

Multi-Class Classification of Bets and Parlays for DraftKings Sportsbook

Major Qualifying Project

Advisor:

D. Richard Brown III

Written By:

Grace Casey

Ryan Dieselman

Jack Fredo

Jackson Lombardi

Cameron Tomko



WPI

Submitted to the Faculty of the Worcester Polytechnic Institute in partial fulfillment of the requirements for the Degree of Bachelor of Science in Data Science. This report represents the work of one or more WPI undergraduate students submitted to the faculty as evidence of completion of a degree requirement. WPI routinely publishes these reports on the web without editorial or peer review.

August 2022 - April 2023

Abstract

In this Major Qualifying Project (MQP), the team partnered with DraftKings Sportsbook to generate user-specific single bet and parlay predictions. To accomplish this, the team utilized clustering algorithms and evaluation methods to group users together. Multi-class classification algorithms including Linear Support Vector, Naïve Bayes, Stochastic Gradient Descent, and a neural network were then used to predict the most likely associations between bets/parlays and groups of users from the best clustering algorithm. The neural network classifier provided the best prediction performance and significantly outperformed random classification as well as classification to the most popular cluster. This project's goal was to help DraftKings create a more personalized experience for their users by more accurately associating potential bets and parlays with each user.

Executive Summary

The sports betting industry has seen significant growth in recent years, with DraftKings emerging as a prominent player in the market. As sports betting becomes increasingly popular, there is a growing demand for personalized experiences to enhance the user betting experience. This goal of this project was to develop personalized recommendations for bets and parlays for users of DraftKings sportsbook.

While DraftKings sportsbook offers almost endless options for individual bets, they also offer parlays, which involve betting on multiple outcomes that must all occur for the bettor to win. With parlays, users have the opportunity to wager a small amount of money in the hopes of winning big if all the outcomes in their parlay are successful. If the first event in a parlay hits, the earnings are carried over and wagered in the next event, and so on, creating the potential for substantial payouts.

Currently, on the DraftKings platform, the same parlays are recommended for every user, without taking into consideration individual user preferences. The goal of this project was to utilize user betting behavior data to generate personalized parlay and individual bet predictions that cater to each user's interests, thereby enhancing the DraftKings betting experience. The team conducted research to explore different approaches and ultimately decided to develop a multi-class classification model that categorizes bets into clusters of users with similar betting tendencies.

The initial step in this process was to develop an accurate method of grouping users together based on their preferences. The team explored various clustering techniques, including K-means clustering, self-organized feature map for simple clustering (SOMSC), clustering categorical data using summaries (CACTUS), spectral clustering, and hierarchical clustering. The performance of these techniques was evaluated using Silhouette Coefficient, Calinski-Harabasz Index, and Davies-Bouldin Index. Evaluating each cluster method with these metrics led to the conclusion that spectral clustering was the most successful technique. The clusters formed by spectral clustering were then fixed and kept static for all remaining steps in the team's analysis.

Term Frequency Inverse Document Frequency (TF-IDF) was then applied on these clustered users to determine the weights of the words (bet information) within each cluster. Several classification algorithms were then tested to classify individual bets to specific clusters, including multinomial naive bayes (MNB), stochastic gradient descent (SGD), linear support vector classifier (Linear SVC), deep neural networks (DNN), and long short-term memory neural network (LSTM NN). Once the feasibility of the classification methodology was confirmed for individual bets, the team repeated these steps to classify two-leg parlays and all-leg parlays.

Similar to clustering algorithms, the classification algorithms needed a method for evaluating and comparing their performances. Metrics including accuracy, precision, recall, and F1 score were used to evaluate classification performance. The DNN model was found to be the most accurate in classifying bets to clusters for individual bets, two-leg parlays, and all-leg parlays, with accuracy scores of 0.540, 0.680, and 0.650, respectively. These numbers showed significant improvement compared to randomly assigning a bet to a cluster (accuracy score = 0.167 for all three datasets) or randomly assigning bets to the most popular cluster (accuracy score = 0.261 average across all three datasets). Implementing this model into DraftKings' current platform would enhance the user betting experience by providing more personalized bet and parlay recommendations.

Acknowledgements

We would like to express our gratitude to all the individuals who have contributed to the success of this project. Their invaluable support and guidance have made this project possible, and we are deeply thankful for their assistance.

We would like to thank our advisor Professor Brown for his unwavering commitment to providing guidance in all aspects of the project, his continuous support, and valuable insights. His dedication has been critical to the successful completion of our MQP report.

We would like to thank Andrea Mock and her team at DraftKings for their valuable input, patience, and willingness to help. We are immensely grateful for the opportunity to work with such experienced professionals and for their guidance in both the data science and business aspects of our project work.

We would like to thank DraftKings, our sponsor, who has provided generous support to WPI for the last five years including five sponsored WPI MQPs. We are grateful to DraftKings for providing invaluable opportunities for WPI students to gain real-world experience and we appreciate their enthusiastic support of this project.

We would like to thank the Faculty of the WPI Data Science department for their support, guidance, and additional help throughout the project. Their contribution has been instrumental in the success of this project.

We are truly grateful to all the people who have supported us throughout this project and helped us achieve our goals.

Table of Contents

<i>Abstract</i>	<i>ii</i>
<i>Executive Summary</i>	<i>iii</i>
<i>Acknowledgements</i>	<i>v</i>
<i>Table of Contents</i>	<i>vi</i>
<i>List of Figures</i>	<i>ix</i>
<i>List of Tables</i>	<i>xi</i>
<i>Table of Acronyms</i>	<i>xii</i>
1. Introduction	1
2. Background	3
2.1 Sports Betting Basics	3
2.1.1 Odds.....	3
2.1.2 Types of Bets	4
2.1.3 Parlays	7
2.2 DraftKings in the Sports Betting Industry	8
2.2.1 Origin and Growth.....	8
2.2.2 Current Platforms.....	9
2.3 Dataset and Features	9
2.3.1 Data Dictionary.....	10
2.4 Machine Learning Techniques	13
2.4.1 Basics of Machine Learning	14
2.4.2 Introduction to Multi-Class Classification	14
2.5 Natural Language Processing	15
2.5.1 Term Frequency- Inverse Document Frequency (TF-IDF)	15
2.5.2 Jaccard Similarity and Jaccard Distance	16
2.6 Clustering Techniques	17
2.6.1 Elbow Curve Method.....	18
2.6.2 K-means Clustering.....	18
2.6.3 Self-Organized Feature Map for Simple Clustering (SOMSC)	19
2.6.4 Clustering Categorical Data Using Summaries (CACTUS).....	19
2.6.5 Spectral Clustering.....	20
2.6.6 Hierarchical Clustering	20
2.7 Clustering Evaluation Methods	21
2.7.1 Silhouette Coefficient.....	21
2.7.2 Calinski-Harabasz Index.....	22
2.7.3 Davies-Bouldin Index.....	22
2.8 Classification Algorithms	22
2.8.1 Naive Bayes Classifier	22

2.8.2 Stochastic Gradient Descent Classifier	23
2.8.3 Linear Support Vector Classification	23
2.8.4 Neural Networks	24
2.9 Evaluation Metrics	24
2.9.1 Accuracy Score	25
2.9.2 Precision, Recall, and F1 Score	26
3. Methodology	28
3.1 Exploratory Data Analysis.....	29
3.2 Cleaning and Preparing the Data	33
3.3 Feature Engineering and Selection	35
3.3.1 New Features	35
3.3.2 Feature Selection.....	37
3.3.3 Bet Sentence Creation.....	39
3.4 Cluster Development	40
3.4.1 Mode Bet of the Users	40
3.4.2 One Hot Encoding.....	41
3.4.3 Summary Matrix.....	42
3.4.4 Elbow Curve Method.....	43
3.4.5 Evaluation and Selection of Clustering Methods	44
3.5 Cluster Evaluation.....	44
3.6 TD-IDF and Train/Test Split	45
3.7 Classification Algorithms Implementation.....	46
3.7.1 Sklearn	46
3.7.2 TensorFlow	47
3.8 Implementing Evaluation Metrics	48
4. Results.....	50
4.1 Cluster Exploration	50
4.1.1 Determination of Optimal Number of Clusters and Clustering Algorithm	50
4.1.2 Exploratory Data Analysis on Bet Distribution in Clusters	51
4.2 Multi-Class Classification Models for Bet and Parlay Association.....	58
4.2.1 Baseline Models	59
4.2.2 Multinomial Naive Bayes (MNB) Model.....	60
4.2.3 Stochastic Gradient Descent (SGD) Model.....	66
4.2.4 Linear Support Vector Classifier (LSVC) Model	72
4.2.5 Deep Neural Network (DNN) Model	78
4.2.6 Long Short-Term Memory Neural Network (LSTM NN) Model	82
4.3 Summary of Results.....	85
5.0 Recommendations and Conclusions	86
5.1 Implementation	86

5.2 Recommendations	89
5.3 Conclusions	90
<i>Appendix A – Summary of Platforms Offered by DraftKings.....</i>	<i>91</i>
<i>Appendix B – Classification Report of Random Baseline.....</i>	<i>92</i>

List of Figures

Figure 1: Example from the DraftKings website showing spread NBA bet.....	5
Figure 2: Example from the DraftKings website showing a total O/U NBA bet. Circled in red is the bet for the over/under of this Bucks vs. Grizzlies game.	6
Figure 3: Example from the DraftKings website of a player passing touchdowns prop event occurring in an NFL game.	6
Figure 4: Example of a parlay bet slip on DraftKings.	8
Figure 5: Example of a four-leg parlay in the dataset.	13
Figure 6: Example of a singular bet in the dataset.	13
Figure 7: Sample of what a user summary matrix would look like between users. The column/row labels 0-4 represent five different unique users.	20
Figure 8: Demonstration of RNN using sequential data.	24
Figure 9: Confusion matrix diagram (Nickolas, 2021).	25
Figure 10: How to calculate accuracy (Kanstrén, 2021).	26
Figure 11: Process flow diagram of project.	28
Figure 12: Boxplot of how many bets each user placed.	29
Figure 13: Comparison of parlay to individual bets in the data set.	30
Figure 14: Distribution of number of legs per parlay.	31
Figure 15: Distribution of the bets per sport when exploring the dataset of all the bets.	32
Figure 16: Depicts the total number of bets per sport per month.	32
Figure 17: Distribution of all the different sports in the dataset before the removal of the less popular sports.	35
Figure 18: Example of six-leg parlay in dataset with missing values for the ODDSAMERICAN column.	38
Figure 19: Snippet of summary matrix for all users.	43
Figure 20: K-means clustering elbow curve used to determine optimal K number of clusters (K=6).	44
Figure 21: 80/20 train test split by each cluster.	45
Figure 22: Distribution of number of individual bets within each cluster.	52
Figure 23: Heatmap of the most correlated words in each cluster.	53
Figure 24: The number of clusters each bet appears in.	54
Figure 25: Distribution of number of two-leg parlays in each cluster.	55
Figure 26: Number of clusters two-leg parlays exist in.	56
Figure 27: Distribution of all-leg parlays in each cluster.	57
Figure 28: Number of clusters all-leg parlays exist in.	58
Figure 29: Classification report of random baseline model for individual bets.	60
Figure 30: Training classification report for individual bets.	61
Figure 31: Testing classification report for individual bets.	61
Figure 32: Training confusion matrix for individual bets.	62
Figure 33: Testing confusion matrix for individual bets.	62
Figure 34: Training classification report for two-leg parlays.	63
Figure 35: Testing classification report for two-leg parlays.	63

Figure 36: Training confusion matrix for two-leg parlays.....	64
Figure 37: Testing confusion matrix for two-leg parlays.	64
Figure 38: Training classification report for all-leg parlays.	65
Figure 39: Testing classification report for all-leg parlays.	65
Figure 40: Training confusion matrix for all-leg parlays.....	66
Figure 41: Testing confusion matrix report for all-leg parlays.	66
Figure 42: SGD training classification report for individual bets.....	67
Figure 43: SGD testing classification report for individual bets.	67
Figure 44: SGD training confusion matrix for individual bets.	68
Figure 45: SGD testing confusion matrix for individual bets.	68
Figure 46: SGD training classification report for two-leg parlays.	69
Figure 47: SGD testing classification report for two-leg parlays.	69
Figure 48: SGD training confusion matrix for two-leg parlays.....	70
Figure 49: SGD testing confusion matrix for two-leg parlays.....	70
Figure 50: SGD training classification report for all-leg parlays.	71
Figure 51: SGD testing classification report for all-leg parlays.	71
Figure 52: SGD training confusion matrix for all-leg parlays.....	72
Figure 53: SGD testing confusion matrix for all-leg parlays.....	72
Figure 54: LSVC training classification report for individual bets.	73
Figure 55: LSVC testing classification report for individual bets.	73
Figure 56: LSVC training confusion matrix for individual bets.....	74
Figure 57: LSVC testing confusion matrix for individual bets.....	74
Figure 58: LSVC training classification report for two-leg parlays.	75
Figure 59: LSVC testing classification report for two-leg parlays.	75
Figure 60: LSVC training confusion matrix for two-leg parlays.....	76
Figure 61: LSCV testing confusion matrix for two-leg parlays.	76
Figure 62: LSVC training classification report for all-leg parlays.	77
Figure 63: LSVC testing classification report for all-leg parlays.	77
Figure 64: LSVC training confusion matrix for all-leg parlays.....	78
Figure 65: LSVC testing confusion matrix for all-leg parlays.	78
Figure 66: Deep neural network model summary for all bets.	79
Figure 67: Deep neural network training accuracy for all individual bets.	79
Figure 68: Deep neural network model summary for two-leg parlays.	80
Figure 69: Deep neural network training accuracy for two-leg parlays.	80
Figure 70: Deep neural network model summary for all-leg parlays.	81
Figure 71: Deep neural network training accuracy for all-leg parlays.	81
Figure 72: LSTM model summary for all bets.	82
Figure 73: LSTM neural network training accuracy for all bets.	83
Figure 74: LSTM model summary for two-leg parlays.....	83
Figure 75: LSTM neural network training accuracy for two-leg parlays.	84
Figure 76: LSTM model summary for all-leg parlays.	84
Figure 77: LSTM neural network training accuracy for all-leg parlays.	85
Figure 78: Flow diagram of implementation.	88

List of Tables

Table 1: Data dictionary describing the contents of the dataset being used.	12
Table 2: The 13 categories of the SPORT feature in the DraftKings dataset and their corresponding acronyms.	12
Table 3: Summarizes the statistics of the boxplot of the number of bets per user.....	30
Table 4: The distribution of wager amount and corresponding term for the quartiles.	37
Table 5: Example of original bet within dataset.	40
Table 6: Example of how a user’s ‘mode of bets’ is created for each user in the dataset.	41
Table 7: Sample of user bet data before one-hot encoding.....	42
Table 8: HOMETEAMNAME example after the data has been one-hot encoded.....	42
Table 9: Model evaluation techniques and their corresponding clustering technique scores used to determine the overall best evaluation procedure.	51
Table 10: Number of bets per cluster.....	52
Table 11: Model evaluation for MNB on individual bets.....	61
Table 12: Model evaluation for MNB on two-leg parlays.....	63
Table 13: Model evaluation for MNB on all-leg parlays.....	65
Table 14: SGD accuracy for individual bets.....	67
Table 15: SGD accuracy for two-leg parlays.....	68
Table 16: SGD accuracy for all-leg parlays.....	70
Table 17: LSVC accuracy for individual bets.....	73
Table 18: LSVC accuracy for two-leg parlays.	74
Table 19: LSVC accuracy for all-leg parlays.	76
Table 20: Machine learning algorithms utilized and their performance.....	86

Table of Acronyms

Word/Phrase	Acronym
Worcester Polytechnic Institute	WPI
Major Qualifying Project	MQP
Exploratory Data Analysis	EDA
National Football League	NFL
National Hockey League	NHL
National Basketball Association	NBA
Major League Baseball	MLB
College Football	CFB
College Basketball	CBB
Stochastic Gradient Descent	SGD
Linear Support Vector Classification	LSVC
Multinomial Naive Bayes	MNB
Deep Neural Network	DNN
Long Short-Term Memory Neural Network	LSTM NN
Term Frequency Inverse Document Frequency	TF-IDF
Natural Language Processing	NLP
Clustering Categorical Data using Summaries	CACTUS
Self-Organized Feature Map for Simple Clustering	SOMSC

1. Introduction

DraftKings provides a platform for various games, including sportsbook and fantasy sports, where users can wager on the outcomes of sporting events. In fantasy sports, players compete against each other and win money if they can create a lineup better than most of the other players in a given contest. In the sportsbook, players compete against DraftKings itself by placing bets with odds set by DraftKings. This project focuses on the DraftKings sportsbook business, which is now legal in 21 states.

One popular type of bet offered by DraftKings are parlays, where users can create a “bet ticket” with multiple linked individual bets, with the winnings rolling over to the next bet if all “legs” of the parlay hit (*Parlay - How to Bet 101 | DraftKings Sportsbook*, n.d.). This makes parlays attractive to users, as they can wager a small amount for a chance to win big. However, the team discovered, through constant communication with the sponsor, DraftKings Sportsbook currently recommends the same, most popular parlay to every user, which lacks personalization.

To address this issue, our team was provided with a sportsbook dataset from DraftKings, containing almost 2 million bets with 22 different features from almost 1,600 users, with the task of classifying user behavior and providing personalized recommendations/predictions for individual bets and parlays¹. Before diving into the data, the team conducted research on sportsbook, types of sports betting, and became familiar with DraftKings’ platforms. The team also explored various clustering methods to group users with similar betting tendencies together and researched different classification algorithms for associating bets and parlays with specific groups of users.

After preparing the data through feature engineering and selection, the users were clustered with several different clustering algorithms. These clustering algorithms were then compared with multiple metrics, the best clustering algorithm was selected, and the clusters were then fixed for the remainder of the project. The clustered data was then used to train multiple classification algorithms to predict the most likely cluster for a given bet or parlay. These

¹ To be clear, this project did not result in a “recommendation engine” in the sense that DraftKings users were presented live recommendations. This project developed methods to predict user behavior that were tested on the static dataset provided by DraftKings.

classification algorithms were then evaluated using multiple performance metrics, e.g., accuracy. The neural network classifier provided the best prediction performance and significantly outperformed random classification as well as classification by most popular cluster. The results of this project will help DraftKings create a more personalized experience for their users by more accurately associating potential bets and parlays with each user, improving their recommendation system, and enhancing user experience.

2. Background

This chapter focused on contextualizing this project within the history of sports betting in the United States, and how DraftKings has grown from a start-up company based in a garage into a leader of online sports betting and daily fantasy sports industry.

2.1 Sports Betting Basics

In the early 19th century, the most common form of betting was betting on horse races. This changed with the establishment of professional baseball, which brought betting to the forefront. Throughout the 20th century, there were several scandals where players confessed or were convicted of throwing games to win money gambling, most notably the Chicago Black Sox in 1919 (Legal Sports Betting, 2022).

While sports betting was quite common throughout this period, it was not yet legal. In 1931, Nevada became the first state to legalize gambling on sports. Still, sports betting was not a massive industry until the 1970s, when Congress lowered the 10% tax on sports bets that bookmakers had to pay (Legal Sports Betting, 2022). The Sports Betting industry has evolved since then, and today, multiple companies such as DraftKings and FanDuel run sportsbook betting through online means such as a website and app (Legal Sports Betting, 2022). Sporting betting through DraftKings Sportsbook is currently legal in 21 states including New England states such as Massachusetts, New Hampshire, and Connecticut (Yakowicz, 2023). As sports gambling has become more prominent in recent years and is the focus of this report, the following sections summarize several key aspects of sports betting including the concepts of odds, different types of bets, and parlays.

2.1.1 Odds

In sports betting, odds determine how much you can win on a bet based on how much you are wagering. In the United States, the most common way of listing odds for bets is through “American Odds”. Other parts of the world use “Fractional Odds” or “Decimal Odds”, but they are more common outside of the United States. It is straightforward to convert between different types of odds (Santaromita, n.d.). For the purpose of brevity, this report will focus on American odds.

In American Odds, odds are either represented with a positive number or a negative number. The negative number represents the favorite for the game, or which outcome is more likely to happen. Conversely, the positive number represents the underdog, or the outcome of the bet that is less likely to happen. The number itself is normally greater than 100, or less than -100, with 'even' odds being represented by a positive 100 (*How to Read Odds - How to Bet 101* / *DraftKings Sportsbook*, n.d.).

The negative number represents how much money a user would have to bet in order to win \$100.00, should the bet "hit". For example, if a bet is listed at -200, a user will have to bet \$200.00 to make a profit of \$100.00 if the bet hits. In this example, the user would receive a "total payout" of \$300.00 if the bet hit. On the other hand, the positive number represents how much money a user would win if they placed a wager of \$100.00. For example, if a bet is listed at +350, a user would make a profit of \$350.00 if that user placed a \$100.00 wager on that bet and the bet hits. In this example, the user would receive a "total payout" of \$450.00 if the bet hit.

2.1.2 Types of Bets

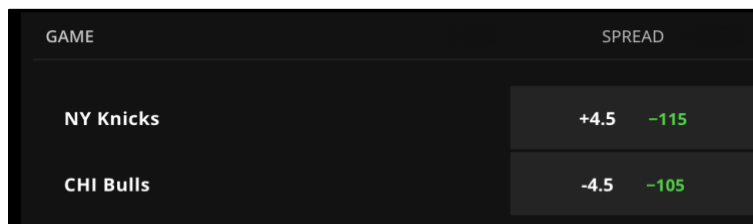
There are 4 major types of bets within sportsbook betting. The major types of bets are moneyline, spread, total, and proposition bets.

A moneyline bet is where the user directly picks the winner of a game. Each game has specific odds associated with the potential outcomes, in which users can place a moneyline bet with. In moneyline bets, the favorite is represented with negative odds, and the underdog is represented with positive odds. For example, a user could bet on an NFL game with the odds "New England Patriots at -150", meaning the New England Patriots are the favored team expected to win. In this case, the user would have to bet \$150 to make a profit of \$100 if the bet wins, netting a total of \$250.

A spread bet is when the user makes a bet on the point separation of the two teams in the game. This type of bet was first created by oddsmakers to increase the number of bets on underdog teams and players. Like moneyline bets, the favorite is represented with a negative number, while the underdog is represented with a positive number. If a user wanted to bet on a favored team, Team A, at -3.5, that user is betting that Team A will win the game by more than 3.5 points. Conversely, if the user chooses to bet on the underdog, Team B, at +3.5, the user is

betting that Team B will either lose the game by less than 3.5 points or win the game. If the user wins, the user will be awarded the amount of money based on the odds set by the oddsmaker and the amount the user bet. (Davidow, Miller, 2019).

Looking at the example in Figure 1, the spread bet for the NY Knicks versus the Chicago Bulls game has the Bulls favored and winning by more than 4.5 points with -105 odds set by the oddsmaker. In this case, the Bulls must win by five or more points to “cover” the spread, i.e., win the spread bet, while the Knicks can lose by four or less points or win the game for them to “cover” the spread. Hence, if a user bet \$105 on the Bulls and the Bulls end up beating the Knicks by five points or more, that user would have a total payout of \$205. On the other hand, if a user bet \$115 on the Knicks, and the Knicks either defeat the Bulls or only lose by four, three, two, or one point, that user would win the bet and have a total payout of \$215.



GAME	SPREAD
NY Knicks	+4.5 -115
CHI Bulls	-4.5 -105

Figure 1: Example from the DraftKings website showing spread NBA bet.

Typically, spreads are set at “.5” rather than a whole number, so the result is firmly on one side or the other, but not always. If the spread was set at a whole number and the result is exactly on that number, that is called a “push”, and the bet is considered a tie. In this situation, the user would be returned the money originally placed on the bet (Davidow, Miller, 2019).

A third type of bets are total bets. Total bets are predictions about the number of points, runs, or goals that occur throughout a game combined between both teams/players. A total bet will come with a line similar to a spread bet. In a total points bet, the line is set at where there is approximately a 50% chance of the total points being above/below that number. This concept is also often called an “over/under bet”. For example, an NFL game between the Patriots and the Dolphins may have an over/under line of 40.5, meaning that a user could predict that the Patriots and Dolphins combined will score more or less than 40.5. Usually, the over/under is shortened to O/U.

TODAY	SPREAD	TOTAL
SGP 8:10PM MIL Bucks	+2 -110	O 226.5 -110
MEM Grizzlies	-2 -110	U 226.5 -110

Figure 2: Example from the DraftKings website showing a total O/U NBA bet. Circled in red is the bet for the over/under of this Bucks vs. Grizzlies game.

Figure 2 shows an example of a total bet for the Milwaukee Bucks versus the Memphis Grizzlies. Both odds are set to -110, implying that the user has a 50% chance of either side of the bet hitting. If the user were to select ‘O 226.5’, the user would be betting that the points combined between both the Bucks and the Grizzlies will total at least 227 points. Rather, if the user were to select ‘U 226.5’, the user would be betting that the points combined between the Bucks and the Grizzlies would total no more than 226 points.

Another kind of bet available are proposition bets. The proposition bet is the most varied type of bet. A proposition, or “prop” for short, is a bet made regarding the occurrence of a particular event throughout a game. Prop bets can be made regarding a player, a team, a game, or a combination. Essentially, a prop bet is betting on whether any particular action or event will occur. For example, a player prop bet could be “Mac Jones O/U 299.5 passing yards” and a team prop bet could be “Patriots O/U 399.5 yards allowed”.



SGP CIN Bengals at TB Buccaneers		SUN 18TH DEC 4:25PM	
PLAYER	OVER	UNDER	
 Joe Burrow	O 1.5 -175	U 1.5 +130	
 Tom Brady	O 1.5 -125	U 1.5 -105	

Figure 3: Example from the DraftKings website of a player passing touchdowns prop event occurring in an NFL game.

Player prop events vary depending on the position the player plays and the sport the bet is for. One common football player prop bet is a passing touchdowns prop for quarterbacks. Figure 3 has a different passing touchdown prop for each of the starting quarterbacks, Joe Burrow and Tom Brady, in the Cincinnati Bengals at Tampa Bay Buccaneers game. Betting ‘O 1.5’ with -125 odds for Tom Brady means that you are betting that Tom Brady will have at least two

touchdown passes. To make \$100 profit off this bet, the user would need to bet \$125. Looking at the passing touchdown prop for Joe Burrow, the underdog bet would be to bet 'U 1.5' with +130 odds. This means that you are betting Joe Burrow will throw zero or one touchdown passes and betting \$100 will create a profit of \$130 if the bet hits (Davidow, Miller, 2019). It is important to note that while prop bets usually take the form of an over/under line where the user picks one side of the line to bet on, a prop bet is still vastly different from a total bet.

There are, of course, other types of bets available in the DraftKings sportsbook including "future bets" on things like the number of wins a team will accrue over the course of a season. The focus of this project, however, is on bets with outcomes that are determined in individual sporting events, the types of which are discussed above, and combinations of these bets into parlays. The next section discusses key aspects of parlays.

2.1.3 Parlays

As discussed, the four different bet types can be placed as individual bets, but two or more bets can be combined to create a parlay. In order to win a parlay bet, all of the individual bets within the parlay must hit (*Parlay - How to Bet 101 | DraftKings Sportsbook*, n.d.). Since the odds for parlays tend to be higher than the odds for most individual bets, this is a way for users to wager a small amount of money to win a relatively large amount of money if every bet on the entire parlay hits. When making a parlay, the user can choose to select bets pertaining to the same game, which is referred to as a same-game parlay, bets that pertain to the same sport, which is usually called a same-sport parlay, or bets that have no relation to each other. Parlays can include as many different bets as the user would like to choose, often described as the number of "legs" in a parlay (Davidow, Miller, 2019).

Figure 4 is an example of a two-leg parlay slip on the DraftKings website. Wagering \$100 for this parlay will have a total payout of \$1,504.00 if both legs of the parlay hit.

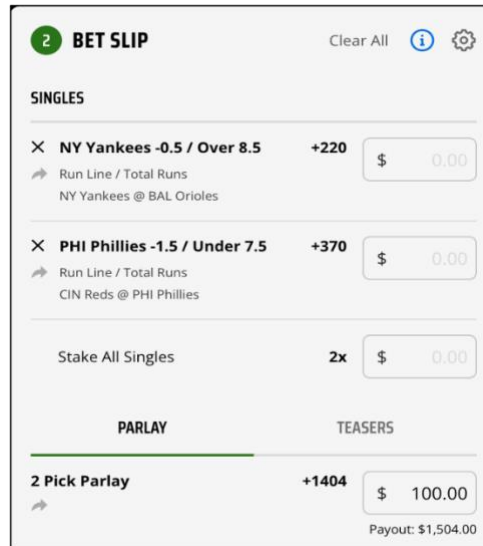


Figure 4: Example of a parlay bet slip on DraftKings.

2.2 DraftKings in the Sports Betting Industry

As sports betting has seen tremendous growth in the industry, the legalization of sports betting created opportunity for companies such as DraftKings to expand and offer new platforms.

2.2.1 Origin and Growth

Founded in 2012, DraftKings is an online sportsbook provider with its headquarters located in Boston, Massachusetts. After a few short years, its founders Jason Robins, Matt Kalish, and Paul Liberman turned the company from a garage-based startup into an estimated \$15.84 billion company as of April of 2023 (DraftKings net worth 2019-2023 | DKNG, 2023). DraftKings offers weekly or daily fantasy sports contests for 15 sports across 8 countries. DraftKings is split up into 5 different verticals: Sportsbook, Daily Fantasy Sports, Casino, Marketplace, and Reignmakers NFT Game.

During May 2018, the Professional and Amateur Sports Protection Act of 1992 was declared unconstitutional by the Supreme Court, allowing states other than Nevada to legalize sports betting (Liptak & Draper, 2018). In November 2018, DraftKings and rival company, FanDuel, announced their intent to merge, which would provide services to over 5 million users. However, the following year, the Federal Trade Commission stated that it would seek to block the merger between the two companies. The two companies combined would have given them

90% of the daily fantasy sports market, which would have been considered a monopoly position (Heitner, 2017).

2.2.2 Current Platforms

DraftKings describes the DraftKings Sportsbook platform as a “one-stop, high-octane jet fuel for all your betting needs” (*Sportsbook / about DraftKings*, n.d.). The company’s app is highly rated and uses world-class technology to provide users with a safe and secure online sports betting experience. DraftKings is currently experiencing unprecedented growth, generating \$855 million in revenue in the last quarter of 2022 (*DRAFTKINGS REPORTS FOURTH QUARTER REVENUE of \$855 MILLION*, 2023). To operate in each state, DraftKings requires a license, and it is currently allowed in 21 states, including online gambling in 5. The recent launch in the state of New York could prove very lucrative due to its large population. However, California, with nearly 40 million residents, recently rejected a proposition to allow private sportsbook operators like DraftKings to partner with a California tribe to allow mobile sports betting (Hughes, 2023). This indicates that the conversation around legalizing sports betting in California will take years, if it happens at all (Waters, 2023). On March 10, 2023, during the course of this project, online gambling and sports betting became legal in Massachusetts, where WPI resides.

The focus of this project is on user behavior in the DraftKings Sportsbook. Additional information about Daily Fantasy Sports, Casino, Marketplace, and Reignmakers can be found in [Appendix A](#). In the DraftKings Sportsbook, users can bet against a myriad of different sports at different levels, placing money on different types of bets including the moneyline, spread, prop, and total lines for each game as described in [Section 2.1.2 Types of Bets](#). By clicking on available bets on both the mobile app and on their Sportsbook website, users can build a “bet ticket” and place a wager. Selecting multiple bets will start creating a parlay and display the changing odds and the total payout as a user adds legs to the parlay.

2.3 Dataset and Features

The goal of this project was to create a model that generates individual and parlay bets that are specific to a user’s betting history using a large dataset provided by DraftKings. Upon completion of this specific bet and parlay suggester, the information this model produces will

enhance the user's betting experience within DraftKings Sportsbook website and application interfaces. The dataset the team used to train our model was a collection of bets gathered between July 2021 and June 2022. Prior to cleaning, prepping, and filtering, the dataset contained 1,941,026 rows with 22 columns. DraftKings extracted this data from their database under the condition that each user has placed no more than 3,000 bets within this timeframe. The types of features that are included in the dataset describe different details of the bets that were placed, e.g., the user placing the bet, the sport, the event of the game matchup, the date the bet was placed, the state the bet was placed in, and more. The features will be further explained in the following section.

2.3.1 Data Dictionary

The table below contains a list of all the features from the original data source as well as features that were created during the data preparation and cleaning stages found in [Section 3.2 Cleaning and Preparing the Data](#). The list contains the feature name, data type, and a brief description. An “*” next to a feature name denotes that it was a feature the MQP team created. Later sections of this report will detail more about the additional features the team made.

Feature	Data Type	Description
USERHASHID	str	Unique user placing the bet
BETTICKETIDHASH	str	Unique bet ticket identifier
PLACEDDATE	str	Date when the bet was placed, includes time
BETDESCRIPTION	str	Type of bet made; includes general types such as moneyline, spread, proposition, or total to specific bets such as “Jose Ramirez Doubles”
LABEL	str	Who/what is being bet on
ISPARLAY	int64	Flag denoting if bet was a part of parlay, 1= parlay, 0 = not parlay

BETSTATUS	str	Whether the bet is settled or cashed
WAGERAMOUNT	float64	Amount of money wagered
PAYOUTAMOUNT	float64	Amount of money that would pay out
ODDSAMERICAN	str	Odds written as + or -
ODDSDECIMAL	float64	Odds as a decimal
ODDSFRACTIONAL	str	Odds as a fraction
SPORT	str	What sport the bet is placed on
EVENTGROUP	str	What league/tournament the bet is placed on
EVENT	str	Game matchup
HOMETEAMNAME	str	Home team name
AWAYTEAMNAME	str	Away team name
HOMETEAMSCORE	str	Home team score
AWAYTEAMSCORE	str	Away team score
STATE --> TIMELINE*	str	Whether the bet has started or finished. Originally was "STATE", changed to "TIMELINE"
JURISDICTION	str	State the bet was placed
EXTERNALIDENTIFIER	str	Country and state reference
BETS_ON_TICKET*	int64	Counts the number of bets including in the parlay

BETTYPE*	str	Created to generalize the “BETDESCRIPTION” feature into one of four categories: moneyline, spread, total (total points), proposition
WAGERAMOUNTNORMALIZED*	str	Contains one of four categories: low, medium, large, highest- calculated from normalizing the values in the WAGERAMOUNT field. Used to better categorize different amounts of spending on bets.

Table 1: Data dictionary describing the contents of the dataset being used.

Most of these features in the dataset are categorical with several unique possible categories. For example, the SPORT feature has 13 different possible categories:

Sport	Acronym
National Basketball Association	NBA
National Football League	NFL
Major League Baseball	MLB
College Basketball	CBB
College Football	CFB
National Hockey League	NHL
Soccer	SOC
Mixed Martial Arts	MMA
Tennis	TEN
Golf	GOLF
NASCAR	NAS
EuroLeague	EL
Canadian Football League	CFL

Table 2: The 13 categories of the SPORT feature in the DraftKings dataset and their corresponding acronyms.

The data cleaning process, which will be discussed in more detail in [Section 3.2 Cleaning and Preparing the Data](#), involved the decision to focus on the most popular sports, namely NBA, NFL, MLB, CBB, CFB, and NHL. Similarly, the JURISDICTION feature, which represents the state where the bet was placed, has several possible categorical values, such as IL, CO, MI, IN, NJ, PA, NY, WV, TN, VA, IA, CT, AZ, NH, LA, WY, and OR. It is worth noting that this data was collected prior to some states legalizing sports gambling and/or online and mobile app sportsbook betting. Columns such as HOMETEAMNAME and AWAYTEAMNAME also

offered a wide range of possibilities, as they could include any team from any of the sports listed above.

The dataset provides information at both the bet and parlay level, where each bet has a unique BETTICKETIDHASH that is only repeated between rows when bets are part of the same parlay. To determine the number of legs in a parlay, the team created the feature, BETS_ON_TICKET, which counts the number of times a BETTICKETIDHASH occurs. To identify which bets are included in a parlay, which is denoted by ISPARLAY = 1, the team can filter on a specific BETTICKETIDHASH, and the specific bets that are part of this ticket will appear. Figure 5 displays an example of a four-leg parlay in the dataset, showing a sample of eight features. Note how the BETTICKETIDHASH is the same for all four bets in the parlay, and how this four-leg parlay includes four different proposition bets across four different NBA games.

	BETTICKETIDHASH	BETDESCRIPTION	BETTYPE	EVENT	HOMETEAMNAME	SPORT	AWAYTEAMNAME	WAGERNORMALIZED
4	49fa6501-23d7-4bae-bb1a-8119270b1382	Series Winner	Proposition	GS Warriors vs DEN Nuggets - Series	GS Warriors	NBA	DEN Nuggets	30.0
182	49fa6501-23d7-4bae-bb1a-8119270b1382	Series Winner	Proposition	MIA Heat vs ATL Hawks - Series	MIA Heat	NBA	ATL Hawks	30.0
229	49fa6501-23d7-4bae-bb1a-8119270b1382	Series Winner	Proposition	PHI 76ers vs TOR Raptors - Series	PHI 76ers	NBA	TOR Raptors	30.0
272	49fa6501-23d7-4bae-bb1a-8119270b1382	Series Winner	Proposition	MIL Bucks vs CHI Bulls - Series	MIL Bucks	NBA	CHI Bulls	30.0

Figure 5: Example of a four-leg parlay in the dataset.

To access information about an individual bet, the same process applies. By filtering for the specific BETTICKETIDHASH, the individual bet will be displayed along with its associated features, as illustrated in Figure 6.

	BETTICKETIDHASH	BETDESCRIPTION	BETTYPE	EVENT	HOMETEAMNAME	SPORT	AWAYTEAMNAME	WAGERNORMALIZED
0	b32a5455-aa5e-41be-b98b-9b2f4f65aa68	Moneyline	Moneyline	BOS Celtics @ GS Warriors	GS Warriors	NBA	BOS Celtics	25.0

Figure 6: Example of a singular bet in the dataset.

2.4 Machine Learning Techniques

In order to generate predictions of user behavior for DraftKings, this project used machine learning techniques to cluster users and to associate bets and parlays with user clusters. Machine learning is a branch of artificial intelligence. Similar to how humans learn, a machine is taught through various computer algorithms; the machine looks at the input of the data and tries to formulate relationships or patterns between the data (Brown, 2021). Machine learning models

allow for great amounts of data to be analyzed at once and at ease compared to the manual input of a human. The team used different machine learning techniques, algorithms, and clustering evaluation methods in the creation of the personalized bet and parlay suggestion model. The report provides a detailed description of the implemented techniques and algorithms and how the team utilized them with the dataset to accurately assign bets to users.

2.4.1 Basics of Machine Learning

The purpose of machine learning falls under one of three categories: to be descriptive, predictive, or prescriptive; it could assist in describing the data, making predictions for the future, or suggesting certain actions to take based on patterns or relationships found (Brown, 2021).

When it comes to types of learning a machine experiences, there are two options: unsupervised and supervised. The main difference between the two types of learning comes from the type of data the machine is learning from. Supervised learning occurs when a dataset has distinct input and output (or target) variables and the algorithm finds patterns in the dataset to create relationships in order to create predictions of the output when given new input data (Berry et al.). On the other hand, unsupervised learning does not involve a target variable, in fact there is no “correct” output (Hiran et al.). Unsupervised learning finds patterns in the dataset that have not been predetermined, interprets the data on its own, processes the findings, and then groups the data together by clusters of similar attributes (Berry et al.). The team determined the dataset to contain unsupervised data as there was no expected output and called upon unsupervised learning algorithms to create clusters and classify the data. Having determined the need for unsupervised learning, the team utilized multi-class classification to classify the bets.

2.4.2 Introduction to Multi-Class Classification

An important branch of machine learning is called "classification", where an algorithm attempts to predict where a data point belongs in a set of predefined classes. The simplest version of classification is called binary classification, whereas the name suggests, there are only two classes that the algorithm makes its prediction between. The natural extension of this idea is multi-class classification, where instead of predicting between two classes, the algorithm must make its prediction between three or more classes.

In order to evaluate a multi-class classification algorithm, a commonly used loss function is multi-class cross entropy, where the difference between the actual and predicted probability distributions across all classes in the problem is calculated. Multi-class classification algorithms can also be evaluated after training and testing due to a variety of different metrics. These metrics include Accuracy, Precision, Recall, and F-1 Measure. These metrics will be discussed in greater detail in [Section 2.9 Evaluation Metrics](#).

2.5 Natural Language Processing

A key step in the team's approach to clustering and classification involved the conversion of bet tickets into "bet sentences". This then allowed the use of natural language processing (NLP) techniques for clustering and prediction. NLP is a branch of machine learning which aims to enable computers to process text and speech in a similar way that humans do. NLP has diverse applications across different disciplines such as translating text between two languages, voice-activated systems, and creating rapid summarizations of large texts (IBM, n.d.). Common examples of NLP in everyday life include customer service chatbots, voice operating systems within GPS systems, and text-to-speech applications.

By leveraging NLP, meaningful insights can be extracted from unstructured text data, enabling complex analysis tasks, and transforming qualitative textual data into decision-making foundations (Porter, 2022). By leveraging NLP, meaningful insights can be extracted from unstructured text data, enabling complex analysis tasks, and transforming qualitative textual data into decision-making foundations (Porter, 2022).

In this project, the team recognized that the dataset primarily consisted of textual data with limited numerical fields and identified the clear potential for leveraging NLP in processing the data to generate sentences based on the bet data. As part of the NLP analysis, the team utilized Term Frequency – Inverse Document Frequency (TF-IDF) to transform count vector data from the textual data, making it amenable to processing by various classification algorithms. The next section provides an overview of TF-IDF and how it was used in this project.

2.5.1 Term Frequency- Inverse Document Frequency (TF-IDF)

Term Frequency – Inverse Document Frequency (TF-IDF) is used to help determine how much weight a specific word in a document carries. The purpose of TF-IDF is to find and reduce

the importance of words that frequently occur in bet descriptions and, therefore, their significance in computing the final similarity scores (Kim & Gil, 2019). TF-IDF is the product of the weight of the term frequency and the weight of the inverse document frequency. Terms with high values have a low significance because the term appears across multiple documents. Conversely, terms with high scores are considered keywords and have a higher importance. The understanding of the TF-IDF value is that it increases when a term appears many times in one document yet appears a few times across all the documents. This is an indication of a keyword in specific papers or across many documents (Kim & Gil, 2019).

In this project, the TF-IDF technique was used to transform the count vector of the training data, which consists of information data. The TF-IDF transformation involved calculating the term frequency (TF) and inverse document frequency (IDF) for each term in the bet description. The TF measures how frequently a term appears in a particular document (i.e., different features of a bet), while the IDF measures the significance of a term across all the documents (i.e., different features of a bet). The product of TF and IDF yields the TF-IDF weight for each term, which helps to determine the importance of words in the bet information data. By applying the TF-IDF transformation to the training data, the models were able to learn from the transformed textual bet data and determine which features in the bets had higher importance for users.

2.5.2 Jaccard Similarity and Jaccard Distance

Jaccard Similarity and Jaccard Distance are ways to compare two sets of terms of qualitative data and turn that score into a number. The two terms are inverses of each other, creating the equation *Jaccard Similarity* + *Jaccard Distance* = 1. The Jaccard Similarity is defined by the ratio of the intersection to the union of the two sets. In other words, the similarity between two sets is the ratio of the shared terms between both sets and the total number of unique terms in the two sets.

$$Jaccard\ Similarity(A, B) = \frac{A \cap B}{A \cup B}$$

$$Jaccard\ Distance(A, B) = 1 - \frac{A \cap B}{A \cup B}$$

Since the Jaccard Similarity is defined as a proportion, it has values ranging from 0 to 1, where 0 correlates to no similarity and 1 correlate to completely identical sets of values. Conversely, Jaccard Distance ranges from 0 to 1 with 0 meaning 0 distance between the points and 1 meaning the maximum possible distance between the data points (“What Is the Jaccard Similarity Measure in NLP?”). An example of Jaccard Similarity is shown below.

Sentence One: “Moneyline, New England Patriots, New York Jets”

Sentence Two: “Moneyline, Boston Red Sox, New York Yankees”

$$\text{Jaccard Similarity}(\text{Sentence 1, Sentence 2}) = \frac{1}{5} = 0.2$$

$$\text{Jaccard Distance}(\text{Sentence 1, Sentence 2}) = 1 - \frac{1}{5} = 0.8$$

The example above provides insight into how Jaccard Similarity and Jaccard Distance were used in the project. Jaccard Similarity returns the fraction of words that appear in both sentences divided by the total number of unique words in both sentences. In this example, the term moneyline is found in both sentences as why ‘1’ appears in the numerator. Both Sentence One and Sentence Two have three individual terms, but as ‘Moneyline’ is found in both, there are only five unique terms. The count of unique terms can be found as the denominator of the Jaccard Similarity. The resulting Jaccard Similarity score is 0.2. As discussed, to calculate the Jaccard Distance, 1 minus Jaccard Similarity is taken. For the example, the Jaccard Distance is $1 - 0.2$, which returns 0.8.

2.6 Clustering Techniques

As this project involves multi-class classification of bets and parlays to groups of users, clustering users together was an essential step in the project. Clustering is a form of unsupervised learning (Alpaydin, 2020) and is a technique used for grouping similar data points together, where data points within each cluster are similar to each other, but different from data points in other clusters (Soni Madhulatha, 2012). The similarity between data points is typically measured using some quantitative distance between two points, e.g., the E distance as discussed above.

The team explored four different styles of clustering, which included K-means clustering, clustering categorical data using summaries (CACTUS), spectral clustering, and hierarchical

clustering. These clustering techniques are described in detail in the following sections and were compared with multiple metrics to determine the most effective method for clustering users together in the context of this project.

2.6.1 Elbow Curve Method

The elbow curve method is a commonly used method for choosing the optimal value of k , or optimal number of clusters for K-means clustering. The elbow curve is a 2D plot where the x-axis represents the number of clusters (k) and the y-axis represents the sum of squared error (SSE). The SSE is the sum of squared distances between each data point and its assigned cluster center. As the number of clusters increase the SSE will decrease since each point should be closer to the assigned cluster. At a certain point, the decrease in SSE will be ridiculously small and level out creating the point of inflection otherwise known as the “elbow”. The idea behind the elbow point is that adding more clusters does not significantly improve the clustering result. The elbow point is identified by observing the plot and finding the point where SSE begins to flatten out. This point is the optimal number of clusters (Syakur et al., 2018).

2.6.2 K-means Clustering

K-means is a common clustering algorithm for unsupervised data, which is nonlabelled data, which works on the basis of comparing each data point to the mean of the data points that make up each cluster and then assigns the data point to the one with the closest mean (Soni Madhulatha, 2012). K-means finds the centroids by randomly assigning data points to clusters and then the centroids are recalculated. From the new centroid, the data points are reassigned based on how close the data points are to the centroid. The process is repeated until the centroids of the clusters does not change (Piech, n.d.). K-means calculates the Euclidean distance of the data points to assign each data point to a cluster. The K-means Clustering algorithm is optimal for large datasets as the procedure does not take the Euclidean distance for all the data point (Penn State, n.d.). In the case of this project, the team utilized K-means to cluster the users of the dataset, which was tested up against the clustering evaluation methods to find an optimal clustering algorithm.

2.6.3 Self-Organized Feature Map for Simple Clustering (SOMSC)

Self-Organized Feature Map for Simple Clustering, also known as SOMSC, uses artificial neural networks to cluster sentences. The algorithm uses a process for unsupervised learning to discover relationships between sentences using brain maps and semantic maps. SOMSC is a subcategory of Self-Organized Feature Map (Kohonen, 1990). SOMSC implements the K-means algorithm, which is mentioned in [Section 2.6.2 K-means Clustering](#), in the form of a neural network. SOM reduces the dimensions of the data to a map and looks for the similarities of the dataset. SOM initializes weights and then iterates through the input data attempting to find a neuron for the inputs using Euclidean distance. As the learning algorithm is iterating over the inputs, the weights are adjusted for the neurons (Pang, 2003).

2.6.4 Clustering Categorical Data Using Summaries (CACTUS)

Clustering categorical data using summaries, or CACTUS, is an efficient and valuable clustering method for categorical data. This method is particularly useful for data with many attributes and unique values, as well as missing data points. CACTUS operates in three phases: first, it summarizes the dataset and the relationship between each attribute. Then, it clusters data points based on the summaries from the previous step to incorporate attribute relationships, and finally, it validates the clusters created (Ganti et al., 1999; Dutta et al., 2005). The clustering phase of the summaries enables it to handle large datasets with many features (Ganti et al., 1999). Any clustering algorithm can be utilized in phase two.

2.6.4.1 Summary Matrix

During the first phase of the summarization step in the CACTUS method, a summary matrix is created which relates each feature to one another to weigh similarities and quantify co-occurrences. These summaries are then utilized to create more accurate clusters of features (Abdu & Salane, 2009). With the particular data at hand, summary matrices for the users would be created to reflect overlapping tendencies to cluster similar users together. The dimensions of the matrix would reflect the number of unique users in the data; for n unique users, the summary matrix would be $n \times n$. Each value within the summary matrix ranges from 0 to 1 where 1 indicates that the users always co-occur together and 0 means the users rarely occur together. Figure 7 is an example of what this summary matrix between users looks like.

	0	1	2	3	4
0	1.000	0.375	0.250	0.250	0.125
1	0.375	1.000	0.125	0.375	0.250
2	0.250	0.125	1.000	0.250	0.250
3	0.250	0.375	0.250	1.000	0.375
4	0.125	0.250	0.250	0.375	1.000

Figure 7: Sample of what a user summary matrix would look like between users. The column/row labels 0-4 represent five different unique users.

The closer the value is to 1, the more often the users overlap in their betting tendencies. Here, user 0 and user 1 have a value of 0.375, which means they have similar betting tendencies 37.5% of the time.

2.6.5 Spectral Clustering

Spectral clustering aims to group data points with high similarity together and separate those with low similarity (Bach & Jordan, 2004). It utilizes the summary matrix by performing eigenvalue decomposition, and then uses the eigenvectors to cluster the data points together based on their similarity (Bach & Jordan, 2004). In the project, after generating summary matrices for the users, the team would perform eigenvalue decomposition and use the resulting eigenvectors to cluster the data points together based on their similarity.

2.6.6 Hierarchical Clustering

Hierarchical clustering is another technique for grouping data points together, specifically using the dissimilarity between data points to create these groups (Nielsen, 2016). The two types of hierarchical clustering are agglomerative and divisive. Agglomerative clustering is the ‘bottom-up’ approach where each data point is initially placed into its own cluster and then is iteratively merged with others who are least dissimilar to them (Nielsen, 2016). Divisive clustering is the opposite of agglomerative; it is the ‘top-down’ approach where all data points start in one cluster and recursively separates data points based on dissimilarities (Nielsen, 2016).

Hierarchical clustering uses a distance measure, such as Jaccard distance, to quantify this dissimilarity in data points before the agglomerative or divisive methods are used.

2.7 Clustering Evaluation Methods

After a clustering algorithm is run, it is crucial to assess the effectiveness of the grouping of data points. This is where cluster evaluation comes in, to determine the validity and quality of the clustering performance (Jain & Dubes, 1988). Given that clustering is a type of unsupervised learning, with no predefined set of labels or classes, evaluation is a critical part of the process (Jain & Dubes, 1988). In general, there are three primary methods for evaluating clustered data: Silhouette Coefficient, Calinski-Harabasz Index, and Davies-Bouldin Index, all of which will be discussed in more detail. The three cluster evaluation methods return scores with different meanings and assess different features of the clustering algorithms. Evaluating the distribution of the clusters is essential in ensuring the clusters are representative of the dataset and in this project's case, users with similar bet history. These evaluations were crucial in identifying which clustering method should be utilized to cluster users together.

2.7.1 Silhouette Coefficient

The silhouette coefficient is computed for each user in the dataset as a measure of its similarity to other users within its own cluster and to users in other clusters (Jajuga et al., 2020). For example, let's consider a dataset with 1000 users, and after applying the clustering algorithm, these users are assigned to different clusters. The silhouette coefficient is then calculated for each of the 1000 users. The silhouette coefficient for a particular user reflects how well that user belongs to its assigned cluster, with values ranging between -1 and 1. A silhouette coefficient score closer to 1 indicates that the user is more similar to other users within its cluster and dissimilar to users in other clusters, suggesting a dense and correctly separated cluster performance, which is considered ideal. On the other hand, a value of -1 indicates that the user is poorly clustered and does not belong to the cluster it was assigned; the user is more similar to users in other clusters rather than its own. A score of 0 denotes overlapping clusters, which may not be the best clustering method for grouping users (Jajuga et al., 2020). The individual silhouette coefficients for all the users are then aggregated to obtain a single coefficient score for evaluating the overall performance of the clustering algorithm.

2.7.2 Calinski-Harabasz Index

The Calinski-Harabasz Index is a metric that evaluates clusters by measuring how closely related each cluster is and the relation to other clusters (Wang & Xu, 2019). The index ranges between 0 and positive infinity. The higher the Calinski-Harabasz Index, the better the clustering performance. The calculation is the ratio between inter cluster and intra cluster dispersion (Wang & Xu, 2019). This metric is also computationally efficient compared to others and is faster to compute (Wang & Xu, 2019).

2.7.3 Davies-Bouldin Index

The third metric mentioned that can be used for cluster evaluation is Davies-Bouldin Index. Similar to the Calinski-Harabasz Index, Davies-Bouldin has a range from 0 to infinity, but in this case, a better cluster is represented by a lower value (Kärkkäinen & Fränti, 2000). The Davies-Bouldin Index utilizes the centroids of each cluster by measuring the similarity between a cluster's centroid and the nearest cluster's centroid (Kärkkäinen & Fränti, 2000). Similar to the other metrics, it also looks within each cluster, specifically the span of scatter, and the separation between all clusters (Kärkkäinen & Fränti, 2000).

2.8 Classification Algorithms

Within the realm of multi-class classification, there is a wide variety of algorithms that can be used. For this project, the team investigated several algorithms, including the Naive Bayes Classifier, Stochastic Gradient Descent, Linear Support Vector Classification, Random Forest Classifier, and Neural Networks. These algorithms were considered for their potential to effectively classify data points into multiple classes based on their features. Further details of these algorithms are found in the following sections.

2.8.1 Naive Bayes Classifier

The Naive Bayes classifier is a simple but effective probability-based classification machine learning algorithm that is based on the Naive Bayes Theorem. The Naive Bayes classifier is a supervised learning algorithm that predicts the class of the training or testing instance with the highest probability. Compared to other classification models, it has many advantages: it is easy to use and understand, automatically ignores missing entries and their corresponding probabilities, requires only one loop to analyze data, and outputs continuous data.

Additionally, the algorithm does not tend to overfit the data to the model (Farid et al., 2014). However, the classifier assumes that the columns of the dataset are not correlated, which reduces computation time.

An extension of the Naïve Bayes classifier is the multinomial Naïve Bayes classifier. This classifier uses a “bag of words”, which is an ordered list of words with corresponding frequencies that serves as a text document, to determine the class with the highest probability when given a specific document. The returned probability is the product of the prior probability of the class and the likelihood of the document, which is the probability of the document given the class (Jurafsky & Martin, 2023). When working with a “bag of words,” two assumptions are made: the position of the words in the list has no relevance to meaning and the probabilities of the words are independent given a class.

2.8.2 Stochastic Gradient Descent Classifier

Stochastic Gradient Descent (SGD) is a common alteration of gradient descent that is also referred to as incremental gradient descent. The SGD classifier is a machine learning algorithm that attempts to find the optimal parameters while reducing the training time. The main focus of gradient descent is to attempt to find different weights that have the best fit of the training data while trying to overcome that the data cannot converge because it is not linearly separable. SGD does a step-by-step approximation of the gradient descent as it updates weights at each step after a specific training instance error is calculated (Mitchell, 1997). The SGD classifier has become widely accepted in the field of data science as a form of multi-class classification because the algorithm reduces computation costs without affecting the accuracy of the model (Wilbur & Kim, 2014).

2.8.3 Linear Support Vector Classification

Linear Support Vector Classification (Linear SVC) is a technique that can be used to reduce the dimensionality of a dataset and remove noise and redundancy, which can result in improved performance of machine learning models, as discussed by Shlens (2014). Linear SVC attempts to create a linear hyperplane that maximizes the distances between the differentiation of clusters. By doing so, each instance of the dataset is classified to a specific cluster (*Linear SVC.*, n.d.).

2.8.4 Neural Networks

Neural Networks are another algorithm that can be utilized as a multi-class classifier. One family of neural networks are called Recurrent Neural Networks (RNN). Instead of being strictly feedforward networks, these RNNs can repeat themselves to add depth to the calculation. For this reason, types of RNNs are very designed for predicting sequential data by observing its predecessors (Camargo et. al., 2019). As shown in Figure 8, RNN uses the output from its previous iteration as one of the inputs to its current cell, or calculation.

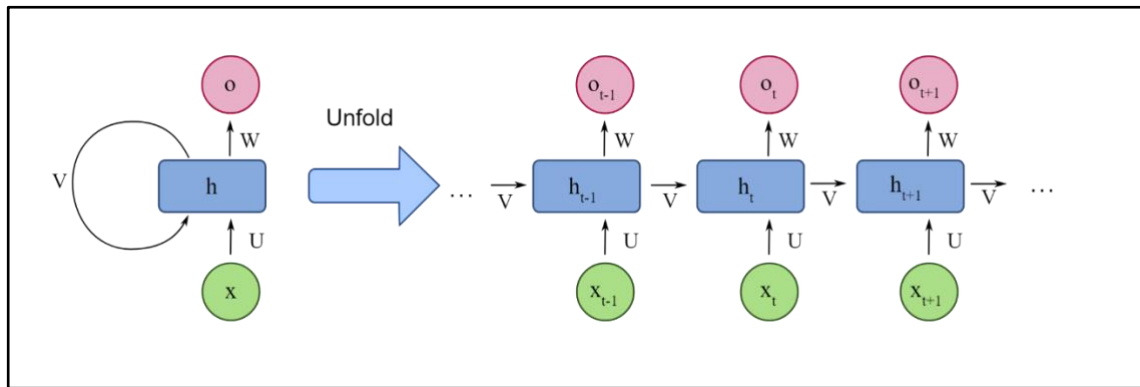


Figure 8: Demonstration of RNN using sequential data.

However, many RNNs struggle to account for long term effects of data. This is fixed by utilizing a specific type of RNN called a Long Short-Term Memory (LSTM) neural network. These specific types of algorithms have storage built in for both short term memory to be accessed instantly, and long-term memory to be held on to a long time to use in many calculations further into the running of the algorithm (Camargo et. al., 2019). Because of its ability to hold onto information for long periods of time while performing other calculations, a LSTM network is particularly powerful in dealing with categorical data (Camargo et. al., 2019). The team employed both a Deep Neural Network and a LSTM Neural Network as algorithms for the multi-class classification.

2.9 Evaluation Metrics

Metrics play a crucial role in the machine learning process. They determine which algorithm performs the best and should be used moving forward (Arguello, 2013). Metrics evaluate the recommendation engine's performance and assess how well it generated predictions based on the data used. Different metrics measure different aspects of the algorithm and

choosing which metric to use is impactful on the recommendation engine’s performance outcome (Tamm et al., 2021). The team established metrics for evaluating the correct classification of bets to clusters. To do so, the team developed an understanding of a confusion matrix that would be suitable for the model, and used accuracy, precision, recall, and F1 score as evaluation metrics. Accuracy was considered the most important metric, as it could be consistently measured across all algorithms. Precision, recall, and F1 score were utilized to gain a deeper understanding of the classification and misclassification of clusters, as well as to identify common mistakes made by each algorithm in the classification process.

2.9.1 Accuracy Score

Before discussing accuracy, it is pertinent to understand the confusion matrix; a confusion matrix is necessary to create the accuracy metrics. A confusion matrix is a tabular way of representing the predicted values of the classification model versus the actual values. In a binary classification problem, the confusion matrix has the dimensions of a 2x2, like the one in Figure 9. The confusion matrix has four classifications: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). Figure 9 illustrates a confusion matrix and shows the four different categories.

		Predicted class	
		Positive	Negative
Actual class	Positive	TP	FN
	Negative	FP	TN

Figure 9: Confusion matrix diagram (Nickolas, 2021).

True positives are data instances that are correctly identified as positive. False positives are data points incorrectly labeled as positive when the actual class is negative. True negatives are data points labeled as negatives when the actual class is also negative. False negatives are positive data points incorrectly labeled as negatives. These four classes make up the formula for accuracy, precision, and recall.

Accuracy is a way to evaluate a classification model. Its calculation is the number of correctly identified items divided by the number of total observations. In our case, the accuracy of our model will be computed by dividing the number of correctly predicted bets by the number of total predictions the team can make. Figure 10 displays three equivalent ways to calculate the accuracy score of a classification model.

$\frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}} = \frac{\text{N. of Correct Predictions}}{\text{N. of all Predictions}} = \frac{\text{N. of Correct Predictions}}{\text{Size of Dataset}}$

Figure 10: How to calculate accuracy (Kanstrén, 2021).

2.9.2 Precision, Recall, and F1 Score

Precision, Recall, and F1 Score are all other metrics that utilize the understanding of a confusion matrix to formulate metrics. Precision is the proportion of positive labeled data points that were correctly identified as positive. The formula for precision is $\frac{TP}{TP+FP}$. Recall is the fraction of true positives over the entire data set of positively labeled data points. The formula for recall is $\frac{TP}{TP+FN}$ (Davis & Goadrich, 2006).

The F1 score is calculated by computing the harmonic mean of precision and recall. The formula for F1 score is $2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$. The harmonic mean is a different way to calculate the average of two values and is often better when working with proportional values, such as precision and recall (Kanstrén, 2021). The F1 score provides a single metric that uses both ratios in a balanced way, and this computation depends on both precision and recall increasing for the F1 score to increase as well.

2.9.2.1. Macro and Micro Precision and Recall

As the number of classes increases in a classification problem, so does the size of the confusion matrix. In multi-class classification, a model must classify instances into three or more different classes, which requires additional evaluation of the classification model (Sokolova & Lapalme, 2009). The precision and recall metrics are now calculated at a macro and micro level. Macro precision and recall averages across all the classes, giving equal weight to each, while

micro precision and recall gives greater weight to the larger classes (Sokolova & Lapalme, 2009). The use of macro and micro precision and recall provides further insights into the performance of the classification model when there are more than two classes.

3. Methodology

The following chapter presents the methods utilized by the team to create personalized bet and parlay recommendations for each user. Our analysis started with cleaning and preparing the raw data for modeling. Feature engineering techniques were applied to extract relevant information and create new features to enhance model performance. Exploratory data analysis was also conducted to gain insights from the data. Subsequently, unsupervised clustering techniques were used to group users based on their betting tendencies and create a target variable for the dataset. Multiple machine learning algorithms were trained and evaluated using precision and recall metrics once the data was prepared for modeling.

Figure 11 provides an overview of the multi-class classification process that transforms the dataset into tailored bet and parlay recommendations for individual users. The process involves several key steps, including clustering, model training, model evaluation, and implementation, which will be comprehensively discussed in the following section.

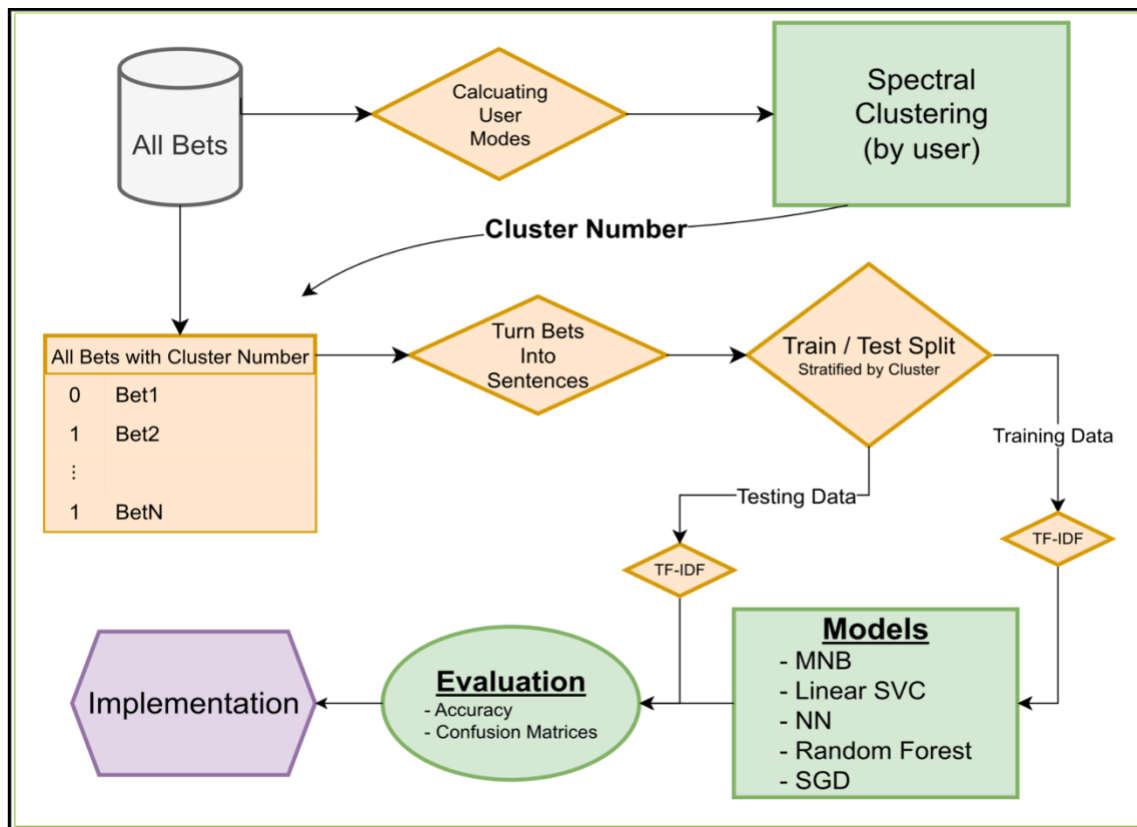


Figure 11: Process flow diagram of project.

3.1 Exploratory Data Analysis

After cleaning the data, the team analyzed the data through exploratory data analysis (EDA). EDA allowed the team to learn about the features of the dataset in an attempt to recognize patterns and trends in the data. EDA is an important step in understanding the data as well as aids in identifying which features might impact the model the most (Hartwig, 1979). The team divided the EDA into three different sections. First, the team learned about the users and their betting history. Second, the team explored the different types of bets that were being placed and if these bets were placed individually or in a parlay. Lastly, the team investigated the popularity of sports being bet on.

To gain insight into the dataset, the team started by looking into the number of users that were featured in the dataset and the total number of bets placed. The team found that there were 1,598 unique users and 781,799 unique bet tickets in the dataset (recall that a unique bet ticket can contain multiple individual bets). The team looked further into the users and generated box plots of the number of bets per user, which is depicted in Figure 12. The statistics from the boxplot of the number of bets per user are shown in Table 3.

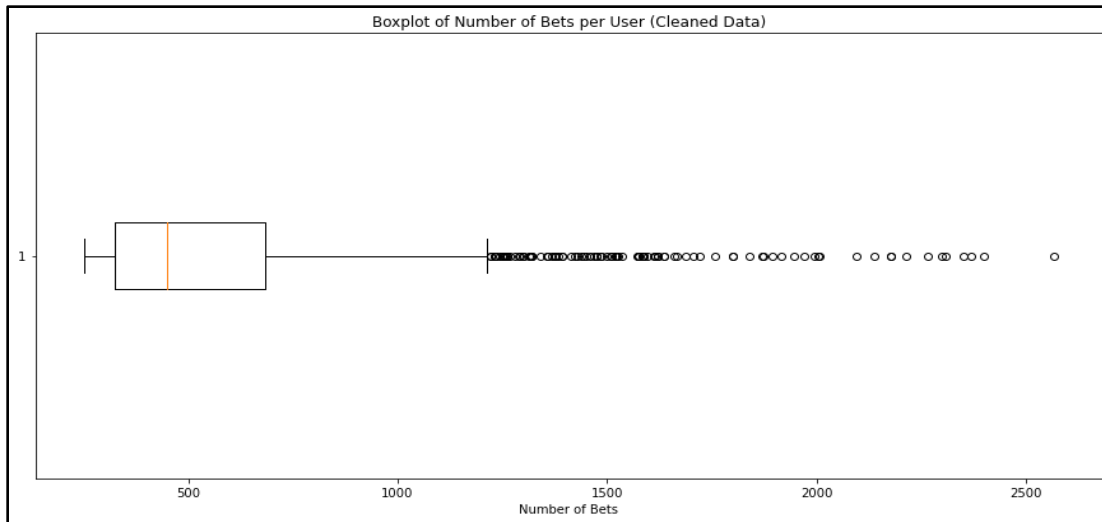


Figure 12: Boxplot of how many bets each user placed.

Statistics of Boxplot of the Number of Bets per User	
Low	2

Lower Whisker	250
Lower Box	324.5
Median	448
Mean	489.24
Upper Box	683
Upper Whisker	683
Max	2333

Table 3: Summarizes the statistics of the boxplot of the number of bets per user.

The next step of analyzing the user betting history was looking at the distribution of bets by the month placed. The next area the team explored was the different types of bets being placed and how they were being placed: either individually or in a parlay. Figure 13 illustrates the percentage of parlay bets versus individual bets. The team found that 64% of the bets in the dataset were not parlay, while the resulting 36% were parlay bets.

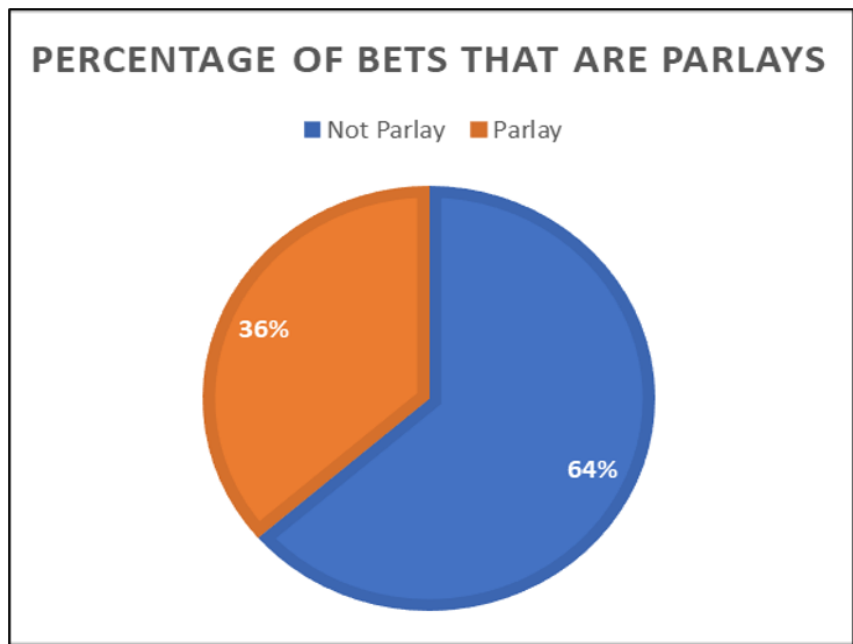


Figure 13: Comparison of parlay to individual bets in the data set.

For parlays, the team conducted research into finding out the lengths of the parlays in the dataset. The team provided a distribution of the number of legs in the parlays and statistics such as median, mean, maximum, and minimum. Figure 14 illustrates the distribution of the number

of legs per parlay. The team discovered the range of the legs from two to 20 legs. The team confirmed that parlays with lower number of legs occurred more often. Two-leg parlays were the most frequent parlays that were bet amongst users.

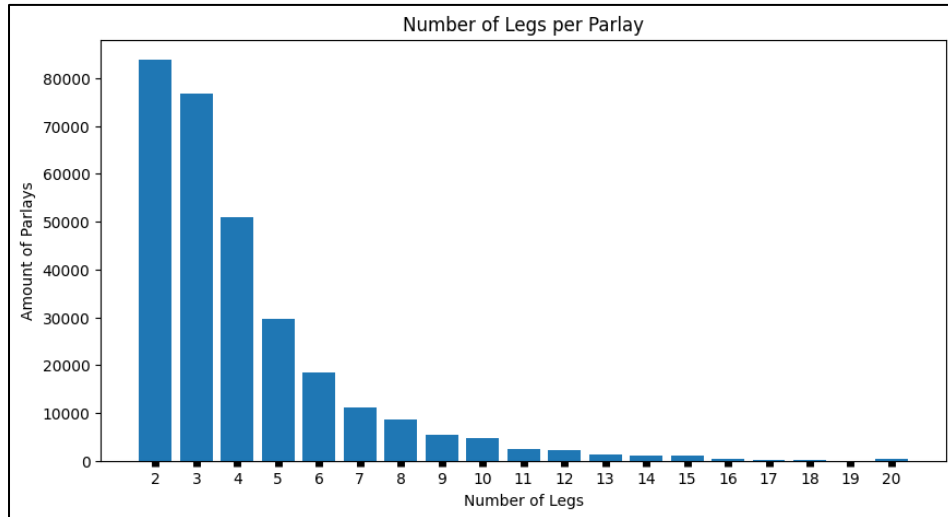


Figure 14: Distribution of number of legs per parlay.

Another feature of the dataset the team focused efforts into investigating was SPORT. The team looked at the distribution of the sport across all bets, individual bets, bets that were included in a parlay, and across the timeline of the data. The data consisted of 13 different sports, but as mentioned above, the team decided to focus on six main sports as they comprise the majority of the dataset. Figure 15 shows the number of bets that pertained to each sport. The team determined that NBA, NFL, MLB, CBB, CFB, and NHL make up 86.17% of the dataset and refer to these sports as the top 6 sports.

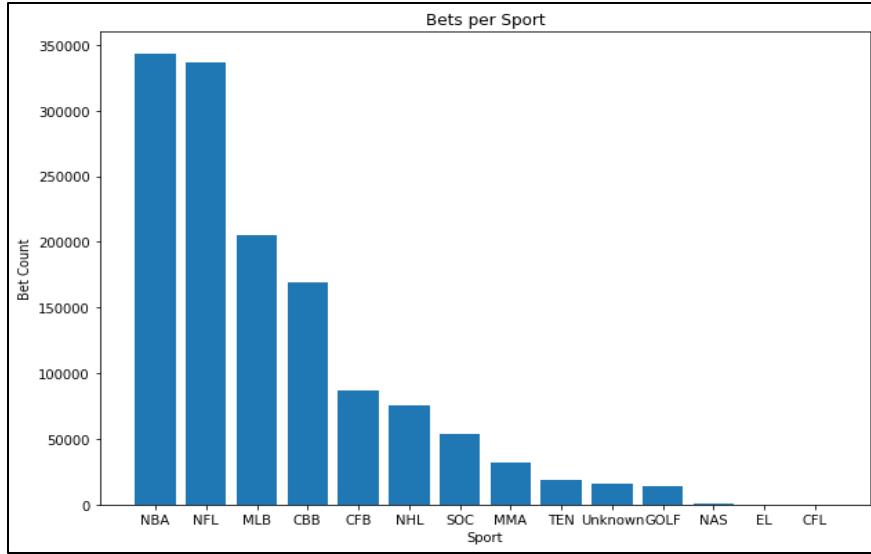


Figure 15: Distribution of the bets per sport when exploring the dataset of all the bets.

Figure 16 provides insight into how the number of bets change per month and how the total number of bets change over the course of a year. The biggest takeaways are that in football is the most popular sport during the NFL season and the summer months of July, August, and June have the least number of bets, which could be because the number of games across all six sports is less than in the other months.

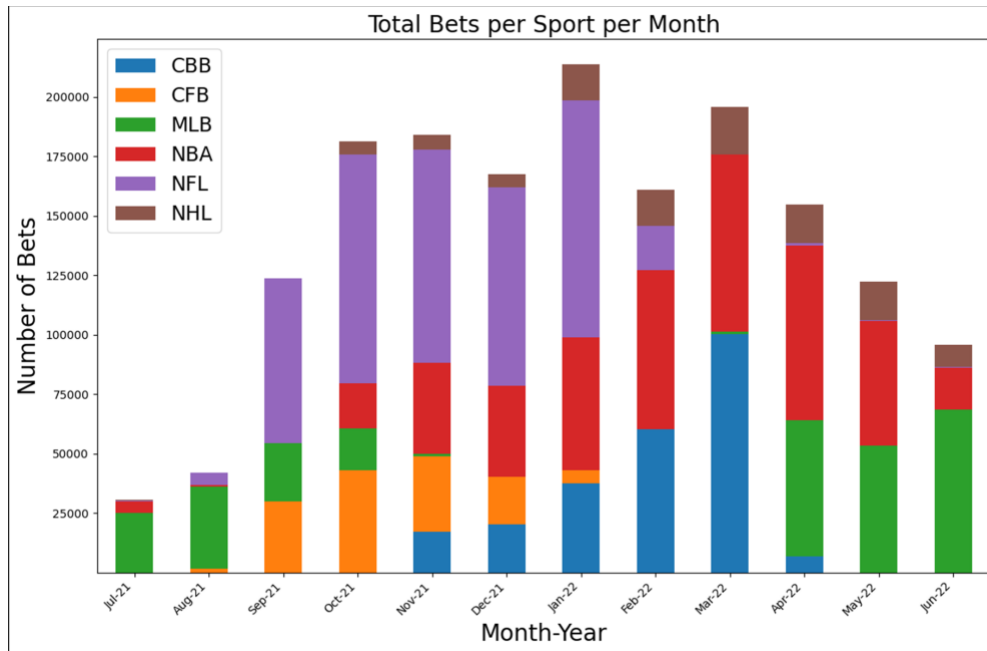


Figure 16: Depicts the total number of bets per sport per month.

Through the exploratory data analysis, the team learned about the dataset including the number of users, number of individual bet tickets, and of those bet tickets, what percentage were parlays. The team used the conclusions drawn during EDA to help learn how the data would be cleaned and what features would be cleaned. EDA helped the team decide on how to process the data and conduct multi-class classification on the dataset.

3.2 Cleaning and Preparing the Data

The first step in the machine learning process was to become familiar with the data. This included learning about the nature of the data as well as cleaning and preparing it for optimum use. This step was vital for data completeness, integrity, and reliability. See [Section 2.3 Dataset and Features](#) for detailed description of each attribute in the dataset.

The dataset provided by DraftKings contained approximately 1.9 million rows with 22 different features. It encompassed betting history from June 2021 to July 2022, involving 1598 unique users. To ensure that the behavior of each user could be tracked effectively, DraftKings imposed a limit of no more than 3,000 bets per individual user in our dataset. This restriction prevented any individual user from disproportionately influencing the data of the entire group.

The initial preprocessing steps involved populating missing data points within the data set. The columns `HOMETEAMNAME` and `AWAYTEAMNAME` contained many missing values. These missing values resulted from users placing bets on events that do not naturally have a “home” team or “away” team but are still associated with one or two teams. However, instead of excluding the column entirely, members of the team manually populated the missing values. Since these columns were crucial for understanding a user’s betting preferences, the group decided to keep them. Additionally, the team made the decision that there should be no distinction between a home team and an away team for judging a user’s preference, as both columns represented an instance of the user placing a bet that is relevant to those two teams. Henceforth, we treated `HOMETEAMNAME` and `AWAYTEAMNAME` as interchangeable. There were two main reasons for the missing team names:

1. The bet being a result of a series, particularly within the NBA and NHL where playoff results are determined by a multiple game series. Since both teams’ host games throughout a series, DraftKings does not record one team as the home team and the other

team as the away team for the bet pertaining to the entire series. However, for our purposes, we wanted to record that the bet was relevant to each of the two teams. In these cases, we were able to take the EVENT attribute and split it up in a way that separated those two team names and insert them into the HOMETEAMNAME and AWAYTEAMNAME attributes for that bet.

2. The bet being a prediction of a title winner (or a similar instance), such as betting which team will win the Super Bowl, World Series, AFC Championship, Eastern Conference Finals, AFC East, etc. Since the bet does not pertain to a specific game, the EVENT attribute reads "2021 Super Bowl Champion" or something similar. As a result, DraftKings does not store a home team or away team for that bet. For our purposes, we wanted to record that the bet is relevant to that team being bet on. In this case, we were able to simply take the LABEL column, which would be the team being bet on, and insert that value into HOMETEAMNAME. Since the bet is only relevant to one team, AWAYTEAMNAME would remain null.

Finally, the last step in the data cleaning and preparation process was to limit the sports for which we would create bet and parlay recommendations. Initially, we had planned to include all sports in the dataset. However, upon closer inspection, we found that the six most popular sports, namely NFL, NBA, MLB, CBB, CFB, and NHL, made up 86% of the total dataset as shown in Figure 17. We realized we did not have enough information for the less popular sports to accurately represent them in our model. Therefore, we made the decision to remove these less popular sports from consideration. As a result, we retained only the six sports (NFL, NBA, MLB, CBB, CFB, and NHL) that were previously mentioned for all future work with the data.

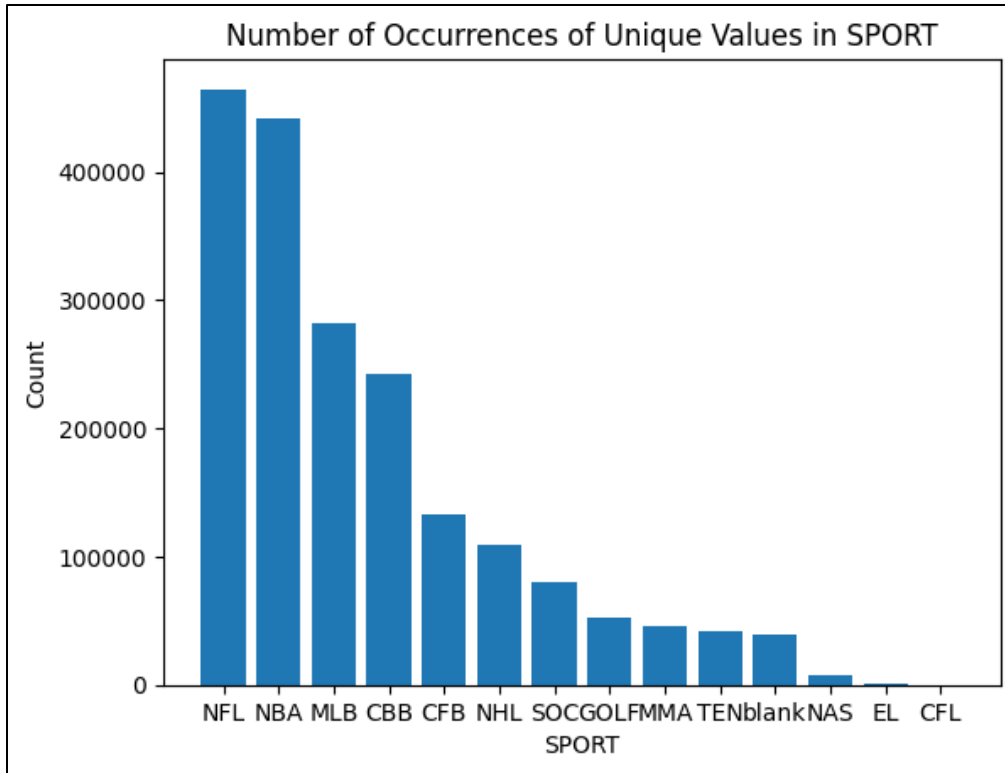


Figure 17: Distribution of all the different sports in the dataset before the removal of the less popular sports.

Filtering of the data by removing the less popular sports resulted in a remaining dataset of approximately 1.7 million rows and 15 features, with a total of 1597 unique users. Only one user was removed in this filtering process.

3.3 Feature Engineering and Selection

After EDA and data cleaning and preparation, we moved forward with additional feature engineering as discussed in the subsequent sections.

3.3.1 New Features

In order to move forward with comparing users and building machine learning models, we had to take some of the pre-existing features within the raw DraftKings data and transform it into new features that would be more useful for our purposes.

3.3.1.1 Bet Type

A large issue that we had in the dataset was the variability within the BETDESCRIPTION attribute. The raw data contained over 29,671 different BETDESCRIPTIONs. Since the

BETDESCRIPTION is one of the most important pieces of information regarding a bet, we knew that it would be an important attribute to use in our machine learning algorithms. This presented our team with 2 issues:

i. 1-off proposition bets:

Within the DraftKings app, users can choose to make a proposition bet out of anything imaginable. Out of the 29,671 different values of BETDESCRIPTION, 29615 of them are some form of a proposition bet, and most of them are not repeated, meaning that exact BETDESCRIPTION was only used a single time by a single user. Examples of this include betting on whether the 5th play of the 3rd Patriots drive of the game will be run or pass, betting number of assists a player will have, betting how many runs would be scored in the 2nd inning of the game, and much more.

ii. Slightly different versions of the same bet description:

Within the customization options that DraftKings provides for its users, they can choose to customize a normal bet in many ways. For example: a user can place a “Moneyline” bet, but they can also place a “Moneyline first half” or “Moneyline 3rd quarter” bet. The same customization applies to Spread bets and Total bets, and adds more variation to the dataset, where in reality, the different bets show nearly identical user behavior.

Given both of these issues with the variability of the BETDESCRIPTION, we choose to simplify the variability down to 4 distinct categories following the main types of bets present in the data set. Therefore, we created the new attribute BETTYPE that can take on one of 4 values:

$$\text{BETTYPE} = \{\text{“Moneyline”}, \text{“Spread”}, \text{“Total”}, \text{“Proposition”}\}$$

This simplification allows us to group together more effectively types of bets based on their core mechanics. Since the simplification from BETDESCRIPTION to BETTYPE does lose a lot of variability, we chose to not delete BETDESCRIPTION so the data would show that the two bets fall under the same umbrella in their BETTYPE, but they can still be different BETDESCRIPTIONs.

3.3.1.2 Normalized Wager

The 2nd new feature we created was related to the WAGERAMOUNT attribute in the dataset. Being numerical, the values of this attribute formed a continuous quantitative distribution as opposed to the distinct categorical values for most attributes. This made it harder

to cluster users, so in order to better group users together based on how much they tend to wager, we created an attribute called WAGERNORMALIZED that could take one 4 values corresponding to each of the 4 quartiles of the WAGERAMOUNT distribution. Table 4 shows the quartiles of the feature WAGERAMOUNT and the corresponding value.

Quartile	Wager Amounts	Wager Normalized Term
First	0.10 - 3.00	Low
Second	3.01 - 10.00	Medium
Third	10.01 - 21.50	Large
Fourth	21.51 – 54,800.00	Highest

Table 4: The distribution of wager amount and corresponding term for the quartiles.

This categorization of the numerical wager amount field allowed us to distinguish users who generally bet smaller amounts vs users who generally bet larger amounts.

3.3.1.3 Bets on Ticket

The third new feature we created was an extension of the ISPARLAY attribute. That attribute was a binary indicator of whether or not that bet was part of a larger parlay or not. We wanted one additional attribute indicating how many legs that parlay included. The new feature called BETSONTICKET is a numerical attribute that represents the number of bets that use this particular BETTICKETIDHASH. If a bet is not part of a parlay, BETSONTICKET will equal 1 because the bet is not part of a larger parlay. However, if the bet is part of a parlay, BETSONTICKET will equal the number of legs in that parlay. This attribute was useful to use for EDA and filtering purposes.

3.3.1.4 Timeline

The dataset originally included an attribute called STATE that tracked whether the bet was completed or not. To not confuse this attribute with the JURISDICTION attribute, we chose to rename STATE to TIMELINE to be clearer on its meaning.

3.3.2 Feature Selection

When deciding which features were important to include in our clustering and machine learning models, the biggest decision we made was to focus on the user's action of placing the

bet rather than the outcome of the bet. This is because regardless of what the outcome of the bet was, the user still placed that bet, and thus it is an example of that user's behavior and preferences. Therefore, any attribute that would not exist at the moment in time that the user placed the bet is not important to our machine learning models. With this line of thinking, we eliminated the BETSTATUS, PAYOUTAMOUNT, HOMETEAMSCORE, AWAYTEAMSCORE, and TIMELINE attributes were eliminated from our data.

Additionally, there were two groups of attributes that the team chose to exclude from this project. The first group of attributes were the geographic attributes of JURISDICTION and EXTERNAL_IDENIFIER. The main reason we chose not to use the user's location as a feature in our model was because DraftKings already utilizes the location in the recommendations they give to users.

The second group of attributes the team decided to not use was the odds attributes of ODDSAMERICAN, ODDSDECIMAL, ODDSFRACTIONAL. We chose not to use the odds as a factor in our machine learning models because since odds frequently change in the time period that a bet is open for, and two users may have placed the exact same bet on different odds, so it would be risky to try to connect the odds with the rest of the bet description attributes. Additionally, an issue that was present throughout the data was that many rows, particularly rows that were part of a larger parlay, would have null values in their odds columns. In Figure 18 below, we can see how the odds were left null in half of the legs of this six-leg parlay.

BETTICKETIDHASH	BETDESCRIPTION	LABEL	ODDSAMERICAN
dd4d59ad-86da-45ef-aa75-6e24b5287df2	Moneyline	WAS Nationals	+2000
dd4d59ad-86da-45ef-aa75-6e24b5287df2	Moneyline	CHI White Sox	NaN
dd4d59ad-86da-45ef-aa75-6e24b5287df2	Moneyline	MIL Brewers	NaN
dd4d59ad-86da-45ef-aa75-6e24b5287df2	Moneyline	NY Yankees	-157
dd4d59ad-86da-45ef-aa75-6e24b5287df2	Moneyline	SD Padres	+205
dd4d59ad-86da-45ef-aa75-6e24b5287df2	Moneyline	TOR Blue Jays	NaN

Figure 18: Example of six-leg parlay in dataset with missing values for the ODDSAMERICAN column.

The last attribute we chose to omit from our models was the EVENTGROUP attribute. In some sports outside of the major 6 (NBA, NFL, NHL, MLB, CBB, and CFB), the EVENTGROUP attribute would be more valuable (such as showing the tournament name for Golf), but in the 6 major sports, EVENTGROUP is nearly identical to SPORT except for some small cases. Examples include bets where SPORT = NFL and EVENTGROUP = NFL Preseason. These cases are small and inconsequential enough that we chose to eliminate the EVENTGROUP attribute and keep SPORT.

After eliminating all the above columns, we moved forward with our project using the following 11 attributes:

[USERHASHID, BETTICKETIDHASH, BETTYPE, BETDESCRIPTION, LABEL, BETSONTICKET, SPORT, EVENT, HOMETEAMNAME, AWAYTEAMNAME, WAGERNORMALIZED]

3.3.3 Bet Sentence Creation

To feed this data into our clustering and classification algorithms, we used the data from these rows to create “bet sentences” where the text from all of the remaining attributes in a row were added to one long string. This was so we could have our algorithms treat the bet as one entity. Each bet sentence is also attached to its own USERIDHASH, BETTICKETIDHASH and value for BETSONTICKET, so we now had the rows of the data in the format [USERIDHASH, BETTICKETIDHASH, bet sentence, BETSONTICKET].

Example of turning a bet into a “sentence”:

An example of an original bet can be found in Table 5 below.

USERIDHASH	f4ee3420-e31e-4cf6-bf64-adf2dabd992c
BETTICKETIDHASH	b32a5455-aa5e-41be-b98b-9b2f4f65aa68
PLACEDDATE	6/2/2022 8:44:42 PM
BETDESCRIPTION	Moneyline
LABEL	BOS Celtics
ISPARLAY	0

BETSTATUS	Bet Settled
WAGERAMOUNT	25
PAYOUTAMOUNT	60
ODDSAMERICAN	140
ODDSDECIMAL	2.4
ODDSFRACTIONAL	7/5
SPORT	NBA
EVENTGROUP	NBA
EVENT	BOS Celtics @ GS Warriors
HOMETEAMNAME	GS Warriors
AWAYTEAMNAME	BOS Celtics
HOMETEAMSCORE	108
AWAYTEAMSCORE	120
STATE	STARTED
JURISDICTION	IL
EXTERNALIDENTIFIER	US_ILLINOIS

Table 5: Example of original bet within dataset.

Taking the features in Table 5, the corresponding “bet sentence” is “Moneyline Moneyline BOS Celtics NBA BOS Celtics @ GS Warriors GS Warriors BOS Celtics Highest”.

3.4 Cluster Development

3.4.1 Mode Bet of the Users

One of the challenges presented in this project was dealing with the variety of bets available to users of DraftKings sportsbook and the corresponding variety of a typical user’s betting history. To effectively cluster users together into classes, the team chose to start our clustering algorithms by calculating the mode bet of each user. The mode bet of a user is defined as the collection of all the modes of all the attributes within that user’s betting history. Therefore, the mode bet for a user would be the most common value for each attribute in that user’s betting

history. If there was no mode of a bet attribute, the mode was determined by randomly selecting one attribute from the user’s history. Table 6 below shows an example of how the user modes calculation was done for our data.

User	Sport	HomeTeam	AwayTeam	BetType
USER1	NFL	New England Patriots	Philadelphia Eagles	Moneyline
USER1	NBA	Boston Celtics	LA Lakers	Proposition
USER1	MLB	New York Yankees	LA Dodgers	Moneyline
USER1	NBA	Boston Celtics	Miami Heat	Moneyline
USER1 MODE	NBA	Boston Celtics	Miami Heat	Moneyline

Table 6: Example of how a user’s ‘mode of bets’ is created for each user in the dataset.

Calculating the users’ modes allows us to be able to one hot encode the modes which was previously unable to be conducted on the full data due to it being too sparse. After one hot encoding is conducted the clustering algorithms can be used to cluster users based on their betting history.

3.4.2 One Hot Encoding

To transform the user bet information from categorical information about which events, teams, and sports the user bet on into numerical data for the machine learning algorithm to be able to process, the team used a one-hot encoding strategy. To do so, each column of categorical information was converted into a matrix where each column represents a categorical value. The matrix was then filled in with 1’s and 0’s, in which 1’s represented the existence of that specific categorical value in the column, and a 0 represented the absence of that value. Because one-hot encoding treats each value as a binary value, 1 or 0, the representation of the data is treated equally by the machine learning algorithm.

As a simple example, the HOMETEAMNAME column in the dataset would be separated into many different columns, each corresponding to the different HOMETEAMNAME values from the data. A user who placed a bet where the HOMETEAMNAME value was Boston Bruins, Boston Red Sox, or Boston Celtics would originally appear as seen in Table 7 below.

User	HOMETEAMNAME
User 1	Boston Bruins
User 2	Boston Red Sox
User 3	Boston Celtics

Table 7: Sample of user bet data before one-hot encoding.

After the one hot encoding has taken place, the bet information would resemble the structure in Table 8.

User	Boston Bruins	Boston Red Sox	Boston Celtics
User 1	1	0	0
User 2	0	1	0
User 3	0	0	1

Table 8: HOMETEAMNAME example after the data has been one-hot encoded.

3.4.3 Summary Matrix

After one hot encoding the data, we next planned to use a summary matrix to compare users based on their one hot encoded feature. The summary matrix proved to be a useful tool for identifying patterns and relationships between different features in the dataset. The goal was to use the summary matrix to establish relationships between users and their top features. The team aimed to cluster users with a high-ranking summary score (closer to 1) to represent closely related users. A snippet of the summary matrix can be found below in Figure 19.

	0	1	2	3	4	5	6	7	8	9	...	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596
0	1.000	0.375	0.250	0.250	0.125	0.625	0.125	0.375	0.375	0.500	...	0.500	0.500	0.250	0.250	0.250	0.375	0.125	0.125	0.375	0.125
1	0.375	1.000	0.125	0.375	0.250	0.375	0.375	0.375	0.125	0.250	...	0.375	0.125	0.125	0.500	0.375	0.500	0.375	0.250	0.500	0.375
2	0.250	0.125	1.000	0.250	0.250	0.375	0.375	0.250	0.250	0.375	...	0.375	0.125	0.125	0.250	0.375	0.125	0.250	0.375	0.250	0.250
3	0.250	0.375	0.250	1.000	0.375	0.375	0.375	0.375	0.125	0.125	...	0.375	0.000	0.125	0.375	0.500	0.250	0.250	0.375	0.500	0.250
4	0.125	0.250	0.250	0.375	1.000	0.250	0.500	0.375	0.250	0.125	...	0.250	0.000	0.250	0.250	0.375	0.375	0.500	0.625	0.375	0.250
...
1592	0.375	0.500	0.125	0.250	0.375	0.375	0.375	0.375	0.250	0.250	...	0.250	0.250	0.250	0.375	0.250	1.000	0.375	0.375	0.375	0.375
1593	0.125	0.375	0.250	0.250	0.500	0.125	0.250	0.250	0.125	0.250	...	0.125	0.000	0.000	0.375	0.250	0.375	1.000	0.375	0.250	0.500
1594	0.125	0.250	0.375	0.375	0.625	0.250	0.625	0.375	0.250	0.125	...	0.250	0.000	0.250	0.375	0.375	0.375	0.375	1.000	0.500	0.250
1595	0.375	0.500	0.250	0.500	0.375	0.500	0.375	0.500	0.125	0.250	...	0.500	0.125	0.250	0.375	0.500	0.375	0.250	0.500	1.000	0.250
1596	0.125	0.375	0.250	0.250	0.250	0.125	0.250	0.250	0.250	0.250	...	0.125	0.000	0.000	0.375	0.375	0.375	0.500	0.250	0.250	1.000

Figure 19: Snippet of summary matrix for all users.

3.4.4 Elbow Curve Method

An important computation that was needed to optimize the model’s performance was the number of clusters that the team separated our users into. To do so, the team plotted an elbow curve using K-means clustering with K values from 1-20. At each value of K, the Elbow curve plots the SSE for each number of clusters. In doing so, it is possible to calculate the optimal number of K clusters when the diminishing returns are not worth the additional cost. For this section, the group used the pyclustering library in python. The pyclustering library is a clustering package that contains different clustering algorithms as well as a function to perform the elbow curve. This function allows the user to store the optimal number of clusters which reduces the human error of picking the point by eye (Umargono et al., 2020). The result of the elbow curve is depicted in Figure 20 below. It is clear to see that the last large decrease in SSE is from 5 total clusters to 6 total clusters and then after 6 it is a stable decrease creating the elbow at 6 total clusters. See also [Section 2.6.1 Elbow Curve Method](#).

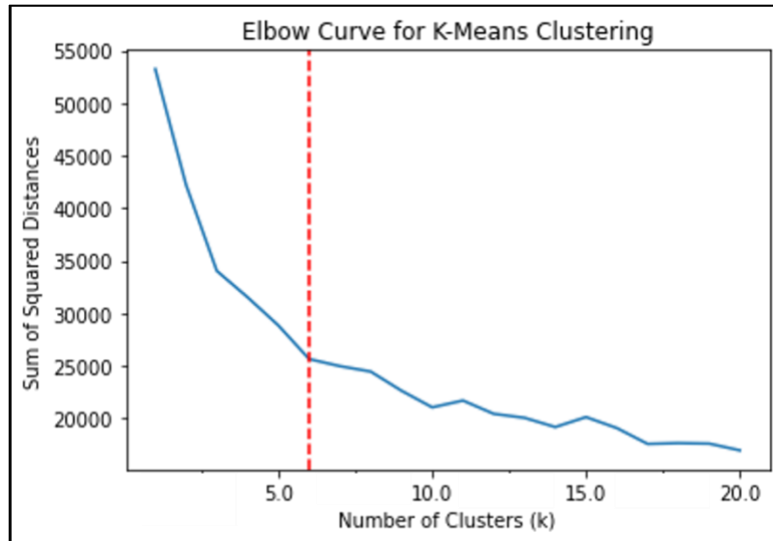


Figure 20: K-means clustering elbow curve used to determine optimal K number of clusters ($K=6$).

Having done this, the team separated our data into 6 different clusters and tested different clustering techniques and compared them with different metrics to determine which clustering algorithm provided the most accurate results when recommending both individual bets and parlays to users in the dataset. Because the team needed static clusters to finalize results, the elbow curve is tailored to the number of clusters from K-means clustering.

3.4.5 Evaluation and Selection of Clustering Methods

In order to determine which combination of clustering algorithms was the most effective in separating users into distinct groups, the team evaluated eight clustering algorithms against three clustering evaluation indexes. The clustering algorithms included K-means (One Hot Encoded), K-means (Summary Matrix), KMeans (Jaccard/Summary Matrix), SOMSC (OneHot), SOMSC (Summary Matrix), SOMSC (Jaccard/Summary Matrix), Spectral Clustering, and Hierarchical Clustering, each of which are described in more detail in [Section 2.7 Clustering Evaluation Methods](#).

3.5 Cluster Evaluation

In order to evaluate how different clusters performed, each clustering algorithm was evaluated by multiple clustering evaluation methods. These methods included Silhouette Score, Calinski-Harabasz Score, and Davies-Bouldin Score, which are described in detail in [Section 2.7](#)

Clustering Evaluation Methods. By evaluating the eight clustering algorithms against the three clustering evaluations, the team was able to gain insight as to which algorithm was the most accurate for our model. In order to evaluate the performance of the algorithms, the team imported python's sklearn package. Importing the silhouette_score, calinski_harabasz_score, and davies_bouldin_score features from the package provided insight as to how each algorithm compared to the rest in different measures.

3.6 TD-IDF and Train/Test Split

Once the team determined which clustering method performed best via cluster evaluations, we kept the user cluster classifications fixed for the rest of the methods moving forward. After joining the cluster information back into the original dataset, the users' bet modes sentences, which was previously created as described in Section 3.4.1 Mode Bet of the Users, now have a designated cluster assignment associated with them. With the data cleaned, prepared, and subjected to feature engineering with a target variable, it was ready for machine learning. The first step of the machine learning process involved splitting the data into training and testing sets, with 80% of the data allocated for training and 20% for testing. The data was stratified by cluster to ensure that both the train and test sets have equal distribution. Figure 21 shows the distribution of each cluster for the train and test sets.

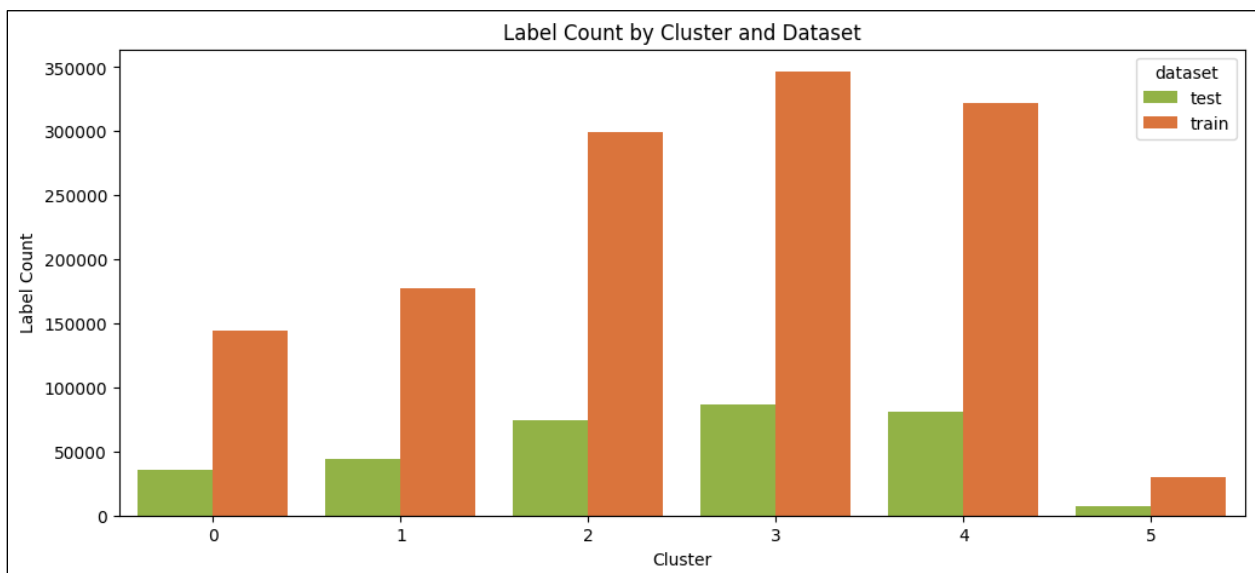


Figure 21: 80/20 train test split by each cluster.

The next step involved applying the Term Frequency-Inverse Document Frequency (TF-IDF) technique to the training data, which transformed it into a matrix of weighted word frequencies (Kim & Gil, 2019). Section 2.5.1 Term Frequency- Inverse Document Frequency (TF-IDF) contains further details on TF-IDF. For example, in a cluster that might contain the words ‘Moneyline, BOS Celtics, NBA, GS Warriors, BOS Celtics, highest’, the word ‘BOS Celtics’ may have a higher weight compared to ‘NBA’ or ‘moneyline’ due to its higher frequency and lower occurrence in other clusters. This technique ensured that the most important words in each sentence and cluster were given more weight in the model. Now that the data was split and transformed using TF-IDF, it was ready to be used in machine learning models.

3.7 Classification Algorithms Implementation

The problem at hand is using textual data of bets to identify the cluster of users who are most likely to make those bets. To achieve this, classification was performed to identify distinct clusters and label bets and parlays according to their associated clusters. There were three machine learning algorithms used for classification. These algorithms were chosen based on their suitability for the evaluation of textual data in the context of our problem. These algorithms are available in the popular sklearn library, which provides a wide range of machine learning algorithms for various tasks, including the classification necessary to solve this problem.

In addition to sklearn, our team also utilized TensorFlow, a powerful Python package developed by Google, for building neural networks. The team chose to use a neural network for the problem as neural networks often perform better than standard machine learning algorithms. TensorFlow offers functionality for constructing neural networks through its sub package Keras. For this project there were two different types of neural networks used to classify the clusters, a Deep Neural Network (DNN) and a Long-Short Term Memory (LSTM) neural network. The following sections will discuss the implementation using the sklearn library as well as the implementation when using the TensorFlow library.

3.7.1 Sklearn

Sklearn, is short for scikit-learn and is the name of the import for the python library. This library provides tools for machine learning and data analysis. Its functionalities include machine learning algorithms, tools for data preprocessing, model evaluation, and training and testing data

creation. Sklearn library offers a wide range of machine learning algorithms that are suitable for various tasks, including classification, which is necessary to solve our problem of classifying user clusters based on bet information.

The decision to use Naive Bayes, SGDClassifier, and LinearSVC algorithms for classification in the sklearn library is based on their suitability for the problem at hand. Naive Bayes is a probabilistic algorithm that is simple to implement and has been known to perform well in many text classification scenarios. SGDClassifier is a linear classifier that is efficient in handling large datasets and high-dimensional data. Since dimensionality from the text data is large, the SGDClassifier was also chosen. LinearSVC is an efficient and scalable linear classification algorithm. Overall, due to the problem dealing with textual data which requires efficient and scalable algorithms, Naive Bayes, SGDClassifier, and LinearSVC algorithms fit this description.

To implement Naive Bayes, SGDClassifier, and LinearSVC, there are three functions that can be imported from sklearn library. After these are imported algorithms can be initialized and then fit using the training X and Y data, where the Y data corresponds to the labels from the clustering step. After this is complete, the training and testing accuracies and confusion matrices can be observed to evaluate the model's performance, which can be found in [Section 3.8 Implementing Evaluation Metrics](#).

3.7.2 TensorFlow

In order to build neural networks in python, our team used the python package TensorFlow, developed by Google. TensorFlow allows users to process data and train a variety of machine learning models, most prominently Neural Networks through the sub package, Keras. TensorFlow encompasses many differences of the machine learning process, from being capable of storing, pre-processing, model training/testing, and storing results. For our project, the team chose to develop two different types of Neural Networks, a Deep Neural Network, and a more advanced Long-Short Term Memory (LSTM) Neural Network.

TensorFlow models have to take in data in the form of TensorFlow objects, so before the data could be passed into the model, the data needed to be converted into sparse Tensor objects. The DNN used the exact same preprocessing steps within Sklearn as the previous models. Due to

dimensionality issues of the data and how TensorFlow has their LSTM layers setup to use 3-dimensional data rather than 2-dimensional data, the LSTM models had to undergo a slightly different preprocessing setup that utilized TensorFlow's built-in encoding layers to handle textual data.

3.8 Implementing Evaluation Metrics

As discussed in the [Section 2.9 Evaluation Metrics](#), selecting a metric is a particularly important step in the machine learning process. There are a plethora of options to evaluate how the multi-class classification algorithms perform, but the team wanted one that represented the unique data the most and provided interpretable results.

Although all the listed metrics are valid and provide insight about the relative success of the classification model, precision, recall and the F1 score were the best fit for our model. In making predictions about users' betting habits, the model created many instances of true positives, false positives, true negatives, and false negatives. With these results, the model generated a large number of proportions which provide the team with the necessary insight to how well it is performing. Precision, recall, and F1 score were the most suitable to deal with these proportional values and return the most balanced product. Having the F1 score factor both precision and recall into its calculation strengthens this balance and requires that both precision and recall have a reasonable value to return a high score. Furthermore, by using precision as a metric, the team utilized confusion matrices to further understand areas of the model where misclassification of bets into the different clusters occurred.

To implement the metrics, the team utilized sklearn to ensure consistency across the classification algorithms besides for the neural network, which implemented tensorflow. The team used sklearn's `metrics.classification_report` package, which generated a report with precision, recall, and F1-score of each cluster. The report also returned the overall accuracy of the model and the macro average and weighted average of each cluster's precision, recall, and F1-scores, which averaged to create a measure for the entire model. For the confusion matrices, the team applied sklearn's confusion matrix which can be found in the metrics package. The confusion matrix gave the team insight into the correct and incorrect predictions of the multi-class model for each cluster. From here, the team learned common mistakes of the models and had a visual tool, which expressed totals of correct and incorrect clustering.

To keep consistency across the model's metrics, the team used accuracy as the metric when measuring the success of the neural network that was created in TensorFlow. Accuracy was the metric that was used across all of the classification algorithms that utilized sklearn. To specify that accuracy was the metric of choice, the team created the architecture of the neural network and compiled the model using categorical cross entropy as the loss function, adam as the optimizer, and accuracy as the metrics. Following compiling the model, the team tested the model, returning loss and accuracy measurements for each epoch.

4. Results

Our goal was to provide personalized recommendations for users on DraftKings by suggesting bets and parlays based on their tendencies. To achieve this, the team first analyzed the DraftKings bet data and determined the optimal number of clusters to divide the 1597 users. As discussed in the Methods chapter (see, e.g., [Section 2.6.5 Spectral Clustering](#)), the team identified spectral clustering as the best user clustering method. Next, the team employed machine learning algorithms, including Multinomial Naive Bayes, Stochastic Gradient Descent, Linear Support Vector Classifier, Deep Neural Networks, and Long Short-Term Memory Neural Networks, to associate a given bet or parlay with a specific cluster. An in-depth description of these machine learning algorithms can be found in [Section 2.8 Classification Algorithms](#). After evaluating the performance of these algorithms using precision, recall, and F1 score metrics at both macro and micro levels, the team found that Deep Neural Networks yielded the best results among the multiclass classification algorithms considered. This section provides an in-depth analysis of the performance of our spectral clustering method and the performance of the five different machine learning models and discusses how we compared the results to determine the best approach for associating bets and parlays with clusters.

4.1 Cluster Exploration

4.1.1 Determination of Optimal Number of Clusters and Clustering Algorithm

The Elbow Curve Method, which is presented in further detail in [Section 3.4.4 Elbow Curve Method](#), determined that six clusters would be the optimal number of clusters that could correctly represent the users and their corresponding bets for the individual bet data. After determining the number of clusters, the team looked at different clustering algorithms to find the best fitting algorithm. Table 9 below describes the clustering evaluation results of the different clustering algorithms. [Section 2.6 Clustering Techniques](#) and [Section 2.7 Clustering Evaluation Methods](#) refer to the clustering algorithms and clustering evaluation techniques found in Table 9. As previously mentioned, the three different clustering evaluation methods, Silhouette Coefficient, Calinski-Harabasz Index, and Davies-Bouldin Index, have different interpretations for their results.

Clustering Algorithm	Silhouette Coefficient	Calinski-Harabasz Index	Davies-Bouldin Index
KMeans (OneHot)	-0.035	0.892	22.945
KMeans (Summary Matrix)	-0.018	0.819	34.906
KMeans (Jaccard/Summary Matrix)	0.136	188.881	2.909
SOMSC (OneHot)	-0.009	0.974	24.216
SOMSC (Summary Matrix)	-0.019	1.020	31.068
SOMSC (Jaccard/Summary Matrix)	-0.019	0.9349	32.848
Spectral Clustering	0.114	194.949	2.173
Hierarchical Clustering	-0.237	1.085	18.741

Table 9: Model evaluation techniques and their corresponding clustering technique scores used to determine the overall best evaluation procedure.

The cluster evaluation discovered that KMeans with Jaccard Similarity on the Summary Matrix and Spectral Clustering were the two algorithms that best separated the data. After further analysis of the cluster evaluation, the team determined Spectral Clustering was the better algorithm as it had the highest Calinski-Harabasz Index result, 194.949, the lowest Davies-Bouldin Index score, 2.173, and the second highest Silhouette Coefficient, 0.114, which was only behind the KMeans with Jaccard Similarity.

4.1.2 Exploratory Data Analysis on Bet Distribution in Clusters

With the six clusters created using the individual bet data, the team conducted exploratory data analysis on the clusters to better understand areas of overlap and the representation of the clusters. The distribution of the number of bets per cluster is shown below in Figure 22. The range of the clusters is from 433,247 to 37,787 bets. Cluster 3 has the highest number of bets and Cluster 5 has the least number of bets. The quantitative values are shown in Table 10.

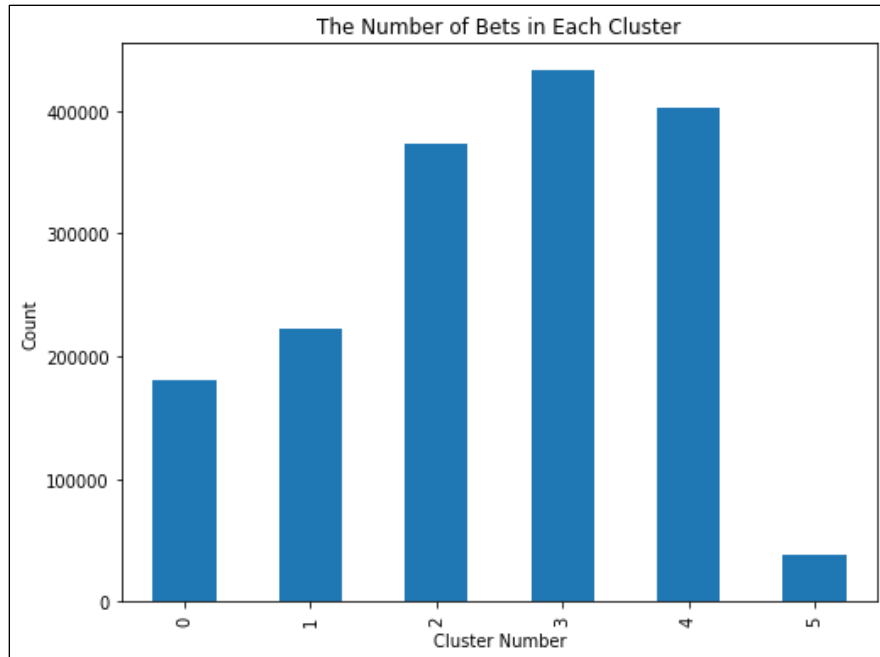


Figure 22: Distribution of number of individual bets within each cluster.

Cluster Number	Number of Bets
Cluster 0	190,546
Cluster 1	222,255
Cluster 2	373,668
Cluster 3	433,257
Cluster 4	402,839
Cluster 5	37,787

Table 10: Number of bets per cluster.

Next, the team looked at the distribution of the clusters and learned about the representation of the cluster. The team investigated attributes of the dataset the clusters were grouped on. Figure 23 is a heatmap which illustrates the correlation of terms to clusters. The heatmap helped the team determine what bet terms were significant to each cluster. Cluster 0 has a strong representation of bets that correlate on the spread. Cluster 1 contained bets that included the highest wager amount. Moneyline was the most recurring term in Cluster 2, meaning that this

cluster contained mainly moneyline bets. For Cluster 3, which is the largest cluster, there was a disbursement of moneyline and proposition bets with three different sized wager amounts. Cluster 4 had the highest count of a term between all the clusters. Bets with a low wager amount was the most common found word in Cluster 4. Lastly, in Cluster 5, there was a smaller number of total correlated words in the cluster, which was because it was the smallest cluster. There correlated words found in Cluster 5 were spread bets and highest wagered amount bets.

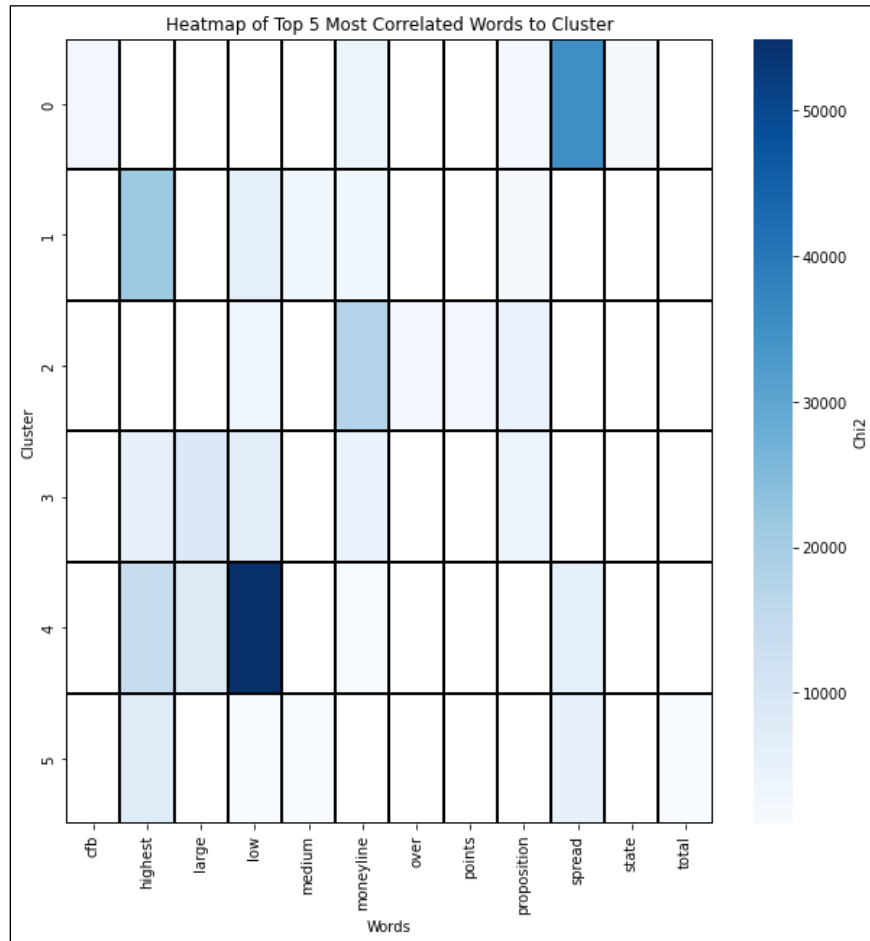


Figure 23: Heatmap of the most correlated words in each cluster.

As the clusters were created based on users and by taking their most frequently occurring instance of an attribute, not based by bet ticket, the team recognized that concerns were to be had about the same betting lines being placed into multiple clusters. The team created clusters to signify similarities in users and thus, grouped users together. Figure 24 illustrates that users can make the same bet, but not be seen as similar users as bets do appear in more than one cluster.

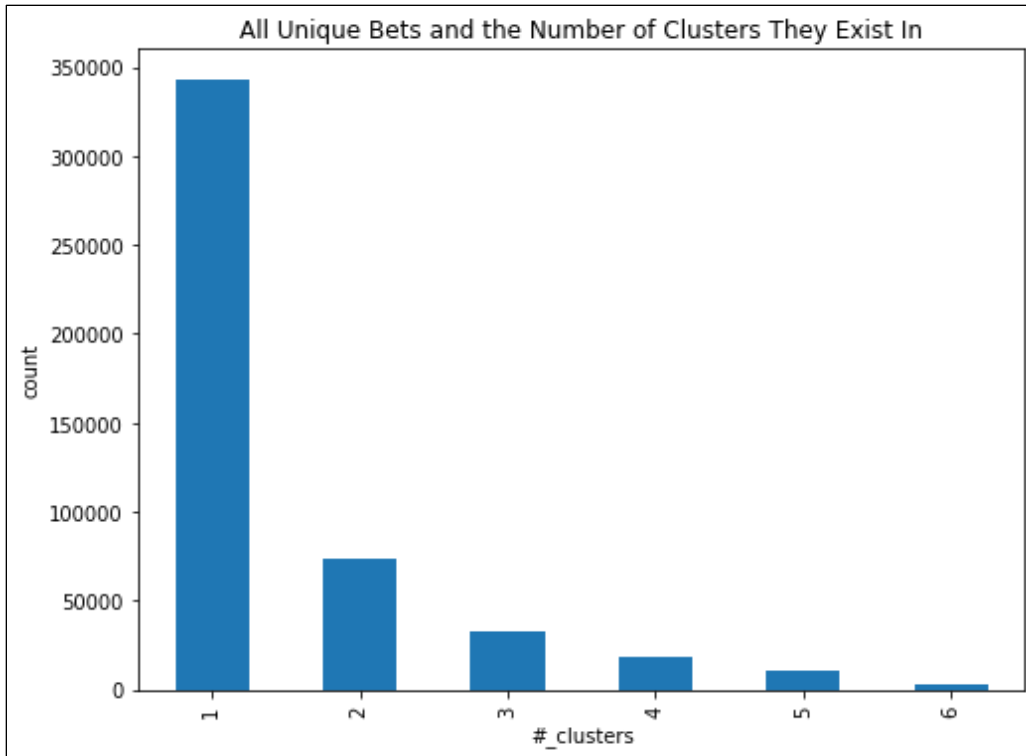


Figure 24: The number of clusters each bet appears in.

Figure 24 provides a visual of the distribution of unique bets among the number of clusters the bet is found in. The graph shows that the number of clusters a bet belongs to decreases as the number of total clusters increases. The highest number of unique bets, which is 71.28% of all unique bets, belong to only one cluster, which further confirms the differentiation of the clusters.

After running initial data analysis on the clusters of the individual bet data, the team applied the same methodology to two-leg parlay data. The dataset consisted of 82,031 two-leg parlays. The data was divided into six clusters and Figure 25 shows the distribution of the two-leg parlays among the clusters.

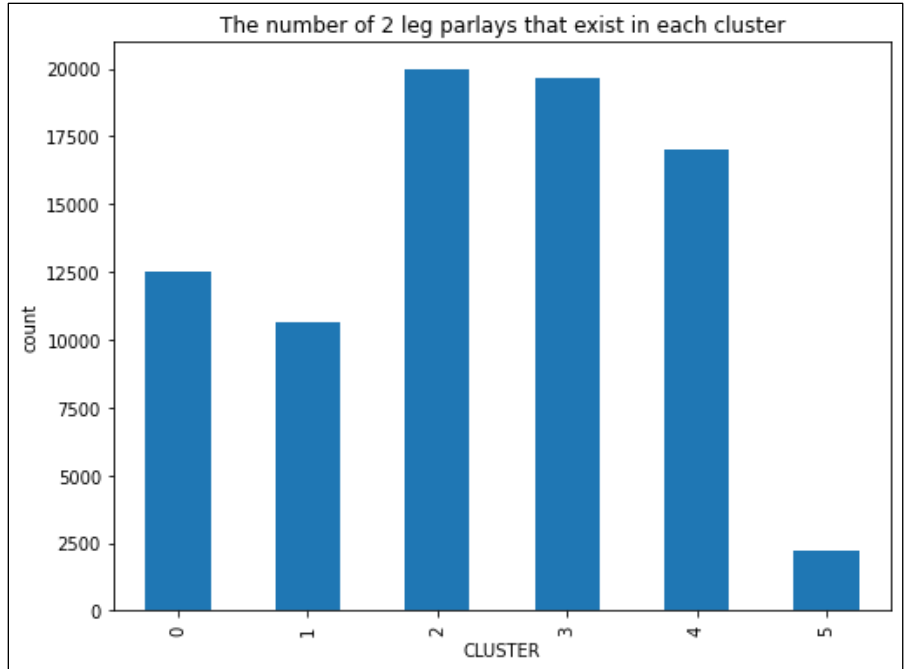


Figure 25: Distribution of number of two-leg parlays in each cluster.

Cluster 2 had the highest number of bets whereas in the distribution of the individual bet data, Cluster 3 had the highest total of bets. Similarly, Cluster 5 had the lowest total of bets for both datasets. A main difference in the distribution of the clusters from different datasets. The highest number of two-leg parlays in a cluster is roughly 20,000, while the largest cluster of individual bets was over 430,000.

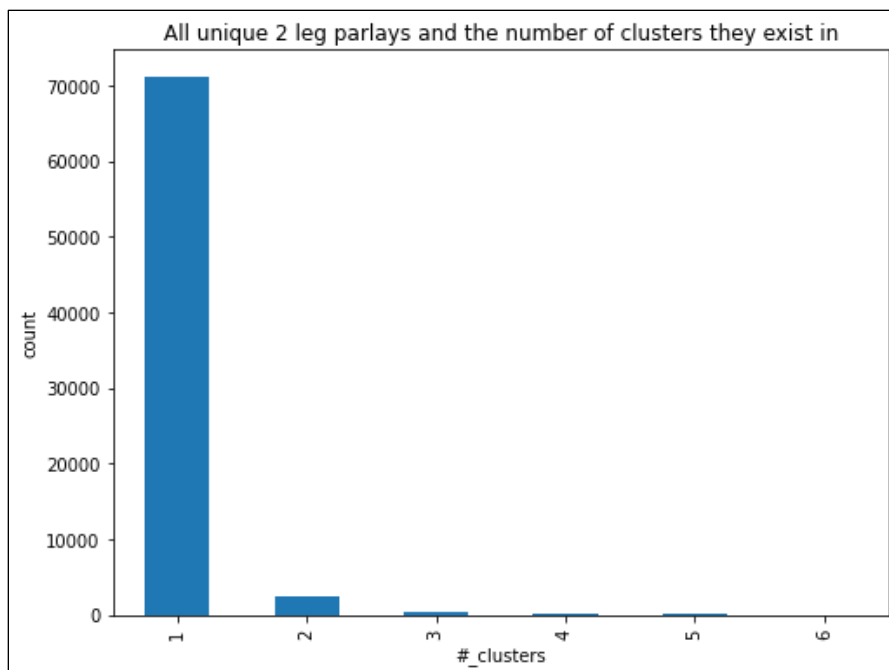


Figure 26: Number of clusters two-leg parlays exist in.

As previously mentioned, the team recognized that bets have the potential to appear in multiple clusters as bets were clustered by user’s mode bet. Figure 26 shows the distribution of bets across the number of clusters. The two-leg parlay clusters proved further improvement on the clustering algorithm as the team saw a significant decrease in the number of bets that appeared in more than one cluster. 71,159 unique two-leg parlays were found to be in only one cluster. Out of the 74,189 unique parlay bets, only 3,030 parlay bets were found to be in multiple clusters, showing that 95.9% of the parlay bets were only in one cluster.

Lastly, the team applied the same methodology to all parlays in the DraftKings dataset. This dataset, which we call “all-leg parlays”, included the two-leg parlays studied previously as well as all parlays with more than two-legs. The DraftKings data included parlays with as many as 20 legs. The distribution of parlays across the six clusters shows similar trends to the distribution of the individual and the two-leg parlay data. Figure 27 shows the number of parlay bets that exists in the six clusters. The total number of parlays in the data set is 294,740. Cluster 3 is the largest cluster with 81,224 parlays. The next largest cluster is Cluster 4 with 73,391 parlay bets. When comparing the distribution graphs of the individual bets, two-leg parlay, and all-leg parlay data, the team discovered that the trends in the individual bet data and the parlay data were similar as the ranking of the number of bets per cluster were in the same order. The

two-leg parlay data showed differences in the ranking as Cluster 2 had the largest count of the two-leg parlays, followed by Cluster 3, which is opposite for the individual bet data and the parlay data.

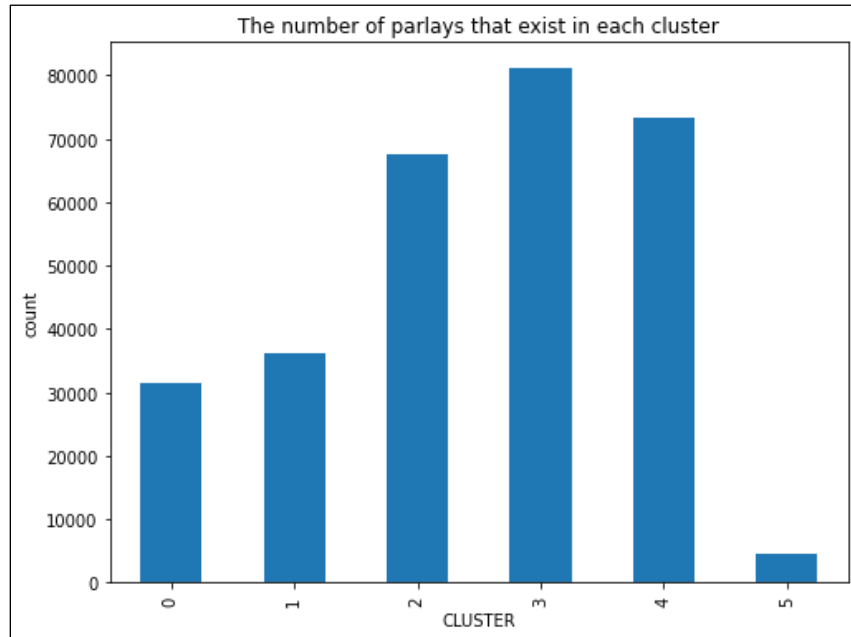


Figure 27: Distribution of all-leg parlays in each cluster.

Figure 28 depicts the number of unique parlays and the number of clusters the parlay appears in. In the parlay dataset, there are a total of 283,900 unique parlays that were made by DraftKings Sportsbook users. When analyzing the clusters, the team concluded that 98.56% of the unique parlays were found in only one cluster. Once again, the number of unique bets that appeared in multiple clusters declined as the number of clusters increased.

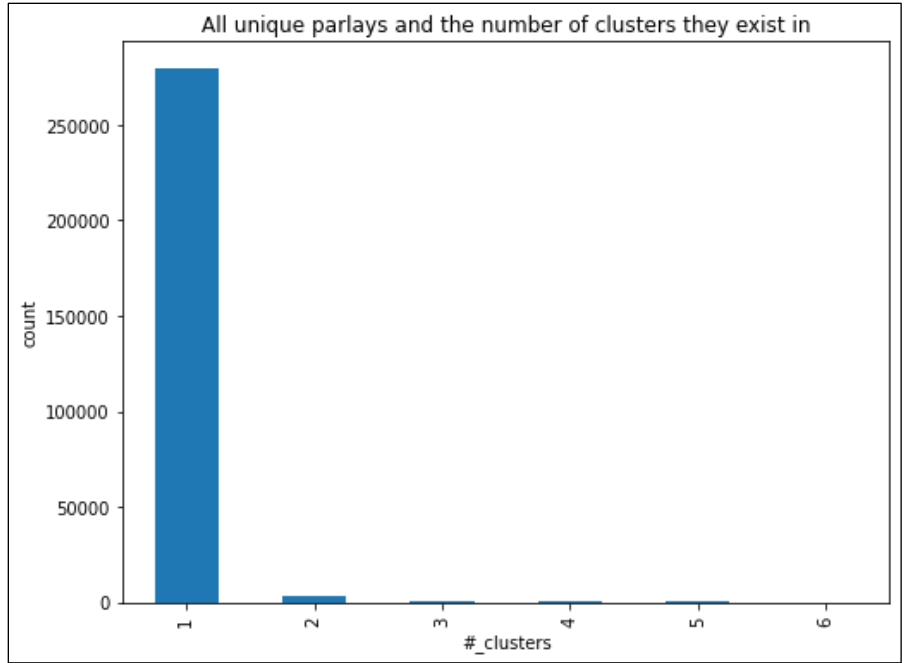


Figure 28: Number of clusters all-leg parlays exist in.

The clustering of the all-leg parlay data had the best results in separating unique parlays into the same cluster in comparison to the individual bet and two-leg parlay clusters.

4.2 Multi-Class Classification Models for Bet and Parlay Association

Once the target variable of predetermined clusters was created, learning algorithms were utilized and compared based on performance. The goal for these algorithms was to learn from a training set of data of users who belonged to certain clusters and therefore had certain tendencies with which to associate. Bets that were a part of the testing set, stratified by user, were then assigned to clusters based on the different attributes that make up the bet. The algorithms’ successes were measured using the metrics of macro and micro precision and recall that were discussed previously in [Section 2.9.2.1 Macro and Micro Precision and Recall](#).

The idea was that assigning bets to certain clusters would provide DraftKings with the opportunity to provide like-users with suggested bets that appealed to them. The algorithms that were utilized were Multinomial Naive Bayes, Stochastic Gradient Descent, Linear Support Vector Classifier, Deep Neural Networks, and Long Short-Term Memory Neural Networks, which are explained in more detail in [Section 2.8 Classification Algorithms](#). After observing the performance of the algorithms on all individual bets, the same algorithms and processes were

repeated for two-leg parlays as well as all-leg parlays. As mentioned previously, the team used accuracy as the metric to maintain consistency when comparing different algorithms. This choice was made because accuracy was consistent across all algorithms, regardless of the package used.

4.2.1 Baseline Models

Before evaluating the performance of the learning algorithms, the team established a random baseline for comparison. In addition to the precision and recall metrics, the team used the performance of the random models as a baseline to compare the effectiveness of the algorithms. If the algorithms were not superior to random models, then there would be no practical value in implementing them.

4.2.1.1 Random Cluster Association

Taking all the data, when the team randomly assigned the test set of bets to one of the six clusters it resulted in an accuracy score of 0.167521, or about 16.8%. This was expected because assigning a bet randomly to 1 out of 6 clusters would have a 16.66% chance of each cluster getting selected.

The full classification report of the completely random model for individual bets can be seen in Figure 29. Across the board, the metrics and calculations were fairly low. The precision is highest for clusters 2, 3, and 4, with a maximum of 0.26, which was not surprising as these were the largest clusters for individual bets. The macro average precision and the weighted average precision (weighted by the size of the clusters) were also still fairly low at .17 and .21, respectively. The classification reports for two-leg parlays and all-leg parlays were essentially the same besides which cluster was the most populous and therefore containing the highest precision values. These classification reports can be found in [Appendix B](#).

	precision	recall	f1-score	support
0	0.11	0.17	0.13	36109
1	0.14	0.17	0.15	44451
2	0.23	0.17	0.19	74734
3	0.26	0.17	0.20	86650
4	0.24	0.17	0.20	80568
5	0.02	0.16	0.04	7557
accuracy			0.17	330069
macro avg	0.17	0.17	0.15	330069
weighted avg	0.21	0.17	0.18	330069

Figure 29: Classification report of random baseline model for individual bets.

4.2.1.2 Most Popular Cluster Association

To create a higher standard to compare the models to, the team also created a baseline of accuracy if the most populated cluster was chosen every time. This calculation had an accuracy score of 0.262521, or 26.3%, meaning that if every bet in the test set was assigned to the most popular cluster, cluster 3 in the case of individual bets, this would result in the bets being correctly assigned to a cluster 26.3% of the time. This accuracy stays the same for two-leg parlays and all-leg parlays except for which cluster is the most populous. The most popular cluster for two-leg parlays is cluster 2 and all-leg parlays is cluster 3. The only other difference to note was the support, or the number of instances in each cluster, which was significantly smaller because the two-leg parlay and all-leg parlay data was a smaller sample than that of individual bets.

4.2.2 Multinomial Naive Bayes (MNB) Model

Overall, the performance of the Multinomial Naive Bayes (MNB) algorithm was good across the three different sets of data that were tested. For each set of data, training and testing accuracy scores were calculated, as well as cross-validation scores. The results showed that the scores for these calculations remained consistent across individual bets, two-leg parlays, and all-leg parlays. Moreover, the training and accuracy scores for each set of data were also consistent with each other, which indicated good performance and no overfitting.

4.2.2.1 Individual Bets

When utilizing MNB on all the bets in the dataset, the accuracy scores for testing and training were remarkably similar, at about 48%. The three model evaluations of the MNB can be seen in Table 11.

Model Evaluation	Accuracy
Training	0.476783
Cross Validation	0.445178
Testing	0.475464

Table 11: Model evaluation for MNB on individual bets.

In addition to these accuracies being similar, the other metrics the team calculated also stayed true to this consistency. Figures 30 and 31 compared the classification reports of the training and testing. The only differences were one disparity in both cluster precision and recall. This consistency in calculations across the board suggested the model is performing well on new data and can be a reliable output.

	precision	recall	f1-score	support
0	0.31	0.36	0.33	144437
1	0.50	0.38	0.43	177804
2	0.45	0.48	0.47	298934
3	0.51	0.48	0.49	346597
4	0.69	0.57	0.63	322271
5	0.15	0.49	0.23	30230
accuracy			0.48	1320273
macro avg	0.43	0.46	0.43	1320273
weighted avg	0.51	0.48	0.49	1320273

Figure 30: Training classification report for individual bets.

	precision	recall	f1-score	support
0	0.30	0.36	0.33	36109
1	0.50	0.38	0.43	44451
2	0.45	0.48	0.47	74734
3	0.51	0.48	0.49	86650
4	0.69	0.57	0.63	80568
5	0.14	0.48	0.22	7557
accuracy			0.48	330069
macro avg	0.43	0.46	0.43	330069
weighted avg	0.51	0.48	0.49	330069

Figure 31: Testing classification report for individual bets.

In addition to the classification reports for the training and testing, the confusion matrices for training and testing are also provided in Figures 32 and 33. The confusion matrices between training and testing represent the same patterns of classification. The project team can address that this confusion matrix looks significantly better than that of the random model. It can be seen

that cluster 4 has the highest number of correct classifications. When cluster 4 was misclassified, it was prominently classified as cluster 2 and cluster 3. The next highest number of correct classifications goes to cluster 3 followed by cluster 2. The misclassification for cluster 3 is found to be cluster 2 and cluster 0. The misclassification for cluster 2 is found to be cluster 3 and cluster 0. From the confusion matrix, it can be seen that the MNB model for individual bets has trouble differentiating between clusters 2 and 3 mostly as well as cluster 0 being predicted often when it is not the right cluster.

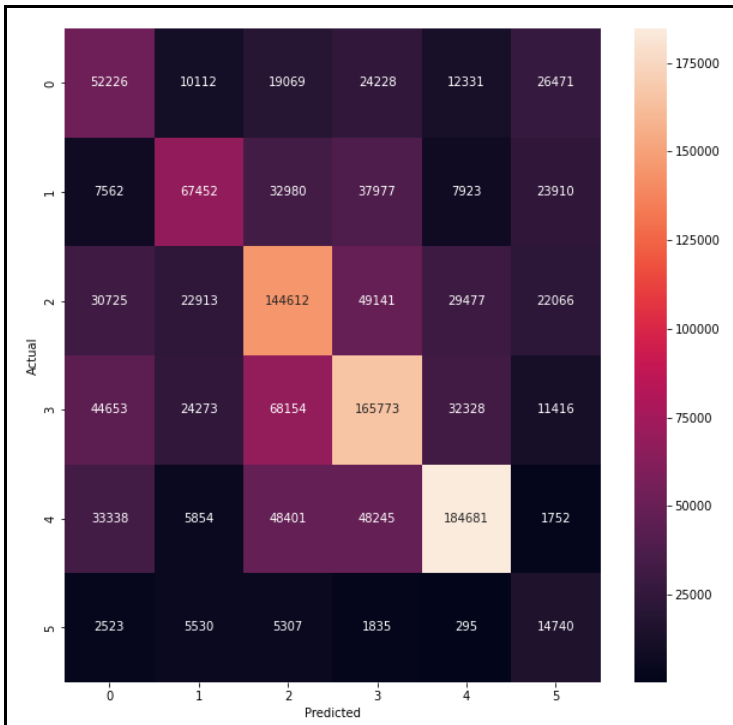


Figure 32: Training confusion matrix for individual bets.

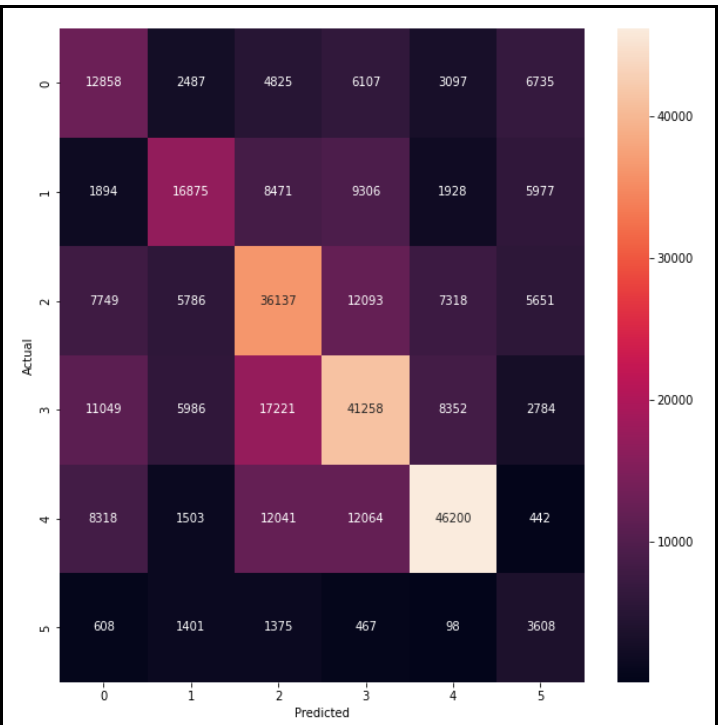


Figure 33: Testing confusion matrix for individual bets.

Compared to both of our random baseline models, this performance was significantly better. Even when looking at the bias model of always selecting the most popular cluster, MNB performed almost 85% better. This could be due to the fact that MNB does well at classifying text and feature categorization.

4.2.2.2 Two-leg Parlays

When performing MNB on two-leg parlay data, the accuracies did decrease compared to the individual bets data, but not a significant amount. The accuracies sat around 43%, which

again, was still consistent between the training, cross-validation, and testing as well as was a significant improvement from the random baseline models. The accuracy scores can be found in Table 12.

Model Evaluation	Accuracy
Training	0.438849
Cross Validation	0.430026
Testing	0.436887

Table 12: Model evaluation for MNB on two-leg parlays.

Comparing the training and testing classification reports for two-leg parlays, there were a few more minor discrepancies between individual precision and recall calculations. These small differences did not affect the overall accuracy, macro averages for precision and recall, or weighted average for precision and recall to much extent. The classification reports can be seen in Figures 34 and 35.

	precision	recall	f1-score	support
0	0.36	0.42	0.39	10012
1	0.44	0.36	0.40	8518
2	0.49	0.40	0.44	15985
3	0.45	0.40	0.43	15713
4	0.65	0.58	0.61	13610
5	0.12	0.52	0.20	1786
accuracy			0.44	65624
macro avg	0.42	0.45	0.41	65624
weighted avg	0.48	0.44	0.45	65624

Figure 34: Training classification report for two-leg parlays.

	precision	recall	f1-score	support
0	0.36	0.42	0.39	2503
1	0.45	0.37	0.40	2130
2	0.48	0.40	0.43	3997
3	0.45	0.41	0.43	3928
4	0.63	0.57	0.60	3403
5	0.12	0.51	0.20	446
accuracy			0.44	16407
macro avg	0.42	0.44	0.41	16407
weighted avg	0.47	0.44	0.45	16407

Figure 35: Testing classification report for two-leg parlays.

Comparing the training and testing confusion matrices for two-leg parlays, the result was found to be pretty much the same as that of the individual bets. The highest number of correct predictions was cluster 4, followed by clusters 2 and 3. The misclassification of clusters 2 and 3 was still very apparent as well as the over classification of cluster 0 that was mentioned in the individual bets section. Overall, the MNB model seemed to have the same issue for individual bets as well as two-leg parlays.

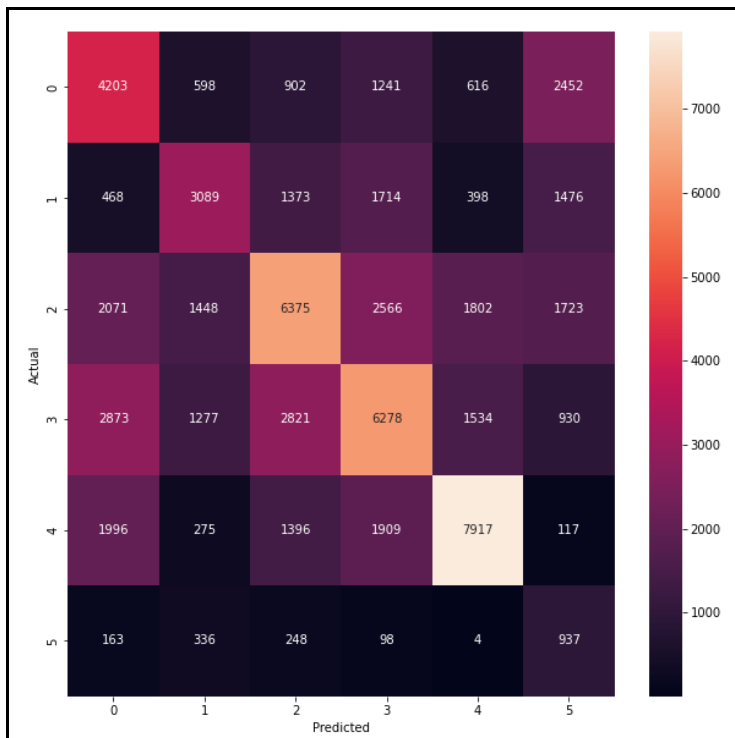


Figure 36: Training confusion matrix for two-leg parlays.

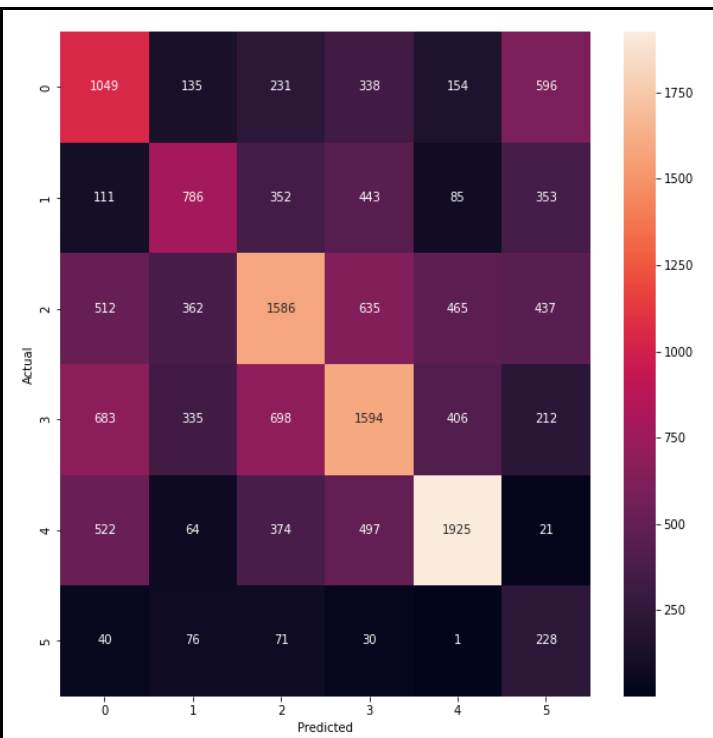


Figure 37: Testing confusion matrix for two-leg parlays.

A major difference between two-leg parlays and individual bets was the difference in the sample size. The support counts and averages were significantly smaller in the two-leg parlay data, which could be the reasoning for the decrease in performance of the MNB.

4.2.2.3 All-Leg Parlays

Lastly, the team looked at how well MNB performs on all-leg parlays. These model evaluation accuracy scores were the highest out of the three forms of data. The training and testing accuracies were around 48.5% while the cross-validation accuracy was about 45%. This difference could suggest a small degree of overfitting in the model. See Table 13 with these values.

Model Evaluation	Accuracy
Training	0.484737
Cross-Validation	0.453048
Testing	0.489347

Table 13: Model evaluation for MNB on all-leg parlays.

Similar to the previous data samples, the classifications reports had only a few small discrepancies between calculations, as shown in Figures 38 and 39. The minor differences only resulted in a .01 difference in the weighted averages for precision. Again, these metrics showed significant improvement from the random baseline models.

	precision	recall	f1-score	support
0	0.32	0.44	0.37	25281
1	0.46	0.38	0.42	28977
2	0.48	0.45	0.47	54110
3	0.52	0.50	0.51	64979
4	0.71	0.57	0.63	58713
5	0.10	0.44	0.17	3732
accuracy			0.48	235792
macro avg	0.43	0.46	0.43	235792
weighted avg	0.52	0.48	0.50	235792

Figure 38: Training classification report for all-leg parlays.

	precision	recall	f1-score	support
0	0.33	0.45	0.38	6321
1	0.46	0.38	0.42	7244
2	0.48	0.46	0.47	13527
3	0.53	0.50	0.51	16245
4	0.71	0.58	0.64	14678
5	0.10	0.42	0.16	933
accuracy			0.49	58948
macro avg	0.43	0.46	0.43	58948
weighted avg	0.53	0.49	0.50	58948

Figure 39: Testing classification report for all-leg parlays.

As shown in the individual bets and two-leg parlays, the training and testing confusion matrices for all-leg parlays are provided in Figures 40 and 41 below. As was outlined in the confusion matrices for individual bets and two-leg parlays, the clusters with the most correct classifications were 4, 3, and 2. The problems of misclassification of clusters 2 and 3 as well as the over classification of cluster 0 were still apparent in the confusion matrices. All three of the datasets experience the same type of misclassification for the MNB models.

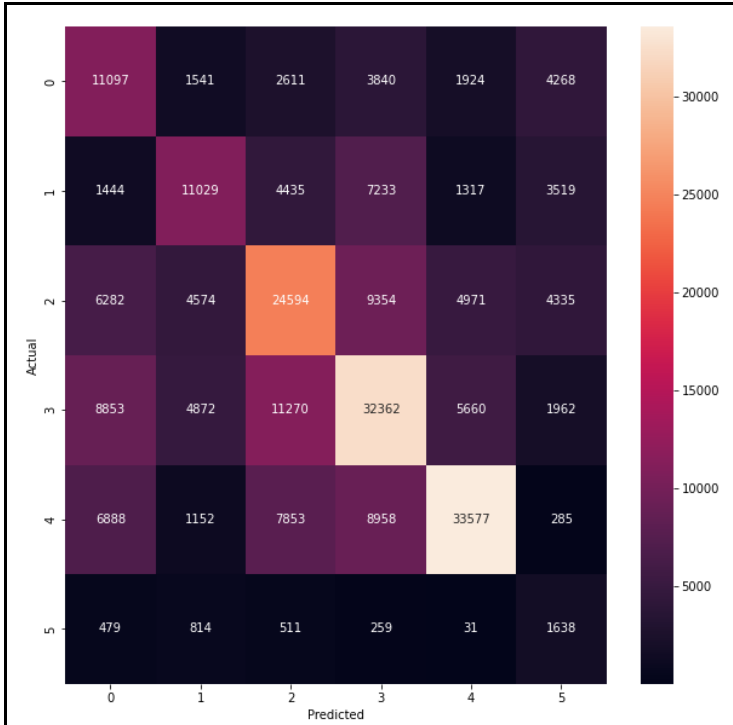


Figure 40: Training confusion matrix for all-leg parlays.

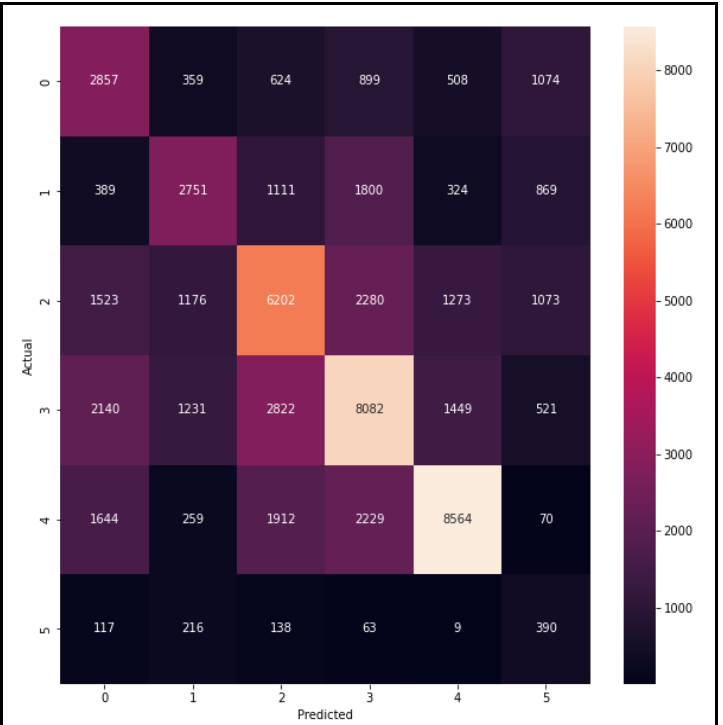


Figure 41: Testing confusion matrix report for all-leg parlays.

A difference to draw attention to would be the sample size of the all-leg parlays. This sample size was a small portion compared to that of the data sample with individual bets. This could have an impact on the performance of the model, as less data points were included.

4.2.3 Stochastic Gradient Descent (SGD) Model

Across the same three datasets, we calculated the accuracy of SGD. Similar to other algorithms, the team evaluated the model over training, cross-validation, and testing accuracy. In general, the model was much better at predicting individual bets than both two-leg parlays and all-leg parlays. The accuracy results can be seen below.

4.2.3.1 Individual Bets

When looking at individual bets, SGD produced similar training and testing accuracy results, both of which were around 0.436. The exact results are displayed below in Table 14.

Model Evaluation	Accuracy
Training	0.436000
Cross Validation	0.503829
Testing	0.435148

Table 14: SGD accuracy for individual bets.

A review of the classification reports for the SGD model on all individual bets, as seen below in Figures 42 and 43, reflects the precision, recall, and f1-score of the model across the different clusters. As seen in these reports, clusters 2, 3, and 4 yielded much better results than the other clusters in the dataset.

	precision	recall	f1-score	support
0	0.31	0.40	0.35	144437
1	0.50	0.36	0.42	177804
2	0.38	0.54	0.45	298934
3	0.51	0.48	0.49	346597
4	0.68	0.35	0.46	322271
5	0.15	0.42	0.22	30230
accuracy			0.44	1320273
macro avg	0.42	0.43	0.40	1320273
weighted avg	0.49	0.44	0.44	1320273

Figure 42: SGD training classification report for individual bets.

	precision	recall	f1-score	support
0	0.30	0.40	0.34	36109
1	0.51	0.36	0.42	44451
2	0.38	0.54	0.45	74734
3	0.51	0.48	0.49	86650
4	0.68	0.35	0.46	80568
5	0.14	0.42	0.21	7557
accuracy			0.44	330069
macro avg	0.42	0.42	0.40	330069
weighted avg	0.49	0.44	0.44	330069

Figure 43: SGD testing classification report for individual bets.

Upon review of the confusion matrices from the SGD model against individual bets further exemplifies the differences in how each cluster performed. Like the classification report recorded, clusters 2, 3, and 4 returned better results than the other three clusters. It can also be recognized that misclassification occurs prominently between clusters 2, 3, and 4.

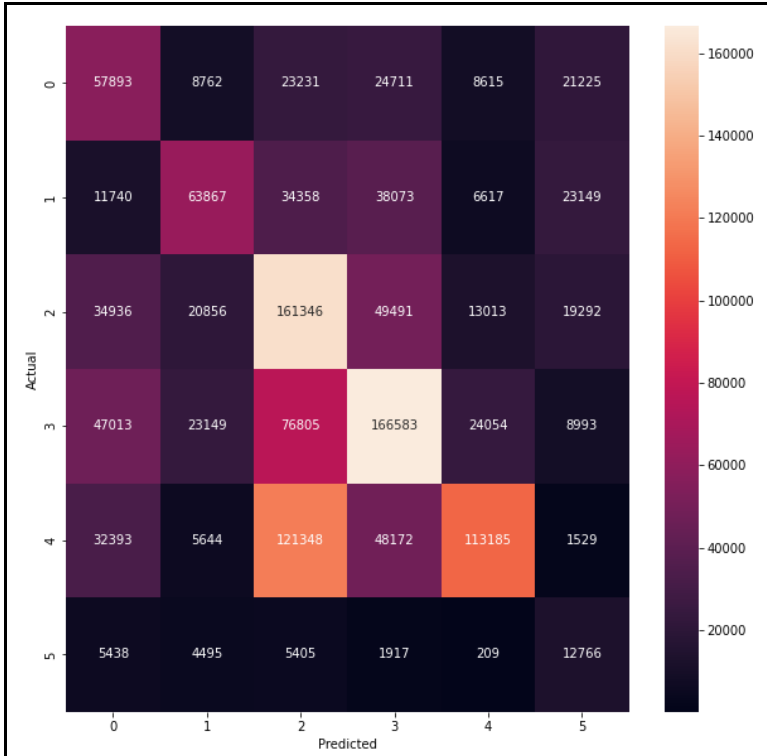


Figure 44: SGD training confusion matrix for individual bets.

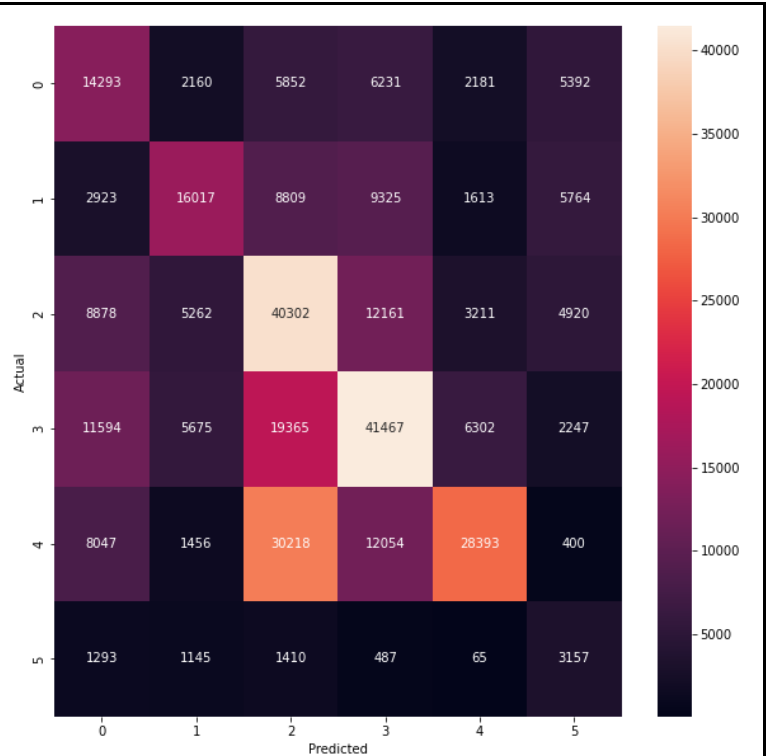


Figure 45: SGD testing confusion matrix for individual bets.

4.2.3.2 Two-leg Parlays

When introducing the SGD model to two-leg parlays, the accuracy notably dropped from the previously calculated 0.436 to roughly 0.382 on both testing and training data. However, this is still a noticeable improvement from the random baseline model. The calculations can be seen in Table 15.

Model Evaluation	Accuracy
Training	0.382268
Cross Validation	0.462147
Testing	0.382824

Table 15: SGD accuracy for two-leg parlays.

The corresponding classification reports for training and testing data on the SGD model display the same decrease in accuracy for each cluster. However, it can be seen in this report that

the model did significantly better on cluster 0 when dealing with two-leg parlays as opposed to all individual bets. The classification reports for SGD on two-leg parlays is recorded in Figures 46 and 47.

	precision	recall	f1-score	support
0	0.36	0.52	0.43	10012
1	0.28	0.43	0.34	8518
2	0.43	0.43	0.43	15985
3	0.41	0.44	0.42	15713
4	0.67	0.16	0.25	13610
5	0.13	0.17	0.15	1786
accuracy			0.38	65624
macro avg	0.38	0.36	0.34	65624
weighted avg	0.44	0.38	0.37	65624

Figure 46: SGD training classification report for two-leg parlays.

	precision	recall	f1-score	support
0	0.37	0.52	0.43	2503
1	0.28	0.43	0.34	2130
2	0.43	0.44	0.43	3997
3	0.42	0.45	0.43	3928
4	0.66	0.15	0.24	3403
5	0.13	0.15	0.14	446
accuracy			0.38	16407
macro avg	0.38	0.36	0.34	16407
weighted avg	0.44	0.38	0.37	16407

Figure 47: SGD testing classification report for two-leg parlays.

The confusion matrices for two-leg parlays on the SGD model gives clearer insight as to how well the model performed within each cluster. As previously stated, the model yielded much better results for cluster 0 compared to all individual bets. However, the model still does not have very accurate results with cluster 5. Here it can be seen that there were a lot of misclassifications happening in this model for two-leg parlays. Here clusters 2, 3, and 4 are misclassified as 0, clusters 3 and 4 are misclassified as 1, clusters 3 and 4 are misclassified as cluster 2, and clusters 0, 2, and 4 are misclassified as cluster 3. From this the team concluded that SGD did a poor job at differentiating the two-leg parlays to their clusters. The confusion matrices can be seen in Figures 48 and 49.

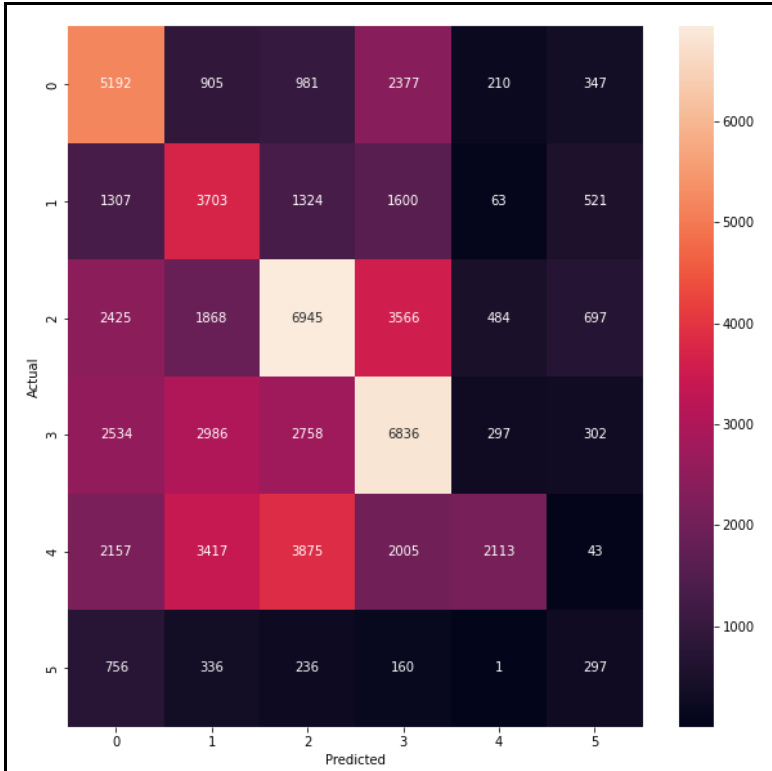


Figure 48: SGD training confusion matrix for two-leg parlays.

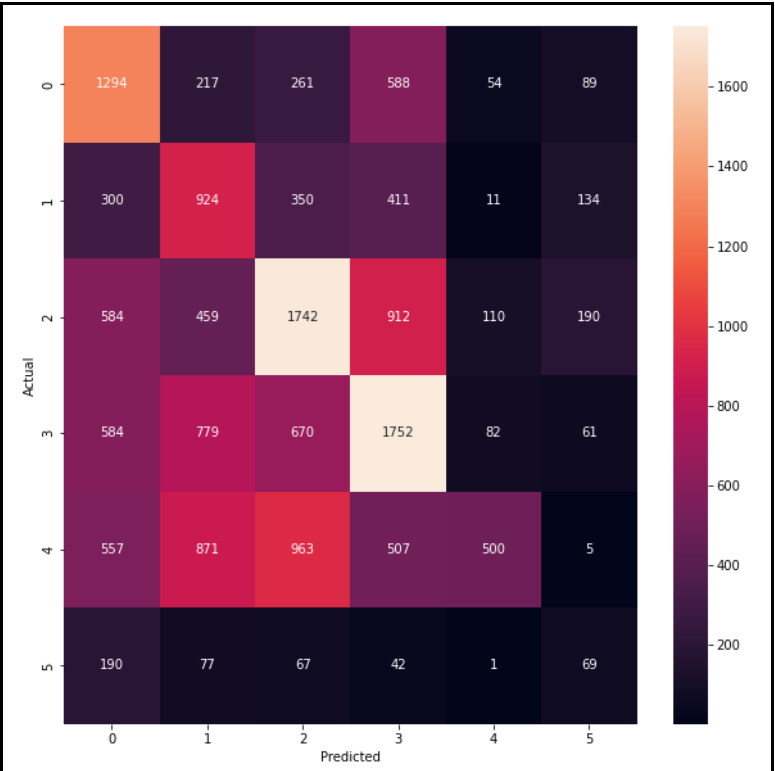


Figure 49: SGD testing confusion matrix for two-leg parlays.

4.2.3.3 All-Leg Parlays

Surprisingly, the SGD model did slightly better when dealing with all-leg parlays. The training and testing accuracies increased a single percent from 0.382 to approximately 0.393. The cross-validation calculation also increased from two-leg parlays to all-leg parlays. The exact results can be seen in Table 16.

Model Evaluation	Accuracy
Training	0.390958
Cross Validation	0.484240
Testing	0.393007

Table 16: SGD accuracy for all-leg parlays.

The following training and testing classification reports in Figures 50 and 51 are consistent with the training and testing accuracies reported for the SGD model on all-leg parlays. From these results, clusters 2 and 3 appear to be returning the best results. The model yielded the best f1-score for cluster 2 on all-leg parlay data than either of the two previous datasets.

	precision	recall	f1-score	support
0	0.27	0.57	0.36	25281
1	0.34	0.41	0.37	28977
2	0.41	0.54	0.46	54110
3	0.54	0.45	0.49	64979
4	0.50	0.12	0.20	58713
5	0.10	0.18	0.13	3732
accuracy			0.39	235792
macro avg	0.36	0.38	0.33	235792
weighted avg	0.44	0.39	0.38	235792

Figure 50: SGD training classification report for all-leg parlays.

	precision	recall	f1-score	support
0	0.27	0.58	0.37	6321
1	0.33	0.42	0.37	7244
2	0.41	0.54	0.47	13527
3	0.54	0.44	0.49	16245
4	0.50	0.12	0.19	14678
5	0.09	0.16	0.12	933
accuracy			0.39	58948
macro avg	0.36	0.38	0.33	58948
weighted avg	0.44	0.39	0.38	58948

Figure 51: SGD testing classification report for all-leg parlays.

The matrices in Figures 52 and 53 further prove this point. As displayed, clusters 2 and 3 do much better than the other clusters on this dataset using this model. Like in previous evaluations, the model still struggles with classifying cluster 5. Misclassifications of 3 and 4 where they are classified as 2 is relevant here as it was in the individual bets. It can also be mentioned that clusters 3 and 4 are also misclassified as 0. There was also a lot of misclassifications that was apparent in cluster 4.

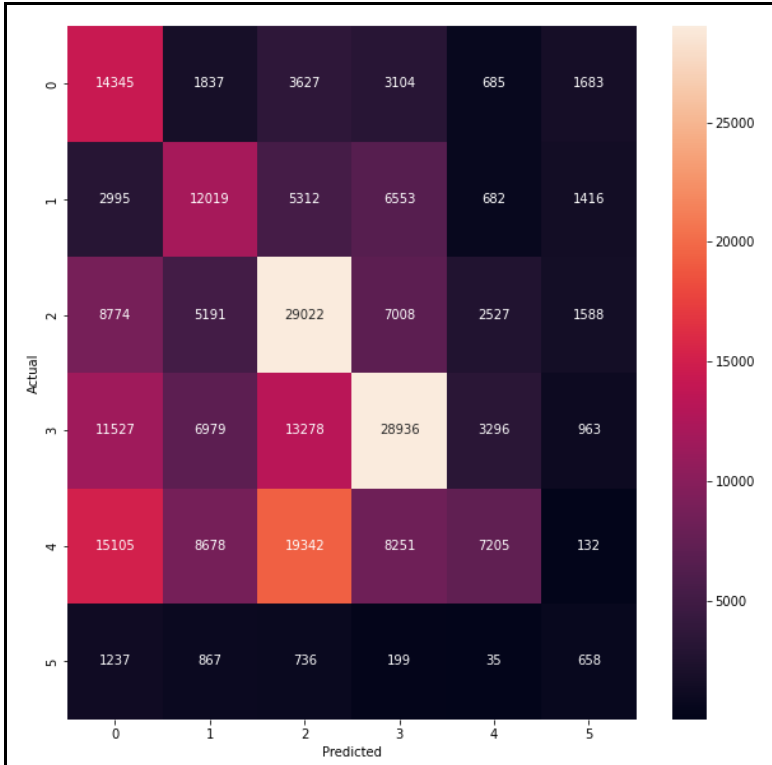


Figure 52: SGD training confusion matrix for all-leg parlays.

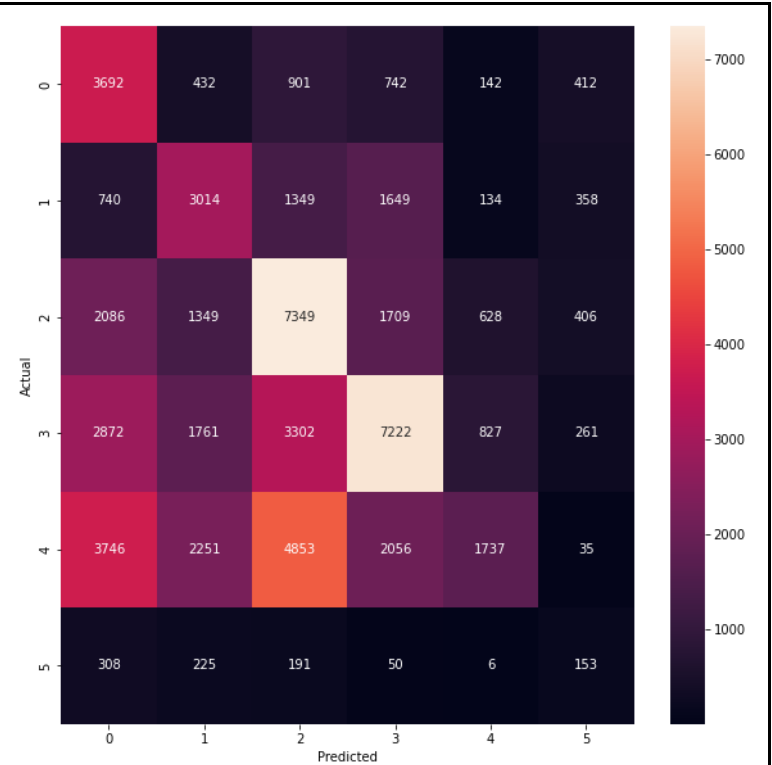


Figure 53: SGD testing confusion matrix for all-leg parlays.

4.2.4 Linear Support Vector Classifier (LSVC) Model

Much like the previous models, LSVC was evaluated for its accuracy against individual bets, two-leg parlays, and all-leg parlays. Its results, which can be seen below, returned consistent results. However, the model did appear to perform better on all-leg parlays than either of the other two datasets.

4.2.4.1 Individual Bets

When the LSVC model was tested on the individual bets dataset, it returned accuracies for both training and testing that were around 0.454. The results can be seen below in Table 17.

Model Evaluation	Accuracy
Training	0.454750
Testing	0.454253

Table 17: LSVC accuracy for individual bets.

The training and testing classification reports for LSVC against all individual bets can be seen below in Figures 54 and 55. The model had much better accuracies with clusters 2, 3, and 4 than it did with clusters 0, 1, and 5. However, LSVC already is showing more promising results when dealing with cluster 5 than SGD.

	precision	recall	f1-score	support
0	0.30	0.40	0.35	144437
1	0.50	0.38	0.43	177804
2	0.41	0.53	0.46	298934
3	0.51	0.48	0.49	346597
4	0.70	0.43	0.53	322271
5	0.15	0.38	0.21	30230
accuracy			0.45	1320273
macro avg	0.43	0.43	0.41	1320273
weighted avg	0.50	0.45	0.46	1320273

Figure 54: LSVC training classification report for individual bets.

	precision	recall	f1-score	support
0	0.30	0.40	0.34	36109
1	0.50	0.38	0.43	44451
2	0.41	0.53	0.46	74734
3	0.51	0.48	0.49	86650
4	0.70	0.43	0.54	80568
5	0.15	0.38	0.21	7557
accuracy			0.45	330069
macro avg	0.43	0.43	0.41	330069
weighted avg	0.50	0.45	0.46	330069

Figure 55: LSVC testing classification report for individual bets.

The representation of the cluster distribution can be seen below in confusion matrices Figures 56 and 57. As recorded in the classification reports for training and testing, clusters 2, 3, and 4 yield the best results of the six clusters. There were a lot of misclassifications happening for the model on individual bets, specifically in clusters 3 and 4 where they are classified as cluster 2.

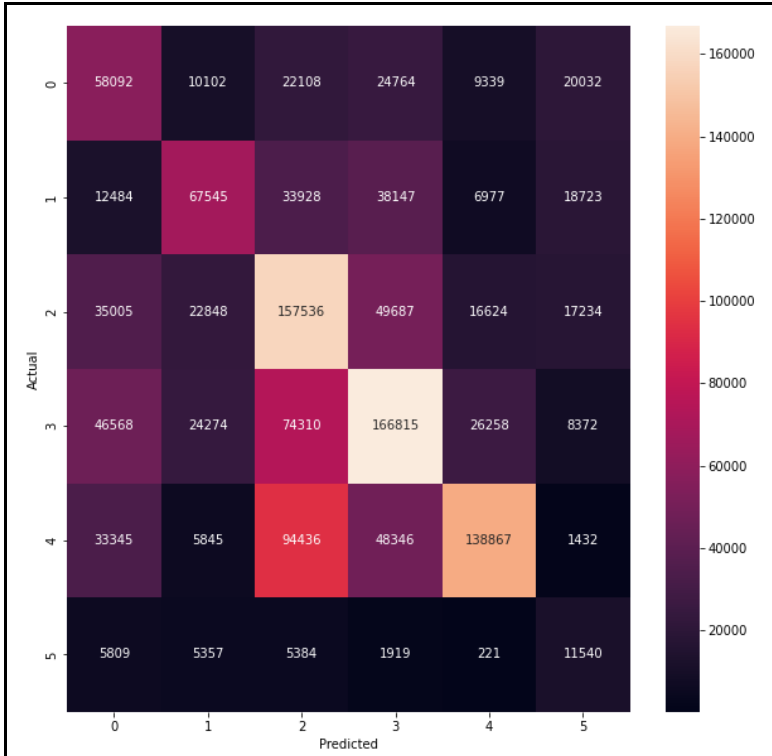


Figure 56: L SVC training confusion matrix for individual bets.

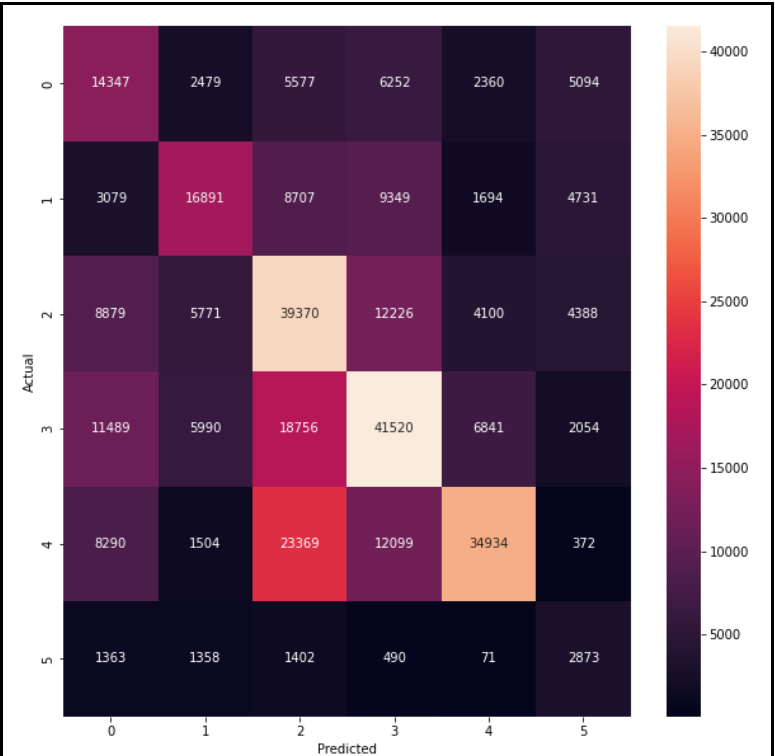


Figure 57: L SVC testing confusion matrix for individual bets.

4.2.4.2 Two-leg Parlays

The L SVC model only performed slightly worse on two-leg parlays than it did on all individual bets. Both training and testing accuracies were calculated as approximately 0.447 each. The results can be seen below in Table 18.

Model Evaluation	Accuracy
Training	0.447443
Testing	0.446029

Table 18: L SVC accuracy for two-leg parlays.

The following training and testing classification reports for the model on two-leg parlays can be seen in Figures 58 and 59. According to the results, the model produced the best results with clusters 2 and 4. The results did, however, indicate worse results when dealing with cluster 3.

	precision	recall	f1-score	support
0	0.37	0.51	0.43	10012
1	0.45	0.36	0.40	8518
2	0.46	0.51	0.48	15985
3	0.47	0.35	0.41	15713
4	0.67	0.50	0.57	13610
5	0.14	0.41	0.20	1786
accuracy			0.45	65624
macro avg	0.43	0.44	0.41	65624
weighted avg	0.48	0.45	0.46	65624

Figure 58: LSVC training classification report for two-leg parlays.

	precision	recall	f1-score	support
0	0.37	0.51	0.43	2503
1	0.45	0.37	0.41	2130
2	0.45	0.50	0.48	3997
3	0.46	0.36	0.41	3928
4	0.67	0.49	0.57	3403
5	0.14	0.41	0.21	446
accuracy			0.45	16407
macro avg	0.43	0.44	0.42	16407
weighted avg	0.48	0.45	0.45	16407

Figure 59: LSVC testing classification report for two-leg parlays.

The confusion matrices in Figures 60 and 61 give further insight as to where the misclassification is occurring. The model often misclassified information as clusters 3 and 4 that were cluster 2. This is worth noting because the same misclassification occurred when the model was dealing with individual bet data. The model also misclassified information as being in clusters 2, 3, and 4 when it was truly cluster 0, which did happen with individual bets as well, but the issue became worse with two-leg parlay data.

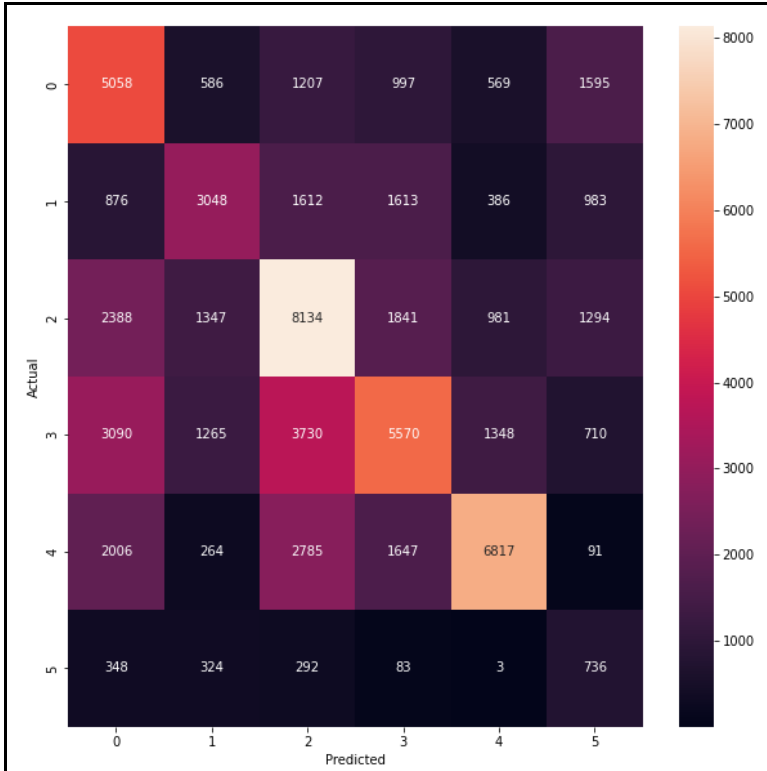


Figure 60: LSCV training confusion matrix for two-leg parlays.

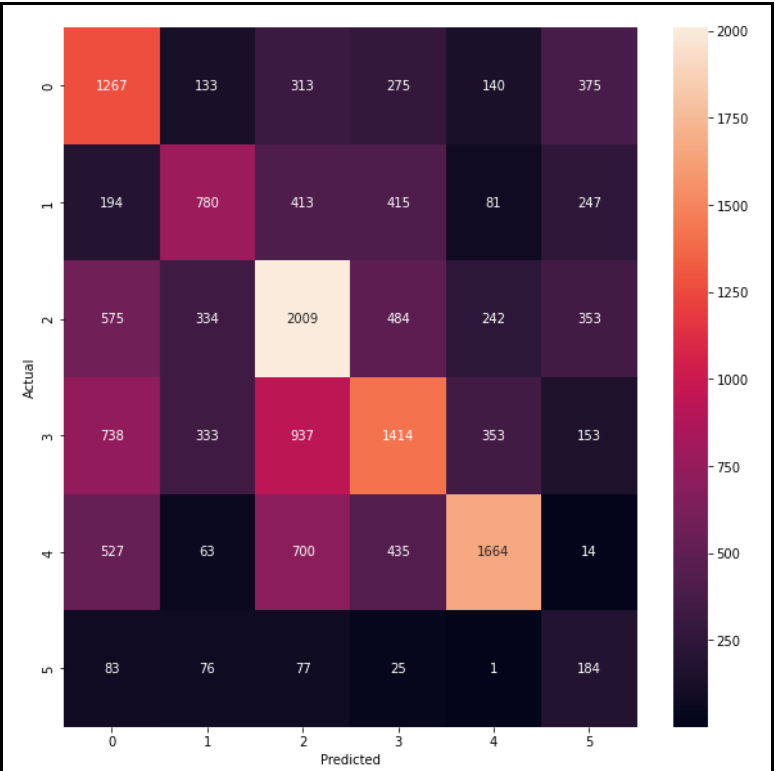


Figure 61: LSCV testing confusion matrix for two-leg parlays.

4.2.4.3 All-Leg Parlays

When the LSCV model’s accuracy was evaluated for all-leg parlay data, it produced the best results. The training and testing accuracies increased with this dataset to roughly 0.473 each. The results can be seen in Table 19.

Model Evaluation	Accuracy
Training	0.471720
Testing	0.475317

Table 19: LSCV accuracy for all-leg parlays.

The training and testing classification report for LSCV on the all-parlay dataset is recorded below in Figures 62 and 63. There was a notable increase in cluster 3, whose f1-score

went from 0.41 to 0.50 from two-leg parlays to all-leg parlays. Cluster 5's f1-score was reduced from 0.21 to 0.16, however.

	precision	recall	f1-score	support
0	0.33	0.44	0.38	25281
1	0.47	0.38	0.42	28977
2	0.43	0.53	0.48	54110
3	0.53	0.48	0.50	64979
4	0.71	0.48	0.57	58713
5	0.11	0.42	0.17	3732
accuracy			0.47	235792
macro avg	0.43	0.45	0.42	235792
weighted avg	0.52	0.47	0.48	235792

Figure 62: LSVC training classification report for all-leg parlays.

	precision	recall	f1-score	support
0	0.34	0.45	0.39	6321
1	0.47	0.38	0.42	7244
2	0.44	0.53	0.48	13527
3	0.53	0.48	0.50	16245
4	0.71	0.48	0.58	14678
5	0.10	0.40	0.16	933
accuracy			0.48	58948
macro avg	0.43	0.45	0.42	58948
weighted avg	0.52	0.48	0.49	58948

Figure 63: LSVC testing classification report for all-leg parlays.

The confusion matrices below for training and testing the LSVC on all-leg parlays can be seen below in Figures 64 and 65. Similar to the results from two-leg parlays, the biggest misclassification that occurred was where the predicted cluster was 2, when the actual clusters were 3 and 4. This is consistent with the results from all individual bets as well. Likewise, there is still misclassification where the model predicted clusters 2, 3, and 4 when the actual cluster was 0.

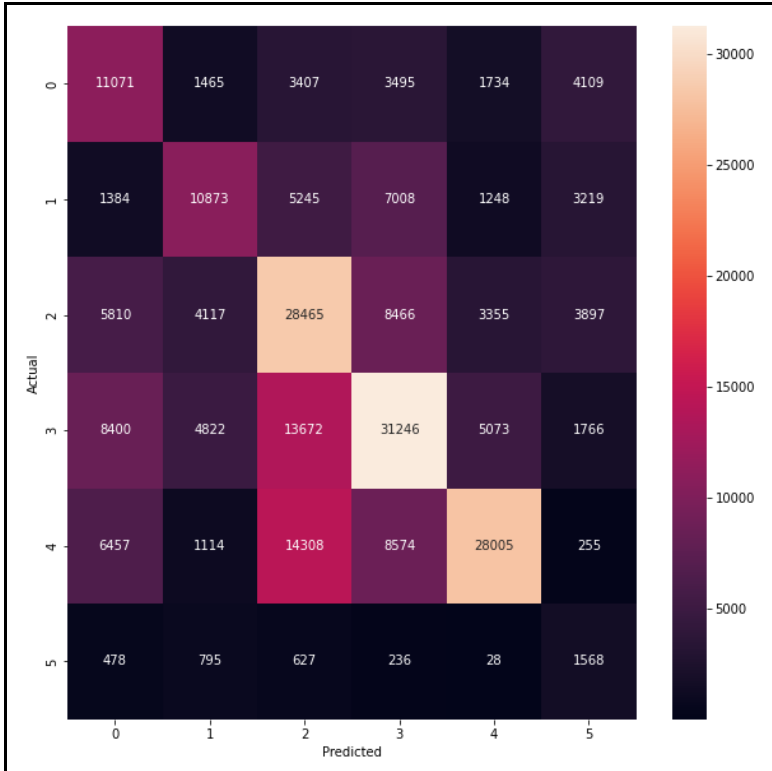


Figure 64: L SVC training confusion matrix for all-leg parlays.

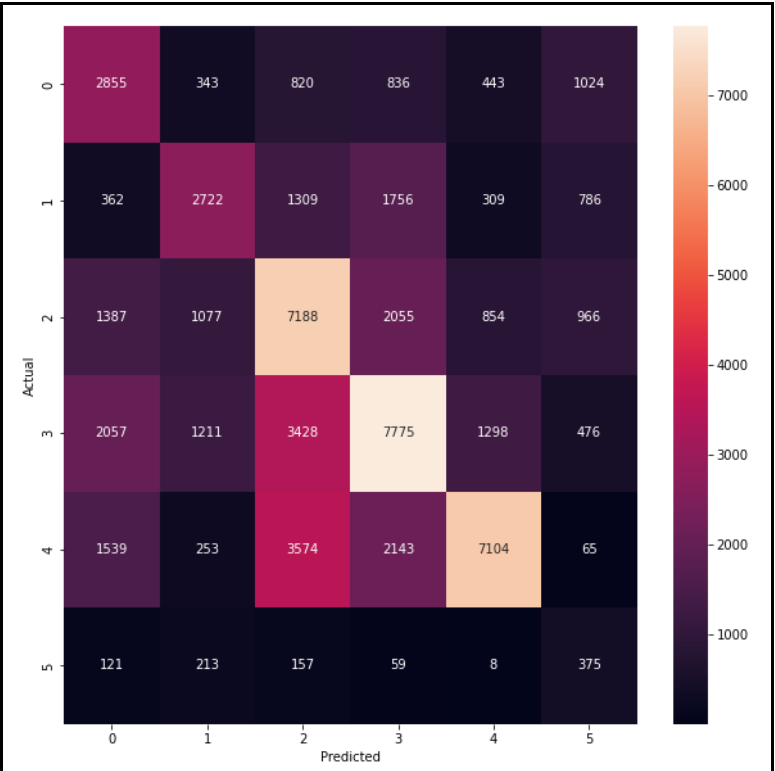


Figure 65: L SVC testing confusion matrix for all-leg parlays.

4.2.5 Deep Neural Network (DNN) Model

The Deep Neural Network uses the same preprocessing setup as the other models. Based on the same X_train data frame that the earlier models can directly take in, the TensorFlow preprocessing takes that data frame and converts it to a sparse tensor object. The first layer of the DNN then takes in that sparse tensor object.

For both the DNN and LSTM NN, the big difference between comparing the results for all individual bets, just two-leg parlays, and all-leg parlays is the size of the vocabulary that results from the sentence creation from all the bets. The team trained the DNN models over 40 epochs, or passes over the data, using a batch size of 100.

4.2.5.1 Individual Bets

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 64)	331264
dense_1 (Dense)	(None, 32)	2080
dense_2 (Dense)	(None, 16)	528
dense_3 (Dense)	(None, 6)	102
=====		
Total params: 333,974		
Trainable params: 333,974		
Non-trainable params: 0		

Figure 66: Deep neural network model summary for all bets.

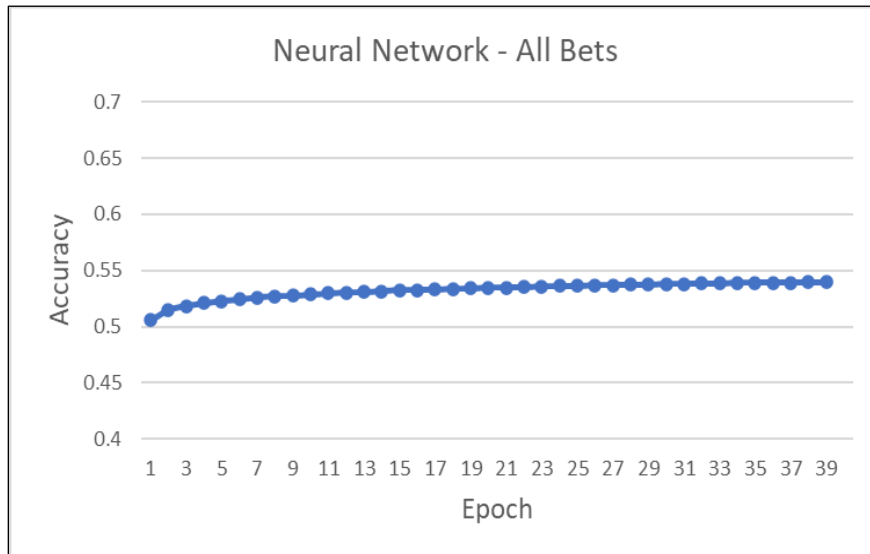


Figure 67: Deep neural network training accuracy for all individual bets.

The vocabulary size for the individual bets category was the largest of all the vocabularies. This is likely due to the amount of one-off proposition bets that are present throughout the data set. These random prop bets are not included in parlays, thus making parlay bets easier to predict than individual bets. The final accuracy after all epochs were completed was 0.5395.

4.2.5.2 Two-Leg Parlays

Layer (type)	Output Shape	Param #
dense_8 (Dense)	(None, 64)	177216
dense_9 (Dense)	(None, 32)	2080
dense_10 (Dense)	(None, 16)	528
dense_11 (Dense)	(None, 6)	102

Total params: 179,926
Trainable params: 179,926
Non-trainable params: 0

Figure 68: Deep neural network model summary for two-leg parlays.

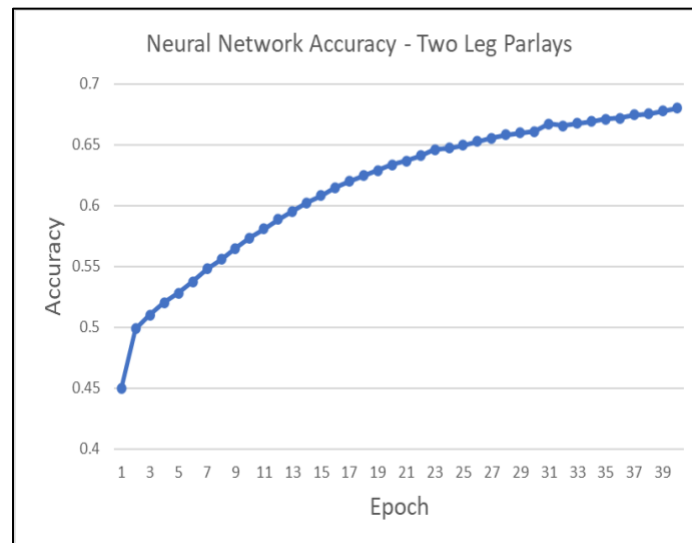


Figure 69: Deep neural network training accuracy for two-leg parlays.

When only looking at two-leg parlays, the vocabulary size dealt with in the first layer is considerably different. As a result of the smaller vocabulary size, the accuracy for the DNN for two-leg parlays in the final epoch was 0.6798, significantly higher than the individual bets results.

4.2.5.3 All-Leg Parlays

Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 64)	240512
dense_5 (Dense)	(None, 32)	2080
dense_6 (Dense)	(None, 16)	528
dense_7 (Dense)	(None, 6)	102

Total params: 243,222
Trainable params: 243,222
Non-trainable params: 0

Figure 70: Deep neural network model summary for all-leg parlays.

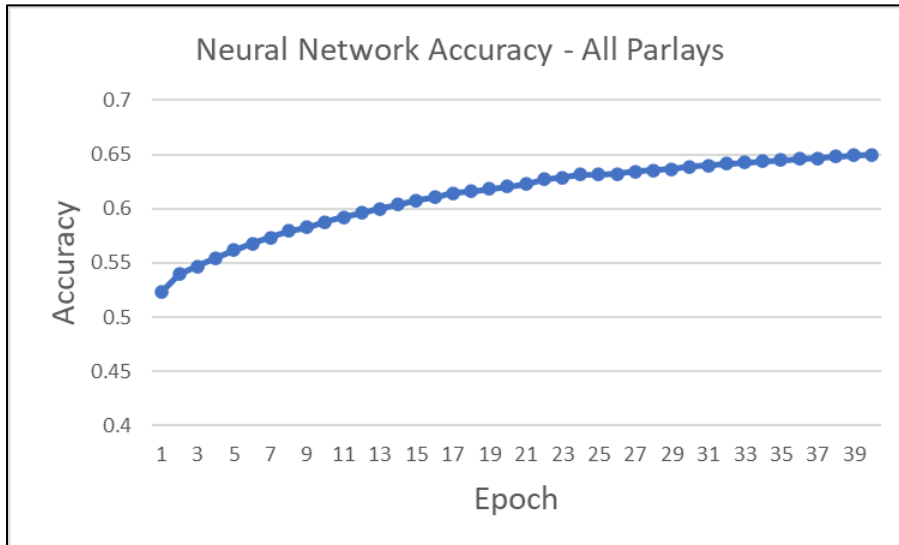


Figure 71: Deep neural network training accuracy for all-leg parlays.

The DNN classifier for the all-parlay dataset still performed very well, with an accuracy of approximately 0.65 in the final epoch. We can see in the first layer that the vocabulary size is greater than just two-leg parlays, but still less than individual bets. This is likely a result of including so many one-off proposition bets throughout the database that were not included in any parlays.

4.2.6 Long Short-Term Memory Neural Network (LSTM NN) Model

The Long Short-Term Memory models had a slightly different preprocessing path compared to the normal models. The LSTM model takes in the raw sentence data and does its own preprocessing with a text_vectorization layer and embedding layer. The embedding layer serves the same purpose as the first layer of the DNN did. Due to how long training time took for the LSTM models, they were trained on far fewer epochs than the DNN was. Additionally, the exponentially longer training time associated with LSTM networks compared to the DNN forced the group to limit the number of epochs so the network does not converge as clearly as the DNN was able to.

4.2.6.1 Individual Bets

Layer (type)	Output Shape	Param #
text_vectorization_1 (TextVectorization)	(None, None)	0
embedding (Embedding)	(None, None, 64)	436544
bidirectional (Bidirectional)	(None, 8)	2208
dense_3 (Dense)	(None, 16)	144
dense_4 (Dense)	(None, 6)	102
=====		
Total params: 438,998		
Trainable params: 438,998		
Non-trainable params: 0		

Figure 72: LSTM model summary for all bets.

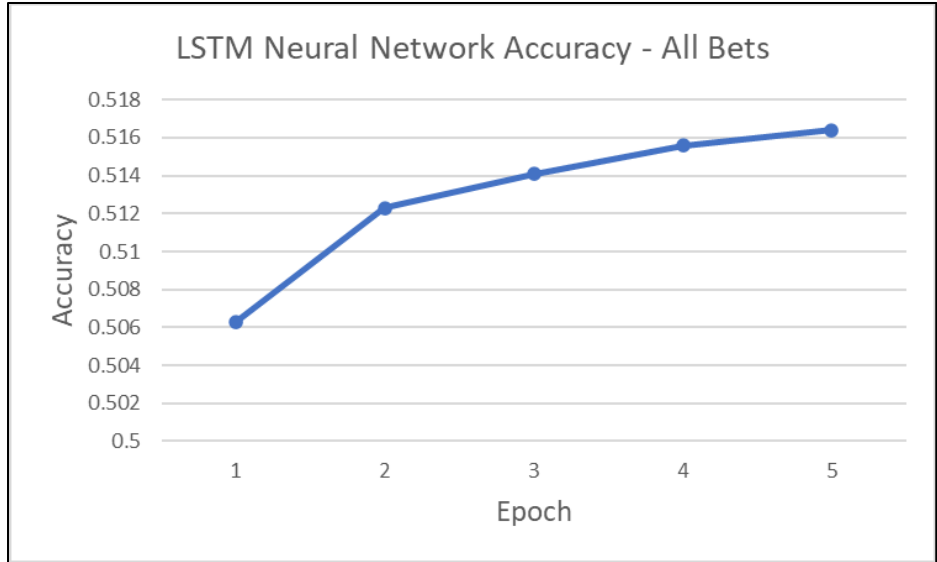


Figure 73: LSTM neural network training accuracy for all bets.

The final accuracy of the LSTM model for all-bets was 0.5164.

4.2.6.2 Two-Leg Parlays

Layer (type)	Output Shape	Param #
text_vectorization (TextVectorization)	(None, None)	0
embedding_3 (Embedding)	(None, None, 64)	185536
bidirectional_4 (Bidirectional)	(None, 128)	66048
dense_18 (Dense)	(None, 16)	2064
dense_19 (Dense)	(None, 6)	102
=====		
Total params: 253,750		
Trainable params: 253,750		
Non-trainable params: 0		

Figure 74: LSTM model summary for two-leg parlays.

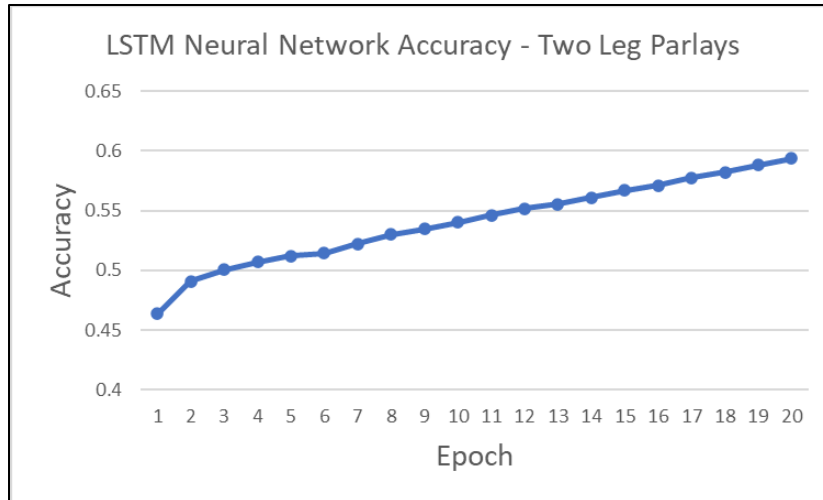


Figure 75: LSTM neural network training accuracy for two-leg parlays.

The final accuracy for the LSTM model for two-leg parlays was 0.5936. This training is a prime example of the time constraints associated with the limited processing power of the free version of Google Collab compared to the power of WPI's Turing Machines. Given more time in the project, the team could've attempted a longer training period of the LSTM network.

4.2.6.3 All-Leg Parlays

Layer (type)	Output Shape	Param #
text_vectorization (TextVec torization)	(None, None)	0
embedding_3 (Embedding)	(None, None, 64)	258048
bidirectional_3 (Bidirectio nal)	(None, 128)	66048
dense_5 (Dense)	(None, 16)	2064
dense_6 (Dense)	(None, 6)	102
=====		
Total params: 326,262		
Trainable params: 326,262		
Non-trainable params: 0		

Figure 76: LSTM model summary for all-leg parlays.

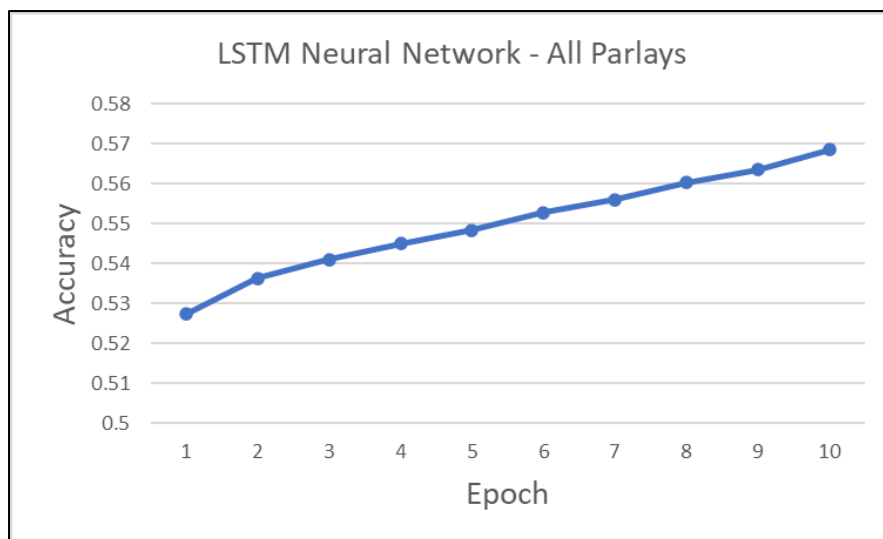


Figure 77: LSTM neural network training accuracy for all-leg parlays.

The final accuracy for the LSTM model for all-leg parlays was 0.5684.

4.3 Summary of Results

In summary, cluster analysis was performed in order to correctly classify similar users together. The optimal number of clusters was determined as six via the elbow curve method and the users were classified into clusters using the spectral clustering method. Once this new cluster feature was joined back into the dataset, the group was able to transform the data with TF-IDF to quantify the importance of certain features/trends. Then the group tested five machine learning algorithms on the three different sets of data (all individual bets, two-leg parlays, and all-leg parlays) and used multiple metrics to determine which had the best performance. Table 20 below summarizes the five classification algorithms, as well as the random cluster association and most popular cluster association baselines, and their testing accuracies for all three datasets.

Model	All Individual Bets – testing accuracy	Two-Leg Parlays – testing accuracy	All-Leg Parlays – testing accuracy
Multinomial Naive Bayes	.475	.437	.489
Stochastic Gradient Descent	.435	.383	.393
Linear SVC	.454	.446	.475
Deep Neural Network	.540	.680	.650
LSTM NN	.516	.594	.568
Random cluster baseline	.168	.168	.168
Most popular cluster baseline	.263	.244	.276

Table 20: Machine learning algorithms utilized and their performance.

The deep neural network had the best performance across the board for all three datasets with the highest testing accuracy. This means that individual bets can be associated with the correct cluster with 2.3 times better accuracy than the probability of selecting the most popular cluster and over 3 times better accuracy than random assignment. The performance for parlays is even better. Two-leg parlays are associated to users 2.8 and 4 times better than the probability of selecting the most popular cluster and a random assignment, respectively. Likewise, the accuracy performance of all-leg parlays performed 2.4 and 3.9 times better than the probability of selecting the most popular cluster and random cluster association, respectively.

5.0 Recommendations and Conclusions

In this chapter, we discuss a potential path toward implementation of the bet/parlay prediction system developed in this project into the DraftKings app. We then discuss our recommendations for future work and our conclusions.

5.1 Implementation

In this section, we will outline how the clustering and classification system developed in this project can be utilized to enhance the user experience of the DraftKings app. Currently, when users load into the app, popular parlays for the day are displayed at the top of the screen. These parlays are not tailored to individual user preferences, however. Using the clustering and

classification system built in the project, we can leverage the clusters of users to make bets and parlays more personalized and easily accessible.

As one way to achieve this, Figure 78 shows a flow diagram of a potential implementation model. The first step in the process is to identify popular bets and parlays, a step that DraftKings sportsbook already does. The next step is for DraftKings to convert the bet information from popular parlays to text. The resulting “bet sentence” is then used as input to the classifier trained on cluster-labeled data. The classifier then determines the cluster of users that is most likely to bet on the popular parlay. Once the cluster is identified, the popular bet/parlay can be recommended to the users in that specific cluster in a new tab called "parlays for me." This tab will store all the popular parlays that are relevant to the user's cluster, making it easy for them to find and bet on parlays that are likely to align with their preferences.

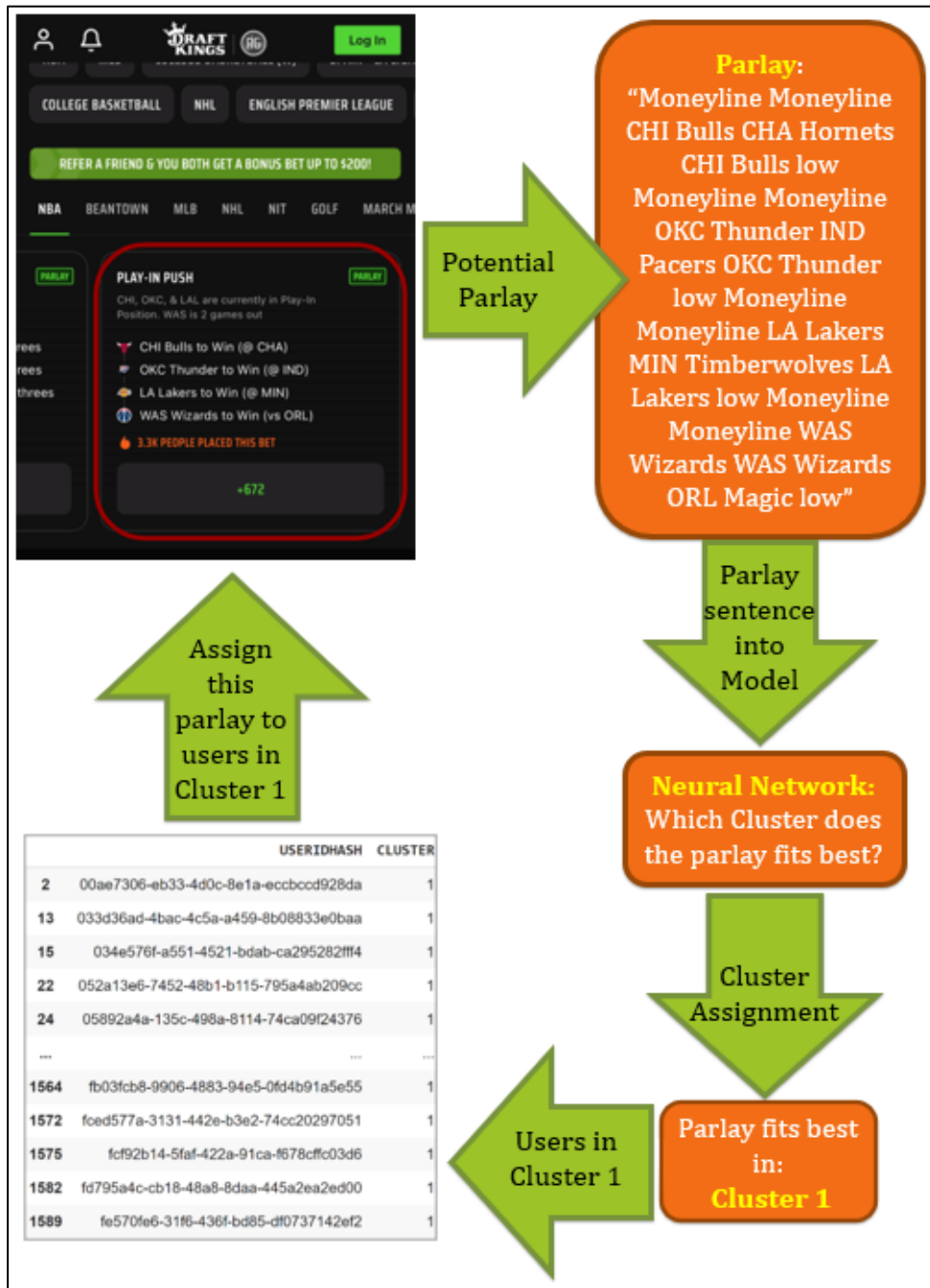


Figure 78: Flow diagram of implementation.

By implementing this feature, we can significantly enhance the user experience of the DraftKings app by providing personalized and relevant parlays to users based on their cluster membership. This can result in users making more parlay bets since the “parlays for me” will be aligned with user preferences. An additional benefit is that the user will be presented less bets/parlays that they are not be interested in. This limits the time and effort the user has when

searching for parlays that they might like and will lead to better user interaction with the DraftKings interface.

Overall, the clustering and classification system developed in this project has the potential to be a valuable tool for enhancing the user experience of the DraftKings app by offering personalized and tailored bets and parlays to users based on their betting behavior and preferences, ultimately providing a more engaging and satisfying betting experience for users.

5.2 Recommendations

Over the course of this project, the team explored many different ideas that were not discussed in this report since they did not contribute to the final implementation of the bet/parlay prediction system. The decision to perform user clustering and multi-class classification occurred relatively late in the project. As a result, the team recognizes that there are opportunities to further develop the clustering portion of the prediction system. Our approach to the user cluster problem involved narrowing down our data to the modes of user bet information. Since we are only focusing on the mode, i.e., the most popular category for each user, better results could be achieved by considering all the users' bets rather than the modes of the categorical columns. With more time, one of the methods that the group proposed would be to create a document for each user containing all their bets and then vectorize to get the word vectors for each user like what was conducted in the multi-classification. This would allow capturing all of the users' textual information, rather than just the modes, and the values can possibly result in more distinctive clustering of the users.

In addition to capturing more than just the modes of user textual information to cluster, with more time the group would have focused more on picking the best clustering algorithm jointly with the optimal number of clusters for each clustering algorithm. Since we wanted to test the machine learning and deep learning results, the group needed a static clustering of users. To do this with limited time, the group used an elbow curve method for K-Means, from which we determined the optimal number of clusters was $K=6$. This number of clusters was used for all of the other clustering algorithms (even though it was only determined to be optimal for K-means) and then the clustering algorithm that performed the best based on Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index was selected. The group recognizes that $K=6$ might not be the optimal number of clusters for each of the clustering algorithms. Due to the lack of

time at the end of the project, this is the method the group used to create static clustering of users that was then tested on the multi-class classification models. The group would have liked to ensure that the optimal number of clusters for each specific clustering algorithm was chosen, and then the optimal metrics to be compared between the algorithms to choose the best possible clustering of users. The way the group would have attempted this is by recording the metrics for different numbers of clusters for each given algorithm and choosing the number of clusters that would perform the best, storing that information to later compare to the other algorithm's optimal results. Bettering the clustering of users will help to better the model's capability to classify which users would like a bet that it is given.

Additionally given more time, the team would have been able to look more into using the WPI Turing machines to gain access to additional processing power. Having great processing power would have allowed the team to run algorithms, particularly the LSTM Neural Networks, for a longer training period to allow additional convergence of the optimizer.

5.3 Conclusions

With sports betting becoming increasingly popular, the demand for personalized betting experience has also grown. This project developed a system that would achieve personalization by taking the popular parlays that are shown to everyone via the DraftKings app and finding a way to help these popular parlays reach their target audience users. The group created a multi-classification model where the users are clustered based on behavioral data using spectral clustering and then using TF-IDF to classify the bets to their respective clusters. For this project, among the various classifiers considered, the deep neural network was most accurate in classifying individual bets, two-leg parlays, and all-leg parlays. The implementation of this model onto the DraftKings platform would improve the user experience by creating more personalized popular parlays. This project provides a promising direction for DraftKings to tailor recommended bets and parlays to the users who are most likely to bet on them.

Appendix A – Summary of Platforms Offered by DraftKings

In DraftKings Daily Fantasy Sports, users receive a “Salary Cap” which allows them to acquire players in their roster. Each player has an assigned cost to them that subtracts from the user’s salary when they choose to select the player. Players gain points based on their performance that count towards the overall score of a user’s team. This provides a refreshing way for users to play fantasy sports all season long without being trapped in the same roster for the duration of the season.

In the DraftKings Casino, users can play many different wager-based games, ranging from traditional casino games like Blackjack, Poker, and Roulette. Alternatively, there are many online DraftKings specific games for users to wager on.

In the DraftKings Marketplace, users have the ability to purchase and sell many different collectibles across sports, entertainments, and culture that will be owned by their account.

In DraftKings ReignMakers, users can build a collection of digital player cards licensed by many major sports organizations and continuously draft those cards throughout the duration of the season. Like in Daily Fantasy Sports, the cards accrue a score depending on the performance of the corresponding player in their competition. Users compete their lineup of these player cards against each other for various monetary prizes.

Appendix B – Classification Report of Random Baseline

As discussed, the random baseline model will contain the same accuracy scores across all three data types of individual bets, two-leg parlays, and all-leg parlays, but will only differ on which cluster is the most popular for each of the datasets. In Figure 1B and 2B below it is evident that cluster 2 and cluster 3 are the most popular clusters for two-leg and all-leg parlays, respectively.

	precision	recall	f1-score	support
0	0.14	0.15	0.15	2503
1	0.14	0.17	0.15	2130
2	0.25	0.17	0.20	3997
3	0.24	0.17	0.20	3928
4	0.21	0.17	0.19	3403
5	0.03	0.17	0.05	446
accuracy			0.17	16407
macro avg	0.17	0.17	0.16	16407
weighted avg	0.20	0.17	0.18	16407

Figure 1D: Classification report for random model baseline model for two-leg parlays.

	precision	recall	f1-score	support
0	0.11	0.16	0.13	6321
1	0.13	0.17	0.15	7244
2	0.23	0.16	0.19	13527
3	0.28	0.17	0.21	16245
4	0.25	0.17	0.20	14678
5	0.02	0.20	0.04	933
accuracy			0.17	58948
macro avg	0.17	0.17	0.15	58948
weighted avg	0.22	0.17	0.18	58948

Figure 2D: Classification report for random baseline model for all-leg parlays.

Works Cited

- Abdu, E., & Salane, D. (2009). A spectral-based clustering algorithm for categorical data using data summaries. *Proceedings of the 2nd Workshop on Data Mining Using Matrices and Tensors*. <https://doi.org/10.1145/1581114.1581116>
- Alpaydin, E. (2020). Introduction to Machine Learning, fourth edition. In *Google Books*. MIT Press. https://books.google.com/books?hl=en&lr=&id=tZnSDwAAQBAJ&oi=fnd&pg=PR7&dq=Introduction+to+machine+learning.+MIT+press.&ots=F3YVd1bvvd&sig=fQr3fMejI3uVEotoI0oIy_vXoLw#v=onepage&q=Introduction%20to%20machine%20learning.%20MIT%20press.&f=false
- Arguello, J. (2013, March 25). *Evaluation Metrics*. UNC School of Information and Library Science; University of North Carolina. https://ils.unc.edu/courses/2013_spring/inls509_001/lectures/10-EvaluationMetrics.pdf
- Bach, F., & Jordan, M. (2004). *Advances in neural information processing systems 16 : proceedings of the 2003 conference ; [Seventeenth Annual Conference on Neural Information Processing Systems (NIPS), held in British Columbia, Canada, from December 8 through 13, 2003]* (S. Thrun, L. K. Saul, & B. Schölkopf, Eds.; Learning Spectral Clustering). MIT Press.
- Berry, Michael W, et al. *Supervised and Unsupervised Learning for Data Science*. Cham, Switzerland, Springer, 2020, link.springer.com/book/10.1007/978-3-030-22475-2. Accessed 2 Feb. 2023.
- Brown, Sara. "Machine Learning, Explained." *MIT Sloan*, 21 Apr. 2021, <https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained>.
- Camargo, M., Dumas, M., González-Rojas, O. (2019). Learning Accurate LSTM Models of Business Processes. In: Hildebrandt, T., van Dongen, B., Röglinger, M., Mendling, J. (eds) *Business Process Management. BPM 2019. Lecture Notes in Computer Science()*, vol 11675. Springer, Cham. https://doi.org/10.1007/978-3-030-26619-6_19
- Davidow, M., & Miller, E. (2019). *The Logic Of Sports Betting*. Ed Miller.
- Davis, J., & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. *Proceedings of the 23rd International Conference on Machine Learning - ICML '06*, 233–240. <https://doi.org/10.1145/1143844.1143874>

- DraftKings Net Worth 2019-2021 | DKNG | MacroTrends*. (n.d.). Retrieved November 15, 2022, from <https://www.macrotrends.net/stocks/charts/DKNG/draftkings/net-worth>
- DRAFTKINGS REPORTS FOURTH QUARTER REVENUE OF \$855 MILLION*. (2023). DraftKings Inc. <https://draftkings.gcs-web.com/news-releases/news-release-details/draftkings-reports-fourth-quarter-revenue-855-million-raises>
- Dutta, M., Mahanta, A. K., & Pujari, A. K. (2005). QROCK: A quick version of the ROCK algorithm for clustering of categorical data. *Pattern Recognition Letters*, 26(15), 2364–2373. <https://doi.org/10.1016/j.patrec.2005.04.008>
- Farid, D. Md., Rahman, M. M., & Al-Mamuny, M. A. (2014). Efficient and scalable multi-class classification using naïve Bayes tree. *2014 International Conference on Informatics, Electronics & Vision (ICIEV)*, 1–4. <https://doi.org/10.1109/ICIEV.2014.6850698>
- Ganti, V., Gehrke, J., & Ramakrishnan, R. (1999). CACTUS—clustering categorical data using summaries. *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 73–83. <https://doi.org/10.1145/312129.312201>
- Hartwig, F., & Dearing, B. E. (1979). *Exploratory data analysis* (No. 16). Sage.
- Heitner, D. (2017). The first five years of DraftKings. *Forbes*. Retrieved November 15, 2023, from <https://www.forbes.com/sites/darrenheitner/2017/04/25/the-first-five-years-of-draftkings/?sh=177d44e45254>
- Hiran, Kamal Kant, et al. *Machine Learning: Master Supervised and Unsupervised Learning Algorithms with Real Examples (English Edition)*. Google Books, BPB Publications, 16 Sept. 2021, books.google.com/books?id=4VVDEAAAQBAJ&lpg=PT25&ots=OLfeBCI0iK&dq=%20unsupervised%20learning%20vs%20supervised%20machine%20learning&lr&pg=PA1#v=onepage&q&f=false. Accessed 3 Feb. 2023.
- How to Read Odds - How to Bet 101 | DraftKings Sportsbook*. (n.d.). Sportsbook.draftkings.com. Retrieved December 10, 2022, from <https://sportsbook.draftkings.com/help/how-to-bet-reading-odds?wpsrc=Organic%20Search&wpaffn=Google&wpkw=https%3A%2F%2Fsportsbook.draftkings.com%2Fhelp%2Fhow-to-bet-reading-odds&wpcn=help&wpsc=how-to-bet-reading-odds>
- Hughes, A. (2023, January 23). California Sports Betting: When Will It Be Legalized? *Gaming Today*. <https://www.gamingtoday.com/california/>

- IBM. (n.d.). *What is natural language processing?* IBM - United States.
<https://www.ibm.com/topics/natural-language-processing>
- Jain, A. K., & Dubes, R. C. (1988). *Algorithms for Clustering Data*. Prentice Hall.
https://homepages.inf.ed.ac.uk/rbf/BOOKS/JAIN/Clustering_Jain_Dubes.pdf
- Jajuga, K., Batóg, J., & Walesiak, M. (2020). Classification and Data Analysis: Theory and Applications. In *Google Books*. Springer Nature.
<https://books.google.com/books?id=qf75DwAAQBAJ&lpg=PA19&ots=OJcbhhKNs1&dq=cluster%20evaluation%20silhouette&lr&pg=PA19#v=onepage&q=cluster%20evaluation%20silhouette&f=false>
- Jurafsky, D. & Martin, J. H. (2023). Naïve Bayes and Sentiment Classification. *Speech and Language Processing*. <https://web.stanford.edu/~jurafsky/slp3/4.pdf>.
- Kanstrén, T. (2021, May 19). A look at precision, recall, and F1-score. Medium.
<https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec#:~:text=F1%2DScore%20is%20a%20measure,than%20the%20traditional%20arithmetic%20mean>
- Kärkkäinen, I., & Fränti, P. (2000, July). Minimization of the value of Davies-Bouldin index. In *Proceedings of the IASTED International Conference on Signal Processing and Communications (SPC'2000)*. IASTED/ACTA Press (pp. 426-432).
- Kim, S.-W., & Gil, J.-M. (2019). Research paper classification systems based on TF-IDF and LDA schemes. *Human-Centric Computing and Information Sciences*, 9(1), 30.
<https://doi.org/10.1186/s13673-019-0192-7>
- Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9), 1464–1480.
<https://doi.org/10.1109/5.58325>
- Legal Sports Betting. (2022, December 16). *History of sports betting in the USA*. LegalSportsBetting.com. Retrieved January 10, 2023, from <https://www.legalsportsbetting.com/history-of-sports-betting-in-the-usa/>
- Linear SVC. (n.d.). Retrieved April 24, 2023, from [//nightlies.apache.org/flink/flink-ml-docs-release-2.1/docs/operators/classification/linearsvc/](https://nightlies.apache.org/flink/flink-ml-docs-release-2.1/docs/operators/classification/linearsvc/)
- Liptak, A., & Draper, K. (2018, May 14). Supreme Court Ruling Favors Sports Betting. The New York Times. <https://www.nytimes.com/2018/05/14/us/politics/supreme-court-sports-betting-new-jersey.html>

- Mitchell, T. (1997). *Machine Learning*. McGraw-Hill Science/Engineering/Math.
- Nickolas, S. (2021, May 31). *What Do Correlation Coefficients Positive, Negative, and Zero Mean?* Investopedia. <https://www.investopedia.com/ask/answers/032515/what-does-it-mean-if-correlation-coefficient-positive-negative-or-zero.asp>
- Nielsen, F. (2016). Hierarchical Clustering. *Introduction to HPC with MPI for Data Science*, 195–211. https://doi.org/10.1007/978-3-319-21903-5_8
- Pang, K. (2003). *Self-organizing Maps*. <https://www.cs.hmc.edu/~kpang/nn/som.html>
- Parlay - How to Bet 101 | Draftings Sportsbook*. (n.d.). Sportsbook.draftkings.com. Retrieved January 10, 2023, from <https://sportsbook.draftkings.com/help/how-to-bet-parlay?wpsrc=Organic%20Search&wpaffn=Google&wpkw=https%3A%2F%2Fsportsbook.draftkings.com%2Fhelp%2Fhow-to-bet-parlay&wpcn=help&wpscn=how-to-bet-parlay>
- Penn State. (n.d.). *14.8—K-Means Procedure | STAT 505*. PennState: Statistics Online Courses. Retrieved April 18, 2023, from <https://online.stat.psu.edu/stat505/lesson/14/14.8>
- Piech, C. (n.d.). *K Means*. Retrieved April 18, 2023, from <https://stanford.edu/~cpiech/cs221/handouts/kmeans.html>
- Porter, A. (2022, March 15). What are the advantages of natural language processing in AI? Capacity. <https://capacity.com/enterprise-ai/faqs/what-are-the-advantages-of-natural-language-processing-nlp/>
- Santaromita, D. (n.d.). *Understanding sports betting odds and how to read them*. The Athletic. Retrieved April 24, 2023, from <https://theathletic.com/2497657/2022/01/25/understanding-sports-betting-odds-and-how-to-read-them/>
- Shlens, J. (2014). *A Tutorial on Principal Component Analysis*. <https://arxiv.org/pdf/1404.1100.pdf>
- Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4), 427–437. <https://doi.org/10.1016/j.ipm.2009.03.002>
- Soni Madhulatha, T. (2012). AN OVERVIEW ON CLUSTERING METHODS. *IOSR Journal of Engineering*, 2(4), 719–725. <https://arxiv.org/pdf/1205.1117.pdf>

- Sportsbook / About DraftKings*. (n.d.). Mt.draftkings.com. Retrieved November 26, 2022, from <https://mt.draftkings.com/about/sportsbook/?wpsrc=Organic%20Search&wpaffn=Google&wpkw=https%3A%2F%2Fmt.draftkings.com%2Fabout%2Fsportsbook%2F&wpcn=about&wpscn=sportsbook%2F>
- Syakur, M. A., Khotimah, B. K., Rochman, E. M. S., & Satoto, B. D. (2018). Integration K-Means Clustering Method and Elbow Method For Identification of The Best Customer Profile Cluster. *IOP Conference Series: Materials Science and Engineering*, 336(1), 012017. <https://doi.org/10.1088/1757-899X/336/1/012017>
- Tamm, Y.-M., Damdinov, R., & Vasilev, A. (2021). Quality Metrics in Recommender Systems: Do We Calculate Metrics Consistently? *Fifteenth ACM Conference on Recommender Systems*, 708–713. <https://doi.org/10.1145/3460231.3478848>
- Umargono, E., Suseno, J., & Gunawan, S. K. (2020, January 1). K-Means Clustering Optimization Using the Elbow Method and Early Centroid Determination Based on Mean and Median Formula. <https://doi.org/10.2991/assehr.k.201010.019>
- Wang, X., & Xu, Y. (2019). An improved index for clustering validation based on Silhouette index and Calinski-Harabasz index. *IOP Conference Series: Materials Science and Engineering*, 569(5), 052024. <https://doi.org/10.1088/1757-899x/569/5/052024>
- “What Is the Jaccard Similarity Measure in NLP?” *Educative*, <https://www.educative.io/answers/what-is-the-jaccard-similarity-measure-in-nlp>.
- Wilbur, W. J., & Kim, W. (2014). Stochastic Gradient Descent and the Prediction of MeSH for PubMed Records. *AMIA Annual Symposium Proceedings, 2014*, 1198–1207.