TO GRANT OR NOT TO GRANT: DECIDING ON COMPENSATION BENEFITS

Group 10 Machine Learning Project

December 2024

Group members

Barata Diogo - 20240258

Deschanel Chloé - 20240693

Leandro Lara - 20211632

Parracho Érica - 20240583

Walihullah Mohammad - 20230768

Abstract

The application of machine learning has transformed many industries like the insurance one. It offers efficient tools for handling tasks like predicting and analyzing claims. Our project explores machine learning in automating and enhancing decision-making processes for the New York Workers' Compensation Board (WCB) by predicting workplace injury claim types. By using historical claims data from 2020 to 2022, we focus on developing a strong multiclass classification model.

Our main challenge was navigating the issue of having a highly imbalanced dataset. To address this, we tried using a re-sampling technique, hyperparameter tuning, and predicting a variable to include in our model. Despite achieving a decent score on our validation and our test data, none of these techniques managed to improve predicting minority classes efficiently. In addition, we wanted to provide the WCB with an app to streamline predicting claim injury type. This proved to be a time-consuming task and errors are still present in the deployment part.

Finally, our main contribution is to demonstrate the potential of machine learning in streamlining processes, like processing claim injury types for WCB. It also emphasises the need for improved techniques to handle highly imbalanced datasets and ameliorate predictions for minority claim types. Our project is publicly available on github here.

Table of Contents

I. Introduction

The application of machine learning has transformed many industries like the insurance one. It offers efficient tools for handling tasks like predicting and analyzing claims. This can enable companies to streamline their processes, increase efficiency, and make informed decisions when dealing with large volumes of claims (Hanafy and Ming, 2022). Existing research on insurance claim prediction seems to mainly focus on binary classification (e.g. Abdelhadi et al., 2020; Baran and Rola, 2022; Hanafy and Ming, 2020; Saikia et al., 2024), but there are some studies that explore multi-class classification.

For instance, the work of Mustika et al. (2019) used a dataset with a large number of missing values and predicted the risk levels for Prudential Life Insurance applicants. XGBoost had the best performance among the models tested, yielding a score of 0.60730 when using the Quadratic Weighted Kappa metric. Similarly, Alamir et al. (2021) proposed a predictive model for classifying motor insurance claim status. The Random Forest model slightly outperformed the Support Vector Machine one, with an average accuracy of 98.36% and 98.17%, respectively. The two models also performed well when evaluating other metrics like F-Score, where Random Forest got 94.91% and 95.37% for the SVM. In another study, Liu et al. (2014) categorized claim counts to predict auto insurance claim frequencies. The Multi-Class AdaBoost Tree performed better compared to others, with a predictive accuracy of 0.8462. Table 1 provides more details on the studies mentioned.

Building on the insights from research on the insurance industry, our project focuses on helping the New York Workers' Compensation Board (WCB) automate the way they handle claims and improve their decision making process. We aim to develop and optimize a predictive model that can identify and classify different types of workplace injury claims. Using historical data from 2020 to 2022, the project addresses the challenge of working with an imbalanced dataset. Figure 1 demonstrates the overall workflow of our project for a better overview.

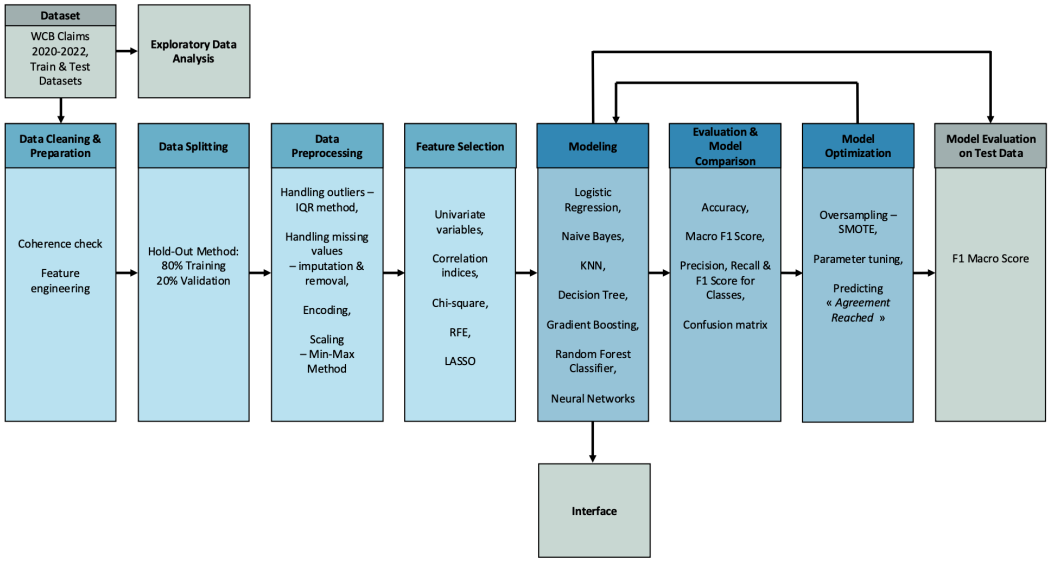

Figure 1: Project workflow

II. Data Exploration and Preprocessing

A) Exploratory Data Analysis

For this project, we have two datasets, one for training our model and the other for its assessment and testing. We began by performing an exploratory data analysis on the training set to gain a better understanding of the data, and to identify issues and incoherences.

Firstly, the training data set has 593,471 entries and 32 unique features. We set the index to *Claim Identifier* and we converted date related columns to date-time type for easier analysis and consistency. When checking for duplicates, we identified 18,349 duplicated entries. These were removed to maintain the integrity of the dataset. Additionally, one duplicated *Claim Identifier* was found and dropped, as it appeared to be an error, given that each claim identifier should be unique. Except for *Assembly Date*, all variables contained missing values. Fourteen of them had the same number of missing values (i.e. 1,096 missing values), and *OIICS Nature of Injury Description* had zero entries. When looking at the statistical analysis for numerical variables, we noticed evident anomalies in variables like *Age at Injury* (range from 0 to 117), *Birth Year* (range from 0 to 2018), and *Accident Date* (range from 1961 to 2023).

Secondly, we divided our variables into numerical and categorical to visually explore them. Starting with the categorical features, the analysis revealed significant insights into the target variables, *Claim Injury Type*. It is evident that it is highly imbalanced. Non-Compensable cases account for the majority of claims (291,078), while others, like Permanent Total Disability and Death fewer cases, 97 and 470 claims respectively [Fig.2]. We then checked categorical variables in relation with our target variable. We found that most claims are filed by males, with the majority being Non-Compensable and Temporary Disability. These two claim types were also the most reported amongst females [Fig.3]. Similarly, Non-Compensable, followed by Temporary Disability are the two main claim types across other variables when checking for target distribution (e.g. *County of Injury, District Name, Medical Fee Region, Industry Code, Carrier Name,* …). However, we did notice that most claims with an attorney or representative were classified as Temporary Disability, rather than Non-Compensable [Fig.4]. A total of 547,238 claims did not reach an agreement, while 26,787 did, with a significant majority (24,535) involving an attorney or representative, suggesting that legal representation may facilitate compensation [Fig.5] [Table 2].

Finally, for numerical variables, we started by plotting histograms and boxplots to check the distribution of our date [Fig.6]. We can confirm that almost all numerical variables contained outliers, which is consistent with the findings from the statistical analysis (e.g. *Age at Injury*). *Hearing at First Date* seemed to be the only variable that had a distribution closest to a normal one. We also checked for relationships between variables using scatterplots, and we found that individuals born later in time tend to have higher average weekly wages and there is a strong negative correlation between *Age at Injury* and *Birth Year*, which is expected as older individuals were born earlier [Fig.7]. Additionally, we explored numerical variables in relation to our target and found that for more severe injury types, the average age at injury is between 45 and 54 years old, while the others are between 38 and 43 years old. Similarly, the violin plot reveals that more severe injury types have higher *Age at Injury* medians, compared to Non-Compensable and Cancelled cases having the lowest medians [Fig.8]. We also found that

independent medical examinations (IME-4) may be important to determine more severe injury type, as Permanent Total Disability and Permanent Partial Disability (Non-Scheduled Loss) claims have more independent medical examinations on average, with 6 counts and 4 counts respectively.

This exploration phase allowed us to identify data issues, such as duplicates, missing values, and inconsistencies in some features, which will be addressed during preprocessing to ensure data quality for further analysis.

B) Preprocessing

The preprocessing phase aimed to remove anomalies and to solve inconsistencies discovered upon our initial analysis, while also preparing the data for the model building.

i) Coherence Check

From the EDA, we can assume that our dataset is based in the USA, which will guide our decisions on how to best handle incoherences. To start, we dealt with *Age at Injury* and *Birth Year*, and with the available information, we tried to calculate one of these two variables based on the other if it was missing. Additionally, the Fair Labor Standards Act (FLSA) legislates a minimum age of 14 for employment (Wage and Hour Division, 2016). Thus, we replaced ages below 14 or above 90 with "NaN", and we ensured that no individual had a birth year beyond 2008, as this would indicate an underage worker. We also looked for inconsistencies in the following variables:

- *Carrier Name* contained many inconsistencies, such as the use of different punctuations and abbreviations for the same carrier. We attempted to standardize these names and managed to reduce *Carrier Name* from 2,046 to 1,560 distinct values.
- *Zip Code* contained invalid entries, such as not having 5 digits, which is not valid in the US (IBM, 2023), or being marked as “UNKNO”, “0000”, or “9999”. These entries are replaced with NaNs instead.
- For date related variables, we ensured that they were chronologically consistent. For instance, there were some cases where the *First Hearing* or *Assembly Date* occurred before the *Accident Date*. In those cases, we replaced these dates with the oldest available date from the individual.
- For code columns, we did not find any inconsistencies when we checked cases where there were multiple codes associated with the same description, or vice versa. Additionally, *WCIO Part of Body Code* had a negative code, -9, which is not present in the list of codes (UDS, 2019, section 16-6). We replaced these instances with the most similar code description.

ii) Feature Engineering

Our dataset, contained many variables that were difficult to interpret, and so, to obtain more information and enhance model performance, we created the following features:

- *C3-Date*, *C2-Date* and *First Hearing Date* were converted into binary features, representing whether the event happened or not. This also allowed us to deal with missing values.

- To get a clearer representation of age, we calculated the current *Age* by subtracting the current year (i.e. assumed to be 2024) with the *Birth Year*.
- We calculated the *Days from Assembly Date*, by subtracting the current year with the year of assembly.
- By extracting the year from *Accident Date*, it allowed us to simplify the feature and enhance understandability.
- To address skewness in highly variable features like *Wage* and *IME-4 Count*, we applied the log transformations. This reduces the range of values, making the data closer to a normal distribution and improving model performance.

Finally, we dropped the original columns that were used to create these new features, as they no longer provided useful information. Additionally, we removed the code description columns, as they duplicated information already present in the code columns.

iii) Data Splitting

Using the Hold-out-Method, the training set was split into features and the target variable (i.e. *Claim Injury Type*), where 80% of data was used for training and 20% for validation. Given the class imbalance, we used stratified sampling to ensure the distribution of the target variable was maintained across both the training and validation datasets. Before splitting the data, we removed rows where the target was missing, since we could not stratify otherwise. Additionally, these contained missing values in most features, limiting the information and confirming our choice to drop them.

iv) Outliers

Outliers were identified through the Interquartile Range (IQR) method, which measures the spread between the 25th and 75th percentiles of the data. These were the variables and their percentage of outliers detected: *Age at Injury* (0.01%), *Age* (0.06%), *IME-4 Count* (0.06%), and *Accident Year* (0.85%). To handle these outliers, we replaced them with the upper or lower bound of each IQR.

v) Missing Values

When dealing with missing values, *OIICS Nature of Injury Description* contained over 80% missing values, and so, was dropped, as it did not provide enough useful information to justify imputation. Our strategy to deal with missing values is detailed in Table 3.

Features	Imputation Method	Description
Categorical Features (i.e. codes and *County of Injury*)	Mode	Categorical features do not have a natural ordering, therefore, imputing with the mean or median is sub-optimal.
Age	Mean	The distribution of age columns is asymptotically normal, thus, imputing with the mean provides a good central estimate.
Age at Injury	Calculations & Mean	Estimate the *Age at Injury* based on *Age* and *Days from Assembly Date*, and then impute with the mean.
Accident Year	Median	Use the median to mitigate against extreme values, as it is not sensitive, unlike the mean.
Average Weekly Wage	Industry-Specific Mean	To reflect the variation in wages across different industries and provide a more appropriate estimate, missing values are filled with industry-specific mean.
IME-4 Count	Constant Value - 0	A missing value here suggests no IME-4 forms were submitted, which is the same as 0.

Table 3: Imputation Methods for Missing Values

vi) Encoding Categorical Variables

We used Ordinal Encoding for categorical variables to transform them into numeric values that models can understand. Despite not having an inherent order in these variables, we chose ordinal encoding for its simplicity and effectiveness, as some features had a huge number of categories. It is important to mention that this method also converted some variables into binary. After encoding, we ensured that any new categories in the validation set were replaced with the mode from the training set to prevent data leakage.

vii) Scaling the Data

To ensure all features are on the same scale and avoid issues with scale-sensitive models, we applied Min-Max Scaling to all features. This scaling method is ideal because it does not assume a normal distribution of the data and adjusts the data to a range between [0, 1], making it easier for algorithms to interpret and model relationships between variables.

III. Multiclass Classification

A) Feature Selection

Feature selection plays a crucial role in ensuring that we get the most relevant information from our data, and to guarantee good model performance and interpretability. In this section, we applied filter, embedded, and wrapper methods according to the characteristics of each variable. Details of which variables were dropped on each method are available in Table 4.

We first started assessing the variance of all features to see if there were any univariate variables and found that *WCB Decision* has the same values across all observations. So we decided to drop it as it

is not informative for the model. After that, the tests were divided by variable types, and we established that for a variable to be dropped they have to be considered irrelevant by all applicable methods.

Starting with categorical variables, we decided to apply the chi-squared test which measures if there is a statistically significant relationship with the target. For this part, we used a significance level of 0.05, and the features with a p-value bigger than this would be considered irrelevant.

For numerical features, we first assessed the relationship between them using Spearman correlation [Fig.9], as it does not assume a specific distribution for our data. We consider redundant variables the ones with a correlation bigger or equal to absolute 0.80. There are 2 pairs of correlated variables (*Age* and *Age at Injury*, *Accident Year* and *Days_from_Assembly Date*). In this case, the ones dropped will be the ones that perform worst on the rest of the methods.

Moving on to wrapper methods considering numerical and binary features, we decided to use Recursive Feature Elimination with logistic regression as the base estimator, which is computationally efficient but can also maintain good performance on classification problems. We evaluated the ROC-AUC score on the validation data to determine the best subset of variables, which shows the model's capability to distinguish between classes. As we can see in Figure 10, the score gradually increases until 10 features, when the value stabilizes, suggesting 10 is the optimal number.

Additionally, the embedded method Lasso was also used for numerical and binary, and we decided to discard the ones with a score equal to 0. This meant that these variables didn't have a contribution to the prediction of the target on this regression.

After combining the results and considering our initially defined criteria, our final decision was that ***Number of Dependents***, ***Age at Injury***, and ***Has_Dependents*** would be discarded. We were only left to decide on the second pair of correlated features, we dropped ***Days from Assembly*** as it had a worse performance on Lasso.

B) Modelling

To address our multiclass classification problem, we tried different models, including simpler ones but also bagging and boosting algorithms. In our training process, we tested different models that suited our scenario, then assessed their performance using a hold-out method and selected the four best models, for which we used optimization techniques to refine them and select the best one.

The models we decided to test were: Logistic Regression, Gaussian Naive Bayes, Decision Tree, K-Nearest Neighbors Classifier, Gradient Boosting, Random Forest, and Neural Network. In all of them but Gradient Boosting, we used the sklearn implementation. For Gradient Boosting, we opted to use XGBoost (2022), an optimized library that allows for parallelization and also enables L1 and L2 regularization.

Given our highly unbalanced target, our main model evaluation metric was the Macro F1-Score, which takes into account precision and recall, and assigns equal weight across all classes. This ensures that the models are not ignoring the minority classes. In case of a similar score on validation, we prioritized a model that demonstrated less overfitting. To better understand our models, we also examined accuracy for a general view of performance, the metrics by class, and the confusion matrix to try to understand where misclassification occurred. In the event of Macro F1-Score ties, we looked at the precision of the minority classes to select the best models.

i) Performance Assessment

Before selecting the models to optimize, we looked at their performance. Most models had good accuracy, but when we examined the Macro F1-Score, the results were not as good, highlighting that most models were only performing well on classes with bigger support. Considering our defined criteria, in Table 5 we can see that the performance between most models doesn't differ significantly. However, we can clearly define KNN classifiers, Neural Networks, Random Forest, and Gradient Boosting as being better. We can also see that tree-based models have high overfitting.

Model	Macro F1 Score Train	Macro F1 Score Val
Logistic Regression	0,338	0,341
Naive Bayes	0,326	0,324
Decision Tree	0,999	0,371
K-Nearest Neighbors	0,480	0,396
Gradient Boosting	0,634	0,444
Random Forest	0,999	0,401
Neural Network	0,440	0,434

Table 5: Macro F1-Score comparison between tested models

When we look at the F1 metric by class [Table 6], we can see that Gradient Boosting outperforms other models across most classes and that the least represented classes have a negative significant impact on the scores of all models. It is important to note that even though Naive Bayes has the worst F1-score, it is the only model able to predict some correct instances in the minority class (7.PTD – Permanent Total Disability).

ii) Optimization

In this section, we explore our optimization approach for our four best models, which include an oversampling technique and hyperparameter tuning.

Initially, we tried to address our class imbalance by using SMOTE, Synthetic Minority Oversampling Technique, which generates synthetic data points between instances of the minority classes (Chawla et al., 2002). Instances were generated until all classes were the same size as the majority class, and then we reassessed the Macro F1-Score [Table 7]. Looking at class specific scores for Gradient Boosting [Table 8], we can see that using SMOTE slightly ameliorated predictions for minority classes like PPD NSL, PTD and DEATH. In addition, using SMOTE improved significantly performance on the validations set for Gradient Boosting and Random Forest, but we decided not to use it as it resulted in more overfitting.

Our next step involved tuning the 4 best models using Randomized Search for Random Forest and Gradient Boosting, to explore a wider parameter space while maintaining computational efficiency, and Grid Search for KNN and neural networks. To avoid data leakage, we predefined a split, by combining the training and validation sets into a single dataset, with a separate fold to differentiate them. This served as the cross-validation splitting strategy. For tuning the models, we tested different parameters

with a particular focus on parameters that address imbalanced class distributions, such as 'weight' on KNN and the 'class_weight' on random forest. We also focused on parameters that reduce overfitting, such as 'max_depth' for both random forest and gradient boosting, and 'gamma', 'colsample_bytree' and 'min_child weight' for gradient boosting.

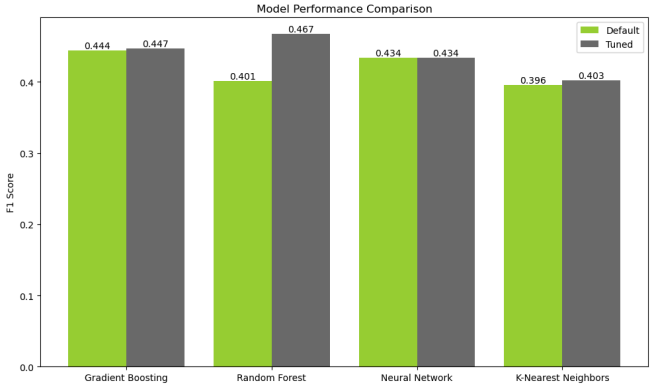

Fig.11: Comparison of Macro F1-Score on validation after and before tuning

In Figure 11, we observe that the results from hyperparameter tuning varied significantly between models. Gradient Boosting and KNN classifiers barely improved in validation performance, and the Neural Networks remained with default as the best parameter combination. In contrast, Random Forest has an F1-Score improvement of over 10%.

When comparing the train and validation performance [Table 9], we can also notice a reduction in overfitting for Random Forest, but it continues to be an issue. Therefore, we decided to choose as our best model the tuned Gradient Boosting, despite being the second-best F1-Score (0.447) on validation, it demonstrated less overfitting, which was also reduced with hyperparameter tuning. This decision provides a better balance between achieving high validation performance and ensuring the model performs well on unseen data. It is important to note that while this model performs well for most classes, it struggles significantly with predicting classes like non-schedule permanent partial disability and permanent total disability, failing to classify any instances of the latter correctly during validation.

IV. Open-Ended Section

A) Predicting *Agreement Reached*

For the first part of our open-ended section, we investigated whether including the *Agreement Reached* feature could improve the prediction accuracy for *Claim Injury Type*. This decision was driven by the fact that *Agreement reached* distribution varies significantly between *Claim Injury Type*, as seen in the EDA. Indeed, some types of injuries have a constant agreement reached outcome. Additionally, the Chi-Squared test for feature selection considered this variable important in predicting outcomes.

After defining the best model to predict *Agreement Reached*, we included this predicted feature into the dataset to understand its impact on predicting *Claim Injury Type*. We tested the model with and without this variable to evaluate the performance change. Even though our model for the binary task

displayed a good classification, using it to input *Agreement Reached* in our original model resulted in a minimal increase in the F1-Score from 0.447 to 0.449. Contrary to our initial thought, it did not significantly enhance the model's performance, suggesting that both models may be underperforming in similar instances.

B) Machine Learning Website for Claim Injury Type Prediction

For the second part, we developed a machine learning website to simulate a real-world scenario and predict the claim injury type, using Python's Streamlit library. The goal of this web application was to create a user-friendly interface that simplifies the prediction process. This can help WCB automate the decision-making process to handle claims, saving time and resources. The development process was as follows:

1. **Model Selection and Preparation:** First, we selected the best performing model which was Gradient Boosting after tuning and serialized it into a **pickle file** (claim_injury_model.pkl) to ensure efficient storage and compatibility with the web application.
2. **Feature Handling:** The application accepts inputs for numerical and categorical features required by the model. We exported three pickle files to encode the categorical features and scale numerical features used in the web application: 1) an Ordinal Encoder for the input features (ordinal_encoder.pkl), 2) a Label Encoder for the target variable (label_encoder.pkl), and 3) the scaler for numerical features (scaler.pkl). This approach ensures consistent encoding of user inputs.
3. **Web Application Interface:** The interface was designed using Streamlit, focusing on simplicity and usability. Users can enter claim details through sliders and select boxes based on the feature types. We used sliders for numerical features and select boxes for categorical features, ensuring a smooth interaction with the application. Once the user has provided the required information, the application processes the inputs, encodes the categorical variables, and prepares the feature set. The data is then passed to the machine learning model to generate a prediction.
4. **Deployment:** For deployment, we only managed to deploy the app through the following github repository here. To run the app locally, copy the repository to the machine, open a terminal in the project directory, and execute the following command: **streamlit run app.py**. For it to work, Streamlit must be installed, launching the web app in the default web browser.

V. Conclusion

In conclusion, throughout the project, we managed to integrate knowledge from our course and gained insights through our own independent research, providing valuable insights for predicting *Claim Injury Types* for the WCB, by using Gradient Boosting.

However, we found that the main challenge was dealing with an imbalanced dataset and correctly predicting minority classes from *Claim Injury Type*. To address this issue, we attempted several techniques to improve our models. First, we tried a Synthetic Minority Oversampling Technique (SMOTE) and it improved our results on validation sets for Gradient Boosting and Random Forest. When looking at the class report for Gradient Boosting, the F1-Score improved for minority classes. However, it was very

minimal and resulted in overfitting, which is an important factor we consider throughout our project. Secondly, we tried tuning our models, which improved the Gradient Boosting model and reduced overfitting. However, once again, this model still struggled predicting the minority class. An additional challenge was the limitation in computational power for parameter testing, which is the reason we had to switch to Randomized Search. Thirdly, we expected *Agreement Reached* to have a higher contribution towards predicting the target, as we saw in our EDA, but this was not the case, and including this variable barely improved our model's performance. Therefore, we believe that a deeper analysis on the impact of each variable for each claim injury type would greatly benefit identifying key factors that influence minority class. Another direction may include gathering more data (e.g. company details, regional economic conditions) to provide additional context that could help predict minority classes.

Finally, despite our limited expertise, we wanted to add value to our project by creating a web application for the WCB to easily predict claim injury types. Although we successfully created the interface, this task was time consuming and we encountered several challenges, especially when encoding values. Additionally, the website app still contains errors, which is the reason why we were only able to deploy it through github.

VI. References

Abdelhadi, S., Elbahnasy K., and Abdelsalam M., (2020), "A Proposed Model to Predict Auto Insurance Claims using Machine Learning Techniques", *Journal of Theoretical and Applied Information Technology*, 98:22, pp.3428-3437.

Alamir, E., Urgessa, T., Hunegnaw, A., and Gopikrishna, T., (2021), "Motor Insurance Claim Status Prediction using Machine Learning Techniques", *International Journal of Advanced Computer Science and Applications*, 12:3, pp.457-463.

Baran, S., and Rola, P., (2022), "Prediction of motor insurance claims occurrence as an imbalanced machine learning problem", *arXiv*, arXiv:2204.06109.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P., (2002), "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, 16, pp. 321-357, DOI: https://doi.org/10.1613/jair.953.

Hanafy, M., and Ming, R., (2022), "Classification of the Insured Using Integrated Machine Learning Algorithms: A Comparative Study", *Applied Artificial Intelligence*, 36:1, DOI: 10.1080/08839514.2021.2020489.

IBM, (2023), "Example: US postal codes". Available at: https://www.ibm.com/docs/en/datacap/9.1.8?topic=data-example-us-postal-codes [Accessed: 2nd December 2024].

Liu, Y., Wang, B. J., and Lv, S. G., (2014), "Using Multi-class AdaBoost Tree for Prediction Frequency of Auto Insurance", *Journal of Applied Finance & Banking*, 4:5, pp.45-53.

Mustika, W. F., Murfi, H., and Widyaningsih, Y., (2019), "Analysis Accuracy of XGBoost Model for Multiclass Classification - A Case Study of Applicant Level Risk Prediction for Life Insurance", *5th International Conference on Science in Information Technology (ICSITech)*. Yogyakarta, Indonesia, 23-24 October, pp.71-77, DOI: 10.1109/ICSITech46713.2019.8987474.

Saikia, D., Barua, R., Gourisaria, M. K., Bandyopadhyay, A., Mishra, R. S., and Bilgaiyan, S., (2024), "Machine Learning Enhancements for Car Insurance Claim Prediction", *15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Mandi, India, June, DOIi: 10.1109/ICCCNT61001.2024.10724028.

UDS, (2019), "NAIC Uniform Data Standard Operations Manual - P&C". Available at: https://ncigf.org/wp-content/uploads/2018/10/UDS-ManualVER2.3-01-2016.pdf [Accessed: 2nd December 2024]

Wage and Hour Division, (2016), "Fact Sheet #43: Child Labor Provisions of the Fair Labor Standards Act (FLSA) for Nonagricultural Occupations". Available at: https://www.dol.gov/agencies/whd/fact-sheets/43-child-labor-non-agriculture [Accessed: 2nd December 2024].

XGBoost, (2022), "XGBoost Documentation", *xgboost 2.1.1 documentation*. Available at: https://xgboost.readthedocs.io/en/stable/index.html [Accessed: 15th December 2024].

VII. Appendix

A) Tables

Authors	Objective of the study	Predictive models tested	Data & place	Findings
Mustika, Murfi, and Widyaningsih, (2019)	Evaluate the accuracy of XGBoost in multiclass classification of risk in Prudential life insurance, compared to other models, with standard imputation preprocessing.	XGBoost, Decision Tree, Random Forest, Bayesian ridge	Dataset Prudential Life Insurance in Indonesia	XGBoost outperformed other models with an accuracy of 0.60730 and demonstrated the ability of handling missing values well.
Alamir et al., (2021)	Build a model to classify and predict motor insurance claim status.	Random Forest, Multi-class Support Vector Machine	Data from primary and secondary sources, including the standard experts of Awash Insurance Company	Multi-class SVM predicted motor insurance claim status better than Random Forest, with accuracies of 98.36% and 98.17%, respectively.
Liu, Wang and Lv (2014)	Build a model to predict auto insurance claim frequencies by framing the task as a multiclass classification problem.	Generalized Linear Model, Neural Networks, SVM, Decision Tree, Adaboost Tree	Dataset from private car Third Party Property Damage claims in Malaysia	AdaBoost performed best in predicting claim frequency, achieving a predictive accuracy of 0.8462.

Table 1: Comparative table of related work

Agreement Reached with Attorney/Representative	
Temporary	23.784
PPD NSL	384
PPD SCH Loss	307
Death	407
PTD	12
Cancelled	4

Table 2: Agreement Reached with Attorney/Representative

Variables	Chi-Squared test	Correlation	RFE	Lasso	Final Decision
Attorney/Representative	Keep	-	-	-	Keep
WCIO Nature of Injury Code	Keep	-	-	-	Keep
WCIO Part Of Body Code	Keep	-	-	-	Keep
Carrier Name	Keep	-	-	-	Keep
WCIO Cause of Injury Code	Keep	-	-	-	Keep
Industry Code	Keep	-	-	-	Keep
Carrier Type	Keep	-	-	-	Keep
District Name	Keep	-	-	-	Keep
Medical Fee Region	Keep	-	-	-	Keep
Gender	Keep	-	-	-	Keep
Zip Code	Keep	-	-	-	Keep
County of Injury	Keep	-	-	-	Keep
First_Hearing	Keep	-	Keep	Keep	Keep
C3_Received	Keep	-	Keep	Keep	Keep
C2_Received	Keep	-	Keep	Keep	Keep
Agreement Reached	Keep	-	Keep	Keep	Keep
Has_Dependents	Discard	-	Discard	Discard	Discard
COVID-19 Indicator	Keep	-	Keep	Keep	Keep
Alternative Dispute Resolution	Keep	-	Keep	Keep	Keep
Age at Injury	-	Discard	Discard	Discard	Discard
Average Weekly Wage	-	Keep	Keep	Keep	Keep
Number of Dependents	-	Keep	Discard	Discard	Discard
Age	-	Discard	Discard	Keep	Keep
IME-4 Count	-	Keep	Keep	Keep	Keep
Accident Year	-	Discard	Keep	Keep(-0.179566)	Keep
Days_from_Assembly Date	-	Discard	Discard	Keep(0.023436)	Discard

Table 4: Feature selection drops for each method

Class	Logistic Regression	Naive Bayes	Decision Tree	K-Nearest Neighbors	Gradient Boosting	Random Forest	Neural Network	Support
1.CANCELLED	0.539	0.499	0.446	0.549	0.597	0.582	0.572	2495
2.NON-COMP	0.900	0.863	0.846	0.897	0.908	0.907	0.905	58216
3.MED ONLY	0.087	0.242	0.217	0.182	0.126	0.117	0.135	13781
4.TEMPORARY	0.756	0.315	0.679	0.745	0.790	0.788	0.777	29701
5.PPD SCH LOSS	0.441	0.503	0.525	0.525	0.645	0.593	0.620	9656
6.PPD NSL	0.000	0.092	0.088	0.004	0.0162	0.002	0.002	842
7.PTD	0.000	0.012	0.000	0.000	0.000	0.000	0.000	20
8.DEATH	0.000	0.061	0.167	0.260	0.467	0.218	0.455	94

Table 6: Comparison of F1-Scores by classes

Model	Macro F1 Score Val	Macro F1 Score Train Train
K-Nearest Neighbors	0.360	0.949
Gradient Boosting	0.460	0.840
Random Forest	0.447	0.999
Neural Network	0.390	0.746

Table 7: Macro F1-Score of the 4 best models using SMOTE

Class	F1-Score
1.CANCELLED	0.590
2.NON-COMP	0.904
3.MED ONLY	0.165
4.TEMPORARY	0.768
5.PPD SCH LOSS	0.647
6.PPD NSL	0.185
7.PTD	0.054
8.DEATH	0.366

Table 8: F1-Score by class with Gradient Boosting using SMOTE

Model	Macro F1 Score Train Default	Macro F1 Score Train Tuned	Macro F1 Score Val Default	Macro F1 Score Val Tuned
K-Nearest Neighbors	0.480	0.998	0.396	0.402
Gradient Boosting	0.634	0.531	0.444	0.447
Random Forest	0.999	0.911	0.401	0.467
Neural Network	0.440	0.439	0.434	0.434

Table 9: Macro F1-Score of the 4 best models before and after hyperparameter tuning

B) Figures

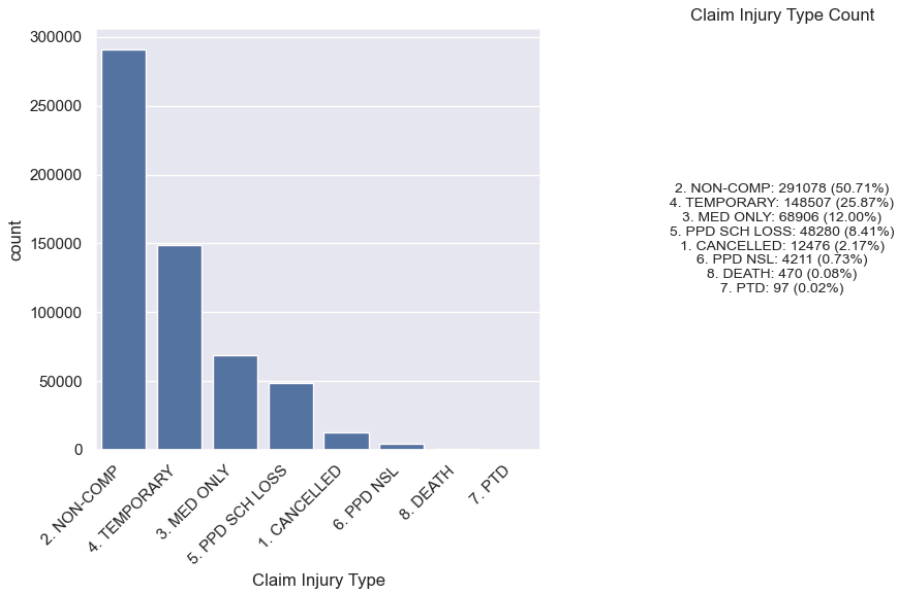

Figure 2: Claim Injury Type Count

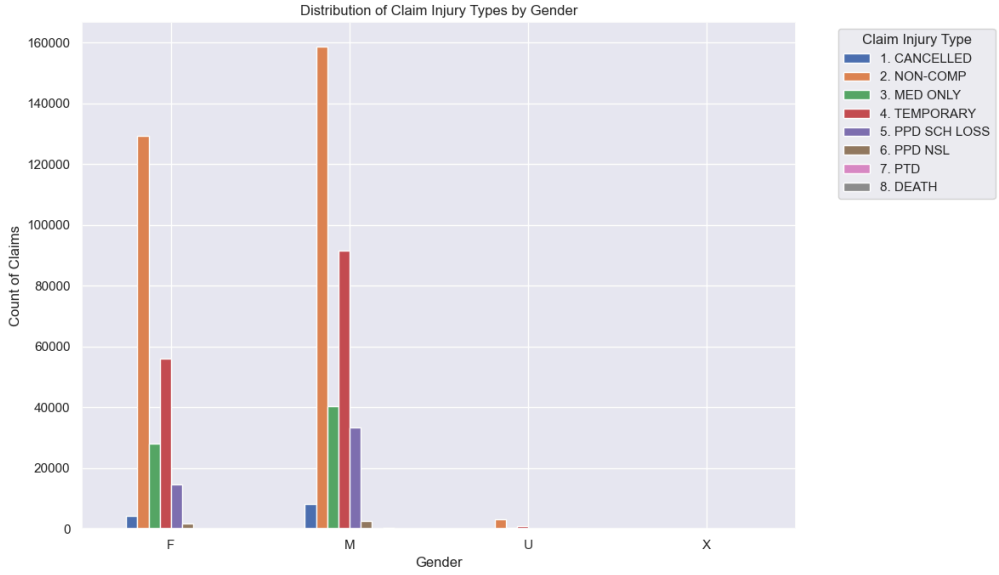

Figure 3: Distribution of Claim Injury Type by Gender

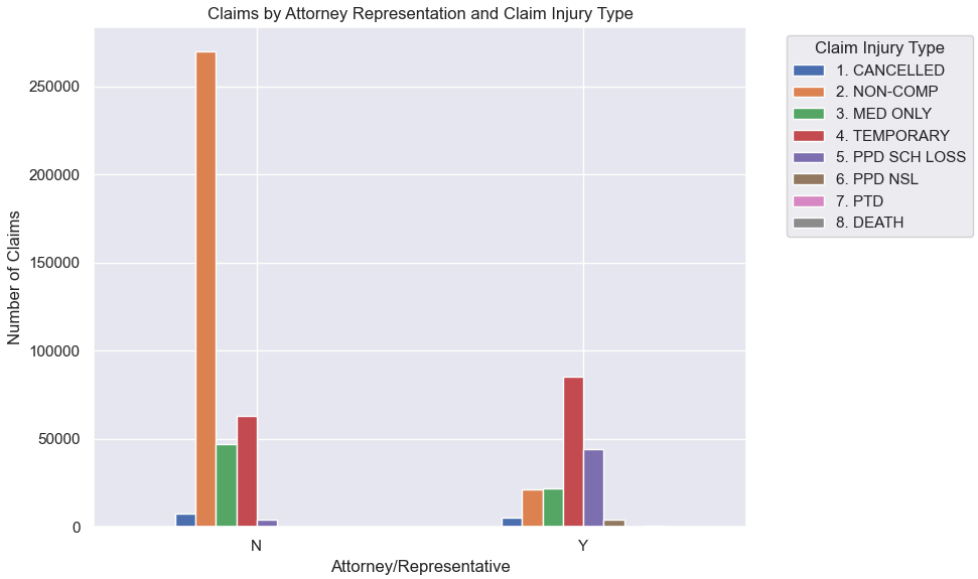

Figure 4: Claims by Attorney Representation and Claim InjuryType

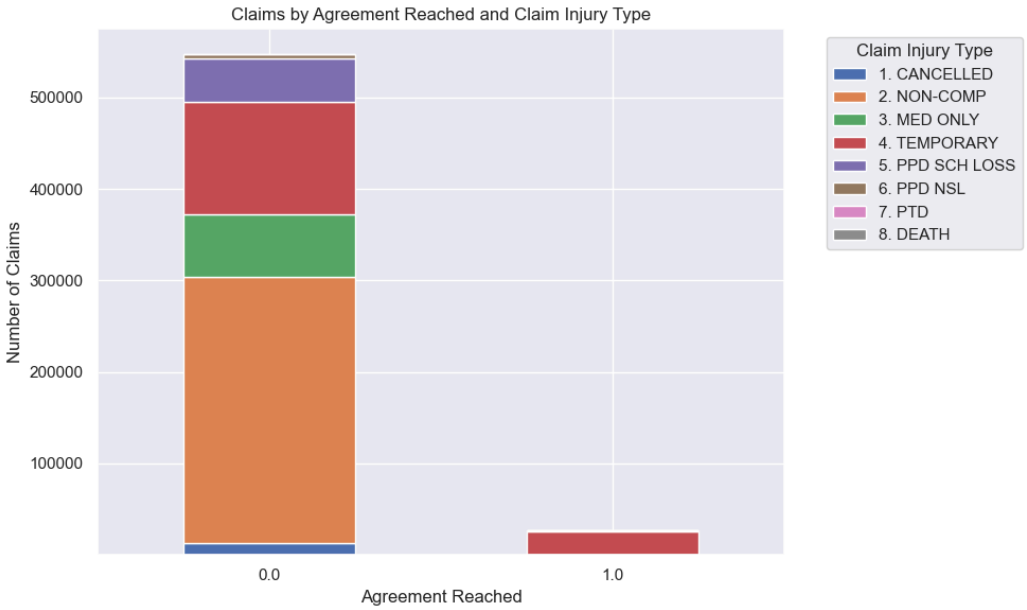

Figure 5: Claims by Agreement Reached and Claim Injury Type

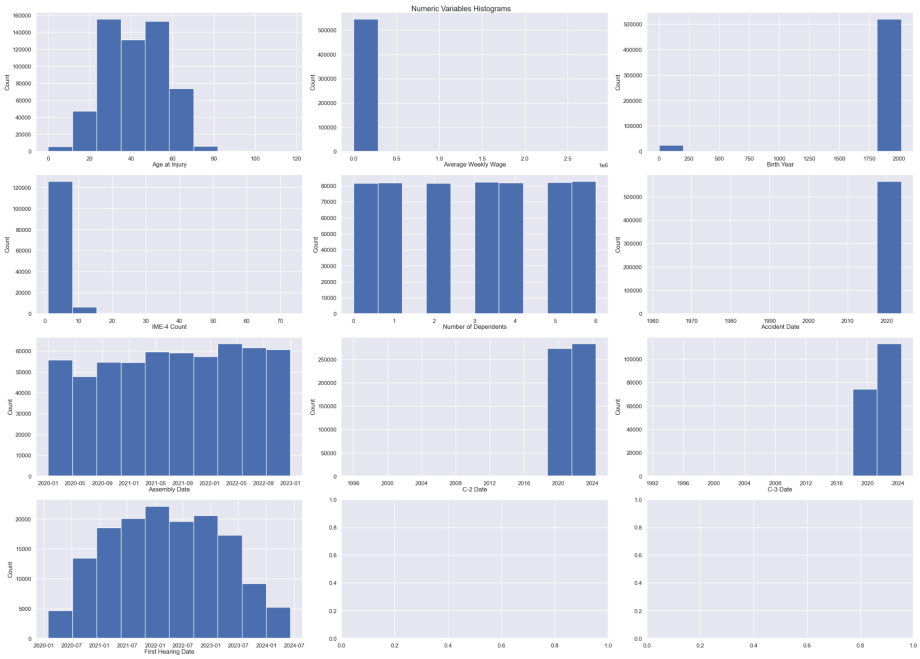

Figure 6: Numeric Variables Histograms and Boxplots

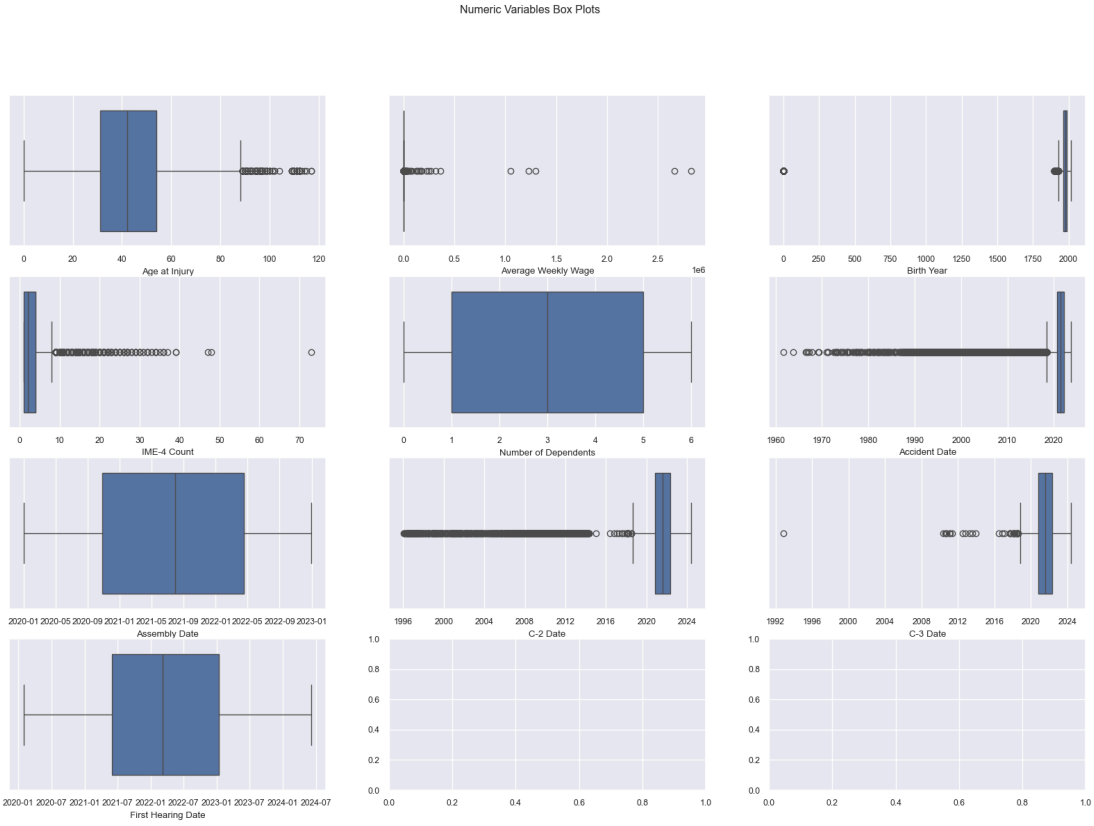

Figure 6: Numeric Variables Histograms and Boxplots

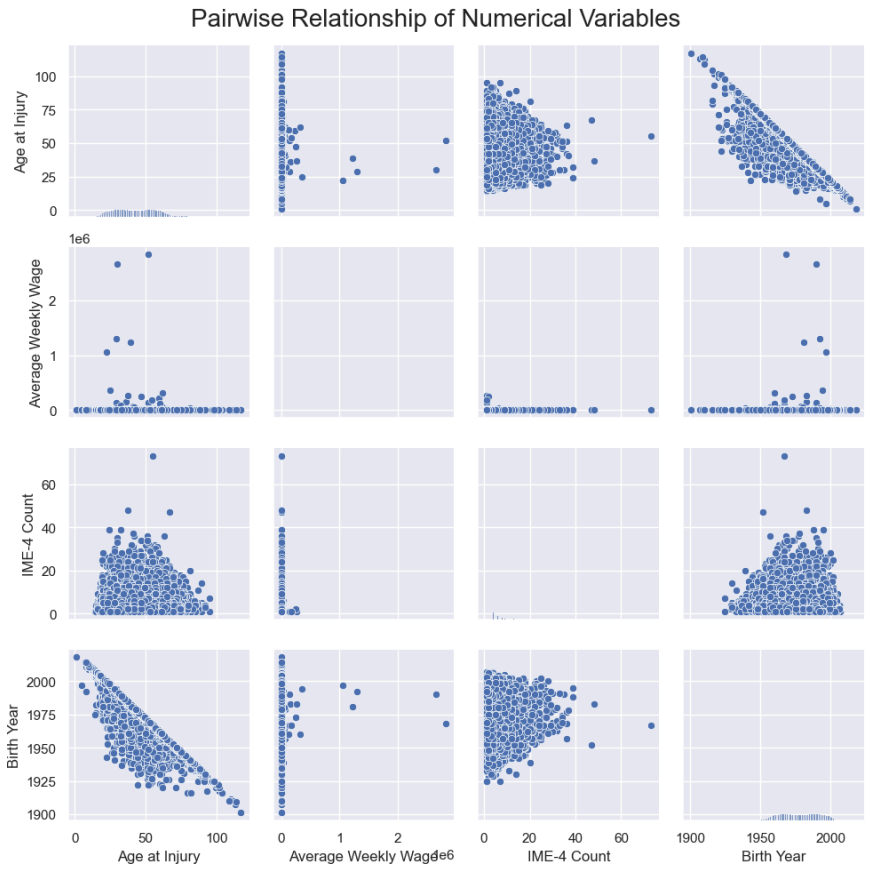

Figure 7: Scatter Plot of Pairwise Relationship of Numerical Variables

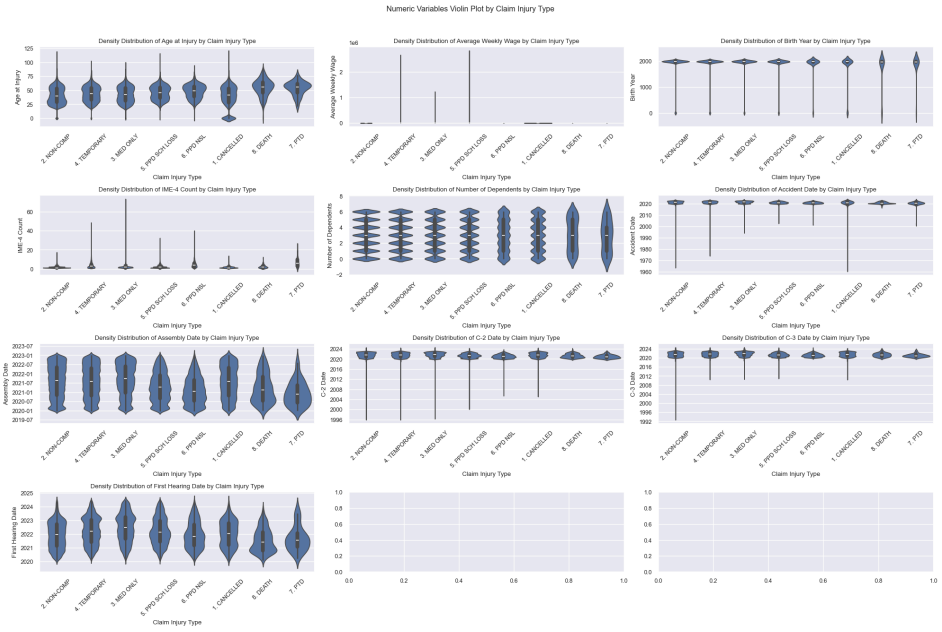

Figure 8: Numeric Variables Violin Plots by Claim Injury Type

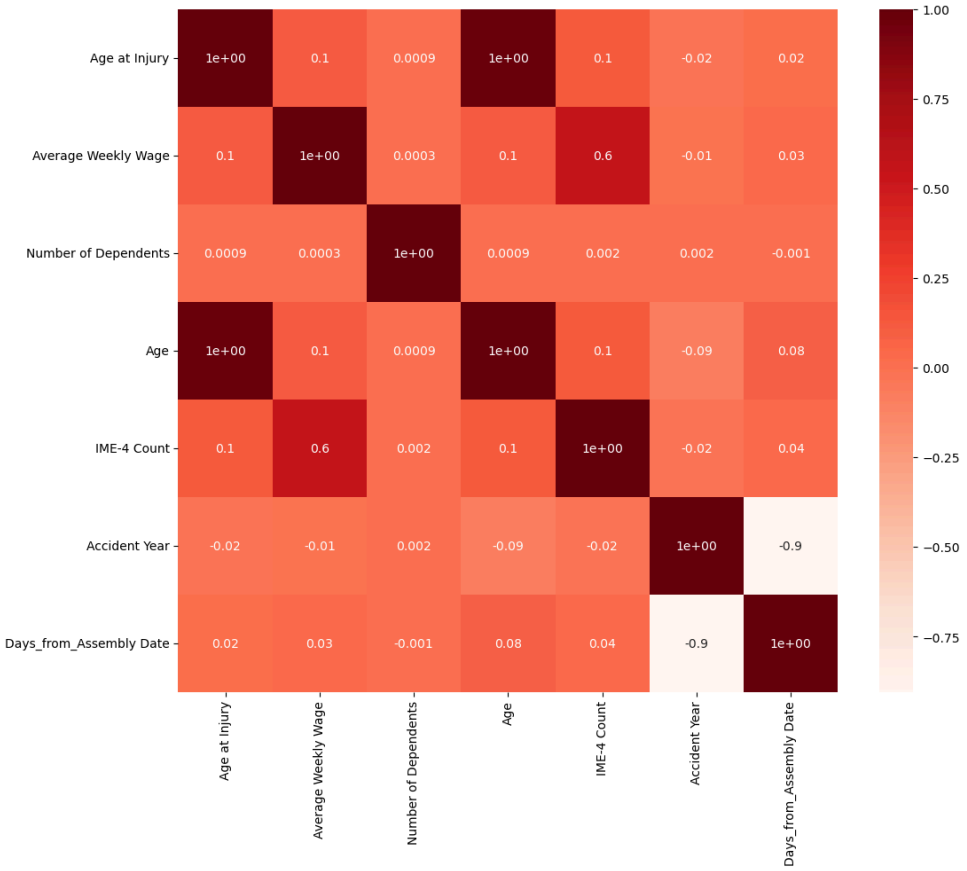

Figure 9: Spearman Correlation Matrix

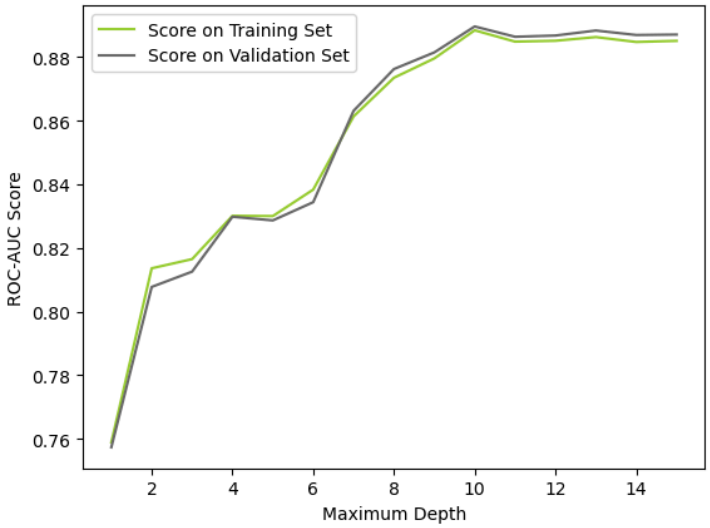

Figure 10: ROC AUC performance across feature subsets