NFL Betting Strategies

Modeling Profitable Approaches to Sports Betting

Georgia Institute of Technology

ISYE 7406 – Data Mining and Statistical Learning

November 26th, 2023

Project Group 42

Shekinah Jacob – sjacob34@gatech.edu - 285
Zachary Jorgenson – zjorgenson3@gatech.edu - 113
Giovanni Marrero – gmarrero@gatech.edu - 934
Daniel Ricci – dricci6@gatech.edu – 957

Abstract

Sports betting has gained widespread popularity across not only the United States, but also the world over the last decade. The most popular sport to bet on in the country is football, as evidenced by the National Football League making nearly $20 billion during the 2022 season. The problem for many bettors is how to consistently win and defeat “The House”, otherwise known as the line that Vegas sets for a particular game. However, with many years of NFL data, modeling techniques can be used to identify weak lines or spreads that were incorrectly priced. This paper will explore NFL game data going back the past 10 seasons, with the goal being to build a predictive betting model to try and identify these inefficiencies in the betting market to help an individual bettor earn a positive return over time. Using a handful of different classification models (logistic regression, quadratic discriminant analysis, and random forest) that were cross-validated and fine-tuned with different parameters, the model with the lowest testing error appears to be a logistic regression approach. We then used this model on our testing dataset on certain games that fit our established criteria (a team our model predicted to win but was an underdog in the betting market). If we had bet $100 on each game that fit the aforementioned criteria, we would have seen a profitable return of $1,770 over the course of the last 10 seasons.

Introduction

It is estimated that $100 billion USD are wagered legally across the course of a single NFL season (Wienerkur, 2023). As sports betting expands in the United States as more states legalize and more sportsbooks launch, sports gambling will continue to grow as a significant sector of the American economy. The legalization of sports betting remains a controversial topic, due largely to the fact that most bettors will see a negative return in their betting efforts, since sportsbooks offer betting odds with statistical probability against bettors winning. For example, if a sportsbook is offering -110 odds Moneyline on both teams for a single game, this means that the sportsbook has set this game at 50-50 actual odds for either team to win. The -110 suggests that a wager of $11 will return $10 in profit, plus the original wager. The expected return in this situation is:

($10 win)(50% chance of win) + ($11 loss)(50% chance of loss) = - $1.00

Since sportsbooks take this small portion of each side of this bet ($1 in the above scenario), also known as the Vig, a bettor must accurately bet 11 out of 21 events, or 52.38% of the time, to break even. Our objective is to create a model which can accurately predict the winning team in NFL regular season games. To return a profit when sports betting, we must predict the accurate team and also beat the Vig that sportsbooks put in their implied odds when offering betting lines. In the instance of having a well-performing model that can accurately predict the winner of a sporting event, we can then check the performance of our model by comparing the profit or loss that would be returned over a sample of games. If this model can accurately predict the winner of 50-50 games 75% of the time, we would then have a new expected return of:

($10 win)(75% chance of win) + ($11 loss)(25% chance of loss) = + $4.70

Our objective is to develop a model that can successfully predict the winner of NFL regular season games well enough to produce a portfolio of bets that generate a profit, successfully beating the sportsbooks and their unfavorable odds.

Most of our relevant data was readily accessible via the *nflreadr* package. Using the provided datasets, we created new variables, **home_team_win** & **away_team_win**, based on the scores from that week. From there, we created lagging variables **home_team_won_last_week** and **away_team_won_last_week** to use in our modeling approaches as a factor of momentum and recent team success. The remaining data came from FiveThirtyEight – a predictive analytics media source – which provided ELO ratings, an advanced metric based on recent team performance.

There is an opportunity to create a model that generates accurate predictions for winners of NFL regular season games. We will create several models using different combinations of predicting variables. With these subsets of variables, we will then create models using logistic regression, random forest, and QDA methods. We will then use Monte Carlo Cross-Validation to determine the best performing model of those generated. Once we have our best performing model, we can then compare our odds versus the odds that sportsbooks provide as well. We will attempt several different betting strategies using the results from our model. Using our test data & the current NFL season, we can then

determine the success of betting on games that we determine to be good candidates for positive expected value.

First, we will bet $100 on the predicted winner of every game. This return may provide a negative return since we are likely exposing ourselves to high risk by betting on games that may have implied odds of a team having. The second betting strategy that we will employ is increasing our threshold to where a team is given an 80% chance to win according to our model. Finally, we will then test betting on only games where the team that our model predicts to win is actually suggested to lose according to the sportsbooks. By utilizing various betting strategies, we open the opportunities to determine how to optimize our betting strategies to return profit.

After generating our models, we quickly realized that logistic regression would be our best modeling approach to predict the winner of a game. Also, we were most profitable when employing our betting strategy of only betting underdogs who our model projected to win. We believe that there would be a great deal of value in expanding similar approaches that bet on teams that have a higher probability of winning than their odds suggest. This way, we would be able to take advantage of any discrepancies in the odds offered, that our model can capitalize on, assuming our model has accurate predictability.

We will discuss our original data sources and then the cleansing and transformations necessary to work with our data. Then, we will discuss findings from our data from our exploratory data analysis, discussing the behaviors of the variables that we have available to us to build our models. We will then discuss the methods that went into building our 15 models as well. After the explanation of our modeling approaches, we will then discuss the results of the models, both tested once and then again with Monte Carlo Cross-Validation. Upon determining our best model, we then will discuss how this translates into several different betting strategies and how this model could be utilized to generate profits, even against the negative odds that sportsbooks provide. Upon reaching our conclusions, we will examine the lessons learned throughout this project.

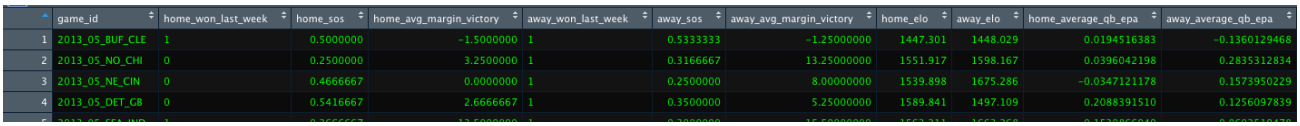
Data Sources

The data is extracted from the *nflreadr* package and it contains all NFL game data, including team information, weather, location, final score, betting odds, starting quarterbacks, margin of victory, if the contest was a divisional game, etc. The rest of the data set is comprised of team metrics up to that point of the season prior to the game beginning. Some of these include quarterback expected points added per play up until that point in the season, a team's strength of schedule entering that game, and lastly ELO ratings, which is a rating based on the team performance up until that game, with recent performance more heavily weighted. Below are snapshots of the entire data frame with the individual game data features in the first screengrab, and team metric variables in the second image.

Individual Game Data:

	game_id	gameday	season	week	home_team	home_score	away_team	away_score	home_rest	away_rest	div_game	temp	wind	home_moneyline	away_moneyline	spread_line	home_team_win
1	2013_05_BUF_CLE	2013-10-03	2013	5	CLE	37	BUF	24	4	4	0	75	6	-185	166	3.5	1
2	2013_05_NO_CHI	2013-10-06	2013	5	CHI	18	NO	26	7	6	0	60	10	-115	104	1.0	0
3	2013_05_NE_CIN	2013-10-06	2013	5	CIN	13	NE	6	7	7	0	68	7	127	-140	-2.0	1
4	2013_05_DET_GB	2013-10-06	2013	5	GB	22	DET	9	14	7	1	59	10	-445	381	10.0	1

Team Metric Season Data:



	game_id	home_won_last_week	home_sos	home_avg_margin_victory	away_won_last_week	away_sos	away_avg_margin_victory	home_elo	away_elo	home_average_qb_epa	away_average_qb_epa
1	2013_05_BUF_CLE	1	0.5000000	-1.5000000	1	0.5333333	-1.25000000	1447.301	1448.029	0.0194516383	-0.1360129468
2	2013_05_NO_CHI	0	0.2500000	3.2500000	1	0.3166667	13.25000000	1551.917	1598.167	0.0396042198	0.2835312834
3	2013_05_NE_CIN	0	0.4666667	0.0000000	1	0.2500000	8.00000000	1539.898	1675.286	-0.0347121178	0.1573950229
4	2013_05_DET_GB	0	0.5416667	2.6666667	1	0.3500000	5.25000000	1589.841	1497.109	0.2088391510	0.1256097839

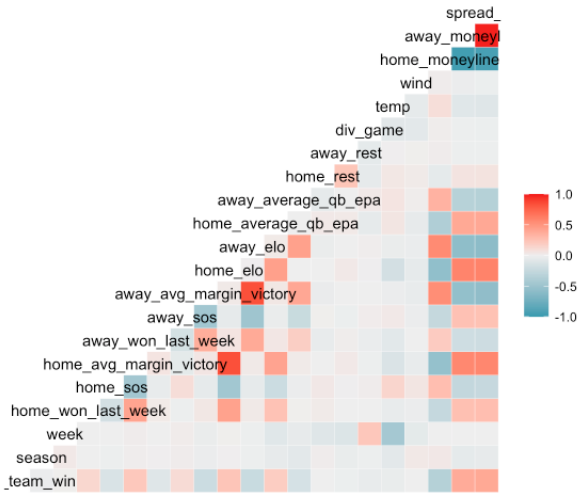
Data Processing and Feature Engineering

The primary data contains 1961 observations with 26 variables from the 2013 to 2022 seasons. Our dataset did not have a lot of missing values in it to begin with. We saw 3 missing values in **home_elo** and **away_elo,** 29 missing values in **home_average_qb_epa**, and 24 missing values in **away_average_qb_epa**. We handled these missing values using a form of median imputation, replacing all occurrences of missing values within the numerical variable data with the median value of that variable. We did see a much larger section of our data with missing values around temperature and wind speed. There were 592 missing values in these fields to be exact. However, after further investigation, all these games with missing temperatures and wind speeds were played at indoor stadiums where there was no wind present, and the temperature was in a controlled environment. We decided to fill these values with the average conditions you would see at an indoor stadium – 0 wind and temperatures between 70-75 degrees. We chose to use 72 across the board for games played indoors.

We feature engineered the binary response variable called **home_team_win** from the **home_score** and **away_score** variables. The value of **home_team_win** is 1 when the **home_score** is greater than the **away_score** and 0 when the **away_score** is greater than **home_score**.

Exploratory Data Analysis

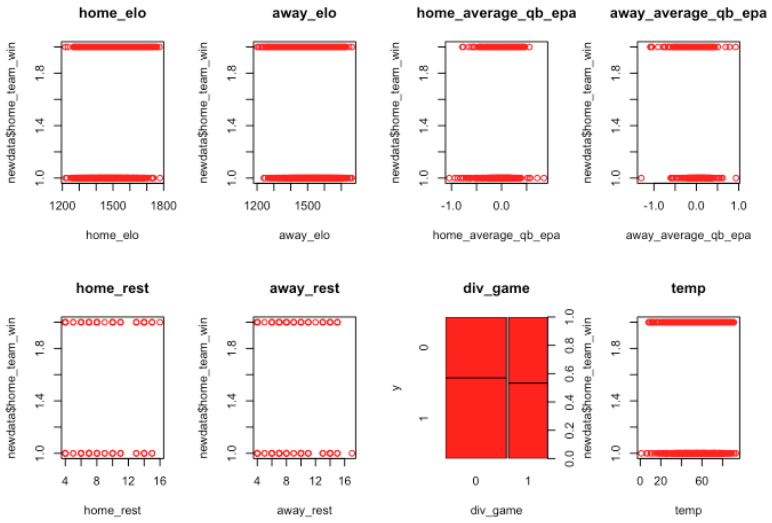
Using a correlation matrix, we analyzed the linear relationship between the response and independent variable. The figure below gives an overview of our correlation matrix.



Correlation Matrix (ggcor) of Response Variable with Independent Variable

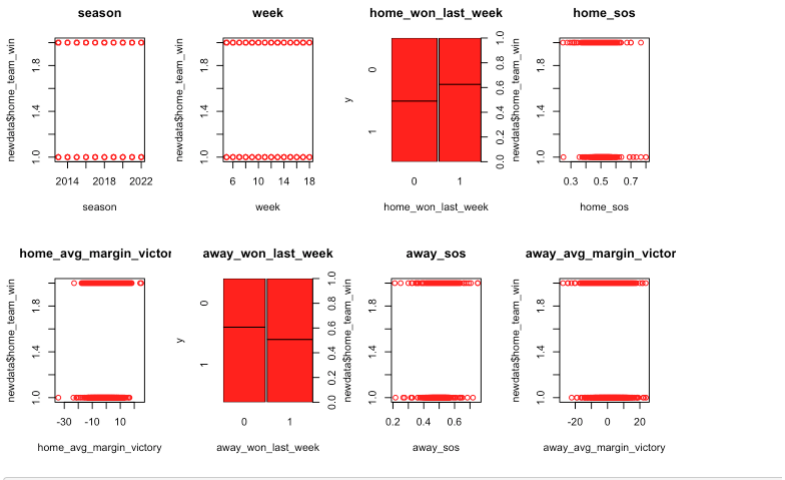
After analysis of the confusion matrix, we concluded that there is a high positive correlation of our response variable (**home_team_win**) with **home_won_last_week**, **home_avg_margin_victory**, **away_sos**, **home_elo**, **home_average_qb_epa**, **away_moneyline**, and **spread**. We also found that there is a high negative correlation of **home_team_win** with **home_sos, away_won_last_week**, **away_avg_margin_victory**, **away_elo**, **away_avarage_qb_epa**, and **home_moneyline.**

We also analyzed the dataset with the help of scatter plots. When the response variable (Y-axis) is a factor equal to two, that demonstrates that the home team won the game. When the factor level is equal to one, it demonstrates that the home team lost the game.

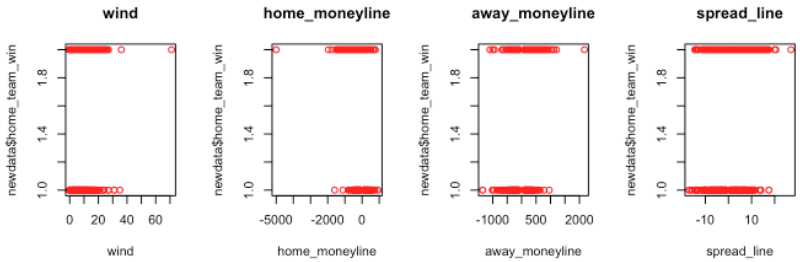


From the plots above, we see that away **average_qb_epa** has some predicting power with our response variable, the higher the **away_average_ab_epa** is, the less likely the home team is to win as noted by the cluster on the lower right portion of that graph. We also see that **home_elo** is more correlated with

the home team winning, which proves that the ELO rating system has some predictive power as to determine if a team will win a game. We also see the same positive correlation with **home_rest**, coupled with a negative correlation with **away_rest**, which shows us that more rest is correlated with a team's success in a given game.



From the plots above, we see patterns among **home_sos** and **home_avg_margin_victory** predicting the home team winning and **away_sos** and **away_avg_margin_victory** predicting the home team will lose.



From both the correlation and scatter plots above, we see again see similar trends between **home_moneyline** and **spread_line** predicting home team winning and **away_moneyline** predicting the home losing. We found that the home teams have a higher chance of winning if they won last week, suggesting that momentum and past performance are positive indicators for future success. The scatter plots of each variable are shown above in the diagram. We can also see similar trends in **home_avg_margin_victory**, **home_average_qb_epa** and **home_elo**. Teams that have higher values in these features are more likely to win the game.

Methodology

With the response variable being a binary variable, we selected Logistic Regression, Random Forest and Quadratic Discriminant Analysis (QDA) methods to model the dataset. We divided the dataset into 90% training data and 10% testing data and performed modeling with the training dataset.

Logistic Regression

Logistic Regression is a supervised machine learning algorithm often used for classification and predictive analytics. This method estimates the probability of an event occurring, based on the given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. Here, a logit transformation is applied on the odds-that is, the probability of success divided by the probability of failure. Based on the categorical response there are three different types of logistic regression. They are as follows:

- **Binary logistic regression:** In this approach, the response or dependent variable is binary in nature—that is, it has only two outcomes (e.g. 0 or 1). Within logistic regression, this is the most commonly and generally used approach and it is one of the most common classifiers for binary classification.
- **Multinomial logistic regression:** In this type of logistic regression model, the dependent variable has three or more outcomes; however, these values have no specified order.
- **Ordinal logistic regression:** This type of logistic regression model is leveraged when the response variable has three or more outcomes, but in this case, these values do have a defined order.

Random Forest

Random forest is a commonly used machine learning algorithm, which combines the output of multiple decision trees to reach a single result. It handles both classification and regression problems. It is an extension of the bagging method as it utilizes both bagging and feature randomness to create an uncorrelated forest of decision trees. The key benefits of random forest are mentioned below:

- **Reduced risk of overfitting:** With the robust number of decision trees in a random forest, the classifier will not overfit the model since the averaging of uncorrelated trees lowers the overall variance and prediction error.
- **Provides flexibility**: Since random forest can handle both regression and classification tasks with a high degree of accuracy, it is one of the most popular methods being used.
- **Easy to determine feature importance:** Random Forest makes it easy to evaluate variable importance, or contribution, to the model. There are a few ways to evaluate feature importance. Gini importance and mean decrease in impurity (MDI) are usually used to measure how much the model's accuracy decreases when a given variable is excluded. However, permutation importance, also known as mean decrease accuracy (MDA), is another important measure.

Quadratic Discriminant Analysis (QDA)

Quadratic Discriminant Analysis (QDA) is a generative model and is closely related to Linear Discriminant Analysis (LDA) model where it is assumed that the measurements from each class are normally distributed. Here there is no assumption that the covariance of each of the classes are identical. This is also one of the most used methods for obtaining a classifier.

Variable Selection

In addition to comparing different modeling methods, we also wanted to experiment with different combinations of variables in each of these models. To do so, we analyzed the variables in the correlation matrix and used our general knowledge of statistics and football to determine what we believed to be important to winning a football game. After that, we used AIC as our selection criteria to grab our set of training variables. We saw an interesting result using AIC: the initial results would grab one of the home/away variables, but not its reciprocal (Ex: AIC selected home team won last week but did not select away team won last week). Therefore, if one of the metrics for home (away) was included, we also included the away (home) metric as well for the 'User Selected' models. With the AIC selection, the EDA from the correlation matrix, and our general football knowledge, we mixed five different combinations of variables to build five different datasets.

Dataset A contains User Selected variables (as described above), Dataset B contains strictly the variables that were selected using AIC, Dataset C contains AIC selected variables and correlation selection, Dataset D contains a combination of AIC, correlation, and user selected variables, and lastly Dataset E contains again contains a different mix of user selected variables.

Dataset A	Dataset B	Dataset C	Dataset D	Dataset E
User Selection	AIC Selection	AIC & Correlation Selection	AIC, Correlation & User Selection	User Selection
away_averagage_qb_epa	away_moneyline	away_average_qb_epa	away_average_qb_epa	away_average_qb_epa
away_avg_margin_victory	div_game	away_avg_margin_victory	away_avg_margin_victory	away_avg_margin_victory
away_elo	home_average_qb_epa	away_moneyline	away_moneline	away_moneyline
away_rest	home_avg_margin_victory	away_rest	away_sos	away_won_last_week
away_sos	home_rest	away_sos	away_won_last_week	div_game
away_won_last_week	home_sos	away_won_last_week	home_average_qb_epa	home_average_qb_epa
div_game	home_won_last_week	div_game	home_avg_margin_victory	home_avg_margin_victory
home_average_qb_epa		home_average_qb_epa	home_won_last_week	home_rest
home_avg_margin_victory		home_avg_margin_victory	home_sos	home_sos
home_elo		home_moneyline	home_moneyline	home_won_last_week
home_rest		home_rest	spread_line	spread_line
home_sos		home_sos		
home_won_last_week		home_won_last_week		
wind				

Model Results

Out of our models, we saw decent performance in a handful of our models when running without cross-validation, including our Logistic Regression Models with datasets B, D, and E as well as Random Forest with dataset B. All these models performed similar to each other in terms of training and testing errors with each being around 23-25% training error and 33-34% testing error.

Results of Model without Cross Validation:

Model Name	Training Error	Testing Error
Log Reg A	0.3010204	0.3677054
Log Reg B	0.25	0.3365439
Log Reg C	0.2602041	0.3456091
Log Reg D	0.244898	0.333711
Log Reg E	0.2346939	0.3416431
Random Forest A	0.0000000	0.3694051
Random Forest B	0.0000000	0.3388102
Random Forest C	0.0000000	0.3427762
Random Forest D	0.0000000	0.3456091
Random Forest E	0.0000000	0.3382436
QDA A	0.1734694	0.4124646
QDA B	0.255102	0.368272
QDA C	0.255102	0.3971671
QDA D	0.2857143	0.3977337
QDA E	0.2346939	0.3705382

While we saw similar results among a few models running each model just one time, we saw a clear standout in our Logistic Regression with dataset B when performing cross-validation. After running the model through 100 rounds of cross-validation, we saw a 35% training error and 32% testing error. This is at least a few points better than any of the other results we examined during cross-validation.

Model Results with Cross Validation:

Model Name	Training Error	Testing Error
Log Reg A	0.3163265	0.3660057
Log Reg B	0.3571429	0.3184136
Log Reg C	0.3162760	0.3501416
Log Reg D	0.2857143	0.3535411
Log Reg E	0.2959184	0.3575071

Random Forest A	0.0000000	0.3773371
Random Forest B	0.0000000	0.3614731
Random Forest C	0.0000000	0.3535411
Random Forest D	0.0000000	0.3597734
Random Forest E	0.0000000	0.3586402
QDA A	0.2551020	0.4294618
QDA B	0.2908163	0.3688385
QDA C	0.2295918	0.4215297
QDA D	0.244898	0.415864
QDA E	0.2346939	0.3983003

Based on the results of our models with cross-validation, we moved forward with Logistic Regression B being our optimal model. This model contained the following feature variables: **home_won_last_week, home_sos, home_average_margin_victory, home_average_qb_epa, home_rest, div_game**, and **away_moneyline.** We used this model to run through a few different betting strategies. In each of these strategies, we will look only at placing a $100 bet on each game.

The first scenario we evaluated was betting on every team that our model predicts to win, that is if our model says the probability of winning is greater than 0.5, we bet $100 on that team. Looking at the data we have and the output of our model, this strategy would have placed bets on 1,765 games and we would have lost $1,925. However, if we adjusted our threshold to place a bet on a team, we can believe that we could limit our losses or come out ahead. The second strategy that we evaluated was using a threshold of 0.8 to place the $100 bet on a team. Under this strategy, we would have placed bets on 166 games and had a positive return of $68. This is a much better result than the first scenario, but still a smaller winning amount. The last strategy that we evaluated was to bet on the underdog. In this scenario, we only bet on the team if the MoneyLine was not favored to win the game in Vegas, but our model predicted the team will win the game with a threshold of 0.5. Under this approach, we only bet on 88 games, however we had a positive return of $1,770. This is a much better amount to win and is likely a result of being able to pinpoint the teams that statistically are more likely to win the game, but in terms of Vegas MoneyLine are not the team that is favored to win the game. If you win your bets in this situation, the payout is going to be higher than it would be if you bet on the team that was favored in Vegas MoneyLine and that team wins.

Conclusion

Perhaps we used our model at the start of this past season to bet on games that fit our criteria – that is, when our model predicts a team to be favored but when Vegas has that same team as the underdog. So far in the 2023 NFL season only nine games meet those criteria. Like we did with our

testing set, if we were to bet $100 on each of these games given the Vegas MoneyLine odds, we would have five wins and four losses and a profit of $286.

The results are encouraging for the first version of the model. However, there are still other improvements that are worth considering for the model's next iteration. Firstly, the betting data that was scraped was only the closing line of the game. That is, the final spread and odds were retrieved only immediately prior to the game beginning. However, another interesting factor to add would be the opening lines and what the spread and odds are when the game first becomes available to the public. With that data, we can track how much a line moves between opening and closing. For example, say a team was favored initially by three points when the line originally came out several days before the game. However, over the course of the week, as more money started to be bet on the other team, Vegas adjusted the line to be only one point right before kickoff. With the line changing from three points to one point, this would give us another metric to use to identify what happens when the spread changes for or against a certain team throughout the week.

There are a few other potential variables that we could add to the next model. One is a variable that measures momentum. For example, the current model only looks at if a team won the previous week. However, some type of momentum variable that measures if the team is on perhaps a three-game or five-game winning streak would account for a longer stretch of the season and if a team has consistently been playing well over the past few weeks. Another factor to consider is how a team fares on the road. Some teams tend to have more success on the road relative to other teams, so adding a type of road strength metric to account for a team's away performance would likely account for better predictions.

With sports betting being allowed in all but 12 states, it has made it much easier for the average sports fan to simply pull out their phone and place their bets on an app or their local casino rather than taking a weekend visit to Las Vegas. This has changed the entire sports betting landscape, particularly in the National Football League, which already was the most popular sport given the magnitude of fantasy football. All the Vegas lines are originally set using many advanced algorithms and analytical techniques. Beating "the house" is not an easy proposition – it is not a mistake on why casinos are some of the most profitable industries in the world. However, people can still make profit using advanced analytics to build machine learning models. Bettors will not always win every bet, but if a model can determine the best games to bet on as well generate above a 50%-win rate, this will result in bigger profits over a longer period of time.

Lessons We Have Learned

Lastly, our group learned a great deal not only from this project, but throughout the course. Firstly, data does not always come clean or in our desired format to analyze. Therefore, we had to learn how to manipulate multiple data frames, feature engineer new variables from existing variables, and merge large data frames. Data cleaning/formatting and understanding the data was an eye-opening process to us. The important lesson what we learned from data/variable selection was data is always huge and will have all random details, but we must extract or select what we need from it. We also had to put some effort into gaining knowledge of the content of the data here about sports and betting.

In addition, the entire group learned how to program more efficiently, in that we can build many highly advanced machine learning models with only a few lines of code using for-loops and testing different parameters during cross-validation. We also learned to interpret the R-code results, graphs, and also to extract the needed information from the output. Through all these tiny footsteps we learned to do some research work from different data sources. Overall, for this project specifically, we learned that there are feasible betting models that can be built to defeat “The House” or at least earn positive returns for an individual bettor over time.

References

NFLreadr R Package. NFL Verse Data, (2023), GitHub repository, https://github.com/nflverse/nflverse-data

Wienerkur, Akiva, et al. “NFL Betting Guide: How to Bet on the Spread.” *Sidelines*, 29 Mar. 2023, www.sidelines.io/betting-guides/nfl/spread

Appendix

```
#load in data
setwd('Documents/GA_Tech/ISYE 7406 - Data Mining and Statistical Learning/Group Project')
data <- read.table(file = "nfl_data.csv", sep = ",", header=T);
set.seed(1356487)

#load packages
library(SmartEDA)
library(ggplot2)
library(GGally)
library(Hmisc)
library(randomForest)
library(gbm)
library(dplyr)
library(rpart)
library(MASS)
library(class)
library(e1071)
library(stats)
library(nnet)
library(zoo)
library(corrplot)

#EDA
dim(data)
summary(data)
str(data)
numdata<- data %>% select_if(is.numeric)
round(cor(numdata),2)

#add a home team won indicator column
data <- data %>% mutate(home_team_win = ifelse(home_score>away_score,1,0))
newdata <-
data[,c('home_team_win','season','week','home_won_last_week','home_sos','home_avg_margin_victor
y','away_won_last_week','away_sos','away_avg_margin_victory','home_elo','away_elo','home_average
_qb_epa','away_average_qb_epa','home_rest','away_rest','div_game','temp','wind','home_moneyline','
away_moneyline','spread_line')]
model1 <- multinom(home_team_win~.,data=newdata)
summary(model1)
step(model1)

ggcorr(newdata)
newdata2<-newdata[,c(2:3,5:6,8:15,17:21)]
c=round(cor(newdata2),2)
corrplot(c,method ="circle")

par(mfrow = c(2,4))
```

```
for( i in 2:9){
  plot(newdata[,i],newdata$home_team_win,
       main = colnames(newdata)[i],xlab = colnames(newdata)[i], col = 'red')
}
par(mfrow = c(2,4))
for( i in 10:17){
  plot(newdata[,i],newdata$home_team_win,
       main = colnames(newdata)[i],xlab = colnames(newdata)[i], col = 'red')
}
par(mfrow = c(2,4))
for( i in 18:23){
  plot(newdata[,i],newdata$home_team_win,
       main = colnames(newdata)[i],xlab = colnames(newdata)[i], col = 'red')
}

#recategorize a tie to a loss
data$home_won_last_week <- ifelse(data$home_won_last_week == 1, 1,0)
data$away_won_last_week <- ifelse(data$away_won_last_week == 1, 1,0)

#limit to numeric value columns
limnumdata<-data %>% select_if(is.numeric)
limnumdata<-limnumdata[,5:23]
ggpairs(limnumdata, columns = 1:19, ggplot2::aes(colour=as.factor(home_team_win)),
        upper = list(continuous = wrap("cor", size = 2)))

#convert categorical columns to factors
data$home_won_last_week<-factor(data$home_won_last_week)
data$away_won_last_week<-factor(data$away_won_last_week)
data$div_game<-factor(data$div_game)
data$home_team_win<-factor(data$home_team_win)

#fill NAs with column median
fill_with_median<-function(x){
  median_value<-median(x,na.rm=TRUE)
  x[is.na(x)]<-median_value
}
#the missing temp and wind are from indoor stadiums...indoor stadiums are usually 70-75 degrees
data$temp[is.na(data$temp)]<-72
data$wind[is.na(data$wind)]<-0
data$home_average_qb_epa[is.na(data$home_average_qb_epa)]<-
median(data$home_average_qb_epa,na.rm=TRUE)
data$away_average_qb_epa[is.na(data$away_average_qb_epa)]<-
median(data$away_average_qb_epa,na.rm=TRUE)
data$away_elo[is.na(data$away_elo)]<-median(data$away_elo,na.rm=TRUE)
data$home_elo[is.na(data$home_elo)]<-median(data$home_elo,na.rm=TRUE)

limnumdata$temp[is.na(limnumdata$temp)]<-72
limnumdata$wind[is.na(limnumdata$wind)]<-0
```

```
limnumdata$home_average_qb_epa[is.na(limnumdata$home_average_qb_epa)]<-
median(limnumdata$home_average_qb_epa,na.rm=TRUE)
limnumdata$away_average_qb_epa[is.na(limnumdata$away_average_qb_epa)]<-
median(limnumdata$away_average_qb_epa,na.rm=TRUE)
limnumdata$away_elo[is.na(limnumdata$away_elo)]<-median(limnumdata$away_elo,na.rm=TRUE)
limnumdata$home_elo[is.na(limnumdata$home_elo)]<-median(limnumdata$home_elo,na.rm=TRUE)


#split training vs testing
modeldataA<-data[,c(9:23,27)]
modeldataB<-data[,c(9:11,17,19,21,25,27)]
modeldataC<-data[,c(9:14,17:21,24,25,27)]
modeldataD<-data[,c(9,12,11,14,10,13,17,18,24,25,26,27)]
modeldataE<-data[,c(9,12,10,11,14,17,18,19,21,25,26,27)]
n=dim(modeldataA)[1]
n1=round(n*0.9)

####Full CV Loop with all models
B=100
logTRA=NULL
logTRB=NULL
logTRC=NULL
logTRD=NULL
logTRE=NULL
rfTRA=NULL
rfTRB=NULL
rfTRC=NULL
rfTRD=NULL
rfTRE=NULL
qdaTRA=NULL
qdaTRB=NULL
qdaTRC=NULL
qdaTRD=NULL
qdaTRE=NULL
logTEA=NULL
logTEB=NULL
logTEC=NULL
logTED=NULL
logTEE=NULL
rfTEA=NULL
rfTEB=NULL
rfTEC=NULL
rfTED=NULL
rfTEE=NULL
qdaTEA=NULL
qdaTEB=NULL
qdaTEC=NULL
qdaTED=NULL
```

```
qdaTEE=NULL
for (b in 1:B){
 set.seed(1352647)
 ##randomly create training and testing data sets
 flag<-sort(sample(1:n,n1))
 CVTrainA<-modeldataA[-flag,]
 CVTestA<-modeldataA[flag,]
 CVTrainB<-modeldataB[-flag,]
 CVTestB<-modeldataB[flag,]
 CVTrainC<-modeldataC[-flag,]
 CVTestC<-modeldataC[flag,]
 CVTrainD<-modeldataD[-flag,]
 CVTestD<-modeldataD[flag,]
 CVTrainE<-modeldataE[-flag,]
 CVTestE<-modeldataE[flag,]

 #Logistic Regression A
 logModelA<-glm(home_team_win~.,data=CVTrainA,family="binomial")
 logModelATrain<-round(predict(logModelA,CVTrainA[,1:15],type="response"))
 logTRA<-c(logTRA,mean(logModelATrain!=CVTrainA$home_team_win,na.rm=TRUE))
 logModelATest<-round(predict(logModelA,CVTestA[,1:15],type="response"))
 logTEA<-c(logTEA,mean(logModelATest!=CVTestA$home_team_win,na.rm=TRUE))

 #Logistic Regression B
 logModelB<-glm(home_team_win~.,data=CVTrainB,family="binomial")
 logModelBTrain<-round(predict(logModelB,CVTrainB[,1:7],type="response"))
 logTRB<-c(logTRB,mean(logModelBTrain!=CVTrainB$home_team_win,na.rm=TRUE))
 logModelBTest<-round(predict(logModelB,CVTestB[,1:7],type="response"))
 logTEB<-c(logTEB,mean(logModelBTest!=CVTestB$home_team_win,na.rm=TRUE))

 #Logistic Regression C
 logModelC<-glm(home_team_win~.,data=CVTrainC,family="binomial")
 logModelCTrain<-round(predict(logModelC,CVTrainC[,1:14],type="response"))
 logTRC<-c(logTRA,mean(logModelCTrain!=CVTrainC$home_team_win,na.rm=TRUE))
 logModelCTest<-round(predict(logModelC,CVTestC[,1:14],type="response"))
 logTEC<-c(logTEC,mean(logModelCTest!=CVTestC$home_team_win,na.rm=TRUE))

 #Logistic Regression D
 logModelD<-glm(home_team_win~.,data=CVTrainD,family="binomial")
 logModelDTrain<-round(predict(logModelD,CVTrainD[,1:12],type="response"))
 logTRD<-c(logTRD,mean(logModelDTrain!=CVTrainD$home_team_win,na.rm=TRUE))
 logModelDTest<-round(predict(logModelD,CVTestD[,1:12],type="response"))
 logTED<-c(logTED,mean(logModelDTest!=CVTestD$home_team_win,na.rm=TRUE))

 #Logistic Regression E
 logModelE<-glm(home_team_win~.,data=CVTrainE,family="binomial")
 logModelETrain<-round(predict(logModelE,CVTrainE[,1:12],type="response"))
 logTRE<-c(logTRE,mean(logModelETrain!=CVTrainE$home_team_win,na.rm=TRUE))
```

```
logModelETest<-round(predict(logModelE,CVTestE[,1:12],type="response"))
logTEE<-c(logTEE,mean(logModelETest!=CVTestE$home_team_win,na.rm=TRUE))

#Random Forest A
rfModelA<-randomForest(home_team_win~.,CVTrainA,importance=TRUE)
##training error
rfTRA<-c(rfTRA,mean(predict(rfModelA,CVTrainA,type = "response")!=CVTrainA$home_team_win))
predictionA<-predict(rfModelA,CVTestA,type="response")
#testing error
rfTEA<-c(rfTEA,mean(CVTestA$home_team_win!=predictionA))

#Random Forest B
rfModelB<-randomForest(home_team_win~.,CVTrainB,importance=TRUE)
##training error
rfTRB<-c(rfTRB,mean(predict(rfModelB,CVTrainB,type = "response")!=CVTrainB$home_team_win))
predictionB<-predict(rfModelB,CVTestB,type="response")
#testing error
rfTEB<-c(rfTEB,mean(CVTestB$home_team_win!=predictionB))

#Random Forest C
rfModelC<-randomForest(home_team_win~.,CVTrainC,importance=TRUE)
##training error
rfTRC<-c(rfTRC,mean(predict(rfModelC,CVTrainC,type = "response")!=CVTrainC$home_team_win))
predictionC<-predict(rfModelC,CVTestC,type="response")
#testing error
rfTEC<-c(rfTEC,mean(CVTestC$home_team_win!=predictionC))

#Random Forest D
rfModelD<-randomForest(home_team_win~.,CVTrainD,importance=TRUE)
##training error
rfTRD<-c(rfTRD,mean(predict(rfModelD,CVTrainD,type = "response")!=CVTrainD$home_team_win))
predictionD<-predict(rfModelD,CVTestD,type="response")
#testing error
rfTED<-c(rfTED,mean(CVTestD$home_team_win!=predictionD))

#Random Forest E
rfModelE<-randomForest(home_team_win~.,CVTrainE,importance=TRUE)
##training error
rfTRE<-c(rfTRE,mean(predict(rfModelE,CVTrainE,type = "response")!=CVTrainE$home_team_win))
predictionE<-predict(rfModelE,CVTestE,type="response")
#testing error
rfTEE<-c(rfTEE,mean(CVTestE$home_team_win!=predictionE))

#QDA Prep
CVTrainA$div_game<-as.numeric(CVTrainA$div_game)
CVTestA$div_game<-as.numeric(CVTestA$div_game)
CVTrainA$home_won_last_week<-as.numeric(CVTrainA$home_won_last_week)
CVTestA$home_won_last_week<-as.numeric(CVTestA$home_won_last_week)
```

```
CVTrainA$away_won_last_week<-as.numeric(CVTrainA$away_won_last_week)
CVTestA$away_won_last_week<-as.numeric(CVTestA$away_won_last_week)
CVTrainB$div_game<-as.numeric(CVTrainB$div_game)
CVTestB$div_game<-as.numeric(CVTestB$div_game)
CVTrainB$home_won_last_week<-as.numeric(CVTrainB$home_won_last_week)
CVTestB$home_won_last_week<-as.numeric(CVTestB$home_won_last_week)
CVTrainC$div_game<-as.numeric(CVTrainC$div_game)
CVTestC$div_game<-as.numeric(CVTestC$div_game)
CVTrainC$home_won_last_week<-as.numeric(CVTrainC$home_won_last_week)
CVTestC$home_won_last_week<-as.numeric(CVTestC$home_won_last_week)
CVTrainC$away_won_last_week<-as.numeric(CVTrainC$away_won_last_week)
CVTestC$away_won_last_week<-as.numeric(CVTestC$away_won_last_week)
CVTrainD$home_won_last_week<-as.numeric(CVTrainD$home_won_last_week)
CVTestD$home_won_last_week<-as.numeric(CVTestD$home_won_last_week)
CVTrainD$away_won_last_week<-as.numeric(CVTrainD$away_won_last_week)
CVTestD$away_won_last_week<-as.numeric(CVTestD$away_won_last_week)
CVTrainE$home_won_last_week<-as.numeric(CVTrainE$home_won_last_week)
CVTestE$home_won_last_week<-as.numeric(CVTestE$home_won_last_week)
CVTrainE$away_won_last_week<-as.numeric(CVTrainE$away_won_last_week)
CVTestE$away_won_last_week<-as.numeric(CVTestE$away_won_last_week)
CVTrainE$div_game<-as.numeric(CVTrainE$div_game)
CVTestE$div_game<-as.numeric(CVTestE$div_game)



#QDA A
qdaModelA<-qda(CVTrainA[,1:15],CVTrainA[,16])
qdaTRPredA<-predict(qdaModelA,CVTrainA[,1:15])$class
qdaTRA<-mean(qdaTRPredA!=CVTrainA$home_team_win)
qdaTREPredA<-predict(qdaModelA,CVTestA[,1:15])$class
qdaTEA<-mean(qdaTREPredA!=CVTestA$home_team_win)

#QDA B
qdaModelB<-qda(CVTrainB[,1:7],CVTrainB[,8])
qdaTRPredB<-predict(qdaModelB,CVTrainB[,1:7])$class
qdaTRB<-mean(qdaTRPredB!=CVTrainB$home_team_win)
qdaTREPredB<-predict(qdaModelB,CVTestB[,1:7])$class
qdaTEB<-mean(qdaTREPredB!=CVTestB$home_team_win)

#QDA C
qdaModelC<-qda(CVTrainC[,1:13],CVTrainC[,14])
qdaTRPredC<-predict(qdaModelC,CVTrainC[,1:13])$class
qdaTRC<-mean(qdaTRPredC!=CVTrainC$home_team_win)
qdaTREPredC<-predict(qdaModelC,CVTestC[,1:13])$class
qdaTEC<-mean(qdaTREPredC!=CVTestC$home_team_win)

#QDA D
qdaModelD<-qda(CVTrainD[,1:11],CVTrainD[,12])
```

```
  qdaTRPredD<-predict(qdaModelD,CVTrainD[,1:11])$class
  qdaTRD<-mean(qdaTRPredD!=CVTrainD$home_team_win)
  qdaTREPredD<-predict(qdaModelD,CVTestD[,1:11])$class
  qdaTED<-mean(qdaTREPredD!=CVTestD$home_team_win)

  #QDA E
  qdaModelE<-qda(CVTrainE[,1:11],CVTrainE[,12])
  qdaTRPredE<-predict(qdaModelE,CVTrainE[,1:11])$class
  qdaTRE<-mean(qdaTRPredE!=CVTrainE$home_team_win)
  qdaTREPredE<-predict(qdaModelE,CVTestE[,1:11])$class
  qdaTEE<-mean(qdaTREPredE!=CVTestE$home_team_win)
}

## Create a matrix with the mean values
error_matrix <- matrix(c(mean(logTRA), mean(logTEA), mean(logTRB),
mean(logTEB),mean(logTRC),mean(logTEC),mean(logTRD),mean(logTED),mean(logTRE),mean(logTEE),m
ean(rfTRA),mean(rfTEA),mean(rfTRB),mean(rfTEB),mean(rfTRC),mean(rfTEC),mean(rfTRD),mean(rfTED),
mean(rfTRE),mean(rfTEE),mean(qdaTRA),mean(qdaTEA),mean(qdaTRB),mean(qdaTEB),mean(qdaTRC),
mean(qdaTEC),mean(qdaTRD),mean(qdaTED),mean(qdaTRE),mean(qdaTEE)), nrow = 2)

# Add row and column names
rownames(error_matrix) <- c("Training Error", "Testing Error")
colnames(error_matrix) <- c("logA", "logB","logC","logD","logE","Random Forest A","Random Forest
B","Random Forest C","Random Forest D","Random Forest E","QDA A","QDA B","QDA C","QDA D","QDA
E")

# Print the resulting 2x2 table
print(error_matrix)

#Logistic Regression B
logModelB<-glm(home_team_win~.,data=CVTrainB,family="binomial")
logModelBTest <- predict(logModelB,CVTestB[,1:7],type="response")
test_data <- CVTestB[,1:7]
response <- CVTestB[8]

#Grab Home Moneyline
moneyline<-data[,c(9:11,17,19,21,24, 25,27)]
moneyline<-moneyline[flag,]
moneyline <- moneyline[7]

final_df_1 <- cbind(test_data,moneyline)

#Calculate Win Probability for Home Team
final_df_1 <- final_df_1 %>%
  mutate(home_team_win_prob = ifelse(home_moneyline <= 0, (-(home_moneyline)/(-
(home_moneyline)+100))
                                     , (100/(home_moneyline+100))),
         away_team_win_prob = 1 - home_team_win_prob)
```

```
#Combine All Data into one Data Frame
final_df_1 <- cbind(final_df_1, response,logModelBTest)

#Change column Name
colnames(final_df_1)[12] <- "home_team_winprob_pred"

#Calculate our Model's Prediction and our winnings
final_df_1 <- final_df_1 %>%
  mutate(home_team_win_pred = ifelse(home_team_winprob_pred >= 0.5, 1, 0),
       winnings1 = case_when(home_team_win == 1 & home_team_win_pred == home_team_win &
home_moneyline >= 0 ~ (home_moneyline/100)*100,
                       home_team_win == 1 & home_team_win_pred == home_team_win &
home_moneyline <= 0 ~ 100 / (-1*home_moneyline / 100),
                       home_team_win == 0 & home_team_win_pred == home_team_win &
home_moneyline >= 0 ~ 100/(-1*away_moneyline / 100),
                       home_team_win == 0 & home_team_win_pred == home_team_win &
home_moneyline <= 0 ~ (away_moneyline/100)*100,
                       TRUE~ -100
       )
)

sum(final_df_1$winnings1)
```