MDSAA

Master's in Data Science and Advanced Analytics

Predictive Modelling for Compensation Benefits

Machine Learning Project

December 2024

Group 51

André Lourenço - 20240743
Carolina Pinto - 20240494
Daniel Caridade - 20211588
Fábio dos Santos - 20240678
Gustavo Gomes - 20240657

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

Abstract

The New York Workers' Compensation Board (WCB) manages over 5 million claims since 2000, with claim review processes becoming increasingly complex and time-consuming. This report presents the development of a machine learning model to automate the classification of claims, improving efficiency in decision-making, whose workflow is presented in figure 1. We address the challenges posed by high cardinality, feature redundancies, and severe class imbalances through comprehensive data preprocessing. Key steps included imputation of missing values, removal of outliers, encoding categorical variables, and feature engineering to generate new predictive attributes. Feature selection was performed using both statistical and machine learning-based methods to retain the most relevant predictors for model building.

Seven algorithms were evaluated using Stratified 5-Fold cross-validation, with Extreme Gradient Boosting emerging as the best performer. After hyperparameter tuning, the model achieved a validation performance of 0.4487 with a low overfitting rate of less than 5%. This model was integrated into an interactive interface designed for WCB to make real-time predictions on new claims. The results demonstrate the effectiveness of machine learning in streamlining the workers' compensation decision-making process, providing WCB with a powerful tool to handle large-scale claim assessments efficiently. All deliverables and project information, including code and documentation, are published on GitHub[1] for further reference and collaboration.

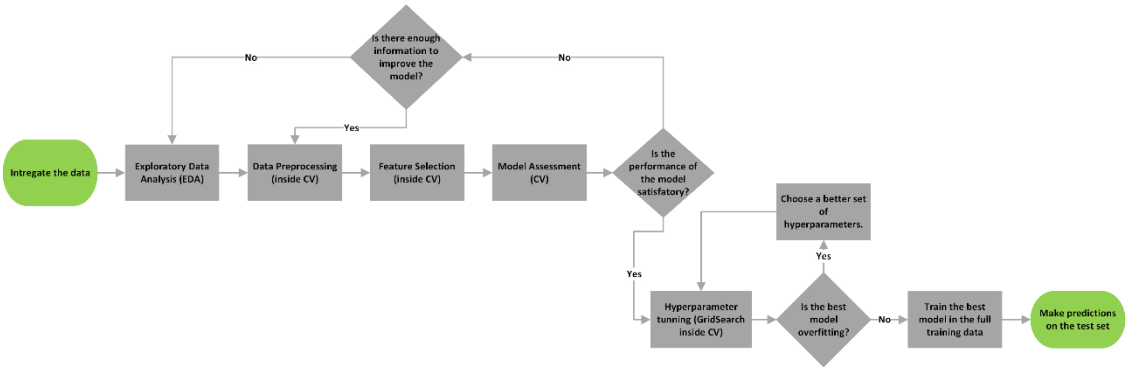


Figure 1: Machine Learning Model Development Workflow

[1] https://github.com/DanieLLL5/Machine-Learning-Project-Group-51.git

Table of Content

1. Introduction

In today's complex regulatory environment, accurately determining injury types for workers' compensation claims is critical for organizations like the Workers' Compensation Board (WCB) with fair and consistent decisions not only enhancing operational efficiency but also building trust with stakeholders while preventing prolonged disputes and financial losses. As claim volumes continue to grow, leveraging data-driven approaches has become essential to streamline decision-making and improve outcomes.

Previous studies have utilized machine learning models to predict the severity of claims; however, these efforts have predominantly concentrated on narrow domains such as agriculture [1] or textual injury surveillance [2]. This study aims to expand the scope of such research by developing a model to evaluate injury types among workers across various industries, utilizing cases classified by the New York Workers' Compensation Board (WCB) providing a broader and more comprehensive analysis of injury patterns.

This project aims to develop a machine learning-based model to assist WCB in predicting ***Claim Injury Type*** by analysing historical claim data, including demographic details, injury descriptions, and other relevant factors. Our methodology followed an iterative process with multiple stages, starting with exploratory data analysis (EDA) and data preprocessing. Several machine learning models, such as Logistic Regression, Decision Trees, Random Forest, and Extreme Gradient Boosting were tested, across key metrics: precision, recall, and macro F1-score.

Beyond prediction, the project delivers a user-friendly interface built with **Tkinter** and **Joblib** libraries, enabling WCB to make predictions easily and efficiently. These actionable insights empower WCB to improve service delivery, optimize resource allocation, and address challenges in managing workers' compensation claims.

2. Data Exploration and Preprocessing

2.1 EDA (Exploratory Data Analysis)

2.1.1 Data Description

The first step in this Machine Learning initiative was to get a brief overview of the provided dataset. The dataset contains 593471 observations and 33 features, being 5 numerical and 28 categorical. Notably, the variables that represent codes (e.g. *Industry Code*) are represented numerically, but represent categorical data that requires encoding.

2.1.2 Missing Values

Our analysis of missing data revealed that, apart from *Assembly Date* and *Claim Identifier*, all features contain missing values. Among these, 14 features have exactly 19,445 missing values, representing rows entirely devoid of data, except for the two mentioned features. The feature *OIICS Nature of Injury Description* consists exclusively of missing values and will be removed during pre-processing. Additionally, features such as *C-3 Date*, *First Hearing Date*, and *IME-4 Count* exhibit over 70% missing data. Furthermore, features related to codes and their corresponding descriptions share identical patterns of missingness, indicating that these values are not missing at random [3]. Consequently, retaining both code and description features is unnecessary.

2.1.3 Descriptive Statistics and Univariate Feature Exploration

2.1.3.1 Numeric Features

The univariate analysis and descriptive statistics revealed that features *Age at Injury* and *Birth Year* contain a minimum value of 0, which is impossible. Furthermore, 50% of workers report an *Average Weekly Wage* of 0. While this could reflect volunteer participation as noted in the metadata, it is unlikely that half of the dataset comprises volunteers, this fact will be addressed during pre-processing. Moreover, this analysis revealed that all features, except *Number of Dependents*, contain outliers. Regarding distributions, *Average Weekly Wage* and *IME-4 Count* are right-skewed, while *Birth Year* is left-skewed due to the anomalous value 0. *Age at Injury* approximates a normal distribution with two peaks, and *Number of Dependents* shows a uniform distribution, confirmed by a Chi-Squared Goodness of Fit test shown in Annex 1, possibly indicating that this feature has likely little predictive power.

2.1.3.2 Categorical Features

The univariate analysis of categorical features revealed several insights, though not all are directly relevant for model development. Concerning the most relevant insights a significant class imbalance was observed in *Alternative Dispute Resolution*, where **99.5%** of workers did not undergo adjudicated processes external to the board, and unknown cases constituted less than **0.1%**, making this minority class insufficient to provide meaningful patterns for model training. For *Gender*, **0.8%** of workers are categorized as Unknown, and less than **0.1%** are non-binary. Additionally, many features exhibit high cardinality, making one-hot encoding impractical due to the risk of feature explosion and increased model complexity, potentially leading to the curse of dimensionality [4].

The target variable also shows severe class imbalance, with the majority class accounting for 50% of the dataset, while the two least common classes collectively represent only **0.1%** (figure 1). This imbalance poses challenges, as models trained without mitigation strategies are likely to perform well on majority classes but poorly on minority ones.

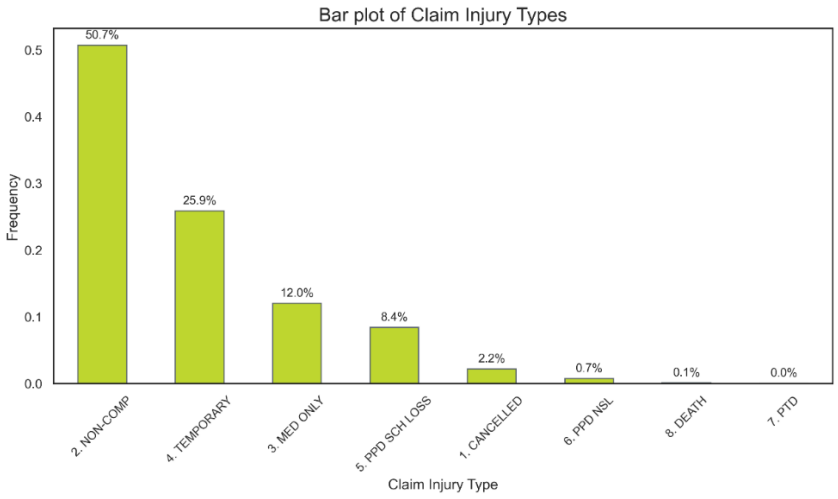


Figure 1: Bar plot of *Claim Injury Type*

2.1.4 Multivariate Analysis

2.1.4.1 Pairwise Relationships of Numerical Features

The analysis of the correlation matrix and pairwise plots revealed strong correlations between several numeric features. *Birth Year* and *Age at Injury* exhibit a negative correlation of **-0.87**, while Assembly Date and Accident Date show a high positive correlation of **0.89**. *C-2 Date* is also strongly correlated with both Accident and Assembly Dates, as illustrated in Figure 2. Additionally, multivariate outliers were detected in the dataset as can be seen in Annex 2.

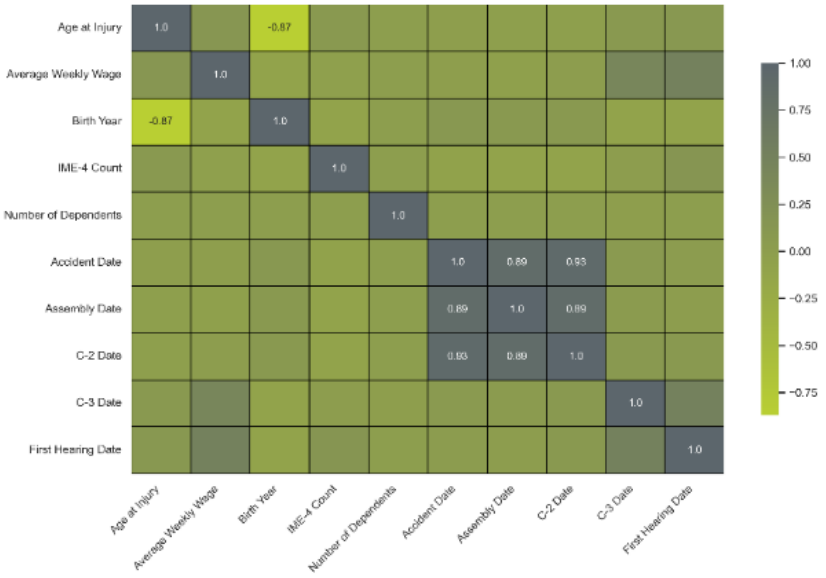


Figure 2: Correlation matrix of the numeric features.

2.1.4.2 Comparing Categorical Features

As previously mentioned, our input features are primarily categorical, making it essential to identify which ones are relevant and which may introduce redundancy. Several sources emphasize the importance of removing highly correlated categorical variables to avoid multicollinearity, which can distort model performance and interpretation [5]. To address this, we analysed the Cramér's V coefficient and observed that certain features, such as *Zip Code* and *District Name*, are strongly associated with *County of Injury*, while *Carrier Type* and *Carrier Name* also exhibit a high association. This suggests that including these features in the model could lead to redundancy, as they provide similar discriminatory power toward the target, nonetheless they can be important to make missing values imputation. The features and their respective Cramér's V coefficient are displayed in Annex 3.

2.1.4.3 Plotting Input Features Against the Target Variable

This analysis revealed that features such as *Age at Injury*, *IME-4 Count*, *Assembly Date* and *Accident Date* exhibit strong discriminatory power toward the target variable, as shown by distinct medians and distributions in the boxplots (Annexes 4, 5 and 6). Variables like *Alternative Dispute Resolution*, *Attorney/Representative*, *Carrier Type*, and *COVID-19 Indicator* also show some discriminatory power, with certain classes displaying distinctive behaviors. For example, when *Attorney/Representative* value equals 'Y', the target class 5. PPD SCH LOSS is very well

represented, whereas it is infrequent when the value is 'N' (Annex 7). Conversely, the *COVID-19 Indicator* shows the opposite pattern (Annex 8). However, for *Alternative Dispute Resolution, Attorney/Representative, and Carrier Type*, the distinctive classes are too minor to be highly meaningful for the model.

2.1.5 Incoherences

Our exploratory analysis revealed several data inconsistencies **that require to be addressed in Data Pre-processing**. For the *Accident Date* feature, we identified 1407 instances where it occurred after the *Assembly Date*, 982 after the *C-2 Date*, 1289 after the *C-3 Date*, and 74 after the *First Hearing Date*. In the *Age at Injury* feature, 5684 cases showed workers injured before age 16, with 5464 of these having an age recorded as 0, and 25140 discrepancies between *Age at Injury* and the difference between *Accident Date* and *Birth Date*. Additionally, there were 168 instances where the *WCIO Nature of Injury Description* field indicated COVID-19, but the case was not identified as such. Lastly, 42011 occurrences of the *WCIO Part of Body Code* contained a negative value (-9), and some Zip Codes did not follow the standard 5-digit format, either containing letters or having fewer than 5 digits.

2.2. Data Preprocessing

2.2.1. Treat Incoherences

In this stage, we addressed the previously identified incoherences. Firstly, we removed rows where the *C-2 Date, C-3 Date, or Assembly Date* occurred before the *Accident Date*. Instances where *Age at Injury* was recorded with values below 16 were replaced by missing values as well as rows where Birth Year or Average Weekly Wage had a value of 0. Additionally, we created a binary flag, *Average Weekly Wage ZERO*, to indicate when the *Average Weekly Wage* was originally 0. The *COVID-19 Indicator* was updated to "Y" for cases where the *WCIO Nature of Injury Description* field explicitly mentioned COVID-19.

For the *Zip Code* feature, we transformed values that could not be converted into integers into missing values, as the American Zip Code format does not support letters within the numbers. Zip Codes below 100 were converted to missing values, as they do not conform to the 5-digit format used for regions in the United States. Zip Codes below 1000 were multiplied by 10 to account for Zip Codes that begin with a leading zero (e.g., 789.0 becomes 07890). Finally, Zip Codes with lengths of 6 or 7 digits, which were mistakenly expressed as floating-point numbers with a ".0" suffix, were converted into integers to correctly represent valid Zip Codes.

2.2.2. Outliers

Outlier removal was performed mainly using the IQR-based method. Specifically, we removed rows from the *Age at Injury* feature that exceeded the upper limit of the IQR, and rows from *Birth Year* that fell below the lower limit. For the *IME-4 Count* variable, the most extreme outliers were capped at a value of 30. In the case of *Average Weekly Wage*, we removed rows where values exceeded 10000, which were considered extreme. This resulted in the removal of less than the **5%** threshold deemed acceptable for outlier treatment.

2.2.3. Feature Engineering

The feature engineering process involved both row-wise and column-wise transformations to optimize the dataset for predictive modelling. For row-wise transformations, the value "U" in the *Alternative Dispute Resolution* feature was recoded as "N," as **99.5%** of observations already belonged to this class, avoid unnecessary imputation. Similarly, the labels "U" and "X" in the *Gender* feature were treated as missing values because they predominantly represented workers in the most common target class. This approach aimed to reduce overfitting by preventing the model from learning irrelevant rules based on rare cases. Additionally, categories such as "5C. SPECIAL FUND – POI CARRIER WCB MENANDS" and "5A. SPECIAL FUND – CONS. COMM. (SECT. 25-A)" were grouped into a new label, "Other," due to their minimal representation (9 rows total) and absence in the test set. Furthermore, codes with identical descriptions were standardized into a single value to avoid introducing biases during training.

Column-wise transformations included removal of the description features (e.g. *Industry Code Description*), as the corresponding codes were retained for training. Features absent in the test set, such as *WCB Decis*ion and *Agreement Reached*, were also dropped, along with *OIICS Nature of Injury Description* due to only having missing values. Categorical variables were transformed into numeric using frequency encoding to ensure compatibility with scikit-learn algorithms that can't handle categorical features natively. Additionally, date features were converted into the number of days since January 1, 2000, since scikit-learn can't handle dates natively.

2.2.4. Missing value imputation

The final step in data preprocessing involved imputing missing values. Initially missing values in *Birth Year* were resolved using the difference between *Accident Date* and *Age at Injury*, while missing *Age at Injury* values were calculated as the difference between *Accident Date* and *Birth Year,* this approach did not solve all the missing values in this variables. For *First Hearing Date*, missing values, indicate that no hearing was held, were assigned a date in the future (2030-01-01) and missing values in IME-4 Count were assigned the value 0, considering no report was received for those claims.

Conditional means based on Cramer's V findings explained previously were used to impute other features: missing *Zip Code* values were assigned the mode grouped by *County of Injury*, *Industry Code* by the mode grouped by *Carrier Name* and by *Carrier Type*, and *Average Weekly Wage* by the mean grouped by *Industry Code*. Missing values in *Gender*, *WCIO Cause of Injury, WCIO Nature of Injury Code, and WCIO Part of Body Code* were imputed with the mode grouped by *Industry Code*. If there were still missing values, they would be imputed with the global mean/mode of the column in the dataset.

For remaining missing values in *Accident Date* and *Birth Year*, a K-Nearest Neighbours Imputer (K=5) was employed, with prior scaling to ensure unbiased imputation, finalizing with imputing the remaining missing values in *Age at Injury* the same way as previously described.

3. Multiclass Classification

3.1. Additional Features

Additionally, to the features that the dataset already provided, our team decided to create some more and explore their relationships towards the target. From all those that were explored during EDA the ones that were selected to be tested in the model were *Time to Assembly*, *C-2 Report Status* and *C-3 Report Status*, whose descriptions can be seen in table 2.

Table 1: Description of the additional features

Feature	Description
Time to Assembly	The number of days between the *Assembly Date* and *Accident Date*, representing the time interval.
C-2 Report Status	Indicates if the report was "Not Received," "Received on or before Assembly," or "Received After Assembly."
C-3 Report Status	Indicates if the report was "Not Received," "Received on or before Assembly," or "Received After Assembly."

The features *C-2 Report Status* and *C-3 Report Status* were created to replace *C-2 Date* and *C-3 Date*, addressing their high correlation and unhandled missing values while preserving predictive power towards the target. Our analysis showed that most C-2 reports were received on or before assembly, indicating their importance in case assembly, as their absence often results in cancelled or "2. NON-COMP" cases. Conversely, most C-3 Reports were not received and appear less significant for case assembly.

3.2. Feature Selection

Feature selection was integrated into the cross-validation process during model assessment, combining filter, wrapper, and embedded methods to identify and exclude redundant or irrelevant features. Using Spearman correlation as a filter method, we flagged features with an absolute correlation coefficient above 0.8 for potential exclusion. Recursive Feature Elimination (RFE) with logistic regression served as the wrapper method, where data was scaled to ensure unbiased results, with the two least important features being removed at each step to enhance computational efficiency, starting with 6 features, as using fewer than six would likely result in an overly simplistic model incapable of accurately predicting the target variable. Finally, Lasso Regression acted as an embedded method, eliminating features whose coefficients equal zero, thereby simplifying the model while preserving relevant predictors. Features deemed irrelevant by at least two methods were excluded from the final model leading to the removal of *Accident Date, Birth Year, Number of Dependents and Medical Fee Region,* since they were removed in most of the folds in cross-validation being coherent with our EDA findings.

3.3. Model Assessment

To identify the best algorithm for our model, we used Stratified K-Fold cross-validation with 5 folds on a stratified 40% sample of the data. This method preserved class distribution, making it ideal for our imbalanced dataset, and integrated the pre-processing and feature selection steps, ensuring a robust and consistent evaluation. Our team tested seven algorithms: Decision Tree, Gaussian Naïve Bayes, K-Nearest Neighbours, Logistic Regression, Multi-Layer Perceptron, Random Forest, and Extreme Gradient Boosting.

The algorithms were evaluated using precision, recall, and macro F1-score. Precision was chosen to minimize false positives, especially for sensitive classes like "8. Death" where incorrect predictions could lead to legal complications. Recall was included to ensure that minority classes like "8. Death" or "7. PTD" are not overlooked, reducing the risk of missing critical cases. Macro F1-score was chosen since it balances precision and recall across all classes, ensuring a fair performance on both majority and minority classes, aligning with Kaggle's evaluation criteria. The performance of the algorithms based on these metrics is summarized in Table 3.

Table 2: Performance Metrics of Evaluated Models (Training and Validation)

	Precision_train	Precision_val	Recall_train	Recall_val	F1_train	F1_val
Decision Tree	0.9999	0.3698	0.9999	0.3759	0.9999	0.3724
Gaussian Naïve Bayes	0.2834	0.2786	0.3211	0.3187	0.1244	0.1238
KNN	0.604	0.439	0.4244	0.3563	0.4590	0.3754
Logistic Regression	0.3756	0.3769	0.2903	0.291	0.2902	0.291
MLP	0.1061	0.0809	0.1323	0.1322	0.0904	0.0901
Random Forest	0.9999	0.5365	0.9999	0.3772	0.9999	0.3931
Extreme Gradient Boosting	0.8585	0.5602	0.6764	0.4253	0.7001	0.4461

Analyzing Table 3, we observe that while MLP, Logistic Regression, and Gaussian Naïve Bayes show low levels of overfitting, their validation performance does not surpass that of Extreme Gradient Boosting, that was optimized in the next stage.

3.4. Model Optimization

To optimize the best model identified in the previous stage, we employed Grid Search within our cross-validation pipeline, focusing on tuning hyperparameters to maximize performance. As Grid Search evaluates all possible combinations in the parameter grid, we used smaller grids to improve computational efficiency focusing our optimization efforts on Extreme Gradient Boosting since it was the best-performing algorithm, though we also ran Grid Search for other algorithms. The results, including the best hyperparameter sets and corresponding performances, are detailed in Annexes 9 and 10. Interestingly, Decision Tree outperforms Random Forest, likely due to the hyperparameters grid provided. However, studies indicate that Random Forest generally outperforms Decision Tree under optimized conditions [6–8].

The key hyperparameters tuned for Extreme Gradient Boosting were: *eta* (learning rate), *gamma* (minimum loss reduction for splits), *max_depth* (tree depth), *reg_alpha* (L1 regularization), and *reg_lambda* (L2 regularization). These parameters were chosen not only to enhance performance but also to mitigate overfitting, specifically a moderate *eta* value allows gradual learning, *gamma* ensures splits are meaningful, *max_depth* balances model complexity, while *reg_alpha* and *reg_lambda* add regularization to prevent overfitting [9 – 10].

Based on the results from cross-validation after applying Grid Search, Extreme Gradient Boosting performance on the validation set improved on average from **0.4461** to **0.4476** in macro F1-score, being (*eta* = 0.2, *gamma* = 0.2, *max_depth* = 8, *reg_alpha* = 5, *reg_lambda* = 5) the hyperparameters used for the majority of the folds and for that reason considered the most optimized hyperparameters. Overfitting was significantly mitigated across all models with the gap between training and validation performance decreased markedly, from **25.4%** to just **6.4%**, in our best-performing model. Finally, we trained our model with the optimized hyperparameters in the entire training dataset and made predictions on the test, whose macro f1-score in 20% of the test set was **0.41**.

4. Open-Ended Section

To enhance user interaction with our predictive model for *Claim Injury Type*, we developed an interactive interface using Python's **Tkinter** library. This library was selected for its simplicity and native support for building graphical interfaces, enabling us to create a lightweight, user-friendly application. To manage model serialization and compatibility with preprocessing steps, we used the **Joblib** library, which efficiently handles large objects like NumPy arrays, making it more suitable than Python's default serializer, Pickle.

The interface was designed with usability in mind, incorporating dropdown menus for fields with predefined options, auto-population of descriptive fields based on selected codes, and date entry widgets for date-specific inputs. Before generating predictions, the interface preprocesses user-provided data to ensure consistency with the format used during model training. This preprocessing includes correcting inconsistencies, handling missing values, and applying frequency encoding.

When a user clicks the "Predict" button, the system seamlessly processes the input data and feeds it into the model. To validate the interface, we conducted comprehensive testing to ensure functionality, accuracy, and alignment with the training pipeline that are expressed below:

Persona 1: Our first persona is a 21-year-old female born in 2003, whose Claim ID is 6166141. She represents a typical young individual navigating a straightforward claim scenario. Her case details, as shown in Figure 3, were entered into the interface for analysis being seamlessly processed and accurately predicting her Claim Injury Type as **"2. NON-COMP"**.

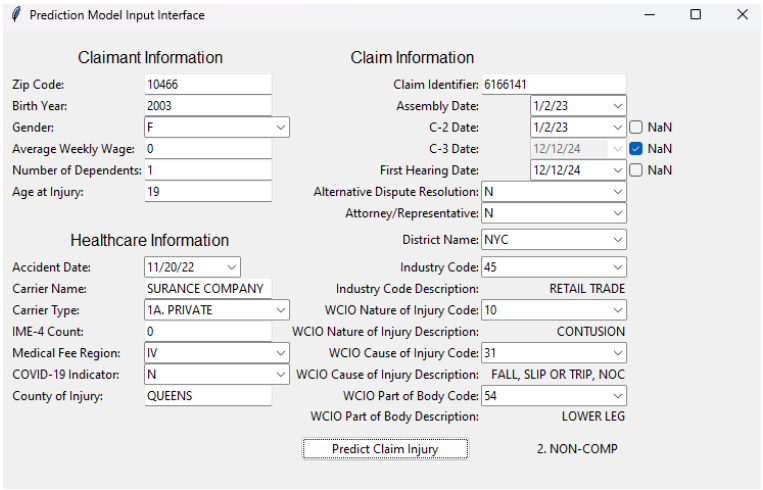


Figure 3: Input Data for Persona 1 in the Predictive Interface

Persona 2: Our second persona is a 36-year-old male born in 1988, with Claim ID 6610113. His case, illustrated in figure 4, posed a unique challenge: the Assembly Date was from 2024, a scenario not represented in the model's training data. Despite this, the interface predicted his Claim Injury Type as **"2. NON-COMP"**, the same as Persona 1. However, the true value was **"3. MED ONLY,"** resulting in a misclassification. This case highlights that while the prediction was incorrect, such errors are expected in real-world applications due to inherent imperfections in our machine learning model, that for obvious reasons can't learn all the possible cases that exist.

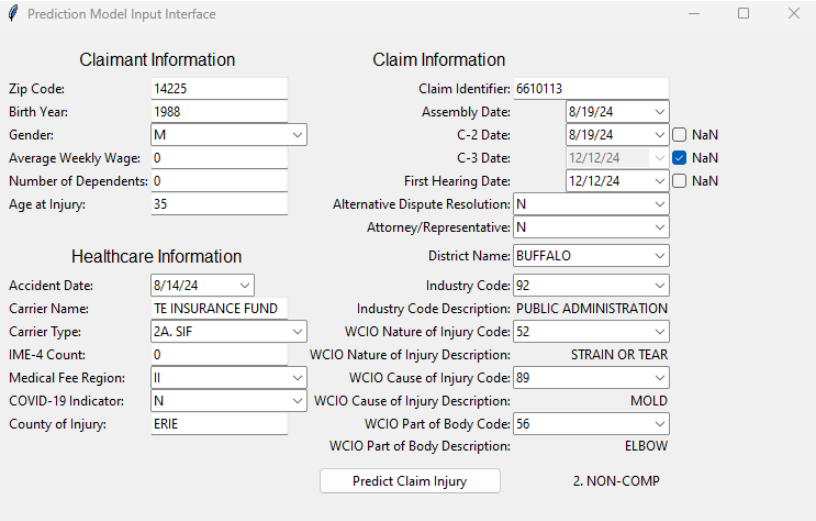


Figure 4: Input Data for Persona 2 in the Predictive Interface

In conclusion, the interactive interface successfully bridges the gap between machine learning predictions and user accessibility, providing accurate results in most scenarios. However, it also has areas for improvement, particularly in addressing misclassifications, such as those observed with Persona 2. Enhancements like better exception handling, clearer error messages, and the ability to auto-populate data for existing claims could further improve the system's usability and efficiency.

5. Conclusion

This project demonstrated that Extreme Gradient Boosting was the most effective algorithm for predicting the Workers' Compensation Board's (WCB) final decision on *Claim Injury Type*, achieving a macro F1-score of approximately **0.45** on the validation set and a moderate-low overfitting rate of **6.4%**, suggesting that it generalizes well to unseen data. The model's success can be attributed to a robust data preprocessing approach combined with cross-validation, which helped improve performance and reduce overfitting. Despite these promising results, there are several areas for improvement that could further enhance the model's performance and usability. Enhancing the pre-processing pipeline and integrating multiple models into a stacking ensemble may yield better results than relying on individual models. Testing additional hyperparameters could also help reduce overfitting and improve overall model performance. Furthermore, experimenting with under-sampling and over-sampling techniques could address class imbalances, thereby increasing the model's ability to accurately predict minority classes. Additionally, developing an executable file for the Open-Ended section would improve

accessibility and usability, providing users with a more direct interface to assess WCB's injury classification decisions.

In conclusion, while the project produced a reasonably performing model and accessible interface, there is considerable potential for further optimization, including enhancing model performance, exploring ensemble techniques, and expanding user interaction capabilities.

6. References

[1] Davoudi Kakhki, F., Freeman, S. A., & Mosher, G. A. (2019). Evaluating machine learning performance in predicting injury severity in agribusiness industries. Safety Science, 117, 257–262. https://doi.org/10.1016/j.ssci.2019.04.026

[2] Vallmuur, K. (2015). Machine learning approaches to analysing textual injury surveillance data: A systematic review. Accident Analysis & Prevention, 79, 41–49. https://doi.org/10.1016/j.aap.2015.03.018

[3] Pham, T. M., Pandis, N., & White, I. R. (2022). Missing data, part 2. Missing data mechanisms: Missing completely at random, missing at random, missing not at random, and why they matter. *American Journal of Orthodontics and Dentofacial Orthopedics*, *162*(1), 138-139.

[4] Cerda, P., & Varoquaux, G. (2022). Encoding High-Cardinality String Categorical Variables. *IEEE Transactions on Knowledge and Data Engineering*, *34*(3), 1164–1176. IEEE Transactions on Knowledge and Data Engineering. https://doi.org/10.1109/TKDE.2020.2992529

[5] James, G. (2013). An introduction to statistical learning.

[6] Madaan, M., Kumar, A., Keshri, C., Jain, R., & Nagrath, P. (2021). Loan default prediction using decision trees and random forest: A comparative study. IOP Conference Series: Materials Science and Engineering, 1022(1), 012042. https://doi.org/10.1088/1757-899X/1022/1/012042

[7] Esmaily, H., Tayefi, M., Doosti, H., Ghayour-Mobarhan, M., Nezami, H., & Amirabadizadeh, A. (2018). A Comparison between Decision Tree and Random Forest in Determining the Risk Factors Associated with Type 2 Diabetes. Journal of Research in Health Sciences, 18(2), 412.

[8] Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random forests and decision trees. *International Journal of Computer Science Issues (IJCSI)*, *9*(5), 272.

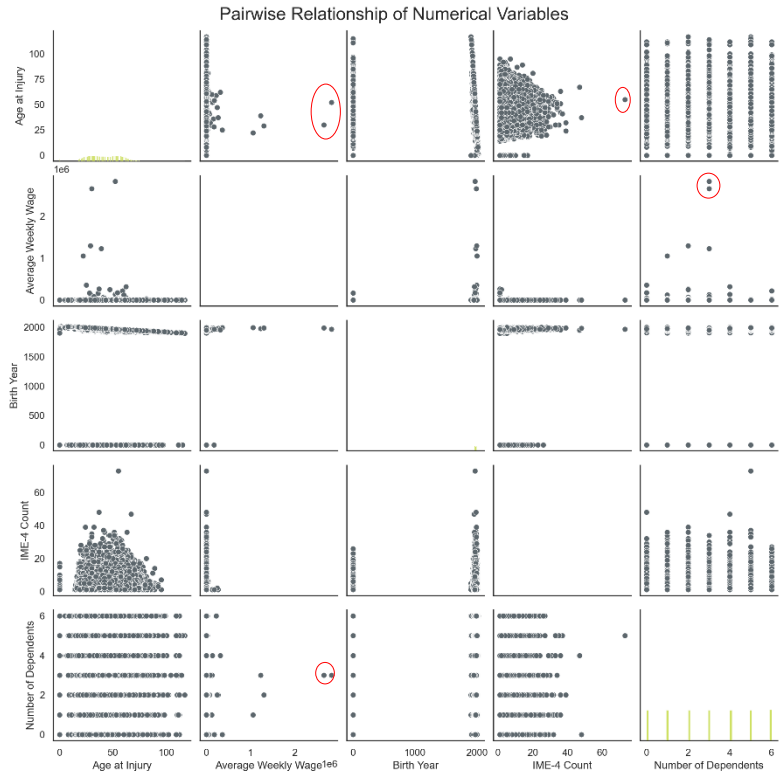
[9] Putatunda, S., & Rama, K. (2018). A Comparative Analysis of Hyperopt as Against Other Approaches for Hyper-Parameter Optimization of XGBoost. Proceedings of the 2018 International Conference on Signal Processing and Machine Learning, 6–10. https://doi.org/10.1145/3297067.3297080

[10] Notes on Parameter Tuning—Xgboost 2.1.1 documentation. (n.d.). Retrieved 13 December 2024, from https://xgboost.readthedocs.io/en/stable/tutorials/param_tuning.html

Annexes

Annex 1: Results of the Chi-Squared Goodness of Fit Test for the Distribution of Number of Dependents

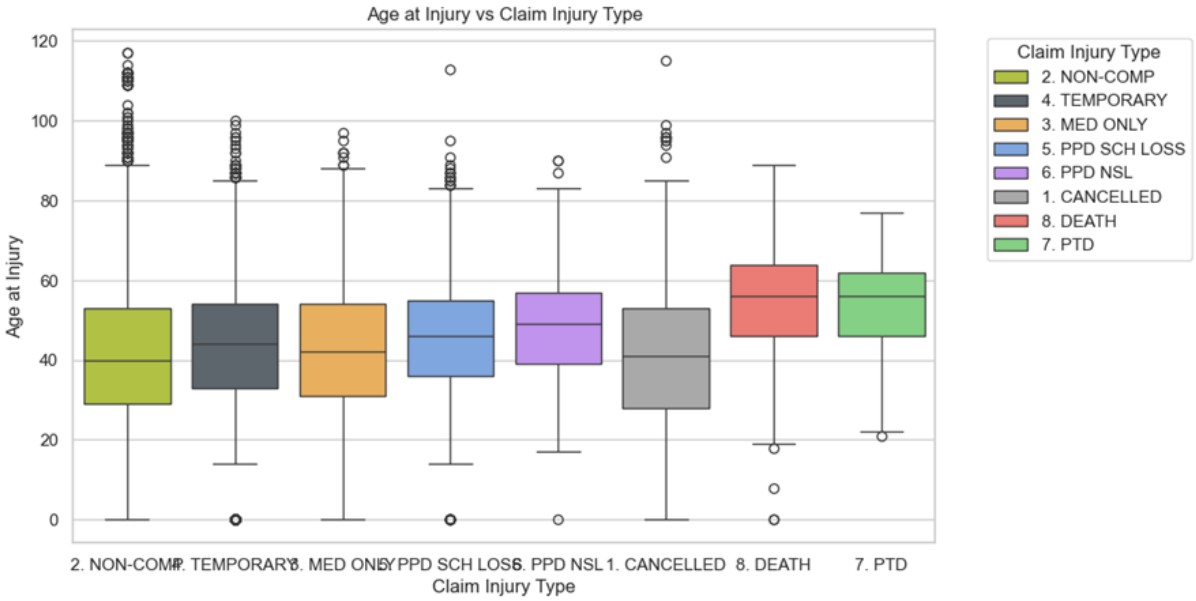
Metric	**Value**
Chi-statistic	10.10
p-value	0.1203
Conclusion:	Number of Dependents follows a Uniform Distribution



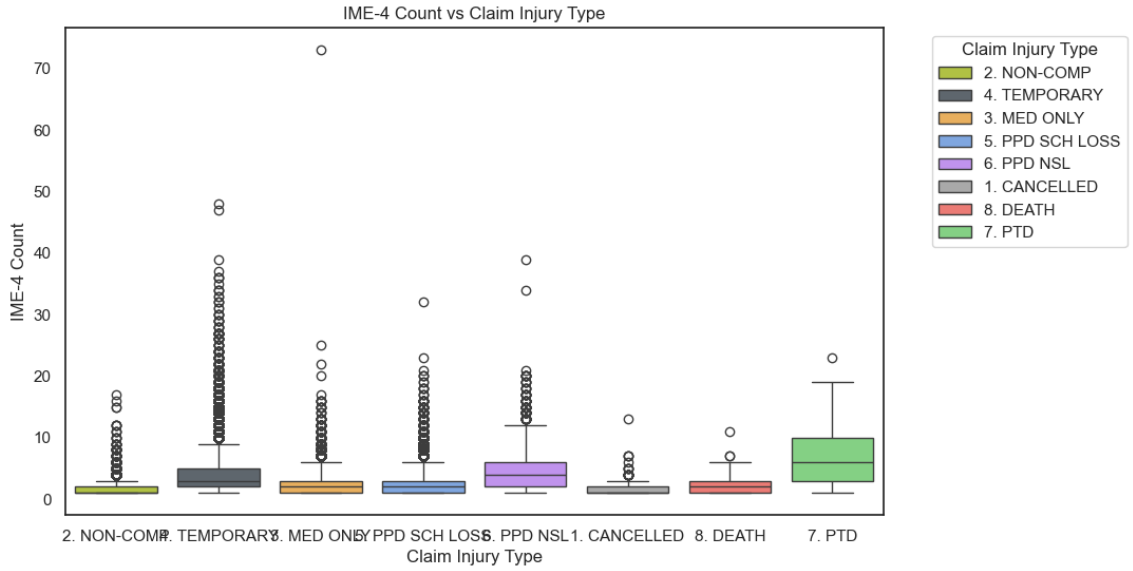
Annex 2: Scatterplots of numeric features highlighting multivariate outliers (indicated by red circles)

Annex 3: Cramer's V for categorical features

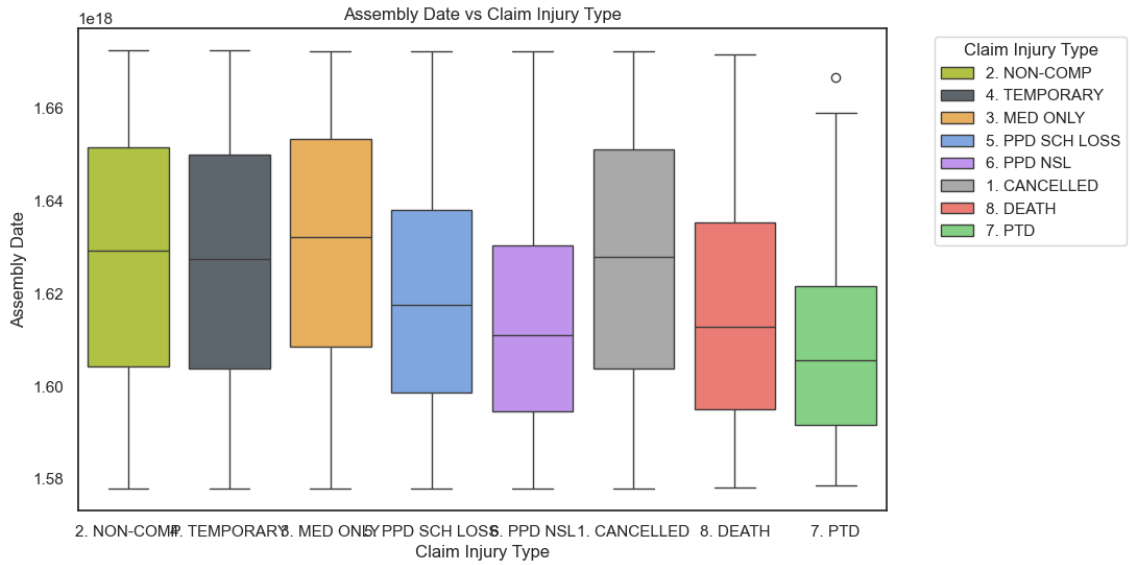
Features tested	Cramer's V	Conclusion
Zip Code and *Medical Fee Region*	1	Strong association
Zip Code and *County of Injury*	0.947	Strong association
Zip Code and *District Name*	0.904	Strong association
County of Injury and *District Name*	0.909	Strong association
District Name and *Medical Fee Region*	0.492	Moderate association
Carrier Name and *Industry Code*	0.488	Moderate association
WCIO Cause of Injury Code and *Industry Code*	0.15	Moderate to weak association
Gender and *Industry Code*	0.254	Moderate to weak association



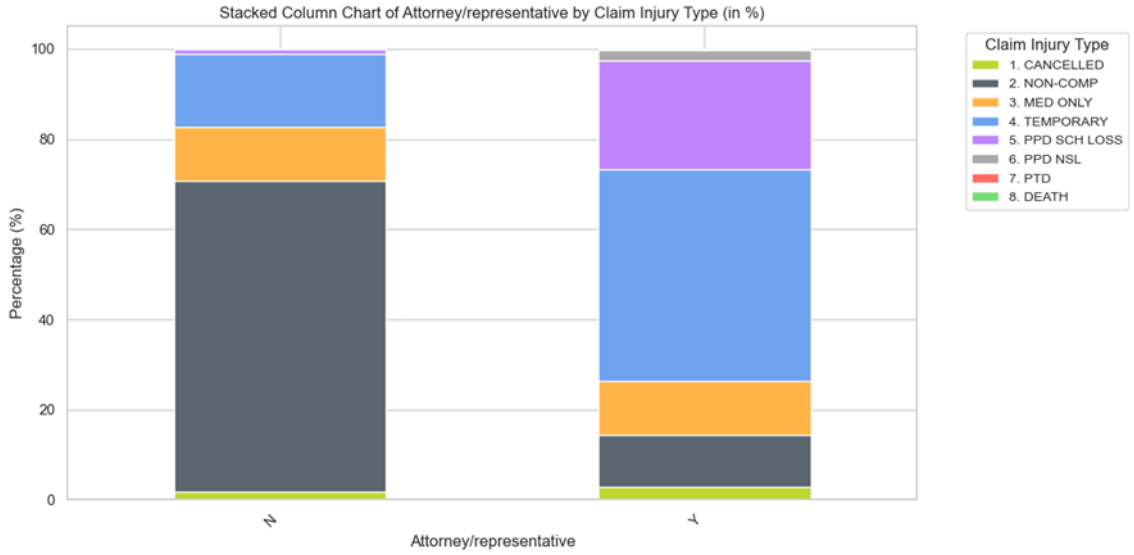
Annex 4: Boxplot of *Age at Injury* by *Claim Injury Type*



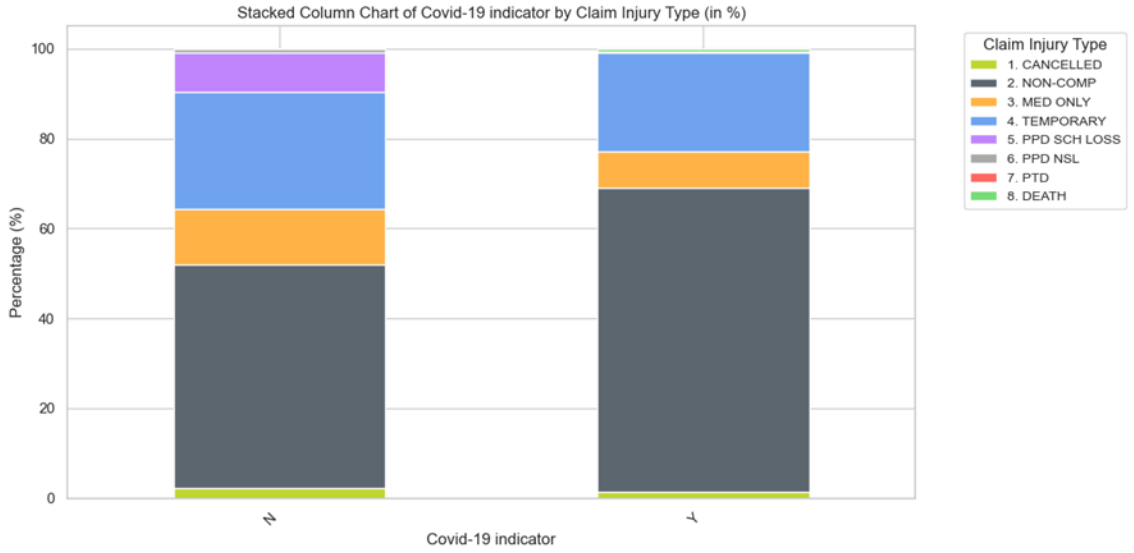
Annex 5: Boxplot of *IME-4 Count* by *Claim Injury Type*



Annex 6: Boxplot of *Assembly Date* by *Claim Injury Type*



Annex 7: Stacked column chart of *Attorney/Representative* by *Claim Injury Type*



Annex 8: Stacked column chart of *COVID-19 Indicator* by *Claim Injury Type*

Annex 9: Optimal Hyperparameter Set from Grid Search to each model

Algorithm	**Hyperparameters**
Decision Tree	max_depth = 10, min_sample_leaf = 10, min_samples_split = 2
Random Forest	max_depth = 20, min_samples_leaf = 1, min_samples_split = 2, n_estimators = 50
Extreme Gradient Boosting	eta = 0.2, gamma = 0.2, max_depth = 8, reg_alpha = 5, reg_lambda = 5
Logistic Regression	C = 0.1, class_weight = balanced, penalty = "l2", solver = "lbfgs"
Naïve Bayes	var_smoothing = 1e-07
Multi Layer Perceptron	activation = "relu", alpha = 0.0001, hidden_layer_sizes = 100, max_iter = 200
KNN	n_neighbors = 5, p = 2, weight = distance

Annex 10: Performance Metrics of Evaluated Models after Grid Search (Training and Validation)

	Precision_train	**Precision_val**	**Recall_train**	**Recall_val**	**F1_train**	**F1_val**
Decision Tree	0.5883	0.5327	0.3872	0.3768	0.4107	0.3962
Gaussian Naïve Bayes	0.2813	0.28	0.3009	0.2917	0.1389	0.1386
KNN	0.9209	0.4269	0.8843	0.3653	0.8913	0.3824
Logistic Regression	0.3377	0.3386	0.4441	0.4344	0.2946	0.295
MLP	0.1339	0.1337	0.1701	0.1701	0.1309	0.1307
Random Forest	0.9608	0.5489	0.6965	0.3732	0.7663	0.3881
Extreme Gradient Boosting	0.7445	0.5575	0.4835	0.4283	0.5119	0.4476