MACHINE LEARNING PROJECT

Master in Data Science and Advanced Analytics

NOVA Information Management School

Universidade Nova de Lisboa

To Grant or Not to Grant

Group 37

Benedikt Ruggaber, 20240500

Ricardo Pereira, 20240745

Francisco Pontes, 20211583

Github: https://github.com/Benerugg/ML-Group37

Fall/Spring Semester 2024-2025

Abstract

The New York Workers Compensation Board (WCB) is responsible for the claim process and has reviewed more than five million cases since the year 2000. This Project's goal is to generate automated classification of claims by predicting the injury type with the help of trained machine learning models. We accomplish this by using labelled data from 2020 to 2022 that was provided for the training and test data sets.

The project began with data analysis and preprocessing, addressing issues such as missing values, outliers, creating new features, encoding and scaling variables. Furthermore, a detailed feature selection was carried out using a variety of methods to identify the optimal variables for the model. After this, various machine learning algorithms were developed and evaluated, with a stacked model having the best performace, achieving a F1 macro score of 0.43291 on Kaggle.

With the implementation of this solution, we enable a more efficient decision-making process in workers' compensation by using the created Web-Interface, reducing the manual evaluation and enhancing the correct output for the claim Identifier.

Keywords: New York Workers' Compensation Board (WCB), Machine Learning, Data Preprocessing, Model Optimization, Feature Engineering, Feature Selection, Claim Classification

Table of Contents

1. INTRODUCTION

The management of workers' compensation claims in New York has become a significant challenge for the Workers' Compensation Board (WCB). It is notable that between the years 2000 and the present, the WCB has received more than 5 million claims. This figure represents a significant increase over previous years. Consequently, the number of claims being reviewed manually has continued to grow. The WCB in New York is responsible for the supervision and administration of workers compensation, the disbursement of benefits to the disabled, and the provision of compensation to volunteer firefighters, volunteer ambulance drivers, and volunteer workers in the civil defense.

This study aims to develop a predictive model that can automate the classification of claims based on injury type, using historical data from 2020 to 2022. The target variable is "Claim Injury Type", which categorizes claims into different levels of severity and compensation. The dataset holds a wide range of features, from basic claim information (Accident Date, Assembly Date, First Hearing Date) to detailed worker characteristics (Age at Injury, Gender, Average Weekly Wage) and injury specific data (WCIO Nature of Injury Code, WCIO Part of Body Code, WCIO Cause of Injury Code).

Our goal is to develop a predictive model that can classify new injury claims. This study will evaluate several machine learning algorithms, comparing their effectiveness in automating the claims classification process. Through feature selection and parameter optimization, we aim to enhance model performance while identifying the key determinants that influence claim outcomes.

The results of recent studies indicate that the implementation of automated classification in workers compensation systems may be a beneficial approach. In the study conducted by Zanke et al. (2024), it was found that 35% of the organizations that had adopted new predictive analytics technology reported noticeable effectiveness. This study shows the significant outcomes in areas directly relevant to the objective of developing a classification model in monitoring the claim progress.

2. DATA EXPLORATION AND PREPROCESSING

The first step to implement this solution was to explore the data that we were given. This analysis takes place in the two datasets provided by the WCB, that contains information about claims related to customer injury types. The training dataset, which was used for model development, contains data from the beginning of 2020 to 2022 with 593,471 records and 32 distinct variables. The test dataset, which was used to evaluate the model's performance, contains 387,975 records and has 30 attributes. Each claim in both datasets is assigned a unique claim identifier number, which serves as the index for both datasets.

The first stage of the data quality analysis revealed several issues. Firstly, inconsistencies in data types were identified and corrected. Secondly, we handled missing values. Additionally, a notable discrepancy was found in the column structure between the training and test datasets. This discrepancy is particularly important because, to generate predictions, it is required to have consistent features across both datasets. ‘Agreement Reached’ and ‘WCB Decision’ aren’t on the test dataset and since we aren’t building a model to predict them, they were removed. There was also an inconsistency where rows from the train dataset had missing values for the Claim Injury Type (target variable; represents 3% of the dataset) we also remove them since these cases were of no benefit to the training of the model as well as the variable *OIICS Nature of Injury Description* due to its high proportion of

missing values (100%). To prepare the data for modelling, we started by splitting the dataset into train (70%) and validation (30%) subsets.

2.1. Data Coherence

The analysis of coherence revealed a number of critical points that required further investigation and correction to ensure the integrity and reliability of the data.

With regard to date coherence, it was found that, in some cases, the order of the dates did not correspond to the expected logical order. Ideally, the timeline would flow as follows: date of the accident, followed by the *C-2 date*, then the *C-3 date*, then the *assembly date* and finally the *date of the first hearing*. It was therefore necessary to correct this temporal inconsistency to maintain data integrity. To resolve this, temporal inconsistencies were corrected, and flags were created to validate the proper order of events. Additionally, values for the *Accident Date* were found extending into 2023, which is inconsistent with the claim period (2020–2022).

Furthermore, we found considerable age-related anomalies within the dataset. *Birth Year* of the workers isn't coherent, since there are values of 0 and 2018 for example which are implausible. To handle this, we replace the values where *Birth Year* is 0 for the difference between the *Accident date* and *Age at Injury on the training dataset*. The *Age at injury* also had values such as 0 and 117, which are incoherent, since there aren't workers with that age. To address this, we established a valid age range and introduced a flag column that marks all ages who within this range.

The *Average Weekly Age* feature displayed extreme values, such as 0 and 2828079. However, further investigation revealed that 58% of the entire dataset contained values of 0. This was not considered as an anomaly, as it could correspond to unpaid volunteers or specific cases where wages were not applicable. We found that there were values in *Carrier Name* that differ by a point or a -, so they represented exactly the same Carrier Name but were treated as different, we treated them using fuzzy [2] matching to standardize and consolidate these values.

2.2. Handling missing values and outliers

The dataset contained 16 features with missing values: *Accident Date*, *Average Weekly Wage*, *Birth Year*, *C-2 Date*, *C-3 Date*, *First Hearing Date*, *IME-4 Count*, *Industry Code*, *Industry Code Description*, *WCIO Cause of Injury Code*, *WCIO Cause of Injury Description*, *WCIO Nature of Injury Code*, *WCIO Part of Body Code*, *WCIO Part Of Body Description and Zip Code*. All the statistics calculated to impute the missing values were based on values from the training dataset, since this way there is no data leakage.

For the datetime features such as: *Accident Date, C-2 Date*, *C-3 Date and First Hearing Date,* we filled the missing values with the modes from the following occurred event, for the *accident date* we filled with the mode from *C-2 Date,* since there were still missing values we filled then with *C-3 Date* and so on...we did this for all the datetime features. For the C-3 Date we also created a binary column to check if there is a missing value or not. In the *First Hearing Date*, 73.7% of the values were missing, indicating that hearings had not yet been scheduled for a majority of the claims. We created a placeholder with a future date and filled the missing values with that value. We also created a binary column that is 1 if there was already a first hearing and a 0 if it's a NaN.

For the *Average Weekly Wage, as* already said previously, we interpreted 0 as someone who didn't receive money because was a volunteer. Because of this, we believe that being a missing value could be interpreted the same, we filled the missing values with a 0. Since IME-4 Count had a lot of missing values, we created a binary column to record if there was a count or not, and we imputed the missing values with 0. For the Birth Year in the train dataset, we decided to fill the missing values with the same logic as we handle the 0's (difference between accident date and age at injury). As for the validation and test we imputed the train dataset's median, to avoid data leakage.

For the categorical data such as: *Industry Code, WCIO Cause of Injury Code, WCIO Nature of Injury Code*, *WCIO Part of Body Code w*e filled the missing values with the mode. *And for the remaining categorical features: Industry Code Description, WCIO Cause of Injury Description, WCIO Part Of Body Description* we filled it with the description where there is the most common code .As for the zip code we created a placeholder and we filled the missing values with that, we also created a binary column of the zip code to check if a row as a valid zip code or not, based on the assumption that a valid zip code in the United States must have exactly 5 digits.

In terms of outliers in Age at Injury, we took care of the outliers by creating a column that flags them. The variables Average Weekly Wage and IME-4 Count have, respectively, a considerable amount of 0's and a lot of missing values, making us more skeptical about dealing with outliers, since values that would be considered normal in a regular case are considered outliers, given that the columns are very imbalanced. We should consider that Average Weekly Wage presents values with a high disparity, which can indicate that the way this question was interpreted by the claimants was different, and the values are probably not comparable. We decided to take a conservative approach and keep the outliers of these variables.

3. MULTICLASS CLASSIFICATION:

3.1. Feature engineering

In order to optimize our model's ability to predict Claim Injury Type, we created these new features:

FEATURE NAME	DESCRIPTION
ACCIDENT_MONTH	Month extracted from Accident Date
ACCIDENT_DAYOFWEEK	Day of week extracted from Accident Date
DAYS_ACCIDENT_TO_ASSEMBLY	Days between Accident Date and Assembly Date
DAYS_ACCIDENT_TO_C2	Days between Accident Date and C-2 Date

DAYS_ACCIDENT_TO_C3	Days between Accident Date and C-3 Date
DAYS_ACCIDENT_TO_FIRST_HEARING	Days between Accident Date and First Hearing Date
REGION_RISK_SCORE	Average IME-4 Count by Medical Fee Region
COUNTY_CLAIMS_NORMALIZED	Claims per county normalized by maximum count
INDUSTRY_CLAIM_VOLUME	Total claims by Industry Code
INDUSTRY_AVG_AGE	Average Age at Injury by Industry Code
INDUSTRY_WAGE_RANK	Average wage percentile by Industry Code
ACCIDENT_DATE_FLAG	Binary flag for inconsistencies in Accident Date sequence
AGE_OUTLIER_FLAG	Binary flag for ages outside valid range (14-88)
C2_AFTER_C3_FLAG	Binary flag if C-2 Date is after C-3 Date
C2_AFTER_ASSEMBLY_FLAG	Binary flag if C-2 Date is after Assembly Date
C2_AFTER_FIRST_HEARING_FLAG	Binary flag if C-2 Date is after First Hearing Date
C3_AFTER_ASSEMBLY_FLAG	Binary flag if C-3 Date is after Assembly Date
C3_AFTER_FIRST_HEARING_FLAG	Binary flag if C-3 Date is after First Hearing Date
C-3 DATE CONVERTED	Binary flag indicating presence/absence of C-3 Date
FIRST HEARING DATE CONVERTED	Binary flag indicating presence/absence of First Hearing Date
IME-4 COUNT CONVERTED	Binary flag indicating presence/absence of IME-4 Count
VALID_ZIP_CODE	Binary flag for valid 5-digit zip codes

Table 1 – New Features

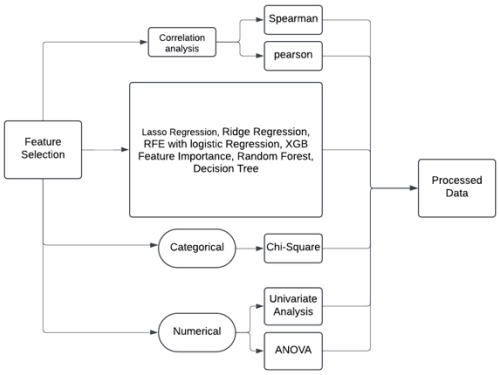
3.2. Encoding and Scaling

To prepare the dataset for modelling, we needed to firstly encode the categorical features. We encoded based on frequency, by replacing each category with the frequency of its occurrence. This was the selected method as it helped us preserve the information about category importance through frequency values, because it is efficient and works well with tree-based algorithms. It also helps avoid the increase in dimensionality that can happen with other encoding methods, since our features had a very large number of unique values in the columns.

After encoding the categorical features, we scaled all the features to ensure they were on a comparable scale. This step was essential to account for the preprocessing applied to the data, particularly since we did not directly remove any outliers. Our scaling method was chosen specifically to handle the characteristics of our data effectively.

3.3. Feature Selection

To get the best model performance, we implemented a variety of methods to analyze the importance of each feature to handle categorical and numerical features. The implemented code followed the structure of this graphic:



Pipeline 1 – Feature Selection

We applied several methods to assess the importance of the features:

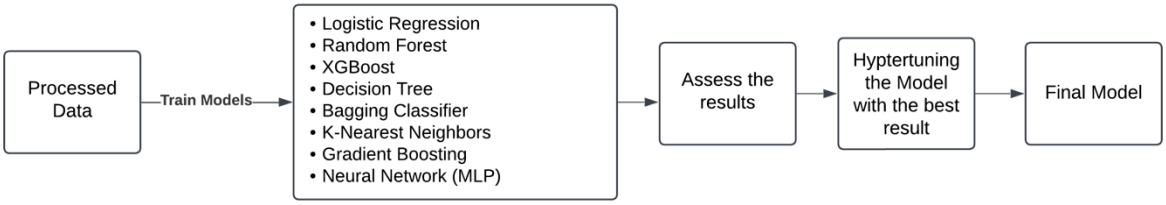
- **Univariate analysis**: For numerical features, we assessed their individual predictive power by measuring their relatioship with the target variable, which evaluates the variance.
- **Correlation analysis:** identified several highly correlated pairs e.g. year of birth and age at injury, which showed a correlation of 0.99, while Days_Accident_to_C2 and Days_Accident_to_Assembly showed a correlation of 0.91. Similarly, Days_Accident_to_C3 showed strong correlations (0.90-0.99) with the related duration measures (Figure 1 Spearman and 2 Pearson).
- **Chi-squared for categorical data:** showed the selection by revealing that all the categorical features have strong dependence on the target variable.
- **RFE with LogisticRegression:** Using logistic regression as the base estimator, RFE served to rank features by iteratively removing the least significant ones. This method was crucial to help us decide the ideal number of features for our model (Figure 3).
- **Random Forest:** This algorithm was used to compute feature importance based on the crease in impurity and its effectiveness in capturing non-linear relationships. It identified average weekly wage, days_accident_to_first_hearing, Attorney/Representative and IME-4-Count as the most significant variables (Figure 4);
- **Decision trees:** using both gini and entropy metrics, this algorithm revealed similar key variables to the first method shown, with Attorney/Representive and WCIO codes also identified as important predictors (Figure 5).
- **Lasso Regression:** This regression helped us highlight the most important predictive features while eliminating redundant ones (Figure 6);
- **Ridge Regression:** Even though Ridge does not output selected features, it gave us a clear perception of the ones that were considered relevant (Figure 7)
- **XGB Feature Importance:** The Extreme Gradient Boosting feature importance has outputted the features that are considered the most relevant for this specific model, based on Weight. This gives us clear insights on what features we should choose, if we decide to choose this model going forward (Figure 8);

- **ANOVA F-Values:** This method selects the features that are the most effective in differentiating target categories based on distances, which gives us a different perspective on feature selection, though only on numerical features.

Regarding the number of features to be chosen, using RFE with logistic regression as shown in Figure 1, we identified that the model achieves optimal performance with 19 features, while avoiding overfitting. By combining all feature methods and their results we could conclude that the selected features for modelling should be: 'Days_Accident_to_First_Hearing', 'IME-4 Count', 'Average Weekly Wage', 'Attorney/Representative', 'WCIO Cause of Injury Code', 'WCIO Part of Body Code', 'Region_Risk_Score', 'C2_After_C3_Flag', 'C3_After_Assembly_Flag', 'C-3 Date Converted', 'Days_Accident_to_C3', 'Carrier Name', 'Birth Year', 'Days_Accident_to_Assembly', 'District Name', 'Medical Fee Region', 'Accident_Month', 'Age_Outlier_Flag', 'Count_Claims_Normalized'.

3.4. Model selection and evaluation strategy

To determine the best model for our task, we have decided to train multiple models and evaluate them using the F1 Macro score. This evaluation strategy aligns with the metric used for Kaggle submissions, ensuring our approach is coherent with the competition requirements. By selecting the model with the highest F1 Macro score, we then plan to optimize performance for the selected model and ultimately get the best results possible.



Pipeline 2 - Modelling

Based on Table 'number,' we conclude that XGB (Extreme Gradient Boosting) achieves the highest score. This algorithm not only demonstrates the best F1 Macro Score on the Validation Data Set and Accuracy but also exhibits a small discrepancy between the Kaggle and Validation F1 Scores. This consistency suggests that XGB is the most reliable model. Coming in as second place, GBM (Gradient Boosting Machine) achieves similar scores, but is computationally expensive. The third best performance belongs to RF (Random Forest), coming close to GBM, while being least expensive computationally. With this in mind, we will focus on further optimizing XGB and RF, to see if we can achieve a better score.

Model	Train Accuracy	Val Accuracy	macro F1-Score Train	macro F1-Score Val	Kaggle Score
Logistic Regression	0.7336	0.7015	0.2698	0.2669	0.29138
Random Forest	1	0.7843	0.9999	0.3881	0.3667
Decision Tree	1	0.6669	1	0.3558	0.23231
Bagging	0.9826	0.7616	0.9687	0.3801	0.34183
KNeighborsClassifier	0.7882	0.7196	0.4271	0.3215	0.30614
GBM	0.7818	0.7802	0.4359	0.3828	0.38161
XGB	0.803	0.7879	0.6424	0.4222	0.41338
Neural Network	0.7762	0.7439	0.3377	0.3622	0.31652

Table 2– Tested models without hyperparametrization

3.5. Model optimization efforts

To optimize our model, we conducted an analysis to evaluate the impact of different parameters on performances for both algorithms. Our primary criterion for optimization was achieving the highest Macro F1 Score, and for each parameter tested, we identified the value that produced the best mean score across 20 training iterations. The optimal parameter values from these experiments were then implemented in our final model configuration.

Model	Train Accuracy	Val Accuracy	macro F1-Score Train	macro F1-Score Val	Kaggle Score
XGB(No Hyperparametrization)	0.803	0.7879	0.6424	0.4222	0.41338
XGB(With Hyperparametrization)	0.8096	0.7897	0.51	0.4247	0.42973
Random Forest(No Hyperparametrization)	1	0.7843	0.9999	0.3881	0.3667
Random Forest(With Hyperparametrization)	1	1	0.7821	0.3889	0.36568
Stacking	0.8316	0.7893	0.5632	0.4159	0.43291

Table 3 – Model optimization efforts

As shown in Table 3, our optimization efforts resulted in a slight improvement in both the Validation F1 Macro Score and the Kaggle F1 Macro Score for the XGB of around 1.6%. (Figure 9 to 14) In contrast, Our Random Forest tuning efforts were in vain, as it made our Kaggle score decrease slightly. (Figure 10 to 15) We tried different sampling strategies to correct the imbalance, but these efforts were unsuccessful, resulting in a lower kaggle F1 macro score. Finally, we have decided to implement a stacking of both tuned models, which improved our kaggle score to 0.43291. Having this in mind, our final solution is represented by a stacked model composed of Extreme Gradient Boosting and Random Forest algorithms.

4. OPEN END SECTION

4.1. Objectives

The main objectives for the open end section of our project was to create a practical Web App for the WCB to quickly generate predictions for the outcome of claims. Therefore, we wanted to transform our model into a tool that could be easily used in everyday usage, without requiring technical expertise from the users.

4.2. Description

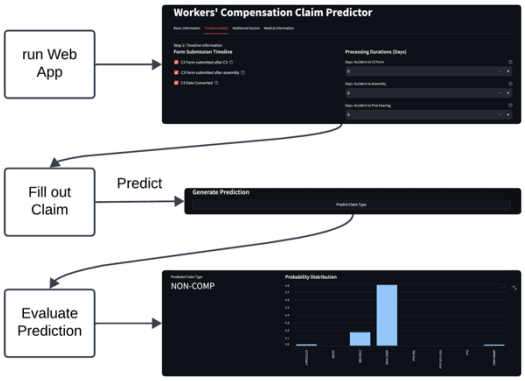
We created an interactive Web-App using streamlit[3][4] that should guide the user through the claim progress. This application is divided into four main areas:

1. basic information
2. timeline with details

3. other factors
4. medical information

Each section features input validation from the selected 19 features used to train the model. Input validation, including simple input controls such as dropdowns, sliders, and checkboxes, ensures data quality and ease of use. Once the user has completed the requisite information, the app provides an efficient method for predicting the outcome. By our probability distribution, the user should be able to understand which claim the model has predicted and proceed with this information.

4.3. Results and Limitations



Pipeline 3 – Result Interface

As shown in the Pipeline 3 we implemented a 3-step process for our claim predictions:

1. **Start the Application**

- User enters the Web App
- Application loads itself with our Model and Mappings
- User get presented our Workers Compensation Claim Predictor

2. **Claim Entry**

- User fills out our sections
- Each field has validation protects from invalid Data

3. **Prediction Output**

Our model generates outputs in 2 fields: **Primary description**, and **Probability Description**, which shows a graphic with all of the possible Outcomes. This graphic was designed to help us understand the Probabilities for each outcome possible.

We wanted to make this project was to make our machine learning model accessible to WCB, providing practical value for real-world claims. However, during implementation, we encountered some technical limitations. The main challenge was that our model was trained on scaled and encoded data. The biggest limitation of this implementations is that we were not able to integrate scaling and encoding processes into the app, which meant that the raw claim information entered by users was not transformed in the same way as our training data prior to prediction.

5. CONCLUSION

When starting this project, we were presented with the challenge of developing a predictive model for the WCB, with the end goal of automizing claim injury type classifications. We were presented with data from the period 2020 to 2022, and we developed a thorough program that consisted of three main phases: Data Exploration, where we analyzed visually and identified the main issues of the data set, Data Pre-processing, where we handled the incoherences of the given data set, and finally, Modelling, where we deploy our solution and evaluate the predictions and results.

Regarding our findings on this project, they were, for the most part, aligned with our expectations. Since we have an imbalance on the classifications, we were expecting tree-based ensembles to have a better score on this problem than other models, which was the case in this project. Regarding feature selection, we were expecting 'Average Weekly Wage' and 'WCIO Cause of Injury Code' to stay for the modelling stage due to, at first glance, seeming to be good discriminating variables for the project. However, there were also some surprises, as it was the case for our feature selection algorithms to not favor 'WCIO Nature of Injury Code' as a relevant feature.

Regarding the limitations of our work, we believe that the hyper-parametrization for our model could have been better, as it only improved our Extreme Gradient Boosting model score by 1.6%, and not improving at all on the Random Forest algorithm. Besides that, we believe that our Open End Section has room for improvement, due to not being able to scale and encode the data when inputting the values for prediction, as mentioned above.

For future work, we suggest developing a more efficient hyper parametrization strategy, maybe with automatized functions with that purpose. We also believe that it would be interesting to create models that predict other variables given, such as agreement reached.

SOURCES

[1] ZANKE ET AL. (2024)
https://www.researchgate.net/profile/Pankaj-Zanke-2/publication/379834411_Optimizing_Worker's_Compensation_Outcomes_Through_Technology_A_Review_and_Framework_for_Implementations/links/661dd3fbf7d3fc2874633b21/Optimizing-Workers-Compensation-Outcomes-Through-Technology-A-Review-and-Framework-for-Implementations.pdf

[2]fuzzy wuzzy
https://github.com/seatgeek/fuzzywuzzy?tab=readme-ov-file

[3] streamlit
https://streamlit.io/

[4] streamlit introduction
https://www.youtube.com/watch?v=xl0N7tHiwlw

APPENDIX

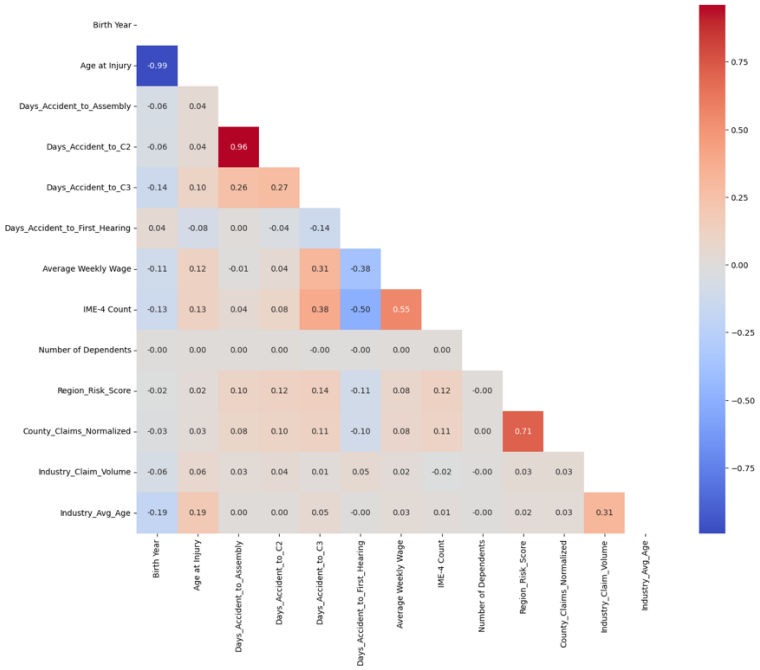


Figure 1 – Spearman Correlation

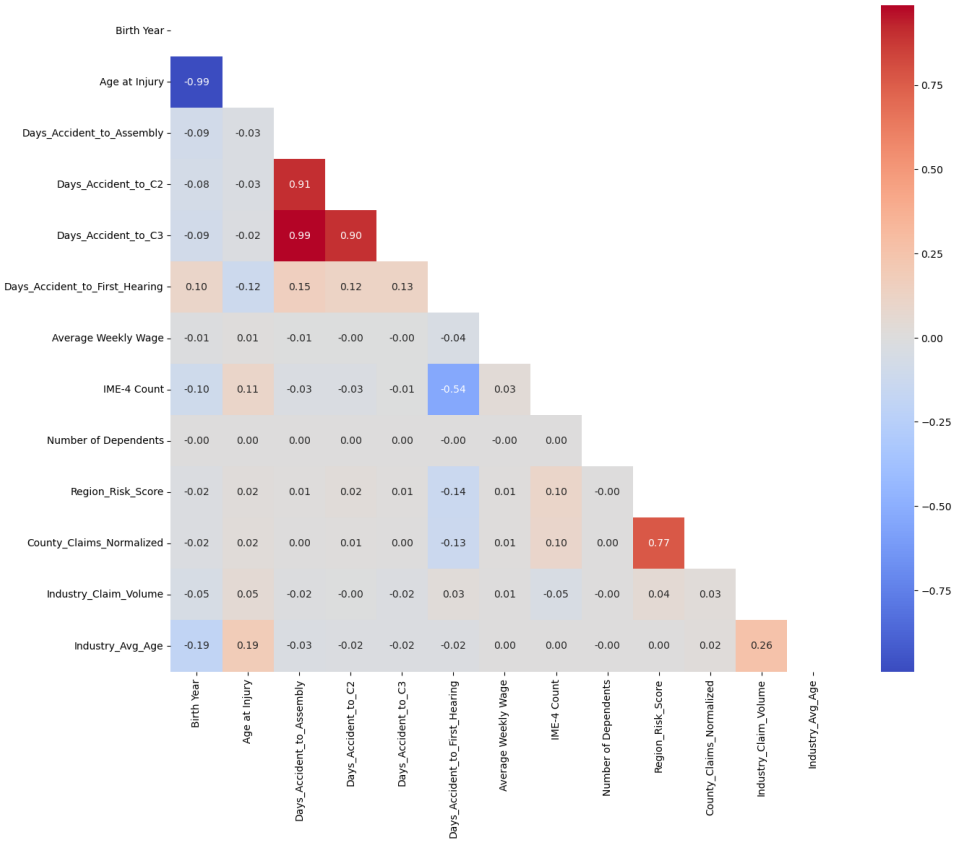


Figure 2 – Pearson Correlation

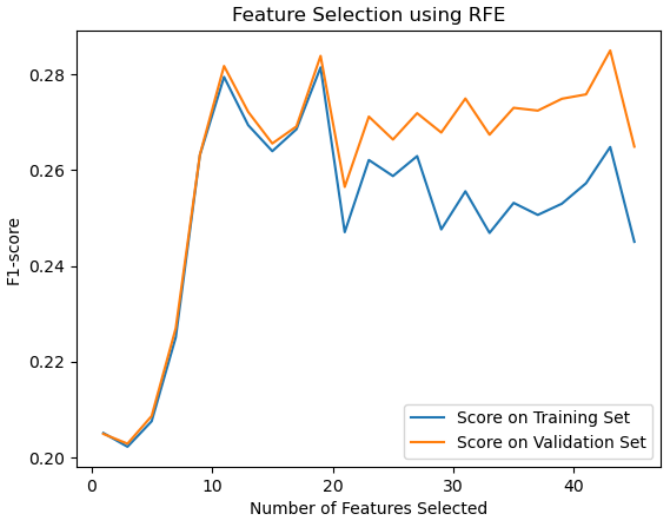


Figure 3 - RFE

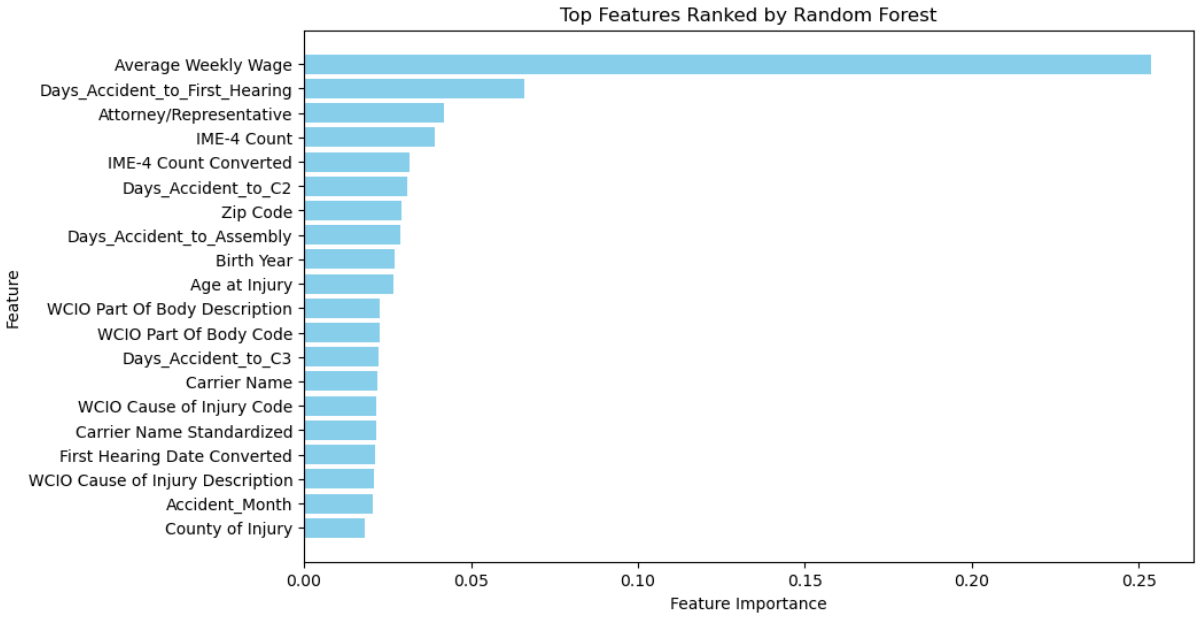


Figure 4 – Random Forest Feature Importance

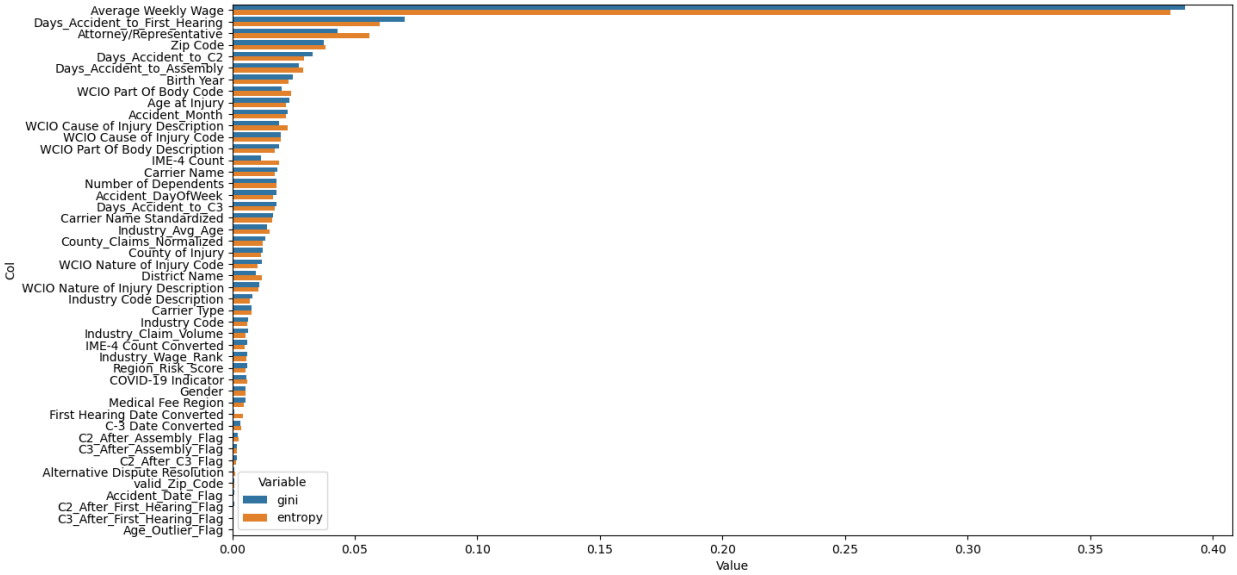


Figure 5 – Decision tree feature importance

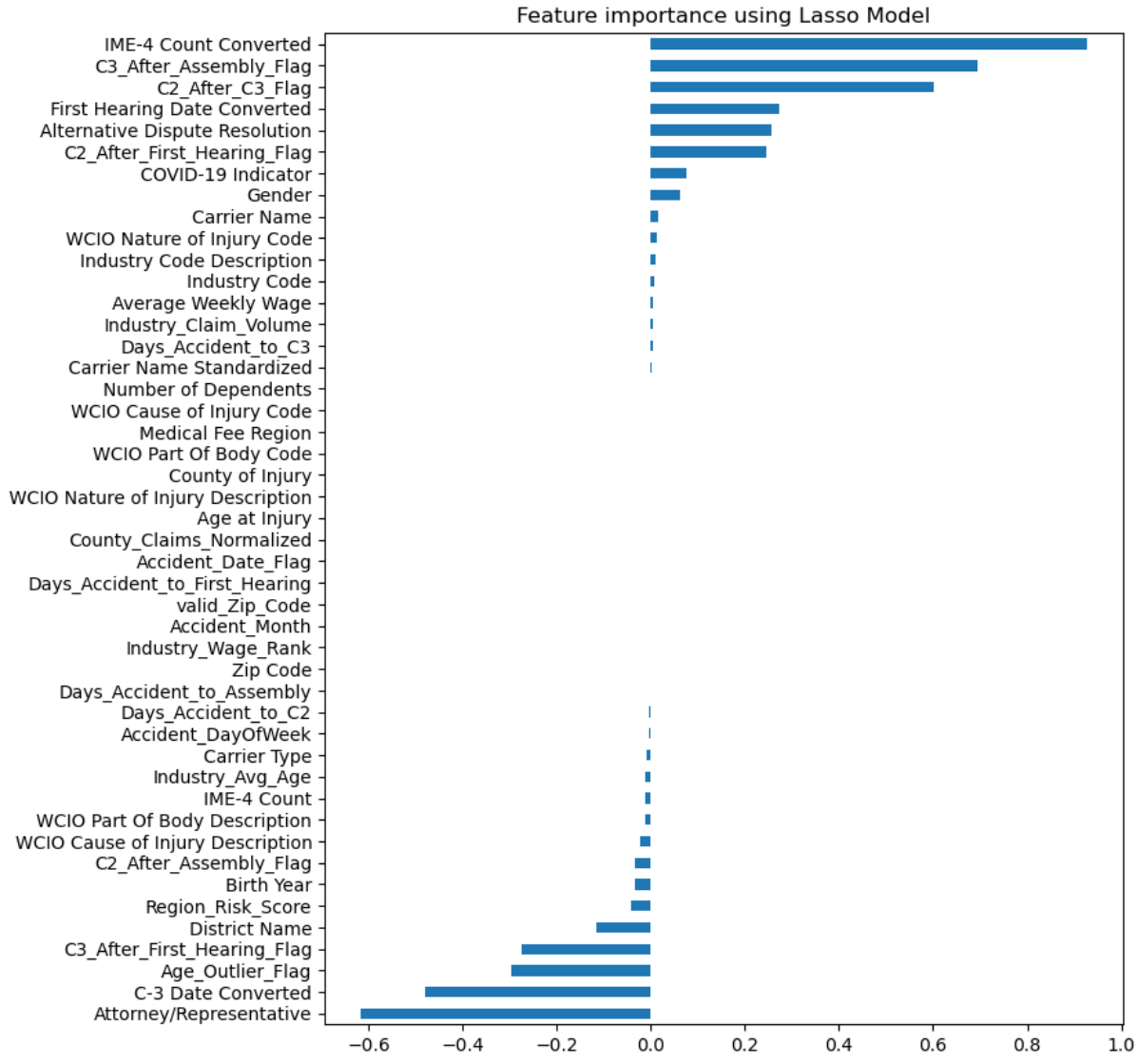


Figure 6 – Lasso Regression Feature Importance

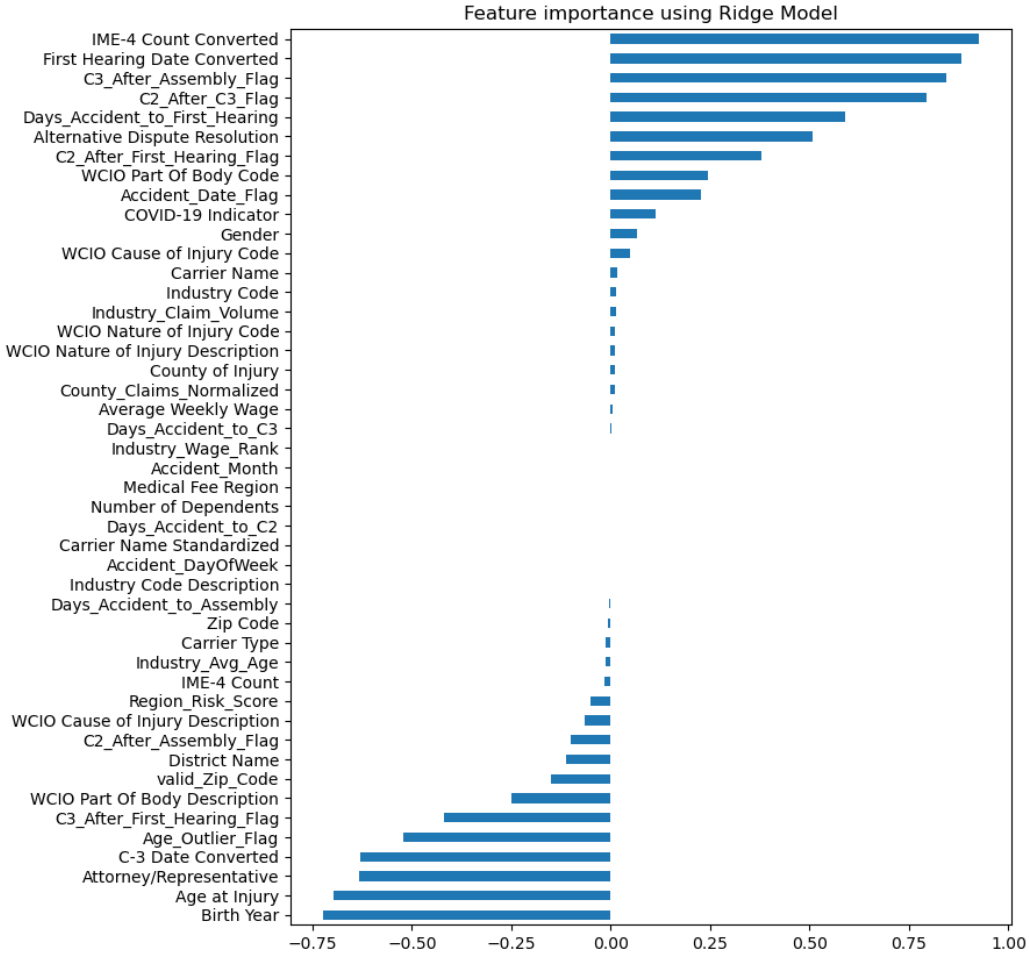


Figure 7 – Ridge Regression Feaure Importance

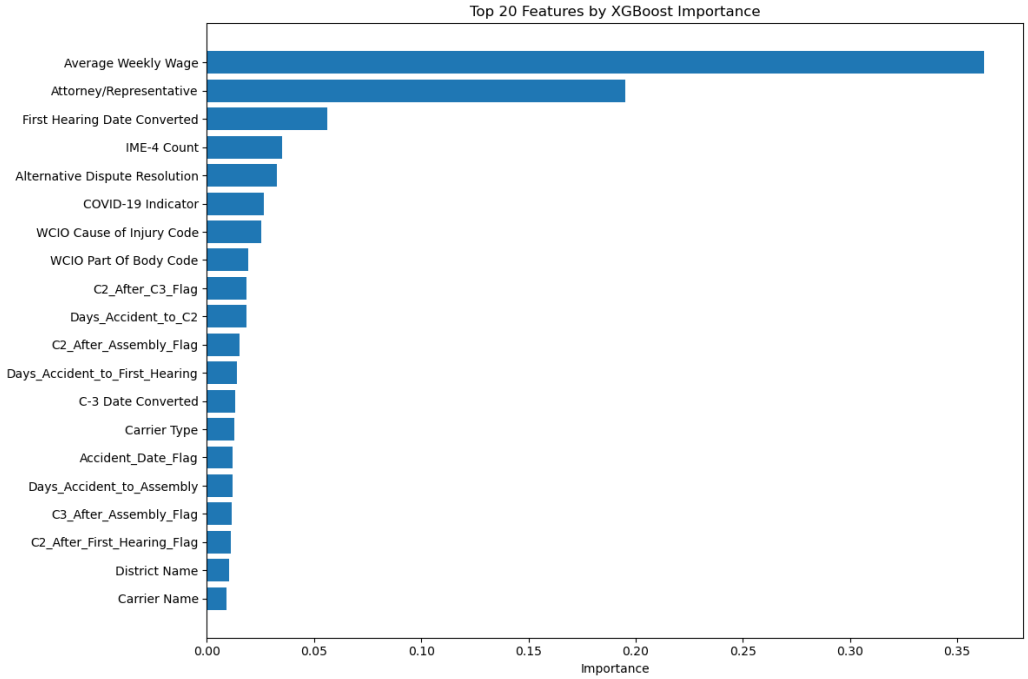


Figure 8 – XGB Feature Importance

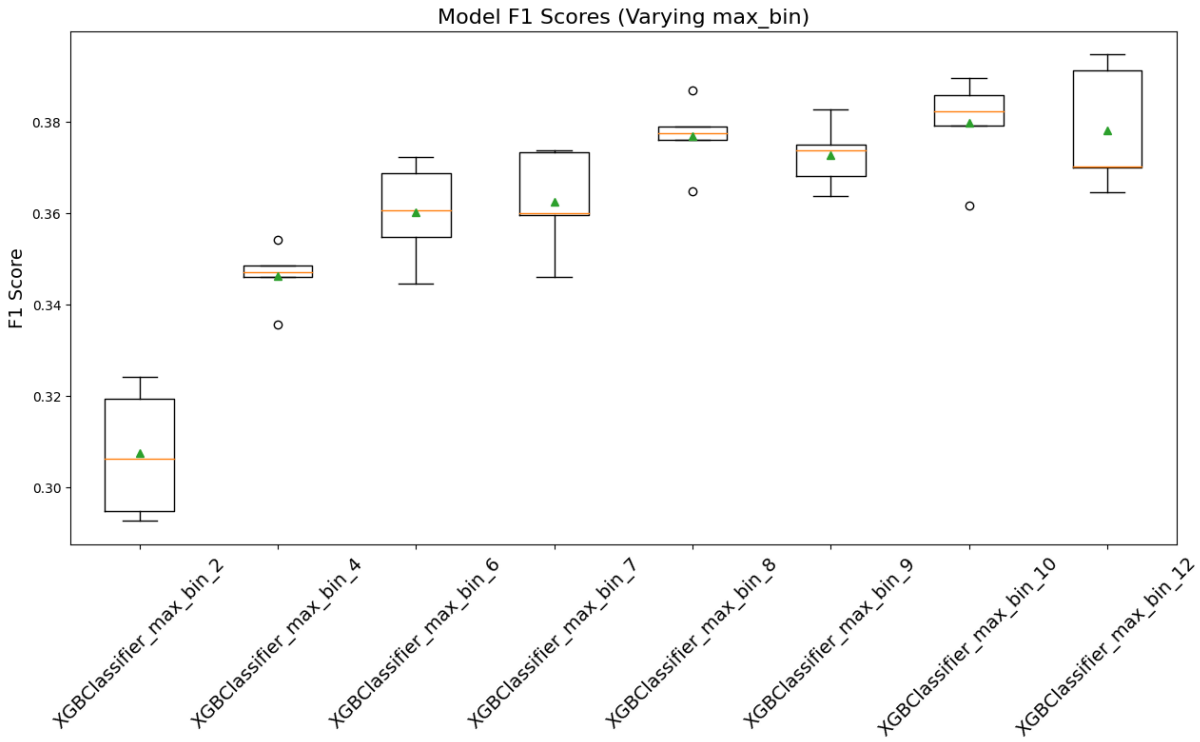


Figure 9 –XGB Max Bin

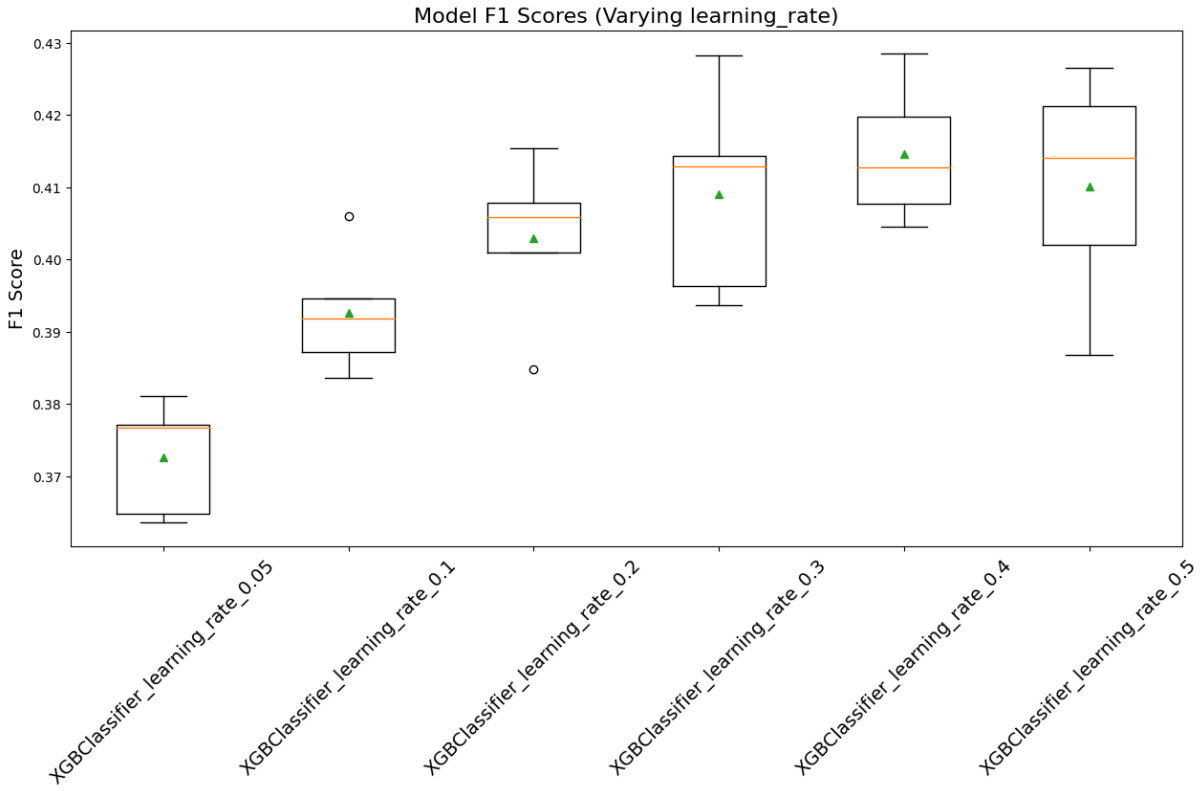


Figure 10 – XGB Learning Rate

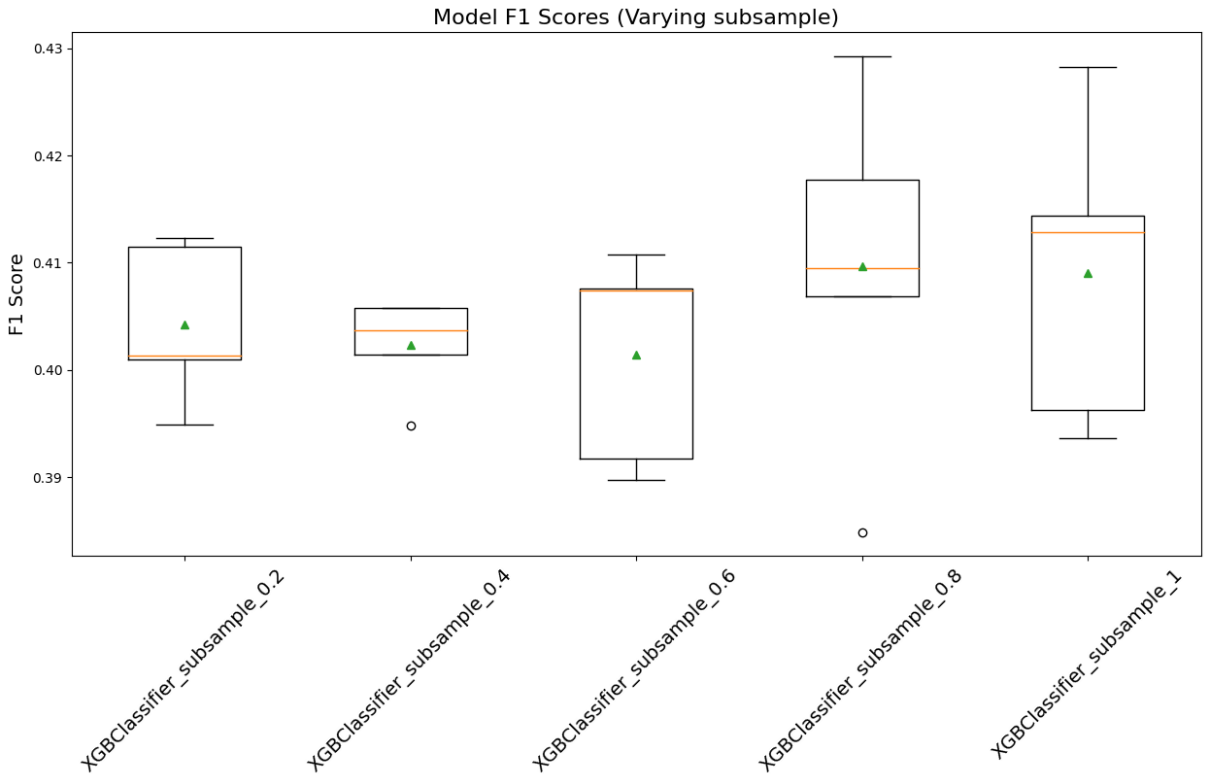


Figure 11 –XGB subsample

Figur

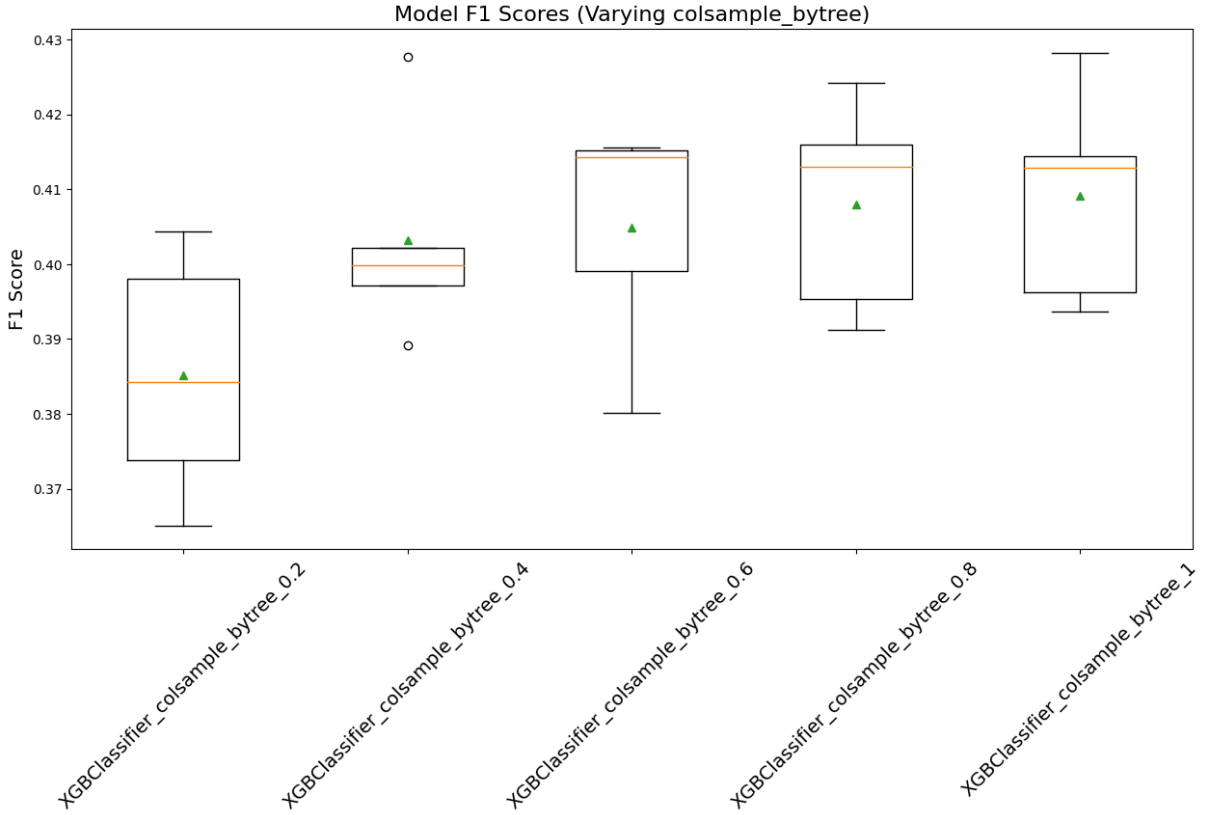


Figure 12 – XGB Colsample by tree

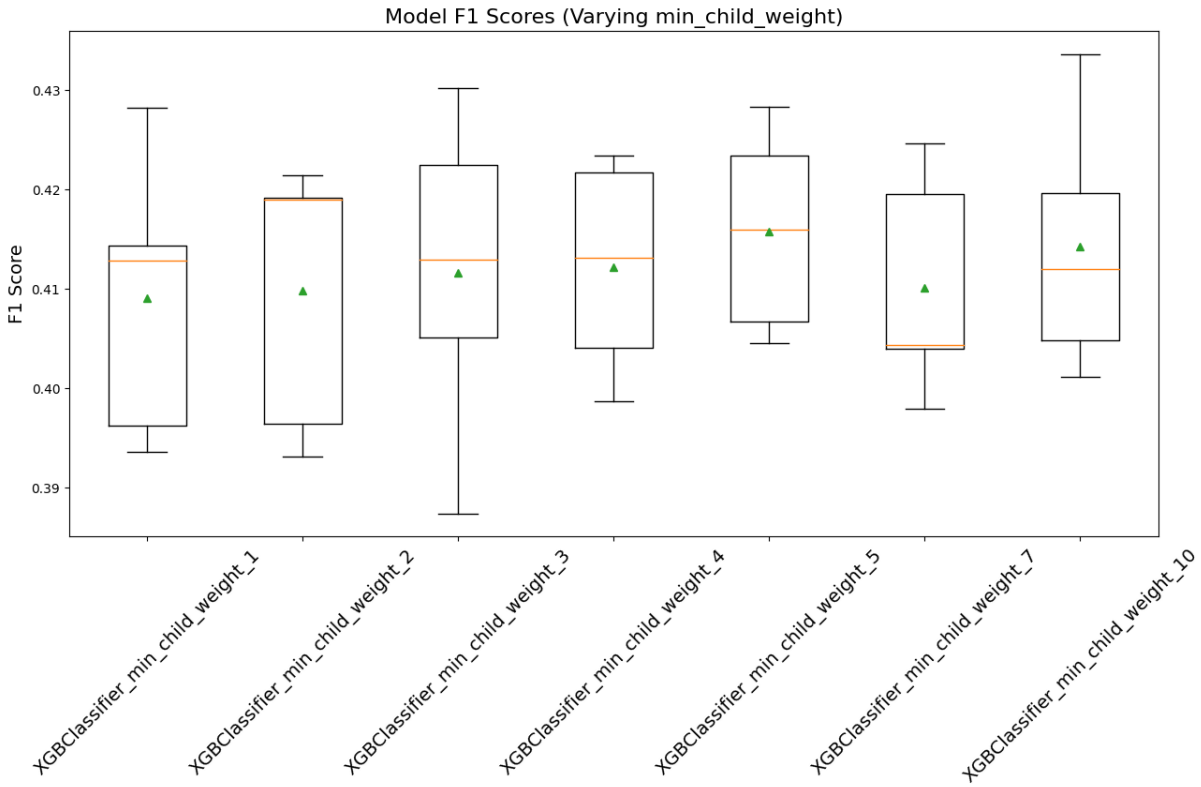


Figure 13 – XGB min child weight

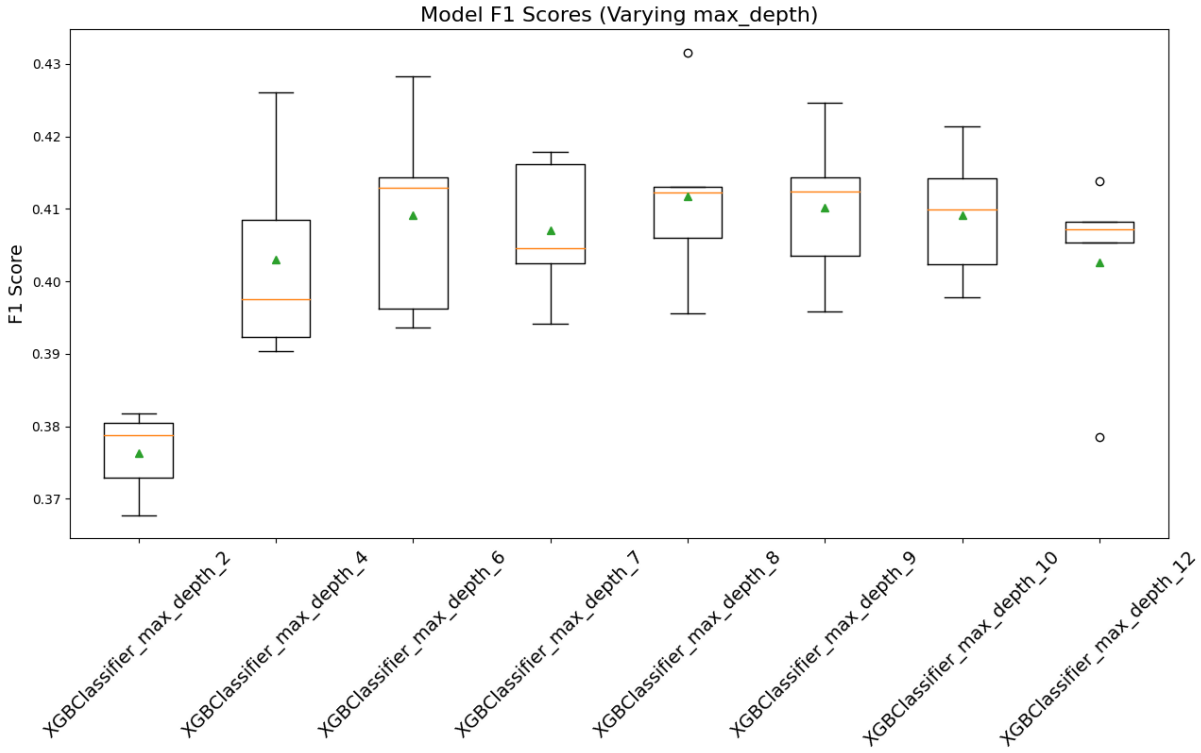


Figure 14 –XGB Max Depth

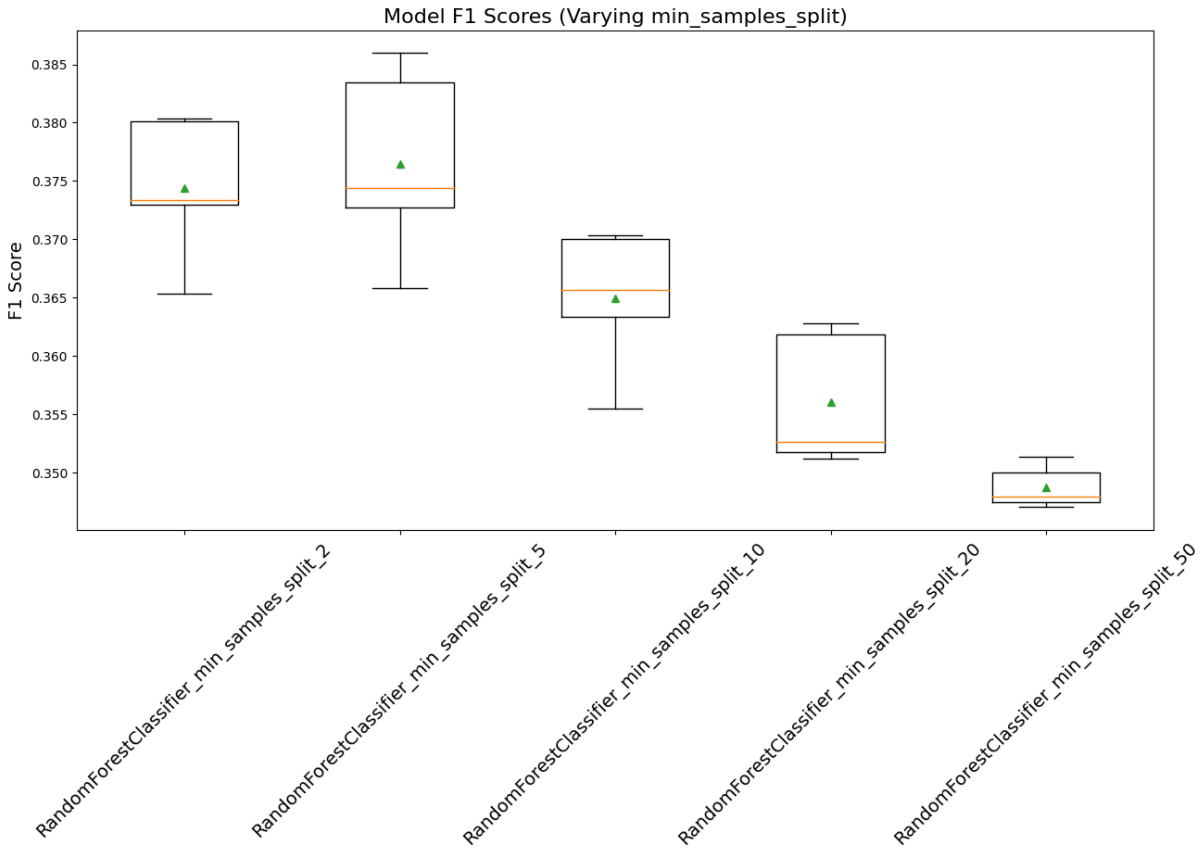


Figure 15 – RF Min Samples Split

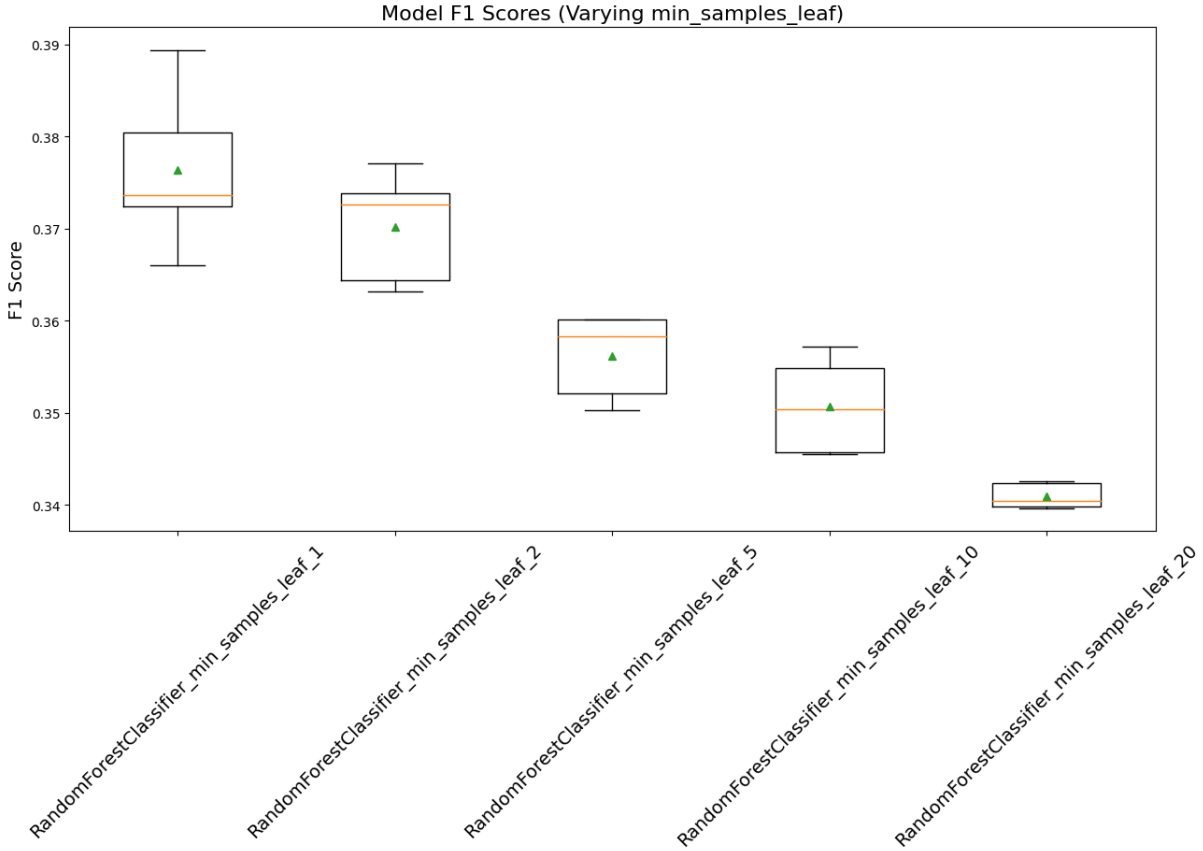


Figure 16 – RF Min Samples Leaf

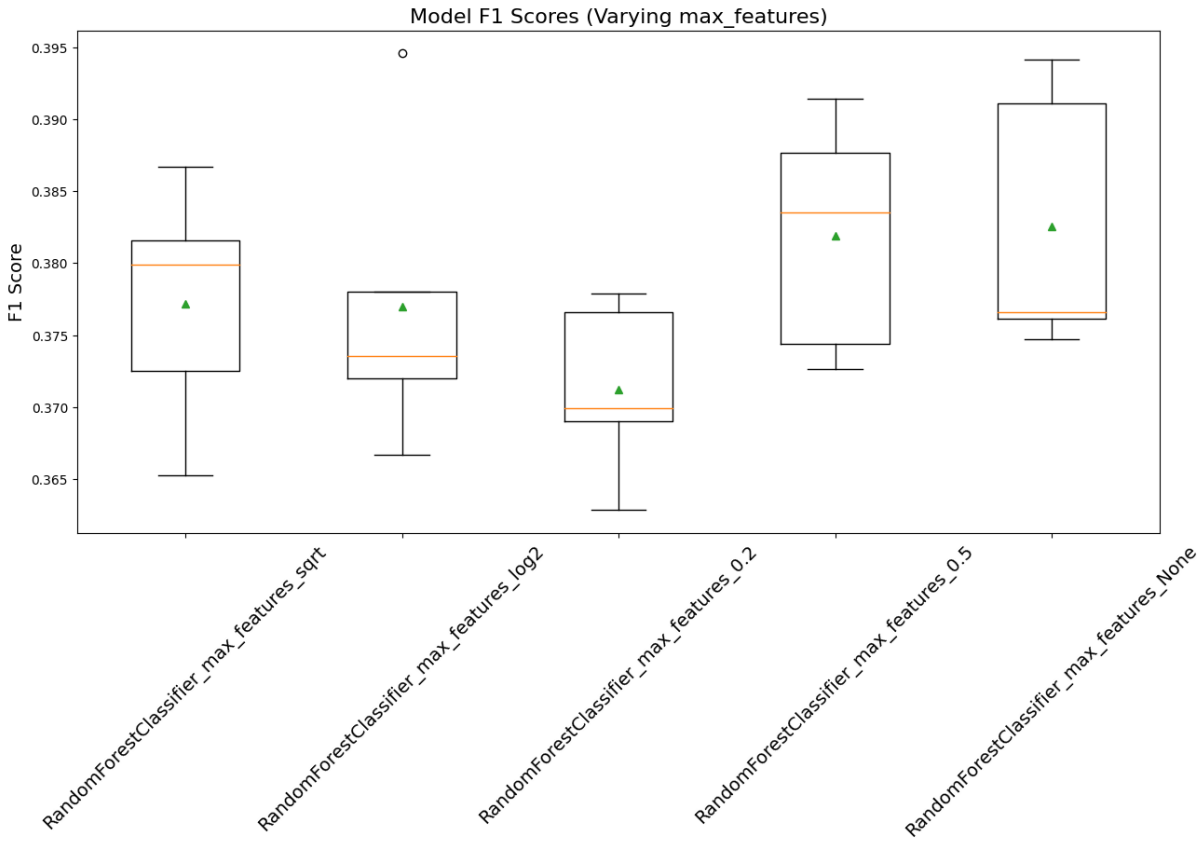


Figure 17 – RF Max Features

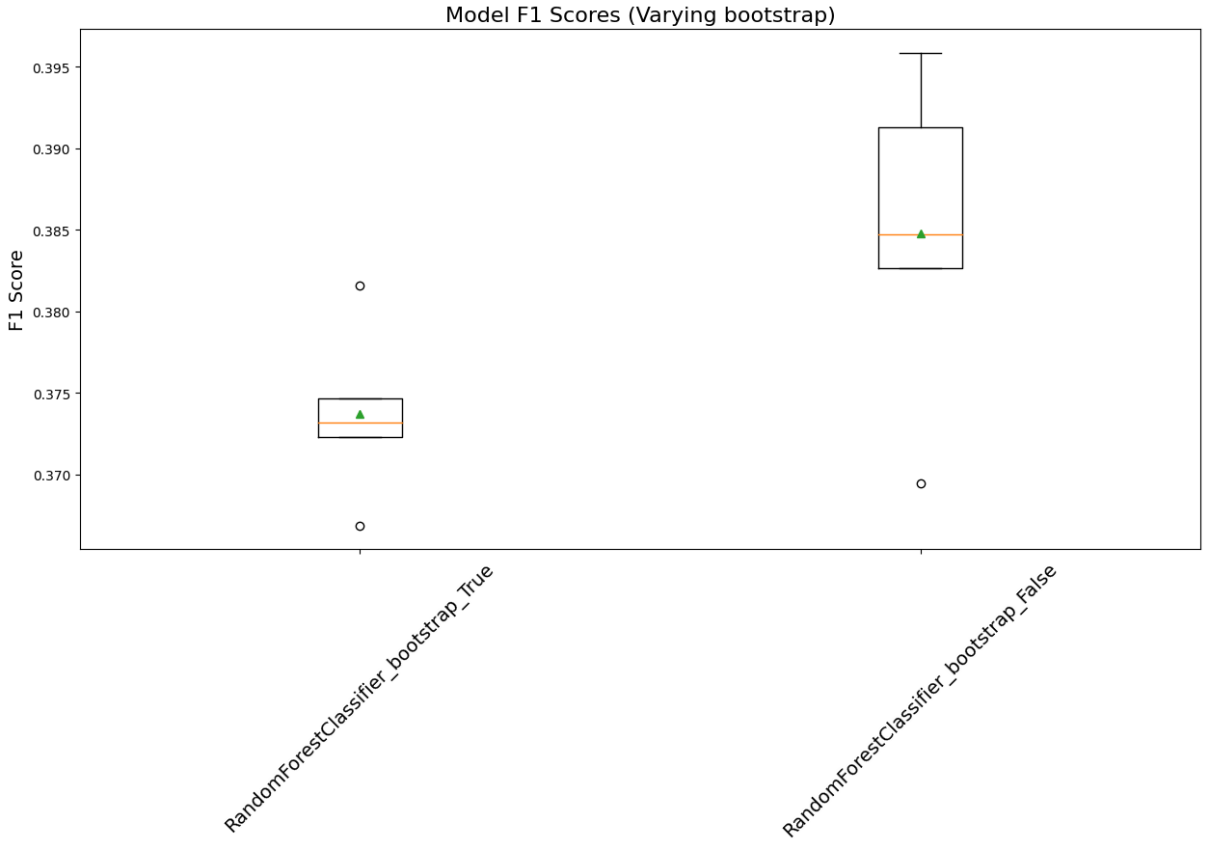


Figure 18 – RF Bootstrap

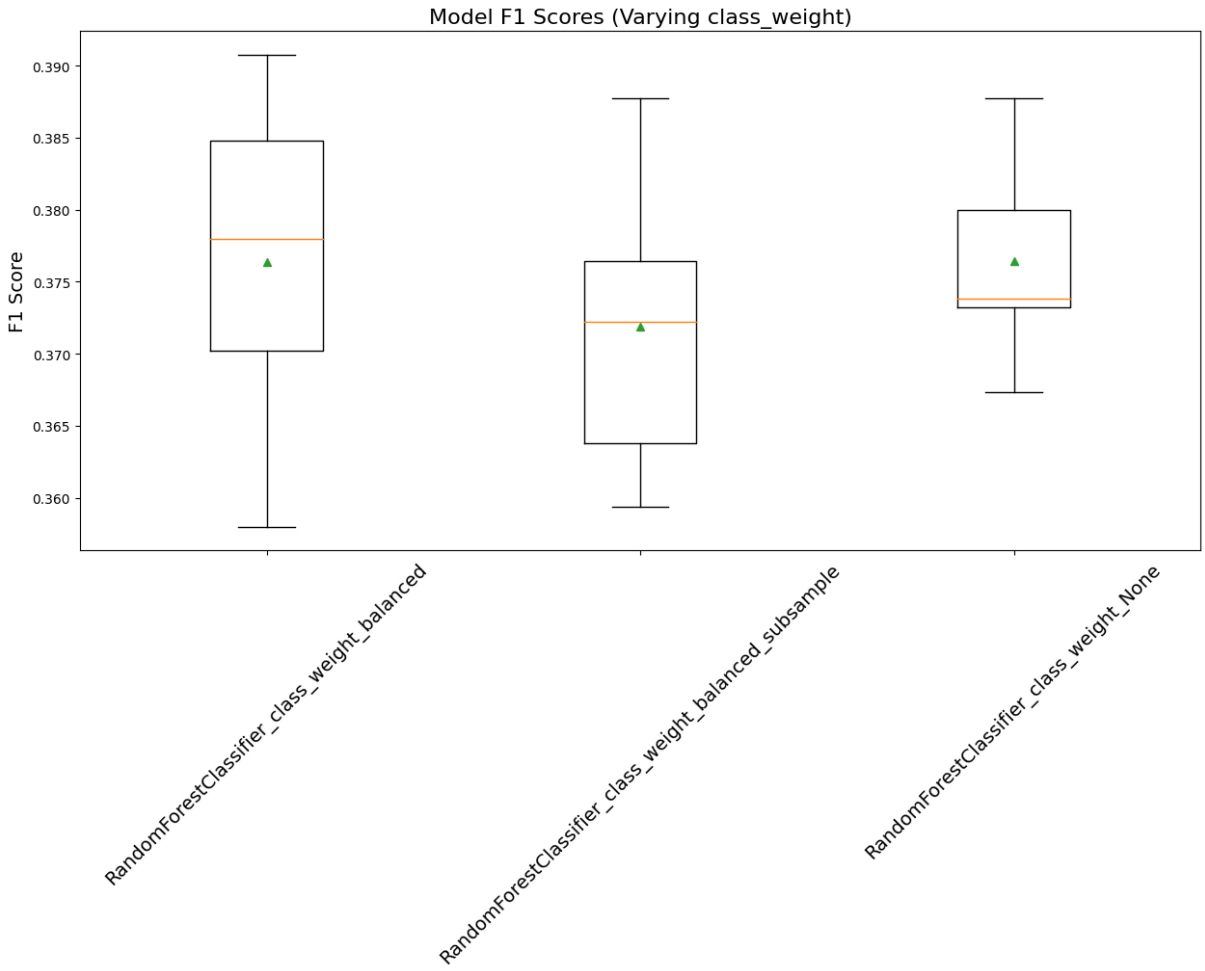


Figure 19 – Class weight

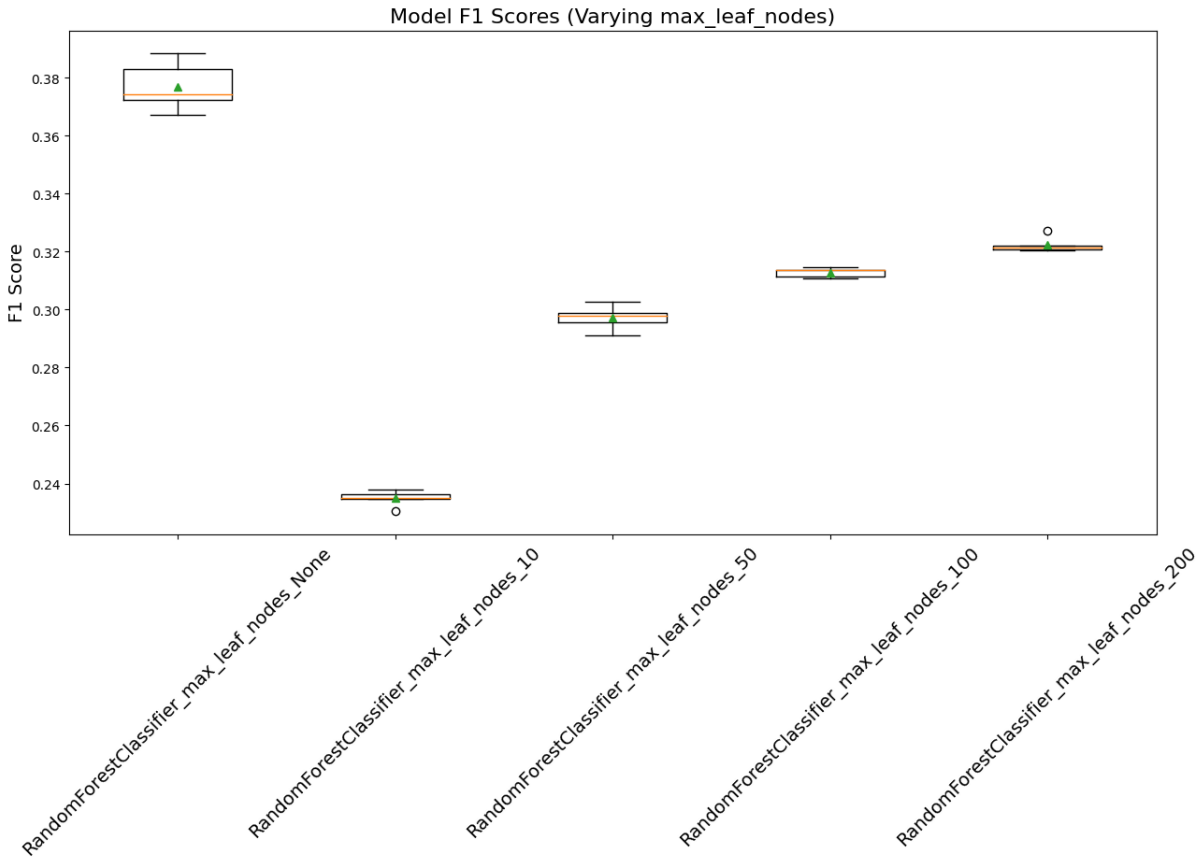


Figure 20 – RF Max Leaf Node

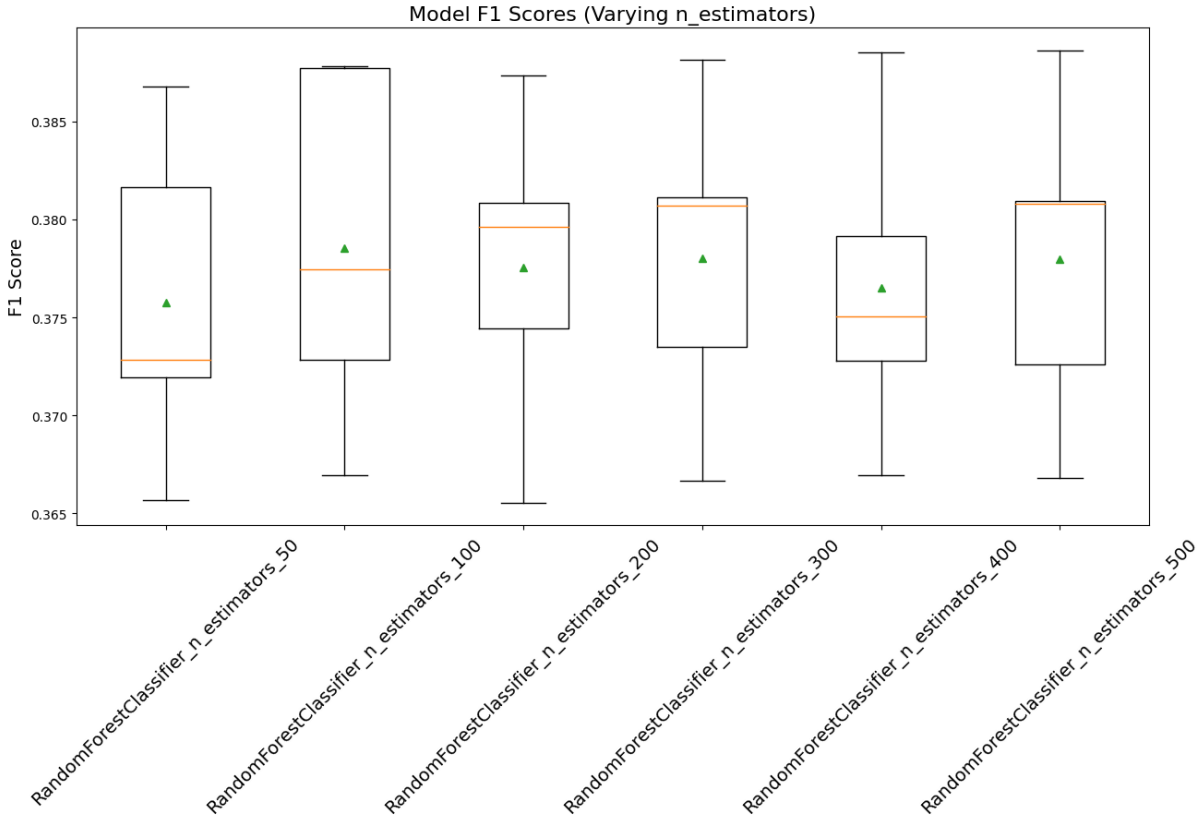


Figure 21 – RF N Estimators