An NFL Quarterback Analysis

Christopher Goodwin

Bellevue University – DSC680

Abstract

For fans of American football, it is well known that the quarterback (QB for short) is the most important position in the game. A football QB, like no other role at any team game, normally reflects his execution on the final score. It's atypical that a bad day for a QB could translate in a good day for the rest of the team. Like no other position, passer's decisions guide the offensive side of the game, the one that leads to score points. Excluding a few anomalies, the QB is the key for wins and championships. (Pensado, 2018)

Our paper will look at game logs for NFL quarterbacks and determine if we can fit a model that can accurately predict whether or not his team won. This analysis will allow us to gain access into how the game of football is currently being played, and may even help coaches gain focus in particular aspects of the passing game.

An NFL Quarterback Analysis

In 2019, the Harvard Business Review posted an article declaring the QB the most important position in all of football, including the coach and the general manager. According to a study of 38 years of win-loss records, these four leader variables—quarterback, coach, general manager, and owner—explained 68.2%, or more than two-thirds, of the variance in team performance. Owners carried the least weight (roughly 11.12% of explained variance), followed by general managers (22.43%), then coaches (29.08%), and finally, quarterbacks (37.37%). (Groysberg, Hecht, and Naik, 2019).

The goal of our paper is to take that even a step further. Within the QB position, what variables factor most heavily into a win. By looking at game logs throughout the course of league history, we can determine which statistics: sacks, interceptions, passing yards, touchdowns lead to the most victories.

The Data

We need to start by examining the data we will be using in our analysis. The *NFL Statistics* was developed by Kaggle user Kendall Gillies. The data was scraped from NFL.com using Python code. Each row of the Quarterback Game Log database represents and individual game for an individual quarterback.

We had to make a few minor updates to the dataset. The first step was removing all entries where Season = ‘Preseason’. The preseason is played much differently than regular season, where an individual player may only play a few plays. Including this data could sway our results. The second step was removing any entries where Passes Attempted = ‘--’. When evaluating QBs, we obviously only want to include players who threw a pass. Whether the data was missing from the scrape, or if the player truly threw no passes, we want to exclude those

game logs. The final purge comes from those game logs where Outcome = ‘T’. This indicates that there was a tie. There are two main reasons for us to eliminate these entries: 1) we are really only trying to predict wins and losses and 2) ties are so rare in the NFL, that it would be unfair to our model if it had to predict them.

Now that we have our final subset, there are a few more steps we need to take. For any null values, the string ‘--’ was uploaded to indicate the lack of data. I made the decision to convert these to zeroes throughout the dataset. There were too many values to exclude them completely from our analysis (i.e., dropping any NA values), so I felt like making them 0 was a good way to go. In addition to that, in order to clean some of the data points up I converted any numeric data to the appropriate format. Some of the integer and float variables were being portrayed as strings or objects, so I made the necessary adjustments. In addition, some of the variables of interest needed be converted to categorical in order to perform our analysis.

The final step was creating some additional variables of our own. The first new variable was called “Team Points”. This column represents the amount of points an individual’s team scored during the game. This value was parsed from the Score variable, which included the combined score for the game. The other variables were encoded categories. I encoded “Home or Away”, where Home was represented by 1 and Away by 0. I also encoded the opponents, which were numbered 0-31 to represent the number of teams in the NFL. The final encoding was for the outcome of the game, with W represented by 2 and L by 0. We are left 17,179 game logs with the following variables: 'Player Id', 'Name', 'Position', 'Year', 'Season', 'Week', 'Game Date', 'Home or Away', 'Opponent', 'Outcome', 'Score', 'Games Played', 'Games Started', 'Passes Completed', 'Passes Attempted', 'Completion Percentage', 'Passing Yards', 'Passing Yards Per Attempt', 'TD Passes', 'Ints', 'Sacks', 'Sacked Yards Lost', 'Passer Rating', 'Rushing Attempts',

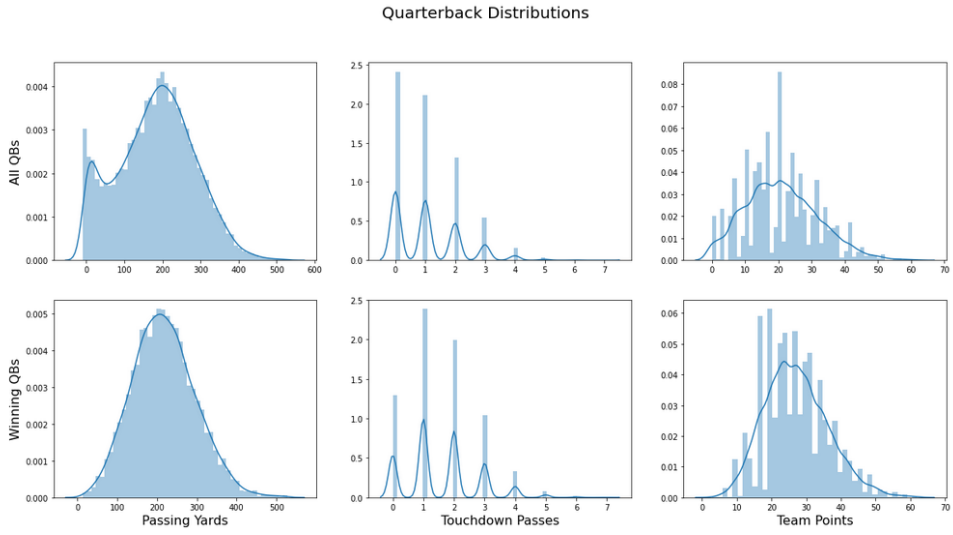
'Rushing Yards', 'Yards Per Carry', 'Rushing TDs', 'Fumbles', 'Fumbles Lost', 'Team Points', 'outcome_code', 'hoa_code', 'opponent_code'.

Initial Analysis

Our first step was to really take a look at the data, and see if we could gain anything just from its general distributions. In order to do this, I compiled some distribution plots for a few variables of interest: Passing Yards, Touchdown Passes, and Team Points. I chose these three variables because based on my general knowledge of the NFL, I thought they may be particularly important.

Figure 1

Distributions of variables of interest



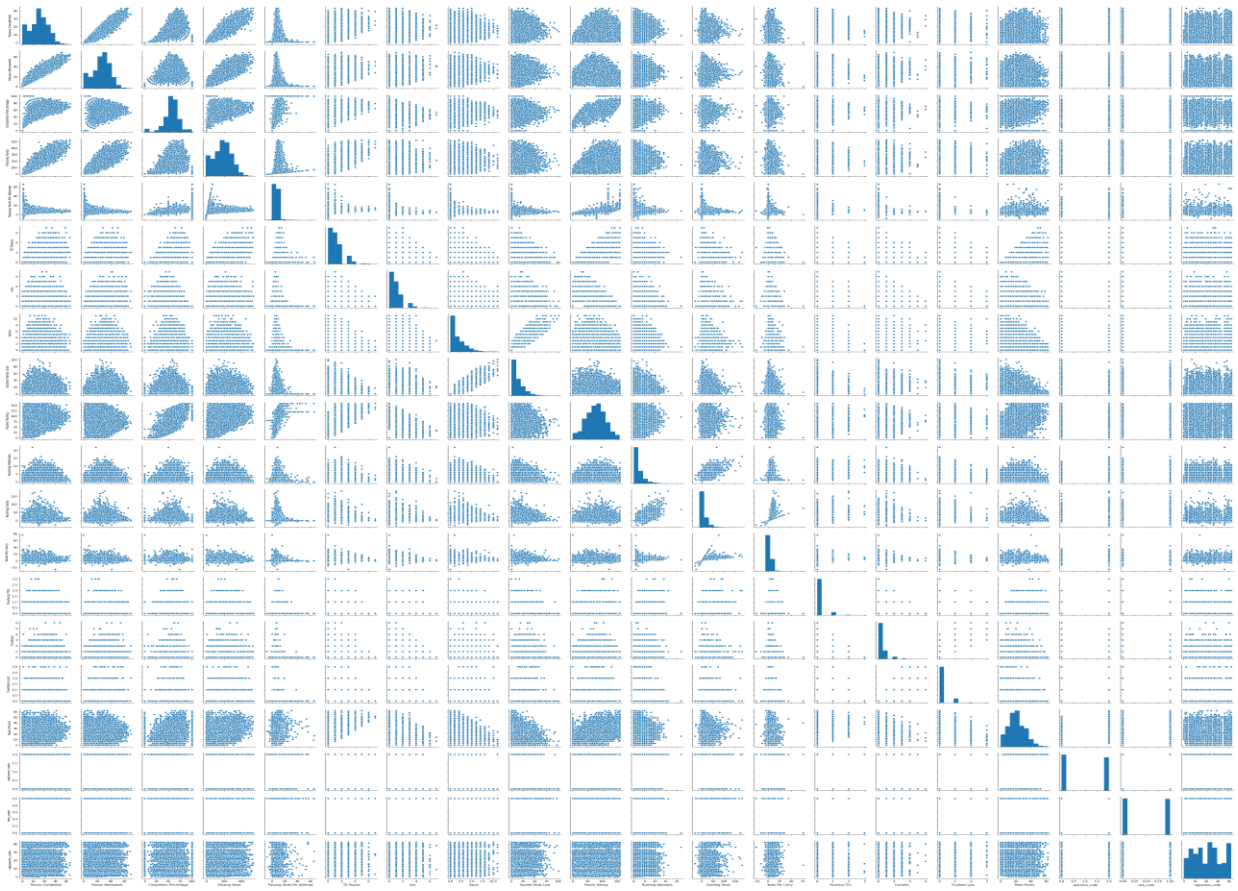
In row 1, we see distributions for the three variables for all QBs. After our initial data cleaning that was described above, these distributions represent the entire dataset. We then have a second row labeled “Winning QBs”. These distributions represent only those quarterbacks who won the game. We see that for all three variables, there is a general shift to the right when we only include winning quarterbacks. This means that there are fewer low values, and more high

values (I.e., quarterbacks who throw for more yards, complete more touchdown passes and score more points tend to win more often).

Another good way to look at our data is to create a pair-wise plot. This creates scatter plots of every variable in our dataset. This could give us some real insight into our data, and how we need to go about building our model. In order to create this plot, we will only look at the numeric data that is included (including our encoded categorical variables that we created). Let's take a look at this plot for our dataset.

Figure 2

Pair-wise plot of our dataset

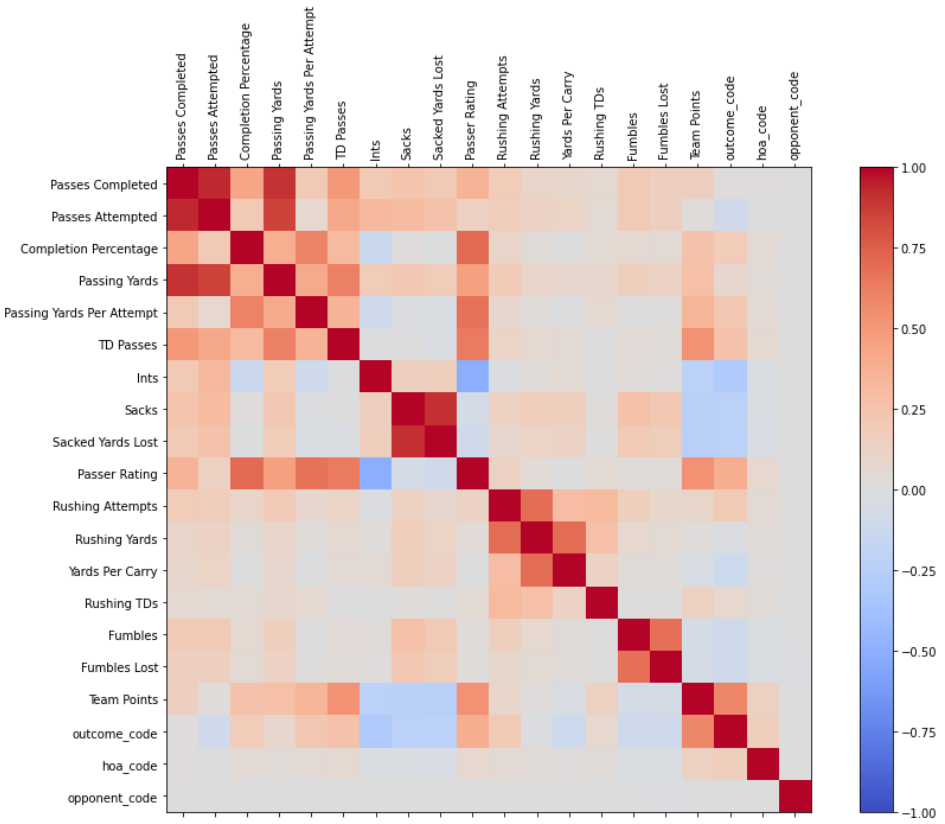


While this is a pretty awesome tool, what does it tell us? Unfortunately for our analysis, it does not tell us much. First off, due the number of variables we have in our dataset, it is difficult

to fully read every individual plot. From this exterior view, the vast majority of scatter plots appear to be very random. I would not say any of these variables appear to be very strongly correlated. And it is even harder to tell the correlation with our decision variable, Outcome, because it is categorical. Perhaps a more interesting look would be a correlation matrix:

Figure 3

A correlation matrix of our dataset



This is much easier to read. As we can see, most of the variables do not seem to have a strong individual impact on the outcome of the game. Team points seems to be the only variable with a noticeable correlation, which makes sense because more points typically lead to more wins. There are also a few variables that have strong correlations with one another: passes completed and passes attempted, sacks and sack yards, passes completed and passing yards. The

decision now is whether or not to include these variables with high correlations. For completeness of analysis, I am going to include them. I think despite the high correlation, each individual statistic plays a role in the ultimate model. We will discuss our models in the following sections.

Decision Tree Classifier.

Our next step is to create a prediction model. The first option I chose was the Decision Tree Classifier. A Decision Tree is a simple representation for classifying examples. It is a Supervised Machine Learning where the data is continuously split according to a certain parameter. Because our the variable we would like to determine is categorical/discrete (Win or Loss), we use a classification tree. Such a tree is built through a process known as binary recursive partitioning. This is an iterative process of splitting the data into partitions, and then splitting it up further on each of the branches. (Chakure, 2019)

Using the *scikit-learn* package in Python, I compiled a DecisionTreeClassifier model. Our X variables were composed of all of the numeric values as previously discussed. Our Y decision variable was of course outcome. I split the data into training and test data (80% train – 13743 records, 20% test – 3436 records) and fit the model to the training data. When predicting the outcome, our model has an accuracy score of 0.7505820721769499, meaning that roughly 75% of the time we can predict the winner of a football just using the QB's game log. That is an impressive start.

Within the model, we can look at the importance factor for each of the variables. The following table lists each feature, and its corresponding importance:

Table 1

Feature Importance for DecisionTreeClassifier

Feature Number	Feature Name	Importance
1	Passes Completed	0.02585
2	Passes Attempted	0.07878
3	Completion %	0.04456
4	Passing Yards	0.04417
5	Passing Yards/Attempt	0.03805
6	TD Passes	0.01258
7	Ints	0.02075
8	Sacks	0.01631
9	Sacked Yards Lost	0.04014
10	Passer Rating	0.06016
11	Rushing Attempts	0.03939
12	Rushing Yards	0.04821
13	Yards Per Carry	0.05542
14	Rushing TDs	0.00323
15	Fumbles	0.01195
16	Fumbles Lost	0.00477
17	Team Points	0.39447
18	Home or Away Code	0.01155
19	Opponent Code	0.04966

What insight can we gain from this table? Right off the bat, we notice how important the Team Points variable is that we created. Almost 40% of the importance is attributed to this one variable. On the other end of the spectrum, we see that some variables are not very important at

all. Surprisingly to me, the home or away variable did not have much of an impact on the model. I would think that being a home team would have a large impact on the number of wins, but that is not the case. Same goes for TD passes. I would have assumed that TD passes would have a great impact on the outcome of the game, but from our analysis here we see that really is not the case.

Because Team Points carried such a large burden of our model, I wanted to try to create another model excluding this variable. I followed the same process, but dropped Team Points from our X variables. This new model achieved an accuracy score of 0.6781140861466822 on our test data, meaning that it was able to successfully predict the winner of 68% of games. In this new model, the variable Passer Rating actually became the most important (importance factor of 0.23335).

Logistic Regression Model

Another classification model that may be useful to us is the logistic regression model. Logistic regression is a machine learning algorithm for classification. In this algorithm, the probabilities describing the possible outcomes of a single trial are modelled using a logistic function. Logistic regression is designed for classification, and is most useful for understanding the influence of several independent variables on a single outcome variable. These reasons encapsulate why I felt like it was the perfect choice for us. (Garg, 2018)

Once again, *scikit-learn* does most of the hard work for us, with model option LogisticRegression. I followed similar steps as before, breaking my data into 80/20 chunks for training and testing. I included all variables of interest, including Team Points, as I was unsure how it would react to this new model type.

After the model was fit to our training data, I used the predict function on our test data. This model achieved an accuracy score of 0.8090803259604191, meaning that this new model correctly predicted the outcome of almost 81% of the games. Specifically, if we look at the confusion matrix for this data, we can see the following:

Table 2

Confusion Matrix for LogisticRegression

N = 3436	Predicted: Win	Predicted: Loss
Actual: Win	1467	323
Actual: Loss	333	1313

We can see that there were 2,780 correct predictions, and only 656 incorrect. This leads to a precision value of 0.81, a recall value of 0.81, and an f1-score value of 0.81 on the entire dataset. These are excellent numbers for our analysis.

Random Forest Classifier

The final model type I will take an in-depth look at is a random forest classifier. This is somewhat of a piggyback off of the Decision Tree algorithm. Random forest classifier is a meta-estimator that fits a number of decision trees on various sub-samples of datasets and uses average to improve the predictive accuracy of the model and controls over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement. The benefits of these models are a reduction in over-fitting and an increase in accuracy over its decision tree counterparts in most cases. (Garg, 2018).

We will use scikit-learn once again, this time using the RandomForestClassifier. We will fit it using our same training data and predict on our same test data. This particular model had an accuracy score of 0.8023864959254947, meaning that it was able to accurate predict the outcome of just over 80% of the games in our dataset. Similar to the decision trees classifier, I wanted to take a look at the importance of each of the variables. The following table has each feature ranked by importance

Table 3

Feature Importance for DecisionTreeClassifier

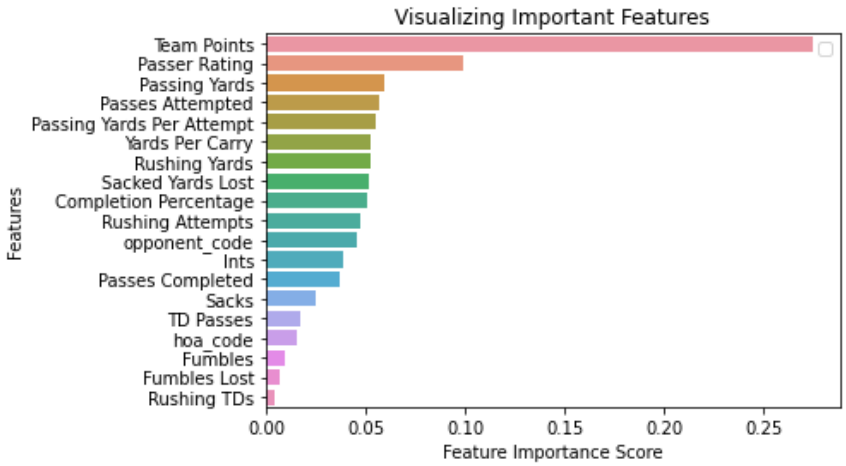
Feature Number	Feature Name	Importance
1	Team Points	0.274760
2	Passer Rating	0.099016
3	Passing Yards	0.059659
4	Passes Attempted	0.056561
5	Passing Yards/Attempt	0.055369
6	Yards Per Carry	0.052724
7	Rushing Yards	0.052182
8	Sacked Yards Lost	0.051952
9	Completion Percentage	0.050599
10	Rushing Attempts	0.047361
11	Opponent Code	0.045367
12	Ints	0.038766
13	Passes Completed	0.036817
14	Sacks	0.025220

15	TD Passes	0.017387
16	Home or Away Code	0.015415
17	Fumbles	0.009665
18	Fumbles Lost	0.006900
19	Rushing TDs	0.004280

Not surprisingly, Team Points once again dominated. It is even more noticeable when we display this as a bar chart:

Figure 4

Bar Chart of Feature Importance for Random Trees Classifier



The graph really puts into perspective just how dominant the Team Points variable is. It also goes to show that Passer Rating is also very important. This was somewhat predictable, given the Decision Tree Classifier we created excluding the Team Points variable. But still, Passer Rating is almost doubly more important than any other variable in the dataset.

The importance of Passer Rating leads me to believe this is why ESPN dedicated time and resources to creating the QBR statistic. ESPN's Total Quarterback Rating (Total QBR), which was released in 2011, incorporates all of a quarterback's contributions to winning,

including how he impacts the game on passes, rushes, turnovers and penalties. Also, since QBR is built from the play level, it accounts for a team's level of success or failure on every play to provide the proper context and then allocates credit to the quarterback and his teammate to produce a clearer measure of quarterback efficiency. (Katz and Burke, 2016)

Because QBR was specifically designed to judge for success, I feel like it would have been a welcomed addition to our dataset. My gut tells me QBR would have been a heavy player in these classification models.

Remaining Models

There are a number of other classifier models that we could use for our analysis here. None of them provided great insight to our specific project, but I have included the accuracy results for each of them in this section, just to show that we did our due diligence in finding the best model for this report.

Naïve-Bayes: This algorithm is based on Bayes' theorem with the assumption of independence between every pair of features. Naive Bayes classifiers work well in many real-world situations such as document classification and spam filtering. This algorithm requires a small amount of training data to estimate the necessary parameters. Naive Bayes classifiers are extremely fast compared to more sophisticated methods. (Garg, 2018) On our test data, this model achieved an accuracy score of 0.7505820721769499, meaning it correctly predicting the outcome of 75% of the games. This is similar to our Decision Tree algorithm, but doesn't match the accuracy of the Logistic Regression or Random Forest Classifier.

K-Nearest Neighbors: Neighbors based classification is a type of lazy learning as it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the k nearest neighbors of each point.

This algorithm is simple to implement, robust to noisy training data, and effective if training data is large. (Garg, 2018) On our test data, this model achieved an accuracy score of 0.7587310826542492, predicting almost 76% of the outcomes. This number seems to be right around the sweet spot for a lot of these models.

Stochastic Gradient Descent: Stochastic gradient descent is a simple and very efficient approach to fit linear models. It is particularly useful when the number of samples is very large. It supports different loss functions and penalties for classification. (Garg, 2018). On our test data, this model only achieved an accuracy score of 0.6731664726426076, making it the least effective model that we tested.

Support Vector Machine: Support vector machine is a representation of the training data as points in space separated into categories by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. This algorithm is effective in high dimensional spaces and uses a subset of training points in the decision function so it is also memory efficient. Unfortunately for us, this model was only able to attain a 0.539289871944121 accuracy score, barely outdoing the flip of a coin.

Next Steps

I think there are definitely some things we can do to improve on our models. Step 1 would be to find or create a more complete data set. By eliminating the entries where Passes Attempted = '--', we lost almost half of our dataset. I can't help but think this hurt the models in the long run. If we had more instances to train the model on, I believe it would improve on its score. In this very basic example, we were able to achieve accuracies of over 80%, and I think the sky is the limit if we have more data.

This also includes additions of some new variables. Total QBR, as we discussed previously, could greatly help the accuracy of our models. We could also include some more modern-day advanced stats. Adjusted net yards per attempt (or ANY/A) uses more modern research by Chase Stuart to estimate the value of touchdowns and interceptions while also incorporating sacks, which evidence suggests has plenty to do with quarterbacks despite being commonly blamed on the offensive line. The indispensable Pro-football-reference.com (PFR) adjusts for era in several key metrics with their index statistics, such as Sack Rate+ (Sack Rate Index) or ANY/A+. PFR measures the number of standard deviations above or below the mean that a player accounts for in a particular category, and multiplies it by 15 to create the index stat. It's not a perfect methodology, but this does an excellent job of putting things in context in terms of key quarterback rate stats. (Barnwell, 2017). If we could find a way to incorporate statistics like this, we really could get somewhere.

I believe a second step would be to include the contributions from other players besides the quarterback. Given more time the rest of the semester, I would love to come back to this project with a method to merge together some datasets with other positions. It would be interesting to me to see if we could combine QB and running back statistics for instance. This would require us to be able to parse individual games from the database tables in order to join instances of the same game, and I unfortunately just could not think of a good way to accomplish that.

Conclusion

Our ultimate goal was two-fold: 1) can we predict who will win a game based on a QB's statistics and 2) what are the leading factors in these predictions. For question 1, I can comfortably say that the answer is yes. With models reaching 80%, I would feel confident

predicting the outcome of a game. For question 2, there were definitely some stats that stood out among the rest. Team points, as well as passer rating, were head and shoulders the most important features in our models. Passing Yards and Passing Yards / Attempt were also very useful.

So what can the industry do with that information? For coaches, it may sound overly simple, but your ultimate goal should be to score more points. Legendary football coach Paul "Bear" Bryant famously said, "Offense sells tickets. Defense wins championships." (Otten, 2018). That appears to be a very outdated sentiment now, as we showed here that the main key to success is scoring points. If you look at the Buffalo Bills for instance, in 2019 they scored 19.6 points per game, which ranked them 23rd of 32 NFL teams. They were able to win 10 games, but lost their first round playoff matchup. Fast forward to 2020, the Bills spent a lot of time and assets ramping up their offense. They are now averaging 27.6 points per game, which ranks them 9th of 32 NFL teams. They currently have 10 wins, with 3 more games to go, and some people are even predicting them to make the Super Bowl. It would appear that offense not only sells tickets, it might also win championships as well.

What about passer rating, what can we do with that information? The NFL passer rating formula ranges on a scale from 0 to 158.3 based on completion percentage, yards per attempt, touchdowns per attempt, and interceptions per attempt. So it essentially encompasses a lot of the other stats in our database. A higher passer rating typically means you are the better quarterback. This further solidifies the position that QB is the most important position on the field, and teams need to dedicate their time to finding the right one for their respective teams. Recruiting, scouting, and drafting a QB needs to be a team's top priority until they find one. Once again

going back to the Buffalo Bills, they missed the playoffs for 17 straight seasons before they drafted Josh Allen. Now they are on track to make it for the third time in four years.

This also may help with the ever growing world of sports betting. January through August, revenue totaled about $178 million, up 16.5% versus 2019, despite the several-month sports hiatus earlier in the year. (Sayre, 2020) Hundreds of millions of dollars are spent every year, so we might as well spend those dollars wisely. If we take into account how much a team scores, or what the QB's passer rating trends are for the season, can we make an educated guess on what the outcome of the game will be? My thinking is yes, absolutely. If you gained nothing else from this report, I hope I could help you win a few bucks.

Appendix I

Codebook for QB Game Log Dataset

The following table outlines the data that is available to us in the game logs dataset.

Column Name	Column Description
Played Id	Unique ID assigned to each quarterback
Name	Quarterbacks Name (Last, First)
Position	Position this individual played
Year	Year in which game was completed
Season	Type of game that was played (Regular season or preseason)
Week	The week of the season the game was played
Game Date	Date game was played
Home or Away	Whether the game was played at home or away stadium
Opponent	Team the QB played against
Outcome	Win, Loss, or Tie
Score	Final score of the game
Games Played	Games Played
Games Started	Games Started
Passes Completed	Number of passes completed by the QB
Passes Attempted	Number of passes attempted by the QB
Completion Percentage	Passes Completed divided by Passes Attempted
Passing yards	Number of passing yards for the QB
Passing Yards Per Attempt	Passing Yards divided by Passes Attempted

TD Passes	Number of touchdown passes by the QB
Ints	Number of interceptions thrown by the QB
Sacks	Number of times QB was sacked
Sacked Yards Lost	Number of yards lost on sacks
Passer Rating	Calculated passer rating
Rushing Attempts	Number of rushing attempts by the QB
Rushing Yards	Number of rushing yards for the QB
Yards Per Carry	Rushing yards divided by rushing attempts
Rushing TDs	Number of rushing touchdowns scored by the QB
Fumbles	Number of total fumbles by the QB
Fumbles Lost	Number of lost fumbles by the QB

References

Barnwell, B. (2017). The NFL stats that matter most. https://www.espn.com/nfl/story/_/id/20114211/the-nfl-stats-matter-most-2017-offseason-bill-barnwell

Chakure, A. (2019). Decision Tree Classification: An introduction to Decision Tree Classifier. https://medium.com/swlh/decision-tree-classification-de64fc4d5aac

Garg, R. (2018). 7 Types of Classification Algorithms. https://analyticsindiamag.com/7-types-classification-algorithms/

Gillies, K. (2016). NFL Statistics. https://www.kaggle.com/kendallgillies/nflstatistics

Groysberg, B., Hecht, E., and Naik, A. (2019). Who's the Most Important Member of an NFL Franchise? https://hbr.org/2019/04/whos-the-most-important-member-of-an-nfl-franchise?utm_campaign=hbr&utm_medium=social&utm_source=twitter

Katz, S. and Burke, B. (2016). How is Total QBR calculated? We explain our quarterback rating. https://www.espn.com/blog/statsinfo/post/_/id/123701/how-is-total-qbr-calculated-we-explain-our-quarterback-rating

Otten, M. (2018). Does defense actually win championships?. https://theconversation.com/does-defense-actually-win-championships-89628

Pensado, E (2008). NFL Analysis: Is QB The Most Important Position In All Of Sports?. https://bleacherreport.com/articles/55776-nfl-analysis-is-qb-the-most-important-position-in-all-of-sports

Sayre, K. (2020). The NFL is Back and Sports Bettors Are Following. https://www.wsj.com/articles/the-nfl-is-back-and-sports-bettors-are-following-11600112703