FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

# Leveraging CRM Insights

## Identifying and Analysing for Bias in Model Training in a CRM's AI

**Maria Beatriz Sousa Gonçalves**


Masters in Data Science and Engineering

Supervisor: Pedro Sanches Amorim, PhD

Co-supervisor: Rita Gonçalves Angélico, MSc

Co-supervisor: Mariana Lourenço Costa, MSc

July 18, 2024

# Leveraging CRM Insights

## Identifying and Analysing for Bias in Model Training in a CRM's AI

**Maria Beatriz Sousa Gonçalves**

Masters in Data Science and Engineering

July 18, 2024

# Resumo

O presente documento descreve o trabalho de auditoria realizado no âmbito da nossa dissertação, que se foca na utilização de informações de uma plataforma de *Customer Relationship Management* (CRM), mais especificamente na identificação e análise de enviesamentos no treino de modelos de Inteligência Artificial nestes sistemas. O CRM em análise é a plataforma Salesforce, com a sua ferramenta de Inteligência Artificial (AI), o Einstein. Atualmente, esta ferramenta aparenta ter mecanismos de prevenção de enviesamentos, que serão o nosso principal tópico de investigação. Esta análise incluirá um teste extensivo de funcionalidades com vários algoritmos disponíveis no portefólio do Einstein, assim como várias métricas de desempenho e indicadores de performance. Como conclusão, será realizada uma análise geral, com o objetivo de compreender resultados e se os mesmo foram alvo de enviesamento intrínseco na plataforma. Se bem sucedida, esta investigação contribui para previsões melhores e imparciais em modelos em vários domínios e contextos como vendas, serviços ou marketing.

# Abstract

This document describes the auditing work performed within the scope of our dissertation, which is centred on leveraging Customer Relationship Management (CRM) insights, specifically on the topic of identifying and analysing for Bias in model training in these systems' Artificial Intelligence (AI). The CRM in the analysis is the Salesforce platform, with its AI assistant, Einstein. Currently, Einstein seems to have bias-preventing mechanisms, which will be our main topic of investigation. This analysis will include extended feature testing with multiple algorithms available on Einstein's portfolio and various performance metrics and evaluators. In conclusion, a general analysis will be conducted in order to understand the results and whether they have been subject to intrinsic bias in the platform. If successful, this research contributes to better and unbiased predictions in models in various fields and environments such as sales, service or marketing.

# Contribution to UN's Sustainable Development Goals

The United Nations (UN) Sustainable Development Goals were officially established in 2015 for all member states. As part of the 2030 Agenda for Sustainable Development, these goals should contribute to a better world. There are 17 goals in total, with a focus on topics such as poverty, inequality, Peace and justice, and climate action (see Figure 1) (UN, 2015b).

The 2030 Agenda for Sustainable Development is divided into five main areas (UN, 2015a):

1. **People**, for issues belonging to topics like poverty, hunger, or inequality;
2. **Planet**, related to the relevant topic of climate change and sustainability;
3. **Prosperity**, for "*economic, social and technological* progress" (UN, 2015a);
4. **Peace**, for a more just and inclusive environment.

This dissertation will be conducted within the scope of bias mitigation and detection, framing it in two goals: "Gender Equality" and "Reduced Inequalities".

The fifth Goal, **Gender Equality**, strives to "*Achieve gender equality and empower all women and girls* (UN, 2015d). Within this Goal, several targets and indicators range from eliminating discrimination and violence against women to promoting more opportunities in multiple sectors and services. According to UN records, at the current progress rate, it will take approximately 286 years to close gender gaps. Our Goal for this dissertation is to audit a technological tool - Einstein Discover from Salesforce - that companies use in real use cases, meaning it can impact individuals' welfare if it still has inherent biases when making predictions. Given our dataset, composed of several demographic attributes (one being gender - "sex" variable -with two possible categories, "Male" or "Female"), we want to understand how biased the predictions can be, given different attributes. Our research will contribute to this Goal by auditing and thoroughly testing Einstein's predictions and exposing flaws and biased results, should they appear in the different experiments. In order to move forward with progress for Gender Equality, it is essential to identify and test every technological solution available to ensure that they are not perpetuating even more gender stereotypes and discrimination. Should Einstein produce biased or discriminatory results, it is our responsibility to reveal them, make possible suggestions to improve the tool and hold its developers and makers accountable. As stated before, our work revolves around bias mitigation and detection, meaning that if we want to go further in the Goal's targets and indicators, it precisely fits in Target 5.1, "*End all forms of discrimination against all women and girls everywhere*", and its respective Indicator 5.1.1, "*Whether or not legal frameworks are in place to promote, enforce and monitor equality and non-discrimination based on sex*". Prediction tools and models reflect their human developers. Most times, all it takes is for a model to be inclined to predict a better income for a man rather than a woman - even if she has a higher degree of education - to uncover more significant issues.


Figure 1: The 17 Sustainable Development Goals.

Advancing to the tenth Goal, **Reduced Inequalities** endeavours to "*reduce inequality within and among countries*". Targets and indicators for this goal range from promoting sustained income growth and equal opportunity to improving safety and responsible migration (UN, 2015c). Within this Goal's targets and indicators, there are three we can highlight as more in line with our work:

- **10.2**: "*empower and promote the social, economic and political inclusion of all, irrespective of age, sex, disability, race, ethnicity, origin, religion or economic or other status*";
- **10.3**: "*Ensure equal opportunity and reduce inequalities of outcome, including by eliminating discriminatory laws, policies and practices and promoting appropriate legislation, policies and action in this regard*;
- **10.4**: "*Adopt policies, especially fiscal, wage and social protection policies, and progressively achieve greater equality*".

By analysing the results from Einstein, we can get an overview of how well Salesforce is achieving these targets. Is the outcome really unbiased? Or is there still discriminatory treatment based on an observation's sensitive attributes? The rationale behind this contribution is the same as before: by performing several tests and experiments on a demographic dataset with many sensitive attributes, we will be either discovering a good prediction tool in compliance with the Goal's target or a [still] biased prediction tool with room for improvement - this would be an opportunity to shape the implementation of these improvements according to the previously highlighted targets.

In the following chapters, we will have the opportunity to analyse all possibilities and state further to which extent this predictor module contributes (or not) to these goals. Either way, our research and work in the matter are already contributions.

# Acknowledgements

I would like to thank FEUP and Deloitte for making this dissertation possible, allowing access to the Salesforce environment, and allowing me to explore Einstein and dive into such relevant topics as bias and discrimination.

I would especially like to thank my orientation team, Pedro Sanches Amorim from FEUP, and Rita Gonçalves Angélico and Mariana Lourenço Costa from Deloitte, who were always available to help, from giving feedback and improvement suggestions to overcoming obstacles and difficulties at any stage of the research and work.

On a more sentimental note, I would like to thank my family. Firstly, I would like to thank my parents for giving me the privilege of a great education and teaching me its importance. Thank you for always supporting me and my work and being there for me anytime. I would not be where I am today or achieved what I have without you.

Lastly, and most importantly, I want to thank my brother, who has been my pillar throughout this whole journey. There are not enough words to describe how much I appreciate everything he has helped with. Thank you for inspiring me and pushing me harder every day to ensure I achieve my potential in this dissertation and my personal and professional life. Thank you for listening to me, for all the advice, and for always making time for me, whatever time zone you found yourself in.

Maria Beatriz Sousa Gonçalves

*"To succeed, planning alone is insufficient.*
*One must improvise as well."*

Isaac Asimov

# Contents

# List of Figures

# List of Tables

# Abbreviations and Acronyms

| | |
|---|---|
| AI | Artificial Intelligence |
| AIC | Akaike Information Criterion |
| App/Apps | Application/Applications |
| AUC | Area Under the Curve |
| AutoML | Auto Machine Learning |
| CRM | Customer Relationship Management |
| CSV | Comma-Separated Values |
| EDA | Exploratory Data Analysis |
| FN | False Negatives |
| FP | False Positives |
| GBM | Gradient Boosting Machines |
| GLM | Generalized Linear Model |
| KPIs | Key Performance Indicators |
| MCC | Mathews Correlation Coefficient |
| ML | Machine Learning |
| PaaS | Platform as a Service |
| SaaS | Software as a Service |
| SDK | Software Development Kit |
| SMEs | Small and Medium Enterprises |
| TN | True Negatives |
| TP | True Positives |
| UI | User Interface |
| UX | User Experience |
| UN | United Nations |
| XG Boost | Extreme Gradient Boosting |

# Chapter 1

# Introduction

## 1.1 Context

As humans have biases and preconceived notions about the world around them, so do the models they build. A model will produce biased or unbiased predictions depending on the prepared data. Even if the creators have their biases, the models shouldn't be used in a context that could negatively affect many individuals (DeBrusk, 2018). With these concerns on the rise, the topic of ethical model development and AI has started to be taken more seriously and considered. It is safe to say that some guidelines have been put into practice, but not to modify data preparation and model training completely.

## 1.2 Motivation

When looking at an established CRM Platform such as Salesforce (Sunkari, 2022), one can wonder how much effort was put into mitigating discriminatory outcomes from its model recommendations and predictions. Narrowing this vision into the CRM's AI - Einstein - how well does it pick up on potential negative (and biased) impact a variable could have on a model and its outcomes and predictions? Does it look at a customer's behaviour and profile and give better offers to the same group? Or does it diversify the company's offerings to find potential opportunities in different groups with different levels of company loyalty? Since Einstein claims to have bias-preventing and flagging mechanisms, the prime motivation for this dissertation is to understand them and explore the extent to which they can be leveraged for future and current CRM projects. Furthermore, in a world filled with discrimination and segregation behaviour, there needs to be technological models that are able to take data and analyse it based on meaningful variables and information instead of other factors related to a human's personal beliefs. In summary, how models are thought out is changing to reflect these newer concerns, but are these changes really impactful on a model's capability to prevent potentially biased outcomes? Has this issue been analysed to the best of its ability so as to improve models? To summarise what has been stated so far, introducing AI solutions in business processes will become increasingly common (DeBrusk, 2018). Consequently, it

is essential to understand how these models are influenced by their developers and which kind of biases they might have from the start, as biased models produce biased or discriminatory outputs. We can conclude that these outputs impact user experiences, customer interactions, perceptions of companies' business practices, ethical practices, and fairness beliefs.

## 1.3 Goals & Objectives

In pursuit of addressing the existing gap in Salesforce Einstein research with data outside of the CRM, this section delineates the goals and specific objectives that guide it. The defined goals will provide a broad framework for the analysis, while the objectives serve as tangible milestones to achieve. By highlighting these goals and objectives, we aim to provide a clear road map for the research process, ensuring a focused and systematic approach to unravelling the complexities inherent in the research problem. We have identified three main goals, which seek to answer the following proposed research questions:

1. To what extent do Salesforce Einstein's models exhibit bias in their outcomes and predictions when potential bias variables are not explicitly flagged?

2. How reliable is the "Analyse for Bias" feature in Salesforce Einstein, and to what degree does it effectively mitigate discriminatory outcomes in the models it assesses?

3. What additional features and capabilities does Salesforce Einstein possess to uphold ethical practices in AI development and deployment beyond the "Analyse for Bias" functionality?

As for the objectives, we have identified three key points and a work plan to follow. For critical points, at the end of this work process, we should have successfully assessed bias in unflagged variables, evaluated the effectiveness of bias-detecting mechanisms and features in Einstein, and identified the existence (if any) of other ethical features in Einstein Discovery. Additionally, we've outlined the most relevant tasks for a work plan aligned with such objectives, listed in Table 1.1.

Table 1.1: Work Plan.

| **Objective** | **Methodology** | | **Expected Outcomes** | | **Implications** | |
|---|---|---|---|---|---|---|
| To evaluate the effectiveness of Einstein Discovery's Analyse for Bias feature in identifying and mitigating bias in its models | 1st Approach | 2nd Approach | Is Effective | Is not Effective | Is Effective | Is not Effective |
| | Without flagging any variables t hat could lead to biased results | Flagg variables and follow Einstein's suggestions to mitigate bias | The second approach should have fewer biased results compared to the first model | There should be no significant difference in the results of the two approaches | It can be used to improve the accuracy and fairness of Einstein AI models without highly compromising the bias-variance trade-off | It highlights the need for further research and development in identifying and mitigating bias in AI |

## 1.4 Dissertation Structure

Beyond the Introduction, this dissertation contains seven additional chapters. In Chapter 2, we present the state of the art of the Einstein Discovery Platform and related work. In Chapter 3, we explain and describe a preliminary experiment with Einstein conducted with the data that will

be used for the rest of the work and some results obtained. We begin by conducting an initial experiment on the platform in Chapter 3, with little to no customisation. In Chapters 4 and 5, we perform the same experiments with only one difference: in the first chapter, every variable is flagged for bias, whilst in the other, no variable is flagged for bias. As Einstein provides an Automated Model option, we dedicate Chapter 6 to the experiment and its related results. Finally, in Chapter 7, we perform two experiments outside the Einstein platform - we ran a Random Forest and XG Boost model and tested two fairness metrics not available in Einstein. Still, in this chapter, test and draw conclusions about the platform's "Upload Model" option. Chapter 8 contains our final conclusions.

# Chapter 2

# Literature Review

In this chapter, we will follow the evolution of customer relationship management throughout the years and the consequential appearance of CRM systems and platforms with the fast development of information technology. Furthermore, we will also perform a brief analysis of topics such as CRM Fairness and the CRM paradox. Additionally, we will have a more in-depth review of Salesforce's available AI solutions, as this dissertation will mainly focus on a bias-diagnostic feature that is only available on Einstein Discovery, which is embedded in the Salesforce platform.

## 2.1 An Overview of CRM

CRM started as a process for achieving and maintaining strong customer relationships through customised individual treatment based on different customer behaviours and responses (Bohling et al., 2006). Throughout the years, CRM evolved into a crucial tool and part of companies' daily operations by creating long-term competitive advantage, improving strategy across departments - such as sales, service and marketing - and increasing profits (Nguyen and Mutum, 2012).

By collecting and analysing customer data provided, CRM manages to leverage this information and generate more value for a company simply by making each customer feel singular and relevant through the process of tailoring communications, thus increasing the retention levels - as it is more challenging to gain a new customer than it is to retain an existing one (Arab et al., 2010).

According to an early study, the role of a CRM varies depending on a company's size. For smaller enterprises, the main objective seems to be budget and process management, whereas, for larger enterprises, the most relevant components are improving strategy and value creation (Bohling et al., 2006). Nevertheless, there also seems to be a consensus on CRM's role in Value Creation, as 65% of respondents highlighted the importance of improving customer experience and strengthening the relationships between product and customer (Bohling et al., 2006).

The seemingly common main objective guiding a company's need for CRM implementation is linking customer information to products and services provided by the firm, with the desired outcome of gaining customer loyalty and retention (Arab et al., 2010).

Various factors impact a successful implementation, such as the value attributed to a product or the satisfaction levels the said product brings to the customers and their needs. Implementing a sustainable strategy may prove too tricky for some, but when executed correctly, it may result in an inimitable competitive advantage and a constant resource for success. By gaining knowledge and insights on customer data, a proficient CRM implementation gives a company the ability and resources to develop and perform an individualised, differential client strategy (Nguyen and Mutum, 2012).

As previously stated, CRM can be implemented and deployed according to a firm's needs, specifically electronic or technological CRM. Many choose to implement a standard solution with little customisation. This stems from a trial and error point of view. By starting with the conventional solution, a firm can test drive the technology without jeopardising its databases and customer records (Adebanjo, 2008). Nevertheless, this strategy may only be sustainable for small to medium enterprises (SMEs). This is because SMEs deal with less customer data on a day-to-day basis, resulting in fewer integrations and relationships to account for.

When analysing more prominent companies, the need for customisation (both for the customer and the company's operations) and scalability becomes a concern and challenge. Along with a standard solution and linking this to market-leading CRMs (such as Salesforce - discussed later in this chapter), it is necessary to purchase additional licenses, which leads to more costs for the company. Furthermore, we can affirm that an SME with fewer resources would not have the possibility to refine and customise its systems to an optimal solution (Adebanjo, 2008). Big companies would most likely have the upper hand on a medium to long-term implementation.

To summarise what has been stated in this section, we can identify a rich-get-richer cycle propelled by CRM and its many (hierarchical) features and capabilities (Adebanjo, 2008).

## 2.2 CRM Fairness and the CRM Paradox

One of the main advantages of CRM is that it continues to increase companies' profits and perceived value for the customer. In order to keep interest and engagement, a firm must adopt retention strategies, such as differentiated offerings based on the customers' behaviours and profiles. Pursuing this type of customised customer treatment can increase the probability of re-patronage. Furthermore, differentiated and individualised treatment is imperative in a CRM environment since companies will always have some customers who are more profitable than others. By relying on CRM technology, a firm can better organise its time and resources for the right customers. These strategies should result in higher customer loyalty and retention and a higher perceived market value (Yu et al., 2015). Nevertheless, personalised offerings may result in a perception of unfairness from a customer's point of view. Putting this in plain terms, a customer not categorised as a priority in a firm's strategic plans will perceive the firm's offerings as unfair and will most likely move on to a competitor. Moreover, it has also been established that the more differential treatment customers receive from a company, the less they reflect on a product or service's prices.

Contrary to this, the perceived value and quality of a firm's services increase the higher the personalised treatment (Yu et al., 2015), which becomes a paradoxical logic when defining an ethical and fair business strategy.

In light of these facts, adopting a differentiated and individualised strategy may result in a high potential for up-selling and cross-selling, higher profits, and long-term customer-company relationships. However, this leads to what's called the CRM Paradox, which states that separating and prioritising some customers over others will simultaneously bring higher value to the company and a higher perception of unfairness from the non-prioritised / non-high value customers (Nguyen and Mutum, 2012).

## 2.3 As-Is Bias Detection and Mitigation Solutions

Although this dissertation will focus on a feature embedded in a CRM, fairness solutions and tools are already being used to preserve ethics in machine learning and AI models. Firstly, we have IBM's AI Fairness 360 - an extensible open-source Python toolkit - designed to detect and mitigate bias in different algorithms, mainly for industry settings. It should offer a singular algorithm evaluation tool for engineers and data scientists. According to its developers, AI Fairness 360 provides a "*platform for researchers to share and benchmark their algorithms*" (Bellamy et al., 2018). Built on top of AI Fairness 360, Python's open-source Fairkit-learn toolkit is also used. Besides containing metrics and algorithms from scikit-learn and AI Fairness 360, it also provides extension points for adding more metrics and algorithms (Johnson et al., 2023).

From a more functional perspective, Microsoft introduced its Responsible AI toolbox, composed of four dashboards, each with a specific capability and functionality: *Error Analysis*, *Explanation*, *Fairness* and *Responsible AI*. It was designed to improve model assessment and better understand fairness-related issues (Microsoft, 2022). The "Explanation" and "Fairness" dashboards are also powered by Python's InterpretML and Fairlearn toolkits, respectively.

Similarly to the "Analyse for Bias" feature from Salesforce's Einstein AI - discussed in detail in section 2.4 - there is Fairness Flow from Meta, which is also a diagnostical toolkit developed to provide insights on different models' ethical performance. Meta leverages this toolkit across its platforms, like Facebook's advertisement system or Instagram's platform.

We would also like to highlight two bias-auditing toolkits: Fairml and Aequitas. Starting with the former, it is an R package that revolves around Generalised Linear Models and Penalised Regression to aid in creating fair and understandable models (Scutari, 2023). Finally, we have Aequitas, an open-source toolkit. As stated before, it is also an auditing tool whose main Goal revolves around " *enabling users to seamlessly test models for several bias and fairness metrics concerning multiple population subgroups*" (Saleiro et al., 2018).

## 2.4 Salesforce CRM

In today's technological world, CRM has many options. However, we keep seeing companies return to market leaders, such as Salesforce, which continues to be top-ranked throughout the years since its release. This section will focus precisely on this CRM.

### 2.4.1 Salesfore CRM Overview

The Salesforce CRM platform was introduced in 2014 and became a well-established solution in 2018 (Sunkari, 2022). It is a multi-tenant CRM, meaning it only has one database with data from several customers (tenants). This strategy seems to be to the liking of Salesforce customers since it is more economical to distribute costs across tenants instead of funnelling all maintenance charges to every user.

Salesforce is structured into multiple clouds, each with a different purpose, keeping a 360 view of the customer and seamlessly integrating customer records. By leveraging sales, marketing, or service cloud, the platform can aid companies in lead and opportunity creation, customer retention, and campaign management, as well as improve customer service with the help of case [escalation] management. Additionally, companies (users) can retrieve insights, analyse sales and market trends, and keep track of Key Performance Indicators (KPIs) with the reporting and dashboard tools (Salesforce, 2023a). These particular features are proven to be crowd-pleasers. Furthermore, companies can keep track of their customers' ever-changing demands and behaviours with quick access to their information and purchase and account history, which, when looking from a long-term relationship point of view, will enable the company to better serve customer requirements and possible issues. One lesser-known but still influential capability is the Chatter feature. This allows teams to communicate and help each other from anywhere in the world (Salesforce, 2023b).

Salesforce has a lot of room for customisation and scalability through processes such as custom objects and records, process automation through workflows, and even heavier customisation for programmers, using the Developer Console feature and Salesforce's object-oriented programming language, Apex. Additionally, Salesforce provides users with the App Exchange, with multiple free and paid tools and apps, each oriented to different Salesforce clouds and market objectives (Sunkari, 2022). Since its release, Salesforce remains the market leader in the pool of platforms giving Software as a Service (SaaS) (Thakkar and Rajaan, 2020).

Throughout the years we have seen Salesforce switch from Classic to the Lightning Platform, thus becoming more user-friendly and with a better User Experience / User Interface (UX / UI) design. This new improvement allowed Salesforce to enter the Platform as a Service (PaaS) market. Moreover, it opened the doors for new tech positions in the workplace, as the Lightning Platform can be customised in a low code / no code manner (Sneh et al., 2018). This means that companies could hire skilled programmers and more functional staff proficient in the platform itself.

To conclude, it is also worth mentioning the learning community that Salesforce offers (Sneh et al., 2018). With resources like **Trailhead** and **Partner Learning Camp**, developers can train

and learn new skills whenever they change to a new project or want to expand their knowledge and prepare or study for upcoming certification exams.

### 2.4.2 Salesforce Einstein Analytics

This section investigates Salesforce's more advanced capabilities and functionalities, such as the **Einstein** module (Salesforce, 2023d).

Einstein is Salesforce's AI for CRM feature. As the CRM platform itself, it seems to be a user-favourite and highly ranked amongst competitors. This component allows for more customised content, such as:

1. Insightful dashboards;
2. Data-based customer recommendations and predictions;
3. Generated content with the new GPT component;
4. Conversational content (Einstein Copilot) allows users to interact with AI to improve task efficiency.

Since Einstein is native to the platform, it reaches all the data stored in the CRM clouds and applications to leverage insights. As of March 2023, and as mentioned in the item list above, Salesforce can now produce AI-generated content with Einstein GPT.

#### 2.4.2.1 As-Is Einstein Applications in Different Clouds

As Salesforce features and capabilities are distributed and assigned to different Clouds, so are Einstein features. In this section, we will cover key Einstein features and their requirements for sales, service, and marketing, in addition to some features for CRM analytics and Admin/Developer specifics.

**Sales Cloud:** Salesforce Sales Cloud is directed towards leads and opportunity management, which, from the point of view of Einstein or translated to an AI setting, should mean less time spent on leads, which eventually would fall through, less time writing emails or even better and more accurate customer dashboards. To summarise, it should help users prioritise leads more likely to convert and spend less time on more routine tasks. Currently, Einstein for Sales has the following key features (Salesforce, 2023c), as listed in Table 2.1.

**Service Cloud:** Salesforce Service Cloud was designed to tackle customer cases and issues. The role of Einstein in this cloud is mainly automating and prioritising case management. Currently, Einstein for Service has the following vital features (Salesforce, 2023c), as listed in Table 2.2.

Table 2.1: Einstein for Sales Cloud - Key features.

| **Feature** | **Description** | **Requirements** |
|---|---|---|
| Einstein GPT | Automate routine tasks | Sales Cloud;<br>Einstein Conversation Insights |
| Deal Insights<br>for Pipeline Inspection | Determine which leads<br>will most likely not be<br>closing in the current month | >=1000 records with >=120 converted<br>in the last 6 months for a local model |
| Lead Scoring | Prioritise leads which<br>are most likely to convert | >=1000 records with >=120 converted<br>in the last 6 months for a local model |
| Opportunity Scoring | Prioritise opportunities<br>which are most likely to convert | >=200 closed-won and 200 closed-lost records<br>in the last 2 years, with a lifespan of at least 2 days<br>for a local model |
| Next Best Action | Deliver recommendations<br>at the optimal time | User-specific requirements<br>like browser or device |
| Activity Capture | Automatically capture data<br>and add to the CRM | >=30 accounts, contact or leads;<br>Compatible with Gmail or Microsoft 365 |
| Forecasting | Predict sales forecasts<br>inside the CRM | Collaborative Forecasting Enabled; Standard fiscal year;<br>Hierarchy has to include at least one forecasting enabled user who<br>reports to a forecast manager;<br>Opportunities must be in Salesforce >=24 months |
| Email insights | Prioritise emails in the inbox | Activity Capture enabled |
| Recommended Connections | Get insights on team members' network<br>to understand who knows a specific customer,<br>and is able to help out on a deal | >=2 users to be connected<br>to Activity Capture and Inbox |
| Sales Analytics | Get insights from sales KPIs | User specif requirements like browser or device |

**Marketing Cloud:** Regarding Salesforce Marketing Cloud, the platform is prepared to tackle customer behaviour and profiling, personalised content (like messages and emails), optimal suggestions, and engagement timing. Currently, Einstein for Service has the following key features (Salesforce, 2023c), as listed in Table 2.3.

**CRM Analytics:** Aligned with Salesforce Tableau, CRM Analytics aims to give actionable insights through dashboards, reports, KPIs, and Machine Learning (ML) functionalities. Einstein for CRM Analytics currently has the following vital features (Salesforce, 2023c), as listed in Table 2.4.

**Admins and Developers:** Through Einstein, Admins and Developers can build customised solutions with the help of Einstein's AI and its low code/no-code features. Currently, Einstein for CRM Analytics has the following key features (Salesforce, 2023c), as listed in Table 2.5.

#### 2.4.2.2 As-Is Einstein Bias Detection Functionalities

Before getting into the specifications of Bias Detection in Salesforce's Einstein, it is relevant to highlight common biases that may appear in AI models, according to Salesforce's Learning Resources (Salesforce, 2024e):

1. **Dataset Bias**: Also called Measurement Bias, it refers to when our data contains mislabelled columns or under/over-representation of a group in the Dataset. This issue can have a significant impact on predicted outputs. When we feed data to the model, if something was mis- or underrepresented in our training set, the expected outcome could be highly biased or just straight incorrect. Putting the case in a simple example: if every woman in the training

Table 2.2: Einstein for Service Cloud - Key features.

| **Feature** | **Description** | **Requirements** |
|---|---|---|
| Einstein GPT | Automate routine tasks | Service Cloud Unlimited Edition;<br>Service Cloud Einstein Unlimited Edition |
| Bots | Automate common questions and doubts | Messaging channels;>=20 examples per language |
| Next Best Action | Deliver recommendations at the optimal time | User-specific requirements like browser or device |
| Case Classification | Automate case classification<br>and improve case field accuracy | >=400 closed cases with description and subject fields populated |
| Case Routing | Automate case triage | >=400 closed cases with routing fields populated |
| Article Recommendations | Select the best articles in real-time<br>to solve customer questions | >=3 knowledge articles in English and use of Lightning Console;<br>Org-specific model training: 500 active articles |
| Reply Recommendations | Decreased time spent writing responses<br>for common questions | Messaging channels;<br>>=1000 chat transcripts and use of Lightning Console |
| Service Analytics | Get insights from contact centre operations<br>and customer profiles | User-specific requirements like browser and device |

data has pink as their favourite colour, when we feed the model data to make predictions, as this data contains women, then the model will most likely predict pink as every woman's favourite colour;

2. **Association Bias**: Regarding the previous example of mislabelled data, we can single out this bias when we label or interpret data according to our beliefs, opinions or stereotypes just because they have proven accurate in past events. We can also identify **Generalisation Bias** in this mix since we often believe that what is valid for one person is true for everyone with the same or similar characteristics;

3. **Confirmation Bias**: It can happen when our models are trained and tuned to rely heavily on pre-existing human notions and beliefs or trends in the data. Additionally, it can refer to an interpretation bias by humans when we analyse our data as a confirmation of what we already believed to be accurate in the first place;

4. **Automation Bias**: This specific bias can be translated into three main things: over human reliance on automated results and outcomes, even if other non-automated data tells us otherwise; the computational assumptions that things will not be that different in the future from what they are now; feeding flawed data to our model;

5. **Societal Bias**: When the outcomes in models perpetuate societal or historical bias;

6. **Survivorship Bias**: Occurs when the model is highly focused on variables and data that "survived" specific selection processes whilst ignoring the rest of the data.

So, how does Einstein mitigate (and try to eliminate) the appearance of bias in its models' results? Einstein offers the 'Einstein Discovery' solution, which should allow users to identify patterns and correlations in their data (Salesforce, 2024d). The "**Analyse for Bias**" feature is embedded into Einstein Discovery. Considering a dataset with potential bias variables like age, gender, ethnicity or nationality, we may conclude that our results could be skewed, biased, or even discriminatory towards a specific group.

***Analyse for Bias*** is supposed to let the user flag potential bias variables and identify correlations between the bias variables and others in the data. After the analysis or training, Einstein

Table 2.3: Einstein for Marketing Cloud - Key features.

| Feature | Description | Requirements |
|---|---|---|
| Einstein Overview | Overall view of all activated Einstein features' performance metrics | Any Einstein for Marketing feature |
| Content Creation | Generate emails and test different message versions | Engagement Corporate or Enterprise Editions; Feature enabled in Setup;Einstein Copy Insights enabled |
| Engagement Scoring | Score customers likelihood to interact with messages sent or to convert via web | Use of the Collect tag for web conversion predictions |
| Engagement Frequency | Predict how many emails to send to customers | Einstein Engagement Scoring |
| Splits | Assign customers to the right journey according to their behaviours or profiles | Einstein Engagement Scoring;Journey Builder |
| Content Selection and Tagging | Automate real-time message penalisation and image tagging | Image-based content |
| Send-Time Optimization | Select Optimal Timing to send emails and messages to subscribers | Account Engagement advanced;Premium Editions |
| Behaviour Scoring | Predict the likelihood of a customer to have interest in a product based on behaviour and engagement patterns | >=6 months of prospect engagement data and >=20 prospects linked to opportunities |

should be able to "***report its findings for disparate impact and proxy variables***" (Salesforce, 2024d), where "disparate impact" refers to data that could lead to discriminatory practices towards a specific group, and "proxy variables" are the ones correlated to the previously flagged ones. This topic will be discussed throughout this dissertation.

#### 2.4.2.3 Einstein Discovery Scope

Einstein Discovery is the solution of the AI feature that allows us to build and make predictions. This solution also includes the **Analyse for Bias** feature. According to Salesforce (Salesforce, 2024b), Einstein Discovery's offer covers the following use cases:

1. **Regression** for numeric outcomes;
2. **Binary Classification** for boolean outcomes, which will be the one selected for this thesis;
3. **Multiclass Classification** for various outcomes, between 3-10 possible results.

This module also allows users to select the data columns they find more relevant for the specific task. To gain insights into the data, the user needs first to create and train a model. After that, Einstein will provide insights and suggestions for improvement. It will also list which variables were the "**Top Predictors**", meaning the variables that had the most impact in the model when making predictions. Furthermore, and in alignment with the suggested improvements, Einstein offers "**Disparate Impact**" insights, highlighting potential bias cases and proxy variables highly correlated with potentially biased ones. Lastly, Einstein Discovery also provides a few model templates. It builds an initial training dataset for everyday business tasks such as maximising customer revenue, maximising deal-winning rates, or minimising the time to close a deal.

To date, in order to successfully create an Einstein model, there are a few required steps, according to **Trailhead**, Salesforce's official learning and training platform:

1. Defining a **Target Outcome**;
2. Building and Preparing the **Dataset** which will be used in future predictions. This is a crucial step since it is here that the users select the columns with a higher analytical value,

Table 2.4: Einstein for CRM Analytics - Key features.

| Feature | Description | Requirements |
|---|---|---|
| Analytics Studio | Build non-existing Applications(Apps), or leverage pre-built templates | Data for custom dashboards;<br>Data from Salesforce Objets for pre-built templates |
| Sales and Service Analytics | Analytics Apps with pre-built KPIs | Sales and Service Cloud data |
| Einstein Discovery | Create custom ML predictions with Salesforce<br>and non-Salesforce data | >=500 rows of data for prescriptive and<br>>=400 rows of data for descriptive per dataset |
| Text Clustering | Extract keywords from large texts using ML | Einstein Engagement Scoring |
| Live prediction with Snowflake | Predictions powered by Snowflake datasets | N/A |
| Prediction Builder | Create custom predictions on any Salesforce object<br>with automated ML, without code | Data in Salesforce objects;>=400 example records with<br>>=100 yes examples and >=100 no examples for binary predictions;<br>>=400 example records for numeric predictions |
| Discovery for Reports | Analyse report data and summarise key takeaways | Lightning Report with >=500 rows of data |

which will have more impact on the final predicted outcomes. Einstein allows users to extract or upload data to the platform, as well as transform said data and make additional configurations;

3. Creating and selecting a **Model**. When creating a model, Einstein will ask the user the Goal of the outcome, which refers to the column values to predict, and if the result should be maximised or minimised, depending on the problem's context. Additionally, there is also an option for creating model versions, meaning that, as the users make improvements or changes to the model, they can be stored in a new version and reflected in the specific version's re-training;

4. **Evaluating** the model. Through the help of performance metrics, the user can measure the quality of a model, meaning that it is possible to understand how useful a model can be in predicting the desired outcomes. Regarding the data which will be used in this thesis and the binary predictions attached to it, the best metrics offered by Einstein to assess model quality are:

   (a) **Area Under the Curve (AUC)**, which is also related to a model's accuracy, refers to the ability of a binary classifier to distinguish between classes (Google, 2024b). This metric should be above 0.5 and below 1.0;

   (b) **Threshold**, corresponding to a defined value - determined for each task in question - which specifies the limit (decision threshold) to where a prediction is true or false. This means that a value above the Threshold is considered true, and a value below the Threshold is considered false. Additionally, AUC will essentially summarise the performance of a model across all the possible classification thresholds.

   In addition to performance metrics, Einstein also allows users to observe the distribution of outcomes. It displays how many true or false values the model predicted, plus a selection of the Top Predictors, which are the variables with the highest correlation to the predicted outcomes.

5. Analyse and Explore **Data Insights**. Insights correspond to information found in the data through training. The Einstein Discovery module explores three types of insights:

Table 2.5: Einstein for Admins & Developers - Key features.

| Feature | Description | Requirements |
|---|---|---|
| Prediction Builder | Create custom predictions on any Salesforce object with automated ML, without code | Data in Salesforce objects;>=400 example records with >=100 yes examples and >=100 no examples for binary predictions; >=400 example records for numeric predictions |
| Recommendation Builder | AI-powered recommendations | >=100 records for Recipient Records; >10 records for Recommended Item Records; >=400 records for Positive Interaction Examples |
| Einstein Search | Decrease search time | Core CRM license |

(a) **Descriptive**, that describe what happened in the past with our data. This could translate to different patterns that were found;

(b) **Diagnostic**, that tries to find patterns in the data to explain why something happened in the data;

(c) **Comparative** ones. This type of insight compares the impact of a specific variable on the predicted outcome with other factors or variables.

By leveraging insights, the user can also investigate potential bias variables and associated insights.

6. Perform changes suggested by Einstein through **Improvements**. After the first training, the user will receive dashboards with suggestions for possible model improvement. These can range from removing a variable from the Dataset to grouping values or variables that similarly impact the predicted outcome.

# Chapter 3

# Preliminary Experiments

In this chapter, we will walk through the first experiment made to test Einstein Discovery's capabilities on a less theoretical and more practical level. The Dataset used for this experiment was retrieved from one of Salesforce's recommended data platforms - data.world[1]. The "**Census Income**" Dataset's main objective is to "***Predict whether income exceeds $50K/yr based on census data***", according to UC Irvine's Machine Learning Repository. The variables included in it are the following stated in Table 3.1 (Becker and Kohavi, 1996).

Table 3.1: Census Income Dataset Variables.

| **Variable Name** | **Variable Type** | **Categorical Values** |
|---|---|---|
| Age | Continuous | - |
| Work Class | Categorical | Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked |
| Final Weight | Continuous | - |
| Education | Categorical | Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool |
| Education - Numeric | Continuous | - |
| Marital Status | Categorical | Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse |
| Occupation | Categorical | Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces |
| Relationship | Categorical | Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried |
| Race | Categorical | White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black |
| Sex | Categorical | Female, Male |
| Capital Gain | Continuous | - |
| Capital Loss | Continuous | - |
| Hours-per-Week | Continuous | - |
| Native Country | Categorical | United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands |

[1] https://data.world

Not all variables were considered for training in Einstein Discovery. The following sections will explain the processes and best practices needed to train a model in Einstein Discovery (Salesforce, 2024c). Moreover, there will also be a "Machine Learning Fundamentals for Binary Classification" section to give more context to the Dataset used for the thesis and other relevant metrics mentioned throughout the following chapters.

## 3.1 Machine Learning Fundamentals for Binary Classification

In this dissertation, we will conduct several experiments on Einstein Discovery with the Census Income, which will mainly predict one of two outcomes: Income greater than 50,000 dollars or less than or equal to 50,000 dollars. This model type will be a Binary Classifier, where the classification tasks will have only two class labels (Brownlee, 2020a). In binary classification, various algorithms are commonly used to classify data into two distinct categories. Logistic Regression is a widely used algorithm that predicts a binary outcome based on one or more predictor variables. It models the probability of the binary outcome using a logistic function. Support Vector Machines, Decision Trees, K-Nearest Neighbours, and Naive Bayes are popular and commonly used algorithms (Brownlee, 2020a). These algorithms are trained on labelled data and used to predict new, unseen data. As reviewed, Logistic Regression and Support Vector Machines are inherently designed for binary classification. At the same time, other algorithms, such as K-Nearest Neighbours and Decision Trees, can also be used for this purpose.

When evaluating the performance of binary classification algorithms, several metrics can be used to assess model effectiveness and quality. Standard performance and evaluation metrics include Accuracy, Precision, Recall, F1-Score and AUC. Another metric is the Matthews Correlation Coefficient (MCC), also used in Einstein (D'Agostino, 2022). Accuracy measures the percentage of correctly predicted outcomes or class labels (Google, 2024a). Precision and Recall are two metrics that go hand-in-hand: whilst Precision refers to the rate of positive values correctly predicted, Recall measures the percentage of actual positive values identified correctly (Google, 2024c). These two metrics are also related to Accuracy. Moreover, there is the F1-Score, which is the harmonic mean between Precision and Recall. The harmonic mean essentially corresponds to the arithmetic mean, but it uses the reciprocal numbers instead (GeeksforGeeks, 2023). Finally, the AUC refers to the ability of a binary classification model to distinguish between positive and negative classes (Google, 2024b).

## 3.2 Creating a Model from Start to Finish in Einstein Discovery

### 3.2.1 Creating a New Dataset

After creating an Application (App), which is Einstein's version of a folder to store the model, the user must make a Dataset. This does not mean creating a Dataset from scratch but rather creating or uploading it to the platform, as shown in Figure 3.1. The initial steps for creating a dataset in

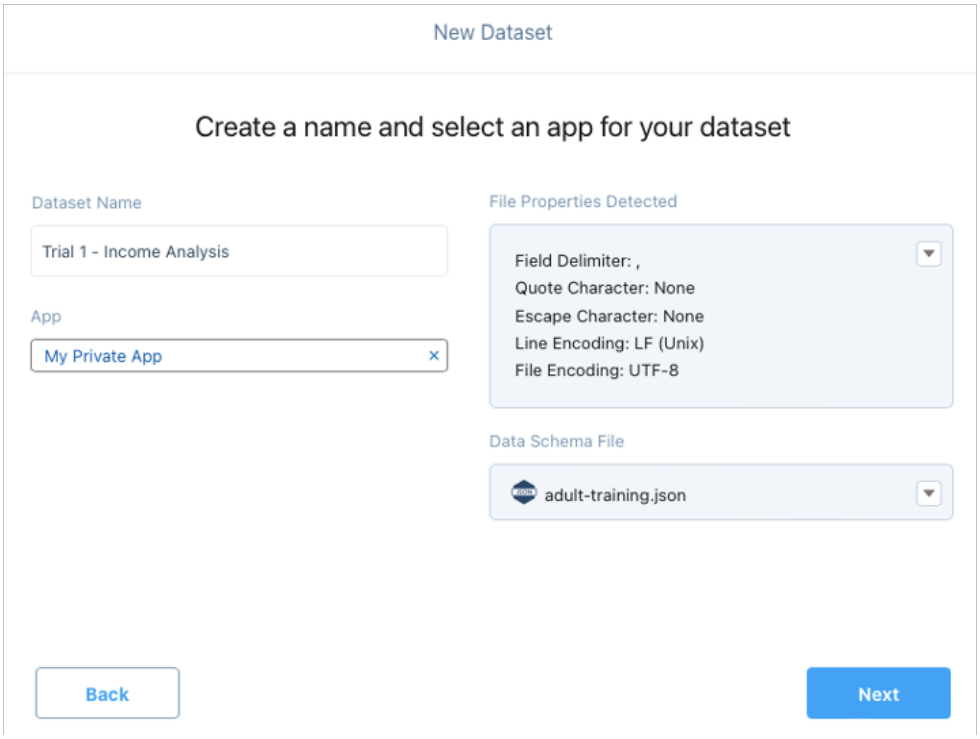


Figure 3.1: Storing a new dataset in Einstein Discovery.

Einstein consist of defining its name in the platform, which App (folder) it will be stored in and most importantly, its properties, such as the file delimiter and the type of encoding. For the first experiment, we created an App called "**My Private App**" - since Apps can be private for a single user or shared with multiple users - for storing the initial testing model. Secondly, aligned with what was stated before, since the name in Einstein does not need to be the same as the uploaded file, we decided to identify this Dataset as "**Trial 1 - Income Analysis**" for a straightforward and self-explanatory name.

When clicking ***Next***, the user can now see the uploaded file in an interactive table format, as shown in Figure 3.2. It's in this step that the first changes or improvements take place. The user can change or add a label to the different columns and select a variable/field type. This will be the basis for the rest of the configurations in the model. It is also to note that if the Dataset in question has a great range of variables, the ***Search fields*** functionality reveals a great usefulness in browsing through all the dataset fields. In this first trial, there was a need to label each column of the Dataset - in Einstein - according to the descriptions provided in data.world's and UC Irvine's repositories. Each column was named according to the information in Table 3.1. As for the ***Field Types***, Einstein seemed to correctly identify each one, which led us to leave each column selection as-is.

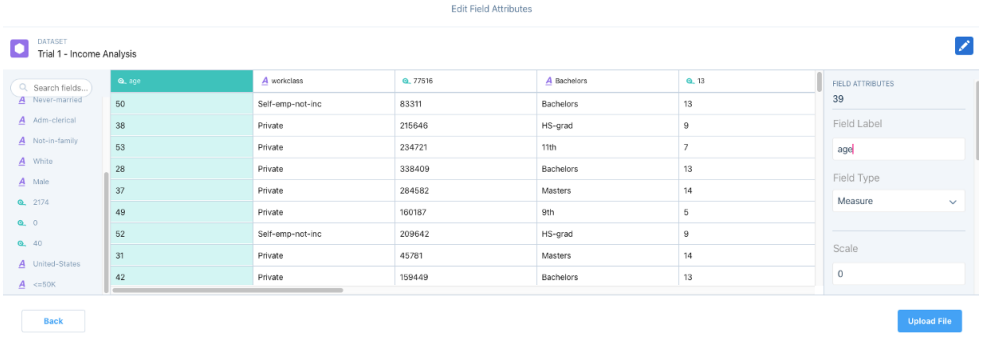


Figure 3.2: Creating a new dataset in Einstein Discovery.

### 3.2.2 Creating a New Model

As shown in Figure 3.3, in order to create a new model, the user needs to select what the model is going to predict, as well as if the result should be maximised or minimised - designed as a solution to more practical use cases, such as ***maximising customer satisfaction*** or ***maximising the number of converted leads*** - which in this case had no use, but since it was a required selection, there was not a way of skipping it (this can also be identified as an improvement point for the platform's flexibility and adaptability with external data sources). Additionally, like with the new Dataset, there needs to be a model name and an App to store it. For the first trial, the variable chosen for the ***I Want to Predict*** component was "**predicted_income**", which corresponds to a binary prediction: an income less than or equal to 50,000 dollars, or greater than 50,000 dollars. After that, the objective chosen was to maximise the predicted income. Finally, the model should be stored in the same App as the Dataset.

After creating the model, Einstein provides two options for configuring a model's columns, as stated in Figure 3.4: ***Automated*** - in which Einstein chooses the "most relevant columns" aligned with what was selected in the previous step - or ***Manual*** - in which the user selects which columns should be considered for model training. In our trial, the manual option needed to be chosen to lead us to the interface where we could flag variables for bias analysis, as we will see in the following step.

### 3.2.3 Configuring the Model's Settings

Once the ***Manual*** option is selected, the user encounters an interface like the one in Figure 3.5. Here, we can configure everything that Einstein needs before model training. Firstly, we can confirm the number of rows in the Dataset, which is quite relevant information since - according to user guides and literature - Einstein needs a **minimum of 400 rows** in most cases to train a model properly. In this trial, the Dataset had **32560 rows**, meaning it met the requirements for successful training. Besides the list of variables to choose from, we also have some correlation information and data alerts, which can be warnings or suggestions from Einstein to improve our model. At least three variables must be selected to start training, one being the outcome variable.

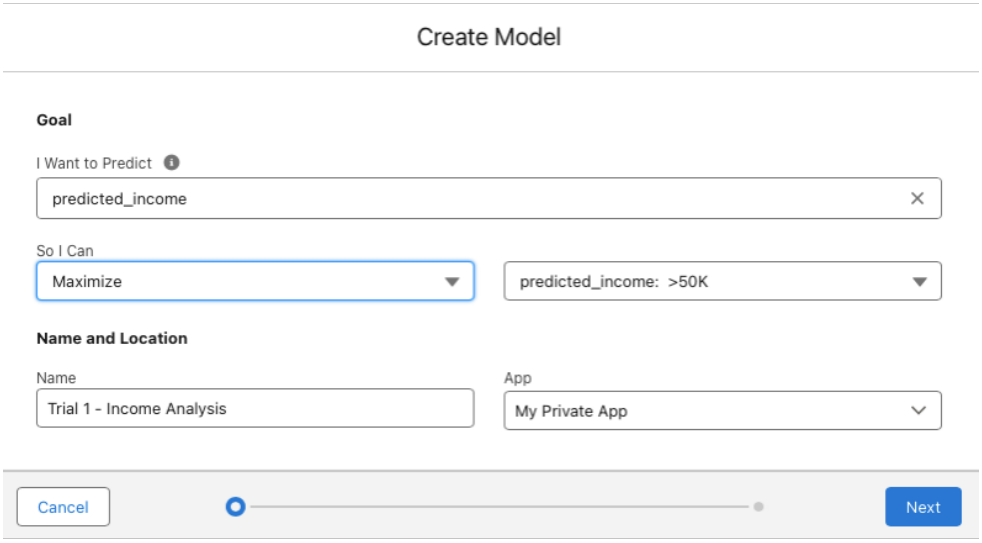


Figure 3.3: Creating a new model in Einstein Discovery.

On the upper right corner - see Figure 3.5 - we have the **Analyse for Bias** functionality. The first step is to select the corresponding checkbox in order to be able to flag the variables for analysis, insights and disparate impact. The variables flagged for bias analysis were: ***age, hours_per_week, native_country, marital-status, occupation, education, workclass, race, sex, relationship***. Flagging a variable for bias means the user is letting Einstein know that a particular variable could lead to misleading or biased predictions. Putting this into the context of our data for this first trial, and according to some frequently preconceived societal notions, variables such as sex, age, race, or even native country could lead to wrongly predicted income values (for example, the expected income for a young woman being less than 50,000 dollars when in reality it is more significant than 50,000 dollars).

In addition to the already reviewed possible configurations, Einstein also allows users to select an algorithm to train with and which validation to select. For the first trial, we decided to leave the default selection, which was the **Generalised Linear Model (GLM)** algorithm and **Automatic Cross Validation**. The GLM algorithm can be described as an extension of Linear Regression that allows for the modelling of exceptional outcomes, which may not be considered in the principle algorithm (Molnar, 2023). Available for algorithm selection was: ***GLM, Gradient Boosting Machines (GBM), XGBoost, Random Forest*** and ***Model Tournament***. For validation, we could choose between ***Automatic Cross Validation, Training/Validation Ratio*** or ***Validation Dataset***. Moreover, it allows a model type selection, which had to be a **Binary Classifier**.

### 3.2.4 Checking Insights and Disparate Impact Alerts

After one training time, the **Data Insights** side tab becomes available. Here, we get access to different graphs regarding various alerts or suggestions that can be applied to the model and Dataset and tested in a second training. Einstein will highlight which variables had the most negative or

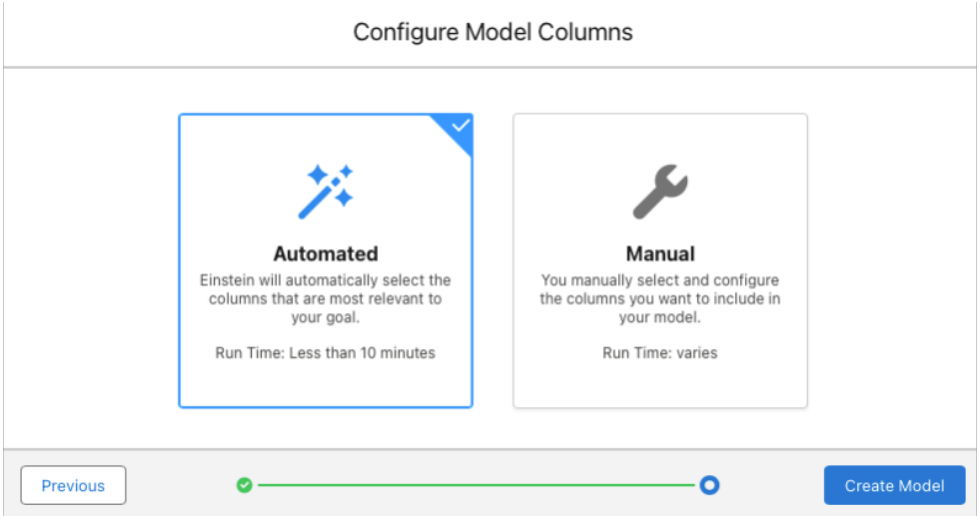


Figure 3.4: Configuring the model's columns.

positive impact on the predictions and which values were most relevant to the outcome. Furthermore, it is also in **Data Insights** that the bias analysis becomes available. For the income analysis, Einstein highlighted ***age*** as the primary source of disparate impact. Along with this find, there are two graphs and related information, as shown in Figures 3.6 and 3.7. In Figure 3.7, we can conclude that the **45 to 50** age range was predicted to have an income greater than 50,000 dollars, **15.83%** more than the global average. In total, **39.91%** of this age range was predicted to have an income greater than 50,000 dollars. On the opposite side, the **17 to 22** age range was the worst predictor for a predicted income greater than 50,000 dollars, with only **0.46%** of the total age range being predicted to have an income greater than 50,000 dollars. According to these insights, **hours_per_week** was the most impactful variable on these outcomes, as an older age range seemingly has more work hours than a younger cohort (1-24 hours and 36-40 hours for a younger and older cohort respectively). Essentially, more work hours combined with an older age leads to a higher probability of a predicted income greater than 50,000 dollars. Einstein also allowed us to ignore the age variable alerts or exclude the highlighted variable entirely.

Finally, there was also a small dashboard divided into two graphs, each answering one of the following questions:

1. When did predicted_income: > 50K occur the most?
2. When did predicted_income: > 50K occur the least?

By analysing the graphs on Figure 3.8, we can safely conclude that - according to Einstein - a predicted income greater than 50,000 dollars occurred the most when the work hours were between **48 and 55** (**44.6%**) and it happened the least when the age was between **17 and 22 years** (**0.5%**). These numbers align with the disparate impact analysis and insights, which highlighted age and hour_per_week as the most relevant variables affecting the predicted outcomes, with the importance of **83.28% for age** and **16.72% for hours_per_week**. We can also conclude that some

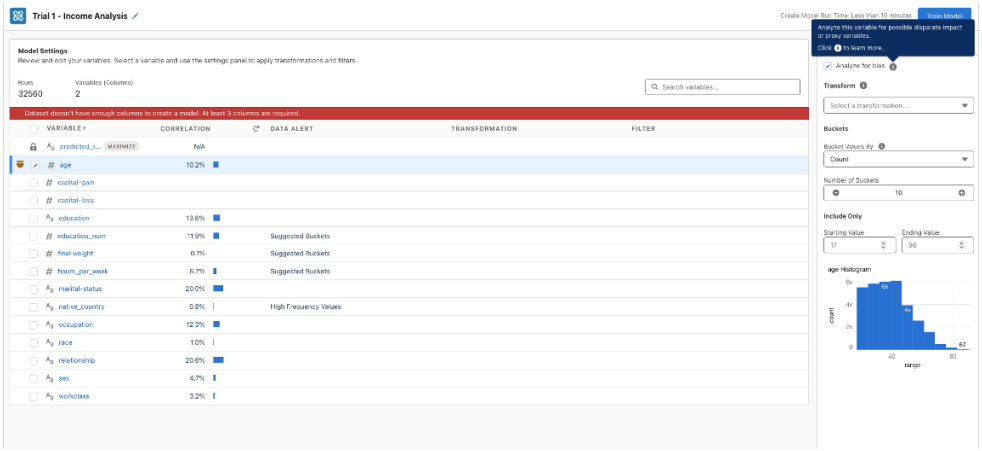


Figure 3.5: Configuring the model's settings.

ranges of hours or age have a different impact when both variables are combined in observation, as is the case of the second insight of the right-hand side graph on Figure 3.8: **hours_per_week between 35 to 40 and age between 22 to 26**. On average, **24.08%** of each age cohort was predicted to earn an income greater than 50,000 dollars.

### 3.2.5 Evaluating Model Performance

The final step in this process is the model evaluation. For this, we can navigate towards the ***Performance*** tab on the left-hand side of the model overview. Here, we encounter four distinct tabs:

1. **Overview**: where we have a summary of the model's performance. We can see the models **Accuracy** and **Threshold**, **Top Predictors** and the **Distribution of the Outcome Variable**, as well as an **Assessment of Deployment Readiness** should our goal be deploying the model to a Production environment. It's also in the **Assessment of Deployment Readiness** card that we can review possible **data alerts** highlighted by Einstein.

2. **Model Evaluation**: divided into 4 tabs of its own - ***Overall Performance***, ***Gain and Lift***, ***Cross Validation*** and ***Coefficients*** - , it provides information regarding different metrics such as **Accuracy**, the **GINI Index**, **AUC** and the **MCC**, or even **Log Loss**. Moreover, Einstein offers help text for each evaluation metric provided, describing each metric and its interpretation to the user. Whilst metrics like AUC, GINI and MCC appear in progress bar-like graphs, **Accuracy Analysis** is presented in a table-like mannered text followed by a brief description of each percentage meaning for **True Positive Rate**, **False Positive Rate**, **True Negative Rate** and **False Negative Rate**, along with the **Maximum Accuracy** percentage value for the respective model and data. Furthermore, for the ***Coefficients*** analysis tab, there is also an option to download a Comma-Separated Values (CSV) file of the findings.

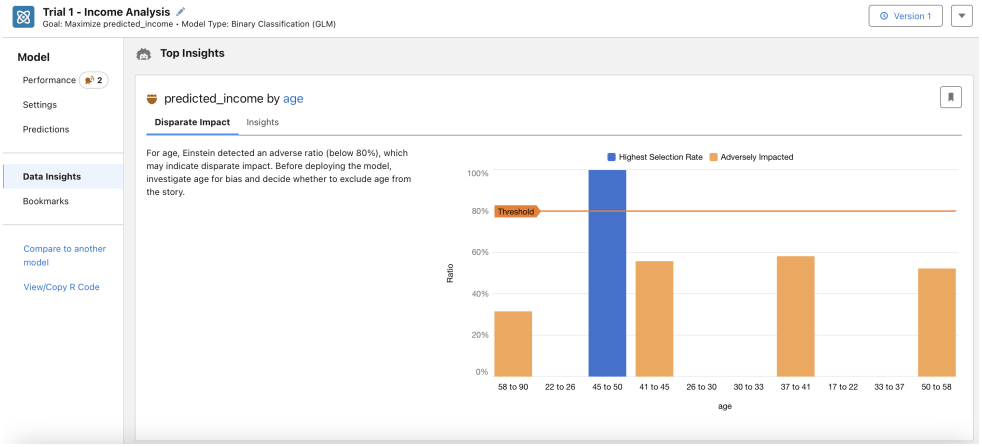


Figure 3.6: Checking disparate impact for *age*.

3. **Prediction Examination**: this tab contains a sample of 100 observations from the training data where the user can select each of them and get more in-depth insights for its outcome. In a more detailed way, for each observation, Einstein provides a card where there is information regarding the **Predited versus Actual Outcome**, **Score** and **Threshold**, **Top Factors**, in addition to the possibility of optimising for a specif metric - the user can choose from: ***Custom***, ***Accuracy***, ***MCC***, ***Informedness***, ***Markedness*** and ***F1-Score***.

4. **Threshold Evaluation**: in this final tab, we have a summary of the model performance, with a confusion matrix and key metrics' values available. Additionally, we get a threshold value, which according to Einstein is *"a value ranging from 0 to 1"*, which *"represents the trade-off between the true positive and false positive rates"*. Furthermore, there is also the option of selecting a specific metric to optimise and define a cost ratio (Einstein provides help text for this).

For this first experiment, we got the following results, stated in Table 3.2 - a Confusion Matrix - and Table 3.3.

Table 3.2: Income Prediction - Confusion Matrix.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 1337 True Positives (TP) | 1059 False Positives (FP) | **55.8% Positive Predicted Value** |
| | **<=50K Negative** | 6504 False Negatives (FN) | 23660 True Negatives (TN) | **78.44% Negative Predictive Value** |
| | | **17.05% True Positive Rate** | **95.72% True Negative Rate** | |

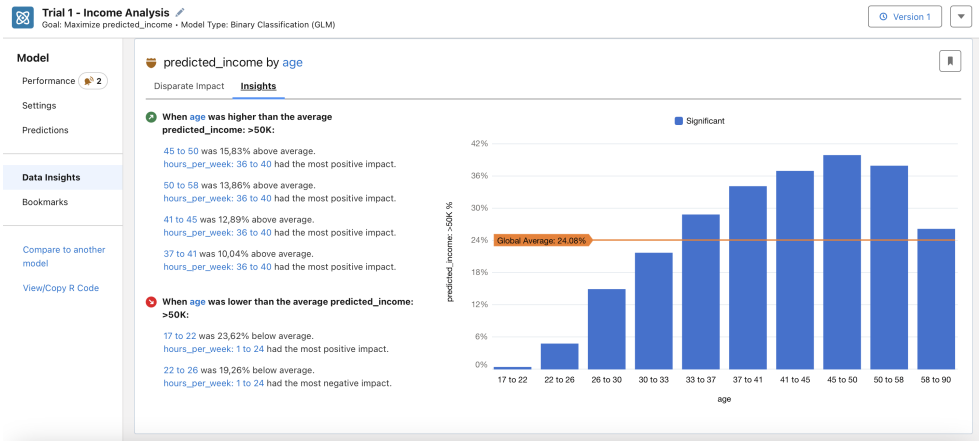


Figure 3.7: Checking disparate impact insights for *age*.

### 3.2.6 Following every Einstein Suggestion

As stated before, after training the model for the first time, Einstein will make suggestions to improve model performance and mitigate potentially biased results. We received **two Data Alerts**: one for the ***age*** variable, for Disparate Impact, and the other was a "***Suggested Buckets***" model improvement suggestion. For the ***age*** alert we can also identify it as **Endogeneity Bias**, rooted explicitly in simultaneous causality (Zaefarian et al., 2017) since we can see that different variables' values can cause each other - i.e. reciprocal causal effects - like the case of ***age*** and ***hours_per_week***. These alerts become more straightforward to understand by analysing Figures 3.10 and 3.9. The next action point was to follow these two suggestions and see the impact of these changes on the new model version's performance. Therefore, we applied the suggested buckets for **hours_per_week** and excluded the **age** from the dataset and model training in Einstein, *ceteris paribus*. To be able to leverage these two improvement suggestions, we had to accept and apply them simultaneously, meaning that if we had only used one of them, Einstein would consider the other as ignored or not accepted, and there is a high probability that the same suggestion would not appear in the new model version. After this, the model needed to be re-trained to have new results compared to the first ones.

When comparing the first results obtained for evaluation metrics with these new ones, we concluded every metric had an increase in value, even a slight one. By analysing Table 3.5, we can see that **Accuracy had an increase of approximately 0.02**, and that the most significant increases occurred in **Recall, Precision** and **Negative Predicted Value**, as well as the **F1-Score**, which went **from 0.261 to 0.447**. Furthermore, and by looking at Table 3.4, we can conclude that the number of True Positives and Negatives and False Positives and False Negatives is slightly more balanced. We can also conclude that there was an increase in percentage in the **Positive and Negative Predictive Values**, along with the **True Positive Rate**. The **True Negative Rate**, which refers to the number of predicted negatives among all the actual negatives, was the only percentage value that suffered a slight decrease. To illustrate this conclusion, we can look at the value for the Negative Predictive Value, which **increased from 78.44% to 81.9%**, meaning that the number of

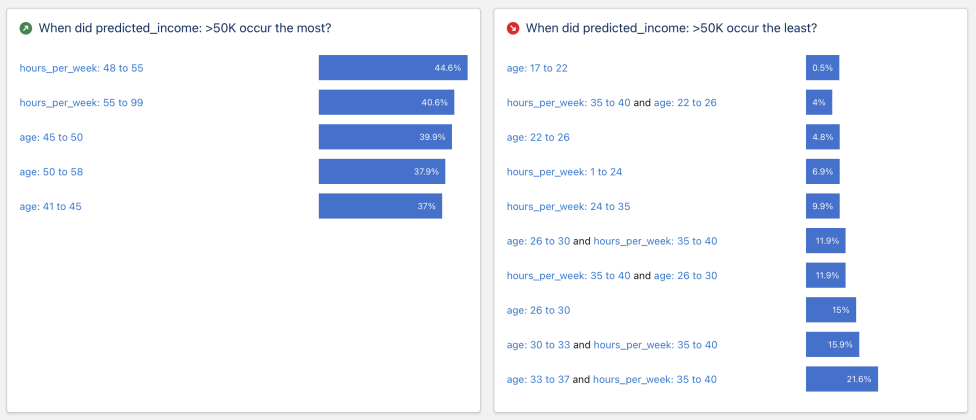


Figure 3.8: Checking which variables had a bigger impact.

actual negatives among all predicted negatives increased.

Furthermore, and as expected, the ***Model Quality*** evaluator improved, now stating that the model is of *Good Quality* and that there were no "**Major Issues Detected**", when previously we had two issues, one being the ***Disparate Impact*** alert for *age*, which we excluded from this new training version. Even though bucketing the *hours_per_week* variable was one of the two suggestions in the first version, Einstein suggested new buckets for this variable as an improvement. As for **Top Predictors**, we saw ***education-num*** replacing *age* with an **Importance of 71.05%**, and ***hours_per_week*** remaining in second place, this time with an **increase in Importance, from 16.72% to 28.95%**. Finally, there were no **Disparate Impact** alerts, even with ***native_country***, ***marital-status***, ***sex*** and ***race*** still flagged for bias. Based on this, we can safely assume that Einstein did not seem to detect that any of these variables would have an unfair impact towards the predicted income for each observation. This also could be a little paradoxical since we did not see a significant increase in Accuracy. However, we could also argue that other variables that were not marked for bias in both versions could be the ones affecting this and other metrics. Some examples can still be: ***education***, ***occupation*** or ***workclass***. These assumptions will be targeted for deeper analysis in the remaining work for this project.

## 3.3 Initial thoughts and Conclusions

Throughout this chapter, we followed a step-by-step approach to creating an Einstein Discovery Model from start to finish. This first experiment was merely an initial step to understand better the model training process and the extent of machine learning and data science knowledge required. Therefore, at every possible selection, we decided to leave the default values, except for the model type, which needed to be a **Binary Classifier**. This aimed to understand the tool's behaviour and which type of insights and dashboards would come up with minimal settings and configuration customisation. From this first experiment, we arrived at the following conclusions for the **platform and its interface**:

Table 3.3: Key Performance Metrics Results.

| Metric | Einstein Description | Result |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.768 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.171 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.558 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.784 |
| F1-Score | Harmonic average of precision and recall | 0.261 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.524 |
| Informedness | Measures how informed the model is about positives and negatives | 0.128 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.342 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.209 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.514 |

- User-friendly UI;
- Low-code / No-code model training, making it more accessible to non-programming users;
- Need for machine learning knowledge to select model algorithms and model types, as well as performance metrics comprehension;
- Help Text and descriptions for every Data Science concept;
- Variety of model quality evaluators as well as written interpretations for every value.

For the **Analyse for Bias** feature, these are the initial impressions:

- Need to manually configure the dataset columns to be able to flag variables one by one;
- "Black box" when it comes to understanding what factors go into the selection of the most impactful (and potentially bias-oriented) variables when first starting to use the tool;
- In this experiment, the solution to mitigate bias in the predictions was to either ignore the Disparate Impact alerts or to exclude the variable entirely from the Dataset in Einstein, which may not be possible to achieve in real-world, commercial-consulting scenarios.

From the preliminary experiment, and by accepting the improvement suggestions, we saw an overall increase in the performance metrics values and an improvement in Einstein's model quality assessment. Furthermore, as stated before, the same variable was targeted for two suggestions, one in each model version. This means that, even with the first improvement suggestion applied, Einstein still picked up on a new way to bucket values to improve the model quality further.

Considering everything, and in light of the already reviewed facts, for the first experiment, we can conclude that Einstein is user-friendly, even for users with less knowledge in the technology

Table 3.4: Income Prediction - Confusion Matrix for Model Version #2.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 2788 True Positives (TP) | 1858 False Positives (FP) | **60.01% Positive Predicted Value** |
| | **<=50K Negative** | 5053 False Negatives (FN) | 22861 True Negatives (TN) | **81.9% Negative Predictive Value** |
| | | **35.56% True Positive Rate** | **92.48% True Negative Rate** | |

field, making it a very accessible tool for every type of company that wishes to use it. As for the **Analyse for Bias** feature, for now, we saw that it requires a manual configuration of the model, and the user needs to flag each variable individually. Einstein will show some preliminary alerts that could affect the model, but it is up to the user to check each variable. Finally, excluding a variable was the solution given in the bias analysis, which in some scenarios could not be a viable option, making it an obstacle for companies with ethical business practices in mind.

Table 3.5: Key Performance Metrics Results for Model Version #2.

| Metric | Einstein Description | Result |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.788 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.356 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.6 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.819 |
| F1-Score | Harmonic average of precision and recall | 0.447 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.532 |
| Informedness | Measures how informed the model is about positives and negatives | 0.28 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.419 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.343 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.512 |

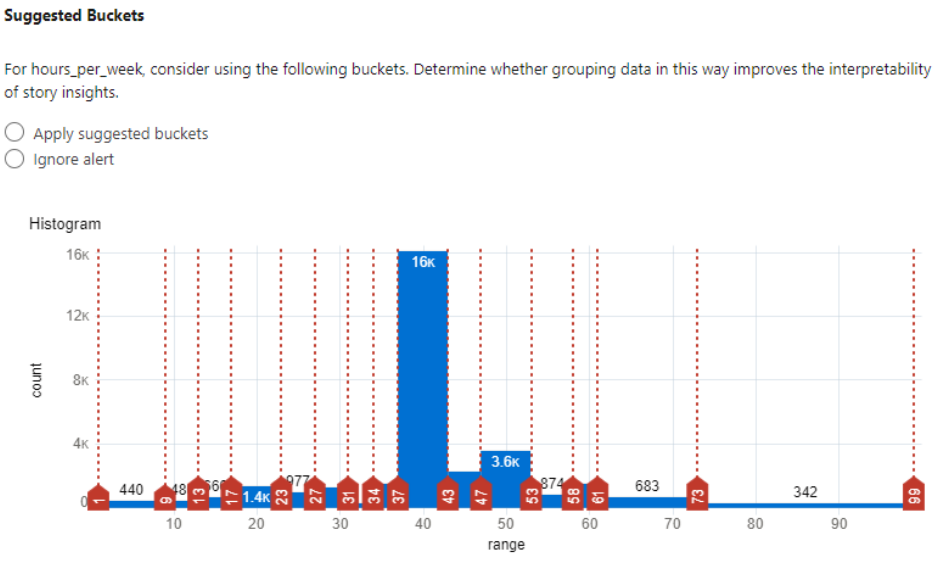


Figure 3.9: *Suggested Buckets* model improvement.

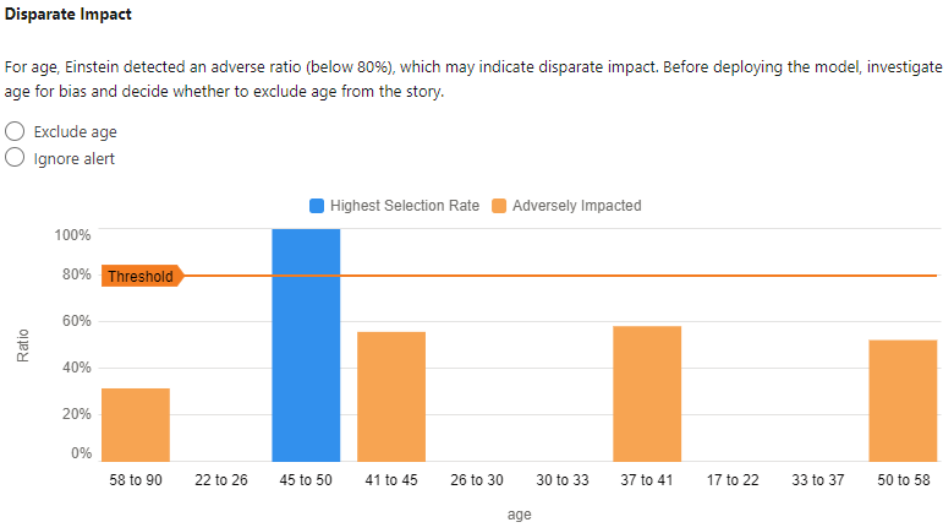


Figure 3.10: Disparate impact suggestion for *age*.

# Chapter 4

# Setting Analyse for Bias on Every Variable

In Chapter 3, we have developed a simple model with low customisation, meaning that, for all selections, we left the default values when training. In this chapter, we will create a new model where all variables in the dataset will be flagged for bias. This will allow us to understand which ones besides *age* are getting picked up for disparate impact. Moreover, we will test the same configurations and customisation in different algorithms the tool provides to select which results are the best-performing model. Additionally, we will choose the "Model Tournament" option to understand if our analysis for the best model is the same as Einstein's. For comparison, we will rely on Accuracy and F1-Score values. Firstly, we analyse **Accuracy** since it represents the proportion of correctly predicted outcomes. As Accuracy alone is not always the best metric, we will then analyse the **F1-Score**, since it is the harmonic average of **Precision** and **Recall**, which results in a more balanced and robust evaluator for model performance. Additionally, the F1-Score is more reliable regarding an uneven class distribution, which is the case for our dataset. We have an overall view of model performance quality with these two values. Moreover, we will also look at the "K-fold Cross Validation Results", with k=4, to further analyse and understand model performance.

Since this chapter is centred on the "Analyse for Bias" feature, we will also focus our efforts on better understanding and discovering Einstein's metrics for identifying potentially unfair or biased behaviour in the model training process. From our preliminary experiment in the previous chapter, we identified that Einstein uses selection rate thresholds to identify disparate impacts on a variable. This is commonly known in the literature as the 80% or "Four Fifths Rule".

## 4.1 Experiment Overview

As shown in more detail in Table 3.1, there are 14 variables in the Census Income dataset, excluding the Income variable, which is the one whose values we are trying to predict accurately. In this new experiment, we will develop a new model in which:

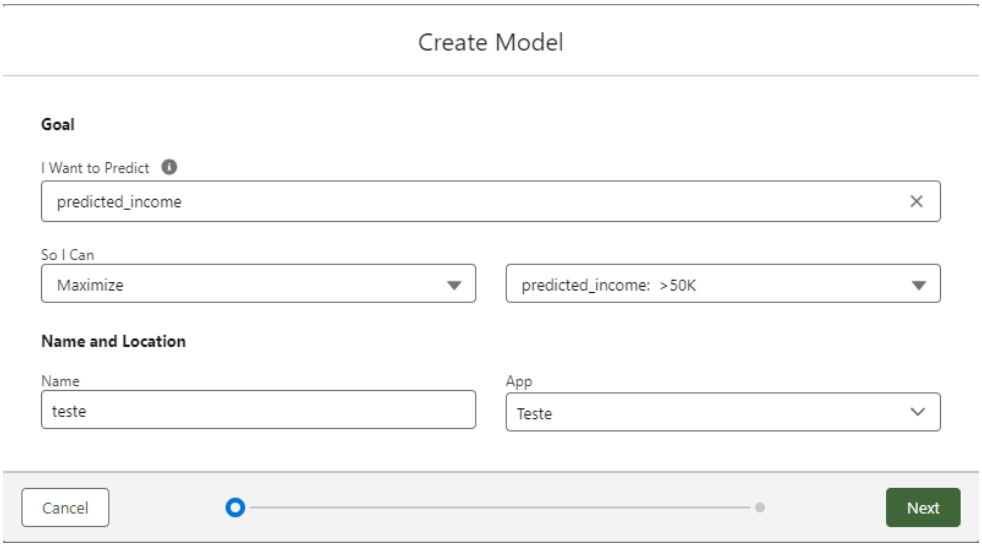


Figure 4.1: Create a new model with all the Dataset variables.

- All variables will be considered for model training;
- All variables will be flagged for bias analysis since every variable refers to demographic attributes.

In the first experiment, we will leave all possible selections as-is. This means that the only difference between this new model and the one in Chapter 3 will be the number of variables considered for model training and those flagged in the "Analyse for Bias" feature. After getting new results and comparing them with the preliminary experiment, we will start improving the model according to Einstein's suggestions and make comparisons again. Finally, with all variables considered, we will begin customising the model by changing the selected algorithm.

## 4.2 First Experiment: Leaving Selections As-Is

With access to an Enterprise version of Einstein and Analytics Studio, we can use up to 50 variables in one dataset. Since our dataset has 15 variables, we will not surpass this limit. The seemingly best practice this time was to create a new model entirely (see Figure 4.1). Additionally, the model columns had to be manually configured (shown in Figure 4.2) in order to get to the Analyse for Bias checkbox. All variables got flagged with the shield icon to maximise the opportunity of receiving disparate alert impacts beyond ***age*** (see Figure 4.3).

### 4.2.1 First Experiment: Leaving Selections As-Is - First Results

Starting with the Data Alerts, we got four alerts, all for disparate impact, doubling our preliminary experiment's results. As expected, ***age*** got the same alert, with the same improvement suggestions: exclude the variable or ignore the warning. The ***hours_per_week*** variable also got an alert, but this time related to bias analysis instead of a *Suggested Buckets* improvement, as shown in Figure 4.4.

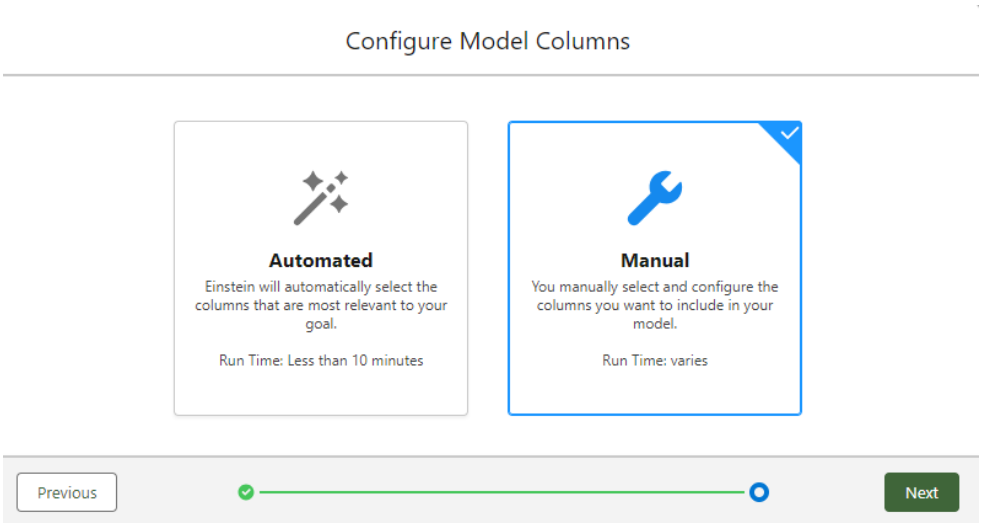


Figure 4.2: Manually configuring model columns.

Two new alerts appeared for Disparate Impact, one for the ***final weight*** variable, and the other for the ***education-num*** (education-numeric) variable, as shown in Figures 4.5 and 4.6.

The Top Predictors for this model were the same variables flagged in the Disparate Impact insights:

- **Age** with 59.4% importance;
- **Education-Numeric** with 26.67% importance;
- **Final Weight** with 8.54% importance;
- **Hours_per_Week** with 5.39%.

Unfortunately, we only got to see either "**Ignore Alert**" or "**Exclude Variable**" suggestions regarding solutions provided to mitigate Disparate Impact. However, it is too early in the analysis to conclude that this is the most common solution provided by Einstein to reduce bias in models. Referring to Table 4.1, we can conclude that, overall, the values for performance metrics increased, mainly the F1-Score, which went from 0.261 to 0.492. Additionally, we see a more balanced confusion matrix and better Precision and Recall results (0.64 and 0.399, respectively), as shown in Table 4.2. As for the 4-fold cross-validation, shown in Table 4.3, we can see that values for AUC were approximately 82% for all folds, which confirms that it performs better than random guessing, meaning that it correctly classified observation about 80% of the time. Furthermore, and by analysing the Mean Per Class Error, we can conclude that the predictions were not correct about 27% of the time. We can safely assume that our model cannot be classified as "perfect" for log loss values, with values rounding the 42% mark for each fold.

Since the variables displayed in Disparate Impact alerts were also the top predictors and relevant to income predictions, we could not exclude them from the data, which meant a deeper analysis of these four variables. After training the model, we can access the "Predictions" tab to

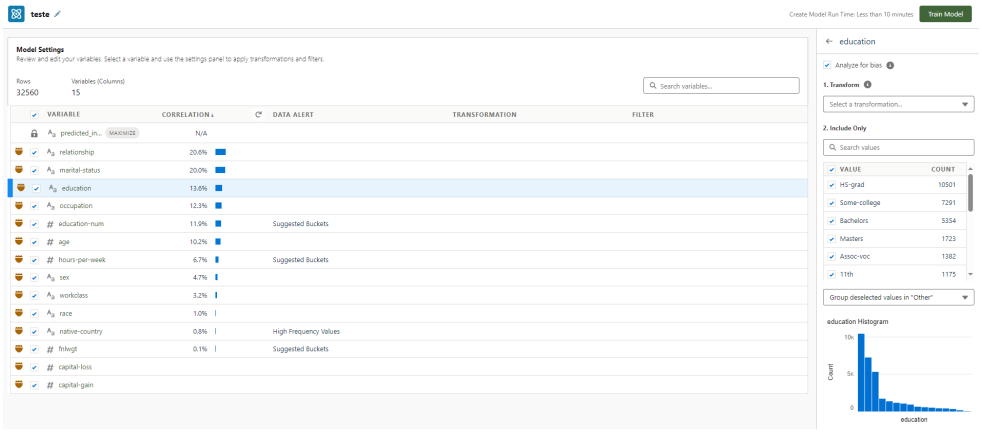


Figure 4.3: Manually configuring model settings.

select a specific group within each variable and see the predicted outcome based on the selected characteristics. In Figure 4.7, we can see that, for an age range of 45 to 50 years old, 13 to 16 years of education, final weight of 131 100 to 158 400 and 48 to 55 work hours in a week, Einstein predicts that individuals who meet this criteria are highly likely to have an income greater than 50,000 dollars with an Accuracy of 80.1%. Furthermore, this was also an expected result since every group selected within these variables was highlighted in the "Data insights" tab as most impactful to obtain a predicted income greater than 50,000 dollars, as shown in Figure 4.8. Since age was the variable with the highest importance for predictions, we decided only to change this range, leaving the rest of the values as-is. This time, we tested the prediction with an age range of 17 to 22 years old (stated as most impactful for predicting an income lower than or equal to 50,000 dollars), 13 to 16 years of education, and a final weight of 131 100 to 158 400, and 48 to 55 work hours in a week. Looking at Figure 4.9, and as expected, we can conclude that the prediction changed to "lower than or equal to 50,000 dollars", seeing that the threshold went from 0.846 to 0.294. Additionally, suppose we wanted to see how to change the prediction and threshold to better results. In that case, Einstein allows the selection of the "Actionable" button for each variable at a time. This will show improvements - if there are any - in the "Top Improvements" section, as shown in Figures 4.7 and 4.9, which should be aligned with other insights provided.

To summarise what has been stated so far, flagging every variable for bias gave us four Disparate impact alerts, which coincided with the Top Predictor variables. The solution to mitigate bias was to exclude these variables from training, which, in this case, since they were top predictors, was not an option.

### 4.2.2 First Experiment: Leaving Selections As-Is - New Model Versions

As aforementioned, excluding variables highlighted for Disparate Impact was not an option since the same variables were top predictors in the model. According to Salesforce Help, a Disparate Impact Alert "*Indicates a significant discrepancy in how different classes are being treated, which can signal bias.*" (Help.Salesforce, 2023). The approach here would be to try buckets to improve

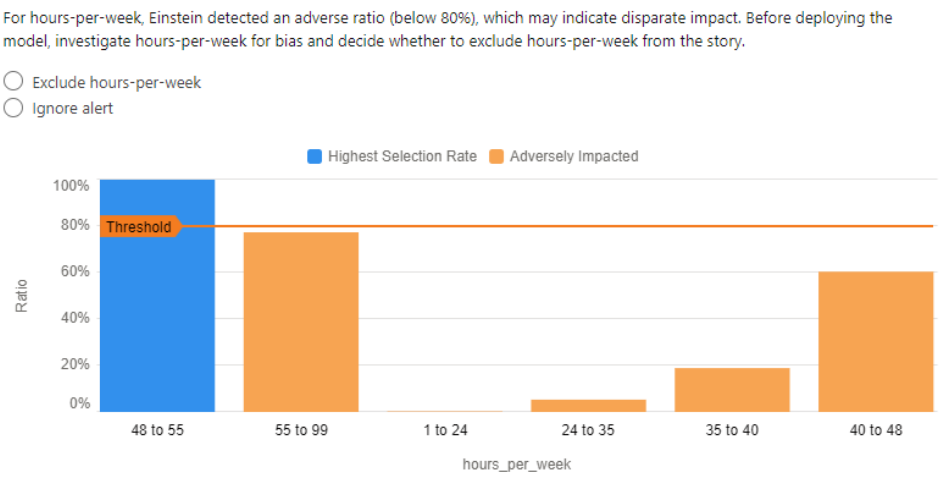


Figure 4.4: Disparate Impact alert for *hours_per_week*.

selection rates in different groupings for each variable highlighted for disparate impact. We started by reducing the number of buckets by half for the **education-num** variable, leaving us with the following value groupings:

- 1 to 4;
- 4 to 7;
- 7 to 10;
- 10 to 13;
- 13 to 16.

Additionally, we added "?" values to the "Other" category in the following variables:

- native_country;
- occupation;
- workclass.

After training the model, although we still got Disparate Impact alerts for the same four variables, the selection rates seem to be more balanced, except for **education_num**, which had the bucket of 13 to 16 years with a greater selection rate than the rest of the groups. This leads us to believe that bucketing this variable into five groups positively impacted other variables' results but did not demonstrate a good impact on itself. Furthermore, this also leads us to conclude that these variables may not lead the model to biased results, meaning they could need configuration adjustments. Moreover, we could not arrive at these conclusions or even experiment further by excluding these variables from the model training. Nevertheless, changing the buckets for the

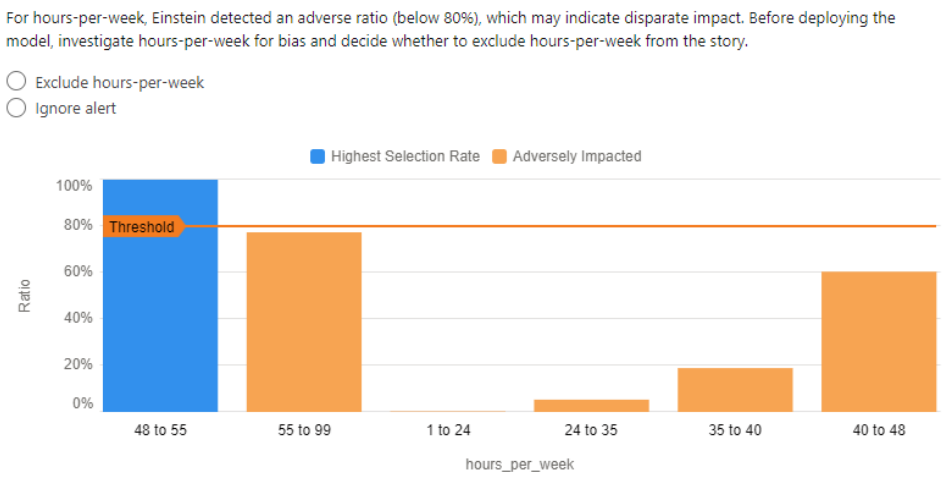


Figure 4.5: Disparate Impact alert for *education-num*.

"education-num" variable was insufficient since the selection rate for this variable did not move from the previous grouping (13 to 16 years).

For a new model version, the main focus lies in balancing the selection rate for education values. This time, instead of bucketing values by count (occurrence), the buckets were configured by width - bucketing *"proportionally within the total range of values"* - as shown in Figure 4.10. The same five buckets were maintained. After training the model again, we saw no changes to disparate impact alerts, with the group of 13 to 16 years being the only one with a selection rate above 80%. The selection rate for the group of 10 to 13 years improved, with a total of 33%. Moreover, we saw a drop in Accuracy, from 0.801 in the first version to 0.799 in this version. To finalise the bucketing analysis and to fully explore all options, we only had the "manual" one left, in which the user defines the intervals for each bucket. The logic behind the manual buckets (shown in Figure 4.11) was the American school years system - since we are using an American dataset -, meaning that the buckets were configured in the following way:

- $[1;5]$: Kindergarten through 5th grade;
- $[6;8]$: Middle School;
- $[9;12]$: High School;
- $[13;15]$ and $[16;+\infty[$: University years.

Once more, there were no positive results to come out of this, seeing that the only groups with significant selection rates were 13 to 16 years with 100% and 9 to 13 years with 16.24% (16.76% decrease compared to the previous model version and the 10 to 13 years group). Furthermore, Accuracy suffered another decrease, from 0.799 to 0.795.

Having already done some analysis of this particular model, it is safe to conclude that:

1. Flagging all variables for bias resulted in four "Disparate Impact" alerts, two more compared to the preliminary experiment detailed in Chapter 3;

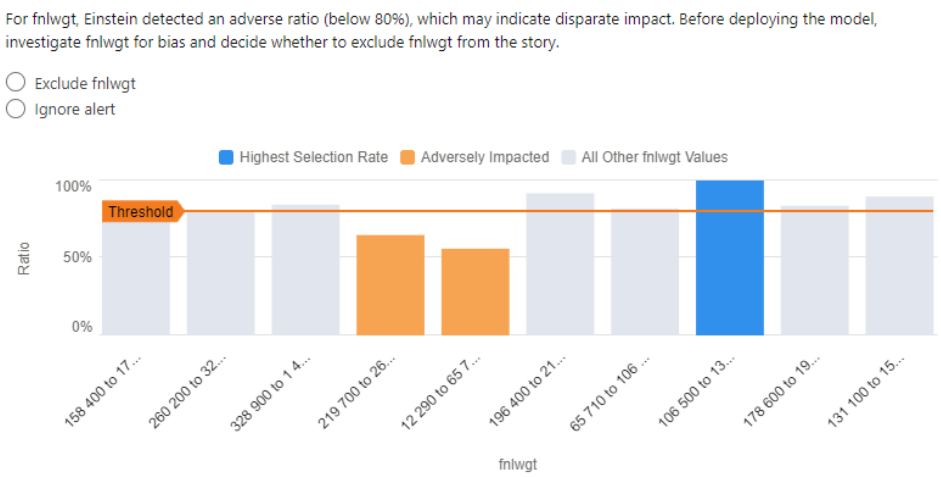


Figure 4.6: Disparate Impact alert for *final weight*.

2. The basis for "Disparate Impact alerts" is set in selection rates. Specifically, when the selection rate is less than 80% for a group within a variable, Einstein raises an alert;

3. By changing some configurations, we were able to attenuate three "Disparate Impact" alerts (*age*, *hours_per_week*, *final weight*), meaning that we were able to achieve more balanced results in selection rates, instead of having one to two highly different than the rest;

4. The variable "*education-num*" was the only one that did not display improvements with different model versions and bucket configurations;

5. Bearing in mind that the selected goal for this model was to maximise predictions for an Income greater than or equal to 50,000 dollars, Einstein did display some bias towards different education levels, seemingly benefiting the 13 to 16 years group above all others;

6. For these model versions, the only suggestion in "Disparate Impact" alerts - besides ignoring the alert - was to exclude the variables from the model. As we have seen, in some cases, the model is not biased towards a variable. It is not configured to its optimal point.

## 4.3 Second Experiment: Changing the Algorithm - GBM

In this section, the major change to the model will rely on the algorithm leveraged for training, which, this time, will be **GBM**. Compared to GLMs, this algorithm should be more accurate for non-linear scenarios since they "account for non-linearity between a predictor and dependent variable and interactions between predictor variables without manual tweaking" (Proksch, 2020). Moreover, they should also "provide accurate results for each variable's incremental impact" (Proksch, 2020).

To better compare the first model versions, we will revert the changes made to the *education-num* variable to 10 buckets, automatically grouped by "Count" by Einstein.

Table 4.1: GLM: Key Performance Metrics Results.

| Metric | Einstein Description | Result |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.801 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.399 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.64 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.83 |
| F1-Score | Harmonic average of precision and recall | 0.492 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.647 |
| Informedness | Measures how informed the model is about positives and negatives | 0.328 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.47 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.393 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.483 |

### 4.3.1 Second Experiment: Changing the Algorithm - GBM - First Results

At first glance, training with GBM did not provide significant improvements in model performance since the increase in value as of:

- 0.1% for Accuracy;
- 0.5% for Recall;
- 0.1% for Precision (equal to Accuracy);
- 0.5% for the F1-Score (equal to Recall).

The threshold slightly decreased, precisely 0.4%, from 0.483 to 0.479. By analysing Tables 4.4 and 4.5, we can conclude that no metric had an impactful increase or decrease in value, leading us to believe that changing the model algorithm from GLM to GBM did not produce better (or worse) results, thus maintaining the same analysis as before. When compared to the preliminary experiment, the value increase is higher. Still, since we could not include all variables in the first model, it is also not a reliable and equitable comparison. As we have seen for performance metrics, cross-validation results were no different compared to the GLM model. According to the results in Table 4.6, there was a slight increase in AUC and a consequent decrease in Mean Per Class Error, but it was not significant enough to identify as an improved model. Log Loss values remained in the 42% mark.

In terms of "Disparate Impact" alerts, the same four variables as the previous section (*age*, *hours_per_week*, *education_num* and *final weight*) were identified, with ***education_num*** being the most unbalanced of the four. Moreover, these alerts again referred to "Top Predictors" within the model, preventing us from excluding the variables from the training data. In terms of order and hierarchy, *age* was once again the most impactful with 50.3% importance, followed by:

Table 4.2: GLM: Income Prediction - Confusion Matrix for First Experiment.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 3130 True Positives (TP) | 1759 False Positives (FP) | **64.02% Positive Predicted Value** |
| | **<=50K Negative** | 4711 False Negatives (FN) | 22960 True Negatives (TN) | **82.97% Negative Predictive Value** |
| | | **39.92% True Positive Rate** | **92.88% True Negative Rate** | |

- **Education-Numeric** with 30.5% importance;
- **Final Weight** with 14.5% importance;
- **Hours_per_Week** with 4.7%.

Additionally, this time, *hours_per_week* was above *final weight*, switching places from 4th to 3rd compared to the previous model's first version.

### 4.3.2 Second Experiment: Changing the Algorithm - GBM - New Model Versions

As for GLM, we replicated the same steps for improvements, starting with bucketing the *education-num* values by count (occurrence) in 5 groups.

We saw a worsened model with an **Accuracy of 0.801** and an **F1-score of 0.473** - meaning a decrease of 0.2% for Accuracy and 2.4% for the F1-Score. The **threshold** value increased to **0.497**. There were no significant changes in "Disparate Impact" alerts. Nevertheless, for the "Top Predictors" section, we saw *education_num* trade importance with *age* for first place. On this version, *education_num* had a 45.71% importance. Overall, there were no significant improvements to the model or new Einstein improvement suggestions. Furthermore, when analysing "Data Impact" alerts, we concluded that three out of four variables, although identified for disparate impact, the selection rates were balanced whereas *education_num* had a clear favourite in the selection, the 13 to 16 bucket, aligned with what we have seen before. For this version, we can conclude that this dataset's highest level of education was the most significant prediction influencer for an income greater than 50,000 dollars. As a first impression assumption, we also take away the fact that Einstein may be inclined towards the highest education values available in the dataset for these prediction tasks.

Maintaining the same process as before, we created a new model version with the only change being the way of bucketing values for education - this time with **width** instead of count (occurrence), ceteris paribus. As expected, no changes or performance improvements were detected, as Accuracy and the F1-Score suffered a 0.1% and 0.3% decrease, respectively.

Table 4.3: GLM: 4-Fold Cross Validation Results

| Metric Name | Einstein Description | Training Set | Validation Set | Fold 1 | Fold 2 | Fold 3 | Fold 4 |
|---|---|---|---|---|---|---|---|
| **Number of Rows** | This is the number of rows used for each fold and set. | 32560 | 32560 | 8148 | 8062 | 8141 | 8209 |
| **AUC** | The Area Under the Curve (AUC) represents the rate of correct classification by a logistic model. An AUC of 0.5 means that the model performs no better than random guessing. An AUC of 1.0 means that the model correctly classifies data 100% of the time, which can indicate data leakage. | 0.823 | 0.823 | 0.817 | 0.815 | 0.815 | 0.817 |
| **GINI** | The Gini Index quantifies how closely this logistic model performs to a theoretically best possible model. | 0.647 | 0.647 | 0.634 | 0.63 | 0.629 | 0.634 |
| **Log Loss** | Logarithmic Loss measures model performance on a scale of 0 to 1, where 0 represents a perfect model (predictions correctly match observations 100%). The less frequently that the predicted probability correctly matches actual observations (lower performance), the higher the log loss. | 0.418 | 0.418 | 0.429 | 0.429 | 0.423 | 0.421 |
| **Mean Per Class Error** | The Mean Per Class Error measures how often the predictions are wrong. The lower the value, the fewer the wrong predictions, and therefore the better the model. | 0.262 | 0.262 | 0.266 | 0.274 | 0.268 | 0.265 |

Finally, to conclude the GBM algorithm section and analysis, all that was left to evaluate was manually bucketing *education*, following the same 'American School System' logic. Unfortunately, the selection rate for education did not get any more balanced than before, and there was a decrease in Accuracy of 0.4% and 3% for the F1-Score.

To conclude this section, it is safe to affirm that GBM did not improve our binary classifier:

1. For the first two model versions, we saw similar - and some identical - performance evaluation values for metrics;

2. Only on the third version did we see any difference in metrics, with a greater decrease in value than for the third model version in GLM.

## 4.4 Third Experiment: Changing the Algorithm - XG Boost

After GLM and GBM, we arrived at the third training algorithm option: Extreme Gradient Boosting. Most commonly known as **XG Boost**, this algorithm is well-liked amongst developers, as it is said and proven to be faster than others without compromising model performance quality (xgboost developers, 2022). According to Nvidia documentation, XG Boost "*provides a parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems*" (Nvidia, 2024). In this section, we will put these statements to the test in a Salesforce context. To obtain a better and more exhaustive comparison, we reverted all new configurations to the *education_num* variable.

### 4.4.1 Third Experiment: Changing the Algorithm - XG Boost - First Results

Pivoting from the previous two algorithms and associated model versions, the change to XG Boost had an immediately noticeable impact since, with all variables flagged for "Analyse for Bias", there was no new "Disparate Impact" alert. Table 4.7 reflects the new model metrics, and as expected for XG Boost, there were improvements, some greater than others. **Accuracy** improved **0.2%** and the **F1-Score** improved **0.7%**, mirroring the improvements of the values for Precision and Recall, which can also be analysed and concluded by looking at Table 4.8. As for cross-validation

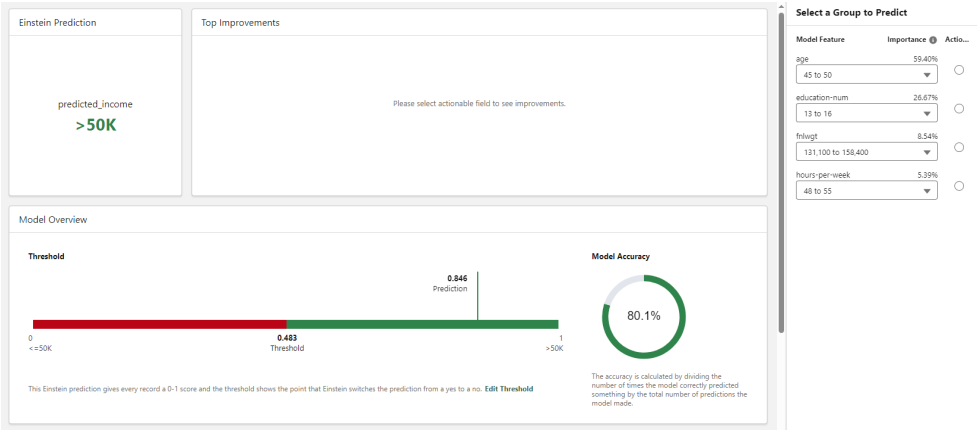


Figure 4.7: Positive prediction with top predictors.

results, AUC did not suffer meaningful changes to its results, and neither did the GINI. Out of the four metrics in Table 4.9, **Log Loss** was the most positively impacted metric for both the training and validation sets. Moreover, we also see a slight decrease in the Mean Per Class Error values, which means that the frequency of incorrect predictions decreased slightly. Additionally, the previous alerts identified in other model versions were marked as "**Resolved**" - as shown in Figure 4.12, meaning that in this version, Einstein did not detect adverse ratios for selection rates within variable groups. Furthermore, there were no improvement suggestions. Contrary to the other model versions, and as stated before, there were no alerts, so, in theory, we could skip the *education_num* bucket configurations. Nevertheless, it would not be a fair comparison towards the other algorithms. These new configurations will serve the purpose of accurate comparisons with an exact match between configurations. Still, they will not be used to mitigate or balance any "Disparate Impact" or biased selection within variables.

### 4.4.2 Third Experiment: Changing the Algorithm - XG Boost - New Model Versions

As per previous sections, the experiment pipeline will be the following:

1. Reducing the number of buckets for *education_num* by half;
2. Training the model with the education variable bucketed by count (occurrence);
3. Training the model with the education variable bucketed by width;
4. Training the model with education variable manually bucketed with the American School System logic;
5. Compare results and analyse improvements, should any occur.

Starting with the count (occurrence) option mentioned in point 2, we obtained similar results as the previous algorithms as we saw the Accuracy value decrease by 0.2%. Contrary to GLM and

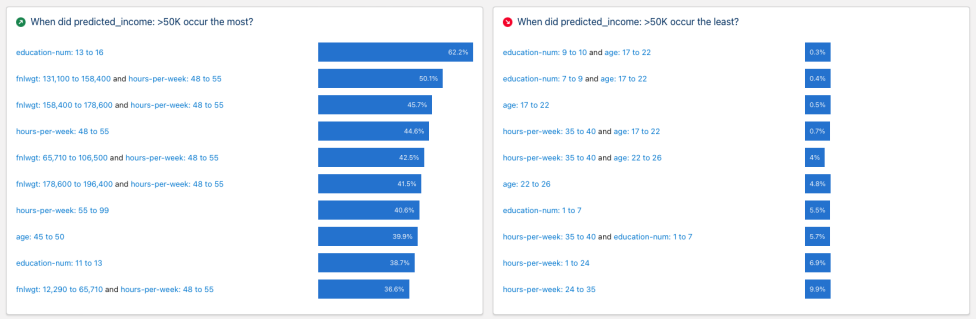


Figure 4.8: Positive and negative impact from variables' groups.

GBM, there was an increase of 0.6%. Additionally, related to the F1-Score, we saw the Recall value increase by 1.5%. For "Top Predictors" there was no surprise when *age*, *education_num*, *hours_per_week* and *final weight* were identified, with ***age*** at the top of the list.

Moving on to the 3rd point, bucketing the values by width (proportion), we again found a worsened model performance. The Accuracy of 0.801 makes this version comparable to the first GLM version. Nevertheless, the F1-Score significantly decreased this time, with a value below the one for the GLM version.

For the 4th and final point and experiment for XGBoost - the American School System logic bucketing - we obtained one of the worst Accuracy values, 0.798. Moreover, the F1-Score and related metrics Precision and Recall also significantly decreased. As expected and reflected in other algorithm's versions, the third and final experiment with bucketing values for *education_num* was the worst performing one.

To summarise the information gathered in these sections regarding the XG Boost algorithm, we found that:

- Out of GLM, GBM and XG Boost, the latter had the best performance so far, with an Accuracy of 80.4% when education_num was with Einstein's default groupings (buckets);
- Out of the three tested algorithms, XG Boost was the one that displayed a bigger difference in model performance metric values when comparing the first version - Einstein default bucketing - with the last version - decreasing bucket number by half and grouping values following the American School System school years grouping;
- The "Disparate Impact" alerts identified in GBM's training versions were "Resolved" when switching to XG Boost, meaning the selection rates within variables' groups were more balanced.

## 4.5 Fourth Experiment: Changing the Algorithm - Random Forest

Once more, reverting all bucket configurations to the default Einstein state, we changed the algorithm to understand which types - if any - of improvements would appear when flagging every

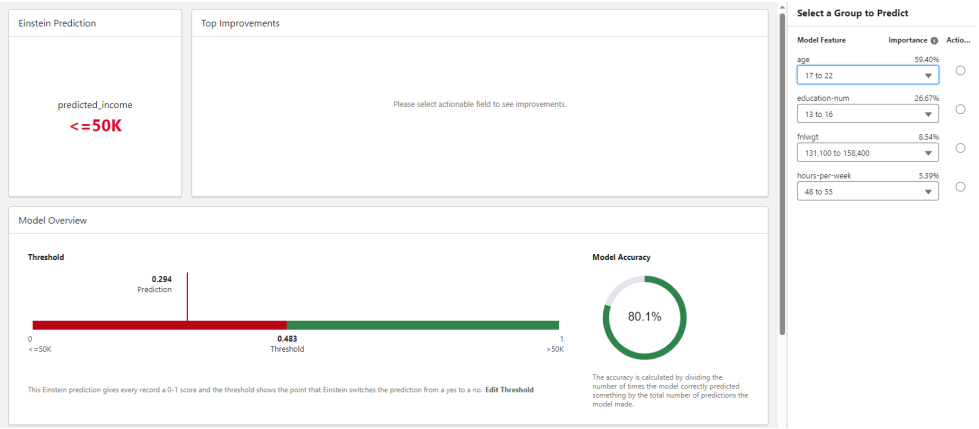


Figure 4.9: Negative prediction with top predictors.

variable for Bias in the dataset. We also performed the same bucket configurations to the *education_num* variables for fair comparisons between versions and algorithms. The algorithm studied in this section was Random Forest. As an ensemble algorithm, it "combines the output of multiple decision trees" to arrive at the predicted results IBM (2024). Moreover, according to IBM's publications, decision trees can be prone to overfitting and biased results, and Random Forest should help to mitigate these issues.

### 4.5.1 Fourth Experiment: Changing the Algorithm - Random Forest - First Results

After analysing Tables 4.10 and 4.11, we concluded that this algorithm, until now, had been the worst performing one, with a starting Accuracy of 0.795 and an F1-Score of 0.461. Of all the metrics, Recall is an excellent example of the difference in value, having decreased from 0.413 to 0.363. These findings were also reflected in the cross-validation (Table 4.12), which demonstrated the worst values for Log Loss and AUC, which increased and decreased, respectively. Consequently, the Mean Per Class Error also increased in value. As per "Disparate Impact", the previously resolved issues resurfaced, with the same variables that we have seen throughout this chapter: *age*, *education_num*, *hours_per_week* and *final weight*. Once more, these variables were also "Top Predictors", preventing us from excluding any of them from the dataset. Furthermore, and as shown in Figure 4.13, it is safe to affirm that this was one of the worst selection rate distributions in education groupings for *education_num*, having "1" through "11" with a 0% selection rate, meaning that the model only showed interest in picking up values in two buckets (11 to 13 and 13 to 16). Additionally, this could have been the opposite had we chosen to minimise predictions for an Income greater than 50,000 dollars or maximise an Income less than or equal to 50 000 dollars. In the following section, we will address this limitation in Einstein in more detail, confirming or discrediting the assumption made in the previous affirmation.

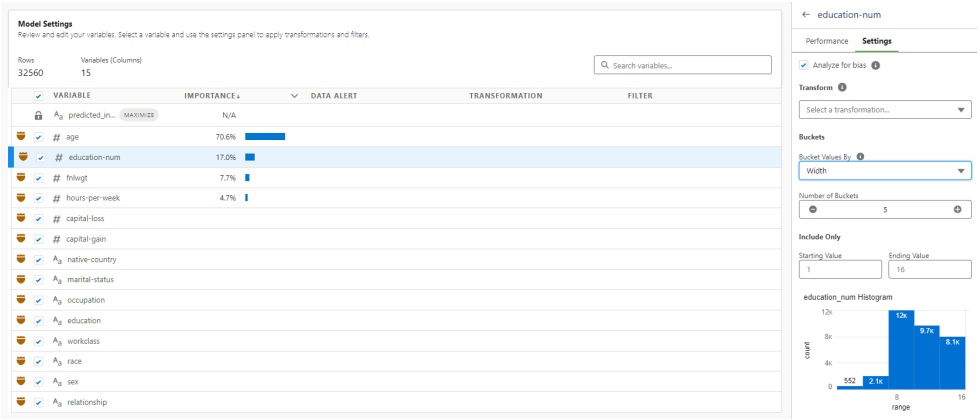


Figure 4.10: Bucketing values by width.

### 4.5.2 Fourth Experiment: Changing the Algorithm - Random Forest - New Versions

Following the previous sections' experiments, we will start by decreasing the buckets for *education_num* to 5 and changing how the values in the buckets are configured. As stated before, we will start with count (occurrence), followed by width (proportion) and end with manually bucketing the values by mirroring the American School System logic. Our findings - combining the first results, the new versions, and the algorithm overall - can be summarised by the following key points and takeaways:

1. When bucketing by count (occurrence), the selection rate for education_num was worse, and the selection rate for other variables did not seem to change. Accuracy decreased by 0.1%, and the F1-Score increased by 2.8%. No new model improvements were detected for other variables;

2. Bucketing values by width (proportion) had once more a negative impact on model quality and performance metrics. Accuracy decreased by 0.3% and the F1-Score by 4.4%, which was the greater decrease in value so far for this metric between all versions and algorithms tested so far. As for "Disparate Impact" alerts, the only improvement occurred for *final weight*, with all selection rates close to or above the 80% threshold, which means that this alert was not far from being resolved and removed from the "Analyse for Bias" alerts;

3. Manually configured buckets, as expected, worsened the model performance to the lowest Accuracy so far: 0.79 (0.1% decrease from the previous version). Additionally, the F1-Score decreased by 5%, which surpassed the previous version's decrease by 0.6%;

4. Overall, Random Forest was the worst-performing algorithm and the most imbalanced regarding selection rates within "Top Predictors"' value groupings.

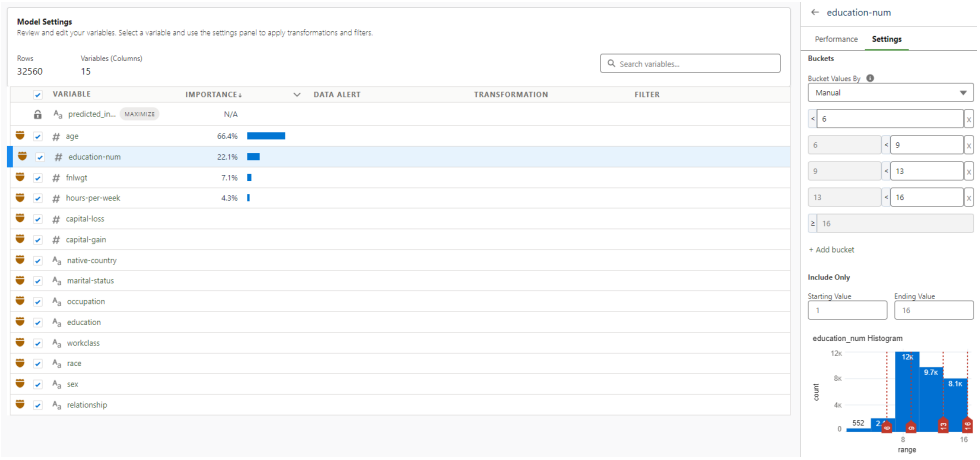


Figure 4.11: Manual configuration of buckets for education.

## 4.6 Fifth Experiment: Model Tournament

In the previous section, we concluded the experiments for the different algorithms provided by Einstein. For this fifth and final section, we selected the "**Model Tournament**" option, which, according to the help text, will have Einstein running all algorithms and then choosing the one that produces the most accurate model. From our analysis, we safely concluded that XG Boost was the best-performing algorithm in both performance metric values and "Disparate Impact" alerts - given that they were all set to "Resolved" when the model was trained with this algorithm. Since this option is set to provide the best model possible, depending on the results, there was a possibility that altering the buckets for *education_num* was not needed.

### 4.6.1 Fifth Experiment: Model Tournament - Results

At the start of the training process, we found that this option took longer since Einstein ran all possible algorithms. Model Tournament works in Einstein in a *blackbox* context since the end-user has no visibility of what happens while the training goes on.

Einstein's winner was XG Boost, as was also concluded by our singular analysis for each algorithm. As per metric results, they were the same as the ones from the previous section, with no value difference. Before moving on to some of the most relevant insights, one more experiment was needed: flagging every variable for bias analysis, removing any missing or "?" values, and excluding *final weight* from the dataset columns set for training. This logic came from the fact that since this variable only represents the number of people, the census believes each entry represents (Lemon et al., 2024), meaning it is not a valid demographic variable. We wanted to understand if this variable affected the model quality positively or negatively. By excluding this variable, we found that, for the best-performing model (XG Boost), this variable had been positively impacting Accuracy, which was 80.4% with it and 79.9% without. As we advanced, the decision was to keep *final weight* on the training data.

Table 4.4: GBM: Key Performance Metrics Results.

| Metric | Einstein Description | Result |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.802 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.405 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.641 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.831 |
| F1-Score | Harmonic average of precision and recall | 0.497 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.647 |
| Informedness | Measures how informed the model is about positives and negatives | 0.333 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.472 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.396 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.479 |

Taking into consideration that the **average for predicted income greater than 50 000 dollars was 24.08%**, some of the most relevant insights showed us that:

- As shown in Figure 4.14, and corroborating what had been analysed before, when education was between 13 to 16 years (highest level of education on the dataset), 62.24% of prediction were for an income greater than 50,000 dollars;
- By analysing Figure 4.15, we concluded that for ages above 33 years old, the percentage of predictions for an income greater than 50,000 dollars was always above the average of 24.08%;
- Only when the number of work hours in a week was above 40 did the percentage of predictions for an income greater than 50,000 dollars lie above the average, as shown in Figure 4.16.

#### 4.6.1.1 Fifth Experiment: Model Tournament

Before moving to the conclusions section to close the chapter, we decided to perform the model tournament, with all variables flagged and all default buckets for variables where this was applicable. Still, this time, with the "minimise" option instead of the "maximise one" for the predicted income (greater than 50,000 dollars). To clarify, on this model, for the "**So I Can**" selection, we chose "minimise", leaving all other selections equal to the previous models analysed in this chapter, as shown in Figure 4.17. For "Top Predictors", the same four previous variables were identified. As expected, the best-performing model was XG Boost, with no Data Alerts. The most significant differences we could locate were for "Data Insights" since this model highlighted the exact opposite of the first model: predictions for an income greater than 50,000 dollars occur the least when the age is between 17 and 26 years old. The number of work hours was reduced (see

Table 4.5: GBM: Income Prediction - Confusion Matrix for Second Experiment.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 3178 True Positives (TP) | 1782 False Positives (FP) | **64.07% Positive Predicted Value** |
| | **<=50K Negative** | 4663 False Negatives (FN) | 22937 True Negatives (TN) | **83.11% Negative Predictive Value** |
| | | **40.53% True Positive Rate** | **92.79% True Negative Rate** | |

Figure 4.18). Essentially, what hurt the previous model was highlighted and positively impacted this one. Moving forward, we decided that all experiments would be based on the maximisation of predicted income (> 50,000 dollars) since this selection seemed to have little to no impact on predictions: for XG Boost Accuracy was still of 80.4%, with an F1-Score of 50.4%. For cross-validation, the results were also the same as the previous XG Boost model version (when configured to maximise *predicted_income > 50K*).

## 4.7 Conclusions

In this section, we performed experiences and tested the different algorithms provided by Einstein, all of which will contribute to answering the second research question: "How reliable is the Analyse for Bias feature in Salesforce Einstein, and to what degree does it effectively mitigate discriminatory outcomes in the models it assesses?". Until now, we have found that variables that were greatly expected to have an adverse impact have not raised any issue within model training in Einstein. Example of such variables would be ***race***, ***sex*** or ***native_country***. Nevertheless, we are working with an income-prediction dataset, which makes the fact that variables such as *education*, *age*, and *hours_per_week* have been the most impactful was not a surprise or even unexpected. In almost every tested algorithm, *education_num* was the variable with the least balanced bucket selection rates, leading us to perform additional configurations. Having tested buckets by count, width or manual configuration, we concluded that Einstein's auto-generated buckets always performed better. However, we can also affirm that balancing variable groups or buckets, although it slightly compromises the model's accuracy, gives those same groups a fairer chance when training the models.

According to our conclusions and analysis, and aligned with the results from the Model Tournament option in Einstein, XG Boost had the best performance, most excellent metric values, and model quality since it did not produce any "Disparate Impact" alerts, having resolved selection rate issues identified in previous versions' algorithms.

Following up on our second research question, we can identify three key takeaways:

Table 4.6: GBM: 4-Fold Cross Validation Results.

| Metric Name | Einstein Description | Training Set | Validation Set | Fold 1 | Fold 2 | Fold 3 | Fold 4 |
|---|---|---|---|---|---|---|---|
| **Number of Rows** | This is the number of rows used for each fold and set. | 32560 | 32560 | 8148 | 8062 | 8141 | 8209 |
| **AUC** | The Area Under the Curve (AUC) represents the rate of correct classification by a logistic model. An AUC of 0.5 means that the model performs no better than random guessing. An AUC of 1.0 means that the model correctly classifies data 100% of the time, which can indicate data leakage. | 0.83 | 0.83 | 0.816 | 0.816 | 0.815 | 0.817 |
| **GINI** | The Gini Index quantifies how closely this logistic model performs to a theoretically best possible model. | 0.66 | 0.66 | 0.633 | 0.631 | 0.629 | 0.634 |
| **Log Loss** | Logarithmic Loss measures model performance on a scale of 0 to 1, where 0 represents a perfect model (predictions correctly match observations 100%). The less frequently that the predicted probability correctly matches actual observations (lower performance), the higher the log loss. | 0.412 | 0.412 | 0.429 | 0.428 | 0.423 | 0.42 |
| **Mean Per Class Error** | The Mean Per Class Error measures how often the predictions are wrong. The lower the value, the fewer the wrong predictions, and therefore the better the model. | 0.261 | 0.261 | 0.261 | 0.261 | 0.267 | 0.267 |

1. It is important to remember that when we were first creating the model, we needed to define if we wanted to maximise or minimise the predicted income, which could have influenced every result and outcome moving forward. As it was concluded before, this impact was minimal;

2. It is safe to assume that the bias-related alerts and issues were most likely tied to the algorithm selected, not to inherent bias within Einstein's configurations and back-end developments. The "Analyse for Bias" feature seems to work well in detecting issues with variables' selection rates, warning the user of the need for further analysis. This is also known in the literature as the "**Four Fifths Rule**", in which the selection ratio of an underprivileged group should be at least 80% of a privileged or majority group (Ronaghan, 2019). This metric is commonly used and is a base for now-known toolkit metrics such as Fairlearn's from Microsoft;

3. When disparate alerts appear, the only possible solution we got until now was to exclude [relevant] variables from training.

As we understand, Einstein provides the "Analyse for Bias" feature as a diagnostic tool based on the 80% rule to identify potential disparate impact on training variables. Following this discovery, we arrive at a potential paradoxical conclusion for Einstein models: the selection rate threshold and consequential disparate impact alerts may not always be reliable or inherently related to fairness in a model. Looking at our use case, we can see that no alerts were raised for selection rates in most sensitive variables such as *sex*, *race* or *native_country*, which refer to an individual's gender and ethnicity attributes. For our models, alerts were raised for factual variables like the number of years that an individual has studied or the number of hours they work in a week. These values are highly related to an individual's income, but their values should directly impact the model's decision when predicting an income, whereas a person's gender or ethnicity should not. Seeing that there were no disparate impacts for such attributes, we can argue that the alerts raised are not fairness-related but mere observations that some groups were favoured over others.


Figure 4.12: Resolved alerts with XG Boost.

Table 4.7: XG Boost: Key Performance Metrics Results.

| Metric | Einstein Description | Result |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.804 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.413 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.645 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.833 |
| F1-Score | Harmonic average of precision and recall | 0.504 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.662 |
| Informedness | Measures how informed the model is about positives and negatives | 0.341 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.478 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.404 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.486 |

Table 4.8: XG Boost: Income Prediction - Confusion Matrix for Third Experiment.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 3242 True Positives (TP) | 1781 False Positives (FP) | **64.54% Positive Predicted Value** |
| | **<=50K Negative** | 4599 False Negatives (FN) | 22938 True Negatives (TN) | **83.3% Negative Predictive Value** |
| | | **41.35% True Positive Rate** | **92.8% True Negative Rate** | |

Table 4.9: XG Boost: 4-Fold Cross Validation Results.

| **Metric Name** | **Einstein Description** | **Training Set** | **Validation Set** | **Fold 1** | **Fold 2** | **Fold 3** | **Fold 4** |
|---|---|---|---|---|---|---|---|
| **Number of Rows** | This is the number of rows used for each fold and set. | 32560 | 32560 | 8148 | 8062 | 8141 | 8209 |
| **AUC** | The Area Under the Curve (AUC) represents the rate of correct classification by a logistic model. An AUC of 0.5 means that the model performs no better than random guessing. An AUC of 1.0 means that the model correctly classifies data 100% of the time, which can indicate data leakage. | 0.831 | 0.831 | 0.815 | 0.813 | 0.81 | 0.815 |
| **GINI** | The Gini Index quantifies how closely this logistic model performs to a theoretically best possible model. | 0.662 | 0.662 | 0.63 | 0.625 | 0.621 | 0.63 |
| **Log Loss** | Logarithmic Loss measures model performance on a scale of 0 to 1, where 0 represents a perfect model (predictions correctly match observations 100%). The less frequently that the predicted probability correctly matches actual observations (lower performance), the higher the log loss. | 0.41 | 0.41 | 0.43 | 0.431 | 0.427 | 0.422 |
| **Mean Per Class Error** | The Mean Per Class Error measures how often the predictions are wrong. The lower the value, the fewer the wrong predictions, and therefore the better the model. | 0.257 | 0.257 | 0.271 | 0.267 | 0.269 | 0.267 |

Table 4.10: Random Forest: Key Performance Metrics Results.

| **Metric** | **Einstein Description** | **Result** |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.795 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.363 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.629 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.822 |
| F1-Score | Harmonic average of precision and recall | 0.461 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.62 |
| Informedness | Measures how informed the model is about positives and negatives | 0.295 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.451 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.365 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.454 |

Table 4.11: Random Forest: Income Prediction - Confusion Matrix for Fourth Experiment.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 2848 True Positives (TP) | 1680 False Positives (FP) | **62.9% Positive Predicted Value** |
| | **<=50K Negative** | 4991 False Negatives (FN) | 23032 True Negatives (TN) | **82.19% Negative Predictive Value** |
| | | **36.33% True Positive Rate** | **93.2% True Negative Rate** | |

Table 4.12: Random Forest: 4-Fold Cross Validation Results.

| Metric Name | Einstein Description | Training Set | Validation Set | Fold 1 | Fold 2 | Fold 3 | Fold 4 |
|---|---|---|---|---|---|---|---|
| **Number of Rows** | This is the number of rows used for each fold and set. | 32560 | 32560 | 8148 | 8062 | 8141 | 8209 |
| **AUC** | The Area Under the Curve (AUC) represents the rate of correct classification by a logistic model. An AUC of 0.5 means that the model performs no better than random guessing. An AUC of 1.0 means that the model correctly classifies data 100% of the time, which can indicate data leakage. | 0.81 | 0.81 | 0.815 | 0.811 | 0.81 | 0.814 |
| **GINI** | The Gini Index quantifies how closely this logistic model performs to a theoretically best possible model. | 0.619 | 0.619 | 0.63 | 0.623 | 0.62 | 0.628 |
| **Log Loss** | Logarithmic Loss measures model performance on a scale of 0 to 1, where 0 represents a perfect model (predictions correctly match observations 100%). The less frequently that the predicted probability correctly matches actual observations (lower performance), the higher the log loss. | 0.434 | 0.434 | 0.433 | 0.435 | 0.429 | 0.426 |
| **Mean Per Class Error** | The Mean Per Class Error measures how often the predictions are wrong. The lower the value, the fewer the wrong predictions, and therefore the better the model. | 0.274 | 0.274 | 0.269 | 0.274 | 0.285 | 0.269 |

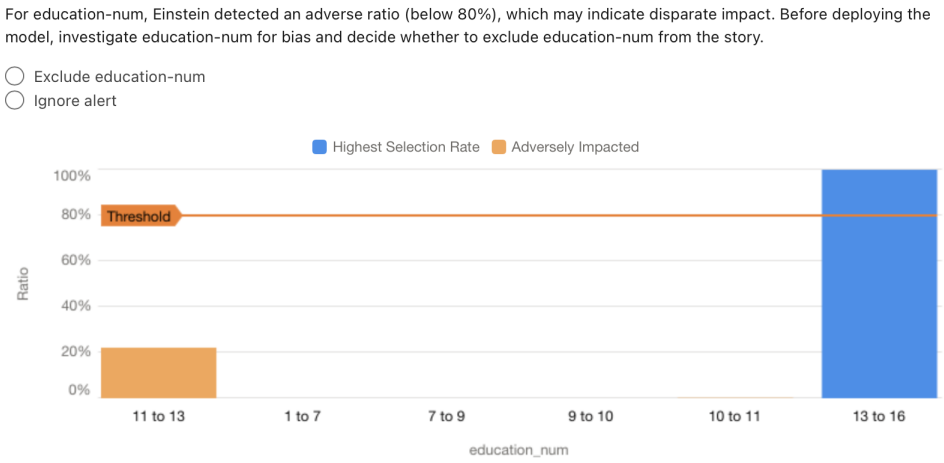


Figure 4.13: Random Forest - Disparate Impact alert for education_num.

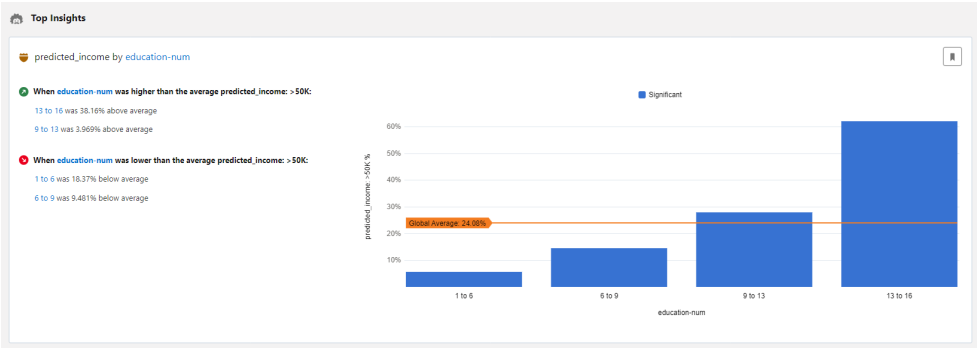


Figure 4.14: Top insights for *education_num*.

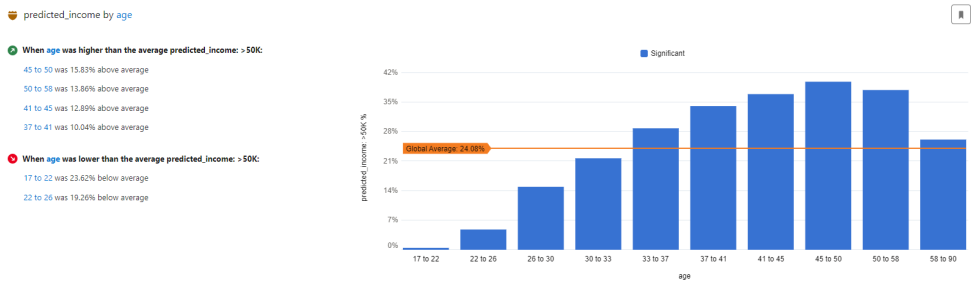


Figure 4.15: Top insights for *age*.

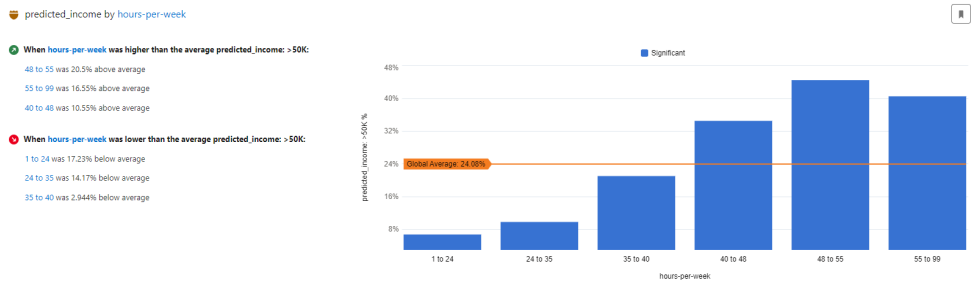


Figure 4.16: Top insights for *hours_per_week*.

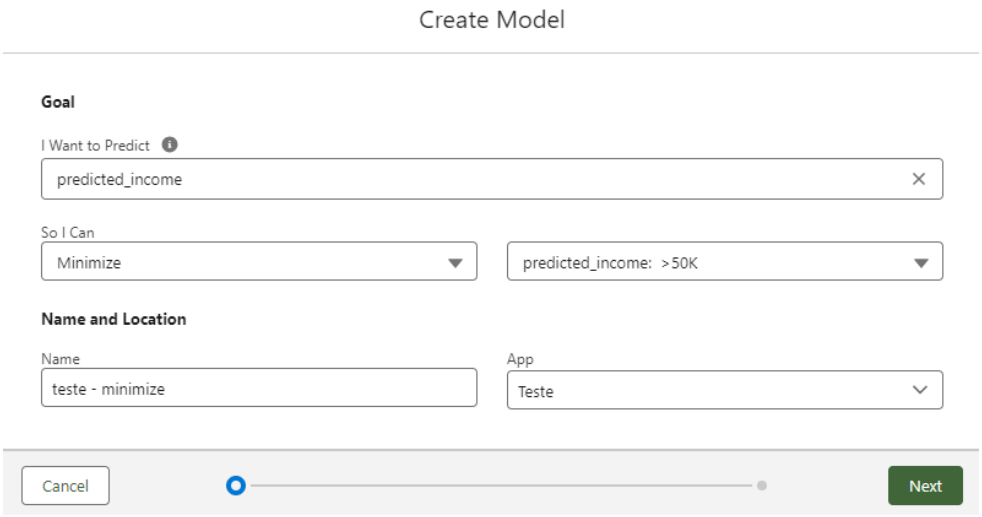


Figure 4.17: Minimising the predicted income greater than 50,000 dollars.

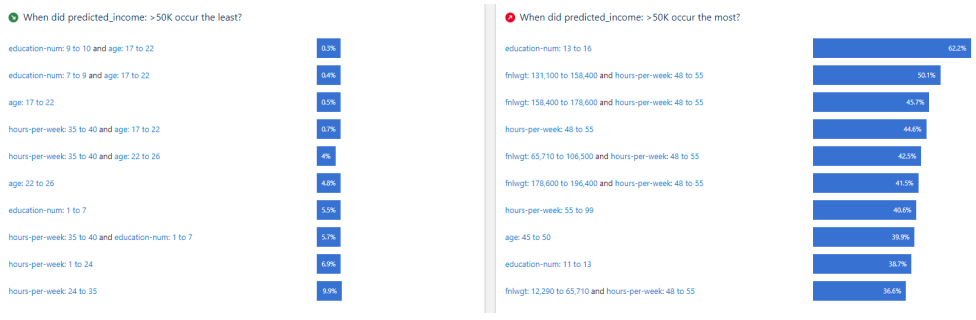


Figure 4.18: Insights: minimising predicted income.

# Chapter 5

# Leaving Analyse for Bias Unchecked on Every Variable

In Chapter 4, we performed various experiments to identify the best-performing model, "Top Predictors". We identified variables for "Disparate Impact": the recurring variables identified in both alerts were *age*, *education_num*, *hours_per_week* and *final weight*. In this chapter, we intend to analyse Einstein's capabilities further by not flagging any variable for bias. This action will most likely prevent the model training from applying its - until now detected - only fairness mechanism and metric, the "Four Fifths Rule". The starting assumption is that Einstein will not use these metrics if variables are not explicitly flagged for "Analyse for Bias".

## 5.1 Experiment Overview

We want to understand what kind of alerts and insights are identified when the model training is not mainly focused on analysing variables' selection rates. As in the previous chapter, the training will include all dataset variables. Any "?" values (unknown values in categorical variables) will be added to the "Other" categories. All possible algorithms will be tested and compared with each other, starting with GLM and all default selections, changing to GBM, XG Boost, and Random Forest, and ending in a Model Tournament after testing different configurations and applying Einstein's improvement suggestions (if there are any). Furthermore, for evaluation metrics, we will continue with Accuracy, Precision, Recall, F1-Score, and others mentioned before, such as Negative Predicted Value or the model's Threshold. Accuracy and the F1-Score will significantly impact our analysis and comparisons as in the previous chapter. Metrics such as AUC, Mean Class Per Error or Log Loss, included in the "K-fold Cross Validation" with k=4, will also be analysed and compared to the equivalent algorithms in the previous chapter.

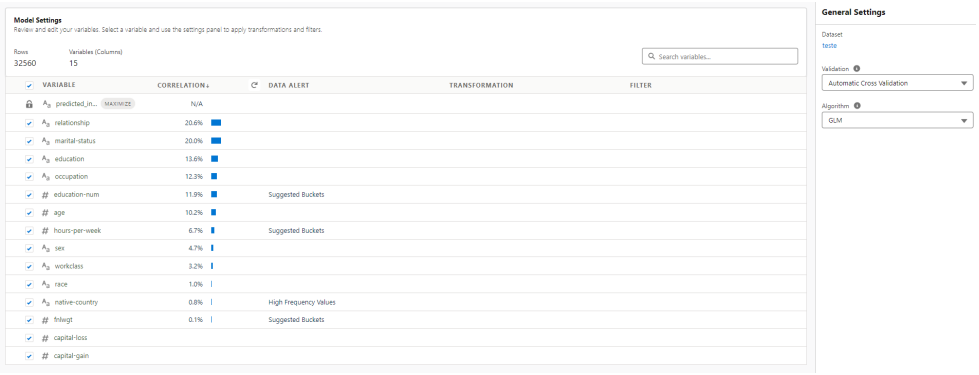


Figure 5.1: Dataset settings: default algorithm selection and no variables flagged for bias.

## 5.2 First Experiment: Leaving Selections As-Is

As per Figure 5.1, no variable will be explicitly checked for bias analysis for this and future versions.

Before training the GLM model, we were able to see that there were already some preliminary alerts and improvement suggestions from Einstein, such as "**Suggested buckets**" or "**High-Frequency Values**" - specifically for the *nativecountry* variable and the "United-States" values, meaning that there are more United States observations than for other countries. After training, we will look for these preliminary alerts on the "Data Insights" tab and "Data Alerts" section. Consequently, the most logical approach should be to follow these improvement suggestions. Moreover, we will also be looking for any alert or insight that could lead to the identification of potentially biased outcomes in the model.

### 5.2.1 First Experiment: Leaving Selections As-Is - First Results

Firstly, and as expected - since "Analyse for Bias" does not interfere with the performance, as we understood from the previous chapter - we discovered that the metric values did not change from the previous GLM version where all variables were flagged. As always, we highlight Accuracy and the F1-Score, which stayed at 80.1% and 49.2% respectively (as shown in Tables 5.1 and 5.2). As for the threshold, our values always lie between 45% and 50% approximately. Moreover, and present in Table 5.3, are the same values for GLM as before, when all variables were flagged for bias analysis. The "Top Predictors" were *age*, *education_num*, *hours_per_week* and *final weight*.

Not checking any variable for bias analysis resulted in "No Major Issues Detected" for model quality assessment, which is directly tied to selection rate and bias analysis. Nevertheless, there were three data alerts, all of which were "Suggested Buckets" improvements- see Figure 5.2. There was no mention or suggestion for the previously highlighted "High-Frequency Values" alert in the *nativecountry* variable (for the United States category). In total, there were three improvement suggestions: one for *education_num*, another for *final weight*, and a final one for *hours_per_week*. The following logical approach was to accept these suggestions and see if the model performance would improve.

### 5.2.2 First Experiment: Leaving Selections As-Is - New Versions

Accepting all suggested improvements did not result in better model performance, as metrics stayed at the same values or suffered a minimal change. The most remarkable differences in metric values we identified were for Recall - from 0.399 to 0.375 - and F1-Score - from 0.492 to 0.474. There were no data alerts for this version, but there were data insights from which we highlighted the following:

- For education, Einstein's training took preference in predicting a greater income (> 50,000 dollars) when the instruction years were above 12 - a high degree of education. The predictions for an income greater than 50,000 dollars was 24.38% above the average for this particular group;
- Only for ages above 33 years old were the predictions for an income greater than 50,000 dollars above average;
- For work hours in a week, the greater the number of hours, the greater the percentage of predictions above average, starting specifically 43 hours and above;
- The optimal values for each of the top predictor variables to achieve their best performance were education between 13 and 16 years, age between 50 to 58 years old, and work hours in a week between 45 and 50.

Table 5.1: GLM: Key Performance Metrics Results.

| Metric | Einstein Description | Result |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.801 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.399 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.64 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.83 |
| F1-Score | Harmonic average of precision and recall | 0.492 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.647 |
| Informedness | Measures how informed the model is about positives and negatives | 0.328 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.47 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.393 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.483 |

## 5.3 Second Experiment: Changing the Algorithm - GBM

GLM performed similarly when compared to having all variables flagged for bias (Chapter 4) with the addition of new model improvement suggestions not identified in the previous experiment.

Table 5.2: GLM: Income Prediction - Confusion Matrix for First Experiment.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 3130 True Positives (TP) | 1759 False Positives (FP) | **64.02% Positive Predicted Value** |
| | **<=50K Negative** | 4711 False Negatives (FN) | 22960 True Negatives (TN) | **82.97% Negative Predictive Value** |
| | | **39.92% True Positive Rate** | **92.88% True Negative Rate** | |

Even without explicitly flagging variables for bias analysis, we got some insights about Einstein's behaviour for variable value selection when predicting income.

In this section, we will perform the same tests as before. Only in this version will we train the model with the GBM algorithm to understand if there are significant or minimal changes to model quality, performance, and biased behaviour. Moreover, in order to make a reliable comparison, we will revert to the improvement suggestions adopted from the GLM model.

### 5.3.1 Second Experiment: Changing the Algorithm - GBM - First Results

From our analysis, model performance with GBM did not improve or worsen compared to the previous GBM test. The performance values were maintained, but three new data alerts emerged, all suggested buckets for *education_num*, *hours_per_week*, and *final weight*. It is interesting to see that when "Analyse for Bias" was checked, there were no "Suggested Buckets" improvements, nor any model improvement suggestion. Since Einstein detected issues with the model quality with alerts for disparate impact, our assumption, for now, is that only when model quality is "good" by Einstein standards does the tool start to check for further developments in model performance.

As we saw in the previous section, metric values did not change, strengthening our earlier assumptions of "Analyse for Bias" being a diagnostic tool and not impacting model training. By analysing Table 5.4 and Table 5.5, we can see that Accuracy improved by 0.1%, whilst the F1-Score improved by 0.5%. It is safe to say that these results are not bad, but at the same time, we have not seen a significant change(improvement) in values since the beginning of our tests. We also conclude from Table 5.6 that the cross-validation results did suffer a slight change in value, not straying away from the same range as the first GBM version. We have also come to understand that Einstein does not seem to display any bias towards an individual's race, ethnicity or sex, as the "Top Predictors" are, until now, variables that make sense to be influential when predicting someone's income, as a more significant amount of instruction years should result in higher-ranking careers, leading to higher salaries. Linked to this, the older the individual, the greater the number of years they had to level up through the ranks, career or education-wise. This is not always true, but from our point of view, these variables should carry a greater weight in predictions when a model tries to decide whether to choose a greater or lesser income.

Table 5.3: GLM: 4-Fold Cross Validation Results.

| Metric Name | Einstein Description | Training Set | Validation Set | Fold 1 | Fold 2 | Fold 3 | Fold 4 |
|---|---|---|---|---|---|---|---|
| **Number of Rows** | This is the number of rows used for each fold and set. | 32560 | 32560 | 8148 | 8062 | 8141 | 8209 |
| **AUC** | The Area Under the Curve (AUC) represents the rate of correct classification by a logistic model. An AUC of 0.5 means that the model performs no better than random guessing. An AUC of 1.0 means that the model correctly classifies data 100% of the time, which can indicate data leakage. | 0.823 | 0.823 | 0.817 | 0.815 | 0.815 | 0.817 |
| **GINI** | The Gini Index quantifies how closely this logistic model performs to a theoretically best possible model. | 0.647 | 0.647 | 0.634 | 0.63 | 0.629 | 0.634 |
| **Log Loss** | Logarithmic Loss measures model performance on a scale of 0 to 1, where 0 represents a perfect model (predictions correctly match observations 100%). The less frequently that the predicted probability correctly matches actual observations (lower performance), the higher the log loss. | 0.418 | 0.418 | 0.429 | 0.429 | 0.423 | 0.421 |
| **Mean Per Class Error** | The Mean Per Class Error measures how often the predictions are wrong. The lower the value, the fewer the wrong predictions, and therefore the better the model. | 0.262 | 0.262 | 0.266 | 0.274 | 0.268 | 0.265 |

### 5.3.2 Second Experiment: Changing the Algorithm - GBM - New Model Versions

There were three improvement suggestions to apply, one of them being for *hours_per_week* to be divided into 14 buckets, as shown in Figure 5.3. Since these are suggestions from the tool, the expectation tends toward a more favourable result than its previous version. Nevertheless, that was not the case for this version, as Accuracy decreased to the same value as the GLM model, and the F1-Score suffered a 2.2% decrease from the previous version. However, we did see a slight improvement in the Precision value, of approximately 0.9%, which means that the number of actual positives - a "positive" being a "> 50K" value - among the predicted positives increased. As for insights, the information seems repetitive since most graphs point to the fact that fewer work hours, younger age, and a lower level of education lead to a more negligible probability of the model predicting an income greater than 50,000 dollars for observations with these profiles.

Additionally, we saw no need to keep training new model versions since Einstein was not able to pick up on any "Disparate Impact" when variables were not explicitly flagged, there were no more alerts to attend to, and there were no issues with model quality - according to Salesforce standards.

## 5.4 Third Experiment: Changing the Algorithm - XG Boost

In the previous chapter (Chapter 4), XG Boost was selected as the best-performing algorithm. Consequently, we expect it to perform at the same or better level. As adjustments were made in variable buckets for GBM, we will revert them for a proper comparison.

The preliminary assumption is that the algorithm will have the same metric values and model quality performance as it occurred for GLM and GBM in this chapter. Moreover, in this chapter, we saw new suggestions for improvement apart from the "Disparate Impact" alerts. To clarify, there is also an expectation that some suggested buckets will be applied to the usual top predictor variables in the model.

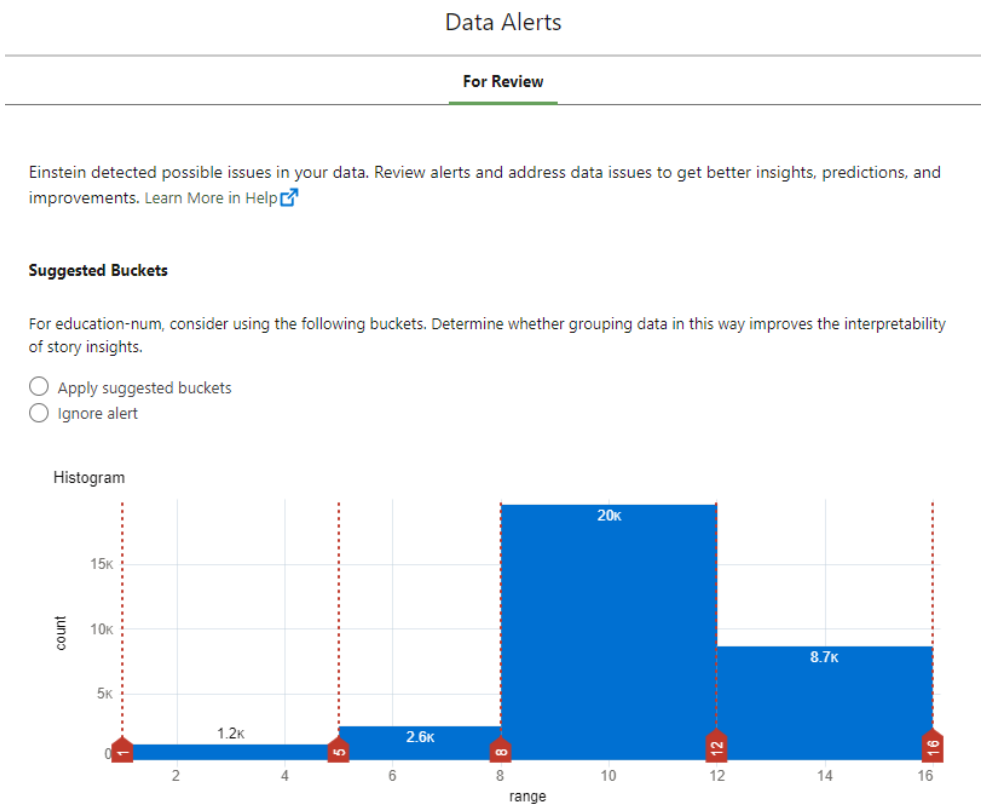


Figure 5.2: Data alert: suggested buckets for *education_num*.

### 5.4.1 Third Experiment: Changing the Algorithm - XG Boost - First Results

As aforementioned, the preliminary assumption was that the algorithm would have the same metric values and model quality performance as it occurred for GLM and GBM. This was partially the case. By analysing Tables 5.7 and 5.8, we conclude that Accuracy maintained its value, but Recall and the F1-Score suffered a 2.4% and 1.6% decrease. Precision improved its value by 1%. Contrary to performance metric values, in Table 5.8, we saw that no value changed for AUC, GINI, Log Loss or Mean Per Class Error compared to the XG Boost version with all flagged variables. Education and age significantly impacted predictions, and three "Suggested Buckets" improvements could be applied to the next model version. As we have concluded, suggested buckets are identified before training and do not seem to change after, no matter the algorithm used. Nevertheless, we applied and tested them to analyse if any metric would suffer a significant improvement. By analysing GLM and GBM training, we only saw metric values worsen.

### 5.4.2 Third Experiment: Changing the Algorithm - XG Boost - New Model Versions

There is a slight paradox when analysing the results: the metrics decreased in value, but no model issues were detected. This means that the optimal training for Einstein - good model quality and

Table 5.4: GBM: Key Performance Metrics Results.

| Metric | Einstein Description | Result |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.802 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.405 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.641 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.831 |
| F1-Score | Harmonic average of precision and recall | 0.497 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.654 |
| Informedness | Measures how informed the model is about positives and negatives | 0.333 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.472 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.396 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.479 |

no problems detected - was not the one with the highest metric values. The performance metrics were similar or the same as GLM and GBM. Nevertheless, this was the best-performing algorithm in this chapter so far. Having already trained with Random Forest and the Model Tournament version, the expectation is that XG Boost will be the best-performing algorithm for metrics and model quality.

## 5.5 Fourth Experiment: Changing the Algorithm - Random Forest

For our fourth and last test for algorithm comparison, we will train the model with the Random Forest algorithm. Since most tests in this chapter got improvement suggestions - notably “Suggested Buckets” suggestions - we expect to see this reflected in this model. Moreover, we want to analyse whether these suggestions, contrary to other algorithms, improve model performance.

### 5.5.1 Fourth Experiment: Changing the Algorithm - Random Forest - First Results

Firstly, and as expected, Random Forest was the worst-performing algorithm from the selections available. Compared to XG Boost, we saw a 1.1% decrease in Accuracy and a 7.7% decrease in the F1-Score. Moreover, for cross-validation results, there was a 2% and 4.3% decrease for AUC and GINI, respectively, and a slight increase in Log Loss and Mean Per Class Error values. This means that there were more wrong predictions in this model. In comparison with the previous chapter’s Random Forest model, there were also some changes in metrics, present in Tables 5.10, 5.11 and 5.12:

- 0.2% decrease in Accuracy;
- 5% decrease in F1-Score;

Table 5.5: GBM: Income Prediction - Confusion Matrix for Second Experiment.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 3178 True Positives (TP) | 1782 False Positives (FP) | **64.07% Positive Predicted Value** |
| | **<=50K Negative** | 4663 False Negatives (FN) | 22937 True Negatives (TN) | **83.11% Negative Predictive Value** |
| | | **40.53% True Positive Rate** | **92.79% True Negative Rate** | |

- 6.3% decrease in Recall;
- 2.7% increase in Precision;
- No changes for 4-fold cross-validation results.

For improvements, we saw the three standard alerts that also appeared for other algorithms: suggested buckets for education, work hours and final weight. Once more, there were no mentions of potential bias or unfairness in the model as the model quality was considered good by Einstein standards (if there were any disparate impact alerts, they would appear in this section).

### 5.5.2 Fourth Experiment: Changing the Algorithm - Random Forest - New Model Versions

Applying the suggested improvements for this algorithm positively impacted model performance, as performance metrics values increased: 0.2% for Accuracy and 4.2% for F1-Score. Meanwhile, for cross-validation results, there was a 0.6% decrease in value for AUC and a 0.6% increase for Log Loss. Additionally, Mean Per Class Error significantly decreased, meaning more wrong predictions overall.

## 5.6 Fifth Experiment: Changing the Algorithm - Model Tournament

As performed in Chapter 4, we will do a Model Tournament to close this chapter's experiments. We want to confirm our selection for the best-performing model with the XG Boost algorithm. However, we will not be training an additional model to minimise the predicted income since we have already concluded that particular selection has little to no impact on results. As for every other test, we will revert the application of the suggested improvements to get an accurate comparison.

As expected, **XG Boost** was selected as the best-performing model, with the same previous metric values and cross-validation results. By performing the model tournament, we could compare our analysis with Einstein's and confirm our conclusions for the best-performing algorithm.

Table 5.6: GBM: 4-Fold Cross Validation Results.

| Metric Name | Einstein Description | Training Set | Validation Set | Fold 1 | Fold 2 | Fold 3 | Fold 4 |
|---|---|---|---|---|---|---|---|
| **Number of Rows** | This is the number of rows used for each fold and set. | 32560 | 32560 | 8148 | 8062 | 8141 | 8209 |
| **AUC** | The Area Under the Curve (AUC) represents the rate of correct classification by a logistic model. An AUC of 0.5 means that the model performs no better than random guessing. An AUC of 1.0 means that the model correctly classifies data 100% of the time, which can indicate data leakage. | 0.827 | 0.827 | 0.816 | 0.814 | 0.812 | 0.816 |
| **GINI** | The Gini Index quantifies how closely this logistic model performs to a theoretically best possible model. | 0.654 | 0.654 | 0.632 | 0.627 | 0.623 | 0.632 |
| **Log Loss** | Logarithmic Loss measures model performance on a scale of 0 to 1, where 0 represents a perfect model (predictions correctly match observations 100%). The less frequently that the predicted probability correctly matches actual observations (lower performance), the higher the log loss. | 0.415 | 0.415 | 0.429 | 0.43 | 0.425 | 0.421 |
| **Mean Per Class Error** | The Mean Per Class Error measures how often the predictions are wrong. The lower the value, the fewer the wrong predictions, and therefore the better the model. | 0.264 | 0.264 | 0.264 | 0.268 | 0.273 | 0.271 |

## 5.7 Conclusions

The purpose of this chapter was to replicate all tests performed in the previous chapter without explicitly flagging any of the variables for bias analysis. We wanted to analyse what type of insights and data alerts would be raised when Einstein is not told to search for disparate impact and imbalance in variables' selection rates. From our tests and experiences in the platform, we arrived at the following conclusions:

1. XG Boost was still the best-performing algorithm from the possible options;

2. For every algorithm, there were improvement suggestions, particularly "Suggested Buckets", which had not appeared when variables were checked for "Analyse for Bias". This happened for every variable which had previously only been identified for Disparate Impact alerts. Even XG Boost, which in Chapter 4 did not display any model issue;

3. Overall, applying suggested improvements did not result in better model performance and evaluation metrics;

4. There was no mention of any potential bias-related issue with each different version and algorithm. There was a preliminary alert - before training the model when configuring settings for the dataset - which mentioned a "High Frequency" alert for *nativecountry*, in which there were more "United States" categorical values than any other value for the variable. Nevertheless, after training, there was no suggested action or improvement to mitigate any negative impact that could occur;

5. Without checking the "Analyse for Bias" checkbox, there seems not to be any bias-preventing tool or diagnostic mechanism to inform users of a potentially biased model.

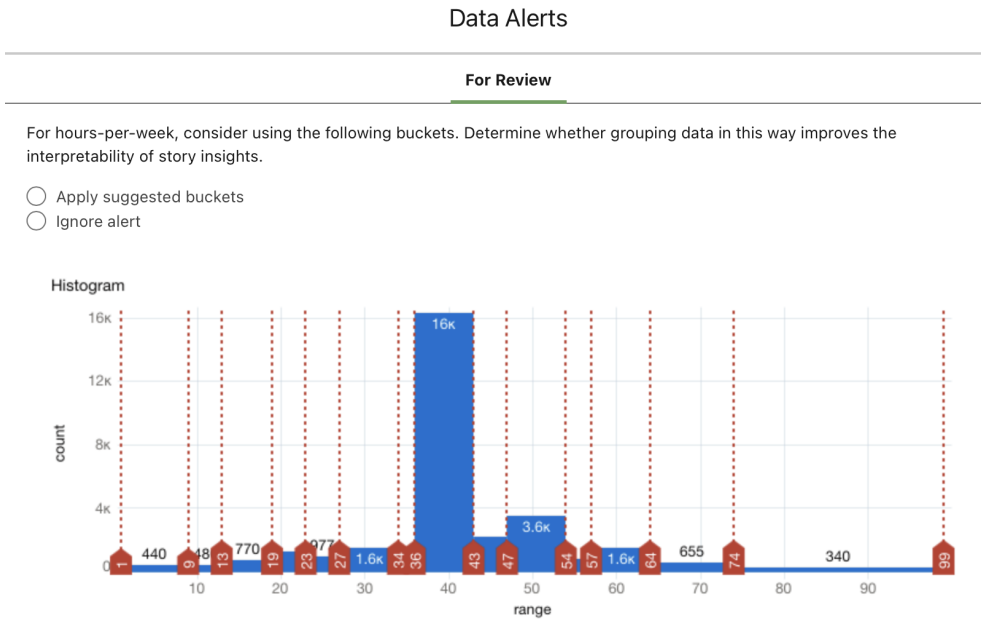


Figure 5.3: Data alert: suggested buckets for *hours_per_week*.

Table 5.7: XG Boost: Key Performance Metrics Results.

| Metric | Einstein Description | Result |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.804 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.389 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.655 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.828 |
| F1-Score | Harmonic average of precision and recall | 0.488 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.662 |
| Informedness | Measures how informed the model is about positives and negatives | 0.324 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.483 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.396 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.504 |

Table 5.8: XG Boost: Income Prediction - Confusion Matrix for Third Experiment.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 3051 True Positives (TP) | 1606 False Positives (FP) | **65.51% Positive Predicted Value** |
| | **<=50K Negative** | 4790 False Negatives (FN) | 23113 True Negatives (TN) | **82.83% Negative Predictive Value** |
| | | **38.91% True Positive Rate** | **93.5% True Negative Rate** | |

Table 5.9: XG Boost: 4-Fold Cross Validation Results.

| **Metric Name** | **Einstein Description** | **Training Set** | **Validation Set** | **Fold 1** | **Fold 2** | **Fold 3** | **Fold 4** |
|---|---|---|---|---|---|---|---|
| **Number of Rows** | This is the number of rows used for each fold and set. | 32560 | 32560 | 8148 | 8062 | 8141 | 8209 |
| **AUC** | The Area Under the Curve (AUC) represents the rate of correct classification by a logistic model. An AUC of 0.5 means that the model performs no better than random guessing. An AUC of 1.0 means that the model correctly classifies data 100% of the time, which can indicate data leakage. | 0.831 | 0.831 | 0.815 | 0.813 | 0.81 | 0.815 |
| **GINI** | The Gini Index quantifies how closely this logistic model performs to a theoretically best possible model. | 0.662 | 0.662 | 0.63 | 0.625 | 0.621 | 0.63 |
| **Log Loss** | Logarithmic Loss measures model performance on a scale of 0 to 1, where 0 represents a perfect model (predictions correctly match observations 100%). The less frequently that the predicted probability correctly matches actual observations (lower performance), the higher the log loss. | 0.41 | 0.41 | 0.43 | 0.431 | 0.427 | 0.422 |
| **Mean Per Class Error** | The Mean Per Class Error measures how often the predictions are wrong. The lower the value, the fewer the wrong predictions, and therefore the better the model. | 0.257 | 0.257 | 0.271 | 0.267 | 0.269 | 0.267 |

Table 5.10: Random Forest: Key Performance Metrics Results.

| **Metric** | **Einstein Description** | **Result** |
|---|---|---|
| Accuracy | Measures the proportion of outcomes that the model predicted correctly | 0.793 |
| True Positive Rate / Recall | Number of predicted positives among all the actual positives: TP/(TP+FN) | 0.3 |
| Positive Predictive Value / Precision | Number of actual positives among the predicted predicted positives: TP/(TP+FP) | 0.656 |
| Negative Predicted Value | Number of actual negatives among all the predicted negatives: TN/(TN+FN) | 0.81 |
| F1-Score | Harmonic average of precision and recall | 0.411 |
| GINI | Quantifies how closely this logistic model performs to a theoretically best possible model | 0.662 |
| Informedness | Measures how informed the model is about positives and negatives | 0.25 |
| Markedness | Measures the trustworthiness of positive and negative predictions by the model | 0.466 |
| MCC | Measures the quality of this logistic model. It provides a more even representation of the four parts of the confusion matrix | 0.341 |
| Threshold | Value ranging from 0 to 1. It represents the tradeoff between the true positive and false positive rates | 0.504 |

Table 5.11: Random Forest: Income Prediction - Confusion Matrix for Fourth Experiment.

| | | **Actual predicted_income** | | |
|---|---|---|---|---|
| | | **>50K Positive** | **<= 50K Negative** | |
| **Predicted predicted_income** | **>50K Positive** | 2349 True Positives (TP) | 1232 False Positives (FP) | **65.6% Positive Predicted Value** |
| | **<=50K Negative** | 5490 False Negatives (FN) | 23480 True Negatives (TN) | **81.05% Negative Predictive Value** |
| | | **29.97% True Positive Rate** | **95.01% True Negative Rate** | |

Table 5.12: Random Forest: 4-Fold Cross Validation Results.

| **Metric Name** | **Einstein Description** | **Training Set** | **Validation Set** | **Fold 1** | **Fold 2** | **Fold 3** | **Fold 4** |
|---|---|---|---|---|---|---|---|
| **Number of Rows** | This is the number of rows used for each fold and set. | 32560 | 32560 | 8148 | 8062 | 8141 | 8209 |
| **AUC** | The Area Under the Curve (AUC) represents the rate of correct classification by a logistic model. An AUC of 0.5 means that the model performs no better than random guessing. An AUC of 1.0 means that the model correctly classifies data 100% of the time, which can indicate data leakage. | 0.81 | 0.81 | 0.815 | 0.811 | 0.81 | 0.814 |
| **GINI** | The Gini Index quantifies how closely this logistic model performs to a theoretically best possible model. | 0.619 | 0.619 | 0.63 | 0.623 | 0.62 | 0.628 |
| **Log Loss** | Logarithmic Loss measures model performance on a scale of 0 to 1, where 0 represents a perfect model (predictions correctly match observations 100%). The less frequently that the predicted probability correctly matches actual observations (lower performance), the higher the log loss. | 0.434 | 0.434 | 0.433 | 0.435 | 0.429 | 0.426 |
| **Mean Per Class Error** | The Mean Per Class Error measures how often the predictions are wrong. The lower the value, the fewer the wrong predictions, and therefore the better the model. | 0.274 | 0.274 | 0.269 | 0.274 | 0.285 | 0.269 |

# Chapter 6

# Automated Model Training

Until now, all experiments have been conducted with the manual training option in Einstein to facilitate the process of variable selection and bias analysis - manually flagging each variable for "Analyse for Bias". In this chapter, we intend to choose the **"Automated"** option - in which Einstein selects the seemingly "most relevant" variables -to understand which features Einstein selects for model training and if there is a possibility the tool automatically flags any feature for bias analysis. All other selections, such as the target variable (*predicted_income*) and the maximisation of "predicted_income: >50K" will remain the same.

## 6.1 Einstein Selections: First Analysis

According to Einstein's description, the Automated model training should be able to select the most relevant variables for the goal in our particular use case. According to Salesforce's training platform, Trailhead, the Auto Machine Learning (AutoML) capacity in Einstein "includes feature engineering, which automatically combs through data and begins identifying the most significant features" (Trailhead, 2024). This section focuses on analysing Einstein's variable and algorithm selections.

Firstly, we can see that GLM was the selected algorithm, and the "Top Predictors" were *age*, *education_num*, *fnlwgt* and *hours_per_week*, as seen in all other models trained in previous chapters. Thirteen of fifteen available variables were chosen, with *capital_loss* and *capital_gain* left out of training. According to Einstein's analysis, these two variables were less impactful than the others in predicting each observation's income.

For evaluation metrics, the model resulted in an **Accuracy** of 80.1% and **F1-Score** of 49.2%, which stand in the average of values obtained in other models trained manually. Regarding the 4-fold cross-validation results, we got an average of 82% for AUC, 42% Log Loss, and 26.7% for Mean Per Class Error. Once more, these values were not better or worse than the ones in previous models. There were three "Suggested Buckets" improvement suggestions: one for *education_num*, one for *fnlwgt*, and one for *hours_per_week*, which are the same ones raised for

the manually trained GLM model in chapter 5, when "Analyse for Bias" was not flagged in any variable.

Finally, there were no "Disparate Impact" alerts since no variable was flagged for bias analysis. This answers the question at the beginning of this chapter, in which our goal was to understand if there was a possibility that Einstein would automatically flag variables for bias. Seeing this did not occur. Our logical next step is to flag the automatically selected variables for bias and train the model again. Additionally, we will accept some suggested improvements alongside the "Analyse for Bias" flags. Moreover, we do not have any fairness metrics to analyse since not even the 80% rule was applied in the training.

It is worth noting that the left-out variables, *capital_gain* and *capital_loss* do not seem to be relevant enough to predict our target variable, but in external models, at least *capital_gain* appear as one of the essential features, as we will see in Chapter 7. GLM was the selected algorithm in this model. As we have concluded from other experiments, this is not the best-performing one, which leaves us with a new question: is there any model tournament occurring in the Automated version, or is the default model - in this case, GLM - selected? We also discovered no data cleanup regarding the "?" values in categorical variables.

## 6.2 Einstein Selections: Bias Analysis and Improvement Suggestions

In this section, we will join the bias analysis feature with Einstein's improvement suggestions from the Automated model. Any "?" values in categorical variables will be grouped in the "Other" categories. Since Einstein provides this option, we are just mimicking what has already been done in previous chapters. Moreover, we will leave the algorithm as GLM but later change it to our best-performing one in Einstein, XG-Boost.

Joining the improvement suggestions and bias analysis resulted in a slight decrease in metrics values, but not significant enough to affirm any detrimental impact. For the model quality checks, applying the bias flags resulted in the "Issues Detected' assessment due to four new data alerts regarding one sensitive variable, *age* and three others: *education_num*, *hours_per_week* and *fnlwgt*. This time, there were two tabs in the data alerts popup, one for the new "Analyse for Bias" alerts and the other for resolved issues, which in this case were the improvement suggestions already accepted.

Einstein provides a "Model Comparison" option, which has not been used until now since there was no experiment in which we have applied both improvements and bias analysis. In this automated experiment, it made sense to compare both versions. The "Model Comparison" option provides the metrics we are accustomed to and some new "Advanced Metrics" that are not shown in the regular interface, such as the Akaike Information Criterion (AIC). The **AIC** compares models and selects the best one based on a score, which is to be minimised. The baseline is that the model with the lowest score is selected (Brownlee, 2020b). Keeping this in mind, the first automated version was the best performer, not only because of the lowest AIC value but also due to lower

loss values. Nevertheless, since the value difference was minimal, we cannot say one model was better.

Looking at the disparate impact alerts, there is one that we can identify as a potentially biased treatment from Einstein's model training in the age variables. By looking at the selection rates, there was a clear difference between individuals lying in categories above forty years old and all others, notably younger. We can further illustrate this by taking two coefficients for two age categories: for the "17 to 22" category, we have a coefficient of -1.2116, and for the "51 to 58" category, we have a 0.5274 coefficient. Since the coefficient "*represents the impact that an explanatory variable (or a pair of explanatory variables) has on the outcome variable*" (Salesforce, 2024a), it is understandable that the model is favouring observations with older ages. Unfortunately, "Analyse for Bias" is a diagnostic tool, and Einstein's only suggestion was to exclude age from the model. As this is the first model where we have joined diagnostics with improvements, we will exclude *age* to analyse the impact on model quality and performance and other disparate impact alerts. Will it prevent an adverse effect on the other flagged variables?

### 6.2.1 Following Disparate Impact Suggestions: Excluding *age* from the model

All metrics worsened by excluding *age* from our model. We can see a decrease of 1.5% in Accuracy and a 10.9% decrease in F1-Score. Additionally, there was a slight increase in Log Loss from 42% to 46.6%. From the higher AIC value, we concluded that, comparably, the previous models performed better overall. Examining this leads us to conclude that excluding *age* is not a viable option for our use case. Here, we identify one case in which the 80% rule was applied but cannot impact a change since the variable in question is too important - high feature importance and correlation with the target variable - to exclude. Overall, excluding this variable resulted in a worse-performing model.

### 6.2.2 Changing the algorithm: Training with XG Boost

As we have seen in previous experiments, models where XG Boost was the algorithm did not raise any disparate impact alerts, which means that the selection rates across groups for different variables - namely education, age, work hours and final weight - were more or less equal, having none of them been below 80%.

We are now using a model with improvements and bias flags to test the overall best-performing algorithm and equal treatment across selection rates.

As expected, training with XG Boost resolved all bias-related alerts, which seemingly displays that this algorithm is pretty equal in selection rates and does not discriminate towards certain groups within sensitive variables, particularly *age*, which had been flagged in the previous (GLM) model. Consequently, no alerts were raised regarding this model, and the model quality was deemed not to have any "major issues". Advancing to our model comparison, we can conclude that there were no noticeable changes in metrics. Additionally, there was a slight decrease in metrics when comparing this model with the previous bias-flagged XG Boost model, where there

were no "Suggested Buckets" improvements to apply. Furthermore, these changes do not reach the 1% mark, meaning that improving the model - according to Einstein's suggestions - did not compromise its overall performance.

From this chapter and previous experiments, we can safely affirm that this algorithm, in Einstein, is better equipped to deal with sensitive variables and possibly mistreated groups.

# Chapter 7

# Comparison to External or Out-of-the-Box Solutions

Until now, every experiment was conducted inside the Einstein platform. In this chapter, we will externally run machine learning models, with attention to algorithms also available in Einstein, for better comparison purposes. Furthermore, we will try to leverage the "Upload Model" option in the platform, which is expected to allow users to upload external Scikit or TensorFlow models.

## 7.1 Experiment Overview

As mentioned in Chapter 2, companies use other solutions to measure, detect, and mitigate their models' biases. In this chapter, we will apply our dataset's best and worst-performing algorithms in Einstein - **XG Boost** and **Random Forest**, respectively - with external Python solutions. This way, we can compare Einstein's performance with other solutions. We will rely on **Accuracy**, **F1-Score**, **Precision** and **Recall** for performance comparison metrics, as they were the most relevant for our Einstein models. Since Einstein only had "Selection Rate" as a fairness metric and detector for disparate impact, we plan to apply other fairness metrics - not available in Einstein - such as **Demographic Parity** or **Equalized Odds** available in **Fairlearn**. Additionally, we want to understand if these metrics use any form of selection rate as a bias-detecting metric, like Einstein.

Since Einstein allows users to upload external scikit-learn models, after concluding our external experiments with Random Forest and XG Boost, we will try to upload these models to the platform and analyse if model metrics align with the ones from the externally run models.

### 7.1.1 Random Forest

We will begin by applying the Random Forest algorithm. Since this was the worst performance in Einstein, our goal is to analyse performance in different settings. We will use the **RandomForestClassifier** algorithm from scikit-learn for this. Regarding further customisations such as hyperparameter values, we cannot perform a replica of an Einstein model since we cannot access them as Einstein end-users.

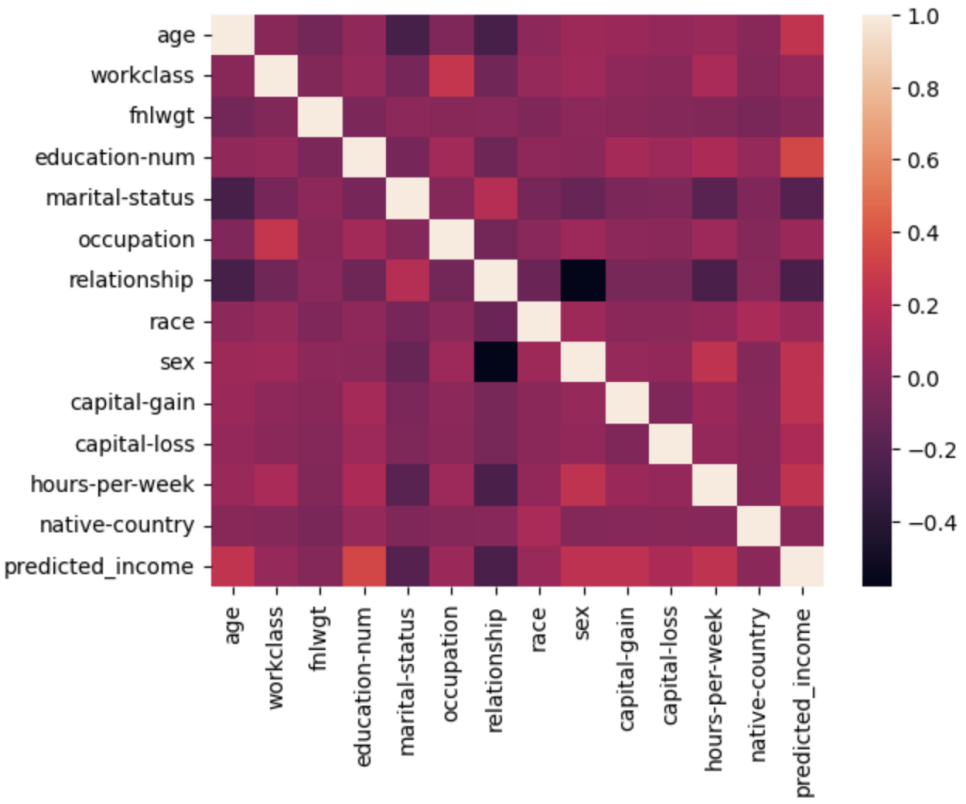


Figure 7.1: Random Forest: correlation between variables.

Firstly, we removed the dataset's blank or *NaN* values as in our Einstein models. Then we used the **OrdinalEncoder** from scikit-learn to transform our categorical values into numerical ones since it seemed to result in a more stable base to train a Random Forest in this particular case. Moreover, we removed the *education* variable from the dataframe since the dataset already had the *education_num* variable, and having two numerical education variables would be redundant. Finally, since all variables would be numerical, we also transformed the binary target variable - *predicted_income* - into a numerical one, with the value "0" for "<=50K" and "1" for ">50K". As part of Exploratory Data Analysis (EDA), we created a heatmap to check if any variables had positive or negative correlations between each other. By analysing it, (Figure 7.1) we discovered that the target variable, *predicted_income* had a significant positive correlation with ***hours_per_week***, ***capital_gain***, ***sex*** and ***education_num***. Additionally, **age** and **capital_loss** also exhibited positive values for correlation, but not as significant when compared to the previously mentioned variables. Comparing these variables with recurring "Top Predictors" in Einstein, we found some common ones like *age*, *hours_per_week* and *education_num*.

We performed an 80/20 split for training and testing to mimic Einstein configurations. Running the model is a straightforward process when using the scikit-learn library. After this, we discovered that performance metrics were significantly better than in Einstein's model:

- **Accuracy** was 86.4% when in Einstein, we only got a 79.5%;

- One of the biggest differences was for the **F1-Score**, which was 69.4% for scikit's classifier, and 46.1% for Einstein;
- **Precision** went from 62.9% to 75.7%;
- **Recall** was the metric with the greatest increase in value, from 36.3% in Einstein to 64.2% in scikit's classifier.

For every model, Einstein presents its "Top Predictors", which we concluded are generally the same for every algorithm tested. Furthermore, we decided it would be relevant and an excellent comparison to discover the feature importance in this model. Replicating Einstein's analysis, we identified the four most important features in this model: *fnlwgt*, *age*, *education_num* and *capital_gain*. It is interesting to discover that ***capital_gain*** significantly impacted correlation and feature importance for the income predictions but was never considered relevant in Einstein's experiments.

After training, testing, and evaluating the model's performance, we could finally test Fairlearn's metrics: Demographic Parity Difference and Ratio and Equalized Odds Difference and Ratio. Starting with Demographic Parity, according to Fairlearn's documentation, it is a "*fairness metric whose goal is to ensure a machine learning model's predictions are independent of membership in a sensitive group*" and "*in the binary classification scenario, demographic parity refers to equal selection rates across groups*" (Fairlearn, 2018). Equalized Odds is a metric that "ensures a machine learning model performs equally well for different groups" (Fairlearn, 2018). To calculate these fairness values we used Fairlearn's **demographic_parity_difference()**, **demographic_parity_ratio()**, **equalized_odds_difference()** and **equalized_odds_ratio()**. These metrics take as parameters the ground truth, the predictions and what the user defines as "sensitive data". For sensitive data we firstly selected *age*, *race*, *sex* and *native_country*. We got a result of 1.0 for demographic parity difference, meaning that our model did not achieve demographic parity. It is safe to assume that selection rates across groups were highly disparate. Analysing the demographic parity ratio of 0.0, the model did not pass the "Four-Fifths Rule" or the 80% threshold already known from our Einstein experiments. Notably, excluding age, all variables selected for "sensitive data" were not flagged for disparate impact in Einstein, meaning there was no selection rate below the 80% mark. The equalised odds metric considers true and false, as well as positive and negative rates. The model is considered fair if these are the same for all groups. Once more, the results for this model were far from ideal, with a value of 1.0 for the difference and 0.0 for the ratio. A fair model would have a value close to 0.0 for the difference and a value close to 1.0 for the ratio. In conclusion, these metrics can result in values between 0 and 1 and are used to audit models for fairness. We could analyse and work with a different fairness tool by applying these functions. Nevertheless, using these metrics on external models is a different analysis entirely. There is no ground for comparison with Einstein since the models were external Python scripts we created, meaning that any fairness audit performed is unrelated to the models and experiments inside Salesforce.

### 7.1.2 XG Boost

Following our previous experience with Random Forest, the worst-performing model in Einstein, we trained an XG Boost model to compare results with Einstein's best-performing model. We removed any blank or NaN values from the dataset as in Random Forest. After that, we transformed our target variable, *predicted_income*, into a numerical one. The value "<=50K" was mapped to the value of "0.0", whilst the value ">50K" was mapped to the value of "1.0". All other variables were left as-is, either categorical or numerical. We also performed simple hyperparameter tuning with the **BayesSearchCV** optimiser from **scikit-optimize**. Before applying the BayesSearchCV optimiser, we first defined the *search_space* dictionary with all default values, which we then defined as the *search_space* parameter for the optimiser. Other parameters included the number of cross-validation folds, which we set as four - to replicate the Einstein configuration -, the number of iterations, set at 10, and finally, the scoring, for which we selected "Accuracy".

For performance evaluation results, we got the following:

- **Accuracy** was 86.8%, which is an increase of 6.4% compared to Einstein's result and 0.4% compared to the Random Forest model in the previous section. Contrary to Einstein, the Accuracy difference between the models is near zero;
- Once more, **F1-Score** had a significant increase in value, from 50.4% in Einstein, to 69.9% in our scikit and xgboost-based model. Compared to Random Forest, it suffered a 0.5% increase;
- **Precision** suffered a 13.9% increase compared to Einstein, and a 1.1% increase compared to the external Random Forest model;
- **Recall** demonstrated an increase of 22.4% in value, with a result of 63.7% for the external model. For Random Forest, the Recall value was greater than XG Boost by 0.5%.

After running a feature importance analysis, we discovered that *capital_gain*, *age*, *capital_loss* and *hours_per_week* were the most important ones in the prediction process of the model, two of which - *capital_gain* and *capital_loss* - were not identified as a "Top Predictor" in Einstein.

Advancing to Fairlearn metrics, we first define the sensitive data as a junction of **age**, **race**, **sex** and **native_country** values. According to our results of "1.0" for demographic parity difference and "0.0" for demographic parity ratio, it is safe to say that our model did not achieve demographic parity when analysing this particular group of defined sensitive data. Furthermore, we achieved the same far-from-ideal results, with a value of "1.0" for equalised odds difference and "0.0" for equalised odds ratio. As aforementioned in the Random Forest section, this was a good exercise for trying other fairness tools in models, but it cannot be compared with Einstein.

## 7.2 Uploading Models to Einstein

As mentioned at the beginning of this chapter, now that we have tested the best and worst-performing algorithms outside of Einstein, we will upload each model separately to the platform,

run them, and explore results and metrics. After this, we will compare them with the external ones.

In order to upload a model to Einstein, we need to comply with the following requirements (SalesforceHelp, 2024):

- Group our model files in a .zip format;
- Include these three files (Help, 2024):
  - **saved_model.pkl**, the scikit-learn model generated with the pickle module from Python;
  - **data_processor.py**: a Python script which must contain:
    - preprocessor() function where any data cleaning or reshaping is performed;
    - postprocessor() function that returns predictions;
    - LABEL_COLUMN which identifies the outcome/target variable;
  - **validation.csv**, a small subset of the dataset used. I should contain the outcome variable (*predicted_income* in our case) and no more than 99 rows and 50 columns;
- When uploading the model:
  - On the "Model Runtime" selection, choose "ScikitLearn: 1.0.2 Python 3.7";
  - On the "Model Type" selection, choose "Binary Classification".

We decided that the model to be tested would be a Random Forest one, as it had the worst performance in Einstein and, in contrast, performed so well in our previous experiment with external models. Moreover, it is a scikit ensemble model (RandomForestClassifier()), which complies with Einstein's requirements. For the "saved_model.pkl", we began by transforming every categorical variable into a numerical one (*education* was dropped since we already had a numerical version). After that, we continued with a standard model training pipeline, with an 80/20 split, and focused on Accuracy, Precision, Recall and F1-Score for evaluation metrics. Lastly, as required, we saved the model in a ".pkl" format. As for the "data_processor.py" we structured it as such:

1. Define the "**LABEL_COLUMN**" - the target variable - and "**COLUMN_NAMES**" - every variable present in the dataset;
2. Create a function to prepare and clean the dataset: clean blank or "NaN" values, transform every non-numerical variable into a numerical one (again, dropping the categorical "education" variable);
3. Define the "**preprocessor**" function to read the dataset;
4. Define the "**postprocessor**" function to obtain the final predictions.

There were some difficulties and obstacles that we encountered when uploading our model:

1. We are restricted to specific versions of Python (3.7.), Scikit-Learn (1.0.2.), Pandas (1.0.5) and Numpy (1.21.5), which, in our case, meant creating a 'conda environment' with Python 3.7., Scikit-Learn 1.0.1. - we found that this version was better accepted than the actual 1.0.2.- Pandas 1.0.5 and Numpy 1.21.5. Notably, these versions are older and not in agreement with the most up-to-date ones;

2. Salesforce provides a help page listing the required files for each model type (Tensorflow or Scikit-Learn). Nevertheless, there is not enough information or examples to be able to create our scripts from that page alone. The solution was to research and communicate with colleagues to understand how to create functioning scripts;

3. We began by creating a standard scikit RandomForestClassifier() pipeline: read and clean the data, create the split, train the model and evaluate. Then, save the model in a '.pkl' format, according to scikit's documentation. Nevertheless, this was insufficient, as Einstein produced an error, stating that loading the model into Scikit was impossible. As already mentioned above, we later discovered that there was a need to involve our pre and post-processing functions in the model training;

4. Once we were finally able to have the "saved_model.pkl" and "data_processor.py" files validated, the model upload had a few failed attempts before entering the platform, mainly because the errors shown were vague, meaning that they were no help in guiding us in the right direction to correct them.

After much testing and trial-and-error attempts, we finally uploaded our model into Einstein. Circling back to the beginning of this experiment, our purpose was to upload an externally developed model into Einstein and then see its performance on the platform. However, we later discovered that this was impossible since, after uploading the model, the only options available were to clone, delete or deploy it, meaning that we could not test it in our developer environment. In an actual project, we would deploy the model to a production-type environment, where the model would run and make predictions, with results appearing in a designated Salesforce field (e.g., a page for an individual where there would be a field called 'Income' where the prediction values would appear). Regarding comparing values and metrics with other Einstein models, our only option was to run the model externally and see the results, as we could not run the model after being uploaded to Einstein.

## 7.3 Conclusions

In this chapter, we trained two models outside of the Einstein platform, Random Forest and XG boost, which in previous experiments were the worst and best-performing, respectively. Since we could not access every model configuration, we replicated what we could. We performed an 80/20 split and a 4-fold cross-validation. We tested the most relevant metrics for our models: Accuracy, F1-Score, Precision and Recall. For Random Forest, we saw a significant increase in every metric.

For Accuracy and F1-Score, respectively, we got a 6.9% and a 23.3% increase. As for XG Boost, there was a 6.4% and a 19.5% increase when compared to Einstein's model. The difference in metric values from Random Forest was not as significant as in Einstein, meaning that Random Forest performed better on the external model. When looking at the Recall value, Random Forest performed better. Overall, the models performed better outside the platform, leaving us with two possible reasons for why this might have happened:

1. Einstein's models are better equipped for bias mitigation, which slightly impacts metrics;
2. Model hyperparameters in Einstein are too general, and since we cannot access or change them, the overall model performance is affected.

Additionally, we evaluated our external models with Fairlearn's fairness metrics: demographic parity and equalised odds. We concluded that our models did not perform at their best. Still, since we cannot apply these same metrics in Einstein, these results become somewhat unnecessary, as we cannot compare them.

Finally, we experimented with the platform's "Upload Model" option. As stated before, we trained the model externally, which gave us a better performance for Random Forest. There were a few obstacles when attempting to upload the model, such as versioning or unclear errors thrown by the platform. In the end, our solution was to set up an environment with all the required - outdated - versions and a few additional checks, such as making sure the predictions were numeric (0 and 1) and the validation.csv file maintained all columns with its original data and missing values, except for the *predicted_income* variable, that should have the predicted values (values obtained by running the developed Random Forest model). Nevertheless, after getting the model into the platform into the platform, in a development/testing environment, there was not much to do besides cloning or deleting the model, meaning that we could not perform any "preview" or run the model with Einstein.

# Chapter 8

# Conclusions

In this final chapter, we will review our overall conclusions and results. Every experiment performed always had one of the initial research questions as a baseline for the work conducted. In the following sections, we will answer these questions and reveal some interesting findings regarding Einstein's predictions and overall performance.

## 8.1 Prediction Analysis

Following the conclusion of all our previous tests and experiments, it was time to perform an overall analysis of the predictions available - Einstein only allows the end user access to "100 randomly sampled observations" - and answer the first research question: **"To what extent do Salesforce Einstein's models exhibit bias in their outcomes and predictions when potential bias variables are not explicitly flagged?"**.

From the **XG Boost model** prediction analysis- in which there were no bias-related alerts - we got the following conclusions:

- For the "**Male**" category on the "sex" variable, the actual values for "income" were more balanced (i.e. there was an adequate distribution for "<=50K" and ">50K" values);
- For the "**Female**" category on the "sex" variable, the actual values for "income" were skewed towards the "<=50K" value;
- The main cause for Einstein to wrongly predict an income for a Male (specifically a lower income) was the individual's education level. Observations with an education below the bachelor level would get a predicted income of "<=50K";
- The main causes for Einstein to wrongly predict an income for a Female (specifically a lower income) were the individual's education level and the fact that there were more "Female" observations in which the actual income was "<=50K". Consequently, there were no observations - from the 100 samples available for analysis - in which Einstein predicted an income greater than 50,000 dollars;

- "Race" and Ethnicity (i.e. "Native-Country") did not seem to unfairly interfere with predictions;
- "Sex" did have an impact on the predicted values. This impact was not based on bias or discrimination towards a certain group but rather on the actual income values from our supervised training data.

Additionally, for the Random Forest analysis, we concluded that - as in XG Boost - the education level was the greatest "influencer" when selecting between "<=50K" and ">50K" for predicted income. Furthermore, there was a bigger number of wrong predictions since education was a target for bias-biased behaviour, with the [12-16] age group being the only one where the selection rate was greater than 80%.

Moreover, in the course of our numerous experiments, we identified some recurring patterns in our results:

1. Disparate Impact alerts and suggested improvements are almost mutually exclusive alerts. When we got a disparate impact alert, there were no suggested improvements;
2. For the "Flagging all variables for Bias" experiment, some of the variables identified for disparate impact were not sensitive, leading us to believe that Einstein does not have a mechanism in place to distinguish sensitive from non-sensitive variables;
3. XG Boost was the better performing algorithm on all experiments, but the value difference for metrics was not of great significance.

## 8.2 Overall Einstein Discovery Audit Results

Throughout this dissertation, we performed many experiments, thoroughly testing the "Analyse for Bias" feature and the extent of its capabilities. At its core, our work revolved around auditing the tool - Einstein - and its intrinsic fairness mechanisms and metrics. In alignment with our second research question, **"How reliable is the "Analyse for Bias" feature in Salesforce Einstein, and to what degree does it effectively mitigate discriminatory outcomes in the models it assesses?"**, on a general overview of Einstein and the Salesforce platform, "Analyse for Bias" is a minor feature - a single checkbox on each variable's "settings" tab, available when manually configuring the dataset for model training. Furthermore, after leveraging the feature, it was clear that the diagnostic aspect of it relied on the "Four-Fifths" rule, applying it to every variable flagged for bias analysis instead of being capable of distinguishing sensitive from non-sensitive variables and applying the rule to the appropriate ones. At times, it came across as an afterthought feature, a small addition to the platform to adhere to trending topics and conversations about ethics in AI. Moreover, automated models do not consider this feature (i.e. flagging variables for bias analysis) even when a dataset might contain definition-sensitive variables. Training a new model version with manually flagged variables is always necessary.

Looking through an end user's perspective, Einstein Discovery has a user-friendly UI focused on clicks, not code. The interface is interactive and intuitive, which makes it relatively easy to use and understand. Several insight possibilities exist, meaning users can draw many conclusions from each use case. However, the Discovery module is highly oriented towards business scenarios where data is already inserted in objects and fields inside the Salesforce platform (CRM). This became clear when performing the configurations for model training - highlighting the "So I Can" selection, where a user needs to select if they either want to maximise or minimise the target variable (e.g. maximise lead conversion) - where we had to choose between maximising or minimising our target variable, "predicted_income" when in reality our only goal was to obtain the correct predicted values for each observation.

As we understood, working with external data proved to be a more significant challenge, specifically for the "Upload Model" experiment. Not having any connection to org data not only hurt the promised easy-to-use, user-friendly interface but also might have affected results since Einstein had no additional knowledge of the data besides the dataset - with intrinsic Salesforce data, the AI would be aware of the data itself and its relationships to other objects or fields.

Finally, to conclude our audit analysis and answer our third and final research question, **"What additional features and capabilities does Salesforce Einstein possess to uphold ethical practices in AI development and deployment beyond the "Analyse for Bias" functionality?"**, we did not find any additional features or mechanisms related to bias analysis and mitigation, having only the "Analyse for Bias" feature and its related "Disparate Impact alerts". From what we understood, Einstein expects the data to be already combed through for different types of biases (described in Chapter 2) before entering the platform and model training.

Overall, Einstein Discovery is not yet robust enough to handle external data and sensitive data detection regarding bias mitigation mechanisms.

# References


D Adebanjo. E-crm implementation—a comparison of three approaches. In *2008 4th IEEE International Conference on Management of Innovation and Technology*, pages 457–462. IEEE, 2008.

Farnaz Arab, Harihodin Selamat, and Mazdak Zamani. An overview of success factors for crm. In *2010 2nd IEEE International Conference on Information and Financial Engineering*, pages 702–705. IEEE, 2010.

Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. Ai fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias, 2018.

Timothy Bohling, Douglas Bowman, Steve LaValle, Vikas Mittal, Das Narayandas, Girish Ramani, and Rajan Varadarajan. Crm implementation: Effectiveness issues and insights. *Journal of Service research*, 9(2):184–194, 2006.

Jason Brownlee. 4 types of classification tasks in machine learning, Aug 2020a. URL https://machinelearningmastery.com/types-of-classification-in-machine-learning/.

Jason Brownlee. Probabilistic model selection with aic, bic, and mdl, Aug 2020b. URL https://machinelearningmastery.com/probabilistic-model-selection-measures/.

Chris DeBrusk. The risk of machine-learning bias (and how to prevent it). *MIT Sloan Management Review*, 15:1, 2018.

Andrea D'Agostino. The explanation you need on binary classification metrics, Aug 2022. URL https://towardsdatascience.com/the-explanation-you-need-on-binary-classification-metrics-321d280b590f.

Fairlearn. Common fairness metrics, 2018. URL https://fairlearn.org/main/user_guide/assessment/common_fairness_metrics.html#the-four-fifths-rule-often-misapplied.

GeeksforGeeks. F1 score in machine learning, Dec 2023. URL https://www.geeksforgeeks.org/f1-score-in-machine-learning/.

Google. Classification: Accuracy | machine learning | google for developers, 2024a. URL https://developers.google.com/machine-learning/crash-course/classification/accuracy?hl=en.

Google. Auc | machine learning | google for developers, 2024b. URL https://developers.google.com/machine-learning/glossary?hl=en#AUC.

Google. Classification: Precision and recall | machine learning | google for developers, 2024c. URL https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall?hl=en.

Salesforce Help. Prepare the upload file for the machine learning model, 2024. URL https://help.salesforce.com/s/articleView?id=sf.bi_edd_model_upload_prepare_file.htm&type=5.

Help.Salesforce. Disparate impact alert, 2023. URL https://help.salesforce.com/s/articleView?id=sf.bi_edd_data_alert_disparate_impact.htm&type=5.

IBM. What is random forest?, 2024. URL https://www.ibm.com/topics/random-forest.

Brittany Johnson, Jesse Bartola, Rico Angell, Sam Witty, Stephen Giguere, and Yuriy Brun. Fairkit, fairkit, on the wall, who's the fairest of them all? supporting fairness-related decision-making, Mar 2023. URL https://www.sciencedirect.com/science/article/pii/S2193943823000043.

Chet Lemon, Chris Zelazo, and Kesav Mulakaluri. Predicting if income exceeds 50000 per year based on 1994 US Census Data with Simple Classification Techniques, 2024.

Microsoft. Microsoft responsible ai toolbox, Mar 2022. URL https://responsibleaitoolbox.ai/.

Christoph Molnar. Interpretable machine learning, Aug 2023. URL https://christophm.github.io/interpretable-ml-book/index.html.

Bang Nguyen and Dilip S Mutum. A review of customer relationship management: successes, advances, pitfalls and futures. *Business Process Management Journal*, 18(3):400–419, 2012.

Nvidia. What is xgboost?, 2024. URL https://www.nvidia.com/en-us/glossary/xgboost/.

Michael Proksch. From glm to gbm, Jun 2020. URL https://towardsdatascience.com/from-glm-to-gbm-5ff7dbdd7e2f.

Stacey Ronaghan. Ai fairness - explanation of disparate impact remover, Apr 2019. URL https://towardsdatascience.com/ai-fairness-explanation-of-disparate-impact-remover-ce0da59451f1.

Pedro Saleiro, Benedict Kuester, Loren Hinkson, Jesse London, Abby Stevens, Ari Anisfeld, Kit T Rodolfa, and Rayid Ghani. Aequitas: A bias and fairness audit toolkit. *arXiv preprint arXiv:1811.05577*, 2018.

Salesforce. Why salesforce platform, 2023a. URL https://www.salesforce.com/eu/products/what-is-salesforce/.

Salesforce. What is chatter?, 2023b. URL https://www.salesforce.com/products/chatter/overview/.

Salesforce. Salesforce einstein: Pricing & readiness guide. *https://www.salesforce.com/content/dam/web/en$_u$s/www/documents/datasheets/einstein − features − cheat − sheet.pdf*, 2023c.

Salesforce. Salesforce artificial intelligence. *https://www.salesforce.com/products/einstein-ai-solutions/*, 2023d.

Salesforce. Help and training community, 2024a. URL https://help.salesforce.com/s/articleView?id=sf.bi_edd_glossary.htm&type=5.

Salesforce. Einstein discovery: Quick look, 2024b. URL https://trailhead.salesforce.com/content/learn/modules/einstein-discovery-quick-look/meet-einstein-discovery?trail_id=wave_analytics_einstein_discovery.

Salesforce. Help and training community, 2024c. URL https://help.salesforce.com/s/articleView?id=sf.bi_edd_intro.htm&type=5.

Salesforce. Ethical model development with einstein discovery: Quick look, 2024d. URL https://trailhead.salesforce.com/content/learn/modules/ethical-model-development-in-einstein-discovery-quick-look.

Salesforce. Responsible creation of artificial intelligence, 2024e. URL https://trailhead.salesforce.com/content/learn/modules/responsible-creation-of-artificial-intelligence/recognize-bias-in-ai.

SalesforceHelp. Bring your scikit-learn model to salesforce, 2024. URL https://help.salesforce.com/s/articleView?id=release-notes.rn_bi_edd_byom_sciKitLearn.htm&release=240&type=5.

Marco Scutari. fairml: A statistician's take on fair machine learning modelling, 2023.

MS Sneh et al. Analysis of business strategies of salesforce. com inc. *Sneha, MS & Krishna Prasad, K.(2018). Analysis of Business Strategies of Salesforce. com Inc. International Journal of Case Studies in Business, IT and Education (IJCSBE)*, 2(1):37–44, 2018.

Saideep Sunkari. A brief review on crm, salesforce and reasons stating salesforce as one of the top crm's. *Salesforce and Reasons Stating Salesforce as One of the Top CRM's (June 18, 2022)*, 2022.

Makrand Thakkar and Rajesh Rajaan. Salesforce crm: A new way of managing customer relationship in cloud environment. *International Journal of Electrical, Electronics and Computers*, 5(3):14–17, 2020.

Trailhead. Salesforce einstein basics, 2024. URL https://trailhead.salesforce.com/content/learn/modules/get_smart_einstein_feat/get_smart_einstein_feat_basics.

UN. Transforming our world: The 2030 agenda for sustainable development | department of economic and social affairs, 2015a. URL https://sdgs.un.org/2030agenda.

UN. The 17 goals | sustainable development, 2015b. URL https://sdgs.un.org/goals.

UN. Goal 10 | department of economic and social affairs, 2015c. URL https://sdgs.un.org/goals/goal10#overview.

UN. Goal 5 | department of economic and social affairs, 2015d. URL https://sdgs.un.org/goals/goal5.

xgboost developers. Xgboost documentation, 2022. URL https://xgboost.readthedocs.io/en/stable/.

Xiaoyu Yu, Bang Nguyen, Sung Ho Han, Cheng-Hao Steve Chen, and Fei Li. Electronic crm and perceptions of unfairness. *Information Technology and Management*, 16:351–362, 2015.

Ghasem Zaefarian, Vita Kadile, Stephan C Henneberg, and Alexander Leischnig. Endogeneity bias in marketing research: Problem, causes and remedies. *Industrial Marketing Management*, 65:39–46, 2017.