nsethi9

Tail Me - Team 46 ML Final Report

Introduction / Background

Unlike professional sports, college football is a collection of nearly 150 teams - many of whom don't receive national attention. For many of these programs, local media outlets dominate their news coverage. In the context of sports betting, lines and odds are set by outsiders on a national scale. Thus, much of the underlying information about a team may be captured by local media and kept a "secret" to these oddsmakers. Additionally, key team statistics—such as returning players, offensive and defensive metrics, and overall roster continuity—often go underexplored at the national level. We aim to explore how capturing local news sentiments, combined with these underexplored team statistics, can provide context that national coverage misses and lead to more accurate predictions of a college football team's season. While others have explored social media and individual-level data to predict outcomes, the concept of analyzing local news coverage is still relatively untapped in this space (IEEE Explore and MDPI). For the example attaached, one study performed sentiment analysis of Tweets which was a similar process to what we looked at. They used an SVM and decision trees in order to classify the Tweets and found high accuracy.

For our first model, we took a fresh approach to predict college football team performance by combining local sentiment with national betting odds. For every FBS team, we scraped local news articles to get a sense of how writers felt about their team's chances before the 2023 season. Using advanced tools, we analyzed the tone of these articles—whether positive, negative, or neutral—and turned that sentiment into a numerical score. We then used this score to create a simple prediction: whether bettors should take the over or under on a team's projected wins for the season. To check how accurate our model was, we compared it against national win projections from Sports Odds History. By mixing local coverage with national odds, we developed a new way to approach sports betting predictions.

For our next model, we decided to try something a little different by using a clustering technique to find groups of teams that had the best chances of winning based on similar stats. We started by collecting all the team statistics from the 2022 season using a detailed sports database College Football Team Data, along with data on how many starters each team was bringing back for the 2023 season. We used total yards, first downs, penalties, turnovers, total yards allowed, and first downs allowed. We knew that teams with more returning starters often perform better, so we turned this into a percentile score to highlight the strength of each team's roster. Then, we combined all of this information and used a method called KMeans clustering, which groups teams with similar profiles together. This allowed us to categorize teams into clusters based on their overall strengths and weaknesses. By analyzing these clusters, we were able to spot patterns that helped predict which teams were more likely to outperform or underperform their expected win totals for the season. This approach gave us a fresh perspective on how different team types might fare, offering sports bettors valuable insights into team success.

Finally, to take our predictions to the next level, we combined both models into one powerful approach. By using a Random Forest classifier, we connected the insights from sentiment analysis with the team clusters, allowing us to predict whether a team would exceed or fall short of their expected win total. Random Forest works by creating many "decision trees" that each make their own prediction, and then combining the results to give a final, more accurate prediction. Think of it like having a group of experts weigh in on the same question, with the final decision being the one most experts agree on. We plotted key data points together to predict a team's win differential for the season. This merger of models gave us a much clearer picture of team performance, turning our predictions into actionable insights. In the end, these three models provided a strong foundation for accurately predicting which teams will outperform expectations, helping bettors make smarter, data-backed decisions for the season ahead.

Problem Definition

Problem

Predicting college football team performance can be challenging, as national odds often overlook both the local sentiment surrounding a team and key team statistics. We aim to explore whether combining insights from local news articles—capturing the emotions and opinions of writers close to the team—with overlooked team metrics like returning players and offensive/defensive statistics that can provide valuable predictions for the upcoming season.

Motivation

Local news and underexplored team statistics often contain critical information that is underutilized by national oddsmakers, especially for smaller programs. By analyzing both the narratives in local reporting and metrics like roster continuity and past performance, we aim to uncover insights that can give sports bettors a significant edge in predicting team success.

Methods

Model 1: Sentiment Analysis and Linear Regression (Supervised)

Data Preprocessing Methods

- We start by gathering the dataset, done by webscraping local news articles in a specified format for all division one teams, aiming for around 10 articles per team. We extract the title and article content of each of these links and export it to a csv.
- Next, we **filter out bad articles**
  - First we **embed all of the titles** of the articles using a hugging face sentence embedding model. We then compare it against a set query and use cosine simililarty of the two embeddings in order to determine whether to drop the article or not. It is worth noting we artifically bump up some of the titles that contain certain keywords (i.e preview, predictions) in order to not incorrectly drop some of the articles.
  - We also drop all articles that could not be parsed correctly
- We then webscrape the odds database for all the teams, concatenating it with the dataframe of the articles to create a singular dataframe

Models Used

There are two main models we implemented so far.

1. Sentiment Analysis
   - We take all of the valid articles and their contents and score them from -1 to 1 on sentiment using a hugging face sentiment analysis model. We do this for each article for each team, followed by averaging them out for each team. The goal of this is to capture the overall sentiment of local news for the given team. Sentiment analysis is a supervised NLP technique that (as the name implies) captures the sentiment of the given text. No preprocessing of the text was required as it is a transformer based model that takes capitalization, punctuation, stopwords, etc. into account; as long as the article content was correctly scraped we passed it in as is.
2. Linear Regression
   - We then apply linear regression, using the sentiment scores for each team as the independent variable and the win differential (calculated as actual wins minus predicted wins) as the dependent variable. We do this method with a continuous outcome rather than simply predicting over and under in order to get a more accurate model. Using over/under as the outcome (binary, would have used logisitic regression) would have ignored many of the naunces associated with each data point. Thus, we opted to use a continuous variable as the outcome and then afterwards convert this to a binary over/under prediction for comparison in order to be more precise.

Model 2: KMeans Clustering (Unsupervised)

Data Preprocessing Methods

- We started by scraping team statistics from the 2022 college football season using a detailed sports database, College Football Team Data. This dataset provided a variety of performance metrics, including offensive and defensive stats for each team.
- Additionally, we gathered data on returning starters for the 2023 season. We turned this into a **percentile ranking** to quantify the strength of each team's roster based on the proportion of returning starters compared to other teams.
- We then combined these statistics into a single dataframe, ensuring all teams had both statistical performance and roster strength data.
- **Feature Scaling** was applied using **Min-Max Scaling** to normalize the data. This ensures all variables (e.g., offensive efficiency, defensive stats, returning starters) are on the same scale, preventing any one feature from dominating the clustering process due to scale differences.

Models Used

There is one main model we implemented for this stage.

1. **KMeans Clustering**
   - **Clustering Teams Based on Statistical Profiles**: We used KMeans, an unsupervised learning algorithm, to group teams based on their overall statistical profile. The algorithm works by assigning each team to one of k clusters based on its proximity to the centroid of each cluster. Initially, we experimented with different cluster counts and decided on **4 clusters**, which we determined by creating an **elbow graph**. An elbow graph plots the Within-Cluster Sum of Squares (WCSS) against the number of clusters, and the "elbow" point indicates the optimal number of clusters, where increasing the number of clusters no longer significantly reduces the WCSS.
   - **KMeans Algorithm**: KMeans begins by initializing k centroids randomly and iterating through the dataset to assign teams to the closest centroids. It recalculates the centroids based on the mean values of the teams in each cluster and repeats this process until the clusters stabilize (i.e., the centroids no longer change). This process helps identify natural groupings within the data.
   - **Cluster Quality Evaluation**: To evaluate the quality of our clusters, we used the **Silhouette Score** and **WCSS (Within-Cluster Sum of Squares)**. The silhouette score measures how similar teams are to their assigned cluster compared to other clusters. A higher score (closer to 1) indicates better-defined clusters. The WCSS (inertia) gives a sense of how tightly the teams in each cluster are grouped, and a lower score indicates better clustering. After running KMeans, we calculated these metrics to ensure the clusters were meaningful and well-separated.
   - **Dimensionality Reduction with PCA**: To reduce the dimensionality of our feature set (i.e., make the data easier to visualize and analyze), we applied **Principal Component Analysis (PCA)**. PCA is a technique that transforms the data into a new coordinate

system, where the greatest variances in the data are captured by the first few components. However, the data we had wasn't linearly separable, so we applied the **kernel trick** to map the data into a higher-dimensional space. This allowed us to capture non-linear relationships in the statistical profiles of the teams, making the data more separable. We then used PCA to reduce the number of features, enabling us to graph the clustering results in 2D, which helped us visually inspect the quality and distribution of our clusters.
    - **Insights**: These insights gave us a clearer picture of which teams were more likely to outperform or underperform their betting lines. By understanding which clusters represented higher and lower chances of success, sports bettors were equipped with a strategic edge, knowing where to place bets on teams to exceed or fall short of their predicted win totals. This approach helped identify potential **value bets**, as well as opportunities to **short** teams that might underperform based on their statistical profiles.

Model 3: Random Forest (Supervised)

Data Preprocessing Methods

- After building the initial models, we combined the insights from the **Sentiment Analysis** and **KMeans Clustering** models to create a unified prediction model using **Random Forest**.
- To prepare the data for this model, we first extracted key features from both models. This included the **sentiment score** (from the Sentiment Analysis model) and the **cluster probability** (from the KMeans Clustering model), which represented each team's placement in the cluster and its likelihood of exceeding or falling short of expectations.
- We combined these features into a single dataset, ensuring that all necessary data points were present and properly formatted for input into the Random Forest algorithm. No further feature scaling was needed, as Random Forest is a tree-based model that handles raw data effectively.
- Furthermore, we ensured that the same test and training sets were used for both the KMeans algorithm and the Random Forest model. This was done to prevent any potential data leakage, ensuring that the test set used in Random Forest was not influenced by the training set from KMeans clustering.

Models Used

There is one main model we implemented for this stage.

1. **Random Forest Classifier**
    - **Merging Insights from Sentiment and Clustering**: We used **Random Forest**, a supervised ensemble learning algorithm, to combine the insights from both the **Sentiment Analysis** and **KMeans Clustering** models into a single unified prediction.

Random Forest works by constructing multiple “decision trees,” each making its own independent prediction. The final prediction is determined by aggregating the results from all the decision trees, with the majority vote deciding the outcome. This approach significantly improves prediction accuracy by reducing overfitting and capturing a wider range of patterns in the data.

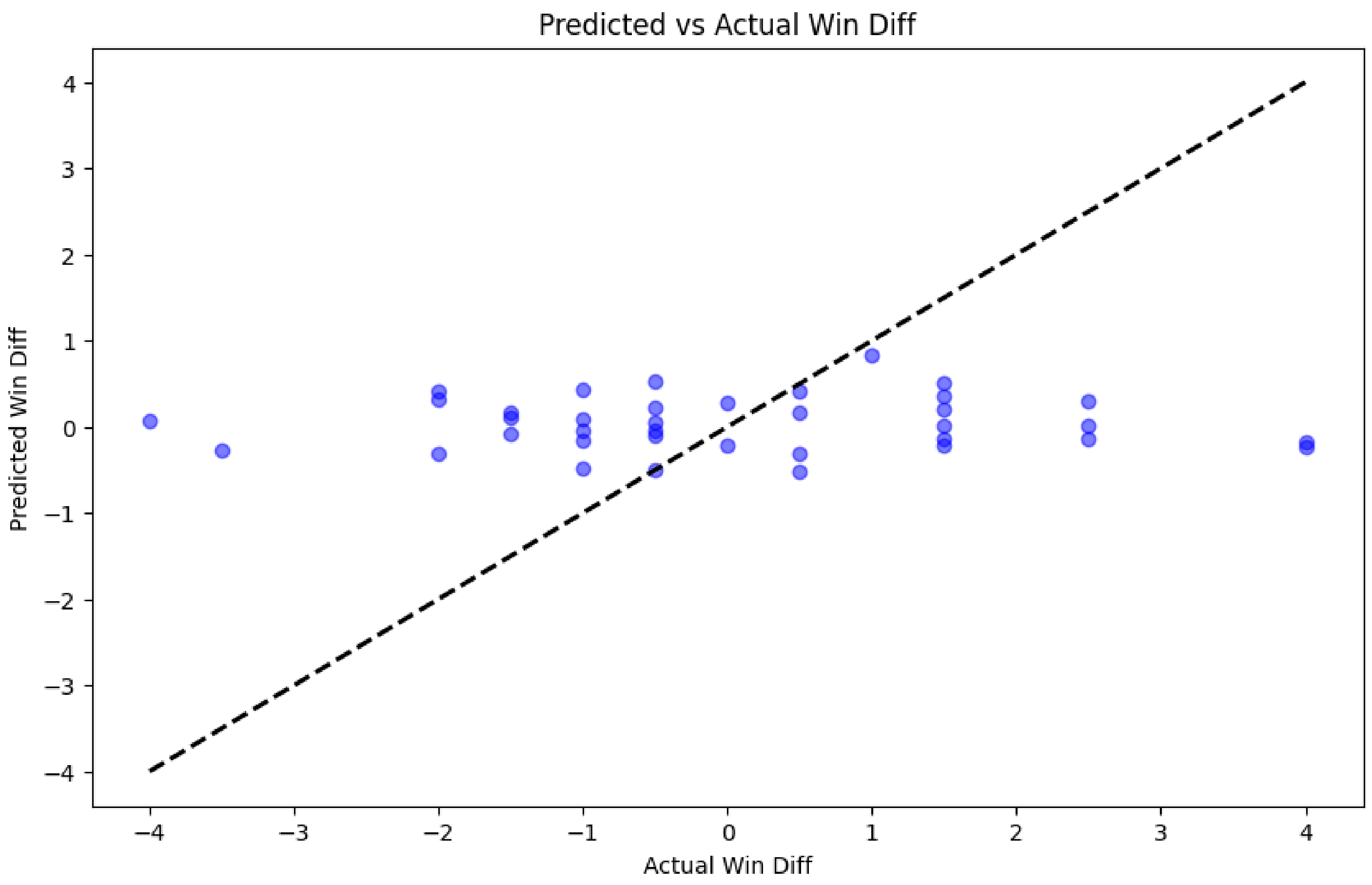
- **Random Forest Algorithm**: Each decision tree in the Random Forest is built by selecting random subsets of the features and using them to make a series of binary splits. The algorithm then calculates a final prediction based on the majority vote across all the trees. The Random Forest model works well with complex data sets and non-linear relationships, making it ideal for combining multiple insights like sentiment and clustering.
- **Outcome Prediction**: Our goal was to predict whether a team would outperform or underperform its expected win total for the season. By inputting the sentiment score and cluster probability into the Random Forest model, we were able to produce a binary prediction (over/under) for each team. The model was trained on historical data, where we used the win differential (actual wins minus predicted wins) as the target variable.
- **Model Evaluation**: To evaluate the quality of our model, we used **accuracy metrics** to ensure our Random Forest model was providing reliable and generalizable predictions. The model was able to account for complex interactions between the sentiment data and statistical cluster information, making it a powerful tool for predicting team performance.

Results and Discussion

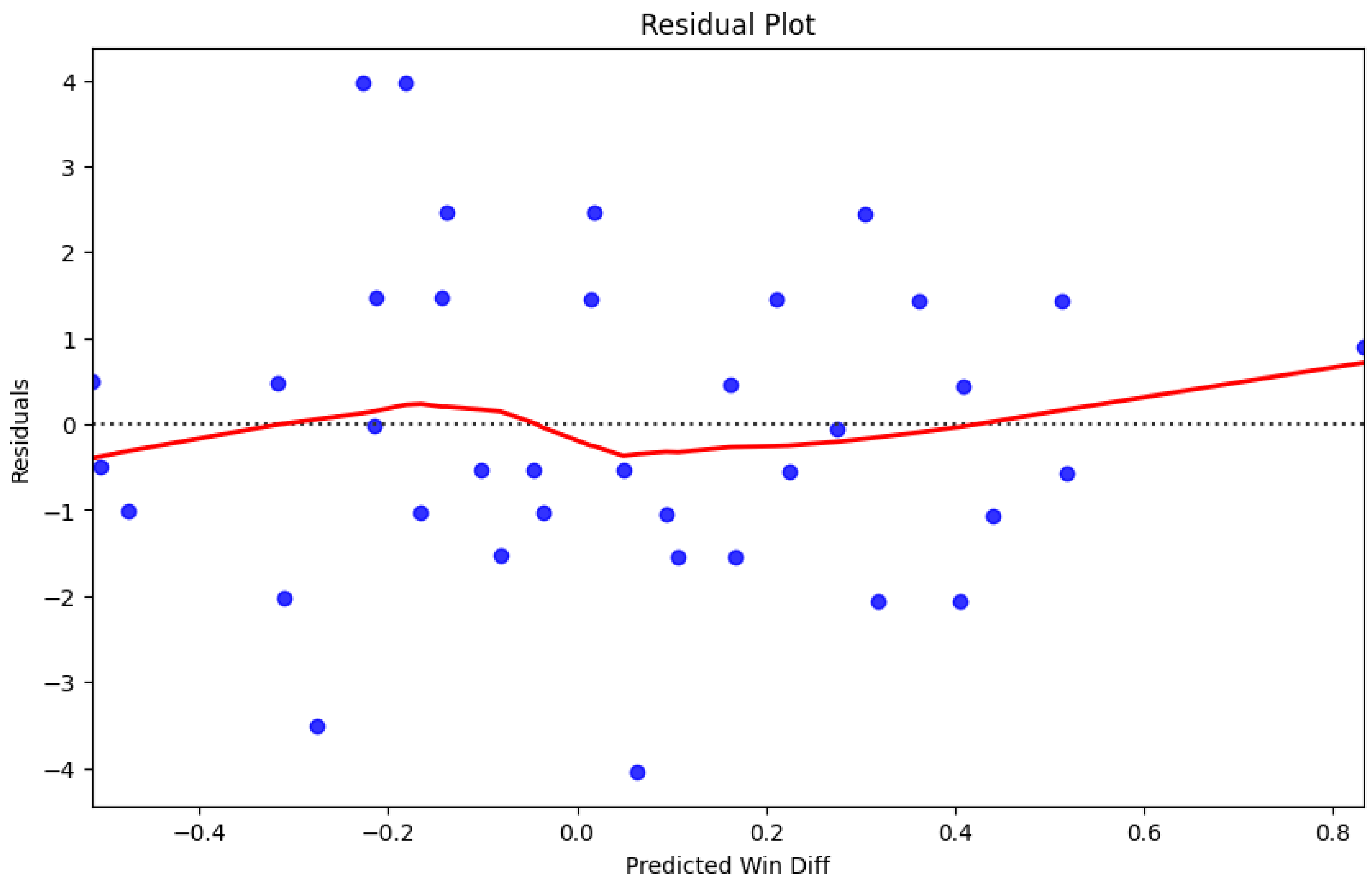
Model 1: Sentiment Analysis and Linear Regression (Supervised)

Scores:

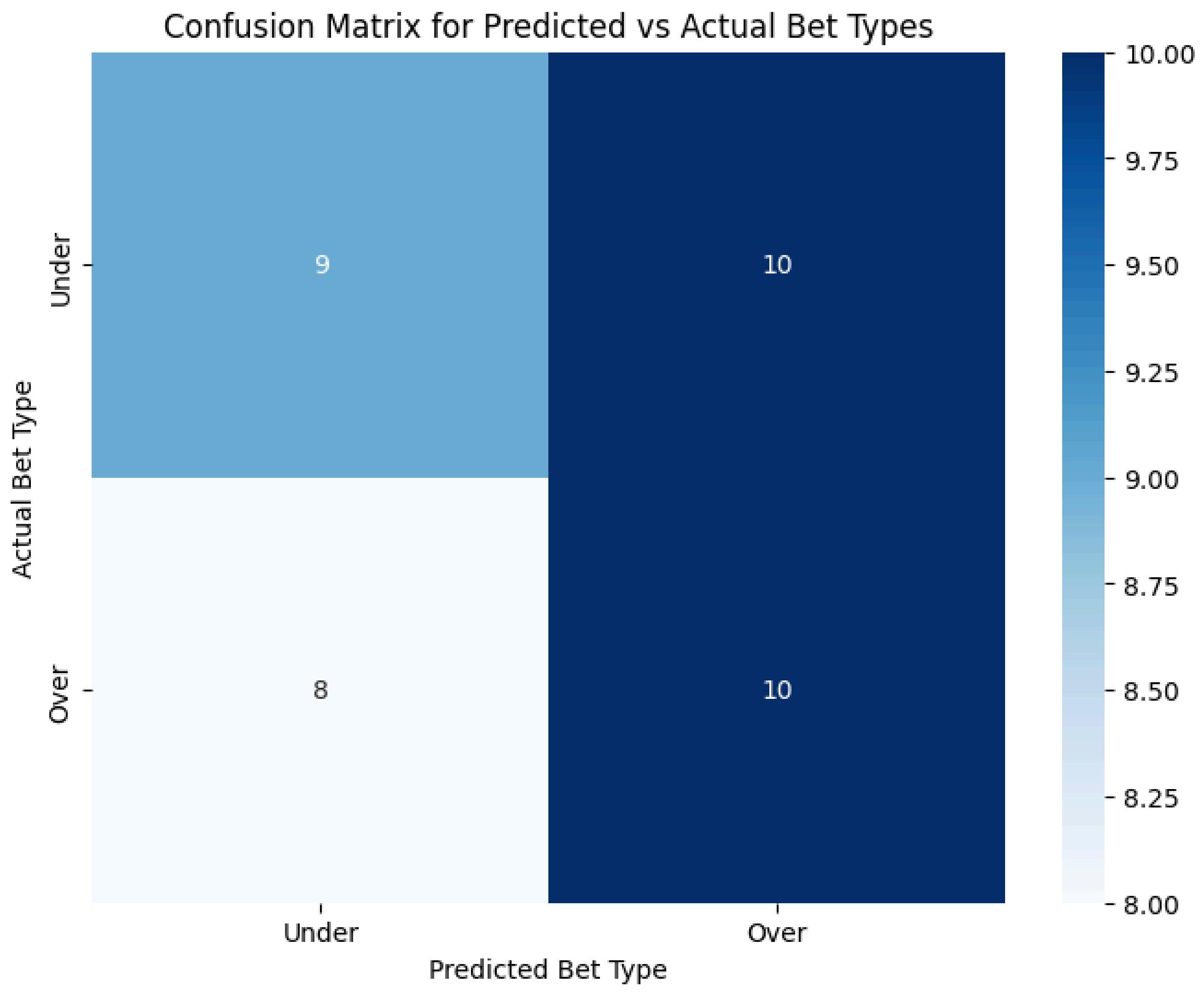
- **Mean Squared Error:** 3.3171764072001624
- **R-squared:** -0.02545207213662004
- **Accuracy:** 0.5135135135135135



- This is a basic plot of our linear regression model (sentiment of articles versus actual-predicted win loss difference)
- Trendline is relatively steep, and the datapoints are at a similar level. Correlation is very subtle.
- However, a steep trendline doesn't necessarily mean a good fit, indicated by the negative R-squared and relatively high MSE.



- This is a residual plot for the linear regression model.
- There seems to be little to no clear pattern in the graph except for outliers towards the high end of the win total. Overall, this is a good sign there is no clear section of bias in our model and is without being influenced by any specific bias.

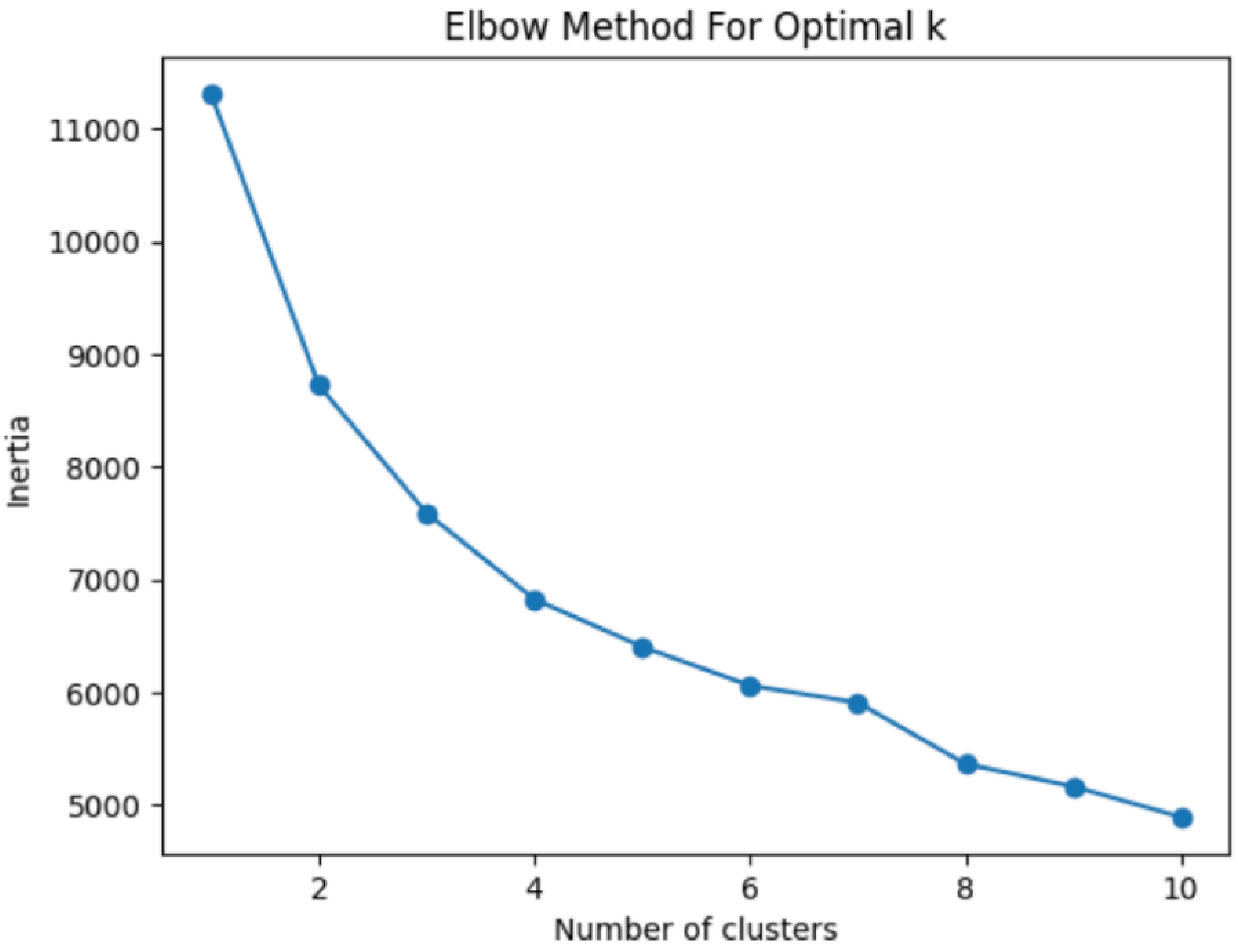


- The model is ever so slightly predicting over more than under, but overall pretty even. There is little trend here as there should be. This is good to know as we likely do not have to adjust the threshold to predict over or under (derived from predicted wins).

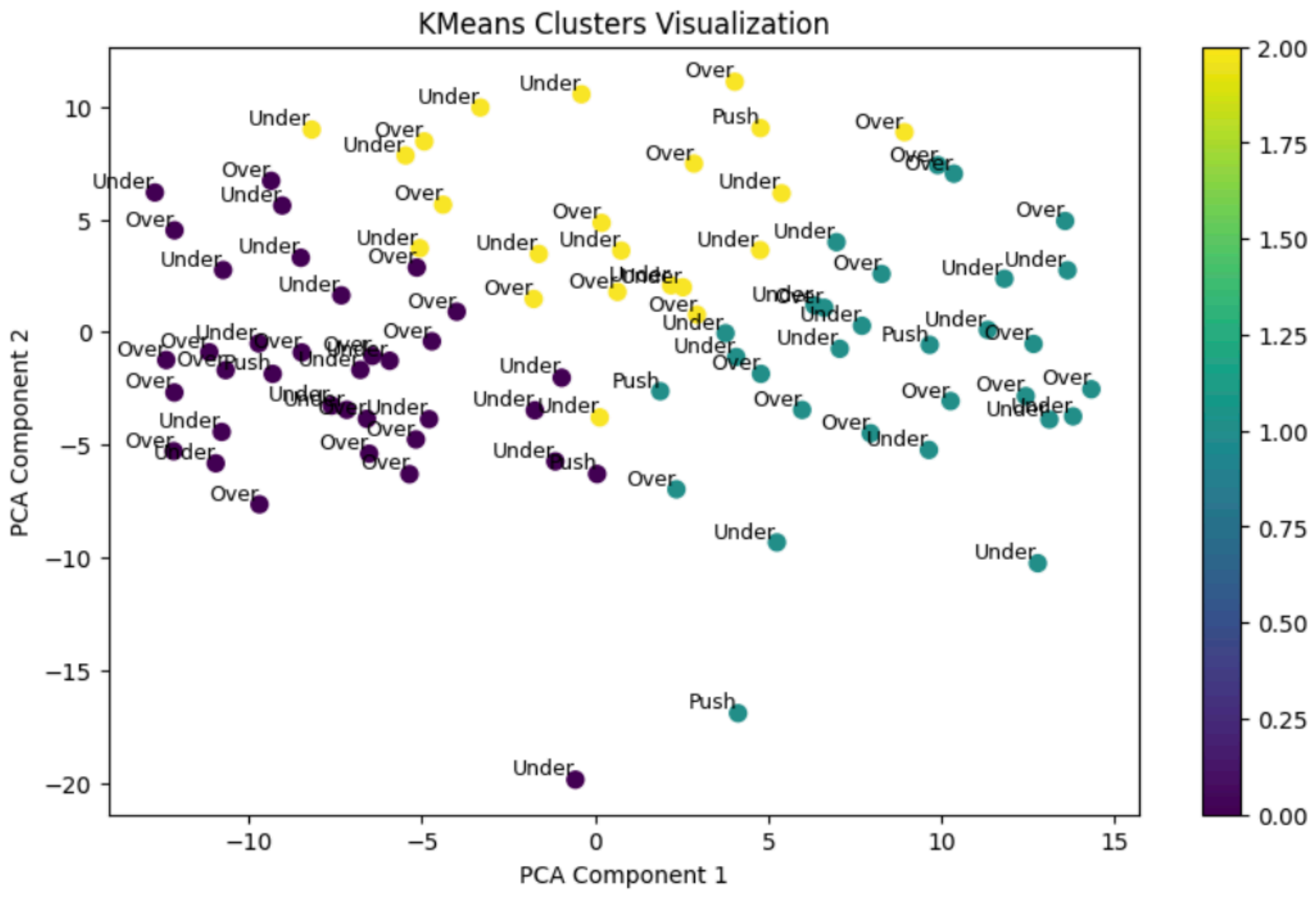
Model 2: KMeans Clustering (Unsupervised)

Scores:

- **Silhouette Score:** 0.209
- **Accuracy:** 0.52381



- This is an Elbow graph, which helps us determine the optimal number of clusters for KMeans.
- The ideal point is typically where the curve transitions from steep to more gradual, indicating diminishing returns from adding more clusters. In our case, this ‘elbow’ occurred at 4, so we chose to create 4 clusters.

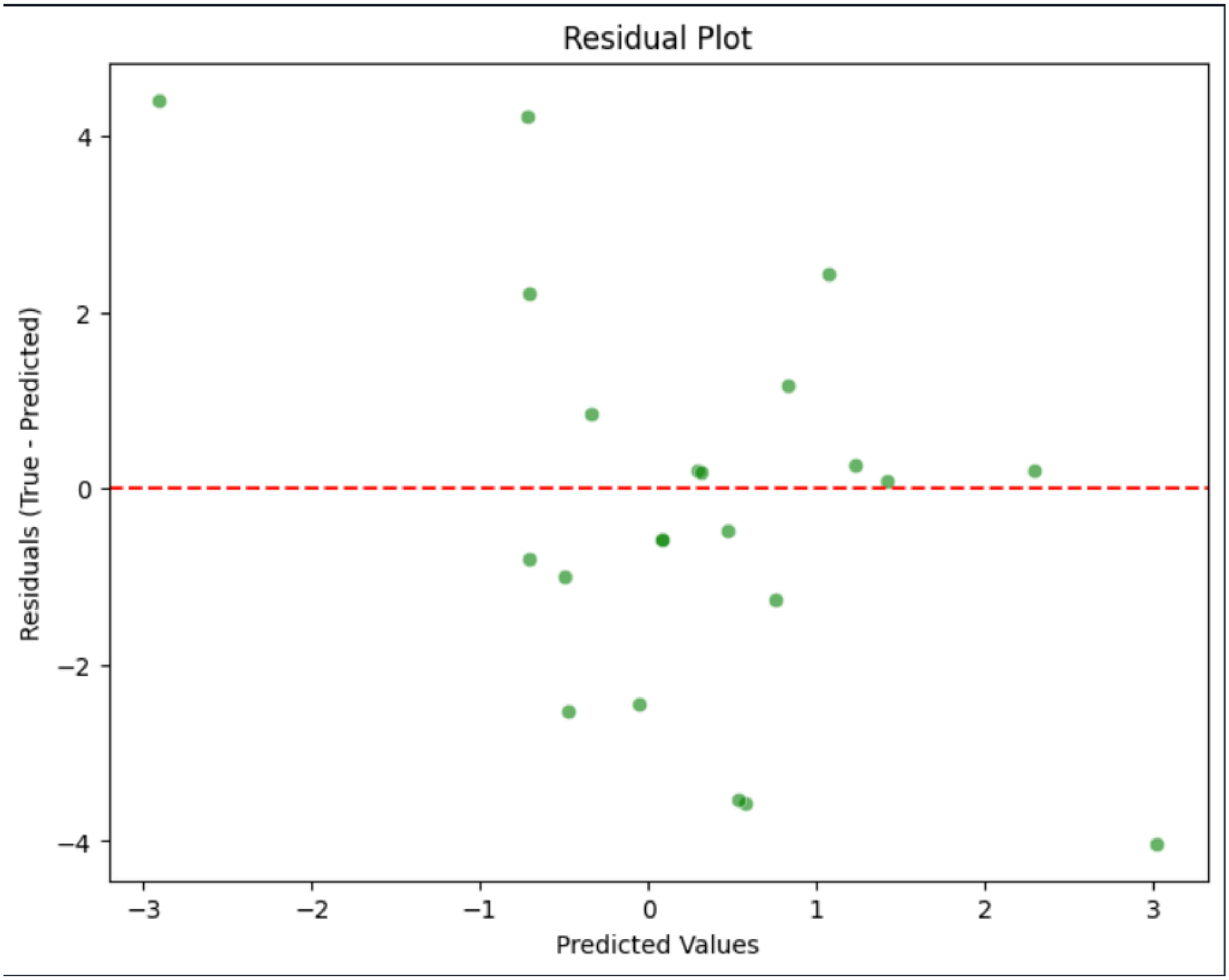


- This is the KMeans clustering output for teams with similar characteristics based on their 2022 season statistics.
- By identifying clusters with similar statistics, we can explore potential trends in betting lines.

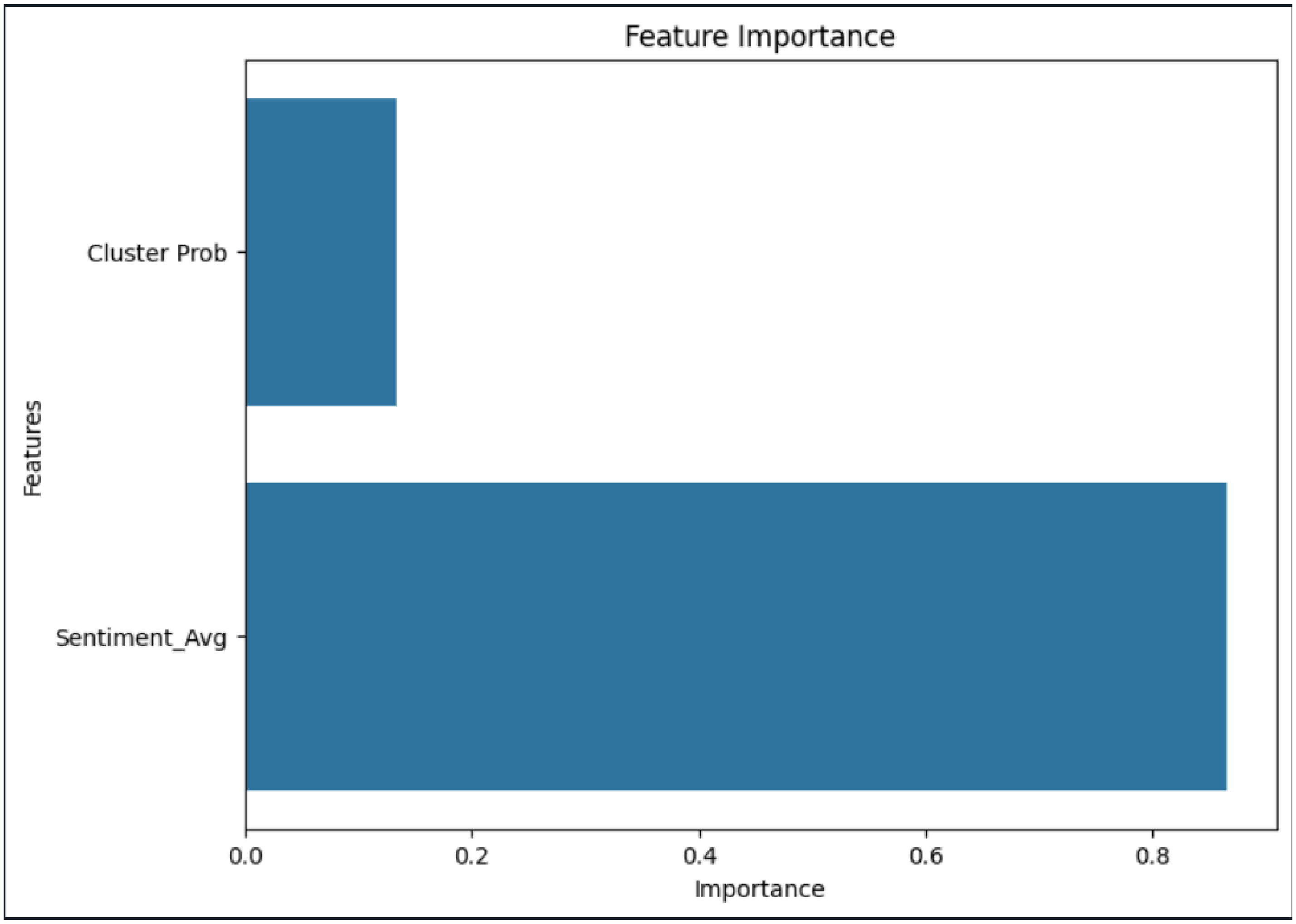
Model 3: Random Forest (Supervised)

Scores:

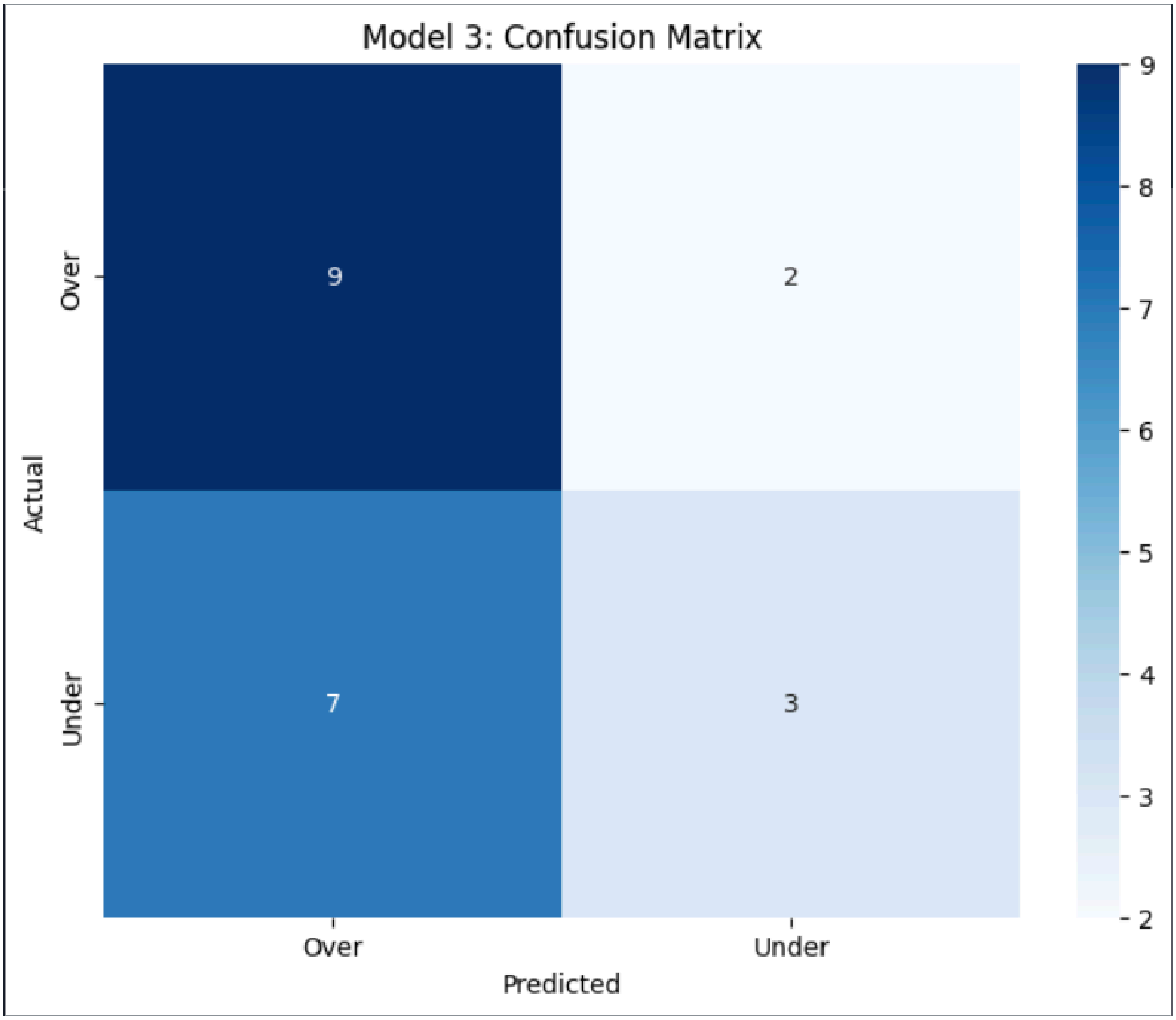
- **Mean Absolute Error:** 1.50
- **R-squared:** 0.02
- **Accuracy:** 57.14285714285714



- This is a residual plot for the random forest model.
- There is not a super clear pattern in this plot, similar to model one. This graph is a good sign there is little to no bias in this model.



- This is a basic feature importance chart of our random forest model.
- It has cluster probability with a .15 feature importance and sentiment average with a 0.9 feature importance.
- This likely means that sentiment average was vastly more important when it came to influencing the model to provide an accurate outcome.



- This model predicts over a good bit more than under, but still predicts the under occasionally. This is a contrast from the first model, where over still had a slight edge, but it was for the most part even. To fix this, there are many things one could do, such as resampling the data, or adjusting the class weights.

Conclusions

Our first model, which used Sentiment Analysis, performed slightly better than random guessing, achieving an accuracy of 51.3% compared to the 50% baseline. This result was expected, as sentiment analysis of articles is likely to capture only minor trends. Coupled with the use of a basic model like linear regression, it was unlikely to yield more than a marginal improvement, especially given the inherently unfavorable nature of sports betting.

Our second model, which incorporated KMeans Clustering, showed a slight improvement with an accuracy of 52.38% in guessing the over/under for team projected win totals. By making predictions using KMeans clustering alone, we demonstrated a correlation between certain team statistics and drew insights into how these similarities connected to projected win totals. However, our accuracy still falls short of the 53% accuracy threshold typically required to be profitable in sports betting, which is generally seen as the minimum necessary for a positive return on investment.

Finally, our third model, which combined sentiment analysis and KMeans clustering data with a Random Forest classifier, achieved a remarkable accuracy of 57.14% in over/under bets for projected team win totals. This is a profitable result, indicating that our sports betting model has successfully outperformed the Vegas odds. We are excited about this outcome and look forward to refining our models in the future to achieve real-world profitability.

Contribution Table

Name	Proposal Contributions
Charlie Hamilton	Github Pages Setup, Proposal Writeup, Gantt Chart for Proposal, Model Selection for Sentiment Analysis / Linear Regression, Data Pre-Processing and Cleaning for KMeans Clustering, Midterm Report, Model Selection for Sentiment Analysis / Linear Regression, Model Comparisons, Final Presentation, Final Recording, Final Report
Nevin Sethi	Proposal Methods, Video Recording for Proposal, Data Sourcing and Cleaning for KMeans, Data Pre-Processing for Sentiment Analysis, Data Pre-Processing for Random Forest, Results Evaluation and Analysis for KMeans, Results Evaluation and Analysis for Random Forest, Google Slides, Final Video, Model Comparisons, Final Presentation, Final Recording, Final Report
Seth Fagin	Introduction and Background, Problem Definition, Video Recording for Proposal, Gantt Chart and Contribution Table, Model Coding for Random Forest, Data Sourcing and Cleaning for Sentiment Analysis, Results Evaluation and Analysis for Sentiment Analysis / Linear Regression, Model Selection for KMeans, Model Coding for KMeans, Model Selection for Random Forest, Model Comparisons, Final Presentation, Final Recording, Final Report

Link to Gantt Chart

Gantt Chart