Ensemble Model for Predicting NFL Game Winners

Patrick Niccolai
University of Vermont
patrick.niccolai@uvm.edu

ABSTRACT

The problem of predicting NFL game winners has many real-world implications, especially in sports gambling. However, much of current game predicting involves biased "gut-feelings." My model aims to remove these biases from game prediction by making statistically informed predictions. Using player data scraped from NFL team's websites, I have created an ensemble classification model. The model works by training 5 linear regression models, each on a position on an NFL team: quarterback, rusher, receiver, tackler, interceptor. Then, I run the player data through these models to get a 0 to 1 number for each player, roughly equivalent to how "good" that player is. Then I take this data and use it to train a classification model, which predicts what team will win a given matchup.

Keywords

NFL game prediction, ensemble model, linear regression, logistic regression, decision tree, random forest, SVC, neural network, K-Nearest neighbors.

1. INTRODUCTION

An accurate model for predicting NFL game winners would have many real world uses. Firstly, NFL gambling is a billion-dollar industry [1]. The ability to accurately predict NFL game winners would have a major impact on NFL gambling. Many gamblers place their bets based off of "gut-feelings," which will likely be biased and inaccurate. These gamblers would benefit greatly from a statistically informed method of predicting game winners.

However, the model does not only need to be used for gambling. There are many reasons one would want to predict NFL game winners. There is an entire industry of sports media, with many people interested in predicting game winners. They could also benefit from a model that accurately predicts NFL game winners.

2. RELATED WORK

One paper, A Hybrid Prediction System for American NFL Results [1], is about a classification model for predicting NFL game winners. In this paper, the authors created a K-Nearest neighbors model for predicting NFL game winners, which has an accuracy of over 80%. In my research I found that a K-Nearest neighbors model was not the most accurate classification model for my data. My model is different from this paper's model because they used team statistics (e.g. a team's current record, a team's total yards gained on offence, etc.) while I used player statistics (e.g. a quarterback's completion percentage, a running back's average yards per carry, etc.).

Another paper, Application of Neural and Regression Models in Sports Results Prediction [2], is about a linear model for predicting distances of javelin throws. In this paper, the authors found that a neural network model was the most effective, which I also found to be the case for my data.

3. METHODS

3.1 Data Collection

There are two types of data I used in this model. The first is player data. To get this data I used web scraping on all 32 NFL team's websites. This caused some problems because many of the websites are formatted differently, specifically they have the tables at different indexes (e.g. the Giants quarterback table is at index 6 while the Patriots quarterback table is at index 0). I got around this by hardcoding the indexes of all tables for each team. Fortunately, the tables themselves are all formatted the same so I could combine them without any difficulty.

The second type of data I used was a table containing the 2020 season schedule, which had all matchups and which team won or lost the game for games that already happened.

3.2 Data Exploration

The main exploratory data analysis for this project involved looking at which stats were correlated with game outcomes. I did the exploratory data analysis on an earlier iteration of the cycle, so I was looking at different data than what ended up in the final model. However, I still learned one useful piece of information, which is that engineered features were less correlated with game outcomes than using the features as they are. Below are three graphs correlating offensive yards with points scored in a game.

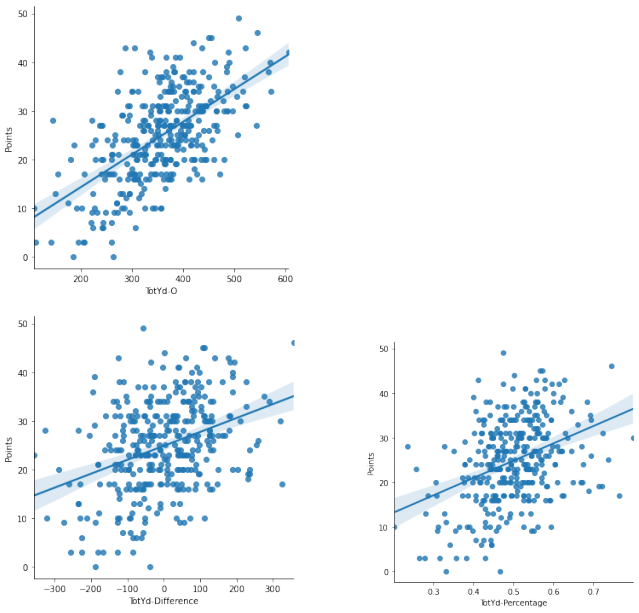

Figure 1: These graphs show the correlation between: offensive yards in a game by one team (top), difference of yards gained in a game between the two teams (bottom left), and the percentage of the total yards in a game one team got (bottom right) and the points scored by the team that game. Note the graph of the plain feature (top) has a higher correlation than the two engineered features (bottom).

3.3 Data Cleaning

The data cleaning part of this project was very important. The primary cleaning that needed to be done was to standardize the number of players on each team. Different teams have different

numbers of players at each position, for example the New York Giants have used 2 quarterbacks this season while the Dallas Cowboys have used 4. Therefore, when training the model most of the teams would have many null values, which would skew the model. To account for this, I only took the top certain number of players for each team. I used 1 quarterback, 3 rushers, 3 receivers, 5 tacklers, and 2 interceptors. This solves the problem because now all teams have the same number of players so there won't be any null values skewing the model. This also more accurately reflects a team's actual lineup.

Then I filled in any null values with 0 because that is what a null represents in this dataset. Then I scaled the data because many of the features were of different magnitudes.

Finally, I cleaned the data for game schedules and outcomes by removing all games that had not happened yet, so the model will only be trained on games that have outcomes.

3.4 Linear Regression Models

The next step in the project was to train a linear regression model using sci-kit learn's [3] linear regression model for each position (QB, RB, WR, tackler, interceptor). First, I randomly split the schedule into training and test data, with 90% of games going into the training data and 10% into the test data. Then, I created a new dataframe for each position, and for each game in the training data and I added a row/rows to that dataframe with the corresponding team's players and the outcome of the game. So, for example if the game in question was played by the Giants and they lost the game, I would add 1 row to the quarterback training dataframe with the Giant's quarterback's stats and a 0 in the win/loss column, 3 rows to the rushing training dataframe with the 3 Giant's rushers stats and a 0 in the win/loss columns, and so on for receiving, tackling, and intercepting.

After the linear regression models were trained, I used them to get a prediction for each player. This prediction was a number from 0 to 1 roughly equivalent to how "good" that player was. These numbers are the data I used for the rest of the project.

3.5 Classification Models

The final step was to train a classification model. First, I created a dataframe with a row for each team and a column for each position, so the dataframe had 32 rows with 1 column for a quarterback, 3 columns for rushers, 3 columns for receivers, 5 columns for tacklers, and 2 columns for interceptors.

Then, using that dataframe I created another dataframe, with a row for each game in the entire dataset of games with outcomes. Each row had the stats from the two teams taken from the previously discussed dataset, along with the outcome of the game, a 1 if the first team in the row won the game and a 0 if the first team in the row lost.

Then I used cross validation with 10 folds to determine which classification model was most accurate out of 6 models: logistic regression, decision tree, random forest, SVC, neural network, and K-Nearest neighbors, all from sci-kit learn [3]. Finally I chose the most accurate model, neural network, and trained a neural network model on the training data so it could be used to predict future games.

4. RESULTS

4.1 Predictive Models

I found that the neural network model was most accurate for predicting NFL game winners with a mean accuracy after 10-fold cross validation of 0.702. Neural network was closely followed by logistic regression, which had a mean accuracy of 0.691. The neural network model was statistically significantly better than decision tree, random forest, and K-Nearest neighbors. All other comparisons were not statistically significant.

Below is a table of all classification models tested and their mean accuracies, along with the p-value from comparing each model to the neural network model.

Model	Mean Accuracy after 10-fold cross validation	P-value vs. neural network
Logistic Regression	0.691	0.733
Decision Tree	0.628	0.034
Random Forest	0.639	0.049
SVC	0.631	0.059
Neural Network	0.702	--
K-Nearest Neighbors	0.617	0.034

Below I've graphed box plots for each model's accuracies after cross validation.

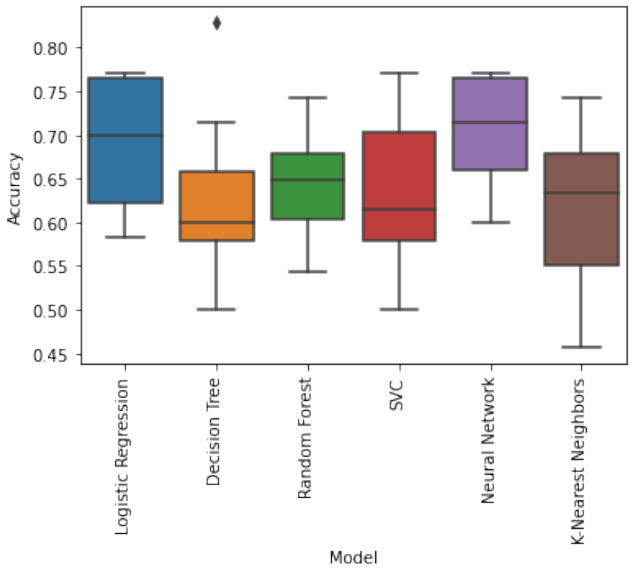

Figure 2: This graph shows the accuracies of each model after 10-fold cross validation. Neural network had the highest mean accuracy.

4.2 Feature Importance

Then I decided to look at which features were most important by comparing the coefficients of the features for the logistic

regression model. What I found was that, unsurprisingly, the quarterback was the most important feature for determining the outcome an NFL game. Interestingly, the second most important feature was interceptions. This surprised me because I assumed the offensive position would be most important, however it makes sense that interceptions are important because even one interception can be game changing.

5. DISCUSSION

5.1 Model Accuracies

Overall, the models were not highly accurate. The most accurate classification model, neural network, had a mean accuracy after 10-fold cross validation of 0.70. This is a good accuracy but not very high considering similar models have gotten accuracies of above 0.80 [1]. I think there are several possible reasons as to why the accuracy is relatively low.

Firstly, the data I am working with is not perfect. I only have season total data (a running back's total number of rushes for the season, a quarterback's completion percentage over the entire season, etc.). Therefore, the model is essentially assuming that once scaled, these stats don't change much week-to-week, which is probably somewhat true but not entirely. However, by using these statistics the model is easily applied to predicting future games.

Secondly, I believe that when looking at player data there is a lot of noise that lowers the accuracy. To account for this I tried removing some noise from the data by removing certain features before training the linear regression model. Specifically, I removed the following features from the quarterback dataframe: total pass attempts, total completions, total yards, total touchdowns, total interceptions, longest pass, total number of sacks, and total number of yards lost per sack. I removed these features because I thought they could be just adding noise because they are less important than features like completion percentage, yards per completion, etc. Removing noise by removing these features did increase the accuracy, but only very slightly.

Finally, I believe the accuracies of the models are low because predicting NFL game winners is a difficult problem. The reason I believe this is that when looking at what games the model predicts wrong, many of them are upsets, where the "worse" team wins the game. Therefore, I believe the model is accurate at picking the "better" team, however the "better" team doesn't always win the game. Further evidence for this is that the model agrees with VegasInsider.com betting odds (i.e. the model's predicted winner is the favorite to win on VegasInsider.com) at a high rate.

5.2 Limitations

I believe the biggest limitation in this model is dealing with injured players. The NFL team's websites don't remove injured players from their datasets, which means that injured players are included in my dataset. Therefore, if an injured player, for example a quarterback, still has enough pass attempts to be their team's leading passer, that quarterback's stats will be used by the model for that team's predictions until another quarterback on that team has more pass attempts than the injured quarterback. Clearly this is not ideal as the injured player will not actually be playing in the game and not be affecting the game's outcome.

Another limitation is the fact that the winner of an NFL game is not only determined by the players. Things like weather, travel, bye weeks, etc. all have important impacts on deciding game winners, and these are not taken into account by the model.

6. FUTURE WORK

I believe the most important thing to work on in the future is the linear regression models. Up to this point, most of the work on the project went into creating the classification models and testing them, while comparatively little work went into the linear regression models. For all positions except for quarterback the linear regression models are trained on all features from the dataset. This is likely not ideal as some features may be less important for evaluating a player, so in the future I would like to work more on choosing the optimal set of features for the linear regression models.

Also, in the future I would like to perform more statistical tests analyzing the models. I have looked at the accuracy for predicting the winning team, but there are many other ways to test the model, like accuracy against VegasInsider.com betting favorites. There are also more p-value tests I could perform to see which models are statistically significantly better than random chance, always picking the team with the better record, etc. These tests would give a better idea of how well the model truly works.

7. CONCLUSION

This project predicts NFL game winners by using an ensemble model on player statistics to predict what team will win a given matchup. The model works by scraping player stats from the 32 NFL team's websites, training 5 linear regression models on the 5 player positions, then using the outputs of these models in a classification model to predict a winner between two teams. The model can predict game winners with an accuracy of 70.2% when using a neural network as the classification model, which was the classification model with the highest accuracy out of logistic regression, decision tree, random forest, SVC, neural network, and K-Nearest neighbors.

8. ACKNOWLEDGMENTS

Thank you to Professor Nick Cheney, who aside from teaching me everything I know about data science helped with the player count standardization, which solved the problem of teams have null values in their datasets and helped the model more accurately reflect an actual NFL team lineup.

9. REFERENCES

[1] Anyama O. U., Nwachukwu E. O., A Hybrid Prediction System for American NFL Results. *International Journal of Computer Applications Technology and Research Volume 4–Issue 1*

[2] Maszczyk A., Gołaś A., Pietraszewski P., Application of Neural and Regression Models in Sports Results Prediction. *Procedia - Social and Behavioral Sciences, March 2014*

[3] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

About the authors:

Patrick Niccolai is an undergraduate computer science major at the University of Vermont. When he is not working he enjoys reading and exercising.