# Predicting Rankings of PGA Golf Players

Reno Domel, Michael Lee, Gavin Jakubik, Jack Rundle, Elijah Nordan

University of Notre Dame

**ACM Reference format:**

Reno Domel, Michael Lee, Gavin Jakubik, Jack Rundle, Elijah Nordan. 2021. Predicting Outcomes of Golf Tournaments. In *Proceedings of ACM conference. ACM, New York, NY, USA, 2 pages.*
https://doi.org/10.1145/1234567890

## 1 INTRODUCTION

### 1.1 Motivation

The Official World Golf Ranking has been around since 1986 and ranks the best male golfers around the world. It is calculated based on the player's performance in tournaments over a rolling two-year span. It has real-world implications as it can determine which golfers are invited into tournaments and may also increase their profitability [1]. In this project, we are attempting to predict the Official World Golf Ranking of the top golfers. This can be used both for betting purposes and as a helpful guide for potential sponsors to determine what golfers may perform well in the future.

### 1.2 Precise Problem Definition

As mentioned above, the Official World Golf Ranking is calculated by the player's performance in tournaments over the previous two years. We aim to predict this ranking based upon the stats of each golfer in the year prior to the two year period, with each data object being a specific player during a specific year. In other words, when trying to predict the ranking of a golfer at the end of year x, the features of each data object are the golfer statistics for the year x-2. The label is the Official World Golf Ranking of the golfer in year x (two years later). Since the ranking is based on two years of performance, the stats for the year x-1 are not used in the prediction because we did not want to include any statistics that would be directly involved in calculating the ranking.

To approach this problem, we wanted to be able to use several classification models, and additionally, we were less concerned with predicting the ranking to the exact right number. Thus, we decided to cluster the Official World Golf Ranking, which we are treating as the label, into a set of 6 different ranges (1-20 = 'A', 21-40 = 'B', 41-60 = 'C', 61-100 = 'D', 101-200 = 'E', 201-300 = 'F'). These 6 classes were chosen because they are the groups we considered to be of interest and would generally be what matters to sponsors and bettors to separate elite versus good versus mediocre professional golfers.

With this label cluster, the problem becomes how does the available data we can find on professional golf players predict their performance and golf ranking in the future?

Here is a visualization of the problem we are solving:

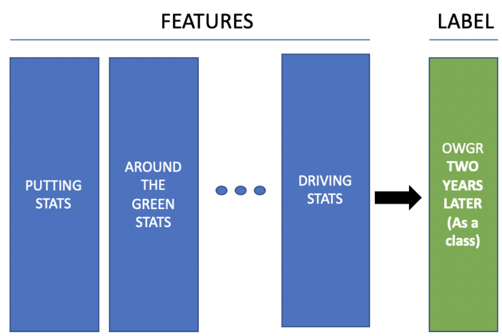


**Figure 1. Problem Visualized**

## 2 RELATED WORK

### 2.1 Related Work

Wiseman has a similar idea of using machine learning in order to predict golf-related outcomes for the purpose of betting. However, instead of trying to predict the Official World Golf Ranking of professional golf players, he sets out to predict the outcome of a singular golf event, given the results of the first round. Simple linear regression, correlation analysis, and multiple linear regression were used to determine the relative importance of data features. Then, the most important data features were used for the final machine learning model [2].

Vhras also makes use of data from the PGA Tour website, but for a very different purpose: predicting individual statistics based off of the score and vice-versa. For example, with data attached to a specific player, one can enter the desired score in Vhras's model and it would output the amount of putts, fairways, and greens of the course. This project can be useful for both casual golfers determining how their skill level matches up to how they are predicted to perform at different courses. It also only uses linear regression [3].

### 2.2 Our methods

Similar to these other projects, we make use of the PGA Tour website to scrape for data. However, we differ in that we plan on determining the Official World Golf Ranking, with the use of Gaussian Naive Bayes, Kernel SVM, Neural Network with KNN, KNN (3 neighbors), and KNN (5 neighbors). Also, our project has implications for being used for potential sponsors to determine the future performance of golf players.

## 3 DATA/EXPERIMENT SETTINGS

### 3.1 Data Sources

To attempt to solve our problem, we first collected raw data to feed into our models. All of our data is scraped directly from the PGA Tour website. We modified an existing scraper to get the statistics we needed from the years we wanted to examine. The scraper crawls through several tables across the PGA Tour domain, returns a variety of stats broken down by player for a given year, then loads them into several CSV files to be manipulated as necessary. The actual data collected will be discussed further in Section 3.3.

### 3.2 Data Processing

Our next step was processing this data into a form that would be usable to build a model. First, we prepared a script to merge the different golf stats (i.e. driving vs. putting) from each table (using player name) then combined that information to get data for the available players for a given year. Then, we combined the different years into one large CSV file. All of the features used were *numerical*, so after obtaining this data, we used MinMax scaling on these features for all models, except the Gaussian Naive Bayes model (reason explained in Section 3.5). In addition, because all features used were numerical, we did not use any encoding method. The full list of 13 features is shown in Section 3.3.

Also, given the classes we defined for our prediction, we had to convert raw rankings into a class, and we removed all players whose future rank is greater than 300 since they do not belong to any groups. Note that this does not mean that we removed players whose current ranking (before the two year period) is currently greater than 300.

Lastly, because of the way we decided how the ranks (A-F) should be established, the data is not fully balanced, with certain ranks appearing more than others.

| Label | Number of Objects with Label |
|---|---|
| A | 247 (14.6%) |
| B | 200 (11.9%) |
| C | 172 (10.2%) |
| D | 268 (15.9%) |
| E | 480 (28.5%) |
| F | 316 (18.8%) |

**Table 1. Label Distribution of Objects**

However, we decided not to use stratified sampling because we hoped that this would make our models bias the classes it needed to predict more often. In addition, we wanted to use as many data objects as we could.

### 3.3 Data Object/Features

Each data object corresponds to a players' performance for a certain year. This object has information on player name, various strokes gained statistics, driving distance and accuracy, green in regulation percentage, etc. All of these features are numerical. The corresponding label is their official world ranking category (A-F) two years after the current year. In our dataset, we have about 1680 data objects, spanning from 2004 to 2019.

The complete list of 13 input features used are: current average of overall golf rating points, average driving distance, amount of times the golfer placed top ten in a tournament in the current year, current year, number of rounds played in the year, greens in regulation percentage, strokes gained putting average, driving accuracy, scrambling percentage, adjusted scoring average, strokes gained off the tee average, strokes gained approaching the green average, and strokes gained around the green average.

### 3.4 Feature Correlation Analysis

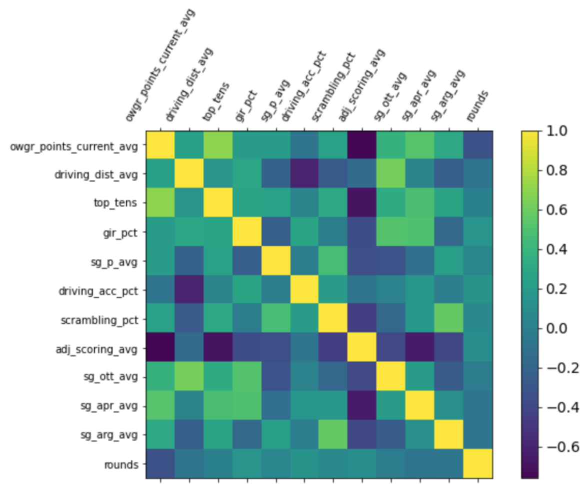


**Figure 2. Feature Correlation Matrix**

To detect any potentially redundant features, we performed correlation analysis. Shown above is a heatmap of correlation coefficients for each feature. The pair of features with the strongest correlation were scoring average and current world golf points average, as well as scoring average and number of top tens. That said, these pairs did not have high enough correlation to deem it unnecessary to remove a feature.

### 3.5 Models Used

For our naive model baselines, we apply two main heuristics. In the first approach, we randomly guess a rank category for each player, where each category has probability directly correlated to the size of accepted ranks in that category. In the second approach, we simply reuse a player's current rank as their predicted rank, categorizing the rank as appropriate. As previously mentioned, the only players included are players whose rankings in two years in the future is less than or equal to 300. Therefore, if the second naive model was a given player whose current rank was greater than 300, it will predict their future ranking to be F (i.e. in 201-300) because it knows they aren't likely to be higher than that.

The actual models we built to beat such naive models were a Gaussian Naive Bayes model, KNN using 3 neighbors, KNN using 5 neighbors, a Neural Network with KNN, and Kernel SVM. Note that the Gaussian Naive Bayes model did not use MinMax scaling on the numerical features. The reason for this was that the model performed better without the features being scaled.

### 3.6 Neural Network with KNN Explained

This was a custom ensemble method we built. We decided we wanted to train the neural network by having it guess a raw rank (a number) so it could use regression. Therefore, unlike the other models, this model uses regression in order to perform classification. This was successful, but in using regression, it struggled to identify players in class F. So, our solution was to use KNN (which was successful in identifying class F) to assist. Therefore, the Neural Network with KNN first uses a neural network to predict a raw rank and then it converts that raw rank to a predicted class. After doing so, it checks to see if KNN predicted class F. If this is the case, the model goes with class F, otherwise it goes with the neural network's prediction.

In addition, using trial and error as well as our knowledge of golf, we removed some features which made our neural network perform worse. Those features were current year, number of rounds played in the year, greens in regulation percentage, scrambling percentage, strokes gained approaching the green average, and strokes gained around the green average.

### 3.7 Validation Methods

In training our model, we utilized 5-fold cross validation, via the sklearn module, with approximately 20% of data used for testing in every fold. This allowed us to use every object as a testing instance at least once, and it allows for more statistically significant results.

## 4 EVALUATION METRICS

Given that this problem consists of doing multi-class classification, the results produced can be really dense and data-heavy. The metrics used to help us understand our results are:

- Accuracy - Averaged over 5 folds, it is simply the number of correct classifications divided by the number of testing instances
- F1 Micro - Averaged over 5 folds, based on precision and recall (on an instance level)
- F1 Macro - Averaged over 5 folds, based on precision and recall (on a class level)
- Precision - This is used on a class level. This is used to determine what percent of the time the model's guesses for a given class X actually are in fact X
- Recall - This is also used on a class level. This measures the percentage of time the model predicted class X when the instance corresponded to label X

The results for the metrics listed above are given in the following section.

## 5 RESULTS

Shown below is the Accuracy, Micro F1-score, and Macro F1-score for all 5 models.

| | ACCURACY | | MICRO F1 | | MACRO F1 | |
|---|---|---|---|---|---|---|
| | Mean | St. Dev. | Mean | St. Dev. | Mean | St. Dev. |
| Naïve Solution 1: Random | 0.186 | 0.020 | 0.186 | 0.020 | 0.142 | 0.011 |
| Naïve Solution 2: Continuity | 0.320 | N/A | 0.320 | N/A | 0.307 | N/A |
| Gaussian Naïve Bayes | 0.371 | 0.039 | 0.371 | 0.039 | 0.313 | 0.022 |
| **Neural Network with KNN** | 0.351 | 0.046 | 0.351 | 0.046 | 0.303 | 0.054 |
| **Kernel SVM - Quadratic Kernel** | 0.344 | 0.038 | 0.344 | 0.038 | 0.237 | 0.044 |
| KNN (5 Neighbors) | 0.283 | 0.008 | 0.283 | 0.008 | 0.241 | 0.014 |
| KNN (3 Neighbors) | 0.249 | 0.019 | 0.249 | 0.019 | 0.223 | 0.016 |

**Table 2. Accuracy, Micro F1-score, Macro F1-score for All 5 Models**

Standard deviation for the Naive 2 Solution is not measured because it did not need to be trained and was thus tested on all available data. This table is visualized in Figure 3 for greater clarity.

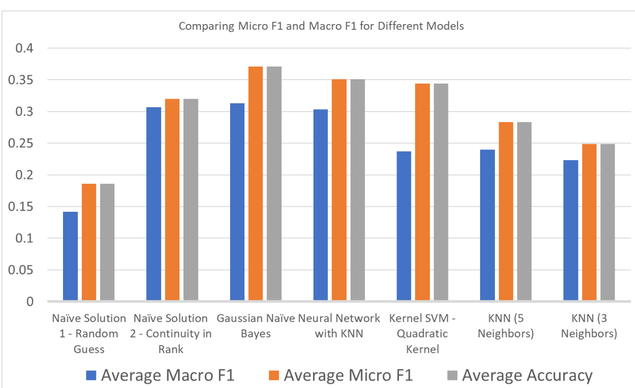


**Figure 3. Visualizing Macro F1-Score, Micro F1-Score, Accuracy**

In addition, the raw data for precision and recall per class (A-F) for each of the models is displayed in Table 3 below. Note that standard deviation is not included for the sake of simplicity given the density of the table.

| | Class A | | Class B | | Class C | | Class D | | Class E | | Class F | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Average Precision | Average Recall | Average Precision | Average Recall | Average Precision | Average Recall | Average Precision | Average Recall | Average Precision | Average Recall | Average Precision | Average Recall |
| Naïve Solution 1: Random | 0.215 | 0.090 | 0.090 | 0.050 | 0.058 | 0.034 | 0.120 | 0.120 | 0.250 | 0.294 | 0.185 | 0.322 |
| Naïve Solution 2: Continuity | 0.546 | 0.600 | 0.230 | 0.265 | 0.148 | 0.160 | 0.204 | 0.210 | 0.352 | 0.321 | 0.350 | 0.316 |
| Gaussian Naïve Bayes | 0.593 | 0.566 | 0.272 | 0.293 | 0.125 | 0.013 | 0.210 | 0.153 | 0.382 | 0.511 | 0.353 | 0.422 |
| **Neural Network with KNN** | 0.640 | 0.456 | 0.291 | 0.217 | 0.200 | 0.118 | 0.209 | 0.209 | 0.380 | 0.526 | 0.351 | 0.319 |
| **Kernel SVM - Quadratic Kernel** | 0.465 | 0.553 | 0.245 | 0.305 | 0.443 | 0.020 | 0.153 | 0.043 | 0.362 | 0.668 | 0.402 | 0.147 |
| KNN (5 Neighbors) | 0.385 | 0.484 | 0.175 | 0.161 | 0.077 | 0.055 | 0.178 | 0.184 | 0.351 | 0.421 | 0.269 | 0.198 |
| KNN (3 Neighbors) | 0.335 | 0.547 | 0.142 | 0.204 | 0.080 | 0.083 | 0.204 | 0.189 | 0.351 | 0.268 | 0.237 | 0.155 |

**Table 3. Precision and Recall Per Class for Each Model**

To understand these results, which will be later discussed, we will show the precision and recall separately in Figures 4 and 5 respectively.

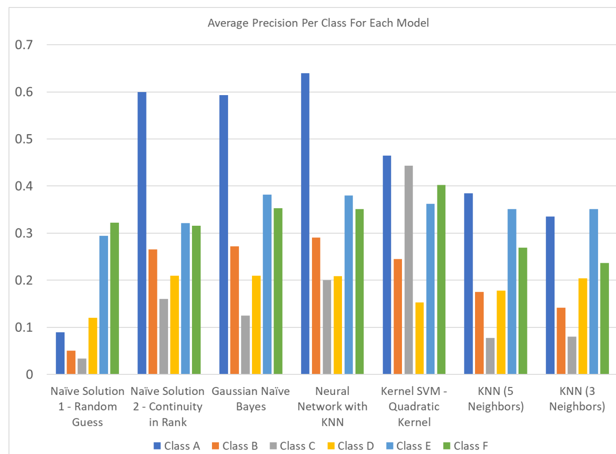


**Figure 4. Precision Per Class for Each Model**

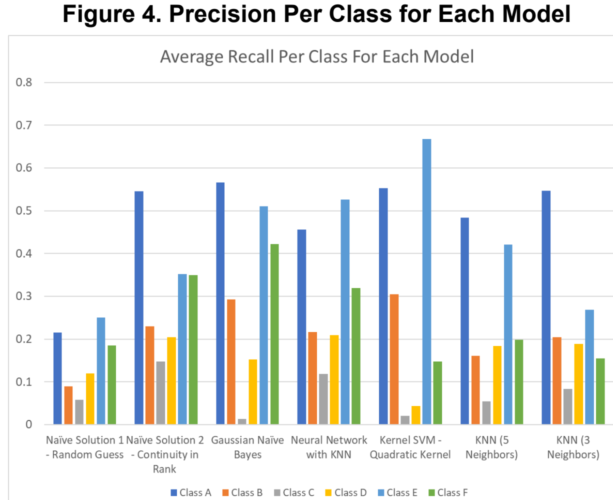


**Figure 5. Recall Per Class for Each Model**

As will be later discussed, the Guassian Naive Bayes model outperformed the intelligent naive guess in all 3 metrics, with about a 5% greater accuracy. The “Neural Network with KNN” model had about a 3% greater accuracy with only a slightly smaller Macro F1. And, lastly the Kernel SVM model had about a 2% better accuracy than the intelligent naive model but had about an average Macro F1 7% less than that model.

Secondly, the results for precision and recall showed that most models struggled with identifying classes C and D, but it was easier for them to identify classes A, E, and F.

Now, the results of each model will be explored separately.

### 5.1 Naive Solutions

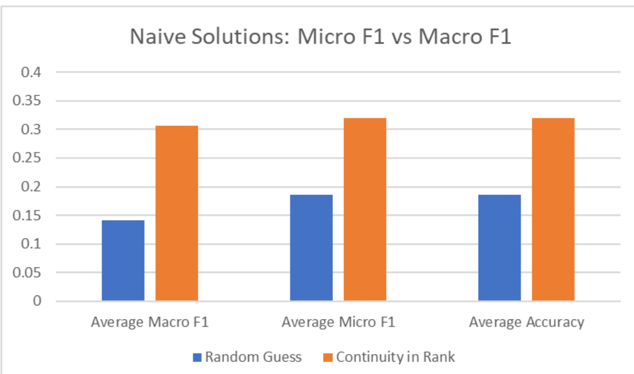


**Figure 6. Accuracy Metrics for Naive Models**

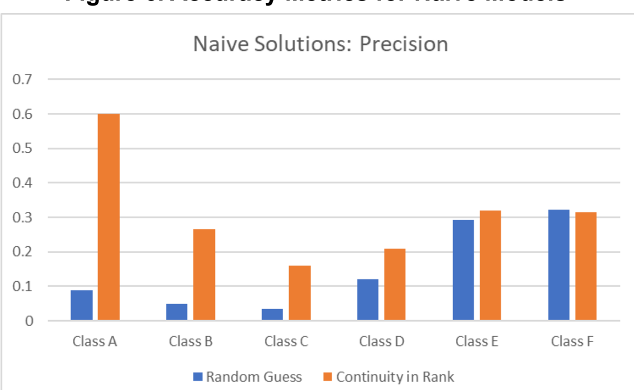


**Figure 7. Precision by Class for Naive Solutions**

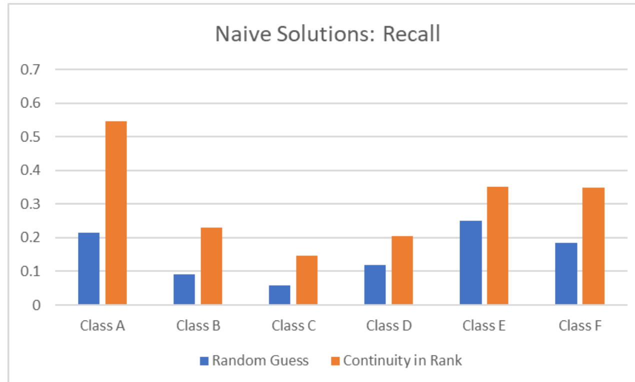


**Figure 8. Recall by Class for Naive Solutions**

As a baseline we implemented two naive solutions, one of which randomly guesses which class each golfer will be in in the next year and the other of which simply assumes that the rankings hold from one year to the next. As expected, the results of our random guess solution were quite poor with an average accuracy of 0.142 and an average Micro F1 of 0.186. The precision measures for this solution were at or around 0.1 for all classes except for class E and F. This is most likely due to the fact that classes E and F encompass larger ranges of golfers than the other classes, making it more likely that golfers are correctly placed into these categories with random assignment. Overall, the results of the random guess solution were in line with what was expected as it makes sense that randomly assigning golfers to a rank would be quite inaccurate.

Our second naive solution was assuming that golfers’ ranks remained the same from one year to the next. This solution was much more effective, having an average accuracy of 0.307 and an average Micro F1 of 0.320. This solution performed well for the Class A golfers, most likely due to the fact that the top tier of professional golfers are talented enough to nearly always perform at the top level of the sport. That being said, the accuracy results for this naive solution were slightly lower than we initially expected, with an overall accuracy of just over 30%. This speaks to the relative difficulty of the problem of predicting golfers’ ranks from year to year. As opposed to other sporting outcomes whose probabilities are nearer to 50%, there is a lot of variance in golf rankings from year-to-year as the collection of professional golfers is quite large.

### 5.2 Gaussian Naive Bayes

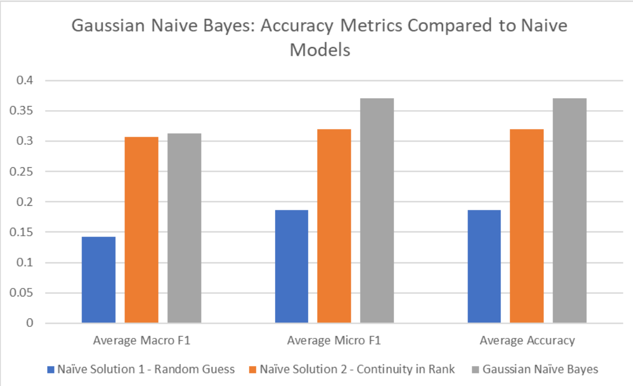


**Figure 9. Gaussian Naive Bayes vs Naive Solutions**

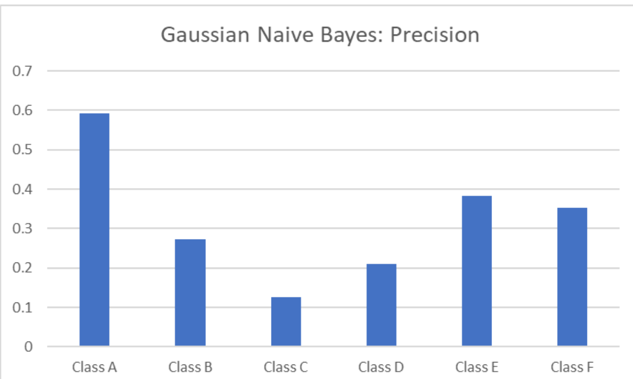


**Figure 10. Precision by Class for Gaussian Naive Bayes**

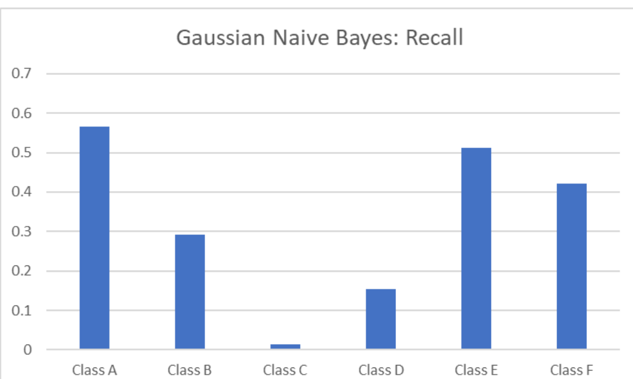


**Figure 11. Recall by Class for Gaussian Naive Bayes**

The first model we implemented was Gaussian Naive Bayes. This model performed quite well, outperforming our intelligent naive guess in all metrics with a 5% increase in accuracy. This was our best performing model overall, with an average accuracy of 0.371 and an average Macro F1 of 0.313. This good performance is mainly due to the model's exceedingly high performance in class F which had a great impact on the overall accuracy of the model due to the relative size of class F. Because class F contains golfers with rankings from 201 to 300, accuracy at this level has a much higher impact on a model's overall accuracy than any other. This is why our Gaussian Naive Bayes model performs significantly better overall than our other solutions despite having relatively similar metrics for the other classes when compared to the other models. Like the other models we implemented, Gaussian Naive Bayes struggled in correctly identifying golfers in class C and D. As discussed later, this is most likely due to the fact that golfers in the "middle-tier" of the professional scene do not differ significantly from one another in terms of true talent, thus making their performances much more difficult to predict with any accuracy.

### 5.3 Kernel SVM - Quadratic Kernel

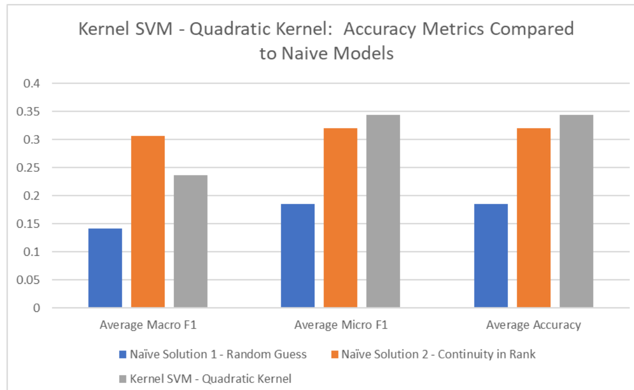


**Figure 12. Kernel SVM vs Naive Solutions**

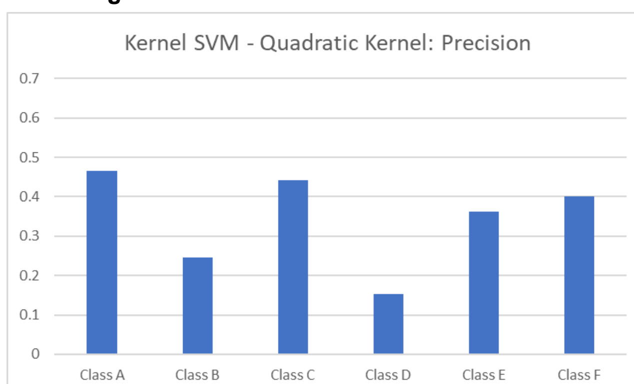


**Figure 13. Precision by Class for Kernel SVM**

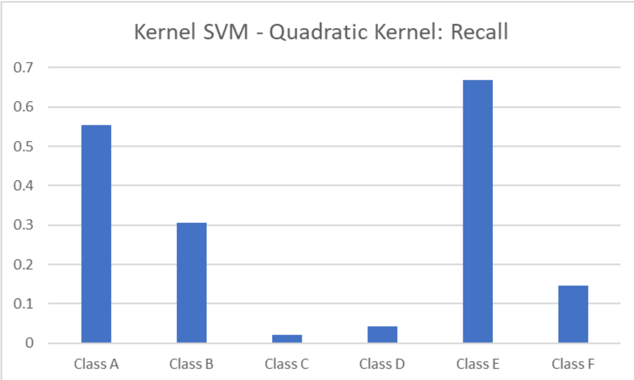


**Figure 14. Recall by Class for Kernel SVM**

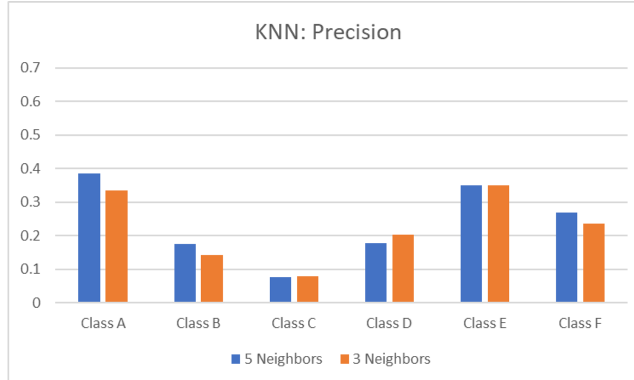


**Figure 16. Precision by Class for KNN**

The next model we implemented was a Kernel SVM (Quadratic Kernel). This model performed well, outperforming our intelligent naive solution with an accuracy increase of about 2.5% compared to the intelligent naive solution. It is only slightly worse than the Gaussian Naive Bayes in terms of accuracy but had an average Macro F1 of .237. These results for average Macro F1 could have been due to the fact that this class had a poor recall for class D relative to other models and a very poor recall for class F in general.

### 5.4 KNN - Five and Three Neighbors

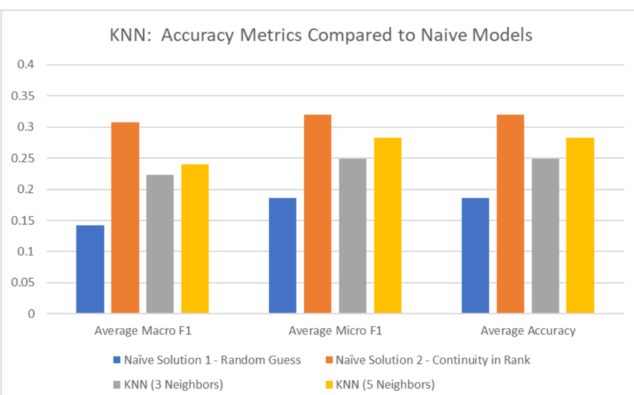


**Figure 15. KNN vs Naive Solutions**

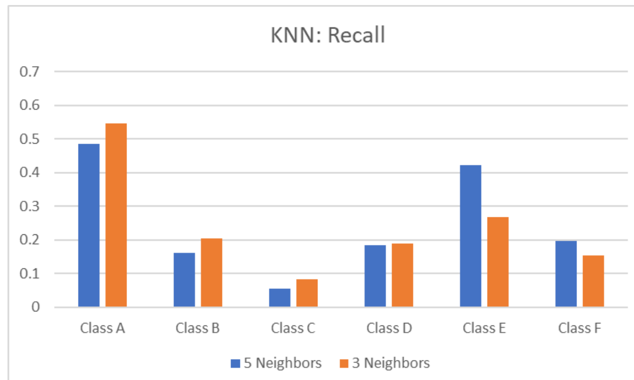


**Figure 17. Recall by Class for KNN**

For our KNN models, we tried with a K of 5 and 3. These models were not particularly great, with both losing to our intelligent naive solution with drops of 8% and 4% in accuracy respectively. As we can see, the KNN with five neighbors performed slightly better than the three neighbors configuration. This is likely because the five neighbors model had more context of the nearby players to predict future ranks from. It is possible that testing with an even higher K could yield better results.

### 5.5 Ensemble - Neural Network with KNN

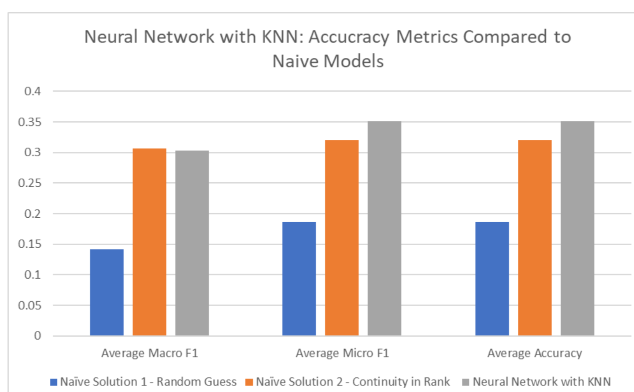

**Figure 18. Neural Network w/KNN vs Naive Solutions**

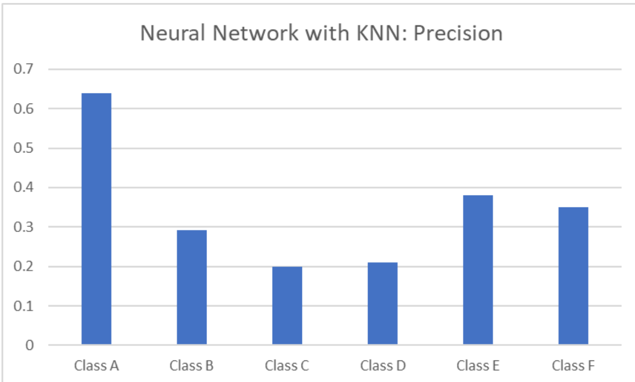


**Figure 19. Precision by Class for Neural Network w/KNN**

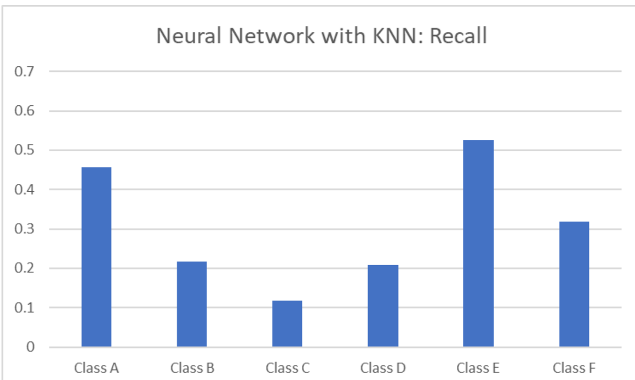


**Figure 20. Recall by Class for Neural Network w/KNN**

The last model we tried was an ensemble method of a neural network and KNN. Our neural network initially struggled to find any players in class F, so we added our KNN with 5 neighbors to assist. This model performed well, improving accuracy from our intelligent naive solution by about 3%. However, it did have a slightly worse average Macro F1 (0.303) compared to the naive model. Overall, we were very pleased with the performance of this model because it combined multiple concepts successfully to get a meaningful result and was quite consistent across all the classes compared to the other models. In the future we would look into making our KNN more accurate to help make this model overall more accurate.

### 5.6 Feature Importance Analysis

Upon inspection of our best model, Gaussian Naive Bayes, we were able to plot the importance of each feature, shown in Figure 21. As expected, the most important features in predicting future rank were current OWGR points average, number of top tens, and adjusted scoring average. Other notably important features were average strokes-gained off-the-tee, driving distance, and number of rounds played.

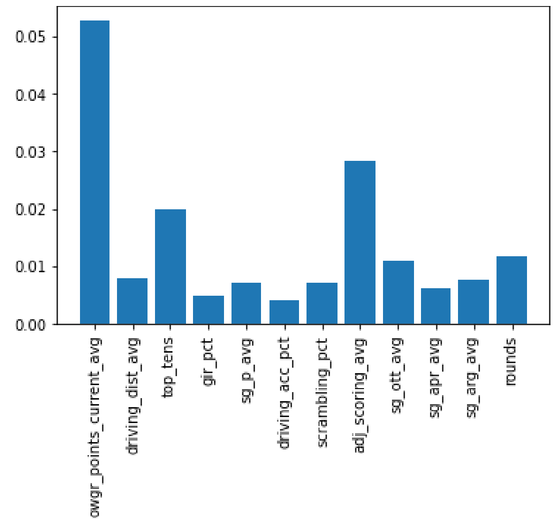


**Figure 21. Feature Importance for Gaussian Naive Bayes Model**

## 6 CONCLUSION

As previously stated, three of our models outperformed the intelligent naive model in terms of accuracy, and one model (the Gaussian Naive Bayes Model) outperformed the intelligent naive model in all three metrics.

In creating our model, we used 5-Fold Validation to ensure that every data object had a chance to appear in the test data. This validation method makes our model less biased, thus making our accuracy result of 0.371 for the Gaussian Naive Bayes Model more significant. As for why this model was the best in terms of accuracy and Macro F1, looking at the recall, we can see that this model significantly outperformed all other models in identifying class F. This gave this model a better accuracy because if a model struggles to identify class F, then it will struggle in overall accuracy given how many objects of class F there are (due to the fact that this represents players 201-300).

This brings us to the discussion of the precision and recall of the models previously presented. As was observed across all models, players in classes C and D were hard to identify. Though there may be more than one reason for why this is the case, the most likely reason is that players belonging to this category are in a moment of flux. In other words, these could be players who were either uncharacteristically good but later may sink back down to class F (or even below the top 300) or they may be players who are young and just beginning to reach their full potential. Class A, on the other hand, was quite easy for most models to identify. We expect that this is due to the fact that players who achieve this rank are likely to stay around this rank because they are elite, well-known players (Tiger Woods, Brooks Koepka, etc.). In addition, perhaps the imbalance in the label distribution could have hurt the model's ability to identify a class, at least for class C. However, it does not seem that the label distribution made too much of a difference because, first, there was not that significant of label imbalance. Secondly, it is clear that many models were much better in identifying class B than class D even though class D had a higher label distribution.

Related to these results of precision and recall, we see that the Neural Network with KNN model was consistent in precision and recall across all classes relative to the other models. Given this fact and the fact that it outperformed the intelligent naive model by about 3% with an insignificant decrease in Macro F1 score, one could also argue that this model is one of the best models given its consistency.

Finally, as for the problem itself, we learned that this problem, like many sports prediction problems, is quite difficult. The intelligent naive model was only able to achieve approximately 32% accuracy! This fact made us more pleased with the results achieved by our models. In addition to the difficulty of the problem, we also learned about the importance of the features themselves. Of course, current rank is the best predictor of future rank. However, among each of the strokes-gained features, our results show that strokes-gained off the tee is the most important, much more so than features associated with shorter shots, like GIR percentage or strokes-gained approaching the green. Furthermore, driving accuracy is the least important feature in our best model. Ultimately, our results seem to underscore a growing theme in professional golf right now, where the best players are shifting their focus almost entirely on driving the ball as far as possible [4].

**REFERENCES**

[1] Golf News Net. "How the Official World Golf Ranking Works: Formula, Points, Averages." *Golf News Net: What You Need to Know about Golf*, 18 Mar. 2020, https://thegolfnewsnet.com/golfnewsnetteam/2020/03/18/how-the-official-world-golf-ranking-works-formula-points-averages-118671/.

[2] Wiseman, Oisin. *Using Machine Learning to Predict the Winning Score of Professional Golf Events on the PGA Tour*. http://norma.ncirl.ie/2493/1/oisinwiseman.pdf.

[3] Vrahas, Mark. "Scikit-Learn Linear Regression for Predicting Golf Performance." *Medium*, Towards Data Science, 27 May 2020, https://towardsdatascience.com/scikit-learn-linear-regression-for-predicting-golf-performance-c92f31b69f92.

[4] Herbert, Brian. "Bryson DeChambeau Is Playing the Long Game, Literally and Figuratively." 6 Oct. 2021, https://www.si.com/golf/news/bryson-dechambeau-is-playing-the-long-game-literally-and-figuratively.