Predicting the NFL

Zachary Owen and Virgile Galle

1 Introduction

As the most popular sports league in the United States, the National Football League (NFL) draws wide popular interest and is the subject of a $3 billion legal betting market each year. In addition, with the current trend toward more liberal gambling laws it is likely that in the near future a portion of the the illegal betting market which is estimated to be at least an order of magnitude larger than the current legal betting market, may move into public view. Therefore, the accurate prediction of game outcomes and point spreads is a task of concern for many in the betting industry as well as for casual fans who simply want to know the chance their favorite team has to win.

In this project we have applied machine learning techniques to a rich set of data on the NFL to predict both the binary win or loss outcomes of football games as well as the associated point spread that is the most common metric used in the legal betting market. To make our predictions we used the database *nfldb* that is made available for public use on Github by Andrew Gallant. This data set is extremely rich in that it contains play by play data for each game played from the 2009 season through the present. It includes informatino from the overall score of each game down to who was credited with making the tackle in each individual play. With this quantity and granularity of data, one of the primary challenges we address in this project is the problem of feature specification and selection. We approach this problem using our prior knowledge on important indicators of team quality in football in combination with machine learning techniques that promote sparse models.

2 Game Outcome Prediction

Our first task was to simply predict the binary win or loss outcome of football games based on data available at the time the game was played. More specifically, we wish to predict whether or not the home team will win versus the away team each game based solely on data concerning each teams' past performance. For the present, we concern ourselves solely with overall accuracy, the percentage of games in which the outcome is predicted correctly, weighting each game equally. An interesting extension would be to evaluate predictions versus the available gambling odds, but due to the difficulty in collecting this data we do not consider it here.

To train our models we will use the games played in the three NFL seasons 2010, 2011, and 2012, while using the corresponding data from the 2013 season for validation and model selection. Finally, we will evaluate the performance of our methods by their predictive ability on the first 12 weeks of the current 2014 season.

There are a number of available benchmarks to which we can compare the machine learning models that we will consider in the following sections. Perhaps the most parsimonious predictive policy is the trivial policy that takes advantage of the well-documented phenomenon of home-field advantage by simply predicting that the home team will win each and every game. For the data sets we use here this method is surprisingly effective. In the 2013 season that we use for validation, the home team won 60.0% of the games they played. For the 2014 season testing set this home-field advantage benchmark accurately predicts 58.9% of games through the first 12 weeks. Another reasonable, conceptually simple benchmark would be to predict the outcome that the 'favorite' wins

each game. This could be defined based on the fair spread implied by the betting market or by comparing the records of the two teams. However, due to the manual nature of collecting the betting favorites and the ambiguity in using teams records we did not use these benchmarks here.

The other primary benchmark that we consider in evaluating our work is that of expert predictions - those of sports writers who post predictions for every game each week of the season and log their performance over time. Even though understanding, analyzing, and predicting the outcome of football games is their full-time occupation, their ability to predict the outcome of games is, perhaps surprisingly, poor. Specifically we chose to compare the accuracy of our models against the predictions of the NFL editors for CBS Sports. In our comparisons we include the records of all the commentators whose game predictions are logged on the site to provide a sense for the general accuracy of football experts. CBS Sports was chosen since it employs experts with a representative sample of predictive accuracy but also includes Jamey Eisenberg, who has perhaps the best performance of any commentator on a major network sports website. As we will see, most of these experts have accuracy between 60.0% and 66.0%, which is only incrementally better than simply predicting that the home team will prevail in each matchup.

In addition, we also evaluate the performance of our methods in comparison to machine learning-based approaches by other teams including those of Microsoft's Cortana AI engine, the CBS Sports prediction machine, and WhatIfSports' game simulations. According to the companies, these predictors use sophisticated analytical tools in order to make their prediction and tend to achieve accuracy toward the higher end of the range of human predictors with typically accuracies between 64% and 68%.

2.1 Defining the Features

Before we are able to fit a model, we must first specify the relevant features we wish our model to take into account. To organize our data we chose to associate a set of data with each individual game played in our training, validation, and testing data sets.

Each of these instances represents an individual game characterized by four main identifying features and the corresponding label, whether or not the home team won the game in question:

- **Year**: (*int*) between 2010 and 2014. Represents the season in which the game has been played
- **Week**: (*int*) between 1 and 17. There are 17 weeks in the regular season of the NFL (each team plays only 16 due to a bye week occurring after week 4 in which each team is not schedule for a game)
- **Home Team**: (*char*) corresponding to the initials ($2-3$ letters) of the team hosting the game
- **Away Team**: (*char*) initials of team playing against the home team
- **Home Win**: ($bool \in \{-1, +1\}$) equal to $+1$ if the Home Team won the game. This is our dependent variable that we would like to predict.

These features are just those that are required to properly identify the game. They are not used directly to predict the outcome of each game.

For each team (Home and Away), we use two different kinds of features based on data from other games that team has played. One issue is that sports statistics from individual games are inherently noisy. Even the best teams and players can have bad games and conversely poor ones can show sudden, brief flashes of brilliance. What distinguishes good teams and players from the rest is the ability to consistently perform at a high lever over time. Therefore it makes sense to use features

consisting of the average of some statistic over a set of games. In order to be fair, notice that if we want to predict a game, we can only use data that would have been available at the time the game was played, meaning data from games previously played. Therefore it is natural to ask: On how many games should we take those averages? It seems clear that games that are close to each other in time will have high correlation between them, while conversely games from three seasons ago are likely to be irrelevant. Therefore we define a model variable **Game Span** ,denoted GS, which is a positive integer and represent a number of weeks (equivalently, a number of games). In English, this model variable states: In order to predict a game, we only consider features coming from the GS games played before this game. This variable is central to the performance of our model. Indeed, if GS is too small, then our model will suppose that games are only affected by a few games that do not represent the overall performance of a team. On the other hand if GS is too large, then the model will believe that games which were played a while ago and maybe in which the team and coach were completely different are still relevant to the outcome of today's game which seems unlikely.

Therefore we immediately see that we will have to perform model selection on GS as the optimal value is not obvious to find. However, before proceeding to model selection, we assume that GS = 10 as it seems to be a reasonable scale of time to consider. We will compare models using this value and then tune this value for the selected models.

The first class of features are the ones that are associated with overall performances over games and we have those for both Home and Away Teams.

- **Home Points Scored**: (*float*) the average number of points scored by the Home team in the previous GS games.
- **Home Points Allowed**: (*float*) the average number of points the GS previous teams scored against the Home team.
- **Home Turnovers**: (*float*) the average number of turnovers that the Home team gave to their opponents in the last GS games.
- **Home Turnovers Forced**: (*float*) the average number of turnovers that they gained against their opponents.

The importance of the first two features here is trivial, the more points you score, the more chance you have to win and the more points you allow, the less likely a team is to win. This idea comes from the fact that better teams are likely to score more points than their opponents on average while simulaneously allowing fewer points and the degree to which a team scores more points than its opponents should have predictive value. The last two features come from the experience of many specialists. It is commonly known that every possession is a scarce resource in football. Losing one ball to the other team on a turnover affects much more a game than most of the other plays.

The other types of features that we considered are based on play results, which represents more granular data. For each team, Home and Away, we computed 8 variables, each of whose values are calculated as the average over all the plays of all GS past games. For below we summarize the definition of the statistics in the case of the home team they are also computed for the away team in each game we consider.

- **Home Pass Eff**: (*float*) The pass efficiency is the ratio between the yards gained when a pass is attempted and the number of pass attempts.
- **Home Pass Def**: (*float*) The same value but when Home team is playing defense, so this is a measure of defensive ability versus opponents' passes.

- **Home Rush Eff**: (*float*) The rush efficiency is the ratio between yard gained during a rushing play and the number of rushing plays.
- **Home Rush Def**: (*float*) The same value but when Home team is in defense, so this is a measure of defensive efficiency versus opponents' rushes.
- **Home Pen Ag Off**: (*float*) Measures the yardage lost in penalties when the offensive team plays.
- **Home Pen For Off**: (*float*) Measures the yardage won in penalties when the offensive team plays.
- **Home Pen Ag Def**: (*float*) Measures the yardage lost in penalties when defensive team plays.
- **Home Pen For Def**: (*float*) Measures the yardage won in penalties when defensive team plays.

These parameters are often cited as significant by experts when studying the outcome of football games. Again, notice that all these measures try to take into account that each possession of the ball is a limited resource and not passing efficiently or getting too many penalties may make a difference in the result of a game.

For each training, validation, and testing game we consider, we created a record consisting of the identifying game data and these statistics for both teams for the previous GS games. Table 1 demonstrates how each games is presented in the data set we derived from *nfldb*. This makes a total of 29 features per game. We use 24 'independent' variables to predict 1 independent variable 'Home Win'.

Instance	*Year*	*Week*	*Home Team*	*Away Team*	*Home Win*
1	2013	9	OAK	PHI	-1

H/A Points Scored	*H/A Points Allowed*	*H/A Turnovers*	*H/A Turnovers Forced*
16.8/20.3	19.1/28.0	0.7/1.1	1.3/1.1

H/A Pass Eff	*H/A Pass Def*	*H/A Rush Eff*	*H/A Rush Def*
6.6/7.4	6.9/7.5	4.5/4.7	3.3/4.0

H/A Pen Ag Off	*H/A Pen For Off*	*H/A Pen Ag Def*	*H/A Pen For Def*
26.6/32.0	16.7/26.6	30.8/21.6	28.2/33.5

Table 1: Example of a instance in our database

2.2 Outcome Prediction using Game Span 10

As discussed above, we use the concept of game span in order to keep the data stationary so the coefficients of our model do not need to adjust to the number of previous games in the season and to account for a degree of continuity in the performance across seasons. As the performance of

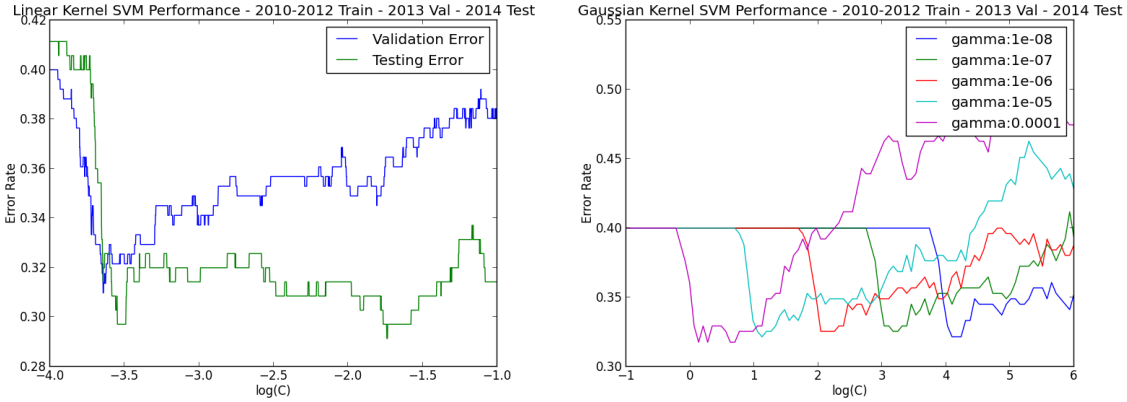
NFL teams tends to change gradually, a team's past performance, even from the previous season is indicative of future success. In this section, we discuss our model fitting procedure and results using statistics averaged over the last 10 games played by each team. This value of gamespan was shown to perform well versus other possibilities as we will discuss later.

Using this data, we fit a number of different model types including kernelized support vector machines (SVM), logistic regression with various regularizers, random forest classifiers, and AdaBoost with decision stumps on the features. All of the models were trained using the 2010, 2011, and 2012 seasons for training, validated using the 2013 season, and tested using the 2014 season. It is important to note that model selection was based entirely on 2013 performance, although we also plot test set performance this is just to help us see how analogous the validation and testing set are under our models. We will summarize the output of each of these in turn, beginning with our first, and ultimately most successful approach, which was to perform this classification using kernelized support vector machines (SVM).

As our data was clearly non-separable, we fitted both linear and Gaussian kernel soft margin SVMs to our data and tuned them using a grid in the log space of the parameters C and γ. Just as in our derivation in class, the parameter C represents the misclassification penalty in the SVM optimization problem objective function and similarly the bandwidth parameter γ corresponds with β with $\frac{1}{2\sigma^2}$ from a typical Gaussian density. We demonstrate our parameter selection method for these SVM models graphically in figure 1. Remarkably, after optimization over the parameters on the validation set, our final Gaussian kernel model and final linear model have the same performance on both the validation and testing sets and in fact they make the same predictions on the test set. For this reason in testing different values of game span we consider it sufficient to test only the linear kernel SVM allowing us to save time in computation and parameter estimation.

Our optimal fitted SVM models perform well on both the validation and testing sets with 69.0% accuracy on the validation set and with 68.6% accuracy on the testing set. Specifically these models outperform all three of the other quantitative models that we benchmarked against including those by Microsoft, WhatIfSports, and the Fox Sports Prediction Engine. Notably, our SVMs also outperform all but one of the human experts we compared with through the first 12 weeks of the 2014 season. The full results of our SVMs as well as our other models and the benchmark comparisons are presented in table 2. One interesting feature of note is that our linear SVM operates with a positive intercept term that is quite large in comparison to the coefficients on the other features. Since we are predicting whether or not the home team will win a given game means that our model takes advantage of the fact that the home team tends to win an outsize percentage of games and thus tends to predict the home team unless the visiting team is far superior in which case it will switch. It is remarkable that this rather simple principle tends to outperform the experts who perform NFL prediction as part of their job.

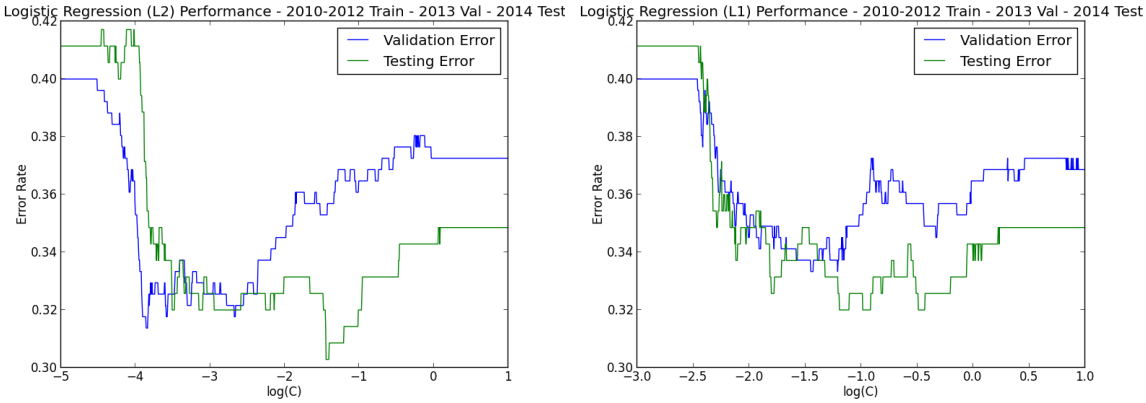
We also tried a variety of other models on this prediction task, but in general these tended to slightly underperform the linear and Gaussian SVMs discussed above. For instance we tested logistic regression with both L_2 and L_1 regularization. We observed that L_1 regularization outperformed L_2 on the test set, despite demonstrating worse performance on the validation set. This is in keeping with the idea that by forcing parameters to zero L_1 regularization better removes noisey and colinear features from consideration. Interestingly, the L_1 regularized model ends up assigning non-zero weight only to the the four features involving points scored and a couple of the terms involving penalty yardage. This is likely because many of our feature such as rushing and passing efficiency are highly correlated with points scored and as such using them in a logistic regression model could introduce additional noise in prediction. However using solely these features in an SVM model results in inferior performance, indicating that if used correctly they still have some predictive value. An illustration of our parameter selection technique for these models is provided in figure 2. Note that the parameter C here is the inverse of the regularization constant. In the feature selection section we



(a) (b)

Figure 1: Parameter selection illustration for our SVM with (a) linear kernel (b) Gaussian kernel. For clarity in the case of the Gaussian kernel, we only plot a subset of the scale parameters in the grid and do not plot testing error.

fit the linear SVM model on the features selected by the L_1 regularized logistic regression and report results. We also tried various non-parametric techniques such as AdaBoost with decision stumps and Random Forest, but we did not find a model of this type with comparable performance to our SVMs.



(a) (b)

Figure 2: Parameter selection illustration for logistic regression using (a) L2 regularization (b) L1 regularization.

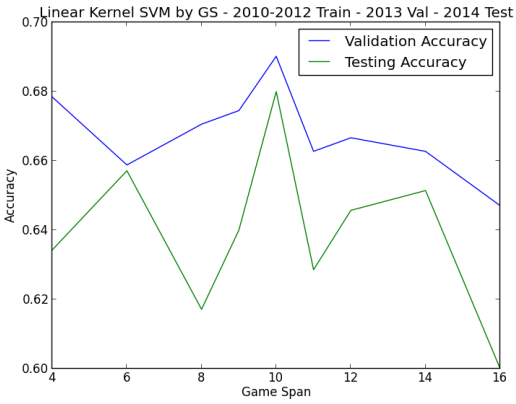
2.3 Optimizing Game Span and further analysis

For ease of exposition we began with a discussion of our predictive models using data about each teams' performance over the last 10 weeks. After observing that the linear kernel SVM appeared to offer superior performance for each value of gamespan that we tested, in order to select an optimal game span we fit an optimized linear kernel SVM to data using each game span in $\{4, 6, 8, 9, 10, 11, 12, 14, 16\}$. Then by observing the performance of each such optimized SVM on the validation set as discussed above, we found the optimal such game span to be 10. This also turned out to be the best game

Benchmarks	Parameters	Validation Record 2013	Testing Record 2014
Home Wins	-	153-102 (60.0%)	103-72 (58.9%)
Pete Prisco	-	-	119-56 (68.0%)
Jason LaCanfora	-	-	104-71 (59.4%)
Will Brinson	-	-	102-73 (58.3%)
Josh Katzowitz	-	-	110-65 (62.9%)
Ryan Wilson	-	-	97-78 (55.4%)
John Breech	-	-	112-63 (64.0%)
Dave Richard	-	-	117-58 (66.9%)
Jamey Eisenberg	-	-	126-49 (72.0%)
Microsoft Cortana	-	-	117-58 (66.9%)
CBS Prediction Machine	-	-	116-59 (66.3%)
WhatIfSports.com	-	-	115-60 (65.7%)
Models	Parameters	Validation Record 2013	Testing Record 2014
SVM - Linear Kernel	$C = 2.27\text{e}^{-4}$	176-79 (69.0%)	120-55 (68.6%)
SVM - Gaussian Kernel	$C = 6.87, \gamma = 1.68\text{e}^{-5}$	176-79 (69.0%)	120-55 (68.6%)
Logistic Regression (L2 Reg.)	$C = 1.38\text{e}^{-4}$	175-80 (68.6%)	111-64 (63.4%)
Logistic Regression (L1 Reg.)	$C = 3.54\text{e}^{-2}$	170-85 (66.7%)	115-60 (65.7%)
AdaBoost Decison Stumps	LearningRate=0.226	163-92 (63.9%)	113-62 (64.6%)
Random Forest (1000 it.)	-	148-107 (58.0%)	101-74 (57.7%)

Table 2: Performance of the benchmarks and our models using a game span of 10 on the validation and testing data sets. Parameters of our models are selected by performance on the validation set from the 2013 NFL season.

span to select for the testing set as well, although we did not use this as criteria for model selection, in hindsight it validates our decision and demonstrates that our validation set accurately reflects the character of the testing set. Because of the superiority oberved for models based on a game span of 10 we restricted our attention to this game span while analyzing our models in detail. A plot of this analysis is given in figure 3.

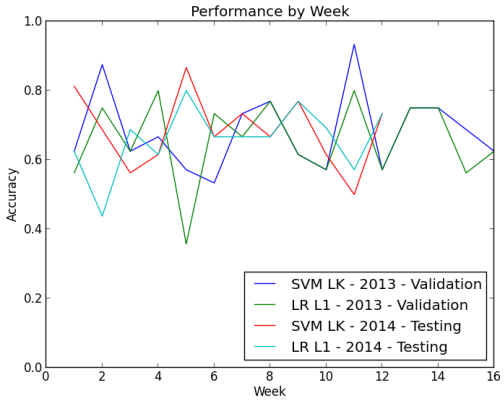


(a)

Figure 3: Selecting game span by optimizing over regularization parameter C in the linear kernel SVM seperately for each gamespan. We observe that the optimal game span in terms of both test and validation accuracy was 10, the case considered in the previous section.

One reasonable concern to have about our model is that the use of a game span that wraps

around the beginning of the current season to the end of the previous season is that it does not take into account the perhaps discontinuous nature of team quality over the off-season. After each season, teams lose and acquire players through free agency, draft new players, and there can even be coaching and other staff changes. It is quite reasonable to expect that these factors would influence a teams' chances of winning, perhaps strongly, between seasons so that weighing the last season as heavily as the game span model does for games in the early part of a season, may be a poor modeling choice. We wanted to test whether or not there was an observable trend of increasing accuracy as the season progresses, which would indicate that perhaps our modeling assumptions were flawed. However, after examing accuracy by week for two of our more accurate models - the linear kernel SVM and the L_1-regularized logistic regression, both fitted using game span 10 - we saw little change in performance as the season progressed. For the SVM model the accuracy appears to be quite stationary over the course of the two seasons. However, for the logistic models, although we do not observe a strong upward trend, we do note that the worst week for this model occurred early in both seasons, perhaps indicating that in this case there is an off-season effect. Modeling this effect would require additional data and so we are not able to consider it here. An interesting extension would perhaps be to use some smoothing technique so that more recent games are weighted more heavily than past games. Due to time constraints and the good performance of our fixed-window model we did not focus on this in favor of instead analyzing our second task of predicting the point spread.



(a)

Figure 4: Performance of two of our best models, a linear kernel SVM and a L_1-regularized logistic regression model, by week of the season on the training and validation sets. From this limited data there appears to be no clear trend of increased accuracy as the season progresses.

3 Point Spread Prediction

As we mentioned in the introduction, most of the bets allowed today in the US are based on a measure called the 'Point Spread'. Here is an example of a typical betting proposition:

New England Patriots -7 vs. Philadelphia Eagles

The team that is thought to be better is called the *favorite*, and since they are expected to win the game, to make the bet fair, they must score more than just a single point more than weaker team. Here the favorite is listed with a minus sign which means that New England is expected to win by more than 7 points. In other words, an Eagles bettor would win if Philadelphia wins by any amount of points or loses by less than 7 points. This is equivalent to saying that the following bet is fair:

Philadelphia Eagles +7 vs. New England Patriots

Therefore from now on, we always give the points spread for the home team and it is negative if the home team is favorite.

Notice that this problem is more general and therefore more difficult than the one we considered in Section 2. Indeed from answering the point spread question, we can infer whether or not a team win is likely to win. In the case of predicting a point spread, we now have a 'continuous' outcome. However, note that in reality it is always an integer. Therefore this problem is most suited to the techniques we learned such as Regression with different types of regularization.

3.1 Defining the Features

For this task, we consider the same set of features presented in Section 2.1. However since we did the game outcome prediction task prior to the piece we introduce here, for this task we introduce 4 new features per team that could also be relevant. These features are percent of conversion of 3^{rd} and 4^{th}. For 4^{th} downs, we use the Laplace correction in order not have to divide by zero and it permits to have more accurate estimates specially in the case of 4^{th} downs that are not that frequent. These conversions are important because they measure a teams ability to retain possession of the football on offense and therefore increasing their chances of scoring and winning as well as the ability of the defense of the team to prevent the opponent from continuing their drives.

- **Home 3D Eff**: (*float* $\in]0,1[$) The percentage of 3^{rd} down attempts converted.
- **Home 3D Def**: (*float* $\in]0,1[$) The percentage of 3^{rd} down attempts converted against the home team.
- **Home 4D Eff**: (*float* $\in]0,1[$) The percentage of 4^{th} down attempts converted (with Laplace correction).
- **Home 4D Def**: (*float* $\in]0,1[$) The percentage of 4^{th} down attempts converted against the home team.

We will see in the feature selection if those parameters make a difference in our models.

3.2 Model Selection

Since our problem is most analogous to a regression problem we will compare three of the most common approaches to such a problem

1. OLS Regression
2. Ridge Regression
3. Lasso Regression

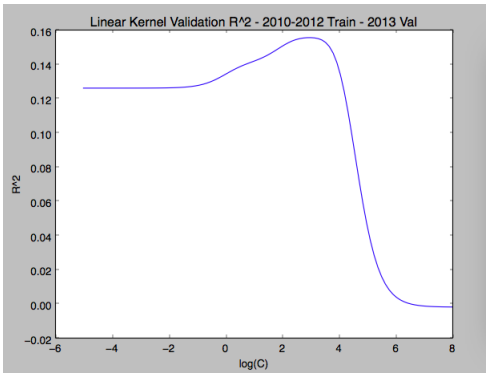
Furthermore, we only exploring the case for linear kernels as the previous part suggests that introducing kernels does not help much with our data.

For the regularized models that we consider, we proceed to feature selection on the C for both penalty terms. We use the common statistical quantity R^2 as the measure of performance of the regressions. It is defined as

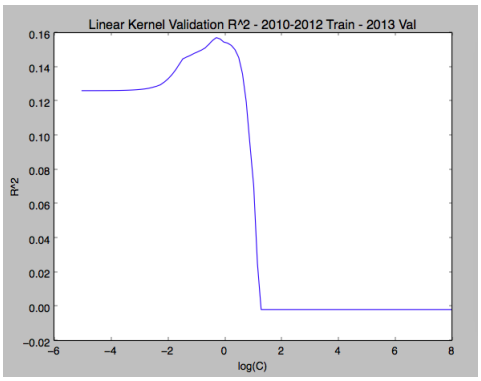
$$R^2 = 1 - \frac{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{y}_{(i)} - y_{(i)}\right)^2}{\frac{1}{n}\sum_{i=1}^{n}\left(\overline{y} - y_{(i)}\right)^2} \tag{1}$$

where $\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_{(i)}$ is the average of the outcomes of our data points. In essence, R^2 measures the performance of our regressions compared to the benchmark of predicting the average of the training set.

The results of our feature selection process for the ridge regression and lasso regression models are present below in Figure 5, which shows the curve to perform model selection for both regressions. The corresponding figures are presented in Table 3.



(a) R^2 as a function of C for Ridge Regression



(b) R^2 as a function of C for Lasso Regression

Figure 5: Feature selection in both Regressions

Type of Regression	R^2 *out of sample*	*best C*
OLS	0.126	
Ridge	0.157	1023.53
Lasso	0.159	0.53

Table 3: Feature Selction

By examining Table 3, one can see that it seems that the Lasso clearly outperforms OLS and outperforms ridge regression by a small amount. Therefore, for simplicity, from now on we focus on the case for Lasso regression. Moreover, this is going to help us as Lasso does feature selection naturally by reducing the weights to zeros for insignificant features. We will also examine the method of backward selection and compare the results with the weights obtained with Lasso (see Table 4).

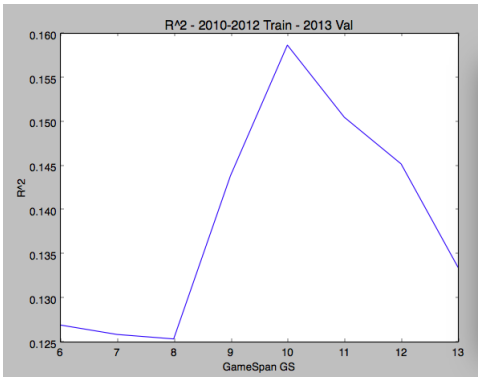
Variable	*Coefficents*	*Variable*	*Coefficents*
Home Pts Scored	-0.698	*Home Pen Ag Def*	0.008
Home Pts Allowed	0.409	*Home Pen For Def*	0.007
Away Pts Scored	0.514	*Away Pass Eff*	1.15
Away Pts Allowed	-0.238	*Away Pen Ag Off*	-0.069
Home Pen Ag Off	0.040	*Away Pen For Off*	-0.127
Home Pen For Off	-0.035	*Away Pen For Def*	-0.079

Table 4: Non-zeros regression coefficients given by Lasso regression

We would like to highlight two points. First, we see that all the signs of the coefficients except for a few (Home Pen For Def and Away Pen For Off) are in line with the intuition of the game. Indeed the more penalty the home team gets in its favor, this should help the team win. However, we see that all the penalties are quite correlated. Therefore it is possible that signs are inverted in some variables.

GS	R^2 *out of sample*	*best C*
6	0.127	0.53
7	0.126	0.72
8	0.125	0.72
9	0.144	0.53
10	0.159	0.53
11	0.151	0.53
12	0.145	0.39
13	0.133	0.39

(a) Table of Values



(b) Graph

Table 5: GS Selection

Second, we see that the variable Away Pass Eff is significant in this model but not Home Pass Eff. This might be strange as we could think that if one feature matters, then the one for the opposite should matter too. However, this can be explained by the fact the following intuition. It seems that playing at Home is very important. The Home team only loses if the team that is visiting is really good, so we only need the feature of the away team. Moreover, it is possible that actually Home Pass Eff is significant. But since the Lasso Regression is taking all the features into account and most of them are correlated, it might set its weight to zero even though it is significant. We will see how it behaves when we do feature selection. Upon suppressing features, it may become significant.

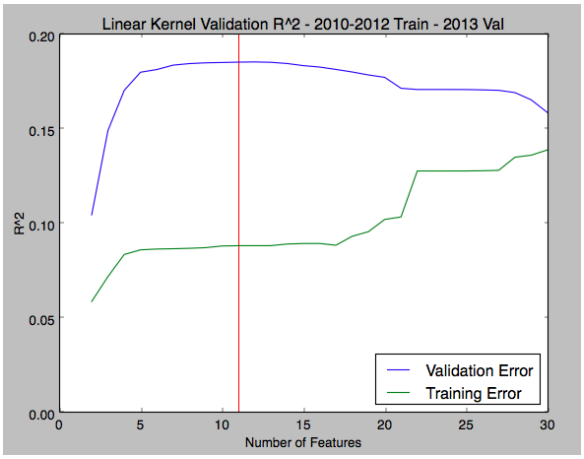
3.3 Selecting the optimal GS and Backward Feature Selection

Now as in Section 2, we need to figure which GS to use with our Lasso Regression. Therefore for each $GS \in \{6, 7, 8, 9, 10, 11, 12, 13\}$, we optimize the coefficient C in the Lasso objective using the validation set and take the one that maximizes the R^2 out of sample.

Given Table 5, it seems that again the best solution is $GS = 10$ which is the one that we had intuitively chosen. Second, we can see that the error does not seem to be convex with GS which is perhaps counter intuitive and difficult to explain. Therefore from now on we take $GS = 10$.

So far we considered that all the features we defined should be used in determining the regression weights. However, we know that there is a tradeoff between accuracy and complexity. Therefore we are going to operate a backward feature selection and see how the R^2 out of sample evolves. What we do is that we suppress the feature that makes the R^2 increase the most (or decrease the least at the end). Doing this we obtain the following curve in Figure 6.

As expected, as the number of feature grows the training R^2 increases since we can fit the data better. The validation R^2 first increases and then decreases as we are overfitting the data. This helps



(a)

Figure 6: Features Selection in the Lasso Regreassion with $GS = 10$

us doing our feature selection by optimizing over the validation R^2 and choosing only 11 features as opposed to 28 at the beginning. This makes our model more interpretable, decreases the number of correlated variables and helps us to avoid overfitting our data. Now we give our final model with the weights given by the Lasso Regression on the 11 selected features. By doing so we increase the validation R^2 to 0.186.

Variable	*Coefficents*	*Variable*	*Coefficents*
Home Pts Scored	-0.664	*Home Rush Eff*	0.255
Home Pts Allowed	0.463	*Home Pen For Off*	-0.050
Home Turnover Forced	-1.488	*Away Pass Eff*	0.266
Away Pts Allowed	-0.354	*Away Rush Def*	-2.284
Away Turnovers	1.961	*Away Pen Ag Def*	0.388
Home Pass Eff	-0.868		

Table 6: Non-zeros regression coefficients given by Lasso regression after features selection

There are many interesting things that these features tell us about football given the data that we have. Most of the coefficients make sense except for instance, Away Pen Ag Def because, the more the away defense commits penalty, the more likely the home team is to win. This can again be explained by the remaining correlation between the coefficients.

Most importantly, these features help us draw the conclusion that it seems very important to have a strong offense when you play at home and that when a team plays 'Away', a rigorous defense with no turnovers, a strong defense a rush and few penalties against you when you defend seems to be the recipe to be able to win such an away game. Note that Away Pass Eff is still significant meaning that you still need to have a decent attack to win.

Finally, we remark that the coefficients are very different from the ones selected by Lasso when all the features are given to it. This means that doing feature selection can help increase the accuracy of your model.

3.4 Benchmark on the Test Set

Now we compare our model with actual data from Las Vegas spread on this year NFL 2014 (source *http://www.cbssports.com/nfl*). We suppose that we are an betting agency that gives spreads on every game like Vegas.

Until week 12, there were 176 games played. We report Vegas and our accuracy on Table 7. We can see that on this season the model is more accurate than Vegas predictions.

	Our Model	*Vegas Odds*
Accuracy	0.574	0.5

Table 7: Accuracies on the 2014 NFL season through week 12

Now we can take the point of view of a gambler who wants to bet against Vegas spreads. We use the following rule. Let P be our prediction of the spread and V vegas prediction. If P and V are the same sign and $|V| \leq |P|$, then that means we agree with Vegas. Otherwise we bet against Vegas. Then if Vegas is right and we bet for Vegas, or Vegas is wrong and we bet agains it, we 'win' this game. Otherwise, we lose it.

Table 8 presents our record for season 2014 through week 12. It appears that using this model we would have win money overall through this season. Also notice that the best professional sports writer from CBS sports, again Jamey Eisenberg, has a record of 93/83 over the first 12 weeks. All the others are 50% or less.

	Wins	*Losses*
Record	98	78

Table 8: Record on betting on the 2014 NFL season through week 12

4 Simulating Games by Kernel estimation of densities

Our last part concerns a related subject which is the simulation of games. It may seem unrelated to our previous work but it is actually quite similar at least in its goals and the motivations. Due to time constraints we were not able to fully complete this section but we describe some of our motivations, assumptions, and the machine learning-related work we have completed below.

4.1 Motivations

One might wonder why someone would want to simulate a game, but there are actually many applications. One of the most famous examples of sports simulation can be found in video games which focus on team management. In these games the player is the GM of a team, buys and sells players while he cannot play directly each games. Therefore the video game has to simulate those games given the players of each team in a way that is as close as possible to reality.

Another application is again betting. Imagine that a game is a random variable (either Bernoulli for Win/Loss or integer for the Spread). The game that is going to be realized in real life is only one realization of this random variable. To get information on the likelihood of this realization, one might want to simulate this game and see what might be a probable outcome. For instance, we can simulate 1000 times a future game and see what is the average spread and chose this as our decision.

Other applications include also situations for game has already been played. Coaches might be interested to see what could have happen if they decided to go for another type of play. A last one could be to make real time decision making for instance going for a play or another by simulating outcomes during a game.

4.2 Assumptions

Of course, one might see that each game depends on an nearly infinite number of factors, some of them being nearly impossible to catch (for e.g the noise in the stadium). Therefore it is necessary to make some assumptions that simplifies the problem.

We made the following series of assumptions:

Play Calls First, we see that we need to predict what are going to be the play that each team is going to make throughout the game. There are several possibilities depending on the position of the play on the field. We consider that if the play is a 1^{st}, 2^{nd} or a 3^{rd} down, then the team on offense can either rush or pass the ball. Therefore we model this decision by a Bernoulli variable which depends on the yardage to gain to obtain a first down (for instance the probability of rushing on 1^{st} down and 14 yards to go is not the same that 3^{rd} down and 1 yard to go). We suppose that each team is independent of each other in this process. So we need to have access to those parameters for each team, each down, each position.

Since each position might be too complicated, we group the situations in the following 'buckets' of situations:

- 0 to 2 yards to go
- 3 to 4 yards to go
- 5 to 7 yards to go
- 8 to 9 yards to go
- 10 yards to go
- 11 or more yards to go

This decomposition is purely subjective and based on our knowledge on the game. Finally, we need to have access to the parameters of Bernoulli variables for each team, each down and each bucket.

Finally for the case of 4^{th} down, we suppose that the offensive team has to choice: either punting or kicking the field goal. This is also a bernoulli variable only based on the overall position on field (which yard line). Notice that here we do not allow for going for a 4^{th} down.

Plays and distributions on the distance Now that we made the decision on the play, we need to predict how many yards the offensive team is going to win with this play. For rushing and passing plays, it can vary continuously in the integers. For field goals, it is again a Bernoulli variables of scoring or not depending on where the team is on the field (between 0 and 30, 31 and 40, 41 and 50 or 51 or more). For punts, the distance is also continuous, then we simulate with a Bernoulli if the opponent team returns and if so how many yards do it gains also by a continuous outcome.

We suppose that for field goals and punting plays, we suppose that this is common to every team to simplify the model as most of those plays are pretty regular and similar to all teams. however for rushing and passing plays we need distributions for each team.

Learning Now we have to learn those parameter from past data. Our training data are all the games of season 2013 before week 13. Our test set is the games after week 13.

All the Bernoulli parameters can be learnt by maximum likelihood: for the ones where we have a lot of observations (punting and field goals since we use the data for all the teams), we use the no correction but for the play call distribution, since the data is quite sparse we use the Laplace correction.

The very interesting part is the estimation of the 'continuous' densities (we consider for simplicity that those are continuous and we round them up to the closest integer). In order to do so, for each team, we construct the appropriate histograms and fit them using kernel density estimation. We use the Gaussian kernel and we use cross validation to optimize the bandwidth (taken a priori in the set $\{0.5, 0.75, 1, 1.25, 1.5, 1.75, 2\}$). Again the distribution is very dependent to the position of the play. Thus we group together different types of plays: For passing plays we look if there is between 0 and 4 yards to go, between 5 and 9 yards to go, 10 yards to go and 11 or more. For rushing plays the buckets are (0,2), (3,4), (5,7) and (8 or more). For each situation and each team, we have learnt a distribution on our train set.

Time of plays We suppose that the time of plays are deterministic. Each plays takes the same amount of time. If it is completed then there is extra time spent. We stop the simulation when the time reached 40 minutes.

5 Conclusion

In this machine learning project, we tackled the popular problem of predicting outcomes of NFL games. We first focused on the simpler problem of predicting the winner of a game using classification techniques. We were able to construct an linear kernel SVM which outperformed the machine learning-based models of three companies as well as most of the sports writes from CBS sports in predicting the outcome of the first 12 weeks of the current NFL season. We then built regression models which predict the point spread of a game, which is the main object of betting interest. Notably, this model was able to "beat" the spread in the 2014 season test set, making fewer errors than any of the experts we consider. Had we bet equal money on each game based on these predictions we would have been able to turn a profit. Our work in this project demonstrates that machine learning techniques are applicable to problems in sports and can boast performance that exceeds that of nominal experts. Beyond prediction, our work demonstrates that these machine learning techniques can also be used to gain insight into the underlying factors that are most indicative of future NFL success.

5.1 Division of Labor

Finally, we briefly describe how the work on this project was divided amongst the team. We break down each major task involved in the project below.

- Find and organize a data set.

  We both collaborated in searching for the initial database. Together we developed the game span system for making the data more manageable. Zach then handled the initial processing of the data from *nfldb* into more ergodic files that we could use for machine learning.

- Complete win or loss prediction task for the outcomes of individual games.

  The coding and analysis for this task was largely designated for Zach. Virgile stayed up to date on this element and assisted by suggesting models and points of analysis to consider.

- Extension of the models above to the regression setting for points scored prediction.

  Virgile was primarily responsible for the coding and analysis of point spread prediction. Zach was following along the whole time, contributing advice about features and models to consider.

- Code football game simulation tool with space for parameter inputs that we estimate and game setting flexibility. Simulation parameter estimate computation and validation.

  Although due to time constraints we were not able to completely finish this section both of us were able to contribute something of value. Zach coded a basic, working football game simulation, while Virgile applied kernel density estimation to get realistic parameter values. Integrating these together remains for future work.