NFL Predictions

Stats 415

Adam Stasinski, Nakoul Makim, Jack Lucci

Can we accurately predict the outcomes for NFL games?

Introduction:

Sports gambling is a billion dollar industry. Many states have introduced legislation to legalize sports gambling in their respective states due to a recent court ruling that allows for the legalization of betting. Sports gambling is already a billion dollar industry, and with the recent legalization, we can expect the industry to continue to grow. This leads to a single question, can we predict sports? This has been a question that humans have been trying to solve for centuries. Currently, Las Vegas sportsbooks are the most accurate at predicting the outcome of sports. They set gambling lines for every sport, including prop bets. The sport that makes the most revenue is the NFL, which brought in $8.2 billion in 2017. This makes the NFL the highest revenue generating sports league in the world. This leads to the billion dollar question, can we predict the outcome of NFL games?

Goals:

Our data exploration involved two questions of interest: 1. Can we accurately be able to predict whether the home team will win or lose and 2. How does our model compare to the spread favorite and over/under line for a game.

Data:

Our dataset, spreadspoke_scores.csv, comes from kaggle.com. This dataset originally contained data on all games for the NFL, from 1979-2019. This dataset includes game results and betting odds for every game. It was created using public websites such as ESPN.com, NFL.com and Pro Football Reference. This data originally contained 17 predictors. These predictors were schedule_date, schedule_season, schedule_week, schedule_playoff, team_home, score_home, score_away, team_away, team_favorite_id, over_under_line, spread_favorite, stadium, stadium_neutral, weather_temperature, weather_wind_mph, weather_humidity, and weather_detail.

We quickly realized that this dataset was not useful for creating predictions on which team would win the game as it was not engineered properly to plug into statistical models. Specifically, the games were listed on a per game basis with the teams as a categorical string for both the home and away teams respectively. In order to fix this, the teams were all converted to binary factors as “h_TeamName” and “a_TeamName” with these columns equal to 1 if the home or away team was equivalent to “TeamName” (home and away done separately). Due to team name changes throughout the years, teams were defaulted/converted to their current team name before the data was sorted. The Date/Time variable was then engineered into 3 numerical variables Day, Month, and Year in order to gain predictive power from the variable. The team_favorite_id was converted into a binary factor for whether or not the home team was favorited. Binary factors were made for whether it was a playoff game and whether the game was played in a dome. A few other columns were renamed into their equivalent predictors and the data was cleaned. The cleaning process consisted of deleting any rows that contained NAN values with the exception of weather related statistics which were more sporadic and were filled with the mean value for their respective column. This resulted in a total of 9444 rows covering just about every game played in the NFL from 1979 to 2019. After this was done we tested the effectiveness of the new predictors using general eye testing and found that the new predictors were not yielding much predictive power with the exception of the betting related variables. In order for us to make accurate predictions, we needed some new and more effective predictors. To do this, we further engineered this dataset with focus on the scoring and winning lagging averages for each team. After engineering the new features, the predictors were now score_home, score_away, Day, Month, Year, spread_favorite, over_under_line, weather_temperature, weather_wind_mph, weather_humidity, is_playoff, played_in_dome, game_week, is_home_favorited, h__x, a__x, where x is the abbreviation for a NFL team, Home_team_wins_prev_y, Away_team_wins_prev_y, Home_avg_pts_prev_y, Away_avg_pts_prev_y, Home_team_home_wins_prev_y, Away_team_away_wins_prev_y, Home_avg_pts_athome_prev_y, and Away_avg_pts_ataway_prev_y, where y represents the number of lagging games to average for the result, with y values of 1, 3, 5, 7, 10, and 15. The predictors is_playoff, played_in_dome, is_home_favorited, h__x, and a__x are all factor variables. The home and away team predictors are all predictors that we created, using scores for teams from their previous games.

We then created a response variable, TARGET_Home_win, which is a binary factor variable that indicates whether the home team won or lost the game. Also, score_home and score_away were omitted from the predictors as they would essentially feed the model the target variable that we were trying to predict.

Overview:

We decided to split 80% of our data into a training set and the other 20% into a test set. Since our response variable, TARGET_Home_win, is categorical, we decided that we would use linear discriminant analysis, quadratic discriminant analysis, K nearest neighbors, logistic regression, and support vector machines as our models. We performed initial testing on all these models and quickly confirmed that the only variables that gave us significant results were the average points and previous win variables that we added to our dataset. We also decided to omit is_home_favorited, spread_favorite, and over_under_line as these were predictors created professionally by Las Vegas Sportsbook (Explained in further below).

Overall, our final resulting predictors were:

{Home_team_wins_prev_y, Away_team_wins_prev_y, Home_avg_pts_prev_y, Away_avg_pts_prev_y, Home_team_home_wins_prev_y, Away_team_away_wins_prev_y, Home_avg_pts_athome_prev_y, and Away_avg_pts_ataway_prev_y}

The y values consisted of 1, 3, 5, 7, 10, and 15

Our final resulting target variable was:

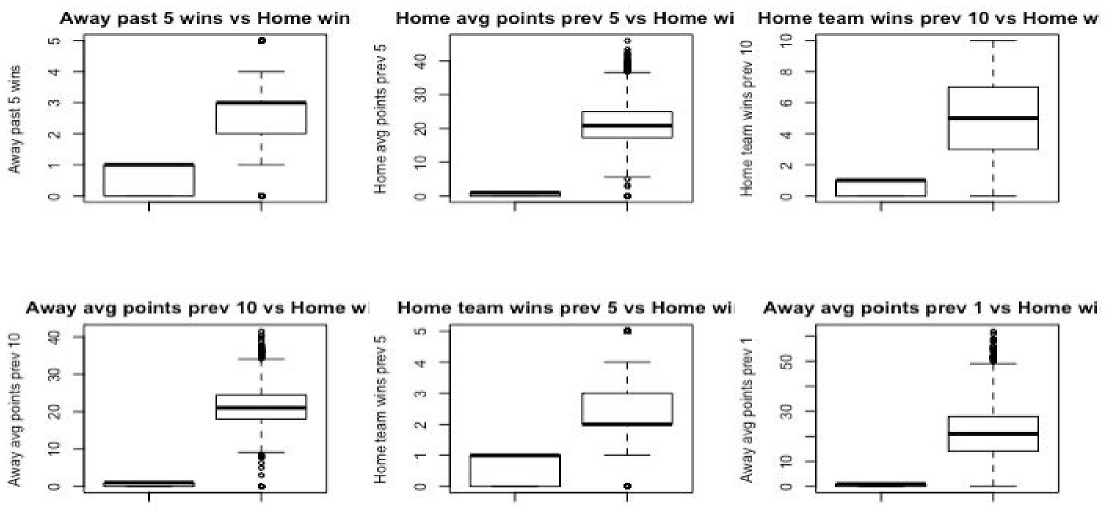
{TARGET_Home_win}

From the above set of predictors, the subset of predictors used for our LDA, QDA, KNN, and logistic regression models would be chosen from data visualization to keep the model to six predictors. For the SVM model, forward and backward selection would be performed to select a new unique subset of predictors to compare with our models.

Our plan was then to compare test errors from all of these models to determine which model gave the most accurate predictions. From there, we planned on comparing the best model we came up with to our "benchmark value", a value calculated from the is_home_favorited predictor. The reason this predictor gave the benchmark value is because it was the professionals pick for whether or not the home team would win the game. We did not expect our accuracy to outperform Las Vegas, because they use many more predictors and predict games on a week to week basis, using much more complicated and ensembled models built over the course of the last 10 to 15 years. However, we wanted to see how close we could get to their accuracy with our own unique predictors as trying to improve their model using their results would have been a near impossible feat in such a short time.

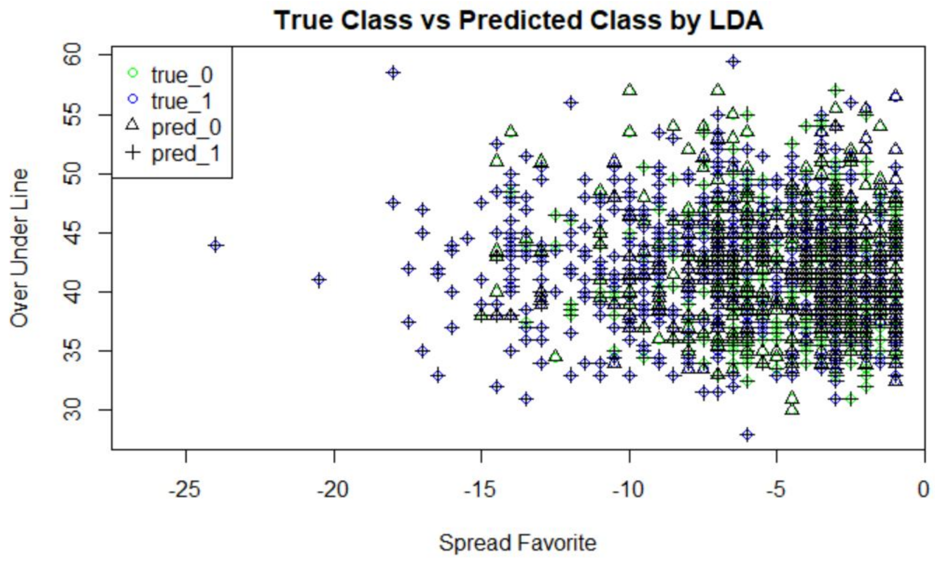
In addition to testing if we could predict the outcomes of games, we also wanted to use these results and compare them to the spread_favorite and over_under_line benchmark predictors that were also professionally done and provided in the original data set. For LDA, QDA, KNN, and logistic regression, we planned on creating classification plots of our TARGET_Home_win variable plotted against the spread_favorite and over_under_line. Our goal was to see what kind of games we accurately predicted. More precisely, we wanted to see how accurately we predicted games with smaller spreads versus larger spreads, and higher over under lines versus lower over under lines.

Box Plots:



We chose the predictors for our models by analyzing side-by-side boxplots of every variable against our binary(factor) response variable TARGET_Home_win. From here, we chose the predictors that gave the largest difference between classes when plotted against our response variable. The figure above shows the six predictors we found had the largest difference between classes and therefore the most likely to have the greatest impact on our response, TARGET_Home_wins. These six predictors are Away_team_wins_prev_5, Home_avg_pts_prev_5, Home_team_wins_prev_10, Away_avg_pts_prev_10, Home_team_wins_prev_5, and Away_avg_pts_prev_1. The predictors, Home_team_home_wins_prev_y, Away_team_away_wins_prev_y, Home_avg_pts_athome_prev_y, and Away_avg_pts_ataway_prev_y, where y represents 1, 3, 5, 7, 10, and 15, were avoided because of collinearity levels with these predictors. Further analysis was conducted using these predictors moving forward.

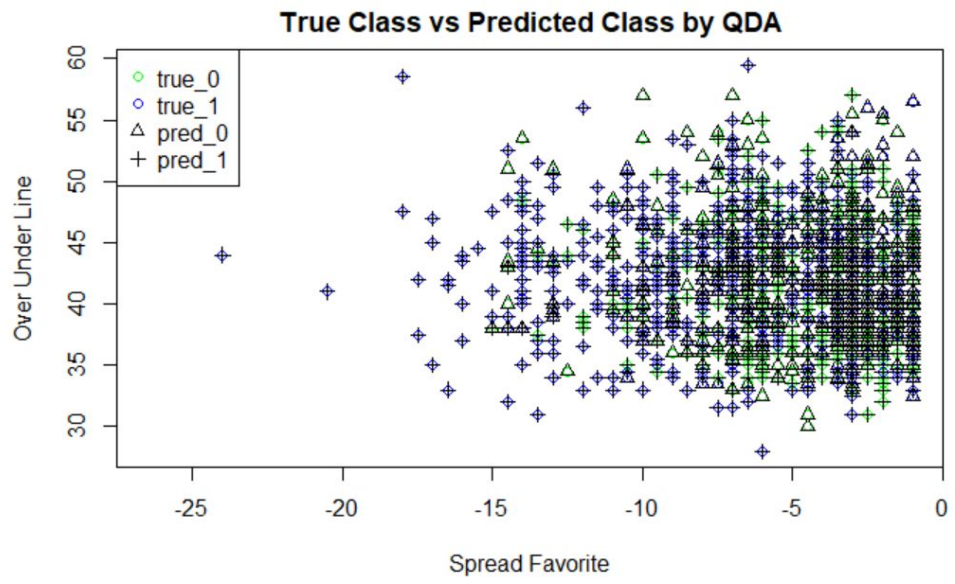
Linear Discriminant Analysis:
We started off by fitting an LDA model on our data. Our model was y = $-0.23871x_1 + 0.04790x_2 + 0.276689x_3 + -0.09169x_4 + 0.06654x_5 + 0.00187x_6$. The LDA model reported prior probabilities for the 2 classes of our response(TARGET_Home_win), 0: 0.4191157 and 1: 0.5808843. We found that this was fairly accurate as it was reported in 2019 that Home teams win approximately 55% to 60% of the time. Our training error was 0.360736 and our test error was 0.3605082. This test error tells us that from LDA, we accurately predicted the winner 63.95% of the time.



Looking at the above plot, it is quite clear that there is not a linear relationship between the over_under_line and spread_favorite. However, the plot does tell us that in our predictions of TARGET_Home_win that we predict games more accurately when the spread is larger. When the spread gets smaller, our model loses accuracy in predicting which team will win. This is expected, because if the spread between two teams is close to 0, it means that the Las Vegas sportsbooks considers the match fairly even and is predicting a close outcome. Naturally, it is difficult to predict the winner of evenly matched teams and thus, this is illustrated in our LDA plot above. We do not see any clear correlation with the over_under_line and our predictions.

Quadratic Discriminant Analysis:

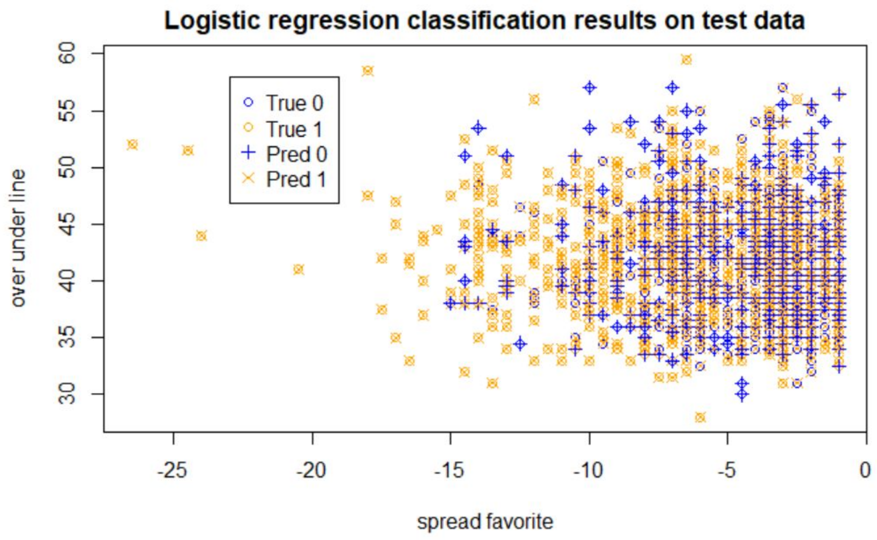
We then fit a QDA model on our data and compared our results to our LDA. The QDA summary does not report the coefficients of the discriminants. Our QDA reported the same prior probabilities as our LDA model, along with training error of 0.3625894 and a test error of 0.3695077. From this, we can conclude that our QDA model correctly predicted the expected outcome 63.05% of the time. Although they performed quite similarly, the performance statistics proved that LDA slightly outperformed QDA.



From QDA, we see that there is not a quadratic relationship between spread_favorite and the over_under_line. As seen in LDA, as the spread_favorite approaches 0, our model loses accuracy in its prediction. This plot is very similar to the LDA plot above, and once again there is no correlation between TARGET_Home_win and the over_under_line.

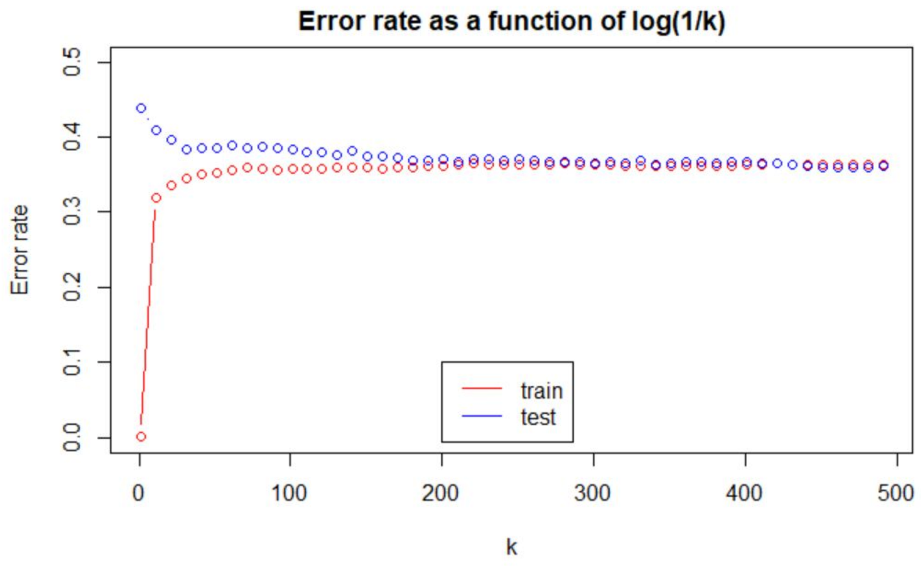
Logistic Regression:

In addition to fitting LDA and QDA models, we also fit a logistic regression model on our data. Our model was y = 0.325007 + -0.141364x_1 + 0.028806x_2 + 0.162064x_3 + -0.053607x_4 + 0.036489x_5 + 0.001116x_6. The model also reported four highly statistically significant predictors, x_1-x_4. These significant predictors were Away_team_wins_prev_5, Home_avg_points_prev_5, Home_team_wins_prev_10, and Away_avg_points_prev_10. From logistic regression, we received a training error of 0.360736 and a test error of 0.3605082, thus indicating that our model predicted the correct outcome with 63.95% accuracy. Our logistic regression model performed the exact same as our LDA model, so either model is a viable option. It is important to note, however, that LDA and QDA models have assumptions that are often more restrictive than logistic regression.

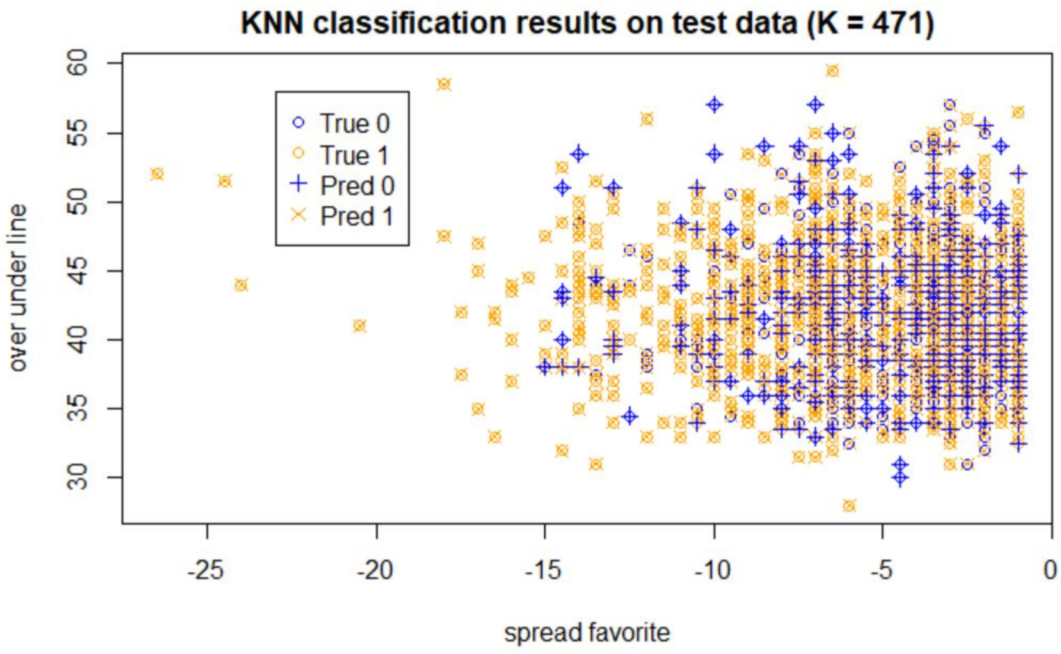


From the figure above, we can see that there does not appear to be any significant relationship between the data and the variables spread_favorite and over_under_line. The figures models very closely to the LDA and QDA plots and this is expected as our models all reported classification errors within 1% of each other. Again, this plot tells us that when the spread is large our model predicts with greater accuracy and when the spread is small, we lose some accuracy due to the inevitable uncertainty of determining which of the evenly matched teams will prevail.

K Nearest Neighbors:
For KNN, we used 50 values for K, incrementing by 10 each time. We did not want to go all the way to the number of rows due to the large amount of rows in our training data, because those large values of K would throw errors. We received the following error rate, using log(1/k) to better visualize the error rate.



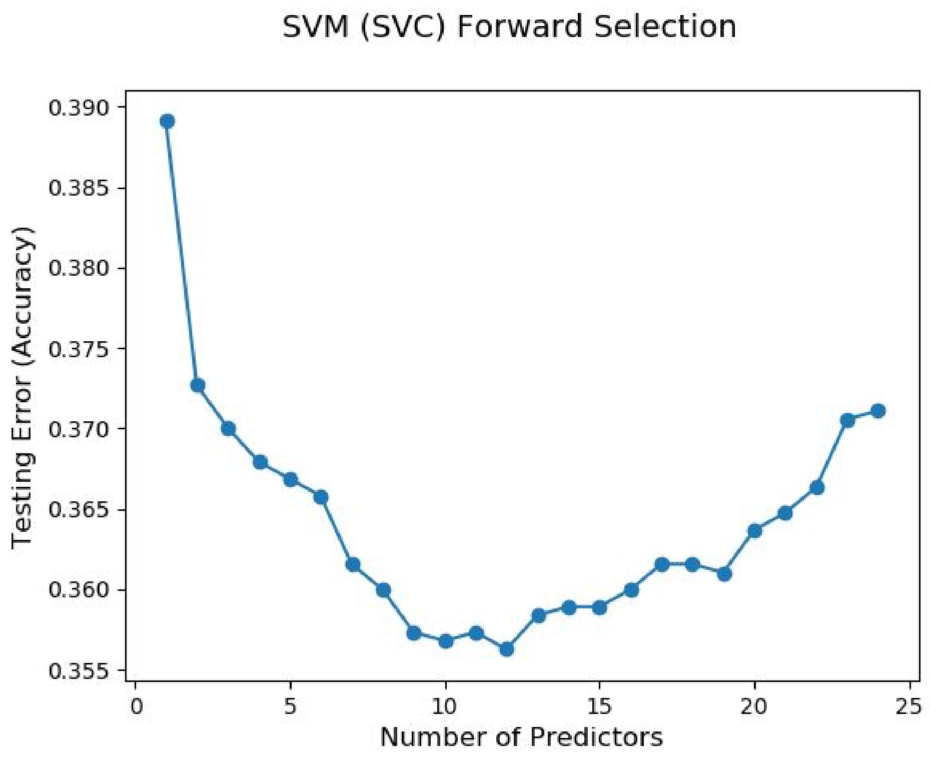
The above error rate plot determined that the the value of K that minimized test error was K = 471. When studying the above plot more thoroughly, we can see that the training error rises above 0.3 very quickly and the test error slowly decreases as K increases. It is not until K gets very large that the test error becomes lower than the training error. For K = 471, the training error was 0.3629865 and the test error was 0.3594494, giving a prediction accuracy of 64.05%. We received the following classification plot.



This plot was the most accurate of all the classification plots. The reason KNN outperforms the previous three methods is because the true relationship is nonlinear. The plot is still similar to the others, but the results are more accurate. KNN does a better job at predicting games with spreads between -15 and -5 than the other plots above. However, once the spread gets to around -5, our accuracies still decrease greatly.

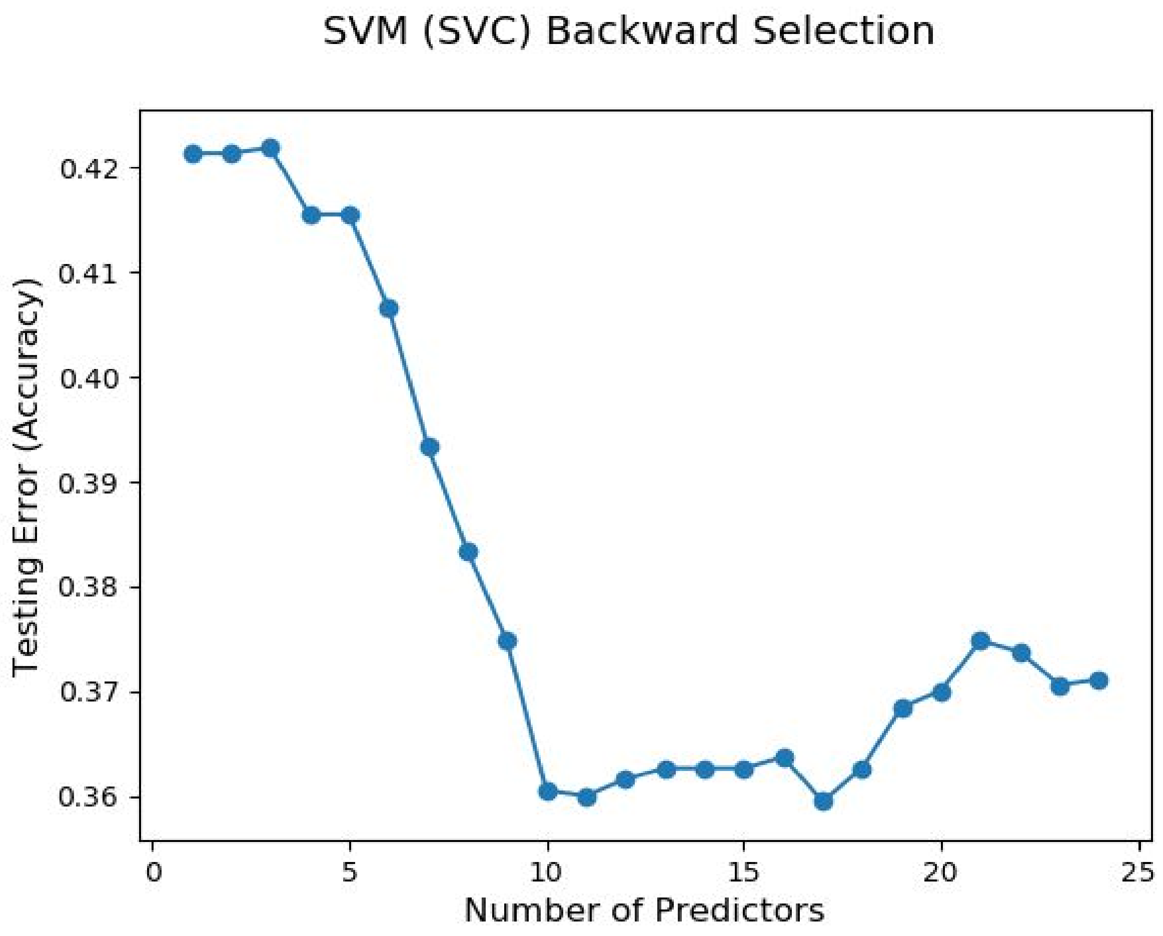
Support Vector Machine:
For the support vector machine we decided to test out two new feature selection techniques while also taking into account the analysis found from the previous implementations. Specifically, we would test forward and backward selection techniques using the set of all of our created features (Games won, points scored, home team home games won, away team away games won, home team home points scored, and away team away points scored in past 1, 3 , 5, 10, and 15 games).

Support Vector Machine (Forward):
For forward selection, the exact method of choice was to add a single predictor to the model based on the model that resulted in the highest overall testing accuracy. This was done without replacement and until the entire set of predictors was exhausted. The accuracy of each additive model were then recorded, and the most accurate was selected as the overall forward selection SVM model. The results for each additive model are seen in the following table:



As seen in the graph, the lowest testing errors were found when N = 10, 11,and 12 predictors. We selected the N = 10 instance as the test errors were only different by a matter .005 and with a lower number of predictors the model would be more robust and would be less likely to overtrain on any given training set. The overall N = 10 SVM forward model had a training error of 0.356539 and testing error of 0.356802.

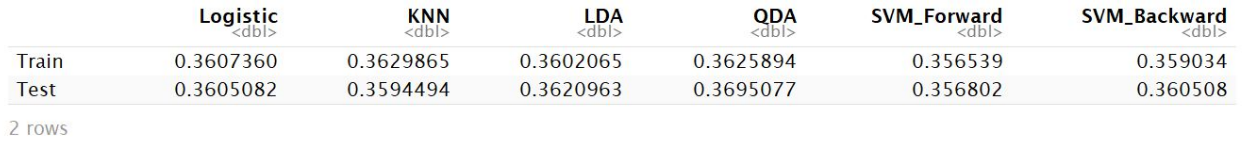
Support Vector Machine (Backward):
For backward selection, the exact method of choice was to begin with all of the predictors included in the model and to delete a predictor at each step based on the predictor whose deletion resulted in the highest testing accuracy. This was done until there was only one predictor left in the set of used predictors. The accuracies of each successive model were then recorded, and the most accurate was selected as the overall backward selection SVM model. The results for each successive models are seen in the following table:



As seen in the graph above, the lowest testing errors were again found when N = 10, 11 and 12 predictors. Once again, we selected the N = 10 instance as the test errors were only different by a matter .005, using the same logic as in forward selection. The overall N = 10 SVM backward model had a training error of 0.359034 and testing error of 0.360508.

Conclusion:

Using the training and test errors calculated from each model, we get the following table:



	Logistic	KNN	LDA	QDA	SVM_Forward	SVM_Backward
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
Train	0.3607360	0.3629865	0.3602065	0.3625894	0.356539	0.359034
Test	0.3605082	0.3594494	0.3620963	0.3695077	0.356802	0.360508

2 rows

When looking at the table above, we can see that the best model is SVM using forward selection followed by KNN. Given that the SVM, specifically with the non-linear kernel, performs the best on our dataset, this would suggest that the data is not linearly separated. This is further seen from the fact that KNN outperforms LDA, QDA, and logistic regression. We conclude that either the KNN or SVM forward selection models are the best choice for predicting NFL outcomes, because the test errors only differentiate by about 0.003. SVM forward selection is slightly more accurate, but uses more predictors. We recommend both models, depending on whether a user wants a smaller, more robust model, or a slightly more accurate model.

Previously, we mentioned that our goal was to get as close to the benchmark value that was calculated by the is_home_favorited predictor. That benchmark testing error was 0.339000. SVM using forward selection gave a best testing error of 0.356802. These values give accuracies of 66.1% and 64.32% respectively. From these values, we were very happy with our results. We can conclude that our predictions are quite accurate. Las Vegas sportsbooks use much more complicated models with more predictors and data. These models are edited weekly and take into account variables such as injuries, team statistics, player statistics, coaching staffs, and many other variables that we do not account for. The fact that our accuracies are off by less than 2% is pretty remarkable and also tells to just how difficult it really is to predict the NFL. It also suggests that the lagging features are very tell-tale when it comes to predicting NFL games.

When comparing the classification models that we created, we can conclude that KNN performed the best. These models allowed us to see what kind of games we accurately predicted. As expected, we predicted games with larger spreads much more accurately than games with smaller spreads. We would have liked to see a little better accuracy in games with closer spreads, but sports are not supposed to be predictable, specifically big games that are supposed to be very close. While there was a relationship between our predictions and the spread, there was no clear relationship with the over_under_line. There was no clear cut relationship between the accuracies of our predictions and how many points Las Vegas sportsbooks determined would be scored in the game.

Individual Contributions:
Adam: I created the models for LDA, QDA, logistic regression, and KNN, as well as their respective plots. I helped put together the presentation as well as the final report.
Nakoul: I helped Adam with the code for LDA, QDA, logistic regression and KNN. I also performed variable selection to determine what predictors were to be used in these models, as well as helped with the presentation and final report.
Jack: I focussed for on the feature engineering portion of the project, engineering the usable predictive features from the original dataset as well as helped put together the presentation and final report. I also created the model for the SVM classifier which I implemented forward and backward selection on.