Starcraft Rank Prediction Analysis

Overview

The purpose of this investigation was to build a supervised classification model to predict Starcraft League Rank. The dataset required minimal pre-processing, where the only major changes made were to replace missing values with K-Nearest Neighbors based imputations. We then explored the dataset by assessing the distributions of our dependent and independent features, specifically focusing on the independent features' relationships with Starcraft League Rank. We then selected 3 subsets of features using tree-based feature importance and tested these subsets on 6 different classification algorithms. Our best model was the Random Forest Classifier, which achieved an accuracy of 42.99%. Future steps for data collection are discussed at the end of the report.

1. Exploratory Data Analysis

The data used in this investigation contains Starcraft player performance data in ranked games. There are 3,395 records and 20 features. The goal is to predict "LeagueIndex", which is a categorical feature with values from 1 to 8 that represent each Starcraft League rank.

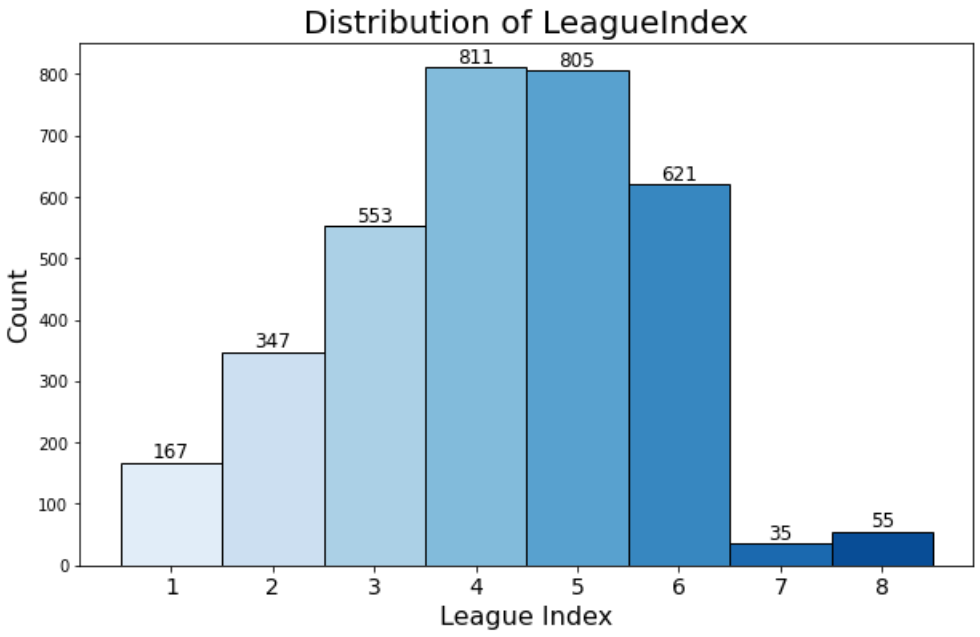

Figure 1. Histogram of LeagueIndex

We can see that the LeagueIndex feature has a relatively normal distribution where the majority of players have a League Index of 4 or 5, which is equivalent to Platinum and Diamond, respectively. One glaring problem is the imbalance of records in each index label, where there are only 35 records with a League Index of 7 and only 55 records with a League Index of 8. We also see this on the lower end of the spectrum where there are only 167 records with a League Index of 1. This imbalance may bias the model towards the majority classes and lead to poor generalization when the model is given new data. Next, we looked at the distributions of all the variables to be used as predictor features.

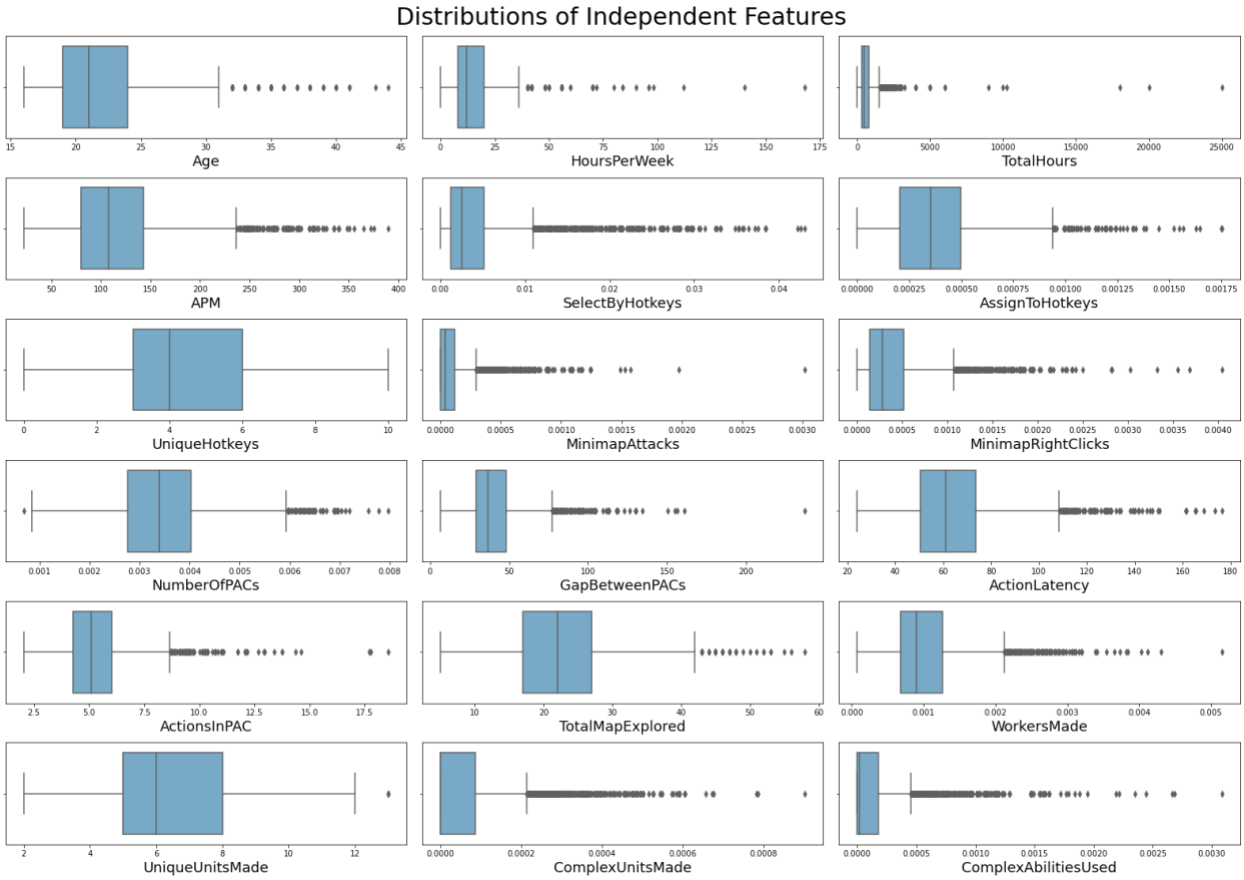

Figure 2. Boxplots of Independent Features

Many of our features have distributions that are skewed to the right, meaning that the averages of these features are much higher than their medians. While we could assume that these metrics just have uneven distributions, it's more likely that this is because of the relatively low number of records for high and low rank players. The majority of our metrics likely rise or fall as skill level increases and having so few records on both ends of the League Index range make these values appear as outliers. To explore this theory, we will look at the distributions of some of our features when categorized by League Index. The features of interest were selected by looking at their correlations with LeagueIndex. A heatmap of these correlations is attached in the appendix (Figure 3).

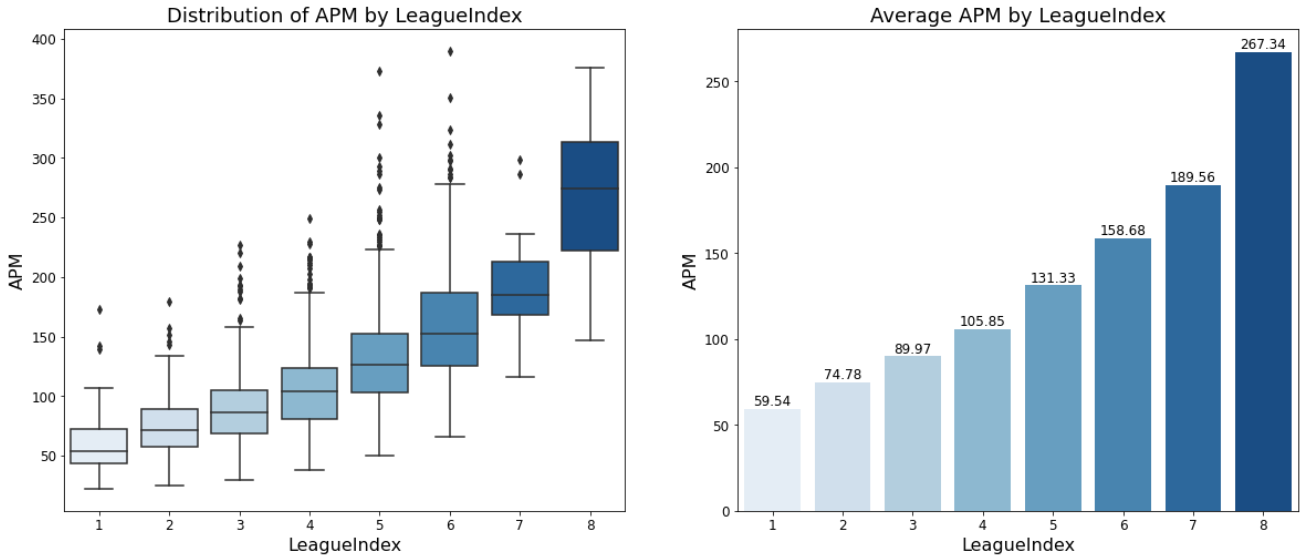

Figure 4. Distribution of APM by LeagueIndex

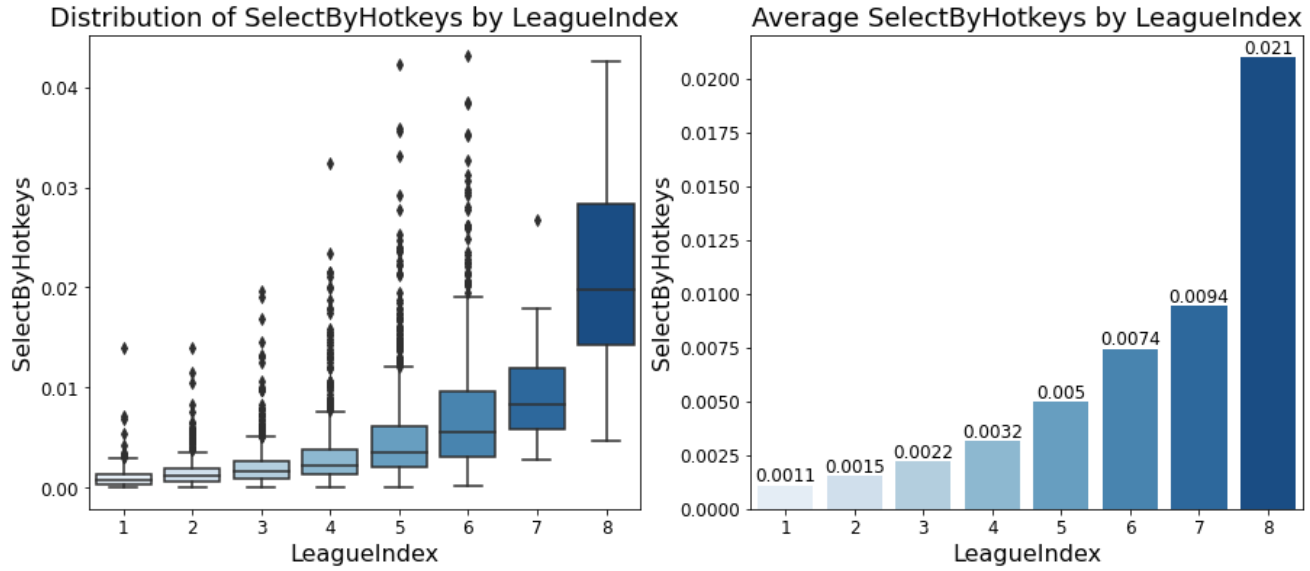

Figure 5. Distribution of SelectByHotkeys by LeagueIndex

Looking at the distributions of APM and SelectByHotkeys, we can see that the medians and averages for both steadily increase as LeagueIndex increases. Although each individual boxplot is still skewed to the right, we can see that the majority of the “outliers” in each boxplot are within the normal range for League Index 8. Now, we will look at the distributions of variables that decrease as League Index increases.

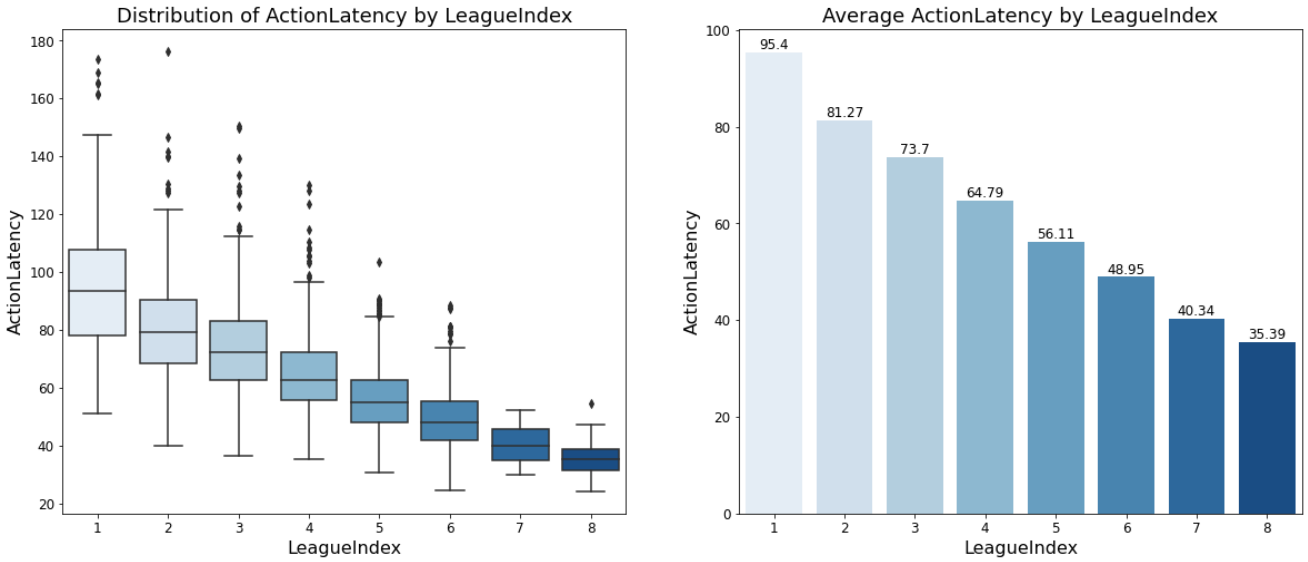

Figure 6. Distribution of ActionLatency by LeagueIndex

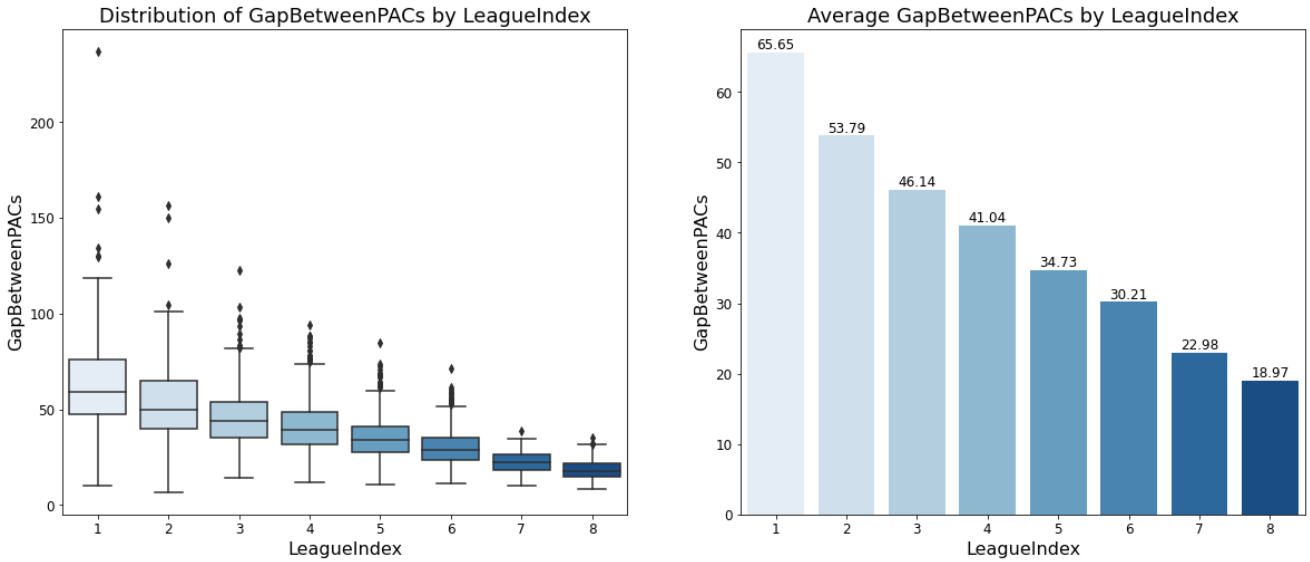

Figure 7. Distribution of GapBetweenPACs by LeagueIndex

Looking at the distributions of ActionLatency and GapBetweenPACs, we can see that these features follow a similar trend but reversed, where the outliers come from the lowest rank. Each of the individual distributions categorized by LeagueIndex follow a normal distribution much more closely which bolsters our initial thoughts that the skewed distributions come from class imbalances.

2. Feature Selection

In order to figure out which of our independent features would be the most important in predicting League Index, we utilized the feature importance technique available in tree-based algorithms like Random Forest. Feature importance refers to a technique that assigns a score to each variable based on how significant they are at predicting our target variable. This technique uses a metric called mean decrease in impurity (MDI) which tells us how much each feature helps reduce uncertainty when making a prediction on the data. The following graph shows the feature importance's of our variables from a Random Forest Classifier:

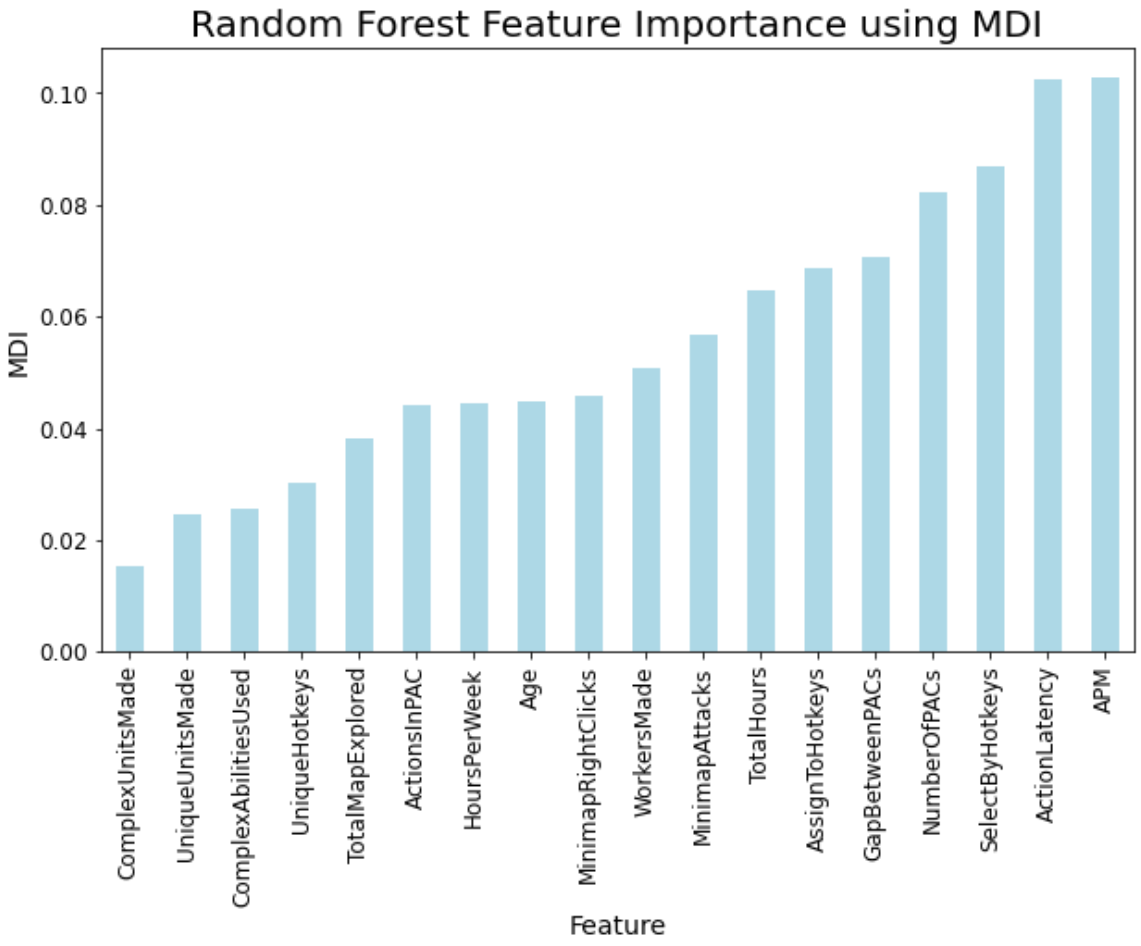

Figure 8. Feature Importance of Independent Fields

From the graph, we can see that APM and Action Latency appear to be the most important features, which makes sense given their strong correlations with the LeagueIndex field. Based on this feature importance ranking, we created subsets of the top 5 and top 10 variables to use for model testing. This was done to test how powerful our top variables are in

being able to predict League Index, and to test if some of the weaker variables introduced more noise into the data. All models were tested on 3 sets of data - the top 5 variable dataset, the top 10 variable dataset and the original dataset with all of the variables.

3. Model Testing

This project implemented six different classification algorithms – Logistic Regression, Decision Tree, Random Forest, Neural Network, K-Nearest Neighbors and LightGBM. For each model, we tuned the parameters using GridSearchCV, which is a technique that tests various combinations of parameters and picks the values that maximize accuracy. GridSearchCV also implements cross-validation, which is a technique that helps us detect overfitting and gives insights on how the model will generalize to unseen data. As previously mentioned, each model was tested on the 3 subsets of data (top 5 variables, top 10 variables, all variables). All values were standardized using Robust Scaler, which is a technique that scales the data according to the Interquartile Range for each variable. The data was divided into training and testing datasets using a 75/25 split. The following table shows the highest accuracies of each algorithm and the dataset it was achieved on.

	Accuracy	**Data Subset**
Logistic Regression	35.92%	Top 10 Variables
Decision Tree	31.9%	Top 10 Variables
Random Forest	42.99%	All Variables
Neural Network	42.16%	Top 10 Variables
K-Nearest Neighbors	38.39%	All Variables
LightGBM	38.28%	All Variables

Table 1. Best Model Comparison for Each Algorithm

Our best performing algorithm was Random Forest, which achieved an accuracy of 42.99% using all of our independent variables. Neural network was close behind at a 42.16% accuracy and used the subset of the top 10 variables. The Decision Tree classifier struggled to predict League Index, achieving a meager 31.9% accuracy. None of the top performing models used the subset of the top 5 variables, which indicates that our most important variables are not strong enough on their own to predict League Index. Using our best-performing Random Forest model, we made predictions on the testing set. The following heatmap shows the confusion matrix from the predictions on the testing set.

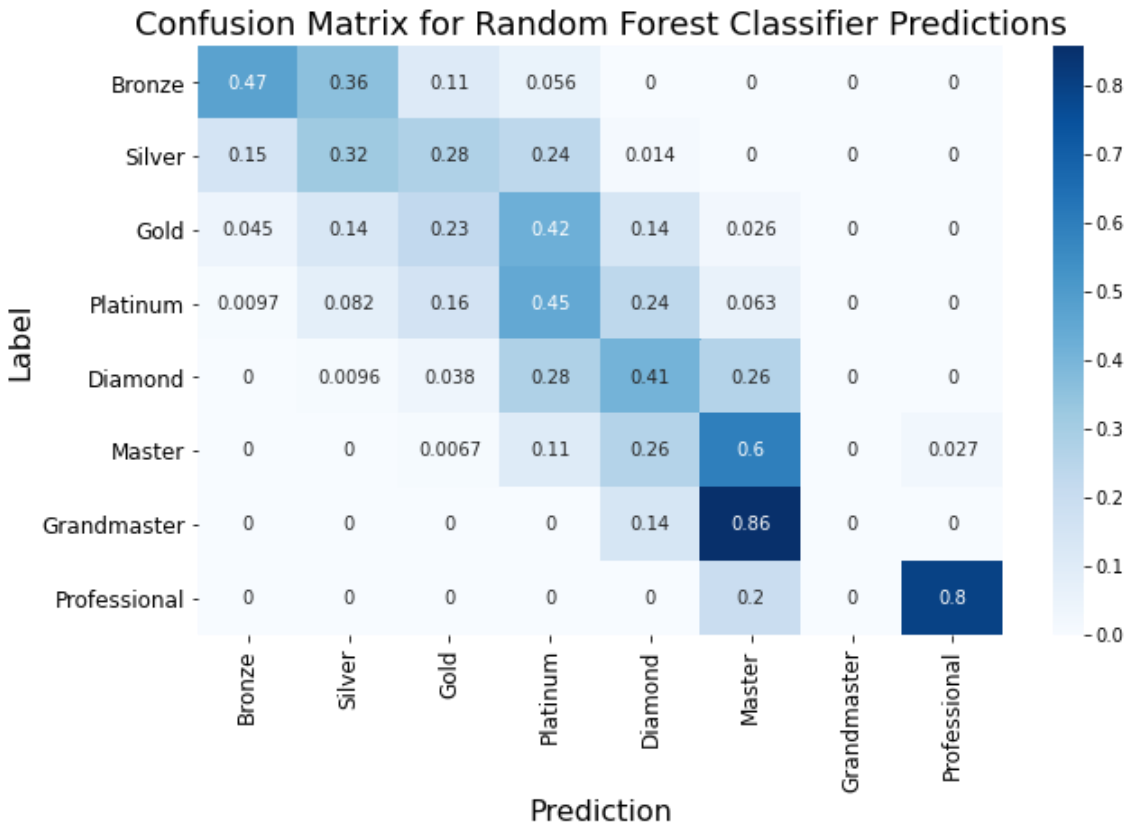

Figure 9. Confusion Matrix for Random Forest Classifier Predictions

The figure above shows the distributions of predictions that the model made for each League Rank. For example, out of all the professional players in the testing set, our model predicted 80% of professional players as professional and 20% as Master. Surprisingly, the model achieved the best accuracy on the professional rank, correctly predicting 80% of the professional rank records. On the other hand, the model incorrectly predicted every record with the Grandmaster rank at a 0% accuracy. This is not surprising given that Grandmaster had the lowest frequency in the dataset having only 35 records. The model also struggled to accurately predict records with the Gold rank, only predicting 23% of those labels correctly. Out of the records with a rank of gold, the model predicted 42% as platinum, which likely shows that there are very minimal differences in characteristics between gold and platinum players.

Overall, we were unable to achieve satisfactory results and cannot accurately predict League Index based on the given dataset. While testing other pre-processing techniques and models may slightly increase accuracy, the results show that this dataset does not capture enough information to accurately predict Starcraft rank. However, from the confusion matrix, we can see that our model was able to accurately predict within 1 rank of the label. For example, for records that had a League Rank of Diamond, 95% of the predictions fell within Platinum, Diamond and Master. While the implications would be different, grouping the League Ranks into categories would achieve a substantially higher accuracy.

4. Future Steps

In terms of future data collection, there are two main topics that should be focused on. First, more data needs to be collected on the minority classes, or ranks 1, 7 and 8. The significant class imbalance discussed at the beginning of the report likely played a role in the low accuracy. Having more data on high-rank and professional level players is especially important as a potential use for this model could be to scout high-ranked players that have professional level mechanics.

In addition to this, data collection should focus on finding other in-game metrics or statistics that have a strong correlation with League Index. From our feature importance analysis, we know that Actions Per Minute was the most powerful variable. One area of interest is to investigate other metrics that track a player's ability to process information or multitask, especially because Starcraft is notorious for overwhelming new players with information overload. While these metrics may not be obtainable, statistics related to reaction time, camera movement and defense may carry the same strength as APM. Personally, the aspect I struggled with the most when I played Starcraft II was resource management, and there were no features that directly represented resource management in our dataset. The following list contains in-game features I am interested in based on my analysis:

- Whether the player won the game
- Average win rate
- Game duration
- Map Selection
- In-Game Race
- Resources Collected
- Resources Used
- Unused Resources
- Resource Collection Rate
- Player Region

Besides the features related to resources, the remaining features focus on gaining a high-level overview of the player's attributes. The dataset contained many micro-level features that help us understand their mechanics but does not include features that give us a well-rounded image of the player. It is nearly impossible to say if these features would improve accuracy, but having more options allows us to expand upon the feature engineering and feature selection process. While there are many technical aspects that could have been improved upon, the overall low accuracy indicates that we need better features and more data to accurately predict Starcraft League Rank.

A. Appendix

Table 1: Best Model Comparison for Each Algorithm

	Accuracy	Data Subset
Logistic Regression	35.92%	Top 10 Variables
Decision Tree	31.9%	Top 10 Variables
Random Forest	42.99%	All Variables
Neural Network	42.16%	Top 10 Variables
K-Nearest Neighbors	38.39%	All Variables
LightGBM	38.28%	All Variables

Table 2: Results from All Algorithms

	Accuracy	Data Subset
Logistic Regression	30.98%	Top 5 Variables
Logistic Regression	35.92%	Top 10 Variables
Logistic Regression	34.63%	All Variables
Decision Tree	28.97%	Top 5 Variables
Decision Tree	30.51%	Top 10 Variables
Decision Tree	31.21%	All Variables
Random Forest	35.45%	Top 5 Variables
Random Forest	41.82%	Top 10 Variables
Random Forest	42.99%	All Variables
Neural Network	36.75%	Top 5 Variables
Neural Network	42.16%	Top 10 Variables
Neural Network	41.58%	All Variables
K-Nearest Neighbors	37.57%	Top 5 Variables
K-Nearest Neighbors	38.16%	Top 10 Variables
K-Nearest Neighbors	38.40%	All Variables
LightGBM	33.80%	Top 5 Variables
LightGBM	36.75%	Top 10 Variables
LightGBM	38.28%	All Variables

Figure 1: Distribution of League Index Feature

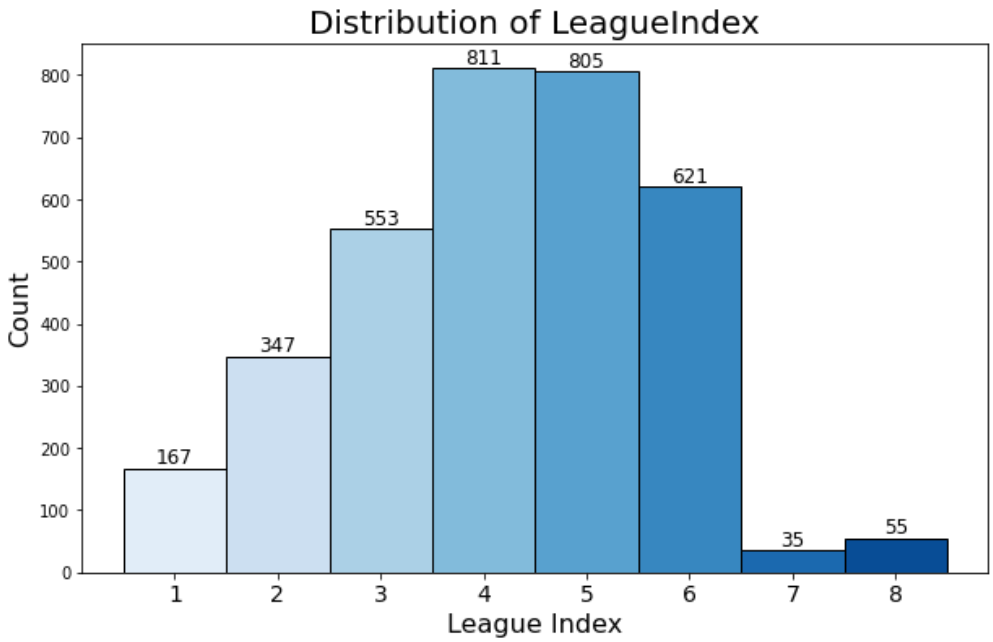

Figure 2: Distributions of All Independent Features

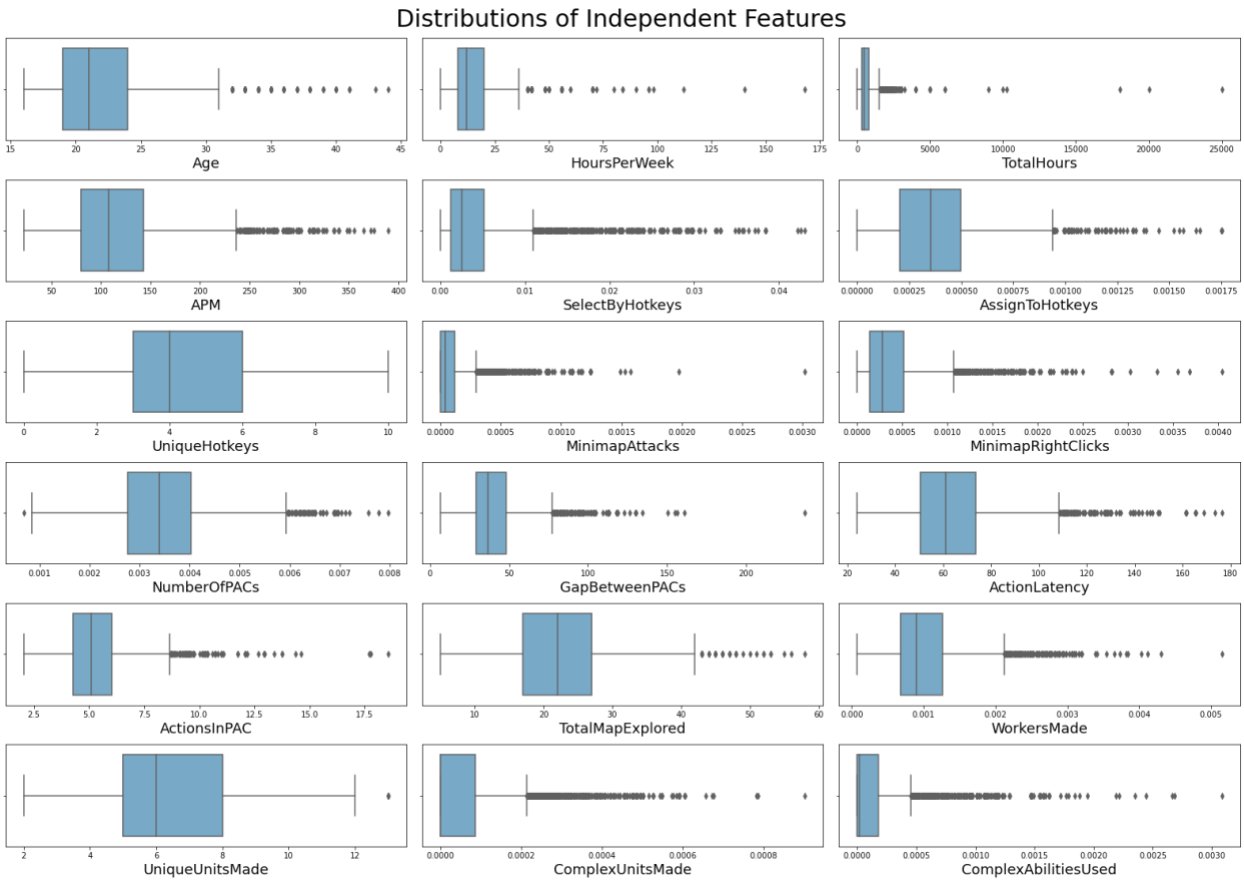

Figure 3: Heatmap of Correlation Matrix

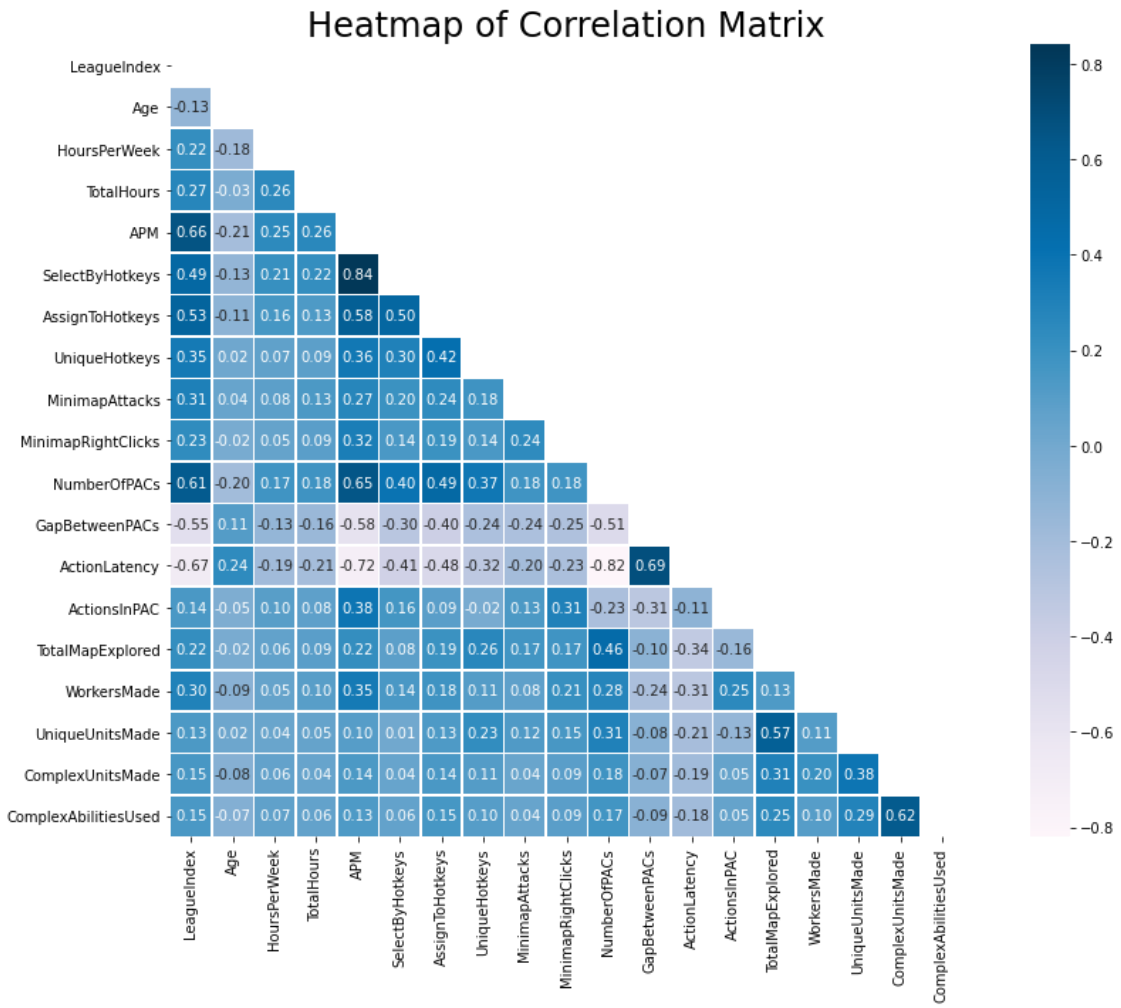

Figure 4: Distribution of APM by LeagueIndex

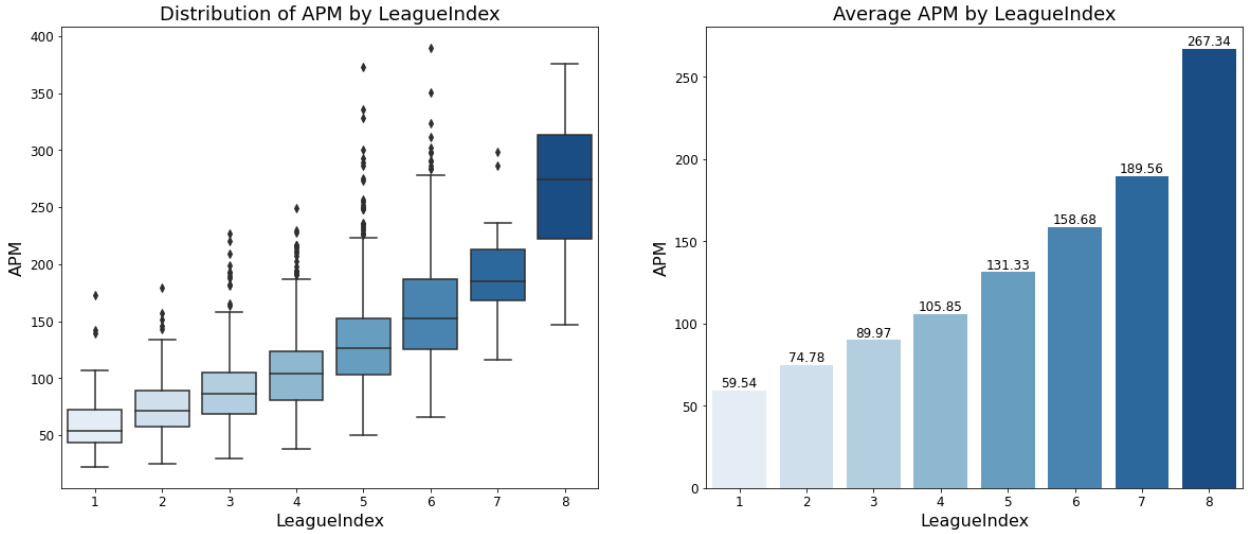

Figure 5: Distribution of SelectByHotkeys by LeagueIndex

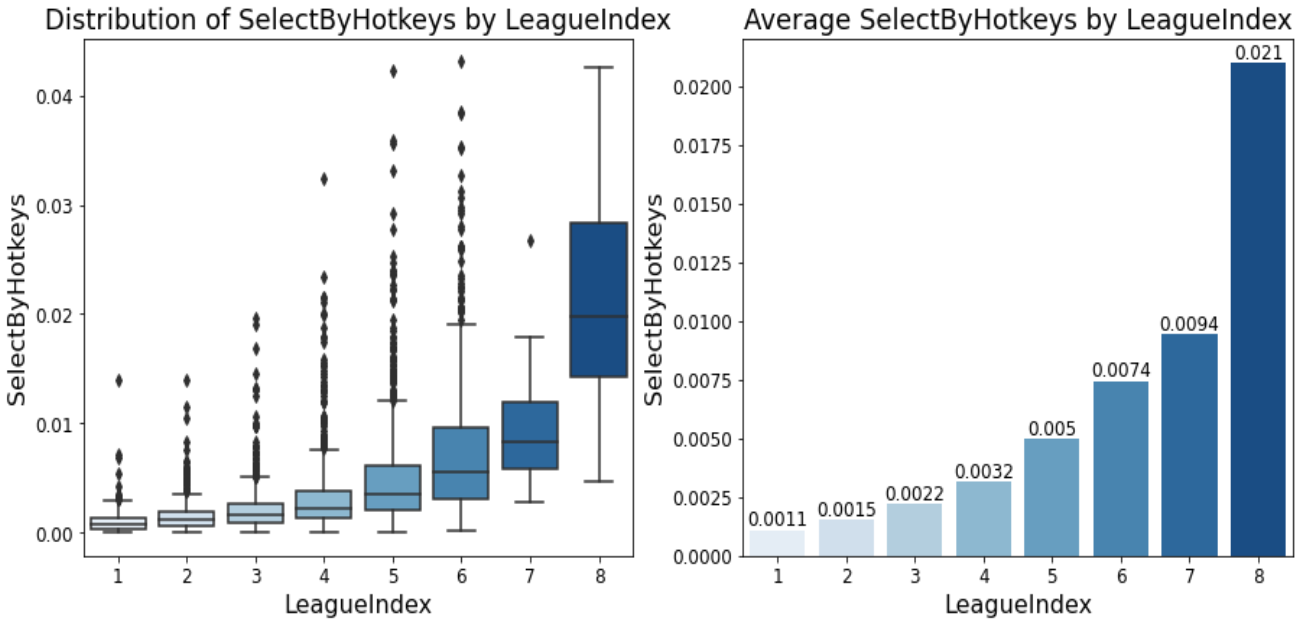

Figure 6: Distribution of ActionLatency by LeagueIndex

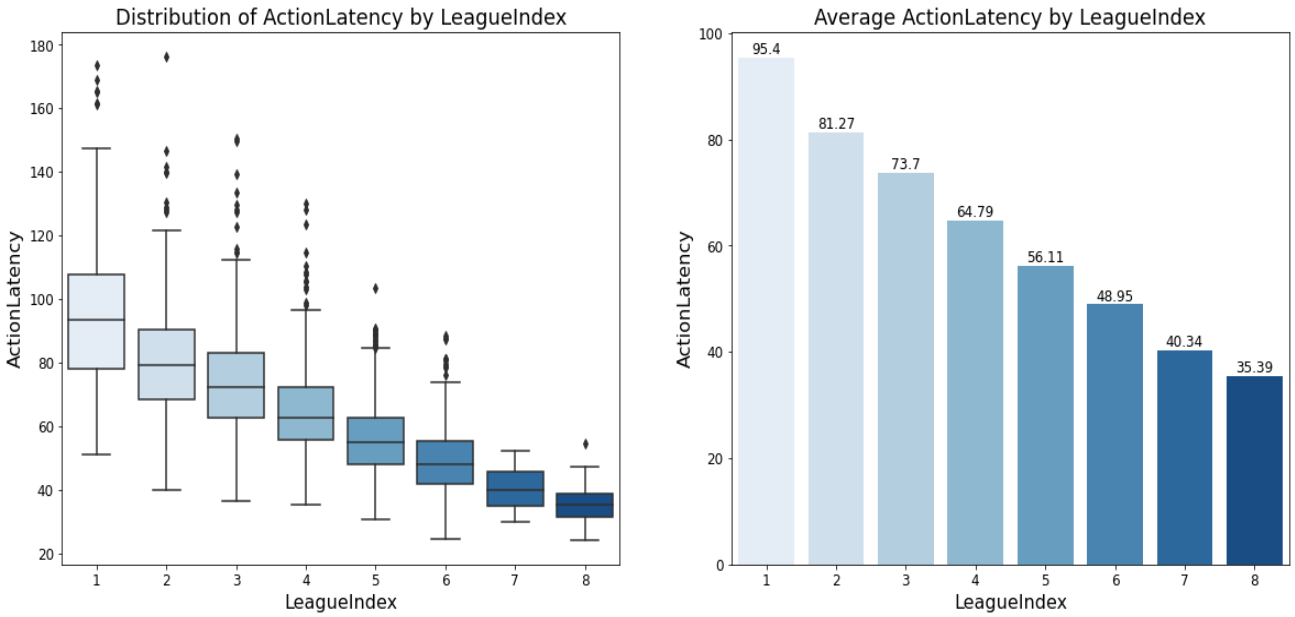

Figure 7: Distribution of GapBetweenPACs by LeagueIndex

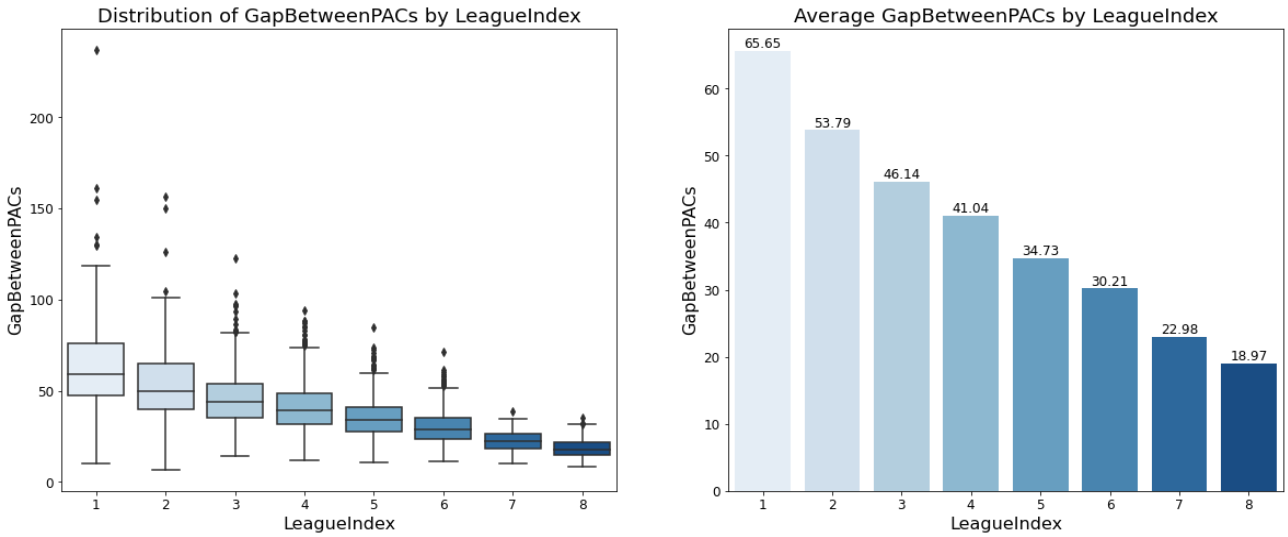

Figure 8: Feature Importance of Independent Fields Using Random Forest

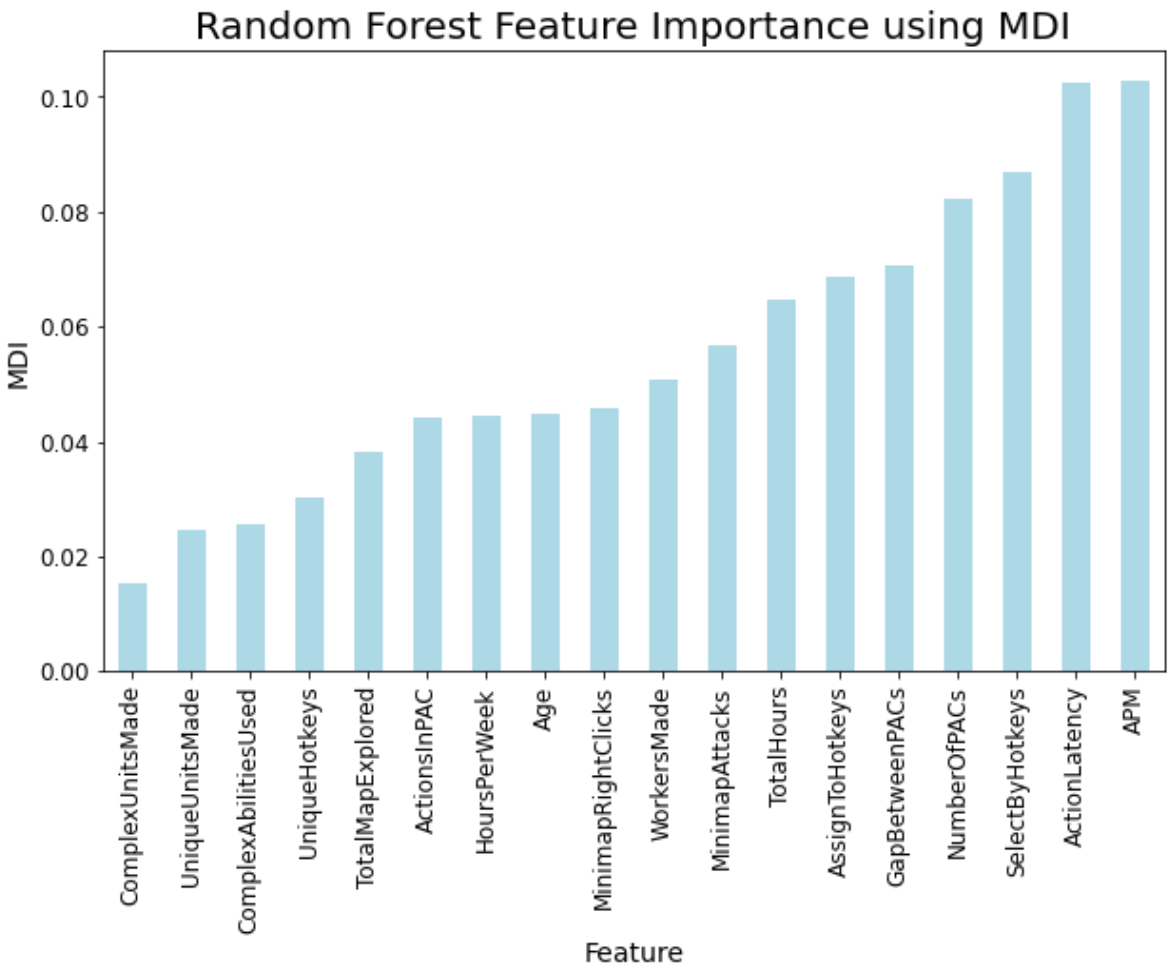

Figure 9: Confusion Matrix for Random Forest Classifier Predictions