MACHINE LEARNING PROJECT

Master in Data Science and Advanced Analytics

NOVA Information Management School

Universidade Nova de Lisboa

To Grant or Not to Grant: Deciding on Compensation Benefits

Fall/Spring Semester 2024-2025

Group 38

Ana Marta Azinheira | 20240496

Bráulio Damba | 20240007

Catarina Ribeirinha | 20240507

Marco Galão | r20201545

Rodrigo Sardinha | 20211627

Table of Contents

1. Abstract....1
2. Introduction....2
3. Data Exploration....2
3.1. Index, Constant Features, Duplicates and Data Types....3
3.2. Missing Values (Initial Treatment)....3
3.3. Coherence Checking....4
4. Cross-Validation....5
4.1. Outliers and Missing Values....6
4.2. Feature Engineering....7
4.3. Encoding and Scaling....7
4.4. Feature Selection....8
4.5. Model Training and Optimization....9
4.6. Final Model....9
5. Open-Ended Section....10
6. Conclusion....10
7. Bibliography....12
8. Annexes....13

1. Abstract

This report presents the culmination of our Machine Learning (ML) course project, aimed at developing a robust predictive model to assist the New York Workers' Compensation Board (WCB) (Bib. 1) in automating the classification of workplace injury claims - a process traditionally performed manually and prone to inefficiencies.

Our work followed the CRISP-DM framework, beginning with business understanding to align our objectives with the New York WCB's needs. During data exploration, we identified key patterns, missing values, inconsistencies, and outliers. In the data preparation phase, we addressed these issues and performed additional steps such as feature engineering and selection to optimize the dataset.

In the modeling phase, we trained a range of ML algorithms, categorized into simpler and more complex models. The simpler models included Logistic Regression, Naive Bayes, and Decision Tree, while the more complex models included Adaboost, Bagging, Random Forest, XGBoost, and Neural Network. After fine-tuning model parameters and preprocessing strategies, and using cross-validation along with the macro F1-score as performance metrics, the Neural Network was selected as the best model. It was chosen for its consistent performance, lower overfitting, and efficient computational cost.

In the deployment phase, we developed a web application to deliver real-time injury type predictions and analytical insights. The application allows stakeholders to upload new claim data and receive automated predictions, while an interactive dashboard offers valuable insights into claim patterns and trends. Ultimately, this solution is designed to integrate with the New York WCB's online claim submission system (Bib. 2), enabling real-time claim classification with minimal manual effort.

Our findings demonstrate the potential of ML to streamline the WCB's decision-making process, significantly reducing manual effort while maintaining accuracy. Limitations, including data quality and model interpretability, were acknowledged.

The full project is available on our GitHub repository (Bib. 3) for further exploration and collaboration.

2. Introduction

The New York WCB is responsible for reviewing and classifying workplace injury claims, ensuring workers receive the compensation they are entitled to. However, the current manual process is labor-intensive, time-consuming, and prone to inefficiencies. With the increasing volume of claims, the need for an automated solution has become critical.

Manual claim classification leads to delays and inconsistencies in decision-making, affecting both the speed and accuracy of compensation delivery. Automation could alleviate these bottlenecks, enhancing efficiency and minimizing human error in the process.

ML has shown great promise in automating classification tasks across industries such as healthcare and retail, where it is used, for example, to predict diseases and optimize inventory management. By training on historical claim data, ML can predict injury types based on data patterns, eliminating the need for manual input. This approach increases efficiency and ensures consistent, unbiased decision-making - key for the New York WCB's operations.

A core aspect of this project is the development of a web application to deploy the ML model. This application allows stakeholders to upload claim data and receive real-time injury predictions. Designed to integrate seamlessly with the New York WCB's online claim submission system, it enables real-time classification with minimal manual effort. Additionally, the interactive dashboard offers valuable insights into claims, further enhancing usability and supporting informed decision-making.

This report outlines the steps taken to develop and deploy the predictive model, from data exploration and preparation to model evaluation and optimization. It also covers the development of the web application, providing a practical solution to the New York WCB's claim classification challenge.

3. Data Exploration

Analyzing the multivariate relationships between the target variable and key features reveals notable patterns. For instance, males are more frequent across all injury types, but their dominance is particularly striking in the "8. DEATH" category (Figure 1). The COVID-19 Indicator shows the highest prevalence in "8. DEATH," suggesting a strong correlation between the pandemic and fatalities (Figure 2). Regarding *Industry Code*, "Educational Services" and "Retail Trade" exhibit similar distributions of injury types, while "Public Administration" and "Transportation and Warehousing" share identical proportions for "2. NON-COMP" and "3. MED ONLY" claims (Figure 3). These insights provide valuable context for understanding claim dynamics and guiding targeted interventions.

3.1. Index, Constant Features, Duplicates and Data Types

As the next step in our exploration, we analyzed all features in the training dataset, including setting the *Claim Identifier* as the index, and identified initial issues such as constant features, duplicates, and incorrect data types, as shown in Table 1.

Table 1 - Index, Constant Features, Duplicates and Data Types Summary

Category	Feature	Issue	Decision
Index	*Claim Identifier*	Inconsistent length	Drop all 9-length *Claim Identifier* records (3.28% of data) due to missing values in all features except *Assembly Date* (Figure 4), and set the 7-length *Claim Identifier* as the index
Constant Features	-	No constant features found, though *Alternative Dispute Resolution* and *COVID-19 Indicator* are highly imbalanced	No constant features to drop
Duplicates	All features	1 duplicate observation	Drop duplicate record to ensure data integrity
Data Types	Code features	Incorrect data type (float)	Convert to object
	Age at Injury *Birth Year* *Number of Dependents*	Incorrect data type (float)	Convert to integer
	Date features	Incorrect data type (object)	Convert to datetime
	Alternative Dispute Resolution *Attorney/Representative* *COVID-19 Indicator* *Gender*	Incorrect type (object)	Convert to binary (Gender was renamed as Male)

3.2. Missing Values (Initial Treatment)

As part of our initial preprocessing, we identified and addressed missing values (Figure 5) where feasible, with the remaining treatment performed in subsequent steps. Table 2 summarizes the actions taken and their rationale.

Table 2 - Missing Values Summary

Feature	Issue	Missing %	Decision
OIICS Nature of Injury Description	All rows missing	100	Drop column. Fully missing, adds no value to analysis.

Feature	Issue	Missing %	Decision
Carrier Type	"UNKNOWN" values	0.48	Change "UNKNOWN" to NaN
Gender	"U" and "X" values	0.82 and 0.008, respectively	Change "U" to NaN; drop "X" due to its insignificance
Medical Fee Region	"UK" values	5.83	Change "UK" to NaN
Alternative Dispute Resolution	"U" values	0.0009	Change "U" (unknown) to NaN
County of Injury	"UNKNOWN" values	0.21	Change "UNKNOWN" to NaN
IME-4 Count	NaN values	76.87	Drop column. High missingness and low value for predictive power (Figure 6)
First Hearing and *C-3 Date*	NaN values	73.73 and 67.39, respectively	Keep columns. Missingness indicates no hearing held or form not received yet

We also decided to drop observations with 10 or more missing values (35.71% of the columns), which affected only 0.99% of the dataset (Figure 7). This minimal data loss helps maintain a high-quality dataset without significantly impacting the analysis.

3.3. Coherence Checking

The coherence checks were conducted to identify and correct inconsistencies, thereby improving the dataset's accuracy and usability (Table 3).

Table 3 - Coherence Checking Summary

Check	Purpose	Findings	Decision
Assembly Date	*Ensure claims are from 2020–2022*	No incoherencies	-
Chronological Order	Verify date sequence: *Accident* <= *C-2* <= *C-3* <= *Assembly* < *First Hearing* (Bib. 4)	179,563 incoherencies (31.37%)	Remove rows where *Assembly Date* <= *Accident Date* (0.86%). Adjust *C-2* and *C-3 Date*'s based on *Assembly Date,* considered the most reliable feature with no missing values (Figure 8)
C-3 Date	Should be filed within 2 years of the Accident Date (Bib. 4)	6,606 (1.16%) exceed limit	Remove rows where *C-3 Date* (use *Assembly Date* as proxy for missing values) > 2 years after *Accident Date*
Minimum *Age at injury*	Ensure compliance with legal working age restrictions	2,050 records (0.36%) for ages 14–17 (allowed under restricted conditions - Bib. 5); 6,938 records (1.23%) for individuals under 18	Remove all records for individuals under 18
Birth Year	Check incoherencies and correlation with *Age at Injury*	5.04% missing, 3.97% zero values; low correlation with *Age at Injury* (-0.03).	Drop *Birth Year*. *Age at Injury* is reliable, has no missing values, and

Check	Purpose	Findings	Decision
			provides the same information without incoherencies
Number of Dependents	Explore distribution	Uniform distribution (Figure 9)	Drop feature due to data quality concerns
Zip Code	Validate *Zip Code* format (Bib. 6 and 7)	14,934 (2.68%) invalid *Zip Code*'s (Figure 10 e 11)	Convert invalid ZIP Codes to NaN
Code Columns	Align codes with references (Bib. 8 and 9)	*WCIO Part of Body Code*: 41,328 (7.41%) have "-9" labeled as "MULTIPLE"	Change "-9" to "90" and update description to "Multiple Body Parts"

The coherence checks led to the removal of 14,948 (2.61%) records from the dataset. With these incoherencies addressed.

4. Cross-Validation

Before modeling, we implemented a cross-validation framework (Figure 12) using Repeated Stratified KFold Cross-Validation (3 splits, 2 repetitions) to ensure robust model selection and consistent target distribution across splits. For each iteration, all preprocessing steps (outlier treatment, missing values imputation, feature engineering, encoding and scaling) were applied to training and validation datasets, avoiding data leakage by processing validation data using only training information.

To address class imbalance (Figure 13), we applied Synthetic Minority Over-Sampling Technique (SMOTE) to generate synthetic samples for minority classes and combined it with undersampling to reduce the size of majority classes. This approach reduced the degree of imbalance while maintaining a meaningful class distribution.

After balancing, we conducted feature selection and used Randomized Search to optimize hyperparameters for multiple models. Each model was trained and evaluated on its respective split, with metrics such as Macro F1-Score, computation time, and optimal parameters tracked in a summary dictionary. This structured approach ensured systematic model evaluation and optimization.

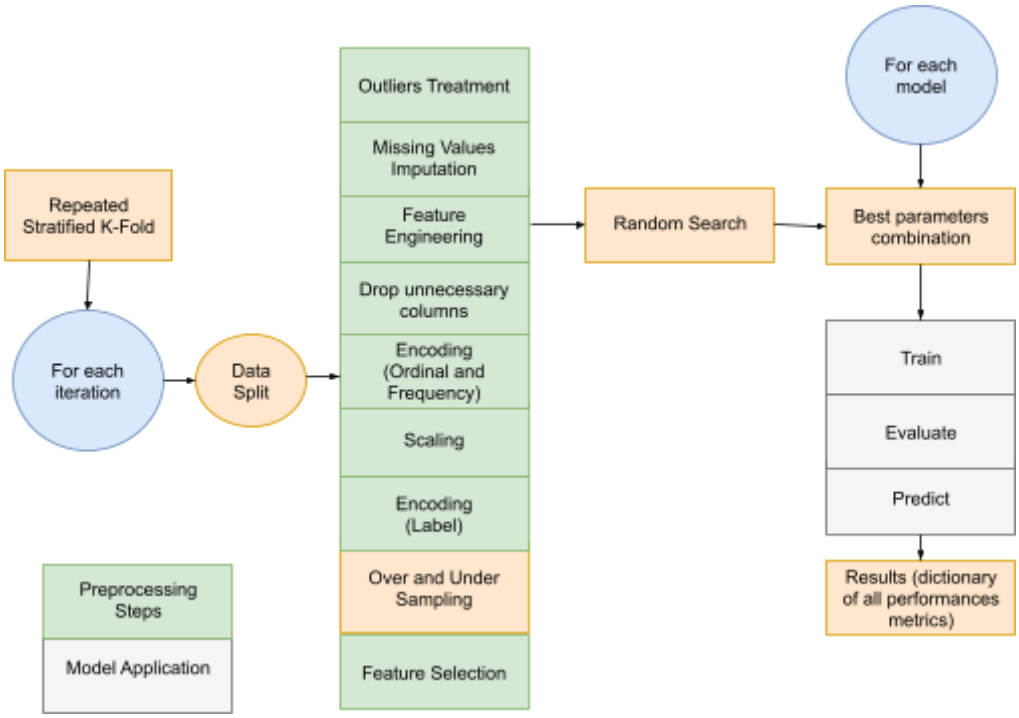

Figure 12 - Cross-Validation Framework

4.1. Outliers and Missing Values

To address outliers (Figure 14), we evaluated IQR, Manual Filtering, and Winsorization (Figure 15). IQR was excluded as it removed 77,977 observations (13.99%). Instead, we applied Manual Filtering and Winsorization to the most problematic variables: *Age at Injury* and *Average Weekly Wage* (Table 4). These conclusions were based on the entire training dataset (pre-split) for reference. After handling outliers, missing values were efficiently imputed using central tendency measures: the median for numeric variables and the mode for categorical ones.

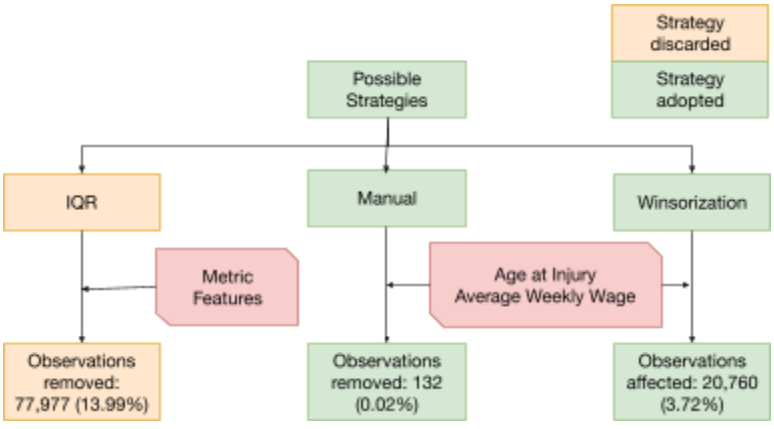

Figure 15 - Possible Outliers Treatments Flowchart

4.2. Feature Engineering

To improve prediction accuracy for our target variable, we created several new features (Table 5).

Table 5 - Feature Engineering summary

Category	Feature Name	Description	Purpose
Missing Date Flags	*C-2 Date Missing*	Binary indicator for missing *C-2 Date*	Highlights absence due to potential delays in claim filing
	C-3 Date Missing	Binary indicator for missing *C-3 Date*	Highlights absence due to potential delays in claim filing
	First Hearing Date Missing	Binary indicator for missing *First Hearing Date*	Indicates unresolved or incomplete cases
Age and Wage-based	*Age Group at Injury*	Groups *Age at Injury* into bins	Categorizes as: "Young Adult" (≤25), "Adult" (26-35), "Mid-aged" (36-45), "Older Adult" (46-60), "Senior" (>60)
	Average Weekly Wage Category	Groups *Average Weekly Wage* into bins	Categorizes as: "Very Low" (≤50), "Low" (51-300), "Medium" (301-1,000), "High" (1,001-2,000)," Very High" (>2,000)
Aggregated	*Average Weekly Wage by Medical Fee Region*	Groups *Average Weekly Wage* by *Medical Fee Region*	Reflects regional earning averages
	Average Weekly Wage by Gender	Groups *Average Weekly Wage* by *Gender*	Highlights gender-based earning differences
	*Age * Avg Weekly Wage*	Interaction term between *Age at Injury* and *Average Weekly Wage*	Combines age and wage effects for analysis
Category-based (Bib. 9)	*Part of Body Category*	Groups *WCIO Part Of Body Code*'s into broader categories	Consolidates codes into: "Head", "Neck", among others
	Nature of Injury Category	Groups *WCIO Nature of Injury Code*'s into broader categories	Consolidates codes into: "Specific Injury", "Occupational Disease or Cumulative Injury", among others
	Cause of Injury Category	Groups *WCIO Cause of Injury Code*'s into broader categories	Consolidates codes into: "Burn or Scald", "Caught In, Under or Between", among others

4.3. Encoding and Scaling

To prepare the data for modeling, we first encoded categorical variables to convert them into numerical values, making them suitable for machine learning algorithms. The Ordinal Encoder was used for ordinal features (*Age Group at Injury*, *Average Weekly Wage Category*), while Frequency Encoder was applied to high-cardinality variables (such as *Carrier Type*, and *County of Injury*) to capture the frequency of each category. The target variable was encoded using Label Encoder. Additionally, we scaled the numerical data using Standard Scaler to standardize the features, ensuring equal contribution from all variables and enhancing model performance.

4.4. Feature Selection

"Relevant features lead to efficient models, not more features! Note that including a large number of features might lead to an overfitting problem" (Swamynathan, 2017) (Bib. 10). To avoid this, we performed feature selection using various methods (filter, wrapper, and embedded), as outlined in Figure 16.

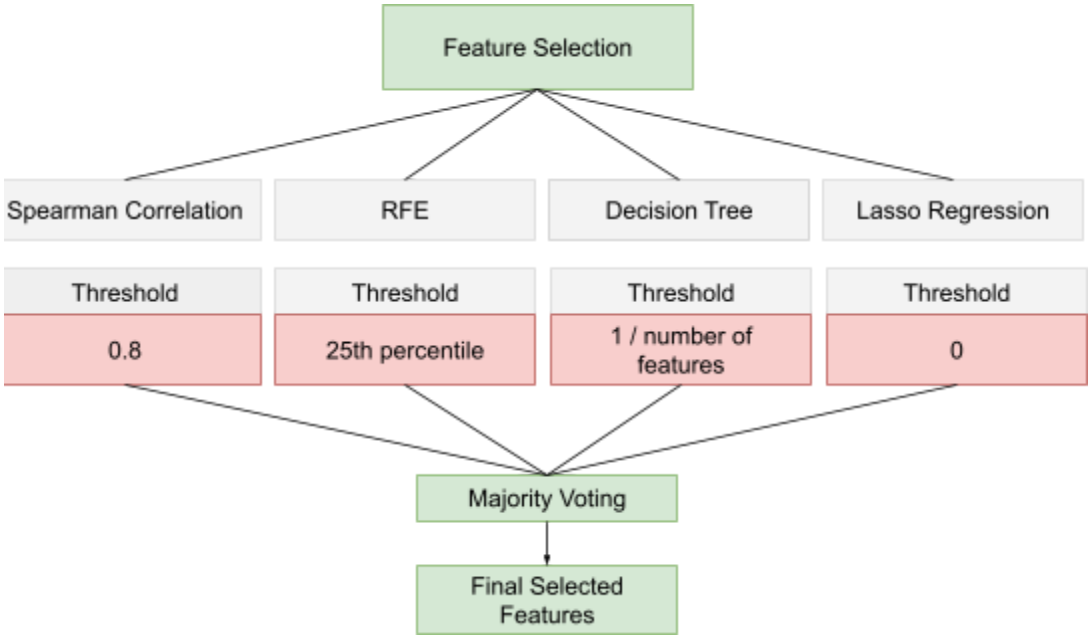

Figure 16 - Feature Selection Flowchart

For Spearman's Correlation, we excluded binary variables as they are not meaningful in correlation analysis. We set a threshold of 0.8 to identify highly correlated pairs, excluding the feature with lower variance.

In Recursive Feature Elimination (RFE), we used Logistic Regression and set the threshold to retain the top 75% of the most important features, based on their ranking from the elimination process.

For Decision Trees, we assessed feature importance using the model's built-in attribute, selecting features with importance greater than the average expected contribution (i.e., 1 divided by the total number of features).

With Lasso Regression, we selected features with non-zero coefficients, while less relevant features had their coefficients penalized and reduced to zero through regularization.

Finally, in Majority Voting, features that were selected by at least 50% of the methods were retained. This ensured that only features consistently identified as important across multiple techniques were included in the final set.

4.5. Model Training and Optimization

Recognizing the challenges of a multi-classification problem, we initially evaluated several algorithms, including Adaboost, Bagging, Decision Tree, Naive Bayes, Logistic Regression, Neural Network, Random Forest, and XGBoost (Text 1).

To streamline the process, we adopted a two-phase approach. First, all models were trained with default parameters, and their performance was evaluated using mainly the mean F1-score on the validation set. Simpler models generally required less training time, but many exhibited overfitting due to untuned parameters, as evidenced by the gap between training and validation F1-scores (Figure 17 and 18). Based on this initial evaluation, XGBoost, Neural Network, Random Forest, and Bagging were selected for further optimization.

In the second phase, we optimized the selected models using Randomized Search over defined parameter distributions (Table 6). Neural Network emerged as the best model, offering minimal overfitting and faster training, despite having a lower validation F1-score than others (Figure 19 and 20).

4.6. Final Model

After evaluating a diverse set of ML models, both in their default configurations and after hyperparameter optimization, the Neural Network was selected as the final model based on its consistent performance, lower overfitting, and reasonable computational cost. After optimization, it achieved a validation F1-score of 0.356, with a relatively small gap to the training F1-score (-0.114), indicating reduced overfitting compared to other models. Additionally, the low variability in the F1-scores across folds further highlights the model's stability and reliability.

Although the models XGBoost, Bagging and Random Forest achieved slightly higher F1-scores, they suffered from higher overfitting and significantly longer optimization times. The Neural Network provided a good balance between performance and efficiency, making it the most practical choice for this task.

The optimized Neural Network used the following parameters:

- Activation function: relu
- Hidden layer sizes: (9, 9, 9)
- Learning rate: constant
- Initial learning rate: 0.01
- Solver: adam

5. Open-Ended Section

The open-ended section of this project focuses on providing a practical, user-friendly solution for automating claim classification and enabling insightful analysis. We developed a web application (Figure 21) for the New York WCB that allows users to easily upload a CSV file containing new claims data. The data is processed through a pipeline similar to the one used during model training, using the same preprocessing steps and model objects created during the prediction phase for the test data. The processed data is then passed to the best-performing model to predict the *Claim Injury Type* for each claim.

The application also features an intuitive and interactive dashboard (Figure 22) designed to help users visualize the predictions effectively. Users can customize their view by selecting specific features, which trigger the display of various visualizations such as box plots for numeric (Figure 23) and date (Figure 24) features, stacked bar charts for binary features (Figure 25), and bar charts for categorical features (Figure 26). These visualizations allow users to analyze the characteristics of claims for each injury type, providing insights into trends and patterns that support informed decision-making.

A demo video (Bib. 11) is provided to guide users through the application, covering the steps for uploading data, generating predictions, and effectively using the dashboard.

Currently, the application is intended for local use. However, future plans include integrating the solution with the WCB's online claim submission system, enabling real-time claim classification and streamlined processing. This will reduce manual effort and improve the overall efficiency of the claims management process.

6. Conclusion

In this project, we developed an automated claim injury type classification system for the New York WCB, using ML techniques. Our approach involved extensive preprocessing, feature engineering, and applying various ML models to ensure the most effective solution. After careful evaluation, Neural Network emerged as the top-performing model. While the score may seem modest, it is a reasonable result given the complexity of the problem and the challenges posed by the dataset.

We adopted a robust methodology throughout, including Repeated Stratified K-fold Cross-Validation to assess model performance and ensure generalization. This rigorous approach, combined with techniques like SMOTE to address class imbalance, helped improve the model's consistency and reliability.

However, there were notable limitations. The dataset itself presented challenges, such as missing data, inconsistencies, and outliers, making preprocessing a critical step. Computational limitations

also impacted model training time, especially for more complex models. Furthermore, the open-ended application is currently designed to run locally, limiting its scalability. In the future, integrating the solution with the New York WCB's online system would allow for real-time predictions, improving efficiency and reducing manual intervention. Additionally, future work could explore the use of text mining techniques to better leverage the categorical features, enhancing model's predictive power.

Overall, this project successfully demonstrated the potential of ML for automating claim injury type classification, providing valuable insights and a user-friendly application. However, improvements in data handling, computational power, and integration with broader systems are necessary for further advancement.

7. Bibliography

Bib. 1 - NYS Workers Compensation Board - Home Page. (n.d.). Retrieved from www.wcb.ny.gov website: https://www.wcb.ny.gov/

Bib. 2 - Employee Claim. (2024). Retrieved from Ny.gov website: https://www.wcb.ny.gov/onlineforms/c3/C3Form.html

Bib. 3 - 38, G. (2024). GitHub - mgalao/machine-learning-project. Retrieved from GitHub website: https://github.com/mgalao/machine-learning-project

Bib. 4 - Workers' Compensation Board All Common Forms. (2022). Retrieved from Ny.gov website: https://www.wcb.ny.gov/content/main/forms/AllForms.jsp

Bib. 5 - New York State Workers' Compensation Board. (n.d.). Hours of Work for Minors. Retrieved from Department of Labor website: https://dol.ny.gov/hours-work-minors

Bib. 6 - ZIP Code. (2021, May 10). Retrieved from Wikipedia website: https://en.wikipedia.org/wiki/ZIP_Code

Bib. 7 - New York Zip Codes 2024. (2024). Retrieved from Worldpopulationreview.com website: https://worldpopulationreview.com/zips/new-york

Bib. 8 - Search NAICS Codes by Industry | NAICS Association. (2019). Retrieved from NAICS Association website: https://www.naics.com/search-naics-codes-by-industry/

Bib. 9 - *WCIO Legacy*. (n.d.). Retrieved from https://www.guarantysupport.com/wp-content/uploads/2024/02/WCIO-Legacy.pdf

Bib. 10 - Swamynathan, M. (2017). *Mastering Machine Learning with Python in Six Steps*. Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-2866-1

Bib. 11 - 38, G. (2024). Demonstration of Open Ended Section. Retrieved from OneDrive website:http://surl.li/nytywm

8. Annexes

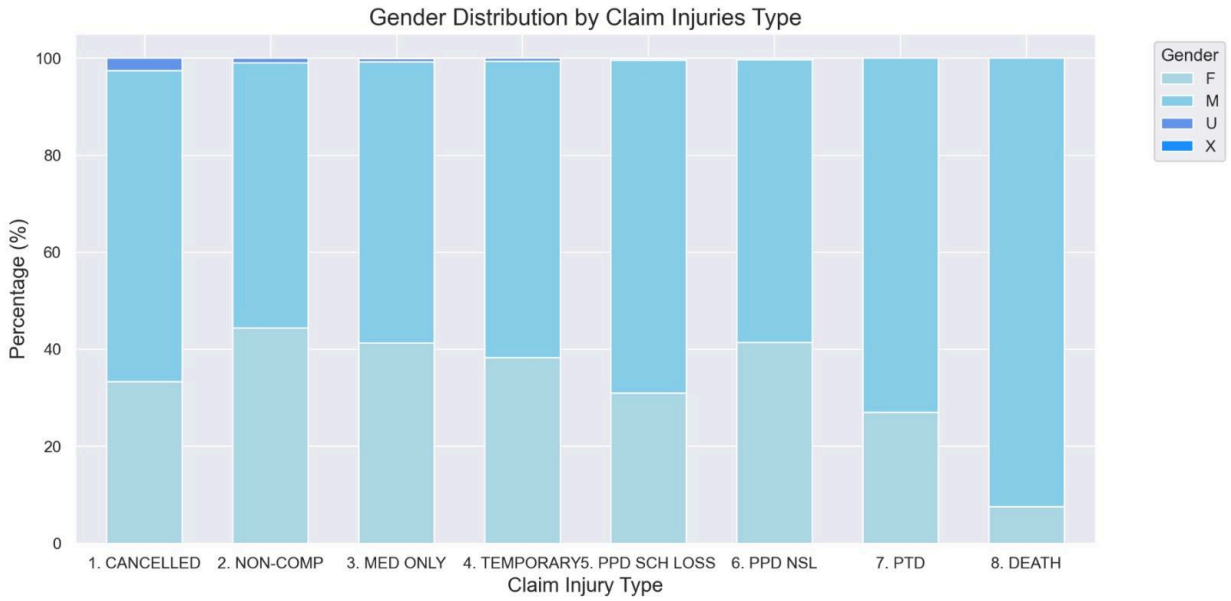

Figure 1 - *Gender* Distribution by *Claim Injury Type*

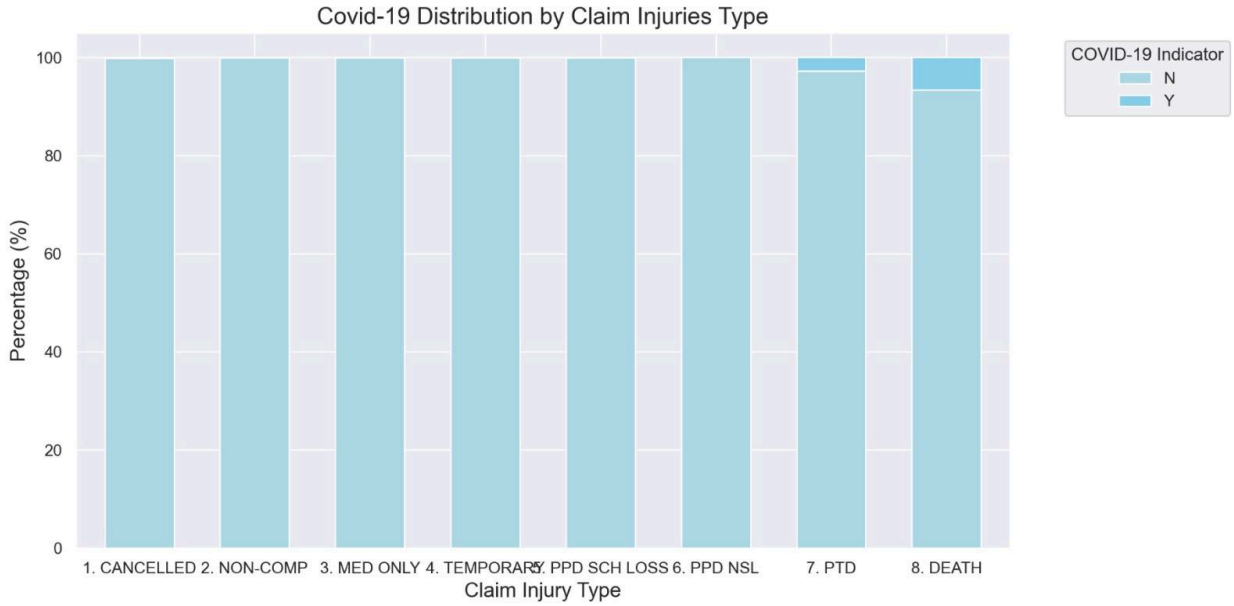

Figure 2 - *COVID-19 Indicator* Distribution by *Claim Injury Type*

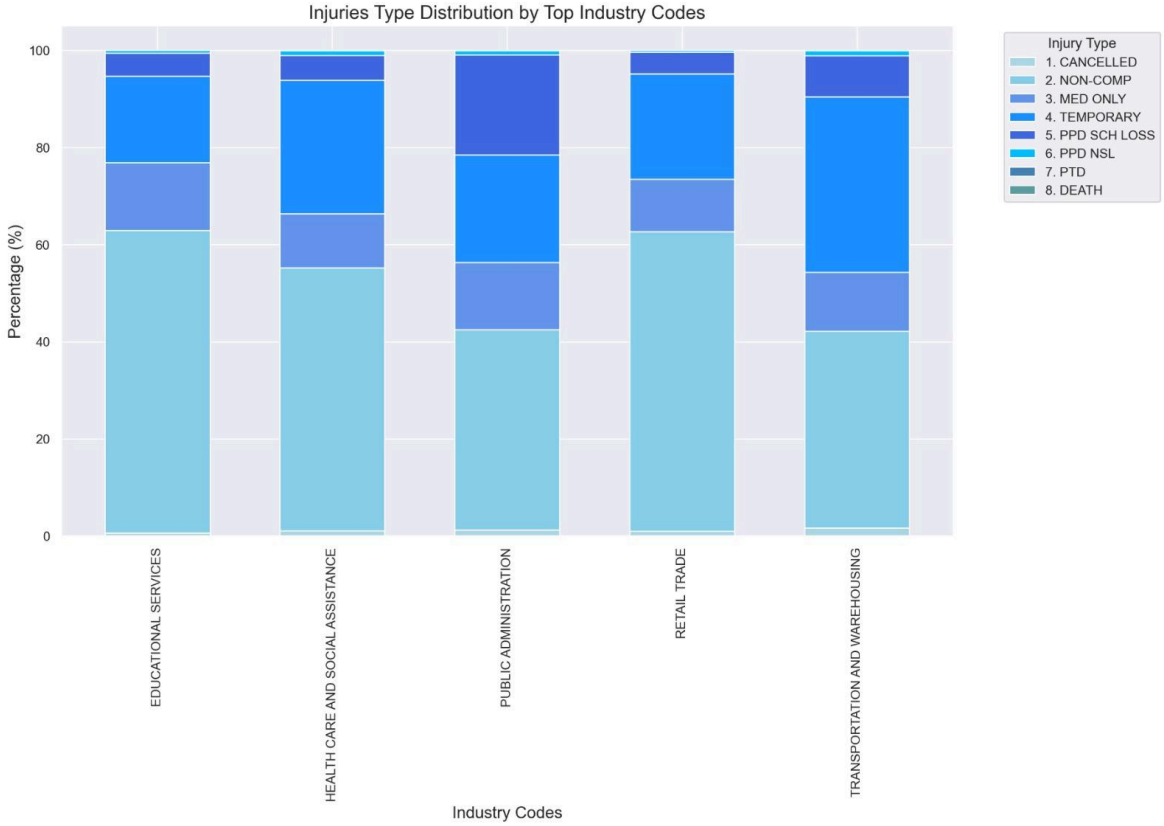

Figure 3 - *Claim Injury Type* Distribution by Top *Industry Code*'s

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Accident Date	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Age at Injury	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Alternative Dispute Resolution	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Assembly Date	19445	1096	2020-06-16	30	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Attorney/Representative	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Average Weekly Wage	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Birth Year	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
C-2 Date	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
C-3 Date	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Carrier Name	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Carrier Type	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Claim Identifier	19445.0	NaN	NaN	NaN	55174917 7.15	259220840.07	100031438.0	328429921.0	554012330.0	777096910.0	999891667.0
Claim Injury Type	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
County of Injury	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
COVID-19 Indicator	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
District Name	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
First Hearing Date	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Gender	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
IME-4 Count	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Industry Code	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Industry Code Description	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Medical Fee Region	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
OIICS Nature of Injury Description	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
WCIO Cause of Injury Code	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
WCIO Cause of Injury Description	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
WCIO Nature of Injury Code	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
WCIO Nature of Injury Description	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
WCIO Part Of Body Code	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
WCIO Part Of Body Description	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Zip Code	0	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Number of Dependents	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Figure 4 - Summary Statistics for 9-length *Claim Identifier* Records

```
OIICS Nature of Injury Description    100.00
IME-4 Count                            76.86
First Hearing Date                     73.73
C-3 Date                               67.38
Birth Year                              5.07
Zip Code                                4.99
Average Weekly Wage                     4.99
WCIO Part Of Body Code                  2.98
WCIO Part Of Body Description           2.98
WCIO Nature of Injury Code              2.73
WCIO Nature of Injury Description       2.73
WCIO Cause of Injury Code               2.72
WCIO Cause of Injury Description        2.72
C-2 Date                                2.54
Industry Code Description               1.73
Industry Code                           1.73
Accident Date                           0.64
dtype: float64
```

Figure 5 - Percentage of Missing Values of All Features

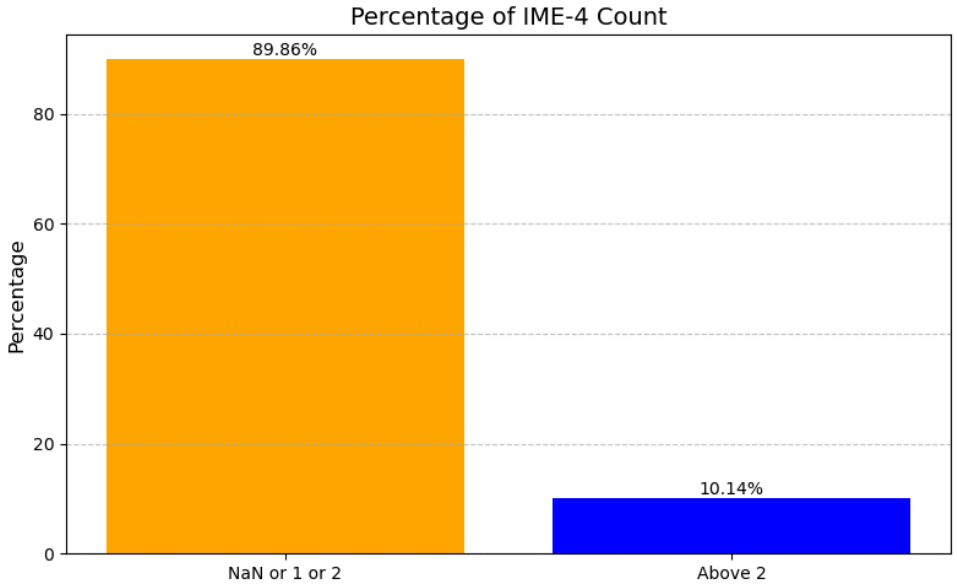

Figure 6 - Percentage Frequency Distribution of *IME-4 Count*

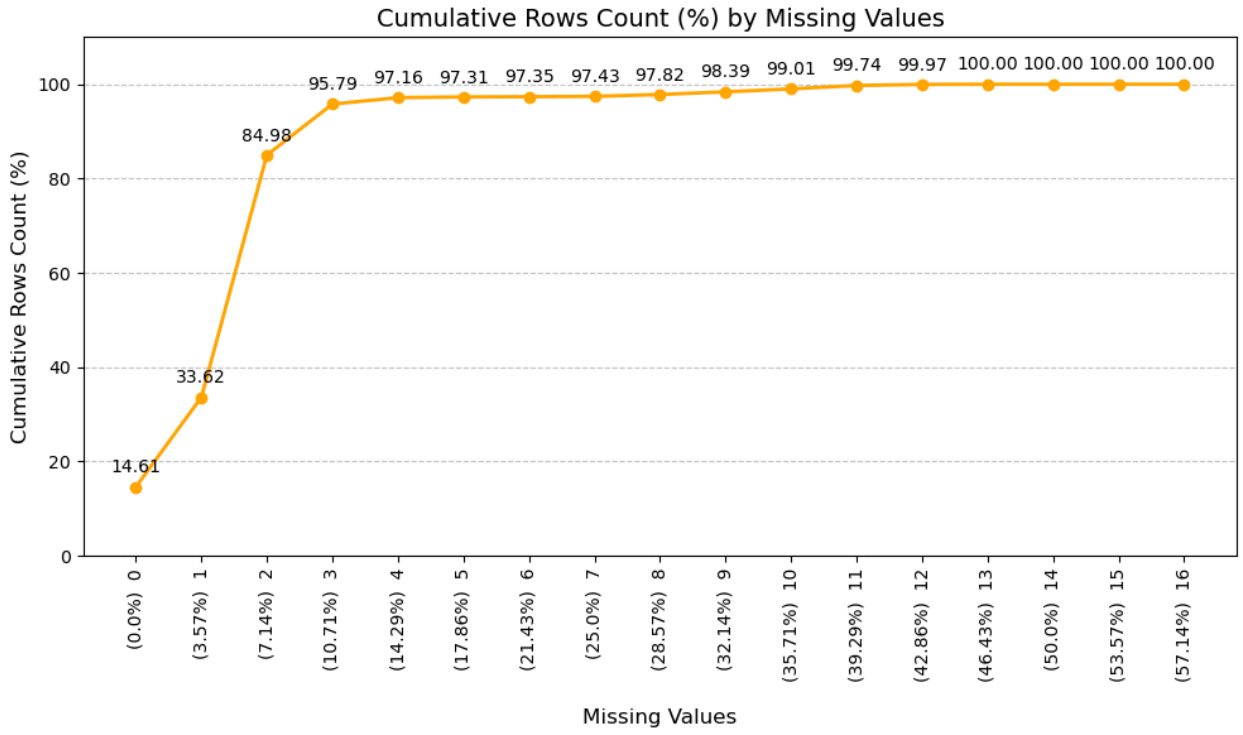

Figure 7 - Cumulative Rows Count (%) by Number of Missing Values

```
Assembly Date           0.00
Accident Date           0.60
C-2 Date                2.28
C-3 Date               67.36
First Hearing Date     73.66
dtype: float64
```

Figure 8 - Percentage of Missing Values of Date Features

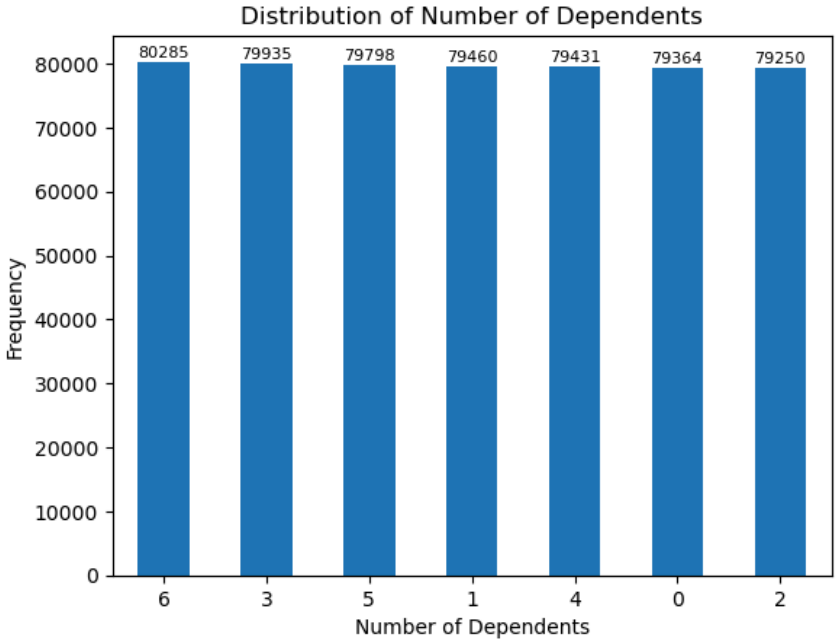

Figure 9 - Frequency Distribution of *Number of Dependents*

```
Claim Identifier
5399802    L1N 5
5404203    T1B0P
5407348    L6Y 1
5412514    JMDMR
5413081    N2P 1
           ...
6143240    V1M2B
6147711    L2P0A
6150118    L6L 1
6152050    L1A 1
6164424    G9B0Y
Name: Zip Code, Length: 14934, dtype: object
```

Figure 10 - Examples of 5-digit ZIP Codes

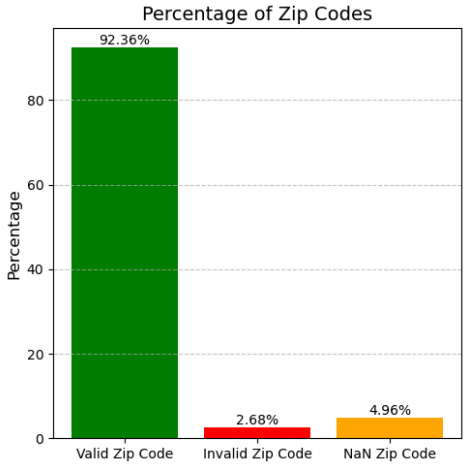

Figure 11 - Percentage Frequency Distribution of ZIP Codes

```
Claim Injury Type
2. NON-COMP          50.85
4. TEMPORARY         26.01
3. MED ONLY          12.09
5. PPD SCH LOSS       8.47
1. CANCELLED          1.73
6. PPD NSL            0.75
8. DEATH              0.08
7. PTD                0.02
Name: proportion, dtype: float64
```

Figure 13 - Percentage Frequency Distribution of *Claim Injury Type* (Before Cross-Validation)

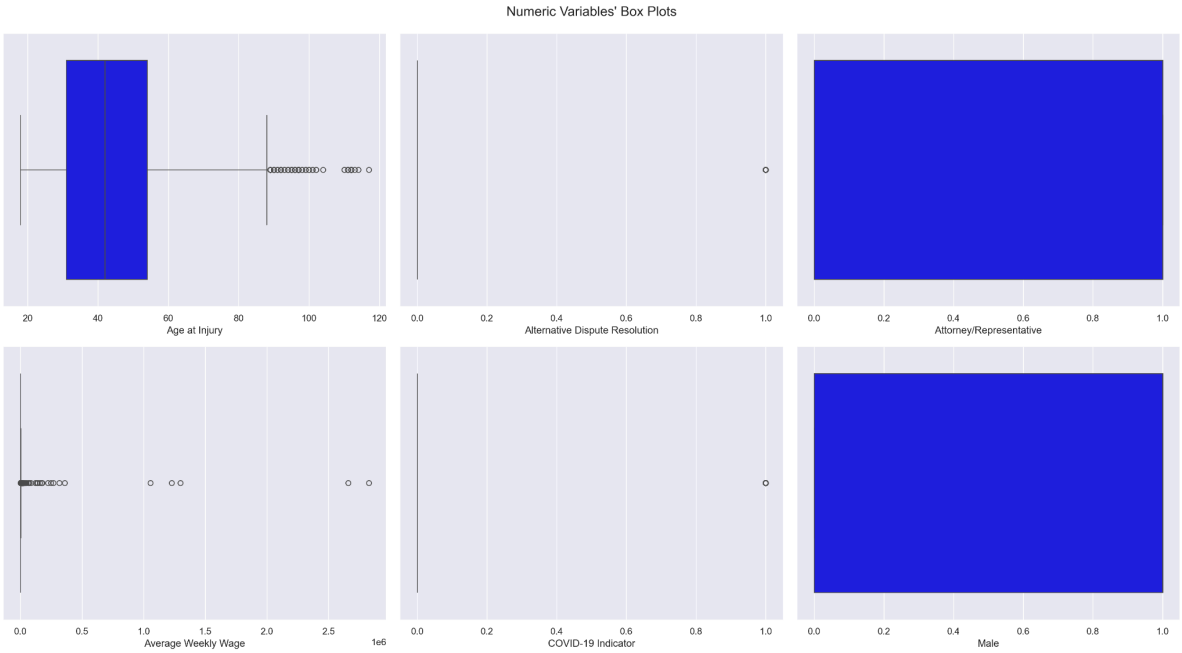

Figure 14 - Numeric Variables' Box Plots

Table 4 - Thresholds of Manual Filtering

Feature	Threshold values	Why?
Age at Injury	*100*	We had already removed the minor individuals, we only have individuals over 18 years old and decided to set a maximum age of 100; while such ages are rare, they are still plausible, whereas values like 117 are unreasonable
Average Weekly Wage	*8000*	We accepted a minimum value of 0, to account for cases where an individual could be a volunteer, may not have worked during the period or had a wage of zero for other reasons (e.g., leave of absence or unemployment). To determine the threshold, we analyzed the boxplot and 8000 was the most suitable value

XGBoost (Extreme Gradient Boosting) is an efficient and powerful machine learning algorithm based on gradient boosting. It builds an ensemble of decision trees sequentially, where each tree corrects the errors of the previous one. XGBoost is known for its speed, scalability, and high performance, often outperforming other models in tasks like classification and regression. It uses regularization techniques to prevent overfitting and handles missing data well. Additionally, XGBoost can be easily tuned for optimal performance and is widely used in machine learning competitions and practical applications.

Text 1 - XGBoost Model Explanation

	F1 Val	F1 Train	F1 Val-Train	Time (s)	Iterations	Best Params
XGBoost	0.376 +/- 0.01	0.668 +/- 0.01	-0.292 +/- 0.01	14.215 +/- 0.30	0.0 +/- 0.0	None
NeuralNetworks	0.362 +/- 0.00	0.552 +/- 0.01	-0.191 +/- 0.00	110.674 +/- 11.10	199.8 +/- 0.4	None
RandomForest	0.353 +/- 0.01	0.918 +/- 0.02	-0.565 +/- 0.02	43.501 +/- 2.23	0.0 +/- 0.0	None
Bagging	0.343 +/- 0.01	0.908 +/- 0.02	-0.564 +/- 0.02	11.462 +/- 0.56	0.0 +/- 0.0	None
LogisticRegression	0.339 +/- 0.00	0.404 +/- 0.01	-0.065 +/- 0.01	4.304 +/- 0.19	100.0 +/- 0.0	None
DecisionTree	0.327 +/- 0.01	0.918 +/- 0.02	-0.591 +/- 0.02	2.444 +/- 0.43	0.0 +/- 0.0	None
GaussianNB	0.258 +/- 0.00	0.311 +/- 0.01	-0.053 +/- 0.01	0.546 +/- 0.18	0.0 +/- 0.0	None
Adaboost	0.216 +/- 0.03	0.340 +/- 0.04	-0.124 +/- 0.02	10.707 +/- 0.84	0.0 +/- 0.0	None

Figure 17 - Default Models' Performance Metrics

Figure 18 - Bar Chart of Default Models' Performance (Validation F1-score)

	F1 Val	F1 Train	F1 Val-Train	Time (s)	Iterations	Best Params
XGBoost	0.383 +/- 0.01	0.645 +/- 0.04	-0.262 +/- 0.04	676.838 +/- 198.33	0.0 +/- 0.0	{'learning_rate': 0.05, 'max_depth': 7.0, 'min...
Bagging	0.376 +/- 0.01	0.682 +/- 0.09	-0.306 +/- 0.08	543.323 +/- 89.14	0.0 +/- 0.0	{'estimator': DecisionTreeClassifier(min_sampl...
RandomForest	0.376 +/- 0.01	0.704 +/- 0.06	-0.327 +/- 0.07	1255.395 +/- 174.69	0.0 +/- 0.0	{'class_weight': None, 'max_depth': 10.0, 'max...
NeuralNetworks	0.356 +/- 0.00	0.470 +/- 0.02	-0.114 +/- 0.02	211.367 +/- 32.45	122.0 +/- 60.2	{'activation': 'relu', 'hidden_layer_sizes': (...

Figure 19 - Optimized Models' Performance Metrics

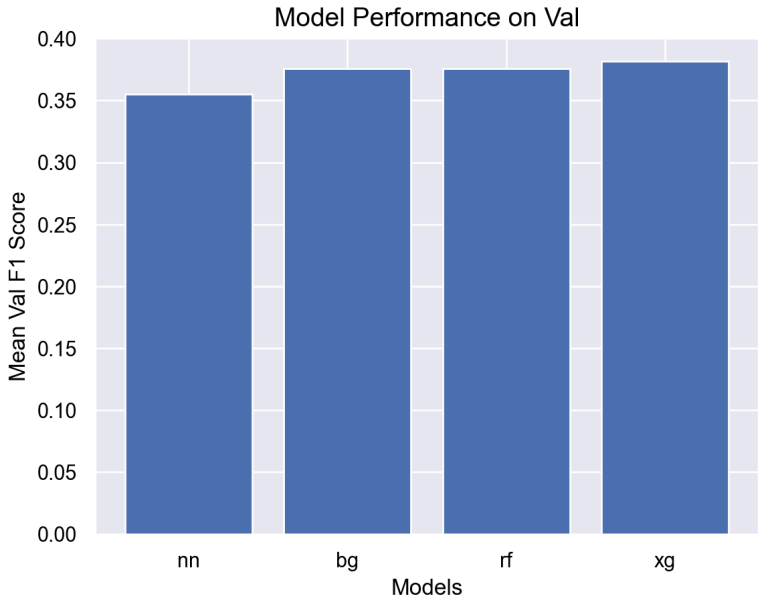

Figure 20 - Bar Chart of Optimized Models' Performance (Validation F1 Validation)

Table 6 - Tested Hyperparameters for each Model

Model Name	Parameters to test
XGBoost	"learning_rate": [0.01, 0.05, 0.1, 0.2], Best: 0.05 "n_estimators": [50, 100, 150, 200], Best: 100 "max_depth": [3, 5, 8], Best: 7 "min_split_loss": [0, 0.1, 0.3], Best: 0.1 "subsample": [0.7, 0.8, 0.9, 1.0], Best: 0.6
Random Forest	"n_estimators": [50, 100, 200, 300], Best: 50 "max_depth": [None, 3, 5, 10, 15], Best: 10 "min_samples_split": [2, 5, 10], Best: 5 "class_weight": ["balanced", None], Best: None "max_samples":[None,0.6,0.8], Best: 0.8 "min_samples_leaf": [1, 2, 5], Best: 1 "max_features": ['sqrt',0.8, 'log2', None], Best: 0.8
Bagging	"n_estimators": [10, 50, 100, 200], Best: 200 "max_samples": [0.5, 0.7, 1.0], Best: 0.7 "max_features": [0.5, 0.7, 1.0], Best: 1 estimator: DecisionTreeClassifier(" max_depth"=d, "min_samples_split"=split, " min_samples_leaf"=leaf, "max_features"=f, "class_weight"=cw)

Model Name	Parameters to test
	for d in [3, 5, 10, None] for split in [2, 5, 10] for leaf in [1, 2, 5] for f in [None, 0.8, 'sqrt', 'log2'] for cw in ["balanced", None]
Neural Network	"hidden_layer_sizes": [(7, 7), (9, 9), (11, 11), (7, 7, 7), (9, 9, 9), (11, 11, 11)], Best: (9,9,9) "activation": ["relu"], Best: relu "solver": ["adam"], Best: adam "learning_rate": ["constant", "adaptive"], Best: constant "learning_rate_init": [0.001, 0.01], Best: 0.01

Figure 21 - Web Application Starting Page

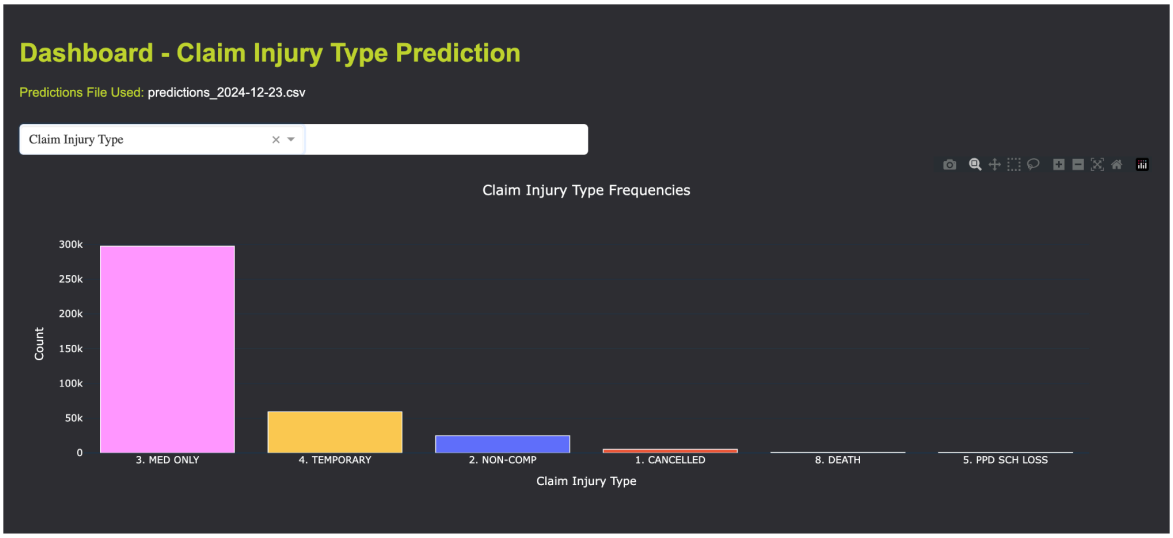

Figure 22 - Bar Chart of *Claim Injury Type*

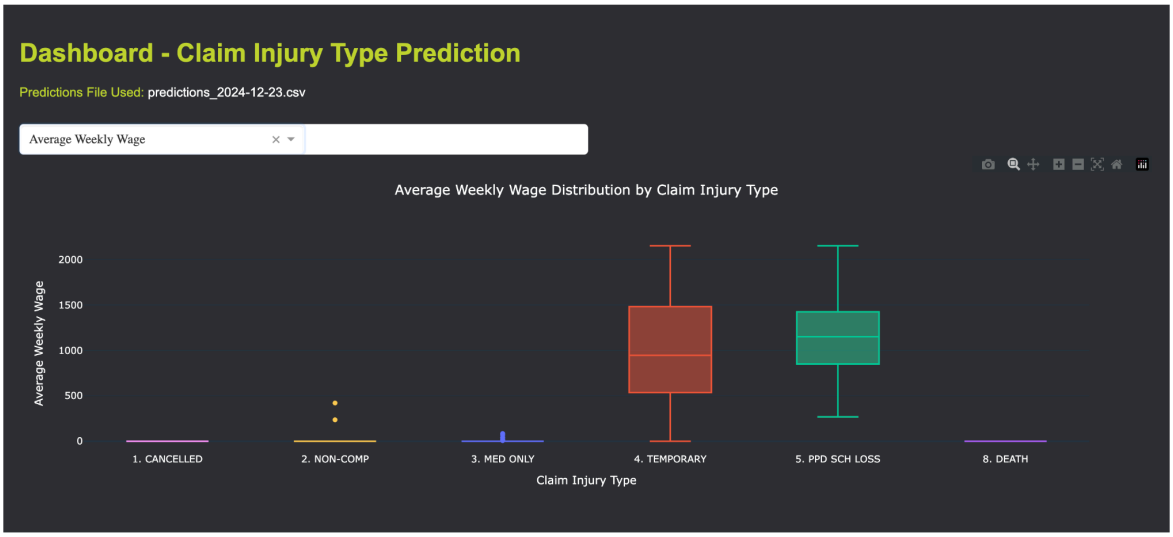

Figure 23 - Box Plot of *Average Wekly Wage* by *Claim Injury Type*

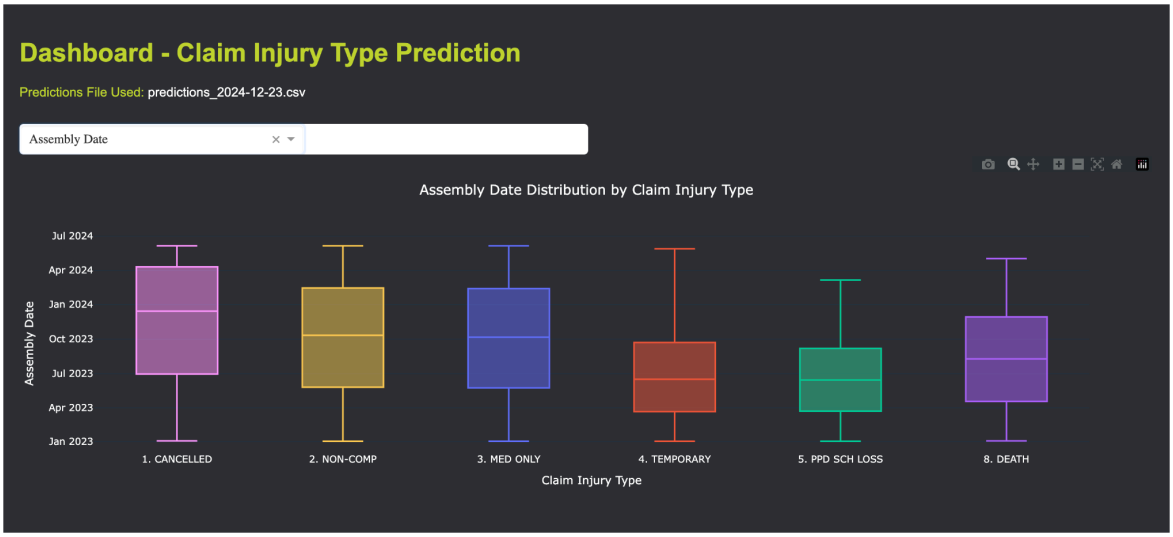

Figure 24 - Box Plot of *Assembly Date* by *Claim Injury Type*

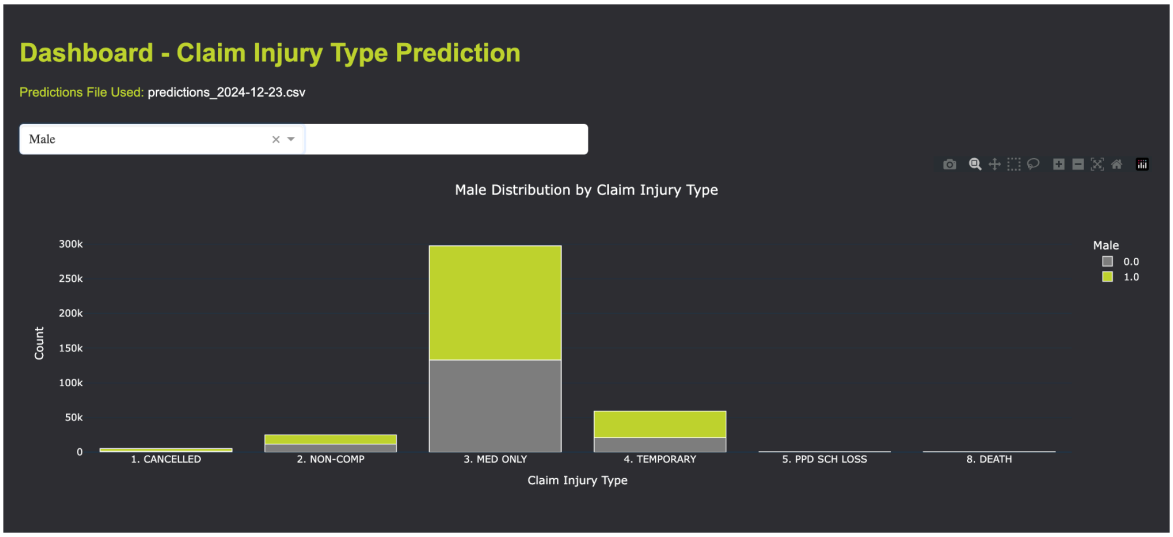

Figure 25 - Stacked Bar Chart of *Male* by *Claim Injury Type*

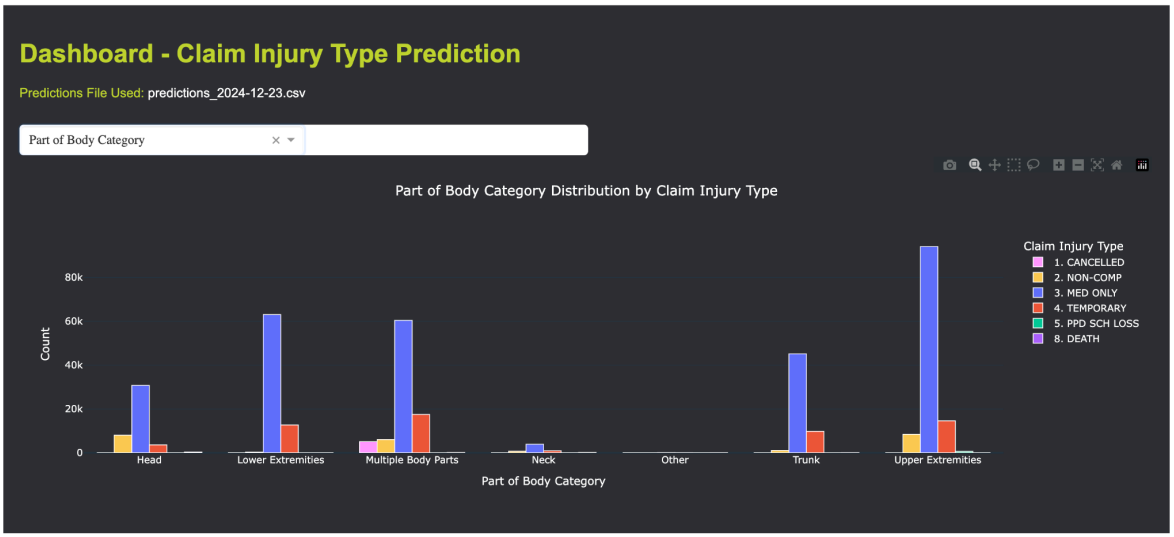

Figure 26 - Bar Chart of *Part of Body Category* by *Claim Injury Type*