Capstone 2 Milestone Report

Predicting Fantasy Points for NFL Quarterbacks

Using regression techniques to build winning fantasy football lineups

Introduction

The fantasy football industry has seen a massive spike in both revenue and the number of fantasy players in the past five years. One of the main reasons for this spike is how the concept of daily fantasy sports (DFS) has revolutionized the industry. Daily fantasy players no longer need to deal with the risk of season-ending injuries to their star players, drafting player for a season that underperform, making trades that end up in favor of the other team-owner, etc. Also, daily fantasy players are able to win money and get paid immediately after the final contest in a slate has been completed, instead of having to wait until the end of the season to collect their winnings. Players have the ability to draft a new fantasy team each week, or every day for certain sports like baseball or basketball. Contests between players who participate in DFS may include anywhere from 2 participants (1 versus 1) all the way up to 500,000+. Entry Fees for participants may range from no cost (usually a promotional contest with a small reward for playing) up to $10,000, and prizes may reach up to $2,000,000 for winning a tournament with a high volume of participants. Daily fantasy sites, such as DraftKings and FanDuel are more popular and lucrative than ever. The motivation of this project is to explore ways to implement machine learning to predict points scored for players, which will help us build profitable daily fantasy sports lineups. Those who enjoy playing NFL fantasy football could use our predictions as recommendations for using players in their lineups to help them make profitable decisions. For the purpose of this project, we will be focusing on predicting fantasy points for the quarterback position, as it is arguably the most important position to predict in fantasy football. We will, however, be collecting data for all positions for future work.

How it Works:

FanDuel and DraftKings both host DFS games in nearly every major sport, which include NFL, MLB, NBA, NHL, and NCAAF among others. For our purposes, we will be exploring professional football (NFL), and using FanDuel's format for building a team. For each NFL contest using FanDuel, the participant is allowed a $60,000 salary cap to draft his or her players. For each team, the participant may choose one quarterback (QB), two running backs (RB), three wide receivers (WR), one tight-end (TE), one flex (a choice of an extra RB, WR, or TE), and one team defense/special teams (DST). The participant chooses players for his/her team, and pays a fee to enter a contest. The site, or host (FanDuel in this case), of the DFS contest automatically takes 10% of the participant's entry fee upon entering the contest. This is known as the "rake", and is how the site makes its profit. Once a participant has entered a contest, in order for his or her entry to win money, it must place higher than a certain percentage of entries in the contest. The amount of money a single entry wins depends on what type of contest it is, as well as how many entries are in it. For example, if someone entered a two-person contest, and paid $1 to enter, that person's lineup must score higher than the opponent to win $1.80, since the host keeps 10% of each entry. Another example is if someone entered a tournament that had 20,000 entries for $5, the host keeps 10% of $100,000, and the remaining $90,000 would be up for grabs among the entries. The top prize might win $40,000, second place might win $15,000, third might win $5,000, and the remaining $30,000 is distributed in a decreasing manner among the remaining top 10-15% of entries in this contest.

Data

Attribute Selection

The attributes that will be used for this project will be mainly individual player statistics based on a player's overall history, and that player's recent history. We will also be using matchup attributes for an instance. Every position within a lineup is different, and thus different attributes will be important. Since we are focusing mainly on the quarterback position, we will need to take FanDuel's scoring system for quarterbacks into account, and will want to use individual overall history attributes, such as pass yards per game, rush yards per game, pass TD's per game, rush TD's per game, pass attempts per game, average fantasy points per game, and others. For current matchup attributes, we will be using defensive points allowed per game, whether the match is home or away, defensive yards allowed per game (pass, rushing, receiving), etc.

Obtaining the Data

For all of the attributes described, there are many sources from which to obtain the data. One source is pro-football-reference.com, which has a database that contains nearly every relevant football statistic for every NFL game played dating back to 1920, which includes weather, venue, and referee data. Another source for obtaining the needed data will be fantasydata.com, which has defensive and offensive rankings for every team in each contest, and data from sports books, which includes the totals for each game. Additionally, fantasydata.com has every relevant fantasy stat needed for individual players.

Data Collection

To collect the data, I built two different data scrapers using Python's BeuatifulSoup4 library. The first scraper was for pro-football-reference.com. This scraper collects data from each individual game played since 2010. The variables collected in this scraper include the game ID, date, teams, metrics that show how strong a team's offensive, defensive, and special teams performance was, and team statistics such as first downs, turnovers, passing yards, time of possession, etc. The second data scraper collects odds data from fantasydata.com. The odds data includes data from every game dating back to 2010. The variables collected include the data of the game, who is favored to win the game, how many points the favorite is favored to win the game by (spread), who the underdog in the game is, the total number of points expected to be scored by both teams combined (total), the moneyline for both the home and away team, the season, and the week. We also will be using fantasy statistics for every player from every game dating back to 2010 using fantasydata.com. Additionally, we will be using data that we collected in the first capstone project, which includes team grades from every game from profootballfocus.com, venue data, and footballoutsiders.com data which measures teams' offensive, defensive, and overall efficiency.

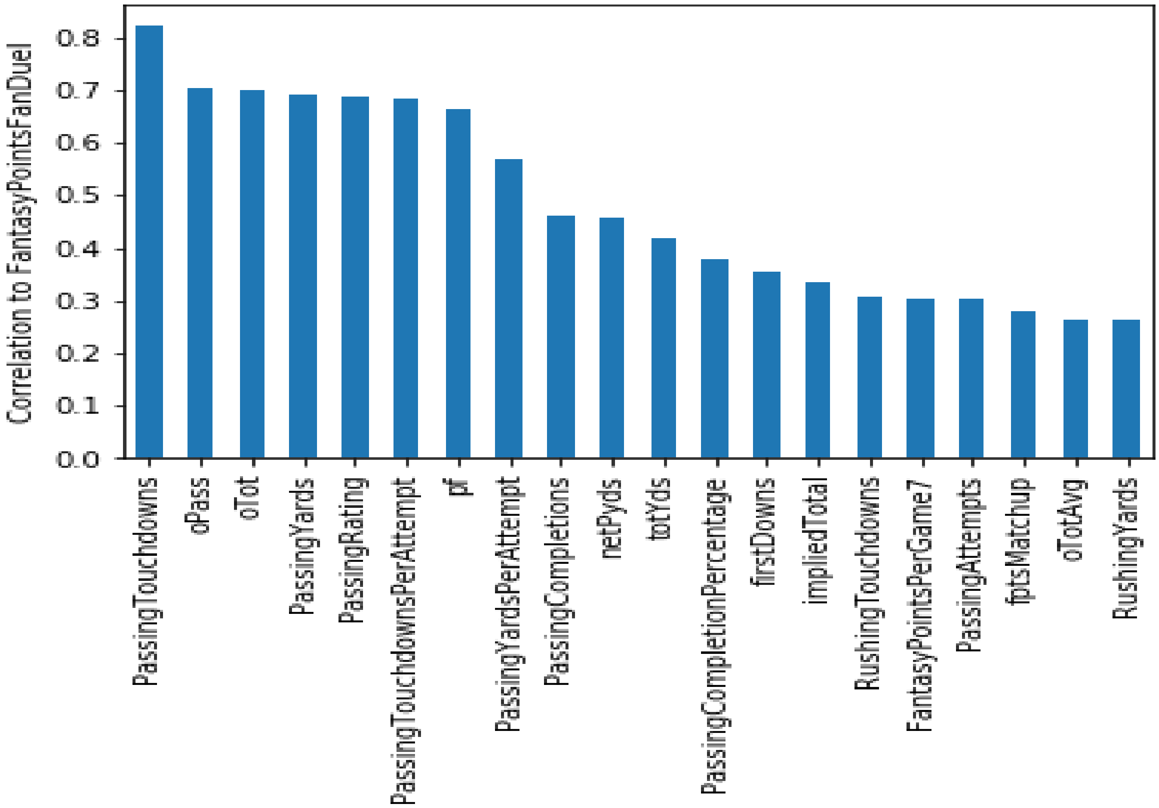
Data Wrangling

For data wrangling, many techniques were used to get the data in a usable format for analysis and modeling. I used Python's Pandas library for a lot of this process. Using read_csv(), the data that was scraped from the .csv files were put into a Pandas dataframe. Column operations were performed on the dataframe to create new variables that could be useful for analysis and modeling. For example, a variable named FantasyPointsPerGame7 was created by averaging each quarterback's fantasy points over his last 7 games going into his upcoming contest. This was done by setting the index for the dataframe by date and name, grouping the dataframe by name, and using a rolling average method for each quarterback in the dataframe. This process was completed for many different variables in which a moving average would be useful, such as passing touchdowns over the last seven games, passer rating over the last seven games, etc.

Furthermore, date columns were dealt with by transforming them into datetime objects, string methods were used on columns to clean string variables, team name columns needed to be cleaned for consistency among all tables, so string methods and dictionaries were used to create one consistency, and joins and merges were used on tables to prepare the data for analysis and modeling.

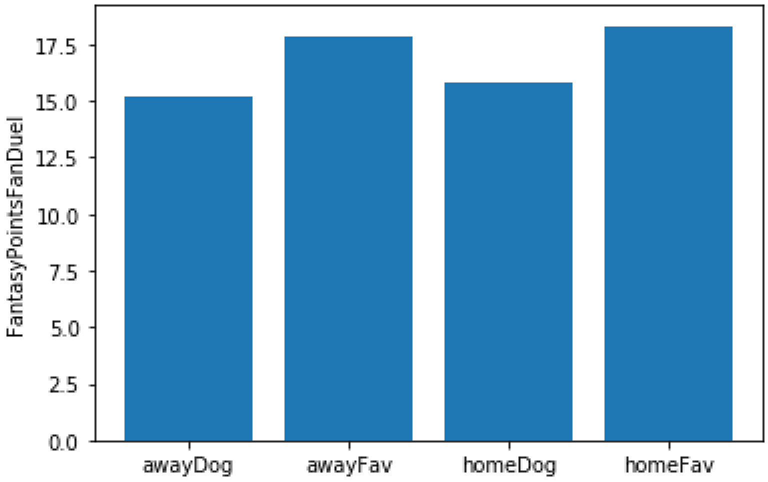
Exploratory Analysis

Now that we have the wrangling process out of the way, we can begin digging into the data to find trends and patterns. We need to keep in mind our main goal, which is to predict fantasy points for quarterbacks. With this in mind, let's first take a look at some correlations to a quarterback's fantasy points.



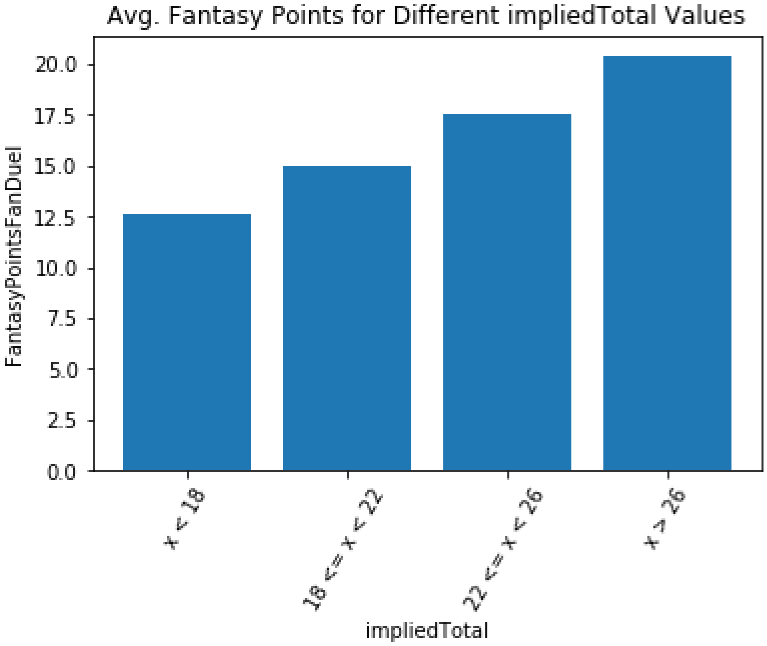
Looking at the bar graph above, we can see that passing touchdowns have the strongest correlation to fantasy points for quarterbacks. The next tier of correlations includes oPass, which is a measure of the quarterback's team's offensive points gained via passing plays, oTot, which is a measure of a qb's team's offensive points gained by the team's offense as a whole, passing yards, passer rating, touchdowns per attempt, and pf, which is a team's total points scored. We notice that passing attempts doesn't crack the top 15 correlations to fantasy points. A common point of emphasis among the fantasy community is that quarterbacks who throw the ball lots of times will score more points. While there is some truth to this, it may be a better strategy to target quarterbacks who are projected to be more efficient, rather than targeting qb's who are projected to throw a lot.

A good way to target efficient quarterbacks is to look at the odds data. The ‘total’ variable indicates how many combined points two teams are projected to score via the Las Vegas sports books. The ‘spread’ variable indicates how many points the team who is favored in a game is expected to win by multiplied by -1. The spread will be negative for a team who is favored to win because this is essentially a ‘handicap’ for the better team. If a team is favored to win by 8, the spread variable will be -8. These sports books are incredibly good at setting the totals of games, as well as spreads, taking bets from both recreational bettors and professionals on the totals and spreads, and turning a profit. We can leverage this information to our advantage when targeting quarterbacks for our lineups.



As we can see from the above image, quarterbacks who are favored to win average more fantasy points than quarterbacks who are underdogs, regardless of whether they are home or away. Being a favorite means that the spread < 0 for a quarterback. In this scenario, quarterbacks average approximately 17.6 fantasy points per game, as opposed to underdog quarterbacks, who average only around 15 fantasy points per game.

The impliedTotal variable takes into account both the spread and the total, and has an even higher correlation to fantasy scoring than the total. Implied total for a quarterback is calculated by taking (total - spread)/2. For example, if a team is favored to win by 7 in a game with a total of 48, the implied total for that team's quarterback is (48 - (-7))/2 = 55/2 = 27.5. This team is expected to score 27.5 points in this particular game. The underdog in this game will be expected to score 20.5, since they are 7-point underdogs. Notice that adding 27.5 and 20.5 gives us our 48-point total. Now that we have a better understanding of impliedTotal, let's look at how we can use it.



Looking at the graph above, we notice that as impliedTotal increases, a quarterback's fantasy output also increases. When a quarterback's implied total is < 18, he should not be a target in our fantasy football lineups, as quarterbacks in this scenario only average 12.5 points per game. Conversely, quarterbacks with an impliedTotal of > 26 should be targeted regularly, as they average over 20 fantasy points per game. This information is valuable, since most fantasy football players target quarterbacks in games with a high total instead of taking both total and spread into account. The ‘total’ variable should be considered, but it can be argued that ‘impliedTotal’ should be weighed more heavily.