MACHINE LEARNING PROJECT

Master's in Data Science and Advanced Analytics

NOVA Information Management School

Universidade Nova de Lisboa

WCB Project Report

Group 20

Duarte Nunes, 20240564

Mariana Gomes, 20211689

Pedro Gaspar, 20240112

Rodrigo Nascimento, 20240565

Yasmine Boubezari, 20230775

Fall Semester 2024-2025

Abstract:

This project focuses on building a predictive model for the automation of claim injury types classification for the New York Workers' Compensation Board (WCB), based on claims data registered from 2020 to 2022. We found similar approaches in the real world that used machine learning models to detect fraudulent claims in real-time reducing fraud-related costs while speeding up review times as well as to automate the analysis of claims documents reducing human errors and maximizing operational efficiency

To reach an accurate multiclass classification model that can be easily and effectively used by WCB workers, the structure approach included data exploration and preprocessing, feature engineering, encoding and scaling, feature selection, model assessment, hyperparameters optimization, final model training and an analytics interface. While dealing with raw data we treated the missing values, incoherencies and outliers found in the appropriate manner and by imputation, like for "Age at Injury" and "IME-4 Count". On the opposite some new variables were created from our data to detect more insights like "Injury Severity" and "Claim Complexity". In feature selection we used filter, wrapped and embedded methods, where RFE with logistic regression gave a result of 27 features to keep, and 30 features to keep for RFE with Ridge classifier, which after comparing all methods lead us to drop 6 features from 33.

The model training conducted on a sample of 10000 observations with different models gave the best weighted F1-scores, 0.712, 0.717 and 0.719 with Random Forest, Gradient Boosting Classifier and XGBoost Classifier, respectively. After the hyperparameters showing the combination "gamma: 0.2, learning_rate: 0.1, max_depth: 5 and n_estimators: 150", for a macro F1-score of 0.437 for the XCBoost Classifier this model was picked up for the final training on the whole dataset.

The interface created by TKInter library helps non-technical workers to easily receive the injury type output, by simply inputting the claim details into the interface.

The kaggle result showed a macro F1 score of 0.34865, which confirms the capacity of the model to predict, with some bias due to unbalanced data.

Key words : Machine learning, classification problem, multivariate analysis, Workers' Compensation Board (WCB), Workers' Compensation Claims, Imbalanced Data Handling, Gradient Boosting Classifier, Claims Automation.

1. Introduction

The New York Workers' Compensation Board (WCB) processes millions of claims annually, a task that is both resource-intensive and time-consuming when handled manually. The increasing complexity of these claims, additional to the need for rapid decision-making, has made automation a pressing need for the WCB.

Our objective in this project is to address this challenge by using machine learning tools to automate the classification of injury types from workers' compensation claims, determine claim eligibility and injury type classification, by building, training and validating a predictive model, and test its performance through a Kaggle competition.

The project is based on a train and a test datasets covering descriptive information about the claim details, worker demographics, and administrative processes.

The training dataset contains claims data from the start of 2020 to the end of 2022. Each claim is associated with a range of descriptive attributes and three key target variables, including the Claim Injury Type, which is the focus of this project. It provides us with the necessary labeled data for building and validating our models and be able to build our final model to predict the target variable.

On the other hand, the test dataset includes claims data from January 2023 onward. It contains the same descriptive attributes as the first one but lacks the target variables to ensure that it will simulate the predictions right.

Similar initiatives in the insurance industry demonstrate the significant impact of automation and predictive models in optimizing claims management. For example, companies like **Anadolu Sigorta** have implemented machine learning systems to detect fraudulent claims in real-time. This has enabled them to reduce fraud-related costs while speeding up review times. These predictive tools analyze massive datasets to identify abnormal patterns in claims, providing fast and accurate analysis that was previously impossible through manual processes. [2]

Another relevant example is **AXA's** approach, where machine learning and deep learning models were used to automate the analysis of claims documents. This solution has not only streamlined the claims processing but also reduced human errors and maximized operational efficiency. These examples illustrate how similar techniques to those in our project can transform claims management, making processes more efficient and precise. [1]

Our final expectation is to deliver a robust and generalizable machine learning model solution capable of accurately classifying claims into their respective **Claim Injury Type** categories, which will be used to facilitate the decision-making process, reduce the reliance on manual intervention, assist in prioritizing cases requiring immediate attention, and hopefully improve the accuracy of claim evaluations.

2. Data Exploration and Pre-processing

2.1 Initial dataset analysis:

After importing the required libraries and our dataset to the notebook, we started exploring our dataset, reviewing its structure, column names, and overall properties. We initially had **33 features**, of which **12 are numerical** and **21 are categorical**, with a total of **593471 unique observations**. In a logical and contextual way, we can divide our features into:

- **Claim Details-related,** which directly describe the event and the circumstances of the claim, include features like “Accident Date”, “County of Injury”, “COVID-19 Indicator”, “WCIO Cause of Injury Description”, “WCIO Nature of Injury Description”, “WCIO Part Of Body Description”.
- **Worker Demographics-related features**, which outline characteristics of the injured worker, include features like “Age at Injury”, “Average Weekly Wage”, “Birth Year”, and “Gender”.
- **Administrative Processes-related features**, which focus on the management and evaluation of claims, include features like “Alternative Dispute Resolution”, “Attorney/Representative”, “Carrier Name”, and “IME-4 Count”.

By using the descriptive statistics and functions like **“.describe()”**, we discovered pertinent information about our features, like the high presence of NaN values, the average age of the claimers that is situated at 42, and some incoherencies on the same variable “Age at Injury” like 117 for the maximum age and 00 for the minimum as well as a negative code (-9.0) in “WCIO Part Of Body Code”, regarding “Age at Injury” we treated in accordance with “Birth Year”, because observations with a value of zero in “Age at Injury” but with a non-null number in “Birth Year” and vice-versa attracted our attention and so we decided to give the most correct treatment, in our perspective, to the values zero and NaN in these features. In this section, we only observed the claims where both columns were zero and we changed the values of only one column to NaN values , in this case “Age at Injury”, however we realized that if we transformed both it would still work in the function we later defined to treat zeros and missing values in these features, we kept it like this just to avoid running the entire notebook again.

We also noticed that the number of claims was very high on the date 2020-03-01, which coincides with the first days of the explosion of the covid epidemic expansion. To confirm this hypothesis, we crossed this date with the feature “COVID-19 Indicator” and appeared that most positive results happened on the 2020-03-01 as we see on Figure 1.

For deeper and clearer understanding of the given data, we used a pie chart to visualize the frequency of our target variable, which showed that the “NON-COMP” represents slightly more than half (50.7 percent) of the injury types claimed, followed by “TEMPORARY” (25.9 percent), then “MED ONLY” (12 percent) , and finally the 3 others left with less than 10 percent each. (Figure 2)

We also studied the distribution of “Gender” on these types and it appeared correlated for all genders as showed in Figure 3. For this variable, we assumed the categories were well-defined (“M” for Male, “F” for Female, “U” for Unknown, and “X” for Non-Binary).

By inversing to the distribution of the Claim Injury type on the Gender feature, we found that “M” (presumed for male) and “F” (presumed for female) were majoritarian, where “M” was more concerned in all claims types hich can be visible on Figure 4 .

We found too that medical fees in region “IV” seem to be very high compared to the others, on the opposite of “UK” where they seem the lowest. (Figure 5)

Curious too about the link between the industry and the claim injury types, we dived into a distribution and found that “HEALTH CARE AND SOCIAL ASSISTANCE” represents the highest industry injured and mostly on the “NON-COMP” injury type, followed by “PUBLIC ADMINISTRATION” and then “RETAIL TRADE”. If we connect it with the findings about gender distribution, it might signify that males are more present in these industries. (Figure 6)

When it comes to the age of claimers, we noticed the presence of some outliers as observed on Figure 7, which will later be treated. From this first analysis of our data, no major issues were found apart from the presence of missing values and outliers.

2.2 Multivariate Analysis:

The pairplot matrix on Figure 8 allowed us to explore and analyze the relationships between the variables described in the dataset, where we found interesting patterns:

- The diagonal of the pairplot shows univariate distributions. With this, we found that "Age at Injury" appears to follow an approximately normal distribution with peaks at certain age ranges, pointing to the concentration of injuries among middle-aged workers.
- The "Average Weekly Wage" seems skewed, which shows the presence of workers earning above the average wage.
- The colors in the plot suggest good separation between some categories, meaning variables like "Claim Injury Type" are strongly correlated with other predictive variables.

  Relationships between variables:

  1- Age at Injury vs. Claim Injury Type: There is a pattern where certain injury types are more common in specific age groups. For example, younger workers might exhibit specific types of injuries related to physical accidents, whereas older workers might show a higher prevalence of occupational diseases.

  2- Average Weekly Wage vs. Claim Injury Type: There is a tendency for less severe injury types to be associated with workers earning lower average wages. This suggests a possible relationship between injury severity and wage level.

  3- COVID-19 Indicator vs. Claim Injury Type: For claims related to COVID-19, a specific concentration of injuries or occupational diseases linked to pandemic conditions are observed.

- Classes Represented by Colors: The matrix shows distinct clusters for certain variable combinations, such as "Age at Injury" and "Medical Fee Region." which can be explained by highly discriminative combinations that may be useful for classification models.
- Correlation Between Continuous Variables: "Age at Injury" and "Year of Birth" show an expected inverse correlation. Additionally, variables like "Average Weekly Wage" may have a weak correlation with temporal variables, pointing out that economic data is relatively stable over time. (Figure 9)
- Outliers and Anomalies: continuous variables like "Average Weekly Wage" present isolated points in the scatterplots, indicating the presence outliers that could either be noise or exceptional cases that should be evaluated and treated to prevent negative impacts on the models.

  We conclude from this plot that variables like "Age at Injury," "Average Weekly Wage," and "COVID-19 Indicator" show significant visual relationships with "Claim Injury Type" and should be prioritized as predictors in the models.

2.3 Cleaning and Pre-Processing:

After having an idea of our data regarding the type, the frequency, its dynamics and also its context, we now go to the cleaning and preparation of data. To better treat our data, we decided that numerical variables containing codes like “Claim Identifier” and “WCIO Cause of Injury Code” were going to have their types changed to objects, because even though their inputs were numeric what they representing was not numeric but categorical, given that this codes represent the descriptions of the corresponding variables, such as, “Industry Code” representing “Industry Code Description”.

- **Missing values:**

Missing values were addressed systematically across train, validation, and test sets, but all transformations were calculated based solely on the train set to avoid data leakage. The same imputation logic was then applied to the validation and test sets without recalculating. This means that the missing values imputation made in the validation and test sets, was based on the information of the training data.

To start we checked variables with more than 50**%** missing values (Figure 10) and then checked according to their descriptions if it made sense eliminating them or not, and we ended up dropping 'IME-4 Count' and 'OIICS Nature of Injury Description' because of their redundancy and irrelevancy regarding our goal, given that missing values didn’t represent anything for these features and having more than 50% of the data missing is not optimal. Variables like ”C-3 Date” and ”First Hearing Date” were kept, as their missing values were reasonable (e.g. events that had not yet occurred).

For the **Missing Date Variables**, they were imputed with context-specific values, because based on our analysis of the descriptions of all these date variables we understood that there was a chronological relation between them and so we could not simply impute missing values in these variables with the mode. Given this, we started with “Accident Date” for which we made a function that would create a random number to simulate the difference between “Accident Date” and “Assembly Date” (which needs to be positive) and then impute the missing values of Accident Date by taking out from the corresponding Assembly Date that difference, since the “Accident Date” cannot be after the “Assembly Date”. Then for the other variables as we referred we imputed them based on logical reasoning, such as, ”No hearing has taken place yet” for C-3 Date, ”Compensation form not received” for First Hearing Date and “Accident/illness report not submitted” for C-2 Date.
The remaining categorical features were treated by imputing the missing values with the mode of the corresponding features.
Regarding the numerical variables, except “Age at Injury” and “Birth Year” we treated them by imputing the mean or the median depending on the relationship between them:

- If the mean and median were close, missing values were filled with the mean.
- Otherwise, the median was used for imputation, in order to not distort even more the distribution of the data distribution.

For “Age at Injury” and “Birth Year” we created a function that if one of the variables was zero or NaN and the other was a non-null number it would calculate the value of the feature with zero or NaN value by making the difference between accident year and the variable with the non-null value.
If we had the situations where either both had NaN values or one with zero and the other with NaN it would impute “Age at Injury” with the median of “X_train” and then do the previous calculation.
When we treated missing values at the same time for the three sets regarding categorical features, in the dates variables we applied the same contextual logic and for the others we used the mode of the features in the training data, for the numeric features we used the mean or the median of the corresponding feature in the training data, all of this to not incur in data leakage.

- **Outliers:**

To perform outlier treatment, we first set adjustments for the variables that were misclassified as numerical while they were considered categorical by us. We identified the outliers using the IQR approach, for each feature we calculated the interquartile range and defined the upper and lower limits to be able to identify the observations that were classified as outliers. We identified outliers in all numeric features except "Number of Dependents".

The outliers of "Birth Year" and "Age at Injury" were removed from the train set only, since these values were implausible as we mentioned during the data exploration. (***Figure 11***, ***Figure12***)

We kept the validation, and test sets intact to prevent data leakage, given we could not remove outliers from the validation and test sets.

We applied the validation set transformations from the train set, without touching the test set, to ensure that the test truly reflects unseen data during the model evaluation. For extreme values in "Average Weekly Wage", we replaced them with the median, as such occurrences were rare but still possible, unlike the previous features.

This meticulous preprocessing process ensured a clean and consistent dataset for modeling while maintaining the integrity of the validation and test sets for unbiased evaluation.

3. Multiclass Classification

3.1 Feature engineering, scaling and encoding

In order to have the best possible model to predict the "Claim Injury Type", we have created five new variables:

Days Between Accident and Assembly Date	Number of days from the Accident Date until the Assembly Date.
Claim Complexity	Classification of the claim as being of "Low", "Medium" or "High" complexity based on the features "Attorney/Representative" and "Days Between Accident and Assembly Date".
Forms Filed Indicator	Represents whether C-2 and C-3 forms were filled. Binary variable: 0 if none was filled, 1 if only one was filled and 2 if both were filled.
Injury Severity	Classification of the severity of the injury as being "Low", "Medium" or "High" based on the variable "WCIO Nature of Injury Description".
Likelihood by Industry	Classification of the different industries as being "Very likely to have an injury", "Likely to have an Injury" or "Not likely to have an Injury" based on the number of claims per industry.

These variables were created with the aim of showing relationships that could exist between variables that were not captured by the already existing variables.

Each variable created had a purpose behind it. For "Days Between Accident and Assembly Date", we wanted to reflect the processing times associated with each claim. With this feature the model can learn patterns such as injuries of certain types that are normally associated with certain carriers and target labels, can be associated with higher values of these features.

For “Claim Complexity”, we believe that different claims might be associated with different complexity levels. The standard level is “Low” and then according to the process time (“Days Between Accident and Assembly Date”) and the presence of an attorney in the process (“Attorney/Representative”) the complexity would change, for example, if there is an attorney and the process time is more than 30 days it is “High”.

“Forms Filed Indicator”, is a binary variable indicating whether both, none or only one of the forms was received (C-2 and C-3 forms).

Regarding “Injury Severity”, this variable was created to better translate the relationship and patterns between the variable “WCIO Nature of Injury Description” and the target variable. We created a severity map where we manually divided the different natures of injuries into “High”, “Medium” or “Low” severities. This was performed with some common knowledge, and a bit of research to try to understand which ones belong to each category. In this map we included all unique labels of training, validation and test datasets.

For “Likelihood by Industry”, we assessed the likelihood of injuries per industry given the nature of the work itself, for example, heavy machinery work vs. office jobs. Claims per Industry were counted and then we labeled the top 5 as “Very likely to have an Injury”, bottom 5 as “Not likely to have an Injury” and the rest as “Likely to have an Injury”. We segmented it like this, because the top 5 industries had much higher values compared to the remaining and the bottom 5 followed the same rationale but reversed.

Before feature selection, we decided to scale our numeric features and encode our categorical ones. We scaled our numeric features using MinMaxScaler, because it preserves the relative distances between data points, it works well with not normally distributed data unlike StandardScaler and because outlier treatment was already done. Scaling ensures all features have the same ranges at least in the training data (validation set can happen to have values above 1 or below 0), avoiding bias in the models due to the different ranges of features. Encoding of features was performed using category_encoders library to frequency encode our features. We believe that frequency encoding is a better solution, because it is ideal for high cardinality features, still reflects categories’ relative importance through their frequency, it avoids the dataset expansion done by One-Hot encoding and does not assume natural ordinal relationships between.

3.2 Feature Selection

In feature selection we decided to keep things simple, nonetheless we used various methods to be able to select the features that we are going to build and test our model with. In this phase we divided the methods according to their type: filter, wrapper and embedded methods. Starting with filter methods we utilized univariate analysis and spearman correlation matrix, as seen on **figure 13,** for numerical features and chi-square testing for categorical features. Univariate analysis lead to no variables being excluded and spearman correlation matrix allowed us to check if there were any redundant features in our dataset and, as expected, among the numeric features “Age at Injury” and “Birth Year” had a correlation coefficient of -0.99, meaning they are almost perfectly correlated and are most likely introducing the same information to our model being the presence of both in our model a bit redundant (0.80 normally is the threshold defined to categorize variables as being highly correlated).

Moving on to wrapper methods we decided to use RFE (Recursive Feature Elimination) with two different ranking models, Logistic Regression and Ridge Classifier, to try to obtain two different outcomes of which features to keep, with this we wanted to see the evolution from one model to the other given the L2 penalization introduced by the Ridge Classifier. In these methods we used all features, numeric and categorical, for the model to say which set of features from all the features in our dataset would be the best one according to the F1 score on the validation set. Using Logistic regression, we obtained that the best set of features is 27 out of 33, meaning that 6 features should be eliminated from the model (**Figure 14**). We decided to use the Ridge Classifier, because this is a model that inserts an L2 penalization into the model. Basically, this penalization is penalizing bigger coefficients by pushing them towards zero to prevent overfitting. We observed a difference from one to the other, with only 3 features being eliminated with Ridge Classifier (**Figure 15**). In the embedded methods we only used Lasso and we chose this model, because this model introduces an L1 penalization, which is similar to L2 but this one is able to eliminate coefficients by setting them equal to zero. Given this, we are going to use Lasso to observe which features he will eliminate. This model eliminated 10 out of 33 features, (**Figure 16**). To reach our final conclusions for the set of features we evaluated all features across three models (Logistic Regression, Ridge Classifier, and Lasso) and, depending on the variable type, the spearman correlation matrix or chi-square, discarding features if two or more models suggested removal. This process eliminated 6 features, leaving a final set of 27 features. (**Figure 17**, **Figure 18)**

3.3. Initial Model Training

To efficiently evaluate multiple models, this initial training phase was conducted on a sample of 10.000 records from the training dataset. We went for this sampling approach to significantly reduce computational costs while being able to assess the relative performance of various algorithms.

The models evaluated were:

- Random Forest.
- Gradient Boosting Classifier.
- Logistic Regression.
- Support Vector Classifier.
- K Nearest Neighbors.
- Decision Tree Classifier.
- XGBoost Classifier.

For XGBoost to work we had to create dummy samples in y_train_sampled, because there was one label in validation that was not seen in the training set. After this, we measured the performance of each with the weighted and macro F1-score, as the dataset exhibited class imbalance. The results from Random Forest Classifier, Gradient Boosting Classifier and XGBoost Classifier showed to be the best performances, as they achieved weighted F1-scores of approximately 0.712, 0.717 and 0.719 and macro F1-scores of 0.398, 0.412 and 0.418, respectively (**Figure 19**). The three models were subsequently chosen for hyperparameter optimization.

3.4. Hyperparameters Optimization

A custom Grid Search function was developed to test multiple hyperparameter combinations for the previously selected models. This function trained and validated Random Forest Classifier, Gradient Boosting Classifier and XGBoost Classifier on separate data to prevent overfitting and ensure reliable performance evaluation. For each model we identified the following best hyperparameters (**Figure 20**):

- XGBoost Classifier: gamma: 0.2, learning_rate: 0.1, max_depth: 5, n_estimators: 150, for a macro F1-score of 0.437.

- Random Forest Classifier: max_depth: 20, min_samples_split: 2, n_estimators: 200, for a macro F1-score of 0.40.
- Gradient Boosting Classifier: learning_rate: 0.1, max_depth: 5, n_estimators: 150, for a macro F1-score of 0.419.

After the hyperparameter tuning, we concluded that the XGBoost Classifier was the best model for this classification, based on the performance of the model assessed with macro F1-score.

3.5. Final model training and Evaluation

To finalize the building of our model, once we had selected our best model, we trained the XGBoost Classifier on the entire training data set using the selected best hyperparameters, which allowed the model to train and learn on all the available data and give the optimal performance.

After training we made the prediction in our reserved test dataset reaching our final goal, which was to build a model that would predict “Claim Injury Type”. We ended up with a score of 0.34865 in Kaggle.

4. Open-Ended section

4.1 Analytics Interface

Since WCB is responsible for assembling and deciding on claims whenever it becomes aware of a workplace injury and manually reviewing all claims is an arduous and time-consuming process, we not only created the model to automate this process of deciding on the “Claim Injury Type”, but we also created an interface to enhance model usability, allowing non-technical users in WCB to use it, since it creates an user-friendly platform making it easy for users to write the different inputs and retrieve the decision about the “Claim Injury Type”. Our idea was to create an analytics interface where people could just introduce the inputs corresponding to the features, we selected in the feature selection section and then it would return the “Claim Injury Type” as an output. This will be possible, because this interface already has built in the model, we choose in the model assessment section, for the interface to be able to give the prediction for the target variable as an output.

Our platform design is very similar to a google forms document (**Figure 21**) and that is what makes it so accessible to anyone, basically the person just needs to write all the inputs asked in the platform and then click on the submit button in the end. Every time this button is activated/clicked there is a function in the code that will check if certain conditions are met (e.g date variables are in the yyyy-mm-dd format) and raise an error message in the screen if they are not or raise a success box saying “Inputs submitted with success” if all conditions are met, giving right after the prediction of “Claim Injury Type”. Besides this, in the interface we also created a function that will scale and encode the inputs using the previous MinMaxScaler and frequency encoding, before inserting the inputs in the model.

This platform was created using the TKInter library, which allowed us to create a customizable interface where we were able to insert dropdown menus for some features, with the unique labels of features that were present in the training, validation and test sets already available for the user to select from or even write a new label if it is not available in the dropdown menu, we also created a slider to select the “Age at Injury” and “Days Between Accident and Assembly Date”, Radiobuttons for features that had limited options, such as, “Yes”, “No” and “Unknown” and a box that allows users to write, for example, the “Carrier Name” including the already existing labels in the datasets or new ones.

The interface was tested by filling in all fields with one of the observations of the training data and then the outcome was a prediction for “Claim Injury Type” according to the inputs. By comparing it with the tables where we have our observations scaled and encoded, we could see that the scaling and encoding was being done correctly, since the values of the variables were the same. We noticed that the prediction given by the model is different from the value in “Claim Injury Type” in the training data, which despite not correctly predicting the target according to “y_train”, can be seen as a good thing, since it means that our model didn’t overfit to our training data.

5. Conclusion

Initial objectives

The main goal of this project was to use machine learning tools to automate the classification of injury types from workers’ compensation claims, creating a model for this complex and time-consuming process. Additionally, we developed an analytics interface to make the model accessible to everyone including non-technical users, so that anyone could take advantage of this model to automate this process.

We began by exploring the dataset, identifying key characteristics of the data, such as the distribution of features, the presence of missing values and outliers. This analysis revealed that the dataset had 33 features (12 numerical and 21 categorical). One of the key insights was the high frequency of certain claims during the COVID-19 and inconsistencies in the “Age at Injury” variable. In the cleaning phase, we treated the missing values and outliers, we made sure we didn’t have any data leakage by applying transformations calculated from the training set to validation and test sets. Features with excessive missing values or irrelevance were dropped, regarding outliers we removed the extreme values in the training data for “Age at Injury” and “Birth Year” given the inconsistency of these numbers (age of 0 or 117 years given that these are claims related to workspace). In feature engineering we created five new features in all sets to try to understand and translate into features some complex relationships between variables like processing times, injury severity and industry-specific risk patterns.

A combination of filter, wrapper and embedded methods was used to evaluate the importance of features. This process led to the elimination of 6 features, which reduced the redundancy a little and improved the computation efficiency for the next step. The final set included 27 features (**Figure 17**, Figure 18), which were the best (most relevant) based on their performance across multiple selection methods.

Then, we proceeded to evaluate 7 models using a subset of 10000 records from the training dataset. The performances were measured with the weighted and macro F1-Score to account for class imbalance and XGBoost Classifier together with RandomForestClassifier and Gradient Boosting Classifier were the top performers with a macro F1-Score of 0.418, 0.412 and 0.398 respectively (**Figure 19**). We then proceeded to use a custom Grid Search function to fine-tune hyperparameters for the three best-performing models. XGBoost CLassifier was the chosen model given its performance on Grid Search with gamma: 0.2, learning_rate: 0.1, max_depth: 5, n_estimators: 150 (**Figure 20**) as its best hyperparameters, this model was then trained on the full training dataset with the 27 features selected in feature selection. Finally, the model’s predictions were formatted for submission to Kaggle, where it achieved a score of 0.34865.

In our open-ended section we ended up only developing a user-friendly interface using the TKInter library. This interface allows users to input claim details and receive predictions on the “Claim Injury Type”, automatizing the process of reviewing claims manually. (**Figure 21**)

Regarding the ideas we had initially for the open-ended section we reached the conclusion that we were being too ambitious by doing all three. One of the ideas was to try to create a model to predict the features "Agreement Reached" and "WCB Decision" and check whether using them as features would increase the model's performance, however we ended up eliminating these two features and so we believe that now it wouldn’t make much sense to try to predict these two features. Regarding the analysis and discussion of the importance of the features for the different values of the target variable we decided not to carry it out, because we decided to fully direct our efforts towards the interface, given that it was requiring much work to be able to deliver a good analytics interface, that would be working perfectly and doing what it was supposed. However, in the multivariate analysis we provided a small and superficial analysis of this topic of feature importance in the different target values. We as a group decided that it would be better to deliver a good open-ended section than a section with all the ideas we proposed to do and have it imperfectly done.

Results vs Expectations

The results were not perfectly aligned with our initial expectations, because we were hoping to have a model with a good score in Kaggle, which for us reflects our work and effort in this project, also meaning that we were able to build a good model to predict the “Claim Injury Type”. We improved a lot our project from the first delivery to this one in some steps, such as, splitting the data before preprocessing, improving methods of imputing missing values taking into account the context of the features like the dates variables (e.g Accident Date), treating the training, validation and test sets, using correct models in feature selection and making use of functions like Grid Search so we could optimize hyperparameters in our final model, and our Kaggle score was always around 0.28, and then in the day before submitting we tried a new model and it gave us a score of 0.35. During the whole project we felt unmotivated and, in the end, we were already giving up, since we believed that our project was very good, but we couldn’t seem to find a model that would give us a good score.

Limitations

Despite the success, there were some limitations in the project, such as the size of our dataset, which made our process time in assessing our models and adjusting our choices low, since we would have to spend around 5 hours to run all the notebook, or at least one hour to be able to train the model in the whole training data. This affected us by losing time waiting out for things to finally give us an output so we could adjust them. While we removed redundant features, some choices might have slightly affected the model’s performance. We were unable to address two ideas of the Open-Ended Section, which could have extended the understandability of the features of the project and enhanced our model and corresponding performance.

Future work

For future work, we could try to explore new models (DL models like neural networks), work with a larger data sample to improve accuracy, maybe developing a system where the model can be updated continuously as new data becomes available.

Another approach we considered, time permitting, is implementing a binary classification to predict "grant" (1) or "not grant" (0). The target variable "grant_status" would be constructed based on conditions derived from the "Agreement Reached", "WCB Decision" and "Claim Injury Type" variables: if an agreement is reached without WCB involvement, it would indicate a "grant". A "Not Work Related" WCB decision would suggest "not grant”. The injury type severity indicated in "Claim Injury Type" would be used as a proxy to refine the classification (more severe injuries are more likely to lead to a "grant", while “cancelled” and “NON-COMP” would lead to “not grant”). Even though this might help the WCB to have less claims to treat and gain time, the model might still need supervised learning due to missing values.

Github link: https://github.com/apocalypsine/MLproject_G20

Annexes:

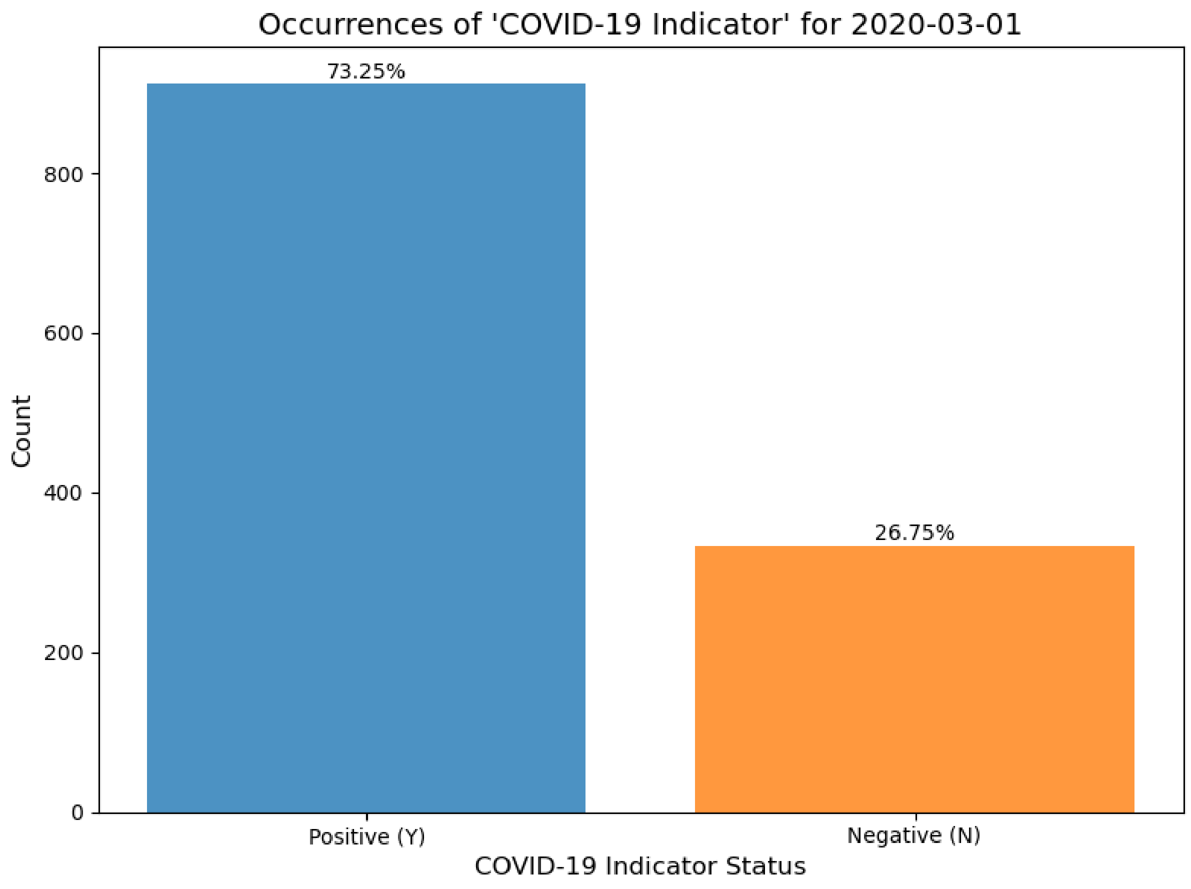


Figure 1: *Distribution of results of COVID-19 indicator on 2020-03-01*

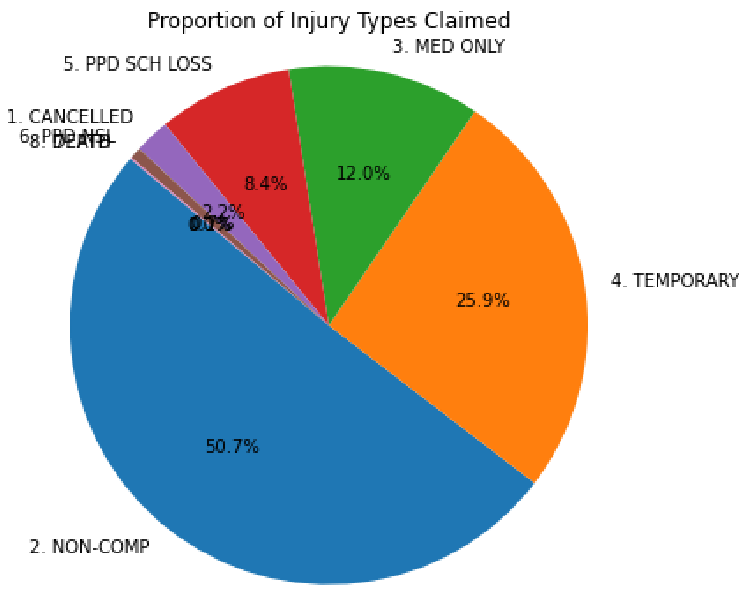


Figure 2: *Proportion of Injury Types Claims in pie chart*

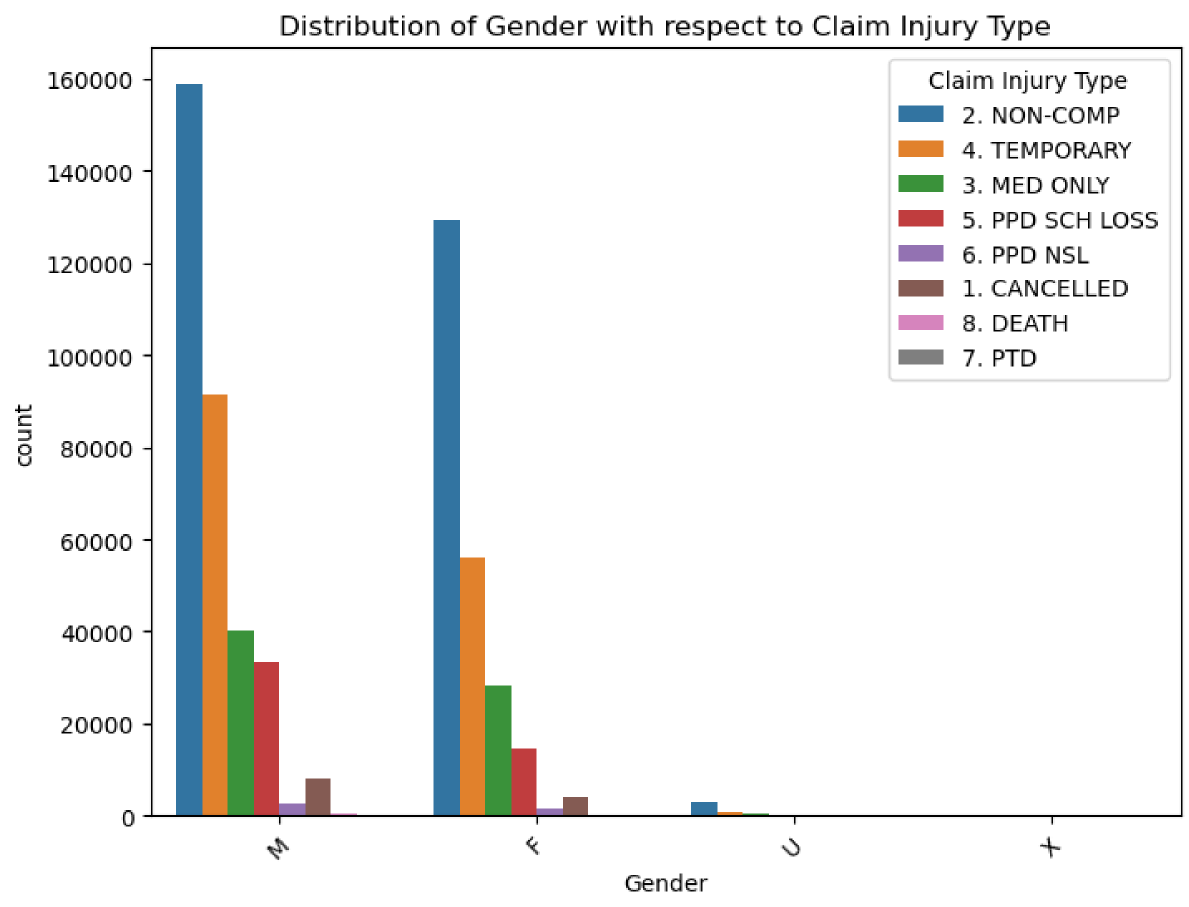


Figure 3: *Distribution of Gender by Claim Injury Types*

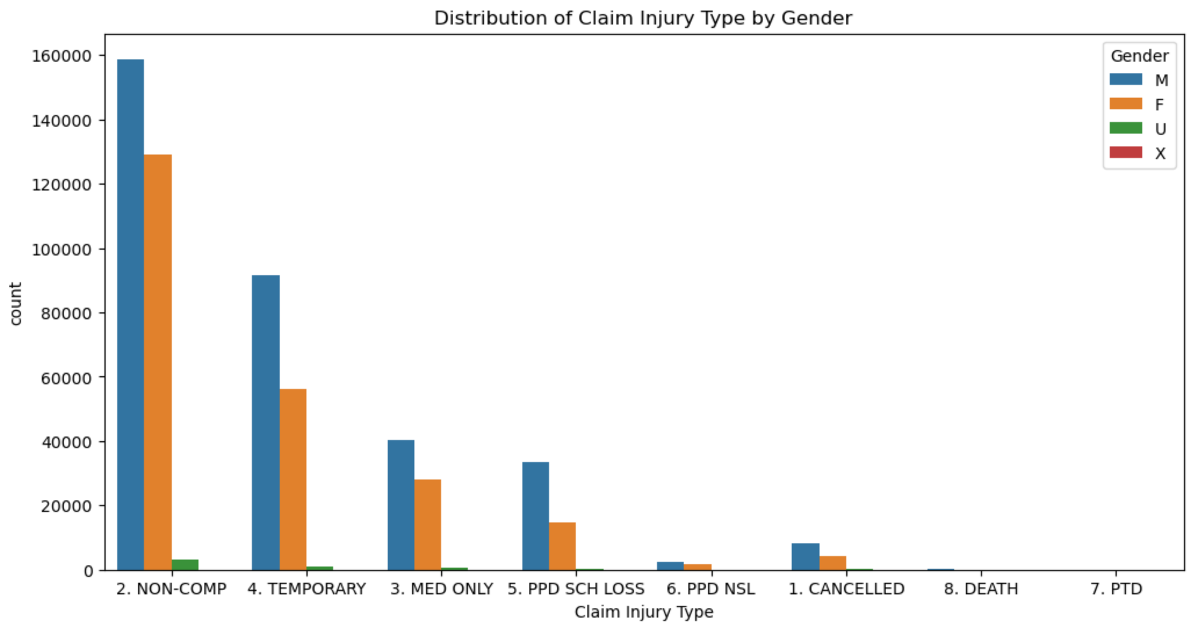


Figure 4: *Distribution of Claim Injury Type by Gender*

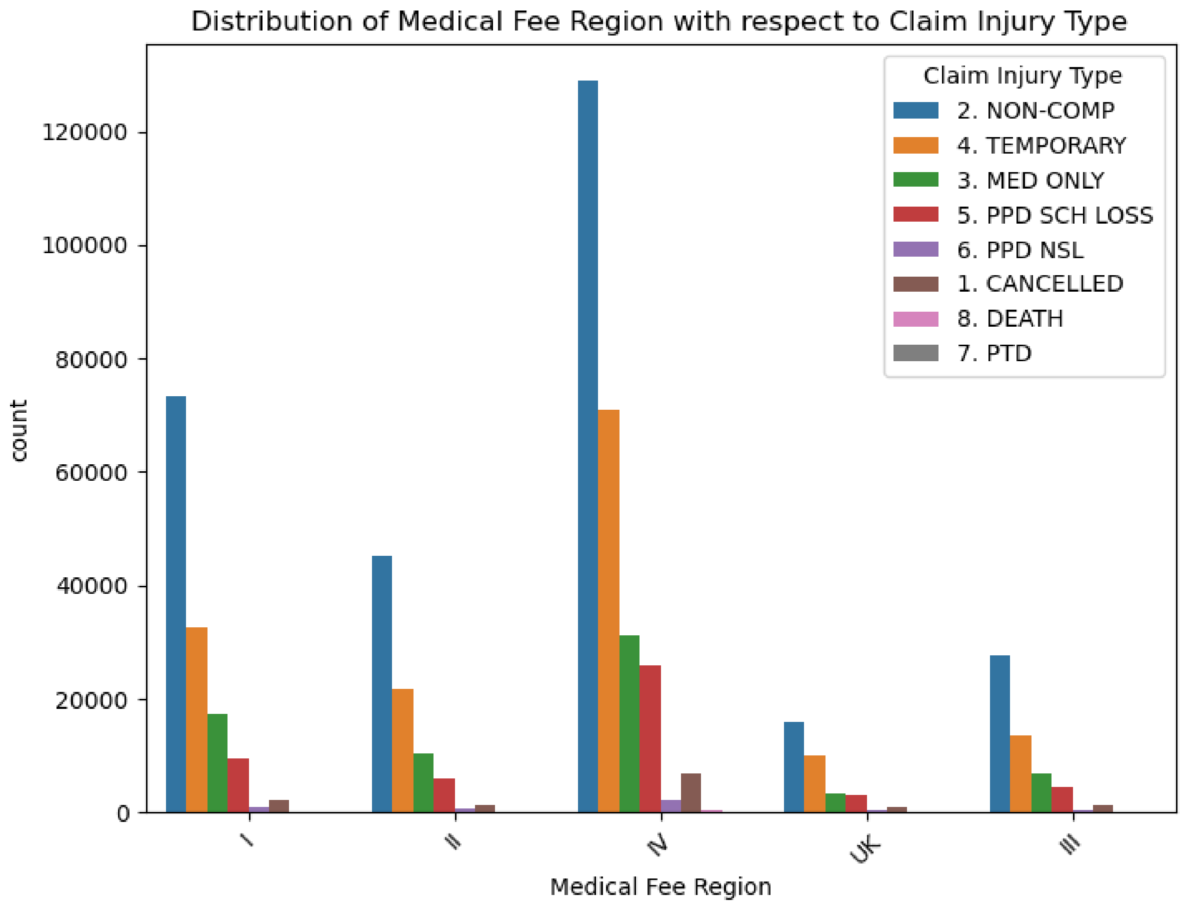


Figure 5: *Distribution of Medical fee region by Claim Injury type*

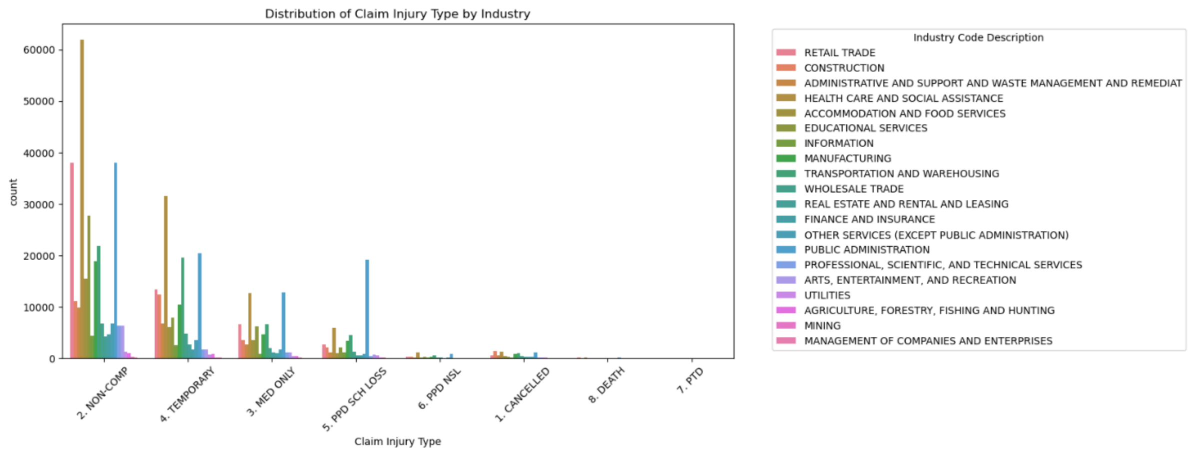


Figure 6: *Distribution of claim injury type by Industry*

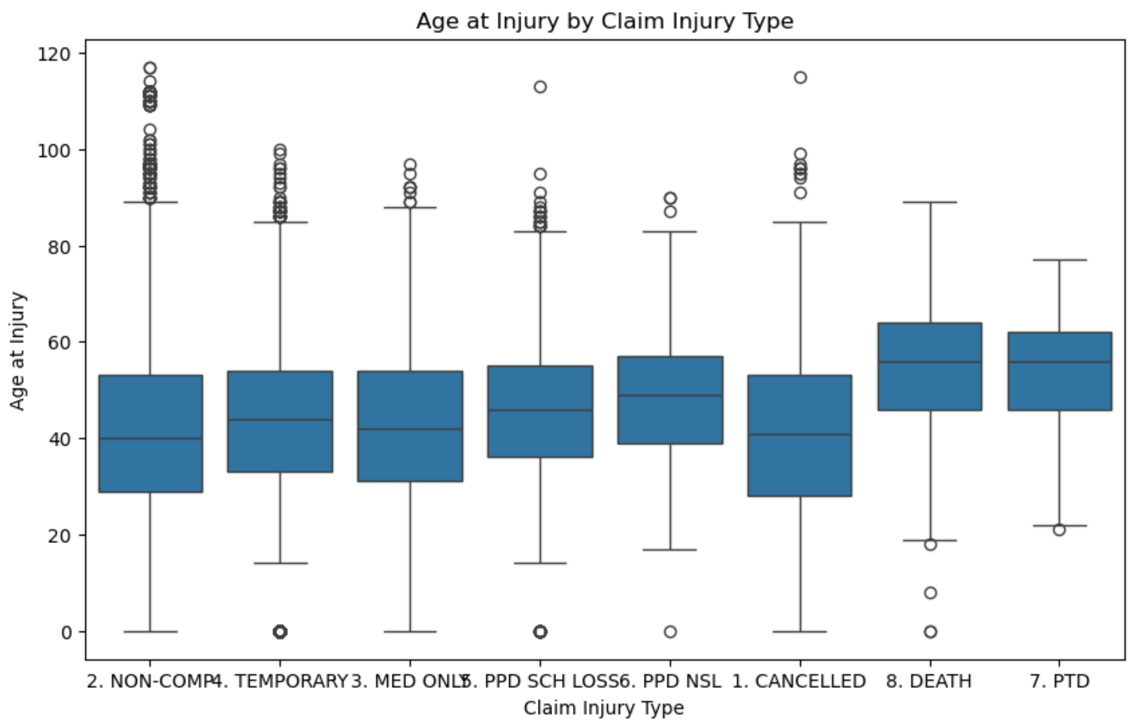


Figure 7: *Boxplot of Age at Injury by Claim Injury Type*

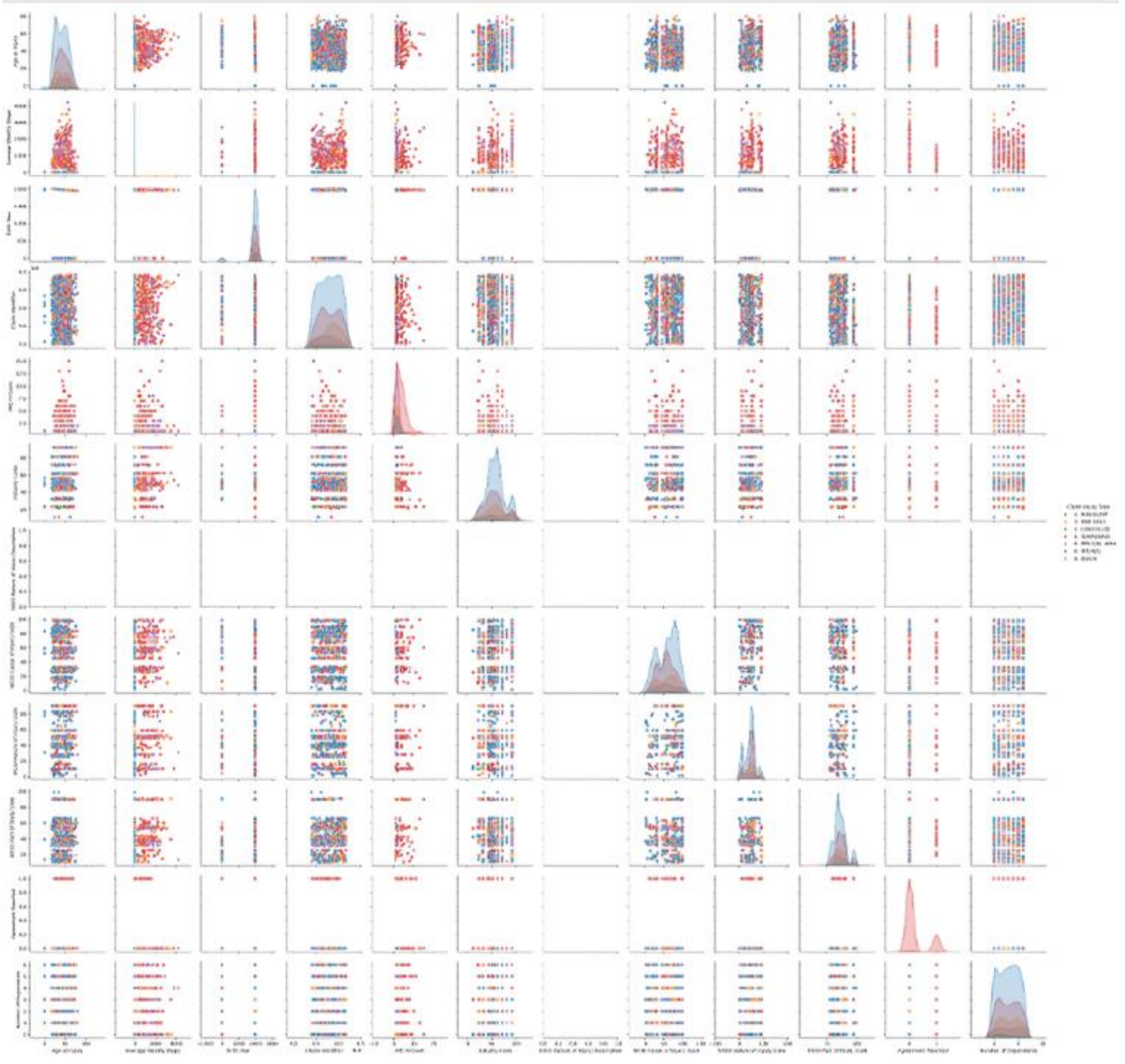
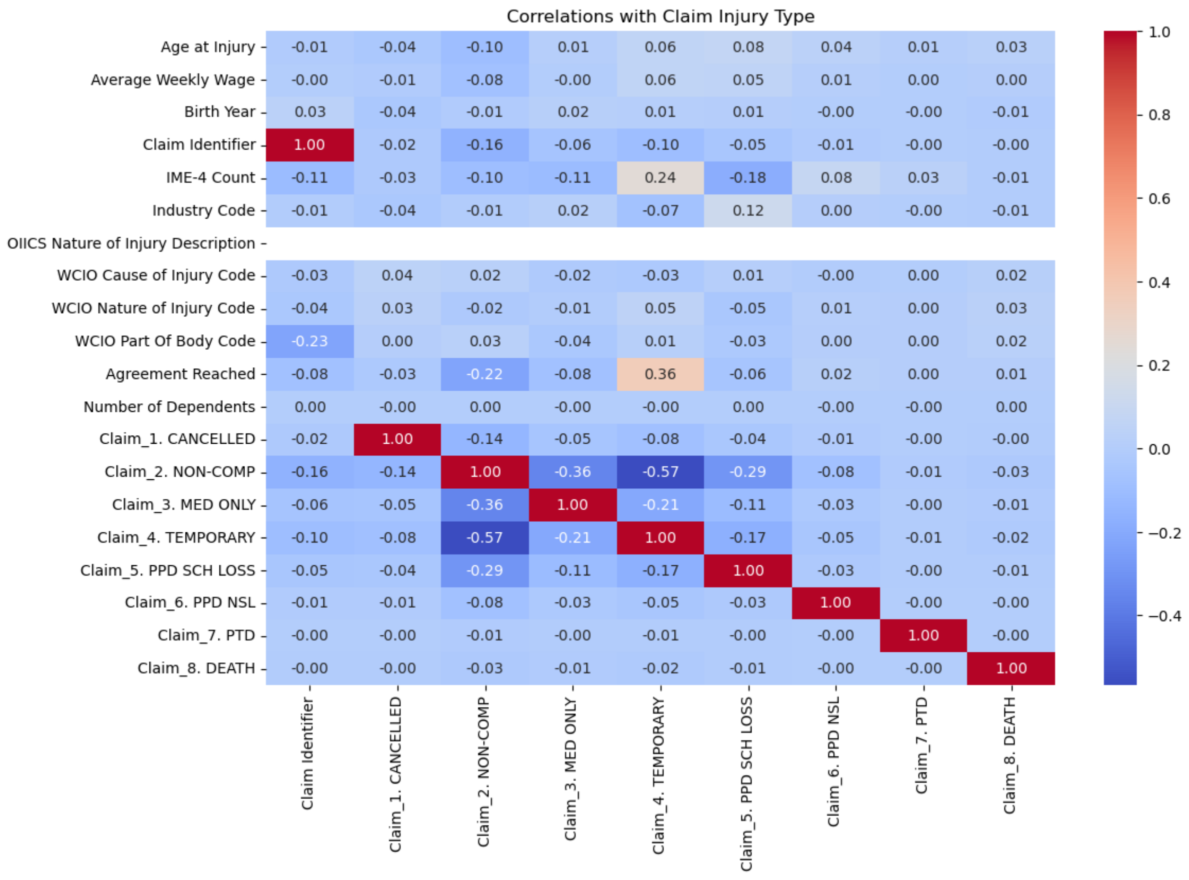
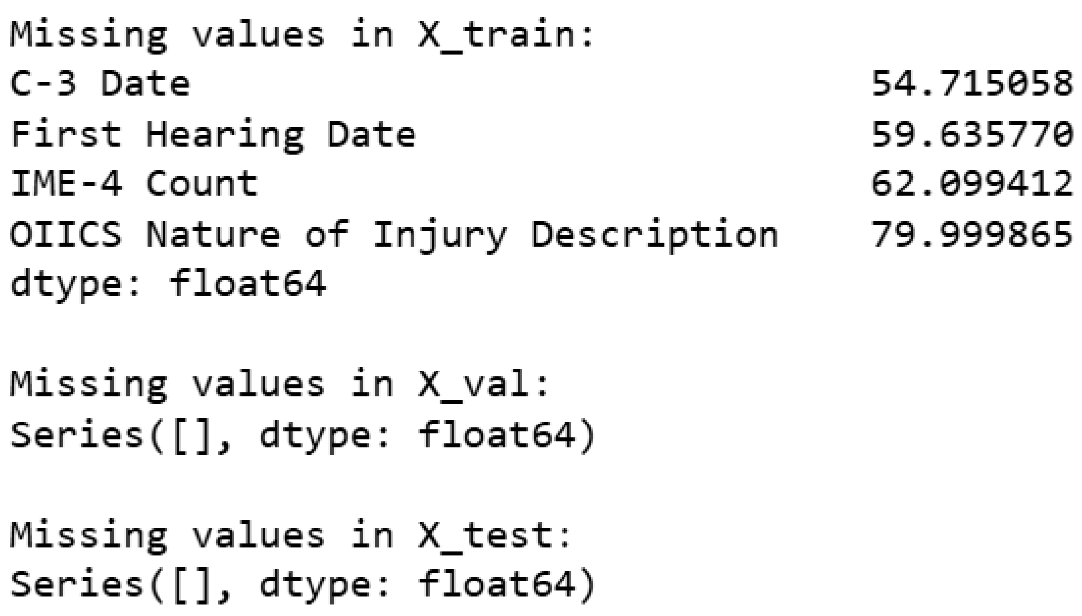


Figure 8: *Multivariate analysis matrix*



	Claim Identifier	Claim_1. CANCELLED	Claim_2. NON-COMP	Claim_3. MED ONLY	Claim_4. TEMPORARY	Claim_5. PPD SCH LOSS	Claim_6. PPD NSL	Claim_7. PTD	Claim_8. DEATH
Age at Injury	-0.01	-0.04	-0.10	0.01	0.06	0.08	0.04	0.01	0.03
Average Weekly Wage	-0.00	-0.01	-0.08	-0.00	0.06	0.05	0.01	0.00	0.00
Birth Year	0.03	-0.04	-0.01	0.02	0.01	0.01	-0.00	-0.00	-0.01
Claim Identifier	1.00	-0.02	-0.16	-0.06	-0.10	-0.05	-0.01	-0.00	-0.00
IME-4 Count	-0.11	-0.03	-0.10	-0.11	0.24	-0.18	0.08	0.03	-0.01
Industry Code	-0.01	-0.04	-0.01	0.02	-0.07	0.12	0.00	-0.00	-0.01
OIICS Nature of Injury Description									
WCIO Cause of Injury Code	-0.03	0.04	0.02	-0.02	-0.03	0.01	-0.00	0.00	0.02
WCIO Nature of Injury Code	-0.04	0.03	-0.02	-0.01	0.05	-0.05	0.01	0.00	0.03
WCIO Part Of Body Code	-0.23	0.00	0.03	-0.04	0.01	-0.03	0.00	0.00	0.02
Agreement Reached	-0.08	-0.03	-0.22	-0.08	0.36	-0.06	0.02	0.00	0.01
Number of Dependents	0.00	-0.00	0.00	-0.00	-0.00	0.00	-0.00	-0.00	0.00
Claim_1. CANCELLED	-0.02	1.00	-0.14	-0.05	-0.08	-0.04	-0.01	-0.00	-0.00
Claim_2. NON-COMP	-0.16	-0.14	1.00	-0.36	-0.57	-0.29	-0.08	-0.01	-0.03
Claim_3. MED ONLY	-0.06	-0.05	-0.36	1.00	-0.21	-0.11	-0.03	-0.00	-0.01
Claim_4. TEMPORARY	-0.10	-0.08	-0.57	-0.21	1.00	-0.17	-0.05	-0.01	-0.02
Claim_5. PPD SCH LOSS	-0.05	-0.04	-0.29	-0.11	-0.17	1.00	-0.03	-0.00	-0.01
Claim_6. PPD NSL	-0.01	-0.01	-0.08	-0.03	-0.05	-0.03	1.00	-0.00	-0.00
Claim_7. PTD	-0.00	-0.00	-0.01	-0.00	-0.01	-0.00	-0.00	1.00	-0.00
Claim_8. DEATH	-0.00	-0.00	-0.03	-0.01	-0.02	-0.01	-0.00	-0.00	1.00

Figure 9: *Heatmap of Features Correlation with Claim Injury Types*



```
Missing values in X_train:
C-3 Date                                   54.715058
First Hearing Date                         59.635770
IME-4 Count                                62.099412
OIICS Nature of Injury Description         79.999865
dtype: float64

Missing values in X_val:
Series([], dtype: float64)

Missing values in X_test:
Series([], dtype: float64)
```

Figure 10: *Output showing variables with high missing values percentages*

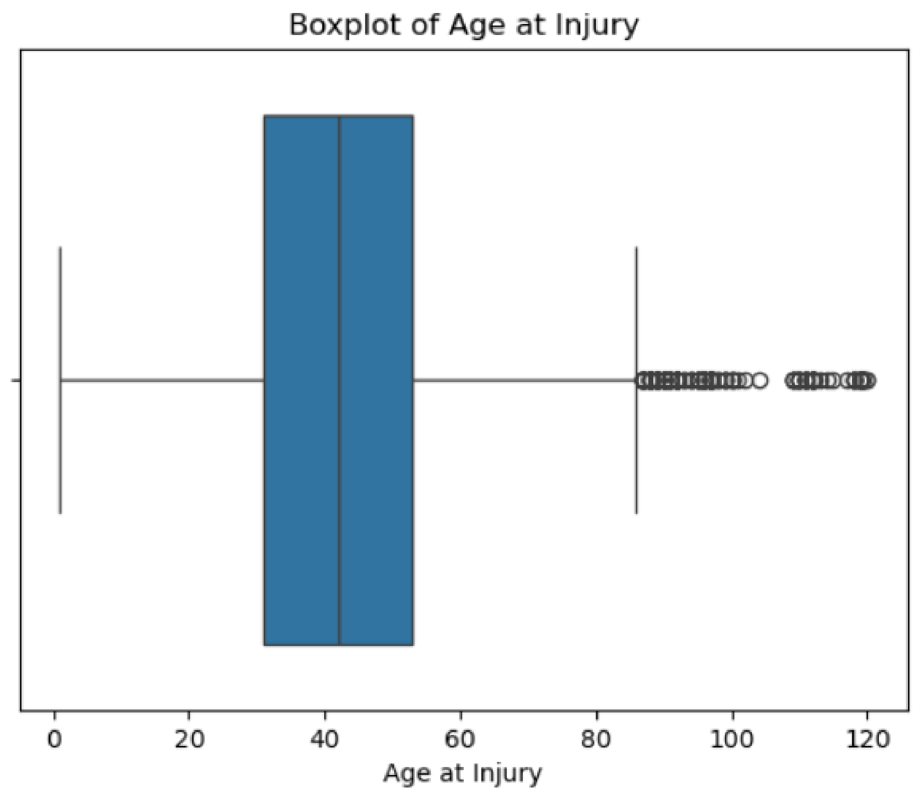


Figure 11: *Boxplot showing Age at Injury Distribution with outliers*

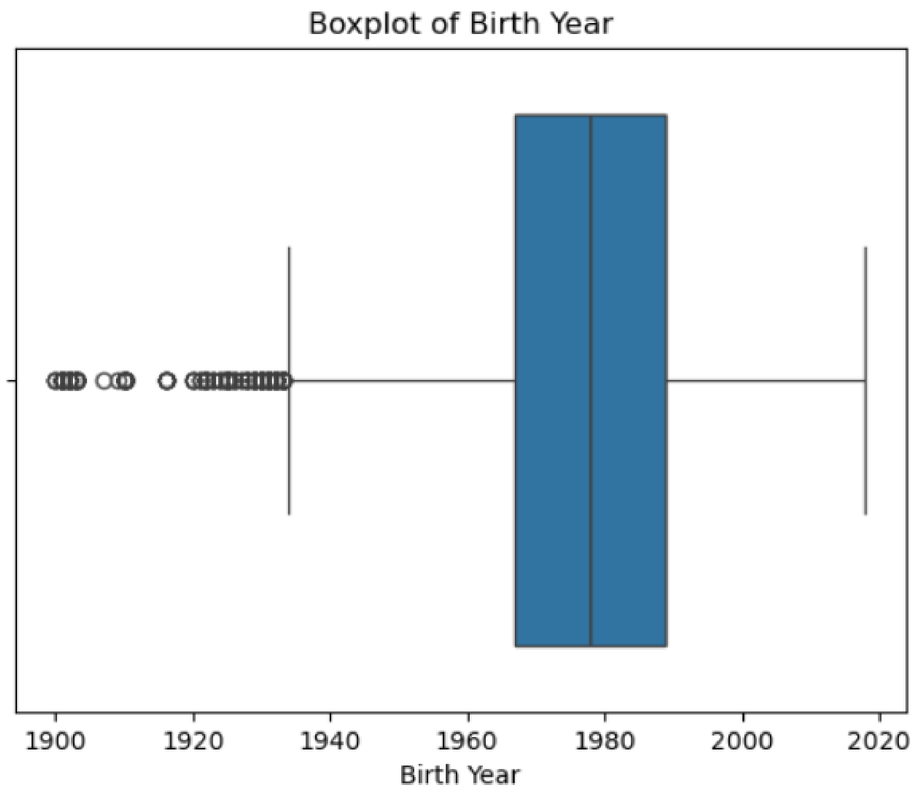


Figure 12: *Boxplot showing Birth Year distribution with outliers*

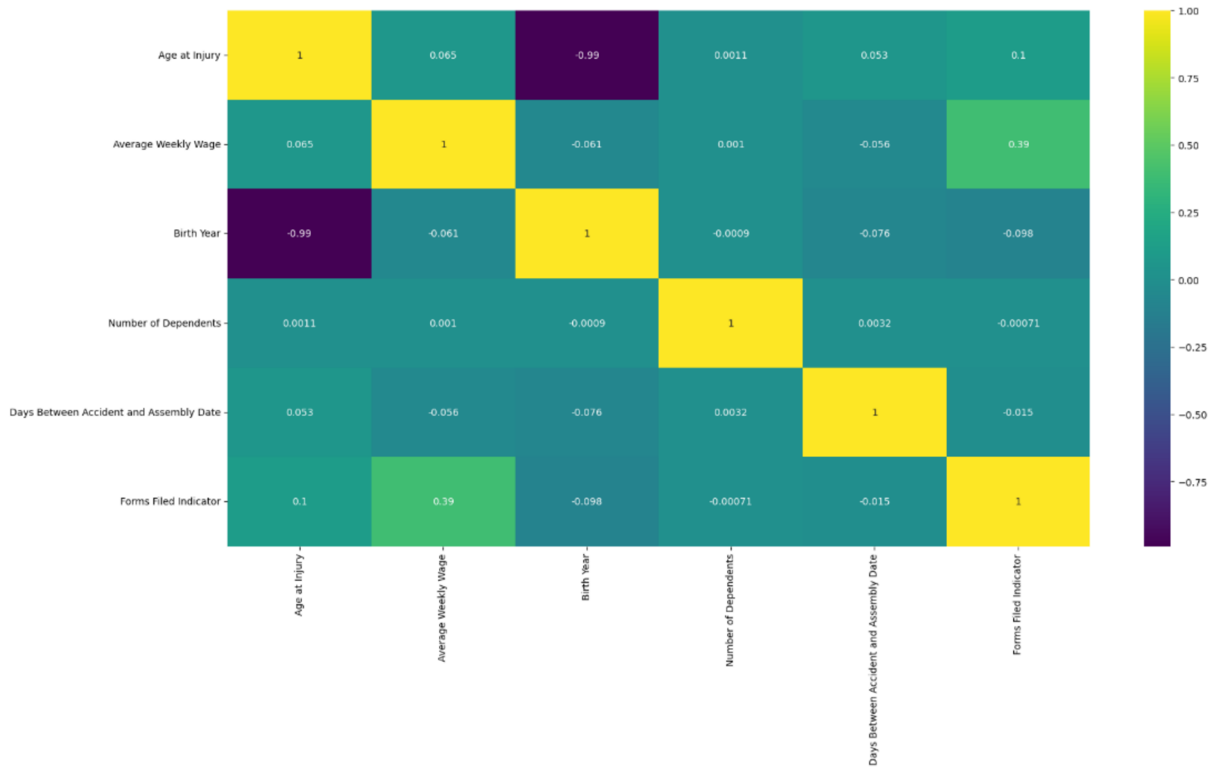


Figure 13: *Correlation matrix of the scaled numerical features*

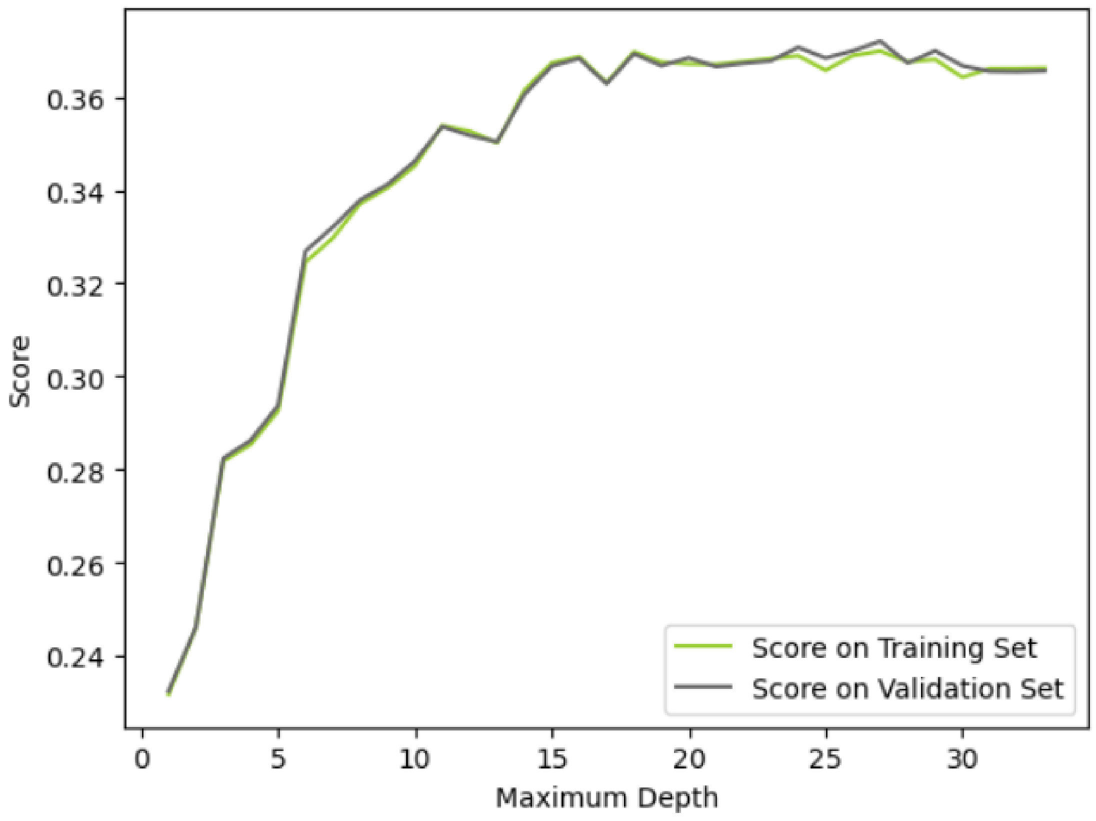


Figure 14: *RFE score using Logistic regression*

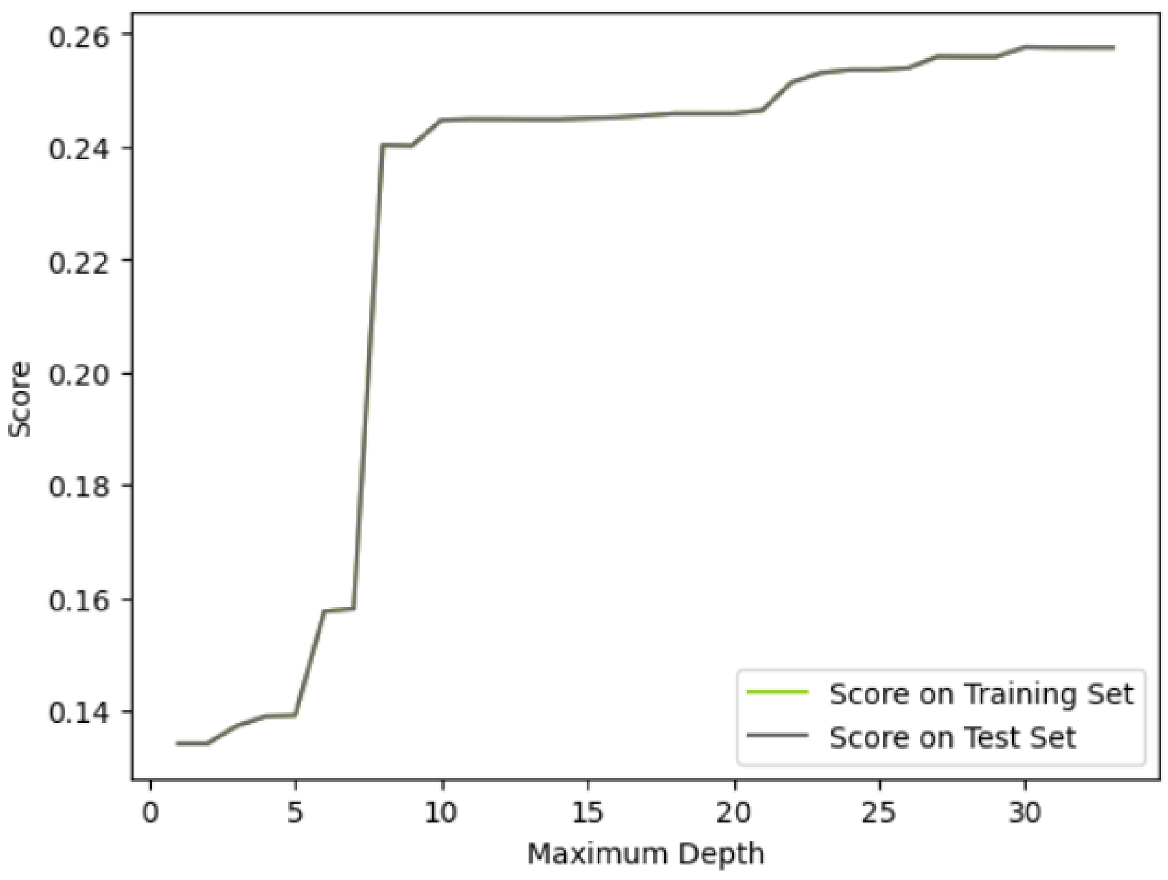


Figure 15: *RFE Score using Ridge Classifier*

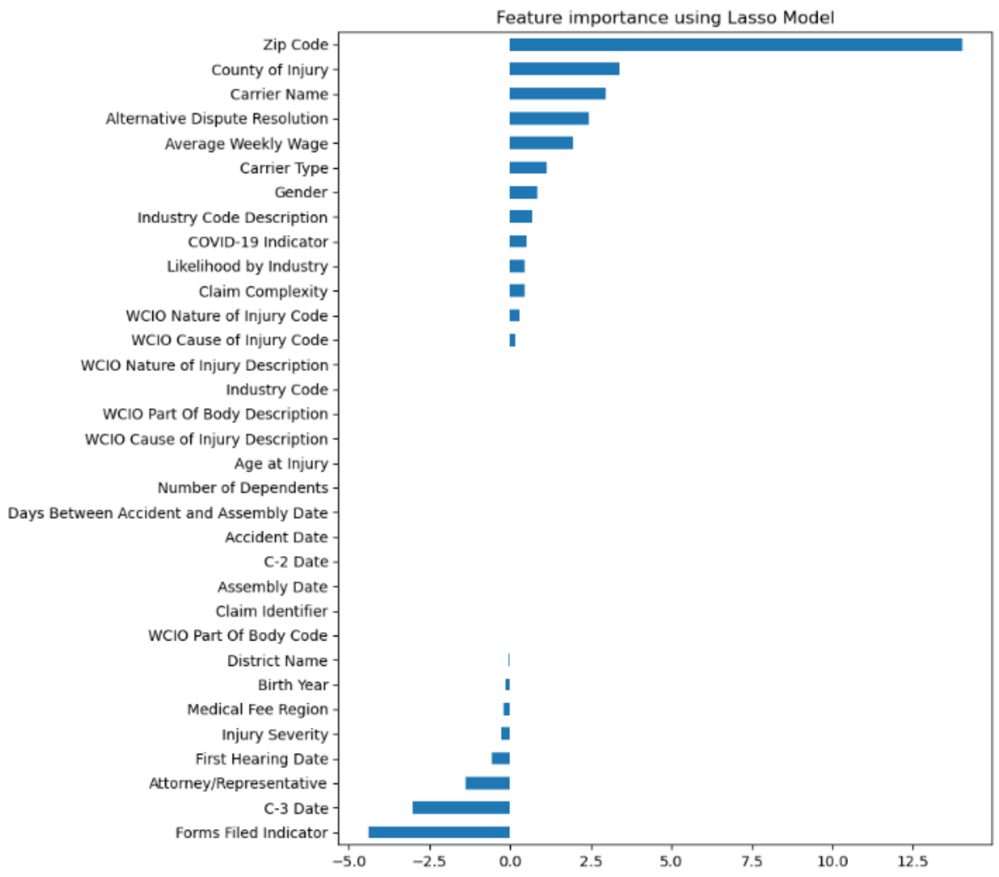
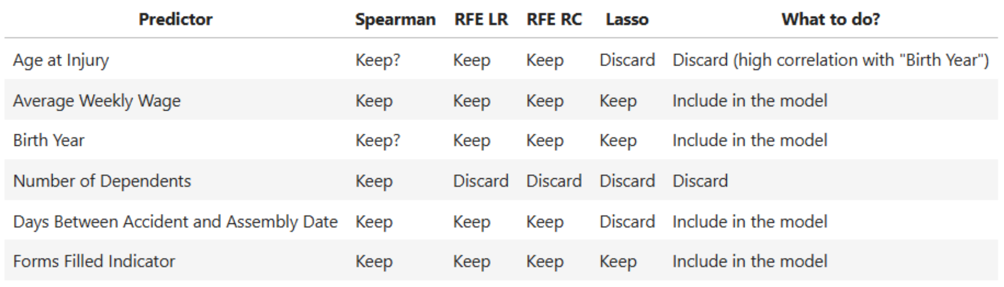


Figure 16: *feature importance using LASSO regression*

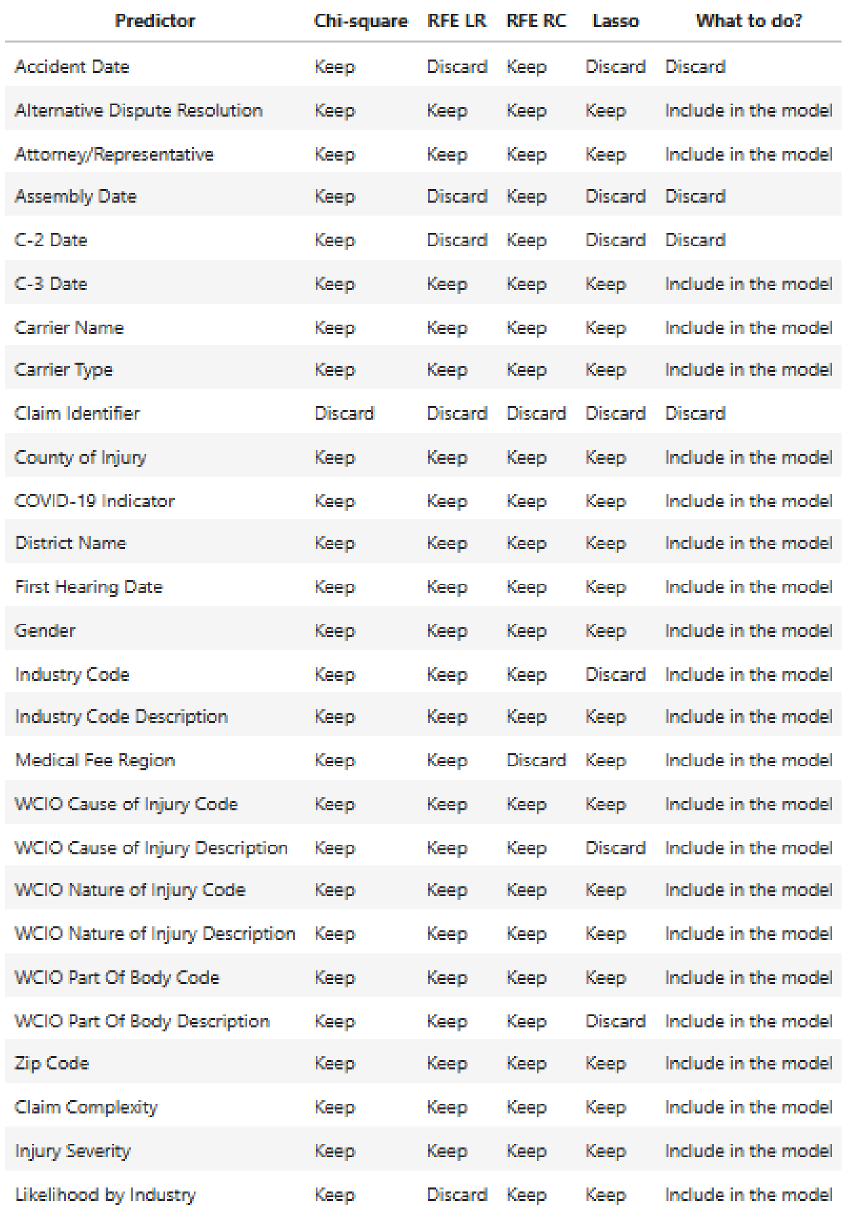
Numerical Data



Predictor	Spearman	RFE LR	RFE RC	Lasso	What to do?
Age at Injury	Keep?	Keep	Keep	Discard	Discard (high correlation with "Birth Year")
Average Weekly Wage	Keep	Keep	Keep	Keep	Include in the model
Birth Year	Keep?	Keep	Keep	Keep	Include in the model
Number of Dependents	Keep	Discard	Discard	Discard	Discard
Days Between Accident and Assembly Date	Keep	Keep	Keep	Discard	Include in the model
Forms Filled Indicator	Keep	Keep	Keep	Keep	Include in the model

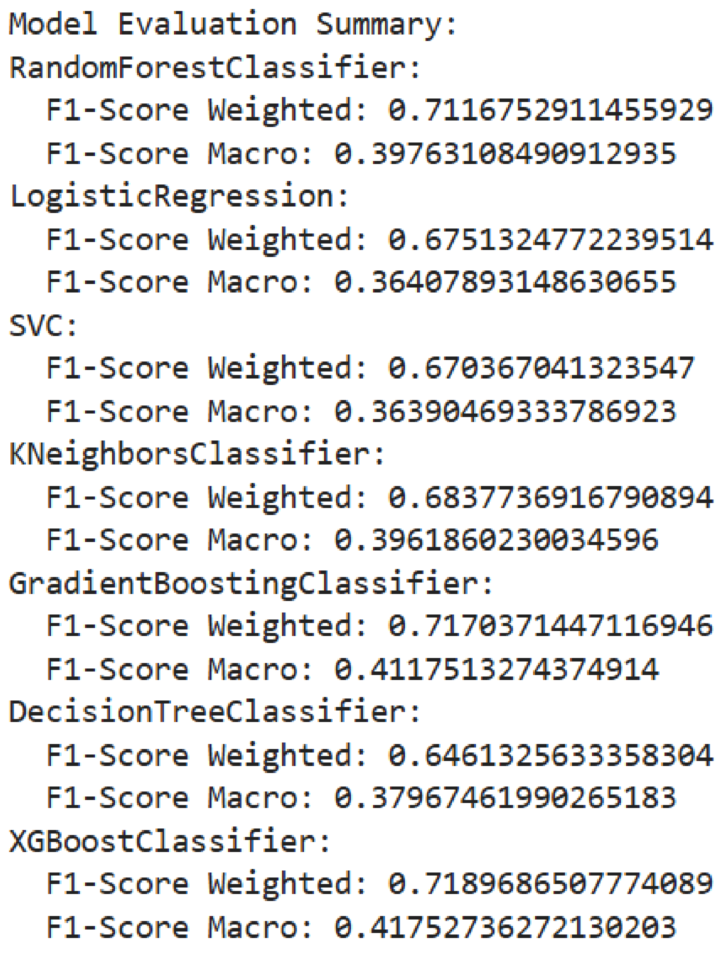
Figure 17: *The final selected features from numerical data*

Categorical Data



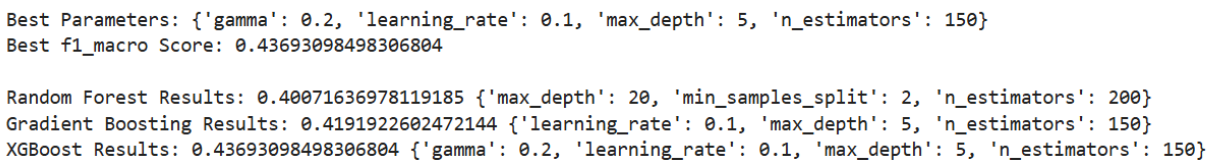
Predictor	Chi-square	RFE LR	RFE RC	Lasso	What to do?
Accident Date	Keep	Discard	Keep	Discard	Discard
Alternative Dispute Resolution	Keep	Keep	Keep	Keep	Include in the model
Attorney/Representative	Keep	Keep	Keep	Keep	Include in the model
Assembly Date	Keep	Discard	Keep	Discard	Discard
C-2 Date	Keep	Discard	Keep	Discard	Discard
C-3 Date	Keep	Keep	Keep	Keep	Include in the model
Carrier Name	Keep	Keep	Keep	Keep	Include in the model
Carrier Type	Keep	Keep	Keep	Keep	Include in the model
Claim Identifier	Discard	Discard	Discard	Discard	Discard
County of Injury	Keep	Keep	Keep	Keep	Include in the model
COVID-19 Indicator	Keep	Keep	Keep	Keep	Include in the model
District Name	Keep	Keep	Keep	Keep	Include in the model
First Hearing Date	Keep	Keep	Keep	Keep	Include in the model
Gender	Keep	Keep	Keep	Keep	Include in the model
Industry Code	Keep	Keep	Keep	Discard	Include in the model
Industry Code Description	Keep	Keep	Keep	Keep	Include in the model
Medical Fee Region	Keep	Keep	Discard	Keep	Include in the model
WCIO Cause of Injury Code	Keep	Keep	Keep	Keep	Include in the model
WCIO Cause of Injury Description	Keep	Keep	Keep	Discard	Include in the model
WCIO Nature of Injury Code	Keep	Keep	Keep	Keep	Include in the model
WCIO Nature of Injury Description	Keep	Keep	Keep	Keep	Include in the model
WCIO Part Of Body Code	Keep	Keep	Keep	Keep	Include in the model
WCIO Part Of Body Description	Keep	Keep	Keep	Discard	Include in the model
Zip Code	Keep	Keep	Keep	Keep	Include in the model
Claim Complexity	Keep	Keep	Keep	Keep	Include in the model
Injury Severity	Keep	Keep	Keep	Keep	Include in the model
Likelihood by Industry	Keep	Discard	Keep	Keep	Include in the model

Figure 18: *The final selected features from numerical data*



```
Model Evaluation Summary:
RandomForestClassifier:
  F1-Score Weighted: 0.7116752911455929
  F1-Score Macro: 0.39763108490912935
LogisticRegression:
  F1-Score Weighted: 0.6751324772239514
  F1-Score Macro: 0.36407893148630655
SVC:
  F1-Score Weighted: 0.670367041323547
  F1-Score Macro: 0.36390469333786923
KNeighborsClassifier:
  F1-Score Weighted: 0.6837736916790894
  F1-Score Macro: 0.3961860230034596
GradientBoostingClassifier:
  F1-Score Weighted: 0.7170371447116946
  F1-Score Macro: 0.4117513274374914
DecisionTreeClassifier:
  F1-Score Weighted: 0.6461325633358304
  F1-Score Macro: 0.37967461990265183
XGBoostClassifier:
  F1-Score Weighted: 0.7189686507774089
  F1-Score Macro: 0.41752736272130203
```

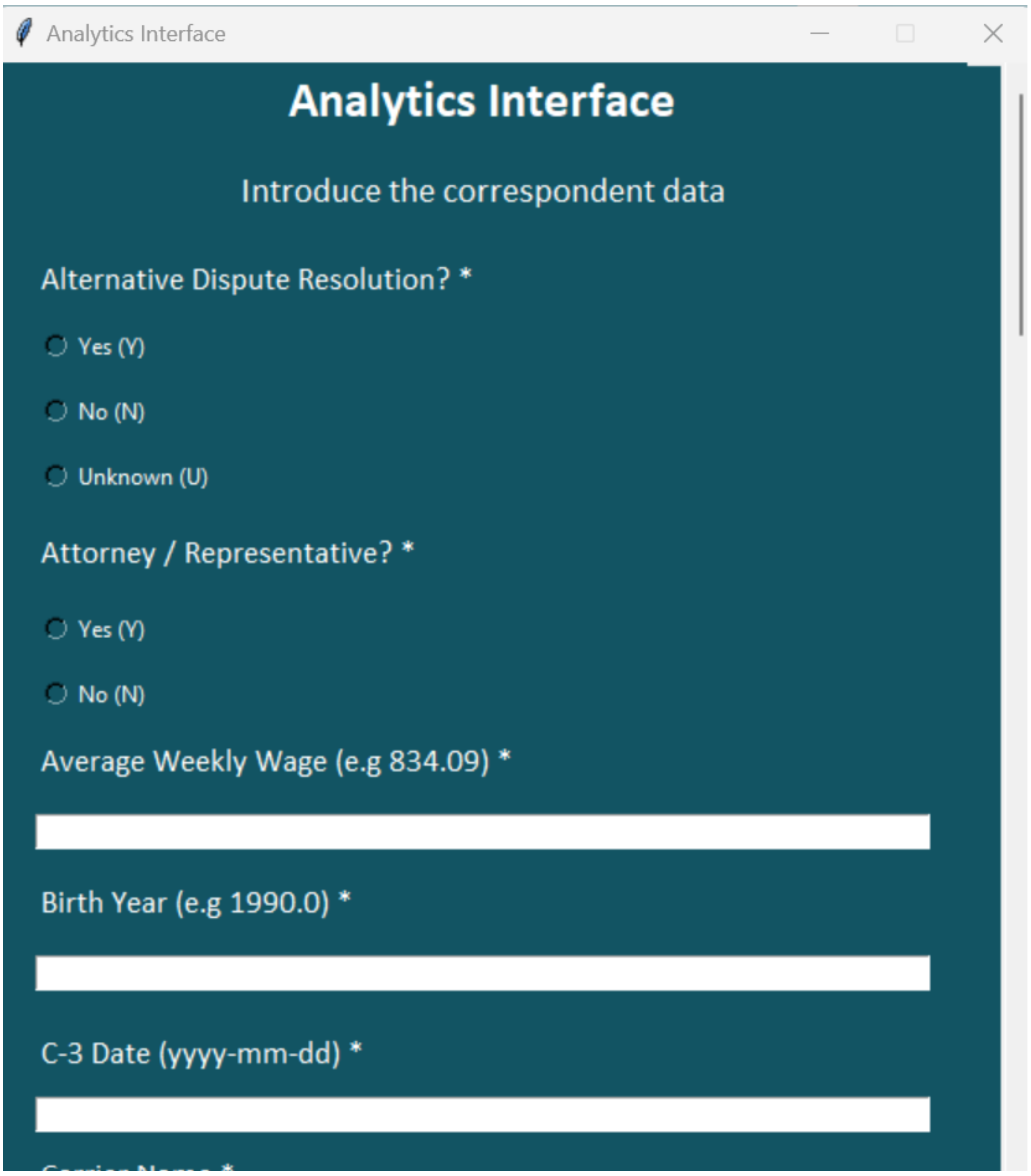
Figure 19: *Summary of the performance of evaluated models*



```
Best Parameters: {'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 150}
Best f1_macro Score: 0.43693098498306804

Random Forest Results: 0.40071636978119185 {'max_depth': 20, 'min_samples_split': 2, 'n_estimators': 200}
Gradient Boosting Results: 0.4191922602472144 {'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 150}
XGBoost Results: 0.43693098498306804 {'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 150}
```

Figure 20: *Best Hyperparameters of each best performer model*


Analytics Interface

Analytics Interface

Introduce the correspondent data

Alternative Dispute Resolution? *

- Yes (Y)
- No (N)
- Unknown (U)

Attorney / Representative? *

- Yes (Y)
- No (N)

Average Weekly Wage (e.g 834.09) *

Birth Year (e.g 1990.0) *

C-3 Date (yyyy-mm-dd) *

Figure 21: *Analytics interface*

References

[1] NxnXzH4HS. (n.d.). How AI is transforming the insurance industry [6 use cases]. *V7*.

https://www.v7labs.com/blog/ai-insurance

[2] Yohn, A. (2024, September 23). *11 Use cases for predictive analytics in insurance*. Duck Creek.

https://www.duckcreek.com/blog/predictive-analytics-reshaping-insurance-industry/