POLITECNICO DI MILANO
Master of Science in Computer Science and Engineering
Dipartimento di Elettronica, Informazione e Bioingegneria

[Title of the Thesis]

Internal supervisor: Prof. Viola Schiaffonati
Internal co-supervisor: Prof. Letizia Tanca
External supervisor: Prof. Pierre Senellart
External co-supervisor: Prof. Karine Gentelet

M.Sc. Thesis by:
Riccardo Corona, 927975

Academic Year 2020-2021

About this template

With this template I want to give you some input on how to structure your thesis if you develop your thesis with me in Politecnico di Milano. Next to the pure structure, which you should reuse and adapt to your own needs, the document also contains instructions on how to approach the different sections, the writing and, sometimes, even the work on your thesis project itself. Sometimes you will also find boxes like this one. These are meant to provide you with explanations and insights or hints that go beyond the mere structure of a thesis.

I hope this template will help you do the best thesis ever, if not in the World, at least in your life.

Florian Daniel
October 12, 2017

Disclaimer: Sometimes I may make statements that are general, if not over-generalized, personal considerations, or give hints on how to do work or research. Be aware that these are just my own opinions and by no way represent official statements by Politecnico di Milano or its community of professors. If something goes wrong with your thesis or presentation, you cannot refer to these statements as a defense. You are the final responsible of what goes into your thesis and what not.

Acknowledgements: The original template for this document was not created by me. I would love to acknowledge the real creator, but I actually do not know who it is. The template has been passed on to me by a former student, who also didn't know the exact origin of it. It was circulating among students. However, to the best of my knowledge at the time of writing, it seems that Marco D. Santambrogio and Matto Matteucci may have contributed at some point with considerations on structure and funny citations. Both were helpful and enjoyable when preparing this version of the template. I will be glad to add more precise acknowledgements if properly informed about the origins of this template.

Supervisors and co-supervisors

If the supervisor is internal to Politecnico di Milano (a professor or researcher), then on the first page use "Supervisor" plus the titles "Prof." and "Dr." for professors and researches, respectively. If the work was co-supervised by someone else, refer to him/her as the "Co-supervisor." If the work was supervised by someone external to Politecnico di Milano, use "External supervisor" for the external supervisor plus "Internal supervisor" for the internal supervisor that mandatorily must co-supervise the work with the external supervisor.

Optionally, here goes the dedication.

Abstract

The abstract is a small summary of the thesis. It tells the reader in few words (up to one/one and a half page of total text) everything he/she needs to understand:

- ☐ the *context* of the work (e.g., chatbots),
- ☐ the specific *problem* approached by the thesis (e.g., the development of personal bots by non-programmers),
- ☐ if applicable, clearly state the *research questions* you would like to answer (e.g., "is it possible to enable non-programmers to do X using A?"),
- ☐ the three/four *core aspects of the proposed solution* (e.g., use pre-defined rules, use machine learning, assisted development, etc.),
- ☐ the *concrete outputs* produced by the thesis (e.g., a state of the art analysis, a conceptual/mathematical model, an application, middleware or API, an empirical study with/without users, etc.), and
- ☐ the *findings and conclusions* that one can draw from the evaluation of the approach (e.g., that under some very specific conditions non-programmers are indeed able to implement own chatbots effectively using the proposed technique).

Checklists
Now and there I propose checklists with items, such as the one just above this box. They are meant for you to check if you included all the content that is relevant and that should be included, in order to make your text complete. When reading your thesis, I will look for all these items.

Writing style

This is a M.Sc. thesis. It's neither Facebook nor Twitter nor an email. This is going to be an official document with legal value that will decide on the final mark of your yearlong university career and perhaps even on your future work perspectives. So, you surely don't want to be judged badly because of grammar errors, flawed/wrong vocabulary or superficial layout and/or text structure. It is a must that what you write is always *correct* content- and language-wise (no false statements or claims, no language mistakes), *readable* (no sentences that cannot be understood) and targeted at the *average-skilled reader* (professors, but also your own colleagues).

Plagiarism

This is a M.Sc. thesis. It's neither Facebook nor Twitter nor an email. This is going to be an official document with legal value that will decide on the final mark of your yearlong university career and perhaps even on your future work perspectives – yes, I plagiarized myself here a little bit. So, you surely don't want to copy/paste material from scientific articles, online resources, books, and similar without adequately acknowledging the holders of the respective intellectual property rights. If you do so, it is a must that you properly *cite* each source where you take text or inspiration from. It is fine to do so – actually, citing someone is a compliment! – but it becomes a crime if the source is not cited. Not only M.Sc. titles but also Ph.D. titles have been withdrawn for fraudulent "reuse" of others' intellectual property. Be aware that Politecnico di Milano, like most higher educational institutions that issue university degrees or scientific publishers, may use specialized software to automatically detect plagiarism.

Sommario

Here goes the translation into Italian of the abstract. If the thesis is written in Italian, no translation into English is needed. Hence, one of the following must be checked:

☐ Thesis written in *English*, properly proofread translation needed

☐ Thesis written in *Italian*, no translation needed, chapter omitted

Acknowledgements

If you would like to thank somebody for given support, this is the right place to do so.

Contents

Chapter 1

Introduction

The introduction is one of the core chapters of your thesis. It expands what has already been said in the abstract with additional details on the content and contribution and on the structure of the thesis. It is meant to introduce the reader to the work he/she will be reading in the rest of the document and, most importantly, to get the reader curious about reading on, knowing more about your work.

1.1 Context: [topic]

This thesis is about describing the work you are doing in your final thesis project. You have been working on it for months, and nobody knows the work better than you do. This is great and exactly how things should be: by doing your thesis project you became an expert – if not *the* expert – in this specific field of research and/or technology.

But attention: being the expert is also dangerous when it comes to explaining others what you did and why you think you did a great work that deserves attention (I give it for granted that you work does so). There are only very few people around you (your supervisor and possible co-supervisor, some friends, maybe someone else) who are as expert as you are in this topic. So, if you start in a full-impact fashion to tell that you implemented an extraordinarily cool, new algorithm to solve X, or that you discovered this extremely surprising finding Y, or that you mathematically proofed that Z, etc. (you got it), your reader will not understand anything. Therefore, before talking about what you actually did, you need to introduce the reader to the context of your work, provide the necessary core definitions that are needed to understand the terminology you will be using in the rest of the thesis (if it's not standard IT terminology).

Therefore:

- ☐ Tell the *research area(s)* your work/project focuses on. If you are doing your thesis with me, likely candidates of research areas are Web Engineering, Data Science, Crowdsourcing, Service-Oriented Computing, Business Process Management.
- ☐ Tell possible *sub-areas* that are more specifically related to what you are doing. Again, if you are doing your thesis with me, likely candidates of sub-areas are chatbots, social knowledge extraction, business process matching/modeling, quality control in crowdsourcing, etc.
- ☐ Make the *heading* of your context section self-explaining by substituting "[topic]" in heading 1.1 with the sub-area most relevant to your work. It should read like "Context: quality control in crowdsourcing" or similar.
- ☐ If needed, introduce some *key definitions* (no need to introduce everything here, but be sure that the introduction does not use terminology the reader may not be familiar with). For instance, if you are working on chatbots, this is definitely a term that needs to be introduced here; it's not yet commonly known but it's crucial for the understanding of the rest of the thesis and introduction.
- ☐ Use *examples* to make definitions and ideas concrete and clear.
- ☐ Throughout, make *references* to the relevant literature.

Use of tenses and pronouns

Writing a thesis is writing a scientific document like scientific articles or research publications. There are two conventions that are usually applied in this kind of publications (admittedly, they may seem somewhat odd if not used to):

First, the most used tense is the *simple present*. The thesis is meant to describe a piece of work, from problem statement, to the conception of a solution, its implementation and evaluation. Yet, it's not a novel about your life, and it's not meant to provide a chronological story about what you did and didn't do. Content is presented in an order that is most effective to convey its message, not in time order. In this spirit, it's much more effective to say "in order to get result A, first we do X, then we do Y and then Z," instead of saying "in order to get result A, we did Y after having done X, then we went on doing Z." The order of actions, their interconnections, inputs and outputs already tell the dependency – if properly described. Most of the times, the most effective way to describe a solution or methodology only becomes clear after trial and error. It's enough to explain the result, not how you got there chronologicallly.

Second, the *pronoun* used to talk about the own work is "our" (work). That is, it is custom to say "we" instead of "I," even if you are writing your thesis alone. However, don't forget about all the people that helped you get there: your supervisor, co-supervisor, colleagues, etc. This may sound strange at the beginning, but, at the other hand, using "I" too often risks to convey the impression that you are self-focused and egoistic, which is never good.

1.2 Scenario and Problem Statement

Now that the reader got the general context of your work and has an intuition of the problem you will be solving in the rest of the thesis, it's time to be clear about which *specific problems* your thesis project is going to solve. One way of doing so is by describing a *scenario* (a description of a real situation, with all its actors, roles, tasks, instruments, etc.) that provides evidence that there are one or more real problems right now that, with the current technology and understanding of the domain, are hard to solve or not solvable at all. If instead the problem(s) can be solved already, it should be evident from the scenario that this is possible only at a prohibiting cost or with unsatisfying guarantees on the quality of the result or not within useful time for the target user.

It's important that the scenario is written in such a way that the reader, after reading it, agrees with you that the problem you are focusing on is a relevant one, one that deserves being studied and solved. Consider that if you convince the reader here that your thesis is needed (after all, that's what this section is about), he/she will be very open to possible solutions and happy to see how you solve it. If instead you fail to convince the reader – let me be harsh – the whole rest of your thesis is useless in the eyes of the reader. This is the worst outcome you want.

Conclude this section by explicitly stating which of the problems evident in the scenario you are approaching. Don't raise false expectations! Never ever tell the reader there are five core problems and then solve only two of them in the thesis, without telling upfront that this is what you intended to do in the first place. As soon as you list problems, the reader wants to see a solution, unless you stop him/her immediately from thinking so by telling that out of the described problems you focus on a subset only, usually because this subset is already a huge research and development problem in its own.

In summary:

- ☐ Describe a *real scenario* that provides evidence of *real problems*.
- ☐ Convince the *reader* that the problems need to be solved.
- ☐ Use an *illustration* or *figure* to help the reader understand.
- ☐ If possible, provide *references* to literature that backs your assessment of the problem.
- ☐ Provide a clear *problem statement* that summarizes what came out of the scenario and your specific focus.

1.3 Methodology

Fixed the problem(s) you want to approach, you can approach it/them in thousands of different ways. Your way is just one of the thousands, and the reader may have (and very likely will have) a very different intuition of how to solve the problem(s) you just pointed out. So, clarify how you intend to proceed:

- ☐ Tell if you follow an existing *methodology* or not; if yes, name it and provide a reference to literature, if available. For example, Design Science [**?**] is a likely methodology to cite here.
- ☐ Tell which of the following *procedures*, *techniques*, *methods* you use in your work and for which purpose (put them also into the right order, so that their application or use makes immediate sense to the reader):
  - ☐ *Systematic literature review, survey*
  - ☐ *Statistical hypothesis formulation and testing*
  - ☐ *Software prototyping*
  - ☐ *Iterative development*
  - ☐ *Participatory design*
  - ☐ *Performance evaluation*
  - ☐ *Comparative studies*
  - ☐ *User studies*
  - ☐ *Expert interviews*
  - ☐ *Simulation/emulation*
  - ☐ *Live experiments*
  - ☐ *Case studies*

- ☐ *Mathematical theorem proofing*
- ☐ *Mathematical modeling*
- ☐ *Pseudocode*
- ☐ *Graphical modeling* (e.g., UML, ER)
- ☐ *Model-driven development*
- ☐ *Automatic code generation*
- ☐ ...

☐ Tell if you use some special *software instruments* that help you in your work. We are of course not talking about Word or Google Search. Perhaps you can tell that you used R for data analysis or some specific modeling instrument for automated code generation or simulation.

1.4 Contributions

Now that the reader knows what you want to solve and how you intend to proceed, you can anticipate the contributions your thesis makes to the state of the art. Attention, a thesis project may produce lots of different *outputs* (e.g., a software prototype, a set of registrations and transcripts of interviews, datasets collected during experiments) and *contributions* (e.g., a demonstration that some software solutions solves a given problem under well defined conditions, a formal proof that some property holds, empirical evidence that something works as expected). The former are all the artifacts produced throughout the work. The latter refer to *new knowledge* (if you are doing a full thesis) or the most important, *final output* (if you are doing a tesina). Sometimes, outputs and contributions overlap, but not necessarily.

Typical contributions are (multiple choices may apply to your thesis):

☐ A *systematic literature review* of the state of the art providing evidence for some argument

☐ The design of a *model* (mathematical, graphical, algebraic, etc.) describing how to solve a real world problem in a reusable fashion

☐ The drawing of *conclusions* (findings) from the analysis of a dataset describing some physical or virtual phenomenon

☐ The implementation of a *software prototype* solving a real world application problem

☐ The design of a *language* (textual, graphical) enabling others to solve own problems or to solve them easier

☐ *Formal proofs* of correctness, completeness or other properties of the proposed models or theorems

☐ *Objective evidence* from empirical studies (e.g., performance analyses or simluations) that demonstrate that the proposed prototype or solution works / works better than existing software or solutions that solve the same/similar problem(s)

☐ *Subjective evidence* from user studies or expert interviews backing the claims of viability of the proposed problem or solution/artifact

☐ A reasoned *argumentation*, e.g., based on a detailed case study, supporting the viability of the proposed problem or solution/artifact

Thesis vs. Tesina

Let me spend some words on the difference between these two. Before that, however, it is important to clarify the very purpose of your final project, be it a thesis or a tesina (a small thesis). The purpose of it is giving you the possibility to show that, after years of attending classes and giving exams, you are also able to *apply* the knowledge you acquired during your studies. In short, it's all about you showing that you are *mature*. Mature form a knowledge perspective, mature from an application perspective, mature from a work/teamwork perspective, mature from an ethical perspective.

It is common that a thesis project is not very well defined in its beginning and that even the supervisor does not really know how to approach a given problem or which problem to focus on in the first place. This may even be annoying to you, but attention: there is no intention behind it. Your supervisor is not withholding information from you to test you or to see if you get something. It's just the nature of real *problem solving*. If things were clear from the beginning, there wouldn't be any problem! Fledging out the problem and agreeing on a solution and methodology is a core part of you demonstrating your maturity – if not the most important one. *How* you proceed from the inception of the thesis idea to the final solution is as important as *what* you find and/or produce in the end.

This being said, a *thesis* in Politecnico di Milano usually requires you to make a contribution to the literature (the so-called state of the art). Making a contribution – from a science point of view – means creating new *knowledge*, that is, finding something that nobody knew before, demonstrating a property that nobody showed before, improving the performance of a given system with a new algorithm, and similar. For a thesis, it is therefore not enough to produce a perfectly engineered solution. It is key that you also demonstrate, provide empirical evidence or proof

that your solutions performs as claimed. Well, for a *tesina* this last demonstration is usually not required, and the focus is on the engineering of the solution. In addition, perhaps in the case of the tesina the solution to be engineered is also less complex then for a thesis, but this depends on the context and on how you want to measure complexity.

1.5 Structure of Thesis

Here you explain the structure of the thesis, so that the reader knows how to read it. Consider that not every reader wants to read through the whole thesis to find some specific information. Actually, only few will do so (your supervisor and co-supervisor, and the possible reviewer for sure). Many more will just leaf through it and look for specific types of information (e.g., the context of your work, your findings, how you implemented something, which technologies you used). It is your duty to accommodate them all. How? By telling them how your thesis is structured.

Therefore, in this section you provide a brief description (2-3 sentences) for *each* chapter that follows this introduction. Use an itemized or numbered list to structure the text, like this:

- ☐ Chapter 2 introduces the state of the art and...
- ☐ Chapter 3 provides...
- ☐ ...

Structuring text
Besides telling the reader how the content of your thesis is organized into chapters, it is important that you master some basic text structuring techniques. To organize your text there are lots of instruments you can use: chapters, sections, sub-sections, paragraphs, itemized lists, numbered lists, code examples, figures, images, screen shots, captions below figures, tables, and so on. Use them all! Don't write text without structure. Never.

Be aware that the structure of your text, that is, how you present your work, conveys a lot of information about how well you actually understand what you are writing about, how much you care about being clear and helping your reader understand, and how much value you give yourself to your own thesis. A well structured presentation of content that the reader can understand and agree with is a huge plus in this respect. Text that lacks proper paragraphs, does not use lists where needed, etc. is a minus and also much harder to read (think about how

much a well structured text can help you go back ten pages and find concepts you know you read about compared to a text that comes without an easy to memorize formatting and structure). When writing, think about some of your textbooks. Since you are doing an engineering degree, I'm sure these are textbooks that make exemplary use of the different formatting instruments available.

Chapter 2

Socio-Ethical Preliminaries

The aim of this chapter is to provide to the reader preliminary notions of ethical and sociological rather than technical nature.

Starting from the concept of *bias*, passing through *discrimination* and *human rights*, we will discuss about *equality*, *equity* and finally *fairness*, by providing definitions and relevant examples from the literature on these topics, with a focus on the data and computer systems perspective. It is important to emphasize that, despite the exhaustiveness of the definitions, these terms often have different meanings depending of the context of use, and there is a lot of debate on how to interpret them and eventually include all their dimensions in computer systems.

Because of the "dual nature" of this research, this chapter has to be seen as complementary to Chapter 3, in which some technical bases will be provided, together with an overview on the tools adopted.

2.1 Bias

Although the word "**bias**" does not have an intrinsically negative meaning (it is informally used to indicate a deviation from neutrality), it is mostly adopted in contexts where it entails a moral and social dimension. As reported in [19]:

> *We use the term bias to refer to computer systems that* systematically *and* unfairly discriminate *against certain individuals or groups of individuals in favor of others. A system discriminates unfairly if it denies an opportunity or a good or if it assigns an undesirable outcome to an individual or group of individuals on grounds that are unreasonable or inappropriate.* [19, p. 332]

Therefore, it is important to underline that unfair discrimination due to bias is strictly related to systematic and unfair outcome, where the word "systematic" is used with the meaning of "regular, which occurs methodically when certain conditions arise".

By following the classification provided in [19], we can distinguish three overarching categories of bias:

- **Preexisting bias**: it has its roots in social institutions, practices and attitudes. Preexisting bias may originate in the society at large or in subcultures and organizations (*societal bias*), but it is also intrinsic in the nature of every human being (*individual bias*), and can enter a computer system either voluntarily or implicitly and unconsciously, even in spite of the best intentions of the system designer. Furthermore, since preexisting bias is often related to historical discrimination of disadvantaged groups, it could lead to the introduction or the exacerbation of representation issues in the data. An example of preexisting (gender) bias is the one present in the society that leads to the development of educational software that overall appeals more to boys than girls [19].

- **Technical bias**: it arises from the resolution of issues in the technical design. Technical bias may originate from design choices, constraints and technological tools, or exacerbate preexisting bias. An example of technical bias, due to technical constraints, is the one of a monitor screen displaying the flight options most relevant to an airline customer: the screen dimension forces a piecemeal representation of the flights and therefore if the ranking algorithm systematically places certain flights on initial screens and other flights on later screens, it exhibits technical bias [19].

- **Emerging bias**: it emerges some time after a design is completed, as a result of changing societal knowledge, population, or cultural values. Emerging bias is strictly related to the specific context of use, and it is the most difficult to detect. An example of emerging bias, caused by a *mismatch between users and system design* due to *different values* (that is, originated when a computer system is used by a population with different values than those assumed in the design), is the one of an educational software embedded in a game situation that rewards individualistic and competitive strategies used by students with a cultural background that eschews competition and promotes collaboration [19].

For the purpose of this research, we will focus on preexisting bias (in

particular, societal bias) and technical bias, but it is important to point out that emerging bias should not be underestimated in the long run, especially when it arises from a mismatch between users and system design due to different values, because society is in constant change and systems should be readjusted or reinvented in order to keep up with the present.

A significant example of preexisting (racial) bias, exacerbated by technical bias, is provided in [3]: a commercial tool called COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) was used in courts in the U.S. to automatically predict some categories of future crime to assist in bail and sentencing decisions. On average, the tool correctly predicted recidivism 61% of the time, but blacks were almost twice as likely as whites to be labeled a higher risk but not actually re-offend. The tool made the opposite mistake among whites: they were much more likely than blacks to be labeled lower risk but go on to commit other crimes.

Other examples, related to gender bias and technology, are given by [21] and [13]. The former is about a research conducted in 2015, in which a tool was used to simulate job seekers that did not differ in browsing behavior, preferences or demographic characteristics, except in gender. One experiment showed that Google displayed adverts for a career coaching service for "$200K+" executive jobs 1,852 times to the male group and only 318 times to the female group. The latter concerns another Big Tech company, Amazon, whose machine learning specialists, back in 2015, discovered that their new recruiting engine was not rating candidates in a gender-neutral way, because the system taught itself that male candidates were preferable by penalizing resumes that included the word "women's".

2.2 Discrimination

Bias can lead to **discrimination**, but what discrimination is and how it occurs is a controversial issue. As reported in [31], often the law, rather than providing a definition of discrimination, defines a list of attributes, called **protected attributes**, that cannot be used to take decisions in various settings. The list is non-exhaustive and includes characteristics such as race, gender, religion, or sexual orientation. Groups of people that are more likely to be discriminated against because of these attributes are therefore classified as *protected groups*.

Trying to elaborate a bit more, we can define discrimination as the result of either one or both the following:

- **Disparate treatment**, or *intentional discrimination*: the illegal prac-

tice of treating an entity, such as a job applicant, differently based on a protected attribute such as race, gender, age, religion, sexual orientation or national origin because of a discriminatory motive.

- **Disparate impact**, or *unintentional discrimination*: the result of structural disparate treatment, in which policies, practices, rules or other systems that appear to be neutral result in a disproportionate adverse impact on a protected group. Disparate impact is not based on a discriminatory motive and the discriminating agent is usually unaware of the discrimination.

Protected attributes are mentioned in the article 2 of the Universal Declaration of Human Rights (UDHR), which states:

> *Everyone is entitled to all the rights and freedoms set forth in this Declaration, without distinction of any kind, such as race, colour, sex, language, religion, political or other opinion, national or social origin, property, birth or other status.* [4]

Discrimination is therefore strictly related to *human rights*, together with the concept of *equality*, since "equal", "equally" and "equality" itself are recurring words in several articles of the Declaration.

2.3 Human Rights

For what concerns **human rights**, there is a lot of debate on how to incorporate them in computer systems by following a "human-rights-by-design" approach [28], in order to contrast the negative effects of the so called "dual-use technologies": products which may serve legitimate societal objectives but are also used to undermine human rights like freedom of expression or privacy. Of course, reaching this goal would require a culture shift and huge efforts from both national governments and businesses, which should design tools, technologies, and services to respect human rights by default, rather than permit abuse or exploitation of them. A similar concept is proposed in [37], where the authors sketch the contours of a comprehensive governance framework for ensuring AI systems to be ethical in their design, development and deployment, and not violate human rights. This framework should be effective in contrasting *ethics washing*: the practice of fabricating or exaggerating a company's interest in equitable AI systems that work for everyone, a sort of side door that companies use to substitute regulation with ethics.

For the purpose of this research, we can define human rights as "inalienable fundamental rights to which a person is inherently entitled simply because she or he is a human being" [32, p. 3]. A few examples are the rights to life and liberty, freedom from slavery and torture, freedom of opinion and expression, the rights to work and education, and the right to the pursuit of happiness. These norms are concerning every human being, regardless of sex, age, language, religion, ethnicity, or any other status.

2.4 Equality & Equity

Equality is generally intended as "an ideal of uniformity in treatment or status by those in a position to affect either" [7]. The concept of equality is often associated with discrimination mostly because of the article 7 of the UDHR, which states:

> *All are equal before the law and are entitled without any discrimination to equal protection of the law.* [4]

This principle is known as "equality before the law", and establishes that everyone must be treated equally under the law regardless of race, gender, color, ethnicity, religion, disability, or other characteristics, without privilege, discrimination or bias.

However, it is important to distinguish between two different political and social theories:

- **Equality of opportunity**:

  > *The idea that people ought to be able to compete on equal terms, or on a "level playing field", for advantaged offices and positions.* [26]

  This principle is based on the notion of *sameness*, where fairness is achieved through equal treatment regardless of people's needs. Equality of opportunity is usually simply referred as **equality**, and from now on we will adopt the same terminology for this research.

- **Equality of outcome**: the idea that people should have access to resources (possibly of a different nature and to a different extent) in order to be able to reach the same condition. This principle is based on the notion of *need*, where fairness is achieved by treating people differently depending on their endowments and necessities. Equality of outcome is also known as **equity**, and from now on we will adopt the same terminology for this research.

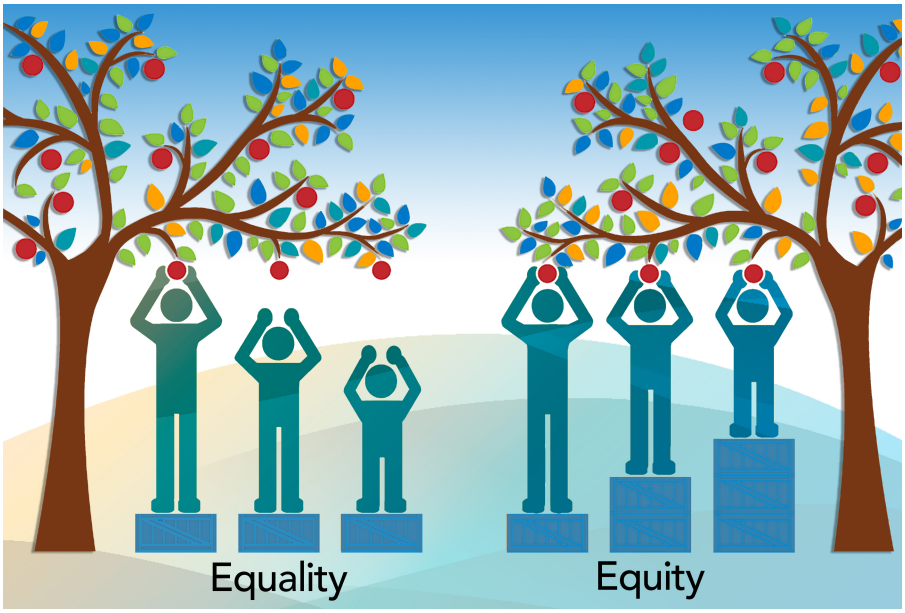


Figure 2.1: Visual example of the difference between equality and equity. ©2014, Saskatoon Health Region. Source: `https://www.nwhu.on.ca/ourservices/Pages/Equity-vs-Equality.aspx`.

Figure 2.1 provides a simple example of the difference between equality and equity. Treating people equally, in this scenario, means to give everyone the same one box to reach the fruit, while treating people equitably means to give them as many boxes as they need to achieve the goal. It is important to notice that equity could require (and often requires) unequal treatment.

Moving on to the data perspective, we can now extend (or restrict, depending on the point of view) the equality and equity concepts to data equality and data equity. Data equality usually refers to transparency of institutions and companies towards customers regarding the information collected on their account, whereas data equity is used in a different context. The authors of [24] distinguish between four different facets of data equity:

- **Representation equity**: bias may arise because of material deviations between the data and the world represented by the data, often with respect to historically disadvantaged and underrepresented groups. Even when dealing with contemporary data, disparities rooted in historical discrimination can lead to representation inequities and therefore to the introduction or the exacerbation of problems. For example, in the U.S. there has been a lot of discussion about racial disparities concerning COVID-19, regarding both availability of testing

(fewer test sites in minorities neighborhoods, historically poorer) and desire of individuals to be tested (black people more suspicious about the medical system, because of their history of unfair treatments) [24]. Another example, already mentioned in Section 2.1, is related to Amazon and a software developed by the company for screening candidates for employment: the software was trained on the already hired employees and since they were mostly males, females became underrepresented in the data and the software was much more likely to mark women as unsuitable for hiring.

- **Feature equity**: bias may arise because not all the features needed to represent a marginalized group of people and required for a particular analysis are available in the data, or because some of these features are voluntarily removed in the decision-making process. As an example, for a specific study involving transgender people, it may be important to distinguish between their birth name and their self-assigned name.

- **Access equity**: bias may arise because of a non-equitable and participatory access to data and data products across domains and levels of expertise due for instance to the opacity of data systems or the need to respect the privacy of data subjects. A classical example is the one of medical records: making them public could lead to the development of new techniques to eradicate diseases, but on the other side most people are very sensitive about sharing medical information because of the simplicity of re-identify anonymized data, and there are a lot of regulatory constraints on such sharing.

- **Outcome equity**: bias may arise because of a lack of monitoring and mitigation of unintended consequences for any group affected by the system after deployment, directly or indirectly (for example, contact tracing apps may facilitate stigma or harassment).

2.5 Fairness

As discussed in Section 2.4, both equality and equity aim to achieve **fairness**, despite the different approaches of the two theories, but what fairness really is is a widely debated topic. A very generic definition, taken from [15], depicts it as "the quality of treating people equally or in a way that is right or reasonable".

In the sociological context, fairness is often seen as a synonym of *justice*, and consequently **social justice** is fairness as it manifests in the society,

described by [6, p. 405] as "an ideal condition in which all members of a society have the same rights, protections, opportunities, obligations, and social benefits". Although the literature on this subject does not always agree on their number, we can delineate five interrelated principles of social justice, by following the classification provided in [1]:

- **Access to resources**: a just society should provide services and resources that are available to each different socioeconomic group, in order to give everyone an equal start in life.
- **Equity**: in unjust societies, there are always disenfranchised groups. These groups need to receive more support from the society than privileged ones, in order to move towards the same outcome.
- **Participation**: everyone in a just society, and not just small groups of individuals, should be able to participate in the decisional processes that affect their lives.
- **Diversity**: a just society should recognize the value of diversity and cultural differences, and develop ad-hoc policies with the aim of breaking down societal barriers.
- **Human rights**: a just society should ensure the protection of everyone's civil, political, economic, cultural, and social rights.

Moving back to the data and computer systems perspective, and recalling the aforementioned concepts of equality and equity, we can distinguish between two different concepts of fairness [17]:

- **Individual fairness**: any two individuals who are similar *with respect to a task* should receive similar outcomes. The similarity between individuals should be captured by an appropriate metric function, usually difficult to determine. For example, deciding whether or not to display a specific advertisement is a classification task, and the definition of individual fairness assumes the existance of a task-specific metric (e.g. the number of clicks made by the users) capable of determining, for any two individuals, how (dis)similar they are for the specific task. Individual fairness is strictly related to the idea of equality.
- **Group fairness** (also known as *statistical parity*): demographics of the individuals receiving any outcome - positive or negative - should be the same as demographics of the underlying population. For example, in the problem of predicting if hiring applicants, assuming to divide

them into groups according to their gender, this means the acceptance rates of the applicants from the groups must be equal regardless of the protected attribute. Group fairness equalizes outcomes across protected and non-protected groups, and is therefore strictly related to the idea of equity.

Although individual and group fairness are not mutually exclusive in theory, in real life it is often hard to conciliate the two approaches. Furthermore, this categorization is not the only possible one: the authors of [34] collected and provided about twenty among the most prominent definitions of fairness, and applied each of them to a case study based on gender-related discrimination, in which the aim was to assign a credit score to people requesting a loan by using "Personal status and gender" as a protected attribute for the decision-making process, operated by a classifier (an algorithm that automatically orders or categorizes data into one or more of a set of "classes", in this case only "good credit score" and "bad credit score").

Among the others, a couple of peculiar definitions, often listed together with individual and group fairness, are the following:

- **Fairness through unawareness**: protected attributes are not used in the decision-making process, and therefore the subsequent decisions cannot rely on them. This "blind" approach relies on *impartiality* and is consistent with the disparate treatment principle, but removing features means losing information, and furthermore there could be features correlated to protected attributes that would not be removed, potentially introducing bias.

- **Counterfactual fairness**: a precise and non-technical definition is provided in [35]:

  > *A model is fair if for a particular individual or group its prediction in the real world is the same as that in the counterfactual world where the individual(s) had belonged to a different demographic group. However, an inherent limitation of counterfactual fairness is that it cannot be uniquely quantified from the observational data in certain situations, due to the unidentifiability of the counterfactual quantity.* [35, p. 1]

  To better clarify the concept, we could imagine a situation in which a software has the task of deciding whether or not to assign a promotion to the employees of a company by looking at their profile that includes,

among the other attributes, sex and race. The software is counterfactually fair if, for each individual, the outcome of the analysis is the same both in the case in which the real values of these attributes are used and in the case in which these values are replaced with others (counterfactuals).

The classifier resulted to be fair depending on the notion of fairness adopted, showing the impossibility of addressing fairness as a unique, broad and inseparable concept. This result is coherent with *Chouldechova's impossibility theorem* [11], which demonstrates, taking three definitions of fairness, the impossibility of satisfying all of them.

Chapter 3

Technical Preliminaries

The aim of this chapter is to provide to the reader preliminary notions about the tools adopted for this research and the related technical knowledge.

We will start by introducing *relational databases*, the *data science pipeline* and *data mining techniques*, and then focusing on more specific concepts such as *linear regression* and *functional dependencies*. An explanation of some needed *evaluation metrics* and *statistical concepts* will follow, and finally a description of the tools will be provided.

As specified in Chapter 2, the complementarity of the preliminaries is one of the key points of this research, and will allow the reader to have two perspectives of a different nature on the same problem.

3.1 Relational Databases

When dealing with computer systems, one of the most basic notions, often inappropriately taken for granted, is the one of "**data**", definable as:

> *Information, especially facts or numbers, collected to be examined and considered and used to help decision-making, or information in an electronic form that can be stored and used by a computer.* [14]

Therefore, a large amount of data stored in a computer in some organized manner is called a **database**. To be more precise, a database is "any collection of data, or information, that is specially organized for rapid search and retrieval by a computer" [8]; while the software that supports the management of these data is called a **database management system (DBMS)**.

The history of databases is deeply interconnected with the history of informatics itself, because the problem of how to store and retrieve information

appeared as one of the initial challenges of computer creators. However, in the past few decades the rapid and enormous evolution of computer systems and databases led to the adoption and the development of the so called "data models". A **data model** [2] is an abstract representation of an information system, which defines the data elements and the relationships between data elements. The aim of a data model is to give a clear and intuitive overview on how a system looks like, by providing a standardized description of its components, in such a way as to facilitate the understanding of the system itself and the possible integration with other systems.

Nowadays, the most widespread data model is the **relational model**, firstly proposed by Codd in [12]. The relational model represents a database as a collection of relations, depicted as tables of values. Each row of the table is a collection of related data values, referring to a real-world entity or relationship between entities. Therefore, we can simply define a **relational database** as a digital database based on the relational model of data. To make it clearer, the following list provides the main terms used in this context, together with a concise explanation, while Figure 3.1 shows them in a trivial example.

- **Table**, or **relation**: modeling of a real-world entity or of a relationship between real-world entities.
- **Row**, or **tuple**: single data record.
- **Column**, or **attribute**: property, or feature, of a relation.
- **Cardinality**: total number of tuples of a relation.
- **Degree**: total number of attributes of a relation.
- **Primary key**: attribute, or combination of attributes, that uniquely identifies a tuple among the others.
- **Domain**, or **data type**: set of values that a specific attribute can assume (for example, integer numbers, or boolean values).
- **Database schema**, or simply **schema**: blueprint of the database that outlines the way its structure organizes data into tables.
- **Database instance**, or simply **instance**: set of tuples in which each tuple has the same number of attributes as one of the relations of the database schema. It specifies the actual content of the database.
- **Integrity constraint**: property that is supposed to be satisfied by all instances of a database schema.

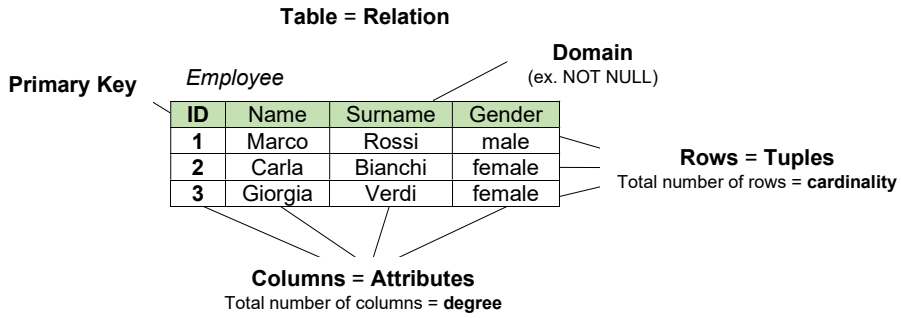


Figure 3.1: Relational model concepts in a trivial example. "Employee" is the name of the real-world entity of reference and therefore of the related table in the model.

Lastly, since this term will be often used in the subsequent sections and chapters, we define a **dataset** as a collection of data. More specifically, since our data are in a tabular format according to the relational model, a dataset simply corresponds to one or more database tables.

Further details on relational databases can be found in [2].

3.2 Data Science Pipeline

Because of the broadness of the concept, there is not a unique and precise definition of data management. In general, we can identify it as the process of acquiring, storing, organizing, and maintaining data created and collected by an organization. In [20], the author, referring to [25], classifies *data management*, together with *analytics*, as one of the two sub-processes to extract insights from data, while the overarching process is referred as **data science pipeline**, or *big data pipeline*. For the sake of clarity, since the term is the one used in [20], we define big data as:

> *Large volumes of high velocity, complex and variable data that require advanced techniques and technologies to enable the capture, storage, distribution, management, and analysis of the information.* [27]

However we preferred to adopt the name of "data science pipeline" instead of "big data pipeline", since we will not deal with big data, which are not a concept strictly inherent to this research.

Since fairness should be addressed in each phase of the data science pipeline, the subsequent list provides a concise explanation of the operations performed in each step, by following the classification proposed in [23], together with the main potential sources of bias.

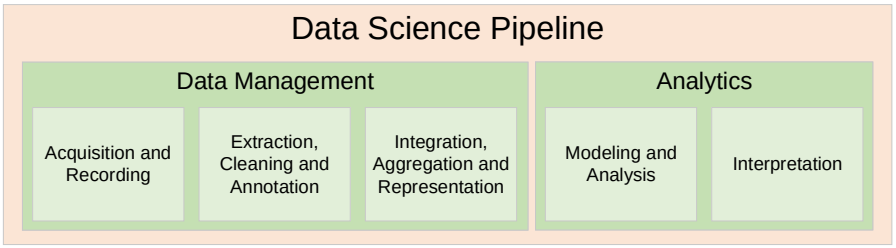


Figure 3.2: Data science pipeline. Image based on the one shown in [20].

- **Acquisition and recording**: data are recovered and captured. In this phase the introduction of bias could derive from some preliminary critical choices we have to deal with, concerning the availability of sources, the identification of who is represented by the data, the definition of what has been measured and of our duties to the people in the data (for example, we may owe them a certain degree of privacy).

- **Extraction, cleaning and annotation**: real data are most of the time messy and dirty, therefore we need to extract the relevant information and clean them, in order to express them in a structured form suitable for analysis. Unfortunately, data cleaning itself is based on assumptions, and wrong assumptions may lead to bias (for example, we may assume missing values in the data as missing at random, while there could be other, maybe ethical, reasons behind).

- **Integration, aggregation and representation**: data analysis often requires the collection of heterogeneous data from different sources, therefore we need to integrate them in order to guarantee syntactic and semantic coherence. Again, we have to rely on assumptions on the world, as for the case of data representation, in which a lot of choices are made in order to decide what to represent, potentially leading to bias (for example, in the context of sentiment analysis we may ascribe sentiment to labels, or we may decide to group age values instead of considering every single year).

- **Modeling and analysis**: before the actual analysis, an abstract model of the data is generated, in order to capture the essential components of the system and their interactions. However, the process of abstraction of concrete data in a conceptual standard model necessarily leads to the loss of information (for example, the relational model provides an intuitive overview of the system, but it does not include any semantics).

- **Interpretation**: a decision-maker, provided with the results of the analysis, has to interpret these results. This process usually requires to examine all the assumptions made and to retrace the analysis, and because of the complexity of the task and the problems that may arise from computer systems (bugs, errors), a human (and therefore impossibly perfectly fair) supervision is needed (for example, the failures of system components can go unnoticed and result in loss of data, or the data format may have changed without being notified, and therefore the system should be equipped with monitoring scripts and mechanisms to obtain user confirmation and correction).

3.3 Data Mining Techniques

> *Data mining is a collection of techniques for efficient automated discovery of previously unknown, valid, novel, useful and understandable patterns in large databases.* [33, p. 80]

Data mining is a broad topic, and usually a variety of procedures are needed in order to gain knowledge from data. However, we can distinguish three main categories of techniques, in each of which the fairness problem should be addressed differently:

- **Preprocessing techniques**: procedures used to transform the raw data in a useful and efficient format. The aim is to improve the overall quality of the data and consequently the data mining results.
- **Inprocessing techniques**: data are subjected to various methods using machine learning and artificial intelligence algorithms to generate a desirable output.
- **Postprocessing techniques**: methods to evaluate the extracted knowledge, visualize it, or merely document it for the end user. The knowledge can also be interpreted and incorporated into an existing system.

For the purpose of this research, we will focus on preprocessing techniques, which constitute one of the most critical steps in the data mining process, since they deal with the preparation and transformation of the initial dataset; while the bias analysis we will perform making use of the tools adopted is part of data (in)processing. Data preprocessing methods are divided into four categories [33]:

- **Data cleaning**: since real-world data are often incomplete, noisy, and inconsistent, some routines are needed in order to fill in missing values, smooth out the noise and correct the inconsistencies. For what concerns *missing values*, these procedures include the removal of the specific tuple, or the filling (manual or automatic) of the missing value by using, for example, a constant (e.g. "unknown"), the mean (for numerical attributes) or simply what is perceived to be the most probable value. Noise instead can be seen as a random error or variance in a measured variable, and some smoothing techniques for *noisy data* are:

  - **Binning**: a data value is smoothed by looking at its "neighbourhood", that is, the values around it.
  - **Regression**: data are fitted to a function, in order to be smoothed according to the function itself. A specific type of regression, useful for our analysis, is *linear regression*, which will be furtherly explored in Section 3.4.
  - **Clustering**: similar data are organized into groups of values, called "clusters". Values that fall outside the set of clusters may be considered as outliers.

- **Data integration**: as mentioned in Section 3.2, data often come from different (possibly heterogeneous) sources, and therefore they need to be combined in order to obtain a coherent model and remove inconsistencies (such as redundancies between attributes, where some can be derived from others).

- **Data transformation**: data are transformed or consolidated in appropriate forms suitable for the mining process. Some techniques used in this context are:

  - **Normalization**: data values are scaled so as to fall within a specified range, such as $(-1.0, 1.0$ or $0.0, 1.0)$.
  - **Aggregation**: new attributes are constructed from the given set of attributes to help the mining process by summarizing or aggregating information (for example, daily sales data may be aggregated so as to compute annual total amounts).
  - **Generalization**: raw (or low-level) data are replaced by higher-level ones, by following a specific hierarchy (for example, the attribute "city" can be generalized to "country").

  - **Discretization**: raw values of numeric attributes are replaced by interval levels or conceptual levels (for example, age values between 15 and 18 could be labeled as "adolescence").

- **Data reduction**: in order to make mining more effective and get better analytical results, several techniques can be applied to obtain a reduced representation of the dataset that is much smaller in volume, yet closely maintains the integrity of the original data. These methods include, among the others, *attribute subset selection*, in which attributes considered as not particularly relevant for the analysis are removed, and *numerosity reduction*, where data are replaced by smaller data representations, such as parametric models.

3.4 Linear Regression

In order to fully understand how one of the adopted tools works (the one we refer to as "Glassdoor Method", described later in Section 3.8), it is appropriate to have a closer look at **linear regression** [22]. As mentioned in Section 3.3, linear regression is a preprocessing technique used to smooth out noise or to find patterns within a dataset, which attemps to model the relationship between two or more variables by fitting data to a linear equation (represented, in the two-variable case, by a straight line in a Cartesian plane). The results from linear regression help in predicting an unknown value depending on the relationship with the predicting variables. For example, the height and weight of an individual generally are related: usually taller people tend to weigh more. We could use regression analysis to help predict the weight of a person, given their height.

We can distinguish between **simple linear regression**, in which a single input variable is used to model a linear relationship with the target variable (as for the example of height and weight), and **multiple linear regression**, where more predicting variables are used.

For simple linear regression, the reference equation is:

$$y = \beta_0 + \beta_1 x + \epsilon$$

Variable x is called *explanatory* or *independent variable*, while y is referred to as *dependent variable*; β_1 is the *slope* of the line, also known as regression coefficient, and β_0 is the *intercept* (the value of y when $x = 0$), while ϵ is the *error* in the estimation of the regression coefficient, also known as residuals, which account for the variability in y that cannot be explained by the linear relation between x and y.

For multiple linear regression, the formula is generalized in order to incapsulate also the other independent variables $(x_1, \ldots, x_n)$ and the related slope coefficients $(\beta_1, \ldots, \beta_n)$:

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_n x_n + \epsilon$$

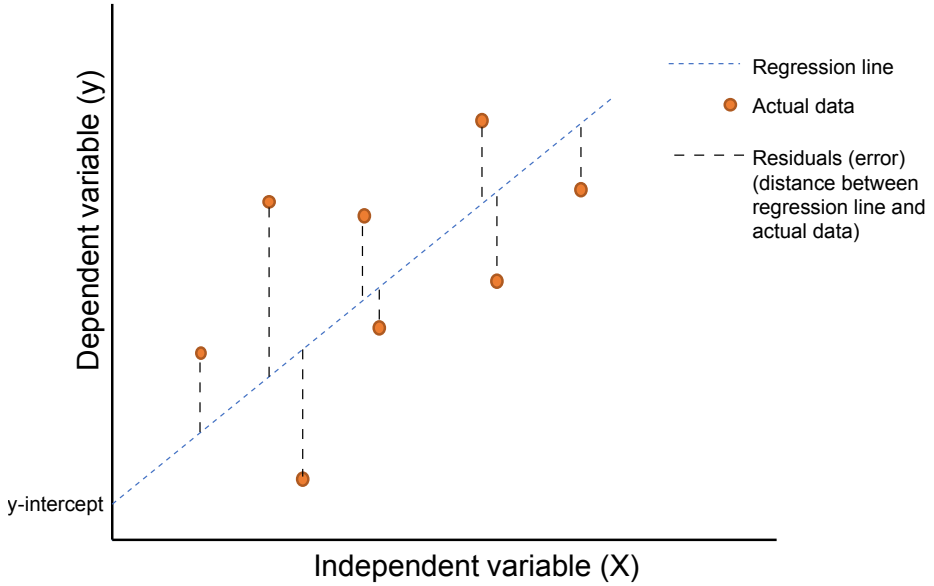


Figure 3.3: Simple linear regression graph.
Source: `https://www.reneshbedre.com/assets/posts/reg/mlr/residual.svg`.

Figure 3.3 shows a simple linear regression graph. It is important to point out that the x and y variables remain the same, since they represent data features that cannot be changed, while the values that we can control are the slope and the intercept. Indeed, there can be multiple straight lines depending upon the values of intercept and slope, and what the linear regression algorithm does is to fit multiple lines on the data points and return the line that results in the least error.

Another important parameter for regression analysis is R^2, also known as *coefficient of determination* (or *coefficient of multiple determination* for multiple linear regression). It is a statistical measure of how close the data are to the fitted regression line, and therefore it indicates how much variation of the dependent variable is explained by the independent variable(s) in a regression model. R^2 values range from 0 to 1 and are commonly stated as percentages from 0% to 100%, where 0% refers to a model that explains none of the variability of the data around its mean, while 100% refers to a model

that explains all the variability of the data around its mean (in this case, all the actual data values would be on the regression line).

Formally, we can define R^2 as:

$$R^2 = 1 - \frac{Unexplained Variation}{Total Variation} = 1 - \frac{\sum_i (y_i - \hat{y}_i)}{\sum_i (y_i - \bar{y})}$$

where y_i is one of the actual data values, $\hat{y}_i$ is the corresponding predicted value, and $\bar{y}$ is the mean of all the y_i values, for $i = 1, \ldots, n$.

Generally speaking, at least for the purpose of this research, the higher the R^2 value, the better the model fits the data.

3.5 Functional Dependencies

One of the tools adopted, FAIR-DB, uses specific classes of integrity constraints, known as dependencies, to detect unfair behaviors in datasets (further details on the tool will be provided in Section 3.9). A **dependency** is a constraint that applies to or defines the relationship between attributes, and it occurs in a database when information stored in a table uniquely determines other information stored in the same table. Basically, dependencies are constraints not necessarily imposed by the system designer but intrinsically satisfied by the data.

Functional dependencies (FDs) are a specific type of dependency, involving two (sets of) attributes of the same relation in which the first uniquely determines the second or, in other words, knowing the value of one attribute (or set of attributes) is enough to tell the value of the other one. The notation to indicate a functional dependency is:

$$A \rightarrow B$$

which can be read as "B is functionally dependent upon A", or "A uniquely determines B", whereas A and B are attributes (or eventually sets of attributes) of a table. A is called antecedent, or *left-hand-side (LHS)*, and B consequent, or *right-hand-side (RHS)*. An example of functional dependency could be the one of a table containing the information about the employees of a company, as in Figure 3.1. Here the *ID* attribute uniquely identifies the *Name* one, because by knowing the employee's ID we can tell what the employee's name is. Therefore, $ID \rightarrow Name$. More specifically, since also the employee's surname and gender are uniquely identified by the ID, we can write $ID \rightarrow Name, Surname, Gender$. Another example is provided in Table 3.1, in which $Temperature, pH, Season \rightarrow Ideal$.

Orange Plantation			
Temperature	pH	Season	Ideal
28	7	Autumn	Y
20	7	Autumn	N
28	7	Winter	N
29	7.5	Autumn	Y
27	6.5	Winter	N
27	7.5	Summer	N
20	6.5	Spring	N
28	7	Summer	N
27	6.5	Autumn	Y

Table 3.1: "Orange Plantation" table. It shows whether or not ambient temperature (°C), soil pH and planting season represent ideal conditions for planting oranges.

Functional dependencies are a very well-known concept for data scientists, especially for those who work on relational models, and further details on them can be found in [2]. However, the constraints imposed by functional dependencies are often too strict for real-world datasets (they must indeed hold for all the tuples of a table), so in the past few years generalizations of FDs have been proposed and started to be considered in their place. **Relaxed functional dependencies (RFDs)** can indeed be simply defined as functional dependencies where some constraints are deleted (relaxed). The authors of [9] distinguished 35 different categories of RFDs, but the ones relevant for our research are the following:

- **Approximate functional dependencies (AFDs)**:

  AFDs are FDs holding on almost every tuple. [9, p. 151]

  In order to quantify how an AFD "almost" holds, several measures have been proposed, including the so called *g3*, defined as "the (normalized) minimum number of tuples that need to be removed from a relation instance in order for an FD to hold" [9, p. 151], whereas "relation instance" is simply a synonym of "relation". The g3 measure is therefore an index whose value ranges between 0 and 1, indicating the percentage of tuples of a table to be removed in order for an FD to hold (0 = none, 1 = all). An example of AFD in Table 3.1 is:

  $$Temperature, pH \rightarrow Ideal$$

  because almost all the values of *Temperature* and *pH* determine the

Ideal value, but this is not true for:

$$Temperature =\text{‘ 28 ’}, pH =\text{‘ 7 ’} \rightarrow Ideal$$

since in most of the cases (two out of three) when *Temperature* =‘ 28 ’ and pH =‘ 7 ’ then *Ideal* =‘ Y ’, but in one case *Ideal* =‘ N ’.

- **Conditional functional dependencies (CFDs)**:

  > *[CFDs] use conditions to specify the subset of tuples on which a dependency holds.* [9, p. 152]

  This type of dependencies allows to catch particular and concrete patterns in the dataset, in fact they make possible to analyze precise values of the tuples and be more specific. An example of CFD, related to Table 3.1, is the following:

  $$Temperature =\text{‘ 28 ’}, pH =\text{‘ 7 ’}, Season \rightarrow Ideal$$

  meaning that, for tuples in which *Temperature* =‘ 28 ’ and pH =‘ 7 ’, the *Season* parameter functionally determines the *Ideal* one. Another example could be:

  $$Season =\text{‘ Summer ’} \rightarrow Ideal =\text{‘ N ’}$$

  interpretable as: “when the attribute *Season* has value Summer, the attribute value of *Ideal* is N”.

- **Approximate conditional functional dependencies (ACFDs)**: FDs obtained by combining the two kinds of relaxed dependencies discussed above. Unifying the two relaxation criteria makes it possible to detect specific and not exact rules, which can highlight anomalies or unexpected patterns in the database, allowing to recognize cases where a value of a certain attribute *frequently determines* the value of another one. An example of ACFD in Table 3.1 is:

  $$Season =\text{‘ Autumn ’} \rightarrow Ideal =\text{‘ Y ’}$$

  which can be read as: “when attribute *Season* has value Autumn, the attribute value of *Ideal* is Y if we delete a maximum number of tuples N from the dataset”. The rule indeed holds for almost all the tuples of the table, apart for the one in which *Temperature* =‘ 20 ’, pH =‘ 7 ’, *Season* =‘ Autumn ’ and *Ideal* =‘ N ’. It is worth to specify that, even though the notation is the same used for CFDs, the relaxation on the number of tuples is implicit in the classification of a rule as an ACFD.

3.6 Evaluation Metrics

The aim of this section is to introduce some evaluation metrics for functional dependencies used by one of the adopted tools, FAIR-DB, described later in Section 3.9.

- **Support**:

$$\text{Support}(X \to Y) = \text{supp}(X, Y) = \frac{\#(X, Y)}{\#tuples}$$

  where $\#(X, Y)$ is the amount of times the (sets of) attributes X and Y appear together in the dataset and $\#tuples$ is the total amount of tuples in the table. The support represents the percentage of records in the dataset that verify the dependency $X \to Y$, and it is therefore an index whose value ranges between 0 and 1.

- **Confidence**:

$$\text{Confidence}(X \to Y) = \text{conf}(X, Y) = \frac{\text{supp}(X, Y)}{\text{supp}(X)}$$

  where $\text{supp}(X)$ is the percentage of tuples in the dataset containing the (set of) attributes X (antecedent, or LHS, of the rule). The confidence shows how frequently the dependency $X \to Y$ is verified, knowing that the antecedent X is verified, and it is therefore an index whose value ranges between 0 and 1. A confidence equal to 1 means that only the valid and exact rules (non-relaxed FDs) will be selected, while decreasing its value implies relaxing the constraint on the number of tuples to be considered. In this context, it can be seen as an analogous metric to the g3 one, mentioned in Section 3.5.

- **Difference**:

$$\text{Difference}(X \to Y) = \text{diff}(X, Y) = \text{conf}(X, Y) - \text{conf}(X \setminus X_p, Y)$$

  where X_p is the subset of protected attributes of the antecedent X of the dependency $X \to Y$. The difference is basically a subtraction between the confidence value of a dependency and the confidence value calculated on the same dependency but excluding all the protected attributes from the antecedent of the rule. Being a subtraction of indices of value between 0 and 1, and given $\text{conf}(X, Y) \geq \text{conf}(X \setminus X_p, Y)$, the difference is also an index whose value ranges between 0 and 1. It indicates how much a dependency is "unethical" (the higher the value,

the more unfair is the dependency), and it gives an idea on the impact of the protected attributes on Y. Last but not least, it is important to point out that the difference is a novel metric, firstly introduced in [5] and specifically designed with the aim of measuring the "ethical" level of a dependency.

3.7 Statistical Concepts

The aim of this section is to introduce some statistical concepts useful for fully understanding the behavior of one of the adopted tools, Ranking Facts, described later in Section 3.10.

The first required notion is the one of a **hypothesis test**, which in statistics is a way to test the obtained result of a survey or experiment on a sample, in order to check if the result is meaningful and extendable to the whole population or if it has happened by chance. In this context, two interpretations are proposed: the first is known as *null hypothesis* (symbolized as H_0), which is the idea that there is no relationship in the population and that the relationship in the sample is caused by errors (informally, this is the "occurred by chance" interpretation); the second is called *alternative hypothesis* (whose symbol is H_1) and it is the idea that the relationship in the sample reflects an existing relationship in the population.

Generally, the rationale behind a hypothesis test is:

1. Assume the null hypothesis true.

2. Determine how likely the sample relationship would be if the null hypothesis were true.

3. If the sample relationship would be extremely unlikely, then *reject* the null hypothesis in favour of the alternative hypothesis. If it would not be extremely unlikely, then *retain* the null hypothesis.

A crucial step in hypothesis testing is to find the likelihood of the sample result if the null hypothesis were true. This probability is called ***p*-value**. Low p-value means that the sample result would be unlikely if H_0 were true and leads to the rejection of the null hypothesis, while high p-value means that the sample result would be likely if H_0 were true and leads to the retention of the null hypothesis. To quantify how low the p-value must be in order to consider the result unlikely enough to reject the null hyphotesis, a parameter known as *significance level* α is used, and its value is usually set to 0.05 (5%). The significance level represents the probability of making the

mistake of rejecting the null hypothesis when in fact it is true (*type I error*): if p-value $> \alpha$ we accept the null hypothesis and the result is considered not statistically significant, otherwise we reject the null hypothesis and the result is said to be *statistically significant*.

A particular type of hypothesis test is the **z-test**, used when data are approximately normally distributed (i.e. the plotted data have the shape of a bell curve on the graph). In order for a z-test to be used, data points should also be independent from each other and the sample size should be greater than 30. The z-test relevant to our research is the *two sample z-test*, which allows to compare two proportions to check if they are the same (H_0) or not (H_1). The reference formula is:

$$z = \frac{p_1 - p_2}{\sqrt{(\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2})}} = \frac{p_1 - p_2}{\sqrt{p(1-p)(\frac{1}{n_1} + \frac{1}{n_2})}}$$

where n_1 and n_2 are the sizes of the samples, p_1 and p_2 are the proportions of the samples, p is the overall sample proportion (total number of "positive" results over total number of people) and σ_1^2 and σ_2^2 represent the variances of the two populations. For the sake of completeness, we define the *variance* as the measure of how far each value in the dataset is from the average value (i.e. the mean, as defined in Section 3.8):

$$\sigma^2 = \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{n}$$

where n is the size of the population, x_i is the i-th value and μ represents the mean. In the z-score formula above, for the sake of simplicity, σ_1^2 and σ_2^2 are approximated by the variance of the Bernoulli distribution (i.e. the probability distribution for a random experiment with only two possible outcomes). To better clarify, let's consider the example of testing two different COVID-19 vaccines: the first works on 248 people out of a sample of 496, while the second works on 23 people in a sample of 100. To check if the vaccines are comparable we firstly calculate the proportions:

$$p_1 = 248/496 = 0.5$$

$$p_2 = 23/100 = 0.23$$

The overall sample proportion is:

$$p = \frac{248 + 23}{496 + 100} \simeq 0.45$$

And finally the z-score is:

$$z \simeq \frac{0.5 - 0.23}{\sqrt{0.45(1 - 0.45)(\frac{1}{496} + \frac{1}{100})}} \simeq 4.95$$

To find out if the obtained result should lead us to accept or reject the null hypothesis, we can look at the known values of z-score, related to the most commonly used α values. For example, the z-score related to $\alpha = 0.05$ is 1.96, meaning that 95% of the area under the normal curve lies within the range $[-1.96, +1, 96]$, and since $4.95 > 1.96$ we can reject the null hypothesis (for z-scores higher than 1.96 with $\alpha = 0.05$, p-value < 0.05).

3.8 The "Glassdoor Method"

After having described the basics, we will now make an overview of the tools used for this research. The first one is a technical guide to analyze gender pay gap in a company, provided by **Glassdoor** in [10].

Glassdoor is a website in which employees and ex-employees of companies anonimously review enterprises and their superiors, with the overall aim of providing insights about jobs and companies and helping people find the most suitable working position for them. The society was founded in 2007 in the U.S. and the website was made available in 2008; since then, the company has grown to become the worldwide leader in the sector.

As specified in the introduction of the guide [10, p. 2], according to a 2016 Glassdoor survey, 67% of the U.S. employees would not apply for jobs at employers where they believe a gender pay gap exists. The purpose of the report is therefore to help HR practitioners in analyzing the internal gender pay gap of their companies, by providing them specific technical knowledge.

First of all, by "gender pay gap" the authors mean:

> *The difference between average pay for men and women, both before and after we've accounted for differences among workers in education, experience, job roles, employee performance and other factors aside from gender that affect pay.* [10, p. 3]

Two measures are proposed in the report, respectively referred as "unadjusted" and "adjusted" pay gap:

- **"Unadjusted" pay gap**:

  > *Average pay for men as a group, compared to average pay for women as a group.* [10, p. 3]

Therefore, the formula for estimating "unadjusted" gender pay gap is:

$$U = \frac{\text{avg}(BasePay)_m - \text{avg}(BasePay)_f}{\text{avg}(BasePay)_m}$$

where $\text{avg}(BasePay)_m$ is the average pay (arithmetic mean of salaries) of male employees, while $\text{avg}(BasePay)_f$ is the average pay of female employees. For the sake of completeness, despite being a basic mathematical concept, we define the *mean* as the sum of a collections of numbers (in this case, salaries) divided by the count of numbers in the collection (the total amount of males or females). Taking n as total number of male employees:

$$\text{avg}(BasePay)_m = \frac{\sum_{i=1}^{n} BasePay_i}{n}$$

and the same holds for female employees.

- **"Adjusted" pay gap**: while the "unadjusted" pay gap is basically a simple comparison of all women with all men, the "adjusted" pay gap compare similarly situated male and female employees, in order to include in the calculation the numerous factors that affect the pay (e.g. job title, or educational level). The estimation of the "adjusted" pay gap is based on linear regression, a concept presented in Section 3.4, and the reference formula is:

$$y_i = \beta_1 Male_i + \beta_2 X_i + \epsilon_i$$

where y_i is the annual salary of worker i, $Male_i$ is a dummy indicator equal to 1 for males and 0 for females, and X_i is a collection of attributes of employees which may be relevant for the calculation (job title, educational level, etc.). The estimated coefficient β_1 represents the approximate pay advantage for men compared to women.

3.9 FAIR-DB

The second tool adopted for this research is called **FAIR-DB** (***F**unction**A**l Dependenc**I**es to discove**R** **D**ata **B**ias*), and it is a framework with the aim of discovering unfair behaviors in datasets, developed at Politecnico di Milano. As the name itself suggests, FAIR-DB is based on functional dependencies, and it falls within the category of preprocessing techniques since it works by finding conditions (constraints) already present in the data. The developers

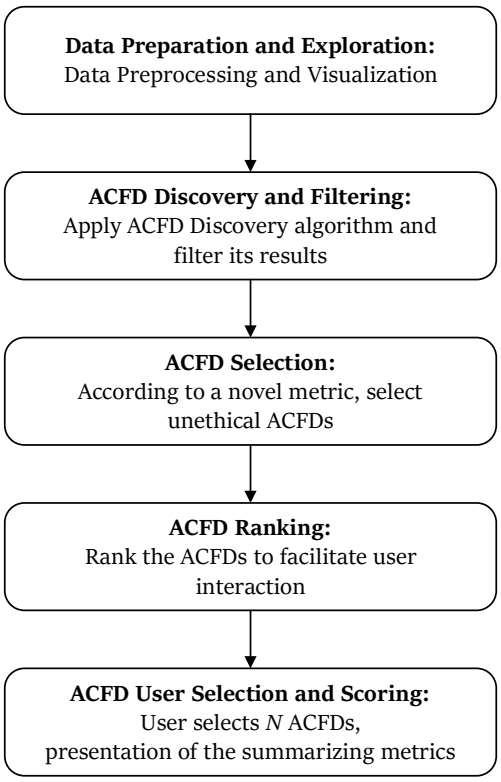


Figure 3.4: Steps of the FAIR-DB framework. Image based on the one shown in [5].

documented the functioning of FAIR-DB in [5], by providing an overview of the tool, together with a clarifying example.

Figure 3.4 shows the framework workflow, while a brief explanation of each phase, as documented in [5], is reported below.

- **Data preparation and exploration**: data are imported and data integration is eventually performed. Data cleaning, feature selection and discretization techniques are also applied in this phase, in order to deal with missing values, select the smallest set of attributes relevant for the analysis and transform data from numerical to nominal data type. Data are finally plotted in order to help the user in identifying groups in the dataset and eventually majority and minority classes.

- **ACFD Discovery and filtering**: the *ACFD Discovery* algorithm, presented in [29], is applied to extract approximate conditional functional dependencies from the dataset. The algorithm takes as input the dataset and three threshold parameters: *minimum support*, *mini-*

mum confidence and *maximum antecedent size* of the ACFD sought. From the output, dependencies not involving at least one of the protected attributes and the target attribute (the one used as reference to search for discrimination, e.g. *Income*) are removed, as well as dependencies containing variables (in which one or more attributes are not assigned to a specific value, e.g. the attribute *Ideal* in *Temperature* =‘ 28 ’, *pH* =‘ 7 ’→ *Ideal*).

- **ACFD selection**: for each ACFD, some metrics are computed to capture the "ethical level" of the dependency. In particular, the *difference* metric described in Section 3.6, as mentioned in the section, is a novel score introduced for this purpose, and a second measure called *p-Difference* is calculated for each protected attribute. The p-Difference indicates how much a dependency shows bias with respect to a specific protected attribute, and it is computed in the same way as the difference, but excluding the attribute from the antecedent of the rule. According to the values of the metrics, the most interesting ACFDs are selected.

- **ACFD ranking**: the ACFDs are ranked in descending order of importance according to *support*, *difference*, or *mean*. The support emphasize the *pervasiveness* of a rule, because it indicates the number of tuples involved by the dependency, so the higher the value, the more tuples are affected by the ACFD. The difference privileges the *unethical aspect* of a rule, because it highlights dependencies where the values of the protected attributes influence most their RHS. The mean is computed as mean of support and difference, and therefore it gives more importance to the rules with the *best trade-off* between pervasiveness and unethical perspective.

- **ACFD user selection and scoring**: the user selects N ACFDs perceived as the most problematic, and the system computes metrics (based on support, difference and p-Difference of the selected rules) to summarize the level of unfairness of the dataset.

3.10 Ranking Facts

The third tool used for this research is called **Ranking Facts**, a Web-based application (there also exists a notebook version) developed by the team of *Data, Responsibly*[1]. Ranking Facts, as the name itself suggests, is a

[1]Available at: `http://demo.dataresponsibly.com/rankingfacts/`.

ranking tool: ranking is an action commonly performed by the vast majority of the algorithms we use every day: Google itself ranks the results of our searches and provides us a list in descending order of relevance, and the same mechanism is used in various contexts of different nature, like dating or hiring applications. These specific scenarios are particularly relevant, because it is people who are ranked, and therefore discrimination against individuals or protected groups could arise, or the outcome could exhibit low diversity. Ranking Facts is based on the concept of *nutritional labels*, in analogy to the food industry, where simple, standard labels convey information about the ingredients and production processes. Similarly, in the tool nutritional labels are derived as part of the complex process that gave rise to the data or model they describe, embodying the paradigm of interpretability-by-design.

As documented in [36], Ranking Facts is a collection of visual widgets with the aim of providing to the user information about the ranking in terms of stability, fairness and diversity. A brief description of how they work, taken from [36], is reported below.

- **Recipe** and **Ingredients**: the former widget succintly describes the ranking algorithm, by listing the attributes used for ranking together with their weights, as specified by the user; while the latter shows, in descending order of importance, the attributes that really affect the ranking.

- **Stability**: it explains whether the ranking methodology is robust on the specific dataset in use. An unstable ranking is one where slight changes to the data (e.g. due to uncertainty and noise), or to the methodology (e.g. by slightly adjusting the weights of the attributes in the recipe) could lead to a significant change in the output.

- **Fairness**: it quantifies whether the ranked output exhibits statistical parity (group fairness) with respect to one or more protected attributes, such as gender or race of individuals. The notion of fairness is defined specifically for rankings and it can be computed comparing only binary categorical attributes (i.e. non-numerical attributes with just two possible values). The summary view of the widget presents the output of three fairness measures:

  - **FA*IR** [38]: ranking algorithm based on the assumption that on a ranking, the desired good for an individual is to appear in the result and to be ranked among the top-k positions. The outcome is therefore unfair if members of a protected group are

p \ k	1	2	3	4	5	6	7	8	9	10	11	12
0.1	0	0	0	0	0	0	0	0	0	0	0	0
0.2	0	0	0	0	0	0	0	0	0	0	1	1
0.3	0	0	0	0	0	0	1	1	1	1	1	2
0.4	0	0	0	0	1	1	1	1	2	2	2	3
0.5	0	0	0	1	1	1	2	2	3	3	3	4
0.6	0	0	1	1	2	2	3	3	4	4	5	5
0.7	0	1	1	2	2	3	3	4	5	5	6	6

Table 3.2: Minimum number of candidates in the protected group that must appear in the top k positions to pass the ranked group fairness criteria with $\alpha = 0.1$. Considering for example $gender$ as a protected attribute with values ' M ' and ' F ', the minimum number of females (or eventually males) appearing in the top-5 with $p = 0.4$ is 1. Table based on the one shown in [38].

systematically ranked lower than those of a privileged group, and a ranking algorithm discriminates unfairly if this ranking decision is based fully or partially on a protected feature.

The *ranked group fairness* criterion used by the algorithm compares the number of protected elements in every prefix of the ranking (i.e. the top-i positions of the ranking, with $i \in [1, k]$) with the expected number of protected elements if they were picked at random using Bernoulli trials (independent "coin tosses") with success probability p. The statistical test also includes a significance parameter α, corresponding to the probability of a type I error, which means rejecting a fair ranking. A clarifying example is provided in Table 3.2.

The algorithm produces a top-k ranking that satisfies the ranking group fairness criterion mentioned above while maximizing *utility*, which means selecting the "best" tuples, assigning them a score based on the relevant attributes used for the evaluation (e.g. picking the most qualified candidates for a job position by looking at their educational level).

- **Proportion** [39]: this measure is based on the concept of z-test, as described in Section 3.7.
- **Pairwise**: also known as *pairwise comparison*, it is a tool for prioritizing and ranking multiple options relative to each other. A matrix is generally used to compare each option in pairs and determine which is the preferred choice or has the highest level of

	Coffee	Wine	Tea	Beer	Sodas	Milk	Water
Coffee	1	9	3	1	$\frac{1}{2}$	1	$\frac{1}{2}$
Wine	$\frac{1}{9}$	1	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
Tea	$\frac{1}{3}$	3	1	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{4}$	$\frac{1}{5}$
Beer	1	9	4	1	$\frac{1}{2}$	1	1
Sodas	2	9	5	2	1	2	1
Milk	1	9	4	1	$\frac{1}{2}$	1	$\frac{1}{2}$
Water	2	9	5	1	1	2	1

Table 3.3: Drink consumption in the U.S. represented in a pairwise comparison matrix. Table based on the one shown in [30].

importance based on defined criteria. At the end of the comparison process, each option has a rank or relative rating as compared to the rest of the options. Table 3.3 provides an example: scores are assigned based on how strongly the consumption of a drink on the left dominates that of a drink at the top. For example, when coffee on the left is compared with wine at the top, since coffee appears to be extremely more consumed, 9 is entered in the first row and second column position. A score of $\frac{1}{9}$ is automatically entered in the second row and first column position. According to the matrix, the final ranking would be:
Sodas : 0.252, *Water* : 0.228, *Beer* : 0.164, *Milk* : 0.148, *Coffee* : 0.142, *Tea* : 0.046, *Wine* : 0.019.

All these measures are statistical tests, and whether a result is fair is determined by the computed p-value.

- **Diversity**: since fairness is also related to representation, this widget shows diversity with respect to a set of demographic categories of individuals, or a set of categorical attributes of other kinds of items, by displaying the proportion of each category in the top-10 ranked list and overall (i.e. considering all the elements in the ranking).

Chapter 4

Experiments

The aim of this chapter is to describe the experiments conducted using the tools introduced in Chapter 3.

For each case study, we will provide a *dataset description*, in order to let the reader understand how the original dataset looks like, then we will discuss about our *data preprocessing* choices, and finally about how the tools performed on the preprocessed dataset, providing further details on the algorithms when needed. Lastly, we will question the results obtained by considering the impact of *other design choices*, verifying how the tools react when different decisions are made.

4.1 Our Societal Focus: Gender Gap

For the purpose of our research, we can simply define social problems as conditions or behaviors that have negative consequences for large numbers of people and that are generally recognized as conditions or behaviors that need to be addressed. We decided to focus on a specific category of social problems, namely those related to discrimination, and in particular on the so called "**gender gap**". According to [16], gender gap is definable as:

> *A difference between the way men and women are treated in society, or between what men and women do and achieve.* [16]

Specifically, the focus of our experiments is *gender pay gap*, already mentioned in Section 3.8, that is, the average difference between the remuneration for men and women in the workforce, or, in other words, a measure of what women are paid relative to men. The experiments are centered on the economical perspective because it is the easiest to measure in the data, being quantifiable for example as a number representative of a person's average

monthly salary, but other facets of the gender gap problem will come into play when dealing with sociological studies.

4.2 Case Study 1: Chicago

4.2.1 Dataset Description

The main purpose of this research is to combine the technological perspective with the sociological one, in order to analyze the strenghts and weaknesses of the adopted tools in real-world scenarios. For this reason, we decided to use real-world datasets, containing information related to public employees of the U.S., and more specifically of public employees working in the cities of Chicago[1] and San Francisco[2]. It is worth to specify that these datasets exist because of The Freedom of Information Act (FOIA): a federal law constituting Title 5 of the United States Code (5 U.S.C. §552), which claims that federal employee salaries must be public information under open government laws.

The **Chicago** dataset we considered includes 31,858 tuples and is made up of 8 attributes, briefly described as follows:

- *Name*: full name of the employee in the form of "Surname, Name".
- *Job Titles*: categorical variable representing the job title of the employee (e.g. POLICE OFFICER). There are 1089 distinct values.
- *Department*: categorical variable representing the job department where the employee works (e.g. POLICE). There are 36 distinct values.
- *Full or Part-Time*: binary categorical variable describing whether the employee is employed full-time (F) or part-time (P).
- *Salary or Hourly*: binary categorical variable describing whether the employee is paid on a hourly basis or salary basis. Hourly employees are further defined by the number of hours they work in a week.
- *Typical Hours*: numerical variable describing the typical amount of work (in terms of number of hours per week) for hourly employees. For salary employees the attribute value is null.

[1] Available at: `https://www.chicago.gov/city/en/depts/dhr/dataset/current_employeenamessalariesandpositiontitles.html`.

[2] Available at: `https://www.kaggle.com/tomtillo/san-francisco-city-payrollsalary-data-20112019`.

- *Annual Salary*: numerical variable describing the annual salary rate. It only applies for employees whose pay frequency is Salary, while for hourly employees the attribute value is null.

- *Hourly Rate*: numerical variable describing hourly salary rates for employees whose pay frequency is Hourly. For salary employees the attribute value is null.

4.2.2 Data Preprocessing

In order to simplify the subsequent bias analysis, we operated some **data transformation** processes on the attributes, choosing what we believe to be the most suitable names. For the Chicago dataset, we renamed *Job Titles* in *Job Title* and *Full or Part-Time* in *Status*. We also performed some **data aggregation**, estimating the *Annual Salary* of hourly employees by using the formula *Typical Hours* $\times$ *Hourly Rate* $\times$ 52, where 52 is a constant representing the number of weeks in a year.

Since our focus is gender pay gap but the original datasets do not contain a *Gender* attribute, we adopted a Python package called `gender-guesser`[3]. The aim of the package is to infer a person's gender from their first name, and the possible outcomes are: unknown (name not found), andy (androgynous), male, female, mostly_male, or mostly_female. The difference between andy and unknown is that the former is found to have the same probability to be male than to be female, while the latter means that the name was not found in the database. For each employee, we split the *Name* attribute to obtain their *First Name*, and then we inferred their gender by using the package. We obtained (out of the total of 31,858 tuples):

- unknown: 2,653 values.

- andy: 184 values.

- male: 20,562 values.

- female: 6,954 values.

- mostly_male: 775 values.

- mostly_female: 730 values.

In order to get coherent results in case of multiple experiments on the same dataset, we decided to remove the tuples related to unknown and androgynous

[3]Available at: `https://pypi.org/project/gender-guesser`.

names instead of randomly assign a gender to them (otherwise, we would have get different numbers at each execution of the preprocessing algorithm). Furthermore, we assumed mostly male names to be effectively related to males and mostly female names to be effectively related to females, and therefore we got 21,337 male values and 7,684 female values for a newly generated *Gender* attribute as a result of this first **data cleaning** process. As we will furtherly discuss in Section **??**, these decisions, together with the adoption of the `gender-guesser` library itself, represent some of the most critical choices we had to deal with, because they can lead to the introduction of (technical) bias.

We also operated **data reduction** by removing the *Typical Hours*, *Hourly Rate*, and *First Name* columns, not relevant for our analysis.

As a consequence of the first data cleaning process, the number of different job titles decreased from 1,089 to 1,057. However, since the FAIR-DB tool used for bias analysis requires user interactions, and in order to lighten the workload and speed up computational times, we decided to remove job titles with less than 100 occurrences.

Our final preprocessed dataset includes 20,309 tuples, of which 16,146 males and 4,163 females, and with 35 distinct *Job Title* values and 20 distinct *Department* values.

We decided to plot the *Annual Salary* values distribution, in order to get a visual overview on the incomes and estimate possible threshold values for the creation of interval levels. The resulting graph is displayed in Figure 4.1.

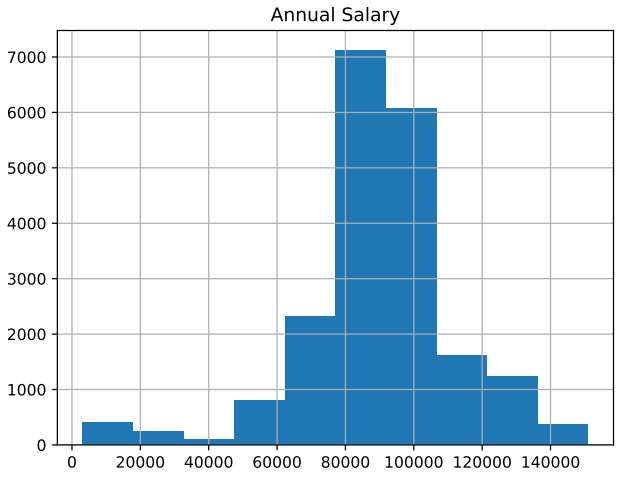


Figure 4.1: Distribution of the Annual Salary values for the Chicago dataset.

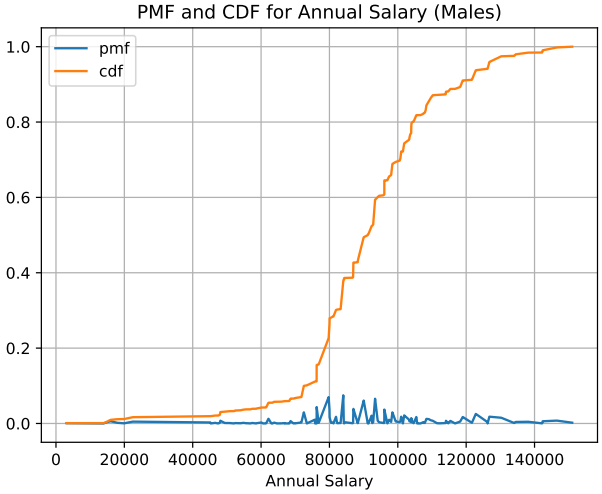
Lastly, we decided to plot the *probability mass function (PMF)* and the *cumulative distribution function (CDF)* of male and female employees, for each *Annual Salary* value. For the laymen, PMF gives the probability that a discrete variable – *Annual Salary* in our case – is exactly equal to a specific value; while CDF is the probability of the variable to be less than or equal to a specific value. The comparison between Figure 4.2(a) and Figure 4.2(b) shows, once again, that women are more likely than men to earn less, and specifically to have an income in the range of [0, 40K] dollars per year.

4.2.3 The "Glassdoor Method"

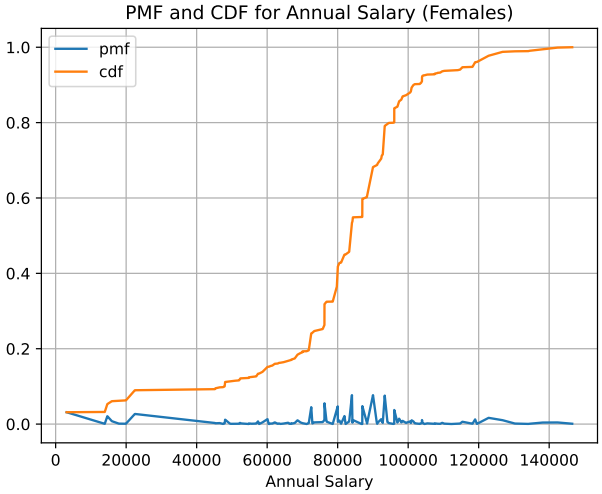
As already specified in Section 3.8, the point of reference for this method is the report published by Glassdoor in 2017 with the aim of helping HR practitioners in analyzing the internal gender pay gap of their companies [10]. Although the report provides a step-by-step guide for the statistical software R, we decided to use Python for our analysis, in order to better integrate the results with the ones from the other tools.

The first step of the analysis, after cleaning up the data and loading them, consists in the creation of a couple of attributes useful for the statistical analysis: *Log Annual Salary* and *Male*. The former is simply the natural logarithm of the annual salary of the employee (i.e. the logarithm to the base of the mathematical constant e, approximately equal to 2.71828), useful because it provides a simple interpretation of the regression results; the latter is a dummy indicator equal to 1 for males and 0 for females, which is the key variable of the analysis: if there is no gap, being male should not provide any advantage, and the coefficient of this variable in the regression should be equal to 0, otherwise its value would give us an estimate of the approximate percentage pay gap between men and women. It is worth to mention that the Glassdoor report also suggests to perform a discretization of age values of employees, grouping them into bins (25−, 25–34, 35–44, 45–54, 55+), but our dataset does not contain any information about the age of the employees.

The report suggests to look at the data before proceeding with the regressions, and it recommends to print a "summary table" displaying the basic statistical information about the dataset. Figure 4.3(a) shows the table related to the Chicago dataset, and it displays the variables *Annual Salary*, *Log Annual Salary*, and *Male* sample size (count), arithmetic mean, minimum and maximum values, and standard deviation (i.e. a measure of the average amount of variability in the dataset – calculated as the square root of the variance – which tells, on average, how far each value lies from the mean). Another useful visualization tool is the so called "pivot table",



(a)



(b)

Figure 4.2: Probability mass function and cumulative distribution function of male (a) and female (b) employees, for each $Annual\ Salary$ value of the Chicago dataset.

displayed in Figure 4.3(b), which provides a high-level summary of the overall difference in pay between men and women by showing the arithmetic mean of the *Annual Salary* attribute values for males and females, together with the number of observations (len) and the median values (i.e. the numeric values separating the higher half of the samples from the lower half). The pivot table is also useful to get a first estimate of the "unadjusted" pay gap:

men on average are paid $92,022.03 per year, while women on average earn $79,790.83 per year – an overall "unadjusted" pay gap of $12,231.20 (13.3% of male pay). Lastly, since we are also interested in the "adjusted" pay gap, it is important to look at the average salaries of men and women employed in the different job titles. Figure 4.3(c) shows the first 8 (out of 35) job titles in alphabetical order, displaying average salaries for men and women and sizes of the samples (i.e. number of men and women employed in the specific job title – information relevant to the problem of representation).

In order to estimate the gender pay gap, the reference linear regression model, as mentioned in Section 3.8, is:

$$LogAnnualSalary_i = \beta_1 Male_i + \beta_2 Controls_i + \epsilon_i$$

The report recommends to run three different models: the first with no controls at all, regressing salary only on the male-female gender dummy (and therefore calculating the approximate overall percentage pay gap between men and women – the "unadjusted" pay gap); the second with the addition of variables related to employee characteristics like highest education, years of experience, and performance evaluation scores; the third including all the possible controls (and finally estimating the "adjusted" pay gap). Due to the lack of attributes, we performed only two linear regressions: the first with no controls and the second including *Job Title*, *Department*, and *Status*.

The results are shown in Figure 4.4: a coefficient of 0.242 on the male-female dummy variable means there is approximately 24.2% "unadjusted" pay gap (therefore, men on average earn 24.2% more than women), but adding to the model all of the controls available in the data the coefficient value shrinks to 0.4% and becomes no longer statistically significant. In this case, we say there is no evidence of a systematic gender pay gap on an "adjusted" basis, after controlling for observable differences between male and female workers, and the big discrepancy between the coefficient values is due to the overrepresentation of men in higher-paying roles and their underrepresentation in lower-paying jobs.

4.2.4 FAIR-DB

FAIR-DB is a tool based on functional dependencies, as already described in Section 3.9, and it operates by following the workflow shown in Figure 3.4.

- **Data preparation and exploration**: this phase is mostly covered by Section 4.2.2. In addition to the preprocessing techniques applied before, we had to deal with the **discretization** of *Annual Salary*

	Annual Salary	Log Annual Salary	Male
count	20309.00	20309.00	20309.00
mean	89514.84	11.35	0.80
std	22067.19	0.42	0.40
min	3120.00	8.05	0.00
max	151026.00	11.93	1.00

(a)

	average	median	len
	Annual Salary	Annual Salary	Annual Salary
Gender			
female	79790.83	84054.00	4163.00
male	92022.03	91338.00	16146.00

(b)

		average	len
		Annual Salary	Annual Salary
Job Title	Gender		
ADMINISTRATIVE ASST II	female	67908.82	102.00
	male	61292.00	9.00
AVIATION SECURITY OFFICER	female	73178.00	42.00
	male	73126.63	149.00
CAPTAIN-EMT	female	146538.00	4.00
	male	147328.84	160.00
CONSTRUCTION LABORER	female	92352.00	54.00
	male	92352.00	337.00
DETENTION AIDE	female	71730.48	50.00
	male	69382.60	131.00
ELECTRICAL MECHANIC	female	104000.00	12.00
	male	104000.00	194.00
FIRE ENGINEER-EMT	female	113901.86	14.00
	male	114411.09	344.00
FIREFIGHTER	female	102326.00	3.00
	male	102630.77	217.00

(c)

Figure 4.3: Summary table (a), pivot table (b) and average salaries of men and women employed in the different job titles (c) for the Chicago dataset.

values, since numbers are not really useful in estimating correlations between attributes (functional dependencies may depend on really specific income values). We decided to create 2 interval levels (or bins) splitting *Annual Salary* values in $\leq$ 90K and $>$ 90K and generating a new *Annual Salary Bin* attribute to store this information, by following the approach presented in [5] by the authors of the tool. *Annual Salary Bin* represents our **target attribute**, while *Gender* is our **protected**

Dependent Variable: Log Annual Salary		
	(1)	(2)
Male	0.242	0.004
Job Title		0.115
Department		0.726
Status		0.362
Constant	11.155	11.070
Controls		
- Job Title	No	Yes
- Department	No	Yes
- Status	No	Yes
Observations	20309	20309
R^2	0.053	0.950

Figure 4.4: Regression results for the Chicago dataset.

attribute. The choice of 90K as threshold was made by looking at the *Annual Salary* values distribution, shown in Figure 4.1.

The histogram of Figure 4.5 shows instead the distribution of the annual salary of employees over their *Gender* attribute, and it is particularly useful because it provides a preliminary visual representation of the discrepancy between the number of male and female employees belonging to the same bin, highlighting the fact that, although the amount of people earning more than $90,000 is larger, there are many more women in the least profitable group.

Lastly, we performed a new **data reduction** operation by removing from the dataset the attributes *Name* and *Annual Salary*, not relevant anymore for our analysis, since the tool will make use of the *Annual Salary Bin* variable.

- **ACFD Discovery and filtering**: as specified in Section 3.9, this phase makes use of the *ACFD Discovery* algorithm, presented in [29]. The authors of the algorithm made available a compute capsule on Code Ocean[4], which works basically as a Web-based application, allowing the user to upload a CSV file (in our case, we exported and uploaded the modified Chicago dataset), set the values of the required parameters (*minimum support*, *minimum confidence*, *maximum antecedent size*), run the algorithm and download the results in the form of a text file. The software also allows the choice of different algorithm implementa-

[4]Available at: `https://codeocean.com/capsule/6146641/tree`.

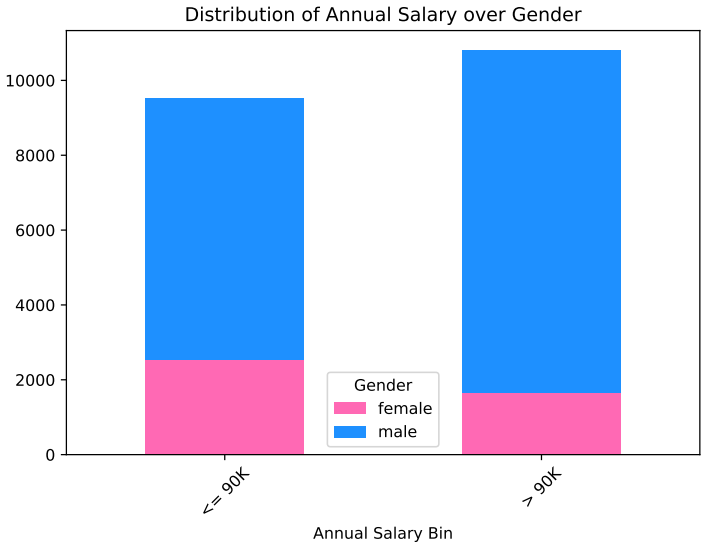


Figure 4.5: Distribution of the $Annual\ Salary$ *values for the Chicago dataset (2 bins).*

tions to be used to extract the dependencies but, as reported in the documentation, the default option FD-First-DFS-dfs is generally the fastest and there are no particular reasons, apart from performances, to choose one implementation over another.

Since our dataset does not contain a large amount of attributes, we decided to keep *maximum antecedent size* $= 2$ (meaning that at most 2 variables will appear in the LHS of the computed rules). For what concerns the confidence, its value is computed as the ratio between the frequency of the dependency over the frequency of the LHS of the rule, and therefore a *minimum confidence* $= 0.8$ seemed to be a reasonable threshold, since the lower the parameter, the more dependencies are generated at each round increasing the computational complexity and making more difficult the subsequent choice of the most interesting ACFDs. Finally, we had to deal with the choice of a proper minimum support, which is quite a delicate operation: if it is too high the risk is to lose information about small groups, if it is too low there could be too many dependencies to analyze. We set *minimum support* $= 100$, because it is a reasonably low number if compared to the total number of tuples in the dataset and to the average amount of tuples for each *Job Title* value ($\frac{20{,}309}{35} \simeq 580$).

The text file generated by *ACFD Discovery*, containing 714 rules, has to be filtered, since the dependencies detected may not involve

the protected attribute or the target attribute; there could also be dependencies in which some attribute values are not specified. The authors of [5] did not use the compute capsule of *ACFD Discovery*, and therefore were able to run the algorithm with an additional parameter, in order to discard the rules not containing the target attribute and its value. We balanced the gap by importing the text file, parsing it in order to extract every rule, and doing the same filtering operation a posteriori of *ACFD Discovery*. This operation resulted in a reduction in the number of dependencies from 714 to 145.

FAIR-DB then proceeds in filtering the rules by following 4 steps:

1. For each dependency, the LHS is separated from the RHS, and from both antecedent and consequent every couple "attribute - value" is stored.
2. For each rule, every couple "attribute - value" is checked, and dependencies with missing values are discarded (being AFDs but not ACFDs).
3. A dictionary (that is, an unordered and indexed data structure similar to a list) of the remaining ACFDs is generated. It is made of two fields: 'lhs' and 'rhs', and each field contains a list of one or more couples "attribute - value".
4. Since each rule of the dictionary contains the target attribute but not necessarily any protected attribute, the dependencies are furtherly parsed in order to satisfy both the criteria.

For the Chicago dataset, the filtering operation resulted in a reduction in the number of dependencies from 145 to 49. For each rule, the metrics *support*, *confidence*, *difference* and *p-Difference*, already introduced in Section 3.6 and Section 3.9, are computed, and the first occurrences are displayed in the form of a table, as shown in Figure 4.6.

- **ACFD selection**: in this phase FAIR-DB selects, among the filtered rules, the most "unethical", by looking at the computed metrics. It is worth to briefly recap the meanings behind the measures:
  - **Support**: it expresses the percentage of records in the dataset that verifies the dependency – the higher the value, the more tuples are involved.
  - **Confidence**: it shows how frequently the dependency is verified knowing that the antecedent is verified – the higher the value, the less approximate is the dependency.

```
Total number of tuples in df2:  49
                                                Rule  Supp  Conf  Diff  GenderDiff
0  {'lhs': {'Annual Salary Bin': '> 90K'},
    'rhs': {'Gender': 'male'}}                  0.45  0.85  0.05         NaN
1  {'lhs': {'Gender': 'female',
            'Salary or Hourly': 'Hourly'},
    'rhs': {'Annual Salary Bin': '<= 90K'}}     0.03  0.89  0.24        0.24
2  {'lhs': {'Annual Salary Bin': '> 90K',
            'Salary or Hourly': 'Hourly'},
    'rhs': {'Gender': 'male'}}                  0.07  0.94  0.13         NaN
3  {'lhs': {'Salary or Hourly': 'Salary',
            'Annual Salary Bin': '> 90K'},
    'rhs': {'Gender': 'male'}}                  0.38  0.83  0.04         NaN
4  {'lhs': {'Annual Salary Bin': '> 90K',
            'Gender': 'female'},
    'rhs': {'Salary or Hourly': 'Salary'}}      0.08  0.95  0.15        0.08
```

Figure 4.6: First 5 filtered dependencies with their metrics for the Chicago dataset (2 bins). NaN (Not a Number) means that the p-Difference has not been computed for the specific rule, since the protected attribute is not in the antecedent.

- **Difference**: it indicates how much a dependency is "unethical" – the higher the value, the more unfair is the dependency.
- **p-Difference**: it indicates how much the dependency shows bias paying attention to the specific value of a protected attribute – the higher the value, the more the rule is discriminatory with respect to the specific protected attribute value.

The selection of the most relevant rules takes place automatically, since the algorithm only keeps the dependencies with a difference parameter value higher than a minimum threshold imposed by the user. We decided to set *minimum difference* = 0.02, in order to keep the majority of the unfair dependencies. This operation resulted in a reduction in the number of dependencies from 49 to 10.

After that, the algorithm performs what the authors call **ACFD completion**. Given the selected ACFDs, the framework computes all the possible combinations for each rule over the protected attributes and the target attribute (performing a Cartesian product between the attributes values). Taking as an example ACFD n.0 of Figure 4.6:

$$AnnualSalaryBin = \text{`}> 90K\text{ '} \rightarrow Gender = \text{`}\ \text{male}\ \text{'}$$

we identify *Annual Salary Bin* as target attribute, whose possible values are $\leq$ 90K and $>$ 90K, and *Gender* as protected attribute,

whose possible values are male and female. Therefore, the possible combinations for the rule are:

$$AnnualSalaryBin = \text{`}> 90K\text{ '} \rightarrow Gender = \text{` male '}$$

$$AnnualSalaryBin = \text{`}\leq 90K\text{ '} \rightarrow Gender = \text{` male '}$$

$$AnnualSalaryBin = \text{`}> 90K\text{ '} \rightarrow Gender = \text{` female '}$$

$$AnnualSalaryBin = \text{`}\leq 90K\text{ '} \rightarrow Gender = \text{` female '}$$

For each newly generated dependency, the evaluation metrics are computed, and a new automatic selection is performed, by keeping the rules with difference greater than the respective minimum threshold. ACFD completion basically allows the user to study all the domain of the protected attributes and of the target class, and the combination computation brings to the surface also small groups that could not be studied otherwise. This operation, for the Chicago dataset, generated 36 dependencies (including the original 10), of which 18 with a difference above the threshold.

- **ACFD ranking**: the dependencies are ranked in descending order of support, difference, or mean, according to the user's choice. As already mentioned in Section 3.9, the support option highlights the pervasiveness of the rules, the difference highlights their unethical aspect, and the mean represents the best trade-off between difference and support. Because of that, we decided to adopt the mean as ordering criterion. The resulting table is finally printed, and Figure 4.7 shows the first 5 dependencies (the others are displayed to the user but are omitted here for the sake of brevity).

- **ACFD user selection and scoring**: this last phase requires interaction from the user, who has to select N interesting dependencies among the ones previously ranked. The system then computes a final scoring outline based on 3 measures:

  - **Cumulative support**: percentage of tuples of the dataset involved by the selected ACFDs – the higher the value, the more tuples are involved.
  - **Difference mean**: arithmetic mean of all the 'Difference' columns of the selected ACFDs. It indicates how much the dataset is ethical according to the chosen rules – the higher the value, the more unfair is the dataset.

```
Number of original CFDs: 10
Number of combinations rules:  36
Number of final rules found:  18
                                          Rule  Supp  Conf  Diff  GenderDiff  Mean
0   {'lhs': {'Annual Salary Bin': '> 90K'},
     'rhs': {'Gender': 'male'}}               0.45  0.85  0.05         NaN  0.25
20  {'lhs': {'Status': 'F',
             'Annual Salary Bin': '> 90K'},
     'rhs': {'Gender': 'male'}}               0.45  0.85  0.04         NaN  0.25
16  {'lhs': {'Annual Salary Bin': '> 90K',
             'Gender': 'male'},
     'rhs': {'Salary or Hourly': 'Salary'}}   0.38  0.85  0.06       -0.01  0.22
12  {'lhs': {'Salary or Hourly': 'Salary',
             'Annual Salary Bin': '> 90K'},
     'rhs': {'Gender': 'male'}}               0.38  0.83  0.04         NaN  0.21
7   {'lhs': {'Gender': 'female',
             'Salary or Hourly': 'Hourly'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.03  0.89  0.24        0.24  0.14
```

Figure 4.7: First 5 selected and ranked dependencies with their metrics for the Chicago dataset (2 bins).

- **Protected attribute difference mean**: for each protected attribute, arithmetic mean of the p-Difference measure over all the selected ACFDs. It indicates how much the dataset is ethical over the protected attribute according to the chosen rules – the higher the value, the more the dataset is discriminatory with respect to the specific protected attribute.

For our research, we selected all the dependencies in which the target attribute appears in the RHS of the rule ($N = 6$ out of 18). This choice is due to the fact that the authors did not specify any criteria or suggestion for the manual selection of the rules, and the algorithm seems to be oriented towards dependencies in which the target attribute is part of the consequent, in fact, as can be noticed in Figure 4.7, rules in which the target is in the LHS have (most of the time) a null p-Difference value, while the same statistical measure is never null when the target is in the RHS. We will furtherly discuss about the impact of this choice in Section **??**. The chosen ACFDs, together with the final scores, are displayed in Figure 4.8.

Among the chosen dependencies, we can detect a correspondency between rule 7 and rule 4: women paid on a hourly basis tend to earn less than \$90K, while men paid on a hourly basis tend to earn more than \$90K. These rules are the ones with the higher support (respectively 0.03 and 0.07), and rule 7

```
Number of tuples interested by the rules:  2310
Total number of tuples:  20309

Cumulative Support:  0.114
Difference Mean:  0.094
Gender Difference Mean:  0.094


Total number of ACFDs selected:  6
                                          Rule  Supp  Conf  Diff  GenderDiff  Mean
7   {'lhs': {'Gender': 'female',
             'Salary or Hourly': 'Hourly'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.03  0.89  0.24        0.24  0.14
27  {'lhs': {'Gender': 'female',
     'Department': 'AVIATION'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.01  0.92  0.13        0.13  0.07
4   {'lhs': {'Gender': 'male',
             'Salary or Hourly': 'Hourly'},
     'rhs': {'Annual Salary Bin': '> 90K'}}   0.07  0.41  0.06        0.06  0.06
29  {'lhs': {'Gender': 'male',
             'Department': 'OEMC'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.01  0.94  0.08        0.08  0.05
24  {'lhs': {'Gender': 'male',
             'Department': 'AVIATION'},
     'rhs': {'Annual Salary Bin': '> 90K'}}   0.01  0.23  0.03        0.03  0.02
30  {'lhs': {'Gender': 'female',
             'Department': 'OEMC'},
     'rhs': {'Annual Salary Bin': '> 90K'}}   0.00  0.18  0.03        0.03  0.02
```

Figure 4.8: Final selected rules and scores for the Chicago dataset (2 bins).

is the one with the highest difference value (0.24). Even though the support is very low – and therefore not many tuples out of the total are involved – it is important to point out that rules 27 and 24 suggest a discriminatory behavior in the subgroup of the AVIATION department, while rules 29 and 30 show that, for what concerns the OEMC department, men seem to be less paid than women.

As for the scoring measures, a cumulative support of 0.114 means that the 11.4% of the dataset is "problematic" (2,310 tuples out of 20,309), while difference mean and gender difference mean (equal because in our dataset *Gender* is the only protected attribute) have a value of 0.094 because of the gap between the difference metric values of the selected ACFDs (above 0.1 for rule 27, above 0.2 for rule 7, below 0.1 for the other rules).

To conclude, we can say that the dataset seems to be quite fair with respect to the group fairness criterion, that is the one on which the tool is based, but more than 10% of the tuples show some bias. Furthermore, the

representation problem is not taken into account, and therefore the tendency of women to be employed in less profitable jobs than men, displayed in Figure 4.2 and Figure 4.6 is ignored.

4.2.5 Ranking Facts

As specified in Section 3.10, Ranking Facts is primarily meant to be a Web-based application with the aim of discovering fairness in a dataset by making use of ranking and providing to the user a collection of visual widgets. However, because of the size of our dataset, we could not use the tool in the form of Web-based application, and we had to opt for the notebook version.

Before importing the dataset, we had to deal with a further **data transformation** process, in which we converted our categorical attributes (*Status*, *Job Title*, *Department*, *Salary or Hourly*) into numerical ones, since the tool can perform ranking only over them. A value of F for the *Status* attribute was therefore converted to 1, while P was converted to 0. The same holds for *Salary or Hourly* (Salary = 1, Hourly = 0), while for *Job Title* and *Department* numbers from 0 to 35 and from 0 to 20 respectively substited the original categorical values.

Once imported the dataset, the tool initially plots the distribution of some attributes specified by the user (in our case, we decided to plot the distributions of the *Annual Salary* and *Gender* values). While the related distribution graphs are not particularly meaningful, the heatmap generated subsequently provides us some preliminary information about the attributes correlations. As Figure 4.9 shows, there seems to be a significant correlation between *Job Title* and *Department*, but mostly important between *Status* and *Annual Salary*, *Status* and *Salary or Hourly*, and finally *Annual Salary* and *Salary or Hourly*, highlighting the fact that hourly paid and part-time employees earn generally less than salary paid and full-time employees, and most of the part-time workers are also being paid on a hourly basis.

The tool then requires the user to specify some attributes to be used for the ranking, together with their weights. The reference formula is:

$$f(x) = w_1 \times Attribute_1(x) + \ldots + w_n \times Attribute_n(x)$$

In our case, the attributes used are *Job Title*, *Department*, *Status*, and *Salary or Hourly*, because they are numerical and none of them is a protected or a target variable. By following the examples provided by the authors of the tool, we decided to set the weights all equal to 1, giving the same importance to each attribute.

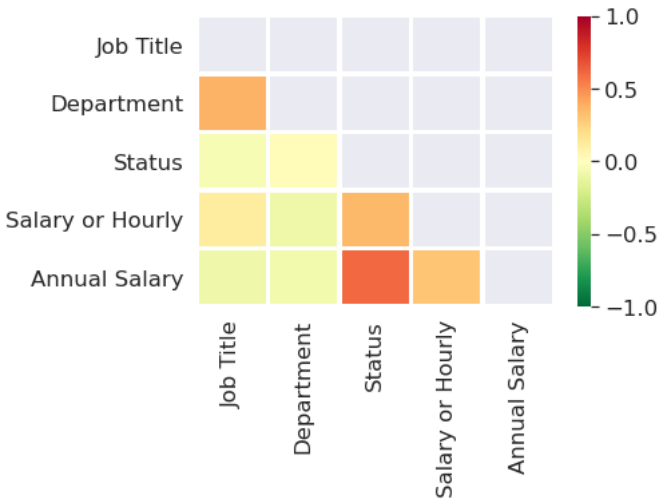


Figure 4.9: Heatmap showing attributes correlations for the Chicago dataset.

By following the widget description list provided in Section 3.10, we will now summarize our results.

- **Recipe** and **Ingredients**: the notebook version of the tool unfortunately does not provide any visual representation. As for the recipe, a couple of summary tables display some statistical measures (median, mean, minimum and maximum value) for the 4 attributes used in the ranking, respectively for the top-10 one and overall. These tables are not particularly useful, and neither is the information related to the ingredients, which tells us that the importance of each attribute used for the ranking is effectively equal to 1.

- **Stability**: this parameter explains whether the ranking methodology is robust on the specific dataset in use. Since an unstable ranking is one where slight changes to the data or to the methodology could lead to a significant change in the output, the label reports a stability score, as a single number that indicates the extent of the change required for the ranking to change. The stability of the ranking is quantified as the slope of the line that is fit to the score distribution, at the top-10 and overall. A score distribution is unstable if scores of items in adjacent ranks are close to each other ($|slope| \leq 0.25$), and so a very small change in scores will lead to a change in the ranking. The ranking used to analyze the Chicago dataset resulted to be unstable both at top-10 (stability at 0.23) and overall (stability at 0.0), and the score distribution is displayed in Figure 4.10.

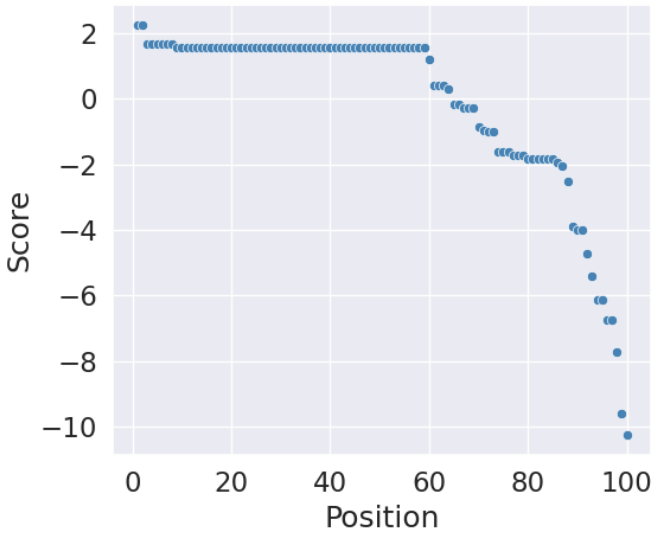


Figure 4.10: Score distribution of the ranking for the Chicago dataset.

- **Fairness**: it quantifies whether the ranked output exhibits statistical parity (group fairness) with respect to one or more protected attributes. The fairness measures adopted are all statistical tests in which the null hypothesis is that the ranking process is fair for the protected group, and whether a result is fair is determined by the computed p-value (a ranking is considered unfair when the p-value of the corresponding statistical test falls below 0.05).

  - **FA*IR**: the ranking resulted to be *fair for males*, with an approximate p-value of 0.94, and *fair for females*, with an approximate p-value of 0.23.
  - **Proportion**: the ranking resulted to be *fair for males*, with an approximate p-value of 0.94, and *fair for females*, with an approximate p-value of 0.35.
  - **Pairwise**: the ranking resulted to be *unfair for males*, with an approximate p-value of 0.0, and *fair for females*, with an approximate p-value of 0.99.

  The results seem to be oriented towards a fair dataset, however the pairwise measure reported an exceptional bias in favor of women, which is quite peculiar because with the previously adopted tools the trend was towards fairness but with a slight tendency in favor of men.

- **Diversity**: it shows diversity with respect to a set of demographic

categories of individuals, or a set of categorical attributes of other kinds of items, by displaying the proportion of each category in the top-10 ranked list and overall. Figure 4.11 shows the predominancy of the male group over the female one, highlighting again a problem of gender representation.

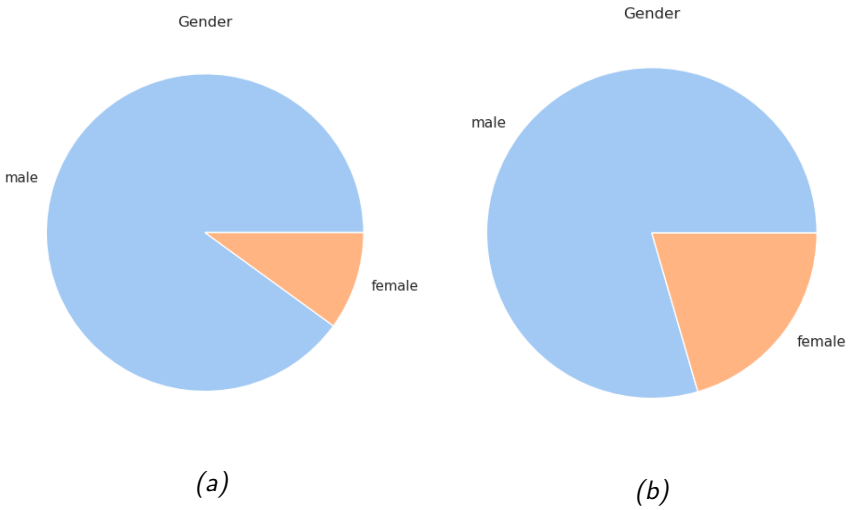


Figure 4.11: Gender diversity widget for the top-10 (a) and overall (b) rankings of the Chicago dataset.

4.3 Case Study 2: San Francisco

4.3.1 Dataset Description

The **San Francisco** dataset we considered includes 357,407 tuples and is made up of 10 attributes, briefly described as follows:

- *Employee Name*: full name of the employee in the form of "Name Surname".
- *Job Title*: categorical variable representing the job title of the employee (e.g. Firefighter). There are 2306 distinct values.
- *Base Pay*: numerical variable describing the annual regular pay for the employee.
- *Overtime Pay*: numerical variable describing the annual overtime pay for the employee.

- *Other Pay*: numerical variable describing other annual pay components for the employee.
- *Benefits*: numerical variable describing the amount of annual benefits for the employee.
- *Total Pay*: numerical variable describing the total annual salary of the employee, benefits excluded (*Base Pay* + *Overtime Pay* + *Other Pay*).
- *Total Pay + Benefits*: numerical variable describing the total annual salary of the employee, benefits included (*Base Pay* + *Overtime Pay* + *Other Pay* + *Benefits*).
- *Year*: numerical variable representing the year of reference (the dataset contains data related to the years 2011 to 2019).
- *Status*: binary categorical variable describing whether the employee is employed full-time (FT) or part-time (PT).

4.3.2 Data Preprocessing

As for the Chicago case, we operated some **data transformation** processes on the attributes of the San Francisco dataset. Specifically, the columns *Employee Name* and *Total Pay* were renamed respectively in *Name* and *Annual Salary*, and the attribute values for *Status* were transformed from FT and PT to simply F and P, in order to keep the algorithm used for the subsequent bias analysis as simple as possible and have a consistent structure across the datasets in use. We also operated a significant **data cleaning**, by filtering the tuples on the *Year* attribute, and keeping only the ones with *Year* = 2019, since they are the most recent and data covering a time span of one year are enough for the purpose of our research. This operation resulted in a reduction in the number of tuples from 357,407 to 44,525.

Again, since the original dataset does not contain any gender-related information, we relied on `gender-guesser` to infer the gender of the employees from their *First Name*, by splitting the *Name* attribute and saving the results in a newly generated *Gender* attribute. We obtained (out of the total of 44,525 tuples):

- unknown: 5,096 values.
- andy: 1,975 values.
- male: 20,636 values.

- female: 14,283 values.
- mostly_male: 1,153 values.
- mostly_female: 1,382 values.

We then removed, as we did for the Chicago dataset, tuples related to unknown and androgynous names, and we assumed mostly male names to be effectively related to males and mostly female names to be effectively related to females, and therefore we got 21,789 male values and 15,665 female values as a result of this **data cleaning** process.

We also operated **data reduction** by removing the *Base Pay*, *Overtime Pay*, *Other Pay*, *Benefits*, *Total Pay & Benefits*, *Year*, and *First Name* columns, since the information concerning the year became no longer useful and for the purpose of this research we are only interested in the total annual salary of employees.

Finally, we performed a last **data cleaning** process by removing job titles with less than 100 occurrences.

Our final preprocessed dataset includes 22,996 tuples, of which 13,688 males and 9,308 females, and with 81 distinct *Job Title* values.

Figure 4.1 shows the *Annual Salary* values distribution, from which we can observe that the lowest paid jobs are the most popular, and the largest group of employees earns less than $50,000.

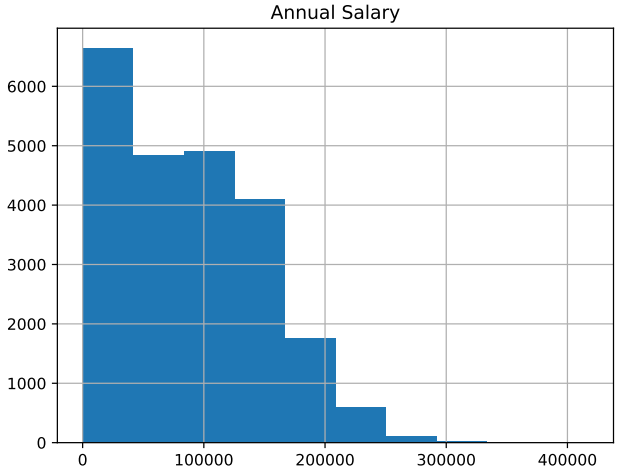


Figure 4.12: Distribution of the Annual Salary values for the San Francisco dataset.

Lastly, Figure 4.2(a) and Figure 4.2(b) display probability mass function (PMF) and cumulative distribution function (CDF) of male and female

employees, for each *Annual Salary* value. PMF is not particularly relevant in this case because of its constant value, but CDF shows that the higher-paying jobs are done almost exclusively by men (even though the graphs look quite similar, the x-axis scale is different and it extends up to $400,000 only on the male one). Again, women are more likely than men to earn less, since the curve on the female graph increases more rapidly compared to the male one (representing a higher probability to have a lower income).

4.3.3 The "Glassdoor Method"

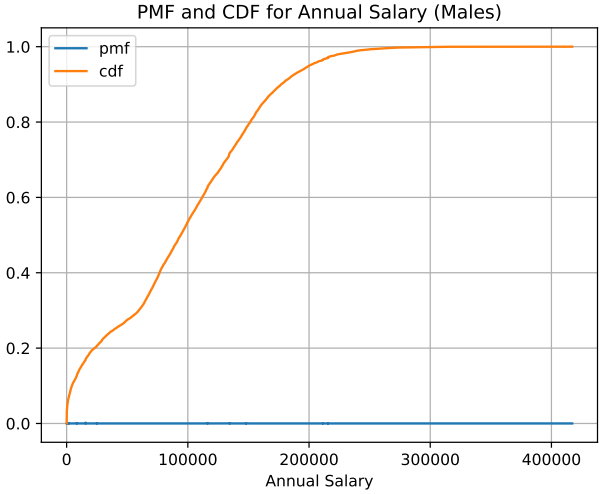
As for the Chicago case, we started by creating the *Log Annual Salary* and *Male* attributes, useful for the statistical analysis. After that, we proceeded in printing the summary and pivot tables, shown respectively in Figure 4.14(a) and Figure 4.14(b). Since men on average are paid $94,370.73 per year, while women on average earn $75,841.50 per year, we got a first estimate of the overall "unadjusted" pay gap of $18,529.23 (19.6% of male pay). Figure 4.14(c) shows the first 8 (out of 81) job titles in alphabetical order, displaying average salaries for men and women and sizes of the samples.

Again, because of the lack of attributes, we could perform only two linear regressions: the first with no controls and the second including *Job Title* and *Status*.

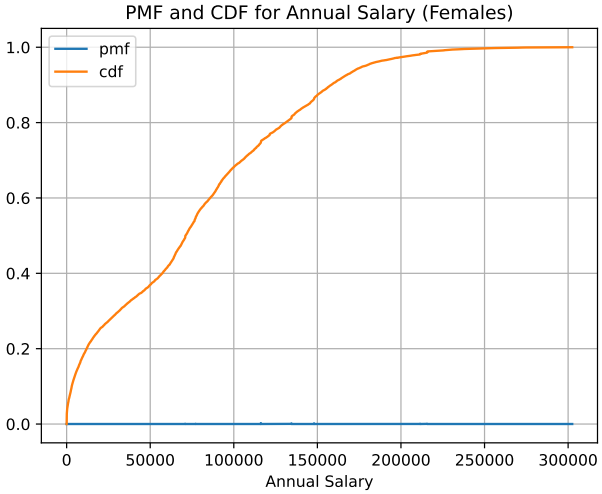
The results are shown in Figure 4.15: a coefficient of 0.304 on the male-female dummy variable means there is approximately 30.4% "unadjusted" pay gap (therefore, men on average earn 30.4% more than women), but adding to the model all of the controls available in the data the coefficient value shrinks to -0.5% and becomes no longer statistically significant. We can conclude that also in this case there is no evidence of a systematic gender pay gap on an "adjusted" basis, after controlling for observable differences between male and female workers, and again the big discrepancy between the coefficient values is due to the overrepresentation of men in higher-paying roles and their underrepresentation in lower-paying jobs.

4.3.4 FAIR-DB

- **Data preparation and exploration**: in addition to the preprocessing techniques applied in Section 4.3.2, we had once again to deal with the **discretization** of *Annual Salary* values and the creation of a new *Annual Salary Bin* attribute. By looking at the graph displayed in Figure 4.1, we decided to keep 90K as threshold value, and therefore to use the same bins ($\leq$ 90K and $>$ 90K) used for the Chicago case.



(a)



(b)

Figure 4.13: Probability mass function and cumulative distribution function of male (a) and female (b) employees, for each $Annual\ Salary$ *value of the San Francisco dataset.*

The histogram of Figure 4.16 shows the distribution of the annual salary of employees over their *Gender* attribute. As we can notice, despite the number of male and female employees belonging to the $\leq$ 90K bin is comparable, almost $\frac{2}{3}$ of the employees belonging to the $>$ 90K bin are men.

	Annual Salary	Log Annual Salary	Male
count	22996.00	22996.00	22996.00
mean	86870.73	10.62	0.60
std	62399.93	2.02	0.49
min	0.01	-4.61	0.00
max	417152.63	12.94	1.00

(a)

	average	median	len
	Annual Salary	Annual Salary	Annual Salary
Gender			
female	75841.50	71100.98	9308.00
male	94370.73	94133.26	13688.00

(b)

		average	len
		Annual Salary	Annual Salary
Job Title	Gender		
Administrative Analyst	female	83282.16	89.00
	male	81709.54	102.00
Assistant Engineer	female	99238.61	63.00
	male	103185.83	100.00
Assoc Engineer	female	123409.06	46.00
	male	125359.07	123.00
Attorney (Civil/Criminal)	female	154756.46	190.00
	male	162993.18	199.00
Automotive Mechanic	female	96526.73	1.00
	male	98182.72	159.00
Automotive Service Worker	female	94050.96	3.00
	male	77288.95	123.00
Behavioral Health Clinician	female	77599.23	123.00
	male	81235.70	33.00
Camp Assistant	female	3897.82	60.00
	male	3949.49	48.00

(c)

Figure 4.14: Summary table (a), pivot table (b) and average salaries of men and women employed in the different job titles (c) for the San Francisco dataset.

Lastly, we performed a new **data reduction** operation by removing from the dataset the attributes *Name* and *Annual Salary*, since the tool will make use of the *Annual Salary Bin* variable.

- **ACFD Discovery and filtering**: we used the compute capsule of the *ACFD Discovery* algorithm as we did for the Chicago case, and we decided to keep the same parameter values: *maximum antecedent size* = 2

Dependent Variable: Log Annual Salary		
	(1)	(2)
Male	0.304	-0.050
Job Title		0.108
Status		0.349
Constant	10.443	11.618
Controls		
- Job Title	No	Yes
- Status	No	Yes
Observations	22996	22996
R^2	0.005	0.476

Figure 4.15: Regression results for the San Francisco dataset.

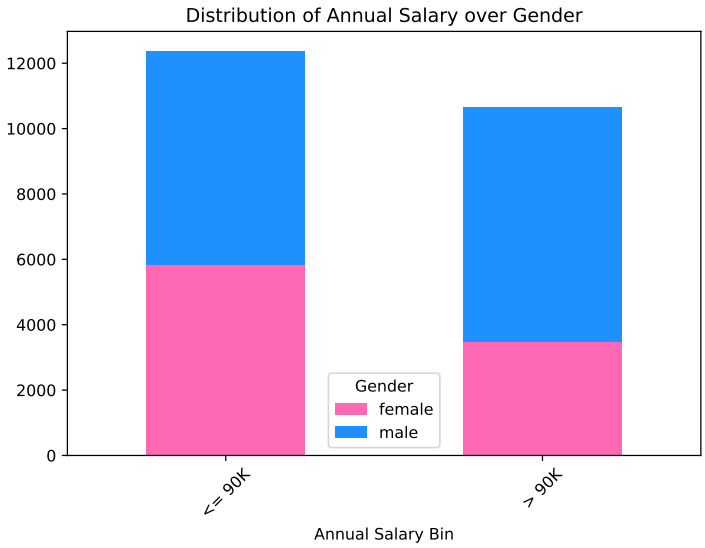


Figure 4.16: Distribution of the Annual Salary values for the San Francisco dataset (2 bins).

because the dataset does not contain many attributes, *minimum confidence* $= 0.8$ to generate a not too high number of dependencies, and *minimum support* $= 100$ because it is a reasonably low number if compared to the total number of tuples in the dataset and to the average amount of tuples for each *Job Title* value ($\frac{22{,}996}{81} \simeq 284$).

The text file generated by *ACFD Discovery*, containing 586 dependencies, was then imported, parsed, and filtered in order to discard the rules not containing the target attribute and its value. This operation

resulted in a reduction in the number of dependencies from 586 to 220. The automatic filtering operations performed by FAIR-DB furtherly reduced this number from 220 to 61.

- **ACFD selection**: as for the previous phase, we decided to maintain the same value for the only parameter required by ACFD selection, and therefore we set *minimum difference* = 0.02 in order to keep the majority of the unfair dependencies. The first automatic selection resulted in a reduction in the number of dependencies from 61 to 10. The subsequent ACFD completion and further selection generated 36 dependencies (including the original 10), of which 16 with a difference above the threshold.
- **ACFD ranking**: Figure 4.17 shows the first 5 dependencies, ranked adopting the mean as ordering criterion (the others are displayed to the user but are omitted here for the sake of brevity).

```
Number of original CFDs: 10
Number of combinations rules:  36
Number of final rules found:  16
                                        Rule  Supp  Conf  Diff  GenderDiff  Mean
4   {'lhs': {'Annual Salary Bin': '> 90K',
             'Gender': 'male'},
     'rhs': {'Status': 'F'}}                  0.30  0.96  0.36        0.04  0.33
5   {'lhs': {'Annual Salary Bin': '> 90K',
             'Gender': 'female'},
     'rhs': {'Status': 'F'}}                  0.13  0.84  0.24       -0.08  0.18
1   {'lhs': {'Gender': 'male',
             'Status': 'P'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.19  0.93  0.03        0.03  0.11
32  {'lhs': {'Annual Salary Bin': '> 90K',
             'Job Title': 'Transit Operator'},
     'rhs': {'Gender': 'male'}}               0.03  0.86  0.06         NaN  0.05
19  {'lhs': {'Gender': 'female',
             'Job Title': 'Assoc Engineer'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.00  0.17  0.09        0.09  0.05
```

Figure 4.17: First 5 selected and ranked dependencies with their metrics for the San Francisco dataset (2 bins).

- **ACFD user selection and scoring**: for the reasons already specified in Section 4.2.4, we selected all the dependencies in which the target attribute appears in the RHS of the rule ($N = 10$ out of 16). The chosen ACFDs, together with the final scores, are displayed in Figure 4.18.

Among the chosen dependencies, we can detect a correspondency between

```
Number of tuples interested by the rules:  5596
Total number of tuples:  22996

Cumulative Support:  0.243
Difference Mean:  0.037
Gender - Difference Mean:  0.037


Total number of ACFDs selected:  10
                                        Rule  Supp  Conf  Diff  GenderDiff  Mean
1   {'lhs': {'Gender': 'male', 'Status': 'P'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.19  0.93  0.03        0.03  0.11
19  {'lhs': {'Gender': 'female',
             'Job Title': 'Assoc Engineer'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.00  0.17  0.09        0.09  0.05
25  {'lhs': {'Gender': 'male',
             'Job Title': 'HSA Sr Eligibility Worker'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.00  0.92  0.06        0.06  0.03
2   {'lhs': {'Gender': 'female',
             'Status': 'P'},
     'rhs': {'Annual Salary Bin': '> 90K'}}   0.02  0.12  0.03        0.03  0.03
16  {'lhs': {'Gender': 'male',
             'Job Title': 'Assoc Engineer'},
     'rhs': {'Annual Salary Bin': '> 90K'}}   0.01  0.95  0.03        0.03  0.02
23  {'lhs': {'Gender': 'female',
             'Job Title': 'Assistant Engineer'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.00  0.16  0.04        0.04  0.02
12  {'lhs': {'Gender': 'male',
             'Job Title': 'Registered Nurse'},
     'rhs': {'Annual Salary Bin': '> 90K'}}   0.01  0.89  0.02        0.02  0.02
11  {'lhs': {'Gender': 'female',
             'Job Title': 'Parking Control Officer'},
     'rhs': {'Annual Salary Bin': '<= 90K'}}  0.00  0.82  0.02        0.02  0.01
20  {'lhs': {'Gender': 'male',
             'Job Title': 'Assistant Engineer'},
     'rhs': {'Annual Salary Bin': '> 90K'}}   0.00  0.90  0.02        0.02  0.01
26  {'lhs': {'Gender': 'female',
             'Job Title': 'HSA Sr Eligibility Worker'},
     'rhs': {'Annual Salary Bin': '> 90K'}}   0.00  0.16  0.02        0.02  0.01
```

Figure 4.18: Final selected rules and scores for the San Francisco dataset (2 bins).

rule 1 and rule 2: apparently, female part-time workers tend to earn more than \$90K, while male part-time workers tend to earn less than \$90K. These rules are the ones with the higher support (respectively 0.19 and 0.02). Even though the support is very low – and therefore not many tuples out of the total are involved – it is important to point out that rules 19 and 16 suggest a discriminatory behavior in the subgroup of the Assoc Engineer job title in favor of men (with the highest difference value for rule 19), and the same

holds for rules 23 and 20 in the subgroup of assistant engineers. Although the complementary rules have not been selected because of a low difference value, rule 12 shows that male registered nurses earn more than the threshold, and rule 11 shows that, for what concerns parking control officers, women tend to have an income lower than $90K. Finally, rules 25 and 26 show that, for the HSA Sr Eligibility Worker job title, men seem to be less paid than women.

As for the scoring measures, a cumulative support of 0.243 means that the 24.3% of the dataset is "problematic" (5,596 tuples out of 22,996), while difference mean and gender difference mean have a value of 0.037 because each rule has a quite low value for the difference metric (below 0.1).

To conclude, we can say that the dataset seems to be quite fair with respect to the group fairness criterion, because even if almost 25% of the tuples seem to be biased, each rule has a low value both for the difference metric – which, as already specified, determines the "unfairness level" – and for the support one – indicating a low percentage of tuples involved. The most impacting dependency is indeed the one regarding male part-time employees, with a support value of 0.19. Furthermore, it is interesting to notice that for traditionally higher-paying jobs – in this case two branches of engineering – men seem to have an economic advantage over women, while the opposite condition occurs only in situations in which the female presence is tipically much more numerous than the male one – that is, in our scenario, the HSA Sr Eligibility Worker job title and the part-time worker status.

4.3.5 Ranking Facts

As for the Chicago case, because of the size of our dataset, we could not use Ranking Facts in the form of Web-based application, and we had to opt for the notebook version. Before importing the dataset, we operated a **data transformation** process, in which the categorical attributes *Job Title* and *Status* were converted into numerical ones.

Figure 4.19 shows the heatmap related to our dataset, and it suggests a really strong correlation between the attributes *Status* and *Annual Salary*, highlighting the fact that part-time employees tend to earn less than full-time employees.

Because of the lack of attributes, we could only use *Job Title* and *Status* as ranking parameters, both with a weight of 1 as done in the Chicago case and by following the examples provided by the authors.

The following list summarizes our results by following the widget description list provided in Section 3.10:

- **Recipe** and **Ingredients**: as for the Chicago case, these widgets did

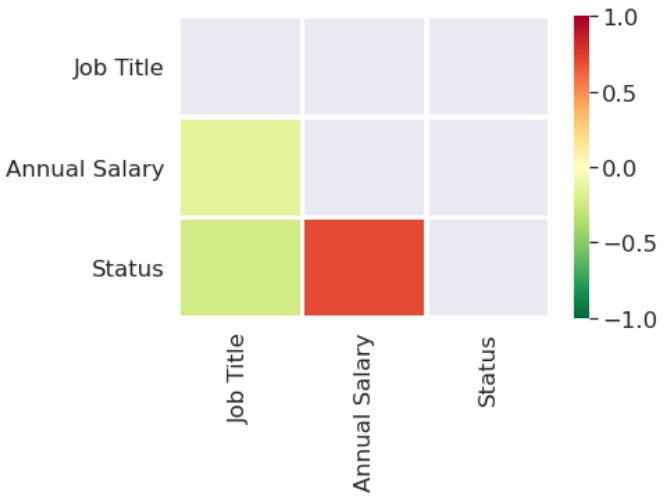


Figure 4.19: Heatmap showing attributes correlations for the San Francisco dataset.

not provide us any particularly useful information. For the sake of completeness, we just report that the importance of each attribute used for the ranking is effectively equal to 1.

- **Stability**: as for the Chicago case, our ranking resulted to be unstable both at top-10 (stability at 0.23) and overall (stability at 0.0). The score distribution is displayed in Figure 4.20.

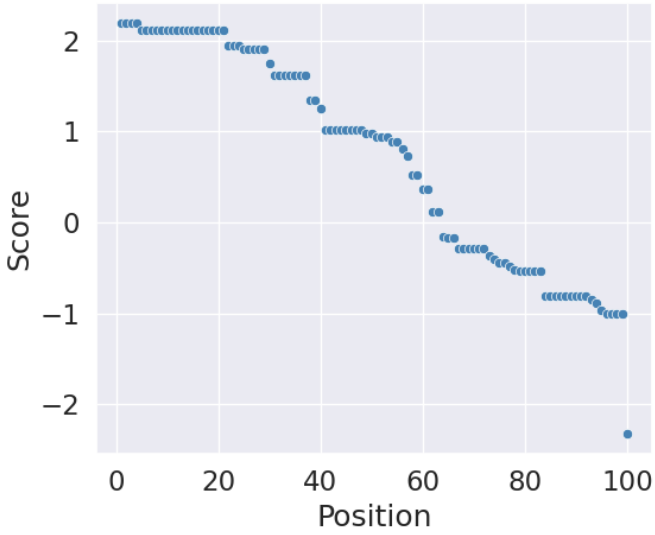


Figure 4.20: Score distribution of the ranking for the San Francisco dataset.

- **Fairness**: recalling the fact that, for the statistical measures adopted,

a ranking is considered unfair when the p-value of the corresponding test falls below 0.05, we will now recapitulate our fairness results:

  - **FA*IR**: the ranking resulted to be *fair for males*, with an approximate p-value of 1.0, and *unfair for females*, with an approximate p-value of 0.0.
  - **Proportion**: the ranking resulted to be *fair for males*, with an approximate p-value of 1.0, and *unfair for females*, with an approximate p-value of 0.0.
  - **Pairwise**: the ranking resulted to be *fair for males*, with an approximate p-value of 1.0, and *unfair for females*, with an approximate p-value of 0.0.

- **Diversity**: Figure 4.21 shows the predominancy of the male group over the female one, especially in the top-10 of the ranking, highlighting again a problem of gender representation.

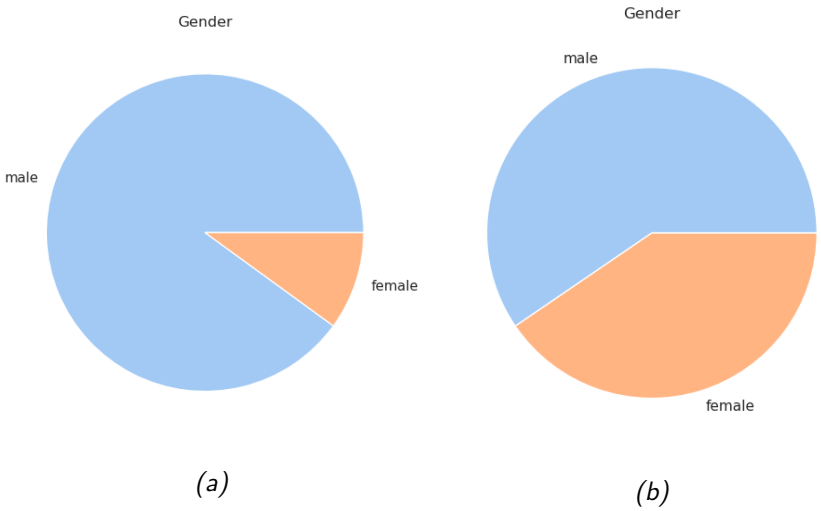


Figure 4.21: Gender diversity widget for the top-10 (a) and overall (b) rankings of the San Francisco dataset.

4.4 Other Design Choices

The aim of this section is to document in a technical way some experiments made on the datasets previously analyzed – Chicago and San Francisco – in which different design decisions were taken. For this reason, the socio-ethical impact of these choices, as well as the impact of other ones not mentioned in this section, will not be discussed here but in Section **??**.

4.4.1 Part-Time Employees Removal

One of the first design choices we had to deal with was concerning part-time employees. The Glassdooor report [10] indeed recommends to only include full-time employees in the gender pay gap analysis (or in case of equal amounts of full-time and part-time employees to conduct two separate analyses) because of the big differences between full-time and part-time workers in the labor market. We initially decided to exclude part-time employees from the datasets, by removing their tuples and subsequently also the *Status* attribute. This choice resulted to be penalizing in both our cases for different reasons:

- **Chicago**: for the preprocessed dataset, the number of male employees decreased from 16,146 to 15,880 (−266), while the number of female employees decreased from 4,163 to 3,790 (−373). Even though the reduction in the number of tuples is not excessive, most of the records removed are related to women, already underrepresented when compared to the number of men.

- **San Francisco**: for the preprocessed dataset, the number of male employees decreased from 13,688 to 9,082 (−4,606), while the number of female employees decreased from 9,308 to 4,696 (−4,612). Even if the number of male and female employees removed is similar, we found the removal of 9,308 tuples from the dataset to have too much impact on the dataset itself.

Furthermore, it is important to notice that the Glassdoor report is meant to be a guide for HR practitioners in analyzing the internal gender gap of a company, while we are performing our analysis not on a company but on public employees of different sectors. Lastly, the removal of the *Status* variable furtherly penalizes datasets already characterized by a low number of attributes, making it more difficult to get concrete results from the tools. Because of these reasons, we decided to include again part-time employees in the datasets and to proceed in our analysis as discussed before.

4.4.2 FAIR-DB: Discretization Using More Bins

For both the Chicago and the San Francisco datasets, we decided to conduct a FAIR-DB analysis using more than just 2 bins. Therefore, we split the *Annual Salary* values in 8 different interval levels (the K specification is implied): 0–39, 40–59, 60–79, 80–99, 100–119, 120–139, 140+.

We will now briefly summarize our results, without going too much into the details of the algorithm but describing the most relevant differences in comparison with the 2 bins analysis.

- **Chicago**: the preprocessed dataset is the same used for the 2 bins analysis, with 20,309 tuples of which 16,146 males and 4,163 females. The histogram of Figure 4.22 shows the distribution of the annual salary of employees over their *Gender* attribute. As we can notice, most women are concentrated in the 80–99 bin, which is the most numerous overall, but the proportion between men and women in lower bins is unbalanced: more than 50% of the employees in the 0–39 bin are females, even though women represent about $\frac{1}{5}$ of the population of the dataset.

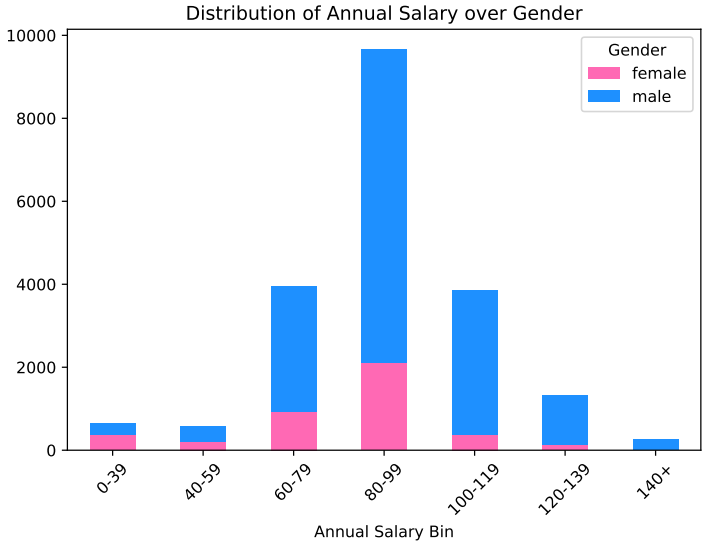


Figure 4.22: Distribution of the Annual Salary values for the Chicago dataset (8 bins).

Since the dataset is the same used for the 2 bins analysis, we ran the *ACFD Discovery* algorithm keeping the same parameter values: *maximum antecedent size* = 2, *minimum confidence* = 0.8, and *minimum support* = 100. The algorithm generated 759 dependencies, and the number shrinked to 193 when we filtered the results keeping only the rules containing the target attribute and its value. The automatic filtering operations performed by FAIR-DB furtherly reduced this number from 193 to 64.

We decided to keep *minimum difference* = 0.02 during the ACFD selection phase, in order to be able to compare the results of the two

analyses and evaluate the impact of the different discretization processes performed. The first automatic selection resulted in a reduction in the number of dependencies from 64 to 28. The subsequent ACFD completion and further selection generated 172 dependencies (including the original 28), of which 66 with a difference above the threshold.

Lastly, we applied the same criterion used in the previous cases for manually choosing the rules (target attribute in the RHS). Unfortunately, we only got $N = 2$ out of 66 dependencies. The chosen ACFDs, together with the final scores, are displayed in Figure 4.23.

```
Number of tuples interested by the rules:  86
Total number of tuples:  20309

Cumulative Support:  0.004
Difference Mean:  0.124
Gender - Difference Mean:  0.124


Total number of ACFDs selected:  2
                                        Rule  Supp  Conf  Diff  GenderDiff  Mean
156 {'lhs': {'Gender': 'male',
             'Job Title': 'ADMINISTRATIVE ASST II'},
     'rhs': {'Annual Salary Bin': '40-59'}}  0.00  0.44  0.23        0.23  0.11
161 {'lhs': {'Gender': 'female',
             'Job Title': 'ADMINISTRATIVE ASST II'},
     'rhs': {'Annual Salary Bin': '60-79'}}  0.00  0.80  0.02        0.02  0.01
```

Figure 4.23: Final selected rules and scores for the Chicago dataset (8 bins).

As we can see, both rules refer to the ADMINISTRATIVE ASST II job title, and they suggest a discriminatory behavior in favor of women, who seem to be more paid for this specific role. This situation did not show up in the 2 bins analysis, because even if apparently there is a gap between salaries of men and women, all the incomes fall below the threshold of \$90K, and therefore no dependencies could have been generated. On the other side, the discriminatory behaviors detected in the 2 bins analysis, related to the AVIATION and OEMC departments, and to female employees paid on a hourly basis, were not detected here, presumibly because the annual salary values reside all in the 80–99 bin.

- **San Francisco**: the preprocessed dataset is the same used for the 2 bins analysis, with 22,996 tuples of which 13,688 males and 9,308 females. The histogram of Figure 4.24 shows the distribution of the annual salary of employees over their *Gender* attribute. We can see

that the ratio of the number of men to women tends to increase: in lower bins there is a balanced proportion of male and female employees, while for higher-paying jobs men are much more numerous than women.

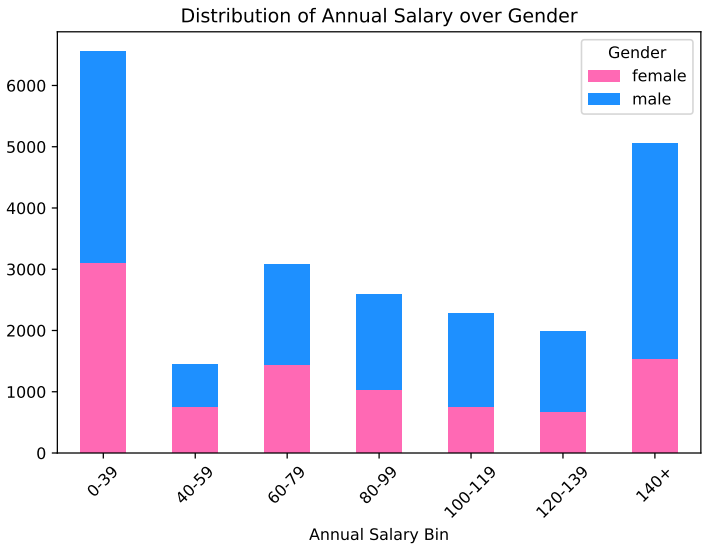


Figure 4.24: Distribution of the Annual Salary values for the San Francisco dataset (8 bins).

Again, we kept *maximum antecedent size* = 2, *minimum confidence* = 0.8, and *minimum support* = 100 for *ACFD Discovery*. The algorithm generated 465 dependencies, and the number shrinked to 144 when we filtered the results keeping only the rules containing the target attribute and its value. The automatic filtering operations performed by FAIR-DB furtherly reduced this number from 144 to 40.

By keeping *minimum difference* = 0.02, the first automatic ACFD selection resulted in a reduction in the number of dependencies from 40 to 20. The subsequent ACFD completion and further selection generated 154 dependencies (including the original 20), of which 62 with a difference above the threshold.

At the end, we got $N = 6$ out of 62 dependencies in the ACFD user selection phase. The chosen ACFDs, together with the final scores, are displayed in Figure 4.25.

The 6 final rules obtained are quite peculiar because they refer just to 2 different job titles. For what concerns junior clerks, men seem to be less paid then women. For the Engineer job title instead, women tend to earn less, and their incomes vary widely, since they fall within

```
Number of tuples interested by the rules:  227
Total number of tuples:  22996

Cumulative Support:  0.010
Difference Mean:  0.037
Gender - Difference Mean:  0.037


Total number of ACFDs selected:  6
                                          Rule  Supp  Conf  Diff  GenderDiff  Mean
51  {'lhs': {'Gender': 'female',
             'Job Title': 'Engineer'},
     'rhs': {'Annual Salary Bin': '120-139'}}  0.00  0.14  0.05        0.05  0.03
40  {'lhs': {'Gender': 'female',
             'Job Title': 'Junior Clerk'},
     'rhs': {'Annual Salary Bin': '40-59'}}    0.00  0.15  0.04        0.04  0.02
32  {'lhs': {'Gender': 'male',
             'Job Title': 'Junior Clerk'},
     'rhs': {'Annual Salary Bin': '0-39'}}     0.00  0.92  0.04        0.04  0.02
43  {'lhs': {'Gender': 'male',
             'Job Title': 'Engineer'},
     'rhs': {'Annual Salary Bin': '140+'}}     0.00  0.89  0.03        0.03  0.02
53  {'lhs': {'Gender': 'female',
             'Job Title': 'Engineer'},
     'rhs': {'Annual Salary Bin': '0-39'}}     0.00  0.06  0.04        0.04  0.02
54  {'lhs': {'Gender': 'female',
             'Job Title': 'Engineer'},
     'rhs': {'Annual Salary Bin': '40-59'}}    0.00  0.03  0.02        0.02  0.01
```

Figure 4.25: Final selected rules and scores for the San Francisco dataset (8 bins).

3 different interval levels (0–39, 40–59, 120–139), while male engineers are paid more than \$140K. As for the Chicago case, the 2 bins analysis could not detect any of these discriminatory behaviors, because in the former case the salary values are all lower than the threshold, while in the latter case the presence of female engineers earning between \$120K and \$139K "balanced" the gap, eluding the algorithm. For the same reasons, none of the discriminatory behaviors detected through the 2 bins analysis has been observed here.

4.4.3 FAIR-DB: Choice of Different Dependencies

As already specified, the ACFD user selection phase of the FAIR-DB framework is a delicate step, because the user needs to manually select N among all the dependencies for the subsequent scoring and calculation of the metrics. In Section 4.2.4 we clarified the reasons why we decided to always keep rules in which the target attribute is in the RHS. However, during our first experi-

ments, we tried to select all the dependencies, in order to check the impact of the user choices on the final results. The experiments were conducted on the same preprocessed datasets, and using the same parameter values; the only difference is indeed the selection of every rule instead of just some of them. The following results refer to the 8 bins analysis:

- **Chicago**: by selecting all the 66 dependencies, we got a cumulative support of 0.856, meaning that 85,6% of the dataset is "problematic" (17,384 tuples out of 20,309). The difference mean value is 0.153, while the gender difference mean is equal to 0.071. The measures have different values because for most of the rules (of which all with the target attribute in the LHS) the p-Difference is null (NaN), while the difference is not.
- **San Francisco**: by selecting all the 62 dependencies, we got a cumulative support of 0.925, meaning that 92,5% of the dataset is "problematic" (21,279 tuples out of 22,996). The difference mean value is 0.153, while the gender difference mean is equal to 0.003.

We can conclude that the user choices in this phase strongly impact the final outcomes, since a dataset results to be fair or not depending on the selected rules. In both our cases, the gap is huge: for Chicago the percentage of "problematic" tuples shifted from 0.4% (as we can see from the cumulative support in Figure 4.23) to 85,6%, while for San Francisco the same percentage shifted from 1% (Figure 4.25) to 92,5%.

4.4.4 Grouping of Job Titles

Another recommendation from the Glassdoor report [10] is the one of grouping together similar job titles, because having too many unique roles with just a few workers in each could make the analysis less reliable. Therefore, we decided to test our tools on a slightly different version of the Chicago dataset, in which the data preprocessing phase consists of one more step: a **generalization** of the *Job Title* values. We opted for the Chicago dataset instead of the San Francisco one mostly because a certain degree of precision is required in classifying job titles, and since the classification has to be performed manually we preferred to deal with 35 distinct *Job Title* values rather than 81. In order to group job titles in a sensible way, we relied on a document published by the Equality Commission for Northern Ireland in 2013 [18]. Even though the document refers to Northern Ireland, it provides an exhaustive list of thousands of different job titles, together with a few introductory sections on how to use the index, and we found useful to rely

on these guidelines in grouping the job titles of our dataset. The document, based on the Standard Occupational Classification 2010 (SOC2010) – a common classification of occupational information for the UK – distinguishes between 9 different major groups:

1. Managers and senior officials
2. Professional occupations
3. Associate professional and technical occupations
4. Administrative and secretarial occupations
5. Skilled trades occupations
6. Personal service occupations
7. Sales and customer service occupations
8. Process, plant and machine operatives
9. Elementary occupations

The index then provides correspondencies between each job title and the major group to which it belongs. Figure 4.26 shows the correspondencies between our job titles and the related major groups, together with the index entries of reference.

4.4.5 Voluntary Introduction of Bias

```
‘ADMINISTRATIVE ASST II’:            4 (Assistant, administrative)
‘AVIATION SECURITY OFFICER’:         9 (Officer, security)
‘CAPTAIN-EMT’:                       1 (Captain, fire)
‘CONSTRUCTION LABORER’:              9 (Worker, construction)
‘DETENTION AIDE’:                    6 (Aide, ward)
‘ELECTRICAL MECHANIC’:               5 (Mechanic, electrical)
‘FIRE ENGINEER-EMT’:                 3 (Engineer, fire)
‘FIREFIGHTER’:                       3 (Firefighter)
‘FIREFIGHTER-EMT’:                   3 (Firefighter)
‘FIREFIGHTER-EMT (RECRUIT)’:         3 (Firefighter)
‘FIREFIGHTER/PARAMEDIC’:             3 (Paramedic-ECP)
‘FOSTER GRANDPARENT’:                6 (Parent, foster)
‘GENERAL LABORER - DSS’:             9 (Operative, cleansing (street cleaning))
‘HOISTING ENGINEER’:                 5 (Engineer, construction)
‘LIBRARIAN I’:                       2 (Librarian)
‘LIBRARY PAGE’:                      4 (Assistant (library))
‘LIEUTENANT’:                        1 (Lieutenant)
‘LIEUTENANT-EMT’:                    1 (Lieutenant)
‘MACHINIST (AUTOMOTIVE)’:            8 (Machinist (garage))
‘MOTOR TRUCK DRIVER’:                8 (Driver, truck)
‘OPERATING ENGINEER-GROUP A’:        2 (Engineer, operations (electricity supplier))
‘OPERATING ENGINEER-GROUP C’:        2 (Engineer, operations (electricity supplier))
‘PARAMEDIC’:                         3 (Paramedic)
‘PARAMEDIC I/C’:                     3 (Paramedic)
‘PLUMBER’:                           5 (Plumber)
‘POLICE COMMUNICATIONS OPERATOR I’:  4 (Assistant, administration (police service))
‘POLICE COMMUNICATIONS OPERATOR II’: 4 (Assistant, administration (police service))
‘POLICE OFFICER’:                    3 (Officer, police)
‘POLICE OFFICER
 (ASSIGNED AS DETECTIVE)’:           3 (Detective (police service))
‘POLICE OFFICER
 (ASSIGNED AS EVIDENCE TECHNICIAN)’: 3 (Officer, police)
‘POLICE OFFICER / FLD TRNG OFFICER’: 3 (Officer, police)
‘POOL MOTOR TRUCK DRIVER’:           8 (Driver, truck)
‘SANITATION LABORER’:                6 (Worker, healthcare (hospital service))
‘SERGEANT’:                          3 (Sergeant)
‘TRAFFIC CONTROL AIDE-HOURLY’:       7 (Assistant, control, traffic, air)
```

Figure 4.26: Correspondencies between $Job\ Title$ values for the Chicago dataset and related major groups, together with index entries of reference.

Bibliography

[1] Social Justice - Overview, History and Evolution, Five Principles. *Corporate Finance Institute*, 2020. https://corporatefinanceinstitute.com/resources/knowledge/other/social-justice/.

[2] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*, volume 8. Addison-Wesley Reading, 1995.

[3] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine Bias. *ProPublica*, 2016. Available at: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

[4] UN General Assembly et al. Universal Declaration of Human Rights. *UN General Assembly*, 302(2):14–25, 1948.

[5] Fabio Azzalini, Chiara Criscuolo, and Letizia Tanca. FAIR-DB: FunctionAl DependencIes to discoveR Data Bias. In *EDBT/ICDT Workshops*, 2021.

[6] Robert L Barker et al. The Social Work Dictionary. 2003.

[7] The Editors of Encyclopaedia Britannica. "Equality". *Encyclopedia Britannica*, 2009. https://www.britannica.com/topic/equality-human-rights. Accessed 7 June 2021.

[8] The Editors of Encyclopaedia Britannica. "Database". *Encyclopedia Britannica*, 2020. https://www.britannica.com/technology/database. Accessed 10 June 2021.

[9] Loredana Caruccio, Vincenzo Deufemia, and Giuseppe Polese. Relaxed Functional Dependencies–A Survey of Approaches. *IEEE Transactions on Knowledge and Data Engineering*, 28(1):147–165, 2015.

[10] Andrew Chamberlain. How to Analyze Your Gender Pay Gap: An Employer's Guide. *Glassdoor Economic Research*, 2017.

[11] Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.

[12] Edgar F Codd. A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM*, 13(6):377–387, 1970.

[13] Jeffrey Dastin. Amazon scraps secret AI recruiting tool that showed bias against women. *Reuters*, 2018. Available at: `https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-thatshowed-bias-against-women-idUSKCN1MK08G`.

[14] Cambridge Advanced Learner's Dictionary. "Data". *Cambridge University Press*, 2013. `https://dictionary.cambridge.org/dictionary/english/data`. Accessed 10 June 2021.

[15] Cambridge Advanced Learner's Dictionary. "Fairness". *Cambridge University Press*, 2013. `https://dictionary.cambridge.org/dictionary/english/fairness`. Accessed 15 June 2021.

[16] Cambridge Advanced Learner's Dictionary. "Gender gap". *Cambridge University Press*, 2013. `https://dictionary.cambridge.org/dictionary/english/gender-gap`. Accessed 28 June 2021.

[17] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness Through Awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

[18] Equality Commission for Northern Ireland. *Index for Classifying Job Titles*. Equality Commission for Northern Ireland, Equality House 7–9, Shaftesbury Square, Belfast BT2 7DP, 2013. Available at: `https://www.equalityni.org/ECNI/media/ECNI/Publications/Employers%20and%20Service%20Providers/Monitoring%20and%20review/IndexforClassifyingJobTitlesfromJan13.pdf?ext=.pdf`.

[19] Batya Friedman and Helen Nissenbaum. *Bias in Computer Systems*. Routledge, 2017.

[20] Amir Gandomi and Murtaza Haider. Beyond the hype: Big data concepts, methods, and analytics. *International journal of information management*, 35(2):137–144, 2015.

[21] Samuel Gibbs. Women less likely to be shown ads for high-paid jobs on Google, study shows. *The Guardian*, 8(7), 2015. Available at: https://www.theguardian.com/technology/2015/jul/08/women-less-likely-ads-high-paid-jobs-google-study.

[22] Paul Glasserman. Linear Regression. *Lecture notes in Managerial Statistics*, 2001. Available at: https://www0.gsb.columbia.edu/faculty/pglasserman/B6014/Regression.pdf.

[23] H. V. Jagadish, Johannes Gehrke, Alexandros Labrinidis, Yannis Papakonstantinou, Jignesh M. Patel, Raghu Ramakrishnan, and Cyrus Shahabi. Big Data and Its Technical Challenges. *Commun. ACM*, 57(7):86–94, 2014. Available at: https://doi.org/10.1145/2611567.

[24] H. V. Jagadish, Julia Stoyanovich, and Bill Howe. The Many Facets of Data Equity. In *24th International Conference on Extending Database Technology (EDBT)*, 2021.

[25] Alexandros Labrinidis and H. V. Jagadish. Challenges and Opportunities with Big Data. *Proceedings of the VLDB Endowment*, 5(12):2032–2033, 2012.

[26] Andy Mason. "Equal opportunity". *Encyclopedia Britannica*, 2019. https://www.britannica.com/topic/equal-opportunity. Accessed 7 June 2021.

[27] Steve Mills, Steve Lucas, Leo Irakliotis, Michael Rappa, Teresa Carlson, and Bill Perlowitz. Demystifying Big Data: A Practical Guide To Transforming The Business of Government. *TechAmerica Foundation, Washington*, 2012.

[28] Jonathon Penney, Sarah McKune, Lex Gill, and Ronald J Deibert. Advancing Human-Rights-by-Design in the Dual-Use Technology Industry. *Journal of International Affairs*, 71(2):103–110, 2018.

[29] Joeri Rammelaere and Floris Geerts. Revisiting Conditional Functional Dependency Discovery: Splitting the "C" from the "FD". In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 552–568. Springer, 2018.

[30] Thomas L Saaty and Mujgan S Ozdemir. Why the Magic Number Seven Plus or Minus Two. *Mathematical and computer modelling*, 38(3-4):233–244, 2003.

[31] Teresa Scantamburlo, Andrew Charlesworth, and Nello Cristianini. Machine decisions and human consequences. *arXiv preprint arXiv:1811.06747*, 2018.

[32] M Sepuldeva, Th Van Banning, Gudrún Gudmundsdóttir, Christine Chamoun, and Willem JM Van Genugten. *Human Rights Reference Handbook*. University for Peace, 2010.

[33] R Tamilselvi, B Sivasakthi, and R Kavitha. An efficient preprocessing and postprocessing techniques in data mining. *International Journal of Research in Computer Applications and Robotics*, 3(4):80–85, 2015.

[34] Sahil Verma and Julia Rubin. Fairness Definitions Explained. In *2018 2018 ACM/IEEE International Workshop on Software Fairness*, pages 1–7. IEEE, 2018.

[35] Yongkai Wu, Lu Zhang, and Xintao Wu. Counterfactual Fairness: Unidentification, Bound and Algorithm. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

[36] Ke Yang, Julia Stoyanovich, Abolfazl Asudeh, Bill Howe, HV Jagadish, and Gerome Miklau. A Nutritional Label for Rankings. In *Proceedings of the 2018 international conference on management of data*, pages 1773–1776, 2018.

[37] Karen Yeung, Andrew Howes, and Ganna Pogrebna. AI Governance by Human Rights-Centered Design, Deliberation and Oversight: An End to Ethics Washing. *The Oxford Handbook of Ethics of AI*, page 77, 2020.

[38] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. FA*IR: A Fair Top-k Ranking Algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1569–1578, 2017.

[39] Indrė Žliobaitė. Measuring discrimination in algorithmic decision making. *Data Mining and Knowledge Discovery*, 31(4):1060–1089, 2017.