NFL Game Prediction: A Machine Learning Approach Using Game Stats

Joseph Lyon, Michael Angell, Jeffrey Angell

CS 478, Winter 2015

Department of Computer Science

Brigham Young University

Abstract

NFL analysts attempt to predict the winner of games each week during the NFL season. A large number of statistics relating to each team, as well as individual members on each team, are used in these predictions. Currently they can achieve about a 60% prediction accuracy rate. We wanted to examine the accuracy rates of different machine learning algorithms and see if we could make any improvement.

In this paper our process for training and testing different machine learning algorithms to predict the winner of NFL games. We first describe how we gathered the data, what features we selected, and the learning models we used. Next, we analyze the initial results of training as well as consider various changes to the data set and feature selection process that could improve accuracy. Then we discuss using the same statistics to predict the point margins for each game and report our accuracy measurements for this. Finally we consider further areas for exploration on this topic.

1 Introduction

In an NFL game the home team wins about 55% of the time. The winner of the coin toss also wins about 55% of the time. As such, analysts are only about 5% better than these other metrics. This does not seem very good considering the vast quantity of statistics available from games dating back several years. If teams A and B were to play each other one-hundred times the “better team”, based on season stats, would clearly be the favorite. However, on any given weekend, team A has just as much chance of winning as does team B. There are several reasons why this is so. A few possibilities are that the underdog that week is incredibly focused and wants to pull of the upset. Or the clear favorite overlooks that week’s game, thinking it is already a decided victory. Also it could be in part due to the extremely high skill and talent levels in the NFL. This is what makes predicting games with any significant level of accuracy very difficult.

1.1 Motivation

We wanted to determine if different machine learning algorithms could improve on the 5% margin and potentially predict the winning team with better accuracy than current analysts. Further exploration led us to predicting point margins. We used multi-layered perceptron and k-nn machine learning models to predict the winning team and point margins of a game.

We first created a model that would

predict the winner of a game given all the stats from the game except the final score. Then we created a model that would take the statistical averages of two teams and predict a winner should the two teams play. We will discuss the results of both of these data sets later on.

2 Methods

2.1 Data Gathering

NFL.com has statistics stored for every NFL game going back several decades. We found an open source python library[1] built to retrieve stats for all NFL games played from the year 2009 to the present. Each NFL season, excluding the pre-season and playoffs, has 256 games. Starting from 2009 and using every game of every season there will be 1536 games available for our data set.

Using the python library we pulled data about these 1536 games and used that to calculate season averages as well as averages over the past three games. This is the data that we used to create our data sets. As we will discuss later on, we were able to double our data by taking each instance and swapping the first and second teams stats so the order was different.

2.2 Data Sets

2.2.1 Game statistics

We ended up creating two data sets that we trained on. The first set had the stats for each game in our time frame. This was fairly simple to retrieve using the python tool since the data it got from NFL.com had all the stats we wanted for this set. This first data set had 3072 instances. As mentioned above we took the 1536 games and duplicated them by switching the order of the two team's stats. Each instance had 14 attributes, namely: total yards, passing yards, rushing yards, turnovers, punts, first downs, and time of possession for each team.

2.2.2 Prediction set

The second data set took more work to create. For every team in the NFL we went over every game they played and stored the stats for that game. Then using the first data set we created we calculated the season and last 3 game average stats for each team up to that point and made that the data instance for that game. Each instance in this set had 26 continuous features that are described in Table 1. This data set had 3072 instance since we took the 1536 games and doubled as described below in the Data Expansion section.

2.3 Selected Models

All of our selected features are continuous values so we wanted to choose learning algorithms that easily handled real-valued features. At first we were doing classification but also planned on doing regression as well. We wanted to use learning models that would be accurate at predicting both classification and regression. Also in terms of time limitations there aren't any real-time requirements for either training or predicting for this problem. So we were not limited in our model selection based on time restraints.

With these things in mind we decided to use a multi-layered perceptron with backpropagation of errors as well as a k-nearest neighbor learner. Both of these models easily handle real-valued features as well as predicting class and regression. We also considered using a decision tree model. This is described further in section 7.2.

Feature	Description	Example Value
homeTotalYdsLast3	Home teams average yards for the last 3 games	321.6667
homeTotalYdsSeason	Home teams average yards for the season	351.8333
homePassYdsLast3	Home teams average passing yards for the last 3 games	247.0
homePassYdsSeason	Home teams average passing yards for the season	281.3333
homeRushYdsLast3	Home teams average rushing yards for the last 3 games	74.6667
homeRushYdsSeason	Home teams average rushing yards for the season	70.5
homeFirstDownsLast3	Home teams average number of first downs for the last 3 games	18.0
homeFirstDownsSeason	Home teams average number of firsts downs for the season	18.6667
homeTurnoversLast3	The average number of turnovers committed by the home team for the last 3 games	0.6667
homeTurnoversSeason	The average number of turnovers committed by the home team for the season	1.1667
homePuntsLast3	Home teams average number of punts for the last 3 games	4.3333
homePuntsSeason	Home teams average number of punts for the season	3.6667
homePosTimeLast3	Home teams average time of possession for the last 3 games	1634.0
homePosTimeSeason	Home teams average time of possession for the season	1689.5
awayTotalYdsLast3	Away teams average yards for the last 3 games	237.3333
awayTotalYdsSeason	Away teams average yards for the season	220.8571
awayPassYdsLast3	Away teams average passing yards for the last 3 games	139.0
awayPassYdsSeason	Away teams average passing yards for the season	128.0
awayRushYdsLast3	Away teams average rushing yards for the last 3 games	98.3333
awayRushYdsSeason	Away teams average rushing yards for the season	92.8571
awayFirstDownsLast3	Away teams average number of first downs for the last 3 games	12.6667
awayFirstDownsSeason	Away teams average number of first downs for the season	12.1429
awayTurnoversLast3	The average number of turnovers committed by the away team for the last 3 games	3.0
awayTurnoversSeason	The average number of turnovers committed by the away team for the season	2.5714
awayPuntsLast3	Away teams average number of punts for the last 3 games	6.3333
awayPuntsSeason	Away teams average number of punts for the season	6.0
awayPosTimeLast3	Away teams average time of possession for the last 3 games	1620.3333
awayPosTimeSeason	Away teams average time of possession for the season	1573.0

Table 1: Feature names with descriptions and an instance example. All the features are continuous values and this example instance was taken from the actual data set used.

3 Initial Results

We first substantiated the correlation of various game statistics and final scores by running our game statistics through both of our planned models. Both of our models were able to predict the winner with about 80% accuracy.

We then used a greedy wrapper on our knn model to determine the most important features for determining a winner. Our wrapper determined that the features that didn't matter in our data set were the year and the week, and that the most important features were number of turnovers and number of punts from both teams. Unfortunately, this didn't allow us to remove as many features as we would've liked.

After removing these unnecessary features, we ran both of our models on our prediction set. Using cross-fold validation with 10 folds, we achieved 60.40% with our backprop model, and 61.11% with our knn model.

4 Data, Feature, and Model Improvements

4.1 Feature Reduction

Feature Reduction seems to be a promising technique for this problem. It makes sense that total number of yards will be related to pass yards and possession time, and indirectly correlated with turnovers and punts.

The greedy wrapper technique was unable to find easily removable features. No matter what we removed to make the model simpler, the accuracy never increased. But there was rarely a significant drop either. Every feature seemed to have a small influence, all similarly removable, so we decided to try principle component analysis instead.

Feature removal results (feature-removed : percent-increased-accuracy-afterward):

0:35 1:44 2:54 3:41 4:38 5:47 6:47
7:46 8:47 9:43 10:49 11:42 12:58 13:52
14:40 15:44 16:48 17:49 18:43 19:49 20:43
21:55 22:47 23:35 24:44 25:42 26:56 27:53

We ran PCA on the data set, and reduced the feature set from 26 to 16. We lost approximately one percent of accuracy in both the MLP and the KNN learners using this new dataset.

4.2 Data Expansion

Using the python module that stored data from the 2009 season and on, we were able to put ~1500 instances in our dataset. Unfortunately, to get more data we would have to do much more primitive crawling of the nfl.com webpage. Instead of trying to do that, we doubled our dataset by adding an instance to every game from the away team's point of view. Our original dataset had the home team on the left, the away team on the right, and an output classifying whether the home team won or lost. The new dataset had new instances with the home team and the away team swapped, and the output classifying whether the left or right team won.

The accuracy of our machine learning dropped 2%, reasonable because we effectively removed the home team information, but there is an interesting side effect. The original data set had a baseline prediction of 56% when always predicting that the home team won, while the new data set has a baseline prediction of 50% when choosing the left or the right team. Getting the same 61% accuracy on the first implies 5% learned, while that accuracy on the second implies 11% learned.

4.3 Regression

In an attempt at getting better prediction results, we tried to run this problem through a regression learner using the final point spread between the teams as the output.

When run through a baseline learner (always return zero, home perspective and away perspective will always cancel out), we obtained a RMSE of 15.704. Unfortunately, when running various learners on this changed problem, we never dropped our RMSE below 15.1. We abandoned this trail pretty quickly. It didn't seem like a regression learner would give us better classification accuracy.

4.4 Models

While training and testing our multi-layered perceptron model we experimented with different numbers of hidden layers and hidden nodes in each layer. We found that the more hidden layers which were added the training time increased exponentially. Also we found that using more than five hidden layers resulted in a drop in prediction accuracy by about 5 percentage points. Ultimately we decided on using two hidden layers with fifteen hidden nodes in each layer. This gave us the best accuracy.

We trained the k-nn model with different k-values ranging from 3 to 21. Using a k-value of 13 gave us the best results. Also for both models we found it necessary to normalize the training data. This also improved our results.

5 Final Results

Dataset	Learning Algorithm	Accuracy
Standard (home team on left)	MLP	60.40%
Standard (home team on left)	K-NN	61.11%
Doubled (both perspectives)	MLP	60.54%
Doubled (both perspectives)	K-NN	58.92%

6 Conclusion

Despite many iterative attempts to improve the prediction accuracy, we could never improve much beyond 61%. This is very similar to the reported accuracy of more professional prediction algorithms. Either professional football is a very difficult problem to predict, and the games can continue to be interesting, or a research team will suddenly and unexpectedly make a lot of money gambling on this.

7 Future Work

7.1 Feature Refinements

There is a lot of data that can be obtained for this problem. We attempted to identify the most obviously useful information, and tried to use it in creative ways, but there's still a lot of potential. Some other possible attributes we might consider are:

Tracking key individual athletes on each team. Knowing who is injured or having an unusual season relative to previous seasons would be helpful information.

Feature manipulation on our current data set. In particular, we would add columns for other averages, such as including a last game column, a last 2 games average column. We would also try removing the entire season average column.

When we created our doubled dataset we dropped the home and away information for each

team. We could try adding a class column back in that tracked that. That would allow us to increase the data amount without losing the information.

Instead of have features for each teams average stats, we could use the difference between the stats. For example instead of have an attribute for average season passing yards for both teams 1 and 2, we could instead have a feature that showed the difference in passing yards between teams 1 and 2. This would reduce the number of features in our data set and as well as potentially learn something new from this combination.

7.2 Other Models

A decision tree would be an interesting model to try. We would use a genetic algorithm to create a single threshold for each column, and then use the classes created in a standard ID3 algorithm. It would be interesting to see if this approach could be at or around the 60% accuracy level. If so this would imply that there is not an exact number you need to reach in order to win or lose a game. But rather there is threshold, that once reached leads to a win or a loss.

References

1 https://github.com/BurntSushi/nflgame.