

Project Documentation

***AEQUITAS*: Bias free AI assisted
recruiting system**

Alessandro Tutone (alessandro.tutone@studio.unibo.it)
Alberto Genovese (alberto.genovese5@studio.unibo.it)

June 2025

Abstract

As the use of artificial intelligence by employers to screen job applicants grows, there are serious risks of discrimination against women, older applicants, and minority groups. Although AI hiring systems promise time and cost savings in recruiting new personnel, in doing so they enable, reinforce, and deepen discrimination against people who are already marginalized in the labor market. During the AI system's life cycle, several techniques can be implemented to highlight and overcome this problem.

Our project investigates a specific software used by the *Adecco Group* in the candidate selection process. The software supports the recruitment process by recommending the best candidates for a given job position and, conversely, suggesting the most suitable positions for candidates. Using the results provided by this software, we analyzed the presence of discrimination for different vulnerable groups. Once a bias was detected, using the AEQUITAS framework, we implemented several techniques in order to operate a mitigation of that bias.

Contents

1	Introduction	4
2	Background and Motivation	4
3	Task Description	4
4	Workflow	5
4.1	Data Explanation	5
4.2	Bias Detection	6
4.2.1	Statistical Parity	7
4.2.2	Disparate Impact	8
4.3	Testing model: Before mitigation	9
4.4	Pre-processing Techniques	10
4.4.1	Disparate Impact Remover	10
4.4.2	LFR (Learning Fair Representations)	11
4.4.3	Reweighting	12
4.5	In-processing Techniques	13
4.5.1	FaUCI	13
4.5.2	Adversarial Debiasing	14
4.5.3	Prejudice Remover	16
5	Conclusion	17
5.1	Results comment	17
5.2	Considerations	17

1 Introduction

AEQUITAS[1] is an innovative framework designed to ensure fairness and trust in AI-based decision support systems. It offers several methods to assess, display, and mitigate bias in AI systems. Thanks to its guidelines, methodologies, and software engineering techniques, it is able to create bias-free systems. In order to validate this framework, various cases are considered that cover distinct domains representative of industrial and digital sectors (hiring), public bodies (health), and sensitive social issues.

AEQUITAS[1] has incorporated six use cases covering three different domains: recruiting, society and economics, and healthcare. For our project, we considered the use case called “Bias free AI assisted recruiting system”[2]

2 Background and Motivation

Algorithmic discrimination caused by AI-enabled recruitment was addressed by several studies, like “*Ethics and discrimination in artificial intelligence-enabled recruitment practices*”[3], where technical and managerial solutions were also explored. In this paper, the findings suggest that AI-enabled recruitment has the potential to improve recruitment quality, increase efficiency, and reduce transactional work. However, algorithmic bias results in discriminatory hiring practices based on gender, race, color, and personality traits. The study indicates that algorithmic bias comes from limited raw data sets and biased algorithm designers. To mitigate this issue, it is recommended to implement technical measures, such as unbiased dataset frameworks and improved algorithmic transparency, as well as management measures such as internal corporate ethical governance and external oversight.

The problem is well known, and the question of whether or not using AI is one of the most asked. The preference of performance over human rights is a debate that is subject of discussion in the community. Another research that addresses this problem is “*Is AI recruiting (un)ethical? A human rights perspective on the use of AI for hiring*”[4], where they analyzed whether AI hiring practices inherently conflict with the concepts of validity, autonomy, non-discrimination, privacy, and transparency, which represent the main human rights relevant in this context.

3 Task Description

The software analyzed is used by the Adecco Group[2] in the candidate selection process. The software supports the recruitment process by recommending the best candidates for a given job position and, the other way around, suggesting the most suitable positions for each candidate.

This software is based on an AI system that extracts relevant information from each candidate and standardizes it against structured data stored in the

internal database. By comparing the attributes of the candidate with the requirements of the job position, the software generates a ranked list of candidates based on their compatibility. The results are then presented to the recruiter.

The purpose of our study is to target cognitive and structural bias that might be associated with existing assisted hiring systems. The analysis of the tool will make it possible to detect and assess possible biased outcomes resulting from the algorithm/software and the human expert in the selection of candidates.

4 Workflow

The general workflow of this project includes an initial part of data understanding to better understand what kind of data we are dealing with, and a following part where data are analyzed to evaluate the fairness level. This dataset is then used to train a baseline model, which serves as a reference to evaluate how performance and fairness are affected by various bias mitigation techniques.

4.1 Data Explanation

The software used by the Adecco Group[2] in the candidate selection phase supports the recruitment process by recommending the best candidates for a given job position and, conversely, suggesting the most suitable positions for each candidate. Indeed, we have two datasets at our disposal: a *direct one* and a *reverse one*.

- **direct_matching_20240213.csv** — For each *job offer*, the top 5 matching *candidates* are listed.
- **reverse_matching_20240213.csv** — For each *candidate*, the top 10 matching *job offers* are listed.

Each dataset contains similar features, but the columns “*match_score*” and “*match_rank*” have opposite meanings. In the *reverse* dataset they represent how much a certain job offer matches a specific candidate, while in the *direct* one, they represent how much a certain candidate matches a particular job offer. Common features are the following: “*cand_id*”, “*job_id*”, “*distance_km*”, “*cand_gender*”, “*cand_age_bucket*”, “*cand_domicile_region*”, “*cand_education*”, “*job_contract_type*”, “*job_sector*”, “*job_work_province*”, “*cand_domicile_province*”, “*job_professional_category*”.

To address our task, we decided to use the *reverse dataset*. Since our objective is to build a classification dataset, we made several modifications to the original data:

1. Adding the *avg_match_score* to assess how, on average, each candidate is evaluated.
2. Adding the *avg_score>97* to discretize the *avg_match_score* column. This is a binary column that will be our **target** to predict.

3. Removing all job-related features, such as “*job_id*” and “*job_sector*”.
4. Grouping all different education levels within 8 different levels.
5. Creating a new column *part_of_italy* to indicate if the candidate is from north, south, or center of Italy. Doing this emphasizes the presence of a bias between northern and southern candidates.
6. Removing all *NaN* values.
7. Using **Ordinal encoding** for categorical data.

4.2 Bias Detection

As a first step, we identified the *sensitive features* to consider. After a few trials, we discovered that a good *sensitive feature* to use is ***cand_domicile_region***, namely the region of origin of the candidate. In order to highlight the presence of a bias inside such feature we decided to create another column, “*part_of_italy*”, that was indeed considered as *sensitive feature*.

As we can see in Figure 1, there is no relevant correlation between the *sensitive features* and the other features of our dataset.

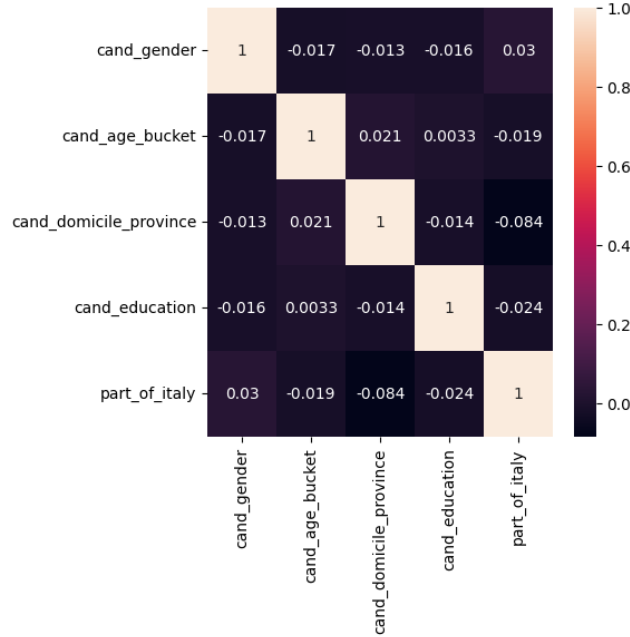


Figure 1: Correlation matrix between *features*

Performing this type of matrix, we look for feasible **proxy features**: highly correlated features with one of the *sensitive features*. Without having found any

proxy, we proceeded to measure the bias level within the dataset. We used two main approaches to assess the fairness of the dataset: *Statistical Parity* and *Disparate Impact*.

After computing the first graphs, we decided to use a different sensitive feature to better visualize the presence of a bias. Therefore, instead of using *cand_domicile_region*, we opted for the *part_of_italy* feature. This makes it easier to see the disparity between North and South of Italy.

4.2.1 Statistical Parity

When we talk about *Statistical Parity*, we refer to the fact that each sensitive group has the same probability of being predicted inside the positive class. We want to adjust the data so that decisions are made fairly and without discrimination.

To compute this metric, we used the function “*statistical_parity_difference()*” from the *FairLib* library[5].

Before applying any kind of mitigation, the results obtained are the following:

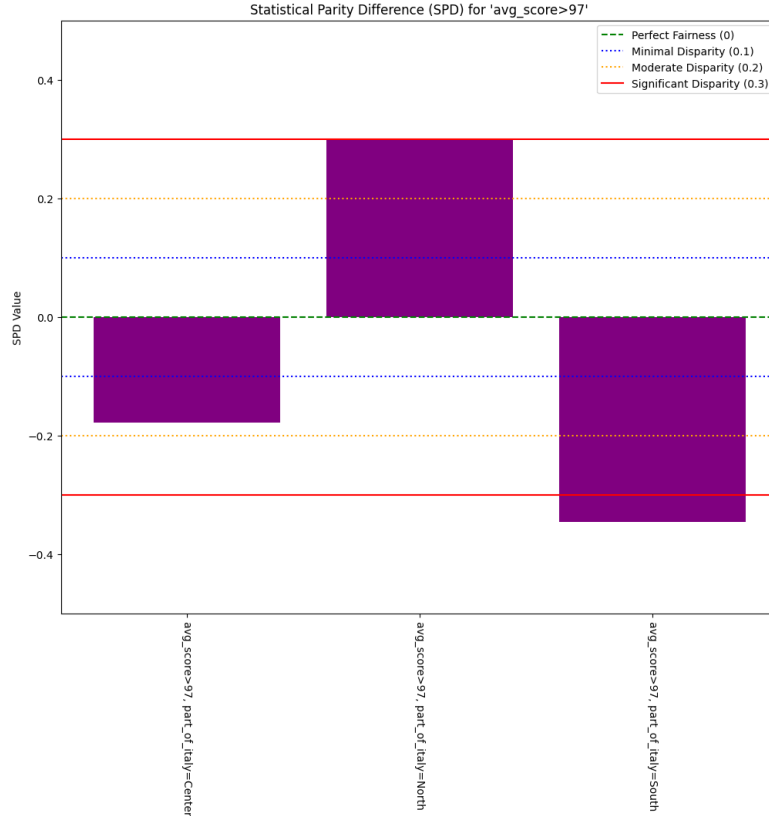


Figure 2: Statistical Parity before mitigation

The graph illustrates the **Statistical Parity Difference (SPD)** for the binary target variable $avg_score > 97$ across the different part of Italy: *North*, *Center*, and *South*. This highlights the disparities in favorable outcomes. A value of zero indicates perfect fairness, yet the results reveal significant imbalance: candidates from the North exhibit a strong positive **SPD** (0.3), suggesting they are disproportionately favored by the model. In contrast, candidates from the South show a negative **SPD** below -0.3 , indicating a marked disadvantage. The Center also experiences a mild negative disparity (-0.15), although less severe. These findings suggest that the model’s decision-making process is not regionally neutral and systematically favors northern candidates, raising concerns about geographic bias that should be addressed through appropriate fairness mitigation strategies.

Statistical Parity Difference may be sufficient to assess the presence of bias, but additional confirmation is needed.

4.2.2 Disparate Impact

When we talk about **Disparate Impact**, we refer to the phenomenon in which a machine learning model disproportionately disadvantages certain groups of people. Using this indicator, we can highlight the presence of unequal treatment among candidates.

Also to compute this metric, we used the function “*disparate_impact()*” from the **FairLib** library[5].

Before applying any kind of mitigation, the results obtained are the following:

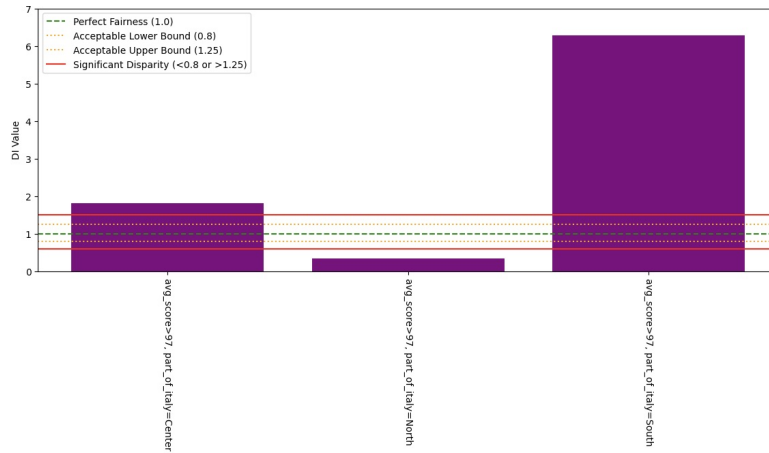


Figure 3: Disparity Impact before mitigation

In this chart, the **Disparate Impact (DI)** chart for $avg_score > 97$ reveals significant fairness issues across different parts of Italy. According to the **DI** values, candidates from the *North* are favored over the other groups ($DI \approx 0.3$),

the **DI** values falls well below the acceptable lower bound of 0.8, indicating significant bias. Conversely, candidates from the *South* show a **DI** value exceeding 6, which is far above the acceptable upper bound of 1.25, suggesting a disfavorable outcome with respect to the other groups. The *Center* region also exceeds the upper bound ($DI \approx 1.8$), though not as severely. These extreme deviations from the fairness threshold of 1.0 highlight clear inequities in how candidates are treated based on geographic region, reinforcing the need for bias mitigation strategies in the model.

With no surprise, the findings of our previous evaluation with the **SPD** are confirmed with the **DI** metric.

4.3 Testing model: Before mitigation

In order to evaluate the effect that each mitigation technique will have on the model performance, we need to compute a baseline. Given the classification dataset we created, we assess the performance. The model we choose to use is a **Random Forest**. To train the model, we use a ***k-fold cross-validation***: the training set is split into **k** smaller sets, and then the following procedure is followed for each of the **k** “*folds*”:

1. The model is trained using **k-1** of the *folds* as training data;
2. the resulting model is validated on the remaining part of the data (i.e., it is used as a test set to compute a performance measure).

The performance measure reported by ***k-fold cross-validation*** is then the average of the values computed in the loop.

Since we want to compute, not only the “classic metrics” such as *accuracy*, *f1-score*, etc., but also the *demographic_parity_ratio*, and the *equalized_odds_ratio*, we had to implement a customized ***k-fold cross-validation*** function.

These two metrics are important in assessing the presence of bias:

- The ***demographic_parity_ratio*** indicates the ratio between the smallest and the largest group-level *selection rate* across all values of the sensitive feature(s). The ***demographic parity ratio*** of 1 means that all groups have the same *selection rate*. With *selection rate*, we indicate the number of candidates predicted to have a *match_score* higher than 97 compared to the total number of job applicants for a particular position.
- The ***equalized_odds_ratio*** indicates whether a classification model provides equal predictive performance across different groups or populations. It focuses on measuring the balance of true positive rates (sensitivity/recall) and true negative rates (specificity) between the groups.

In the following table it is possible to see the results obtained by the model without having performed yet any kind of mitigation technique:

<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>	<i>F1-Score</i>	<i>Dem. Par. R.</i>	<i>Eq. Odds R.</i>
0.72	0.67	0.65	0.66	0.045	0.029

Fairlib[5] provides a set of tools to measure and improve fairness in machine learning models, covering a variety of fairness metrics, pre-processing, in-processing, and post-processing techniques. Once bias has been assessed, the use of these algorithms allows us to mitigate the level of discrimination.

4.4 Pre-processing Techniques

These algorithms operate on the data before they are used to train the model. The pre-processing techniques implemented are three: *Disparate Impact Remover*, *Learning Fair Representations*, and *Reweighting*.

4.4.1 Disparate Impact Remover

The *Disparate Impact Remover* is a pre-processing technique that transforms *feature values* to increase group fairness while preserving as much data utility as possible. By doing so, bias in the dataset is mitigated before model training. This algorithm offers a tunable repair level to control the trade-off between fairness and data utility. By looking at the two following graph, we can see how the algorithm has an impact on the model performance.

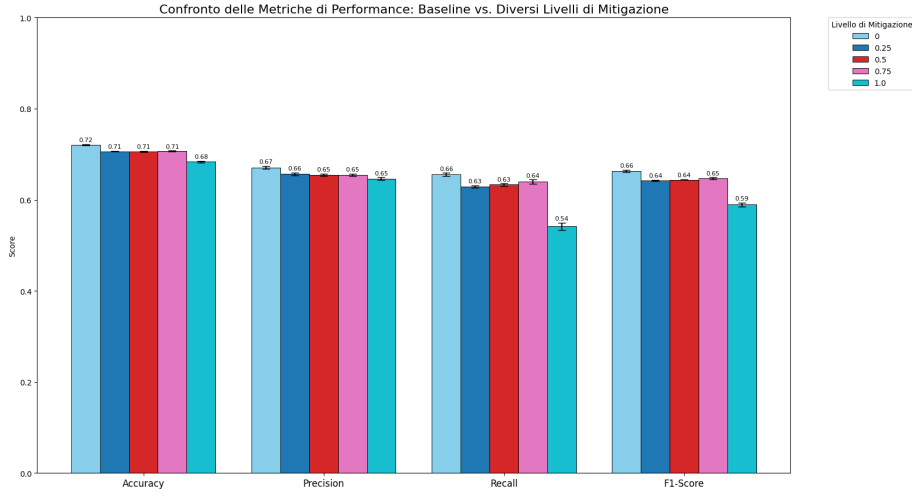


Figure 4: Disparity Impact Remover effect on performance

Performance measures are influenced by the mitigation we carried out by using this pre-processing technique. Increasing the mitigation level makes the model perform worse, as we can see from Figure 4.

Once we have assessed the behavior of the model, we have evaluated whether the mitigation was effective. Indeed, we used the metrics *Demographic Parity Ratio* and *Equalized Odds Ratio* to assess it. By looking at Figure 5, we can see that both metrics reached 1, which means that all groups now have the same *selection rate*.

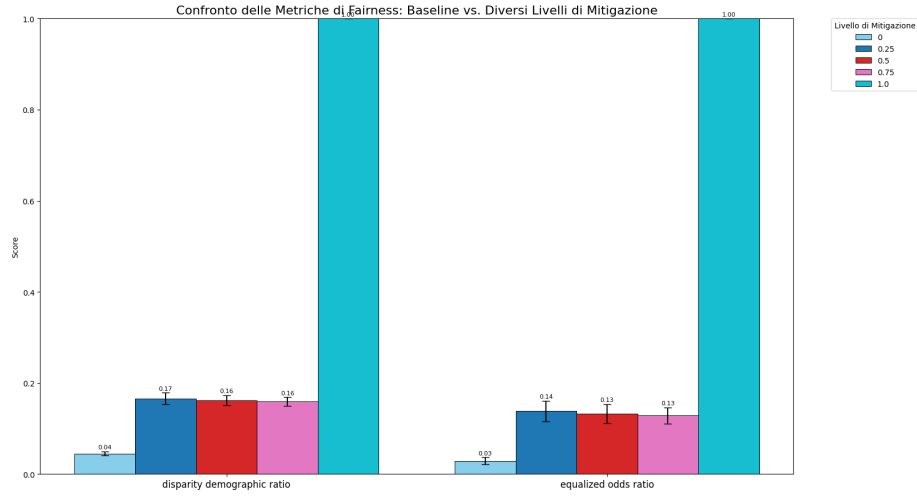


Figure 5: Disparity Impact Remover effect on fairness metrics

4.4.2 LFR (Learning Fair Representations)

This other pre-processing method aims to create a fair and accurate classification model by learning intermediate representations that mitigate bias with respect to sensitive attributes[6]. The method ensures that the learned representations do not retain information about the membership in the protected group, thus promoting fairness in the classification decisions.

Applying this algorithm on our data, gave us the following results:

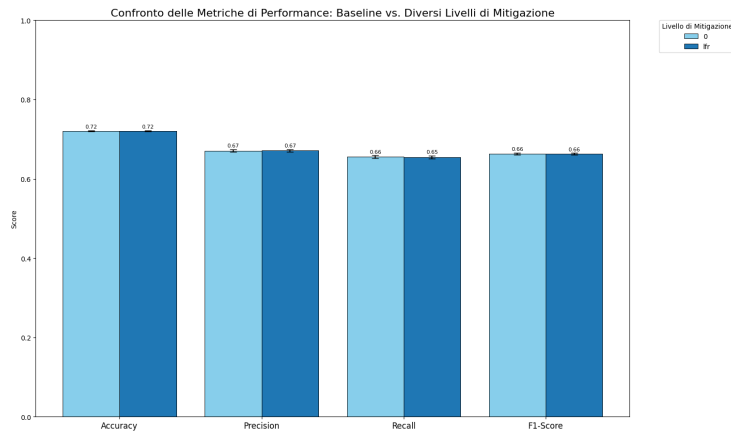


Figure 6: Learning Fair Representations effect on performance

This pre-processing technique was not effective on our dataset; indeed, as we can see from both Figure 6 and Figure 7, no relevant change was highlighted

on the model.

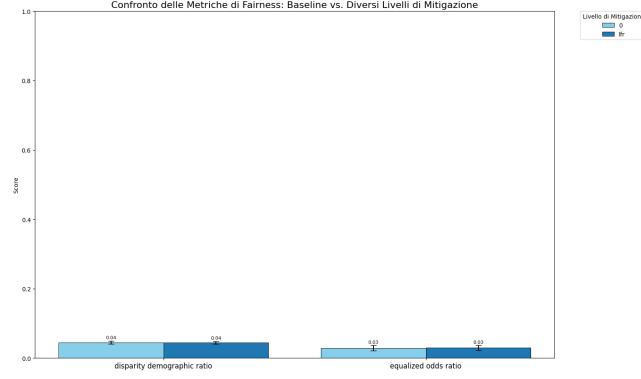


Figure 7: Learning Fair Representations effect on fairness metrics

4.4.3 Reweighting

Reweighting is a bias mitigation method that reweights observations within a data set to ensure fairness between different subgroups within a sensitive attribute. In practice, it weights the examples in each (group, label) combination differently to ensure fairness before classification[7]. As a result of **reweighting**, the Statistical Parity Difference (SPD) of all subgroups goes towards 0 and the Disparate Impact metric becomes close to 1.

For our dataset, these are the results we obtained:

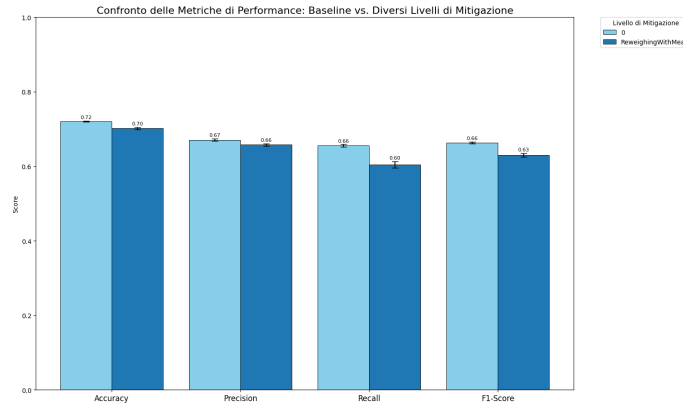


Figure 8: Reweighting effect on performance

The statistical composition of the dataset, or other factors, affected the performance of this other technique. Also as we have seen before, both the

performance and the fairness were unchanged after we applied the mitigation. Despite the level of mitigation we used, no effect was seen on the model.

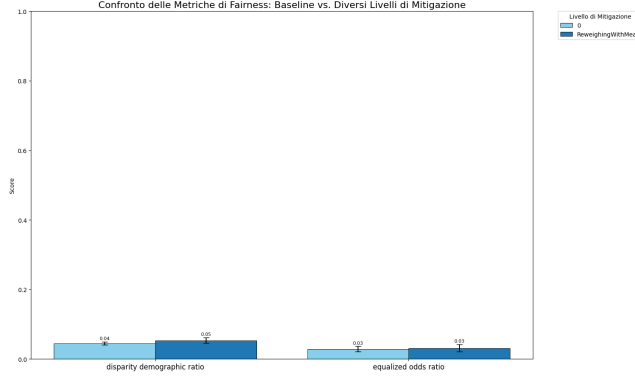


Figure 9: Reweighting effect on fairness metrics

4.5 In-processing Techniques

The other set of techniques we applied are the in-processing techniques that modify the learning algorithm itself to reduce bias during the model training phase. Thanks to these algorithms, fairness is ensured directly during the training process of the model; this allows the model to learn more unbiased decisions.

Given the fact that these algorithms work with neural networks, we created one that is able to perform binary classification.

We defined a feedforward architecture with four hidden blocks, each performing a linear transformation followed by batch normalization. It progressively reduces the data dimensionality, starting from an initial *input_dim* and mapping it down through 512, 256, 128, and finally to 64 features.

Once we have implemented the network, we use it to test the in-process algorithms given by the library.

4.5.1 FaUCI

FaUCI stands for **Fairness Under Constrained Injection** and is an in-processing fairness algorithm that incorporates fairness constraints directly into the model training process through regularization. This algorithm works by adding fairness-specific regularization terms to the loss function, which penalize the model when it produces unfair predictions. The model learns to balance between predictive accuracy and fairness during training.

According to the documentation, the model is trained with a standard loss function (e.g., BCE loss by classification), and a regularization term for fairness is added to penalize unfair predictions.

We can tune the regularization weight parameter to control the trade-off between fairness and accuracy.

- **weight = 0**: No fairness constraint (standard model)
- **weight > 0**: Increasing values impose stronger fairness constraints

FaUCI supports several fairness metrics as regularization targets such as *Statistical Parity Difference* and *Disparate Impact*.

By applying *FaUCI*, we obtain the following results:

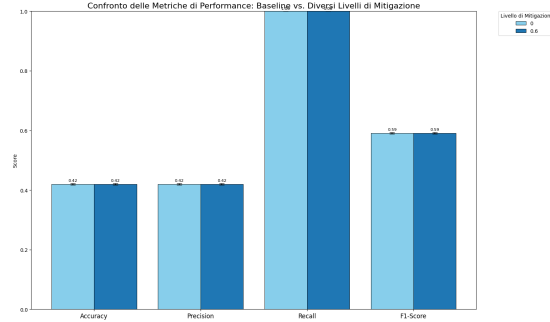


Figure 10: FaUCI effect on performance

The algorithms of in-processing seem not working, our neural network keeps its former performance, and no improvement was assessed on the fairness level.

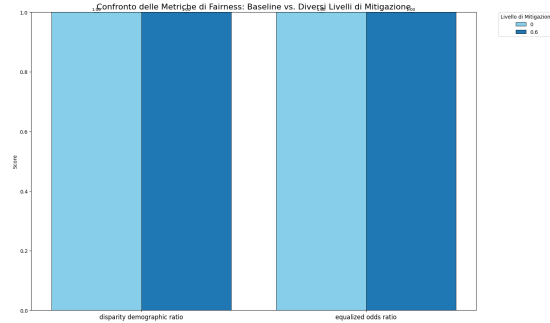


Figure 11: FaUCI effect on fairness

4.5.2 Adversarial Debiasing

Adversarial Debiasing is a method that uses adversarial learning techniques to remove unwanted correlations between predictions and sensitive attributes. This algorithm uses two networks to achieve its objective: a **predictor** network, which learns to predict the target variable, and an **adversary** network, that attempts to predict the sensitive attribute from the **predictor**'s output. The **predictor** aim is to maximize accuracy while also minimizing the ability of the

adversary network to predict the sensitive attribute, and the **adversary** tries to predict the sensitive attribute given the **predictor**'s output.

According to the documentation, a gradient reversal layer is employed between the predictor and the adversary networks to ensure that the **predictor** learns to make predictions that are accurate and unbiased with respect to the sensitive attribute.

The strength of the fairness constraint is controlled by a parameter lambda (*adversary weight*), where:

- **lambda = 0**: No fairness constraint (standard model)
- **lambda > 0**: Increasing values enforce stronger fairness constraints

Using this fairness algorithm gave us these results:

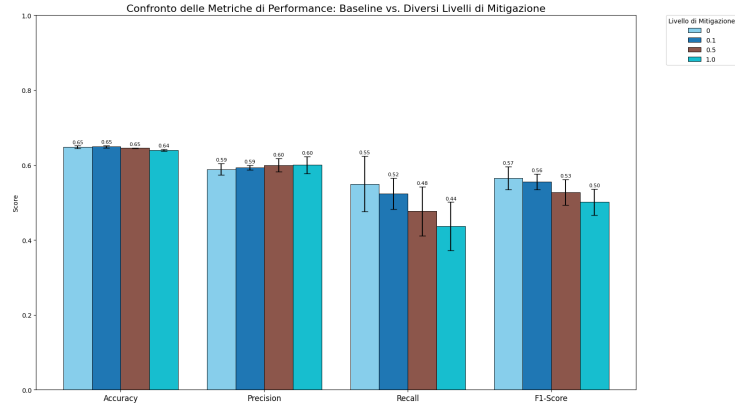


Figure 12: Adversarial Debiasing effect on performance

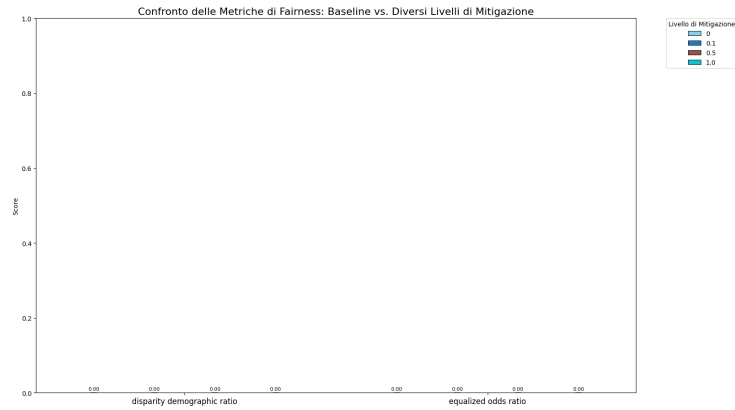


Figure 13: Adversarial Debiasing effect on fairness

4.5.3 Prejudice Remover

The Prejudice Remover Regularizer is a method designed to enforce fairness in classification tasks by reducing indirect prejudice. By applying this method, we want to ensure that predictions are not unfairly influenced by sensitive features[8]. The algorithm implemented in the library is designed to be computationally efficient and easy to implement, making it a practical solution for fairness-aware machine learning models. We want our model to be able to produce outputs that are not enforced by a prejudice induced by the sensitive features.

The results we obtained after using the Regularizer are the following:

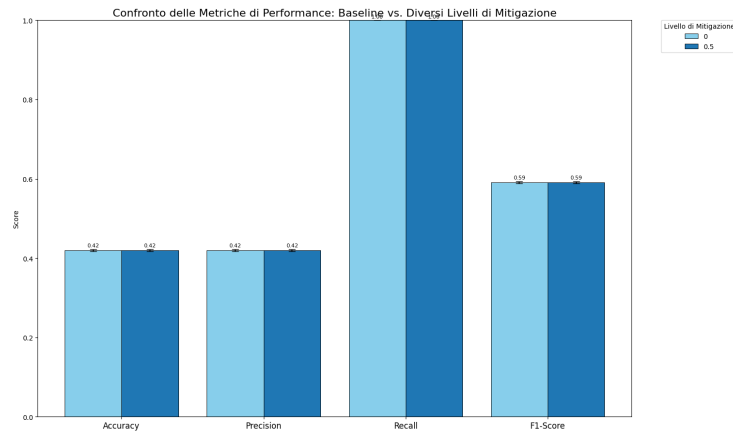


Figure 14: Prejudice Remover effect on performance

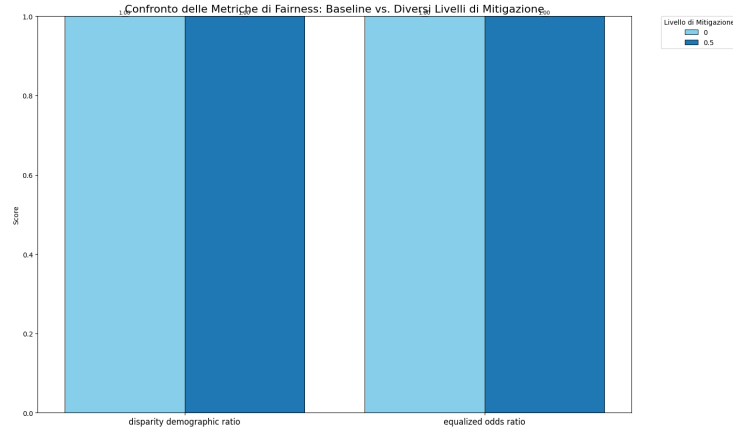


Figure 15: Prejudice Remover effect on fairness

5 Conclusion

5.1 Results comment

The consistent failure of the in-processing mitigation technique we applied was due to the poor performance of our neural network. Although the model used for the pre-processing technique achieved decent results, the neural architecture we designed was not capable of learning effectively.

The network's performance is somewhat acceptable, considering that the task was to predict whether a given candidate would reach a certain score based solely on features such as education level, age, and province of residence, but since we wanted to address the fairness level of such network, these performance reached were still too poor.

More valuable results could likely have been obtained using a more comprehensive dataset with a broader range of relevant features. Such a dataset would have enabled the network to learn more meaningful patterns for making accurate predictions.

On the other hand, we obtained the expected results with our Random Forest model. The performance achieved by this model was sufficient to allow us to apply the pre-processing techniques. In fact, we were able to clearly observe how the model's performance varied according to the level of mitigation applied.

5.2 Considerations

Using a dataset to train an AI model with the only goal of achieving the highest possible accuracy without taking the nature of the data into consideration may result in both excellent performance and significant discrimination. Often, when implementing an AI algorithm, the primary objective is to maximize performance, while the issue of discrimination in its decisions is typically overlooked.

Raising awareness about the problem of discrimination in AI decision-making is a fundamental step towards obtaining more fair AI systems. It is important to give more space to data exploration and to assessing bias within the data before using them.

Creating systems that operate with decent performance, while also avoiding prejudices in making decisions is possible. As we discovered with this assignment, using pre-processing and in-processing techniques is an effective way to find a balance between accuracy performance and fairness. However, choosing the right parameter remains a challenging part of training; the tuning process will be context-dependent.

To conclude, we can say that the pursuit of top performance regardless of how it is achieved is a mindset that is incompatible with the goal of creating systems that are also fair. Today, it is important to reconsider priorities when developing models. Making unbiased decisions, even if using a biased dataset, is possible and it should be pursued whenever feasible.

References

- [1] AEQUITAS, “Welcome to the aequitas framework for ai fairness,” <https://aequitas-home.readthedocs.io/en/latest/index.html>, 2025.
- [2] Aequitas, “Use case hr1: Bias free ai assisted recruiting system,” <https://aequitas-home.readthedocs.io/en/latest/usecases.html>, 2025.
- [3] Z. Chen, “Ethics and discrimination in artificial intelligence-enabled recruitment practices,” *Humanities and Social Sciences Communications*, vol. 10, no. 1, pp. 1–12, 2023.
- [4] A. L. Hunkenschroer and A. Kriebitz, “Is ai recruiting (un) ethical? a human rights perspective on the use of ai for hiring,” *AI and Ethics*, vol. 3, no. 1, pp. 199–213, 2023.
- [5] Aequitas, “Fairlib - python library for fairness metrics and processing,” <https://github.com/pikalab-unibo-students/master-thesis-dizio-ay2324/tree/master>, 2025.
- [6] H. AI, “Learning fair representations,” https://holisticai.readthedocs.io/en/latest/getting_started/bias/mitigation/preprocessing/bc_learning_fair_representations_lfr.html, 2025.
- [7] MathWorks, “Bias mitigation in credit scoring by reweighting,” <https://www.mathworks.com/help/risk/bias-mitigation-for-credit-scoring-model-byreweighting.html>, 2025.
- [8] H. AI, “Prejudice remover regularizer,” https://holisticai.readthedocs.io/en/latest/getting_started/bias/mitigation/inprocessing/bc_prejudice_remover_prejudice_remover_regularizer.html, 2025.