Predicting the Winners of Popularity Contests using Twitter Sentiment Analysis

Author: Ethan LeSage (C2059252)

Supervisor: Dr. Fernando Alva Manchego

Qualification: BSc Computer Science G400

School: School of Computer Science and Informatics, Cardiff University

Completed: 12/05/2023

Abstract

The popularity of social media sites goes hand-in-hand with the popularity of modern reality television shows. Many television programmes encourage viewers to live-tweet their opinions on social media networks, specifically Twitter, by sharing hashtags and interacting with the audience on these platforms.

Therefore, it is likely that the sentiment expressed by users on Twitter is a good litmus test for determining public sentiment towards contestants. The purpose of this paper is to determine whether data acquired from Twitter could be used to predict contestant voting shares, and by extension, winners, of television popularity contests.

The proposed solution is a machine learning model which uses a dataset of tweets and their related sentiment as calculated by a sentiment analysis model. The model takes contestant-wise dataset statistics as input, and outputs the vote share predicted to be received by each contestant.

The dataset was collected using the Twitter API and analysed with TweetNLP. The model was a Support Vector Regression model to which features were iteratively added. Other data pre-processing and post-processing was implemented to normalise both the input features and the predicted vote share. These features were analysed with an ablation study.

The model developed was able to predict voting share and order of contestants in the final with accuracy significantly greater than a strong baseline model.

This accuracy level was sufficient to prove that Twitter data, combined with an intuitive machine-learning model, can be used to predict contestant voting shares with accuracy nearing that of betting exchanges.

Acknowledgements

I would like to thank my supervisor Fernando Manchego for his guidance throughout this project. Without his input and encouragement this project would likely not have been possible. Additionally, his support has encouraged me to potentially pursue a career in machine learning or data science.

Thanks to the developers of TweetNLP at Cardiff University, whose project has helped boost the accuracy of this project.

I would also like to thank all of the faculty at Cardiff University who have taught me throughout my degree, all of whom have contributed to this project indirectly through their teaching and support.

Contents

1 Introduction

In this project, we aim to build a model that can accurately predict the audience voting percentages of TV popularity contests using sentiment analysis applied to tweets. Sentiment analysis is the process of analysing text and classifying the author's sentiment as positive or negative *(Camacho-Collados, J. et al. 2022.)*.

In this project, the phrase "TV Popularity Contests" refers to television programmes that involve the selection of a winner through one or more public votes. These types of programmes have made popular British television for the past 22 years *(Big Brother (TV Series 2000-). 2023.)*. These shows often have a large proportion of their viewers live-tweeting as the show takes place *(Carmody. 2013)*. Due to the presence of massive amounts of real-time sentiment data, we hypothesize that it should be possible to predict the outcome of these votes by analysing Twitter sentiment.

Specifically, this project involves building a model to predict the voting outcomes of "I'm a Celebrity Get me Out of Here" (also known as "I'm a Celeb"). This programme was chosen because it involves the least number of outside factors of any popularity contest. Contestants only ever leave the show because of a public vote unless they choose to leave. This separates it from other programmes like Love Island, where contestants may be eliminated for other reasons that are unrelated to public sentiment *(Love Island (2015 TV Series). 2023)*.

1.1 Motivation

Over the past decade social media has exploded in popularity, with a large proportion of users sharing their opinions online. For instance, in 2014, 72% of Twitter users reported tweeting about a television show while it was live *(Twitter. 2014)*.

When trying to predict the outcome of elections or other popularity contests, the traditional method is to use polling to gauge public opinion *(Hillygus, D. 2011).* In more recent elections, attempts have been made to predict or nowcast the outcome of elections using sentiment analysis of social media, predominantly Twitter, as in *(Page, L. 2015).*

We believed it was possible to substitute polling for Twitter sentiment analysis to predict or nowcast the outcome of TV popularity contests. In essence, we aimed to predict how people would respond to a poll by analysing statements they have previously made on Twitter.

1.2 Objectives

The main aim for this project is to design, implement and evaluate a model that can predict the voting share of candidates in the final vote of *I'm a Celeb*. The research questions we have defined are as follows:

RQ1. Is it possible to design and implement a model capable of predicting (nowcasting) the winner of *I'm a Celeb* on the day of the final to an accuracy of 80%?

RQ2. Is it possible to design and implement a model capable of predicting (nowcasting) the winner and voting share of *I'm a Celeb* on the day of the final with accuracy close to or exceeding the implied probability presented by betting exchanges?

These two research questions lead to the following null hypothesis which we will aim to refute:

H1'. It is impossible to define and create a model capable of predicting (nowcasting) the winner of *I'm a Celeb* on the day of the final to an accuracy of 80%.

H2'. It is impossible to design and implement a model capable of predicting (nowcasting) the winner and voting share of *I'm a Celeb* on the day of the final with accuracy close to or exceeding the implied probability presented by betting exchanges.

Of the two research questions, the second was harder to answer because betting exchanges (also known as prediction markets) are for all intents and purposes the most accurate predictor of competitions *(Franck. 2016).*

In order to answer these two questions, another question was defined as an intermediate step. This was as follows:

RQ3. Is it possible to build a dataset containing at least three thousand tweets for each episode of the competition, spanning all seasons and episodes that are available?

This question was needed because the creation of an adequately sized dataset would be required before building any models.[1]

1.3 Approach

This problem was approached by iterative development on a base model. The project started with building a dataset of 346,000 tweets (almost all that were available to be collected), concerning the last 14 seasons of *I'm a Celeb*.

Next, a baseline model was built to analyse tweets and predict vote counts for each season. This baseline model was a very simple non-ML model which did little more than tally up the number of positive tweets mentioning each contestant. This enabled evaluation by comparing predicted vote shares to the real voting shares published by ITV.

Using what was learned from the baseline model, a more complex machine learning based model was built. This was iteratively developed until completion to create the final model. Changes made throughout development include feature design, architecture modifications, as well as data quantities and filters. For each change evaluation was performed using cross evaluation to gain the values of three metrics; Mean Absolute Error, Absolute Error On-the-Margin, and Degree of Correct Order. This iterative development was stopped upon answering the two research questions outlined above.

To evaluate the final model, researchers compared it's results to the baseline model and to betting exchange predictions in order to answer the two research questions.

[1] Please note, research question number 4 from the initial plan stated we would evaluate the effectiveness of this model against alternative scenarios such as elections. Due to time constraints, this research question was not attempted in the final project. We instead decided to focus on maximising the accuracy of our model with the time available.

1.4 Contributions

The first major contribution was the creation of a large, cleaned dataset. This was collected by scripts that used the Twitter API. Sentiment values were annotated using TweetNLP. The dataset was stored in final processed CSV files.

The next big contribution was the creation of three models: a baseline Twitter model, a main machine-learning Twitter model, and a machine-learning betting exchange model for comparison. The main machine-learning model was the bulk and focal point of this project. It used 10 well-selected features to make predictions on the voting share of candidates. The betting exchange model used betting exchange odds to make predictions on voting share. This model was constructed to aid in evaluating the machine learning Twitter model.

To evaluate the Twitter machine learning model, it was compared to three benchmarks: Guessing, achieving 80% accuracy in predicting the winner, and achieving similar accuracy to the betting exchange model overall.

Final analysis was also conducted, comparing the impact of selected features throughout development, and deciding which were the most important. These important features are recommended to other researchers for similar machine learning projects.

2 Background

2.1 Context Review

2.1.1 TV Popularity Contests

The phrase "TV Popularity Contests" refers to television programmes that involve the selection of a winner through one or more public votes. These programmes have existed in the UK for over 22 years now, the first being Big Brother. *(Big Brother (TV Series 2000-). 2023.)*. This format of programme includes 'The X Factor', 'Dancing on Ice', 'Strictly Come Dancing' and many others.

User engagement through voting is a concept which immediately grew upon its inception. The ability to charge users directly through voting over premium phone lines was highly profitable, and so many different programmes have been created since Big Brother. For example, season 7 of 'The X Factor' pulled in £5m through voting charges alone *(Plunkett, J. 2010).*

2.1.2 Betting Exchanges or Prediction Markets

As stated in research question 2, one goal of this project is to build a model with accuracy close to or exceeding that of betting exchanges.

With traditional betting, a bookmaker sets the odds for the outcome of a competition or other event. Customers give the bookmaker their money, and if the outcome does occur, the bookmaker returns the customers money as well as their winnings. These bookmakers generate profits by offering less-than-fair odds. On average, bookmakers have a 6% profit margin built in to their odds *(Smarkets. Unknown Date)*.

A betting exchange is more complicated. A betting exchange is a marketplace where individual users can act as either the bookmaker or the customer. When placing a bet at an exchange, the gambler who is betting on an event occurring (known as a back bet) is matched to another gambler, who is betting that the event will not occur (known as a lay bet). The owners of the exchange act as an independent third party, who collects money from both sides, and pays all the money out to the winner, after subtracting a fee *(NewBettingSites. Unknown Date).*

These exchanges act similarly to a stock market, but with these markets the odds are what fluctuate and determine betting, whereas in a stock market the price of stock fluctuates and determines purchasing. Figure 1 below shows how these odds fluctuate throughout the course of a football match.

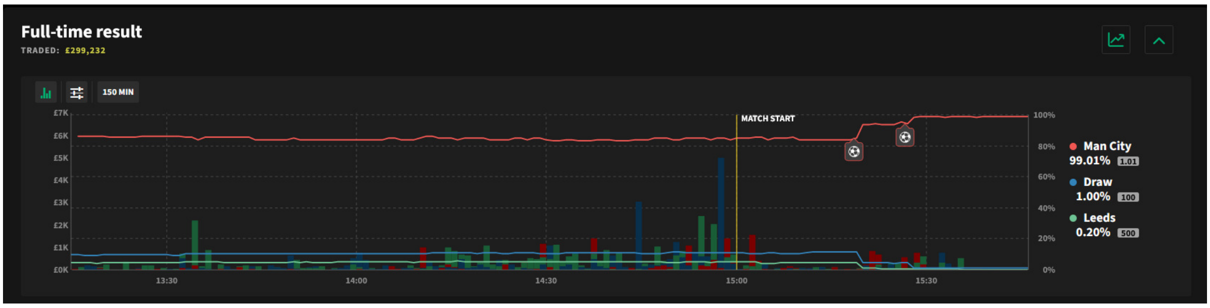


Figure 1 – Example of a betting exchange acting similarly to a stock market, note the changes in odds as teams score.

Because gamblers are only matched together if they can agree on fair odds, the market is encouraged to come to a consensus on which odds are fair. The large number of gamblers who participate in these markets, combined with the lack of a profit margin, means the odds offered by exchanges are usually more accurate than any prediction made by a single person *(Franck, E. Verbeek, E. and Nuesch, S. 2016.).*

> *"The nature of the market as a peer-to-peer platform means that supply and demand becomes the overriding principle behind the odds that are offered, which benefits the bettor."*
> (NewBettingSites. Unknown Date)

The accuracy of these markets is the reason we used them as a benchmark. If our model were to exceed the accuracy of these markets, it would be one of the best systems for predicting the winner of *I'm a Celeb*, without insider information, in the world.

2.2 Solutions to the Problem

In this section the main technologies employed by researchers for building the model were considered.

2.2.1 Sentiment Analysis

> *"The sentiment analysis task… consists of predicting the sentiment of a tweet with one of the three following labels: positive, neutral or negative."*
> (Camacho-Collados, J. et al. 2022.)

A sample of tweets large enough to be representative of the show's audience needed to be in the hundreds of thousands. As such, manual identification of positive and negative tweets by a human being would not have been time efficient. Instead, an automated system of deciding whether a tweet represents a positive or negative opinion had to be implemented. This is called sentiment analysis.

By employing sentiment analysis, huge numbers of tweets can be assigned sentiment values in a relatively short time period when compared to a human annotator completing the same task.

2.2.2 Regression

A technique would be needed to convert the large number of sentiment analysis results into a single value representing the predicted percentage a given contestant would receive in the final vote.

The relatively large amount of data available on Twitter means this project lends itself well to machine learning. Any machine learning technique which takes a number of numeric features and outputs a continuous numeric value is referred to as a regression technique, so this was chosen for the project.

"[Regression is] a statistical technique that is concerned with fitting relationships between a dependent variable, y, and one or more independent variables, x_1, x_2, …,"
(A Dictionary of Computer Science. 2016)

A regression-based system for this project would require a large list of features (independent variables) for each contestant. For example, these features may include the number of positive tweets mentioning a contestant, the number of likes received by positive tweets mentioning a contestant, etc. The regression model would take a large number of voting results (dependent variables) and attempt to find a correlation automatically. The resultant model could then be used to predict as-yet-unseen voting results.

A few libraries are available which integrate into Python code with relative ease and supply developers with regression models. One of these is scikit-learn, which was chosen for use in this project due to ease of implementation and rigorous documentation *(Pedregosa, F. et al. 2012).*

The specific regression model we chose from scikit-learn was support vector regression, or SVR. SVR is a model which trains itself by essentially finding a plane of best fit within the feature set provided *(Sethi, A. 2020.).* When in use, the model maps the features provided to an intersect on the plane and returns the result.

This model was selected because it is supposedly effective with a large number of training features *(Scikit-learn Developers. 2023).* As we were not aware how many features we would be using, this was important. Further, the model is quite flexible, and widely adopted in the machine learning space, meaning that support would possibly be easier to find.

2.3 Related Work

This section includes analysis of papers relevant to the project. Four main papers are discussed, and the findings from all four were later applied in some way to the project.

2.3.1 Machine Learning for Predicting Elections in Latin America…

(Brito, K. and Adeodato, P. 2023) develop a machine learning based approach to predict elections in Latin America based on social media data. This paper attempts to answer the following research questions:

1. Is it possible to define a process and create a machine learning (ML) model capable of predicting election results based on the social media (SM) performance of candidates?
2. Is it possible to define a process and create an ML model capable of performing daily nowcasting of election results based on the SM performance of candidates?

These two research questions are addressing a very similar problem to this project, however in this paper the researchers are using data from Facebook and Instagram as well as Twitter. This is not possible in our case because the APIs for these social media sites are far more exclusive than Twitter's.

More specifically, throughout this paper the researchers use large quantities of data from these social media sites paired with polling results throughout elections in Latin America to build a model capable of predicting the outcomes of said elections. One notable difference with the researchers technique is that they are defining a process which will create a new model for every election, learning from polls and data throughout the election, and finally making a prediction at the end. Our project was not able to use such a technique, as public voting data is only made public for *I'm a Celeb* at the end of each season.

For the Twitter-based features of their model, the researchers used the following metrics from table 1:

Feature Name	Description
TTPosts	Sum of posts in the period
TTLikes	Sum of likes in the period
TTRetweets	Sum of retweets in the period
TTLikesPPost	Average of likes per post in the period
TTRetweetsPPost	Average of retweets per post in the period

Table 1 – Twitter features and descriptions from the researcher's model. 'Period' refers to the 300 days before an election.

These features are notable because they have been found to be effective for this application of a regression model.

> *"Results demonstrated that it was also possible to achieve a high level of accuracy in predicting the final vote share of the candidates, providing competitive or even better results than the traditional polls."*
> (Brito, K. and Adeodato, P. 2023)

Therefore, those features were deemed likely to be effective when applied to our research problem as well. This information was carried forward and investigated in relation to our topic later in section 3.3.

Further, the authors suggest a couple of effective evaluation metrics to assess the performance of the prediction models:

1. Mean Absolute Error (MAE): The average absolute error between prediction and actual results for all candidates.
2. Absolute Error on the Margin (AEOM): The absolute value of the difference between the margin separating the two leading candidates in the prediction and the actual vote.

These two metrics are adopted into our project and outlined in more detail in section 4.3.

2.3.2 How Efficient is Twitter: Predicting 2012 U.S. Presidential Elections…

(Attarwala, A et al. 2017) attempt to use Twitter data with sentiment analysis and support vector machines (SVM) to predict the winner of US elections and compare this to the accuracy of prediction markets (betting exchanges).

This paper was important because the objective is very similar to that of this project, which also aims to predict the winner of a vote using Twitter data, with accuracy close to a betting exchange.

Using a dataset of 40m tweets the researchers built and trained a sentiment analysis model which classifies the sentiment of each tweet towards each candidate in the 2012 US presidential election. This was built by identifying the following features representing statistics of each tweet:

1. Unique Parts of Speech Count
2. Average sentence length
3. Number of election keywords
4. Number of positive and negative sentiment keywords
5. Average number of positive and negative sentiment keywords per sentence

3000 sets of feature-classification pairs were fed into a SVM model for training. This approach was considered for our project, but upon later evaluation of existing sentiment analysis models in section 3.2, the model from this paper was deemed inferior. The results of evaluation of this model can be seen in table 2 below:

Model	Precision	Recall	F1
SVM Custom Model	0.8	0.027	0.052

Table 2 – Precision, recall, and F1 for the researcher's custom model applied to positive tweets

The final vote share on any given day, for any candidate $C1$ is calculated from the now-classified tweet sentiment as follows:

$$\frac{\frac{Pos_{C1}}{Neg_{C1}}}{\frac{Pos_{C1}}{Neg_{C1}} + \frac{Pos_{C2}}{Neg_{C2}}}$$

$$\begin{aligned}&where:\\&Pos = Count\ of\ positive\ tweets\\&Neg = Count\ of\ negative\ tweets\\&C1 = Contestant\ 1\\&C2 = Contestant\ 2\end{aligned}$$

Using this system to predict the vote share, the researchers were able to predict the winner of the election successfully. However, no correlation was found between the predicted vote share and the prediction made by betting exchanges.

The rigorous nature of this unsuccessful paper demonstrates how difficult it can be to create a model which predicts with accuracy rivalling betting exchanges. While this paper did attempt to solve a similar problem to ours, only the structure was carried forward into our project. We also used the results of tweet classification to predict a vote, but in our case a second model was used to map sentiment of tweets to vote share, rather than a simple equation.

2.3.3 Do Retweets Indicate Interest, Trust, Agreement?

Metaxas, P (2014) investigates the significance of retweets in relation to sentiment, through use of a survey in which 316 participants took part. This paper was looked into to establish whether retweets should be included as a feature of our prediction model.

The demographics of these 316 participants was highly educated (43% having a master's degree or higher), but evenly representative of gender (43% male and 54.4% female) as well as race (46% white). The findings should be relevant to all of Twitter, including tweets regarding *I'm a Celeb*.

One important finding was that 82.5% of the participants responded that they retweet a few times per week, or more rarely. 20.7% said they retweet a few times per year or less. This shows that users do not retweet considerably often, so retweets must be of some value since they occur so rarely.

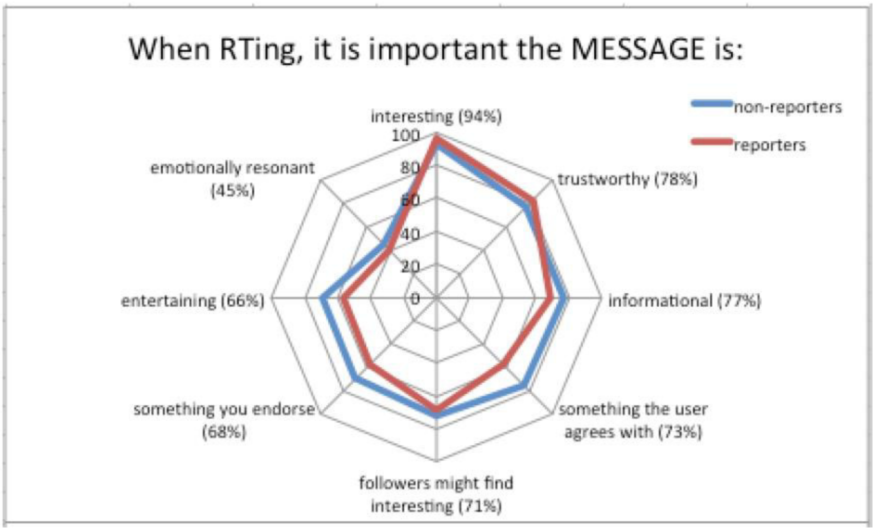


Figure 2 – Important factors determining a retweet for reporters and non-reporters

Figure 2 from the paper shows that of non-reporters (the main demographic for tweets inputted to our model), 68% believe it is important that they endorse a tweet before retweeting, and 73% believe it is important that they agree with a tweet before retweeting. This shows that retweets do in fact indicate some level of agreement. As such, it was decided that including retweets in our model would likely be beneficial.

2.3.4 A Clustering Analysis of Tweet Length and its Relation to Sentiment

Mayo, M. 2015 aimed to determine whether length of tweet relates to sentiment. We used this to decide whether a length limit should be imposed on tweets inputted to the model.

In this paper, researchers aimed to determine whether the number of characters in a tweet correlates in any way with the tweet's sentiment score. The findings of this paper would allow us to determine whether a tweet length limit should be employed, and if so, what value would be appropriate.

The researchers gathered a dataset of English-language tweets with a size of several hundred megabytes. These tweets were analysed for sentiment using a version of AFINN-

111 *(Wormer, T. 2022)*[2], which had been modified to enable the evaluation of terms frequently used on Twitter. They then divided the dataset into two clusters, cluster #0 which contained shorter tweets, and cluster #1 which contained longer tweets.

Attribute	Full Data	Cluster #0	Cluster #1
length	61.7673	41.1478	104.3943
sentiment	2.0732	1.2322	3.8120

Table 3 – Sentiment extremity results for dataset divided by length

Figure 4 shows the results of the analysis of these two clusters. Cluster #0 has a centroid sentiment-extremity score of 1.23. Cluster #1 has a much higher sentiment-extremity score. This shows that there is greater presence of extreme sentiment among longer tweets. The average sentiment polarity (positive vs negative) of the tweets in the dataset had no correlation with tweet length, however.

> *"This is… intuitive, since a single, focused, coherent thought from the human mind will have a tendency to progress linearly in its reasoning. Any progressive reasoning is likely to carry the previous portion of its argument forward. Hence, a positive thought, idea or opinion is likely to continue on a continuous trajectory to its completion."*
>
> (Mayo, M. 2015)

Essentially, a longer thought process is likely to result in a more extreme opinion since a conclusion is more likely to be reached upon completion. Our model needed to make use of extreme sentiment rather than ignore it, so we decided to not limit our dataset by tweet length.

[2] Page last edited in 2022, but the AFINN-111 model itself predates 2015.

3 Preliminary Investigations

This section outlines some experiments to determine characteristics of the data that were to be collected and used in the model.

3.1 Selecting an Appropriate Popularity Contest

In order to select an adequate TV popularity contest for our experiments, we established the following selection criteria (table 4).

#	Criterion	Reasoning
1	Contestants must be eliminated solely as a result of the public vote, with no intervention from judges or from the production company.	This allows us to isolate public sentiment as the sole cause of an elimination or victory, minimising outside factors in deciding or predicting the winner.
2	Public voting results must be publicly available for a large number of seasons, at least for the final vote.	This will enable us to evaluate and analyse our results through every step of iterative development of the model.
3	Every contestant which has not yet been eliminated must be present for the production of each episode, meaning no new contestants may enter after the first few days of the competition.	This criterion also helps eliminate outside factors, ensuring that a contestant who is predicted to win will not be usurped by a preferred candidate who was not present for the original prediction.

Table 4 – Selection criteria for easy application of an ML model to TV popularity contests

Despite the large number of TV popularity contests available, very few satisfy all three of these requirements, as can be seen in table 5.

TV Popularity Contest	Violation Number(s)	Suitability
Love Island	1, 3	Not Suitable
The X Factor	3	Not Suitable
Strictly Come Dancing	1	Not Suitable
Dancing on Ice	1	Not Suitable
I'm a Celebrity Get Me Out of Here	None	Suitable
Big Brother	1, 2	Not Suitable

Table 5 – Popularity contests and their suitability

I'm a Celeb was the only show tested which satisfied all three requirements, and so it was chosen for this experiment.

3.2 Selecting a Sentiment Analyser

Many different sentiment analysis models exist, and a few were considered for this project. Namely, Google's 'Cloud Natural Language API' *(Google. Unknown Date)*, the open source 'TextBlob' *(TextBlob. 2020),* and Cardiff University's 'TweetNLP' *(Comacho-Collados, J. et al. 2022)*.

Google's API was immediately discarded because of high pricing. The API is priced at $1 per 1,000 pieces of text analysis *(Google. Unknown Date)*, meaning a dataset of over 100,000 tweets would cost over $100. Not knowing the possible size of our dataset, this option was eliminated. In the end our dataset reached 346,000 tweets, so Google's API would have cost us $346. This left just TweetNLP and TextBlob as the two free contenders.

TweetNLP and TextBlob have some key differences. TextBlob is a general purpose NLP (Natural Language Processing) library with a large variety of different functions, one of which is sentiment analysis. TextBlob is designed for any text, not specifically tweets. It is designed, in part, for speed and efficiency. TweetNLP is designed and trained for use on tweets. It has a smaller variety of functions, one of which is sentiment analysis. One important feature of TweetNLP is that it makes use of emojis when calculating sentiment, whereas TextBlob does not.

TweetNLP attempts to categorise tweets into either positive, negative, or neutral, but TextBlob categorises them into only positive or negative, on a scale of -1 to +1. Having a neutral category is advantageous to us because not all tweets are expressing a clear positive or negative sentiment towards each contender.

To determine whether we should use TweetNLP or TextBlob, we devised a test to determine which of the two was more accurate for our specific use case. A sample ($n = 100$) of *I'm a Celeb*-related tweets was collected. Of these tweets, 70 were positive, and 30 were neutral or negative (as classified by manual annotation).

The experiment was performed by applying both sentiment analysis techniques to these tweets, recording whether each tweet was categorised as positive, negative, or neutral. Then, a researcher manually recorded the true sentiment of these tweets. An extract of the experiment's recordings can be seen in appendix item B. By comparing the model-predicted sentiment and the sentiment annotated by a human researcher, we could compute the accuracy of each model using precision, recall and F1.

Sentiment Processor	Precision	Recall	F1
TweetNLP	0.9344	0.6628	0.7755
TextBlob	0.8548	0.6386	0.7311

Table 6 – Precision, recall and F1 results for TweetNLP and TextBlob applied to positive tweets

Table 6 shows the results of this analysis for positive tweet classification. As can be seen, TweetNLP achieved marginally better results for all three metrics. Importantly, for TweetNLP precision was very high, meaning that positive results reported by the model can be relied upon.

In the end, TweetNLP was chosen because it was more accurate, it was designed specifically for tweets, and it was built mostly in-house at Cardiff University. This meant support would likely have been easier to acquire if needed.

3.3 Characteristics of Tweets to be Collected

Before starting the project, a number of different tweet characteristics were investigated that could be enforced during the iterative development of the model. These ideas were all posited to potentially increase the accuracy of the model upon implementation.

3.3.1 Tweet Length

This avenue was explored because it could be the case that shorter tweets are more straightforward and therefore easier for a sentiment analysis model to classify. As explained in section 2.3.4, *(Mayo, M. 2014)* suggests that longer tweets tend to have more extreme sentiment associated with them. This means that longer tweets should be included because their sentiment is often more clearly defined. Further, until 2017 tweets were limited to 140 characters, and from then have been limited to 280 characters *(Reimann, N. 2023.).* As such there is already a limit enforced by Twitter. We therefore chose not to enforce a length limit.

3.3.2 Tweets Mentioning Multiple Contestants

Many tweets express contrasting opinions for two or more contestants. With a traditional system of whole-tweet sentiment analysis, this kind of tweet will only be attributed one sentiment value. Since both contestant's names will be found in the text, both contestants will erroneously be given the same score.

```
I actually wish it was Edwina is the final and not Mel #ImACeleb
```

Figure 3 – Example of a real tweet expressing differing sentiment towards multiple contestants

The model available within TweetNLP says the sentiment for figure 3 is most likely negative, even though the tweet is expressing positive sentiment towards Edwina and negative sentiment towards Mel.

There are two potential routes to combat this issue. One would be to split tweets according to which contestant is mentioned in each part. This was unlikely to work successfully because it would be hard to automate this process while retaining all the relevant context embedded within the tweet. The above example would need to be separated into the two sentences in figure 4 to retain context.

```
I wish Edwina was in the final
I wish Mel was not in the final
```

Figure 4 – Example of division of tweet which adequately retains context

The other alternative was to simply filter out all tweets that mention multiple participants. This was very likely to work and improve accuracy but would considerably limit the size of the dataset.

In the end, experiments showed that filtering out tweets which mention multiple participants reduced the dataset too much and damaged the accuracy of the model overall. As such, this type of tweet was not filtered out, and a more accurate way to classify them was left as future work.

3.3.3 Weighting Likes, Retweets and Replies

On Twitter, likes and retweets are the most popular method for expressing agreement towards a given tweet. It would likely be beneficial to assign weightings to tweets according to these factors.

The problem this presents is deciding what strength of weighting to apply to each tweet depending on their likes and retweets (assuming use of a non-ML model). To the best of our knowledge, little research has been performed into this problem. According to *(Metaxas, P. 2014),* 73% of non-journalists agreed with the statement "When retweeting, it is important the message is something the user agrees with". Therefore, a weighting could be applied to every tweet according to the following:

$$Weighting = 1 + 0.73 * Retweets$$

Alternatively, different weightings could be iteratively assigned to retweets through grid search until the accuracy of the model is maximised, before incorporating the optimal weightings permanently.

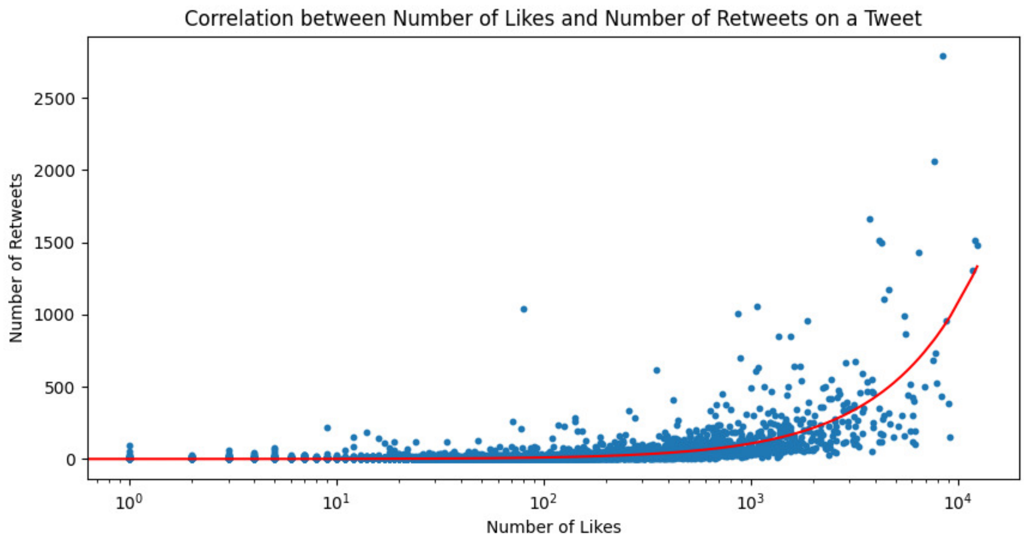


Figure 5 – Correlation between number of likes and retweets received by a tweet, with line of best fit

For likes, we were unable to find any research into appropriate weighting. By analysing a preliminary dataset of 10,000 tweets, we were able to find a strong positive correlation between the number of likes and the number of retweets received (illustrated in figure 5). The Pearson correlation coefficient of this was 0.75. This may justify disregarding likes since they are so closely correlated to retweets.

Retweets are a strong signal but do not necessarily indicate agreement, and likes are a weak signal that almost always indicate agreement. As such, an ensemble approach which considers both values together could be considered.

3.3.4 Prolific Tweeters

In an audience vote, each member of the audience will only be allowed to vote once. In the case of later seasons of *I'm a Celeb*, each member of the audience is given five votes on the app for free *(ITV. 2022)*, however in this situation most audiences will give all five votes to their favourite contestant anyway. As such, prolific tweeters who post many times in favour of one contestant should only have their opinion counted once. To achieve this, we can use a unique anonymous identifier for each Twitter user in the dataset and keep track of which users we have already counted in the model, disregarding any future tweets they make, or potentially including further tweets but with a much reduced weighting.

In the dataset, each tweet was stored with a hash of the user's Twitter handle as an anonymous identifier, so this would be possible. However, this information was not considered in the final model because it was not found to be helpful for gaining accuracy. It was found that counting just one tweet per user harmed accuracy, having reduced the dataset in size by 38%.

3.3.5 Window Size

When making a prediction, it was important to decide how many days' worth of tweets to feed into the model to acquire a result. If a prediction is made on the final day, it may be beneficial to use tweets from the past week to inform the model's decision. Alternatively, tweets from prior days may only contribute noise, making it beneficial to only feed tweets from the current day into the model. This was another parameter that needed to be fine-tuned through grid search during model development.

In the final model it was found that a window of three days struck the perfect balance between maximising data, and minimising noise from tweets which are no longer relevant to sentiment.

4 Outline of Approach

In order to answer our research questions, we proposed an approach that consists of the following steps:

1. Build dataset of significant size.
2. Build models.
    a. Baseline non-ML model.
    b. Final ML Twitter model.
    c. Betting exchange model (For comparing to the Twitter model).
3. Evaluate final ML Twitter model.

4.1 Dataset Creation

The dataset built for this project needed to contain as many tweets as possible with the intention of maximising coverage and accuracy of the final model. We aimed to acquire a large proportion of the total available tweets. Ideally coverage between seasons would be relatively even, and coverage between contestants would be representative of the number of times they had each been mentioned.

To ensure the dataset was manageable, the tweets were stored as CSV files, broken down into one file per day per season. This ensured that data could be replaced, viewed, and edited with relative independence to prevent large computing overheads.

This is explained in further detail in section 5.1.

4.2 Model Creation

In order to answer our two research questions, an initial base model was created which was iteratively developed upon. This base model gave us a baseline accuracy which needed to be beaten.

Next, our machine learning based model was developed, which was the bulk of this project, and the model which would be evaluated at the end. After each stage of iterative development, the model was re-evaluated using metrics defined in section 4.3 to determine the value of the newest changes.

This development methodology ensured that each change had a positive effect on accuracy, because evaluation was performed at regular intervals, allowing in depth comparison between iterations.

Finally, an alternative betting exchange based model was built, which makes predictions of voting share, using only information provided by betting exchanges. This could be used to determine whether we had met our goal of answering research question number two.

This method of model creation is explained in much further detail in section 5.

4.3 Evaluation Metrics

Throughout the iterative development of the project, evaluation needed to take place after each new version of the model to determine the degree to which performance has improved. These metrics were largely inspired by a paper from *(Brito, K. and Adeodato, P. 2023).*

1. **Mean Absolute Error** (MAE):

$$MAE = |pv_1 - av_1| + |pv_2 - av_2| + \cdots$$
$$where:$$
$$pv = predicted\ vote\ share$$
$$av = actual\ vote\ share$$

The predicted percentage of votes for each contestant are subtracted from the actual percentage of votes the contestant has received. This tells us how close our model is to predicting vote shares perfectly.

2. **Absolute Error on the Margin** (AEOM):

$$AEOM\ (per\ season) = \big||pv_w - pv_s| - |av_w - av_s|\big|$$
$$where:$$
$$pv = predicted\ vote\ share$$
$$av = actual\ vote\ share$$
$$w = winner$$
$$s = second\ place$$

This is the absolute value of the difference between the margin separating the two leading contestants in the predicted vote and the real vote. This metric allows us to determine if the model is correctly predicting the difference in vote share between contestants.

3. **Degree of Correct Order** (DCO):

$$DCO = \frac{1}{3} * cp$$
$$where:$$
$$cp = places\ predicted\ correctly$$

This metric represents the degree to which the three contestants in the final of each season have been placed in the correct order. After computing the predicted vote share, each contestant is assigned a predicted place of first, second, or third. This is compared against the positions attained by the contestants in the real vote by recording the number of correct positions as a fraction. Interestingly, for a single three-way vote this metric can only take one of three values; 0 (if no contestants have been placed correctly), $\frac{1}{3}$ (if one contestant has been placed correctly), or 1 (if all contestants have been placed correctly). $\frac{2}{3}$ is not a possible value because in a set of three it is impossible to assign just two positions correctly.

5 Methodology

5.1 Dataset

This section outlines the creation, expansion, and evaluation of the dataset which was used for both the baseline and final Twitter models.

5.1.1 Creation of the Dataset

The collection of data for constructing the dataset was done using the Twitter API, which has strict limitations imposed by Twitter. To avoid reaching rate limits, queries would need to be constructed in such a way to minimise the collection of irrelevant tweets. It was therefore very important that only tweets regarding the current contestants and the hashtag ‘imaceleb’ were collected.

14 JSON files were created manually containing all the relevant details for every season of *I’m a Celeb* that we would analyse (appendix item A). The contents of these files include:

- A list of all contestants in the season.
- A list of all nicknames associated with each contestant.
- A list of the dates each contestant was eliminated.
- The end date of the season.
- The air time of each episode.
- The ITV-published results of every vote throughout the season.

Next, a script was constructed to take the details for each season from the relevant JSON file and create queries to collect tweets from every day the season.

The script starts by specifying the date range of the query, then adds every nickname of every contestant still in the competition, concatenated with OR statements. This process is performed for the final six days of every season. The query concerning the last day of season 14 can be seen in figure 6.

```
(imaceleb) since:2014-12-07 until:2014-12-08 ((@carlfogarty)OR(carl)OR(fogarty)OR(foggy)OR(@JakeQui
ckenden)OR(jake)OR(quickenden)OR(@MsMelanieSykes)OR(melanie)OR(sykes)OR(mel)
```

Figure 6 – Query used to collect tweets from the final day of season 14

The tweets retrieved by each query are stored in a CSV file containing the following columns:

1. Username associated with the author of the tweet, hashed with XXH32. (This hash algorithm was chosen due to the relative speed of this algorithm relative to SHA and MD5 *(Collet, Y. 2021)*).
2. Datetime that the tweet was made.
3. Like-count of tweet.
4. Retweet-count of tweet.
5. Reply-count of tweet.
6. Quote-tweet-count of tweet.
7. Tweet source label (e.g., “Twitter for iPhone”)
8. Tweet contents

This process resulted in a dataset of 346,000 tweets which matched our very specific queries, thereby minimising the collection of irrelevant tweets.

Many tweets included in the dataset had been posted after the results of the vote had been revealed. Since we are trying to make a prediction, we only want tweets which occurred before the vote. To remove this bias, we deleted all tweets from the dataset made after 9:45PM, because contestant eliminations on the show were found to have always occurred after this time. This reduced the dataset in size to just 208,000 tweets.

After this dataset had been created, a script was built which used TweetNLP to find the positive, neutral, and negative sentiment probabilities for each tweet. This information was appended to the tweet in its respective CSV row. As such, the new columns present in the finalised dataset are as follows:

9. Probability of the tweet representing positive sentiment, as calculated by TweetNLP.
10. Probability of the tweet representing neutral sentiment, as calculated by TweetNLP.
11. Probability of the tweet representing negative sentiment, as calculated by TweetNLP.

The dataset was divided into two subsets. A development set was formed which includes data from all seasons 9-17. A held-out dataset was kept aside for final evaluation, which included all data from seasons 18-21. The development set needed to be larger than the held out set because of the small number of seasons available to work with.

5.1.2 Expansion of the Dataset

Later in development, we experimented with the window size of the model for a prediction made on the final day of the competition. We found that including data from all 6 days in our dataset gave the lowest level of inaccuracy and theorised that expanding the dataset to allow for even greater window sizes may be beneficial to the model.

In order to create this larger dataset, we used the same code from the creation of the initial dataset but used it to collect data for an additional 4 days. We then appended this data to the initial dataset to create our new expanded dataset.

The new dataset includes data from the last 10 days of each season. This new dataset contains 304,660 tweets, an increase of 46.5% over the previous version (After removal of post-vote tweets).

5.1.3 Evaluation of the Dataset

The dataset acquired for this project consists of 304,660 tweets, spread across 14 seasons, with data from the last 10 days per season. This may seem like a small dataset at first, but according to the Twitter API there are only 316,357 tweets in existence which match our requirements. This means our sample represents 96.3% of the population.

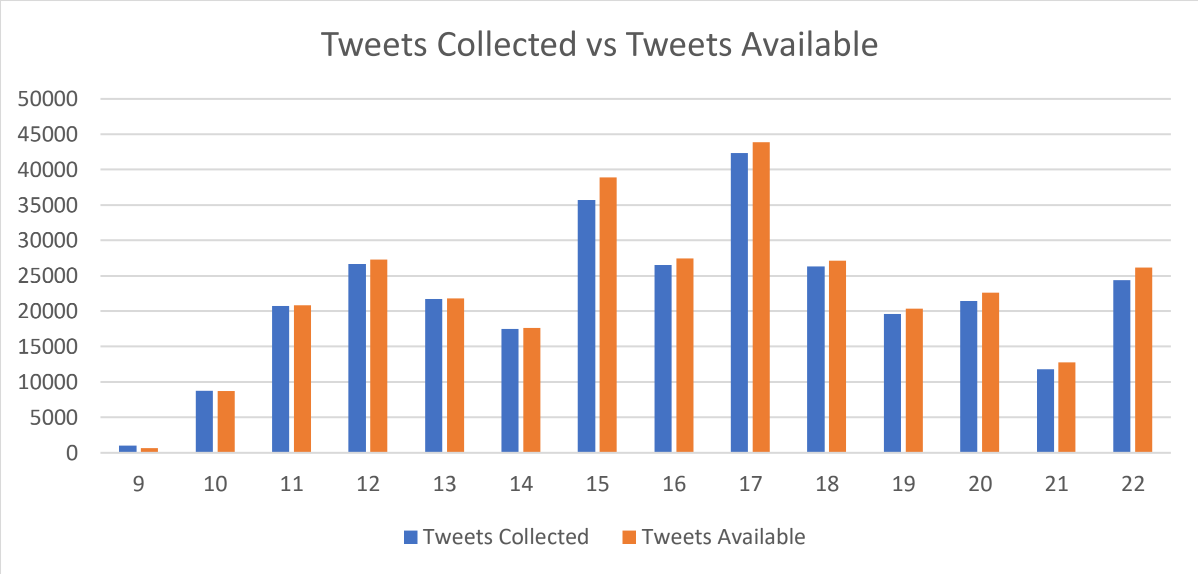


Figure 7 – Visualisation of the number of tweets available and collected for each season

As shown in figure 7, for every season the dataset contains almost every tweet that matches our query. Curiously, for seasons 9 and 10 we have managed to acquire more tweets than the Twitter API says are available. We believe there must be some inaccuracy in the Twitter API 'count' method, which was the tool used to find how many tweets were available.

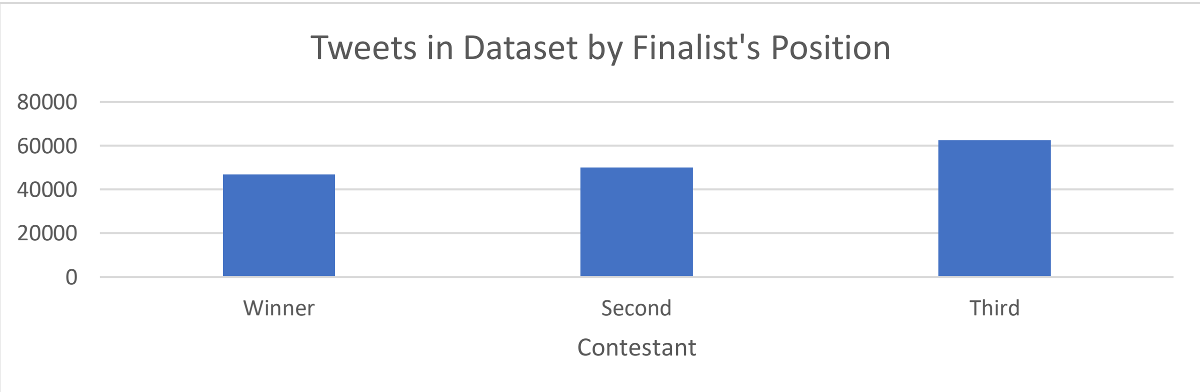


Figure 8 – Visualisation of the number of tweets in the dataset for first, second, and third place contestants

Figure 8 visualises the number of tweets in the dataset for each finalist, regardless of season. Interestingly, this chart shows that the winning contestants are actually tweeted about less than the second and third place contestants. From this we can infer that the volume of tweets concerning a contestant does not indicate their voting percentage. Therefore, more fine-grained methods such as sentiment analysis needed to be performed to establish a relationship between tweets and votes.

5.2 Baseline Model

A baseline non machine learning model was built in order to determine if a simplistic approach could be enough to obtain accurate predictions. It would also aid in learning about which features are most important before creating the final machine learning model.

5.2.1 Creation of Non Machine Learning Baseline Model

In this model, points are tallied up for each contestant, and these are used to make a prediction on voting share. To start with, a point is gained for every positive tweet mentioning the contestant. A positive tweet is defined as any tweet with a positive sentiment probability greater than a minimum predetermined 'confidence level'.

It was necessary to implement parameter tuning to decide a suitable confidence level for the model to filter by. Grid search was employed on the development dataset with intervals of 10 percent over the set $\{0, 1\}$. This grid search found the optimal confidence level to be 0.3. Grid search was then implemented with intervals of 1 percent over the set $\{0.20, 0.40\}$.

It was found that the most suitable confidence level was 35%. This was subject to change throughout the project because subsequent changes to the model called for the optimum confidence level to be re-revaluated.

After finding the number of points for each contestant, the predicted vote share is calculated like so:

$$PredVoteShare_{Contestant\ 1} = \frac{Points_{Contestant\ 1}}{Points_{Contestant\ 1} + Points_{Contestant\ 2} + Points_{Contestant\ 3}}$$

This model was altered to also subtract one from a contestant's points every time a negative tweet is found mentioning their name. Similarly, a negative tweet is any tweet with a negative sentiment probability greater than the confidence level. To find a suitable value for this new negative confidence level, both the positive and negative confidence levels were tuned using grid search with increments of 1%. With a positive confidence level of 18% and a negative confidence level of 95%, we were able to achieve a MAE of 9.63%.

The optimum negative confidence level of 95% stands to reason, because with this format of television programme the audience can only vote for a participant, not against. People expressing negative sentiment towards a candidate cannot directly act on this negative sentiment, other than by influencing the opinion of others.

Another experiment that was performed was in modifying the window size. Window size refers to the number of days' worth of data, prior to the final, that are inputted into the model to make a prediction. Theorising that tweets made a long time before the final may no longer be relevant when it comes to the audience vote, we used grid search on the window size, and found that all 6 days of the dataset is optimal.

Because the entire dataset's capacity of 6 days was found to be the optimal window size, the dataset was expanded to 10 days per season (as explained in section 5.1.2). To establish a new optimum window size, grid search was once again performed, this time ranging from 6-10 days. We found the optimal window size to be 7 days.

A problem with the code was found in the previous experiment. When searching for contestant's nicknames within tweets, the model was case-sensitive. As a result, contestants names were often not detected inside of tweets.

To remedy this, after our grid search experiments had been performed, the code was changed to convert all text into uppercase before comparison. Before this error was fixed, only 62,970 tweets could be found containing the names of each year's three finalists. After correction, 141,270 tweets could be found.

It was decided that weightings would be added to each tweet's value according to the retweets received by a tweet. Prior to this, each contestant gained one point for every positive tweet that was made about them. This was changed so that points were awarded according to the following formula (If positive sentiment probability > positive confidence level):

$$Points\ Gained = 1 + Weighting * Retweets$$

Initially we tried a weighting of 0.73, in accordance with research outlined in section 3.3.3, which indicated that a retweet may represent 73% agreement with the statement of the tweet. This resulted in a MAE of 9.10%, which was worse than the accuracy of the previous model. To find an improved weighting we used grid search. The model had the lowest inaccuracy when a weighting of just 0.20 was applied to each retweet. This model can be explained as follows:

$$\begin{aligned}
&FOR\ Tweet\ in\ Dataset\ IF\ Contestant\ Nickname\ is\ Found:\\
&\quad IF\ (Likelihood\ of\ Tweet\ being\ Positive) > 0.18: Points = Points + 1 + 0.2 * Retweets\\
&\quad IF\ (Likelihood\ of\ Tweet\ being\ Negative) > 0.95: Points = Points - 1
\end{aligned}$$

These points are converted into vote shares using the same algorithm as before:

$$PredVoteShare_{Contestant\ 1} = \frac{Points_{Contestant\ 1}}{Points_{Contestant\ 1} + Points_{Contestant\ 2} + Points_{Contestant\ 3}}$$

5.2.2 Evaluation of Non Machine Learning Baseline Model

To evaluate the model, the predicted voting percentage for each candidate was compared to their actual voting percentage revealed by ITV. One outlier was found, which is illustrated below:

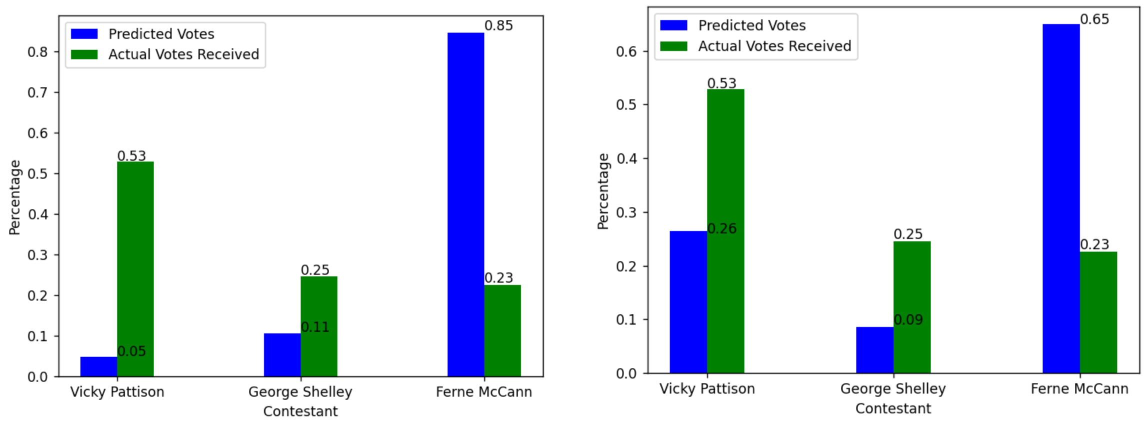


Figure 9 – Season 15 results. Left: Before Twitter handle correction. Right: After Twitter handle correction

For season 15 the model was incredibly inaccurate (Fig. 15, Left), predicting that Vicky Pattison would receive just 5% of the vote, when in fact she received a majority of 53%. Upon closer inspection one cause of this error was found. In the information JSON file for this season, Vicky Pattison's Twitter handle was recorded as @VickyPattison. At the time of the show, her handle was @VickyGShore. The model had been looking for an incorrect keyword to identify tweets concerning her. After updating the dataset, the model was more accurate but still largely incorrect for this season (Fig. 15, Right), predicting that Vicky Pattison would receive 26% of the vote.

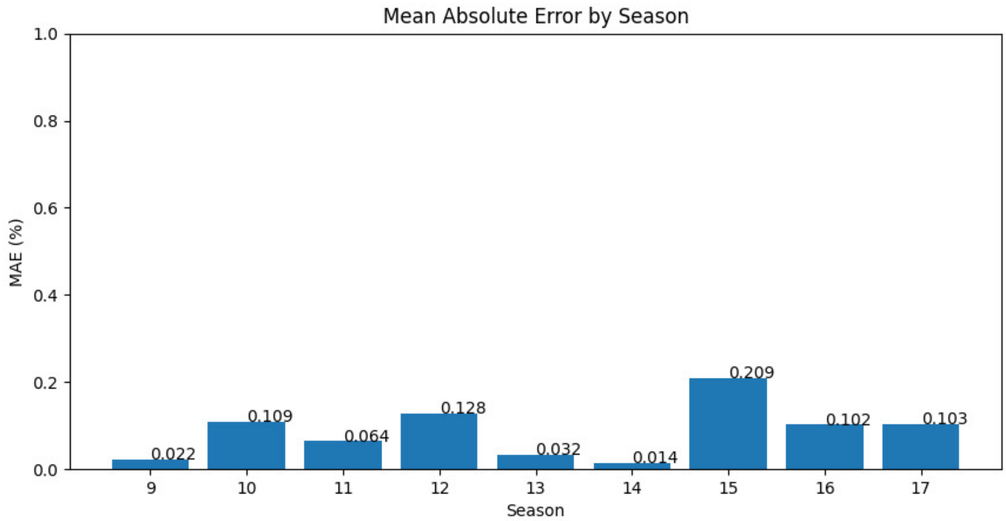


Figure 10 – MAE results for non-ML model by season

Figure 10 (above) demonstrates MAE by season for this baseline model. There is great variation in accuracy. For season 14 the model was incredibly close to the correct voting percentages with a MAE of just 1.4%, whereas for season 15 the model was completely wrong with a MAE of 20.9%. Overall, the baseline model has a MAE of 8.69%. This is a high but acceptable level of inaccuracy. The main problem with this model is not the average accuracy, but the lack of consistency.

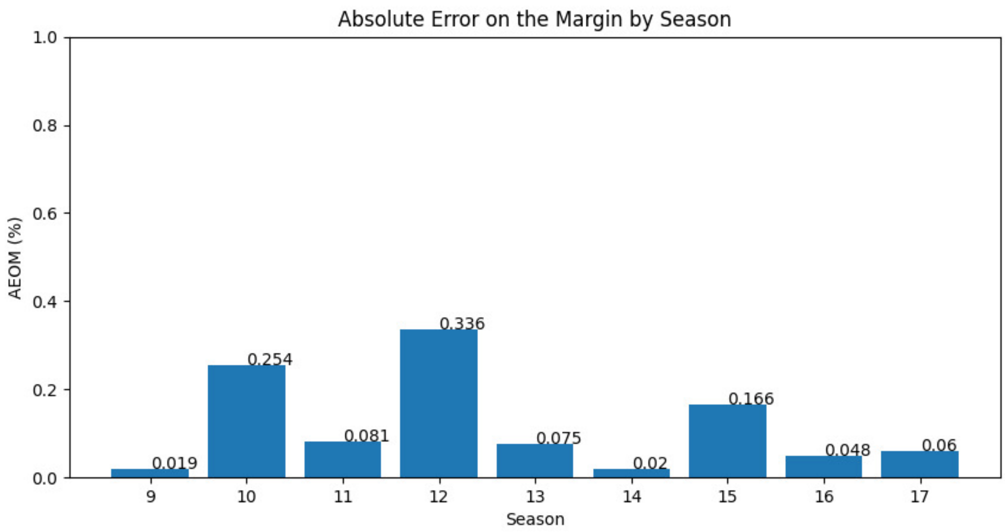


Figure 11 – AEOM results for non-ML model by season

Figure 11 shows AEOM for our baseline model. This figure demonstrates even greater variability, with an overall AEOM value of 11.78% for this baseline model. Season 12 has an AEOM of 33.6%, meaning the model predicted the margin between first and second place completely incorrectly. Many other seasons demonstrate similarly poor AEOM.

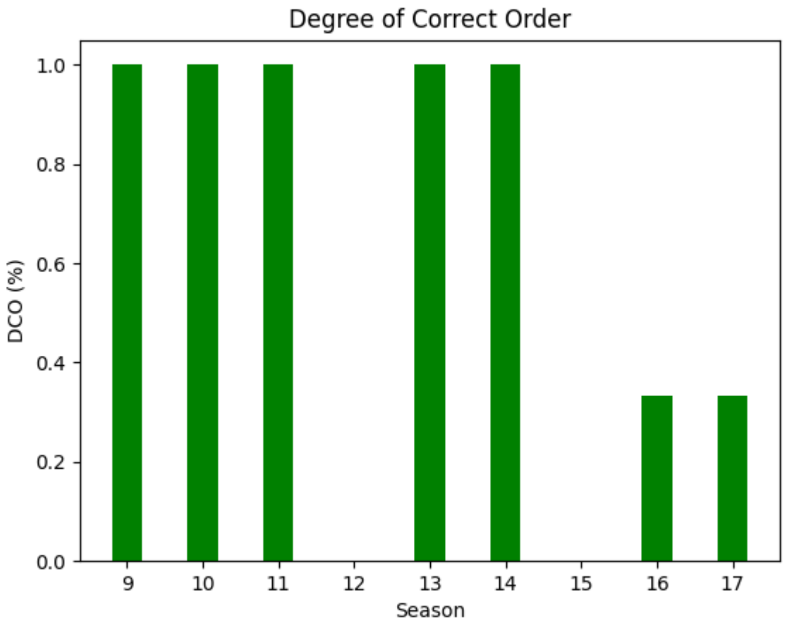


Figure 12 – DCO results for non-ML model by season

Finally, figure 12 shows the Degree of Correct Order by season. Surprisingly, this chart shows that for five seasons the model predicts the order of the three final contestants perfectly. For seasons 16 and 17, one contestant of the three was placed correctly, and for seasons 12 and 15 no contestants were placed correctly. Overall, the DCO for this model is 62.96%. Again, this model demonstrates decent overall accuracy, but high levels of inconsistency.

5.3 Machine Learning Twitter Model

This section outlines the creation, improvement, and initial evaluation of the main machine learning, Twitter-based model.

5.3.1 Creation of Sufficient Data Points by Employing 'Combinations'

The first issue presented by the use of a machine learning model was the extremely limited number of data points available with which the model could be trained. The model was intended to be trained using the final results from each season in a 9-season development set. This meant that for training and cross evaluation the model was limited to just $3\ Contestants * 9\ Seasons = 27\ Data\ Points$. This would not be enough for the training of a traditional machine learning model, so it was necessary to devise a method to extract more training points.

Daily results per celebrity[10]

	Day 13	Day 15	Day 16	Day 17	Day 19	Day 20	Day 21	Day 22		Trials
								Round 1	Round 2	
Vicky	Immune	1st 24.05%	1st 24.09%	1st 26.68%	1st 29.81%	1st 34.03%	1st 37.01%	1st 52.92%	Winner 70.17%	8
George	Immune	2nd 16.08%	2nd 17.01%	2nd 15.49%	3rd 13.93%	3rd 16.1%	3rd 16.42%	2nd 24.53%	Runner-up 29.83%	7
Ferne	3rd 13.33%	6th 6.89%	4th 9.02%	4th 9.47%	2nd 17.9%	2nd 16.63%	2nd 18.41%	3rd 22.55%	*Eliminated* (Day 22)	7
Kieron	4th 10.16%	7th 6.48%	6th 7.65%	7th 7.35%	5th 12.09%	4th 12.54%	4th 15.48%	*Eliminated* (Day 21)		8
Jorgie	2nd 19.17%	3rd 11.38%	3rd 11.68%	3rd 13.76%	4th 12.31%	5th 12.16%	5th 12.68%	*Eliminated* (Day 21)		8
Tony	7th 7.4%	5th 7.01%	5th 7.72%	6th 7.5%	6th 7.07%	6th 8.55%	*Eliminated* (Day 20)			5
Duncan	Immune	8th 6.32%	7th 7.34%	5th 9.42%	7th 6.89%	*Eliminated* (Day 19)				3
Lady C	1st 20.73%	4th 9.41%	8th 7.31%	8th 6.25%	*Withdrew* (Day 17)					8
Chris	8th 6.37%	9th 4.67%	9th 4.43%	9th 4.07%	*Eliminated* (Day 17)					3
Yvette	6th 8.28%	10th 4.36%	10th 3.75%	*Eliminated* (Day 16)						3
Brian	5th 8.97%	11th 3.34%	*Eliminated* (Day 15)							5
Susannah	9th 5.58%	*Eliminated* (Day 13)								3
Spencer	*Withdrew* (Day 5)									1

Figure 13 – Chart showing the results of every elimination and final from season 15

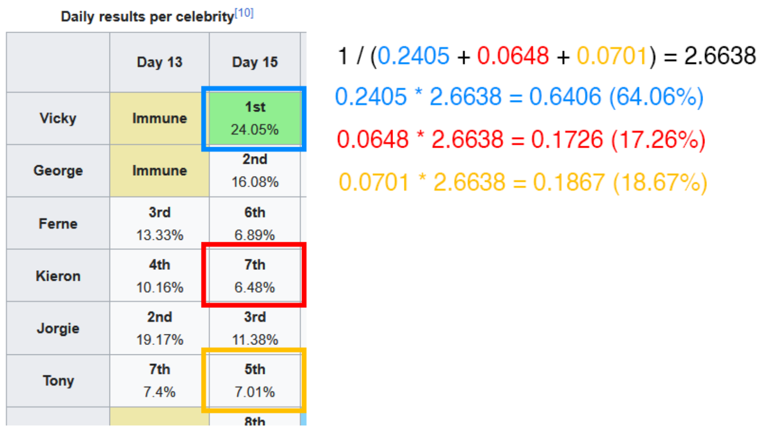
For seasons 8-18 of the show, ITV published the voting results for every elimination throughout the entire season (Fig. 13). We devised a method to maximise the number of datapoints using the results of all of these votes, which occurred almost daily.

The method we have devised uses every single combination of three contestants on any given day to create a large number of theoretical three-contestant votes which never actually occurred. Each of these three-contestant votes are, in essence, a simulation of a final vote in which only three contestants within the combination remain.

For example, on a day with four contestants A, B, C and D:

$$Contestants\ Remaining = \{A, B, C, D\}$$
$$Theoretical\ Final\ Votes = \{\{A, B, C\}, \{A, B, D\}, \{A, C, D\}, \{B, C, D\}\}$$

For day 15 of season 15 (Fig. 13), there are 11 remaining contestants. This means we are able to create $11C3 = 165$ theoretical final votes.


Daily results per celebrity[10]

	Day 13	Day 15
Vicky	Immune	1st 24.05%
George	Immune	2nd 16.08%
Ferne	3rd 13.33%	6th 6.89%
Kieron	4th 10.16%	7th 6.48%
Jorgie	2nd 19.17%	3rd 11.38%
Tony	7th 7.4%	5th 7.01%
		8th

1 / (0.2405 + 0.0648 + 0.0701) = 2.6638
0.2405 * 2.6638 = 0.6406 (64.06%)
0.0648 * 2.6638 = 0.1726 (17.26%)
0.0701 * 2.6638 = 0.1867 (18.67%)

Figure 14 – Demonstration of the construction of theoretical final vote results

For every possible combination, the contestant's voting results as part of the entire voting pool are normalised to represent their respective voting shares if these three were the only contestants remaining for that vote. These combinations can be used to replace the final votes when training the model, because each combination represents the results of its own three-contestant vote.

Figure 14 shows how these theoretical voting counts would be calculated for the combination $\{Vicky, Kieron, Tony\}$ on day 15 of season 15. The calculation you see in this example is performed for every possible combination of three contestants from every day of every season.

As can be seen in table 7, using this method we went from 42 total points $(14\ seasons * 3\ contestants)$ to 12,165 in the entire dataset. We went from 27 points to 11,211 in the development set. This made it far easier to train the model.

Season	# Combinations	# Data Points
9	297	891
10	504	1512
11	514	1542
12	350	1050
13	422	1266
14	204	612
15	519	1557
16	541	1623
17	386	1158
18	314	942
19	1	3
20	1	3
21	1	3
22	1	3
Total:	4055	12165

Table 7 – Data points created through the combination method

To evaluate and develop the model, we used every season from 9-17 as the training and validation set, through a season-wise cross validation process. For every season from 9-17 the model was trained using every combination of contestants from every other season and validated using just the final for that season.

For example, for season 9 we train the model with every combination from seasons 10-17 (a total of 10,320 data points) and validate the model with the final from season 9 (3 data points). This method was employed because it maximises the number of data points available to train and validate.

5.3.2 Feature Design

Features were added to the model iteratively. If a feature was found to yield an improvement upon evaluation, it was retained for the next iteration of the model. If not, it was discarded.

The machine learning model was built to use features which are found by scraping the dataset in search of each contestant's nicknames within the relevant CSV file. The features present in the finished model are as follows:

A. Number of positive tweets directed towards contestant X (As a percentage relative to the other two contestants in the vote)
B. Number of negative tweets directed towards contestant X (As a percentage relative to the other two contestants in the vote)
C. Number of positive tweets directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)
D. Number of negative tweets directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)
E. Average number of retweets on a positive tweet regarding contestant X
F. Average number of retweets on a negative tweet regarding contestant X
G. Number of "Win Indicators" found directed towards contestant X (As a percentage relative to the other two contestants in the vote)
H. Number of "Win Indicators" found directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)
I. Average number of replies on a positive tweet regarding contestant X
J. Average number of replies on a negative tweet regarding contestant X

We will now break down the design and implementation of each feature.

A and B – Numbers of Positive and Negative Tweets Directed Towards Contestant

The first two features implemented into the model were as follows:

A. Number of positive tweets directed towards contestant X (As a percentage relative to the other two contestants in the vote)
B. Number of negative tweets directed towards contestant X (As a percentage relative to the other two contestants in the vote)

Features A and B represent the percentage share of positive and negative tweets concerning each contestant in each combination as features. The 33% confidence level was used because previous experiments (section 5.2.1) found this was optimal when applied to both positive and negative probability equally.

These figures had to be represented as a percentage share rather than a whole number, because there is variation in the number of tweets we have acquired for each season. To maintain consistency in the features with which we are training the model, every feature we added would need to be normalised and represented as a proportion relative to the other two candidates in the combination, rather than a whole exact figure. For example, feature A is constructed like so:

$$Share\ of\ Positive\ Tweets_{Contestant\ 1} = \frac{\#\ Positive\ Tweets_{Contestant\ 1}}{\#\ Positive\ Tweets_{Contestant\ 1} + \#\ Positive\ Tweets_{Contestant\ 2} + \#\ Positive\ Tweets_{Contestant\ 3}}$$

C and D – Features Regarding the Next Best Contestant

C. Number of positive tweets directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)
D. Number of negative tweets directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)

Before the addition of these two features, the model was making a prediction based off the share of tweets directed towards each contestant, without knowledge of how votes are broken up between the other two contestants in the vote.

$$Contestant\ A\ Positive\ Tweet\ Share = 0.45$$
$$Contestant\ B\ Positive\ Tweet\ Share = 0.30$$
$$Contestant\ C\ Positive\ Tweet\ Share = 0.25$$

In the above example, 45% of the positive tweets directed towards these three candidates are directed towards contestant A. This contestant is likely preferred by the public because he has a higher share than the other two contestants.

$$Contestant\ A\ Positive\ Tweet\ Share = 0.45$$
$$Contestant\ B\ Positive\ Tweet\ Share = 0.50$$
$$Contestant\ C\ Positive\ Tweet\ Share = 0.05$$

In this new example above, contestant A still has 45% of the vote, but is not necessarily preferred because contestant B has 50% of the vote.

This demonstrates that knowledge of the share of positive sentiment is not enough information to determine whether a contestant is preferred, unless said contestant has >50% of the vote.

To provide our model with all necessary information, we need to provide information regarding the 'next best' contestant's sentiment as well. This was done by adding the positive and negative tweet share for the next best contestant as features inputted for every contestant into the model.

This means that for any contestant C1 as part of a combination, the features for this contestant will be as follows, where C1 is the contestant whose vote we are trying to predict, and C2 is the contestant with the highest percentage of positive tweets out of the other two contestants in the combination:

$$C1\ Features = \{\%Pos_{C1}, \%Neg_{C1}, \%Pos_{C2}, \%Neg_{C2}\}$$

As an example, the features for three made up contestants are seen below, constructed from information in table 8.

	Contestant 1	**Contestant 2**	**Contestant 3**
% Positive Tweets	0.2	0.25	0.55
% Negative Tweets	0.1	0.15	0.75

Table 8 – Data with which the feature set is created

$$Contestant\ 1\ Features = \{0.2, 0.1, 0.55, 0.75\}$$
$$Contestant\ 2\ Features = \{0.25, 0.15, 0.55, 0.75\}$$
$$Contestant\ 3\ Features = \{0.55, 0.75, 0.25, 0.15\}$$

This feature was added because it allows us to include contextual information about the stiffness of the competition for each candidate. We were already representing each candidate's statistics relative to the others, but now by including the statistics about the best competitor, the model can determine more effectively whether a candidate is likely to beat the competition.

E and F – The average number of retweets

E. Average number of retweets on a positive tweet regarding contestant X
F. Average number of retweets on a negative tweet regarding contestant X

We now wanted to add more features according to the attributes of the tweets we are already using, namely likes and retweets. In our dataset we have a likes and retweets column, telling us the number of times each of these actions have been applied to each tweet. It was theorised that information about the number of likes and retweets received by positive and negative tweets will give us information regarding how popular the sentiment expressed by that tweet is. For example, if a tweet saying "I am sick of Carl Fogarty" receives 500 retweets, this indicates that the sentiment is shared by other Twitter users.

We experimented with the addition of features regarding likes as well, but we found that for both positive and negative tweets, the average number of likes did not help to improve fidelity of the model, so only features regarding retweets were added.

It is important to note that features E and F represent the *average* number of retweets, not the proportion relative to the other contestants. For example, for feature E the value is calculated like so:

$$\frac{Total\ Number\ of\ Retweets\ on\ Positive\ Tweets\ Regarding\ C1}{Total\ Number\ of\ Positive\ Tweets\ Regarding\ C1}$$

G and H – Presence of Win Indicators

G. Number of "Win Indicators" found directed towards contestant X (As a percentage relative to the other two contestants in the vote)
H. Number of "Win Indicators" found directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)

When looking through the dataset, it was noticed that there are some words which, when mentioned alongside the name of a contestant, strongly indicate a desire for that contestant to win. These words are as follows:

$$\{\text{King, Queen, Vote, Win, Winner, First}\}$$

To use this data in the model, we implemented a basic algorithm which searches a given tweet and checks whether any of these words are mentioned. This analysis is run every time a contestant's name is found within a tweet. We call the presence of one of these words a "Win Indicator". For every contestant in each combination, the ratio of win indicators between all three contestants is recorded and passed to the regression model as a feature.

We also attempted to implement the same system but for "Loss Indicators", with the following terms:

$$\{\text{Bottom, Lose, Last, Eliminate, Kick}\}$$

However, we found the presence of these loss indicators to not correlate with percentage of the votes, showing no change in model accuracy when implemented. As previously mentioned, this makes sense because members of the public who express strong distain for a contestant are not able to vote against them, whereas members of the public with strong approval of a contestant are able to vote in favour. Because of this we left loss indicators out of the model and only implemented our win indicators feature.

Initial experiments showed that this feature significantly improved the model's performance, so we added another feature regarding win indicators. This feature is the share of win indicators found for the best competitor to the contestant in question. It was calculated similarly to features C and D.

I and J – Replies to Tweets

I. Average number of replies on a positive tweet regarding contestant X
J. Average number of replies on a negative tweet regarding contestant X

Finally, theorising that the average number of replies may also help to inform us of spreading sentiment, these two features were implemented and added into the inputs of the model. Again, these features are averages, not proportional to the other contestants.

5.3.3 Other Improvements

Although the main focus throughout development was the list of features inputted into our regression model, other changes had to be implemented along the way that did not involve features.

Speed-of-Execution Overhaul

Partway through development, our machine learning model had swollen considerably in complexity and was taking 57 minutes to run on average. This needed to be tackled because this wait time between test runs was seriously inhibiting development, forcing long wait times for every step of cross evaluation.

We identified one area where the program was seriously inefficient and large gains in speed could be achieved. Recall from earlier that for every season of the program we are now constructing hundreds of data points to train the model with.

To construct each of these unique data points, the program loops through every combination and scrapes the dataset to acquire the information needed to construct features for training. This involves looping through every tweet in the dataset and searching for names, phrases, and sentiment scores. This data is unique to contestants, not data points. After this information has been acquired it is normalised into fractions unique to each point, representing the contestant's share of positive and negative sentiment inside of the unique combination. As such, this is a lengthy and time-consuming process.

To avoid unnecessary repetition within data acquisition stage, the program was modified to scrape the dataset once per contestant, rather than once per data point. The information acquired is stored in a bank of data for each season, containing the information for each contestant. This is then normalised after these contestants are grouped into combinations, before being fed into the model for training. For example, for season 15 day 1, the model now scrapes the dataset once for each of the 12 contestants, where previously it was scraping the dataset once for each of the 1557 data points.

This massively reduced execution times from an average of 57 minutes to just 4 minutes.

Parameter Tuning

During development two different parameters needed to be tuned to maximise performance.

The first was the window size. Throughout most of development the model used a window size of 7 days, as this was found to be optimal for our non-machine learning model. This was later checked using grid search. Through grid search we found a new optimal window size of just three days, which caused a significant improvement to MAE and AEOM.

The next was the positive and negative confidence levels. Previously a confidence level of 0.33 was used for both positive and negative, because this was found to be suitable for the non-ML model. These were both tuned with grid search to find new optimum values of 0.3 and 0.35, respectively.

Final Result Normalisation

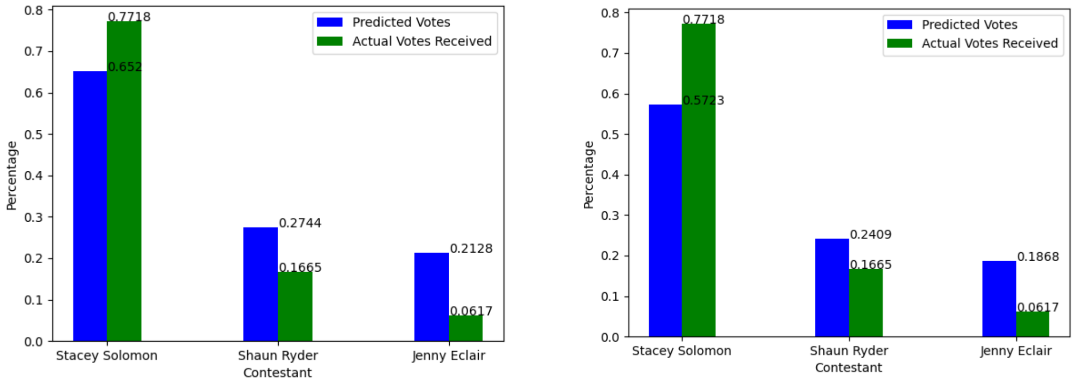


Figure 15 – Results for season 10. Left: Before output normalisation. Right: After output normalisation

The next change made to the model was the addition of a normalisation function. As is shown in figure 15 (left), the model was found to sometimes predict that the overall share of the vote will exceed 100%. In this example, the votes equal $0.652 + 0.2744 + 0.2128 = 1.1392$ or 113.92%. This is because the regression model predicts each contestant's share of the vote independently of the others, so it is not able to ensure that they total 100%.

To remedy this, we added a normalisation function which multiplies every predicted vote by a normalisation factor to ensure that the total always equals 100%. As you can see in the next figure (Figure 15, right), with the normalisation function the total of all predicted votes for a final now sums to 100% as expected.

Experimentation with Model Type

The final step in optimising our model was to experiment with different regression models. So far throughout the project we had used Support Vector Regression, with the intention of finding other more optimal solutions later on.

Regression Model	Mean Absolute Error (Seasons 11-17) (%)
Bayesian Ridge	7.50
Quantile	10.84
Kernel Ridge	7.74
Stochastic Gradient Descent	6.36
Nearest Neighbours	8.74
PLS	6.317
Decision Trees	8.99
SVR	**4.64**

Table 9 – MAE results for all regression models tested

We tested a large number of different regression models to replace SVR. As can be seen in table 9, the regression model we originally chose is superior to all the other models we tested, and so was retained in the finished model.

5.3.4 Model Summary

To aid in understanding, included below are two flowcharts (Figure 16) which outline the steps the model takes in order to train, and reach a final output for a season.

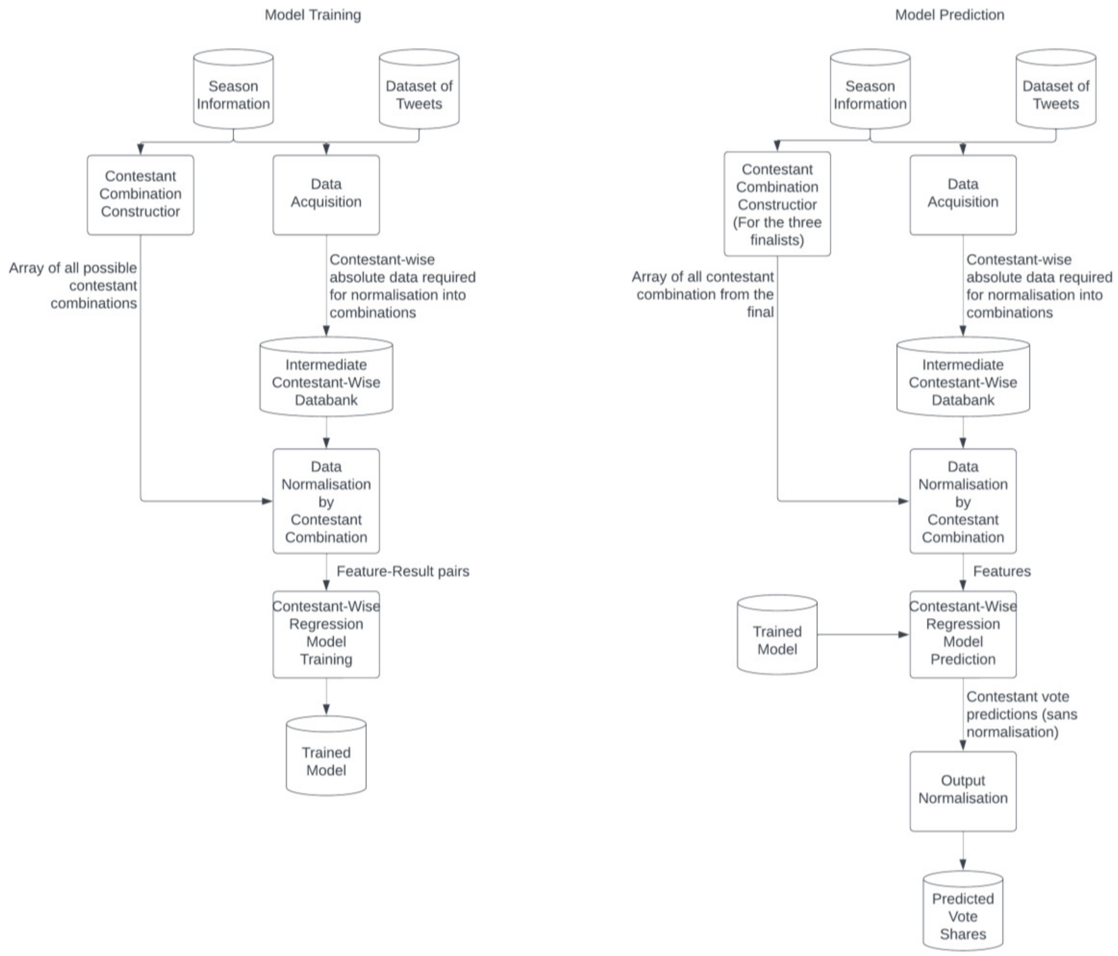


Figure 16 – Flow charts for model training and prediction steps

The first steps of training and prediction are the same: Season information files are used by the data acquisition function to collect data regarding each contestant from the dataset of tweets. These are stored in a databank to prevent unnecessary recollection of data. The contestant combination constructor takes data from the season information file and constructs all necessary/possible combinations. The data normalisation step takes the combinations and constructs the features necessary for the model, for every contestant, using data from the databank.

For training, this data, along with the contestant vote shares, is passed to the model training step, and a finalised model is produced.

For predictions, the normalised features are passed to the model without contestant vote shares. The model predicts the vote shares of every contestant in each combination. These vote shares are then normalised by the output normalisation step, before finally being stored or presented.

5.3.5 Feature Analysis

To analyse the improvements made throughout the development of this model, an ablation study was conducted.

Improvement Summary	MAE	AEOM	DCO
Initial Model, Features A and B	9.56	17.09	66.66
Features C and D added, window size adjusted to 3 days	8.49	14.31	59.24
Features E and F added	7.63	13.19	59.24
Feature G added	6.86	10.68	59.24
Features H, I and J added, vote normalisation implemented	5.54	8.03	70.37

Table 10 – Ablation study of features and accuracy-related improvements

Table 10 shows the results of this ablation study. This study was performed by collecting information recorded throughout development, rather than removal of features in post. Each row of improvements summarises one iteration of development. As can be seen, every step of development helped to improve the model's MAE by at least 0.77%. AEOM was reduced drastically between iterations, with a minimum improvement of 1.12%.

The row in which feature G was added is important, because it was a relatively primitively constructed feature. This was the 'win indicators' feature, which simply searched for words implying a victory within each tweet and recorded the number of occurrences.

The improvement of 2.65% to AEOM with the last iteration of development can be attributed to our output normalisation. Only added in this last step, the output normalisation system ensures that all contestant vote shares total 100%. Recall that AEOM represents the difference in the margin between the predicted first place margin, and the actual first place margin of victory. AEOM was affected so drastically because the differences between contestants were being partly corrected by removing the total error of the three candidates together.

Throughout development DCO stayed roughly the same, before making a large improvement with the last step of development. This cannot be attributed to normalisation because normalisation does not change the order of candidates. Instead, features H-J are what caused this jump.

5.3.6 Model Evaluation Against Baseline

To evaluate this model and compare it to the baseline non-ML model built earlier, season-wise cross evaluation was employed.

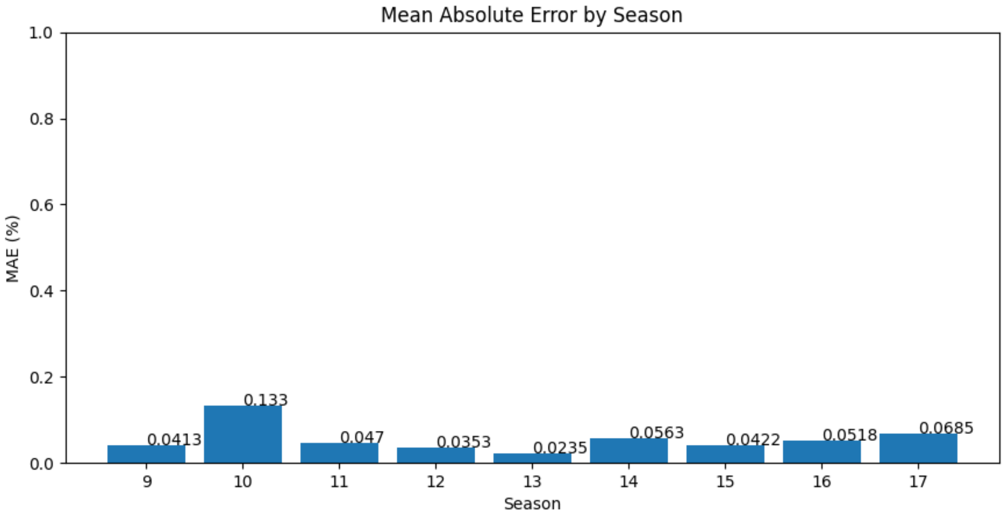


Figure 17 – MAE for Twitter ML model by season

Figure 17 shows MAE for our finalised machine learning model. The development of a machine learning model as outlined has reduced MAE across the entire development set by 3.15% from 8.69% to just 5.54%. If we only consider seasons 11-17 (as seasons 9-10 are lacking in data due to a lack of Twitter usage in these years), MAE now sits at just 4.64%, down from 9.31% with the non-ML model.

Furthermore, the MAE is far more consistent than it was previously, with only season 10 reaching a MAE above double digits.

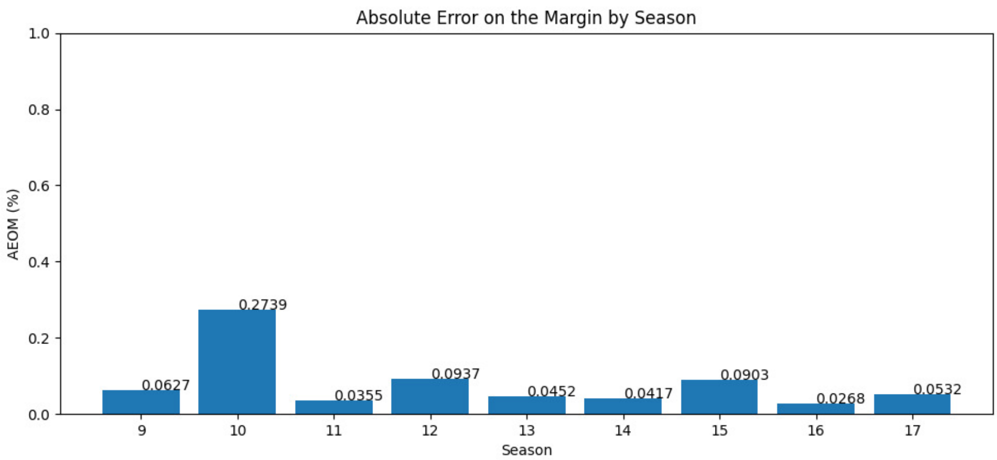


Figure 18 – AEOM for Twitter ML model by season

Figure 18 shows AEOM for our new model. This graph also shows a drastic improvement of 3.75% from 11.78% to just 8.03%. Again, it is far more consistent, with the only outlier being season 10. We have investigated but have been unable to find the cause of this outlier. For seasons 11-17, AEOM now sits at 5.52%, down from 7.86%.

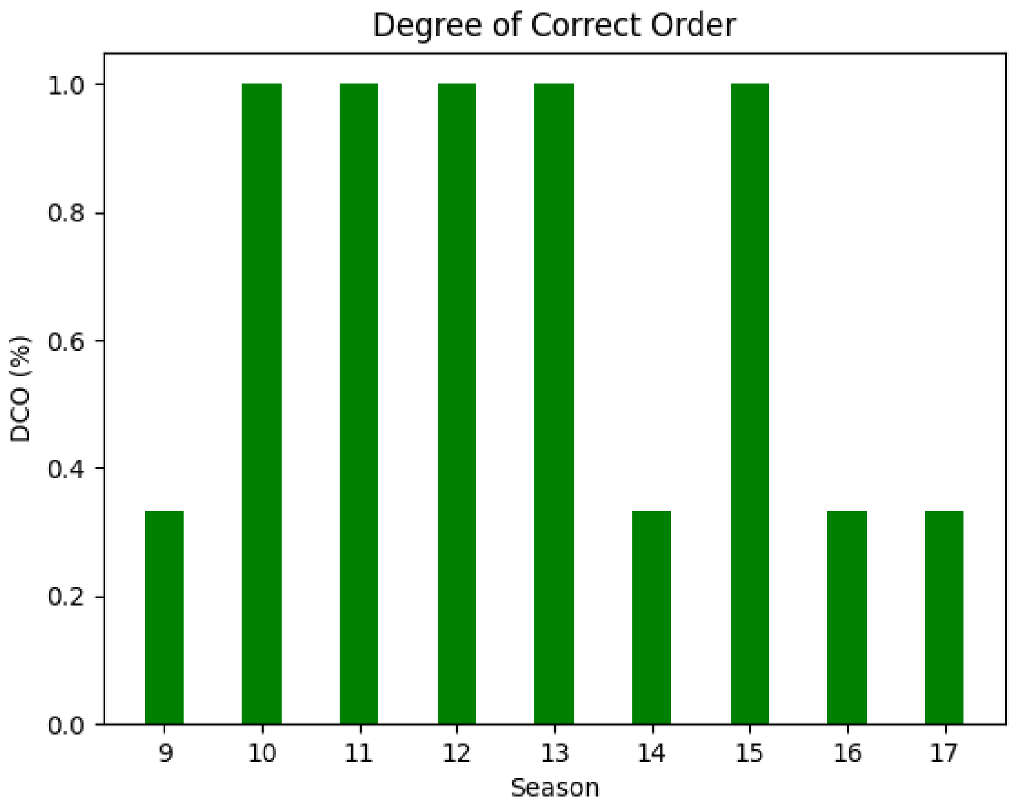


Figure 19 – DCO for Twitter ML model by season

Finally, DCO has improved as well. It has increased from 62.96% to 70.37%. The model still only predicts places perfectly 5 out of 9 times, but for all other seasons at least one contestant has been placed correctly.

Finally, the first place contestant is now correctly placed 7/9 times, making for an accuracy for first place prediction of 77.78%. The previous non-ML model placed 6/9 first place contestants correctly, for an accuracy of 66.67%.

One interesting point is that this new model is placing season 15 contestants correctly, with MAE of just 4.42%. This is important because the non-ML model failed completely with this season, placing all contestants incorrectly and achieving a MAE of 20.9%. This inspires confidence in our model because it means some correlations have been found which can overcome drastic outliers in the ratio of positive and negative tweets.

5.4 Betting Exchange Machine Learning Model

Recall that research question 2 aimed to build a model with accuracy close to that of betting exchanges. In order to answer this research question, it was necessary to create data regarding the accuracy of a betting exchange's odds.

5.4.1 Creation of the Betting Exchange Machine Learning Model

Odds presented on a betting exchange represent the betting population's prediction of who is most likely to win. Using betting odds, we can calculate the likelihood of a contestant winning, according to the betting population. This is called the implied probability *(Sohail, S. 2023),* and it is calculated as follows:

$$Implied\ Probability = \frac{1}{Decimal\ Odds} * 100$$

Implied probability represents the likelihood of a contestant winning but does not represent their predicted voting share. In order to calculate this, we once again needed to make use of a regression model.

Thankfully, many functions from the previous model could be reused, with only the input features to the regression model needing changing. All of the previous 10 features were replaced with just a single feature, which was the betting exchange's odds of that contestant winning. These odds were acquired from Betfair's 'Historical Data Service' *(Betfair, 2018)* and stored in the season information JSON files.

5.4.2 Model Summary

A flowchart can be seen in figure 20 below, which summarises the process undertaken by our Betfair model for both training and output.

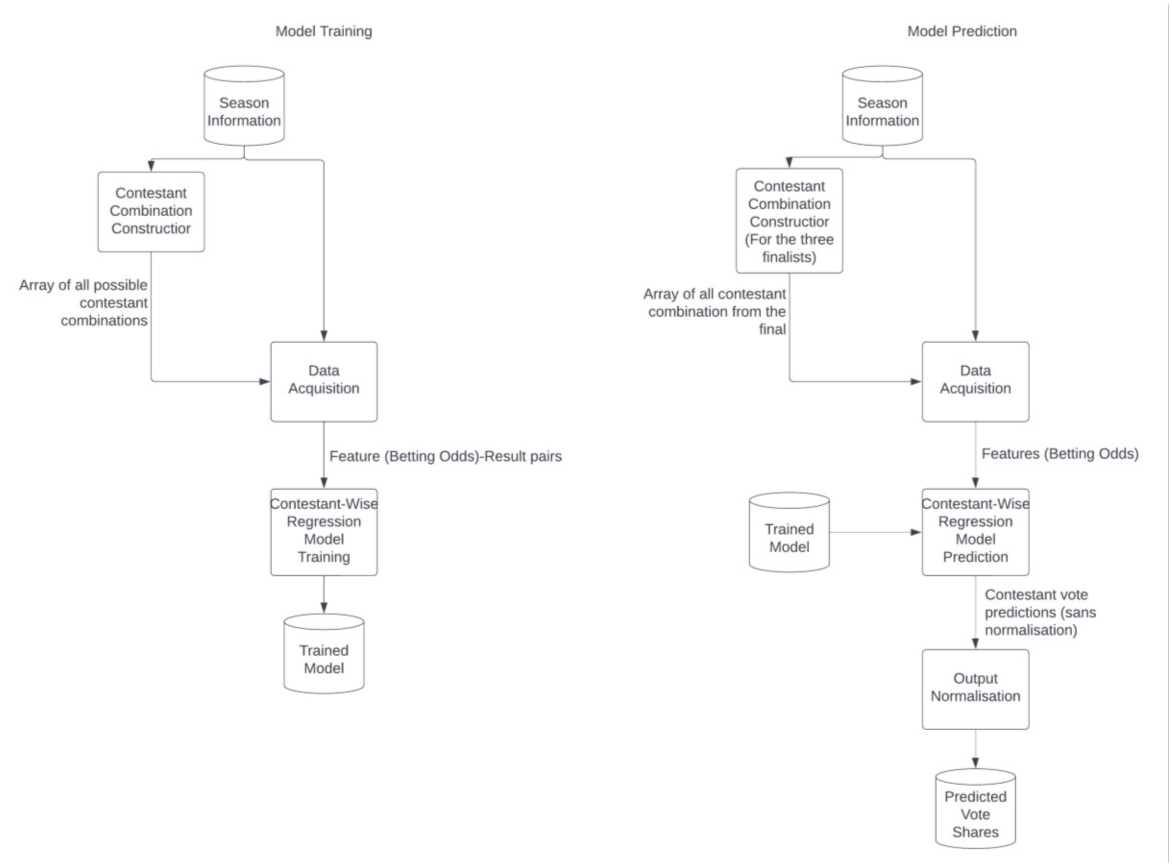


Figure 20 – Flowchart for Betting Exchange ML model for training and prediction

This flow chart shows how many functions from the previous model could be used, and how many functions could be removed entirely as they were no longer relevant.

5.4.3 Model Evaluation

We do not have any data for the model's performance on our development set, because the Betfair historical data portal only provides data for seasons 18-21. Instead, this model was developed using our held-out dataset.

Please note that this has not caused any overfitting, because this model was not iteratively developed to improve accuracy, it was simply implemented, and the results were accepted. This is because the purpose of this model was not to create accurate predictions using betting exchange data, but rather to simply map implied probability to voting share.

As such, there are no season 9-17 graphs to be presented here, and the results are presented in section 6.3 instead.

6 Final Evaluation

In this section we will be evaluating our model to determine whether we have met our goals. Recall from earlier that our goals were to refute the following two null hypotheses:

H1'. It is impossible to design and implement a model capable of predicting (nowcasting) the winner of *I'm a Celeb* on the day of the final to an accuracy of 80%.

H2'. It is impossible to design and implement a model capable of predicting (nowcasting) the winner and voting share of *I'm a Celeb* on the day of the final with accuracy close to or exceeding the implied probability presented by betting exchanges.

For our final evaluation we will be testing our finished regression model when applied to a held out dataset consisting of data from seasons 18-21 of *I'm a Celeb*.

6.1 Evaluation against Guessing

Before determining whether our model had beat the standards set out in both of our null hypothesis, we tested our model against the most basic possible prediction method, guessing without any contextual knowledge.

In a situation where you must guess the voting share of three contestants in a vote, the best way to minimise your error would be to guess that each contestant received exactly one third of the overall vote.

We found that a model which guesses 33.33% for every contestant achieved a MAE of 14.53%. This was 8.7% worse than our machine learning model, which achieved a MAE of 5.83% across our entire held-out set. Next, the guessing model achieved an AEOM of 28.67%, which was 20.75% worse than the ML model. Finally, the DCO was 33%, as every contestant was predicted to come in a three way tie for first place. This was far worse than the ML model, which achieved a DCO of 58.33%.

This means that our model was much better than the very low benchmark of guessing. This does, however, give some important context to each of our evaluation metrics, because the values achieved by our guessing model can be considered to be the lowest that are reasonably possible.

6.2 Null Hypothesis 1

When applied to the held out dataset, our results in terms of correct place predictions are summarised in table 11.

Season	First Place Correct?	Second Place Correct?	Third Place Correct?
18	Yes	No	No
19	No	No	Yes
20	Yes	Yes	Yes
21	Yes	Yes	Yes

Table 11 – Contestant position accuracy by for Twitter ML model by season

From this table we can see that for three out of four seasons, the first place contestant was predicted successfully. This makes for a correct-winner rate of 75%. This may seem shy of 80%, however we have only tested our model against the four seasons in our held-out set.

This means our correct-winner rate could only be one of the following values: $\{0.25, 0.5, 0.75, 1.0\}$. As such, a correct-winner rate of 75% can be considered to successfully refute our first null hypothesis.

6.3 Null Hypothesis 2

For our second null hypothesis, we attempted to build a model capable of predicting (nowcasting) the winner and voting share of *I'm a Celeb* on the day of the final with accuracy close to or exceeding the implied probability presented by betting exchanges.

To determine whether this hypothesis was successfully refuted, we will compare each of our metrics (MAE, AEOM, DCO) for both our final Twitter regression model, and our Betfair regression model.

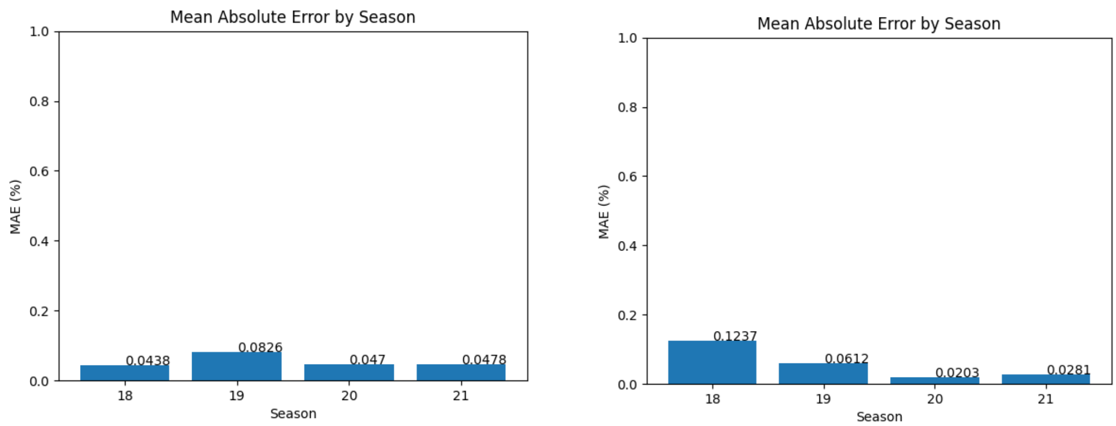


Figure 21 – MAE by season, Left: Betting exchange ML model, Right: Twitter ML model

For MAE, we need to compare the two graphs in figure 21. The results of our Betfair model can be seen on the left, and the Twitter regression model on the right. Overall, the Betfair model makes for more consistent results, with far less variation than our model. However, overall, the total MAE value is only slightly lower than our model (5.54% vs 5.83%).

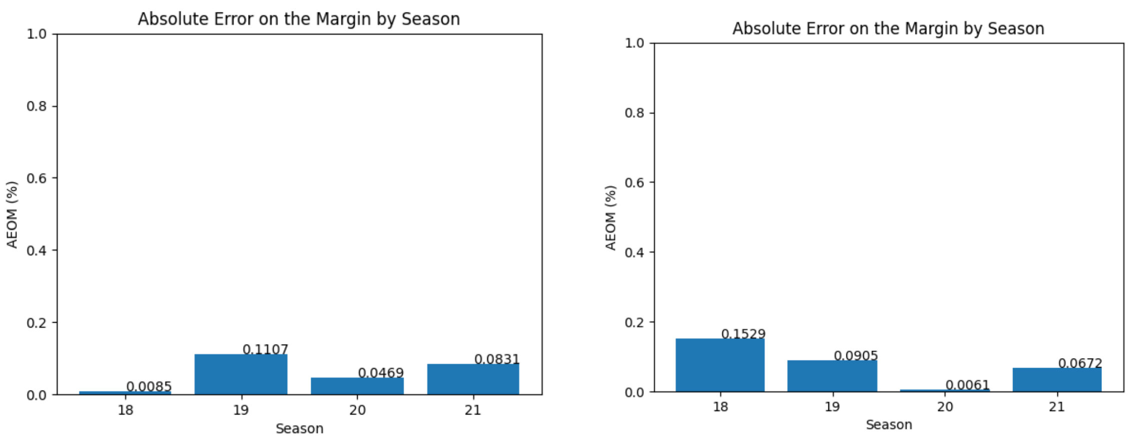


Figure 22 – AEOM by season. Left: Betting exchange ML model, Right: Twitter ML model

Figure 22 shows our results for AEOM. Again, left are the results of the Betfair model, and right are the results of the final Twitter regression model. Both models have a similar level of variation when it comes to AEOM, but the Betfair model still wins out with an overall AEOM value of 6.23%, versus the Twitter model's AEOM of 7.92%. For this metric, the Betfair model has beaten the Twitter model by a significant margin, but the Twitter model has still achieved a good result.

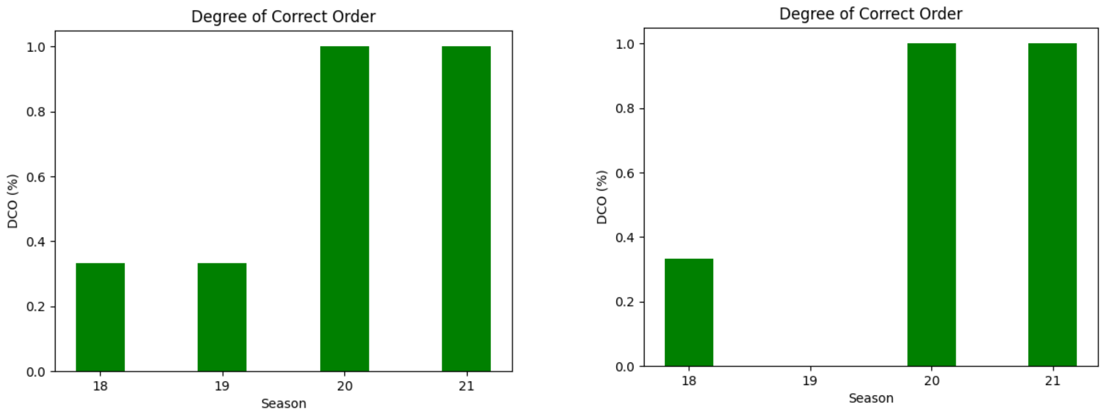


Figure 23 – DCO by season. Left: Betting exchange ML model, Right: Twitter ML model

Figure 23 shows the DCO for both models. On the left is Betfair, and the right is Twitter. Our Twitter regression model is close in this metric, with a DCO of 58.33% against the Betfair model's DCO of 66.65%. While this may seem like a considerable delta between the two models, this difference in DCO is attributed to just one difference in their predictions. For season 19, the Betfair model predicted third place correctly, and the Twitter model did not. The overall results can be seen in table 12, where we have marked in bold the only difference between the two models.

	Betfair Model Results			**Twitter Model Results**		
Season	P1	P2	P3	P1	P2	P3
18	Correct	False	False	Correct	False	False
19	False	False	**Correct**	False	False	**False**
20	Correct	Correct	Correct	Correct	Correct	Correct
21	Correct	Correct	Correct	Correct	Correct	Correct

Table 12 – Contestant position accuracy by season for betting exchange ML model and Twitter ML model

This shows that while there may be a difference in MAE, and a considerable difference in AEOM, overall, the models are incredibly close in their predictions, and in fact much closer than we were expecting.

Finally, using the above table we need to compare the correct-winner rate between the two models. The correct-winner rate is the same for both models, and in fact so is the correct-runner-up rate as well.

Therefore, we can conclude that our model is capable of nowcasting the winner with an accuracy matching that of the odds presented by betting exchanges, and our model is capable of nowcasting the vote share with an accuracy close to that of the betting exchanges. As such, we have also disproven our second null hypothesis successfully.

7 Conclusions and Future Work

7.1 Conclusions

Recall that our research questions outlined at the beginning of this project were as follows:

RQ1. Is it possible to design and implement a model capable of predicting (nowcasting) the winner of *I'm a Celeb* on the day of the final to an accuracy of 80%?

RQ2. Is it possible to design and implement a model capable of predicting (nowcasting) the winner and voting share of *I'm a Celeb* on the day of the final with accuracy close to or exceeding the implied probability presented by betting exchanges?

RQ3. Is it possible to build a dataset containing at least three thousand tweets for each episode of the competition, spanning all seasons and episodes that are available?

Through evaluation of our final model, we have found that it was possible to reach an accuracy of 80% in predicting the winner, thereby answering research question 1. The Twitter model succeeded in providing sufficiently accurate voting share predictions, that the first place contestant was predicted successfully three times out of four.

We also found that it was possible to predict the voting share with accuracy close to that of betting exchanges. The accuracy of this model is approaching the accuracy of our Betfair-informed comparison model, despite the use of independent variables (features) that are less directly correlated to voting share. This is important because as discussed earlier, betting exchanges (of which Betfair is the largest), are widely considered to be the most accurate systems for predicting the outcome of events like *I'm a Celeb*.

Finally, a dataset averaging far more than three thousand tweets per day was successfully built, answering research question three as well. This dataset spanned 14 seasons, which was the most available with significant Twitter data.

In summary, the Twitter model and our development was very successful, as we have successfully answered all three of our research questions. As outlined in section 5.3.5, every iteration of development throughout this project brough accuracy gains, as a result of thoroughly evaluated iterative development. Most of our assumptions about the problem were correct, and speedbumps such as limited data points were swiftly and effectively overcome.

7.2 Future Work

Throughout development a large number of avenues of development were theorised, many of which did not make it into the final model. The decision to not implement each of the following changes may be for a variety of reasons, but the main limitation was time constraints.

7.2.1 GPT3/3.5/4 as a Sentiment Analysis Model

As discussed earlier, at the start of this project TweetNLP was chosen as the sentiment analysis model which would be used to classify tweets as positive or negative. Later in development, it rose to our attention that GPT3 or GPT4 could be used to replace TweetNLP and potentially gain accuracy in sentiment classification.

A small experiment was performed to determine whether this may be the case. The experiment from section 3.2 was repeated, with the same sample of tweets. As a reminder, we will outline this experiment briefly.

Each of our three candidate sentiment analysis models (TextBlob, TweetNLP, GPT3) were applied to a sample of 100 tweets from our dataset. Each tweet was then classified as positive, negative, or neutral by a researcher. GPT3's results were approximated by using ChatGPT instead, an online chatbot which is built on GPT3.5. This was done because GPT3 itself requires an API access key. Finally, the output for each model was compared with the recordings made by the researcher to find precision, recall, and F1.

Sentiment Processor	Precision	Recall	F1
TweetNLP	**0.9344**	0.6628	0.7755
TextBlob	0.8548	0.6386	0.7311
ChatGPT	0.88	**0.8571**	**0.8684**

Table 13 – Precision, recall and F1 for TweetNLP, TextBlob, and ChatGPT applied to positive tweets

Table 4 shows the results of this analysis. As can be seen, ChatGPT performs worse in terms of precision, as a result of a higher occurrence of false positives. This means that of all positive classifications *made by the model*, less are correct. However, recall is drastically improved, meaning of all the positive tweets *in the dataset*, more are classified correctly.

Another benefit of GPT3 may be the potential for more accurate classification of sentiment between different contestants mentioned within the same tweet. Recall that we considered potential routes to combat this issue in section 3.3.2, including limiting our dataset to just tweets mentioning one contestant. In the final model this issue was not tackled. An example of this type of tweet can be seen in figure 24, below.

```
I actually wish it was Edwina is the final and not Mel #ImACeleb
```

Figure 24 - Example of a real tweet expressing differing sentiment towards multiple contestants

We tested ChatGPT to determine whether it was capable of breaking down the above tweet and determining differing sentiment within the same body of text.

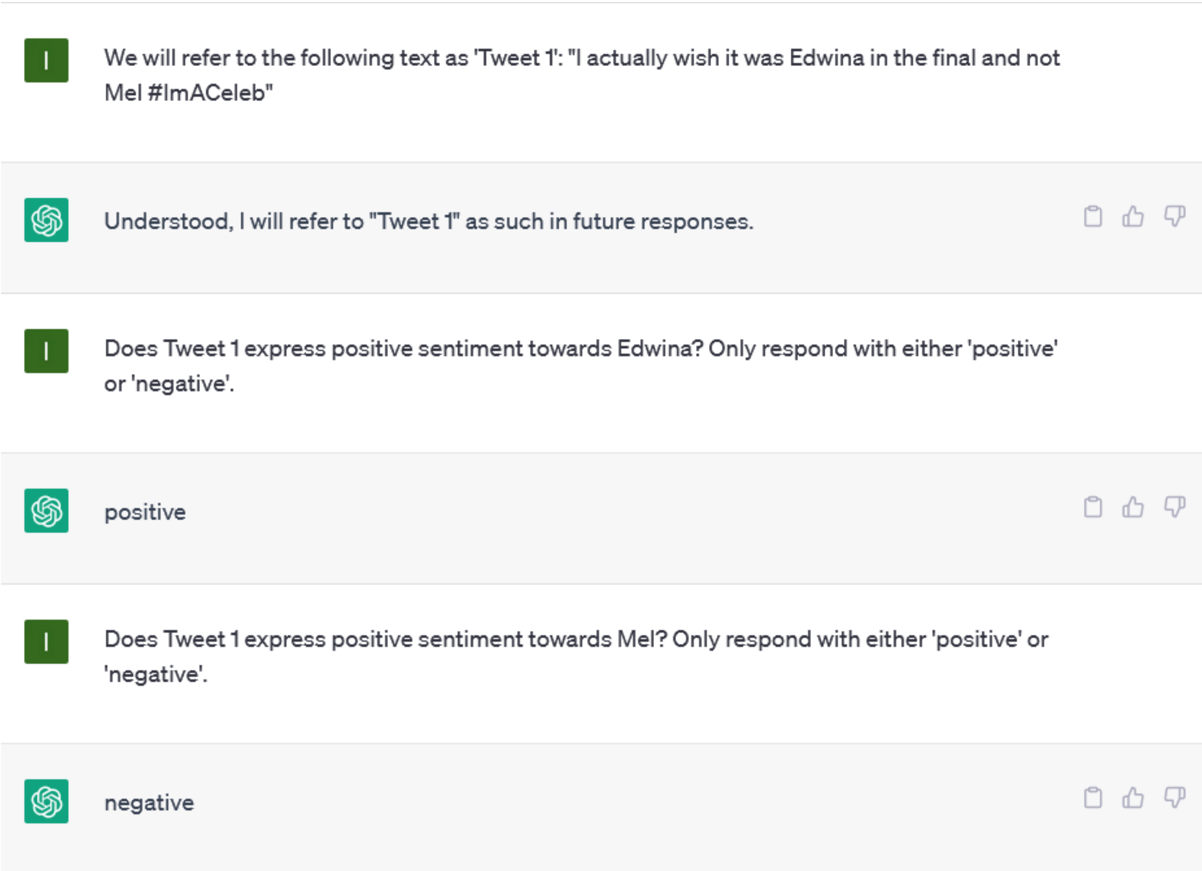


Figure 25 – Demonstration of ChatGPT identifying differing sentiment towards multiple contestants in the same tweet

Figure 25 shows how this experiment was performed. As you can see, it is in fact possible for ChatGPT to determine differing sentiment, as well as to respond in a consistent manner which could be fed into our model. If implemented, this may increase accuracy by enabling more accurate prediction of sentiment within our dataset.

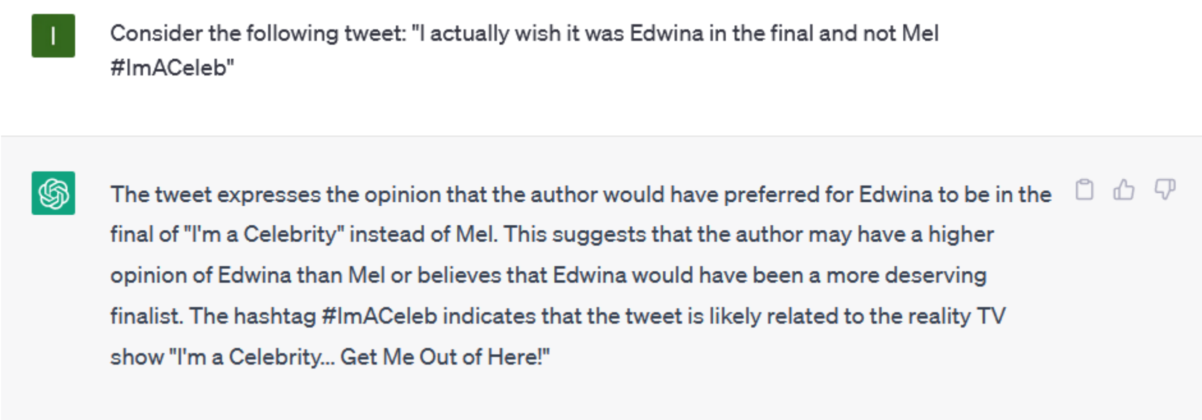


Figure 26 – ChatGPT justification for sentiment classification

Furthermore, figure 26 demonstrates ChatGPT's 'reasoning' behind the sentiment classification it gave for the two contestants. This level of contextual 'awareness' is something which is completely impossible for a traditional sentiment analysis model.

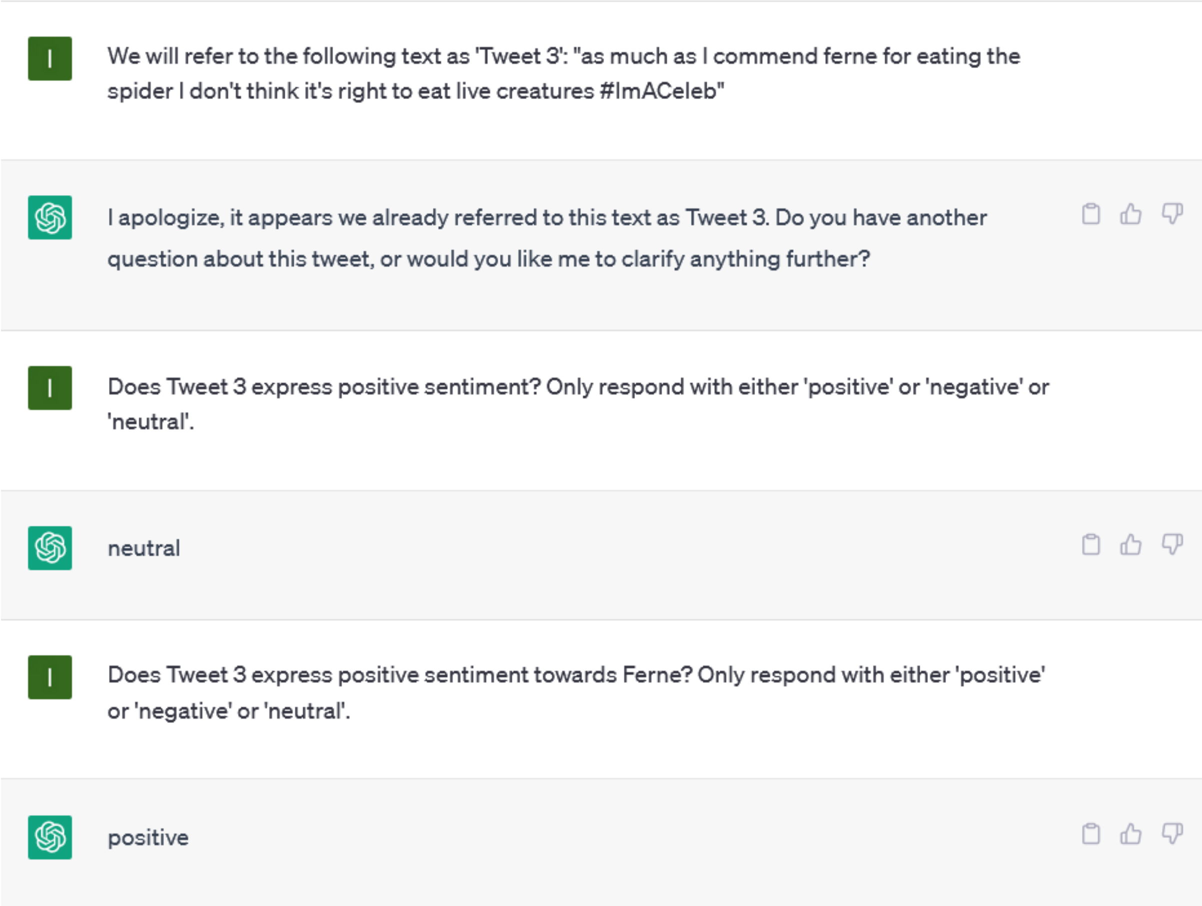


Figure 27 – Identification of differing sentiment towards multiple subjects within the same tweet

As shown in Figure 27, this system of asking the model whether a tweet is expressing positive sentiment towards a specific contestant could be implemented for all tweets, not just those referring to multiple contestants.

In this example the overall sentiment of the tweet was classified as neutral. It is unlikely that the negative sentiment in this tweet was directed towards Ferne, and more likely that the user was expressing negative sentiment towards the format of the show. When asked if the tweet was expressing negative sentiment towards Ferne, GPT3 was able to discern that the user was still expressing positive sentiment towards her, despite the sentiment being neutral overall.

While these ideas for replacing our existing sentiment analysis model have been shown to likely be highly effective, they were not implemented or tested further due to time limitations within this project.

7.2.2 Spam Filtration

When searching for *I'm a Celeb*-related tweets, it becomes apparent that there are a large number of contestant-related fan accounts. These fan accounts, despite likely being run by just one person, tweet hundreds of times throughout each competition and therefore potentially skew the voting results.

Figure 28 – Example of a Twitter fan account

Figure 28 shows one of these fan accounts on one day in 2015. As can be seen, these accounts tweet many times. If these accounts were to amass a large number of retweets, then the information they provide may be valuable because the sentiment they express is clearly shared by others, however in this example we can see that nobody is retweeting or liking their tweets, yet they make their way into the dataset.

It was found previously that filtering tweets to just one per user in the dataset limits our dataset too drastically, and therefore harms our results. One potential route to combat spam like this, without limiting our dataset, may be to implement a cap of 5 or 10 tweets per user entered within the dataset.

One option could also have been to remove tweets from accounts with a contestant's name in their username. In this scenario, tweets from '@FerneFans' would not be seen by the model. This was not possible for our dataset, because usernames were not stored directly, instead being replaced by hashes.

This system could improve accuracy; however, we do not believe that the noise created by these accounts affects accuracy a great deal. In addition, this system would have taken considerable time and effort to implement. As such, it was omitted from our final model.

7.2.3 Hyperparameter Tuning

In order to improve accuracy of the model, hyperparameter tuning may be implemented with grid search and cross evaluation to reduce inaccuracy. This was attempted briefly, but our lack of in-depth knowledge regarding machine learning meant that we did not know which parameter values to attempt. We attempted to use some standard forum-suggested values with grid search, but these were far from optimal and not tailored to our use case. No gains to accuracy were made as a result.

In future, it will be worth researching machine learning in much greater detail, enabling us to attempt new regression models, and implement hyperparameter tuning for real accuracy gains.

7.2.4 Division of Model Features by Day

Recall that the 10 features we input into our SVR regression model are as follows:

A. Number of positive tweets directed towards contestant X (As a percentage relative to the other two contestants in the vote)
B. Number of negative tweets directed towards contestant X (As a percentage relative to the other two contestants in the vote)
C. Number of positive tweets directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)
D. Number of negative tweets directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)
E. Average number of retweets on a positive tweet regarding contestant X
F. Average number of retweets on a negative tweet regarding contestant X
G. Number of "Win Indicators" found directed towards contestant X (As a percentage relative to the other two contestants in the vote)
H. Number of "Win Indicators" found directed towards the best competitor to contestant X (As a percentage relative to the other two contestants in the vote)
I. Average number of replies on a positive tweet regarding contestant X
J. Average number of replies on a negative tweet regarding contestant X

These features are constructed using statistics about every tweet in the three-day window. We found that with each day we add to the window, more noise is introduced as we begin to collect sentiment which is no longer true or relevant to the contestants on the day of prediction. In the end, three days was found to be the optimal window size.

One option for expanding our window size may be to duplicate the features above for every single day prior to the final. For example, if retaining a three day window, we would construct each feature for each day in our window and input a total of 30 features into the model for prediction rather than 10. This would enable the model to assign lower weights to tweets made a longer time ago, and higher weights to more recent ones. This may also enable us to expand our window size, as noisier data from day three onwards can be assigned a lower weighting but still considered.

This would constitute only a small change to our model, and would likely come with some small accuracy gains, however this was still not implemented. The main reason for this was due to time constraints, but also for simplicity of development and explanation.

7.2.5 Addition of Betfair Data into the Model

It was found earlier that Betfair data on its own, with just the single feature in the comparison model, is actually slightly more accurate than the entire Twitter model with 10 features. As such, it is very likely that the addition of Betfair data into the Twitter model would boost accuracy overall.

This could be done in one of two ways, either data from Betfair could be added as one or more features into the current feature pool, or an ensemble method like a voting regressor could be used to combine the outputs of two different models into one, like in figure 29.

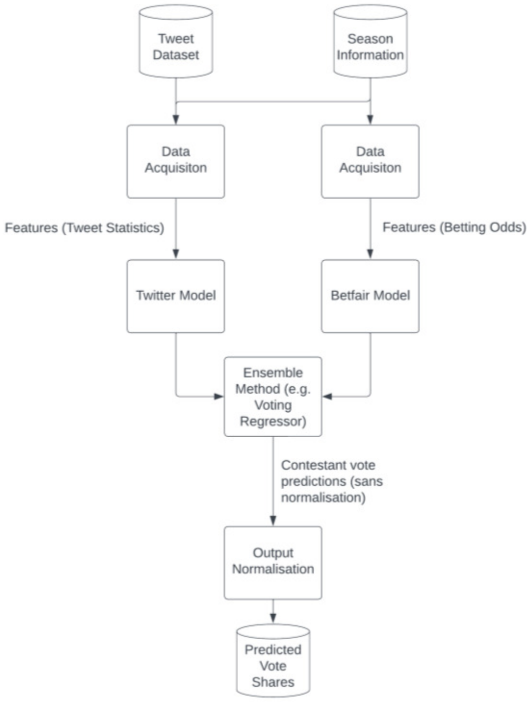


Figure 29 – Flowchart for potential ensemble method

This system, if implemented, would certainly result in far more accurate predictions overall. However, this was not implemented because it was not in line with the research questions of this project, which specifically outline the prediction of voting results using Twitter data.

In future work, if the aims and objectives were to shift to involve simply producing the most accurate model using all data available, this would definitely be a system worth implementing.

8 Reflection on Learning

Through the completion of this project, I have amassed and honed a wide variety of skills, both technical and non-technical.

8.1 Technical Skills

8.1.1 Machine Learning

The completion of this project required applying new technologies which I had never used prior. One of these was machine learning. Earlier in my degree I had taken a module on artificial intelligence, but this module mostly involved the theory and maths behind the concept. This project was more data-science related, requiring me to learn how to leverage existing models to gain useful insights.

Specific skills and concepts which I have learned relating to machine learning include implementing regression algorithms, types of regression algorithms, cross evaluation, evaluation metrics, and more.

I have found data science to be so interesting throughout this project that I have signed up for a course on Python for data science, which I will start now this project has been completed. I am also now seriously considering a career in machine learning, whereas previously I was too intimidated by the concept to attempt this.

8.1.2 Natural Language Processing

This project required the use of existing natural language processing models to evaluate the sentiment of tweets. In order to select an appropriate model, I needed to research and understand what they were and how they worked. I also needed to understand how the output of these models worked, in order to evaluate their performance and implement them in my model.

The field of NLP seems to be one which is rapidly growing and improving. This was shown, in part, by my evaluation of ChatGPT, a recent breakthrough in artificial intelligence which seems to be far better than the existing models I tested earlier. I think that in the near future NLP technology will reach very high levels of accuracy.

8.1.3 Twitter API

Prior to completing this project, I had used a large number of Python and REST APIs before, but none which were as difficult to use as Twitter's. The low rate limits and the strict querying I had to perform in order to prevent exceeding those limits meant building a querying system was difficult and time consuming. It did, however, force me to code in a way which was strict and efficient.

8.2 Soft Skills

8.2.1 Academic Writing

This project required a very large quantity of academic writing to be performed. This is something which I have never done before. I was not aware of rules and best practices which surround academic writing, e.g., writing in the past tense, and the passive.

This is one skill in which I have learnt a great deal, because I had very little experience before undertaking this project.

8.2.2 Project Management

This project was one of, if not the largest projects I have ever undertaken. It was certainly the largest that needed to be completed before a given deadline. Previously the large projects I have undertaken personally have not been for presentation to others, and so have not required rigorous planning, documentation, and testing, and as a result were not coded very cleanly.

This project has forced me to regularly consider deadlines, pace, and planning in order to complete. The ability to plan, document and evaluate my code as I work is one which will stick with me, and I have no doubt this will be of great value in the workplace.

8.2.3 Research

One important skill this project has helped me hone is the ability to find and read academic papers. I have previously read academic papers when given them by a lecturer, or when linked in a newspaper article, but I have never been required to find papers relevant to a research area on my own. In addition, I have only previously skim-read papers, whereas throughout this project I pored over papers, making notes as I went, looking for findings which may be relevant to my project.

I especially found it hard to find papers about the importance of likes and retweets on Twitter, as very little research seems to have been performed in this area. I later realised this was because most analysis of Twitter sentiment is performed with a machine learning model, in which case weightings do not need to be explicitly defined, rather they are found automatically by the model during training.

8.2.4 Critical Thinking

This project has significantly honed my critical thinking skills. Before undertaking this project, I was not aware that the creation of input features for a machine learning model was so critical-thinking heavy. I found that I was constantly brainstorming and evaluating new ideas, trying to figure out whether they would improve accuracy. It was through this critical thinking that I thought of my idea to train the model with contestant ‘combinations’ rather than real final-vote data points. This concept was crucial to the project’s success because it alleviated a massive bottleneck in the number of data points available.

8.2.5 Independence

This project boosted my independence greatly because I was required to work by myself to complete a very large body of work by the deadline. However, it would be unfair to state that I was truly independent throughout the course of the project. This project would not have been possible without help from my project tutor Fernando. Weekly meetings with him helped keep the course on track and inspired me with the ideas that have made this project successful.

8.2.6 Ethics

Before starting this project, I needed to refresh my knowledge of the Cardiff University Computer Science faculty's ethics regulations. I needed submit a full ethics application to allow the use of Twitter data. Although this was relatively straightforward, one precaution I did need to take was the anonymisation of Twitter data by hashing usernames. Abiding by a strict framework of rules to achieve ethical approval was something which I had been taught about but never put into practice.

8.3 Summary of Learning

In summary, this project has taught me a huge number of skills and knowledge, both through self-learning, trial and error, and through my supervisor. I am extremely glad that I chose this project initially, even though I had been doubting my ability to create a project which was outside of my comfort zone. It has opened my mind to the possibility of different and exciting career paths, and I will definitely be creating more machine learning projects in future in my own time.

References

Big Brother (TV Series 2000-). 2023. *IMDB.* Available at: https://www.imdb.com/title/tt0257295/ [Accessed: 6 February 2023].

Carmody, T. 2013. *Better than Nielsen: Twitter breaks down TV behavior by demographics, device, and genre*. Available at: https://www.theverge.com/2013/1/10/3861954/twitter-uk-study-real-time-television-ads [Accessed: 5 February 2023].

Love Island (2015 TV series). 2023. *Wikipedia*. Available at: https://en.wikipedia.org/wiki/Love_Island_(2015_TV_series) [Accessed: 6 February 2023].

Mayo, M. 2014. A Clustering Analysis of Tweet Length and its Relation to Sentiment. TSYS School of Computer Science, Columbus State University doi: https://arxiv.org/abs/1406.3287

Mataxas, P. 2014. Do Retweets indicate Interest, Trust, Agreement? (Extended Abstract). Department of Computer Science Wellesley College, Wellesley doi: https://arxiv.org/abs/1411.3555

Reimann, N. 2023. *Twitter Boosts Character Limit To 4,000 For Twitter Blue Subscribers* Available at: https://www.forbes.com/sites/nicholasreimann/2023/02/08/twitter-boosts-character-limit-to-4000-for-twitter-blue-subscribers/ [Accessed: 03 March 2023].

Brito, K. and Adeodato, P. 2023. Machine learning for predicting elections in Latin America based on social media engagement and polls. *Government Information Quarterly* 40 Issue 1. doi: https://doi.org/10.1016/j.giq.2022.101782

Franck, E. Verbeek, E. and Nuesch, S. 2016. Prediction accuracy of different market structures - bookmakers versus a betting exchange. *International Journal of Forecasting* 26, pp. 448-459. doi: 10.1016/j.ijforecast.2010.01.004

Twitter. 2014. *Study: Exposure to TV Tweets drives consumers to take action - both on and off of Twitter*. Available at: https://blog.twitter.com/en_us/a/2014/study-exposure-to-tv-tweets-drives-consumers-to-take-action-both-on-and-off-of-twitter [Accessed: 29 March 2023].

Page, L. 2015. *Computer scientists use Twitter to predict UK general election result* Available at: https://warwick.ac.uk/newsandevents/pressreleases/computer_scientists_use/ [Accessed: 29 March 2023].

Google. Unknown Date. *Cloud Natural Language pricing.* Available at: https://cloud.google.com/natural-language/pricing [Accessed: 18 April 2023].

A Dictionary of Computer Science. 2016. 7th ed. Oxford University Press.

Camacho-Collados, J. et al. 2022. TweetNLP: Cutting-Edge Natural Language Processing for Social Media. doi: https://doi.org/10.48550/arXiv.2206.14774

ITV. 2022. *I'm a Celebrity... Get Me Out of Here 2022 – Vote.* Available at: https://www.itv.com/terms/articles/im-a-celebrity-get-me-out-of-here-2022-vote [Accessed: 21 April 2023].

Collet, Y. 2021, *xxHash.* Available at: https://cyan4973.github.io/xxHash/ [Accessed: 24 April 2023]

Betfair. 2018. *Betfair Historical Data* Available at: https://historicdata.betfair.com [Accessed: 28 April 2023].

Wormer, T. 2022. *afinn-111* Available at: https://www.npmjs.com/package/afinn-111 [Accessed: 01 May 2023].

Scikit-learn Developers. 2023. *Support Vector Machines* Available at: https://scikit-learn.org/stable/modules/svm.html#regression [Accessed: 05 May 2023].

Hillygus, D. 2011. *The Evolution of Election Polling in the United States*. doi: http://www.jstor.org/stable/41345918

TextBlob. 2020. *TextBlob: Simplified Text Processing* Available at: https://textblob.readthedocs.io/en/dev/ [Accessed: 05 May 2023].

NewBettingSites. Unknown Date. *Why are Betting Exchange Odds better than Fixed Odds?* Available at: https://www.newbettingsites.co/articles/why-are-exchange-odds-better-than-fixed-odds/ [Accessed: 06 May 2023].

Plunkett, J. 2010. *The X Factor phone votes bring in £5m* Available at: https://www.theguardian.com/media/2010/dec/13/x-factor-phone-votes [Accessed: 10 May 2023].

Smarkets. Unknown Date. *How do bookmakers make money* Available at: https://help.smarkets.com/hc/en-gb/articles/214180825-How-do-bookmakers-make-money [Accessed: 10 May 2023].

Pedregosa, F. et al. 2012. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research doi: https://doi.org/10.48550/arXiv.1201.0490

Sohail, S. 2023. *The Math Behind Betting Odds and Gambling* Available at: https://www.investopedia.com/articles/dictionary/042215/understand-math-behind-betting-odds-gambling.asp [Accessed: 10 May 2023].

Attarwala, A et al. 2017 How Efficient is Twitter: Predicting 2012 U.S. Presidential Elections using Support Vector Machine via Twitter and Comparing Against Iowa Electronic Markets. *2017 Intelligent Systems Conference*, doi: https://doi.org/10.1109/IntelliSys.2017.8324363

Sethi, A. 2020. *Support Vector Regression Tutorial for Machine Learning* Available at: https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/ [Accessed: 12 May 2023].

Appendix

A: JSON Season Information File Contents

The below image shows the contents of one of our season information files. This file specifically contains all the relevant information surrounding season 20 of *I'm a Celeb*.

seasons > {} season-20.json > ...


```json
{
    "contestants":[
        "Giovanna Fletcher",
        "Jordan North",
        "Vernon Kay",
        "Shane Richie",
        "Mo Farah",
        "AJ Pritchard",
        "Jessica Plummer",
        "Russell Watson",
        "Victoria Derbyshire",
        "Beverley Callard",
        "Ruthie Henshall"
    ],
    "nicknames":{
        "Giovanna Fletcher": ["@MrsGiFletcher", "giovanna", "fletcher"],
        "Jordan North": ["@jordannorth1", "jordan", "north"],
        "Vernon Kay": ["@vernonkay", "vernon", "kay"],
        "Shane Richie": ["@realshanerichie", "shane", "richie"],
        "Mo Farah": ["@Mo_Farah", "mo", "farah"],
        "AJ Pritchard": ["@Aj11Ace", "AJ", "pritchard"],
        "Jessica Plummer": ["jessica", "plummer"],
        "Russell Watson": ["@russellthevoice", "russell", "watson"],
        "Victoria Derbyshire": ["@vicderbyshire", "victoria", "derbyshire"],
        "Beverley Callard": ["@Beverleycallard", "beverley", "callard"],
        "Ruthie Henshall": ["@RuthieHenshall", "ruthie", "henshall"]
    },
    "eliminations": {
        "Giovanna Fletcher": "04/12/20",
        "Jordan North": "04/12/20",
        "Vernon Kay": "04/12/20",
        "Shane Richie": "03/12/20",
        "Mo Farah": "02/12/20",
        "AJ Pritchard": "02/12/20",
        "Jessica Plummer": "01/12/20",
        "Russell Watson": "01/12/20",
        "Victoria Derbyshire": "30/11/20",
        "Beverley Callard": "30/11/20",
        "Ruthie Henshall": "29/11/20"
    },
    "end-date": "04/12/20",
    "air-time": "21:00",
    "third-place-vote": {
        "Giovanna Fletcher": 0.3945,
        "Jordan North": 0.3406,
        "Vernon Kay": 0.265
    },
    "final-vote": {
        "Giovanna Fletcher": 0.5027,
        "Jordan North": 0.4973
    }
}
```

B: TweetNLP and TextBlob Experiment

The table below shows an extract of the 100 tweets from the experiment in which the accuracy of TweetNLP and TextBlob was evaluated.

Tweet Contents	Tweet NLP	TextBlob	Human	TweetNLP Correct?	TextBlob Correct?
@fernemccann had the hardest challenge. That SPIDER!!! #ImACeleb	positive	negative	neutral	0	0
Just about to watch celeb - think George should win! #ImACeleb	positive	positive	positive	1	1
Vote for George! @higeorgeshelley #georgeofthejungle #imaceleb	neutral	negative	positive	0	0
George to win!!! #ImACeleb	positive	positive	positive	1	1
Only one person deserves to win #ImACeleb and thats @fernemccann. She has proved you can conquer your fears if you put your mind to it	positive	positive	positive	1	1
That spider though. as if #ImACeleb #fernetowin @fernemccann	negative	negative	neutral	0	0
@fernemccann is an absolute machine!! Allllll the votes to her. She's definitely done the most horrendous trials #ImACeleb #fernetowin	negative	positive	positive	0	1
Hahaha George why do you want #moussaka?? #ImACeleb	neutral	positive	neutral	1	0
@fernemccann what a woman #hero #ImACeleb	positive	negative	positive	1	0
@fernemccann just no words. I salute you after that bushtucker trial. #ImACeleb	positive	negative	positive	1	0
No way did Ferne eat that spider?! #ImACeleb	negative	negative	neutral	0	0
@fernemccann has to win after that! Cannot believe she ate a spider #ImACeleb	negative	positive	positive	0	1
#ImACeleb what a trial wow ferne respect oh my life no way could half the world do that	negative	negative	positive	0	0
Vicky to win! #ImACeleb	positive	positive	positive	1	1
Who will come 3rd... I predict Ferne #ImACeleb	neutral	negative	negative	0	1
@fernemccann you hero!!!!! FERNE TO WIN!!!! #ImACeleb https://t.co/pvXn2EKH57	positive	positive	positive	1	1
@imacelebrity @fernemccann deserves to win after that trial.. @higeorgeshelley @VickyGShore all fab celebs their all winners #ImACeleb	positive	positive	positive	1	1
as much as I commend ferne for eating the spider I don't think it's right to eat live creatures #ImACeleb	negative	positive	negative	1	0
@higeorgeshelley GOOD LUCK #ImACeleb https://t.co/rsoHVDgp91	positive	positive	positive	1	1

Table of Figures

Figures

Tables