2.3.1 Evaluation Paradigms

User studies, online evaluations, and offline evaluations utilizing historical data sets are the three basic methods of assessment. Users are involved in the first two types (the main difference lies in how the users are recruited)[1].

1. **Offline Evaluation with Historical Data Sets:** Ratings and other historical data are used. Because standardized frameworks and assessment metrics have been created for such circumstances, offline approaches are the most preferred strategies for system evaluation[1].
2. **Online Evaluation:** Although it can be used to predict and detect faults in a system without users, real users are frequently used as test subjects. Users are typically picked at random, and the various algorithms are evaluated on each sample of users. These approaches are based on comparing two algorithms. To do this, we divide the user segment into groups, with each group using a different algorithm for a set length of time while maintaining all other circumstances as identical as conceivable. The groups will be compared using certain metrics after the conclusion of the procedure. This form of assessment is only feasible if a substantial number of people have previously signed up. As a result, using online evaluation during the startup period is difficult[1].
3. **User Studies:** It occurs when an AI is in contact with real users, as is the case in recommender systems. Users are actively recruited and asked to engage with the system in order to complete specified activities. Before and after the engagement, the user can provide feedback, and the system can record information about the interaction. These details are then utilized to draw conclusions about the user's preferences[1].

Offline evaluation will be the type used to infer results about the models created during the course of this dissertation.

2.3.2 Design in offline evaluation

It is vital to develop models that are not grossly overstated or understated in terms of efficiency. Analysts commonly make the mistake of utilising the same data for parameter tuning and testing when evaluating models. Because parameter tuning is a part of training, and including test data into the training phase leads to overfitting, vastly overestimating the model's efficiency. To avoid this, data is usually divided into three categories: training data (used to develop the training model), validation data (used for model selection and parameter tuning), and testing data (used to assess the efficiency of the final model)[1].

In general, real-world datasets are not split into training, validation, and test datasets. As a result, the ability to automatically split the elements of a real dataset into these parts is crucial. We'll examine the two most common division procedures.[1].

1. **Hold-Out:** A portion of the data, usually small, is hidden, with the remainder used to build the training model. Calculating the accuracy of forecasting the concealed entries yields the overall efficiency. Because the evaluation inputs are hidden throughout training, the stated efficiency is not the effect of overfitting to the provided data set. Nevertheless, such an approach undervalues the actual efficiency. As a result, the assessment will be pessimistic[1].
2. **Cross-Validation:** The data is divided into k equal parts. The left standing $(k-1)$ parts are used for training, with one of the k parts serving as a test. In other words, certain entries are hidden during each training session, and the efficiency is then calculated across these entries. When k is large, this approach may properly estimate the true efficiency[1].

2.3.2.1 Accuracy Metrics in Offline Evaluation

The accuracy of an offline experiment must be tested over the test set once the assessment plan has been approved. RMSE and MAE are the most frequent metrics used to assess the regression model efficiency. They have a lower value when the performance is better[1]. The RMSE is not the same as the MAE, thus calculating the MAE value from the RMSE is complex (and vice versa). Researchers must choose the best instrument for the assignment. We should expect different results because the RMSE and MAE are defined differently. Because it is less resilient to outliers, RMSE tends to grow higher than MAE as the distribution of error magnitudes gets more varied[9].

In terms of classifying models, the goal is to determine how closely the classification indicated by the model matches to the actual categorization of the instance. The obtained results are often represented using a confusion matrix.[26].

		Actual Class	
		Positive (P)	Negative (N)
Predicted Class	Positive (P)	True Positive (TP)	False Positive (FP)
	Negative (N)	False Negative (FN)	True Negative (TN)

Figure 2.8: Confusion matrix

There are different measures for evaluation the performance of the model and through the values present in the matrix, one can quickly obtain the values of most metrics. One of the most common measures used is accuracy, that is calculated with $\frac{TP+TN}{TP+FP+FN+TN}$, but since ignores the distinctions between types of errors and is depending on class distribution in the dataset we will not use it[26]. We will use:

- **Precision:** $\frac{TP}{TP+FP}$, fraction of predictive positive cases that were correctly classified.
- **Recall:** $\frac{TP}{TP+FN}$, fraction of actual Positives that were correctly classified.
- **F1-score:** $2 \times \frac{precision \times recall}{precision + recall}$, maintains a balance between the precision and recall.
- **AUC ROC:** illustrates how effectively the probability in the positive classes are separated from the probabilities in the negative classes.

2.4 Summary

In this chapter we introduced and explained several concepts and factors, whose understanding is central to the development of the experiments that will allow us to obtain results and draw conclusions. Besides referring to what machine learning is and the differences between its variants, we also introduce the concept of distributed machine learning and federated learning, ending the chapter with an explanation of the various evaluation methods, among which some of them will be used throughout this dissertation to classify and compare the performance of models.

In the next chapter we will present results that were obtained in experiments similar to the one we are going to perform, and introduce the algorithm that we will use to build our models.

Chapter 3

State of the art

3.1 Federated learning: challenges and answers

Federated learning is an example of the more common process of "bringing the code to the data, rather than the data to the code", and it tackles the core issues of data privacy and ownership. It is a distributed machine learning method that allows training on a huge corpus of decentralized data stored on devices such as smartphones [7].

The basic feature of federated learning to assure privacy is that users have total power over local data, emphasizing data owners' privacy protection. In a federated learning environment, there are two main sorts of privacy protection systems. Using encryption methods, and/or to include privacy noise into the model parameters[43].

Identify N data owners as $\alpha_1, \dots, \alpha_N$, all of whom desire to develop a machine-learning model by combining their data $\beta_1, \dots, \beta_N$. Centralized machine learning combine all data and utilise $\beta = \beta_1 \cup \dots \cup \beta_N$ to create a model. A federated-learning method is a learning process in which data owners work together to create a model where the owner α_i does not reveal its information β_i to others [42].

Some obstacles in federated learning must be overcome in order to properly secure users' privacy.

1. **Privacy protection:** We must guarantee that the model does not leak personal information about users[43].
2. **insufficient information:** In classical machine learning, a big quantity of data is necessary to train a model with a brilliant performance, but in a distributed context, the amount of data on each device is suboptimal. On the other side, gathering all data in a centralized manner might incur significant costs. As a result, federated learning needs each device to utilize local data to train the local model, which is subsequently uploaded to the server and aggregated into a global model[43].

3. **Statistical heterogeneity:** In the federal environment, there are a high number of end devices, and the data contained by such devices could be Non-IID[43].

Much of the present federated learning research is focusing on "black-box" models such as neural networks [17][21]. However, in our study, we focused on experiments carried out in the context of federated learning using models that are interpretable for humans, such as tree models or regressions.

Tree-based federated learning

One of the hypotheses that is most used when one wants to create interpretable models is the use of decision trees. In [37] [40] and [17] is demonstrated that models based on decision trees in the context of federated learning, not only have the benefit of being interpretable, but are also able to have very satisfactory results, better than those obtained in centralised models, and even better than those obtained in existing state of the art federated learning techniques.

The steps involved in a decision trees' learning process are the following:

1. Choose the feature that divides the training data most effectively. Typically, the "most effective" feature is the one that maximizes information gain.
2. Create subgroups of the training data based on the selected feature.
3. For every subgroup, repeat steps 1 and 2.

This is carried out repeatedly until the subgroups exhibit the desired consistency level regarding the target variable.

In a federated learning scenario, each dataset D owned by some $p \in P$ (where p stands for "partie" and P for "set of parties") includes a collection of cases that are all characterized through the same set of categorical features θ and a class attribute C . The aggregator starts the decision tree model with a root node. Then, the feature $F \in \theta$ that maximizes information gain is selected based on feedback from each party in P and each potential value of F generates a child node. The feature F is then erased from θ . For each of the child nodes, this operation is repeated until one of the following situations occurs:

- θ is empty.
- The tree reaches its maximum depth, which was predetermined.
- Responses have too much noise, and because of it, using them does not benefit the model.

Regression-based federated learning

Numerous logistic regression implementations for vertical federated learning already exist, which is mainly due to the fact that there is an ease of use and widespread application for logistic regression methods in numerous binary classification tasks. However, most of these current methods are based on a design that includes three parties:

- party A, which has both partial features and labels for examples
- party B, which has just partial features
- party C, which serves as a coordinator between party A and party B

Nevertheless, in [31] is established one concept for the implementation of a parallel distributed logistic regression, in a vertical federated learning context, with a use of only two parties, allowing them to run federated logistic regression directly. This system's architecture offers a number of important benefits, such as avoiding the practical challenge of hiring a reliable third-party coordinator that can be trusted by all parties, and reducing the complexity of system deployment.

The results obtained in [31] proves that logistic regression with the use of two parties has great efficiency and scalability.

3.2 AMRules algorithm

In this dissertation we intend to develop rules based models that perform well in the context of federated learning. In order to do so we decided to use an specific algorithm.

There are multiple obstacles in developing model rules from data streams. One of which is the fact that is not possible to retain all of the data in memory and learn from them all since the dataset is no longer fixed and accessible before learning, while another is the fact that numerous consecutive scans over the training dataset are not permitted, the system must gather the important data as it comes in and make splitting decisions iteratively. Data from various origins may be included in the training data is also one of the main obstacles . AMRules is an incremental approach for learning model rules that addresses these obstacles[3].

The only memory required by the AMRules algorithm is the necessary to keep enough statistics on the rules. Also, AMRules can learn very quickly and because of that is the algorithm that performs better, including for medium-sized datasets. Additionally, it offers capabilities for explicitly detecting changes that signal when the learning process is evolving, which is important for comprehending the mechanics of evolving streams[3].

The algorithm begins with a default rule pointing to a data structure κ initialized to NULL and an unused rule set. The κ is used to store the statistics required to expand a rule and predict outcomes. The algorithm determines if a new training example is addressed by any rule from

rule set. Using the outcome values of the instances the rule applies to, the required statistic is adjusted. When an example that is not covered by any of the existing rules is received, the default rule is updated. [3].

The rule gets deleted from the rule set whenever a change is noticed. Otherwise, it is possible that the rule will be expanded, but for this to happen it is necessary that a particular number of cases is evaluated.

3.2.1 Expansion of a rule

It's important to introduce the evaluation metric used in the attribute selection procedure before talk about how rules are expanded. SDR, or standard deviation reduction, is a metric used to assess the worth of a particular split. It can be calculated effectively in an incremental manner. When you have a sample S of size N a potential split hA over attribute A would separate the examples in S into two subsets, SL and SR , with sizes NL and NR [3].

The formula for SDR measure of the split hA is given by:

$$SDR(hA) = sd(S) - \frac{NL}{N}sd(Sl) - \frac{NR}{N}sd(Sr)$$

$$sd(S) = \sqrt{\frac{1}{N}(\sum_{i=1}^N yi^2 - \frac{1}{N}(\sum_{i=1}^N yi)^2)}$$

The procedure for adding a new condition to a rule is as follows. The SDR score for every attribute is calculated for each attribute value. If the sample average's upper bound is less than 1, the real mean is likewise less than 1. In this situation, the condition attribute value $\leq$ certain value or attribute value $>$ certain value is added to the rule. Unfortunately, in terms of SDR values, two splits are frequently highly close, if not identical, and although the intervals shrink significantly as more cases are examined, it is still hard to pick one division over the other. In certain circumstances, an error threshold is employed. If the value goes below this level and the splitting condition is still not fulfilled, the split is formed on the best split with a higher SDR score and the rule is expanded[3].

3.2.2 Prediction Strategies

AMRules can produce a set of rules ordered or unordered, each one of them use different prediction methodologies to obtain optimal prediction. Only the first rule that covers an example is utilized to forecast the target example in the former. In the latter, all of the example's rules are applied to the forecast, and the final prediction is determined by a weighted vote. The weights are

originally set to random values between -1 and 1. The output is calculated using the current weights whenever a new example is received. Then, each weight is updated utilizing $\vartheta \leftarrow \vartheta + \eta(\hat{y} - y)\iota$, where $\hat{y}$ is the output, y the real value, η is the learning rate, and ι the current example [3].

Ordered rule sets must take preceding rules and remaining combinations into account, and concentrate on one rule at a time, resulting in fewer rules than the unordered technique. Because they can be understood independently, unordered rule sets are more modular[3].

The results obtained in [3], besides showing that this algorithm has a very competitive performance in terms of MAE and RMSE, also demonstrate that, in general, the use of unordered rule sets produces better results than those obtained when using ordered rule sets.

3.3 Contributions of our study

We must begin by saying that, unlike the vast majority of research and studies carried out in the area of distributed machine learning and/or federated learning, the work carried out in this dissertation will focus on the creation and study of interpretable models. Within interpretable models, tree-based models are those that have more research and, consequently, whose potentialities are more documented. Due to all these factors, a study on the efficiency of rule-based models in the context of distributed machine learning and/or federated learning will allow us to draw ideas that have not yet been drawn and thus contribute to the research already done, and the one to be done in the future.

Besides the contribution already mentioned above, it is also important to say that in relation to the methodology that was developed for performing the tests, both in the aggregation phase of the rules, where different methods were used, and in the division phase of the original dataset, it allowed us to extract ideas that, to the best of our knowledge, no other study has extracted.

3.4 Summary

In this chapter we presented the results and insights that have already been obtained when using federated learning with white box models and we introduced AMRules, the algorithm that will serve as the backbone of our tests and, consequently, results. We also explained the contributions that our work, and the results that come from it, make to the research already carried out.

In the next chapter we will present the results obtained in the initial tests and proceed to explain the aggregation models used throughout this dissertation.

Chapter 4

Development

All the code that was developed and that allowed us to achieve the results obtained is available for analysis in [44].

We will put aggregation approaches for distributed machine learning to the test in this dissertation. To do so, we needed to decide which models and parameters to utilise, as well as which datasets to use.

Datasets play a crucial role in the development of machine learning models and its quality can directly affect the efficiency of ML models [12]. The information contained in it is used not only to train, but also to test the performance of a specific model.

Different types of datasets exist, each with with different data and values stored, resulting in the existence of various algorithms. Algorithms are designed to produce the best possible models for a specific type of target value. For example, if we want to predict a continuous variable, such as the price of a product, regression algorithms are more adequate than classification algorithms. Other factors, such as the number of examples in the dataset and the correlation between the attributes and the target value to be predicted, may also affect the model's performance.

Generally, it is extremely important to go over data cleaning, analysis, and exploration strategies in order to have a better understanding of the dataset and prepare the data to be used [10]. The aforementioned strategies were applied to a few datasets as part of this project, allowing us to draw various conclusions about model performance under certain parameters.

4.1 Aggregation methods

In order to simulate a federated learning environment and achieve the objectives we have set ourselves, it is necessary to develop "local" models. In the case of the experiments we carried out, models were trained using a dataset that is smaller than the original. Once these models have been developed, they must be joined together to create a final model.

Each local model is originated using datasets with some distinctions, making it a strong possibility that the models generate different sets of rules, and even default rules, among themselves.

The challenge at this stage will then lie in how we should aggregate the rules generated by all the local models. This way we will compare the outcomes using five different models, where 4 of those 5 are originated through aggregation strategies. These models will be:

- **Baseline:** Centralised model. It will be used as a reference point to evaluate the results obtained by the models originated through a distributed strategy. The results obtained by this model will be represented by a blue dot throughout this dissertation.
- **Merged:** Aggregate model. The union of the local models will consist of aggregating the rules generated by them, without any type of treatment. The results obtained by this model will be represented by an orange dot throughout this dissertation.
- **Merged_withoutRepetition:** Aggregate model. The union of the local models will consist of aggregating the rules generated by them, taking care not to add rules that already exist in the final model. The results obtained by this model will be represented by a green dot throughout this dissertation.
- **Merged_defaultRules:** Aggregate model. The union of the local models will consist of aggregating the rules generated by them, and the default rule of the final model will be the rule with the highest weight among all the models. The results obtained by this model will be represented by a purple dot throughout this dissertation.
- **Merged_defaultRulesWithoutRepetition:** Aggregate model. The union of the local models will consist of aggregating the rules generated by them, taking care not to add rules that already exist in the final model, and the default rule of the final model will be the rule with the highest weight among all the models. The results obtained by this model will be represented by a red dot throughout this dissertation.

4.2 Tuning Parameters

Three datasets, called 2dplanes, Ailerons and Elevators, which are available at [36], were used at a very early stage of the project. These datasets were used to determine whether we will use unordered or ordered set of rules, and if drift detection is advantageous for the development of the experiments we will perform. We will also perform initial tests to discover the impact of the number of local models on the performance of the final model.

4.2.1 Results

	MAE		RMSE		TIME	
DATASETS	Unordered	Ordered	Unordered	Ordered	Unordered	Ordered
2dplanes	2.296825	2.81124	2.937709	3.737955	3.91794	13.98175
Ailerons	0.000218	0.000236	0.000272	0.000298	2.37062	2.72627
Elevators	0.003129	0.00363	0.004863	0.006176	1.5465	1.93425

Figure 4.1: Ordered vs unordered results

	MAE		RMSE		TIME	
DATASETS	Drift	No drift	Drift	No drift	Drift	No drift
2dplanes	2.14146	2.14146	2.918202	2.918202	5.88047	3.88839
Ailerons	0.000289	0.000289	0.000495	0.000495	2.74402	2.33903
Elevators	0.003433	0.003433	0.005602	0.005602	1.79450	1.44170

Figure 4.2: Drift detection vs no drift detection results

We may first conclude that unordered has a smaller error when compared to the Ordered. We also concluded that the Unordered variant was faster to execute, most likely due to the fact that Ordered rule sets specialise one rule at a time and must take preceding rules and remaining combinations into account, while unordered rules are more modular and can be understood independently.

In relation to the drift detection, there was no advantage when it was used, and its use contributed to a slower prediction model.

The next question we tried to answer was "In what way the number of merged models will affect the model's efficiency and complexity?".

2 MODELS			
MODELS:	MAE	RMSE	N° RULES
Baseline	1.747387	2.255209	27
Baseline_merg	1.780316	2.32301	37
no_rule_rep_mer	1.780357	2.323029	34
best_df_rule_mer	1.800324	2.336978	37
best_df_no_rep_m	1.800367	2.336997	34

Figure 4.3: 2 models merged results

4 MODELS			
MODELS:	MAE	RMSE	N° RULES
Baseline	1.747387	2.255209	27
Baseline_merg	2.514788	3.32848	44
no_rule_rep_mer	2.514219	3.327922	33
best_df_rule_mer	2.536544	3.352746	44
best_df_no_rep_m	2.535973	3.35219	33

Figure 4.4: 4 models merged results

8 MODELS			
MODELS:	MAE	RMSE	N° RULES
Baseline	1.747387	2.255209	27
Baseline_merg	1.646473	2.099886	60
no_rule_rep_mer	1.645934	2.099577	26
best_df_rule_mer	1.708072	2.158565	60
best_df_no_rep_m	1.707564	2.158309	26

Figure 4.5: 8 models merged results

16 MODELS			
MODELS:	MAE	RMSE	N° RULES
Baseline	1.747387	2.255209	27
Baseline_merg	2.145366	2.658452	76
no_rule_rep_mer	2.146973	2.659793	28
best_df_rule_mer	2.138815	2.65306	76
best_df_no_rep_m	2.140467	2.654455	28

Figure 4.6: 16 models merged results

32 MODELS			
MODELS:	MAE	RMSE	N° RULES
Baseline	1.747387	2.255209	27
Baseline_merg	2.260928	2.807477	94
no_rule_rep_mer	2.262329	2.809725	16
best_df_rule_mer	2.249083	2.795784	94
best_df_no_rep_m	2.250631	2.798278	16

Figure 4.7: 32 models merged results

To start, it is necessary to note that the dataset 2dplanes with a test size of 20% was used to obtain these results.

Looking at the results, we concluded that, when there are fewer models (2 and 4 in this case) and more models (16 and 32 in this case), the error is far from being the smallest possible

(achieved with 8 models in this experiment). For this reason, we can expect that the right number of models to use will be determined mostly by the characteristics of the utilized dataset and by the split strategy used.

4.3 Summary

In this chapter we present the results obtained in the initial tests, and observe that unordered rules present better results than ordered rules and that the number of models used when applying the distributed strategy affects its efficiency. Using the information collected in this chapter, in the next one we will present and analyse the results that were obtained on the datasets presented above.

Chapter 5

Experiments and results

In this chapter we will present the results obtained. Afterwards, we will analyse the results, focusing on the cases with more potential to explain the differences between the decentralized and centralized methods.

5.1 Experimental methodology

The models will be tested using three different strategies for dividing cases among the various models.

1. **Random split:** The cases are divided in an ad hoc manner.
2. **Target split:** The cases are sorted based on the target variable's value.
3. **Attribute split:** The cases are sorted based on the value of a certain attribute. This final strategy will also make use of percentages (66 percent, 80 percent, and 100 percent) to express the likelihood of a certain case with value x on attribute y ending up belonging to the model specified for cases with value x on attribute y .

After the split, the original dataset will have originated K datasets with similar dimensions, and in each of these a model will be trained. After having created K models we will proceed to aggregate the rules that constitute them using one of the aggregation methods presented in section 4.1. Lastly, we will evaluate the results obtained and compare them, among the different aggregation methods, and also in relation to a centralized model.

The datasets used throughout the different tests, which allowed obtaining the results that are demonstrated in this section, are the following:

- **Diamonds:** Available at [2], it is a dataset consisted of 53940 cases that represented diamonds, each of which was distinguished by 10 variables that provided information about each diamond. Used in regression tests.

- **Traffic Deaths:** Available at [14], it is a dataset consisted of 8010 cases. Each case shows the number of deaths in a specific year (1990-2019) and country. It was used in regression tests.
- **Used cars:** Available at [29], it is a dataset containing information provided by Craigslist about car sales. It consisted of 426880 cases, each one of them with 25 variables. It was used in regression tests.
- **Wine:** Available at [20], it is a dataset containing information about red variations of Portuguese "Vinho Verde". Consisted of 1143 cases, each one of them with 12 variables. It was used in classification tests.
- **Salary:** Available at [19], it is a dataset that contains information about 32561 people, each one characterised by 15 variables. It was used in classification tests.
- **Heart disease:** Available at [28], it is a dataset that contains information about 19795 people, each one characterised by 18 variables. It was used in classification tests.

The EDA for all these datasets can be found in the appendixes of this dissertation. This EDA provided an in-depth knowledge, necessary to, for example, choose attributes to perform attribute split.

5.2 Evaluation methodology

In order to easily find performance patterns and ensure that the results are not affected by external conditions to the model's performance, the same evaluation methodology was used in all tests developed.

This methodology consisted in dividing the originated datasets into training dataset and test dataset, with the proportions of 70% and 30% respectively. The training dataset was then used to develop the models, while the test dataset was used to evaluate the results. It is worth noting that the training and test datasets were the same for the testing of all the aggregation methods presented in section 4.1, thus ensuring that all models were developed and evaluated with the same dataset.

In order to quantify the characteristics of the created models, it is necessary to use metrics. The metrics chosen throughout our tests were MAE, RMSE, Precision, Recall, F1-Score and ROC AUC. These allow us to have a very clear idea about the performance of the created models, whether they are regression or classification models.

All tests were developed on a computer with a 6-core processor and 8 gigabytes of memory ram.

We will consider as improvement any experiment in which the error of the decentralized model has decreased when compared to the centralized model, regardless of the number of models

used to achieve that improvement. All the results of the experiments, which enabled the graphs and conclusions that follow, will be present in the appendix section.

It is hoped that the differences between these three strategies, together with the characteristics of the dataset and of each test performed, would allow the discovery of distinct patterns in the predictive ability of the models.

5.3 Regression results

Initially, we will consider the results in an aggregated way, where all the results are represented without any kind of distinction between cases. Afterwards, we will group them by dataset in which the results were generated and by the strategy that originated them.

In the following figures and analysis, it is important to know the following information in relation to the legend I have given :

- **Greatly improve:** Represents the cases in which the error (calculated through the average between the MAE and RMSE values obtained) had a decrease greater than 15%.
- **Improve:** Represents the cases in which the error (calculated through the average between the MAE and RMSE values obtained) had a decrease between 1% and 15%.
- **Decline:** Represents the cases in which the error (calculated through the average between the MAE and RMSE values obtained) increased.

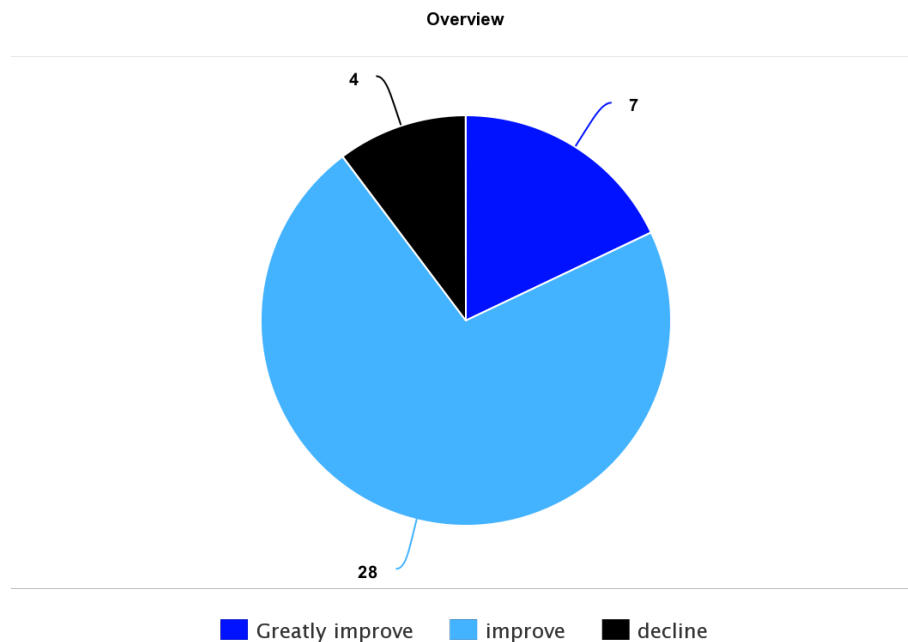


Figure 5.1: Regression results overview

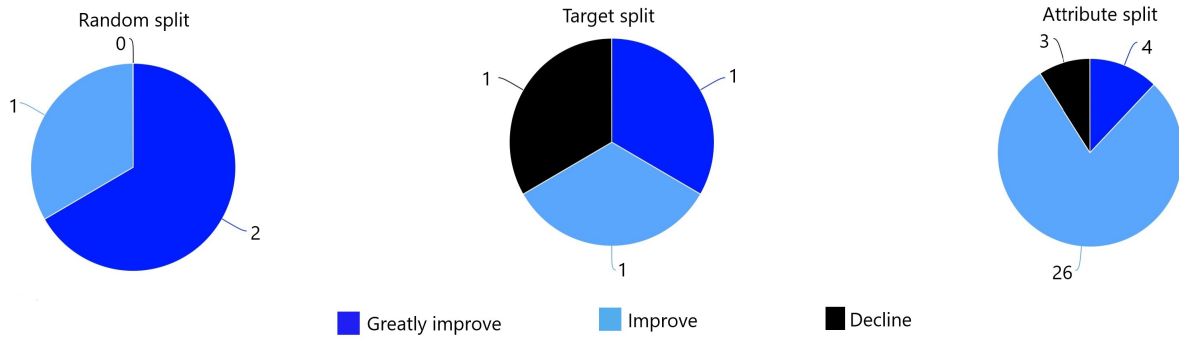


Figure 5.2: Regression results divided by strategy

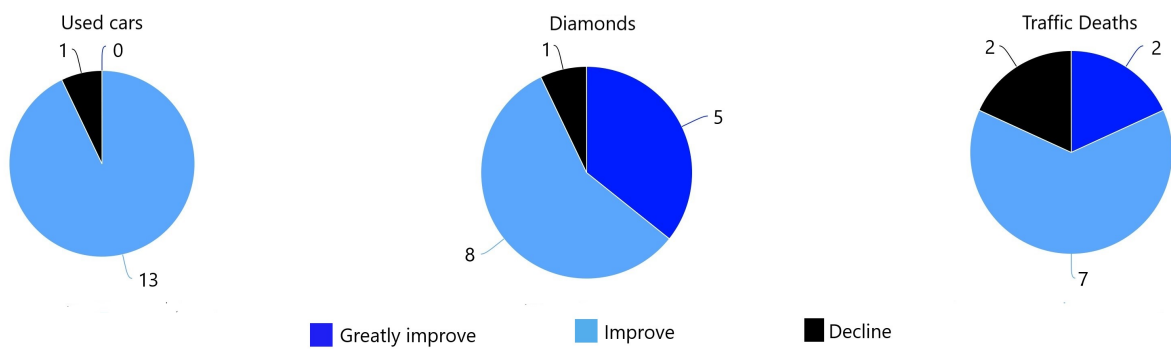


Figure 5.3: Regression results divided by dataset

We were able to quickly verify that 39 different experiments were conducted, and of these, 35 presented positive results. Among the 35 cases in which the error decreased compared to the error existing in the centralized model, on 7 occasions this decrease was very significant.

We also noticed that in 4 occasions, the MAE and RMSE of the final model, when a decentralized strategy was used, was always higher than the MAE and RMSE obtained using the centralized model.

We then specifically identified the cases (filtering them by the splitting method used and by the dataset to which they correspond) where the error decreased substantially or where the error increased, when the decentralized strategy was used.

Regression analysis

By analysing the graphs obtained, and the differences between very positive and negative cases, it was possible to notice that:

1. Random split was the only method that has always presented positive results. In the two smallest datasets this improvement was quite substantial, mainly in the case of the Traffic deaths dataset. This fact leads us to believe that, in small datasets, splitting the

High decrease of error	Increase of error
Attribute split: Diamonds: Color(1)	Attribute split: Diamonds: Clarity(0.8)
Attribute split: Diamonds: X(1)	Attribute split: Traffic deaths: pop_bin(1)
Attribute split: Diamonds: X(0.8)	Attribute split: Used cars: Transmission(0.8)
Attribute split: Traffic deaths: Entity(0.8)	Target split: Traffic deaths
Target split: Diamonds	
Random split: Traffic death	
Random split: Diamonds	

Table 5.1: Case specification depending on the result obtained (regression).

dataset using the value of an attribute or target value generates bad results, and one of the explanations for this may be that there is overfitting in the models created, and this overfit is not counter-balanced when the models are merged. However, the efficiency or otherwise of this method is dependent on the number of models used, since the results are very volatile. It is therefore imperative that, before proceeding to the final model's creation, you find out the correct number of divisions to which the dataset will be submitted (which will originate K base models to be merged)

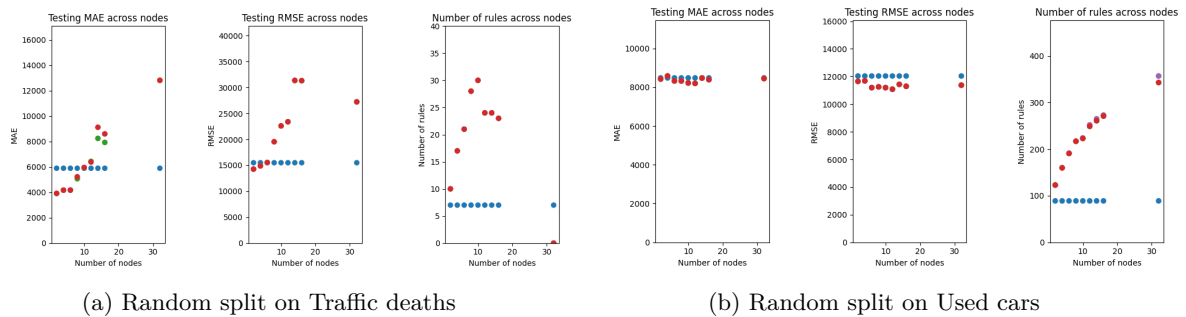


Figure 5.4: Comparison between results obtained in Random Split

We can therefore see that, although the error has a more accentuated decrease in small datasets, in large datasets the performance is not so susceptible to oscillations related to small changes in the number of models used.

2. Target split had very different results when used with the Diamonds and Traffic deaths datasets. This way, we tried to find characteristics in the distribution of these variables that could explain the differences in the results obtained.

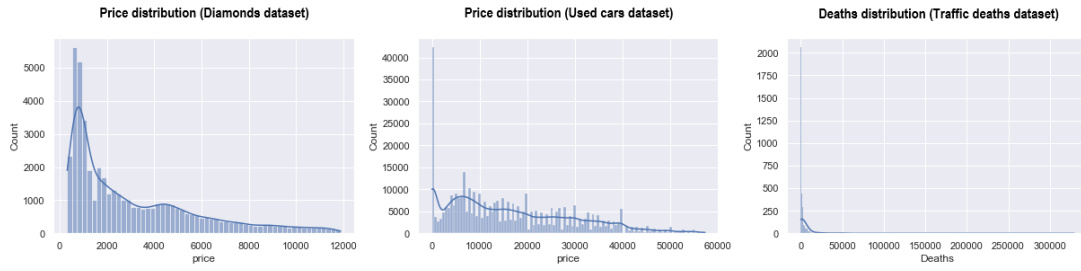
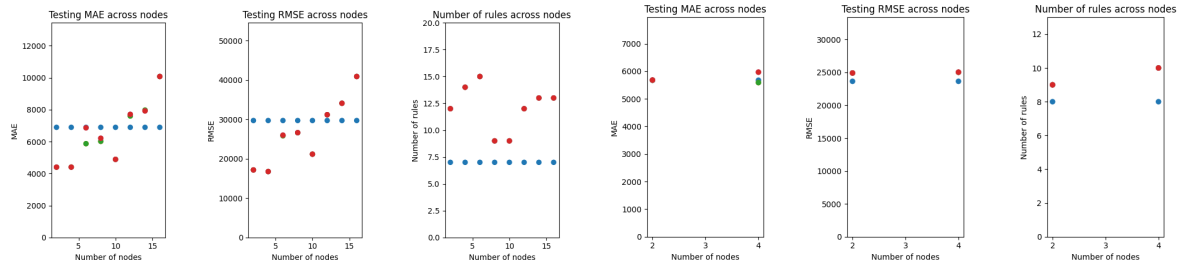


Figure 5.5: Targets distribution

With the help of the figure we were able to observe that the variable deaths in the traffic deaths dataset has a very erratic distribution, since it doesn't seem to be a normal sequence of values, something that doesn't happen in the other 2 datasets. A plausible hypothesis for this is that, if the target variable of the model is a continuous attribute that does not present a sequential distribution, the use of the target variable to split the dataset is not recommended.

3. Dataset Used cars always presents results where the error has dropped, with the exception of when the attribute transmission (with probability 80 percent) is used for the dataset split.
4. The Use of the entity attribute or the pop_bin attribute when splitting the dataset Traffic deaths generates very different results.



(a) Attribute split on Traffic deaths using Entity

(b) Attribute split on Traffic deaths using Pop_bin

Figure 5.6: Traffic deaths splitted through Entity and Pop_bin

5. The Use of the color and X attributes or the clarity attribute when splitting the Diamonds dataset give very different results.

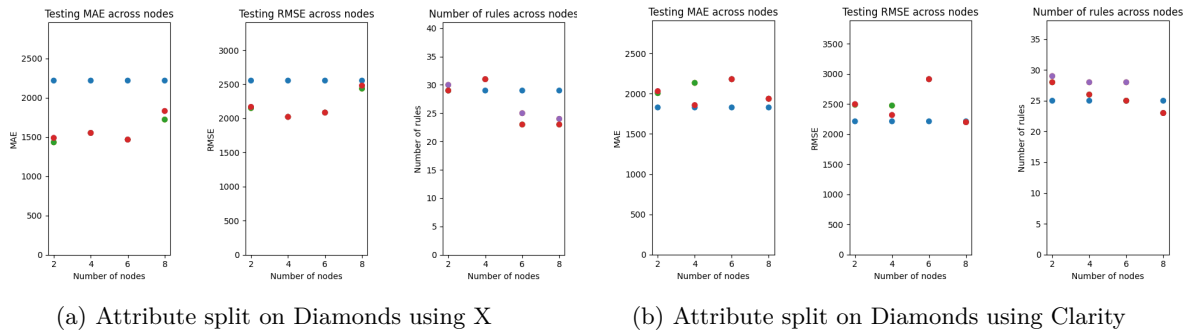
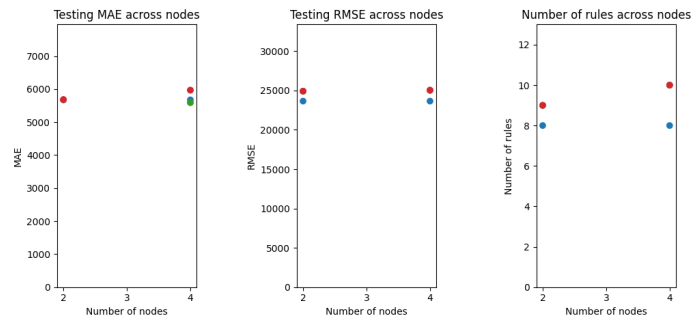


Figure 5.7: Diamonds splitted through X and Clarity

6. The attribute split and target split methods can either show positive or negative results, and the characteristics of the variable are crucial to the results obtained.
7. Through the results obtained when using the `pop_bin` attribute for the split in the Traffic deaths dataset, it is safe to say that a high correlation between the attribute used and the model's target is not in itself an indication that the model's efficiency will improve when using the attribute in question for the split.

Figure 5.8: Traffic deaths splitted through `Pop_bin`

8. The results obtained when using the Entity attribute show that having several different values is not a sufficient reason to generate several different models (for example, one model for each different value). The ideal number of models has to be empirically determined, since it depends on several factors, such as the number of distinct values in the attribute, but also the size of the datasets originated through the division, and that will be used by each model.

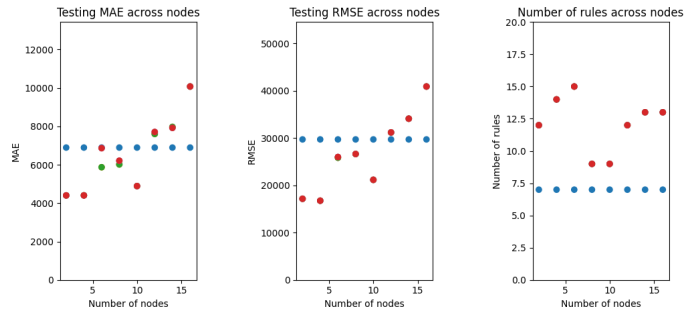


Figure 5.9: Traffic deaths splitted through Entity

9. The results obtained using the Entity attribute on the Traffic deaths dataset and the Region and State attributes on the Used cars dataset lead us to believe that geographic attributes concerning physical boundaries are generally good attributes to perform the splitting of the original dataset.

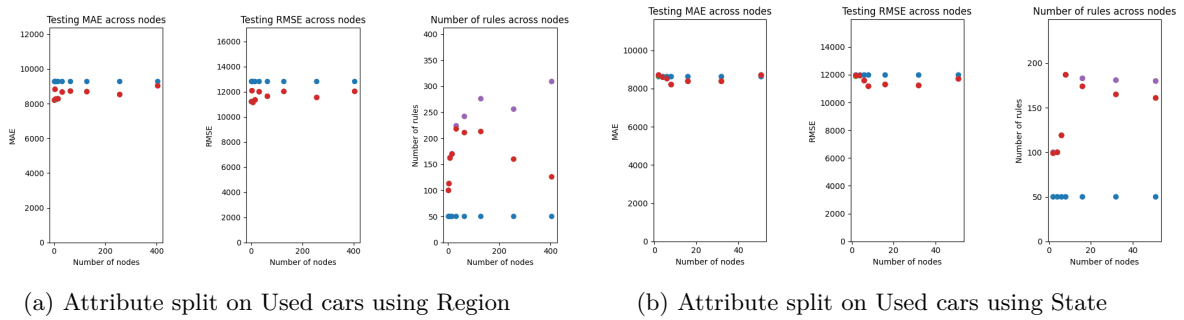


Figure 5.10: Used cars splitted through Region and State

10. The worst performance obtained in models that use the Used cars dataset was obtained through the use of the Transmission attribute. This fact leads us to hypothesize that, usually, in large datasets, it is generally not advisable to use attributes with few distinct values to split the original dataset.

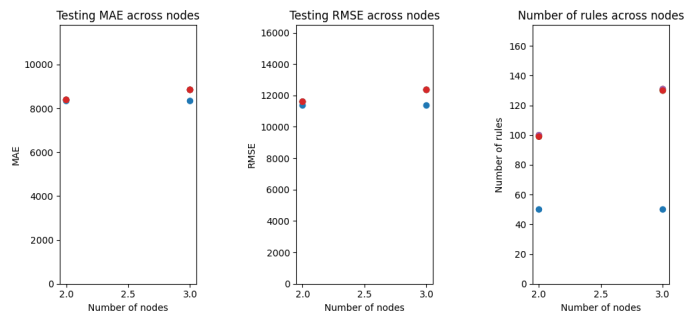


Figure 5.11: Used cars splitted through Transmission

5.4 Classification results

As done in the regression analysis, we will start by looking at the results in an aggregated way, where all the results are represented without any kind of distinction between cases. Afterwards, we will group them by dataset in which the results were generated and by the strategy that originated them.

In the following figures and analysis it is important to know the following information in relation to the legend I have given:

- **Greatly improve:** Represents the cases in which the effectiveness (calculated through F1-score and/or AUC ROC) had a increase greater than 25%.
- **Improve:** Represents the cases in which the effectiveness (calculated through F1-score and/or AUC ROC) had a increase between 1% and 25%.
- **Decline:** Represents the cases in which the effectiveness (calculated through F1-score and/or AUC ROC) decreased.

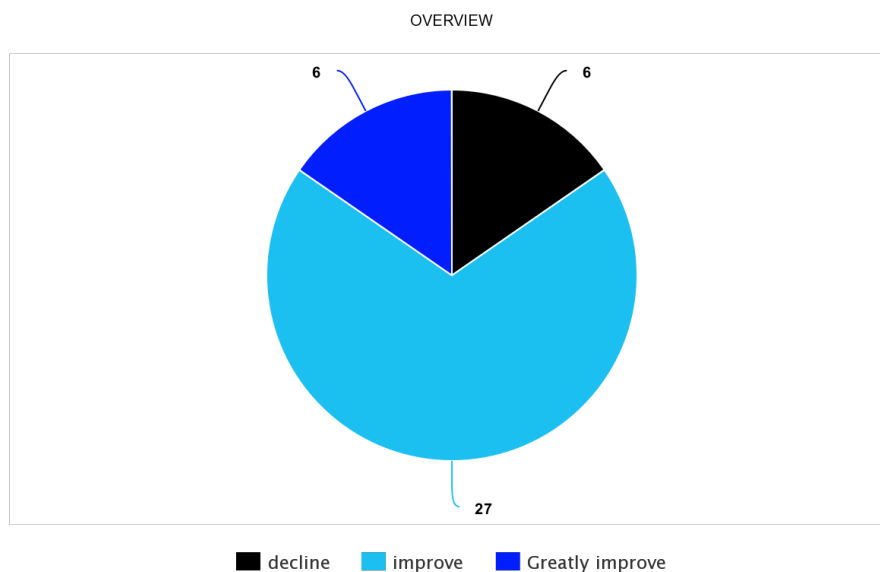


Figure 5.12: Classifications results overview

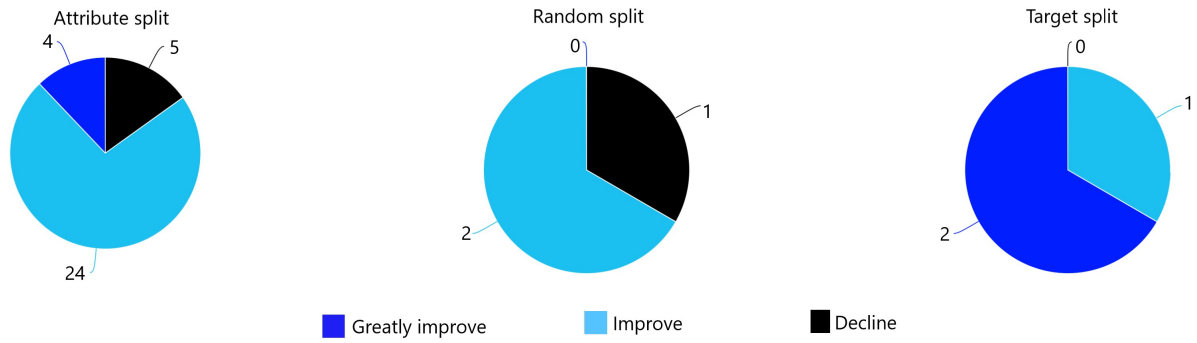


Figure 5.13: Classifications results divided by strategy

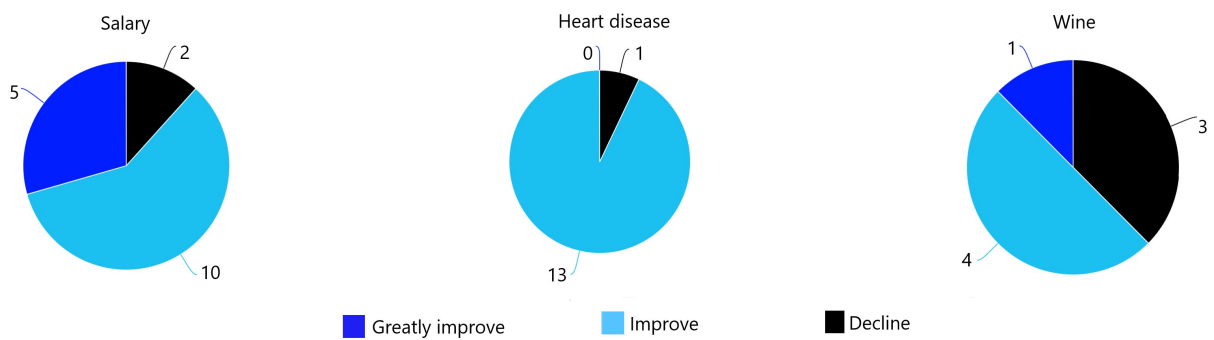


Figure 5.14: Classifications results divided by dataset

We were able to quickly verify that 39 different experiments were conducted, and of these, 33 presented positive results. Among the 33 cases in which the effectiveness increased compared to the centralized model, on 6 occasions this increase was very significant.

We also noticed that in 6 occasions, when a decentralized strategy was used, the effectiveness was always lower than the obtained using the centralized model.

We then specifically identified the cases (filtering them by the splitting method used and by the dataset to which they correspond) where the effectiveness increase substantially or where the effectiveness decreased, when the decentralized strategy was used.

High increase of performance	Decrease of performance
Attribute split: Salary: Occupation(0.66)	Attribute split: Salary: Occupation(1)
Attribute split: Salary: Occupation(0.8)	Attribute split: Heart disease: BMI_bin(1)
Attribute split: Salary: Relationship(0.8)	Attribute split: Salary: Relationship(1)
Attribute split: Salary: Native_county(0.8)	Attribute split: Wine: Acidity(0.66)
Target split: Salary	Attribute split: Wine: Alcohol(0.66)
Target split: Wine	Random split: Wine

Table 5.2: Case specification depending on the result obtained (classification).

Classification analysis

By analysing the graphs obtained, and the differences between very positive and negative cases, it was possible to notice that:

1. Target split was the only method that always showed positive results. In the two smallest datasets this improvement was quite substantial, mainly in the case of the Wine dataset. However, you need to be careful with the number of models created, and pay attention to the balance of the target variable, as you can easily undermine the efficiency of the model by increasing the number of models created.

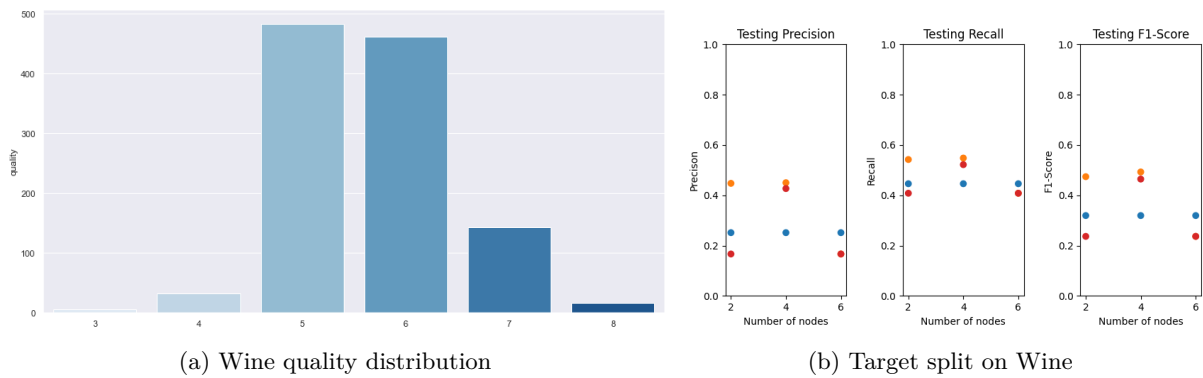


Figure 5.15: Target distribution affect on target split

2. Random split showed a negative result when it was tested on the smallest dataset.
3. Dataset Heart disease always presents results where the error has dropped, with the exception of when the attribute BMI_bin (with probability 100 percent) is used for the dataset split.
4. The results of the Target split and Random split methods are quite different when using the Wine dataset.

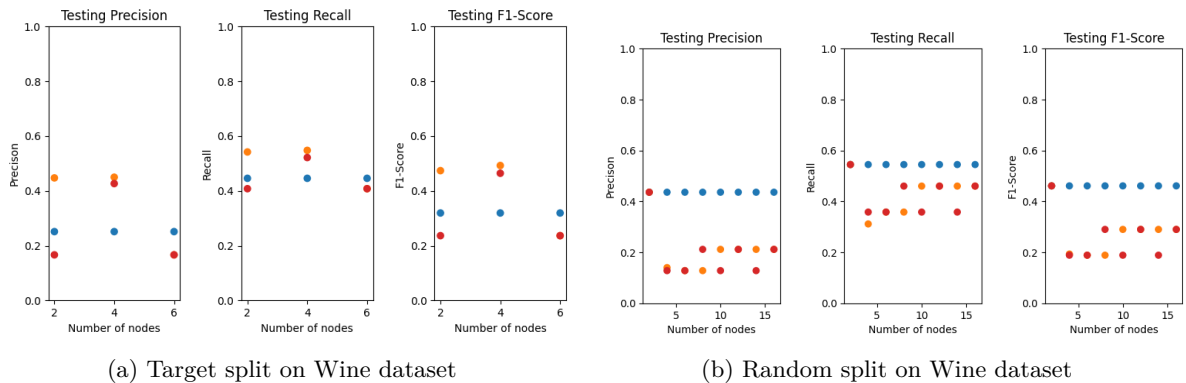
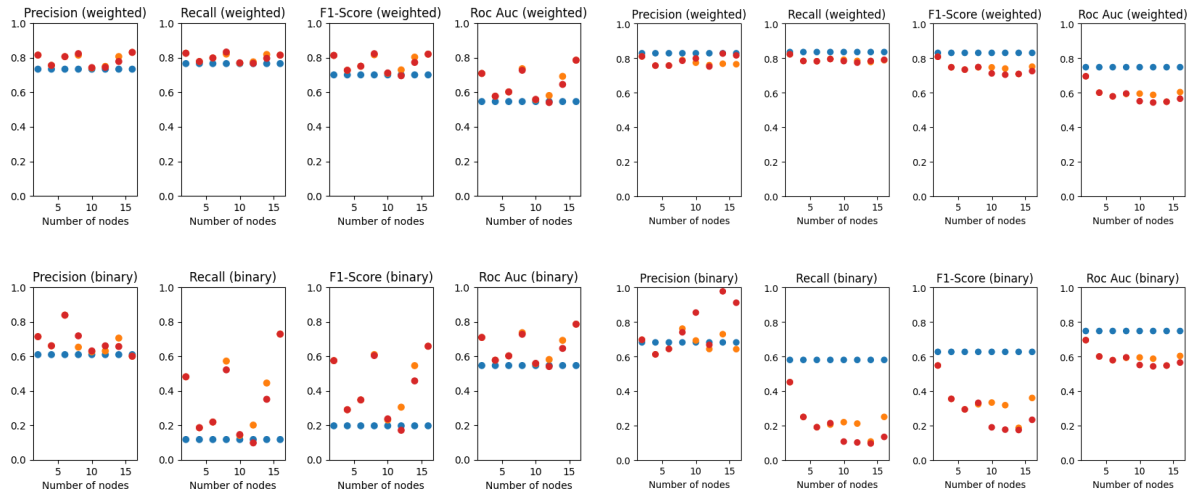


Figure 5.16: Random split vs target split on Wine dataset

5. Wine dataset has a much higher rate of negative results than the other datasets
6. The likelihood of a given case ending up belonging to the model specified for such cases had a great influence when the Occupation and Relationship attributes were used to perform the split in the Salary dataset.



(a) Attribute split on Salary using Occupation (66) (b) Attribute split on Salary using Occupation (100)

Figure 5.17: Salary splitted through Occupation

We noticed that 100 percent probability gives poor results on larger datasets, while 66 percent gives poor results on small datasets. For instance, attribute split on Salary dataset had an excellent performance when used Occupation and Relationship variables with 66 and 80 percent probability. and a decrease of performance when used with 100 percent probability. We believe that this phenomenon can be explained through the occurrence of overfitting of each model in the first case and underfitting in the second.

7. The attribute split method can show either positive or negative results, and the characteristics of the variable are crucial to the results obtained.
8. The random split method proved to be deficient on small dataset sizes (Wine dataset has 1143 cases only). In datasets of larger sizes it presented cases in which it had a higher efficiency than the centralized method, but this efficiency depended on the number of models used.

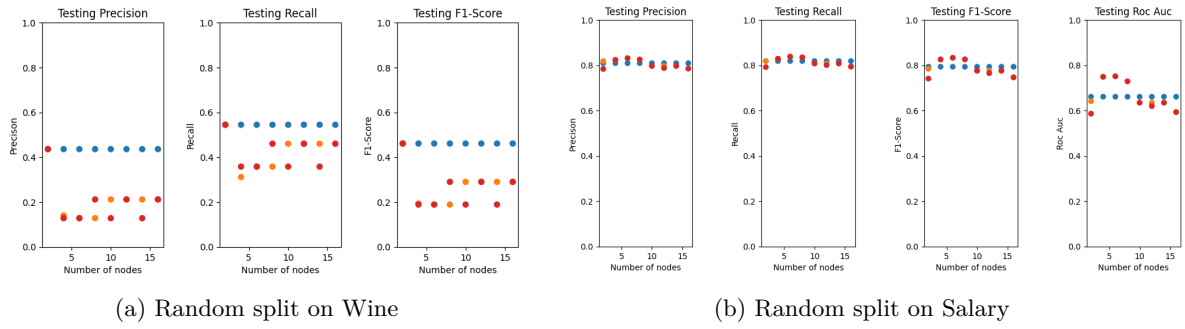


Figure 5.18: Random split: Wine and Salary

9. Since the Heart disease dataset had a negative result only when the BMI_bin variable (with a 100 percent probability) was used to perform the division, we can conclude that it is important that the variable used for the division has an influence on the target variable value, something that was not the case in BMI_bin.

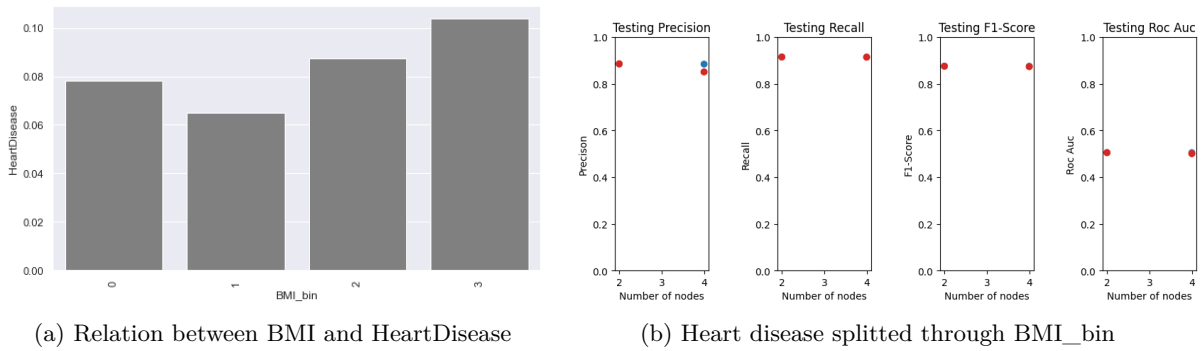


Figure 5.19: BMI_bin affect on Heart disease attribute split

5.5 Summary

In this chapter we proceed to the exposition and analysis of the results. In order to allow a deeper analysis, we divide the results by relevant factors, such as the strategy used for the initial division of the dataset.

The results obtained and the analysis performed for each specific case allowed us to find patterns and draw general conclusions, which will be presented in the next chapter.

Chapter 6

Conclusions

In this dissertation we have conducted a study regarding the use of distributed machine learning and the model aggregation methods to be used. In order to conduct the experiments using distributed learning, we used AMRules on classification and regression tasks with multiple datasets. Results have led to the following conclusions:

- The differences between regression and classification tasks have a big influence on the strategy we should employ. A clear case of this is the fact that the random split and target split strategies presented very different results depending on the task at hand. In regression tasks, random split performed better than in classification tasks, and the target split strategy performed better in classification tasks than in regression tasks. Thus, we conclude that the differences between these two strategies have a great impact on their effectiveness, and, in order to get the most out of each of them, we must first analyse the task we want to perform.
- The performance of target split is related to the target variable's distribution. For instance, on regression tasks, if the target variable of the model is a continuous attribute that does not present a sequential distribution, that is, the extreme values of the target variable are very different from the most frequent values and there is no "middle ground" between them, as we can see in the dataset [14], the target split strategy is not recommended, while on classification tasks, if the target variable is a multiclass one in which there is a large difference in the number of cases for each target value, it is important to be careful about the number of models to be employed, since we obtained poor results both when we used one model for each distinct value of the target variable and when we used only two models for the totality of target variable values.
- The size of the dataset has a large effect on the results obtained. Whether in regression or in classification tasks, datasets larger than 300000 cases presented a high stability in the results, showing little oscillation even when the number of models, aggregation method and splitting method are different from test to test, while in smaller datasets the results were greatly affected by those issues and presented a high volatility. This may explain

the fact that, on one hand, the improvements have never taken the proportions that they managed to take in the smallest datasets (notice that the Used cars and Heart disease datasets had no cases present in the tables, under "high decrease of error" or "high increase of performance"), and also that they were the datasets that presented the lowest percentage of cases in which the model performance decreased (7% in contrast to 16% in the joint results of the other 4 datasets used). Thus, it is possible that as the dataset size increases, the performance of distributed machine learning models will converge to that of centralized models.

- In general, the different methods we used for model aggregation did not translate into high differences. The predictive performance between methods was very similar; in most of the tests performed exactly the same. The difference that most stood out was the number of rules in the merged model, and consequently, its complexity.
- Correlation between attribute and target variable showed not to be a sufficient factor in itself to state that the value of a certain attribute is or is not the best reason to generate a model, and other factors, such as the number of distinct values of the attribute, have an influence on the model's performance. Even so, some patterns were found which gave certain indications as to whether a given attribute will perform well or not. For instance, we noticed that geographic attributes concerning physical boundaries are generally good attributes to generate a model, and also that the probability value had a great impact on the performance of the decentralized model, possibly causing overfit/underfit on the generated models.

6.1 Future work

As a way of ending this dissertation, I will mention the limitations that were present throughout it, and suggest some indications for future research and testing.

Initially, although I have used datasets with remarkable dimensions, it is important to note that a large portion of the challenges that machine learning seeks to answer involves datasets with much larger dimensions [16]. Due to the processing power I had during the dissertation, the computational costs that these datasets entail made it impossible to use them during the tests. Thus, in order to realize the potential of distributed federated learning in all the challenges that exist today, it is important to test the models on datasets with computational costs that a simple personal computer simply cannot match.

It would also be very interesting to be able to test and analyse the results that this technique would have in a true federated learning environment, i.e., instead of splitting a pre-existing dataset and creating models for each split using the same device, we would use different devices, where each device would pre-process the data and generate a model that would be aggregated with other models created by other devices.

Finally, it would be valuable to study the efficiency of these methods for tasks related to recommender systems. A recommender system's purpose is to create relevant suggestions for products that may be of interest to a group of users. Recommendations of books on Amazon or films on Netflix are concrete examples of how recommender systems work and their importance nowadays[23]. In addition to regression and classification tasks, it is one arm of machine learning that has been widely used in recent years.

Appendix A

Exploratory Data Analysis

A.1 Regression datasets

A.1.1 Diamonds

Initially, the dataset consisted of 53940 cases that represented diamonds, each of which was distinguished by attributes that provided information about each diamond (size, clarity, price, and so on). The goal of the model created with this dataset is to predict the value of the attribute "price" with the least amount of error possible.

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

Figure A.1: Uncleaned diamonds data

In order for the data to be analyzed and used by the models in a more efficient manner, the following steps were produced in the initial phase:

1. Remove the column "Unnamed: 0" because it serves no purpose.
2. Change the "cut," "color," and "clarity" values in the columns to numeric values.
3. Outliers were removed using the interquantile range approach (A total of 6746 cases were removed).
4. Reorganize the columns so that the "price" column, which represents the target value of our models, appears at the end of the dataset.

	x	y	z	carat	cut	color	clarity	depth	table	price
0	3.95	3.98	2.43	0.23	2.0	1.0	3.0	61.5	55.0	326.0
1	3.89	3.84	2.31	0.21	3.0	1.0	2.0	59.8	61.0	326.0
3	4.20	4.23	2.63	0.29	3.0	5.0	5.0	62.4	58.0	334.0
4	4.34	4.35	2.75	0.31	1.0	6.0	3.0	63.3	58.0	335.0
5	3.94	3.96	2.48	0.24	4.0	6.0	7.0	62.8	57.0	336.0

Figure A.2: Cleaned diamonds data

After cleaning the data, it was time to analyse the information contained within them.

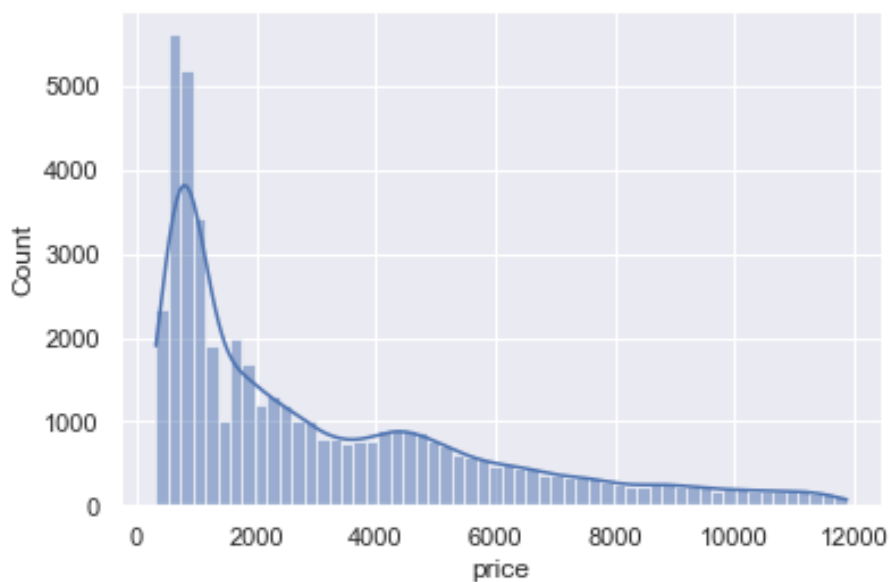


Figure A.3: Diamonds price distribution

Initially, we examined the target value distribution and its link to the various attributes.

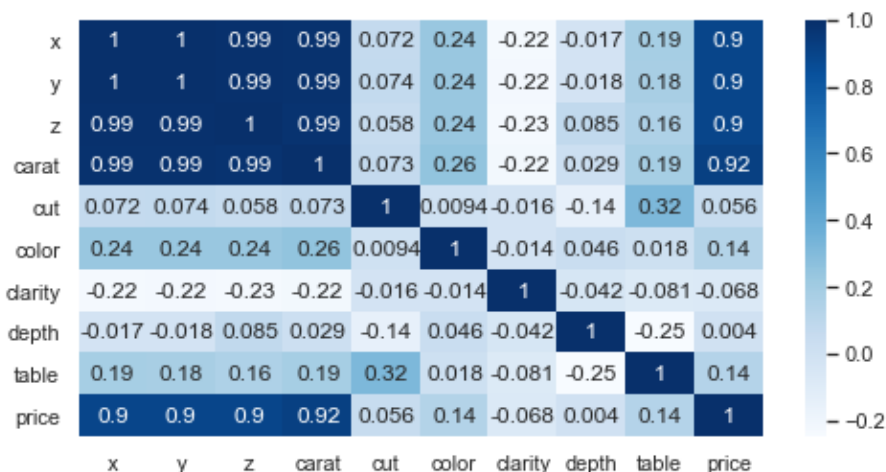


Figure A.4: Diamonds correlation

It was possible to conclude that the quantity of diamonds available decreases as the price rises, and that the attributes "carat" and those related to the size of the diamond ("x", "y", and "z") have the greatest influence on the price. It was also noted that, despite "clarity" and "color" not having a strong correlation with the desired value, they do with "carat", "x", "y", and "z".

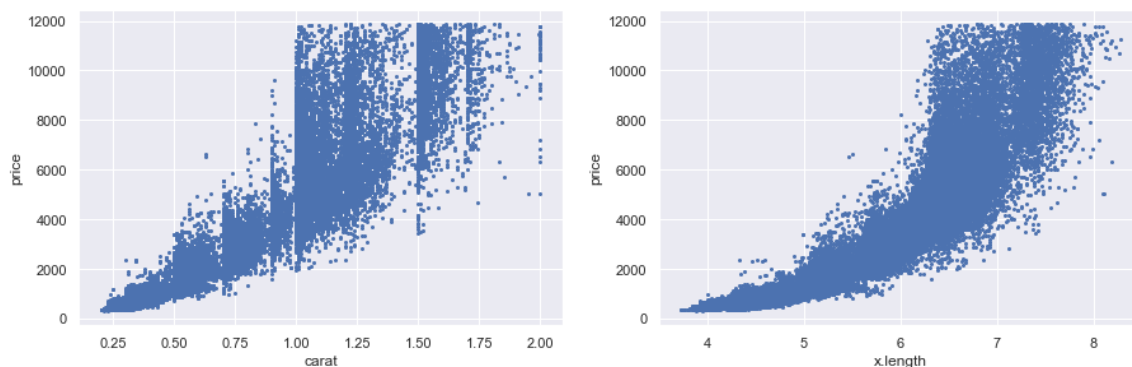


Figure A.5: X and carat correlation with price

To see if the attributes "clarity" and "color" had an influence on the value of "price," graphs were created that show the average price for each different value of "clarity" and "color."

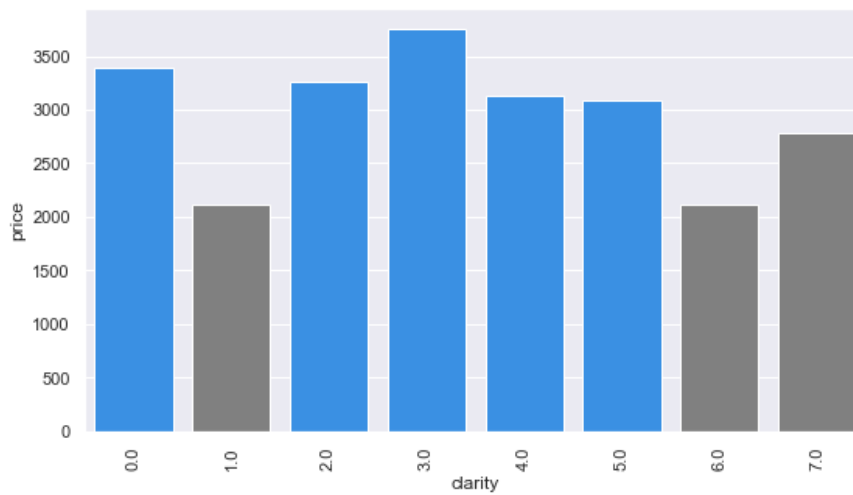


Figure A.6: Average price for each value of clarity

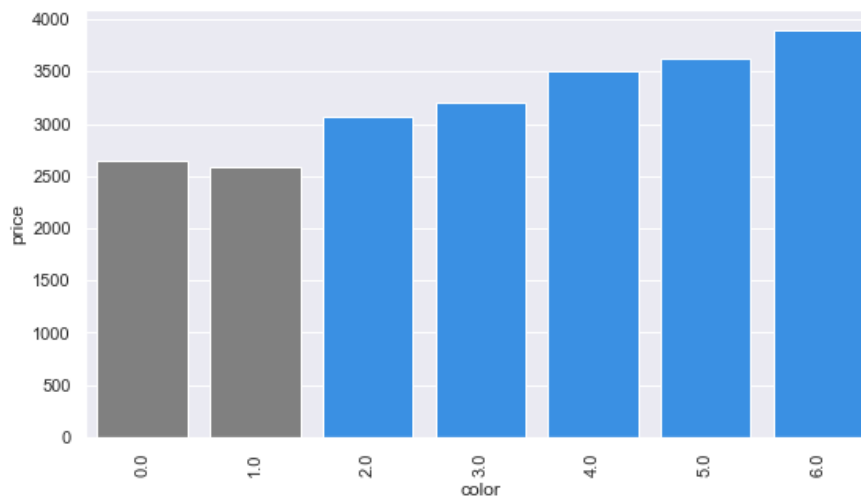


Figure A.7: Average price for each value of color

As a result, we were able to conclude that the four attributes mentioned have an effect on the final value of the target variable. In this way, we'll use the attributes "clarity," "color," "carat," and "x" to conduct experiments that will allow us to draw conclusions about the influence that certain parameters have on model efficiency. To carry out these experiments on the attributes "carat" and "x," it was necessary to resort to binning (four distinct values in the case of "carat" and eight in the case of "x").

	x	x_bin	y	z	carat	carat_bin	cut	color	clarity	depth	table	price
0	3.95	0	3.98	2.43	0.23	0	2.0	1.0	3.0	61.5	55.0	326.0
1	3.89	0	3.84	2.31	0.21	0	3.0	1.0	2.0	59.8	61.0	326.0
3	4.20	0	4.23	2.63	0.29	0	3.0	5.0	5.0	62.4	58.0	334.0
4	4.34	1	4.35	2.75	0.31	0	1.0	6.0	3.0	63.3	58.0	335.0
5	3.94	0	3.96	2.48	0.24	0	4.0	6.0	7.0	62.8	57.0	336.0

Figure A.8: Final state of the dataset

A.1.2 Traffic deaths

The smallest used dataset, initially consisting of only 8010 cases. Each case shows the number of deaths in a specific year (1990-2019) and country. The goal of the model created with this dataset is to predict the value of the attribute "deaths" with the least amount of error possible.

	Entity	Code	Year	Deaths	Sidedness	Historical_Population
0	Afghanistan	AFG	1990	4154	0	12412311.0
1	Afghanistan	AFG	1991	4472	0	13299016.0
2	Afghanistan	AFG	1992	5106	0	14485543.0
3	Afghanistan	AFG	1993	5681	0	15816601.0
4	Afghanistan	AFG	1994	6001	0	17075728.0

Figure A.9: Uncleaned traffic deaths data

In order for the data to be analyzed and used by the models in a more efficient manner, the following was produced in the initial phase:

1. Remove the "Code" column because it serves no purpose.
2. Remove all cases that have NA values.
3. Change the type of values in the "Entity" columns to numeric values.
4. Reorganize the columns so that the column "Deaths", the target value of our models, appears at the end of the dataset.
5. Remove cases where "Historical_Population" is greater than 1500000000, as these are obvious errors.

At the end of the operation, there were 6240 cases in the dataset.

	Sidedness	Historical_Population	Entity	Year	Deaths
0	0	12412311.0	0	1990	4154
1	0	13299016.0	0	1991	4472
2	0	14485543.0	0	1992	5106
3	0	15816601.0	0	1993	5681
4	0	17075728.0	0	1994	6001

Figure A.10: Cleaned traffic deaths data

First, we looked at how target deaths were distributed in relation to the attributes "Historical_Population" and "Sidedness".

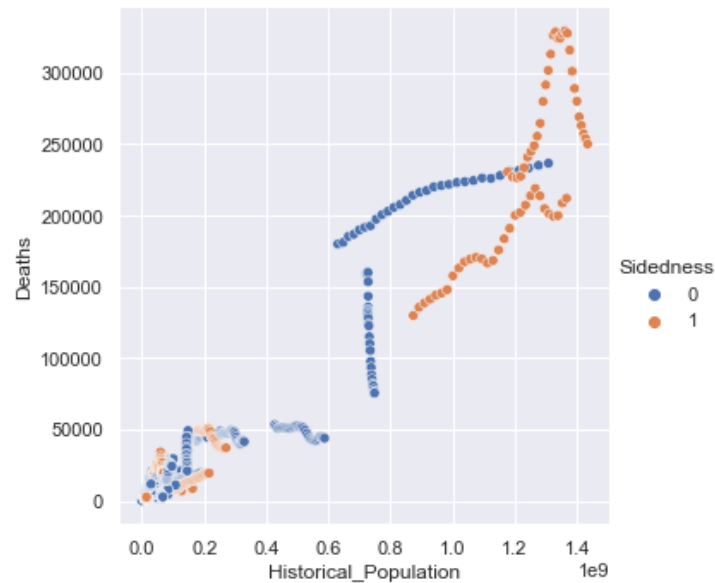


Figure A.11: Correlation between Historical_Population, Sidedness and Deaths

We can see from the data analysis that there is a strong correlation between "Historical_Population" and "Deaths," and that the two are related. As would be expected, more populated countries have a higher absolute quantity of fatal accidents.

Another question to which we sought an answer was "What is the impact of "Entity" and "Sidedness" on the average value of "Deaths"? " given that the correlation matrix only provides completely reliable information between two continuous variables (as in the cases of "Historical Population" and "Deaths"). The following graphics were created in order to provide this response.

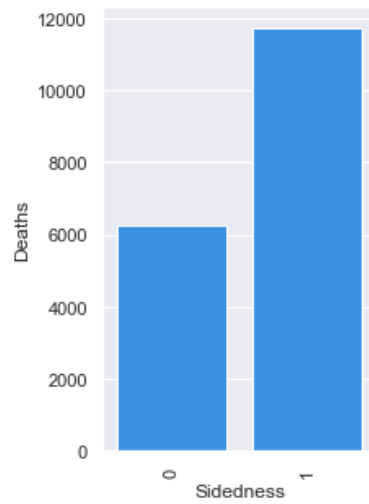


Figure A.12: Average Deaths for each value of Sidedness

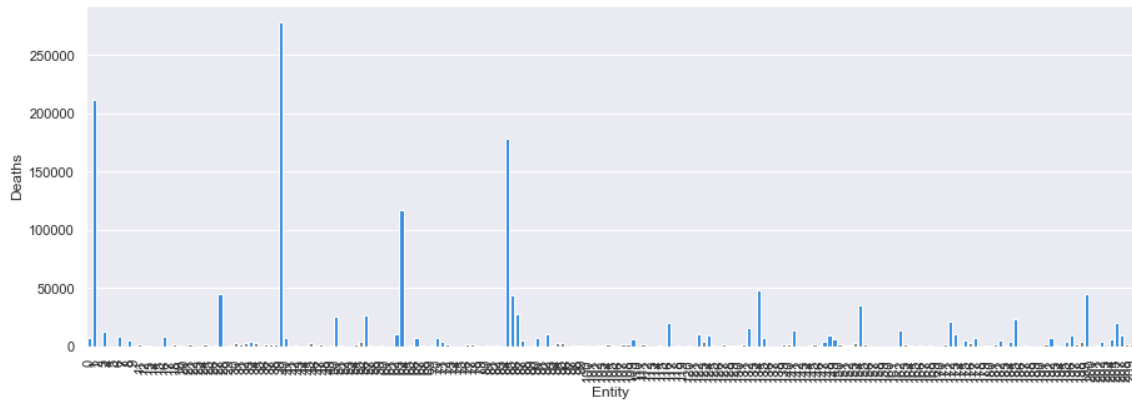


Figure A.13: Average Deaths for each Entity

As a result, we were able to conclude that the three attributes mentioned have an effect on the final value of the target variable. In this way, we'll use the attributes “Historical_Population”, “Entity”, e “Sidedness” to conduct experiments that will allow us to draw conclusions about the influence that certain parameters have on model efficiency. To carry out these experiments on the attribute “Historical_Population”, it was necessary to resort to binning (four distinct values).

	Sidedness	Historical_Population	Entity	Year	pop_bin	Deaths
0	0	12412311.0	0	1990	0	4154
1	0	13299016.0	0	1991	0	4472
2	0	14485543.0	0	1992	0	5106
3	0	15816601.0	0	1993	0	5681
4	0	17075728.0	0	1994	0	6001

Figure A.14: Final state of the dataset

A.1.3 Used cars

Contains information that Craigslist provides on car sales including columns like price, condition, manufacturer, odometer, and 18 other categories. The goal of the model created with this dataset is to predict the value of the attribute "price" with the least amount of error possible. It was initially made up of 426880 unique cases, making it the largest and most complex dataset used during the experiments.

```
Data columns (total 26 columns):
#   Column              Non-Null Count  Dtype
---  -
0   id                   426880 non-null  int64
1   url                   426880 non-null  object
2   region                426880 non-null  object
3   region_url           426880 non-null  object
4   price                 426880 non-null  int64
5   year                  425675 non-null  float64
6   manufacturer          409234 non-null  object
7   model                 421603 non-null  object
8   condition             252776 non-null  object
9   cylinders              249202 non-null  object
10  fuel                  423867 non-null  object
11  odometer              422480 non-null  float64
12  title_status          418638 non-null  object
13  transmission          424324 non-null  object
14  VIN                   265838 non-null  object
15  drive                 296313 non-null  object
16  size                  120519 non-null  object
17  type                  334022 non-null  object
18  paint_color           296677 non-null  object
19  image_url             426812 non-null  object
20  description            426810 non-null  object
21  county                0 non-null       float64
22  state                 426880 non-null  object
23  lat                   420331 non-null  float64
24  long                  420331 non-null  float64
25  posting_date          426812 non-null  object
dtypes: float64(5), int64(2), object(19)
```

Figure A.15: Uncleaned used cars data

In order for the data to be analyzed and used by the models in a more efficient manner, it was produced in the initial phase:

1. Delete useless columns ("id","url","region url","image url","county," and "description")
2. Columns with more than 30% NA values ('condition', 'cylinders', 'VIN','size', 'paint color', 'drive') were erased.
3. Substitute the NA values in the remaining columns for the mode value in that column.
4. Use interquartile range approach to remove outliers.
5. Change the columns with "object" types to "numerical" types.
6. Retrieve more information (hours, days of the week, and months) from the variable "posting date"

7. Reorganize the columns such that the "price" column, which represents the target value for our models, appears at the end of the dataset.

```
Data columns (total 16 columns):
#   Column                Non-Null Count  Dtype
---  -
0   region                 399441 non-null  int16
1   year                   399441 non-null  int32
2   manufacturer            399441 non-null  int8
3   model                  399441 non-null  int16
4   fuel                   399441 non-null  int8
5   odometer                399441 non-null  float64
6   title_status            399441 non-null  int8
7   transmission            399441 non-null  int8
8   type                   399441 non-null  int8
9   state                  399441 non-null  int8
10  lat                    399441 non-null  float64
11  long                   399441 non-null  float64
12  posting_weekday         399441 non-null  int64
13  posting_hour            399441 non-null  int64
14  posting_month           399441 non-null  int64
15  price                   399441 non-null  float64
dtypes: float64(4), int16(2), int32(1), int64(3), int8(6)
```

Figure A.16: Cleaned used cars data

We then started the data analyses by looking at the distribution of the target variable and the connections she had with the other attributes.

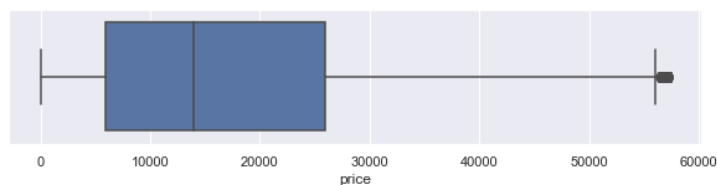


Figure A.17: Cars price distribution

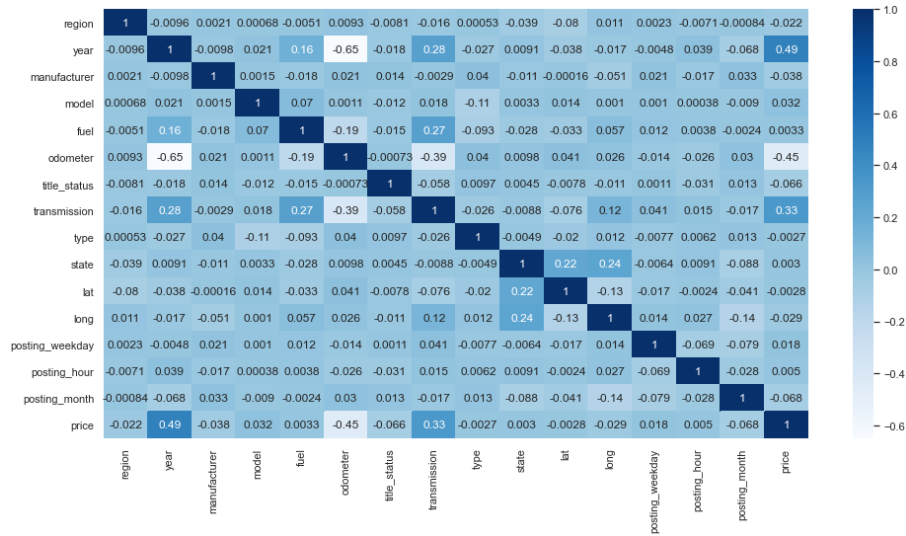


Figure A.18: Used cars correlation

Immediately seeing the strong correlation between "price" and the attributes "odometer," "year," and "transmission". This relationship is easily discernible by the following graphs, which show the average "price" for each year and transmission type and the relation between "odometer" and "price".

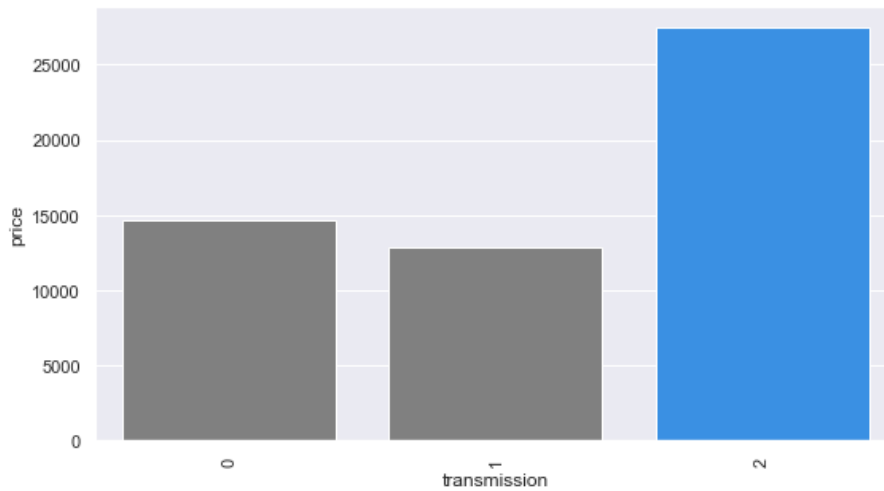


Figure A.19: Correlation between transmission and car price

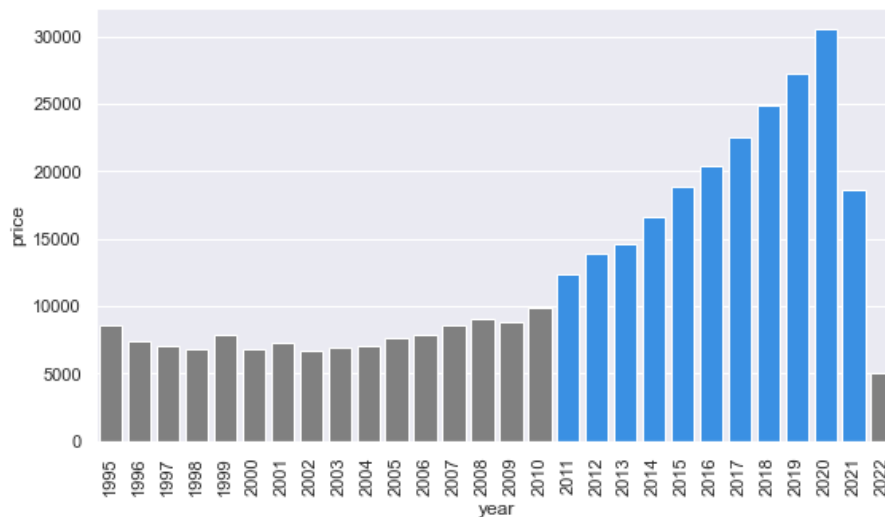


Figure A.20: Correlation between year of the car and car price

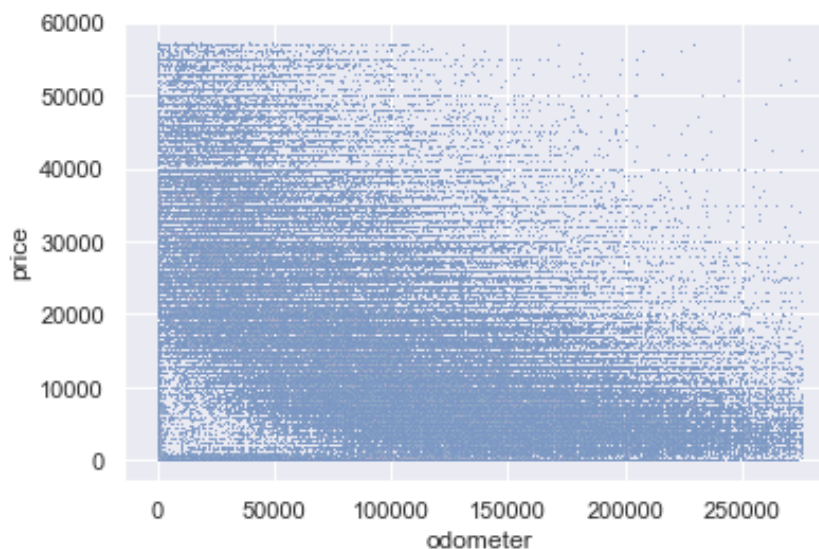


Figure A.21: Correlation between odometer and car price

We can easily conclude that "price" values are often higher in cars from 2011 to 2021, as well as when the transmission has a value of 2. We also came to the conclusion that the variables "odometer" and "price" are linked, since that when the value of "odometer" rises, the value of the variable "price" tends to fall.

Another question to which I sought an answer was "What is the impact of "state" and "region" on the average of "price"? " given that the correlation matrix only provides us with completely reliable information between two continuous variables (as in the case of "odometer" and "price"). The following graphics were created in order to provide this response.

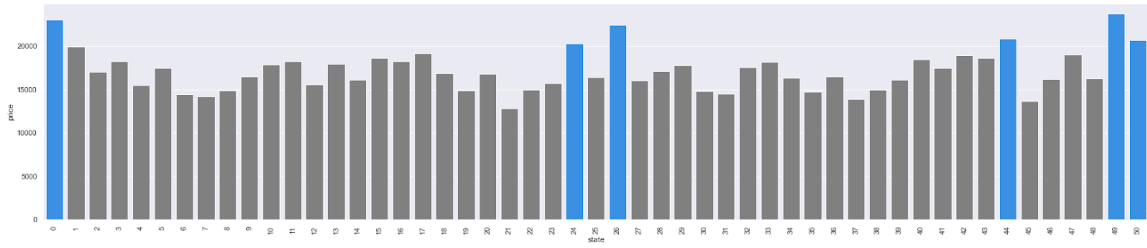


Figure A.22: Correlation between state and car price

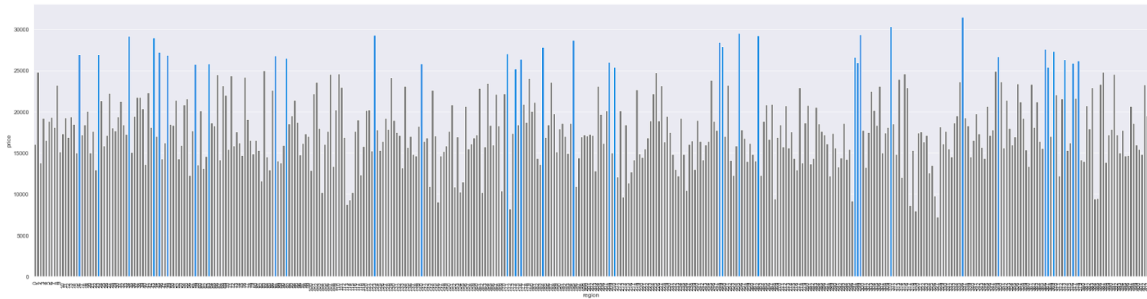


Figure A.23: Correlation between region and car price

With the visual aid created, we deduce that the attributes "state" and "region" are also capable of causing a large fluctuation in the target value "price".

As a result, we choose to use the attributes "state", "region", "year", and "transmission" to conduct experiments that allow us to draw conclusions about the influence that certain parameters have on model efficiency. We won't be using the "odometer" since it would be necessary to discretize the variable, which would be unhelpful in a dataset of this size.

A.2 Classification datasets

A.2.1 Wine

This information is about red variations of the Portuguese "Vinho Verde" wine. The dataset details the amount of different compounds found in wine as well as their impact on its quality. The model objective is to forecast the quality of wine based on the information provided.

Initially, the dataset consisted of 1143 cases and columns.

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	id
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	0
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5	1
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5	2
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6	3
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	4

Figure A.24: Uncleaned wine dataset

In order for the data to be analyzed and used by the models in a more efficient manner, the column "id" was deleted in the initial phase

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

Figure A.25: Cleaned wine dataset

We began the data analysis by looking at the total number of wines for each distinct value of wine quality, and if there were any factors that were responsible for improving or worsening the quality of the wine.

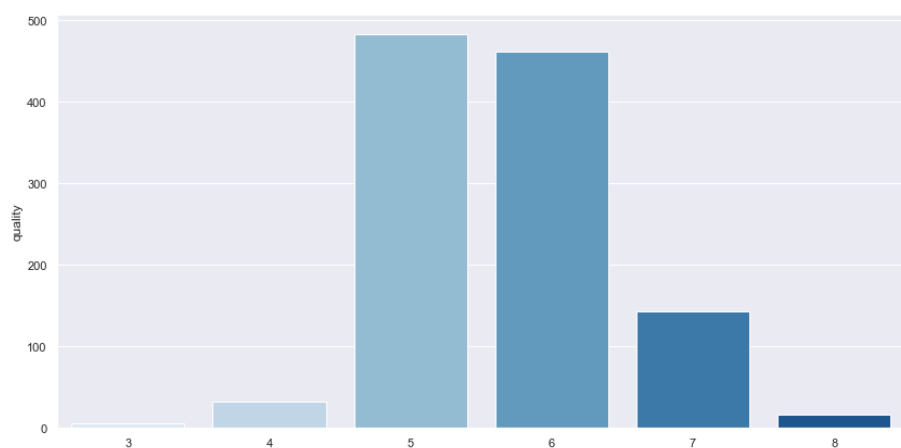


Figure A.26: Wine quality distribution

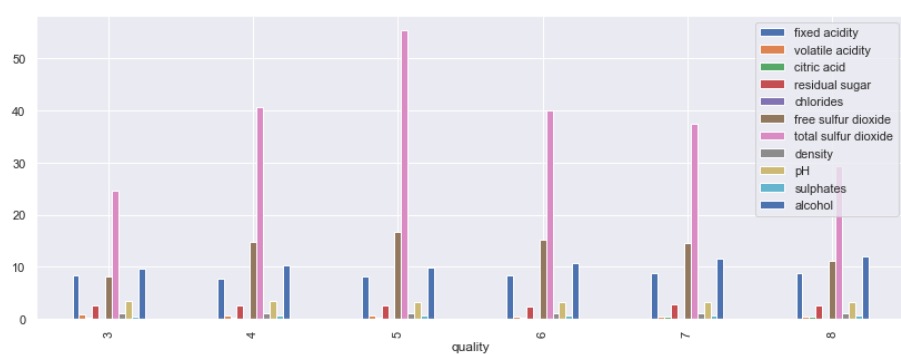


Figure A.27: Mean values for each wine quality

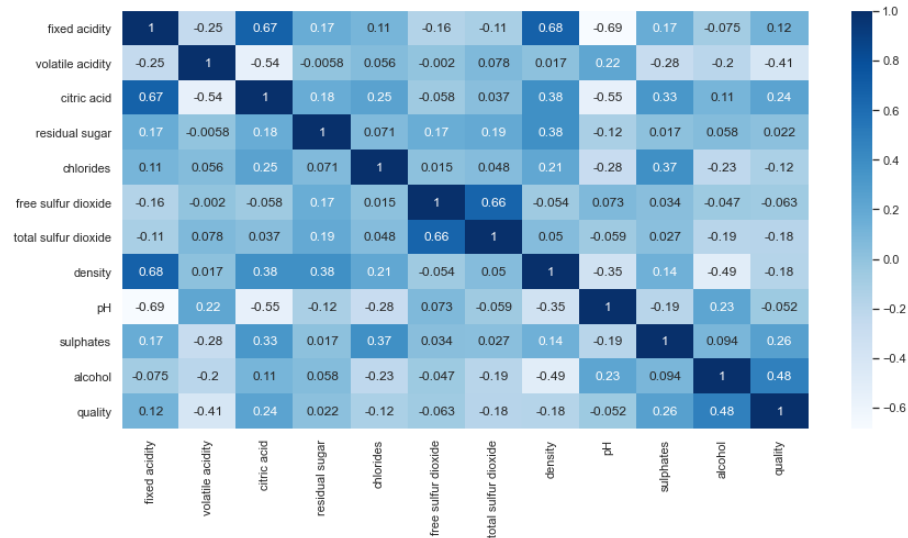


Figure A.28: Wine quality correlation

We can clearly conclude that the classes are unbalanced because there are more average wines than outstanding or terrible wines. It is also possible to conclude that there is no attribute that suffers a significant change when the quality of the wine rises or falls. As seen in the graphs of mean values and correlation, the only attributes that have a clear relationship with the wine's quality value are "alcohol" and "volatile acidity", with this relationship represented by small changes on the value of the attributes. We may conclude from the graph analysis that the quality of the wine tends to increase when the value of "alcohol" rises and the value of "volatile acidity" drops.



Figure A.29: Relation between alcohol and quality

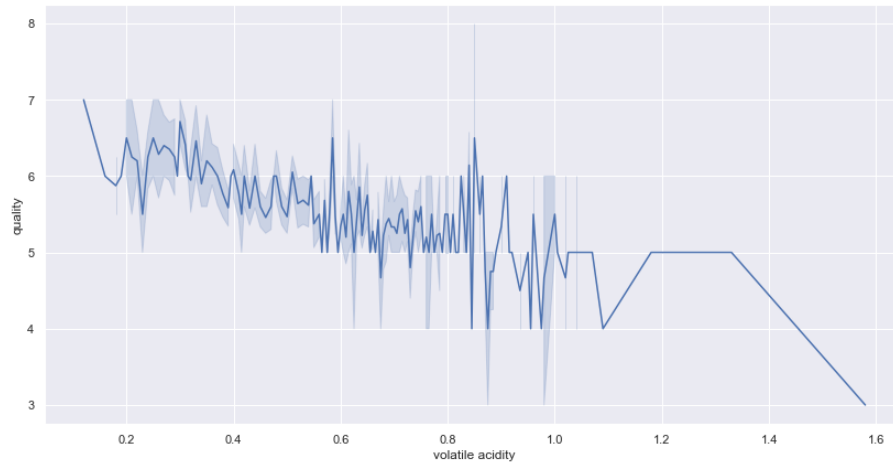


Figure A.30: Relation between volatile acidity and quality

As a result of this EDA, we will use the attributes "alcohol", and "volatile acidity" to conduct experiments that allow us to draw conclusions about the influence that certain circumstances have on model efficiency. In order to do that it was necessary to divide the attributes in bins. Each attribute was divided into 4 distinct bins, each one of those bins represents close to 25% of all data, so, every bin used as nearly the same amount of cases.

	fixed acidity	volatile acidity	acidity_bin	alcohol_bin	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	4	1	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	4	2	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	4	2	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	1	2	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	4	1	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

Figure A.31: Final state of the dataset

A.2.2 Salary

The primary goal of utilising this dataset is to forecast if an income surpasses \$50,000 per year based on census statistics. To accomplish so, we have access to 32561 different examples, each of which is originally characterised by 15 variables.

```

#   Column      Non-Null Count  Dtype
---  -
0   age         32561 non-null    int64
1   workclass    32561 non-null    object
2   fnlwgt       32561 non-null    int64
3   education    32561 non-null    object
4   education_num 32561 non-null    int64
5   marital_status 32561 non-null    object
6   occupation    32561 non-null    object
7   relationship  32561 non-null    object
8   ethnicity     32561 non-null    object
9   sex          32561 non-null    object
10  capital_gain  32561 non-null    int64
11  capital_loss  32561 non-null    int64
12  hrs_per_week  32561 non-null    int64
13  native_country 32561 non-null    object
14  salary        32561 non-null    object
dtypes: int64(6), object(9)

```

Figure A.32: Original salary data

We then started the data analyses by looking at the distribution of the target variable and the connections she had with the categorical attributes.

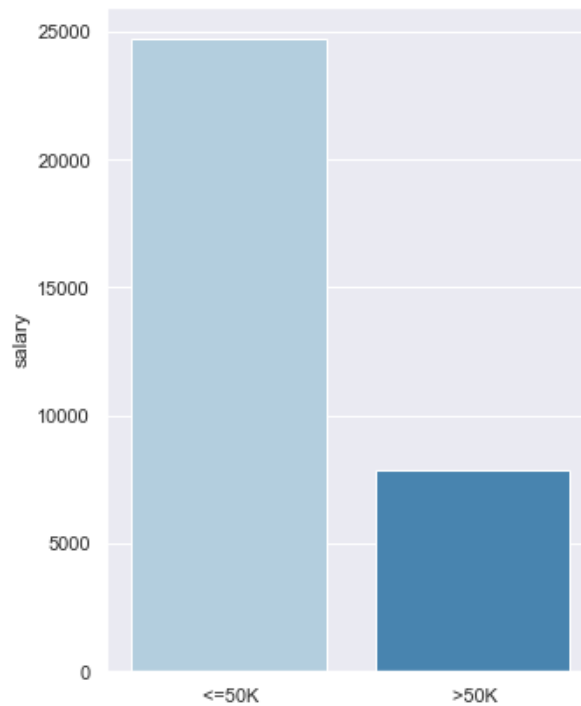


Figure A.33: Salary distribution

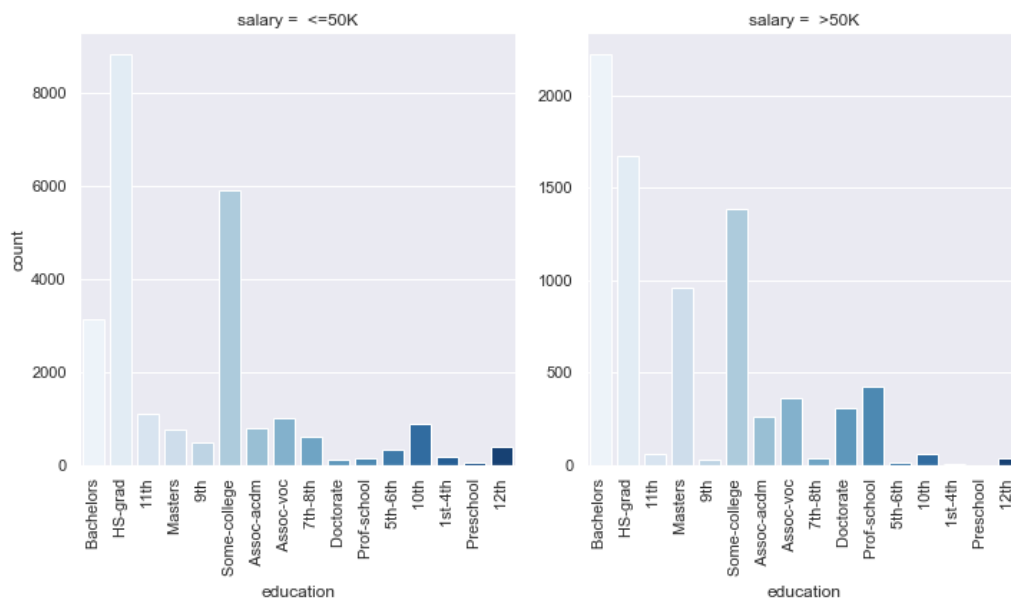


Figure A.34: Relation between education and salary

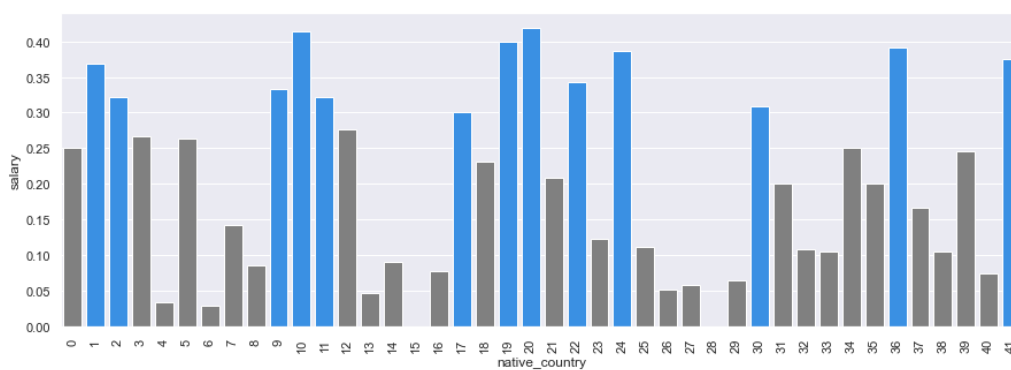


Figure A.35: Relation between native country and salary

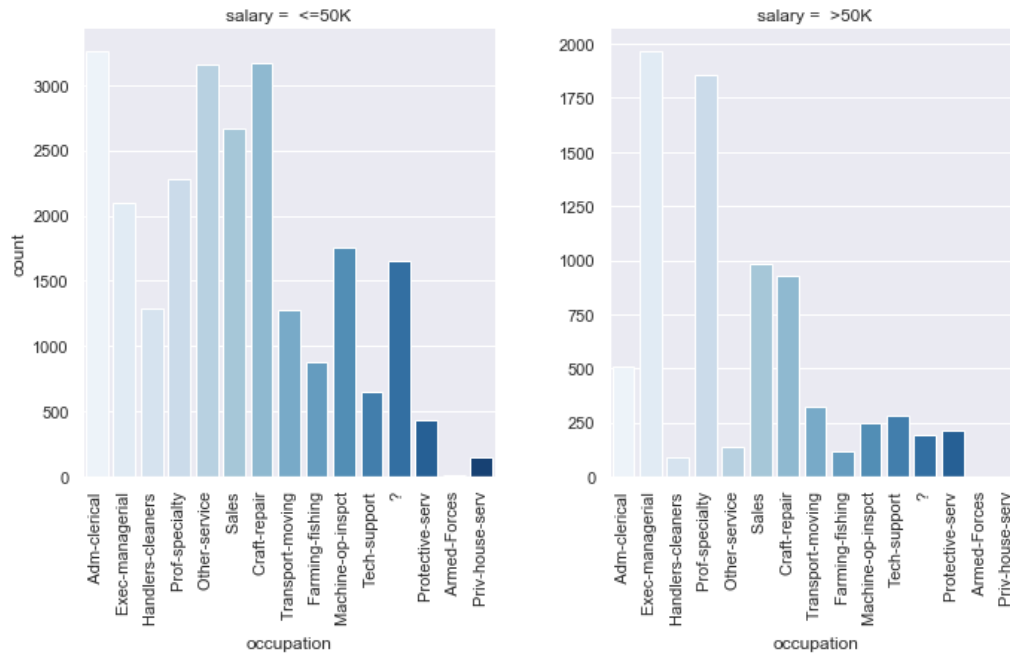


Figure A.36: Relation between occupation and salary

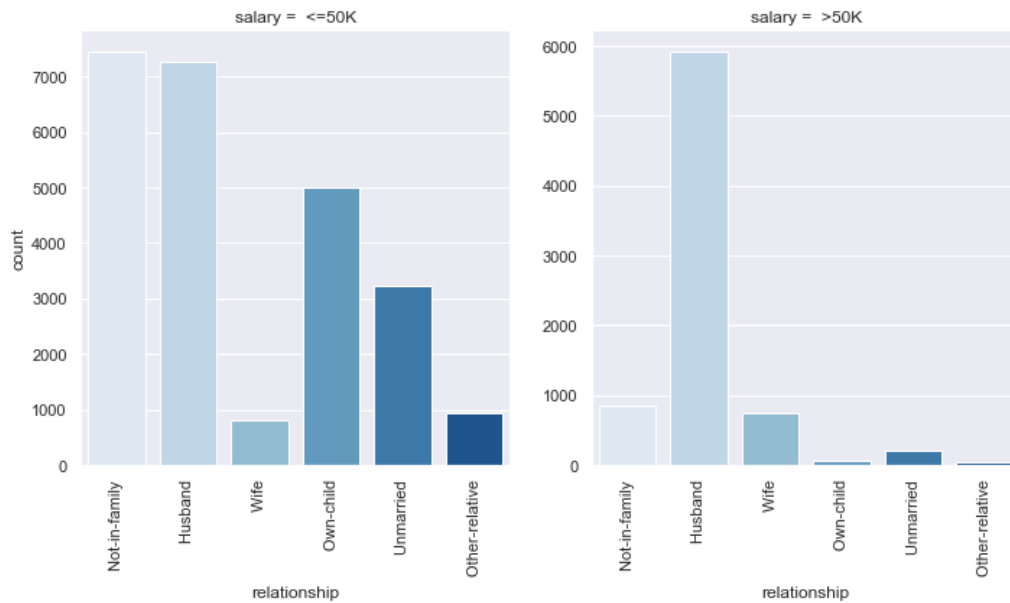


Figure A.37: Relation between relationship and salary

When we compare cases where the salary is less than or equal to 50000 to cases where the salary is greater than 50000, we find very different values in various attributes. The attributes "occupation", "native country", "relationship", and "education" are some of the attributes in which these distinctions occur. Following that, we wanted to know if any of the numerical attributes in the dataset influenced the salary level.

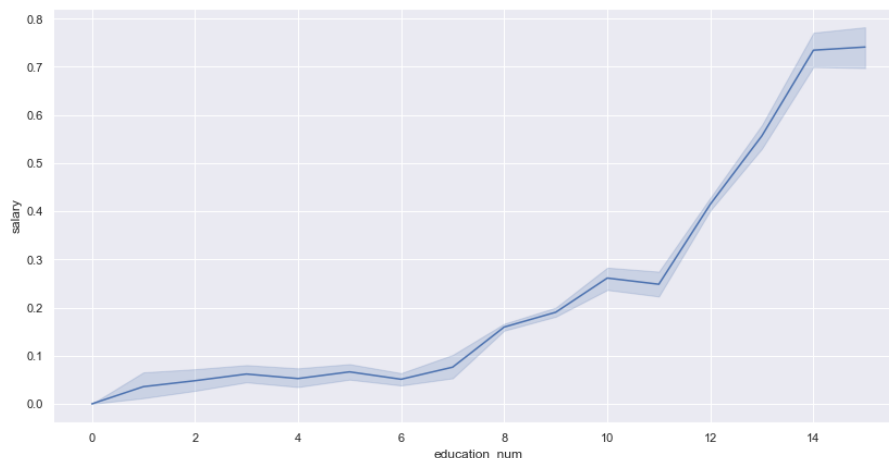


Figure A.38: Relation between education num and salary

As we can see from the figure above, "education_num" has a direct impact on the probability of a person receiving a salary higher than 50000 since, when "education_num" increases, the probability of a salary higher than 50000 also increases.

Because of all that was demonstrated about this dataset, we will use the attributes "education_num", "native_country", "education", "relationship" and "occupation" to conduct experiments that will allow us to draw conclusions about the influence that certain parameters have on model efficiency.

In order to prepare the dataset to be used by the models in a more efficient manner, it was at least necessary to:

1. Delete useless "fnlwgt" column
2. Change the columns with "object" types to "numerical" types.

#	Column	Non-Null Count	Dtype
0	age	32561 non-null	int64
1	workclass	32561 non-null	int8
2	education	32561 non-null	int8
3	education_num	32561 non-null	int8
4	marital_status	32561 non-null	int8
5	occupation	32561 non-null	int8
6	relationship	32561 non-null	int8
7	ethnicity	32561 non-null	int8
8	sex	32561 non-null	int8
9	capital_gain	32561 non-null	int64
10	capital_loss	32561 non-null	int64
11	hrs_per_week	32561 non-null	int64
12	native_country	32561 non-null	int8
13	salary	32561 non-null	int8

dtypes: int64(4), int8(10)

Figure A.39: Final state of the dataset

A.2.3 Heart disease

The biggest used classification dataset, initially consisting of 319795 distinct cases. Each case shows if a person has an heart disease. The goal of the model created with this dataset is to predict the value of the attribute "HeartDisease" with the highest accuracy possible.

```

#      Column      Non-Null Count  Dtype
---  -
0      HeartDisease    319795 non-null  object
1      BMI              319795 non-null  float64
2      Smoking          319795 non-null  object
3      AlcoholDrinking  319795 non-null  object
4      Stroke           319795 non-null  object
5      PhysicalHealth    319795 non-null  float64
6      MentalHealth     319795 non-null  float64
7      DiffWalking      319795 non-null  object
8      Sex              319795 non-null  object
9      AgeCategory      319795 non-null  object
10     Race              319795 non-null  object
11     Diabetic          319795 non-null  object
12     PhysicalActivity  319795 non-null  object
13     GenHealth         319795 non-null  object
14     SleepTime         319795 non-null  float64
15     Asthma            319795 non-null  object
16     KidneyDisease     319795 non-null  object
17     SkinCancer        319795 non-null  object
dtypes: float64(4), object(14)

```

Figure A.40: Original heart disease dataset

We then started the data analyses by looking at the distribution of the target variable and the connections she had with the attributes presented in the dataset.

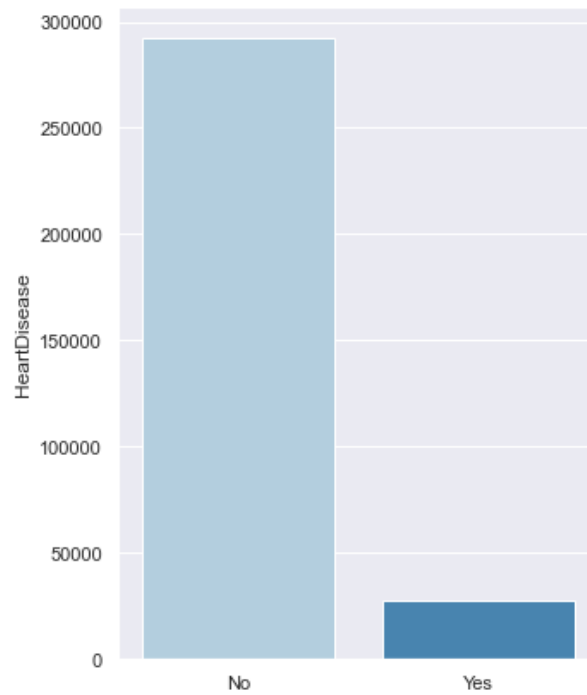


Figure A.41: HeartDisease distribution

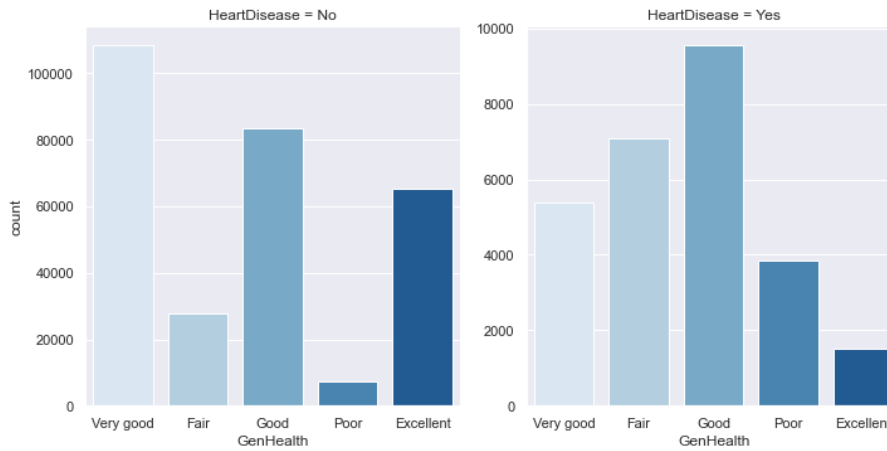


Figure A.42: Relation between GenHealth and HeartDisease

Initially, we found that there is a significant difference in the number of cases for the two distinct values of "HeartDisease" and that "GenHealth" has very different distributions based on the target variable value. Following that, we will consider the impact of "BMI", "AgeCategory", and "Diabetic" on the value assigned to "HeartDisease". To answer these questions, and in order to prepare the dataset for more effective usage by the models, it was required to:

1. Change the columns with "object" types to "numerical" types.
2. Create a new column (BMI_bin) with 4 different values.

3. Reorganize the columns such that the "HeartDisease" column, which represents the target value for our models, appears at the end of the dataset.

```

Data columns (total 19 columns):
#   Column              Non-Null Count  Dtype
---  -
0   BMI                  319795 non-null float64
1   BMI_bin              319795 non-null category
2   Smoking              319795 non-null int8
3   AlcoholDrinking      319795 non-null int8
4   Stroke               319795 non-null int8
5   PhysicalHealth        319795 non-null float64
6   MentalHealth          319795 non-null float64
7   DiffWalking          319795 non-null int8
8   Sex                  319795 non-null int8
9   AgeCategory          319795 non-null int8
10  Race                 319795 non-null int8
11  Diabetic              319795 non-null int8
12  PhysicalActivity      319795 non-null int8
13  GenHealth             319795 non-null int8
14  SleepTime            319795 non-null category
15  Asthma               319795 non-null int8
16  KidneyDisease         319795 non-null int8
17  SkinCancer            319795 non-null int8
18  HeartDisease          319795 non-null int8
dtypes: category(2), float64(3), int8(14)

```

Figure A.43: Cleaned heart disease dataset

We created graphs that show the likelihood of a case including "HeartDisease" for each of the previously mentioned attributes.

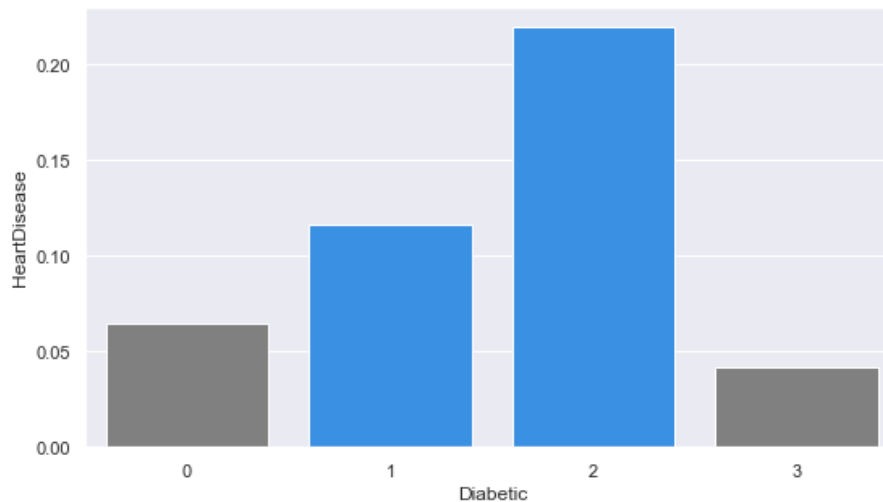


Figure A.44: Relation between Diabetic and HeartDisease

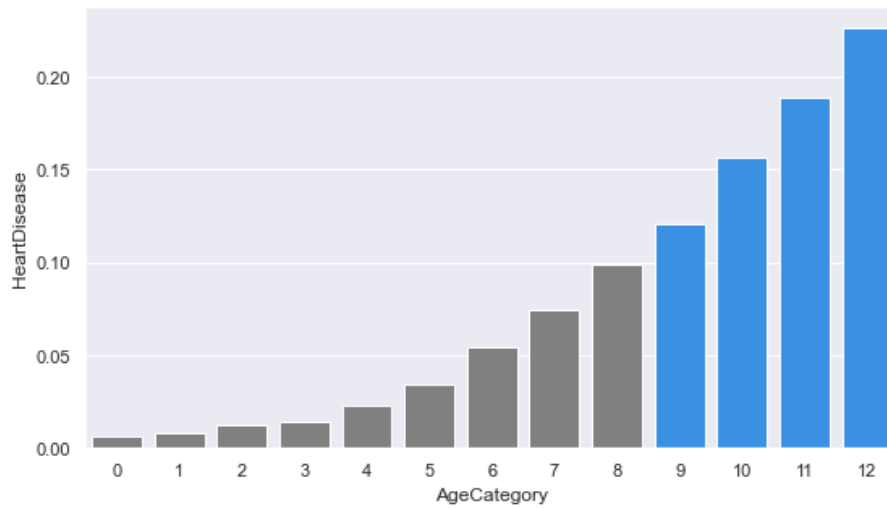


Figure A.45: Relation between AgeCategory and HeartDisease

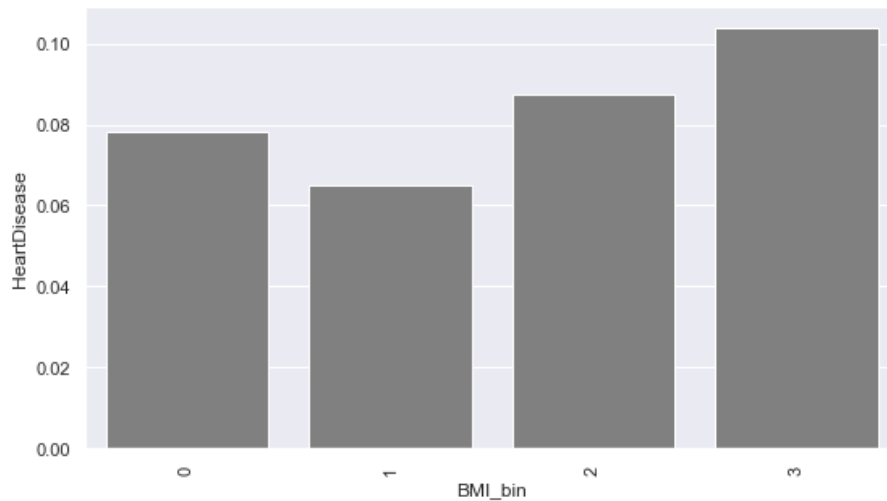


Figure A.46: Relation between BMI and HeartDisease

It is easy to conclude from the created visual aid that the attributes "AgeCategory" and "Diabetic" have a significantly influence on the value of "heartDisease", whereas the attribute "BMI_bin" (created from the values of "BMI") does not cause a significant fluctuation in the value of "heartDisease".

we will use the attributes "GenHealth", "Diabetic", "AgeCategory", and "BMI_bin" to conduct experiments that will allow us to draw conclusions about the influence that certain parameters have on model efficiency.

Appendix B

Regression results charts

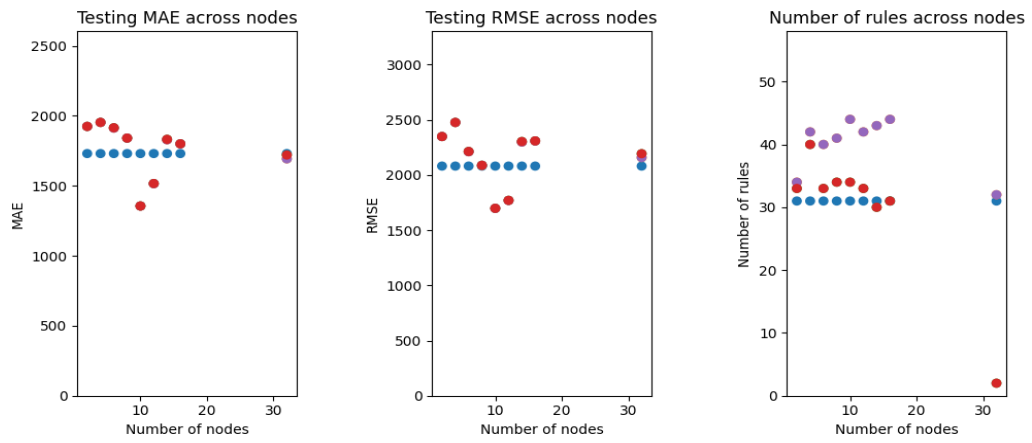


Figure B.1: Random split on Diamonds dataset

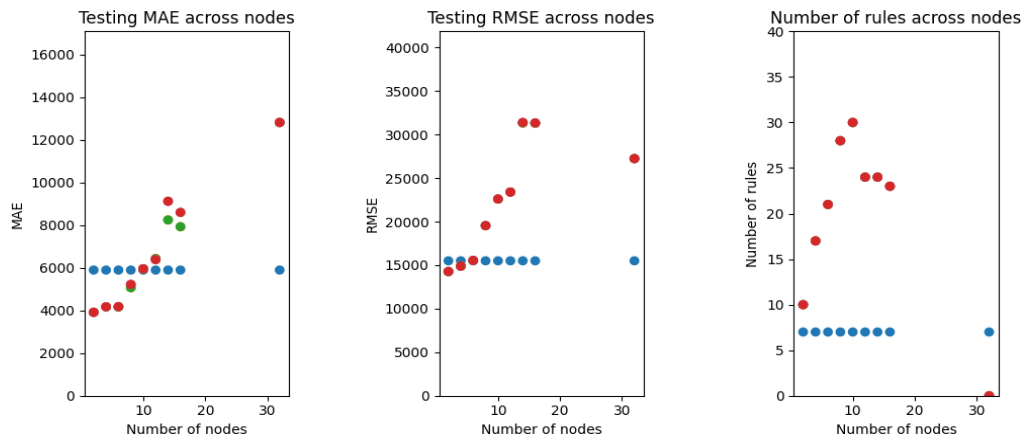


Figure B.2: Random split on Traffic deaths dataset

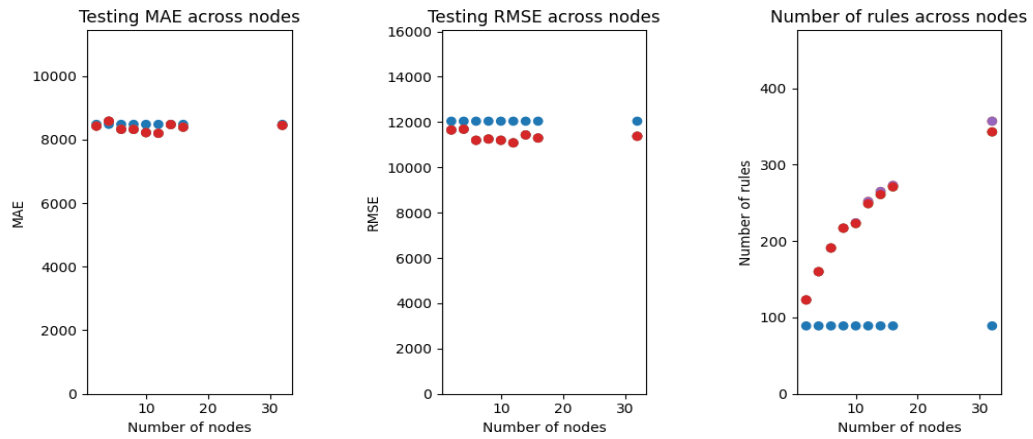


Figure B.3: Random split on Used cars dataset

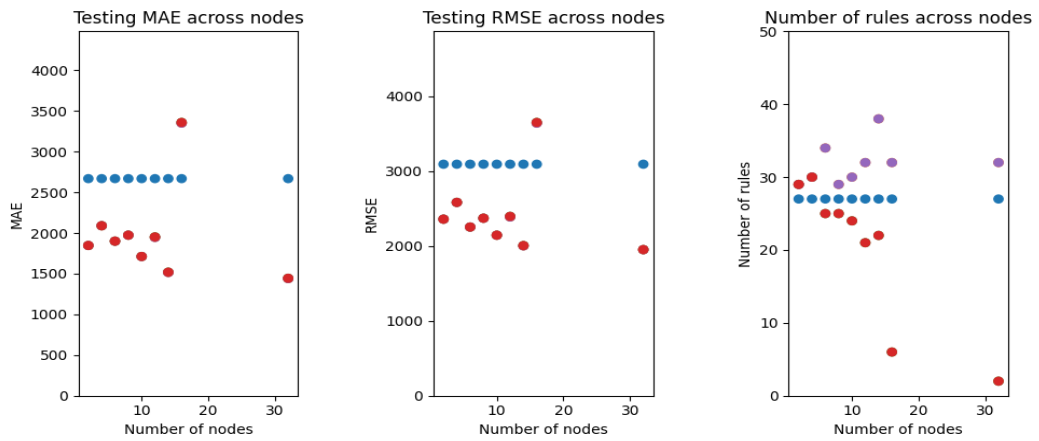


Figure B.4: Target split on Diamonds dataset

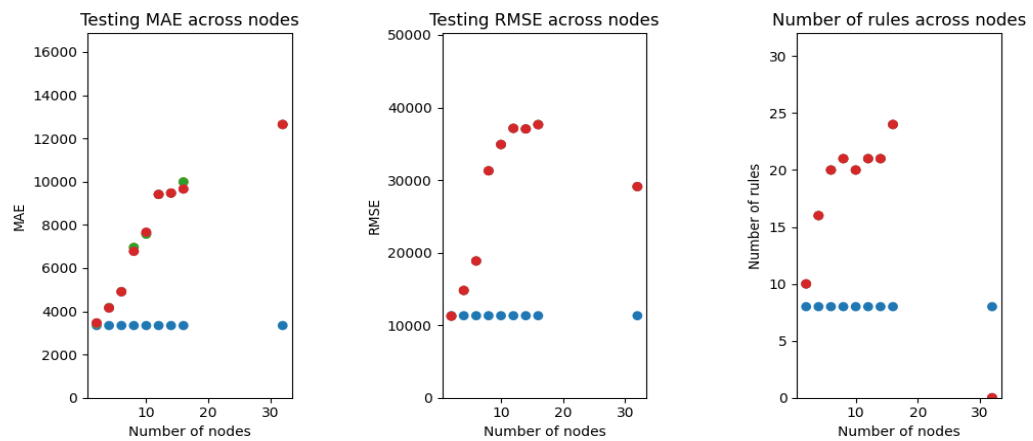


Figure B.5: Target split on Traffic deaths dataset

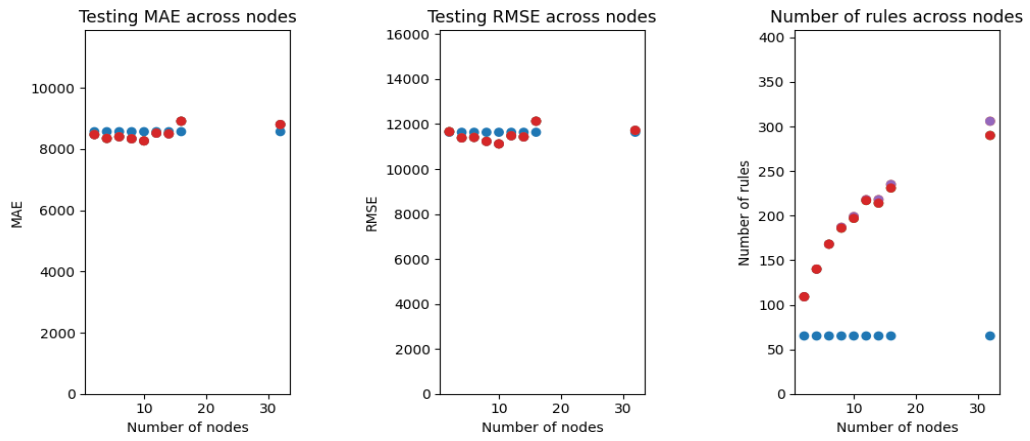


Figure B.6: Target split on Used cars dataset

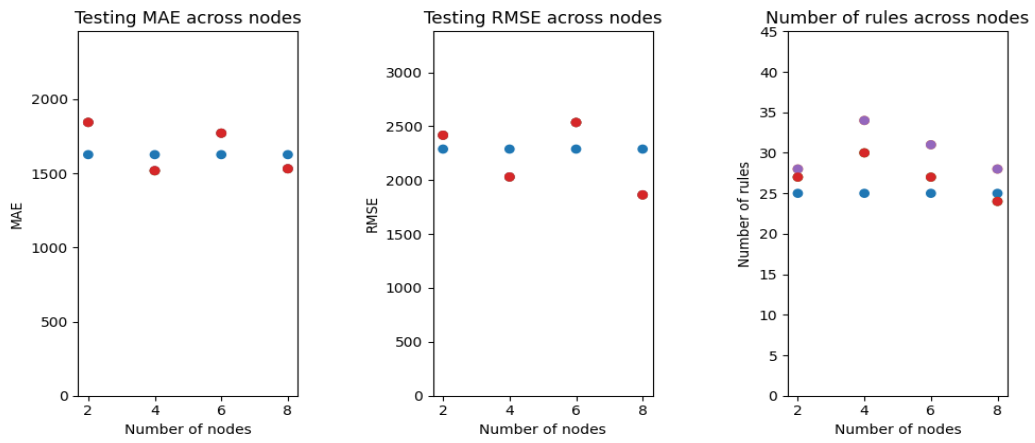


Figure B.7: Attribute split on Diamonds dataset using carat with 66% probability

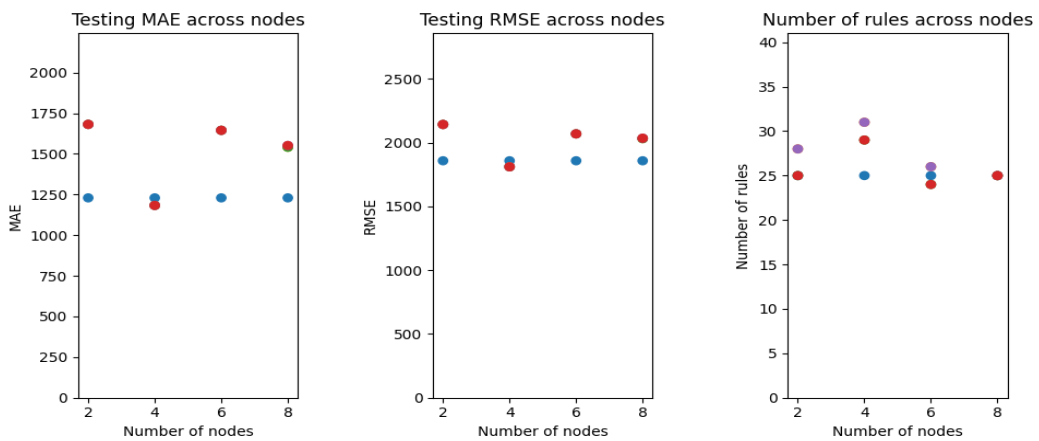


Figure B.8: Attribute split on Diamonds dataset using carat with 80% probability

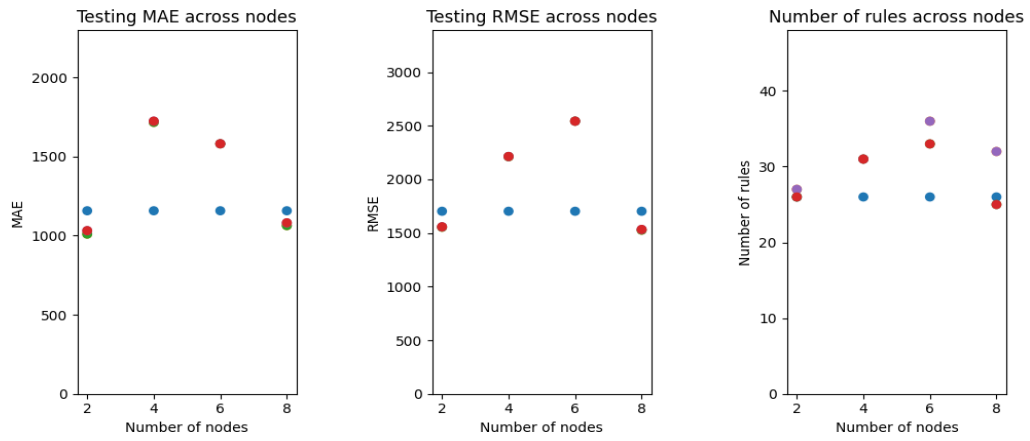


Figure B.9: Attribute split on Diamonds dataset using carat with 100% probability

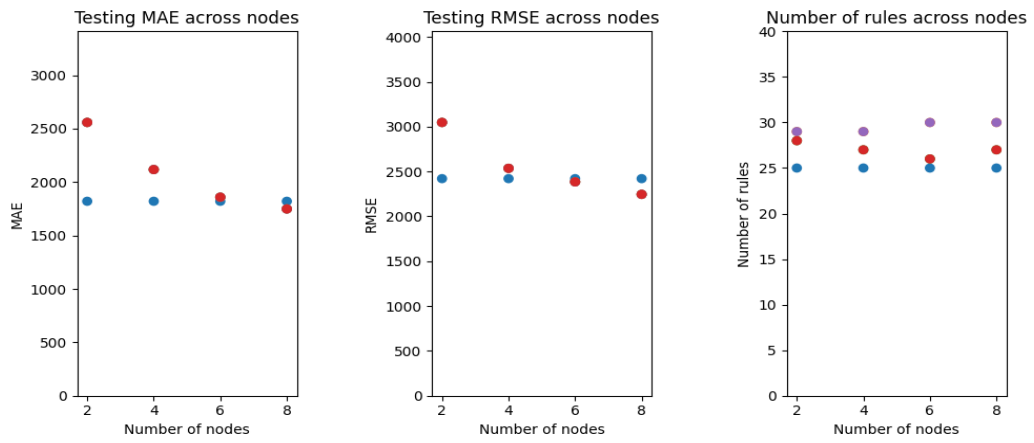


Figure B.10: Attribute split on Diamonds dataset using clarity with 66% probability

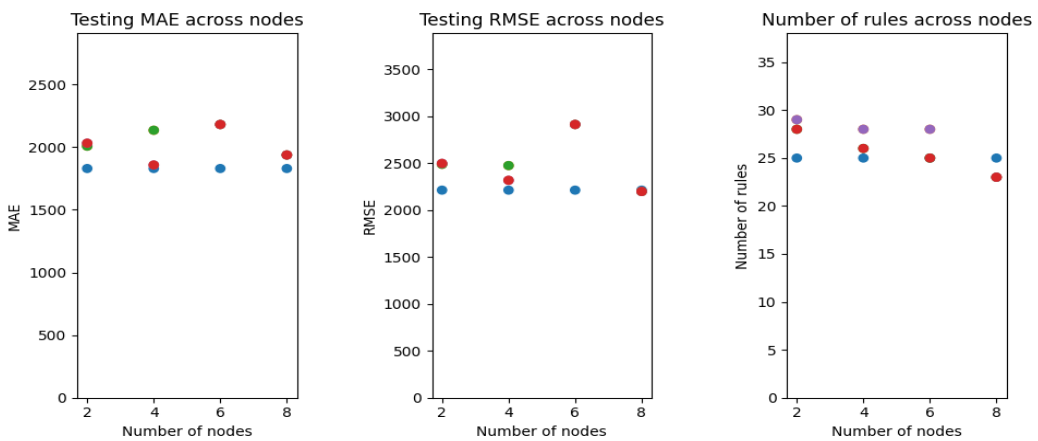


Figure B.11: Attribute split on Diamonds dataset using clarity with 80% probability

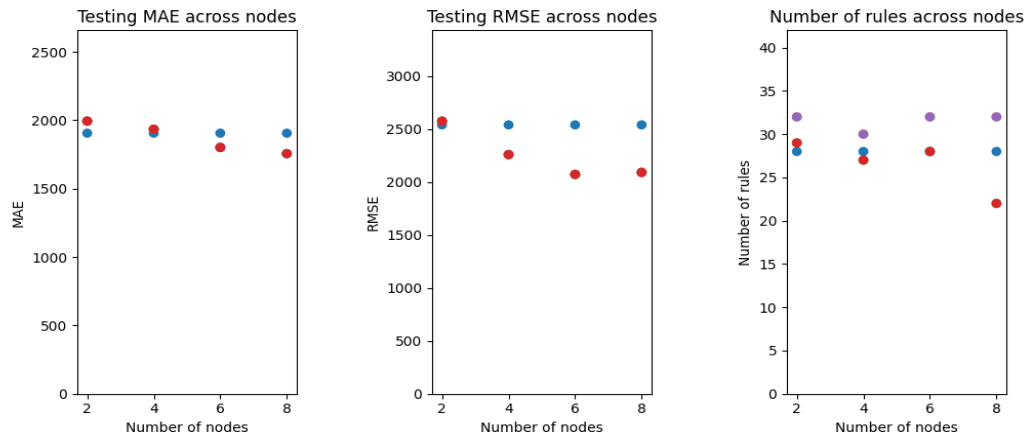


Figure B.12: Attribute split on Diamonds dataset using clarity with 100% probability

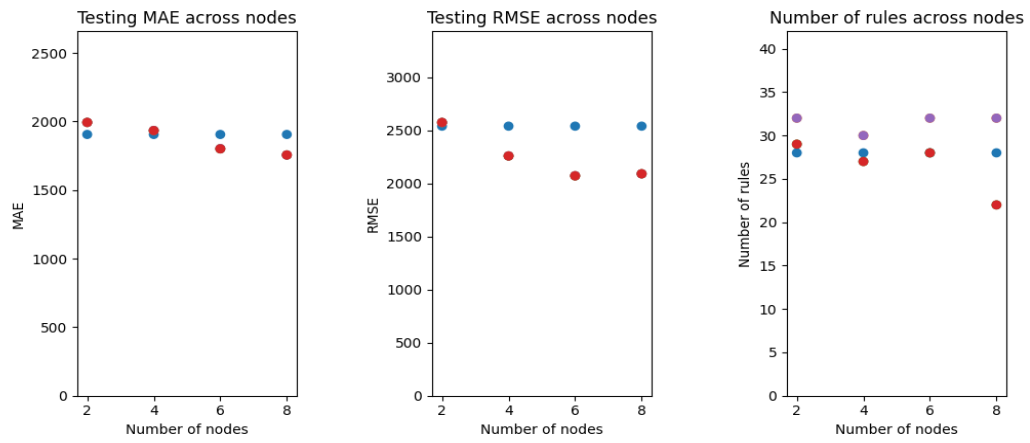


Figure B.13: Attribute split on Diamonds dataset using clarity with 100% probability

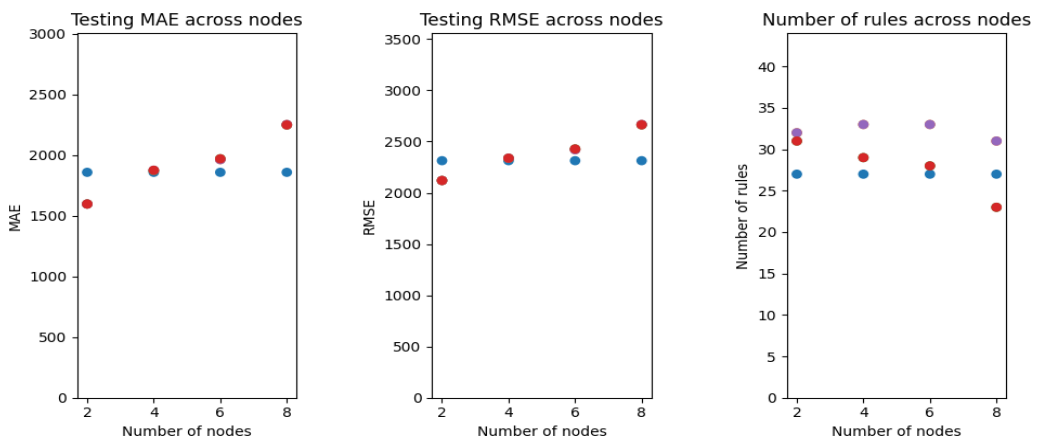


Figure B.14: Attribute split on Diamonds dataset using color with 66% probability

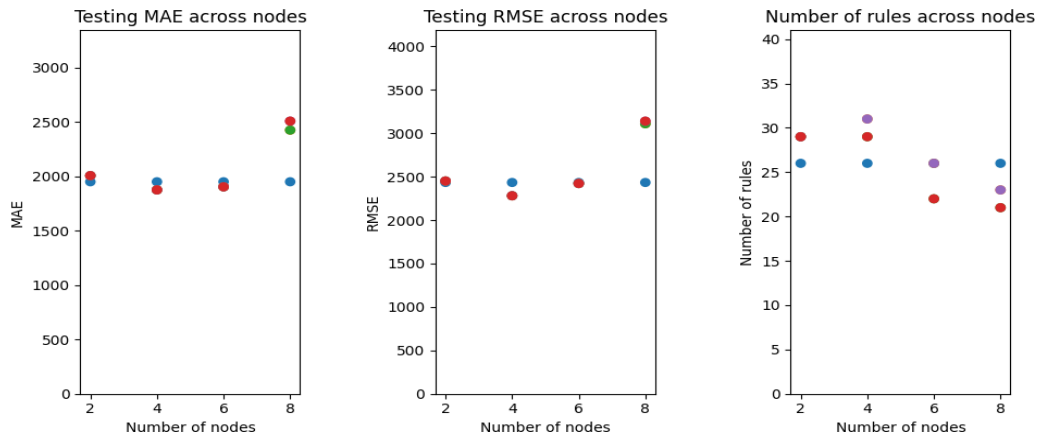


Figure B.15: Attribute split on Diamonds dataset using color with 80% probability

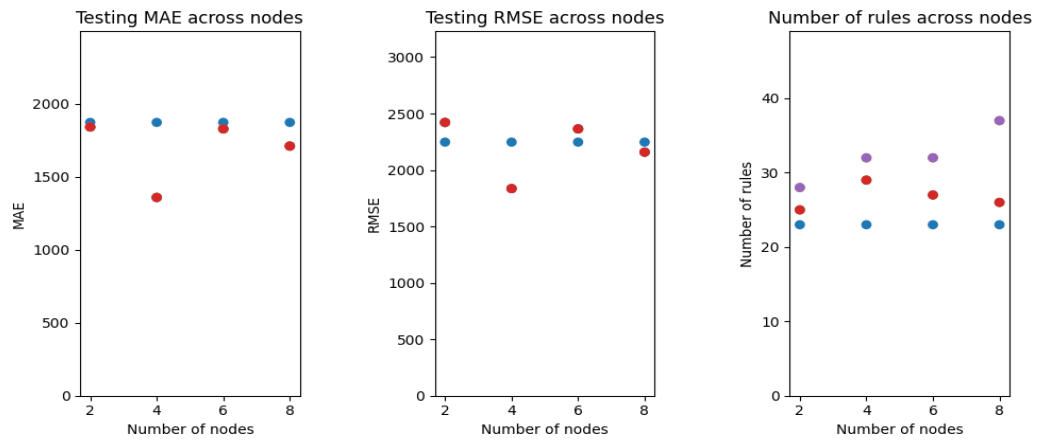


Figure B.16: Attribute split on Diamonds dataset using color with 100% probability

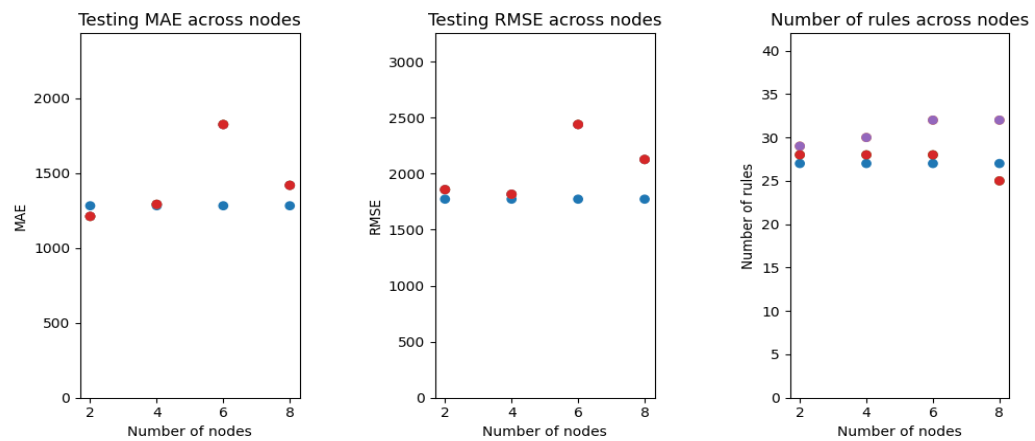


Figure B.17: Attribute split on Diamonds dataset using x with 66% probability

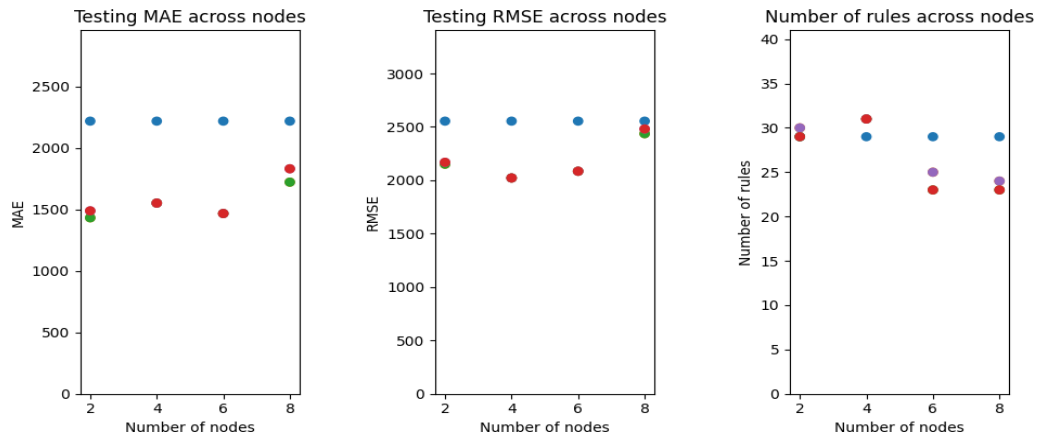


Figure B.18: Attribute split on Diamonds dataset using x with 80% probability

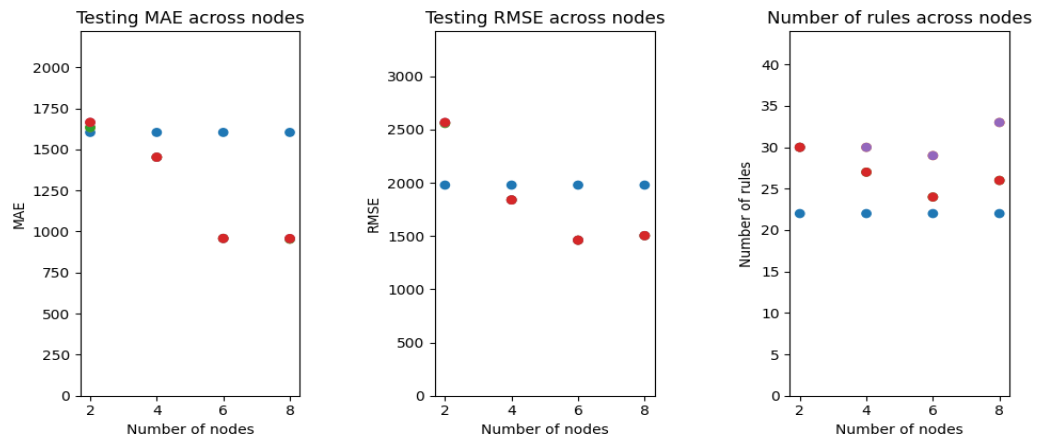


Figure B.19: Attribute split on Diamonds dataset using x with 100% probability

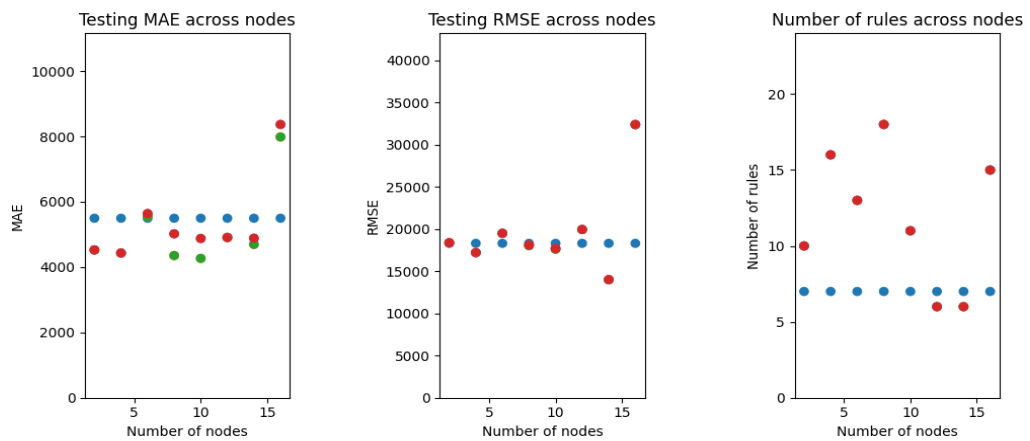


Figure B.20: Attribute split on Traffic deaths dataset using $Entity$ with 66% probability

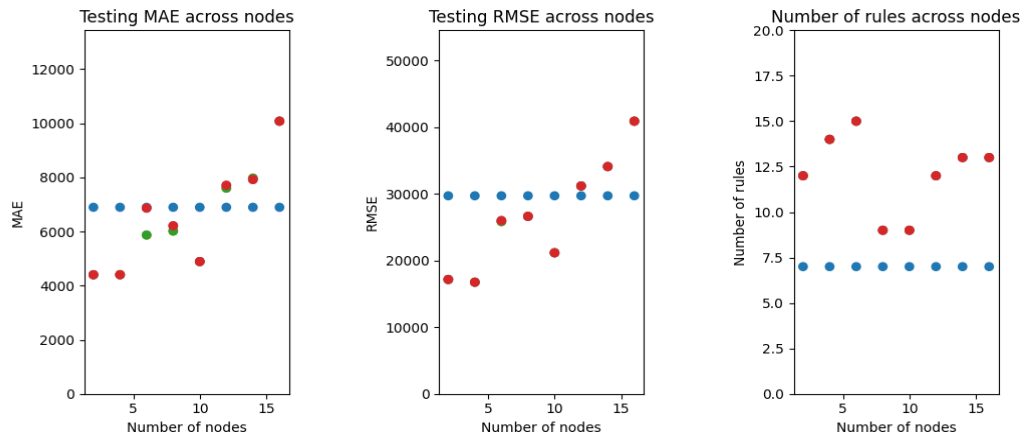


Figure B.21: Attribute split on Traffic deaths dataset using Entity with 80% probability

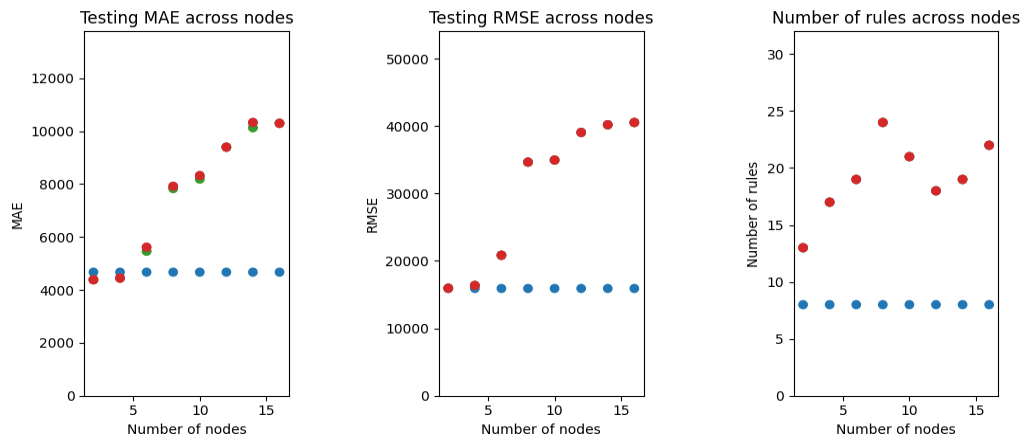


Figure B.22: Attribute split on Traffic deaths dataset using Entity with 100% probability

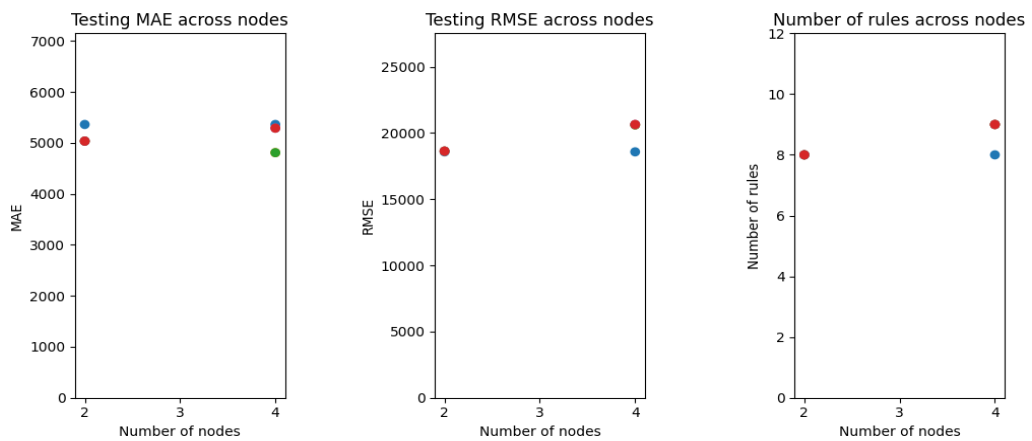


Figure B.23: Attribute split on Traffic deaths dataset using pop_bin with 66% probability

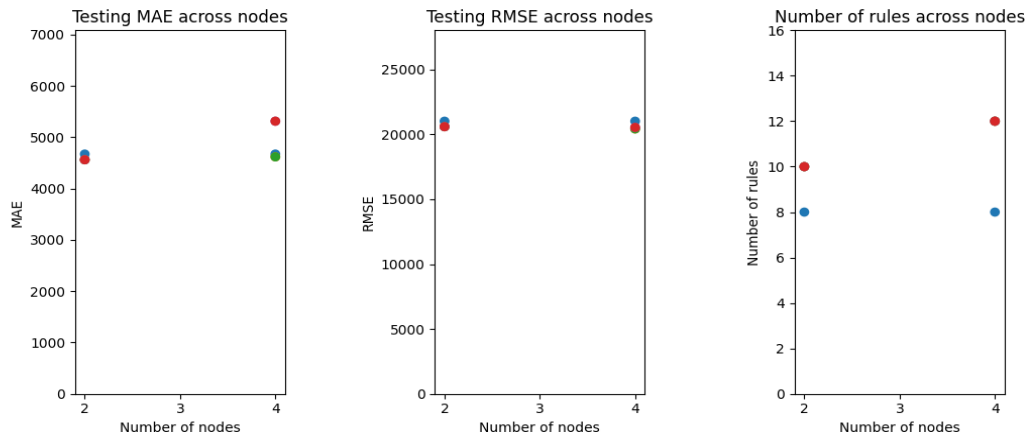


Figure B.24: Attribute split on Traffic deaths dataset using pop_bin with 80% probability

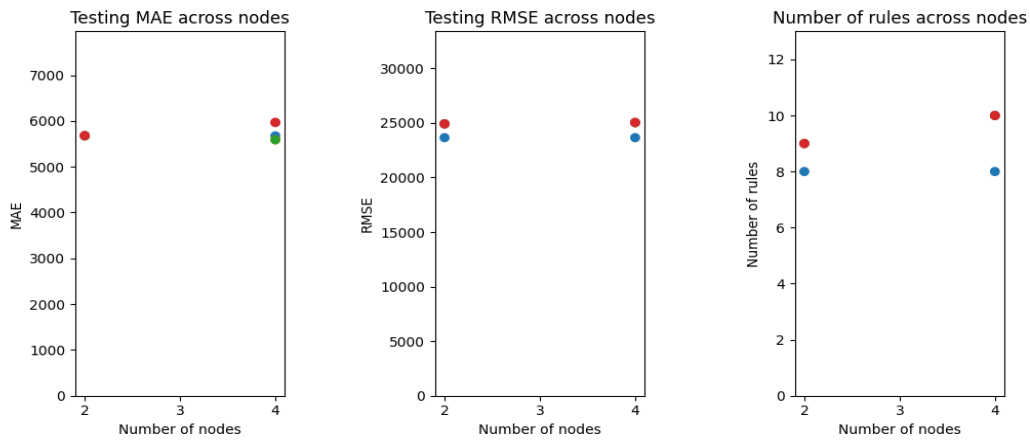


Figure B.25: Attribute split on Traffic deaths dataset using pop_bin with 100% probability

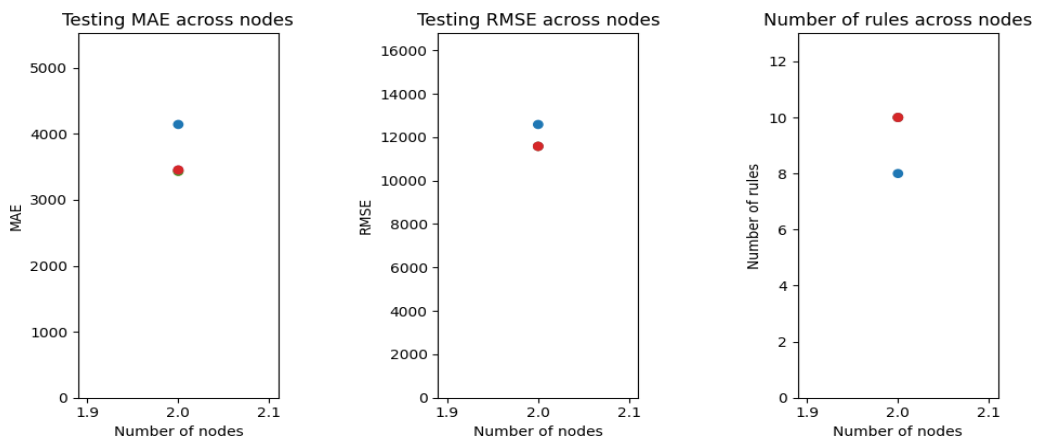


Figure B.26: Attribute split on Traffic deaths dataset using Sidedness with 66% probability

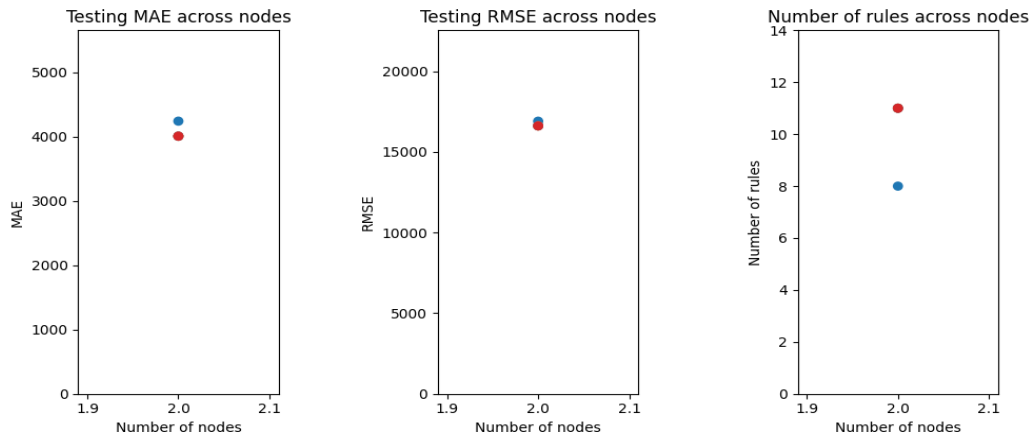


Figure B.27: Attribute split on Traffic deaths dataset using Sidedness with 80% probability

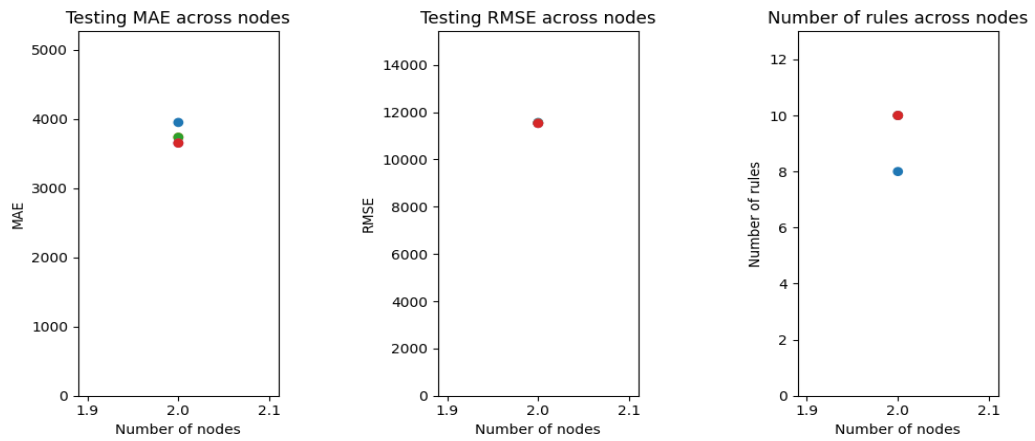
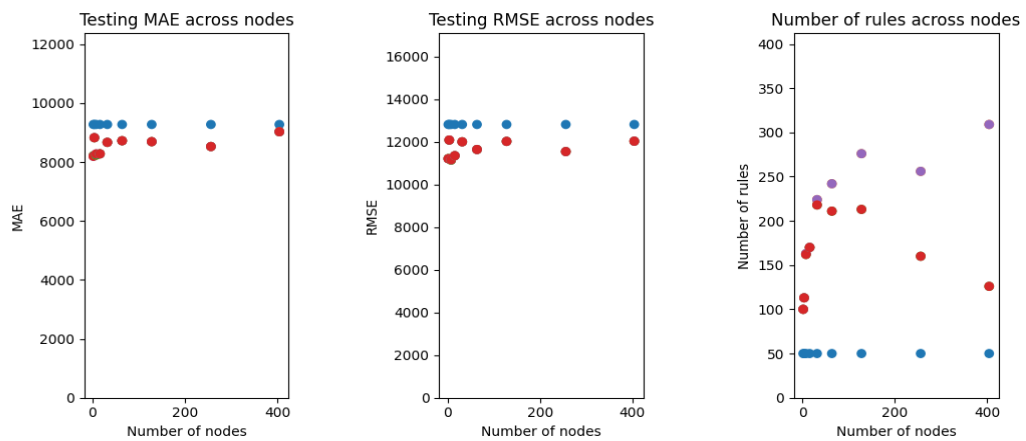


Figure B.28: Attribute split on Traffic deaths dataset using Sidedness with 100% probability



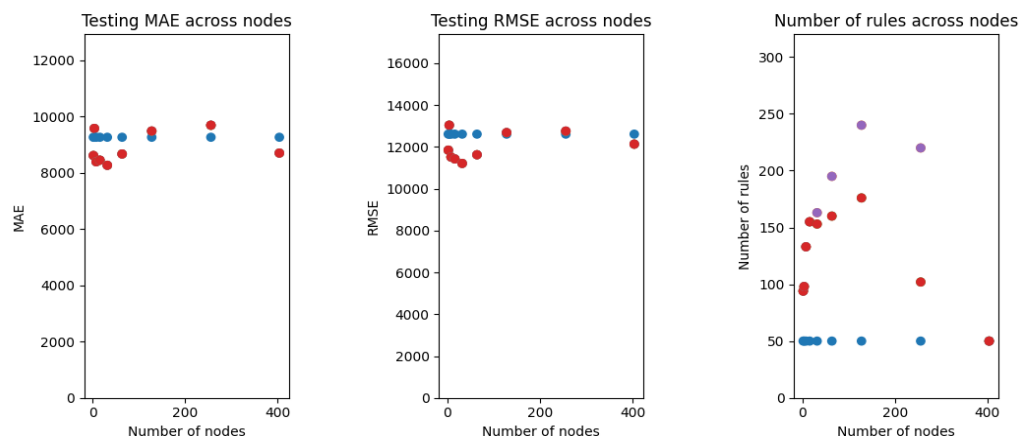


Figure B.30: Attribute split on Used cars dataset using region with 80% probability

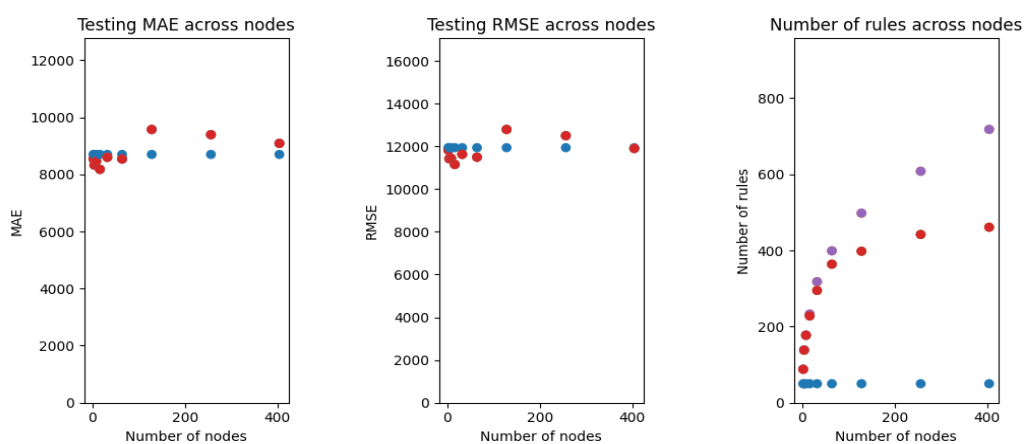


Figure B.31: Attribute split on Used cars dataset using region with 100% probability

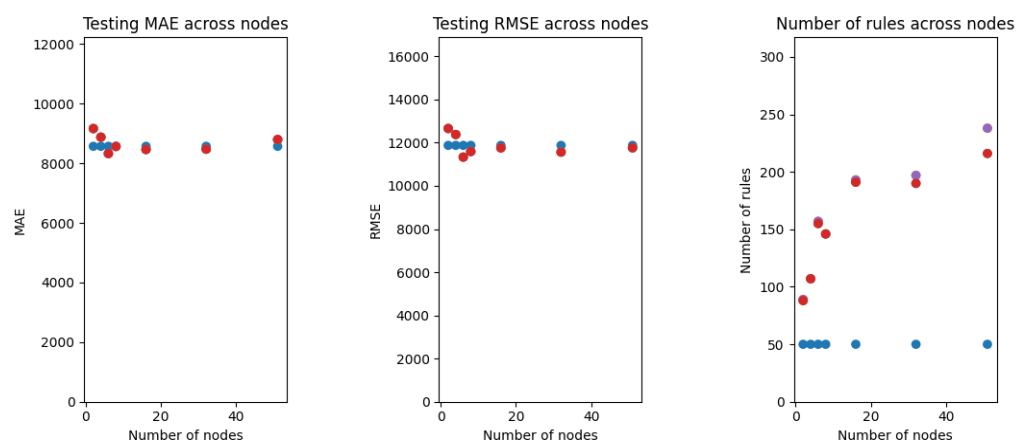


Figure B.32: Attribute split on Used cars dataset using state with 66% probability

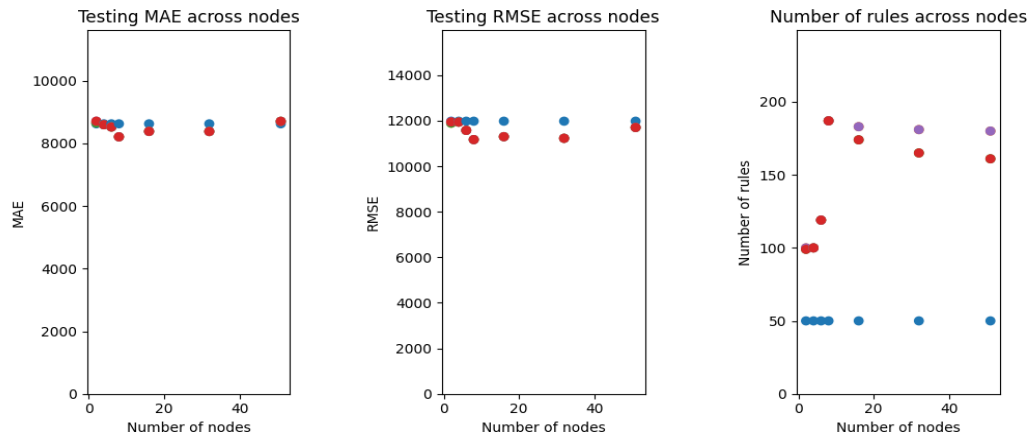


Figure B.33: Attribute split on Used cars dataset using state with 80% probability

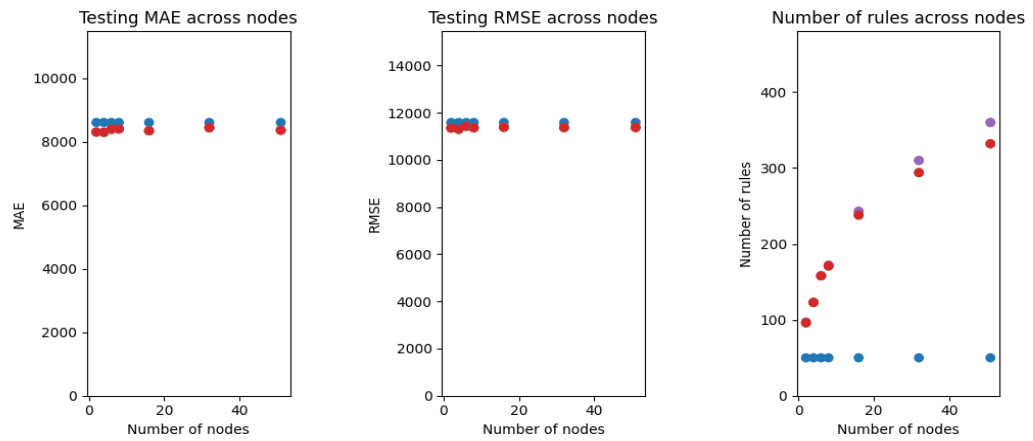


Figure B.34: Attribute split on Used cars dataset using state with 100% probability

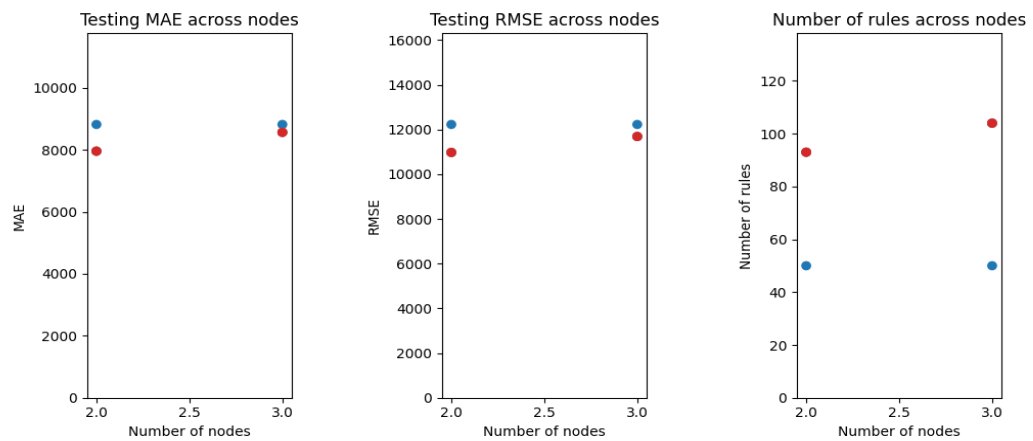


Figure B.35: Attribute split on Used cars dataset using transmission with 66% probability

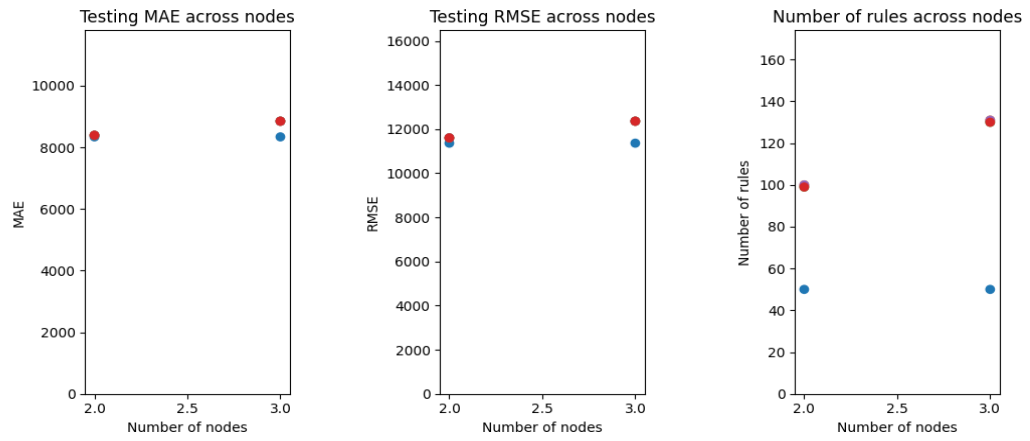


Figure B.36: Attribute split on Used cars dataset using transmission with 80% probability

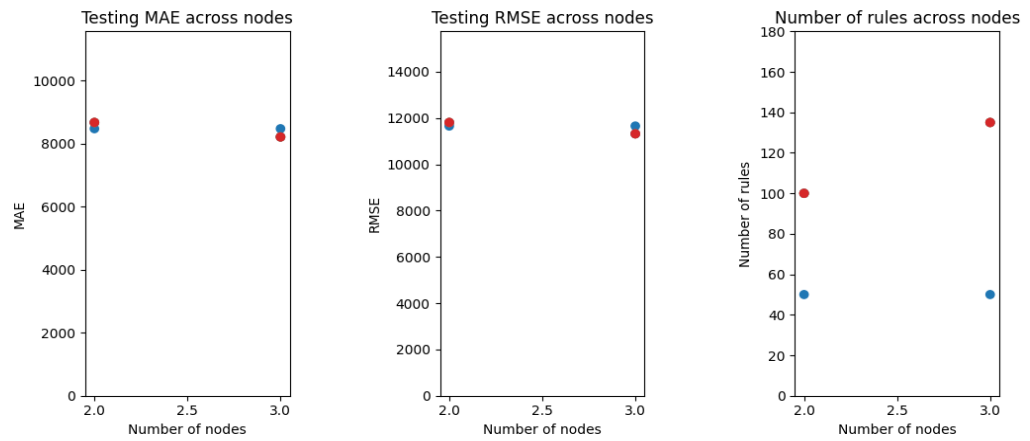


Figure B.37: Attribute split on Used cars dataset using transmission with 100% probability

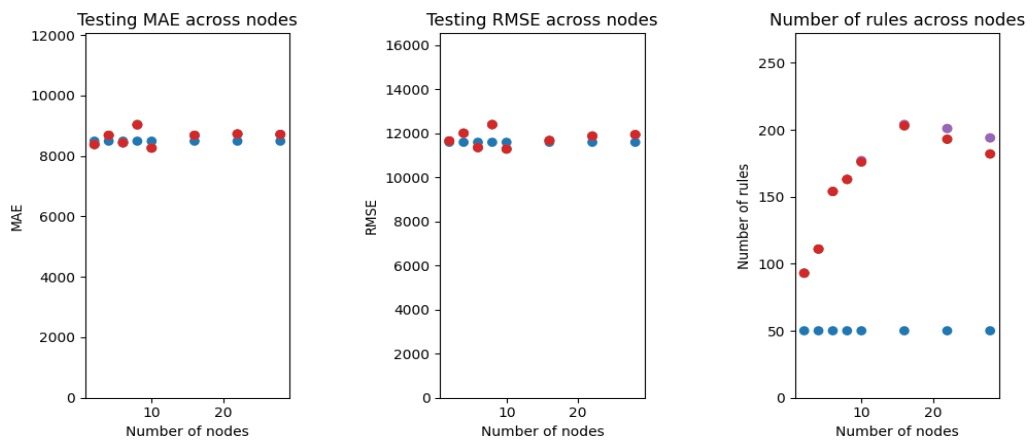


Figure B.38: Attribute split on Used cars dataset using year with 66% probability

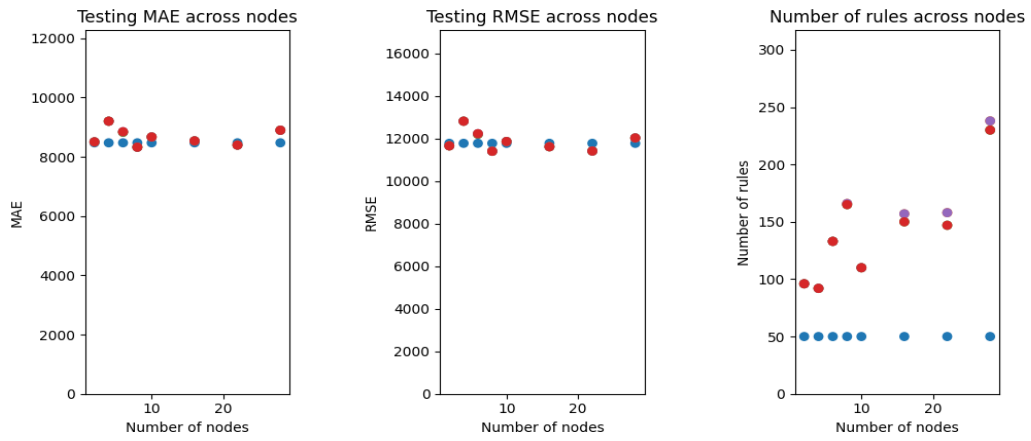


Figure B.39: Attribute split on Used cars dataset using year with 80% probability

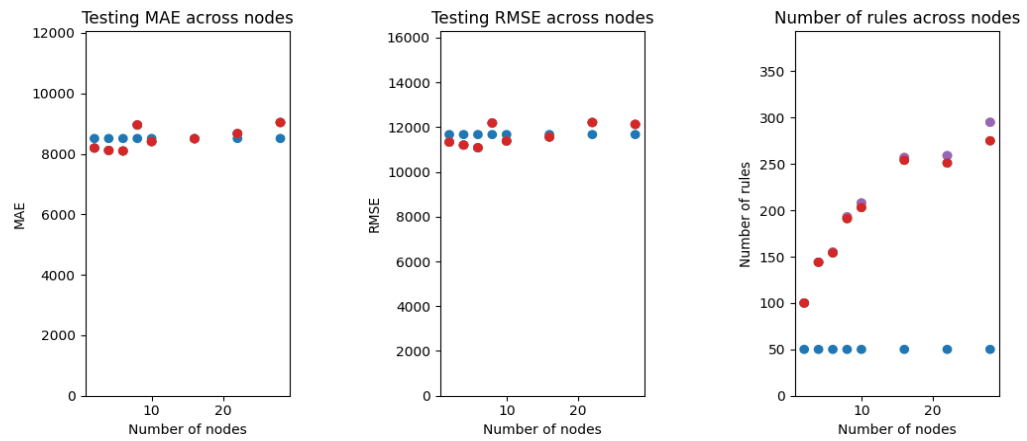


Figure B.40: Attribute split on Used cars dataset using year with 100% probability

Appendix C

Classification results charts

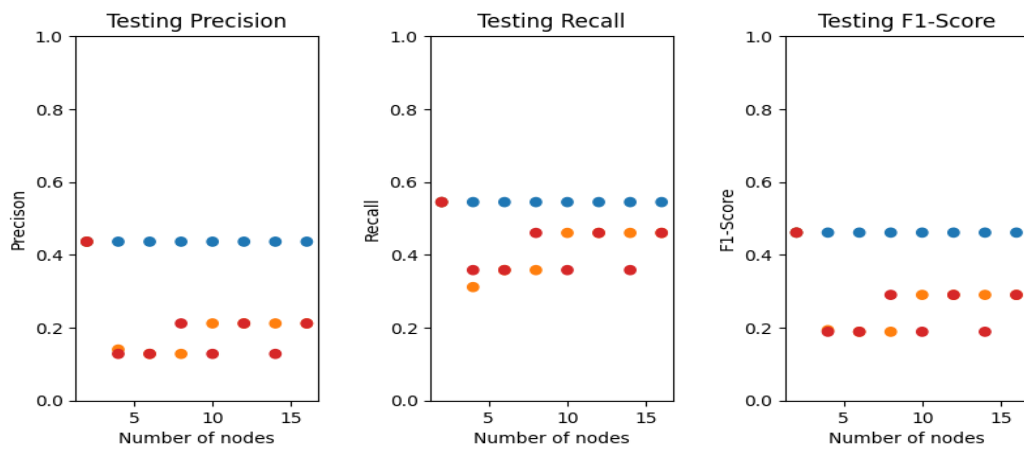


Figure C.1: Random split on Wine dataset

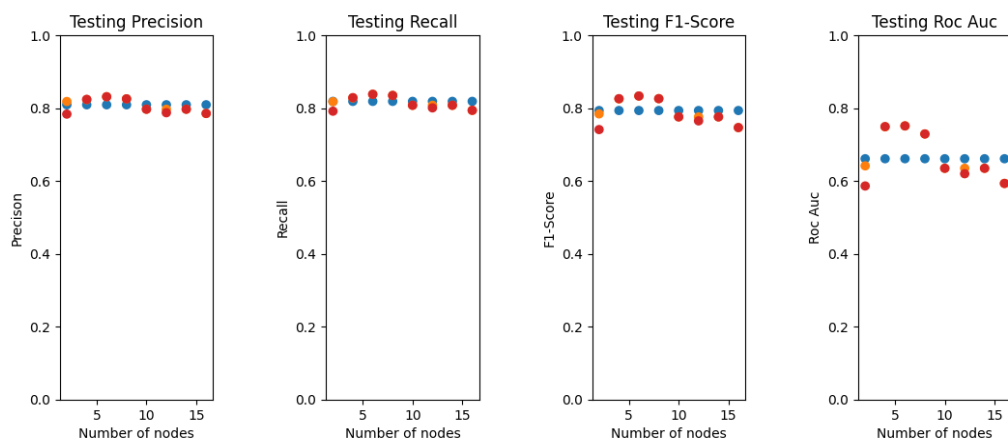


Figure C.2: Random split on Salary dataset

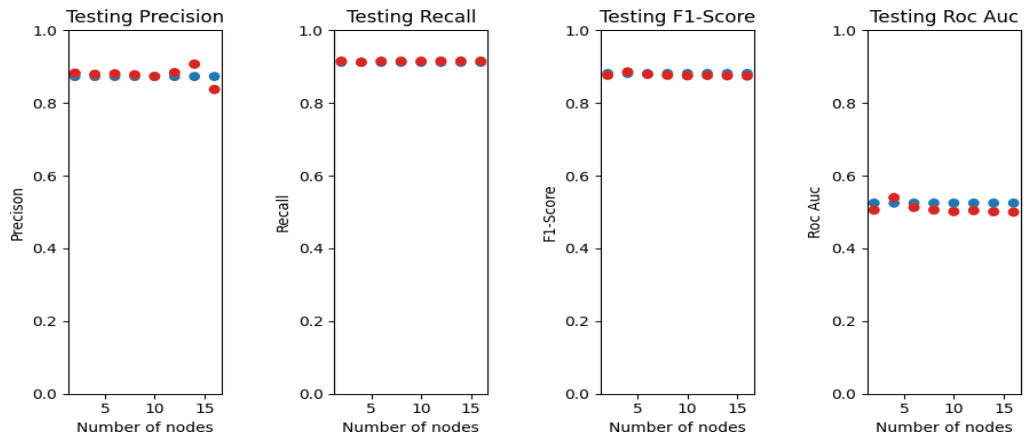


Figure C.3: Random split on Heart disease dataset

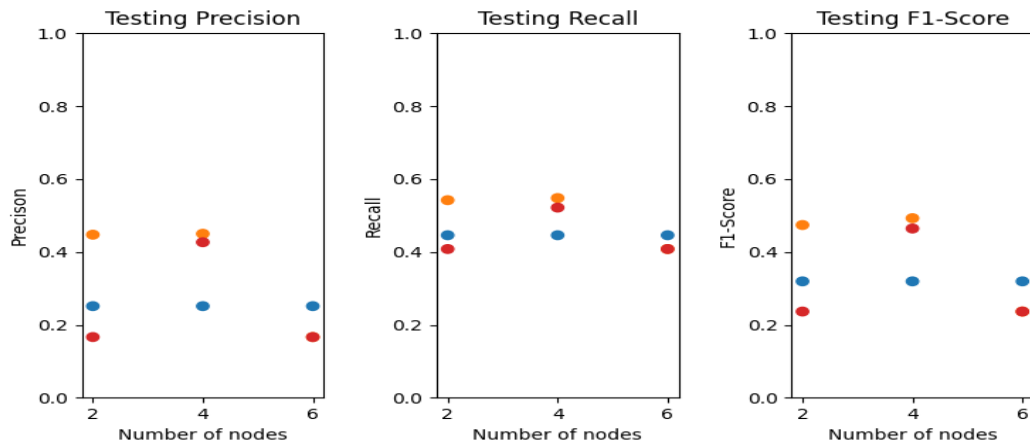


Figure C.4: Target split on Wine dataset

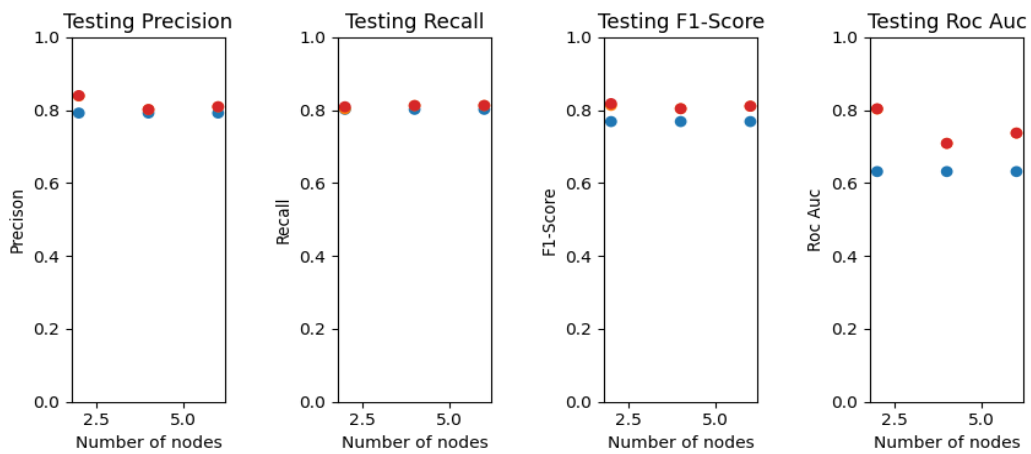


Figure C.5: Target split on Salary dataset

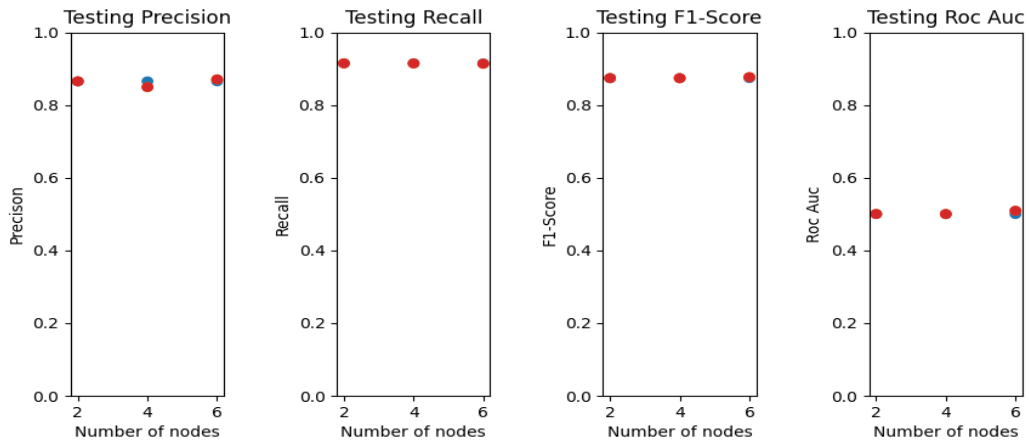


Figure C.6: Target split on Heart disease dataset

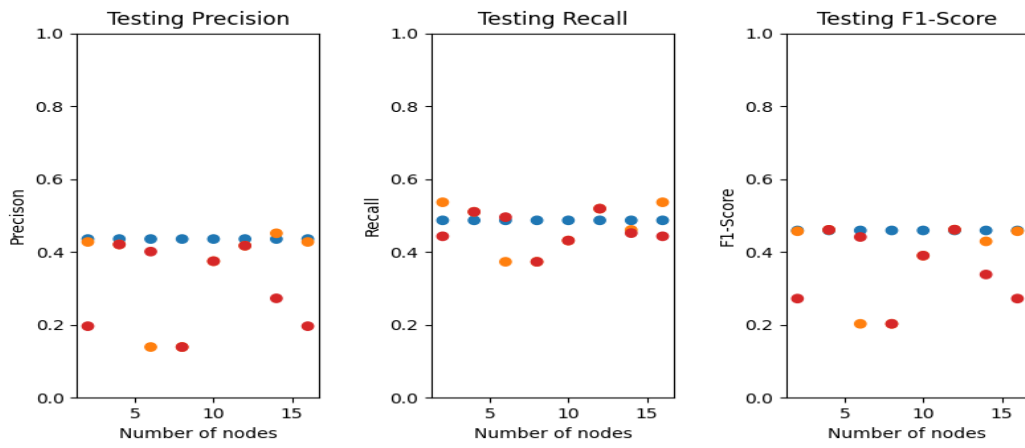


Figure C.7: Attribute split on Wine dataset using acidity with 66% probability

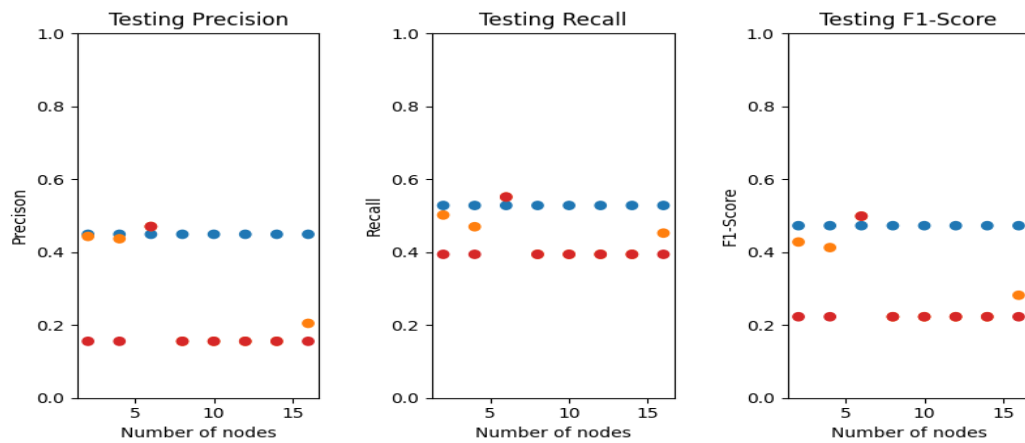


Figure C.8: Attribute split on Wine dataset using acidity with 80% probability

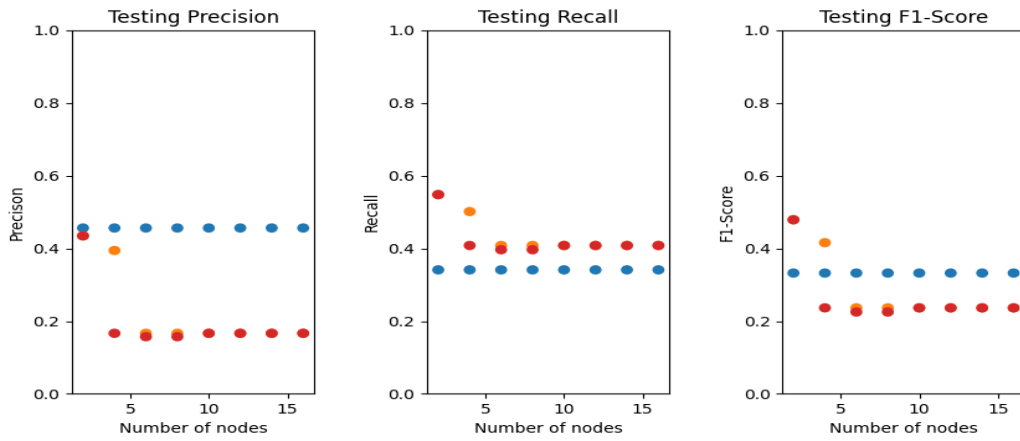


Figure C.9: Attribute split on Wine dataset using acidity with 100% probability

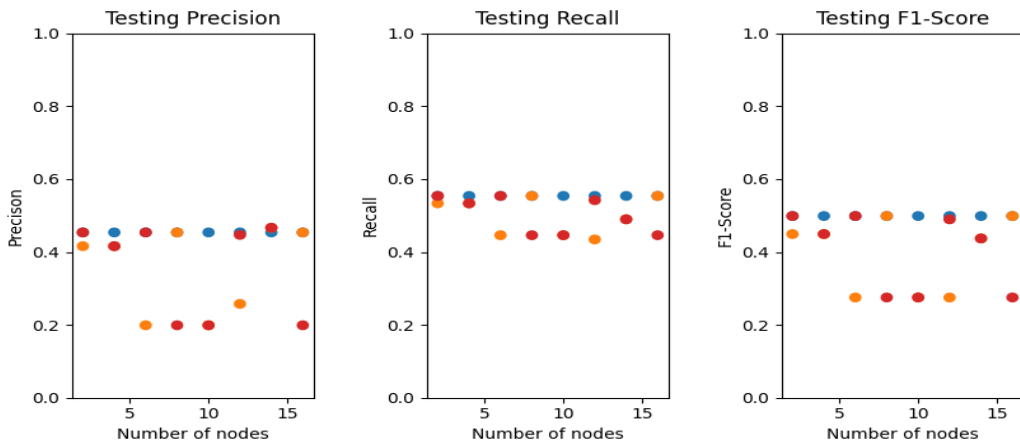


Figure C.10: Attribute split on Wine dataset using alcohol with 66% probability

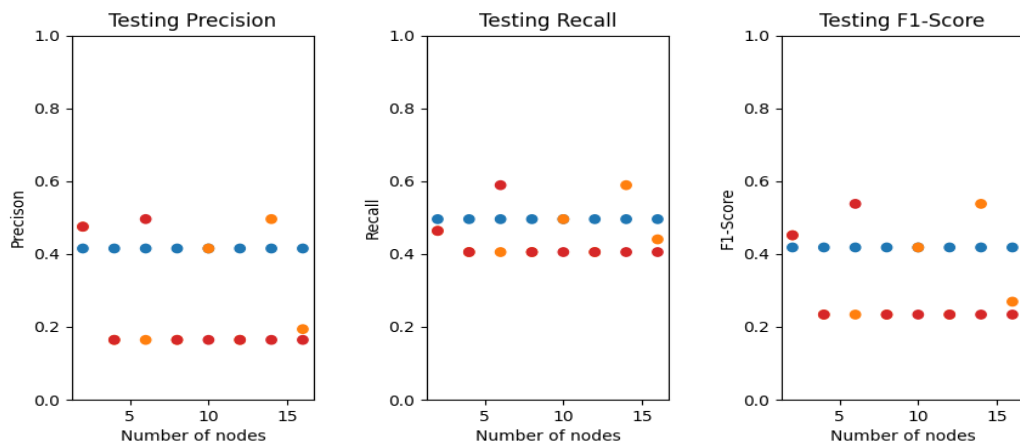


Figure C.11: Attribute split on Wine dataset using alcohol with 80% probability

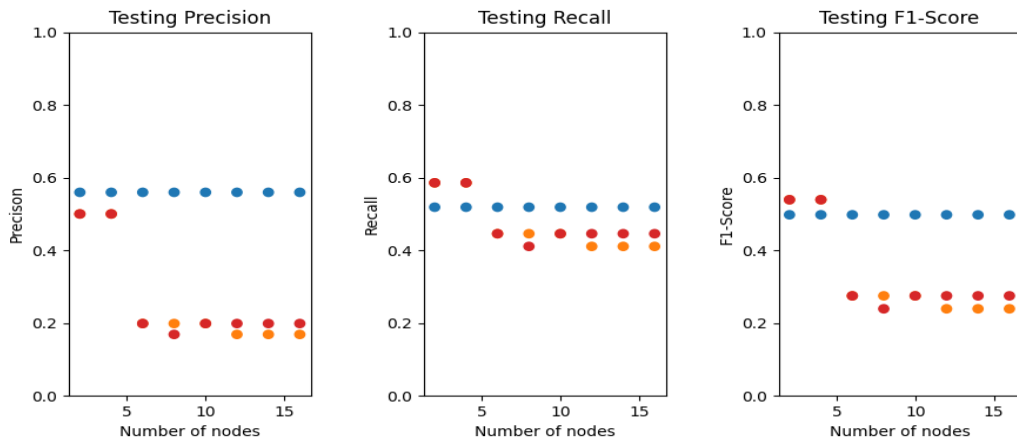


Figure C.12: Attribute split on Wine dataset using alcohol with 100% probability

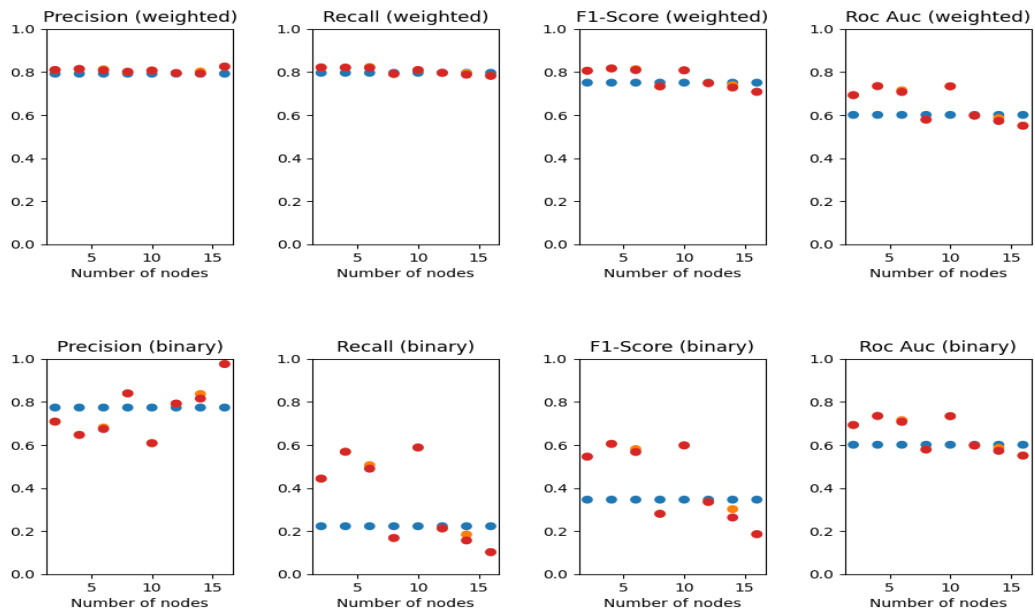


Figure C.13: Attribute split on Salary dataset using education with 66% probability

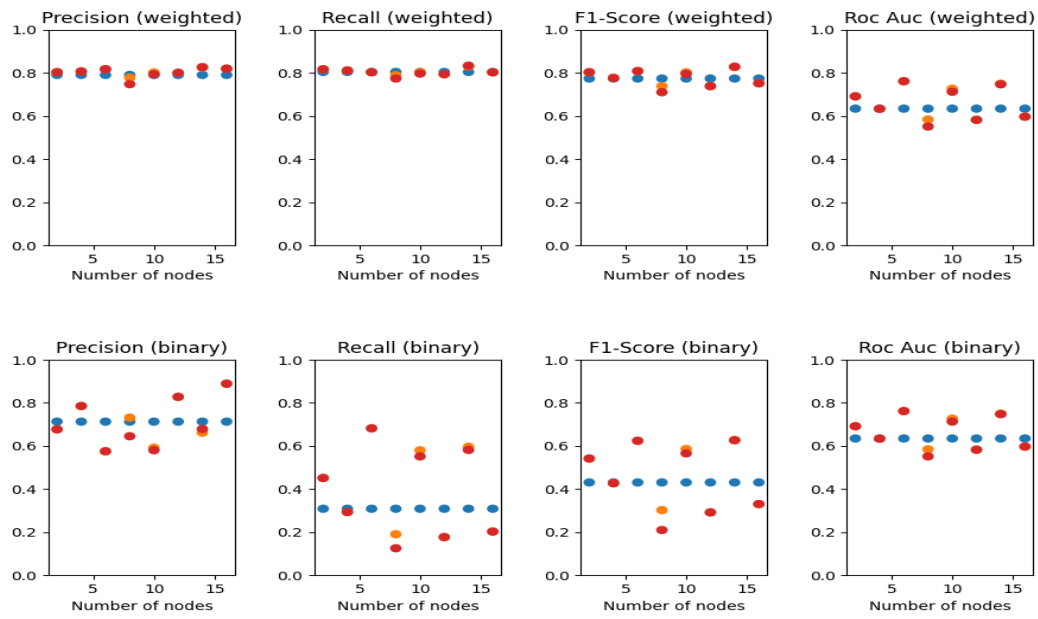


Figure C.14: Attribute split on Salary dataset using education with 80% probability

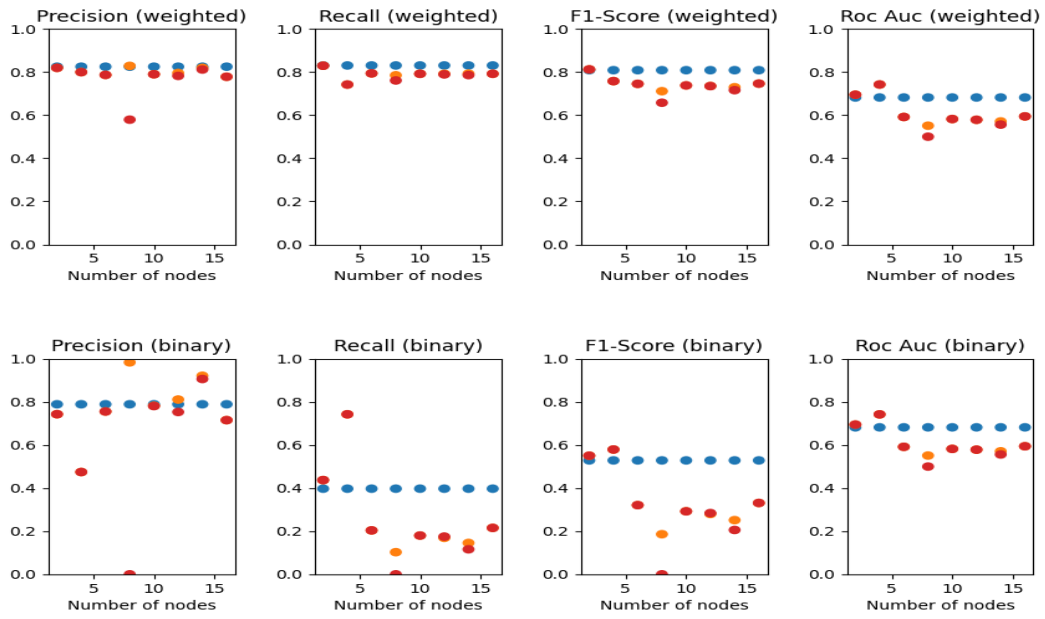


Figure C.15: Attribute split on Salary dataset using education with 100% probability

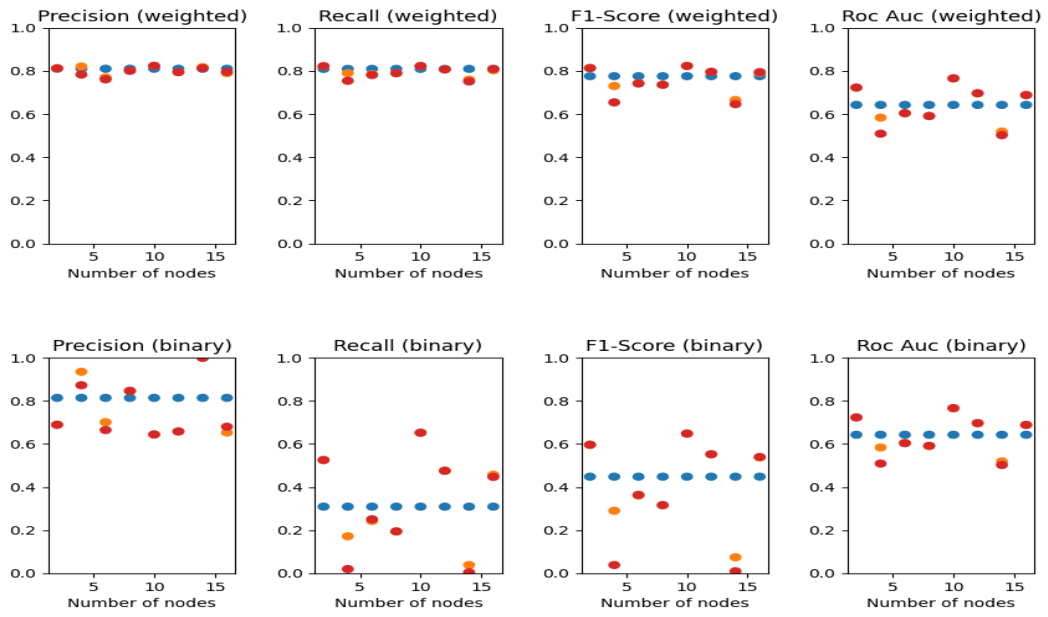


Figure C.16: Attribute split on Salary dataset using education num with 66% probability

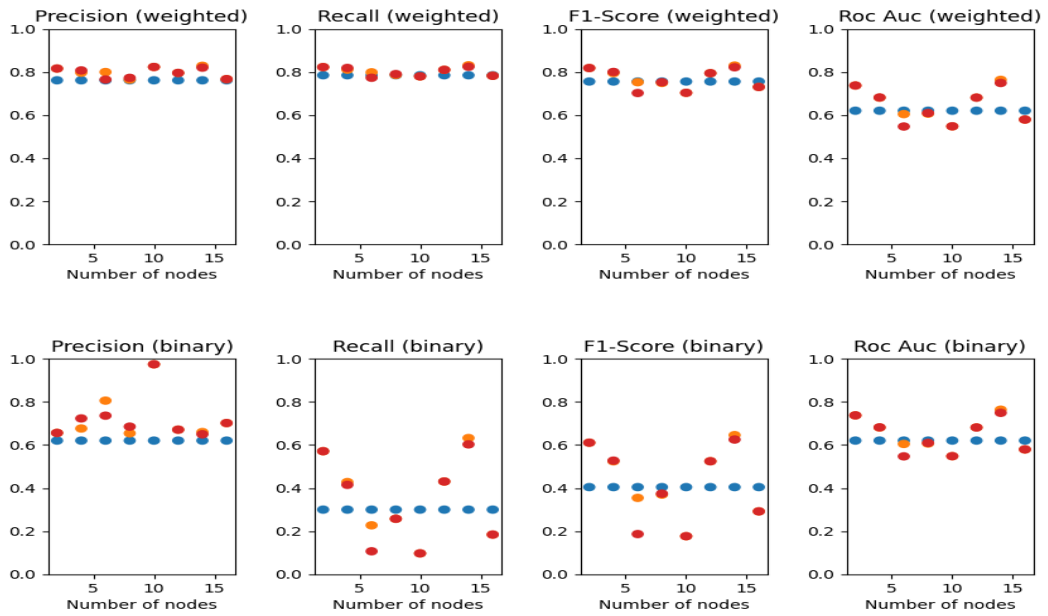


Figure C.17: Attribute split on Salary dataset using education num with 80% probability

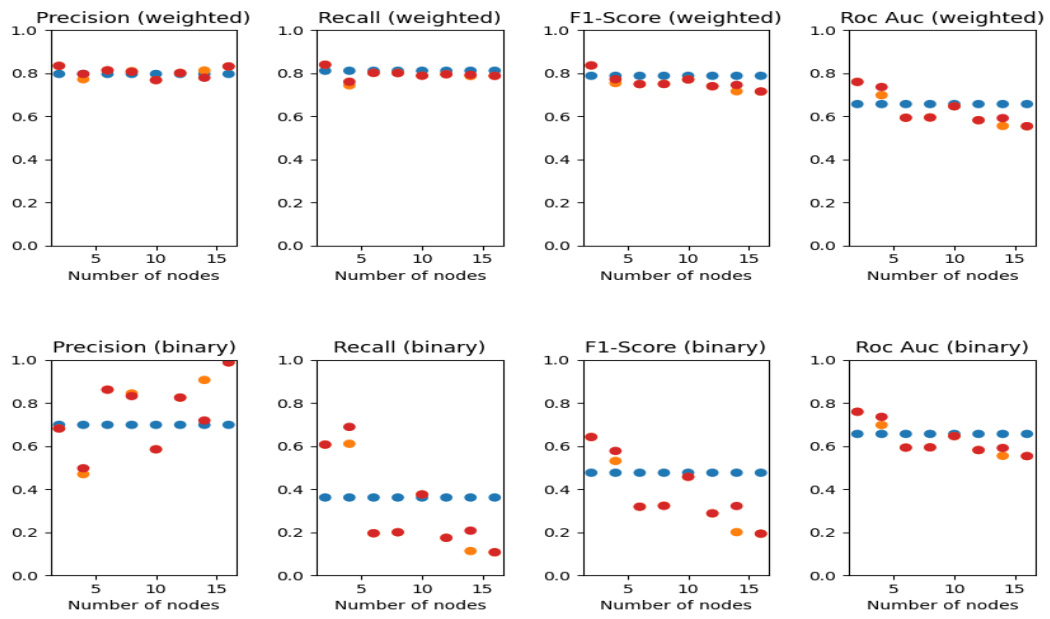


Figure C.18: Attribute split on Salary dataset using education num with 100% probability

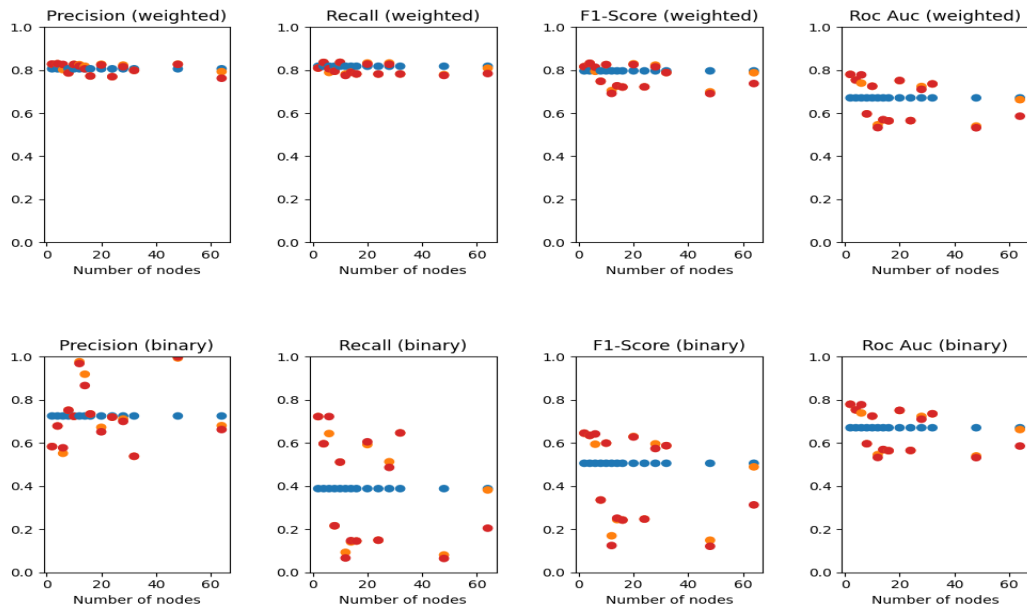


Figure C.19: Attribute split on Salary dataset using native country with 66% probability

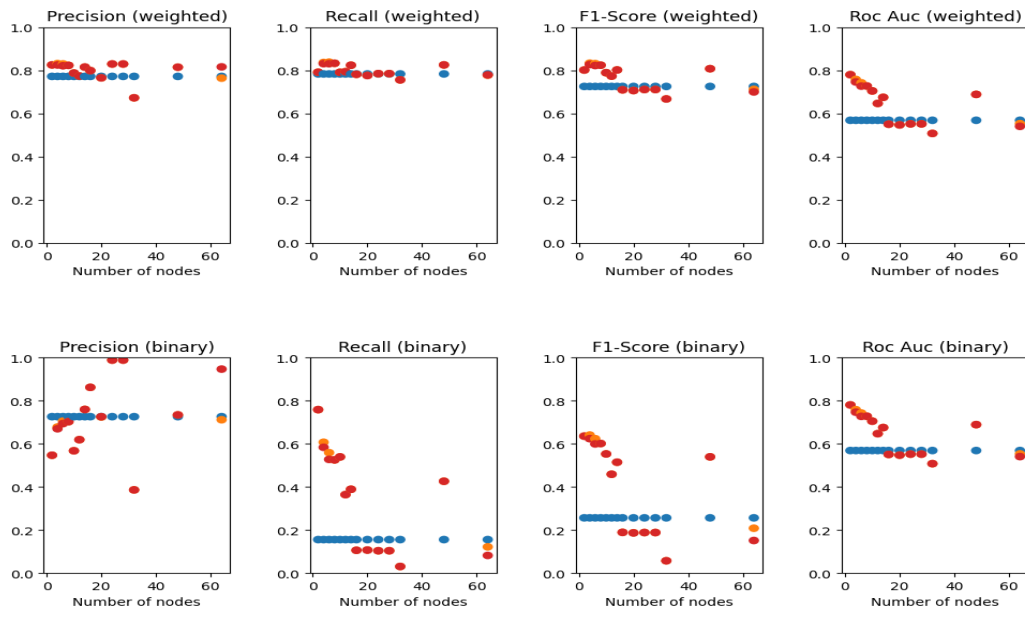


Figure C.20: Attribute split on Salary dataset using native country with 80% probability

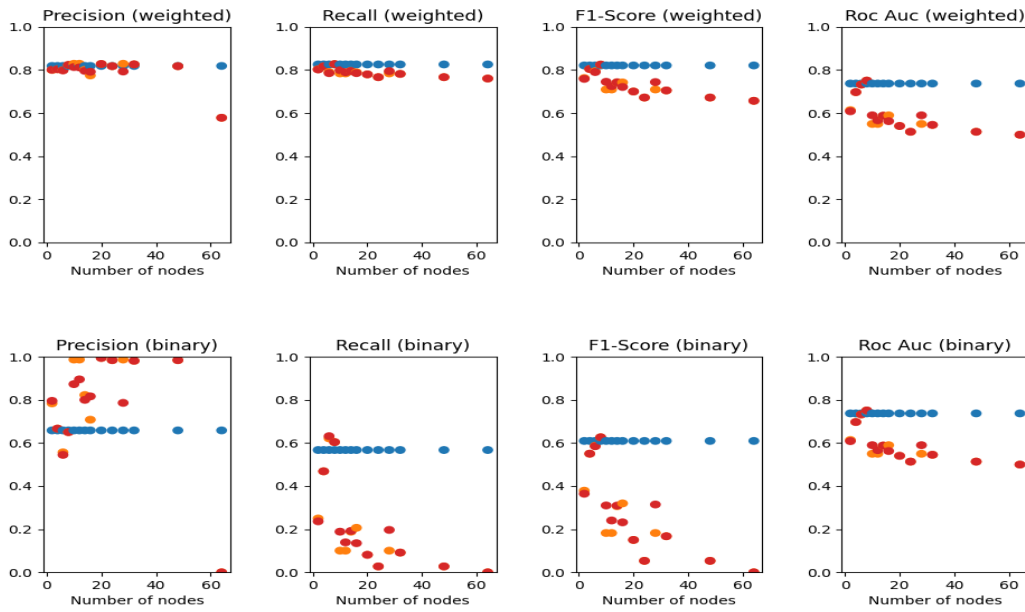


Figure C.21: Attribute split on Salary dataset using native country with 100% probability

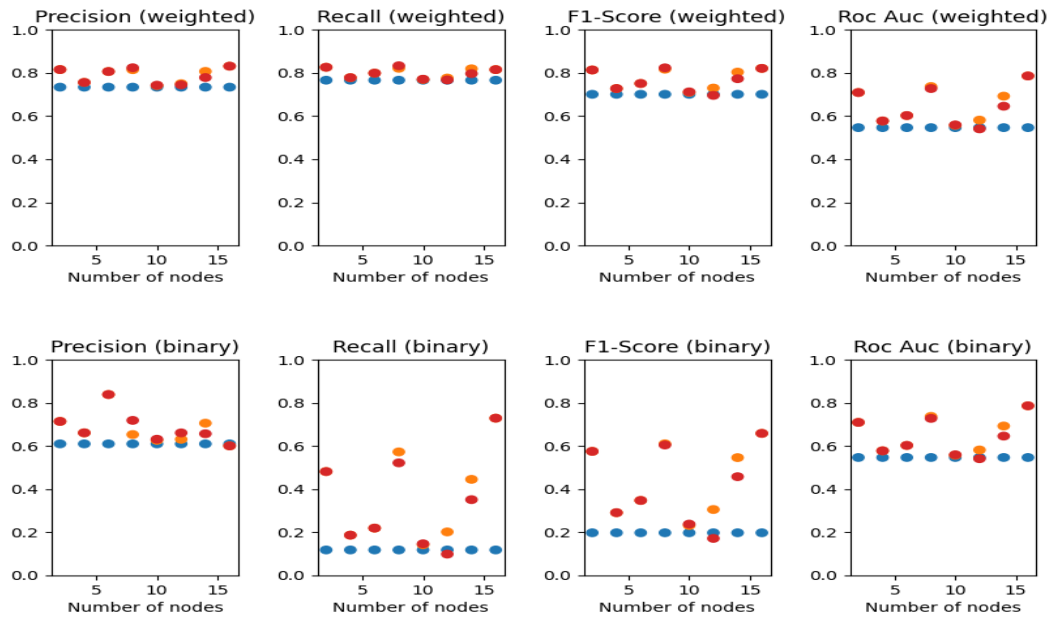


Figure C.22: Attribute split on Salary dataset using occupation with 66% probability

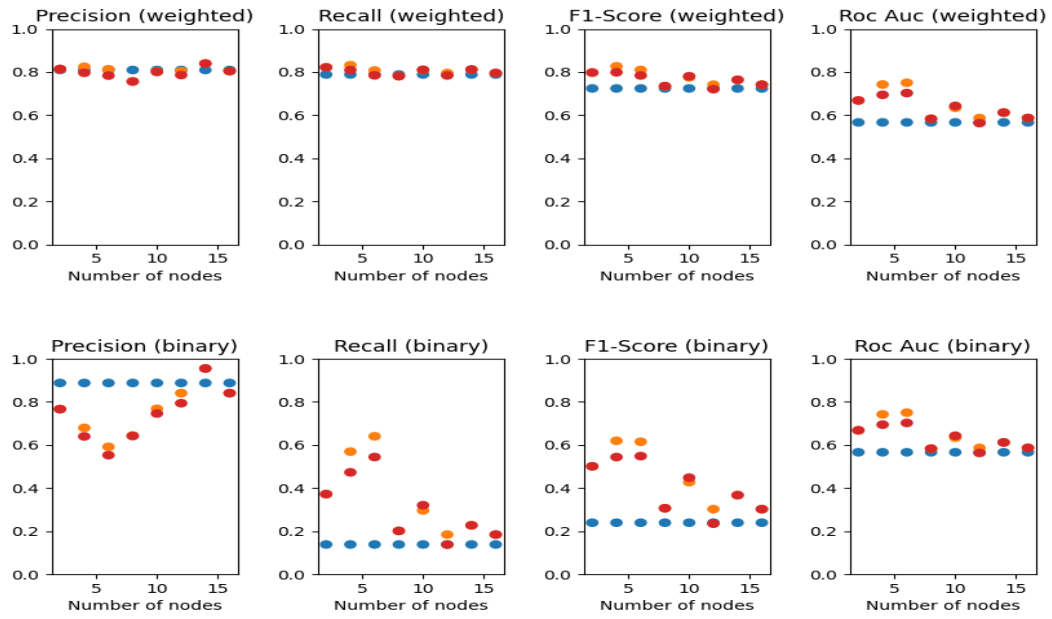


Figure C.23: Attribute split on Salary dataset using occupation with 80% probability

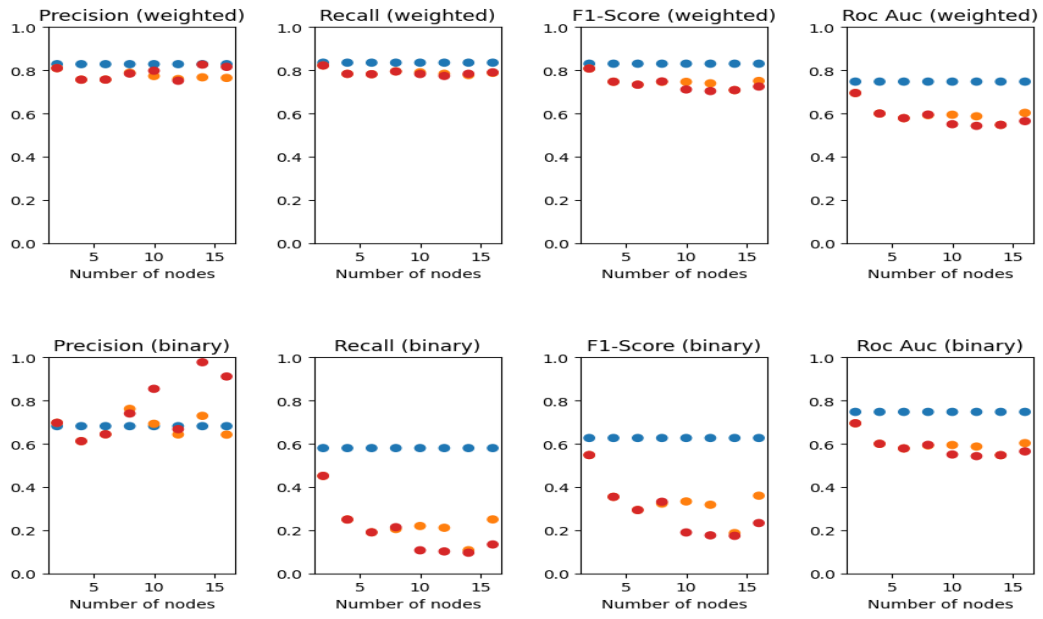


Figure C.24: Attribute split on Salary dataset using occupation with 100% probability

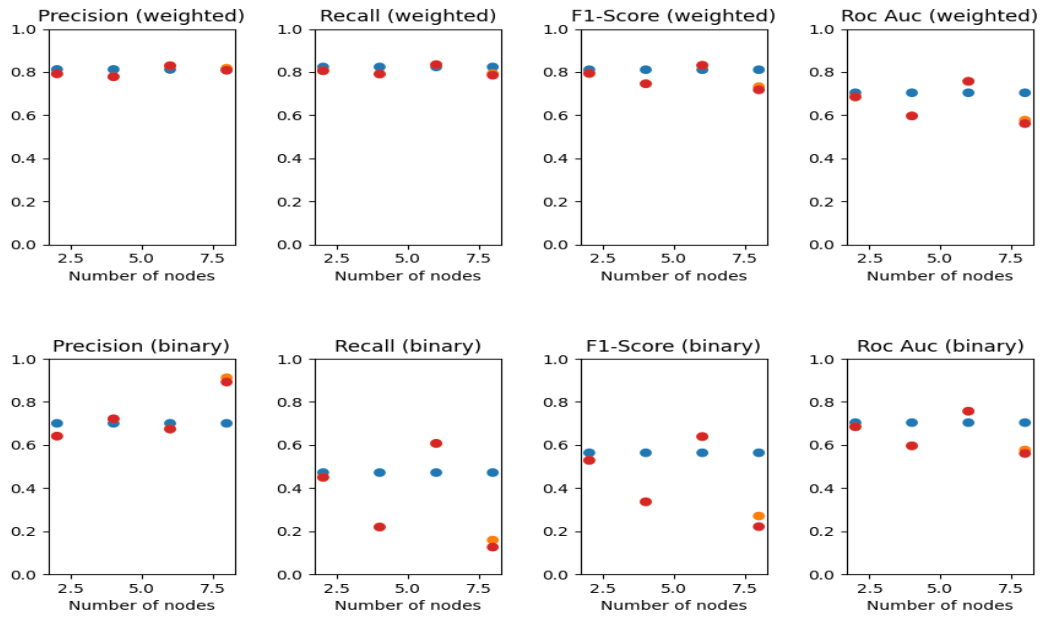


Figure C.25: Attribute split on Salary dataset using relationship with 66% probability

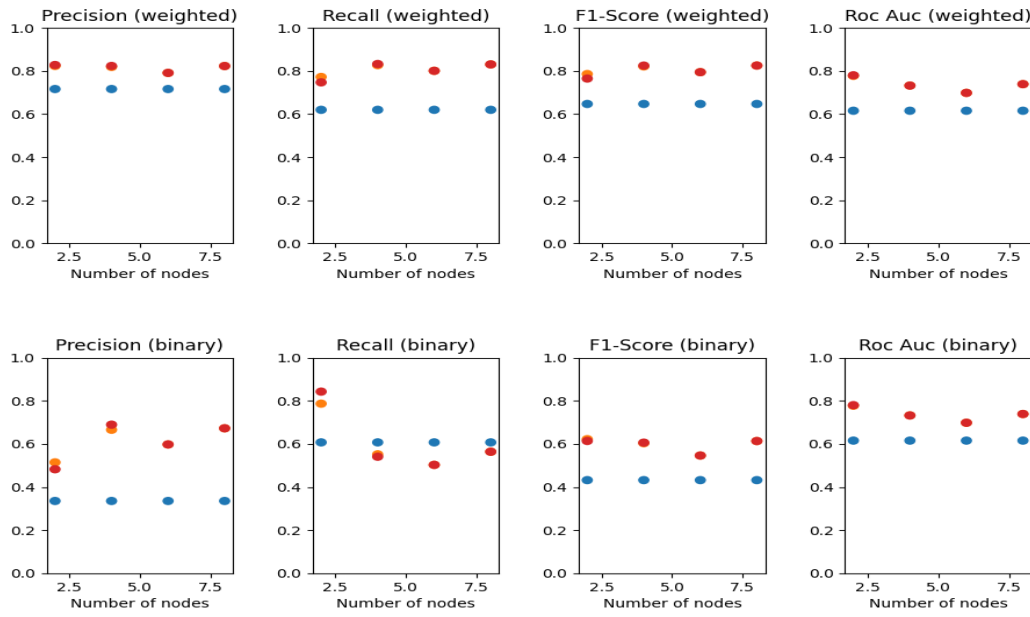


Figure C.26: Attribute split on Salary dataset using relationship with 80% probability

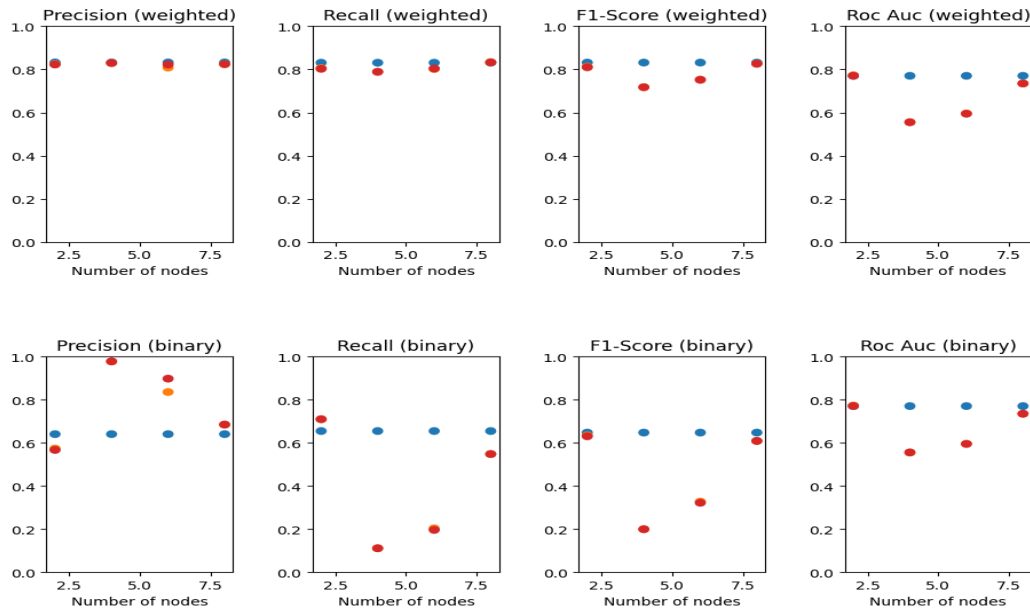


Figure C.27: Attribute split on Salary dataset using relationship with 100% probability

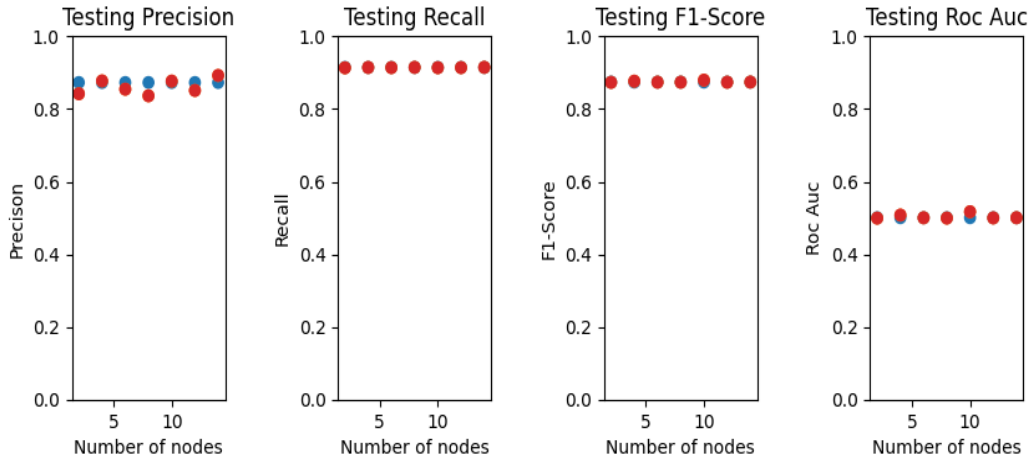


Figure C.28: Attribute split on Heart disease dataset using age with 66% probability

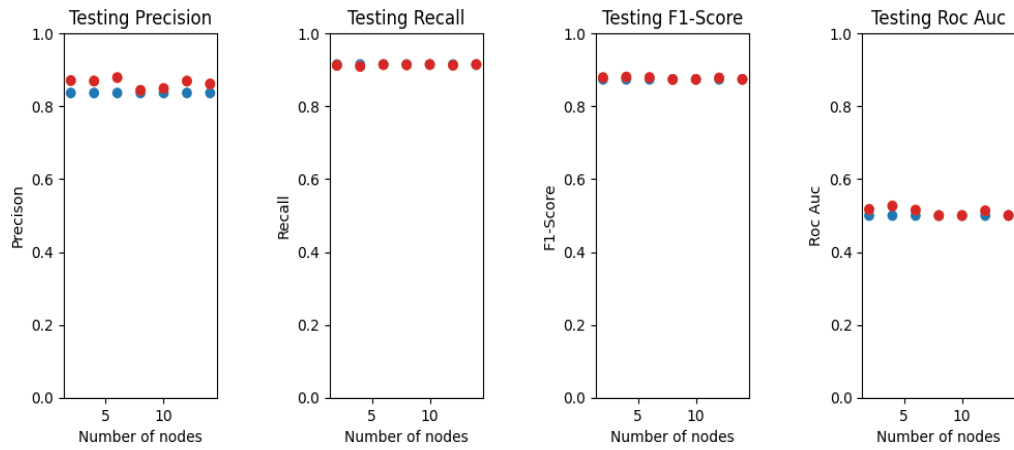


Figure C.29: Attribute split on Heart disease dataset using age with 80% probability

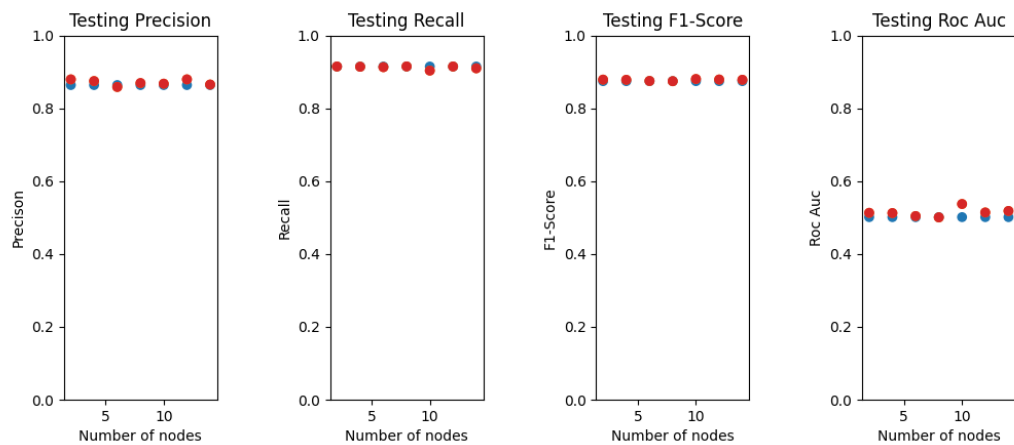


Figure C.30: Attribute split on Heart disease dataset using age with 100% probability

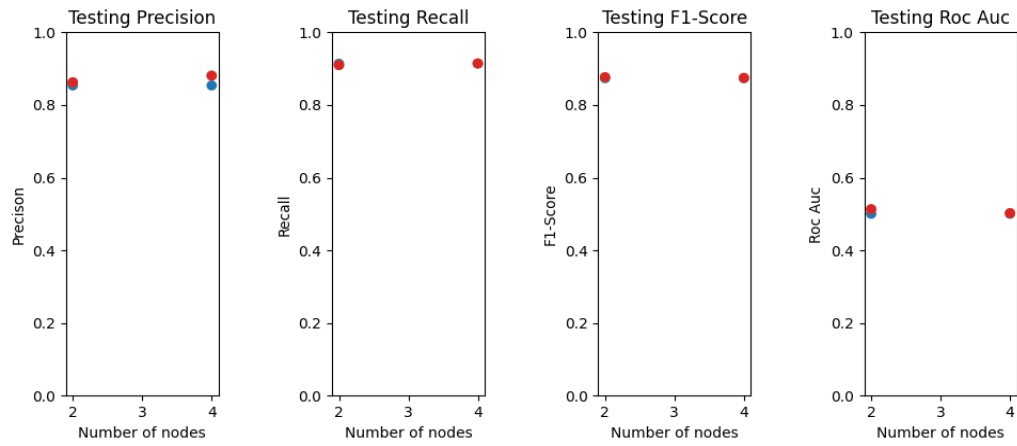


Figure C.31: Attribute split on Heart disease dataset using bmi_bin with 66% probability

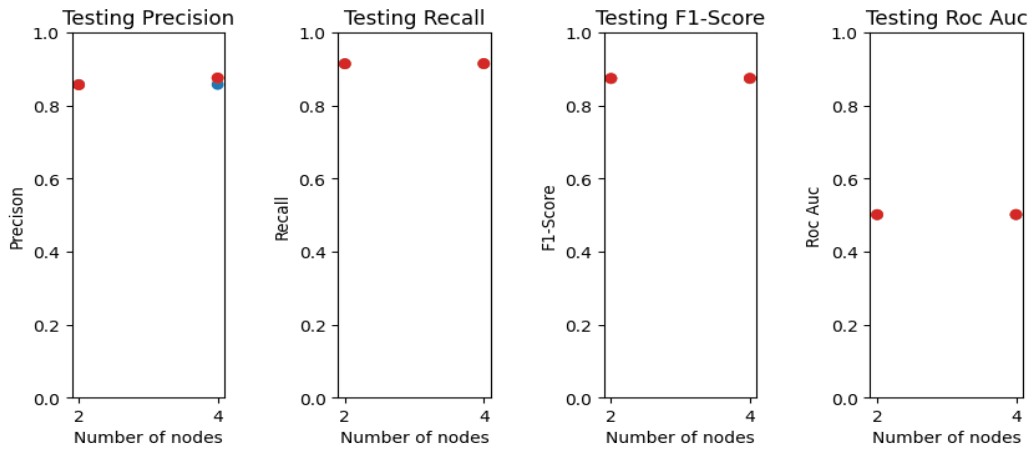


Figure C.32: Attribute split on Heart disease dataset using bmi_bin with 80% probability

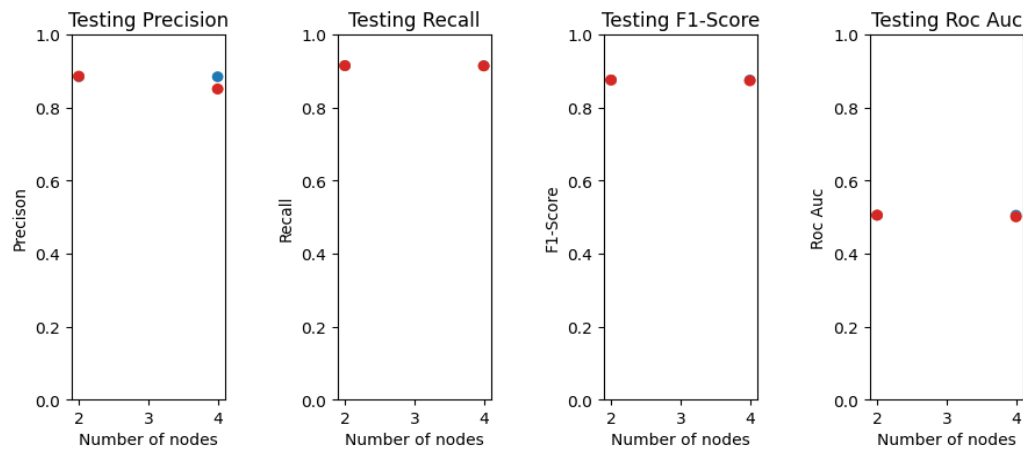


Figure C.33: Attribute split on Heart disease dataset using bmi_bin with 100% probability

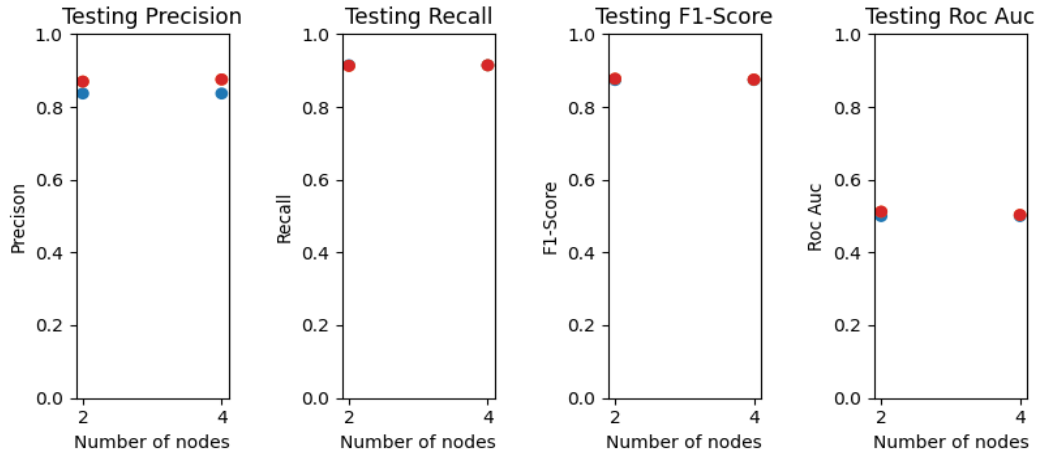


Figure C.34: Attribute split on Heart disease dataset using diabetic with 66% probability

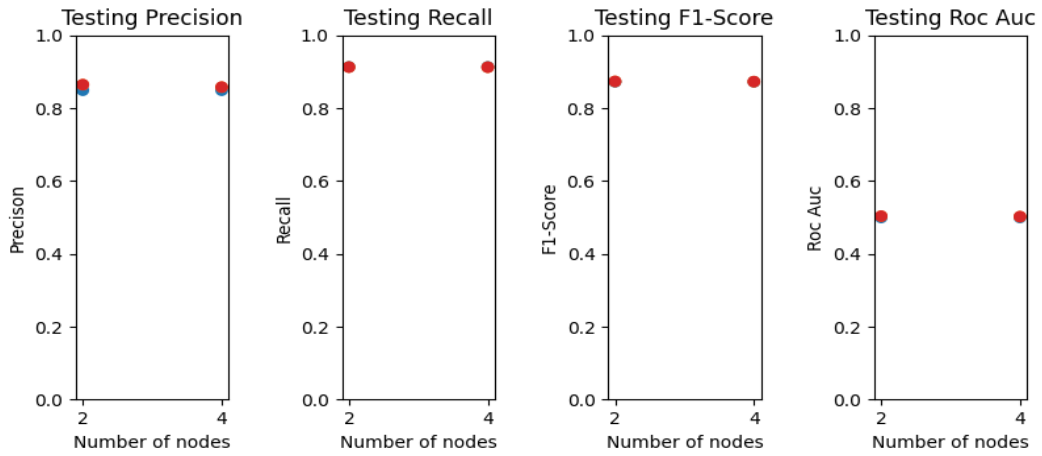


Figure C.35: Attribute split on Heart disease dataset using diabetic with 80% probability

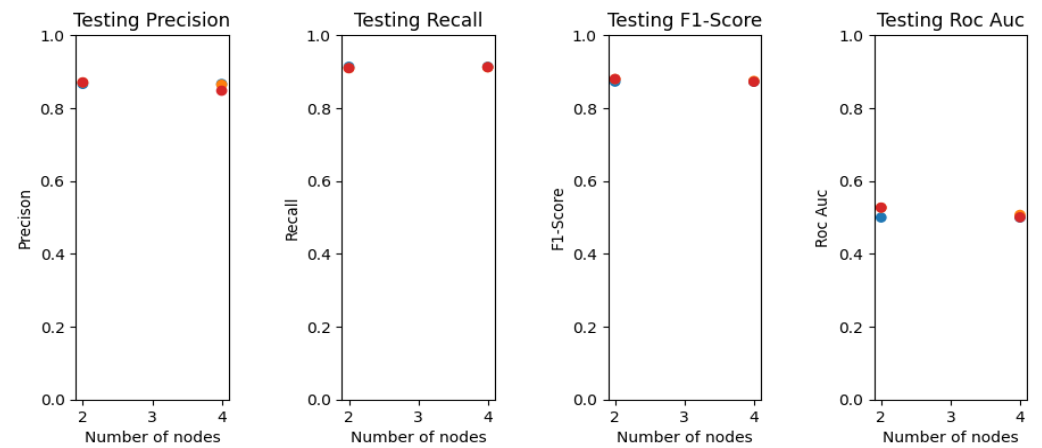


Figure C.36: Attribute split on Heart disease dataset using diabetic with 100% probability

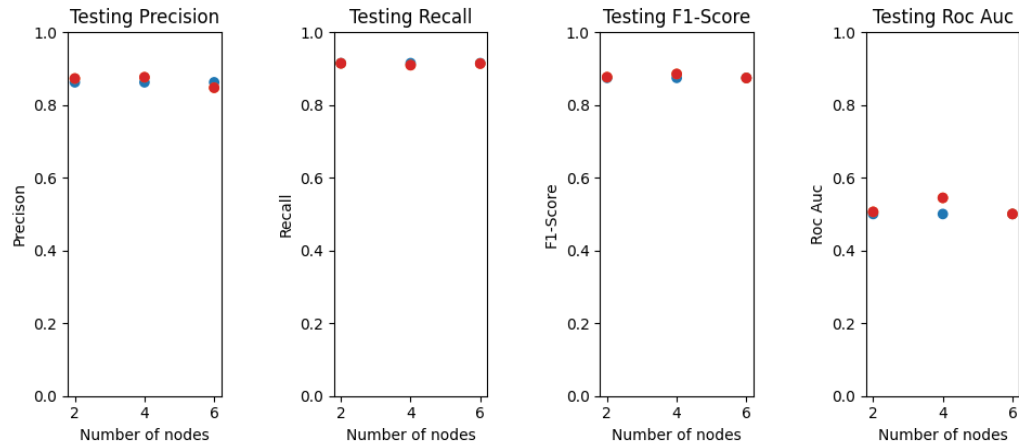


Figure C.37: Attribute split on Heart disease dataset using genHealth with 66% probability

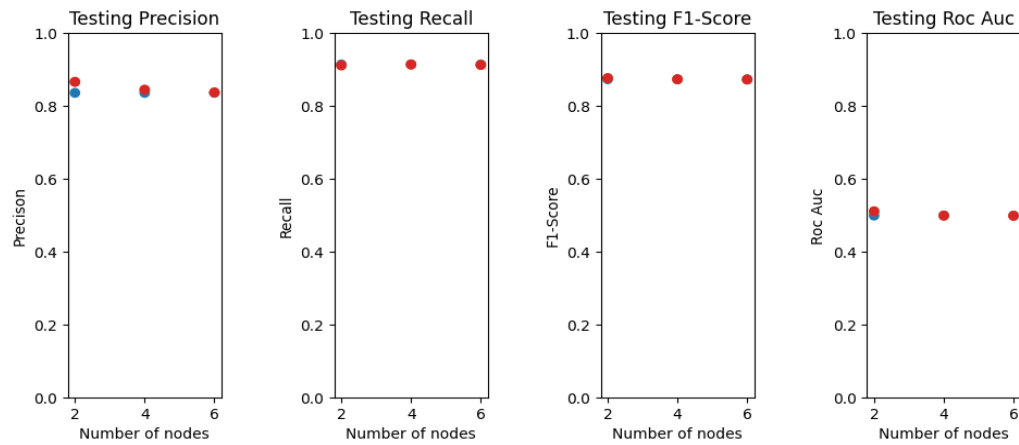


Figure C.38: Attribute split on Heart disease dataset using genHealth with 80% probability

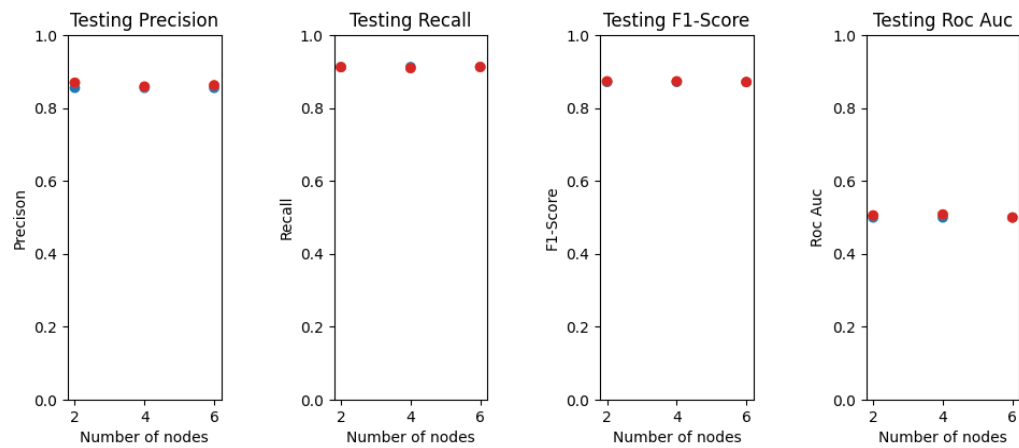


Figure C.39: Attribute split on Heart disease dataset using genHealth with 100% probability

Bibliography

- [1] Charu C Aggarwal et al. *Recommender systems*, volume 1. Springer, 2016.
- [2] SHIVAM AGRAWAL. diamonds. <https://www.kaggle.com/datasets/shivam2503/diamonds>, 2022. [Online; accessed 13-05-2022].
- [3] E. Almeida, Carlos Ferreira, and João Gama. Learning model rules from high-speed data streams. *CEUR Workshop Proceedings*, 1088:10–16, 01 2013.
- [4] Taiwo Ayodele. *Machine Learning Overview*. 02 2010. ISBN: 978-953-307-034-6. doi:10.5772/9374.
- [5] Daniel Bashir, George D Montañez, Sonia Sehra, Pedro Sandoval Segura, and Julius Lauw. An information-theoretic perspective on overfitting and underfitting. In *Australasian Joint Conference on Artificial Intelligence*, pages 347–358. Springer, 2020.
- [6] Garfield Benjamin. Privacy as a cultural phenomenon. *Journal of Media Critiques*, 3, 06 2017. doi:10.17349/Jmc117204.
- [7] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roseland. Towards federated learning at scale: System design. In A. Talwalkar, V. Smith, and M. Zaharia, editors, *Proceedings of Machine Learning and Systems*, volume 1, pages 374–388, 2019.
- [8] Erica Briscoe and Jacob Feldman. Conceptual complexity and the bias/variance tradeoff. *Cognition*, 118(1):2–16, 2011.
- [9] Tianfeng Chai and R.R. Draxler. Root mean square error (rmse) or mean absolute error (mae)?– arguments against avoiding rmse in the literature. *Geoscientific Model Development*, 7:1247–1250, 06 2014. doi:10.5194/gmd-7-1247-2014.
- [10] Chris Chatfield. Exploratory data analysis. *European Journal of Operational Research*, 23 (1):5–13, 1986. ISSN: 0377-2217. doi:[https://doi.org/10.1016/0377-2217\(86\)90209-2](https://doi.org/10.1016/0377-2217(86)90209-2).
- [11] Pádraig Cunningham, Matthieu Cord, and Sarah Jane Delany. Supervised learning. In *Machine learning techniques for multimedia*, pages 21–49. Springer, 2008.

- [12] Emily Denton, Alex Hanna, Razvan Amironesei, Andrew Smart, and Hilary Nicole. On the genealogy of machine learning datasets: A critical history of imagenet. *Big Data & Society*, 8(2):20539517211035955, 2021.
- [13] Rubia Fatima, Affan Yasin Chouhan, Lin Liu, Wang Jianmin, Wasif Afzal, and Awaid Yasin. Sharing information online rationally: An observation of user privacy concerns and awareness using serious game. *Journal of Information Security and Applications*, 48:102351, 07 2019. doi:10.1016/j.jisa.2019.06.007.
- [14] SHIV KUMAR GANESH. road-traffic-deaths-1990-to-2019. <https://www.kaggle.com/datasets/shivkumarganesh/road-traffic-deaths-1990-to-2019>, 2022. [Online; accessed 13-05-2022].
- [15] Zoubin Ghahramani. Unsupervised learning. In *Summer school on machine learning*, pages 72–74. Springer, 2003.
- [16] Yaobin He, Haoyu Tan, Wuman Luo, Shengzhong Feng, and Jianping Fan. Mr-dbscan: a scalable mapreduce-based dbscan algorithm for heavily skewed data. *Frontiers of Computer Science*, 8(1):83–99, 2014.
- [17] Saikishore Kalloori and Severin Klingler. Cross-silo federated learning based decision trees. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, pages 1117–1124, 2022.
- [18] Jens Kirchner, Andreas Heberle, and Welf Löwe. Classification vs. regression - machine learning approaches for service recommendation based on measured consumer experiences. In *2015 IEEE World Congress on Services*, pages 278–285, 2015. doi:10.1109/SERVICES.2015.49.
- [19] UCI MACHINE LEARNING. salary. <https://www.kaggle.com/datasets/uciml/adult-census-income>, 2022. [Online; accessed 13-05-2022].
- [20] UCI MACHINE LEARNING. wine. <https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009>, 2022. [Online; accessed 13-05-2022].
- [21] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 2021.
- [22] Wei Liu, Li Chen, Yunfei Chen, and Wenyi Zhang. Accelerating federated learning via momentum gradient descent. *IEEE Transactions on Parallel and Distributed Systems*, 31(8):1754–1766, 2020.
- [23] Prem Melville and Vikas Sindhwani. Recommender systems. *Encyclopedia of machine learning*, 1:829–838, 2010.
- [24] Netflix. Netflix. <https://research.netflix.com/research-area/machine-learning>, 2022. [Online; accessed 7-august-2022].

- [25] Nils J. Nilsson. Introduction to machine learning. an early draft of a proposed textbook, 1996.
- [26] Jasmina Dj. Novaković, Alempije Veljović, Siniša S. Ilić, Željko Papić, and Tomović Milica. Evaluation of classification models in machine learning. *Theory and Applications of Mathematics and Computer Science*, 7(1):Pages: 39 –, Apr. 2017.
- [27] Diego Peteiro-Barral and Bertha Guijarro-Berdiñas. A survey of methods for distributed machine learning. *Progress in Artificial Intelligence*, 2, 03 2012. doi:10.1007/s13748-012-0035-5.
- [28] KAMIL PYTLAK. heartdisease. <https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease>, 2022. [Online; accessed 13-05-2022].
- [29] AUSTIN REESE. used cars. <https://www.kaggle.com/datasets/austinreese/craigslist-carstrucks-data>, 2022. [Online; accessed 13-05-2022].
- [30] Lior Rokach and Oded Maimon. Clustering methods. In *Data mining and knowledge discovery handbook*, pages 321–352. Springer, 2005.
- [31] Yang Shengwen, Ren Bing, Zhou Xuhui, and Liu Liping. Parallel distributed logistic regression for vertical federated learning without third-party coordinator. *arXiv preprint arXiv:1911.09824*, 2019.
- [32] Spotify. Spotify. <https://research.atspotify.com/machine-learning/>, 2022. [Online; accessed 7-august-2022].
- [33] staista. Statista. <https://www.statista.com/statistics/871513/worldwide-data-created/#statisticContainer>, 2022. [Online; accessed 7-august-2022].
- [34] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [35] Ellen Tannam. Whitebox. <https://www.siliconrepublic.com/enterprise/white-box-machine-learning>, 2022. [Online; accessed 7-august-2022].
- [36] Luís Torgo. Regression. <https://www.dcc.fc.up.pt/ltorgo/Regression/>, 2022. [Online; accessed 7-august-2022].
- [37] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM workshop on artificial intelligence and security*, pages 1–11, 2019.
- [38] Laurens Van Der Maaten, Eric Postma, Jaap Van den Herik, et al. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10(66-71), 2009.
- [39] Joost Verbraeken, Matthijs Wolting, Jonathan Katzy, Jeroen Kloppenburg, Tim Verbelen, and Jan Rellermeyer. A survey on distributed machine learning. *ACM Computing Surveys*, 53:1–33, 03 2020. doi:10.1145/3377454.

-
- [40] Yuncheng Wu, Shaofeng Cai, Xiaokui Xiao, Gang Chen, and Beng Chin Ooi. Privacy preserving vertical federated learning for tree-based models. *arXiv preprint arXiv:2008.06170*, 2020.
 - [41] Liu Yang, Ben Tan, Vincent W Zheng, Kai Chen, and Qiang Yang. Federated recommendation systems. In *Federated Learning*, pages 225–239. Springer, 2020.
 - [42] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.
 - [43] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021. ISSN: 0950-7051. doi:<https://doi.org/10.1016/j.knosys.2021.106775>.
 - [44] ÂngeloGomes. github. <https://github.com/agagomes/FederatedMachineLearning>, 2022. [Online; accessed 19-09-2022].

