NOVA IMS
Information Management School

MACHINE LEARNING

TO GRANT OR NOT TO GRANT

DECIDING ON COMPENSATION BENEFITS

Professors:
Roberto Henriques
Ricardo Santos

André Oliveira 20211539
Bernardo Faria 20240579
Hassan Bhatti 20241023
João Marto 20211618
Miguel Mangerona 20240595

INDEX

ABSTRACT

The New York Workers' Compensation Board (WCB) is responsible for processing claims related to workplace injuries. This task involves the manual review and decision-making on millions of claims, which can be extensive and complex. To address this challenge, the WCB approached us to develop a machine-learning solution that automates and streamlines the classification of injury types in claims. Our task was to create a predictive model capable of categorizing claims into predefined categories, such as Non-Compensable, Temporary, and PPD Schedule Loss.

The project followed a structured approach, beginning with data cleaning and preprocessing of the historical claims dataset. We performed outlier removal, handled missing values, and employed feature engineering to ensure data quality. Key features such as *Average Weekly Wage*, *IME-4 Count*, and *Medical Fee Region* were selected using techniques such as *Spearman*'s correlation, *Chi-squared* tests, *LassoCV*, and *Recursive Feature Elimination* (RFE).

We then trained and evaluated multiple machine learning models, including *Decision Trees*, *Random Forest*, *XGBoost*, *LightGBM*, and *Extremely Randomized Trees*. Additionally, we used Cross-validation to enhance model reliability and minimize overfitting. The *XGBoost* and *LightGBM* models were the best performers based on their F1 scores, indicating a strong predictive accuracy.

To facilitate real-time predictions, we developed a *Streamlit*-based application that provides an interactive interface. The app allows users to input claim data manually or load predefined examples and runs predictions simultaneously across the *Random Forest*, *XGBoost*, and *LightGBM* models. It displays the results, enabling users to compare the models effectively.

The final solution offers a robust and efficient tool for automating claim classification, which could significantly reduce the manual effort and time required by the WCB. This project underscores the effectiveness of machine learning in enhancing decision-making processes in large-scale administrative tasks, with the potential to improve accuracy, consistency, and efficiency.

Keywords: Machine Learning, *XGBoost*, Claim Classification, Feature Engeneering

1. Introduction

We were given the mission of using data from articles relating to previous accidents at work and the resulting decision of the New York Workers' Compensation Board (WCIO) regarding their compensation, to be able to create a supervised learning model that could predict these same WCIO decisions for future accidents.

It will be through exploration, analysis and treatment of this very extensive dataset with high amounts of inconsistencies that we will carry out pre-processing in order to not only create new variables that can help us with the model, but also apply other types of transformations that allow us to achieve a more reliable data set (transformations of distributions, fill in missing values, correct things that logically don't make sense, etc.).

After this treatment, we will move on to selecting those that will be considered the most relevant features and that can most influence our variable that we want to predict (Claim Injury Type) and, through a set of techniques, we will group together those that are in accordance with these same criteria so that we can move on to the end and the final objective of this project, which is based on the creation and optimization of the best possible model capable of predicting this target with the greatest possible precision.

There are some similar works developed with different goals. In the thesis mentioned with a project made for Multicare, the goal of the project is to build a predictive model to predict health insurance claims. Throughout the work, the processes are similar and even the models used are common to our project, such as Random Forest, Decision Trees, etc.

2. Data exploration and Preprocessing

2.1 Initial Overview of the Data

To address this problem, we began by importing the required packages and datasets. At this stage our goal was to get an intuition of the data and gather key insights that could guide the following cleaning and preprocessing steps.

The training dataset contains 593,471 rows and 33 columns, while the test dataset contains 387,975 rows and 30 columns. Notably, the test dataset does not include the three target variables—*Claim Injury Type*, *Agreement Reached*, and *WCB Decision*.

A preliminary analysis revealed several important characteristics. The target variable *Claim Injury Type* is highly imbalanced, with the majority class dominating the dataset while minority classes are severely underrepresented. Approximately 3% of the target variable's entries are missing. Additionally, the *Zip Code* feature is stored as an object type, which is unexpected given its numeric composition. Features such as *C-3 Date*, *First Hearing Date*, and *IME-4 Count* have high proportions of missing data, while *OIICS Nature of Injury Description* contains no values at all. The *WCB Decision* feature is uniform across all records, containing a single constant value. Furthermore, some records indicate children reporting workplace injuries and having dependents, which seems strange given the dataset's origin in the United States.

Before proceeding with cleaning and preprocessing, duplicates in the *Claim Identifier* column were checked to ensure it could be used as the index, and rows with missing target variable values were dropped. Worth noting that all modifications made to the training dataset throughout this section were mirrored in the test dataset for consistency.

2.2 Description Variables

To handle the description variables, we leveraged information from the WCIO and NAICS websites to group these features, as outlined in *Table 1*:

Feature Name	Feature Description	Additional notes
Cause Severity	Groups *WCIO Cause of Injury Description*.	- Groups provided by the WCIO website. - *WCIO Part Of Body Code* had an invalid code (-9), that was replaced with 90 based on its distribution. (see *Figure 1* in the annex).
Body Part Risk	Groups *WCIO Part of Body Description*.	
Nature of Injury Risk	Groups *WCIO Nature of Injury Description*.	
Industry Category	Groups *Industry Code Description*.	- As per the NAICS website, *Industry Code* entries with the same description were grouped as one. - In the absence of broader NAICS grouping, industries were grouped based on our own interpretation of sectors with likely distinct injury or claim resolution profiles.

Table 1 - New Descriptive Variables

Note: Missing values in all code descriptions and their respective codes were replaced with 0, based on the distinct distribution of these codes relative to other groups, and allowing for easier identification and handling of these cases later in the process.

The *Carrier Name* feature included records where variations in formatting made identical entities appear different. We used the *rapidfuzz* library to identify and consolidate similar names into a single entity. At this stage, no further changes were made to the *Carrier Name* feature, as it contains many unique values.

Thereafter, the descriptive features *WCIO Cause of Injury Description*, *WCIO Part Of Body Description*, and *WCIO Nature of Injury Description* were dropped, as they were no longer necessary. The feature *OIICS Nature of Injury Description* was also removed, due to the absence of any information.

2.3 Variables with High Number of Missing Values

Rather than simply dropping the features with a high number of missing values, we addressed them as follows:

- ***C-3 Date:*** Replaced with a binary feature (*Claim Report Received*) indicating the presence (1) or absence (0) of the date; the original feature was dropped.

- ***First Hearing Date:*** Replaced with a binary feature (*Hearing Held*) indicating whether a hearing had occurred (1) or not (0); the original feature was dropped.
- ***IME-4 Count:*** Missing values were assumed to indicate no forms received, as the minimum value of the feature was 1. The missing values were replaced with 0, and the feature was maintained.

2.4 Date Variables

To address the date-related features (*Accident Date*, *Assembly Date*, and *C-2 Date*), we followed the steps outlined in *Figure 1*.

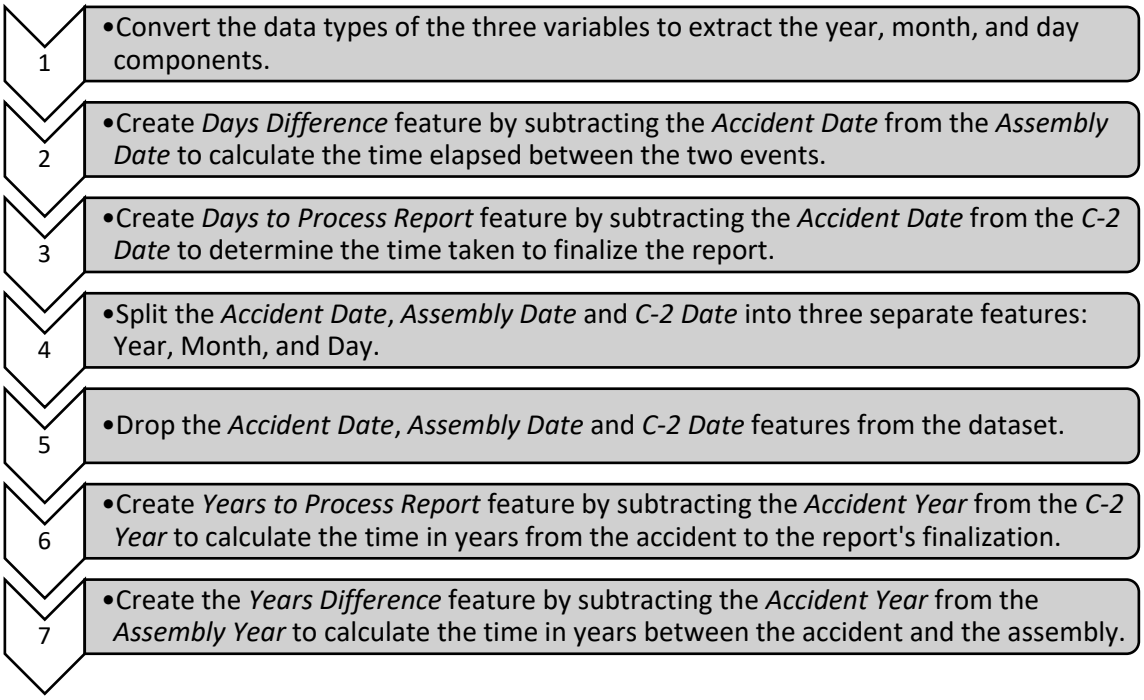

Figure 2 - Date Variables Process

Note: For all the new features, both dates must be present; otherwise, the value is set to *np.nan*. These variables provide a deeper understanding of the data by capturing related time intervals.

2.5 Categorical variables

The categorical variables were processed as follows: The *Covid-19 Indicator*, *Gender*, *Attorney Representative*, *Alternative Dispute Resolution*, and *Medical Fee Region* were label encoded, with "unknown" values replaced by missing values. The *Carrier Type* "unknown" value was replaced by 0, following the same approach applied to the WCIO variables, due to its distinct distribution. For the *County of Injury* and *District Name* variables, "unknown" values were replaced with missing values, and these will only be encoded after the dataset split to avoid data leakage. Lastly, the *WCB Decision* variable was dropped due to its univariate nature.

Next, we will focus on the categorical variables that require more detailed handling.

2.5.1 Zip Code

We found that the *Zip Code* feature is classified as an object because some zip codes contain non-numeric values. Furthermore, we found that these codes are from outside the USA. To address this, we grouped these non-numeric codes manually and considered creating an *Origin Country* feature, but this was discarded as most codes were from the USA. Ultimately, we assigned numeric labels to the groups, enabling conversion of the variable to float.

2.5.2 Birth Year, Accident Year & Age at Injury

The treatment here primarily involved fixing inconsistencies, leveraging the interconnection between these three variables, as any two can be used to derive the third. The changes made are outlined below:

- Rows with *Age at Injury* set to 0 were replaced with missing values, as we considered this value as invalid. The same process was then applied to *Birth Year*.
- When the *Age at Injury* feature was missing and the other two variables were available, we computed the correct value. The process was then applied to *Birth Year*.
- In some cases, the *Age at Injury* did not align with the difference between the *Accident Year* and *Birth Year*, even with a 1-year margin of error (e.g., a person born in 2003 could be either 20 or 21 years old). In these instances, the *Age at Injury* was adjusted to reflect the difference between the *Accident Year* and the *Birth Year*.

We then created a new feature, *Age Group*, which categorized individuals into groups, with each group's value representing the midpoint of the corresponding age range. The intervals between groups were kept consistent, with particular attention to separating "Minors" and "Elderly" (retirement age in the USA), as these age groups are less common in the workforce and relevant for workplace injury analysis.

2.5.3 Average Weekly Wage

The *Average Weekly Wage* variable contains many zero values, a problem which may indicate missing records. Further investigation revealed that records with a positive wage rarely correspond to a 'CANCELLED' or 'NON-COMP' claim (check *Figure 3* in the annex). Given the impact of wage values on claim outcomes, missing values will be handled post-split to avoid data leakage. Additionally, cases where *Industry Code* is missing (now set to 0) will be treated as unemployed or volunteers, and thus an *Average Weekly Wage* of 0 will be assigned to these records.

3. Multiclass Classification

3.1 Data Split

To prepare the dataset for the next steps, we label-encoded the target variable *Claim Injury Type*. Training started with a stratified split – since, as we had seen, the target variable was very

imbalanced. We split the data into 70% for training and 30% for validation, using random state (42) to ensure reproducibility and avoid bias. Worth noting that we excluded the feature *Agreement Reached* from the split, as this feature is not present in the test data and will not be predicted for now.

3.2 Feature Encoding and Preprocessing

At this stage, and although some variables have already been manually encoded, others remain in string format and require further processing. These were handled as follows:

- ***Carrier Name*:** This feature has many unique values, several of which have very low frequencies. However, we consider the Carrier to be an important factor influencing the outcome. Therefore, we assigned numerical labels to each Carrier, grouping those with fewer than 100 occurrences under "Other". For observations in *X_val* or the test set not present in *X_train*, we assigned a value of -1 to indicate new, unseen data. After encoding, the *Carrier Name* feature was renamed as just Carrier.
- ***County Of Injury:*** This feature still had missing values, which were imputed with the mode of the corresponding district. This seemed a suitable approach, since the district variable had no missing values. The variable was then encoded.
- ***District Name:*** This variable was encoded, following the same process and logic applied for *Carrier Name* and *County of Injury*. Variable was renamed as *District*.

Additionally, we created a new feature, *Zip Code Frequency*, which counts the number of occurrences of each zip code in the training data to assess the potential influence of frequent or rare zip codes on Carrier decisions. For new zip codes in the test or validation data, this variable is filled with 0 to indicate "new" locations.

3.3 Outliers

We began by categorizing the features into metric and non-metric groups to facilitate outlier treatment and enable more customized visualizations. Notably, the *Industry Code Description* feature will not be used in the analysis. It is retained solely to address its missing values, after which it will be removed.

Box plots for the metric features were plotted to identify outliers (see *Figure 4* in the annex), and manual limits were defined to exclude rare or extreme cases unlikely to be relevant for training:

- Accidents occurring before 2018 were excluded, as they represent distant events with minimal relevance to the model, as shown in the box plot.
- Cases where the *Average Weekly Wage* was greater than $50,000 were excluded, as it represents an extreme income bracket that is rare and unlikely to significantly impact the model. An extra margin was given, as the distribution will be adjusted later, with the more extreme values being converged.
- Ages under 13 or over 80 were excluded for different reasons:
    - Ages under 13, although not identified as outliers in the box plot, were removed due to data inconsistencies that likely result from erroneous data entries (only 32 observations, which are unlikely to affect predictions).

    - Ages over 80 were excluded as are actual outliers, and while accidents can occur at this age, their low frequency makes them less relevant for training.

- Extreme values for *IME-4 Count*, as observed in the boxplot, were also excluded.

For the treatment of outliers in these features, we applied the IQR method, with a few exceptions. For *Days to Process Report* and *Days Difference*, we only removed data above the 90th percentile. Although the boxplots indicate potential outliers, the distribution of these variables will be adjusted later, converging the extreme values. Similarly, outliers for *Years to Process Report* and *Years Difference* were not treated, as we have already restricted the dataset to accidents from 2018 onward, effectively removing any extreme values.

After these adjustments, we retained 91.86% of the data, well above the recommended 5% threshold. However, we firmly believe that the extreme values, which could bias our model, have been appropriately addressed (*Figure 5*, in the annex, shows the box plots after these changes).

Alongside this, we plotted histograms for the metric features (*Figure 6* in the annex), which provided several insights:

- *Age at Injury* is the feature closest to a normal distribution.
- Features like *Average Weekly Wage*, *Days Difference*, and *Days to Process Report* exhibit a right-skewed distribution and may benefit from smoothing for better model performance. While *IME-4 Count* also shows a right-skewed distribution, its values are closer to each other, so no adjustments are necessary.
- There are no longer any observations where the processing time exceeds one year.
- The feature *Number of Dependents* has an unusual distribution, with values ranging from 0 to 6. Notably, there is a disproportionate number of observations with 6 dependents, but no records exceed this value. Due to its lack of realism and precision, this variable will be removed.

Finally, we plotted histograms for the non-metric features. While most features exhibited expected distributions, it is worth mentioning that the feature *Nature of Injury Risk* is predominant in one specific value (1) compared to the rest.

3.4 Missing Values

For the remaining missing values, we decided to fill them with the mode, with one exception: *Average Weekly Wage*. We performed a t-test for numerical variables and a chi-square for categorical variables to identify which feature would have the highest weight in determining whether the wage was 0 or greater than 0. For the t-test, *IME-4 Count* had the most intriguing result, while for the chi-square highlighted *Attorney/Representative* as the most significant categorical variable.

We used this information to plot stacked histograms (*Figure7* and *Figure 8* in the annex) to visualize these relationships and found that *IME-4 Count* appeared to have a greater influence on wage values. Thus, we applied the following strategy for imputation:

- If *IME-4 Count* > 0, we filled missing *Average Weekly Wage* values with the mean wage based on the Industry Code.
- If *IME-4 Count* = 0, we imputed the missing wage values with 0.

We could now drop *Industry Code Description* as it was no longer required.

3.5 Changing Feature Distributions

After addressing missing values, we focused on handling right-skewed distributions identified during outlier treatment. For *Days to Process Report* and *Days Difference*, we applied a logarithmic transformation using *log(x+1)* to prevent negative values and maintain interpretability. For *Average Weekly Wage*, given the importance of distinguishing between values equal to 0 and those greater than 0, we applied a square root transformation instead of a logarithmic one. This approach smoothens the distribution while preserving the integrity of values closer to 0. These changes improved the distributions of these variables, making them more suited for model training (*Figure 9* and *Figure 10* show the changes made in these distributions).

Table 2, present in the annex, shows a summary of every variable we created, before we proceed with feature selection.

3.6 Feature Selection

Before proceeding with any feature selection analysis, it is crucial to scale the numerical values. We opted for *StandardScaler* over *MinMaxScaler* because the latter restricts the data to a range from 0 to 1. For variables like *Average Weekly Wage*, and even after some treatment to make values more convergent, such scaling would compress some values too close to zero, risking confusion with nulls. *StandardScaler*, on the other hand, provides a more even distribution. While null values no longer align at zero, this approach prevents extreme values from dominating, ensuring a more balanced representation.

We employed five feature selection techniques: *Spearman* correlation, *Lasso* regression, *Recursive Feature Elimination* (RFE), *Chi-Squared*, and *Random Forest*. The evaluation process involved assessing the outputs of these methods and making final decisions based on a majority vote. A feature was retained if the majority indicated "keep"; otherwise, it was discarded.

The table below lists the features ultimately selected. Refer to the annex for a comprehensive table of all features (*Table 3*).

FEATURE	SPEARMAN	RFE	LASSO	CHI-SQUARED	RANDOM FOREST	DECISION
AVERAGE WEEKLY WAGE	Keep	Keep	Keep	-	Keep	Include
BIRTH YEAR	Keep	Keep	Keep?	-	Keep	Include
DAYS TO PROCESS REPORT	Keep	Keep	Keep	-	Keep	Include
IME-4 COUNT	Keep	Keep	Keep	-	Keep	Include
MEDICAL FEE REGION	Keep	-	-	Keep	Discard	Include
WCIO NATURE OF INJURY CODE	Discard	-	-	Keep	Keep	Include
WCIO PART OF BODY CODE	Keep	-	-	Keep	Keep	Include
CLAIM REPORT RECEIVED	Keep	-	-	Keep	Keep	Include
HEARING HELD	Keep	-	-	Keep	Keep	Include
CARRIER TYPE	Keep	-	-	Keep	Discard	Include
ATTORNEY/REPRESENTATIVE	Keep	-	-	Keep	Keep	Include

Table 3 - Features kept after Feature Selection

An important case that deserves an explanation revolves around the features *Attorney/Representative*, *Hearing Held* and *Claim Report Received*. Despite a correlation of around 0.7 between the first and the latter two, we chose to retain all three. While collinearity exists, these features are deemed relevant to the final outcome. For instance, a case may have an attorney without a hearing, or a claim report may be submitted without a representative. Given their potential importance in the final decision - especially when the outcome is neither *Cancelled* nor *Non-Comp* - we decided to include all three in the models.

Additional considerations:

- Even when a feature received a majority "keep" vote, we examined its potential collinearity with other selected features. In such cases, the feature with the highest importance score from the *RandomForestClassifier* was retained, as this model is a good indicator given that we will also use it for prediction purposes. Table x provides an example with the variable *Days Difference*. Despite the majority vote to keep it, the feature was discarded due to multicollinearity with *Days to Process Report*. *Table 4* shows an example of this, with feature *Days Difference.*

FEATURE	SPEARMAN	RFE	LASSO	CHI-SQUARED	RANDOM FOREST	DECISION
DAYS DIFFERENCE	Discard	Keep	Keep	-	Keep	Discard (multicollinearity)

Table 4 - Feature discarded for Multicollinearity

- For ties between "discard" and "keep" votes, if any decision included uncertainty (e.g., marked with a question mark), the feature was classified as "discard" and excluded from the model. *Table 4* illustrates an example of this case, with feature *Assembly Month*.

FEATURE	SPEARMAN	RFE	LASSO	CHI-SQUARED	RANDOM FOREST	DECISION
ASSEMBLY MONTH	Discard	Keep	Keep?	-	Discard	Discard

Table 5 - Feature discarded due to tie

Finally, check *Figure 11* in the annex for a *Spearman* correlation matrix of the 11 variables kept.

3.7 Models

For model selection, we implemented five algorithms: *Decision Trees*, *Random Forest*, *XGBoost*, *LightGBM*, and *Extremely Randomized Trees*. Our goal was to compare the performance of each model using just a few parameters with the same values. The models with the best performance will be the focus for further tuning to achieve better scores. The models chosen for this evaluation were based on Decision Trees, as it is one of a few models which can deal with nominal variables and perform well on multiclass classification problems.

We used cross validation to apply the models, for two key reasons. First, it provides more reliable results since the performance is based on multiple splits. Second, it gives a better generalization of data by testing the model on different data subsets.

After reviewing the scores of these models (*Figure 12* in the annex), we chose to focus on *XGBoost* and *LightGBM* due to their high overfitting. We plotted a box plot to compare these two models, as shown below:

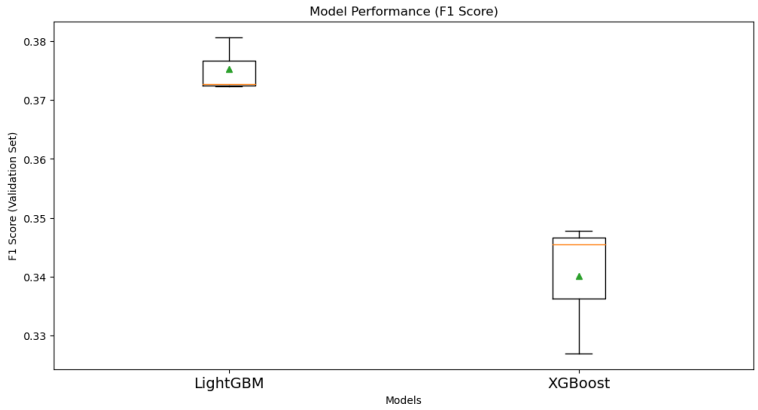

Figure 13 - Evaluation of the models

LightGBM clearly indicates better results compared to *XGBoost* so we will stick with this model for optimization. We will now fine-tune the model parameters to reduce it and achieve better results. Below are summarized our findings for each parameter

- ***Max Depth*:** Perhaps one of the key parameters for controlling overfitting, it is clear to see (annex *Figure 14*) that the model stagnates around the value 10. Therefore, that is the ideal" value for our model.
- ***Number of Estimators*:** Our goal is to quickly identify the optimal number of estimators, which determines the number of trees in the model. Fewer estimators reduce *LightGBM*'s computational cost. As shown in Figure 15 (annex), performance plateaus around 400, with minimal improvement at 500. Given the dataset's complexity, we prioritize execution speed over negligible gains.
- ***Learning Rate*:** After plotting the box plot, it is easy to spot the drop once we reach 0.12 (annex *Figure 16*). With this in mind, we will stick with the value of 0.1 for our model as it shows the best result.
- ***Min_Child_Samples*:** This parameter specifies the minimum number of samples required to create a new child node in a tree. The box plot revealed that, despite some stagnation at around 50, we chose min_child_samples = 100 to ensure no possible overfitting in the test data. As such, that is the value we will select (*Figure 17* in the annex for the box plot).

Thus, we achieved our final model, *LightGBM* with the parameters outlined above. This model achieved an F1 score of 0.4133 on the validation set. Check Annex for the resulting Confusion Matrix (*Figure 18*).

4. Open Ended Section

After obtaining all the results from the models, we thought it would be interesting to have a practical and simple application of the work done. To do this, we thought of a Web App. Our goal was to use the notebook we had developed as a database so that, by entering the inputs (variables) into an application, it could generate the desired output (prediction).

The development process began by choosing the support with which we were going to develop the application. By taking advantage of the *pickle* library, we were able to create a copy of the model developed and to choose the previously selected features. We then turned to *Streamlit*, a powerful tool for building interactive web applications with Python, to start developing the user interface. With these tools, we created the files to load into the application.

The application is structured into three main sections: the first section provides a small presentation of the project, giving users an overview of the work. The second section focuses on the prediction functionality, where users can input data and receive predictions. Three different models are used for these predictions, and the user **can compare the results of each in a graph.** There are also some pre-defined examples of scenarios, so you don't need to fill all the information. Finally, the third section displays a visualization of the distribution of the ages of workers.

5. Conclusion

With the completion of this project, we were faced with many obstacles that ended up being resolved or not.

Considering this, the entire data set ended up requiring pre-processing that, in general, not only filled in a lot of missing data but also made all the data consistent and making sense (there were now no Assemblies before Accidents, something that was checked).

One of the things also discovered throughout this exploration was the fact that the absence or absence of data on each feature almost always had a greater impact than simply a forgotten input. A missing value often meant a specific Claim Injury Type value and therefore the way in which missing values were handled was a bit beyond the usual where we usually, perhaps, would fill it in with the most common value of that feature instead of taking any precautionary measures.

Finally, although we believe that the model we use to predict the target in the test data is the most optimized and the one that presents the best results. It is also worth noting its inefficiency when it comes to constant and accurate forecasts. For example, the model created by us has some difficulties in specifying the values corresponding to the value "3. MED ONLY" and, instead, usually predicts "2. NON COMP" or "4. TEMPORARY" instead.

Annex

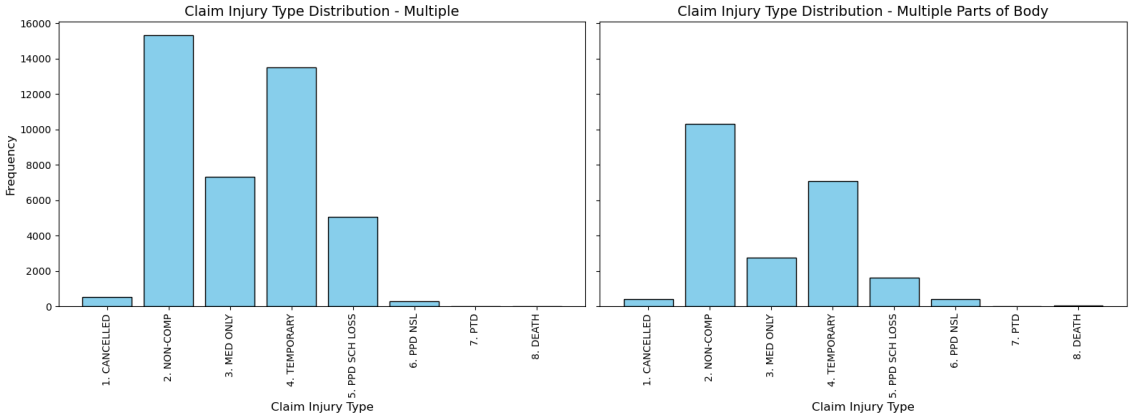

Figure 2 - Comparison between code 9 and 90

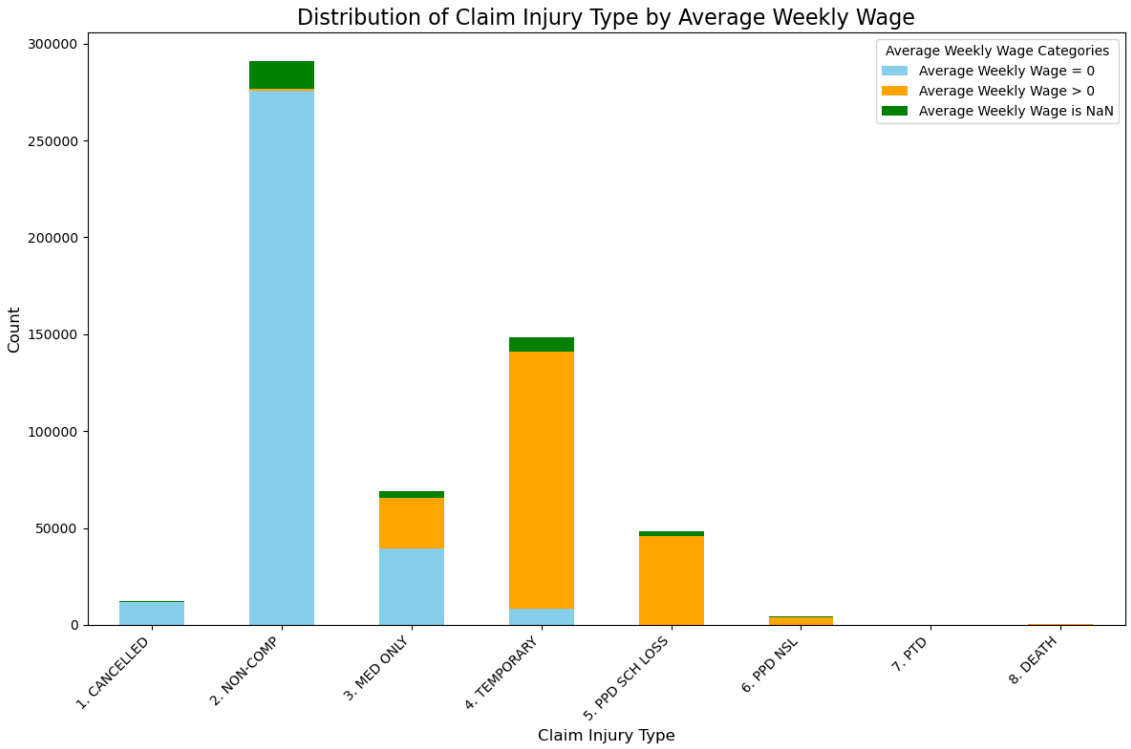

Figure 3 - Distribution of Claim Injury Type by Average Weekly Wage

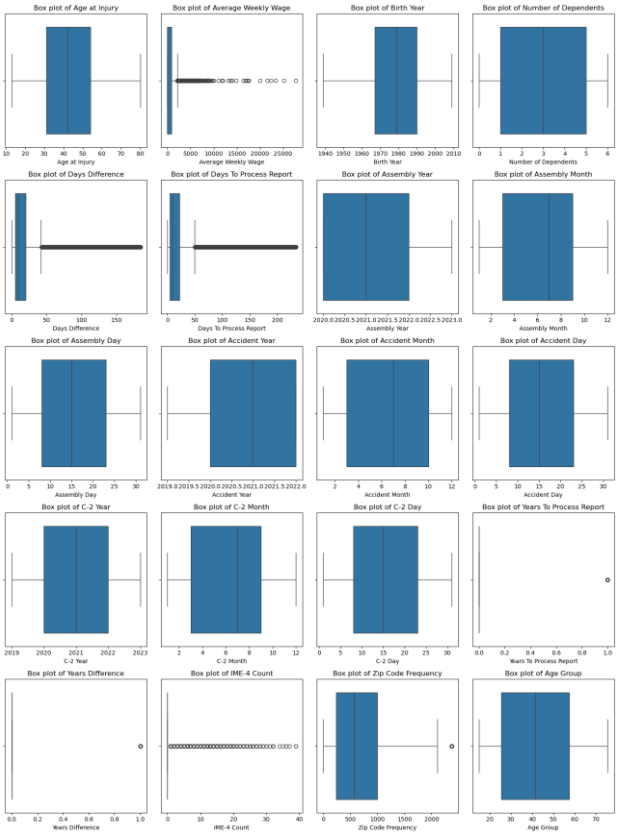

Figure 4 - Box plots before adjustments

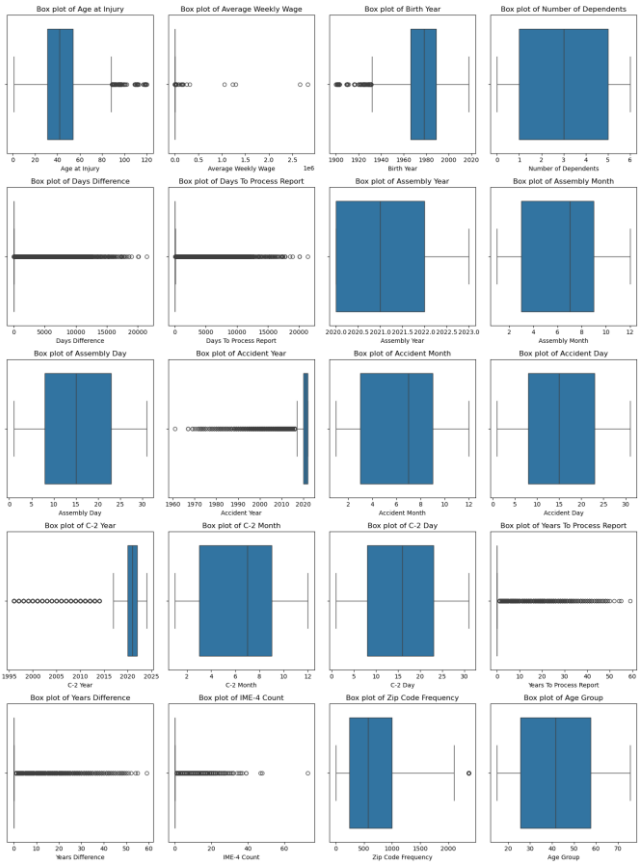

Figure 5 - Box plots after adjustments

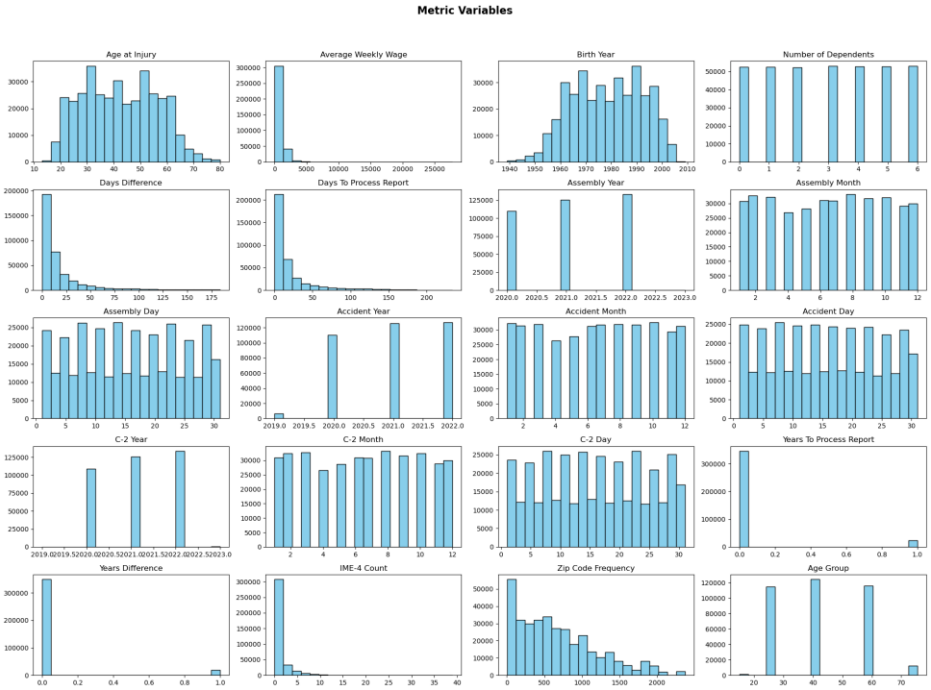

Figure 6 - Histograms for metric features

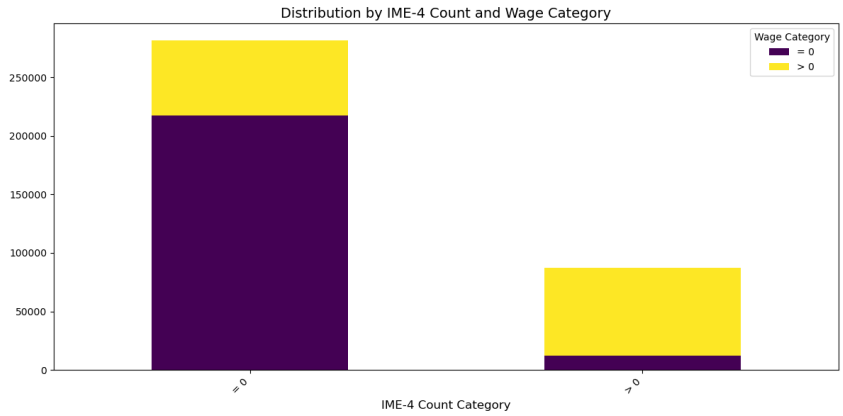

Figure 7 - Distribution by IME-4 Count and Wage category

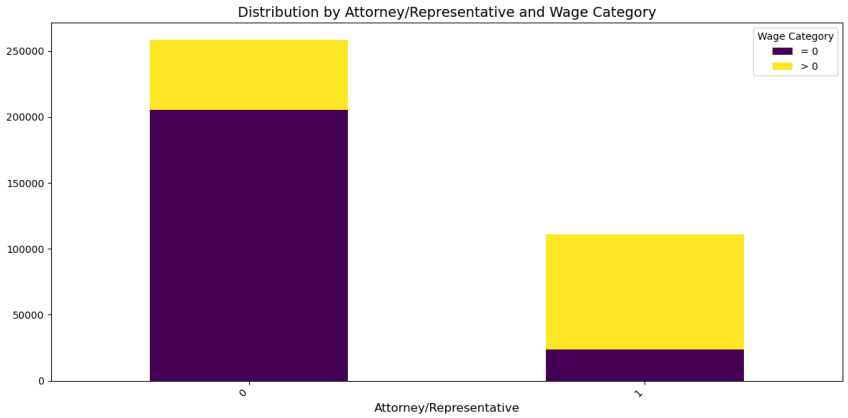

Figure 8 - Distribution by Attorney/Representative and Wage Category

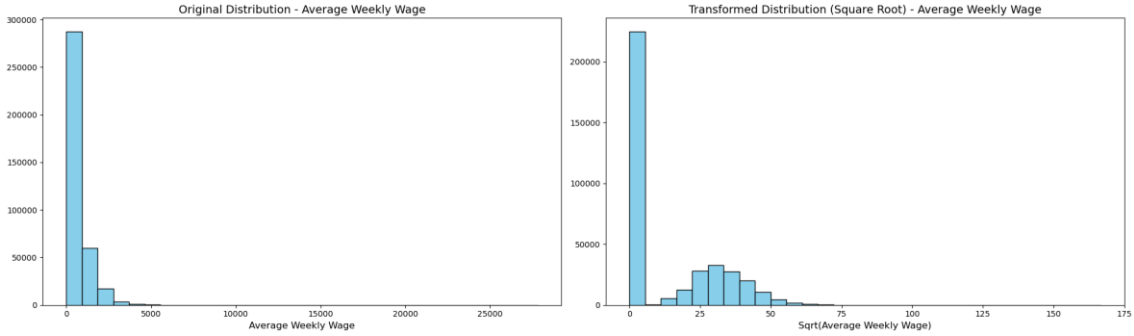

Figure 9 - Change in distribution Average Weekly Wage

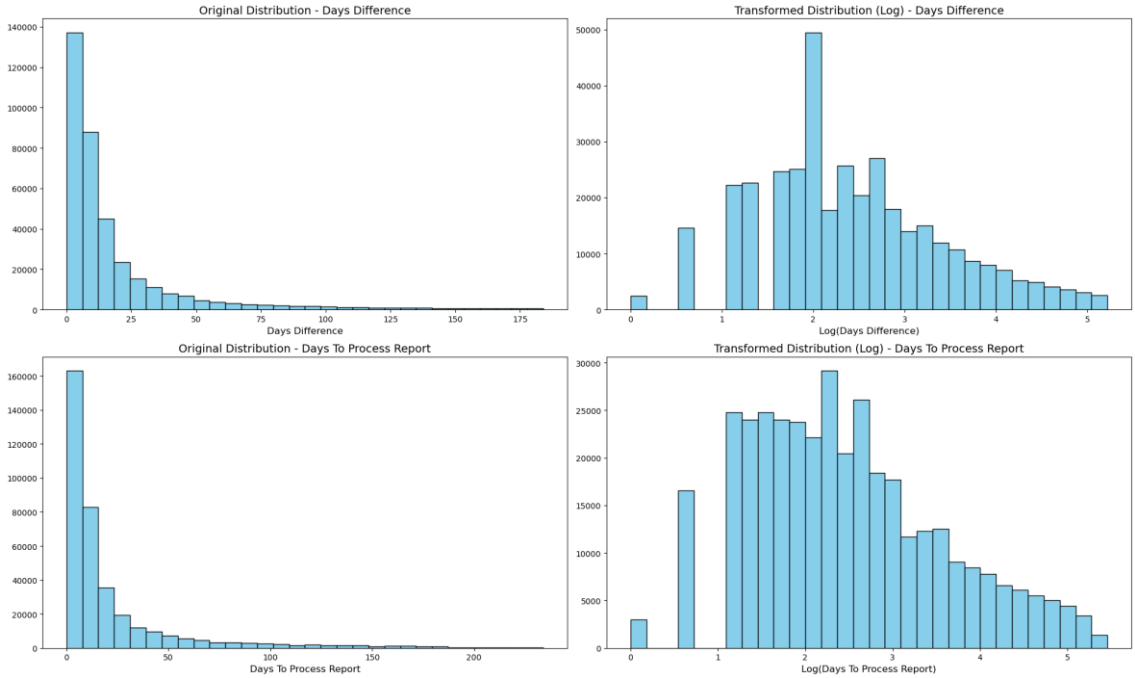

Figure 10 - Change in distribution Days Difference and Days to Process Report

Feature Name	Feature Description	Purpose
Cause Severity	Set causes of injury into numeric categories, from 0 to 10: other, burns, caught, cuts, falls, motor vehicle, strain, striking against, struck, rubbed, miscellaneous, respectively.	Simplification of the WCIO Cause of Injury Description
Body Part Risk	Set body parts into numeric categories, from 1 to 6: head, neck, upper extremities, trunk, lower extremities and multiple body parts, respectively, with 0 for other.	Simplification of the WCIO Part Of Body Description
Nature of Injury Risk	Set nature of injuries into numeric categories: 1 (specific injuries), 2 (occupational disease or cumulative injuries), 3 (multiple injuries), and 0 (other).	Simplification of the WCIO Nature of Injury Description
Industry Category	Industries divided by 5 categories. (Public Sector, Services Commerce, Productive Sector, Technology and Finace and other)	Simplification of Industry Code Description
Claim Report Received	Binary feature indicating whether a claim report (C-3) was received: 1 if the C-3 Date has a value, 0 if it does not.	Simplification of C-3 Date.
Hearing Held	Binary feature indicating whether a hearing was held: 1 if the First Hearing Date has a value, 0 if it does not.	Simplification of First Hearing Date
Days Difference	Number of days between Assembly Date and Accident Date	Obtaining the numbers of days passed between these dates
Days to Process Report	Number of days between C-2 Date and Accident Date	Obtaining the numbers of days passed between these dates
Origin County	Simplification of Zip Code, with a split in three regions	Simplification of the Zip Code (dropped latter due to low relevance)
Assembly Year, Month and Day	Simplification of Assembly Date, with a split in different features	Splitting of the Assembly Date to simplify operations using these features
Accident Year, Month and Day	Simplification of Accident Date, with the splitting in different features	Splitting the Accident Date to simplify operations using these features
C-2 Year, Month and Day	Simplification of C-2 Date, with the splitting in different features	Splitting the C-2 Date to simplify operations using these features
Years to Process Report	Number of years between the C-2 Year and the Accident Year	Obtaining the numbers of years passed between these dates
Years Difference	Number of years between the Assembly Year and the Accident Year	Obtaining the numbers of years passed between these dates
1Age Group	Feature that returns the midpoint of the group age where the person belongs	Simplification of Age at Injury by dividing ages in groups
District	Maps each unique district to a numeric value. Unseen districts are encoded as -1.	Simplification of District Name
Carrier	Maps each unique carrier to a numeric value, grouping low-frequency carriers into the category 'Other'. Unseen Carriers are encoded as -1.	Simplification of Carrier Name
Zip Code Frequency	Calculates the frequency of each Zip Code	Simplification of Zip Code

Table 2 - List with all new features

FEATURE	SPEARMAN	RFE	LASSO	CHI-SQUARED		RANDOM FOREST	DECISION
AGE AT INJURY	Keep	Discard	Discard	-		Discard	Discard
AVERAGE WEEKLY WAGE	Keep	Keep	Keep	-		Keep	Include
BIRTH YEAR	Keep	Keep	Keep?	-		Keep	Include
DAYS DIFFERENCE	Discard	Keep	Keep	-		Keep	Discard *
DAYS TO PROCESS REPORT	Keep	Keep	Keep	-		Keep	Include
ASSEMBLY YEAR	Keep	Discard	Keep	-		Discard	Discard*
ASSEMBLY MONTH	Discard	Keep	Keep?	-		Discard	Discard
ASSEMBLY DAY	Discard	Discard	Discard	-		Discard	Discard
ACCIDENT YEAR	Keep	Discard	Discard	-		Discard	Discard
ACCIDENT MONTH	Discard	Discard	Discard	-		Discard	Discard
ACCIDENT DAY	Discard	Discard	Discard	-		Discard	Discard
C-2 YEAR	Discard	Keep	Discard	-		Discard	Discard
C-2 MONTH	Discard	Discard	Discard	-		Discard	Discard
C-2 DAY	Discard	Discard	Discard	-		Discard	Discard
YEARS TO PROCESS REPORT	Discard	Discard	Discard?	-		Discard	Discard
YEARS DIFFERENCE	Discard	Discard	Keep?	-		Discard	Discard
IME-4 COUNT	Keep	Keep	Keep	-		Keep	Include
ZIP CODE FREQUENCY	Discard	Keep	Discard?	-		Discard	Discard
AGE GROUP	Keep	Discard	Keep?	-		Discard	Discard *
ALTERNATIVE DISPUTE RESOLUTION	Discard	-	-	Discard		Discard	Discard
ATTORNEY / REPRESENTATIVE	Keep		-	-	Keep	Keep	Include? *
CARRIER TYPE	Discard		-	-	Keep	Discard	Discard
COUNTY OF INJURY	Discard		-	-	Keep	Discard	Discard
COVID-19 INDICATOR	Keep		-	-	Keep	Discard	Discard
GENDER	Keep		-	-	Keep	Discard	Discard
INDUSTRY CODE	Discard		-	-	Keep	Discard	Discard
MEDICAL FEE REGION	Keep		-	-	Keep	Discard	Include
WCIO CAUSE OF INJURY CODE	Discard		-	-	Keep	Discard	Include
WCIO NATURE OF INJURY CODE	Discard		-	-	Keep	Keep	Include
WCIO PART OF BODY CODE	Keep		-	-	Keep	Keep	Include
ZIP CODE	Discard		-	-	Keep	Discard	Discard
CAUSE SEVERITY	Discard		-	-	Keep	Discard	Discard *
BODY PART RISK	Keep		-	-	Keep	Discard	Discard *
NATURE OF INJURY RISK	Keep		-	-	Keep	Discard	Discard *
INDUSTRY CATEGORY	Discard		-	-	Keep	Discard	Discard
CLAIM REPORT RECEIVED	Keep		-	-	Keep	Keep	Include
HEARING HELD	Keep		-	-	Keep	Discard	Include

Table 6 – List with results of Feature Selection

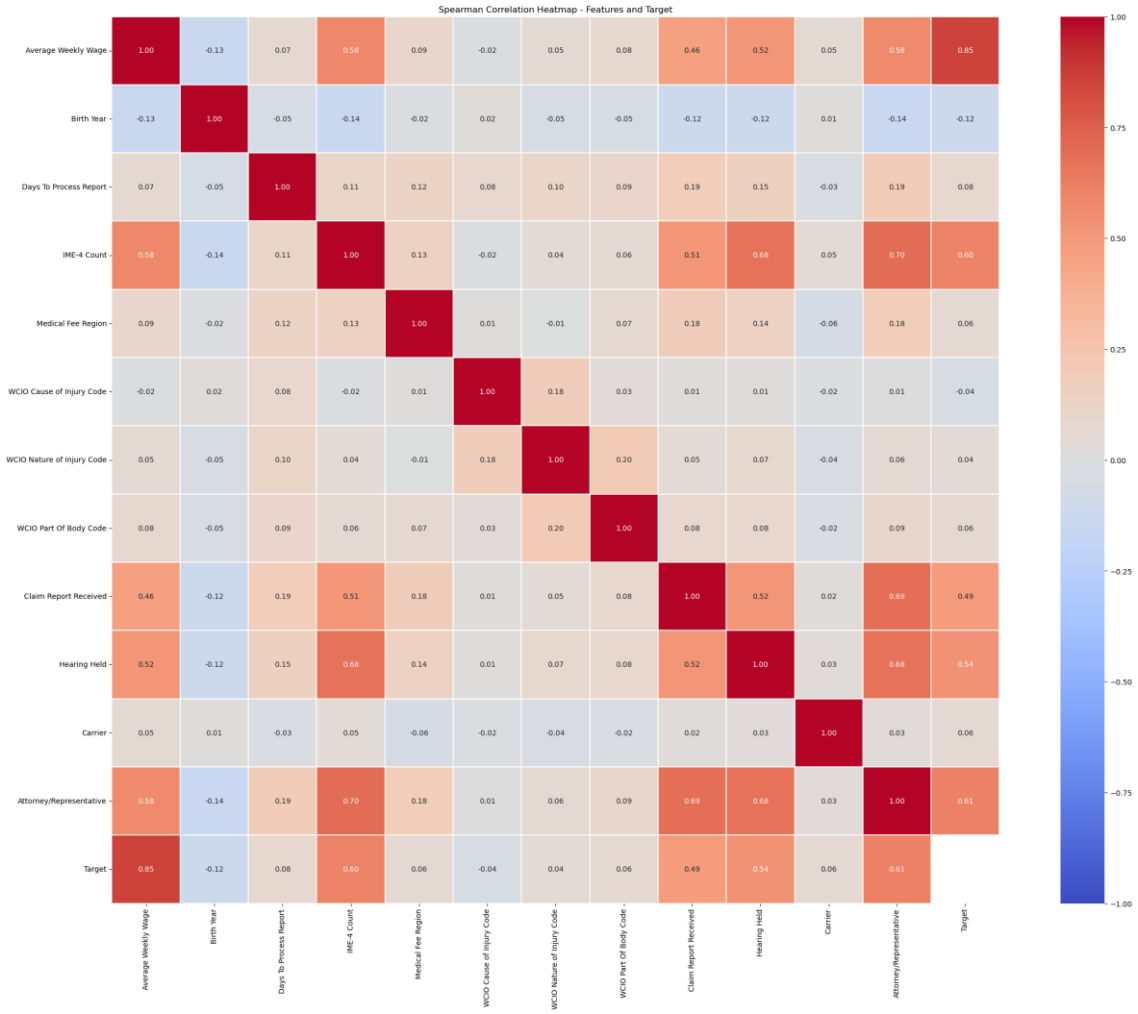

Figure 11 - Spearman Correlation Matrix with selected features

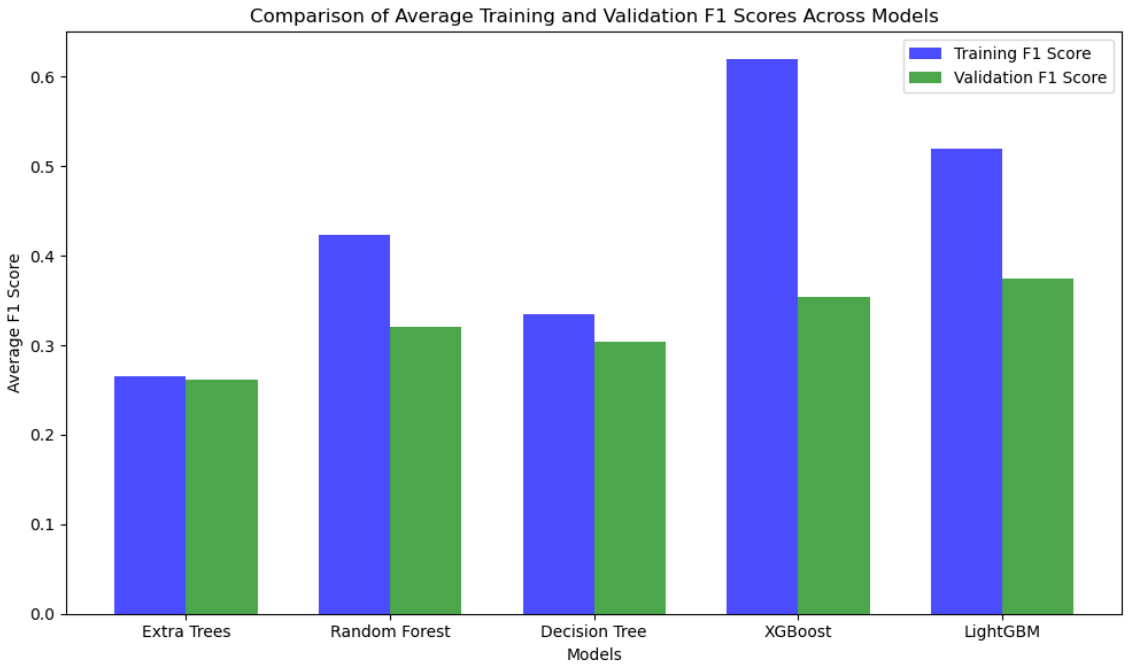

Figure 12 - Comparison of the scores of the models

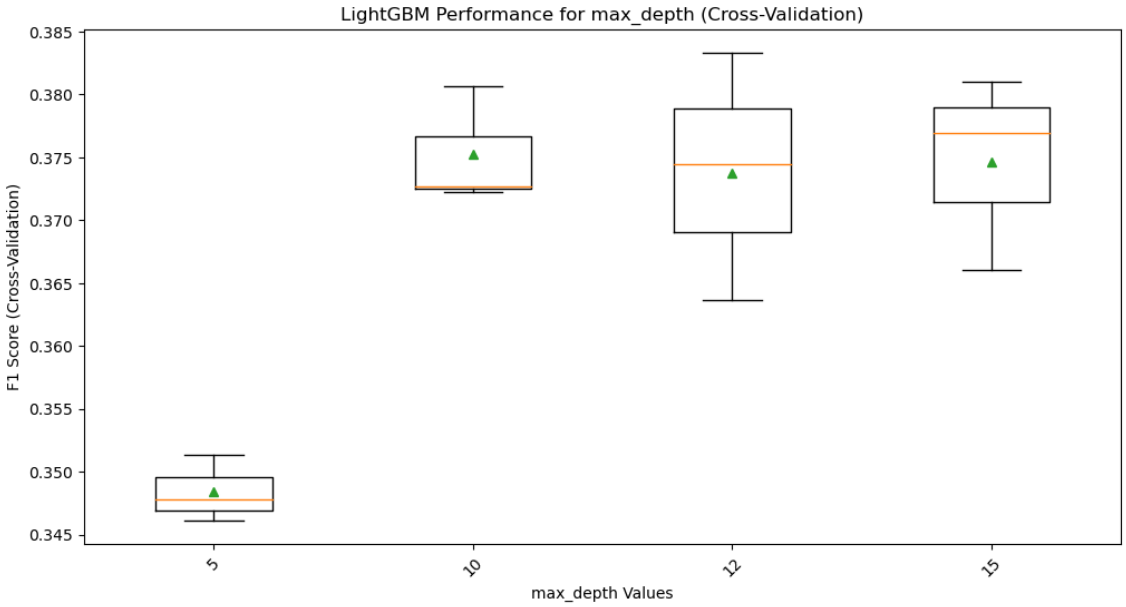

Figure 14 - Box plot LightGBM performance for Max Depth

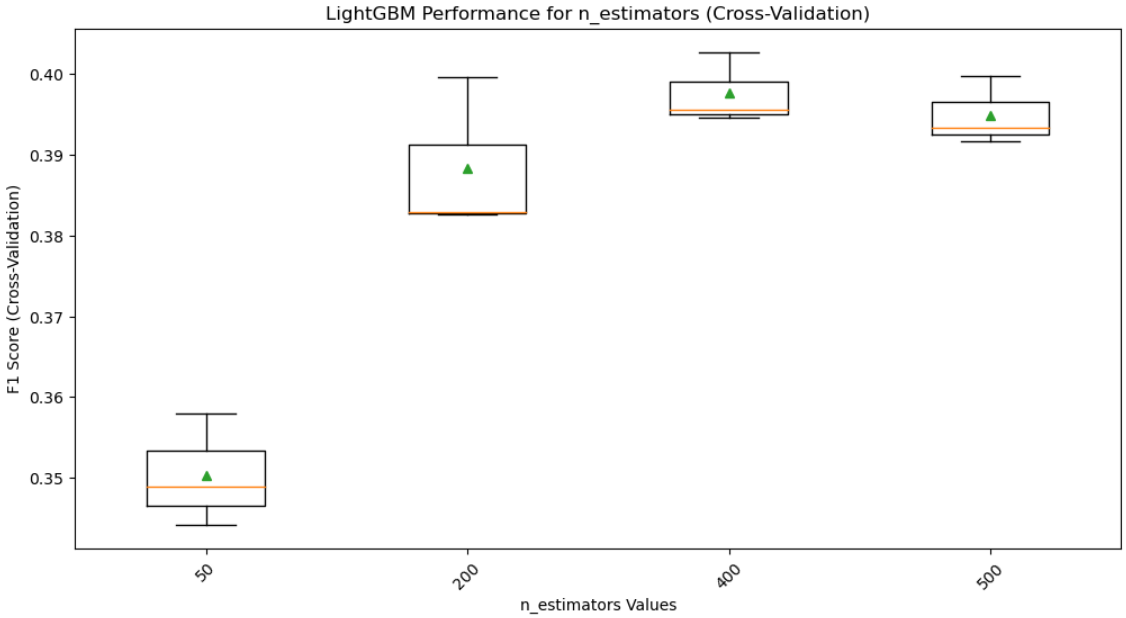

Figure 15 - Box plot LightGBM performance for number of estimators

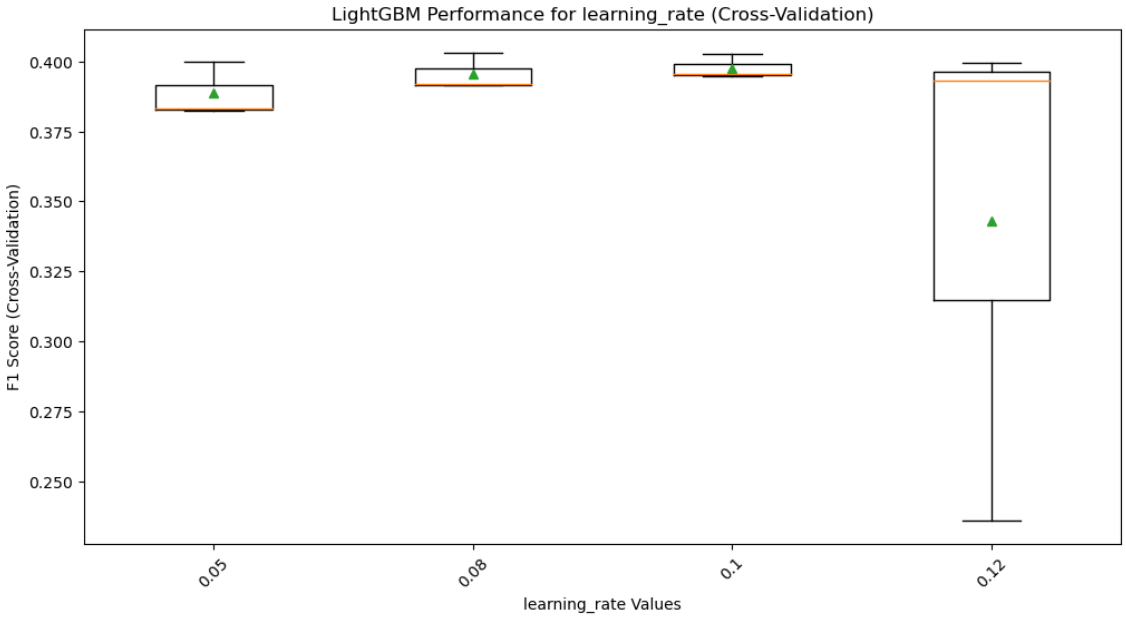

Figure 16 - Box plot LightGBM performance for Learning Rate

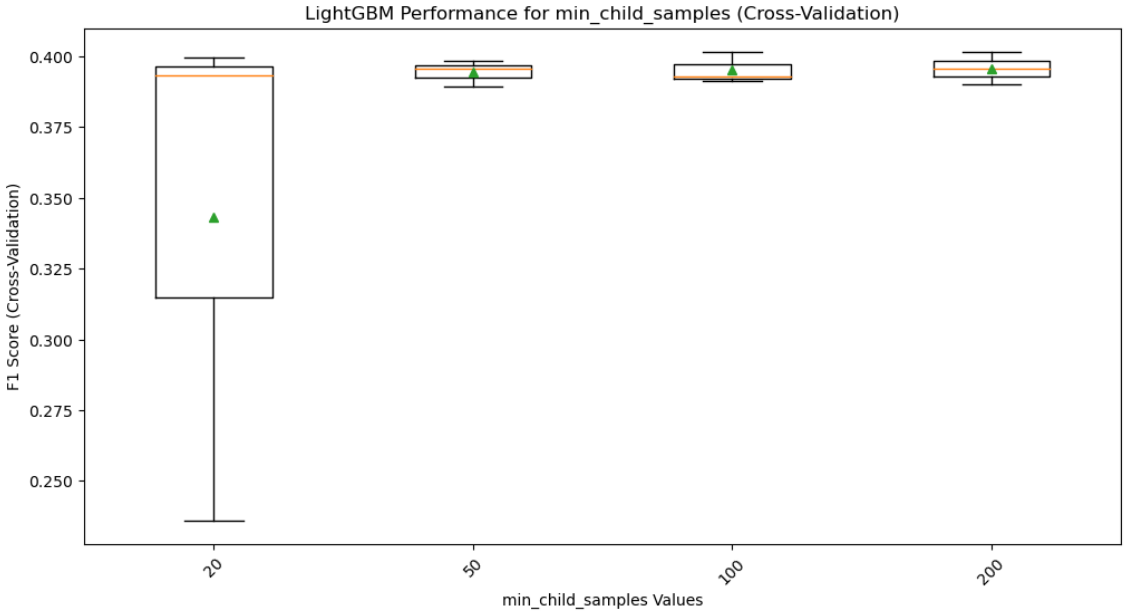

Figure 17 - Box plot LightGBM performance for Min_Child_Samples

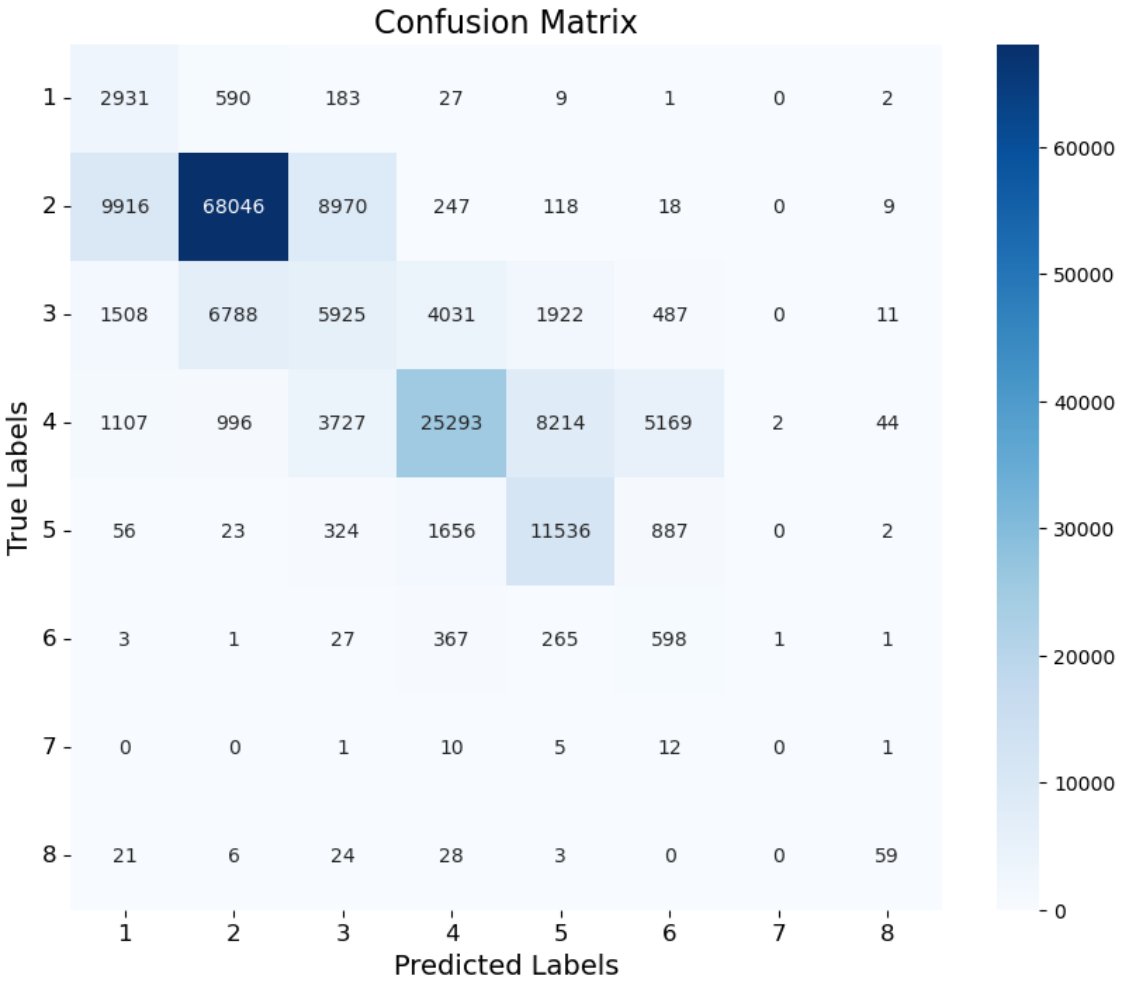

Figure 18 - Confusion Matrix

References

https://run.unl.pt/bitstream/10362/148525/1/TCDMAA2882.pdf

https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMClassifier.html

https://xgboost.readthedocs.io/en/stable/

https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestClassifier.html

https://xgboost.readthedocs.io/en/release_0.90/parameter.html

https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LassoCV.html