PREDICTING A QUARTERBACK'S FANTASY FOOTBALL POINT OUTPUT FOR DAILY FANTASY SPORTS USING STATISTICAL MODELS

by

NICHOLAS AARON KING

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2017

To my parents and faculty who have constantly supported and encouraged me in the pursuit of this degree.

ACKNOWLEDGEMENTS

I would like to thank Professor Aera LeBoulluec for her endless support during my time at the University of Texas at Arlington. She was the first one who introduced the field of data science to me and sparked my interest in it. There are a number of opportunities that I have experienced in the past two years at UTA that I would not have been blessed with had she not gone out of her way to help me. I am confident that my time at UTA would have been far less challenging and rewarding had she not been here.

This thesis research was no different. Dr. LeBoulluec enthusiastically supported my ideas and let me run with them whenever inspiration struck. She also knew exactly when to reel me back in and focus on a topic or idea until its completion. I have learned an immense amount throughout this work, many things I would not have had the opportunity, or time, to learn in a traditional classroom setting. I know that this information has positioned me well for a future career in data science.

Finally, I want to thank my parents and brother for their continuous encouragement. Without their support and love I would have had a hard time convincing myself to go back to school and an even harder time finishing. They were always, and continue to be, in my corner.

April 12, 2017

ABSTRACT

PREDICTING A QUARTERBACK'S FANTASY FOOTBALL
POINT OUTPUT FOR DAILY FANTASY SPORTS
USING STATISTICAL MODELS

NICHOLAS AARON KING, M.S.

The University of Texas at Arlington, 2017

Supervising Professor: Aera LeBoulluec

In the new age of daily fantasy sports (DFS), fantasy football has become an enormous revenue generator for DFS sites, such as DraftKings and FanDuel. Both companies are valued over $1 billion. However, previous analysis done by popular DFS site Rotogrinders has shown that only the top players are consistently winning, the top 10 players much more frequently than the remaining 20,000 players. Using complex statistical models they're able to identify top athletes and value picks (based on an athlete's draft 'salary') that the average player might not be aware of.

There is a need to evaluate which methods and algorithms are best at predicting fantasy football point output. These methods could then be applied to future DFS contests outside of football to see if they predict other fantasy sports point output well. There are few resources and little literature available on this subject. Several factors contribute. Daily Fantasy Sports are still relatively new, and many people are still just starting to get involved in them. Also, very few people have published their work on their custom models or significant variables, since they are generally

developing these models for personal use in an attempt to gain an edge in DFS contests and win money. Thus, there is little to no motivation to make their research or methods publicly available.

This research will attempt to predict the weekly point output of a quarterback based on a variety of attributes and metrics. Finding the important variables and statistical models and learning how to address the volatility in week-to-week performances for a quarterback will allow us to expand this to other player positions in the future. In addition to understanding the best algorithms to apply to weekly point prediction and the best variables to use to predict a quarterback's output, this research also seeks to answer the question that is a currently being debated in courtrooms across the country - should DFS be considered a legal game of skill, or a game of luck, and therefore online gambling?

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

LITERATURE REVIEW & INTRODUCTION

There is little research currently available on building projections for weekly fantasy football and even less on the models behind them. The majority of fantasy football research and articles focuses on the traditional, season-long format that has been the norm for decades. This is the established way of participating in fantasy football and has the largest following. There is little to no motivation for daily fantasy sports (DFS) players with statistical models to make their work public. This lack of research and published work means there is potential to set a baseline, though. Establishing some effective models and outlining the algorithms behind them will allow for more players to become engaged in DFS contests and may encourage others to improve upon this research. This can lead to improvements in projections across the entire fantasy sports industry, not just football.

There are several reasons why there is so little research to be found on projecting DFS fantasy football results. For starters, DFS has only been around several years. The two largest companies in DFS sports, FanDuel and DraftKings, were founded in 2009 and 2012 respectively, and it took several years for them to gain traction and popularity. Even though these two companies have spent hundreds of millions of dollars on advertising and marketing, they still only represent a small percentage of the fantasy sports world. It's estimated that 57 million people in North America play some kind of fantasy sport, yet FanDuel and DraftKings combined have about 5 million users, less than 10% of the market [1].

There are other reasons that there has been so little research. The most important factor is that there is no motivation for people with effective projection models to make their work public. DFS is built on making fantasy football a cash game, and for many of the top players it is their sole source of income, with some players making over $1 million during the season. Their statistical models are kept secret to ensure they have an advantage over the competition and continue winning. These top players use their custom-built models and enter hundreds of lineups every weekend. The models are built in such a way they can pull real-time information and an athlete's salary and can output hundreds of lineups using optimized combinations of players they project to do well, while also staying under the salary cap limit that the DFS sites impose. In addition, they can update their entries with the click of a button. An average participant would have to manually go in and remove an athlete if it was discovered at the last minute that the athlete would not be playing due to an injury; but the top players, with advanced models, can automate the entire process and have the athlete removed and their hundreds of entries updated with a single command. Therefore, they never have to worry about accidentally starting an athlete who isn't playing.

The lack of published research can also be attributed to a lack of NFL data in general. The NFL is the most popular and profitable sports league in the United States, but until recently, has lagged way behind other leagues in sophisticated use of data analysis, and has historically had the least amount of data available. The challenge has to be due in part to the complexities involved with tracking the players on the field. There are so many offensive and defensive sets, plus constant player substitution, that the high amount of activity and variables in play make it difficult to gather meaningful player data [2].

This paper will attempt to investigate the limited data and literature available to analyze the best algorithms in projecting a NFL quarterback's weekly fantasy football performance. Understanding how each model works and comparing its results to others will help determine what models and algorithms show promise in projecting fantasy football moving forward and which ones might not be worth an individual's time if they were attempting to build their own projection models. The seven statistical models that were employed in this research were: backwards stepwise regression, support vector regression (SVR), regression tree, random forests, boosting, artificial neural networks (ANN), and principal components regression (PCR).

A backwards stepwise regression is one of the simplest models that was tested. In this automated form of a regression the model is initially run with all of the x variables considered, and then the least significant variable during each step is removed. The process stops when the only variables remaining are considered significant and have p-values that are less than or equal to the pre-determined value to remove (the α value, in this research we used α=0.10). A backwards stepwise regression models the relationship between x-variables (predictors) and the y-variable (response) by fitting a linear equation to the data. This equation is composed of coefficients that are associated with each x variable found to be significant in the prediction of the response variable - in our case, *DK_points_scored*. Support vector regression (SVR) comes from the popular algorithm called Support Vector Machine (SVM) and was also investigated. The SVR performs a regression like the previous examples, but also introduces *epsilon boundaries* around the line that represent error thresholds. Any errors or predictions that are greater than the ϵ bands are penalized. The goal is to find a function $f(x)$ that deviates from y_n by a value no greater than ϵ for each training point x, and at the same time is as flat as possible [4].

Tree-based methods were also considered in this research. This includes regression tree, random forests, and boosting. Regression trees are a type of decision tree (as are classification trees) and are useful when your data has lots of features that interact in complicated ways. A regression tree sub-divides the feature space into smaller regions where the interactions between variables are more manageable. This space is then subdivided further, a method called *recursive partitioning*, until the chunks of space are manageable enough to fit simple models to them. Each of the terminal nodes, or leaves, of a tree represents a cell of the partition, which includes a simple model that applies to that cell only. You can find what cell a sample point x belongs to by starting at the root (upper) node of the tree and answering a sequence of questions through the branches of the tree to get to the correct cell [5]. In our case, the regression tree uses these partitions to predict the fantasy points of a quarterback's performance. Random forests are an extension of these ideas. The random forests algorithm grows hundreds or thousands of regression trees and then averages their output to generate results. It is considered an *ensemble* method, because it takes numerous *weak leaners* (in this case a single regression tree), and combines them to form a single *strong learner* (our entire forest) [6]. Boosting is similar to the random forests method and averages many regression trees. Random forest trees are grown in parallel with no interaction, though, while the trees in the boosting algorithm are grown sequentially, each to re-weighted versions of the training data [7]. This allows for each tree to get *boosted* from the learning of the previous tree.

Neural networks are known for their pattern recognition, forecasting, prediction, and classification abilities. They have been successfully applied in a wide variety of fields. A neural network was investigated in this research to see how it might perform on fantasy football data. A multilayer perceptron (MLP) feedforward artificial neural network was the variety explored. As its name implies, the information only moves

forward from the input layer, to the hidden layers, and finally to the output layer. The MLP used in this research consisted of six input layers, two hidden layers with three and two neurons, and one output layer. In our model this output layer represented the projected value for *DK_points_scored*.

Finally, a principal components regression (PCR) was performed. PCR is a regression technique based on principal component analysis (PCA). The idea behind a PCR is to calculate the principal components and then use some of these components as predictors in a linear regression model (fitted using the typical least squares model), similar to a MLR. Principal components are sets of of linearly uncorrelated variables. In some cases a small number of principal components can be enough to explain over 90-95% of the variance in the data. This is why employing only a few components can often perform better than using a large number of variables in a model. Thus, a PCR can help with dimensionality reduction, in addition to avoiding multicollinearity between predictors and overfitting a model [8].

It seems reasonable to suggest that daily fantasy sports will only grow in popularity as more people become familiar with the concept of DFS itself. DFS continues to have it's legality challenged in some courtrooms, but it appears public attitudes have changed. Even during the course of this research, several states have already made it legal, with my home state of Iowa voting on the concept in later 2017. As of January 2, 2017, only 10 states remain that block all formats of daily fantasy sports [9]. As the NFL invests more money and resources into tracking data and data analytics, one would expect to see more individuals and hobbyists getting involved with DFS fantasy football. There has also been talk about having separate contests on some DFS sites where only players with predictive models are able to play. This would allow people who may care more about statistics and modeling than football to 'compete' and pit their model against another player's.

Other people who have read this work will also be able to take it further on their own. The methods presented in this research and the error metrics charted allow for others to have a baseline and to begin building their own predictive models. I also see more avenues that this research could follow. In the future I would investigate which models and variables best predict the other positions on a fantasy roster. Once this is decided the research turns to an optimization problem - how to create lineups that offer the highest projections, while still staying below the $50,000 salary cap most DFS sites impose. I would also look at writing a script of code to automate the entire process. Using Python software there are flexible ways of developing an entire program that will pull up-to-date values for important variables from web sources - called 'web scraping' - and automatically run them in the model. When this is completed, a host of alternate lineups that offer high projections can then be fed into DraftKings. Once this process is automated it would most closely resemble the models used by pros, where only a few clicks are needed to enter dozens or hundreds of different contests. Clearly, there is still much that can be done. The research presented here, however, has provided a solid foundation upon which to build and offers a portfolio of work that can help pave the way for future participants to create their own models to project weekly point output.

CHAPTER 2

DAILY FANTASY SPORTS

2.1 Background

Daily fantasy sports are still a somewhat new phenomenon. While traditional fantasy sports have been around for decades, DFS sites are only a few years old. The two largest players in the DFS world are Draft Kings and FanDuel. Together, these two companies control around 95% of the daily fantasy market [10]. FanDuel was established in late 2009, while DraftKings followed in early 2012. In the early years they fought to secure funding from venture capitalists and, eventually, U.S. professional sports organizations [11]. Both companies have been valued above $1 billion. While daily fantasy games exist in nearly every sport (i.e., football, soccer, baseball, basketball, golf, auto racing, etc.) fantasy football is the most popular and will be the focus here.

2.2 Differences Between Traditional and DFS Formats

It's important to understand the difference between DFS fantasy football, and traditional fantasy football. A traditional fantasy football league consists of 10-12 teams, each one run by a member of the league. Before the NFL (National Football League) season starts, members of the league draft athletes they want on their team. A draft order is followed and all team owners draft until their roster is filled. Usually this roster is made up of a quarterback (QB), two running backs (RB), two wide receivers (WR), a tight end (TE), a 'flex' spot (where a RB, WR, or TE can be started), a defense/special teams unit, and a kicker. These players comprise your

OVERVIEW | STATS | NEWS | PROJECTIONS | SCHEDULE | RANKS

STARTERS			2016 SEASON				OFFSEASON				
SLOT	PLAYER, TEAM POS	ACTION	PRK	PTS	AVG	LAST	PROJ	OPRK	%ST	%OWN	+/-
QB	Drew Brees, NO QB	MOVE	3	323	20.2	20	--	--	86.5	99.4	+0
RB	David Johnson, Ari RB Q	MOVE	1	393	24.6	6	--	--	98.8	100.0	+0
RB	Latavius Murray, Oak RB	MOVE	13	197	12.3	3	--	--	69.7	96.6	+0.1
WR	Davante Adams, GB WR	MOVE	9	241	15.1	23	--	--	73.7	91.2	+0.8
WR	Pierre Garcon, Wsh WR	MOVE	24	192	12.0	13	--	--	25.5	46.3	+0
TE	Jimmy Graham, Sea TE	MOVE	4	183	11.4	10	--	--	78.8	97.0	+0.2
FLEX	Willie Snead, NO WR Q	MOVE	32	184	11.5	11	--	--	39.5	77.7	+0.1
D/ST	Packers D/ST D/ST	MOVE	20	75	4.7	1	--	--	26.2	55.8	-2.7
K	Caleb Sturgis, Phi K	MOVE	4	144	9.0	8	--	--	36.1	45.0	+0.3
BENCH			2016 SEASON				OFFSEASON				
SLOT	PLAYER, TEAM POS	ACTION	PRK	PTS	AVG	LAST	PROJ	OPRK	%ST	%OWN	+/-
Bench	Randall Cobb, GB WR	MOVE	51	143	8.9	0	--	--	7.7	78.7	-0.2
Bench	Marvin Jones, Det WR	MOVE	43	165	10.3	12	--	--	21.5	75.1	-0.2
Bench	Eagles D/ST D/ST	MOVE	6	127	7.9	15	--	--	25.1	46.1	+1.9
Bench	Blake Bortles, Jax QB	MOVE	8	260	16.3	14	--	--	36.6	63.4	+2.5
Bench	Tim Hightower, NO RB	MOVE	39	114	7.1	8	--	--	15.8	54.8	-1.4
Bench	Wendell Smallwood*, Phi RB IR	MOVE	71	47	2.9	0	--	--	0.1	7.0	-0.5
Bench	Malcolm Mitchell, NE WR Q	MOVE	81	91	5.7	0	--	--	1.1	34.1	-6.2
IR	Eddie Lacy*, GB RB IR	MOVE	74	40	2.5	0	--	--	0.3	11.5	+0.7
IR	Jamaal Charles*, KC RB IR	MOVE	110	12	0.8	0	--	--	0.4	11.2	+1.1

Figure 2.1. Standard format fantasy football roster on ESPN.

starting lineup. You also draft whatever players you like to leave on your 'bench' - players you may need to start later in the season. Benches usually range from 5-9 players and can include spots for players on injured reserve (IR). An example of a fantasy football roster is shown in Fig. 2.1.

Each week in the league two teams are matched up against each other. The fantasy points your teams score are a direct reflection of how your individual players do in their actual games. For example, in standard scoring, a RB is awarded 1 point for every 10 rushing or receiving yards, and 6 points for a touchdown. So if your RB rushes for 100 yards, catches a 12-yard pass, and scores a touchdown, he would score 17 points. Scoring settings are different for each league and website used, but it's clear that if your athletes are doing well in their real individual games from a

statistical standpoint, then your fantasy team will be doing well also. Traditional fantasy football is also similar to the NFL in the sense that you can make trades, cut underperforming players, and add athletes from waivers or free agency. It's also the nearly the length of an actual NFL season. The NFL regular season lasts for 17 weeks, fantasy football usually lasts 13-14 weeks, with 2 weeks of playoffs following. A winner is determined before the NFL playoffs begin.

The length of competition is a significant difference between DFS football and the traditional format. DFS sites rarely have these season-long commitments. They run an accelerated form, with most competitions taking place over the course of a week or single day. DFS allows you to draft a new team each week and eliminates the other responsibilities of season-long commitments, such as trading, dropping or adding athletes, or having to manage a bottom-feeding team all season, which keeps players engaged.

DFS football is also more money driven. These websites earn their living from making fantasy football a cash game. Each week you pay to enter a team, from a couple of dollars, to several thousands, depending on the type of league and competition you'd like to face. DFS sites receive a portion of the entry fee from each player and then pay out the rest to the winner. While the traditional format pits one member's teams against another in a head-to-head battle, DFS sites offer you the chance to play in nationwide contests where the highest scoring roster takes the winnings among all entries in that league. This has led to large payouts for many members.

2.3 Benefits of Statistical Models

DFS sites also add a salary component into the drafting process. Traditional fantasy sports allow a participant to draft the best athlete possible that is still on the board when it's their turn. DFS sites allocate participants a fixed salary cap

of $50,000 that must be used to draft an entire roster. This means not only does a participant have to pick and choose which athletes they think will do well, but also athletes that the participant can afford. This puts a premium on finding 'sleepers' or value picks. For example, an elite-level quarterback with an easy matchup might cost $9,000, and a mid level QB with an average matchup might cost $6,500. If a player chooses the elite QB they've now spent nearly 20% of their salary cap on one player and still need to draft additional athletes (2 RBs, 2 WRs, 1 TE, 1 FLEX, 1 D/ST, 1 K) to fill out their team. So if a participant can find a QB, or any position for that matter, who is relatively cheap, but will give the same point output as an expensive QB, they have already positioned themselves to do better since they will now have more money to spend on other high-level athletes. An example of a nearly ideal (2nd place finisher) DraftKings lineup in a nationwide contest is shown in Fig.2.2 [12]. This entry won the player $350,000.

This is how the top players with statistical models have done so well. They are able to generate hundreds of different lineups where they try to optimize the relationship between projected QB point output and cost. Finding those sleeper picks also benefits them in the nation-wide contests even more. Picking athletes that have a low ownership percentage can provide separation between their entries and thousands of others. For example, if a player and 20% of other nationwide participants start an elite quarterback like Aaron Rodgers it won't give your team a lot of separation if he does well, since so many other people also started him. But if a participant started Andy Dalton, whom just 2% of entries started, and he does well, then they've just potentially passed up thousands of entries that didn't pick him and saved roster money doing so. Spotting a value pick like Dalton allowed the player to have more money to spend on other positions than if they had paid more for Aaron Rodgers. This is similar to nationwide March Madness competitions for

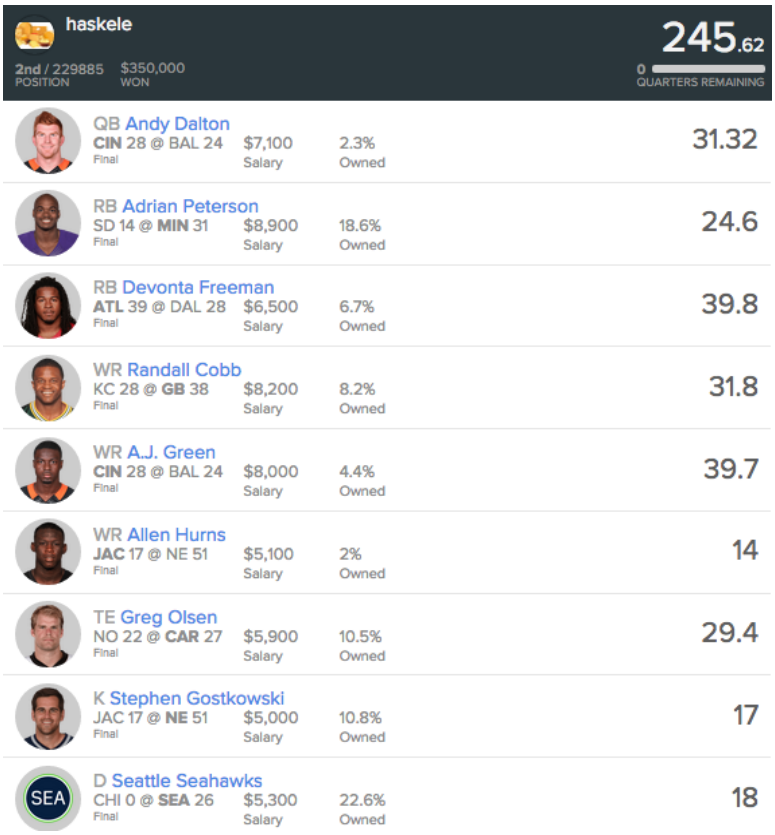

Figure 2.2. Roster of $350,000 winner on DraftKings.

NCAA basketball. People who pick four #1 seeds to reach the Final Four rarely win big since it's often what the majority does. The individuals who pick the big upsets and pick lower-seeded teams to go far usually end up winning, since fewer people's brackets have those games called correctly. If this research can identify variables and algorithms used to accurately predict weekly fantasy point output for QBs, then we can make predictions for each of the starting QBs in the NFL and pick which one might offer an elite-level point output for the lowest cost.

Finally, it's also important to briefly point out how challenging projecting weekly point output has been and remains. Almost all websites that formulate fantasy football projections refuse to keep their projections up past the current week.

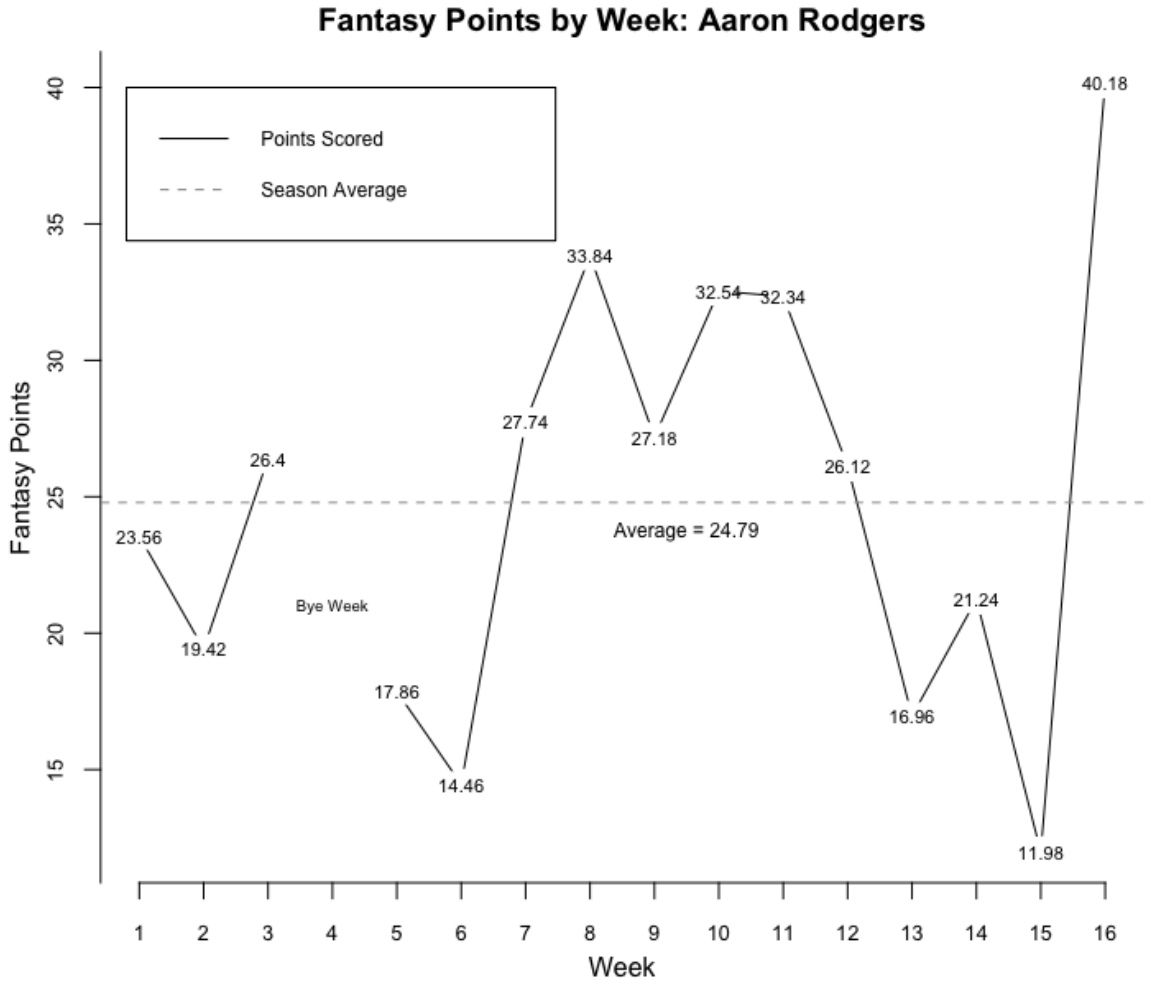

Figure 2.3. Aaron Rodgers' DraftKings point output by week..

This is for a simple reason - they don't want visitors or paying subscribers to see how *in*accurate their projections might have been. Finding data on historical weekly projections is challenging because of this. Fig. 2.3 shows how hard it is to project an athlete's, or in our case, a quarterback's output on a week-to-week basis.

Aaron Rodgers was the top fantasy football quarterback during the 2016 season. One might then expect that an elite QB would put up consistently good performances, but this was not the case. There's no better example of this than Rodgers' weeks 15 and 16, where he put up his season low, only to follow that performance with over

three times as many points, his season high. This is what makes fantasy football so tough to predict, and why a well-developed predictive model can give a player a huge advantage.

This paper is organized as follows. Chapter Three will discuss the data used and the preprocessing steps that took place in order to have clean data on which to run the algorithms. Chapter Four will discuss the models and methods employed. In the Results section all of the models will be compared based on their error metrics. The Conclusion will offer a summary and the implications of this research.

CHAPTER 3

DATA PREPROCESSING

3.1 Data Aggregation and Cleansing

The data used in this research was scraped together from several websites and sources. Originally data was culled from a GitHub user who had aggregated large amounts of historical NFL data dating back to 2009 [13]. I brought this database into a SQL search and used it to find all of the names, teams, birthdates, heights, weights, opponents, home games, and years professional for the dataset. NFL Combine results were found via the NFL Draft Scout website [14]. The NFL Combine is an annual pre-NFL Draft event, where college athletes are invited to essentially show off their skills and athleticism in front of NFL teams scouts in the hopes of improving their draft stock. They do a variety of speed, agility, and strength tests. Pro-football-reference.com was used to find other crucial stats, as well as calculate several additional metrics, such as adjusted net yards per passing attempt (ANY/A). I also paid for a subscription to Fantasy Data [15] to find snap and utilization percentages, as well as previous weekly projections and DraftKings salaries. Since FanDuel is currently being challenged in Texas I decided to focus specifically on DraftKings' salaries and projections, though the two would not vary drastically.

Additional projections were found via CBS Sports, FantasyPros, FanDuel, Fantasy Football Nerd, and Fantasy Football Today. Finding historical projection information from these sites was extremely hard. Once a week has passed the projections are typically removed from websites. This is for one simple reason - these sites don't want to show how wrong they were. Obviously there is so much volatility in project-

ing week-to-week points, that websites, especially large ones like ESPN, don't want you to see how off their projections for a player might have been that week. This is the challenge in gathering historical data on fantasy projections.

Weekly QB performances from weeks 1-7 of the 2016 NFL season were used for a training dataset, while weeks 13-16 were used for a validation dataset. A general practice in developing predictive models is to split the data into training, validation, and testing sets. It is common to cut the train and validation data into roughly a 70/30 split. Each model is trained and 'learned' on the training set and then validated to see how well it generalizes to data it has never seen before with the validation set. Finally, the best chosen model is applied to the testing dataset. In this research the testing dataset composed of 16 samples from Aaron Rodgers' season, from weeks 1-16. The training set in our research led to a study of 43 different quarterbacks and 212 games played. Thus this dataset had 212 samples. The validation dataset featured 33 QBs, with 126 samples. Sample numbers in the original training and testing sets were reduced by 37 and 90, respectively, in order to focus only on QBs that were identified as starters. This is logical, as a player using DraftKings would never select a second or third string QB for their week's team, despite several of these backup QBs logging statistics during games due to a starter's injuries or poor play.

Imputation was done using column means where necessary. This was the case when working with rookie quarterbacks. Many of the variables in the datasets were values corresponding to performance metrics from the previous season. As a rookie, of course, you don't have a previous season's results. Imputing these missing values with the column mean is a typical practice done in data mining and allowed us to work with a full dataset in the research.

Data types were specified as well, including coding several variables as factors or categorical variables. Many of the models used in data science and machine learning,

however, can only work with datasets that are strictly numeric. To process our data to run these models we dummy coded the categorical predictor variables. Dummy coding uses only ones and zeros to convey all of the necessary information for the factor or categories [16]. For example, in the case of the home_game variable the values were 'Yes' or 'No.' To dummy code this we simply made 'Yes'=1, and 'No'=0. This allows R to run the algorithms and models without losing any of the information in our predictors.

Standardization was also done to ensure that the scale of any one predictor didn't have more influence than another. Standardization is another common practice in data mining and involves subtracting the column mean from each value in the column and then dividing by the column standard deviation. This makes comparison between predictors much easier. Standardization does not alter the correlations among the variables [17].

3.2 Variables Considered

After the preprocessing took place our datasets were now all numerical and standardized and had no missing values. Our final training, validation, and testing datasets evaluated over fifty different attributes, with the dependent variable being the weekly fantasy points scored by the quarterback in a standard DraftKings league, *DK_points_scored*. More in-depth descriptions on these variables can be found in Appendix B.

A list in Table 3.1 shows all the variables that were considered. These are all of the variables that were considered in the research, with a brief description for each.

Variable Name	Description	Variable Name	Description
Name	Name of QB	height	Height of QB
weight	Weight of QB	years_pro	Years in NFL
combine40	Combine 40-yd Time	wonderlic	Combine Wonderlic Score
home_game	Home Game	FtsyAcesOwnPct	FantasyAces Ownership %
starter	Started Game	Top50WRs2015	ESPN Top 50 WRs Count
RB900yds2015	Team RB over 900 yds	ESPN_ADP	ESPN ADP
MFL_ADP	MyFantasyLeague ADP	TDs_2015	2015 Pass TDs
RedZnTDs2015	Red Zone Pass TDs	VBD_2015	Value-Based Drafting
CompPct_2015	Completion %	TDPct_2015	Touchdown %
QBR2015	QB Rating	ANY_A_2015	Adjusted Net Yds/Pass Att
Yards_2015	2015 Pass Yards	RushYds_2015	2015 Rush Yards
RushTDs2015	2015 Rush TDs	GmsStarted2015	Games Started
GmsPlayed2015	Games Played	2015TmPassPct	Team Pass %
SnapPct2015	2015 Team Snap %	RushPct2015	2015 Rush %
UtilPct2015	2015 Utility %	SackPct2015	2015 Sack %
PassAtt2015	2015 Pass Attempts	Ints2015	Interceptions Thrown
Sacks2015	2015 Sacks Taken	NCAAstarts	College Starts Total
NCAAcomp	College Comp. %	Pass_262760	Pass 26-27-60 Rule
NFLstartsPre16	Starts pre-2016	RushAtt2015	Rush Attempts
Fum2015	2015 Fumbles	NetPts2015	2015 Net Points
AvgPPG2015	Avg Pts per Game	AvgPPstart2015	Avg. Points per Start
DKpoints2015	2015DraftKings Points	Opp_Rank	Opponent Defense Rank
Opp_Pos_Rank	Opp. Against QB Rank	DK_salary	DraftKings Cost
DK_Proj	DraftKings Projection	FanDuel_Proj	FanDuel Projection
CBS_Proj	CBS Projection	FProsStd_Proj	FantasyPros Projection
FFNerd_Proj	FFNerd Projection	Fftoday_proj	FFToday Projection
PreviousWeekPts	Previous Week Points	Avg_Proj	Avg. Projections
DK_points_scored	DraftKings Pts Scored		

Table 3.1. Variables Considered in This Research

3.3 Scoring

Finally, it's important to understand how a quarterback in a DraftKings league accumulates points [18]. There are different scoring totals for different contest formats on DraftKings which vary from those on FanDuel, Yahoo, EPSN, etc. Table 3.2 shows how the dependent variable in our dataset, *DK_points_scored*, was calculated for each QB. Thus, a QB who threw for 325 yards, 2 TDs, and 1 interception while rushing

for 10 yards would be expected to score 23 points in a DraftKings league. In Fig. 3.1 a histogram of the *DK_points_scored* variable is shown. The data appears to be normally distributed and ready for use in our models. As far as relevant descriptive statistics go, the mean is 17.26 points and the median is 16.17 points. The quartiles for 0% (minimum), 25%, 50%, 75%, and 100% (maximum) are -1.00, 12.04, 16.17, 22.10, and 40.18 points, respectively.

Finally, we needed a metric to evaluate the accuracy of each model. Typically, regressions use R^2 to quantify the unexplained variance and accuracy. Not only can this metric be misleading, but it is not applicable to many of the models developed. Instead the accuracy was calculated by finding the root mean squared error (RMSE) and the mean absolute error (MAE) of the model on the validation dataset. This allowed for a comparison between models, regardless of technique.

Table 3.2. DraftKings Scoring Summary

Offensive Play	**Points Awarded**
Passing Touchdown (TD)	+4
25 Passing Yards	+1 (+0.04 pt/per yd)
300+ Yard Passing Game	+3
Interception	-1
10 Rushing Yards	+1 (+0.1 pt/per yd)
Rushing TD	+6
100+ Yard Rushing Game	+3
10 Receiving Yards	+1 (+0.1 pt/per yd)
Reception	+1
Receiving TD	+6
100+ Yard Receiving Game	+3
Fumble Lost	-1
2 Point Conversion (Pass, Run or Catch)	+2
Offensive Fumble Recovery TD	+6

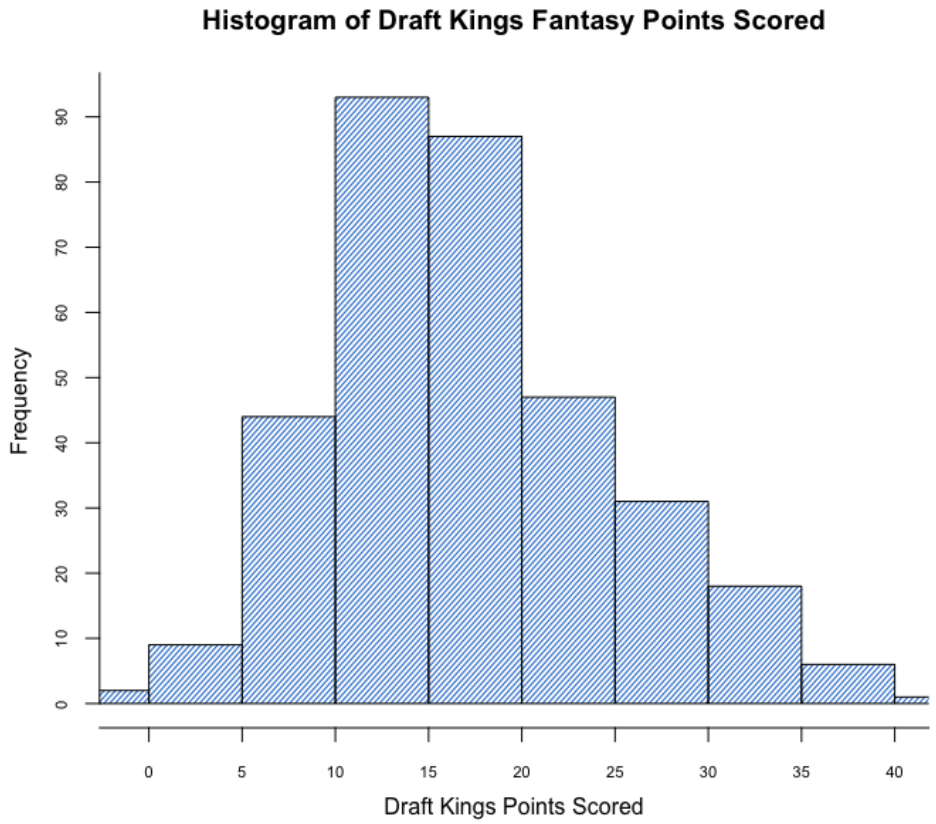

Figure 3.1. Histogram of the response variable DraftKings points scored.

CHAPTER 4

METHODS AND MODELS

4.1 Initial Work

To begin, each of the following models were tested on the full training dataset. Average to below average results and error metrics were found on all models. In an attempt to explain more of the variation in the models more variables were continually added to the training, validation, and testing datasets. Many different websites have accumulated or created their own variables to try and predict a player's fantasy point output. This is called *feature engineering*, and is a popular technique in data mining. Feature engineering involves taking information from the data and manually creating new variables/features that should better explain the unknown variation and improve the model. Several variables were feature engineered here, including ones like *ANY_A_2015*, adjusted net yards per passing attempt in 2015 and *Pass_262760*, a binary variable indicating if a player got a minimum of 26 on his NFL Combine Wonderlic test, made at least 27 starts at QB in college, and had at least a 60% completion rate in college. Much of the work in our research involved continually finding and creating new variables that would hopefully improve the accuracies of all the models tested.

As stated before, this research initially evaluated over 50 different features. Running the models on a dataset with so many variables was eventually found to be detrimental. In essence, the models were having a hard time finding the *signal* through all of the *noise*. In statistics the signal is the information of interest, while the noise is the randomness in our given data. By reducing the features and dimensions

of the data we eventually began to discover lower values for the error metrics RMSE and MAE.

It was found that the data suffered from a lot of multicollinearity. This meant that many of the predictor variables were highly correlated with each other. An example of this might be between variables that tracked statistics from the previous 2015 NFL season. We aggregated stats from 2015 such as passing yards, sacks, interceptions, touchdowns, utilization percentage, etc. Of course the more a QB was utilized, the more sacks he took and the more passing yards he threw for. And of course the more times a QB throws, the more interceptions and touchdowns they would be expected to throw. The correlations between a lot of our predictor variables was an issue. In order to reduce the number of variables in the dataset, and thus the noise and multicollinearity, an ANOVA (analysis of variance) test was done and variance inflation factors (VIF) were calculated. Once these studies were done our dataset was reduced from over 50 variables to 10. The models showed their greatest accuracies when evaluated on the training validation, and testing sets with the seven variables identified in Table 4.1.

Moving forward, these were the variables used in training, validation, and testing datasets. At this point it's interesting to note that the variables to predict output per week and per year are much different. Notice this dataset does not use much historical information for the quarterback, such as a previous season's touchdown or interception total, average draft position, or even total points scored last year. When projecting yearly totals for quarterbacks it can be shown that this historical information is important and necessary. For weekly projections, however, there is so much more volatility and variation, that a previous season's statistics have little to do with predicting a performance in a given week. From a bigger picture this would appear to make sense. In the past three seasons (2014-2016), our top QB, Aaron Rodgers

Table 4.1. Selected Variables Used in Models

Variable Name	Description
years_pro	Number of years a QB has been in the NFL
RedZnTDs2015	Number of TDs scored by QB in the red zone in 2015
RushYds_2015	A QB's rushing yards in 2015
Opp_Rank	Opponent's defensive rank that week
DK_salary	DraftKings' salary of player
CBS_Projected	CBS Sports' point projection
FantasyProsStd_Projected	FantasyPros' standard projection
FFNerd_Projected	Fantasy Football Nerd projection
PreviousWeekPts	Fantasy points scored in previous week
Average_Projected	Average projected points across all sources
DK_points_scored	DraftKings points scored amount (y-variable)

has thrown for 38, 31, and 40 touchdowns, with 5, 8, and 7 interceptions [19]. Another top QB, Drew Brees, has thrown 33, 32, and 37 touchdowns and 17, 11, and 15 interceptions [20]. Because of this consistency, it's much easier to use historical information and predict yearly results than it is to predict weekly. This explains why many of the variables collected for this research were found to be unnecessary for weekly projections. If we had been considering yearly projections our datasets would have looked much different. With the variables used in the models following, one can see that we are harnessing the power of other models and using them to our advantage, to create an even more precise model.

The following sections individually highlight each model and algorithm employed on our reduced dataset. Many are popular and well-known methods that are frequently used in the data science community. These models are grouped into sections for Regressions, Tree-based Methods, Artificial Neural Networks, and Principal Component Analysis. A summary will follow that will compare the individual models to each other and evaluate which ones performed the best.

4.2 Regressions

4.2.1 Backwards Stepwise Regression

First, a backwards stepwise regression was considered. This type of regression is based off of a multiple linear regression (MLR). A MLR model gives each predictor a separate slope coefficient in a single model. In general, suppose there are p distinct predictors; then the MLR model takes the form:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \epsilon, \tag{4.1}$$

where X_j represents the jth predictor and β_j quantifies the association between that variable and the response. We interpret β_j as the average effect on Y of a one unit increase in X_j, holding all other predictors fixed [21].

In MLR the regression coefficients $\beta_0, \beta_1, ..., \beta_p$ in (3.1) are unknown, and must be estimated. Given estimates $\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_p$, we can make predictions using the formula

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + ... + \hat{\beta}_p X_p. \tag{4.2}$$

The parameters are estimated using the same least squares approach used in simple linear regression. $\beta_0, \beta_1, ..., \beta_p$ are chosen to minimize the sum of the squared residuals. The values $\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_p$ that minimize the sum of squared residuals are the multiple least squares regression coefficient estimates [21]

A backwards stepwise regression is a semi-automated process of building a model by successively removing variables based solely on the t-statistics of their estimated coefficients. This type of regression is especially useful for sifting through large numbers of potential independent variables and/or fine-tuning a model by poking variables in or out [22]. In our case it was used to reduce the number of independent variables we were evaluating. Instead of running a MLR on all the variables in the dataset, we can instead elect this method to reduce the number of variables and come out with simplified, more accurate result.

To understand how it works, consider that we have a large number of independent variables and want to extract the best subset of them for use in our predictive model. The backwards stepwise option lets you begin the process with all of the variables currently in the model and proceeds backwards, removing one variable at a time (forward stepwise adds one variable at time). At each step the R software calculates which variables are currently in the model and computes the t-statistic for its estimated coefficient, squares it, and reports it as the 'F-to-remove' statistic. For each variable not in the model, R computes the t-statistic that its coefficient would have if it were the next variable, squares it, and reports it as the 'F-to-enter' statistic. At the next step, the program automatically enters the variable with the highest F-to-enter statistic, or removes the variable with the lowest F-to-remove statistic. Under the backward method, the process begins with all the variables in the model and successively removes the variable with the smallest F-to-remove statistic. The process stops when no variables in or out of the model have F-statistics on the wrong side of the threshold [22].

A summary of the output is shown in Table 4.2. Four of the 10 variables were chosen. The coefficient value is shown for each. Our research determined the linear equation to predict the dependent variable to be:

$$DK\_points\_scored = (-4.20) - (0.004)RushYds\_2015 + (0.002)DK\_salary - (0.144)PreviousWeekPts + (0.733)Average\_Projected$$

So a quarterback with a salary of \$9500 (a pretty good QB), who scored 30 points the week before, ran for 274 yards in 2015, and was projected for 23 points, would be expected to score [-4.20 - (0.004*274) + (0.002*9500) - (0.144*30)

Table 4.2. Backwards Stepwise Regression Summary

	Dependent variable:
	DK_points_scored
(Intercept)	−4.203124
RushYds_2015	−0.0042585
DK_salary	0.0020115
PreviousWeekPts	−0.14402808
Average_Projected	0.73299265
Observations	212
Note:	$p < 0.1$

+(0.733*23)] = 26.17 points. This appears to make sense as we would expect the point total to fall a bit from 30 back down towards the mean of 17.26 points.

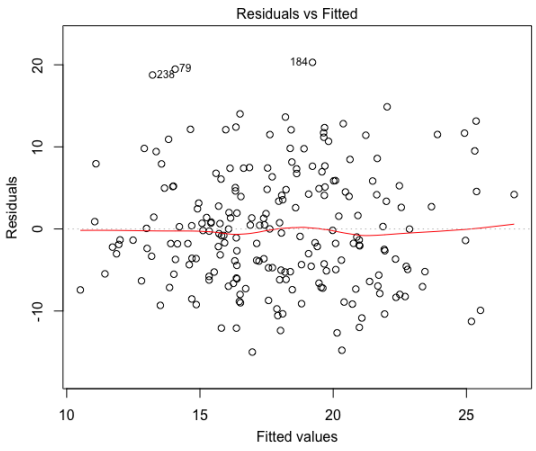

Figure 4.1. Residuals vs. fitted values.

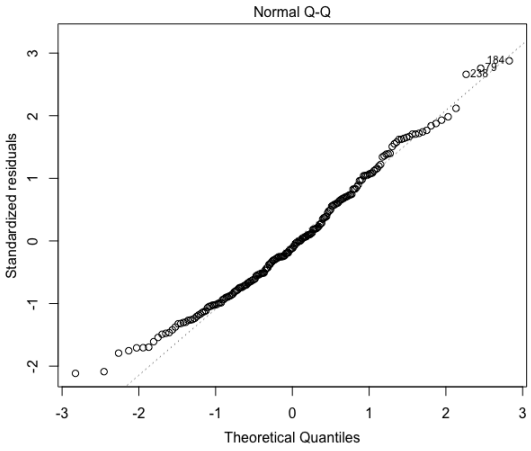

Figure 4.2. Normal probability plot.

A plot of the residuals in Fig. 4.1 from the backwards stepwise regression shows that variance is constant. A normal probability plot (Q-Q plot) of the residuals in Fig. 4.2 shows that, although there is still some departure from normality, it is slightly less pronounced.

4.2.2 Support Vector Regression

Finally, a more unique form of regression, support vector regression, (SVR) was studied. SVR is based off of the same principles as the popular classification algorithm support vector classification (SVC), and an extension of the classifier, the support vector machine (SVM). The SVM is a generalization of a simple classifier called the maximal margin classifier and is meant for binary classification in which there are two classes, making it perfect for our use. It is based on the idea of using a hyperplane to separate the two classes of data. The maximal margin classifier, or *optimal separating hyperplane*, is the separating hyperplane that is the farthest from the training observations. That is, we can compute the perpendicular distance from each training observation to a given separating hyperplane; the smallest such distance is the minimal distance from the observations to the hyperplane, and is known as the margin. The maximal margin hyperplane is the separating hyperplane for which the margin is the largest - that is, we select the hyperplane that has the farthest minimum distance to the training observations [21]. Once this is established we can take samples from the validation data and classify them based on which side of the maximal margin hyperplane the sample lies.

The SVC does *not* perfectly separate the two classes, in the interest of giving greater robustness to individual observations, and better classification of *most* of the training observations. Traditionally, a classifier based on a separating hyperplane (like a Linear Classification or the Perceptron Algorithm) can have sensitivity to

individual observations, and may overfit the training data. SVC works to avoid this by using a hyperplane that is not a perfect separator. This means it will have some observations on the wrong side of the margin or hyperplane, but overall will do a better job of classifying the other remaining observations. A simplified example of this is shown in Fig. 10 [23].

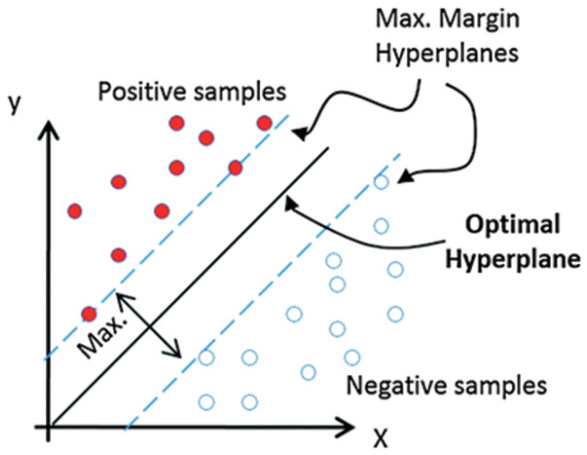

Figure 4.3. Sample of the optimal hyperplane the SVC seeks to find.

In our case, support vector regression is based off of the same principles as the SVC, with a few small differences. Since our research doesn't focus on predicting a class, but instead on outputting a continuous value, it is hard to predict the information at hand, which has infinite possibilities. Therefore, it allows for an error term, ϵ, which forms boundaries of the regression line [24]. Any errors greater than the epsilon threshold are penalized, which improves the SVR model's accuracy. SVR also tries to reduce model complexity. The main ideas between the SVM and the SVR are the same, however: to minimize the error and individualize the hyperplane that maximizes the margin [21].

Fig. 4.4 shows the idea of the SVR. The regression line is in the middle, bounded by the epsilon lines, with the acceptable data points inside. The algorithm performs the necessary separation [25]. The support vectors are the points that are found on the boundary lines. Using the `e1071` package in R and its associated `svm()` function, we were able to run the SVR and fine tune the ϵ value that produced the smallest root mean squared error (RMSE) and mean absolute error (MAE) values.

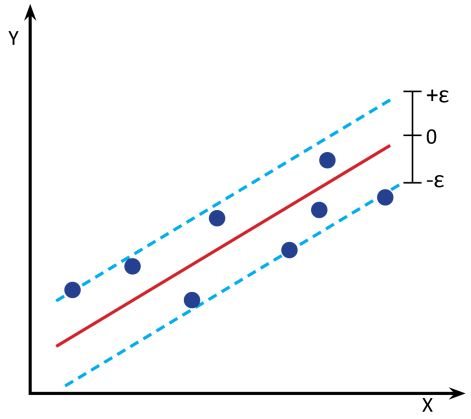

Figure 4.4. SVR shown with margin of tolerance (epsilon).

4.2.3 Regression Summary

After running these two regressions we were able to compare their results to one another. With the lowest values for RMSE and MAE, it was clear here that the support vector regression performed the best.

Table 4.3. Regression Models Summary

Statistical Model	**RMSE**	**MAE**
Stepwise Backward Regression	7.46	5.80
Support Vector Regression	7.30	5.60

4.3 Tree-Based Methods

4.3.1 Regression Trees

Decision trees are frequently used in data mining to create a model that predicts a continuous or categorical variable based on the values of numerous independent variables. The most popular way to do this is through the use of the CART (Classification and Regression Trees) decision tree methodology. The CART methodology was introduced in 1984 by Leo Breiman, Jerome Friedman, Richard Olshen and Charles Stone as an umbrella term to refer to classification and regression trees [26]. In the case of classification trees, the target variable is categorical and the CART algorithm seeks to identify the 'class' that the target variable would fall into. Since the target variable in this research is continuous a regression tree is employed. Tree-based methods such as this are rather simple and are often easily interpreted, which make them popular methods when communicating information to an audience.

Regression trees are different from a classical approach such as the linear regressions previously mentioned. However, neither model is necessarily better than the other - it depends on the problem at hand. If the relationship between the features and the response is well approximated by linear model, then a linear regression will likely produce better results. But if there is a highly non-linear and complex relationship between the features and response variable then decision trees may show lower error metrics than a classical regression. There are several different types of tree-based methods that were explored for this research.

The CART algorithm is structured as a sequence of questions, the answers to which determine what the next question, if any, should be. The result of these questions is a tree-like structure where the ends are terminal nodes, at which point there are no more questions [26]. At each node in the tree, we apply a test to one

of the inputs, say X_i. Depending on the outcome of the test, we go to either the left or the right sub-branch of the tree. Eventually we come to a leaf node, where a prediction is made. This prediction aggregates or averages all the training data points which reach that leaf.

When data, such as ours, has lots of features which might interact in complicated, nonlinear ways, we can sub-divide, or partition, the space into smaller regions, where the interactions are more manageable. We then partition the sub-divisions again - this is *recursive partitioning*, as in hierarchical clustering - until finally we get to chunks of the space which are so tame that we can fit simple models to them. To read a regression tree one must begin at the root node of the tree and ask a series of questions about the features. The interior nodes are labeled with questions, and the edges or branches between them labeled by the answers [5]. A plot of the regression tree developed in our research is shown in Fig. 4.5

There are essentially two steps to building a regression tree.

1. We divide the predictor space - that is, the set of possible values for $X_1, X_2, ..., X_p$ - into J distinct and non-overlapping regions, $R_1, R_2, ..., R_J$.
2. For every observation that falls into the region R_j, we make the same prediction, which is simply the mean of the response values for the training observations in R_j.

The goal is to find regions $R_1, ..., R_j$ that minimizes the RSS (residual sum of squares), given by:

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{Rj})^2, \tag{4.3}$$

where $\hat{y}_{Rj}$ is the mean response for the training observations within the jth box. At each splitting point all of the predictors are considered as well as all possible values for the cutpoint for each of the predictors. The chosen predictor and associated cutpoint

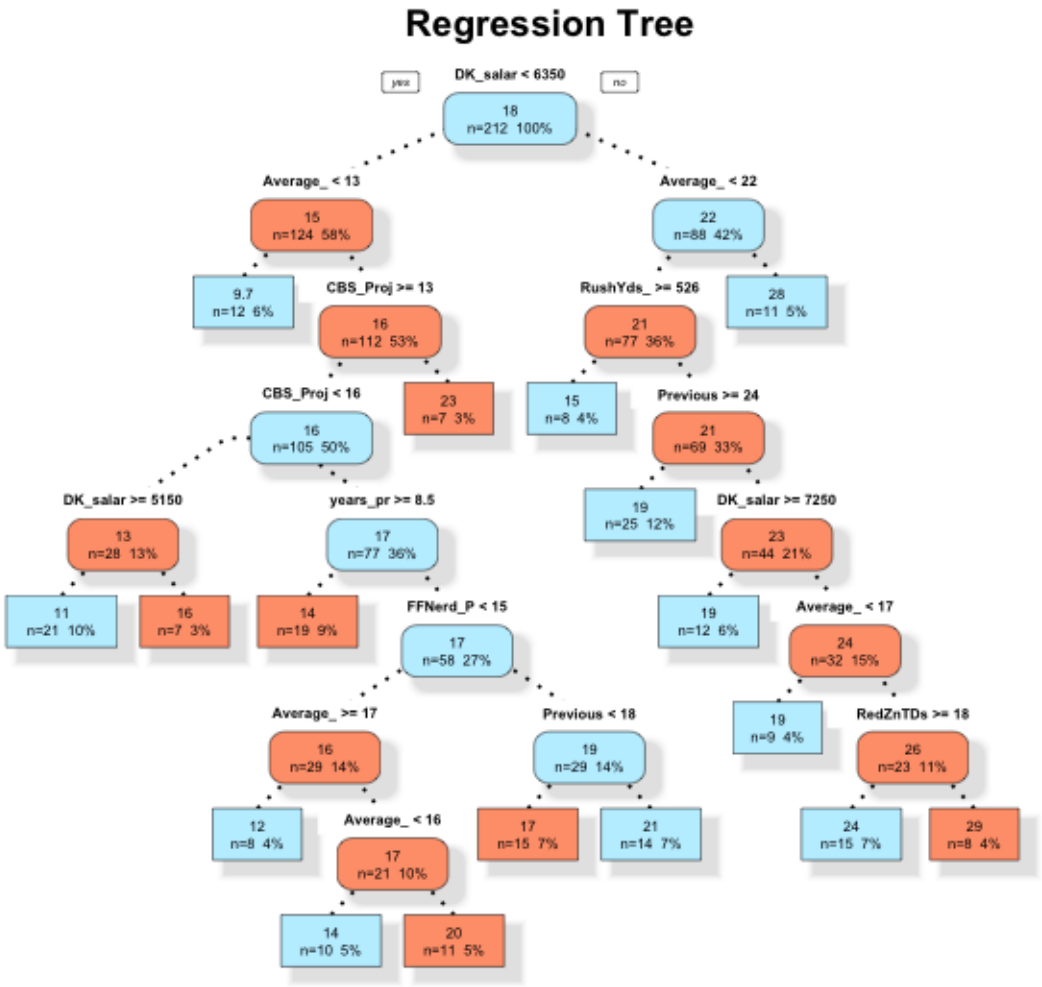

Figure 4.5. Regression tree with *DK_salary* as the root node. Answering questions moves one through the branches to find an answer in a terminal node.

are selected such that the resulting tree has the lowest RSS. The process is repeated at each split; instead of splitting the entire predictor space each time, however, we split one of the two previously identified regions. This process continues, with the smallest RSS value being the selection criteria. Once the regions $R_1, ..., R_j$ have been created, we predict the response for a given validation observation using the mean

of the training observations in the region to which the validtion observation belongs [21].

As stated before, regression trees have numerous advantages, such as their interpretability and their ease of explanation, but generally do not offer the level of accuracy that most other regression approaches can offer. Aggregating many regression trees, however, through methods like *random forests* and *boosting* can produce much more accurate models. These two methods were applied to our dataset with much better results than would have been expected with a single regression tree.

4.3.2 Random Forests

Random forests are an ensemble of different regression trees and are commonly used for nonlinear multiple regression. This method constructs hundreds or thousands of multiple singular regression trees and then outputs the mean prediction of the individual trees. This method is much more powerful than a basic regression tree.

The model is fit to the target variable using all of the independent variables. For each independent variable the data is split at various points. At each point the sum of squared error (SSE) is calculated between the predicted value and the actual value. Then the variable resulting in the minimum SSE is selected as a node to split on [27]. This process is continued until the entire dataset is covered.

In the context of regression trees, bootstrap aggregation, or bagging, is frequently used. Bagging is a general procedure for reducing the variance of a statistical learning method. The key to bagging is that trees are repeatedly fit to 'bootstrapped' subsets of observations. Here we bootstrap by taking repeated, random samples from the training dataset. Therefore, a number of different bootstrapped training datasets are generated. In the context of regression trees this means that B regression trees are constructed using B bootstrapped training sets. Then we average out the resulting

predictions, which reduces the variance [21]. In our work 500 trees are combined into this single procedure.

Bagging was used in our work with random forests. When building decision trees for random forests, a random sample of m predictors are chosen as split candidates from the full set of p predictors. Typically m is only 1/3 of the predictor variables, p. The main difference between bagging and a standard random forests model is the choice of predictor subset size m. If $m = p$ then this simply amounts to bagging. After running the random forests method on our dataset we were able to output some variable importance plots and calculate the RMSE and MAE metrics. Fig. 4.6 shows several variables in the model and two different measures of variable importance. The first is based on the mean decrease of accuracy in predictions when a given variable is excluded from the model. The second measures the total decrease in node impurity that results from splits over that variable. These results indicate that the two most important variables are the Average Projected points from all sources considered and FantasyPros' projected points. It's interesting to note that the previous week's points scored was not considered very important. One might expect that if a player had put up several consistent performances in a row that this would be a good indicator of future success, but as our earlier example of Aaron Rodgers' weeks 15 and 16 show, this is not necessarily the case.

4.3.3 Boosting

Another tree-based method that was tested was *boosting* - an additional approach for predictions resulting from a decision tree. Like bagging, boosting is a general approach that can be applied to many different methods for regression. Bagging involved creating multiple copies of the original dataset using the bootstrap, fitting a separate decision tree to each copy, and then combining all of the trees in

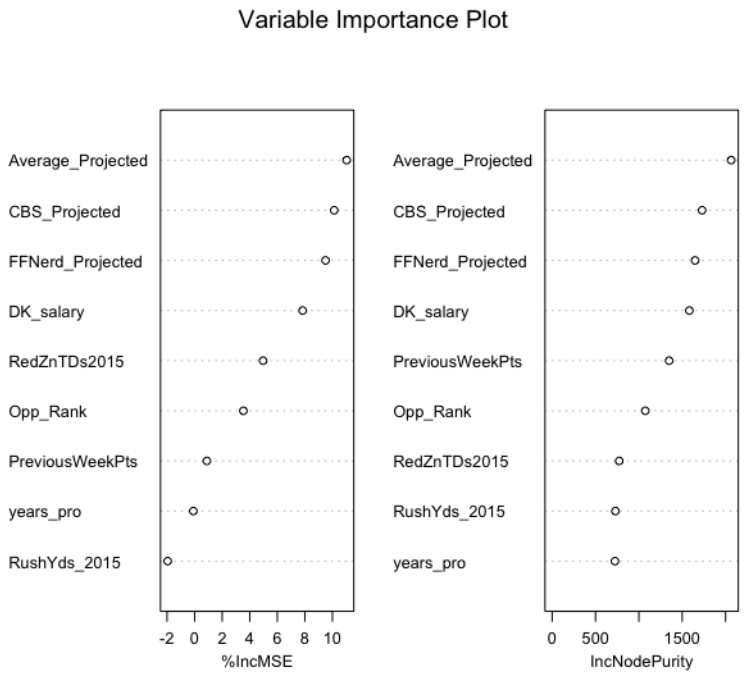

Figure 4.6. Importance of individual variables in the random forest.

order to create a single predictive model. Each tree was independent and grown on a bootstrap dataset. Boosting works similarly, but the trees are grown sequentially, using information from previously grown trees. Boosting also doesn't involve sampling like bagging does; each tree is fit on a modified version of the original dataset. The idea behind boosting is to fit decision trees to the residuals from the model, rather than the outcome variable of *DK_points_scored*. Each new decision tree is added into the fitted function in order to update the residuals from the model. Since we are addressing the residuals, the model slowly improves in the areas where it does not perform well. This algorithm is considered a *slow learner* in the data science world, as it gradually improves the model offering small improvements in the residuals. Typically slow learning models models perform well.

Boosting and bagging also differ in that the construction of each tree in boosting depends strongly on the trees that have already been grown [21]. In R the boosting algorithm is run with the `gbm` package, which allows us the option to set parameters for the distribution and the number of trees to sequentially grow. In our research we found the best results growing 5000 trees.

4.4 Artificial Neural Network

Artificial neural networks (ANN) and the math behind them can be extremely complex. Essentially, neural networks are processing algorithms that are loosely modeled after the structure of the human brain, composed of layers and interconnected nodes, which contain an 'activation function.' They are best at identifying patterns or trends in data and thus are well suited for predictions. An artificial neuron is a device with many inputs and one output. Patterns are presented to the network via the 'input layer,' which communicates with one or more 'hidden layers,' where the actual processing is done via a system of weighted connections. The hidden layers then link to an 'output layer' where the answer is displayed [28]. An example of a neuron is shown in Fig. 4.7 [29].

A multilayer perceptron (MLP) network is a feedforward artificial neural network that was used in our research. As implied by its name, a feedforward network does not form a cycle. It is one of the simpler networks, in which information only moves in one direction - forward - from the input layer, to the hidden layers, and finally to the output layer. This method learns through *backpropagation*. This means that the connection weights are changed after each piece of data is processed, based on the amount of error in the output compared to the expected result. This is an example of *supervised learning* [30]. Each node is a neuron with a nonlinear activation function [31]. The activation function in a MLP uses a nonlinear function that maps

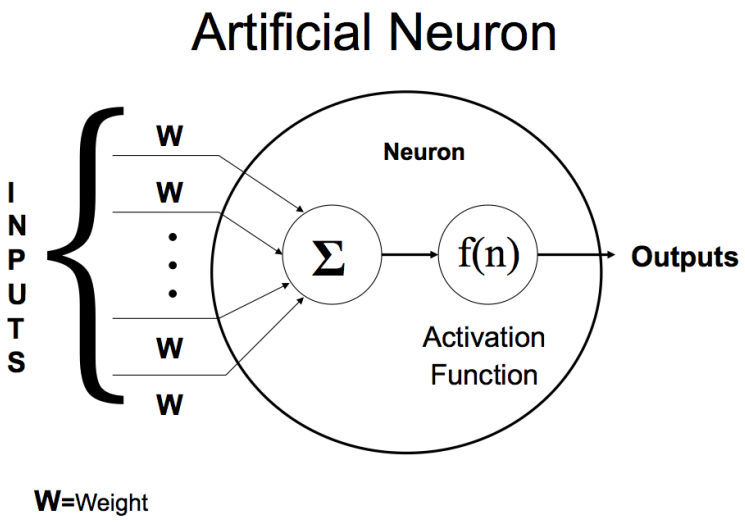

Figure 4.7. Diagram of an artificial neuron within a neural network.

the weighted inputs to the output of each neuron. The two main activation functions that are currently used in applications are both sigmoids (a function having an 'S' shaped curve) are described with:

$$y(v_i) = \tanh(v_i), \text{ and}$$

$$y(v_i) = (1 + e^{-v_i})^{-1}$$

Here y_i is the output of the ith node (neuron) and v_i is the weighted sum of the input synapses [32]. An example of a standard MLP neural network is seen in Fig. 4.8.

In our research we used a neural network with two hidden layers with a configuration of 6:3:2:1. This means that the input layer had six inputs, the two hidden layers had three and two neurons each, and the output layer represented a single value, the *DK_points_scored* variable. This combination was found through trial-and-error, testing various configurations, and determining the best number of hidden layers. For most applications one or two hidden layers is enough. As far as the number of neurons to use, it should be between the input layer size (six) and the output layer size (one), usually 2/3 of the input size [33]. This would translate to using four or five neurons

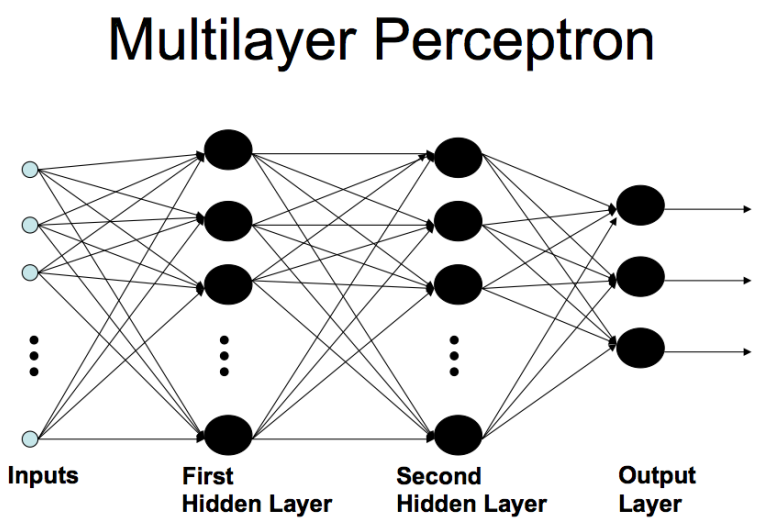

Figure 4.8. Diagram of a multilayer perceptron neural network.

for our model. Obviously five total were chosen here as five neurons performed better than four in our work.

Fig. 4.9 shows the graphical representation of our neural network with the weights on each connection. The black lines show the connections between each layer and the weights on each connection. The blue lines show the bias term added in each step. The bias can be thought of as the intercept of a linear model.

The net is basically a black box, however, yielding us little information on fitting, the weights, and model, so it is hard to explain their outcome as one might be able to with a simpler linear model like a multiple linear regression. After running the model and predicting the values of the validation set like we did for previous models we find the error metrics similar to our earlier stepwise backwards regression. We can compare the two models' outputs in a single plot in Figure 4.10. The regression line is shown in the middle, with the predictions for each model concentrated loosely around the line.

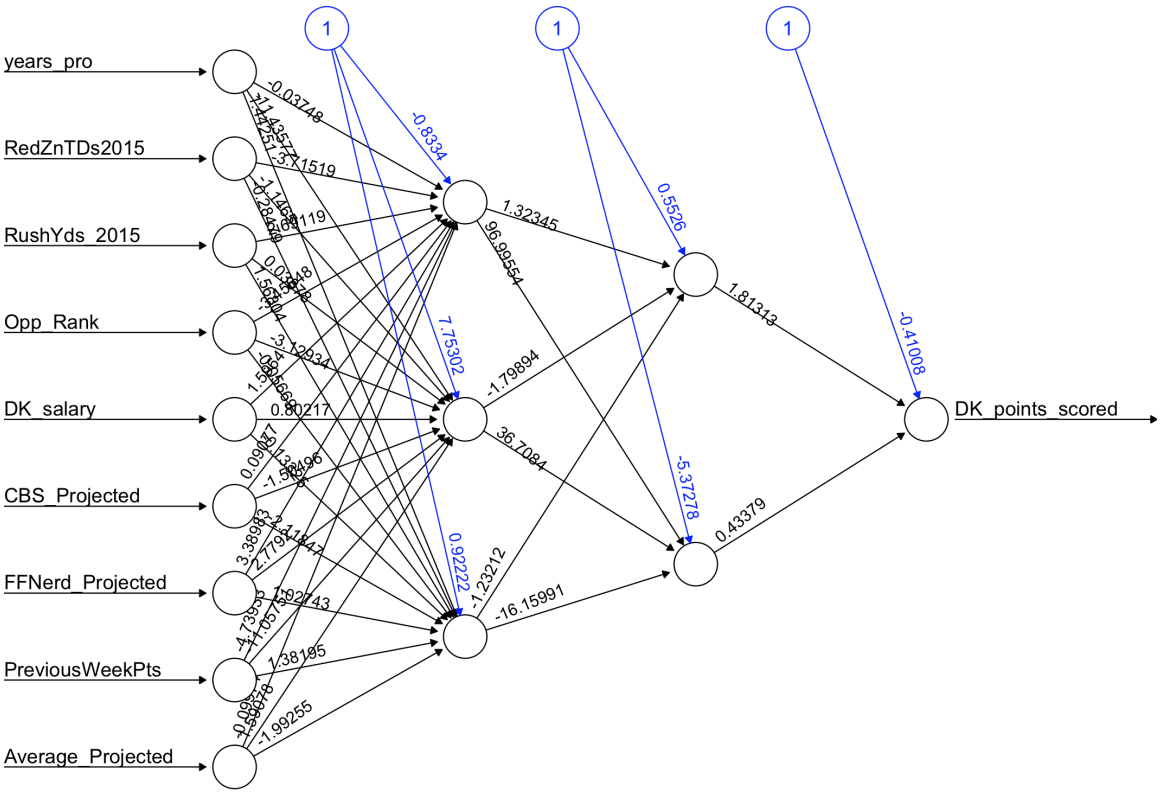

Figure 4.9. Graphical representation of the 6:3:2:1 neural network.

4.5 Principal Component Analysis

The final method that was evaluated was a Principal Component Analysis (PCA), or a Principal Components Regression (PCR). A PCA is often used to obtain a low-dimensional set of features from a large number of variables. A PCA models the variation in a set of variables in terms of a smaller number of independent linear combinations (principal components) of those variables [34]. The analysis refers to the process by which principal components are computed, and the subsequent use of these components in understanding the data. When faced with a large set of correlated variables, as our data reflects, principal components allow us to summarize the set with a smaller number of representative variables that collectively explain most of

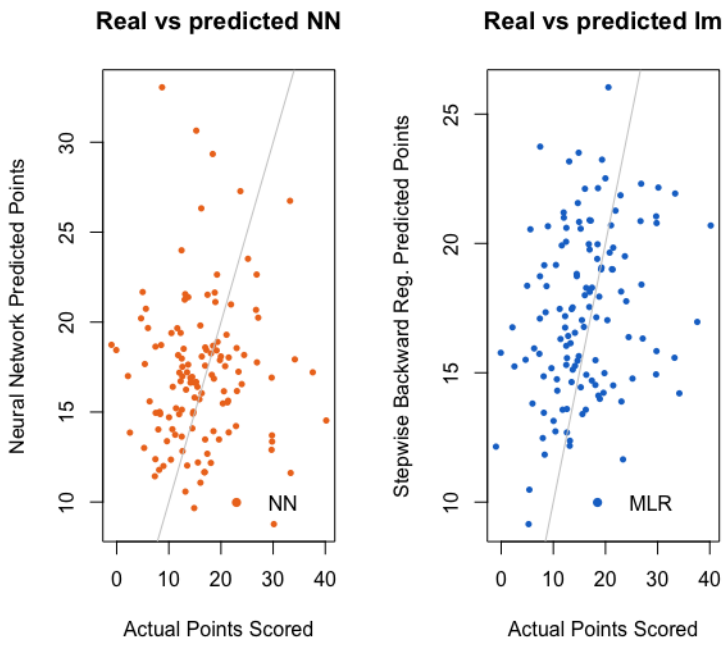

Figure 4.10. Neural network output compared to the backwards stepwise regression.

the variability in the original set. PCA is considered an 'unsupervised' approach, since it involves only a set of features $X_1, X_2, ..., X_p$, and no associated response variable Y [21].

The principal components regression (PCR) approach involves constructing the first M principal components, $Z_1, ..., Z_M$, and then taking these components and using them as the predictors in a linear regression model that is fit using the least squares method (least squares regression is a linear fit of a regression line that has the smallest possible value for the sum of the squares of the residuals). The key idea is that often a small number of principal components suffices to explain most of the variability in the data, as well as the relationship with the response. In other words, we assume that the directions in which $X_1, ..., X_p$ show the most variation are the directions that are associated with Y. This assumption is not always guaranteed, but is reasonable

enough to provide good results. If we are to assume that this is true, then fitting a least squares model to $Z_1, ..., Z_M$ will lead to better results than fitting a least squares model to all our predictors $(X_1, ..., X_p)$, since nearly all of the information contained in the predictors is already present in the principal components [21]. If you were to use $M = p$, in which the number of principal components were equal to the number of predictors, then you would simply be performing a least squares regression. Thus one can begin to see how the PCA/PCR method makes effective use of reducing the dimensionality of the data. Reducing the number of predictors and principal components ended up improving our accuracy metrics. During the running of the PCR method on our dataset it was shown that the optimal number of principal components to use was six. The left side of Fig. 4.11 shows that the first principal component explains the largest proportion of variance, while the right side shows that we can explain about 95% of the variance using six principal components.

Once all of the principal components have been computed, they can be plotted against each other in order to produce low-dimensional views of the data as shown in Fig. 4.12. It is important to note, however, that while the PCA/PCR method was one of the most accurate methods we employed, it is a dimensionality reduction method and *not* a feature selection method such as a random forest or standard multiple linear regression. This is because each of the M principal components used in the regression is a linear combination of all p of the *original features* [21].

Much like the artificial neural network earlier, it is important to perform the PCR method only after normalizing or standardizing each variable. This ensures that all variables are on the same scale. The PCR was run on our dataset, with the ideal number of principal components shown to be six. Fig. 4.12 represents the principal component scores for the first two components and the loading vectors in a single 'biplot' display.

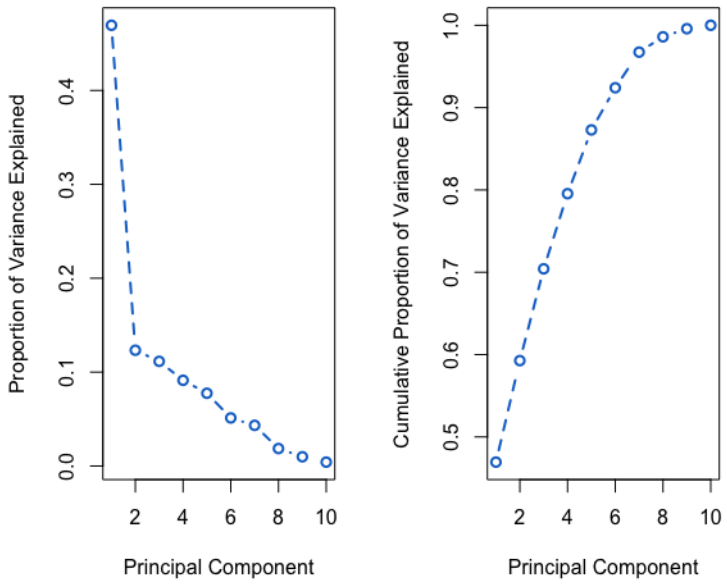

Figure 4.11. With six principal components about 95% of the cumulative proportion of variance is explained.

Fig. 4.12 shows that the first loading vector places approximately equal weight on the variables for several different websites' projections and a QB's average projected points, a little less weight on a quarterback's salary on DraftKings, and much less weight on previous week points, red zone TDs, and years played in the NFL. This makes sense when considering Fig. 4.6 from the Random Forest. That model also determined that variables like points from the previous week and years pro were poor predictors of current week output. Variables that are found close together in this plot indicate that they are somewhat correlated with each other. As with the random forest method, the accuracy of the model's output was measured by computing the RMSE and the MAE.

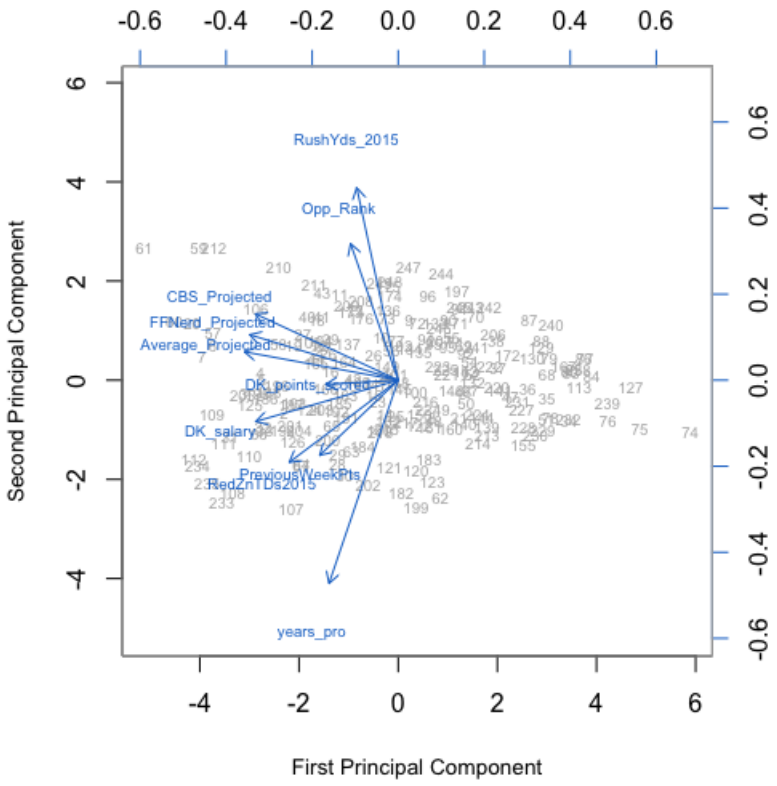

Figure 4.12. Depicting the first two principal components of the QB data.

CHAPTER 5

RESULTS

The table shown below gives a summary and comparison of the results for each model tested. The Principal Component Regression is far better than the rest of the competition, with lower root mean squared error and mean absolute error values than any other method.

Table 5.1. Final Rankings of All Models Investigated

Rank	**Statistical Model**	**RMSE**	**MAE**
1	Principal Component Regression	4.36	3.51
2	Support Vector Regression	7.30	5.60
3	Stepwise Backward Regression	7.46	5.80
4	Boosting	7.59	6.01
5	Random Forests	7.68	6.06
6	Baseline/Best Guess	7.80	6.19
7	Regression Tree	8.53	6.57
8	Artificial Neural Network	8.77	6.67

It is also important to note, though, that most all of the models investigated here gave better results than the 'best guess' model that was based strictly on the average value of the response variable DK_points_scored. This would suggest that there *is* skill involved with setting a successful roster in DFS fantasy football.

The results of the principal components regression were also compared to a leading popular sports website, CBS Sports, which offers fantasy football projections. Fig. 5.1 shows that our model applied to the testing dataset is already more accurate than CBS Sports', which was found to have a RMSE and MAE of 7.32 and 5.80,

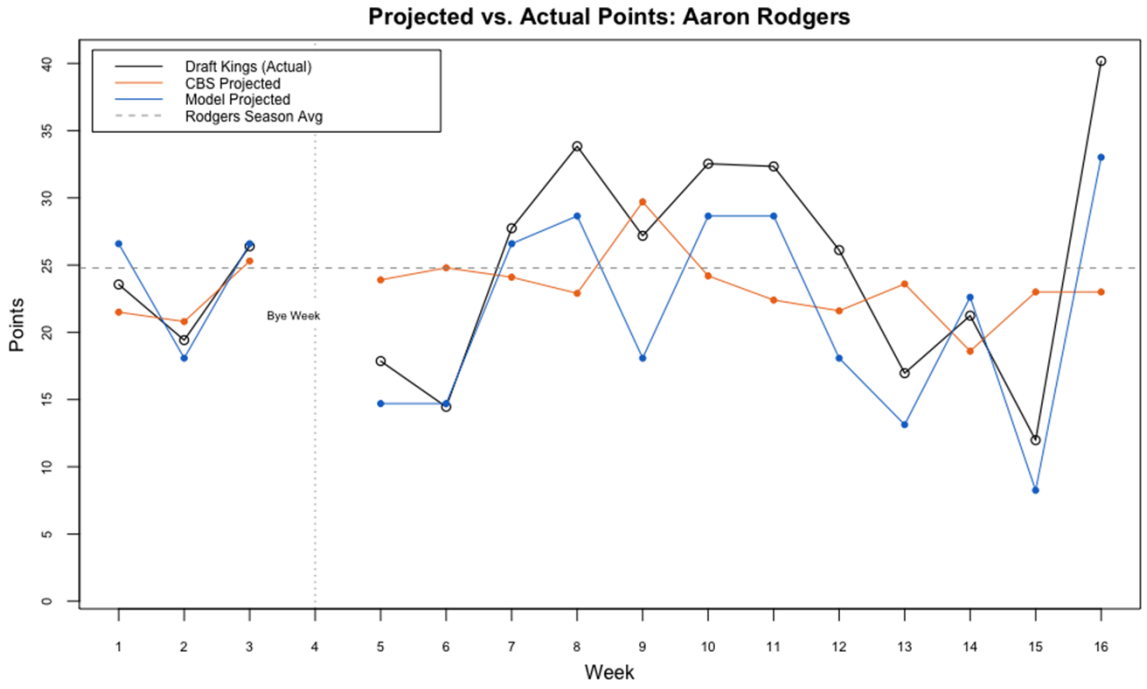

Figure 5.1. Aaron Rodger's actual DraftKings points compared to projections from CBS Sports and the PCR model developed in this study.

respectively. This is important because it reflects that our model has the potential to output more effective results than other mainstream offerings. The popularity and reach of websites like CBS Sports and ESPN is undeniable. There's no doubt that many of the participants in DFS competitions consider their projections when drafting a team. If our model is able to output more accurate results than sites like these, then we are already getting a leg up on the competition.

CHAPTER 6

CONCLUSION

6.1 Moving Forward

It is encouraging to see favorable results with a variety of different algorithms. The next step would be to try to fine-tune each of these models more in an attempt to further reduce the RMSE and MAE metrics and explain more of the unknown variation. Exploring the effect of injuries and how to handle them in the data might also yield better results. Some quarterbacks' point projections were found to be way off because they left the game early with injuries. Other interactions that could be further researched and quantified might be the effect certain stadiums or teams have had on a quarterback in the past, or how strong of a role a coach, offensive coordinator, or opposing defensive coordinator and his typical scheme appears to play in point output. Continuously investigating or engineering new variables and features that could explain more variation is essential as well. Of course, like any model, a larger dataset only provides more accurate insights, so another year's worth of data would be enormously helpful.

It would also be interesting to see if the principal components regression method worked 8as well at projecting other fantasy positions, such as running back and wide receiver. Oftentimes these positions can have even higher variability than the quarterback position, since they handle the ball a lot less and can be more impacted by specific personnel on the defense. For example, a receiver that is matched up against an elite cornerback or a running back facing a stout defensive line will often find opportunities to perform, and consequentially points, harder to come by. In the short

term, however, the most impactful improvements that can be made in this research moving forward can be summed up in two words: optimization and automation.

Once accurate and highly-effective models are identified to project individual positions on a fantasy team this research will focus more on the optimization problem of expected point output versus costs. Currently, the process of data collection and transformation, running models, and interpreting results into useful knowledge is all done manually. With more work, and an improvement in programming skills, I will be able to automate this entire process in Python. This will allow me to write a web-scraping script to automatically pull the necessary data from the right websites as they're updated in real-time. Once that occurs the data can automatically be read into the appropriate statistical model and the results incorporated with a lineup optimizer that factors in an athlete's cost and the DraftKings salary cap. The entire process could all be done with just a few clicks and could result in numerous 'optimal' lineups that can be fed back into DraftKings contests. This would allow for diverse lineups that are all projected to do well, and is similar to what the top participants are doing who are winning big.

6.2 Implications

This research could also help in the prediction of other sports - fantasy or live, which could be a huge financial windfall for some. Historically, most sports in general have been hard to predict because of the high variability and number of factors that play a role in every single competitor, team, play, game, match, series, etc. The statistical models presented here might be able to offer improved predictions in the outcomes of sports beyond football. From my personal standpoint, or perhaps the reader's, the methods shown here might also have introduced some algorithms that were unfamiliar previously. Some of the regression techniques presented here such as

the support vector regression, Yeo-Johnson transformation, and principal components regression, can be applied to a multitude of other problems and might help answer questions or provide accurate solutions in other fields.

Within the realm of fantasy football, though, this research can highlight statistical models and methods that may prove to be very effective for fantasy projections now and in the future. And to address the initial question, this research would appear to support the idea that daily fantasy football *is* in fact a game of skill, and not luck. With a well-developed, robust statistical model, a player would seem to have an advantage over an individual using their best guess, or projections off of a popular free sports website, as highlighted in the research. State governments and policy-makers would seem to agree as well - only 10 states remain where all formats of DFS are blocked [9]. During the course of this research several states determined fantasy football was a game of skill, most notably New York, where a public debate and legal battle raged on for quite some time, since both FanDuel and DraftKings are based in New York City. One would assume that in the near future DFS will be totally legalized across the country, and not considered online gambling.

APPENDIX A

R CODE

APPENDIX B

FULL VARIABLE DESCRIPTIONS

In this appendix, a more in-depth description of the variables are given.

B.1 Full Variable Names and Descriptions

1. Name: Name of quarterback
2. height: Height of quarterback
3. weight: Weight of quarterback
4. years_pro: Number of years played in the NFL
5. combine40: NFL Combine 40-yard dash time
6. wonderlic: NFL Combine Wonderlic test score
7. home_game: Home game (1=yes, 0 = no)
8. FntsyAcesOwnPct: FantasyAces.com ownership percentage of athlete
9. starter: Was QB a starter in game (1=yes, 0=no)
10. Top50espnWRs2015: Number of ESPN Top 50 WRs in previous year
11. RB_over900yds2015: Did team have a RB over 900 yards in previous year (1=yes, 0=no)
12. ESPN_ADP: ESPN Fantasy Football average draft position
13. MFL_ADP: MyFantasyLeague.com average draft position
14. TDs_2015: Passing TDs in 2015
15. RedZnTDs2015: Number of passing TDs in red zone in 2015
16. VBD_2015: Value-Based Drafting (player value in 2015)
17. CompPct_2015: Completion percentage in 2015
18. TDPct_2015: Touchdown percentage in 2015
19. QBR2015: Quarterback rating from 2015
20. ANY_A_2015: Adjusted net yards per passing attempt
21. Yards_2015: Passing yards in 2015
22. RushYds_2015: Rushing yards in 2015

23. RushTDs2015: Rushing touchdowns in 2015
24. GamesStarted2015: Number of games started at QB in 2015
25. GamesPlayed2015: Number of appearances at QB in 2015
26. X2015_TeamPassPct: Team's passing percentage in 2015
27. SnapPct2015: Percentage of team's snaps the QB took in 2015
28. RushPct2015: Percentage of plays the QB had a rushing attempt in 2015
29. UtilPct2015: Percentage of plays that the QB either ran or passed on in 2015
30. SackPct2015: Percentage of plays a QB was sacked on in 2015
31. PassAtt2015: Number of passing attempts in 2015
32. Ints2015: Number of interceptions thrown in 2015
33. Sacks2015: Number of sacks taken in 2015
34. NCAAstarts: Number of starts at QB in college
35. NCAAcomp: College completion percentage
36. Pass_262760: At least 26 college starts, a 27 on the Wonderlic test, and a 60% NCAA completion rate (1=yes, 0=no)
37. NFL_startsPre2016: Number of NFL starts before the 2016 season
38. RushAtt2015: Rushing attempts in 2015
39. Fum2015: Number of fumbles in 2015
40. NetPts2015: Net points in 2015 (penalized for turnovers)
41. AvgPPG2015: Average points scored based on games played in 2015
42. AvgPPStart2015: Average points scored based games started in 2015
43. TotalDKpoints2015: Total DraftKings points scored in 2015
44. Opp_Rank: Defensive rank of opponent
45. Opp_Pos_Rank: Defensive rank of opponent against QB position
46. DK_salary: Salary of DraftKings player before game
47. DK_Projected: DraftKings point projection

48. FanDuel_Projected: FanDuel point projection
49. CBS_Projected: CBSsports.com point projection
50. FantasyProsStd_Projected: FantasyPros.com point projection
51. FFNerd_Projected: FantasyFootballNerd.com point projection
52. Fftoday_proj: Fantasy Football Today point projection
53. PreviousWeekPts: Fantasy points scored in previous week
54. Average_Projected: Average projected points based on projections across all sources
55. DK_points_scored: Actual DraftKings points scored by a QB that week. (Y-variable)

REFERENCES

[1] W. Hobson, "Daily fantasy sites draftkings, fanduel reach agreement to merge," *The Washington Post*, November 2017.

[2] J. Lindsey. (2016) The nfl is finally tapping into the power of data. [Online]. Available: https://www.wired.com/2016/01/the-nfls-impending-data-revolution/

[3] S. Weisberg, "Yeo-johnson power transformations," October 2001.

[4] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer, 1995.

[5] C. Shalizi. Lecture 10: Regression trees.

[6] D. Benyamin. (2017, January) A gentle introduction to random forests, ensembles, and performance metrics in a commercial system. [Online]. Available: https://citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics/

[7] T. Hastie. Trees, bagging, random forests and boosting.

[8] M. Alice. (2017, January) Performing principal components regression (pcr) in r. [Online]. Available: https://www.r-bloggers.com/performing-principal-components-regression-pcr-in-r/

[9] C. Grove. (2017, February) What are the states where you can play daily fantasy sports? [Online]. Available: http://www.legalsportsreport.com/daily-fantasy-sports-blocked-allowed-states/

[10] A. Crupi. (2016, November) Fantasy sports sites draftkings, fanduel september spend tops 100 million. [Online]. Available: http://adage.com/article/media/draftkings-fanduel-spe/300658/

[11] E. Fisher. (2017, November) Daily fantasy pushes to continue growth streak. [Online]. Available: http://www.sportsbusinessdaily.com/Journal/Issues/2015/03/16/Marketing-and-Sponsorship/Daily-fantasy.aspx

[12] D. Roberts. Everything you need to know about the draftkings and fanduel data scandal. [Online]. Available: http://fortune.com/2015/10/05/draftkings-fanduel-data-scandal/

[13] A. Gallant, "Nfl db," GitHub Relational Database, September 2016.

[14] F. Cooney. (2017) The sports exchange: Nfl draft scout. [Online]. Available: http://www.nfldraftscout.com/ratings/dsprofile.php?pyid=72049draftyear=2011genpos=QB

[15] F. LLC, "Fantasy data: Nfl stats," Exportable CSV Files, 2017.

[16] I. for Digital Research and U. Education. What is dummy coding?

[17] J. Caldwell. Centering and standardizing: Don't confuse your rows with your columns.

[18] DraftKings. Contest format: Salary cap.

[19] NFL. Aaron rodgers career stats.

[20] ——. Drew brees career stats.

[21] G. James, *An Introduction to Statistical Learning with Application in R*, 6th ed., G. Casella, Ed. Springer, November 2015.

[22] R. Nau. Stepwise and all-possible-regressions.

[23] C. Fernandez-Lozano, "Improving enzyme regulatory protein classification by means of svm-rfe feature selection." *Molecular BioSystems*, vol. 5, pp. 1063–71, May 2014.

[24] A. Kowalczyk. Svm tutorial.

[25] D. S. Sayad. Support vector machine - regression (svr).

[26] V. Rao. (2017) Introduction to classification and regression trees (cart). [Online]. Available: http://www.datasciencecentral.com/profiles/blogs/introduction-to-classification-regression-trees-cart

[27] A. Sharma. How does random forest work for regression?

[28] J. Burger, "A basic introduction to neural networks," April 1996, powerPoint Presentation.

[29] R. Dua, "A brief overview of neural networks," October 2003, powerPoint Presentation.

[30] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Prentice Hall, 1998.

[31] F. Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, 1961.

[32] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems*, vol. 2, pp. 303–314, 1989.

[33] M. Alice. Fitting a neural network in r; neuralnet package.

[34] JMP. Overview of principal component analysis.

BIOGRAPHICAL STATEMENT

Nicholas A. King currently lives in Dallas, but was born in Iowa City, Iowa, in 1988 and was raised in Davenport, Iowa. He received his Bachelor's degree in Landscape Architecture from Iowa State University in Ames, Iowa in 2013, and his M.S. degree at The University of Texas at Arlington in 2017. While studying at UTA he has been awarded the Kelcy Warren Graduate Engineering Fellowship and COSMOS (Center on Stochastic Modeling, Optimization, and Statistics) scholarship. He spent last summer working as a Big Data Analyst in a healthcare consulting firm in Chicago, using data and predictive models to asses risk for insurance companies in the healthcare industry. His research interests include the application of data science and data mining methods to real-world data and challenges. Nicholas is a member of COSMOS, the Institute of Industrial and Systems Engineers, the Society for Health Systems, and IEEE.