Sports Analytics with Natural Language Processing: Using Crowd Sentiment to Help Pick Winners in Fantasy Football

Sports Analytics with Natural Language Processing: Using Crowd Sentiment to Help Pick Winners in Fantasy Football

Benjamin Wesley Hendricks

A Thesis in the Field of Software Engineering

for the Degree of Master of Liberal Arts in Extension Studies

Harvard University

May 2022

Abstract

We develop a method for maximizing potential in the sports analytics *picking winners* problem. We do so by: i) leveraging BERT-based, large-scale sentiment analysis of tweets about players. ii) statistical modeling incorporating sentiment analysis, weather, stadium, and other professional projections to create player projections. iii) an integer programming algorithm to construct player lineups. Our experiments show that this method outperforms similar lineup creation using solely existing professional projections by an order of magnitude, with player projections alone outperforming professional projections by nearly 30%. These results suggest a new approach to sports prediction modeling that relies on natural language processing and state-of-the-art language models to incorporate the *wisdom of the crowd*.

Acknowledgements

Thank you to my advisor and thesis director, Hongming Wang. Your guidance and encouragement have made this thesis possible!

Thank you to my employer, LinkedIn, for helping fund my studies through their Education Reimbursement Program.

Thank you to my parents, David, and Catherine Hendricks, for all the support throughout my life and for instilling a desire always to learn more.

Most importantly, thank you to Brittney, my wife, for her unwavering support and understanding as I have spent countless hours working on this thesis and the coursework that prepared me for it.

Contents

List of Figures

List of Tables

Chapter I.
Introduction

There is a rapidly growing industry in generating predictions from the vast amount of data our world produces and stores. The variety of recently developed techniques to use that data in a predictive context is astounding, and humanity is just getting started. Of course, some things are more straightforward to predict than others. Still, there is a particular class of predictions we will focus on here: projecting an ideal ensemble of individual items after assembling predictions for those items.

This multi-layer prediction problem mirrors many real-world scenarios: assembling a stock market portfolio requires first projecting individual stock performance, and then assembling a set of such stocks that ideally maximize overall returns; producing a film requires first finding and hiring a cast that's affordable given the film's budget, and then actually building the story from their work to make the movie as good as it can be; lastly, playing fantasy sports (and fantasy football in particular) requires predicting individual player's performance, then combining multiple players into a lineup that maximizes the potential total points scored (Hunter et al., 2019).

The *picking winners* problem and related solutions apply to all these examples, where a selection of entries is needed (new movies for a studio or player lineups for fantasy sports) that maximizes the probability of any single selection having exceptional performance (Hunter et al., 2019). While part of the picking winners problem, this entry selection under constraint has strong similarities to various covering problems

(Hunter, 2016). This similarity is essential as it reveals a solution mechanism to solve such problems, namely a dynamic programming approach that can generate solutions in polynomial time (Karp, 1975). These selection problems are classified as NP-hard, but through probabilistic approximations, unexpected solutions can be found deterministically (Karp, 1975). These probabilistic approximations are essentially greedy solutions, in which entries are constructed under constraints by maximizing expected value while respecting the given restrictions.

Before finding dynamic programming solutions to ensemble creation, though, we will need individual projections. These should be as accurate as possible, which begs the question: where should these predictions be sourced? While specialists in most fields could represent an expert opinion: broker to help with stock portfolio creation; an executive producer who's made many films; pro sports analyst who is paid to evaluate players; the expert opinion is not necessarily the most accurate!

As is described by Surowiecki, "we feel the need to "chase the expert." The argument [here] is that chasing the expert is a mistake, and a costly one at that. We should stop hunting and ask the crowd (which, of course, includes the geniuses as well as everyone else) instead." (p. 15) (Surowiecki, 2004). To ask the crowd would mean turning to the wealth of data available on the Internet, particularly on social media sites such as Twitter, where everyday people share thoughts and ideas about anything one can think of. Individuals regularly comment and attempt to project outcomes for various events within these online social spheres, from player performance for fantasy sports to company outlooks concerning the stock market.

Furthermore, professional projections are typically costly to produce and are therefore not typically updated in real-time. By not being updated in real-time, breaking news about an entity, which would affect such a professional projection, is not accounted for when the projection is most needed: when assembling a lineup as part of the *picking winners* problem. More importantly, the black-box nature of these projections means that consumers don't know what factors have been accounted for and what factors haven't.

In the specific case of fantasy sports, there is a plethora of social activity surrounding individual players on platforms like Twitter, which can provide last-minute information about a player's upcoming performance that professional projections may not include. In addition, injuries are common in professional sports. They can happen just hours, or even minutes, before a contest, meaning that social media may be the only place such critical information is available. By leveraging the crowd's wisdom in real-time, a professional projection can be strengthened, as mentioned by Dunnington (Dunnington, 2015). Furthermore, retrieving such information in a sports context was shown to be possible in Aloufi and El Saddik's work. They used sentiment analysis on a 54,000 soccer Tweet corpus, finding several methods with reasonable accuracy in determining sentiment specific to the game of soccer (Aloufi & Saddik, 2018).

Many factors affect an NFL player's game day performance, and predictions will never be 100% accurate given the truly stochastic nature of professional NFL football. For example, as described by Dunnington, NFL players have a high chance of injury, a low number of games in which coaches vary usage of key players based on plans they do not publicize, and large roster sizes. All these factors make projections more complicated than less constrained projection spaces like professional basketball or hockey (Dunnington, 2015).

Another critical factor is the weather, where adverse conditions have been shown to affect athletic performance and have shown correlations to betting outcomes in NFL games. Specifically, home-field advantage, in which a team may be better acclimatized to cold or hot conditions, plays a role in player performance and should be something we account for (Borghesi, 2007). Furthermore, home-field entails a fan advantage, where a team's fans will try to disrupt the opponent team with loud cheers during their offensive huddles, making it harder to hear plays being called (Schalter, 2013). That advantage, though, was not present in the 2020 season given the COVID-19 pandemic and the playing of all NFL games in empty stadiums (Schalter, 2021).

Both significant factors, injury, and weather, are typically discussed in social media just before games, especially on Twitter which provides near real-time dissemination of information. As shown by Sinha et al., Twitter is a vibrant space for data about NFL games, with a substantial volume of tweets appearing shortly before games, leading to the latest up-to-the-minute information (Sinha et al., 2013). Before diving into the effectiveness of using this online sentiment data for projections, we will review other work in the area and ensure all readers have a shared understanding of the core game concepts needed to understand the ensuing experimentation. The rest of this thesis is therefore structured as follows. In Chapter 2, we review related work in three core areas leveraged in subsequent experiments, notably Natural Language Processing, Sports Analytics, and Data Modeling. In Chapter 3, we dive deeper into fantasy football itself, followed by Chapter 4, discussing the specific problem at hand and its formulation. Then, Chapters 5 through 8 describe the data, feature selection, algorithms, and experimental results. Finally, we conclude with Chapter 9.

Chapter II.
Related Work

There are three significant research areas from which we draw, with areas of overlap that we will attempt to exploit. First is Natural Language Processing (NLP), specifically in the subdomain of sentiment analysis when applied to online social data (such as tweets from Twitter). The second is sports analytics, with a focus on NFL football. The third is in data modeling via regression and neural networks.

2.1. Natural Language Processing

Social media, especially Twitter, is becoming an ever-growing component of keeping up with sports news and events. As described by Hirsh et al. in their research, fantasy sports players are seeking information on Twitter and other discussion forums to help them play (Hirsh et al., 2012).

Identifying sentiment from Twitter data is standard practice; it is especially interesting in the sports context for various reasons, some separate from the projection goals of this thesis. For example, Gratch et al. took 2014 World Cup tweets to evaluate what makes a sporting event exciting and what factors affect the fan emotional response (Gratch et al., 2015). An interesting finding from this work is that excitement relates most to negative sentiment within the fan reactions, which makes sense in today's outrage-fueled social media world.

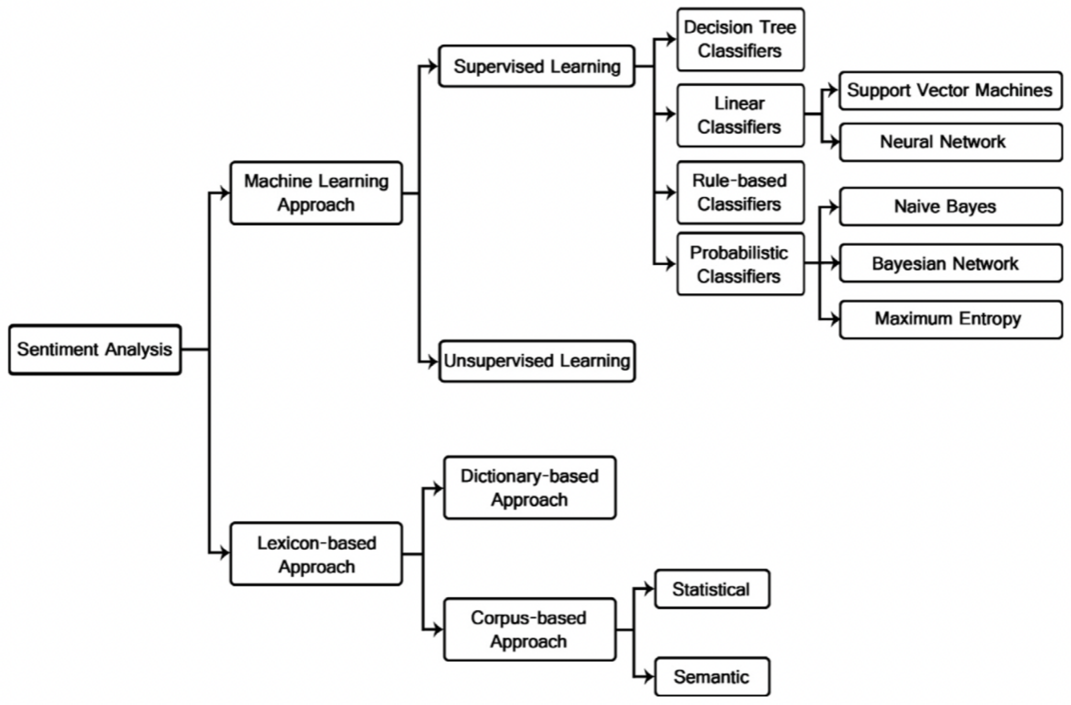
Furthermore, Gratch et al.'s work shows how to leverage tweets for sentiment analysis. By drawing a sample and manually annotating them to compare against the sentiment model's classification, Gratch et al. showed a validation mechanism to ensure the sentiment analysis matched human expectations. Given the use of "off the shelf" sentiment analysis, this could reveal potential idiosyncrasies in sports-related tweets (Gratch et al., 2015).

It would seem like there will be such idiosyncrasies in NFL player-related tweets, given the domain dependency challenges present in this NLP task (Islam & Zibran, 2017). An example of this domain dependency is Islam and Zibran's work in recognizing sentiment from software engineering oriented text. Islam and Zibran improved upon state-of-the-art sentiment analysis tools by adding a lexicon-based classifier, which has a specific software engineering lexicon consisting of several lists of words such as *list of booster words*, *list of phrases*, *list of negations*, *sentimental words* (Islam & Zibran, 2017).

Many different techniques can be used to classify sentiment from text, as shown from the Figure 2.1 (Medhat et al., 2014).

The primary focus for this thesis was the Neural Network approach within the Machine Learning approach sub-tree. However, based on Islam and Zibran's findings, a lexicon-based model may provide more accurate sentiment classification. Our approach, though, follows from the work of Vaswani et al., who found a new transformer-based neural network architecture without any recurrence nor convolution (Vaswani et al., 2017). This simplified transformer architecture is the basis of Devlin et al.'s work on a bidirectional encoder representation of these transformers, a natural language model more commonly known as BERT (Devlin et al., 2019). This model, developed by Google, has seen continued iteration by both Google and other research teams since its introduction, IBM most recently using PoWER-BERT, a BERT model variation with significant inference acceleration while only minimal

Figure 2.1: Tree of sentiment analysis techniques, ranging from lexicon based to machine learning



accuracy loss, using progressive word-vector elimination (Goyal et al., 2020). It is with PoWER-BERT, via IBM Cloud's APIs, that we will evaluate sentiment from tweets.

Others have also used sentiment analysis on sports tweets for various other reasons. For example, Xu and Yu looked at player-generated tweets from the NBA to evaluate whether polarity in their tweets could be related to in-game performance (Xu & Yu, 2015). This work leveraged a lexicon from Finn Arup Nielson, which includes over 2400 words with manually labeled scores between -5 and 5 for polarity (Årup Nielsen, 2011). This approach showed a strong correlation between extracted mood (via sentiment analysis) and in-game performance, especially when mood was extracted the same day as the game (Xu & Yu, 2015). We will attempt to leverage this temporal locality by searching for Tweets as close to the game's start time as possible.

While player emotion may relate to the game outcome, Twitter is also a public forum where we should glean "wisdom of the crowd" insights. Similar to Sinha et al.'s work, Kampakis and Adamides used hashtag-based searches to retrieve their initial Twitter corpus (Sinha et al., 2013)(Kampakis & Adamides, 2014). These searches provide significantly more tweets than what any individual produces on their own, with three months of soccer games generating roughly 2 million tweets (Kampakis & Adamides, 2014). Kampakis and Adamides revealed an important insight, though: hashtag based searches may bundle results from separate entities together, for example, "Saints," a nickname for Southampton FC in the English Premier League, which is the name of an NFL team, the New Orleans Saints (Kampakis & Adamides, 2014).

2.2. Sports Analytics

However, there has been little work identifying tweets for individual players, similar to the mentioned team-based hashtag approach. Additionally, as noted by King, there has generally not been much writing about Daily Fantasy Sports player projections, in any sports, likely due to the lack of motivation to make such work public when these models can provide advantages in online games with real money on the line (King & Leboulluec, 2017). Interestingly, many papers claim to be the first in their respective area, which is partially true, yet conceals the deep connective tissue between this work.

One key area of research within Daily Fantasy (DFS) has been attempts to show DFS to be a game of skill and not simply gambling (Rychlak, 1992). By using her lineup construction logic to generate random lineups to evaluate their performance against optimized lineups, Sarah Newell showed that winning in DFS is not random (Newell, 2017).

While King did evaluate both regression models and a neural network model for individual players, the narrow scope of quarterbacks only combined with limited results for neural networks makes this an exciting area for further investigation (King & Leboulluec, 2017).

The primary dataset used for this research was the 2020 NFL season, which adds an interesting caveat to our use of stadium and fan features, given that for much of the season, stadiums were empty due to COVID-19. These ”ghost games” have been shown to have an impact in a variety of ways, from players to referees (Fischer et al., 2020). However, given the difficulty in acquiring data for multiple seasons, we will leave performing a similar analysis on a season without ”ghost games” to the reader.

Ever since Moneyball revolutionized the baseball world just twenty years ago, there has been an ever-increasing momentum shift towards measuring professional athletes in various ways to maximize performance. By leveraging players' statistical performance to determine which players to select for a team and which players to play, Billy Beane ignited an analytics revolution that now has players wearing activity trackers at all times in most professional sports leagues (Lewis, 2004). In addition, there are now conferences dedicated to the topic, such as the MIT Sloan Sports Analytics Conference running since 2005.

Since before Moneyball, there has been analytics, although the catalyst shifted focus from team analysis to individual player analysis. Prediction in sports has a long history; linear statistical models were already being used in the 1970s to attempt to predict game outcomes (Harville, 1980). Later, Hal Stern provided a normal random variable approximation for game outcomes in 1991 that is still in use today (Stern, 1991). Pro Football Reference has extended this model to be able to evaluate win probabilities within a game by adding further information to the model about the current game situation (down/distance/field position) (PFR, 2021).

Projections for individual players have received less attention, and although many sports websites produce such estimates, they do not reveal how they assembled them. Some writing has been looking to model player performance; for example, Lutz showed how using support vector regression and neural networks to model player performance can be effective. These results were limited, however, to a small set of players, positions (Lutz, 2015). Interestingly, using a similar approach as King by only looking at quarterbacks, the neural network approach provided less predictive power than the regression approach. Given the expansive nature of neural network types and data formulations, it is likely that this was not a failure of the approach generally but of these authors' specific chosen path. Furthermore, given the limited available data, it could simply be a problem of not providing the neural network with enough examples to train.

2.3. Data Modeling

While there is a dearth of data, that has never stopped enthusiasts from throwing their hat into the arena. As mentioned in the introduction, Hunter et al. developed an integer programming approach to assembling full DFS lineups with minimal player projections modeling. Instead, Hunter et al. relied on existing professional website projections from RotoGrinders and Daily Fantasy Nerd (Hunter et al., 2019). Using a similar integer programming approach, Sarah Newell showed another way of constructing lineups leveraging standard deviation of player projections as a critical feature and showing through stochastic lineup generation that DFS is not a game of chance at all (Newell, 2017).

There have been several other examples of integer programming use cases in the sports world. For instance, Sharp et al. constructed cricket lineups (Sharp et al., 2011), or Özlü and Sokol, who used integer programming to schedule MLB scouts (Özlü & Sokol, 2016).

Haugh & Singal took a different route than most, though, and explicitly attempted to model opponent behavior to help construct lineups, something that Hunter eschewed to focus on optimizing for lineup mean projection (Haugh & Singal, 2019). Using Dirichlet regression, Haugh & Singal found success in DFS contests with many other entrants. Modeling opponent behavior in combination with modeling the players' potential points is a unique approach not taken by others. That said, most integer programming lineup explorations described relied on fixed projections from other sources, an opportunity to further improve upon these modeling strategies for DFS contests.

Tangentially, neural networks have seen an explosion of use cases in recent times, an example of which being statistical language modeling. These models have state-of-the-art performance in their respective domains (De Mulder et al., 2015), as do neural networks used in machine understanding of images and video media. In the sports context, these models have been used for predictions of soccer game outcomes (Zhang et al., 2021). There are many neural networks types, ranging from Convolutional Neural Networks (CNNs) to Recurrent Neural Networks (RNNs). We will focus on a specific form of RNNs called Long Short Term Memory (LSTM) that have time-series-based characteristics that are interesting for predicting performance for a repeating entity. These models are nonlinear, unlike regression, which could be better suited to sports contests where existing raw data does not capture the multitudes of potential factors players may face (Zhang et al., 2021).

Chapter III.
What is Fantasy Football

Fantasy football is a derivative game atop actual professional American football player performance. This derivative game originated shortly after World War II (Green, 2014), only 20 or so years after the first pro football league was formed (History.com, 2009). In this game, individual players are selected to create a "lineup," whose actual in-game performance dictates scores and outcomes in the fantasy world. Today, most professional sports have a fantasy version, with Fantasy Basketball, Baseball, Hockey, Soccer, and Football all being highly popular in the United States. While all fantasy sports are generally similar, we will further focus on fantasy football going forward.

In countries outside the US, fantasy football typically relates to soccer, but fantasy football refers to the NFL in the US. This proved frustrating in our research, given the many "fantasy football" papers found that were actually about soccer!

Regardless, the NFL is the largest professional sports league in the world in terms of yearly revenue. As a result, it has spawned an immense fantasy sports industry reliant upon this success. Over five years ago, the industry was already estimated to total $26 Billion in annual spending across the US and Canada, which has only grown since then (Wong, 2015).

3.1. NFL Football

NFL Football has been around in the United States for almost 100 years and is a game where two teams, each with more than 50 players, compete in a highly physical athletic contest. Each team has only 11 players on the field at any time, but there are many opportunities to rotate players out and three different "units" that each team needs to field: offense, defense, and special teams. Offensive and defensive players make up the bulk of where fantasy points are scored, but special teams are often critical to game outcomes.

While similar to rugby in its physicality, American football has far more stoppage in play, which creates an extremely high pace of play while the ball is in motion, paired with many moments of "rest" in-between plays. A game is 60 minutes, comprised of four 15-minute quarters, and each offense typically gets on average 60 plays per game to try and win. Of course, the number of plays is highly variable, depending on many in-game circumstances, but this average helps to show just how few opportunities there are in a football game to have an impact. Unlike a sport like soccer or basketball, which is far more free-flowing, there are far more limited opportunities in NFL football to score, making fantasy football so interesting.

3.2. Fantasy Football

There are now many forms of fantasy football, but we will focus on a specific type: Daily Fantasy Football. In this particular fantasy sports type, players are selected for their performance in a single game and earn points based on their performance in that game. The better they play, generally, the more points they will score. Other forms of fantasy football focus on creating a team for an entire season, while other forms focus more on game outcomes themselves instead of individual players.

In all cases, projections for individual players are critical.

To create these projections, let's first look at how points are scored in fantasy football to understand better what needs to be projected. NFL football is a highly complex sport, with several distinct positions filled by players with a wide range of talents. On the field at any one time are 22 players, 11 on offense and 11 on defense. Daily fantasy football relies primarily on offensive players and only provides a choice of an entire defensive team, leading to two distinct sets of scoring rules.

Defensive units are scored based on their ability to stop opposing offenses and earn the most points when they can score points themselves – for example, by intercepting a pass and scoring a touchdown. Offensive players are scored based on their ability to score points, including the progress down the field it takes to do so. For context, a touchdown, or the scoring of 6 points by an offensive squad, is awarded whenever a player reaches the end zone (each end of a football field) with the ball. This can happen either by running the ball into the end zone or catching a pass in the end zone.

Running and catching help describe the primary offensive NFL positions, as players specialize in one or the other. Quarterbacks (QB) are throwing specialists responsible for throwing passes to other players to score points. Wide receivers (WR) are catching specialists, and are responsible for catching passes from QBs, hopefully in the end zone! Running backs (RB) are running specialists and are handed the ball by QBs to run the ball for a touchdown. These three offensive positions make up most fantasy points in a daily fantasy football game. Tight ends (TE) are another player position type, similar to WRs. However, they are typically bigger and stronger to help them act as blockers for RBs in run situations. An offensive unit of 11 players typically includes five linemen, imposing and powerful players that block defensive players to prevent QBs, WRs, and RBs from getting tackled. These players, however,

Offense	
Action	Points
Passing TD	+4 Pts
25 Passing Yards	+1 Pt (+0.04 Pts/Yards)
300+ Yard Passing Game	+3 Pts
Interception	-1 Pt
Rushing TD	+6 Pts
10 Rushing Yards	+1 Pt (+0.1 Pts/Yards)
100+ Yard Rushing Game	+3 Pts
Receiving TD	+6 Pts
10 Receiving Yards	+1 Pt (+0.1 Pts/Yards)
100+ Yard Receiving Game	+3 Pts
Reception	+1 Pts
Punt/Kickoff/FG Return for TD	+6 Pts
Fumble Lost	-1 Pt
2 Pt Conversion (Pass, Run, or Catch)	+2 Pts
Offensive Fumble Recovery TD	+6 Pts

Table 3.1: DraftKings Offense Scoring Rules

are not selectable in DFS contests.

With that context, Table 3.1 shows the specific scoring rules DraftKings uses for offense, which we will refer to going forward (DraftKings, 2021). Also, Table 3.2 shows the specific scoring rules for defense.

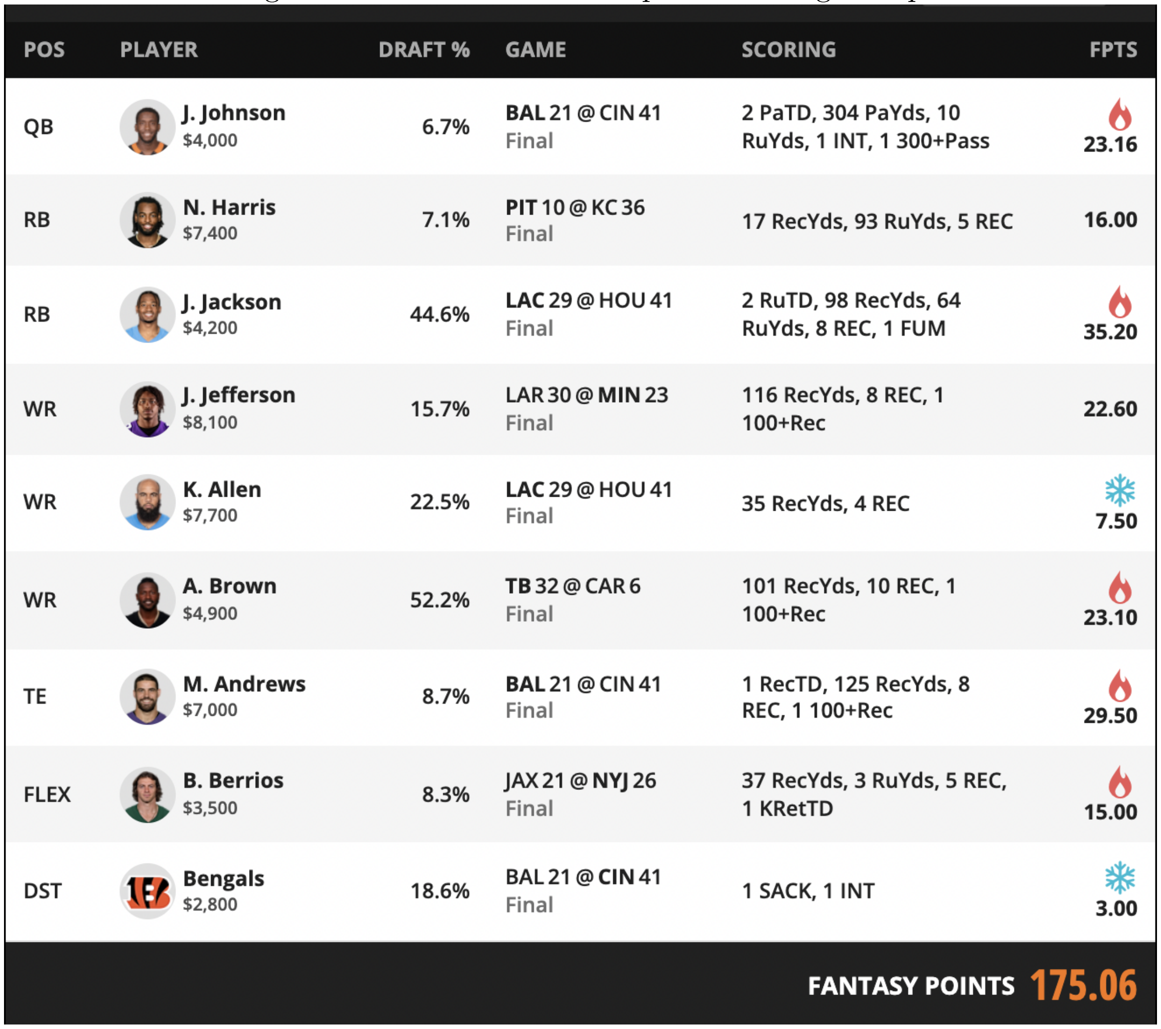
To better visualize how these rules would apply, Figure 3.1 shows a lineup with player in-game performance along with their fantasy points scored on DraftKings.

Estimating the size of the fantasy football market is tricky, as there are so many aspects to consider. In 2015, total annual spending on fantasy football was estimated at $26B, but this doesn't account for all time spent doing fantasy activities (Wong, 2015). By leveraging data about the average income of fantasy players and estimating the amount of time spent, the tangible and intangible economic activity may surpass $50B. This number is larger than the NFL's revenues and provides a complex market acting as a fantastic testbed for various analytical research projects,

Defense	
Action	Points
Sack	+1 Pt
Interception	+2 Pts
Fumble Recovery	+2 Pts
Punt/Kickoff/FG Return for TD	+6 Pts
Interception Return TD	+6 Pts
Fumble Recovery TD	+6 Pts
Blocked Punt or FG Return TD	+6 Pts
Safety	+2 Pts
Blocked Kick	+2 Pts
2 Pt Conversion/Extra Point Return	+2 Pts
0 Points Allowed	+10 Pts
1-6 Points Allowed	+7 Pts
7-13 Points Allowed	+4 Pts
14-20 Points Allowed	+1 Pt
21-27 Points Allowed	+0 Pts
28-34 Points Allowed	-1 Pt
35+ Points Allowed	-4 Pt

Table 3.2: DraftKings Defense Scoring Rules

Figure 3.1: Screenshot of sample DraftKing lineup



POS	PLAYER	DRAFT %	GAME	SCORING	FPTS
QB	J. Johnson $4,000	6.7%	BAL 21 @ CIN 41 Final	2 PaTD, 304 PaYds, 10 RuYds, 1 INT, 1 300+Pass	23.16
RB	N. Harris $7,400	7.1%	PIT 10 @ KC 36 Final	17 RecYds, 93 RuYds, 5 REC	16.00
RB	J. Jackson $4,200	44.6%	LAC 29 @ HOU 41 Final	2 RuTD, 98 RecYds, 64 RuYds, 8 REC, 1 FUM	35.20
WR	J. Jefferson $8,100	15.7%	LAR 30 @ MIN 23 Final	116 RecYds, 8 REC, 1 100+Rec	22.60
WR	K. Allen $7,700	22.5%	LAC 29 @ HOU 41 Final	35 RecYds, 4 REC	7.50
WR	A. Brown $4,900	52.2%	TB 32 @ CAR 6 Final	101 RecYds, 10 REC, 1 100+Rec	23.10
TE	M. Andrews $7,000	8.7%	BAL 21 @ CIN 41 Final	1 RecTD, 125 RecYds, 8 REC, 1 100+Rec	29.50
FLEX	B. Berrios $3,500	8.3%	JAX 21 @ NYJ 26 Final	37 RecYds, 3 RuYds, 5 REC, 1 KRetTD	15.00
DST	Bengals $2,800	18.6%	BAL 21 @ CIN 41 Final	1 SACK, 1 INT	3.00

FANTASY POINTS 175.06

but has received seemingly little academic interest thus far (Goff, 2013).

The following evaluation of modeling methods for fantasy football projections will show that while it may appear like a gambling game, there is an ability to, with skill, improve your chances in these large game formats (FSGA, 2021). DraftKings makes this explicit, sharing data that a small subset of players, often thought of as *sharks*, win the majority of contests from the majority of contestants, thought of as *minnows*.

The approach to model player projections is to leverage as many relevant data sources as possible, including one representing ”the crowd,” to improve existing, professional fantasy platform projections. Two modeling approaches will be attempted: regression and neural networks. These will leverage features from these various sources and several engineered features based on data from those sources.

Chapter IV.
Problem Statement

We intend to develop an algorithm that will produce projections for fantasy football players and use those projections with another algorithm we will develop to create an ensemble from those projections. The first algorithm will take an ensemble approach, leveraging data from other existing models and retrieving information, and applying natural language processing techniques, specifically sentiment analysis, to that data.

Specifically, we will retrieve tweets about NFL players and retrieve sentiment about those players from their respective tweets using IBM Cloud's NLP tools. This data will then be combined with multiple other data sources, including modeled data from NFL.com and FantasyData.io.

The resulting dataset we can refer to as $\boldsymbol{X}$, where each row contains information about a specific player for a particular game. The label vector $\boldsymbol{y}$ contains the actual points scored by that player for a given row in $\boldsymbol{X}$. We will then apply a machine learning algorithm to $\boldsymbol{X}$ and $\boldsymbol{y}$ to produce a function Φ, for which the input is a row x_i of $\boldsymbol{X}$ resulting in $\Phi(x_i) = y_i$.

Given these projections, we can then construct a team, with an objective to maximize projected score while respecting the positional and salary constraints imposed by the game. So from $\boldsymbol{X}$ we aim to assemble a 9-player team, in the DraftKings game the positions must match:

$$\begin{pmatrix} p_1 \to QB_1 & p_2 \to RB_1 & p_3 \to RB_2 \\ p_4 \to WR_1 & p_5 \to WR_2 & p_6 \to WR_3 \\ p_7 \to TE_1 & p_8 \to F_1 & p_9 \to D_1 \end{pmatrix}$$

There is an associated cost for each player, represented by their salary, which we can denote as follows:

$$p = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_9 \end{pmatrix} \quad s = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_9 \end{pmatrix}$$

In addition, the lineups' salary sum must remain under the $50,000 DraftKings salary cap.

Consider, then, that p_k is represented by x_i, a row in our dataset $\boldsymbol{X}$, with an associated salary s_k, then using our function ϕ we can note the construction of a lineup as:

$$max \sum_{k=1}^{9} \phi(x_i)$$

over all i subject to

$$\sum_{k=1}^{9} s_k \leq 50,000$$

This formulation is a form of the knapsack problem, so an integer programming solution leveraging dynamic programming will help produce ideal lineups given the projections from our regression step. We are looking to solve these several steps in the following chapters.

Chapter V.
Data Sources & Processing

Data used for this analysis was the entirety of the 2020 season via several sources: NFL.com, Draftkings, FantasyData.io, Twitter, IBM Cloud, WeatherData Github repository.

All of this data is organized in a single location through a coordinated set of scripts, with 8,498 data points representing 17 weeks of football across 32 teams, 256 games, and roughly 500 players per week.

There are only 16 games that any single player could potentially play within this dataset, limiting the amount of data we can use to model future player performance. For a point of comparison, NBA & NHL players have an 82 game regular season, and MLB players have a 162 game regular season, providing far more game data per player. In addition, injuries are more common in NFL football than in most other pro sports, leading to very few players even playing an entire 16 game season. In our dataset, including over 900 players who participated in the 2020 NFL season, only 18, or 2%, have 16 data points available. Note that there are 17 weeks in an NFL season (as of 2021, there are now 18 weeks with 17 games), but each team only plays 16 games with a bye week as a semi-random point of the season.

5.1. NFL.com

This source is powered by a Python scraper with two main phases, each phase having two separate scripts for offensive players and defensive teams. First is a bash script that saves raw CURL HTML output into text files, followed by a python script that reads the text file and converts it into a CSV of players or teams. There are two pairs of scripts, one that fetches all offensive players and the other that fetches all defensive teams.

This data source provides projections from an existing model from NFL.com, which has no public information available, so it must be treated as a black box. However, it is reasonable to believe that this model may already account for features we include within our approach.

5.2. Draftkings

This source is powered by manual intervention to download a CSV file on the Draftkings website each week, performed weekly during the 2020 season. This CSV contains the weekly set of players available to draft and their salaries. Furthermore, a second CSV was downloaded each week of the 2020 season with the results from the primary contest, which includes actual fantasy points scored by each player, along with the percentage of contest entrants who drafted them.

5.3. FantasyData.io

FantasyData.io is a subscription-based API data source. It allows for free historical access, so the data used for this research was free, but data for the current season would have incurred a cost. A set of projections for all players in the 2020 season was retrieved using this API.

5.4. Weather Data Repository

URL: *https://github.com/ThompsonJamesBliss/WeatherData/tree/master/data*

This source is an open-source repository with 2020 weather data for all NFL stadiums during the regular season, with an easy-to-download CSV file available. All stadium-based and weather features originated from this data source.

5.5. Big Data: Twitter

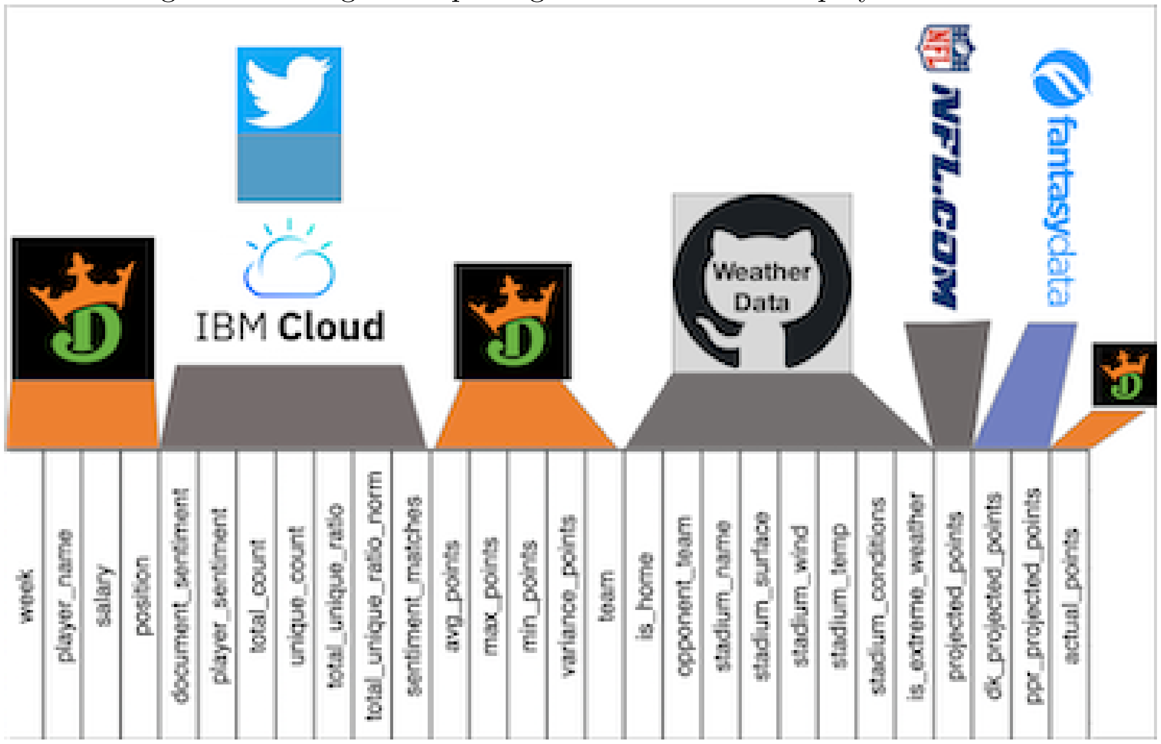
This source is powered by the Twitter API, which is subscription-based but has an Academic Access lane that we used to retrieve up to 10 million tweets per month for free. Over 30 million tweets were gathered, capturing tweets about all players, during each week of the 2020 season, within the 24-hour window before Sunday NFL games. This API provides JSON output, which we saved in raw form for more flexibility in further analysis.

Unlike the other data sources, which primarily deal with small amounts of data, this was the sole big data source used. In addition, the 24-hour window was selected to reduce the sheer number of tweets retrieved to accelerate processing. It would be possible, though, and potentially beneficial to search for all tweets within a given week about a player, but this would require increased data caps from Twitter, which could be expensive. So instead, we relied solely on the Academic Access provided for free, which limited us to 10M tweets/month.

5.6. Data Processing

IBM Cloud was used to process all 30M+ tweets gathered via Twitter and generate sentiment data from these tweets. Initial processing was performed locally on a Macbook Pro. Then, IBM's API took that processed tweet data and retrieved

Figure 5.1: Diagram depicting all data sources employed in thesis



JSON objects, including document and entity sentiment, from that tweet data. While this is a paid API, our usage levels did not incur any costs.

Several additional Python scripts transform the disparate data sources described so far into a single, organized CSV file per game week, with rows representing players and a standard set of columns with 26 features to be used in modeling.

The unique key used to match up data from most of these various sources is simply the player name, with the team name as a second key used for stadium and weather features. Since all sources are saved in CSV files before cleaning, this aggregation script primarily reads many different CSV files from disk to produce a single, aggregated version of all of that data.

Figure 5.1 shows the feature columns and their respective sources.

Chapter VI.
Feature Engineering

With so many data sources, it was critical to find the optimal set of features that could be used to impact projection potential materially within our regression model. However, before finding such an optimal set, experimentation was needed to evaluate various features and their derivatives to see which ones had the most significant impact during regression. Therefore, limiting the number of features given our limited data was critical to ensure regression had enough examples per bin.

There is an uneven distribution of features, with 9 features (4 engineered) from DraftKings, 7 features (1 engineered) from the weather data source, 6 features (all engineered) from IBM + Twitter, 2 (non engineered) from FantasyData, and lastly 1 non-engineered feature from NFL.com. This distribution is worth calling out from the initial set of 26 features as the distribution of features in the final models that were optimal is not the same.

The feature engineering is primarily based on the statistical analysis of the fantasy points scored by players, generating a few derivatives such as point average, min and max, and variance. There is also considerable feature engineering for the sentiment-based features, described further below. As mentioned in Landers and Duperrouzel, "the features used by previous investigations are inherently encoded within the fantasy points themselves and a rich feature set can be obtained by studying these derivative elements alone" (Landers & Duperrouzel, 2019). Furthermore,

Features				
ID	Feature Name	Type	Engineered	Source
1	week	Numerical	False	DraftKings
2	player_name	Categorical	False	DraftKings
3	salary	Numerical	False	DraftKings
4	position	Categorical	False	DraftKings
5	document_sentiment	Numerical	True	IBM + Twitter
6	player_sentiment	Numerical	True	IBM + Twitter
7	total_count	Numerical	True	IBM + Twitter
8	unique_count	Numerical	True	IBM + Twitter
9	total_unique_ratio	Numerical	True	IBM + Twitter
10	sentiment_matches	Categorical	True	IBM + Twitter
11	avg_points	Numerical	True	DraftKings
12	max_points	Numerical	True	DraftKings
13	min_points	Numerical	True	DraftKings
14	variance_points	Numerical	True	DraftKings
15	team	Categorical	False	DraftKings
16	is_home	Categorical	False	Weather
17	opponent_team	Categorical	False	Weather
18	stadium_name	Categorical	False	Weather
19	stadium_surface	Categorical	False	Weather
20	stadium_wind	Numerical	False	Weather
21	stadium_temp	Numerical	False	Weather
22	stadium_conditions	Categorical	False	Weather
23	is_extreme_weather	Numerical	True	Weather
24	projected_points	Numerical	False	NFL.com
25	dk_projected_points	Numerical	False	FantasyData
26	ppr_projected_points	Numerical	False	FantasyData

Table 6.1: Full Feature Set

professional point projections likely incorporate features in this set, making it essential to reduce the feature set to the collection of features without such overlap.

6.1. Large Scale Sentiment Feature Engineering

Several engineered features derive from data gathered via Twitter's API. In addition, unlike the work of Sinha et al., we will not only be retrieving Tweets based on teams but based on specific players as well. To perform this search, we use the Twitter API search functionality with a player name (or team name, for defenses) as the keyword, bound to a single day.

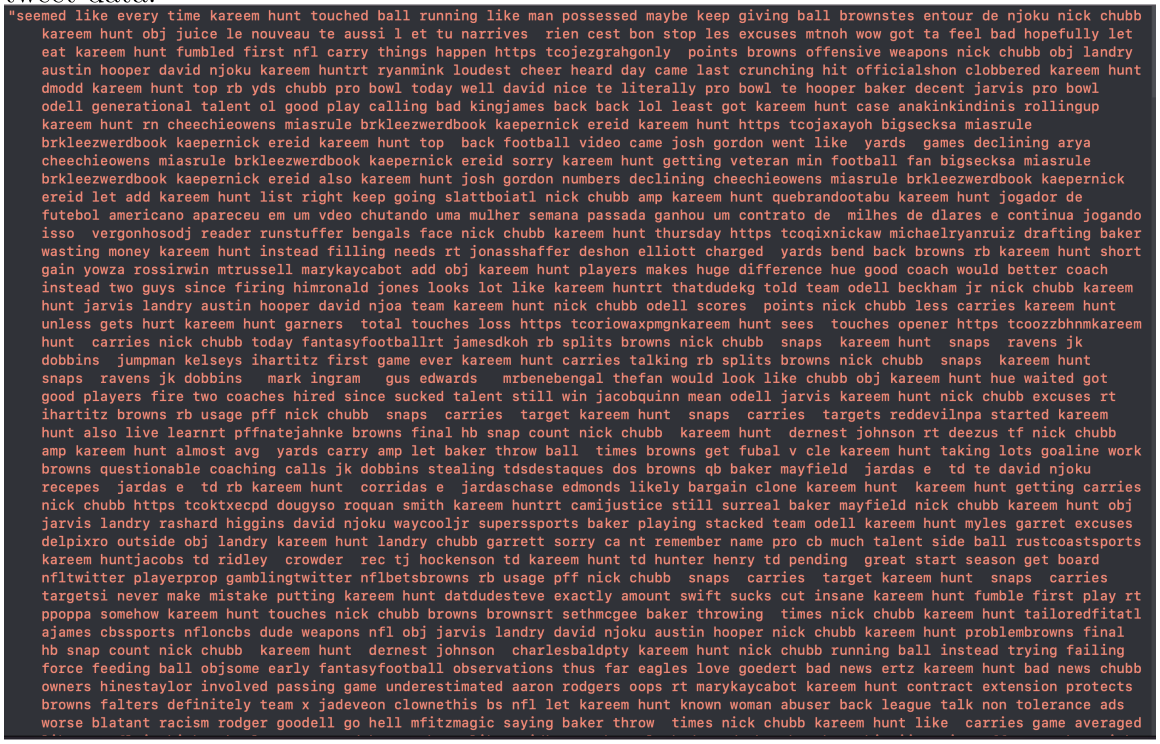
While the ideal solution would have been to compute a time range of tweets to search for by player's game time, we fetched Tweets from Saturdays, given that most games happen on Sundays. There are weekly Thursday and Monday games, though, so we would expect improved model performance if further research takes this extra search step.

This approach allowed us to retrieve all Tweets about a player on each game week's Saturday, which are then concatenated into a single string after de-duplication, followed by a couple of further preprocessing steps: (1) remove all emojis, (2) remove all punctuation and tokenize, (3) remove digits and symbols, (4) remove emails and websites, (5) remove empty spaces, (6) standardize all stopwords into lowercase, (7) join all tokens back together and lemmatize.

The resulting string is then passed to IBM Cloud's natural language understanding API, which as input, takes a text blob up to 50,000 characters. Finally, it outputs a JSON blob including document sentiment scores, recognized entities, and respective sentiment scores.

Unlike in Gratch's work, where every individual Tweet was run through sentiment analysis, where a manual annotation could compare the accuracy of polarity

Figure 6.1: Screenshot of a sampling from input to sentiment analysis, preprocessed tweet data.



```
"seemed like every time kareem hunt touched ball running like man possessed maybe keep giving ball brownstes entour de njoku nick chubb
    kareem hunt obj juice le nouveau te aussi l et tu narrives  rien cest bon stop les excuses mtnoh wow got ta feel bad hopefully let
    eat kareem hunt fumbled first nfl carry things happen https tcojezgrahgonly  points browns offensive weapons nick chubb obj landry
    austin hooper david njoku kareem huntrt ryanmink loudest cheer heard day came last crunching hit officialshon clobbered kareem hunt
    dmodd kareem hunt top rb yds chubb pro bowl today well david nice te literally pro bowl te hooper baker decent jarvis pro bowl
    odell generational talent ol good play calling bad kingjames back back lol least got kareem hunt case anakinkindinis rollingup
    kareem hunt rn cheechieowens miasrule brkleezwerdbook kaepernick ereid kareem hunt https tcojaxayoh bigsecksa miasrule
    brkleezwerdbook kaepernick ereid kareem hunt top  back football video came josh gordon went like  yards  games declining arya
    cheechieowens miasrule brkleezwerdbook kaepernick ereid sorry kareem hunt getting veteran min football fan bigsecksa miasrule
    brkleezwerdbook kaepernick ereid also kareem hunt josh gordon numbers declining cheechieowens miasrule brkleezwerdbook kaepernick
    ereid let add kareem hunt list right keep going slattboiatl nick chubb amp kareem hunt quebrandootabu kareem hunt jogador de
    futebol americano apareceu em um vdeo chutando uma mulher semana passada ganhou um contrato de  milhes de dlares e continua jogando
    isso  vergonhosodj reader runstuffer bengals face nick chubb kareem hunt thursday https tcoqixnickaw michaelryanruiz drafting baker
    wasting money kareem hunt instead filling needs rt jonasshaffer deshon elliott charged  yards bend back browns rb kareem hunt short
    gain yowza rossirwin mtrussell marykaycabot add obj kareem hunt players makes huge difference hue good coach would better coach
    instead two guys since firing himronald jones looks lot like kareem huntrt thatdudekg told team odell beckham jr nick chubb kareem
    hunt jarvis landry austin hooper david njoa team kareem hunt nick chubb odell scores  points nick chubb less carries kareem hunt
    unless gets hurt kareem hunt garners  total touches loss https tcoriowaxpmgnkareem hunt sees  touches opener https tcoozzbhnmkareem
    hunt  carries nick chubb today fantasyfootballrt jamesdkoh rb splits browns nick chubb  snaps  kareem hunt  snaps  ravens jk
    dobbins  jumpman kelseys ihartitz first game ever kareem hunt carries talking rb splits browns nick chubb  snaps  kareem hunt
    snaps  ravens jk dobbins   mark ingram   gus edwards   mrbenebengal thefan would look like chubb obj kareem hunt hue waited got
    good players fire two coaches hired since sucked talent still win jacobquinn mean odell jarvis kareem hunt nick chubb excuses rt
    ihartitz browns rb usage pff nick chubb  snaps  carries  target kareem hunt  snaps  carries  targets reddevilnpa started kareem
    hunt also live learnrt pffnatejahnke browns final hb snap count nick chubb  kareem hunt  dernest johnson rt deezus tf nick chubb
    amp kareem hunt almost avg  yards carry amp let baker throw ball  times browns get fubal v cle kareem hunt taking lots goaline work
    browns questionable coaching calls jk dobbins stealing tdsdestaques dos browns qb baker mayfield  jardas e  td te david njoku
    recepes  jardas e  td rb kareem hunt  corridas e  jardaschase edmonds likely bargain clone kareem hunt  kareem hunt getting carries
    nick chubb https tcoktxecpd dougyso roquan smith kareem huntrt camijustice still surreal baker mayfield nick chubb kareem hunt obj
    jarvis landry rashard higgins david njoku waycooljr superssports baker playing stacked team odell kareem hunt myles garret excuses
    delpixro outside obj landry kareem hunt landry chubb garrett sorry ca nt remember name pro cb much talent side ball rustcoastsports
    kareem huntjacobs td ridley  crowder  rec tj hockenson td kareem hunt td hunter henry td pending  great start season get board
    nfltwitter playerprop gamblingtwitter nflbetsbrowns rb usage pff nick chubb  snaps  carries  target kareem hunt  snaps  carries
    targetsi never make mistake putting kareem hunt datdudesteve exactly amount swift sucks cut insane kareem hunt fumble first play rt
    ppoppa somehow kareem hunt touches nick chubb browns brownsrt sethmcgee baker throwing  times nick chubb kareem hunt tailoredfitatl
    ajames cbssports nfloncbs dude weapons nfl obj jarvis landry david njoku austin hooper nick chubb kareem hunt problembrowns final
    hb snap count nick chubb  kareem hunt  dernest johnson  charlesbaldpty kareem hunt nick chubb running ball instead trying failing
    force feeding ball objsome early fantasyfootball observations thus far eagles love goedert bad news ertz kareem hunt bad news chubb
    owners hinestaylor involved passing game underestimated aaron rodgers oops rt marykaycabot kareem hunt contract extension protects
    browns falters definitely team x jadeveon clownethis bs nfl let kareem hunt known woman abuser back league talk non tolerance ads
    worse blatant racism rodger goodell go hell mfitzmagic saying baker throw  times nick chubb kareem hunt like  carries game averaged
```

recognition, an aggregation of Tweets is used here (Gratch et al., 2015). Manual annotation is possible on individual Tweets, but with each player having an average of 4,100 tweets per week, the preprocessed text being analyzed looks something like what is shown in Figure 6.1.

IBM Cloud provides an NLP API that leverages an underlying neural network based on the pre-trained BERT-Base, English Uncased model, fine-tuned on IBM Claim Stance Dataset (Google, 2020)(Bar-Haim et al., 2017). The core BERT model is a 12-layer, 768-hidden layer, 110M parameter model (Google, 2020). IBM benchmarked the model against Sentiment140 and IMDB Reviews datasets, the former composed of Tweets. On this Twitter-based dataset, IBM's model has over 80% recall, especially impressive given the dataset was not used for training at all (IBM,

Number of Tweets		
Week	Average	Max
1	7508	167825
2	6728	194193
3	5950	135327
4	5052	185808
5	6250	123013
6	5647	123013
7	4708	127807
8	4368	146047
9	4228	139789
10	3282	77953
11	3892	87001
12	4523	136604
13	4384	127868
14	3279	123025
15	3560	90101
16	4751	150491
17	4987	220690

Table 6.2: Number of Tweets

2021).

Through all the above, we arrive at the six engineered features from this data: (1) document_sentiment, which represents the IBM Cloud sentiment score for the entire preprocessed tweet text blob, (2) player_sentiment, which represents the IBM Cloud sentiment score for the recognized player (which was impressively recognized in 100% of attempts), (3) sentiment_matches, a derived Boolean representing whether document_sentiment and player_sentiment share the same sign (as in, they are both positive, or both negative, or not), (4) total_count, which represents the total number of Tweets captured for the given player in that game week, (5) unique_count, which represents the total number of Tweets captured after de-duplication, and lastly (6) total_unique_ratio, which represents the ratio between total and unique counts.

6.2. Weather & Stadium Features

Weather and stadium information was largely unmodified, with a single engineered feature developed to reduce all weather features into one. Specifically, is_extreme_weather is an engineered Boolean representing whether or not the stadium is outdoors AND either (1) the wind speed is above 15 MPH, or (2) the temperature is below 35 degrees Fahrenheit. These thresholds were approximated based on prior analysis, which showed a clear difference in performance at these thresholds (Mancuso, 2019)(Cheema, 2020)

Chapter VII.
Algorithms

The primary algorithm used is simple, ordinary least squares regression. Multiple variations of input features and training/test splits were attempted with this modeling approach. An LSTM RNN was also used to model this data, where hyperparameters were tuned to uncover the best LSTM for the 2020 NFL season dataset. Both algorithms employed a shared validation function, which measured R^2 of predicted results against the R^2 of professional sports site projections. Therefore, an algorithm's success was measured in the improvement of R^2 over existing professional predictions.

7.1. Regression

With ordinary least squares the function $\phi(x_i) = y_i$ is better represented by:

$y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \epsilon_i$

Where x_{ip} represents the $p^t h$ feature for that player column. The coefficients β_p are best known as the intercept in a single variable regression and what needs to be determined first for the model to be used. The previous equation can also be formulated in terms of matrices:

$\mathbf{y} = \mathbf{X}\beta + \epsilon$

where:

$$X = \begin{pmatrix} X_{11} & X_{12} & \dots & X_{1p} \\ X_{21} & X_{22} & \dots & X_{2p} \\ \vdots & \vdots & \dots & \vdots \\ X_{n1} & X_{n2} & \dots & X_{np} \end{pmatrix} \quad \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix} \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

Which yield the following quadratic minimization function to find optimal β coefficients:

$\hat{\beta} = arg_{\beta} min S(\beta)$ with $S(\beta) = \sum_{i=1}^{n} |y_i - \sum_{j=1}^{p} X_{ij}\beta_j|$

$\therefore \hat{\beta} = \beta + ((X^T X)^{-1} X^T \epsilon$

Once we solve this, we can use $\hat{\beta}$ in $\mathbf{y} = \mathbf{X}\beta + \epsilon$ to create predictions, and we can evaluate those predictions by calculating the coefficient of determination R^2:

$R^2 = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2} = 1 - \frac{RSS}{TSS}$

We are looking to maximize R^2 within its [0,1] range.

7.2. Neural Network

An LSTM is a neural network aimed at maintaining memory while projecting values from a sequence of data, for which a season of week-by-week data is seemingly an ideal match. Unlike a traditional neural network, an LSTM can process data sequentially and maintain its hidden state throughout. This is done via hidden states within an LSTM gate, as seen in Figure 7.1.

There are several different types of LSTMs, we will leverage a many to one LSTM, where multiple weeks of player data are input, and a single week's projection is output. A basic example of one of these can be seen in Figure 7.2

To perform training for this type of neural network, we need to apply a stochastic optimization algorithm, for which Adam has become a popular choice. Adam provides a more computationally efficient alternative to standard gradient descent by

Figure 7.1: Gate architecture of LSTM network

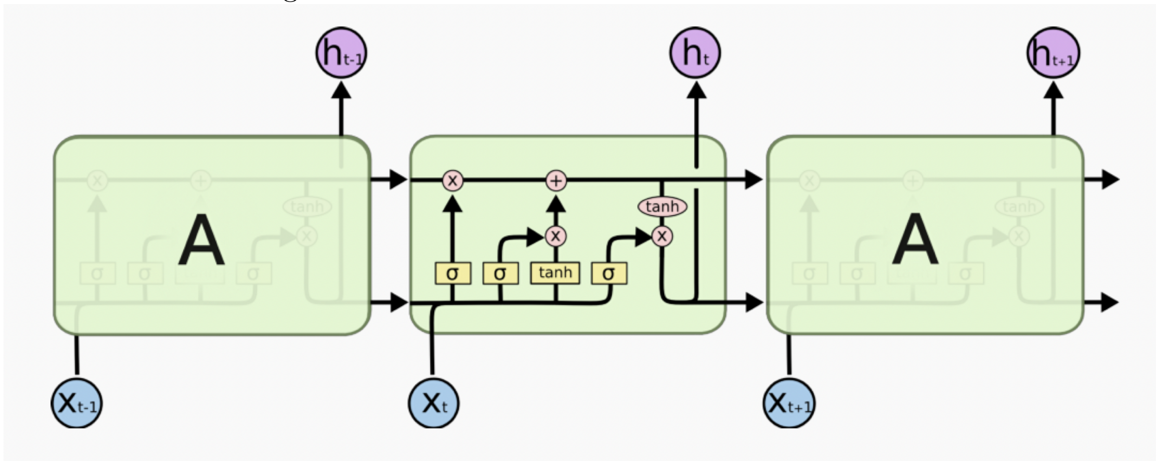
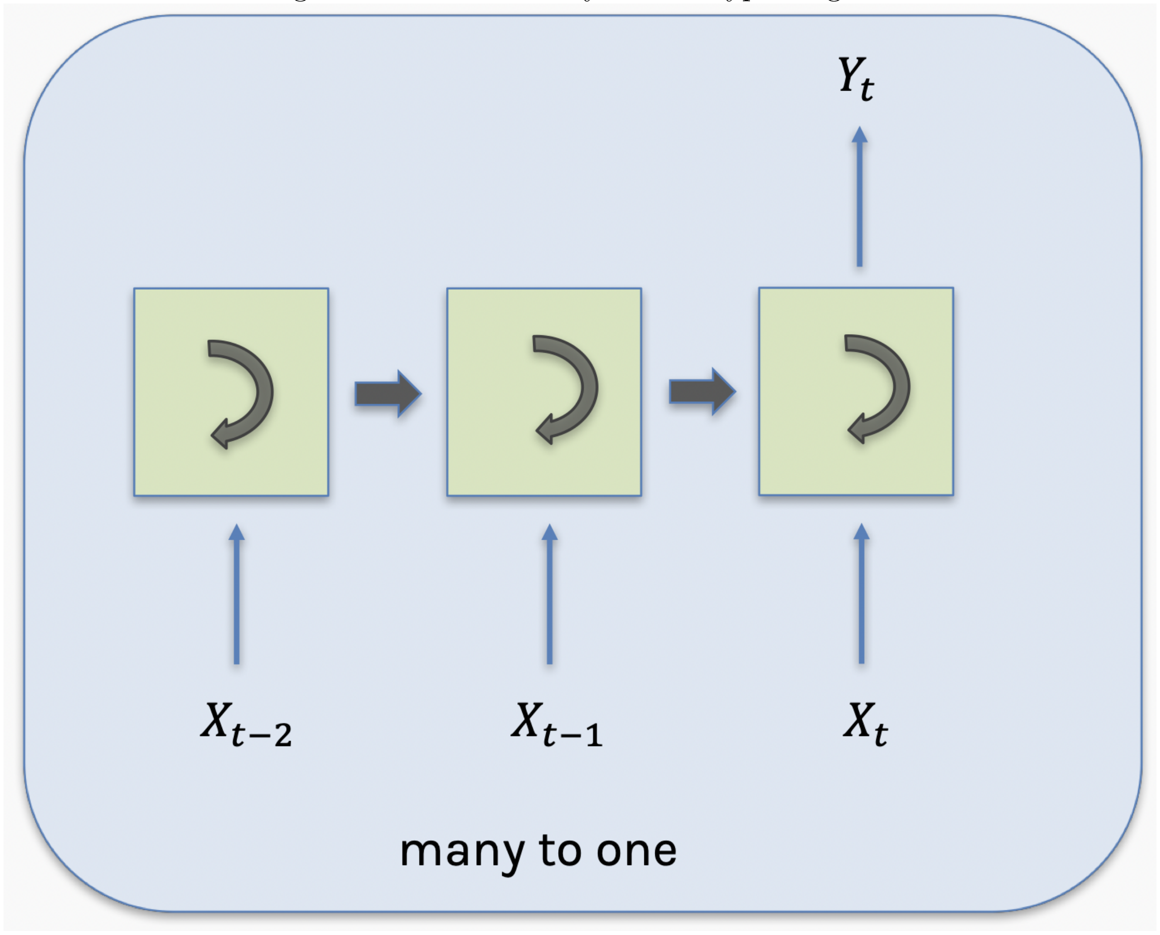


Figure 7.2: LSTM Many to One type diagram



combining the advantages of both AdaGrad and RMSProp (Kingma & Ba, 2017). This optimizer is also the default for Google's BERT language model.

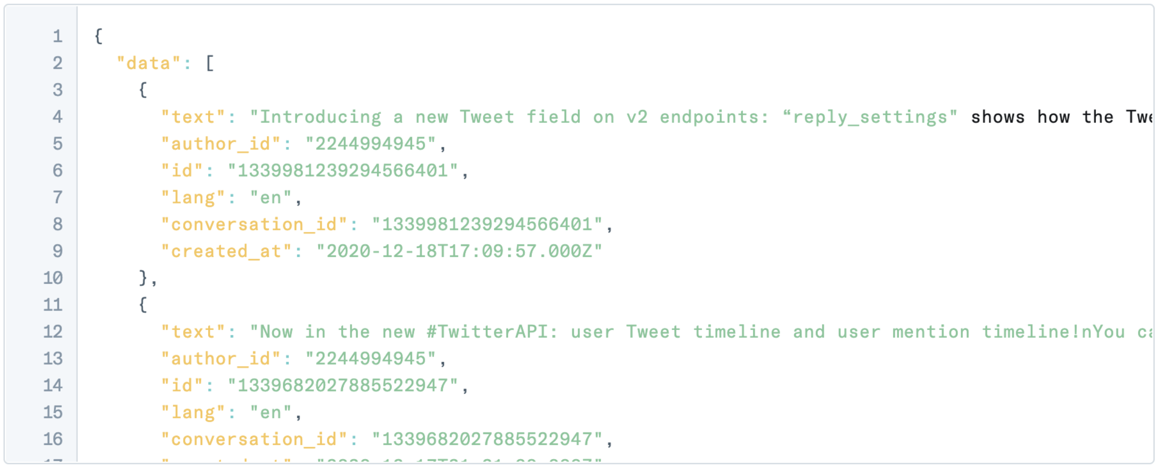
Chapter VIII.
Experimental Results

8.1. Big Data: Twitter

As previously described, we used IBM Cloud's NLP tools to perform sentiment analysis on fetched tweets. Still, some experimentation was needed to optimize the rest of our modeling output. The 30M tweets mentioned were split into weekly folders, each containing a JSON file for each player. These files were quite rarely empty; only 3-5 players had no tweets returned per week from roughly 300 players we searched for tweets for. The largest of these files was over 500MB, with thousands and thousands of Tweets retrieved, in this case for the Browns defense. As shown in table 6.2, the average number of tweets per player was never less than 3,000 tweets per player per week, with the max often over 100,000. By using file size as a proxy for the number of tweets, it is visible that the more "popular" a player, the more tweets there will be. This top-heavy distribution of tweets was not seen to be an issue, given the top-heavy nature of fantasy sports generally.

These JSON files were essentially JSON arrays, with each element representing a Tweet under Twitter's API definition, as shown in Figure 8.1. The search API and its full documentation can be found at https://developer.twitter.com/en/docs/twitter-api/tweets/search/api-reference/get-tweets-search-all.

The Twitter API has rate limits in place for both security of their data and to ensure their servers don't get overloaded, which drastically slowed down our acquisi-



```
{
  "data": [
    {
      "text": "Introducing a new Tweet field on v2 endpoints: “reply_settings" shows how the Twe
      "author_id": "2244994945",
      "id": "1339981239294566401",
      "lang": "en",
      "conversation_id": "1339981239294566401",
      "created_at": "2020-12-18T17:09:57.000Z"
    },
    {
      "text": "Now in the new #TwitterAPI: user Tweet timeline and user mention timeline!nYou ca
      "author_id": "2244994945",
      "id": "1339682027885522947",
      "lang": "en",
      "conversation_id": "1339682027885522947",
```

Figure 8.1: Twitter JSON Format

tion of tweets. Given a 10M tweet per month quota, this retrieval process was split up into three distinct chunks, with our script built such that it would resume from where it had last been paused. Each of the 10M tweet fetches we made in consecutive months took several days to complete.

Each request was limited to a maximum of 500 results. However, we found that the Twitter API had a considerably higher error rate when doing so and that 100 results per request was the sweet spot where Twitter's API finally started to error out only rarely. With a cap of 1 request per second, and 300 requests every 15 minutes, we were limited to retrieving 120,000 results per hour.

$$\frac{300requests}{15minutes} * \frac{60minutes}{1hour} * \frac{100results}{1request} = 120,000\frac{results}{hour}$$

To hit the 10M tweet cap, then, it typically took 80 hours, often more because of random errors that could throw off the long-running script, which would happen to coincide when Twitter was particularly active, or some big news event broke.

8.2. Natural Language Processing

While many fields could be of interest, the primary focus was entirely on the "text" field, which contains the Tweet content viewed on the platform, so text files were created that collated all tweet text together. These were then used as input for sentiment analysis, but not before performing preprocessing on these tweet dumps.

Our first attempt simply ran lemmatization over the raw collated tweet text and dumped that into IBM Cloud's sentiment analysis. This yielded the expected sentiment scores for each player as desired, which we could then use for the rest of our modeling.

However, upon further analysis, we discovered that there was still a lot of junk in the tweet text that IBM was analyzing by only doing lemmatization. We also learned just how many duplicates there were. Given that IBM's API only supports sentiment analysis in chunks of 50,000 characters, and some players had hundreds of thousands if not millions of characters of tweets, this indicated that information was likely not getting analyzed within this input.

We therefore de-duplicated tweets in preprocessing, noting the total count and unique count of tweets as features that could be useful in subsequent modeling steps, since the ratio between total and unique could potentially indicate something about a player's expected performance, for example, if a lot of people were re-tweeting (essentially duplicating) the same tweet.

In addition, several other standard NLP preprocessing steps were taken, given the high volume of punctuation and emojis within these tweets that were unlikely to provide value in sentiment analysis. So, in addition to lemmatization and de-duplication, we removed all emojis, tokenized all tweets, and removed punctuation, stop words, websites and email addresses, symbols, digits, and stopwords. Doing so reduced helped increase the amount of data processed by IBM Cloud's NLP API. As shown in the next section, this helped increase the predictive power of our regression model thanks to the improved sentiment scores output from the NLP performed by IBM Cloud.

There is, of course, room to explore further here, as we might have gotten too aggressive in removing characters to fit under the 50,000 character limit imposed by IBM's API. Similarly, we could have attempted to chain multiple requests to the NLP API in 50,000 character chunks, blending the results to achieve a single sentiment score per player, but this goes beyond our current scope.

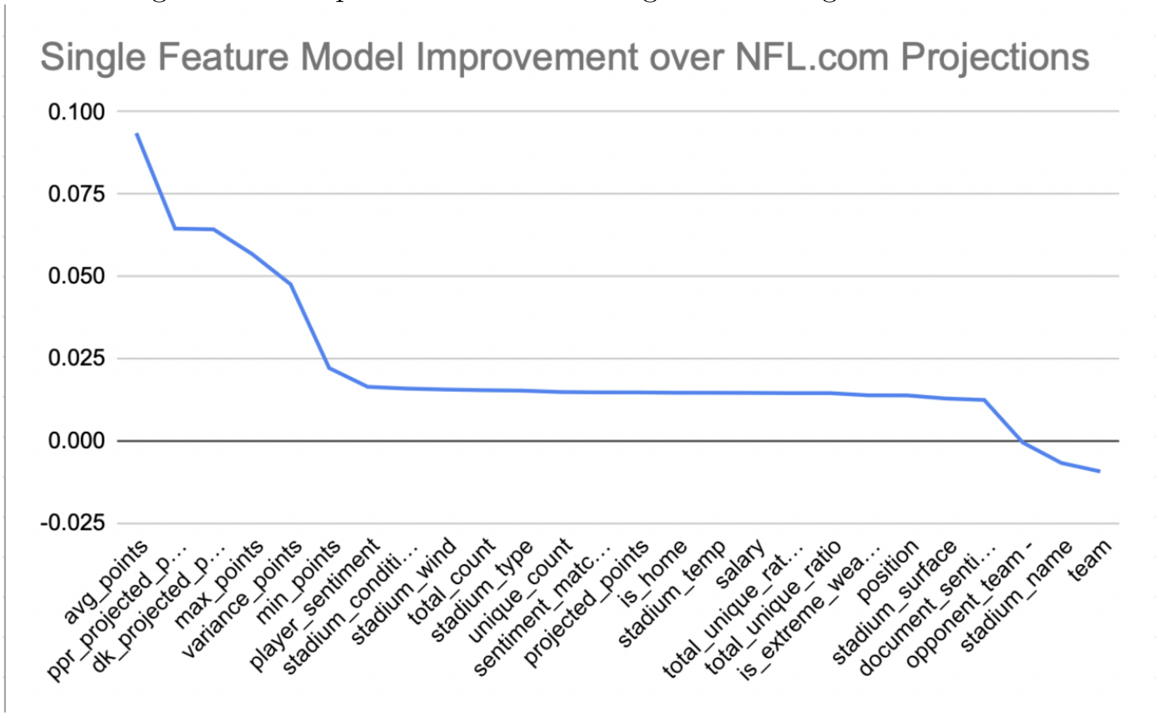
One of the engineered features, F_{10}, was shown to be of critical value as it helped reveal the difference between the document as a whole, as in the full content of tweets fetched for a player, and what IBM Cloud's API recognized as the player and the sentiment associated with that specific entity. The feature simply represents whether the two sentiment scores matched in sign and proved more valuable to our regression model than any other Twitter-derived feature we constructed.

8.3. Regression

We will leverage the first few weeks of the 2020 season to build trust around performance, without the need to retrieve additional data from previous seasons (Landers & Duperrouzel, 2019). We take two approaches in splitting data between train and test sets: (1) use all preceding weeks as train data, (2) use only preceding 4 weeks of data as training data. With 17 weeks in the 2020 season, this means there are 13 folds of data on which we perform testing and training.

Of the 26 features described, only a subset would prove valuable in improving predictive power. We evaluated predictive power by comparing the R^2 value of our regression output against the R^2 value of the NFL.com projections we use as an input

Figure 8.2: Graph of R^2 values for single feature regression models



feature.

Given the number of base features would make combinatorial exploration of all feature combinations computationally expensive, we took a manual approach to retain control of feature combinations to leverage our understanding of the game. We began by creating a regression model with each of the 26 features to establish a single feature baseline R^2 value that we could use to rank features by predictive power on an individual basis, shown in Figure 8.2.

A full combinatorial expansion of features was attempted but quickly ran into the combinatorial explosion. The 6-feature selection combinatoric expansion took over 18 hours on a high-end laptop (new M1 Max Macbook Pro).

From this, the initial model was selected to include all features with positive R^2 values, which included the majority of features. The feature set was then trimmed

1-Feature Models		
Feature	R^2	Diff to Overall
avg_points	0.6684	0.0934
ppr_projected_points	0.6396	0.0644
dk_projected_points	0.6394	0.0642
max_points	0.6319	0.0567
variance_points	0.6227	0.0475
min_points	0.5973	0.0221
player_sentiment	0.5915	0.0163
stadium_conditions	0.591	0.0158
stadium_wind	0.5907	0.0155
total_count	0.5905	0.0153
stadium_type	0.5904	0.0152
unique_count	0.59	0.0148
sentiment_matches	0.5899	0.0147
projected_points	0.5898	0.0146
is_home	0.5897	0.0145
stadium_temp	0.5897	0.0145
salary	0.5897	0.0145
total_unique_ratio_norm	0.5896	0.0144
total_unique_ratio	0.5896	0.0144
is_extreme_weather	0.589	0.0138
position	0.5889	0.0137
stadium_surface	0.588	0.0128
document_sentiment	0.5876	0.0124
opponent_team	0.5746	-0.0006
stadium_name	0.5684	-0.0068
team	0.5658	-0.0094

Table 8.1: Single Feature Regression Modeling Results

down to the set of $F_3, F_4, F_6, F_7, F_9, F_{10}, F_{11}, F_{20}, F_{22}, F_{24}, F_{26}$

This set of features yielded an R^2 improvement over NFL.com projections of 0.1308, 0.5752 vs. 0.4444, which is also 0.0820 greater than the R^2 of projections from FantasyData.io, which had an R^2 of 0.4932. This is a 29% increase in R^2 compared to NFL.com 17% increase compared to FantasyData.

This set of features was further validated by exploring model combinations, exploring all remaining combinations of two features in addition to locked model features. This progressively locked more and more features until improved R^2 stopped improving, which occurred when adding the 12^{th} feature to the model, leaving us with the 11 feature model as described above.

To be more precise, the method used to explore various feature combinations for these models was to set a specific set of features as "locked", and then use an iteration utility to create all 2 feature combinations for the remaining unlocked features, and then generate regression models with each of these feature combinations to find the best ones. We iteratively did this, gradually locking more features as they showed up repeatedly when reviewing results from the 2 feature model combinations. This allowed us to quickly explore the feature space and find improvements in R^2 without having to explore **all** feature combinations.

Further experimentation revealed the power of the included sentiment feature, where improvement over NFL.com projections decreased when skipping tweet preprocessing steps and only lemmatization was performed. We found the overall R^2 improvement of 0.1337, 2.2% higher than 0.1308 previously found. It would be interesting to see further variations of text preprocessing steps and their effect on the model, as these variations have shown an impact.

To further cross-validate our results, we removed features from each data source to evalute the best model with a limited set of features. When removing

N-Feature Models			
Features	Feature Count	R^2	Diff to Overall
F_{11}, F_{26}	2	0.6839	0.1087
F_{11}, F_{25}	2	0.6837	0.1085
F_{11}, F_{3}	2	0.6394	0.0978
F_{11}, F_{3}, F_{4}, F_{26}	4	0.6971	0.1219
F_{11}, F_{3}, F_{4}, F_{25}	4	0.6964	0.1212
F_{11}, F_{3}, F_{7}, F_{26}	4	0.6945	0.1193
F_{11}, F_{3}, F_{6}, F_{4}, F_{26}	5	0.7001	0.1249
F_{11}, F_{3}, F_{22}, F_{4}, F_{26}	5	0.6996	0.1244
F_{11}, F_{3}, F_{7}, F_{4}, F_{26}	5	0.6996	0.1244
F_{3}, F_{4}, F_{11}, F_{6}, F_{26}, F_{7}, F_{9}, F_{22}, F_{10}, F_{20}	10	0.706	0.1308
F_{3}, F_{4}, F_{11}, F_{6}, F_{26}, F_{7}, F_{9}, F_{22}, F_{27}, F_{20}	10	0.706	0.1308
F_{3}, F_{4}, F_{11}, F_{6}, F_{26}, F_{7}, F_{9}, F_{22}, F_{20}, F_{24}	10	0.706	0.1308
F_{3}, F_{4}, F_{11}, F_{6}, F_{26}, F_{7}, F_{9}, F_{22}, F_{10}, F_{20}, F_{27}	11	0.706	0.1308
F_{3}, F_{4}, F_{11}, F_{6}, F_{26}, F_{7}, F_{9}, F_{22}, F_{27}, F_{20}, F_{24}	11	0.706	0.1308
F_{3}, F_{4}, F_{11}, F_{6}, F_{26}, F_{7}, F_{9}, F_{22}, F_{20}, F_{24}, F_{27}	11	0.705	0.1307

Table 8.2: Combinatorial Expansion Result Highlight

FantasyData data, we found a 13 feature model as the one with the greatest R^2 improvement of 0.0977 (features: F_3, F_4, F_5, F_6, F_7, F_9, F_{10}, F_{11}, F_{16}, F_{20}, F_{22}, F_{24}, F_{26}).

When removing sentiment data from Twitter, we found a 10 feature model as the one with the greatest R^2 improvement of 0.0875 (features: F_3, F_4, F_{11}, F_{12}, F_{13}, F_{16}, F_{20}, F_{22}, F_{26}, F_{24})

When ONLY using data from Twitter, we surprisingly still got results that performed better than NFL.com projections! While not as large an R^2 improvement as the models mentioned above, just using features from Twitter led to an R^2 improvement over NFL.com projections of 0.0179 (features: F_3, F_5, F_6, F_7, F_8, F_9, F_{10}). While not as powerful a result as our other models, this provides a key insight: using just Twitter data, we can assemble projections for entities that perform similarly to professional projections. This could prove useful in other domains, such as stock

Positions		
Position	R^2	Diff to Overall
QB	0.663	0.0878
WR	0.5239	-0.0513
RB	0.5986	0.0234
TE	0.5002	-0.075
DEF	0.1096	-0.4656

Table 8.3: Positional Results Split Out

market projections, where a streamlined process could be created to project price movements from social media data alone.

Interestingly, when removing weather and stadium data, we found an 8-feature model with an improved R^2 of 0.1350, another 1% better than previously found. Features for that model: F_3, F_4, F_5, F_7, F_{10}, F_{11}, F_{24}, F_{26}.

Another interesting finding was that 4 weeks of data was the optimal look-back window for training, yielding the most significant improvement in R^2 compared to NFL.com projections. However, when using the whole season of data available in preceding weeks, weeks later in the season showed weaker projection accuracy. Further evaluation of the 4-week number could yield further improvements; different positions can potentially use other lookback windows. We break down the positional differences next.

8.3.1 Positional Differences

The analysis was then extended to review whether predictive power varied based on the player's position. This revealed drastic positional differences:

While most positions are roughly centered around the average R^2 for all positions, defenses have a very low R^2, indicating a different modeling strategy may be ideal for such a different player type. Given that all other positions represent an individual player, while the defense position represents an entire team, requiring a different modeling strategy seems intuitive. It is also interesting that quarterbacks and running backs are the only positions with better position-specific R^2 than average.

When connecting these results to the general media's focus when discussing NFL teams, their performance, and how that relates to the fantasy world, it's no surprise that the position that gets talked about the most is quarterbacks. What's also interesting, though, is that offensive lines, which make up almost half the offensive team but score no fantasy points directly, are also a critical factor. These players' performance is not directly modeled in the work that we have done, but they play a crucial role in the success of quarterbacks and every other offensive player. This suggests that using information about an offensive line's performance, such as their overall rank relative to other teams, could provide further projection power to aid in the projections of other fantasy players.

Similarly, defensive teams, which have very low R^2 values, are likely impacted by the matching offensive line they are playing against. A defense playing a weak offensive line is more likely to score more points since a weak offensive line will not protect their quarterback as effectively, and disruptions to the quarterback result in points for the defensive unit. Therefore, while there is potential to separate players by position to improve projections, offensive line data should be valuable for predictions for all positions.

8.4. Neural Network

To create an LSTM neural net, we first had to heavily manipulate our simple regression data frame into the necessary input format for a neural network. Precisely, we needed to create individual data frames per player, which revealed an uneven distribution of data points per player – some had 12 weeks of data, others only 8, and yet others 16. This was addressed by setting a threshold for the number of data points a player needed to be included, determined by the number of players it left included in the dataset. The value that worked best was 12, leaving roughly half of all players for the 2020 season (480 of 950). All players were then given 0 padding to reach 17 weeks, such that all player data frames were symmetric.

Furthermore, encoding had to be done before this separation to ensure all categories were considered during 1-hot encoding and all values considered within the scaled distribution. Categorical variables were one-hot encoded, and numerical variables were scaled using the standard scaler (std-mean scaling), leaving us with 164 columns.

An initial batch was selected randomly, with 32 players, over 4 weeks, providing an input array of size 32x4x164. Hyper-parameters such as # of neurons and # of epochs were tested linearly, neuron-count not showing any material difference in results through variation, and epoch count showing a leveling off in the 20-50 epoch range for most models created.

The rest of the player set was then tested against this neural net, providing a projection for each player and therefore an R^2 against their actual scores, which were not shown to be better than professional projections.

8.5. Integer Programming

With projections in hand from our regression model, which outperformed existing NFL.com projections, unlike our neural network-based projections, we then turn to an integer programming algorithm to create lineups. While an existing IP solver could be leveraged by formulating constraints in the solver's constraint language, a custom solver was instead explicitly built for this purpose. By doing so, our solver had mechanisms to lock and exclude players manually as part of the lineup creation process, which provided easy ways to test variations to the input set of players to observe the variation in lineups.

This custom solver was coded as a dynamic programming solution to the knapsack problem with a few extra custom constraints, such as the positional constraints set by DFS platforms. It took a data frame of players with their salaries, projected points, and positions and output the top 10 lineups. On my personal laptop, this typically completed in under 30 seconds, especially after discovering that the bottom half of players available in any given week were duds and unlikely to be worthy of making a 9-player lineup. We significantly accelerated the solver by excluding the bottom half of players by projected points.

Admittedly, this solver is not provably optimal, and this can be seen by the non-idempotent output given a shuffled input set of players. Shuffling the input player list and sorting by different factors (projections, ratio of projections to salary, salary) revealed that the best lineups were output from an initially sorted list by projection. Existing available IP solvers likely do this input shuffling automatically. From our findings, using a sorted input produced the best results and was far less computationally expensive than iterating over multiple input orderings.

To evaluate the effectiveness of the regression-based projections, we tasked the solver with creating lineups for both sets of projections for the 2020 season, evaluated against actual DraftKings DFS results to determine the success of lineups in relation to the best lineups created for that week. Note that only weeks 5-17 were evaluated since the regression model uses the past 4 weeks of data to train on, so week 5 was the first week for which we could generate projections using this model.

Using the NFL.com projections, 13 lineups averaged a projected outcome of 146.8 points against an actual outcome of 122.1 points. This meant an average percentile finish in weekly $1 million DraftKings DFS contests of 34.57%, earning an average of \$19.23 per week (on a \$20 entry fee). While not bad, these projections fed through our solver earned just under the break-even point and would not be deployed further.

Using our regression model's projections, 13 generated lineups averaged a projected outcome of 156.4 points against an actual outcome of 187.0 points. This meant an average percentile finish of 6.98%, earning an average of \$246.54. This is an outstanding 1100% potential return on \$260 (13 * 20) in entry fees.

Given the increased accuracy of our regression model's projections, it is not surprising that these performed better when assembled into a lineup. Still, the magnitude of improvement is far more substantial than expected for a marginal improvement in accuracy. This would suggest that further accuracy improvements, as small as they may be, could provide a significant advantage in lineup creation for DFS contests.

Chapter IX.

Conclusion

9.1. Natural Language Processing

Acquiring, processing, and storing 30M+ tweets was performed on a single laptop; however, due to the rate-limiting imposed by Twitter's API, there are inherent restrictions on what can be done with Twitter data. Nevertheless, all tweets were successfully processed and then analyzed through IBM Cloud's sentiment analysis, providing valuable features for our regression modeling. Impressively, IBM Cloud's NLP tools, which leverage a custom trained BERT-based model, recognized the player entity in all tweets, providing both a document and entity sentiment score for each player.

Our processing approach to the 30M+ tweets was admittedly very rough, paired with search queries that likely included a lot of noise. Nevertheless, there is ample opportunity to add more validation to input tweets for each player to ensure they are of a certain quality and are referencing the right player. This is especially true for the DEF position, for which the ”player name” search query returned results for the whole team and not just the defense.

Several other features could be extracted from Twitter, though. First of all, many players are on Twitter themselves, and many are pretty active. Given the highly emotional nature of professional sports, extracting sentiment from a player's tweets could provide insights into the mental state of that player that our model could then leverage. Furthermore, many professionals in the sports industry often provide critical updates but aren't players. While their input has likely been codified into professional website projections like NFL.com, extracting sentiment from these pros has the opportunity to be a valuable feature.

Outside of purely extracting sentiment, specific keywords could be monitored to identify injuries to players. For example, if many tweets have the word "OUT" and a player's name, the player is likely injured. Furthermore, it's possible that injury is late-breaking and therefore not considered within the professional website projections. This would then be a precious source of players to exclude from modeling should they not be playing.

Furthermore, as was shown by our model that only used features derived from Twitter data, our modeling approach can potentially prove valuable in contexts outside of fantasy sports where only tweets are readily available. This suggests that for any entity for which projections are helpful, if there are tweets about that entity, our modeling approach could be leveraged to create projections for those entities, such as stocks or companies to invest in.

9.2. Modeling: Regression & Neural Networks

While we were able to successfully create improved projections via our basic regression model, there are countless other models we could have attempted that likely could result in similar, if not better, results. Neural networks were tried, and while we successfully created an LSTM that generated realistic projections, the performance of these projections was still measurably worse than NFL.com projections. That does not mean that LSTMs can't work for these types of projections, though, and indicate that we likely needed more data for an LSTM and that perhaps a simpler neural network may have performed better. An example of such a more straightforward model would be an Auto-Regressive neural network, which could take advantage of the time-series properties of LSTMs without needing as much input data.

We could improve our projections for our regression model, boosted decision trees, random forests, or other boosting mechanisms. These have been used by others assembling fantasy projections with success.

In addition, it is possible that splitting up players by position to create separate models is advantageous for all kinds of models, given the significant variation in positional projection accuracy we found. Such an approach could quickly exacerbate the problem of scarce data, though, which could be compensated by using more historical data. That said, our regression model performed best with less data vs. more, with our best regression model only requiring the previous 4 weeks of data.

Additional features could also be engineered and experimented with, in addition to the aforementioned potential improvements to twitter-based features. Specifically, the avg_points feature proved highly valuable, and this simply represented that player's average points over the past 4 weeks. We did not attempt variations on this number of weeks included, though, nor did we have multiple average values, which could provide further insight for our model as to the trajectory of this player of the past few games. Additionally, given the nature of sports, there is a lot of "momentum" that can be built up by a team. As such, a trailing 2-week, 3-week, and 4-week average may show differences that highlight that momentum for both a team and individual players, which could further improve projections.

9.3. Integer Programming

Creating lineups via integer programming is effective, but several aspects could be improved or modified. First, creating a custom integer programming solver for fantasy football provided additional flexibility in debugging and experimentation but potentially did not yield the most optimal results. Given the prevalence of commercial and open-source linear programming solvers, employing one of these could have reduced the scope of new code writing while providing stronger guarantees around the output.

Using such an online solver would have likely made it easier to create multiple lineups while limiting their correlation. Unfortunately, this proved more difficult than anticipated within the created IP solver and prevented us from attempting this lineup portfolio creation that would provide the best chances of winning a DFS contest.

Lastly, it is possible that a neural network could be trained directly on previously completed DFS contests and their entries (and results) to bypass the majority of this thesis' data collection and modeling steps. This is likely to be quite tricky to set up, though, given the vast number of players who are ”new” every game week, with some of these players outperforming expectations and therefore being key to consider for lineups, but for which a neural network trained on previous contests would have no knowledge of.

Aside from these potential further research areas, we have shown that lineup creation when using improved projections can yield positive outcomes in the Daily Fantasy Football game space. The potential profitability of our model illustrates the financial potential in a limited context, which could be further broadened. However, in the context of fantasy football, these results also help show how winning in fantasy football is not random, which is what one would expect should the game be a gamble. Instead, we see that a positive expected value can be achieved through careful, deliberate selection, indicating fantasy football is more a game of skill than pure gambling. The stochastic nature of sports will always leave things up to chance, though, which perhaps is by default how gambling is defined, but unlike the lottery, there are clear actions one can take to improve one's chances of winning.

Especially in the current explosion of fantasy football and related advertising, there is a perception that fantasy football is gambling. It should be considered such for the vast majority of players since they do not employ the techniques discussed in this research. However, there is a game of skill to be played for the few who use these techniques, with potentially large winnings available to the best out there.

9.4. Final Remarks

Overall, our method proved effective to improve professional site projections and generate strong lineups to compete in daily fantasy *picking winners* contests. The key to our success was the combination of state-of-the-art natural language processing models to provide sentiment analysis and ensemble models using that sentiment data to create projections. With so many pieces to our system built for this research, countless variations could be attempted going forward. Of those pieces, the BERT NLP model proved quite effective in providing sentiment classification, but perhaps a BERT model specifically pre-trained on sports lexicons may perform even better. Our linear regression model uses a prevalent statistical modeling approach, and the projections it output showed a 30% improvement over existing professional projections.

Another step for this work is to engineer the data collection process to happen live quickly. Then, these projections can be leveraged in actual contests instead of just retrospectively on previously conducted contests. Before that can happen, 2021 season data should be assembled to validate these models further. Hopefully, this analysis also shows how such an approach could be used in other domains, as mentioned. Using our newly developed method combined with faster and more efficient computational resources makes it possible to compute models on a single multi-core machine that used to require supercomputers. There is simply a need for more research in this area, which we hope will occur based on these findings, whether for use in fantasy football or not. We live in a vast acceleration of our ability to capture, analyze, and model data. We'll need as much research into these methods as possible to understand and expand on these models in the future.

Chapter IX.
References

Aloufi, S. & Saddik, A. E. (2018). Sentiment identification in football-specific tweets. *IEEE Access*, 6, 78609–78621.

Bar-Haim, R., Bhattacharya, I., Dinuzzo, F., Saha, A., & Slonim, N. (2017). Stance classification of context-dependent claims. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers* (pp. 251–261). Valencia, Spain: Association for Computational Linguistics.

Borghesi, R. (2007). The home team weather advantage and biases in the nfl betting market. *Journal of Economics and Business*, 59(4), 340–354.

Cheema, A. (2020). Analyzing the effect of weather in the nfl. https://www.thespax.com/nfl/analyzing-the-effect-of-weather-in-the-nfl/. Accessed: 2021-12-05.

De Mulder, W., Bethard, S., & Moens, M.-F. (2015). A survey on the application of recurrent neural networks to statistical language modeling. *Computer Speech Language*, 30(1), 61–98.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert:

Pre-training of deep bidirectional transformers for language understanding. https://arxiv.org/abs/1810.04805.

DraftKings (2021). Rules & scoring. https://www.draftkings.com/help/rules/1/1. Accessed: 2021-11-30.

Dunnington, N. (2015). Fantasy football projection analysis. https://economics.uoregon.edu/wp-content/uploads/sites/4/2015/03/Dunnington_Thesis_2015.pdf. Accessed: 2021-12-10.

Fischer, K., Haucap, J., & Haucap, J. (2020). Betting market efficiency in the presence of unfamiliar shocks: The case of ghost games during the covid-19 pandemic. https://ssrn.com/abstract=3692914. Accessed: 2021-12-10.

FSGA (2021). Who we are. https://thefsga.org/about-the-fsga/. Accessed: 2021-11-25.

Goff, B. (2013). The $70 billion fantasy football market. https://www.forbes.com/sites/briangoff/2013/08/20/the-70-billion-fantasy-football-market/?sh=72b84595755c. Accessed: 2021-12-10.

Google (2020). Bert readme, google research. https://github.com/google-research/bert/blob/master/README.md. Accessed: 2021-12-05.

Goyal, S., Choudhury, A. R., Raje, S. M., Chakaravarthy, V. T., Sabharwal, Y., & Verma, A. (2020). Power-bert: Accelerating bert inference via progressive word-vector elimination. https://arxiv.org/abs/2001.08950.

Gratch, J., Lucas, G., Malandrakis, N., Szablowski, E., Fessler, E., & Nichols, J. (2015). Goaalll!: Using sentiment in the world cup to explore theories of emotion. In

2015 International Conference on Affective Computing and Intelligent Interaction (ACII) (pp. 898–903).

Green, C. (2014). 'wink': Wilfred 'bill' winkenbach invented fantasy football way back in 1962 with gopppl in oakland. https://web.archive.org/web/20150929163914/http://www.newsnet5.com/sports/wink-wilfred-bill-winkenbach-invented-fantasy-football-way-back-in-1962-with-gopppl-in-oakland. Accessed: 2021-12-01.

Harville, D. (1980). Predictions for national football league games via linear-model methodology. *Journal of the American Statistical Association*, 75(371), 516–524.

Haugh, M. B. & Singal, R. (2019). How to play fantasy sports strategically (and win). http://dx.doi.org/10.2139/ssrn.3393127. Accessed: 2021-12-01.

Hirsh, S. G., Anderson, C., & Caselli, M. (2012). The reality of fantasy: uncovering information-seeking behaviors and needs in online fantasy sports. *CHI '12 Extended Abstracts on Human Factors in Computing Systems.*

History.com (2009). Professional football is born. https://www.history.com/this-day-in-history/professional-football-is-born. Accessed: 2021-12-03.

Hunter, D. S., Vielma, J. P., & Zaman, T. (2019). Picking winners in daily fantasy sports using integer programming. https://arxiv.org/abs/1604.01455.

IBM (2021). Text sentiment classifier, ibm developer staff. https://developer.ibm.com/exchanges/models/all/max-text-sentiment-classifier/. Accessed: 2021-12-10.

Islam, M. R. & Zibran, M. F. (2017). Leveraging automated sentiment analysis in software engineering. In *Proceedings of the 14th International Conference on Mining Software Repositories*, MSR '17 (pp. 203–214).: IEEE Press.

Kampakis, S. & Adamides, A. (2014). Using twitter to predict football outcomes. https://arxiv.org/pdf/1411.1243.pdf.

Karp, R. M. (1975). On the computational complexity of combinatorial problems. *Networks*, 5(1), 45–68.

King, N. & Leboulluec, A. (2017). Projecting a quarterback's fantasy football point output for daily fantasy sports using statistical models. *International Journal of Computer Applications*, 164, 22–27.

Kingma, D. P. & Ba, J. (2017). Adam: A method for stochastic optimization. https://arxiv.org/abs/1412.6980.

Landers, J. R. & Duperrouzel, B. (2019). Machine learning approaches to competing in fantasy leagues for the nfl. *IEEE Transactions on Games*, 11(2), 159–172.

Lewis, M. (2004). *Moneyball*. WW Norton.

Lutz, R. (2015). Fantasy football prediction. https://arxiv.org/abs/1505.06918.

Mancuso, J. (2019). 'football weather'-diving into the effects of weather on nfl qb performance. https://towardsdatascience.com/football-weather-diving-into-the-effects-of-weather-on-nfl-qb-performance-f0edb420623d. Accessed: 2021-12-30.

Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5(4), 1093–1113.

Newell, S. (2017). Optimizing daily fantasy sports contests through stochastic integer programming. http://hdl.handle.net/2097/35393. Accessed: 2021-12-30.

PFR (2021). The p-f-r win probability model. https://www.pro-football-reference.com/about/win_prob.htm. Accessed: 2021-12-10.

Rychlak, R. J. (1992). Lotteries, revenues and social costs: A historical examination of state-sponsored gambling. *Boston College Law Review*, 34(1).

Schalter, T. (2013). How do nfl fans really affect games? https://bleacherreport.com/articles/1885183-how-does-the-12th-man-really-affect-nfl-game. Accessed: 2021-12-15.

Schalter, T. (2021). Fans are back at nfl games. but home-field advantage isn't (yet). https://fivethirtyeight.com/features/fans-are-back-at-nfl-games-but-home-field-advantage-isnt-yet/. Accessed: 2021-12-30.

Sharp, G., Brettenny, W., Gonsalves, J., Lourens, M., & Stretch, R. (2011). Integer optimisation for the selection of a twenty20 cricket team. *Journal of the Operational Research Society*, 62.

Sinha, S., Dyer, C., Gimpel, K., & Smith, N. A. (2013). Predicting the nfl using twitter. https://arxiv.org/abs/1310.6998.

Stern, H. (1991). On the probability of winning a football game. *The American Statistician*, 45(3), 179–183.

Surowiecki, J. (2004). *The Wisdom of Crowds: Why the Many are Smarter Than the Few and how Collective Wisdom Shapes Business, Economies, Societies, and Nations*. Doubleday.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. https://arxiv.org/abs/1706.03762.

Wong, K. (2015). The fantasy sports industry, by the numbers. https://www.nbcnews.com/business/business-news/fantasy-sports-industry-numbers-n439536. Accessed: 2021-12-10.

Xu, C. & Yu, Y. (2015). Measuring nba players' mood by mining athlete-generated content. In *2015 48th Hawaii International Conference on System Sciences* (pp. 1706–1713).

Zhang, Q., Zhang, X., Hu, H., Li, C., Lin, Y., & Ma, R. (2021). Sports match prediction model for training and exercise using attention-based lstm network. *Digital Communications and Networks*.

Årup Nielsen, F. (2011). A new anew: Evaluation of a word list for sentiment analysis in microblogs. https://arxiv.org/abs/1103.2903.

Özlü, O. & Sokol, J. (2016). An optimization approach to designing a baseball scout network. *European Journal of Operational Research*, 255(3), 948–960.