# Python II – Group Assignment

Group B

Christian Barba, Carlos Blazquez, Diana Fernandez,
Dominik Roser, Hiba Shanna, Mark Hourany

# Intro to FairDetect

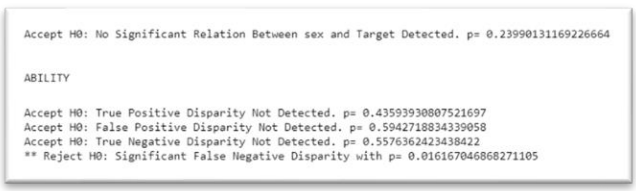
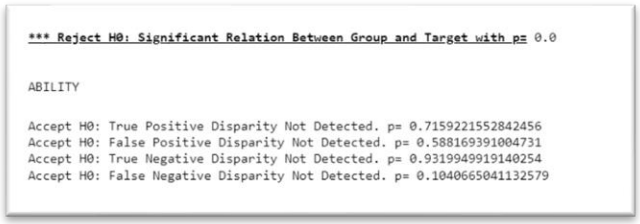
FairDetect framework allows us to detect and understand machine learning bias by defining 3 major fairness metrics: representation, ability, and performance, and then it defines a series of tests to "identify" levels of disparities between sensitive groups. FairDetect is structured into 3 main core steps:

1. Representation: Compares the representation of sensitive variables and its association with the target
2. Ability: Compares the ability of the sensitive variables
3. Predictive: Compares the distribution of the dataset as opposed to the predicted set

Subsequently, FairDetect allows us to deep dive into the model's predictions, for us to understand how classifications were made and isolate the most unprivileged groups, in a way clear steps can be taken to eliminate and/or mitigate the adverse effects.

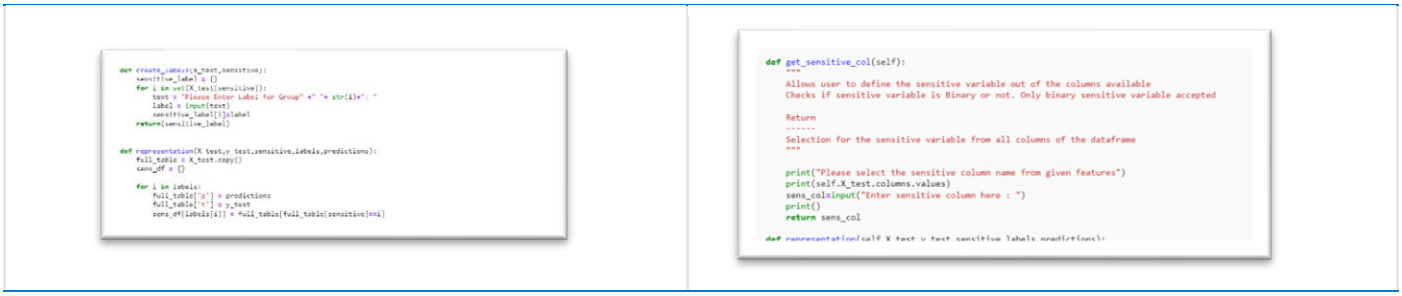
# FairDetect Improvements Explained

**Improvement 1:** Whenever a rejection of the null hypothesis happens and bias is detected, the statement is printed out in bold letters to show. This helps the user directly to spot bias detection in a visual way.

| FairDetect | FairDetect Improved |
|---|---|
|  |  |

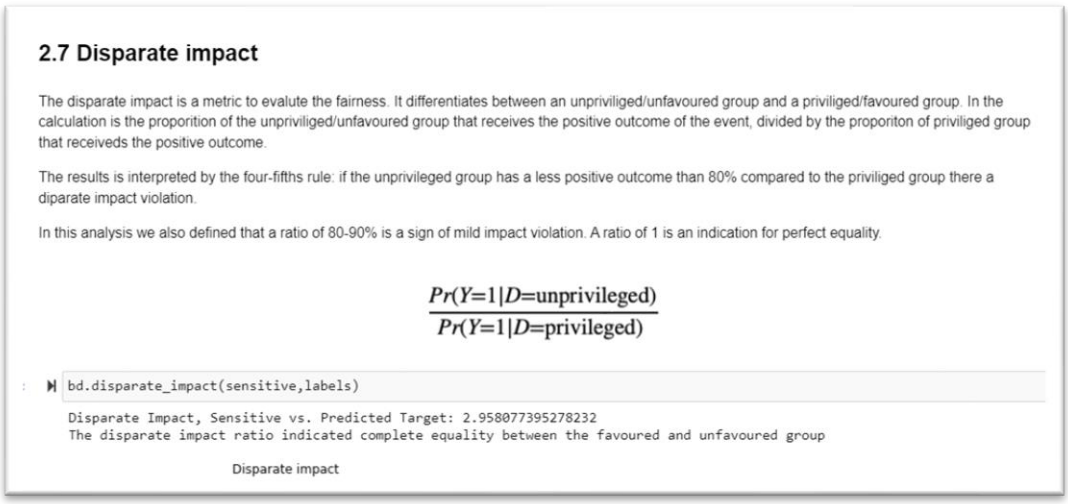
**Improvement 2:** Added Docstrings understanding the functionality of the larger part of the code: the class FairDetect and its methods. Anyone using the library would be able to know all about the FairDetect package like description, package modules, etc., by could simply using the help function to get all the information. This lowers the time of understanding a library and giving easy to access to those users with less knowledge in Python coding syntax.

| FairDetect | FairDetect Improved |
|---|---|



```
def get_sensitive_col(self):
    """
    Allows user to define the sensitive variable out of the columns available
    Checks if sensitive variable is Binary or not. Only binary sensitive variable accepted

    Return
    ------
    Selection for the sensitive variable from all columns of the dataframe
    """

    print("Please select the sensitive column name from given features")
    print(self.X_test.columns.values)
    sens_col=input("Enter sensitive column here : ")
    print()
    return sens_col
```

**Improvement 3:** Introduced Disparate Impact Ratio in the class: t ratio of positive outcomes (Loan_Status=1) in the unprivileged group) divided by the ratio of positive outcomes in the privileged group. This is a common metric in bias detection. In further iterations an improvement to be made would be in the method of disparate impact itself. At the moment there is a hard coding that the favoured group is equal to the sensitive group with the value 0 and unfavoured group is equal to the sensitive group with value 1. A more dynamic data entry could be possible in the next version.



## 2.7 Disparate impact

The disparate impact is a metric to evalute the fairness. It differentiates between an unpriviliged/unfavoured group and a priviliged/favoured group. In the calculation is the proporition of the unpriviliged/unfavoured group that receives the positive outcome of the event, divided by the proporiton of priviliged group that receiveds the positive outcome.

The results is interpreted by the four-fifths rule: if the unprivileged group has a less positive outcome than 80% compared to the priviliged group there a diparate impact violation.

In this analysis we also defined that a ratio of 80-90% is a sign of mild impact violation. A ratio of 1 is an indication for perfect equality.

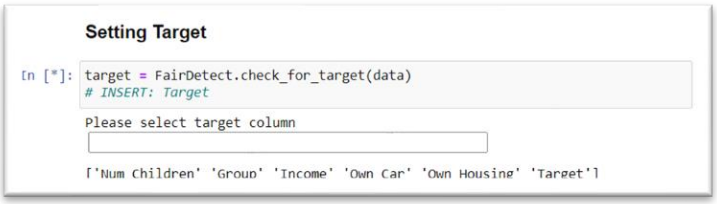
$$\frac{Pr(Y=1|D=\text{unprivileged})}{Pr(Y=1|D=\text{privileged})}$$

```
bd.disparate_impact(sensitive,labels)
```

```
Disparate Impact, Sensitive vs. Predicted Target: 2.958077395278232
The disparate impact ratio indicated complete equality between the favoured and unfavoured group
```

Disparate impact

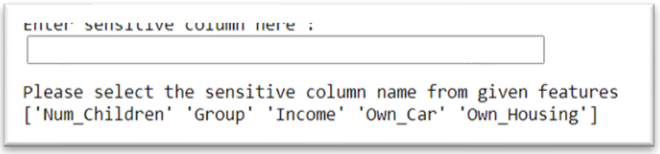
**Improvement 4:** Introduced **user experience** improvements that helps users select critical features for their bias detection and ML exercise which saves them the hassle of encoding the same features (target, sensitive, labels, etc.) for every piece of redundant code. Introduced codes for feature selection which prompts users to select their **Target Value, Sensitive Value and Label Values**



**Setting Target**

```
In [*]: target = FairDetect.check_for_target(data)
        # INSERT: Target
```

```
Please select target column

['Num_Children' 'Group' 'Income' 'Own_Car' 'Own_Housing' 'Target']
```



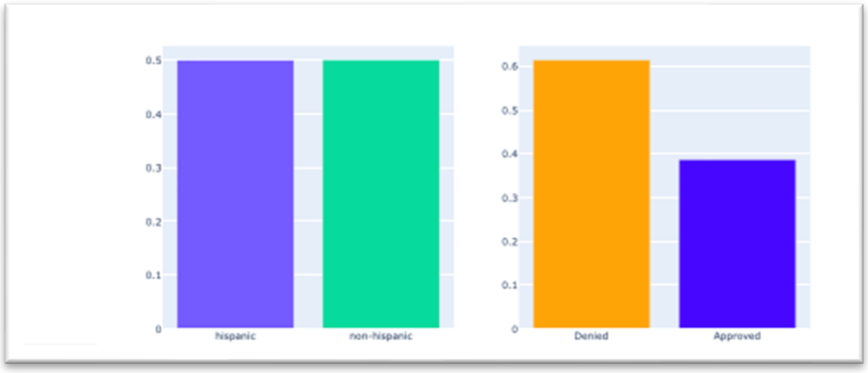
```
Enter sensitive column here :

Please select the sensitive column name from given features
['Num_Children' 'Group' 'Income' 'Own_Car' 'Own_Housing']
```

**Representation of Sensitive Variables**

```
sensitive=bd.get_sensitive_col()
labels=bd.create_labels(sensitive)
bd.identify_bias(sensitive,labels)

#INSERT label 1: hispanic
#INSERT in label 2: non_hispanic
```

```
Please select the sensitive column name from given features
['Num_Children' 'Group' 'Income' 'Own_Car' 'Own_Housing']
Enter sensitive column here : Group

Please Enter Label for Group 0: T
Please Enter Label for Group 1: F

Enter Target names below
Please Enter name for target predicted 0: Hispanic
Please Enter name for target predicted 1: Non-Hispanic
REPRESENTATION
```

Why did we introduce these changes? Just imagine users looking at representation charts and seeing their labels encoded with 0s and 1s. It will get confusing to those users less familiar with typical standard statistical terms. The example above just shows that we are now able to have 0s represented by 'Hispanic' target labels and 1 represented by 'Non-Hispanic' target labels. In addition, the target event is directly labeled instead of using the statistical terms of 0 and 1 in our case "denied" and "approved"




| | hispanic | non-hispanic |
|---|---|---|
| Denied | 0.804943 | 0.423006 |
| Approved | 0.195057 | 0.576994 |

*** Reject H0: Significant Relation Between Group and Target with p= 0.0

**Improvement 5:** Introduced Quality control method that checks if Target value selected (entered via point number 4 above) was binary and raises  out error in case it wasn't:

```
def check_for_target(self):
    """
    Target variable label definition. Only binary target variable accepted
    """
    print(data.columns.values)
    target_column = input("Please select target column")
    if not(set(data[target_column].unique())==set([0,1])):
        raise ValueError('Only binary target variable')
    else:
        return target_column
```

Improved the **understand_shap** method which allows us to not only to view feature importance for one class member, in fact it allows us to generate the same visualisations for the other class member (Hispanic and non-hispanic)

```
print("Model Importance Comparison")
plt.subplot(1, 2, 1) # row 1, col 2 index 1
shap.plots.bar(shap_values.cohorts(sens_glob_coh).abs.mean(0),show=False)
plt.subplot(1, 2, 2) # row 1, col 2 index 1
shap_values2 = explainer(affected_class.drop(['t','p'],axis=1))
shap.plots.bar(shap_values2)
#shap.plots.bar(shap_values2)
```

**Improvement 6 (Bonus):** We accomplished to store the results of our fair detect analysis with SQLite and SQL Alchemy. We selected for our case the KPIs of the p-value of TPR, FPR, TNR and FNR as well as the disparate impact. After each KPIs is calculated is is stored dynamically together with an ID and the date in which it was saved.

| | Id | date | p_TPR | p_FPR | p_TNR | p_FNR | disparate_impact |
|---|---|---|---|---|---|---|---|
| 0 | 9952 | 2022-07-31 09:36:49.960062 | 0.715922 | 0.588169 | 0.931995 | 0.104067 | 2.958077 |

**Trade-off:**

The introduction of the improvements unfortunately increased the time until the libraries were loaded. Given the marginal increase of loading vs. the value the introduced improvements provide, we would argue that the trade-off is small.

Loading libraries and functions time before improvement: **2.73 sec**

Loading libraries and functions time after improvements**: 4.84 sec**

Using FairDetect before improvement: **0.5 sec**

Using FairDetect after improvement**: 0.7 sec**

# Intro to Aequitas

Aequitas, a second bias framework has been introduced, in order to compare different aspects as functionality, usefulness, methodology regarding bias detection.

Aequitas was created by the Centre for Data Science and Public Policy at the University of Chicago. Users can easily audit models for various bias and fairness indicators in connection to numerous population sub-groups using this open-source tool. Aequitas offers thorough guidance on how to apply its findings in a public policy environment, taking the ensuing interventions and their ramifications into account. Through both seamless integration in the machine learning workflow and a web app designed for non-technical users verifying these models' results, it is meant to be utilised by both data scientists and policymakers.

# Tool Comparison

Table below resumes the main differences between FairDetect and Aequitas with respect to main characteristics:

| Characteristic | FairDetect | Aequitas |
|---|---|---|
| Functionality | **1. Attribute to be selected for training the model**: Target/ Classifier Variable. Sensitive variable needs to be defined as an additional step<br>**2.Fairness Criteria:**<br>• Representation<br>• Ability<br>• Predictive Parity<br>**3. Fairness Threshold:** No<br>**4. Requires data pre-processing**<br>**5. Statistics:** Does provide p-values for each fairness criteria | **1. Attribute to be selected for training the model**: variable to be audited for the overall population considered for intervention. requires a reference class to be selected for each attribute and compares the remaining sub-groups with the reference one ⇒ uses disparity as a measure of bias<br>**2.Fairness Criteria:** Very comprehensive set of 10 metrics for each class which facilitates identifying where there might be a bias. Among them:<br>• Equal Parity<br>• Proportional Parity<br>• False Positive Parity<br>• False Negative Parity<br>• Precision<br>**3. Fairness Threshold:** Provides a parity range for a measure to be considered fair<br>**4. Requires data pre-processing**<br>**5. Statistics:** does not provide p-values but a True/False logic. |
| Usability | 1. **Format Available:** Python Library<br>2. **Dataset Consideration:** Require One hot encoding before training the model.<br>3. **Classes Design:**<br>• Identify bias() – Identify areas of bias in the data set<br>• Understand. Shap() – features impacted by bias<br>3. **Visualizations:** Ability to choose the metric to be measured<br>• Bars<br>• Confusion Matrix<br>• Shapley waterfall plots<br>4. **Use Cases:** Works well for different use cases, both socioeconomics, public policy or marketing oriented | 1. **Formats Available**<br>• Web Audit Tool http://aequitas.dssg.io/<br>• Python Library<br>• Command Line Tool<br>2. **Dataset Consideration:** Continuous variables must be discretized prior to passing the data into the classes.<br>3. **Classes Design:**<br>• Group() – Define Groups<br>• Bias() – Calculate Disparities<br>• Fairness() – Assert Fairness<br>3. **Visualizations:** Ability to choose the metric to be measured<br>• Treemaps<br>• Disparity Plots<br>• All up reports<br>4. **Use Cases:** is oriented mostly to public policy and requires a clear and ex-ante social target for the model. |

# Tool Comparison and Case Studies

Case Studies 1: Synthetic Credit Card

| Synthetic Credit Card | FairDetect | Aequitas |
|---|---|---|
| Goal | The Synthetic Credit Card data set predicts the likelihood of individuals of getting a credit card approved or denied. Our goal is to identify bias using FairDetect and Aequitas framework and compare analysis and results | |
| Data | • Income (in the Income column),<br>• The number of children (in the Num_Children column),<br>• Whether the applicant owns a car (in the Own_Car column, the value is 1 if the applicant owns a car, and is else 0), and<br>• Whether the applicant owns a home (in the Own_Housing column, the value is 1 if the applicant owns a home, and is else 0)<br>• **The Group column breaks the users into two groups (where each group corresponds to either 0 or 1).** The column as breaking the users into two different races, ethnicities, or gender groupings. For this exercise, 0 would correspond to a Hispanic user, while 1 corresponds to a non-Hispanic user. | |
| Bias Identification | **Representation Bias:** The rejection of the null hypothesis: confirming the significant relationship between the group Hispanic / Non - Hispanic and Outcome: Credit Card Approved/Denied<br>**Ability Bias:** FairDetect did not identify any disparity among the<br>**Predictive Bias**: The Chi square test indicates there is no significant disparity in the results of the prediction as compared to the distribution of the data se | **False Positive Rate (FPR)** is the fraction of individuals whose credit card was denied, and the model misclassifies with an approved prediction. FPR is quite low across all groups and labels. However, we observed the medium income group having a relatively bigger FPR rate compared to other groups.<br>**False Negative Rate (FNR)** is the fraction of individuals with approved credit cards and the model misclassifies with a denied prediction. 2 groups raise our concerns here: low-income group and the Hispanic group as FNR rate are significantly higher than for the rest of groups withing their category.<br>**False Discovery Rate (FDR)** is the fraction of individuals who the model predicts to have an approved credit card but for whom the card was denied. Hispanic and low-income groups are again the concern here.<br>**False Omission Rate (FOR)** is the fraction of individuals who the model predicts to have a denied credit card but for whom the card was approved. Medium income group seems to get a benefit from the model<br>**Precision** is the fraction of individuals who the model predicts to have an approved credit card about whom this prediction is correct. Quite high for all groups except for low income and Hispanic group. |
| Specific Group Comparison | **Group 0 (Hispanic) that have mis correctly been denied credit card (Target 0)**: the model gives slightly more importance to income for Hispanics as opposed to non-Hispanics while the other parameters have roughly the same impact.<br>**Group 1 (non-Hispanic) wrongly classified denied credit card (Target 0):** we can see that all parameters have the same relative importance however income has a smaller contribution or in other words, has been undervalued for the positive outcome | We can verify that for "hispanic" in the sensitive class (Group), the differences in False Discovery Rate (FDR) and False Negative Rate (FNR) are both above 3 times the one of the reference class (non-hispanic).<br>Another aspect that can be highlighted is the 1.5 Predicted Prevalence disparity of Owner of housing.<br>If we check FDR and FNR for hispanic we see that those disparities are marked as significant (TRUE).<br>In a nutshell, the tool detects significant disparities in:<br>- hispanic<br>- medium and high income<br>- owning a house<br>- owning a car |

Case Studies 2: Modcloth

To prove the generalisation of the improved FairDetect library also a second dataset besides the synthetic credit card approval was chosen. We decided to leverage the dataset of Modcloth, an American online retailer of indie and vintage-inspired women's clothing.

| ModCloth | FairDetect | Aequitas |
|---|---|---|
| Goal | **ModCloth is an e-commerce website which sells women's clothing and accessories.** Many products in ModCloth include two human models with different body shapes and measurements of these models. Users can optionally provide the product sizes they purchased and fit feedback ('Just Right', 'Slightly Larger', 'Larger', 'Slightly Smaller' or 'Smaller') along with their reviews.<br>Therefore, our source of bias is the dimension of human body shape. With the use of **Fairdetect** and **Aequitas** frameworks, we want to understand the existence of the association between product image and user identity in consumers' product selections | |
| Data | There are 2 variables of interest:<br>**User identity:** is the perception of oneself; It calculates the average size each user purchased and classify users into 'Small' and 'Large' groups based on the same standard as the product body shape image.<br>**Product image:** the public impression of a product; attributes of the human models included in the product pictures are used to generate this data set. Products with only one human model wearing a relatively small size ('XS', 'S', 'M' or 'L') are labelled as the 'Small' group while products with two models (an additional model wearing a plus-size: 1X', '2X', '3X' or '4X') are referred as the 'Small&Large' group. | |
| Bias Identification | Comparison of the sensitive attribute User Identity Group (Large=0 and Small=1) and the target variable (Model Attribute: Product Image: Small=0, Small & large=1).<br>**Representation Bias:** User Identity is an unbalanced class, with users identified with identity group large equals to 18% and those identified with small equals 82%. Reject the null Hypothesis of non significance relation between the sensitive variable and the target variable. This means THERE IS a relationship between the user identity and the product image.<br>**Ability Bias:** In the false negative rate (FNR) there is a significant difference between large and small, large enough to reject the null hypothesis, meaning there is presence of false negative disparity and there is a bigger chances of marketing misrepresentation of users identified with large sizes than users identified with small sizes<br>**Predictive Bias**: The model is not further exacerbating the any observed bias. Whatever is present in the dataset is observed in the predictions. | **False Positive Rate (FPR):** is the fraction of individuals who identified with a large size the model misclassifies with small model image. FPR is quite low across all groups and labels.<br><br>**False Negative Rate (FNR):**is the fraction of individuals identified with a small body size and the model misclassifies with a small & large size model image. 1 groups raise our concerns here: very small size fit group seem to be given very often a wrong model image compared to the rest of the fit attributes.<br><br>**False Discovery Rate (FDR):** is the fraction of individuals who the model predicts to have a small size but for whom their individual perception of their body is large. Very large size fit seems to be impacted in this category. |
| Specific Group Comparison | • Brand is the most significant variable for both user identity types, large and small, however is relatively more important for small size users than large size users.<br>• Fit is twice as relevant for users identified with large sizes compared to people identified with small sizes.<br>• For the affected group and target, most relevant variable is category, followed by size and fit, which is significantly more relevant for this group than for the entire population | Predicted Positive Rate Disparity (ppr) is present in all of the 4 categories n our analysis: Fit, User_Attr, Category and Rating, if compared to the reference attribute selected of each of them. |