Problem Chosen **2023** **Team Control Number**
C **MCM/ICM** 2312970
Summary Sheet

Summary

Math is everywhere, even in the games we play. Math is involved in game design, leading to the creation of entire fields of mathematics. And on the consumer side, mathematics can be used to model games, find optimal strategies, and predict the game's popularity. In this paper, we want to model Wordle's popularity and score dynamics. Wordle is a puzzle game offered by the New York Times where players have to guess a solution word based on clues given to them by previous guesses. This interaction produces a large volume of data, players report on Twitter. We want to analyze this data to predict game popularity and score trends.

As in any data analysis project, we begin by cleaning the data. This was done in two steps. First, we took a surface look at the data to identify errors in its structure, such as incorrectly spelled words or data that stands out. Second, we perform a numerical analysis to identify outliers quantitatively. Once the data is either fixed or removed, we move on to modeling. To determine the number of reports on a given day, we first perform data exploration by generating graphs relating the number of reports to the contest number. The relationship looks to be exponential decay. We perform gradient descent to find the best fit, as well as an upper and lower bound to perform interval prediction.

Word difficulty is an essential and foundational piece in Wordle. After all, some words are more difficult to guess than others. How can we determine what makes a word difficult? We carefully employ K-means clustering, an algorithm to group objects in 'clusters' that have similarities. Based on these clusters, we can determine a proper process for constructing an accurate prediction model.

In our search to find an additional method of estimating word difficulty, we created a neural network. It takes in a word as input and outputs the estimated percentage of players that take each number of tries. The results of this method were promising but sub-optimal due to the relatively small data set. We, therefore, move on to a more probabilistic approach that we call the **Four Planes Model**, relying on the word and letter frequency to determine the average number of tries to guess a word correctly. For this model, we defined a word's **difficulty score** to input into the model. We also determine that the distribution of tries on any given day is normal. Using these two pieces, we construct a predicted distribution for the percentage of players taking each number of guesses on a given day for a given solution word.

In an attempt to account for the hard mode data, we began by looking for correlations between the proportion of hard mode players to other variables. We found that the hard mode proportion increased in a logarithmic fashion over the time period. While this data is very interesting, we saw the number of reports exponentially decrease, the opposite of the proportion of hard mode players. We could not find any meaningful relationship between the hard mode proportion and the player scores or the number of reports, so this statistic was not utilized in our calculation of word difficulty or in forming the predicted try distribution of the data.

We used a variety of methods to perform analysis on the data set provided. We discovered many possible metrics to measure the difficulty of a word with relative accuracy. We also identified trends in the number of reports over time, along with the portion of players reporting hard mode.

Finally, to assist the New York Times puzzle editor, we wrote a letter summarizing our data and offering suggestions to improve the game's popularity.

Contents

1 Introduction

1.1 Background

Wordle is a puzzle game from the New York Times with daily contests in which players try to guess a 5-letter word based on clues given for previous guesses. They have six tries to guess the correct word. With each guess, they will be informed that either a) a letter is not present in the solution, b) a letter is present in the solution, but not in the correct place, or c) a letter is present in the solution and in the correct place.

In addition to this base set of rules, players can challenge their vocabulary by activating ”Hard Mode”. This adds a rule that each subsequent guess *must* use the known letters.

In addition to the game, there is also a community of players who report their scores on Twitter. A bot aggregates all of this information and publishes information related to the reports, including the total number of reports, the number of those that were played in hard mode, and the percentage of players that guessed the word in one, two, three, four, five, or six tries along with a percentage of players who failed.

1.2 Problem Restatement

The New York Times has asked us to perform an analysis of this data to answer several questions regarding the data. These questions include the following.

- The number of reported results vary daily. Develop a model to explain this variation and use your model to create a prediction interval for the number of reported results on March 1, 2023. Do any attributes of the word affect the percentage of scores reported that were played in Hard Mode? If so, how? If not, why not?
- For a given future solution word on a future date, develop a model that allows you to predict the distribution of the reported results. In other words, to predict the associated percentages of (1, 2, 3, 4, 5, 6, X) for a future date. What uncertainties are associated with your model and predictions? Give a specific example of your prediction for the word EERIE on March 1, 2023. How confident are you in your model's prediction?
- Develop and summarize a model to classify solution words by difficulty. Identify the attributes of a given word that are associated with each classification. Using your model, how difficult is the word EERIE? Discuss the accuracy of your classification model.
- List and describe some other interesting features of this data set.

Using the results of our analysis, we must also write a one- to two-page letter to the Puzzle Editor of the New York Times summarizing our results.

1.3 Assumptions

To simplify the modeling process and provide more clear and concise insights into the data, we make the following assumptions.

- The data provided in this problem is representative of the entire population of players. We make this assumption because the sample size is relatively large.
- The same proportion of overall players will report their score each day. We make this assumption because we do not see huge isolated spikes in the data.
- For our initial models, we will assume that each player has the same skill level, and disregard the ”hard mode” player proportion. We make this assumption for the sake of simplicity and defer the ”hard mode” player proportion to later models.
- The difficulty of the given challenge is proportional to the average number of tries it takes to complete. This assumption is made on the basis of common sense. If a task takes more tries to do correctly, it is by definition more difficult.
- The average number of tries to complete is can be viewed as a continuous linear difficulty scale. This assumption was made for the sake of simplicity in modeling and predicting.

1.4 Variables

We define each row in the data set as a set of variables described as follows.

Symbol	Description
n	the total number of rows in the cleaned data set.
n_d	the total number of rows in the ”dirty” data set.
i	the row number for the associated variable.
d_i	the date of the contest on row i.
c_i	the contest number of the contest on row i.
w_i	the corrected solution word of the contest on row i.
r_i	the number of reports for the contest on row i.

2 Mathematical Model

2.1 Data Cleaning

Upon visual analysis of the data, we found several errors. To begin, there were several words that were not strings of 5 characters. This is impossible since the game only uses five letter words. Instead of removing this data, we were able to use an algorithm to pull the dates of these errors. We then used the Wordle Archive (16) managed by the game’s creator, Josh Wardle, to determine the correct string. The strings were altered manually to rescue these data points.

For example, the 525th contest word is entered as ”clen”. By utilizing the Wordle archive, we can see that the real contest word was ”clean”, and add the missing ’a’ character to the string. This does not change any of the numerical values of the data, but it would interfere with our calculations and models involving word difficulty. Another word that would interfere was ”naive” with an accented ’i’. We replaced the accented ’i’ with a normal one to make data parsing easier.

While creating models and lines of best fit for the data, we noticed there were several points of data that did not seem to fit with respect to the number of reports for the given day. When trying to identify outliers, one usually employs a measure of deviation from the mean. However, we concluded that it would not be advisable to use the standard deviation from the mean to determine outliers, as the mean is sensitive to those outliers.

Instead, we decided to use a process which is focused on the median, a measure that is less sensitive to outliers. To make this decision, we looked at the breakdown point of both the median and mean measures. The breakdown point is defined as ”the maximum proportion of observations that can be contaminated . . . without forcing the estimator to result in a false value” (10). For a median measure, the breakdown point is 50%. For a mean measure, the breakdown point is 0%, as a sufficiently large measure will produce an outlier mean.

Using the median metric, we can follow the precedent set by Rousseeuw and Hubert (13) of eliminating outliers of more than three Median Absolute Deviations from the median. This will seek out any outliers which can then be removed from the data set. Due to the large variance in the overall data set, we decided to go with a sliding window technique for the identification of outliers. For window size $2w + 1$, where w is the number of days in the window to either side of the value being tested, there are $m = n_d - 2w$ unique windows into the n_d rows of the ”dirty” data set. With M_i being the median value for window i, the Median Absolute Deviation is calculated by

$$\text{MAD} = 1.483 \cdot \underset{i=1,\ldots,n}{\text{median}} |r_i - M_i|$$

The constant 1.483 is a correction factor that makes the Median Absolute Deviation unbiased at the normal distribution (13). This correction factor is calculated by using the inverse phi function as follows

$$1.483 = \frac{1}{\Phi^{-1}(0.75)}.$$

Note that this is the same calculation to find the minimum standard deviation to put the value into the 4th quartile of the normal distribution, and is the default scaling factor for any embedded MAD function (2). To use this correction factor, it must be ensured that the distribution of the number of reports for any given window is normal. To test the normality of the data points in the sliding window, we utilized the Shapiro-Wilk test. Checking each window for normality using this method yielded a mean p-value of 0.504, which is indeed above the threshold of $\alpha = 0.05$ needed to not reject the null hypothesis that the points in the window are normally distributed.

In failing to reject the null hypothesis that the windows are normally distributed, we were able to proceed to the identification and removal of outliers in the number of reports. Using $w = 2$, we performed Median Absolute Deviation analysis on each window, yielding figure 1.

There is only one outlier that is identified by this method. The number of reports was an order of magnitude smaller than its surroundings, most likely indicating a data entry error. We removed this row from the data file and proceeded with the modeling process.

2.2 Daily Report Prediction

To start, we came up with a model to predict an interval of reports on a given day. We began with data exploration, creating a line graph of reports over time to get a qualitative feel for the shape of that data.

Overall, the right side of the peak of popularity seems to demonstrate exponential decay. To focus in on this aspect of the graph we removed the first 50 rows. Determining if this graph was truly an

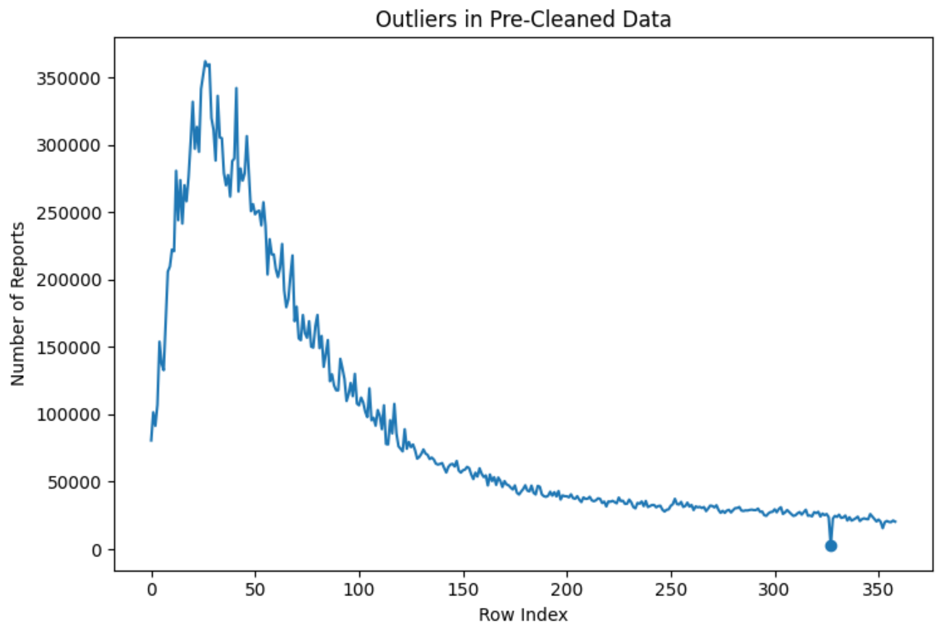


Figure 1: The Outliers Identified by the Shapiro-Wilk Test

exponential decay would be done in two parts. First, an exponential function would need to be found that best fits the given data. Second, the goodness of fit would need to be tested.

2.2.1 Gradient Descent

Gradient descent is an iterative numerical technique to minimize some cost function over a number of variables. In this case, the function to be minimized was the sum of the squared errors. The variables for this function were the parameters for the exponential curve that we were trying to fit to the data.

To perform gradient descent one must start at some point, find the direction that decreases the cost function the most with respect to the all of variables, move in that direction a small amount, and repeat until some condition has been met. This is exactly what we did in our code. Our exponential curve function was defined as

$$f_{a,b,c}(x) = e^{a(x-b)} + c$$

where a, b, and c were the variables to minimize the cost function over. The cost function was defined as

$$c(a, b, c) = \sum_{i=1}^{n} (f_{a,b,c}(c_i) - r_i)^2$$

Where c_i is the contest number for row i.

Arbitrary initial values were set for a, b, and c. The cost function was evaluated at and around those variables, they were adjusted slightly in the direction of steepest descent, and the process was repeated. The loop would break when the cost function had reached a local minimum and no movement was made between iterations. This found $a = -0.020$, $b = 883.201$, and $c = 25417.726$ to be the best fit for the data. The results of this can be seen as the curves in figure 2 on page 6. The next step was to evaluate just how good the "best" fit was.

2.2.2 Goodness of Fit

It is often possible to find a parameters for a curve of best fit for any data set and function. Quadratic data has some linear "best" fit, regardless of the overall "goodness" of the fitted curve. There are many methods available to mathematicians to measure the goodness of fit (9). One of these methods that is available to us is that of the root-mean-square deviation.

The root-mean-square deviation is defined as

$$\text{RMSD}(\hat{\theta}) = \sqrt{\text{E}((\hat{\theta} - \theta)^2)}$$

where $\hat{\theta}$ is an estimator for the parameter θ. Using the best fit values that we obtained for a, b, and c we get the following formula for RMSD.

$$\text{RMSD}(a, b, c) = \sqrt{\frac{\sum_{i=1}^{n}(r_i - f_{a,b,c}(c_i))^2}{n}} \approx 135577.646$$

This number looks pretty large, but considering that tens of thousands of people play Wordle daily maybe is actually an indicator of a good fit. This number alone does not make it easy to evaluate the goodness of fit. For that, we perform one more transformation on the RMSD.

To normalize the RMSD of an estimator one must divide the RMSD by the range of the data set. The range of a datset is found by subtracting the minimum value found in the set from the maximum value found in the data set. Performing this operation on the Wordle data set produces the following.

$$\text{NRMSD} = \frac{\text{RMSD}}{\max_{i=1}^{n}(r_i) - \min_{i=1}^{n}(r_i)} \approx \frac{135577.646}{346354} \approx 0.391$$

An NRMSD value of 0.391 is acceptable. Colloquailly, NRMSD values from 0.2 to about 0.5 are said to be indicator of a good fit. With this statistic, we proceeded to create the model to predict the number of reports for a given day.

2.2.3 Interval Estimation

Using a similar principle of gradient descent with a slightly modified cost function, we found fitted curves of upper and lower bound having $a = -0.014$, $b = 1139.706$, $c = 23693.704$ and $a = -0.006$, $b = 2057.598$, $c = 2498.933$ respectively. These curves can be seen in figure 2 on page 6.

The cost function for each of them was modified to more harshly penalize deviations on one side of the data, resulting in a fit that would "prefer" to be above or below all points. For the upper bound curve, deviations below the data points were raised to an exponent of 4, while those above the line were only squared. The opposite process was taken for the lower bound.

Using the fact that the contest on March 1, 2023 is contest number 620, we can predict the upper and lower bound for the number of reports submitted for that contest by finding $f(620)$ for both the upper and lower bound. Doing so indicates that the upper bound of reports on March 1, 2023 is 25,140 and the lower bound is 10,892.

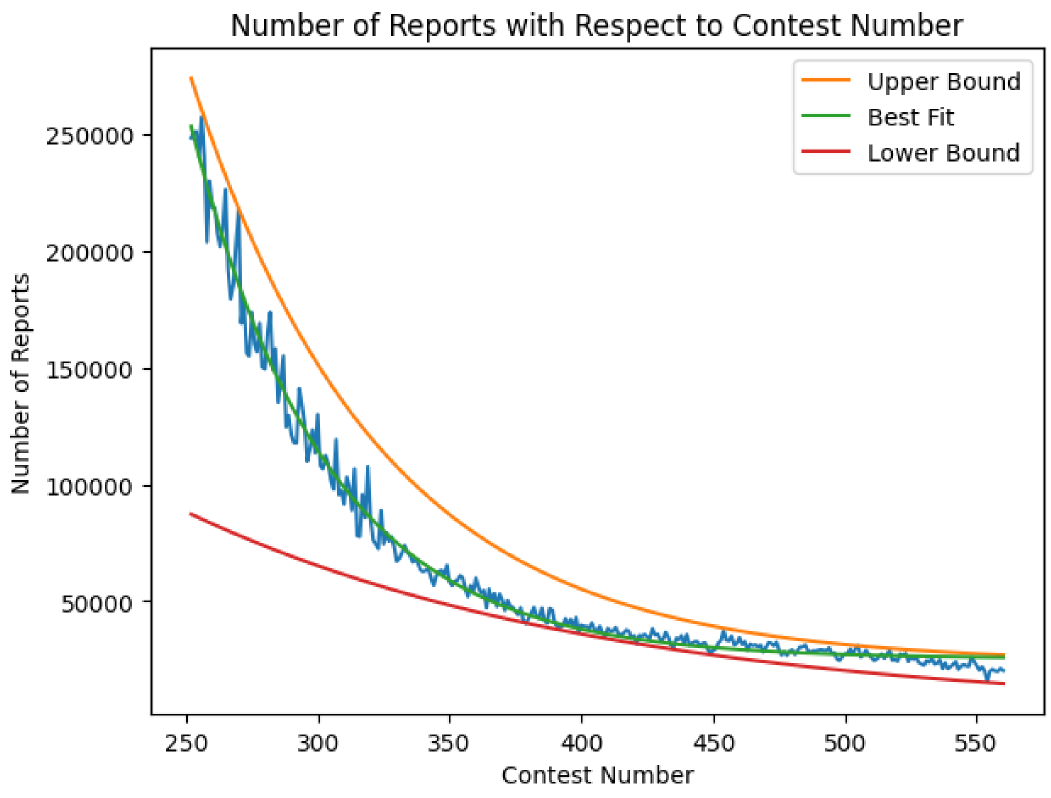


Figure 2: The Fitted Curves for the Data

2.3 Clustering

Clustering is a valuable tool to determine how we can group specific data sets by the likeness of the present variables. To produce a meaningful prediction on the attempted tries for a future date, we must first determine what makes a word less or more difficult. Our assumption is that the difficulty of the word is directly proportional to the amount of tries it takes to correctly guess it. Therefore, we can cluster the data using a unsupervised learning algorithm to group words based on their difficulty.

2.3.1 K-means Clustering

K-means clustering is a form of unsupervised learning which essentially takes a table of data and groups it into K clusters based on their similarity. The algorithm assigns each observation to the nearest cluster, based on a distance metric, with the goal of minimizing the sum of squared distances between the observations and their assigned clusters.(4) In our current case, we use the numbers of tries as our data to determine the difficulty of the word. For example, words that have a higher average amount of tries are more difficult than words whose guesses took fewer tries to determine.

Each cluster has a 'centroid', the mean of the data points in a cluster. The algorithm attempts to minimize the sum of squared distances between each data point and the centroid of the cluster it belongs to.(8) The algorithm works in a series of steps to produce these clusters.

1. Initialize the centroids: The algorithm randomly assigns K data points from the data set provided.

Each of the centroids will be used as starting points for each of the clusters. If we let $m_k(t)$ be the centroid of the k-cluster at step t, we then have

$$\{m_1(0), m_2(0), \ldots, m_k(0)\}$$

as the centroids of the specified clusters at step 0. In our current case, we have k = 4.

2. Assign data points to each cluster: The algorithm then assigns each data points to the nearest pre-defined cluster centroid. The distance metric used in our case is Euclidean distance. If we let $C_i(t)$ be the cluster where the ith object belongs at step t, we have that

$$C_i(t) = \underset{1\leq k\leq K}{\arg\min} = d(z_i, m_k(t)) \qquad i = 1, 2, \ldots, q$$

assigns each object to the nearest centroid at step t. d, in our case, is the euclidean distance metric to determine the distances between all data points.

3. Calculate new centroids: After the data points have been assigned to the respective clusters, the algorithm then calculates the new centroids for each cluster by taking the mean of the data points present inside that cluster. Therefore, we have that

$$\{m_1(t), m_2(t), \ldots, m_k(t)\}$$

are the new centroids at step t.

4. Repeat steps 2 and 3: The algorithm then repeats these steps, assigning objects and updating centroids, until the centroids no longer move or until a maximum number of iterations is reached. The centroids no longer move when the intra-cluster variance

$$V(t) = \sum_{k=1}^{K} \sum_{i:C_i(t)=k}^{q} (z_i - m_k(t))^2$$

does not change.

5. Output the clusters: We output the K clusters which contain the assigned observations and visualize as necessary.

2.3.2 Cluster Results and Process

There are different methods to the K-means clustering algorithm. The most common K-mean clustering methods are: "Hartigan-Wong", "MacQueen", and "Lloyd-Forgy" methods. We determined the Hartigan-Wong method was the most appropriate, as it is essentially an improved method of MacQueen and Lloyd-Forgy methods. (8) The forming of clusters is heavily influenced by the choosing of initial centroids. In other terms, the other methods may fail to converge. Depending on the initial assignment of the centroids, these methods may get stuck in a local optima, thus prone to error.
After employing our K-means algorithm, the clusters shown in figure 5 on page 10 were outputted. The visualizations of these clusters are shown in a PCA (Principal Component Analysis) plot. The farther the observations are from each other, the less alike they are. You can see clearly defined borders

which indicate accurate and healthy clusters (described below). At a glance, the red cluster contains observations of which we deem as ”difficult”. It takes some human analysis and intuition to determine, for yourself, the similar characteristics each cluster contains. The blue cluster, in this case, would be the easier words. Words that are harder to guess are in the red cluster. We can see this since the words present in the red cluster are less utilized on a daily basis in conversation thus providing some slight verification in our results from our visual judgement. Another conclusion to these clusters, as expected, is the construct of the word itself further elaborated in the *Difficulty score* section. That is, letter placements and vowel usage. It’s not enough to intuitively assume these clusters are correct so we also employ a ’silhouette analysis’ and additional materials to verify the integrity and accuracy of these clusters. In addition, the scoring metric described in that section, which is a culmination of our modeling techniques, shows that in figure 4, our scoring metric obeys our clustering analysis.

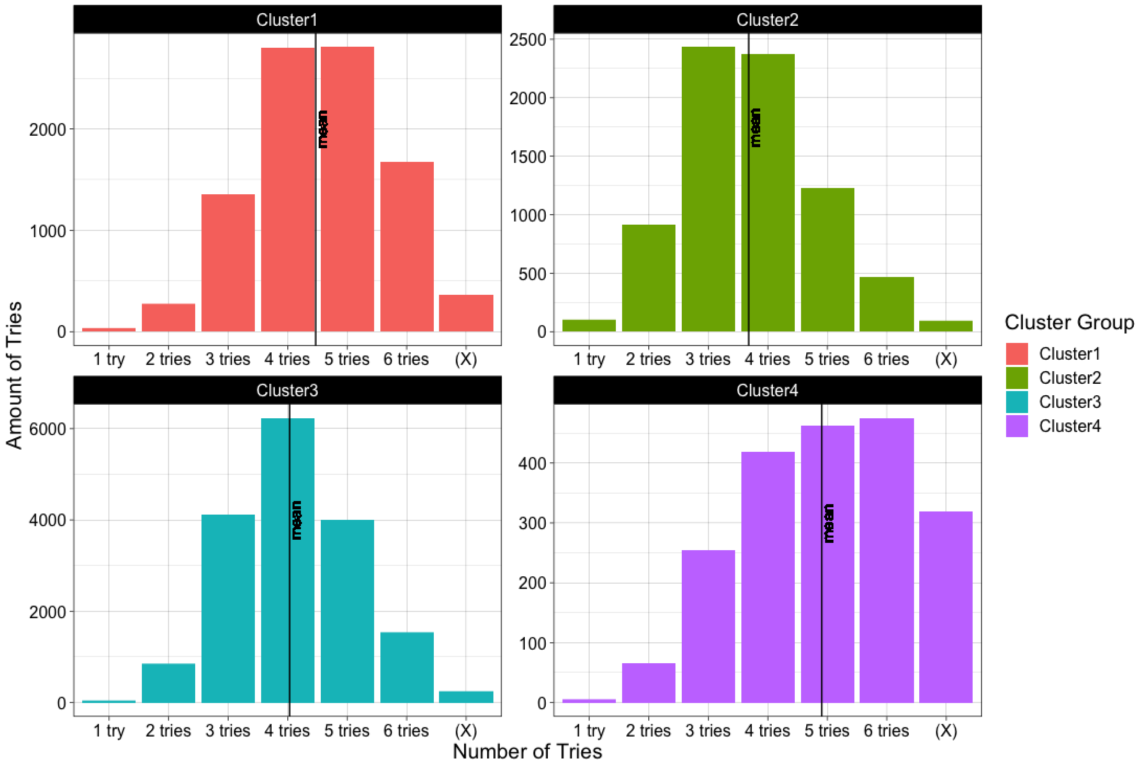


Figure 3: Distribution of the 4 Clusters

2.3.3 Clustering Verification and Analysis

How can we decide what the optimal amount of clusters are? In some instances, more of fewer should be utilized to produce more robust and reliable clusters. The more observations that lie in between the clusters but do not necessarily lie within one is considered to influence validity of the clusters. The more observations that lie outside a cluster, the less reliable it is. Therefore, we employ what is called Silhouette Analysis for verifying the validity. (12) Clusters with a large silhouette width are well clustered and those that have a small silhouette width tend to have more objects lying in between

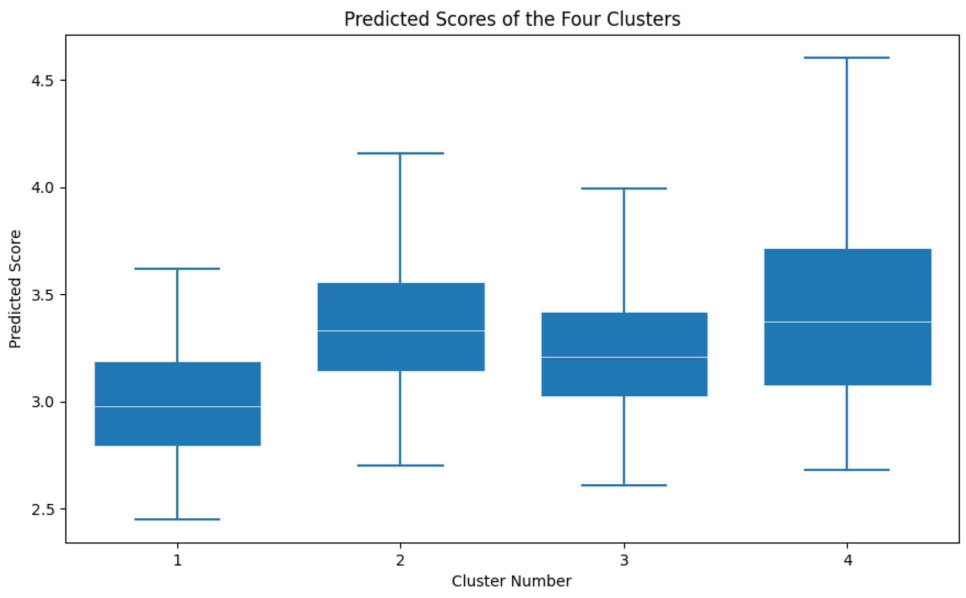


Figure 4: The Scores with Respect to Cluster Number

clusters. The following equation is used to determine this silhouette width.

$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \qquad s(i) = 0 \text{ if } |C_i| = 1$$

where $a(i)$ and $b(i)$ is given as the following

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$$

$$b(i) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i, j)$$

$a(i)$ is the mean distance between i and all other data points in the same cluster, where C_i is the number of objects belonging to cluster i. $b(i)$ will be the smallest mean distance of i to all other points an any other cluster. From the above definitions, it is clear that

$$-1 \leq s(i) \leq 1$$

Therefore a silhouette width with a value closer to 1 implies the data is appropriately clustered and badly clustered if closer to -1. We performed silhouette analysis on our clusters for a range k values (number of clusters). The optimal amount of clusters is 3, with a silhouette score of around 3.30. However, we grouped our clusters in 4 groups with a silhouette score of around 3.00. For our modeling, its really important to see noticeable differences amongst the difficulty of words. We were confident that having 4 clusters was sufficient to group them appropriately and to improve our scoring. It's also important to note that the original centroids are randomly chosen. Depending on the original placement of them may result in different clusters. The differences among them are slight and have very little effect on our model. For additional verification, observe Figure 3. For each Cluster, we plotted the total amount of tries with respect to the number of tries. The mean amount of tries it takes to correctly guess the word are noticeably different among each cluster. If the clusters are grouped incorrectly, we would expect mean values that are too close to one another. Cluster 2 had the least amount of average tries while Cluster 4 had the most amount of average tries.

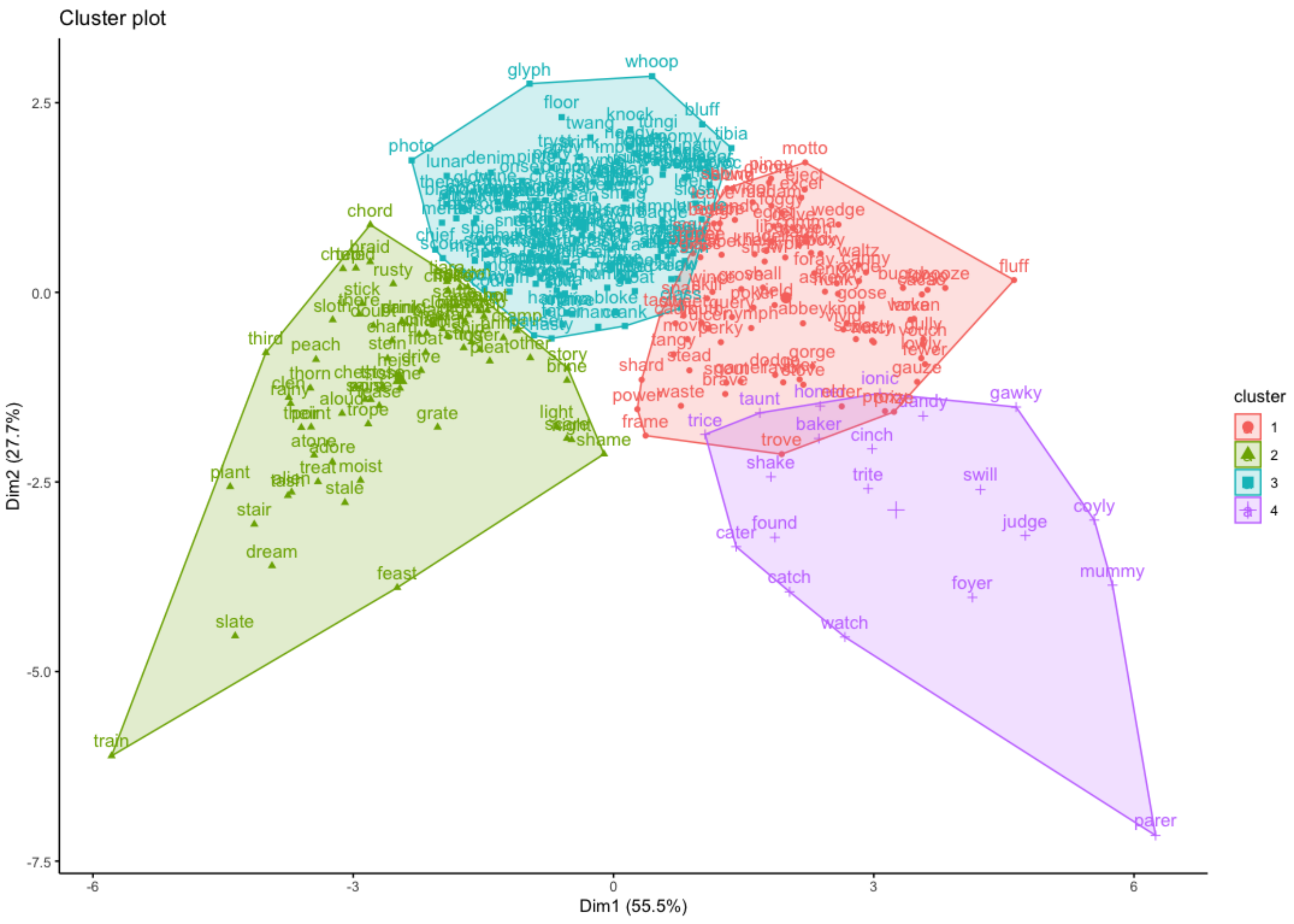


Figure 5: Clustering of words with k = 4

2.4 Neural Network

In our search to quantify what attributes of a word make it a ”difficult” word we found that a neural network trained on the data could possibly be beneficial in quantifying the difficulty of a given word based on only its textual attributes. Neural networks are machine learning techniques that simulate the structure of a brain to complexly transform some input into an output. A neural network is made up of nodes and the edges between them. The nodes are referred to as ”neurons” and the relationships between them. Standard neural networks have a layered structure with l layers, where layer i has n_i nodes and each of those nodes has n_{i+1} edges connecting it to all of the nodes in the next layer for all $i = 1, \ldots, l - 1$.

The first layer of the network is called the "input layer" and its nodes' activations are directly assigned corresponding to some inputs. The last layer in the network is called the "output layer". This is where the final output of the neural networks is. Each node in the final layer represents some characteristic that has been identified by training the networks. The layers in between are called "hidden layers" and they exist to transform the input data in more complex ways before it gets to the output layer.

These edges between nodes represent a transformation of an input value $x \in [0,1]$ to an output value $y \in [0,1]$ where $y(x) = \sigma(z(x))$ and $z(x) = W_{i,j}x + b_{i,j}$. In this equation $W_{i,j}$ represents the "weight" from node i to node j, which can be conceptualized as how much the value of node i affects the value of node j. The value of $b_{i,j}$ represents the "bias" from node i to node j, which can be conceptualized as an offset that allows the network to shift the activation value while still maintaining the weighted relationship between the nodes. Finally, $\sigma(x)$ represents the standard logistic function which is used because it maps from $\mathbb{R}$ to $[0,1]$ in a smooth way. Its equation is written as follows.

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

What may seem complicated and scary is really just linear algebra at its core. Each layer of nodes can be thought of as a column vector l_i of size n_i and each set of edges can be thought of as a weight matrix W_i of size $n_{i+1} \times n_i$ connecting layer i and $i+1$ and a bias vector b_i of size n_{i+1} connecting layer i and $i+1$. Calculating the value of the next layer is as simple as $l_{i+1} = W_i \cdot l_i + b_i$. Repeating this for layers $i = 1, \ldots, l-1$ will activate the output layer with the final values on the nodes.

The complicated part comes with training the neural network. A standard neural network uses gradient descent and back-propagation to adjust all of its weights and biases to minimize the cost of the training data in preparation for real data. This is done with some heavy calculus using partial derivatives with respect to the weights and biases. Overall, after each training iteration, the weights and biases of the network should improve in some way to more accurately predict the output given an input. The following sections describe our specific experiences in preparing the data for the network, training the network on that data, and finally evaluating the network.

2.4.1 Data Preparation

To train the model, we first needed to transform the given word into a format that is easily parsed by the network. Our word is a string consisting of 5 characters, 26 choices per character. One option, possibly a bad one, would be to have 5 input nodes to the network with each node representing a letter in the word. The activation of the input nodes would correspond to the position of the letter in the alphabet scaled from 1 through 26 to $[0,1]$. This approach does not make much semantic sense as the alphabet is not continuous and no letter is meaningfully "closer" to another in this case.

Because of the discrete nature of the alphabet, we opted for a $26 \cdot 5 = 130$ node input layer where each node represented the presence of a given letter in that group of 26 possible letters. This discretized the input while also taking into account the unrelatedness of letters in this particular situation. For example, "these" is not really similar to "thesd" for the purposes of extracting characteristics of the placement of the letters in the word itself.

For the output layer of the training data, we normalized the distribution of guesses before a correct answer to $[0,1]$ and had 7 nodes in our output layer. The network would take in a word in the form of

130 0's or 1's representing the letters of the word, transform that data somehow, and then give us the percentage of people that would take each number of tries to guess correctly.

2.4.2 Training

Because of the relatively small data set, we decided to use every row from the given file as training data. To do this we took each word from the data set, ran it through the algorithm to turn it into the correct format for the network, and compared the output with the actual distribution of number of guesses. Our network had layer sizes of 130, 64, 48, and 7. We trained it for 12 hours on the entire provided Wordle data set and proceeded to evaluation and testing.

2.4.3 Evaluation and Prediction

Evaluating a neural network on the training data is typically advised against, but on such a small data set we believed that it would be best if we did evaluate on the training data for the sake of having a larger number of evaluation points. Figure 6 on page 12 shows the results of our neural network's predictions. We fitted a linear regression to the points and there was indeed a positive correlation between the actual average number of tries and the predicted average number of tries for a given word. Although there is mostly an under-prediction, some adjustment could possibly be made for a more accurate prediction without increasing variance.

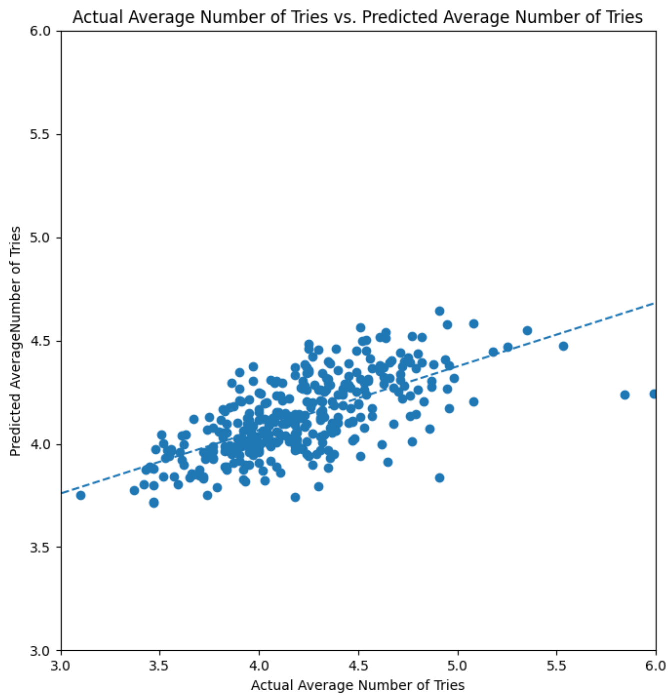


Figure 6: The Results of Testing the Network Against the Training Data

Using the average number of tries to guess a word as the measure of difficulty is a simple but naive method. If it was to be used, this neural network prediction model could predict the average number of tries that any given word would take to guess. This is a model that can map a word to a difficulty rating. If we discretize this with evenly sized and spaced bins, we can categorize the words into any number of difficulties.

2.5 Predicted Difficulty and distribution of Tries

2.5.1 Assumptions and Data Preparation

In order to predict the difficulty and distribution of tries for a future contest, we must make several assumptions. The first is that the number of tries can be considered to be a continuous, linear scale by which we can use classical statistic measures. For example, we can consider a problem that takes an average of 6 tries to be twice as difficult as a problem that averages 3 tries to solve.

In an attempt to predict the distribution of of a given day based on a word, we need to estimate the distribution the tries follow, and create a measure of the word difficulty. We will call this measure the words *difficulty score*.

2.5.2 Try distributions

To further analyze and predict future contests based on this data, we need to understand more about the proportion of users taking a specific number of tries. If we want to predict the proportion of players who solve the puzzle in *n* tries, we must understand the distribution of attempts. Since each day has over two-thousand players, conventional statistical wisdom and the Central Limit Theorem suggest that we can assume their scores will be normally distributed. (5) However, this does not account for differences in playing style. I.e. some players use strategies that maximize the probability they will solve the problem, while sacrificing the number of moves needed to solve it. An example of this strategy would be using a vowel heavy first guess such as ”adieu” or ”query”. Then using the secondary two guesses to identify as many letters as possible, without worrying what letters were found prior. Then, once all the letters of the word are identified, use the remaining guesses to order the letters inside of the word as outlined by Groux.(7) Others will use a riskier strategy where once a letter is found, they will always use it in a subsequent guess. This strategy is similar to playing in hard mode, however it may lead to the player solving the puzzle in less turns. With these biases in the data, we want to test each day to ensure they follow a normal distribution.

We will start indexing each day as a vector with the percentage value of each score indexed. A discrete sample of size one-thousand was generated for each day. This population followed the same try distribution set by the original vector. This population was then fit to a normal distribution using the embedded fit functions in the R program. This fit was then test versus the sample to determine the significance. Similarly, a random control sample of equal size was generated based on the parameters of the fit. The sample was also test against this control population. The Shapiro-Wilk test discussed on page 3 of this paper unfortunately cannot be used here as it breaks down for samples of more than fifty values.(15) Thus, here we must use the discretized version of the Kolmogorov-Smirnov Test and comparison tests, embedded in the R program, to determine if these are from the same distribution (1). Both of which will test the null hypothesis that both populations are from the same distribution. Higher p-values are positive results in our case, as we would fail to reject the null hypothesis that they are from the same distribution. However, we run into an additional problem. Our sample data is bounded from above by the maximum score of 7, failing to solve the puzzle. When plotting a normal distribution, the distribution will predict that several players will obtains scores of 8 or 9. This is solved simply by replacing those scores with the maximum value of 7, and same with the minimum value of 1. As we can see in the violin plot in figure 7 on page 14, the two populations are very closely distributed. However, this bounded normal distribution loses its normality. Thus, we are unable to use

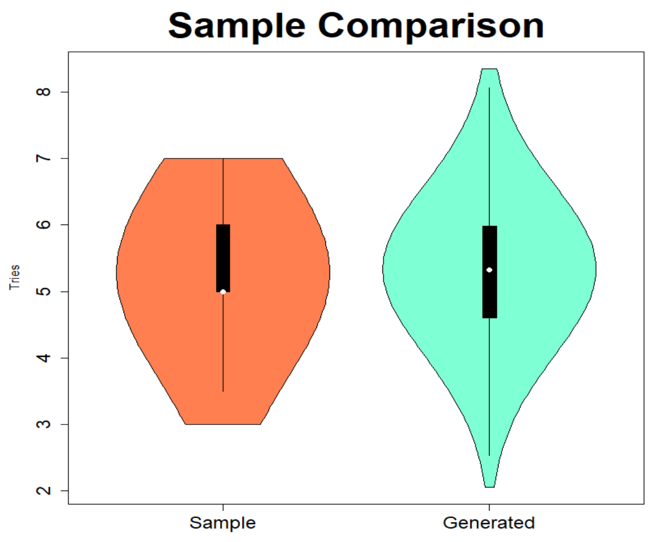
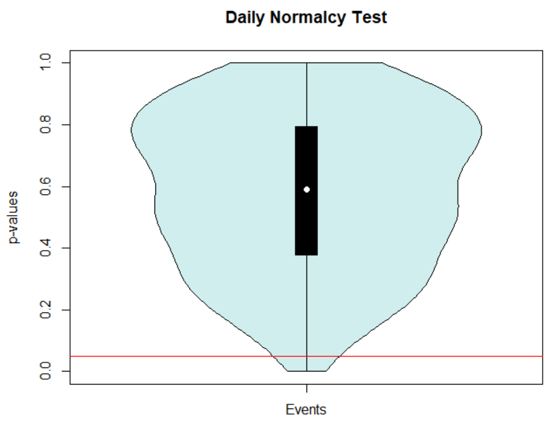
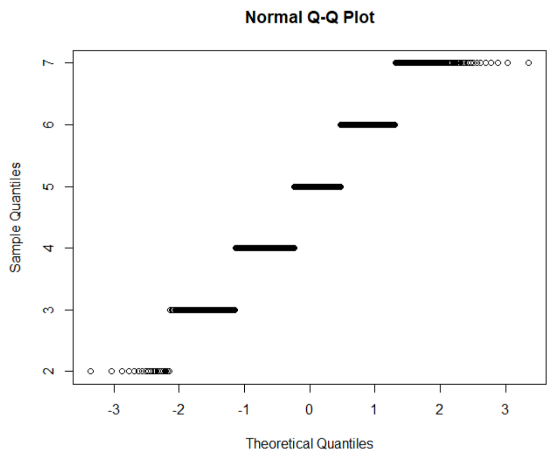


Figure 7: Comparison of the shifted generated population versus the sample distribution

the Kolmogorov-Smirnov test to confirm the normality of these distributions. In this case, we resorted to our comparative tests. If we use a standard test to determine the probability that these two samples are from the same population, then we are able to confirm normally.



(a) Figure 8a: Normality p-value frequency



(b) Figure 8b: Representative Q-Q plot

Figure 8: Daily Normality Measures

Figure 8 displays the fact that in most days, we failed to reject the null hypothesis with an average p-value of 0.6. In fact, more than 97% of the days failed to reject the null hypothesis. This indicates that we can safely assume that each day's distribution is normal. As an additional qualitative measure, we produced discretized quartile-quartile plots for each day. The Q-Q plot is used to measure residuals from a data sample to the normal distribution. The Q-Q plot works by taking a given sample $X_n = \{x_1, \ldots, x_n\}$, and comparing like positioned elements with a known distribution.(11) For example, the two comparison elements being compared would be

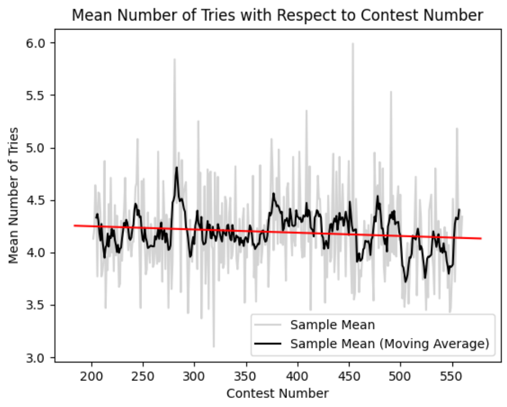
$$\text{pos}(x_j : X_n) = \text{pos}(\eta_j : N(\mu, \sigma^2)).$$

For a normal distribution, these data should fall along a straight increasing line

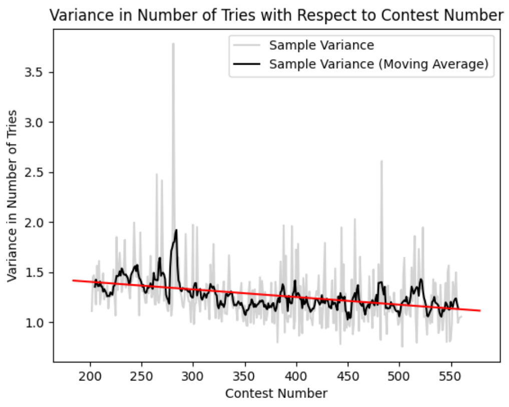
$$x_j \approx \mu + \sigma\psi,$$

where ψ_j is the q_j^{th} quartile of a $N(0,1)$ distribution.(11) For a discretized normal distribution, this will appear as a step function following the same line. In producing a Q-Q plot for each day, we visually confirmed our results of normality for the try distribution of each day. Thus, our assumption that each player is of the same skill holds, regardless of playing strategy. This is also important as we can use a normal distribution to predict the try distribution on future dates.

Now that we know that the distribution of tries for each day is normal, we need to examine what determines the parameters. We will begin with time dependence. If we plot the values of the mean versus time, we can see this graph is very noisy, figure 9. Thus, we will find local averages by utilizing a rolling window. This process will be similar to that outlined earlier in the paper



(a) Figure 9a



(b) Figure 9b

Figure 9: Time dependence of Variance and Mean Try values

The running average is plotted in black, and we can note that visually, there may be a linear relationship in both graphs. We used a linear fitting function to determine the line of best fit. From there, we tested the fit of these lines and found that the linear relationship was not significant in the mean ($p = 0.1331$), and was significant in the variance ($p = 4.722 \times 10^{-7}$). Therefore, the mean number of tries for each day does not depend on the time, but the characteristics of the word, and the variance is linearly dependent on time. The line of best fit for the variance is graphed in red,

$$\sigma^2 = 1.5573769547899334 - 0.0007619209747874947 * c_n.$$

We were able to determine the trend of the variance which can be used to predict the distribution on future dates.However, this is an issue because we need to create a discretized prediction based on a continuous normal distribution. But now, we have another issue of using a continuous distribution to produce a discrete population. However, we can follow the precedent set by Roy in using the continuous normal distribution to generate a new discrete population (14). This also makes sense, as the normal distribution is used for quasi continuous variables in many areas such as class test scores.

2.5.3 Difficulty score

As discussed earlier in the paper, there are many ways to estimate the difficulty of a given word. Using some data exploration techniques like clustering, one can find certain aspects of words that group them together. One of these aspects is the multiplicity of letters within the word. In Wordle, there are four types of multiplicity: single, double, double double, and triple. A word with a single multiplicity is one where no letters are repeated, that is, all five letters appear only once in the word. A word with a double multiplicity has some letter that appears twice throughout the word, such as ”paper”. A word with a double double multiplicity has two pairs of repeated letters. There are four of these in the provided data set, one of which is ”madam”. Finally, a word with a triple multiplicity has a single letter repeated three times throughout the word. There are only two of these words in the provided data set: ”mummy” and ”fluff”.

Another aspect that could affect the difficulty of a word is the letter frequencies in the word. A word that contains a lot of the most frequently used letters is more likely to be guessed by a smart player, who would choose words that include these most common letters in them. Maybe the sum of the relative frequencies of the letters in a word affects the average number of tries needed to guess the word correctly. We calculated relative letter frequency from a corpus of $333,333$ unigrams (6) and used it in further calculations. The letter frequency score $s(w)$ of a word w is calculated as follows where w_i is the i-th letter of the word and $f(l_i)$ is the relative frequency of the given letter.

$$s(w) = \sum_{i=1}^{5} f(l_i)$$

Using this large corpus of words and their frequencies we were also able to calculate the relative frequency of the word itself, denoted by $F(w)$. We used this in our calculation of word score, but as it would come out after some regression it was not a very influential factor in number of guesses before success.

Overall, we devised a scoring function that we call the **Four Planes Model**. That would take into account all of these aspects of a word. The scoring function effectively places the word's difficult score on one of four planes based on its multiplicity and adjusts the score further based on the sum of letter frequencies and also the word frequency. The difficulty score function $d(w)$ is defined as follows.

$$d(w) = m(w) \cdot (\alpha \cdot s(w) + \beta \cdot e^{F(w)} + \gamma)$$

Where $m(w)$ is the multiplicity multiplier as provided by the following function definition.

$$m(w) = \begin{cases} m_1 & \text{if word } w \text{ has single multiplicity} \\ m_2 & \text{if word } w \text{ has double multiplicity} \\ m_{22} & \text{if word } w \text{ has double double multiplicity} \\ m_3 & \text{if word } w \text{ has triple multiplicity} \end{cases}$$

This makes $d(w)$ a function of 7 parameters: α, β, γ, m_1, m_2, m_{22}, and m_3. Similar to how it was used earlier in the paper to determine the upper and lower bounding curves for the number of reports with respect to contest number, we can use gradient descent to fit these parameters to minimize the squared errors with respect to actual average number of tries. Starting with arbitrary values for the parameters, gradient descent finds $\alpha = -3.903$, $beta = -4.450$, $gamma = 12.281$, $m1 = 0.613$, $m2 = 0.656$, $m22 = 0.699$, and $m3 = 0.702$ to be the best fit. For the parameters.

Upon inspection, word frequency had little to no effect on the average number of tries. This is reflected in our decision to discard the dimension for the purposes of figure 10 on page 17. For accuracy's sake, we kept the word frequency as a variable in other calculations.

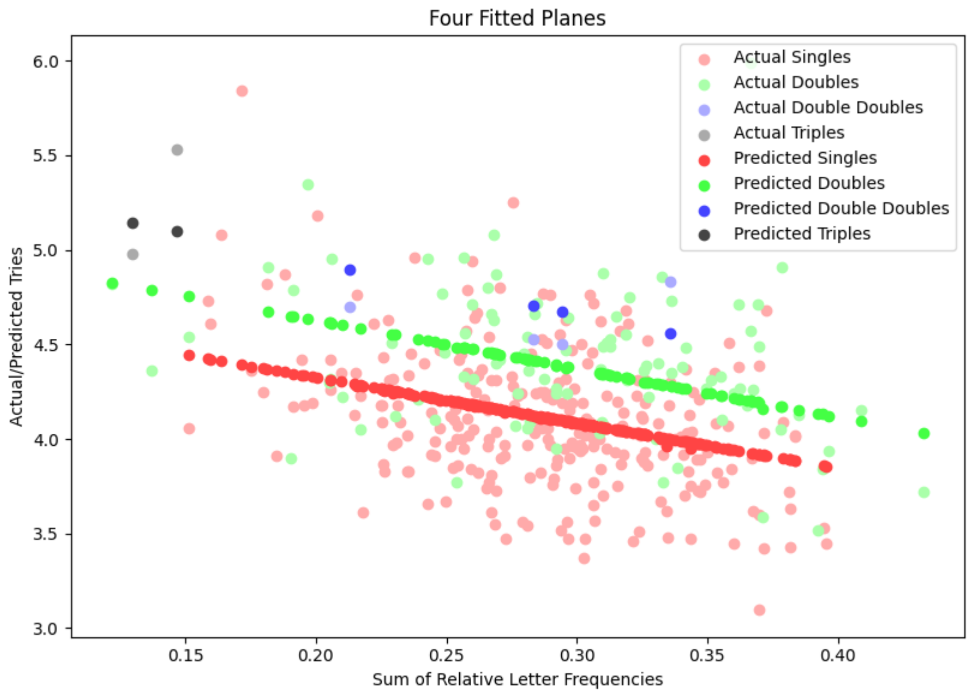


Figure 10: The Fitted Planes for the Data

The model fitting resulted in four lines within the scoring function. One can see that the ref plane which represents the scores of words with single multiplicity is the lowest of the lines. Triple and double double multiplicity words had the highest lines, corresponding to a higher actual average number of tries before correct guesses. One aspect to note about this graph is that the sum of relative letter frequencies and the actual/predicted tries are inversely related. Using this scoring method we can move on to generate the distributions of guesses for a given word.

2.5.4 Model and Extrapolation Techniques

We would like to know the try distribution for the word ”eerie” on March 1, 2023. To calculate this, we will begin by using our scoring metric to calculate the mean number of tries that it will take. Performing this scoring function on ”eerie” results in a score of 4.229, meaning that on average ”eerie” will take 4.229 tries to solve. Similarly, we can use our line of best fit to determine the variance of this distribution. We know that March 1st, 2023 will be the 620^{th} contest. Thus we can see that $\sigma^2 = 1.557376 - 0.00076192097478749(620) = 1.084985$. From the mean and variance we are able to generate a population based on the normality of the data. From this population, we must then discretize the population into buckets which are counted as our numbers of tries. This will be our estimate of the distribution on a given day.

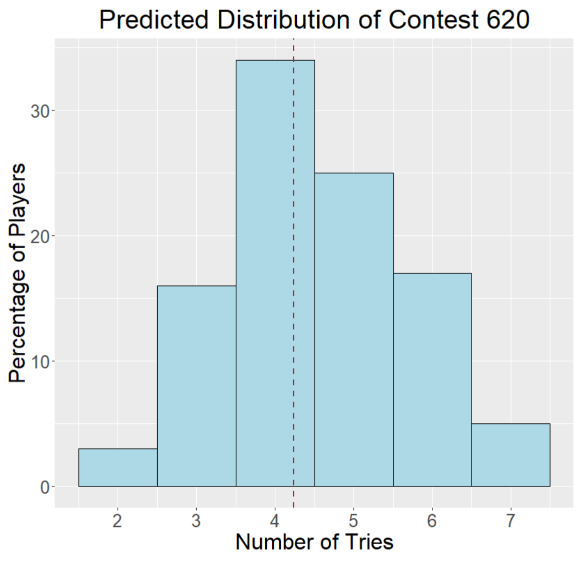


Figure 11: Histogram of try distribution prediction

The predicted distribution for the word ”eerie” on March 1, 2023 is shown in Figure 11 on page 18. We can see that most players will guess the word ”eerie” in 4 tries.

	1 try	2 tries	3 tries	4 tries	5 tries	6 tries	failure
Percentage	0	3	16	34	25	17	5

Table 1: Numerical representation of try distribution

2.6 Model Justification

Overall, there are many approaches that one could take to analyzing the number of reports and measuring word difficulty. As shown in this paper, numerical techniques such as clustering or gradient descent are critical to obtaining quantitative measures on real-world data. Furthermore, statistical techniques like the Shapiro-Wilk test and the measure of Normalized Root-Mean-Square Deviations for goodness of fit tell a mathematician just how closely the ”best” fit is to the actual data it was fit to.

We believe that the four-planes scoring method that we used is a good fit for the data. Along with that, we believe that because the distribution of tries on any given day is normal we can predict the distribution for a given day using a normal distribution.

3 Results

3.1 Sensitivity Analysis

We performed sensitivity analysis on our scoring model. Figure 12 shows how changes in α, β, and γ affect the scores. In our model, α was least sensitive to changes and γ was most sensitive. Changes in α resulted in very small changes to the score, while changes in γ could alter the score by nearly 0.5 points. Similarly, mere 5% changes in β could produce scores varying by up to 0.2 points.

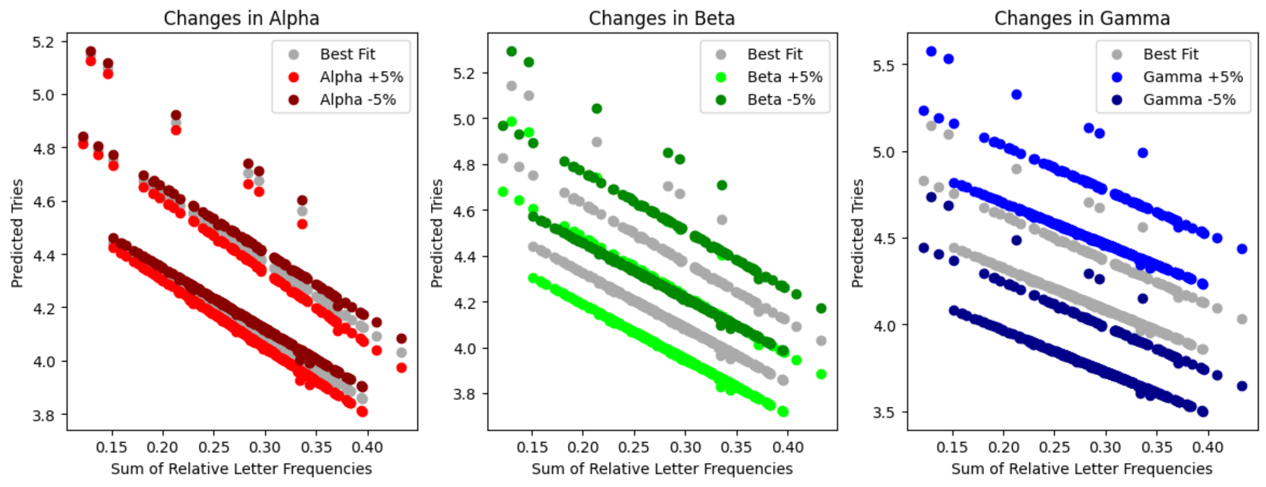


Figure 12: The Results of Sensitivity Analysis

3.2 Model Strengths and Weaknesses

3.2.1 Strengths

- **Robust:** We carefully cleaned the given data and designed our model to hold up against possible outliers. We also ensured that all of our assumptions about the distributions of the data were not evidently false.
- **High Word and Letter Frequency Accuracy:** Using a corpus of one third of a million words we were able to accurately calculate the absolute and relative frequencies of words and letters in the English language. We used these calculations to provide a more robust scoring model.

3.2.2 Weaknesses

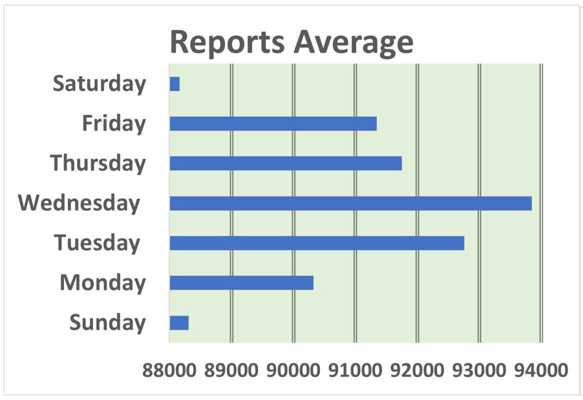
- **Single Data Point Days:** Because each day only has a single data point for number of reports and the fact that the number of reports decays over time, we had to test normality on a sliding window rather than the data set as a whole.

- **Small Provided Data Set:** With regard specifically to the neural network model, the provided data set was relatively small. For machine learning, huge data sets are preferred for optimal results.
- **Linear Regression:** For the estimation of some parameters, linear regression fails at too large of an extrapolation. For example, in figure 8b we estimate variance with respect to time using linear regression. Because variance can not be negative this regression will fail at a sufficiently large contest number.
- **No Outliers at the Edges:** While using the 5 day window for a running average gives a good indicator of outliers, this method will fail to pick out any outliers that may be in the first or last $(n-1)/2$ days where n is the window size.

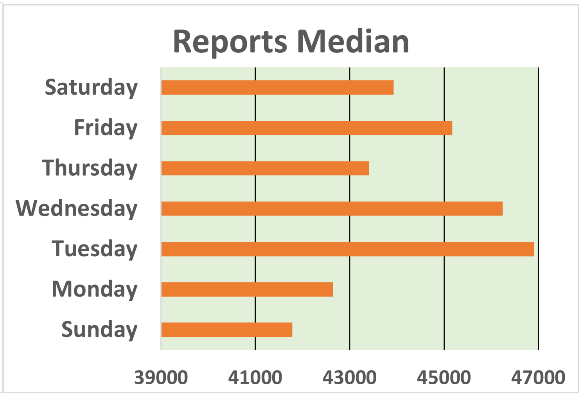
3.3 Some Other Interesting Features of this Data

3.3.1 Days of the Week

In our initial exploration of this data, we wanted to find any obvious trends or relationships. Using basic excel conditional functions we were able to find a curious distribution of the popularity of the game throughout the week as shown in figure 13.



(a) Figure 13a



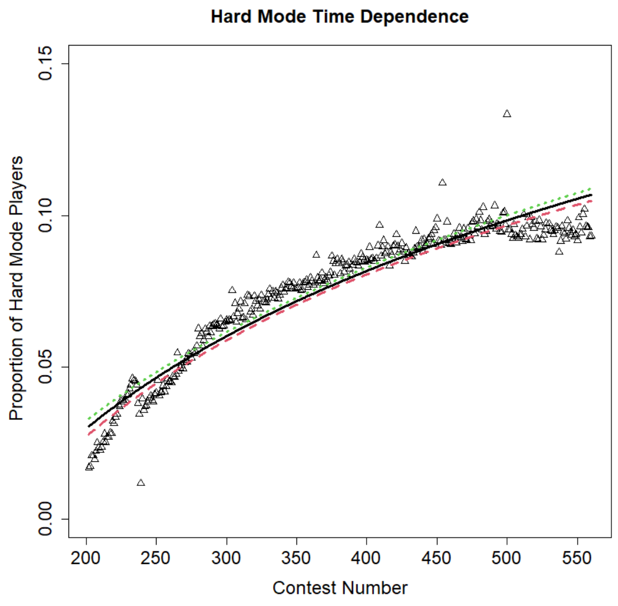
(b) Figure 13b

Figure 13: Time dependence of Variance and Mean Try values

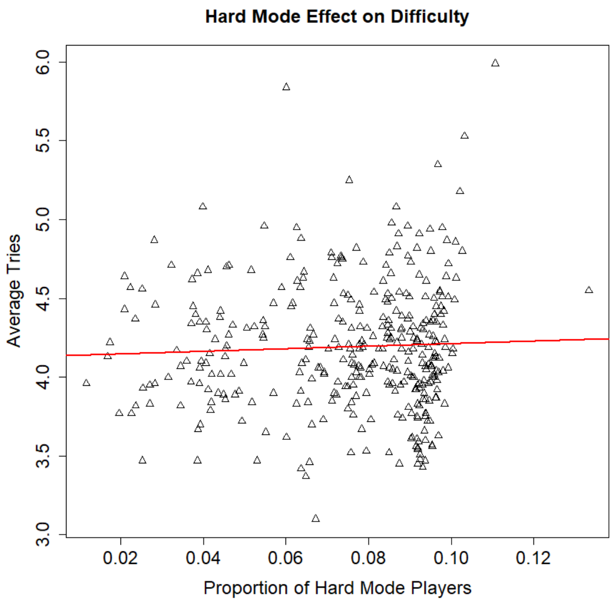
Conventional wisdom would suggest that games and puzzles are much more popular on the weekend rather than weekdays. This is because people are busy at work and with commitments throughout the week and often binge on games during the weekend. However, this data suggests that Wordle is most popular on Tuesdays and Wednesdays, and least popular on the weekends. There is also a large skew in the average of each game throughout the day, likely resulting from the large uptick in values in the first two months of the data. While these differences are interesting, we decided to omit them from our predictions, as the skew between the median and mean was very large.

3.3.2 Hard Mode

While examining the effect of players in hard mode, we came across two interesting pieces of data. First, we plotted the proportion of players playing in hard mode over time in figure 14a. Visually, this scatter plot seems to follow logarithmic growth. Indeed, we fit a logarithmic curve to the data along with a 99% confidence interval displayed by the dashed red and green lines. The summary statistics for this line failed to reject the null hypothesis with a p-value of 0.21, making it a good enough fit for the data. The consistent increase in proportion of players opting to play in hard mode may be due to the fact that as people get used to the game they try to challenge themselves with this mode. However, this would assume that when players play in hard mode, it requires them to make more guesses. As we can see in figure 14b, there is no correlation between the proportion of hard mode players and the average number of tries for the problem. Fitting this data was near impossible, and we were unable to attain any relationship between the two variables. Thus, the proportion of hard mode players did not effect the player scores.



(a) Figure 14a



(b) Figure 14b

Figure 14: Time dependence of Variance and Mean Try values

In addition to these metrics, we also looked at the correlation between the proportion of players using hard mode and the total number of reports. In this graph, we were unable to obtain a confident relationship as there are many confounding factors. For one, motivation theory has consistently produced valuable research on people's willingness to share results. This is tied to protecting the ego, self confidence, and self efficacy. Castagnetti and Schmacker found that people will be less willing to share information about their scores when it could effect their ego (3). With this in mind, we made the assumption that the proportion of hard mode players had no effect on the number of reports, as we cannot quantify these psychological factors. One final remark on the data is that we expected to see major upticks during periods of holiday break. Around the new year, we expected to see much larger number of people reporting their score, however this increase did not occur.

Dear Puzzle Editor,

Through our analysis of the Wordle data, we found several intriguing time-dependent relationships between the word and the amount of publicity the game receives. Our analysis began by determining a model to predict the number of reports on a given day. To do this, we needed to clean some of the data provided to us by correcting some words and removing outliers. We noticed a significant increase in reports at the start of the year; however, from here, we were able to model the number of reports using exponential decay. We used gradient descent to find the parameters of the curve. The curve and confidence interval were a good fit, and we were able to predict that you will see anywhere from 10,892 to 25,140 reports on March 1st, 2023.

Next, we wanted to begin to classify different characteristics of words. To do this, we utilized principle component analysis and k-means clustering to group the words into four groups based on difficulty. We then analyzed the components of each word to determine what gave them their difficulties. This led us to conclude that there are three major aspects that determine word difficulty, discussed in the next paragraph.

We then devise a model that predicts the distribution of the number of tries on a given day for a given word. We found that the number of tries was not related to the day itself but the word difficulty. We developed a word difficulty measure based on the data provided. The measure took in the letter frequencies, word frequency, and the presence of any double, triple, or double-double letters as noted in the k-means clustering. We developed a model that created four planes based on word characteristics, i.e., the presence of any repeated letters. This model was then used to predict a given word's mean number of tries. To obtain the distribution, we needed to test our assumption that each day was normally distributed. This was difficult given that the data was discrete, and we were comparing it to a continuous distribution. However, this is common, as it is practiced in classroom statistics across the entire educational system.

From here, we needed to understand if the parameters of the normal distribution were dependent on any other variables. We discovered that the variance of the distribution would be linearly related to time, at least over this short period. Using a linear fit, we were able to predict the variance on future dates. We then obtained the difficulty score from the proposed word, and we were able to generate the distribution on that date. As you requested, we predict that on March 1st, 2023, the average number of guesses will be 4.229, with a variance of 1.08. This leads us to believe that 5% of the players will fail, and 34% will complete it in 4 tries.

While our model will be accurate in predicting the number of players and the score distribution, we may suggest some changes in order to increase the popularity of the game. For one, there should be a higher percentage of people playing on the weekend. May we suggest a weekend challenge, where the words are more difficult or rare than usual, to entice more players to play? We may also suggest some visual changes to increase the satisfaction of a victory. Many video games use classical conditioning to reward and install repeated behavior with bright and colorful rewards.(17) In addition, many games can increase player retention and daily usage by creating an experience system and possibly giving players certain badges when they have completed arbitrary objectives. These changes may be able to resurrect the popularity of the game and sustain it for years to come.

Regards,

Team 2312970

References

[1] *Kolmogorov–Smirnov Test*, Springer New York, New York, NY, 2008, pp. 283–287.

[2] *Mad: Median absolute deviation*, RDocumentation, (2023).

[3] A. Castagnetti and R. Schmacker, *Protecting the ego: Motivated information selection and updating*, European Economic Review, 142 (2022), p. 104007.

[4] G. De Soete and J. D. Carroll, *K-means clustering in a low-dimensional euclidean space*, (1994), pp. 212–219.

[5] Y. Dodge, *Central Limit Theorem*, Springer New York, New York, NY, 2008, pp. 66–68.

[6] A. Franz and T. Brants, *All our n-gram are belong to you*, 2006.

[7] C. Groux, *Use this foolproof 'wordle' strategy to win every time.*, 2023.

[8] J. A. Hartigan and M. A. Wong, *Algorithm as 136: A k-means clustering algorithm*, Journal of the Royal Statistical Society. Series C (Applied Statistics), 28 (1979), pp. 100–108.

[9] R. J. Hyndman and A. B. Koehler, *Another look at measures of forecast accuracy*, International Journal of Forecasting, 22 (2006), pp. 679–688.

[10] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, *Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median*, Journal of Experimental Social Psychology, 49 (2013), pp. 764–766.

[11] J. I. Marden, *Positions and qq plots*, Statistical Science, 19 (2004), pp. 606–614.

[12] P. J. Rousseeuw, *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis*, Journal of Computational and Applied Mathematics, 20 (1987), pp. 53–65.

[13] P. J. Rousseeuw and M. Hubert, *Robust statistics for outlier detection*, WIREs Data Mining and Knowledge Discovery, 1 (2011), pp. 73–79.

[14] D. Roy, *The discrete normal distribution*, Communications in Statistics - Theory and Methods, 32 (2003), pp. 1871–1883.

[15] S. S. SHAPIRO and M. B. WILK, *An analysis of variance test for normality (complete samples)†*, Biometrika, 52 (1965), pp. 591–611.

[16] J. Wardle, *Wordle archive*, 2023.

[17] Q. Zhao, Y. Zhang, M. Wang, J. Ren, Y. Chen, X. Chen, Z. Wei, J. Sun, and X. Zhang, *Effects of retrieval-extinction training on internet gaming disorder*, Journal of Behavioral Addictions, 11 (2022), pp. 49 – 62.