For office use only	Team Control Number	For office use only
T1 __________	2300879	F1 __________
T2 __________		F2 __________
T3 __________	Problem Chosen	F3 __________
T4 __________	C	F4 __________

2023
MCM/ICM
Summary Sheet

Time Series Analysis for Predicting Wordle Results

Wordle is a popular puzzle currently offered daily by the New York Times. Players try to solve the puzzle by guessing a five-letter word in six tries or less, receiving feedback with every guess. We are asked to construct some models to predict the the number of results and the percentage of hard mode results on March 1st. Meanwhile, we are also asked to create a model to evaluate word difficulty and predict the distribution of different try times on March 1st given the solution word EERIE.

Since the real world time series data are usually messy, we first use Hampel Filter to eliminate the noisy data. Then we manually reduce the deviation of some particular data. And we also use statistical tests such as Granger Casualoty test to test the relation between different time series, ADF and KPSS tests to test the stationarity of time series and ACF and PACF measures to determine the parameter for our model.

We built a word difficulty estimation model that can determine the difficulty of a word quantitatively based on three criterion, including the frequency of word, the frequency of letters in this word and the repetition times of letters. The model can calculate the word difficulty score and distinguish easy and hard word.

We built a univariate time series forecasting model that can predict the number of reported results and the number of hard mode results. The model uses Autoregressive Integrated Moving Average (ARIMA) to analyze the pattern of past values to predict the future. We use two ways of calculation to cross validate the accuracy of our model in predicting the percentage of hard mode.

We built a multivariate time series forecasting model that can predict the distributions of different try times. The model uses Vector Autoregression (VAR) to analyze the past data of different distributions, the number of reported results, the number of hard mode results and word difficulty.

The main advantage of our approach is that the model utilizes both univariate and multivariate time series analysis methods to better capture the pattern in past data and also the relation between different time series. In addition, our model selection and application is based on rigid statistical tests.

Key Words: Autoregressive Integrated Moving Average (ARIMA), Vector Autoregression (VAR), Hampel Filter, Granger Casuality Test, ADF Test, KPSS Test, Autocorrelation, Partial Autocorrelation

Contents

1 Introduction

1.1 Problem Background

Wordle is a puzzle game initiated by the New York Times. It offers a five-character word puzzle to the public everyday. The game has both easy and hard modes, and a user can take a guess for at most six times. The game has attracted many users across the world. With reported results accumulated in Twitter for about a year, it becomes interesting to see if the number of possible reports in the future could be predicted.

1.2 Problem Restatement

This paper wants to focus on the following questions:

1. Develop a model to explain the variation of the number of reported results and use model to create a prediction interval for the number of reported results on March 1, 2023.
2. Define attributes of words and use them to build model to estimate word difficulty. Identify the relationship between attributes of word and the percentage of hard mode. And use this model to determine the difficulty for the word EERIE.
3. Develop a model to predict the distribution of the reported results. Give a specific example of prediction for the word EERIE on March 1, 2023

1.3 General Definitions

- Wordle: a puzzle game offered by New York Times for players to solve a five-letter word in six times or less, with the help of hint from each try.
- Yellow tile: a letter in the final word but has been put in the wrong location.
- Green tile: a letter both exist in the final word and in the right location.
- Gray tile: a letter which does not exist in the word at all.
- Regular mode: the game mode to guess the right form of the word in less than or equal to six times with the previous hint given. each time players could guess for the right word without limitation of choices of characters.
- Hard mode: the game mode to guess the right form of the word in less than or equal to six times with the previous hint given, but players must use the words with yellow tile in the following tries.
- Difficulty of words: the approximately average efforts and time for players to come up with the specific word in continuous thinking in the Wordle.

- Frequency: the times a word appears in the specified corpus or dictionary.
- Repetition: the maximum number of times a character occurs among all characters in the word.

2 Assumptions and Notations

2.1 Assumptions

1. The players choose whether they play hard mode before they play the game and thus the percentage of hard mode has nothing to do with word difficulty.
2. The reported number of results and the number of hard mode results are not related to other data provided and thus we use univariate time series forecasting.
3. The distribution of different tries is related to the percentage of hard mode results and word difficulty and thus we use multivariate time forecasting.

2.2 Notations

1. Num1: time series of the reported number of results per day
2. Num2: time series of number of results that are in hard mode per day
3. Ratio: time series ratio of people choosing hard mode among the reported results

3 Data Processing and Analysis

3.1 Data Cleaning

After we loaded the data sets, we dropped the empty column and then change the name of "Number of reported results" to **"Num1"** and "Number in hard mode" to **"Num2"** for convenience. And then add another column called **"Ratio"**, which represents the percent of number in hard mode among all number of reported results. Then we sort values according to contest numbers so that the data is sorted by time. For contest number 545, we changed the word "rprobe" to "probe" as specified in the notice. Finally, we drop all the empty entries.

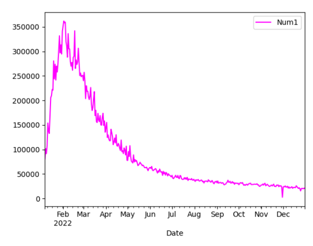

Figure 1: Line plot of Num1

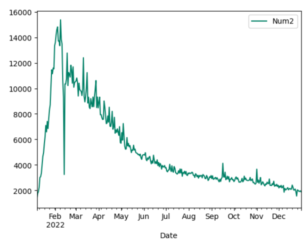

Figure 2: Line plot of Num2

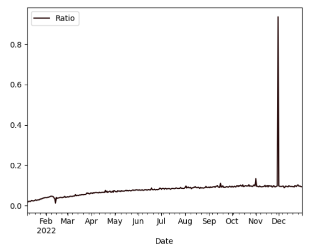

Figure 3: Line plot of Ratio

After cleaning the data, we perform a simple line plot with x-axis representing date and y-axis representing total number of reported results, number in hard mode, and their percent respectively. From the figure, it is apparently there are some points where the curve change dramatically, which reminds us that the data may contain a few data entries errors. Therefore, we want to filter the noisy data.

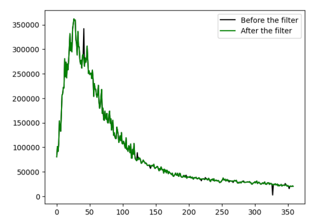

Figure 4: Line plot of Num1 before and after filtering

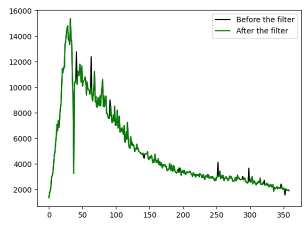

Figure 5: Line plot of Num2 before and after filtering

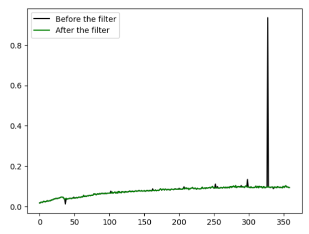

Figure 6: Line plot of Ratio before and after filtering

Hampel filter can identify outliers and replace them with more representative values. The filter is basically a configurable-width sliding window that slide across the time series. For each window, the filter calculates the median and estimates the window's standard deviation σ using the median absolute deviation. For any point in the window, if it is more than 3 σ out from the window's median, then the Hampel filter identifies the point as an outlier and replaces it with the window's median. We apply Hampel filter to our Num1, Num2 and Ratio columns because these are the critical columns for subsequent analysis and also are shown to have potential errors.

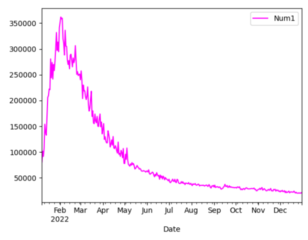

Figure 7: Line plot of Num1 after filtering

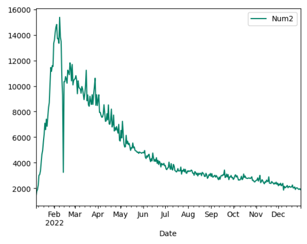

Figure 8: Line plot of Num2 after filtering

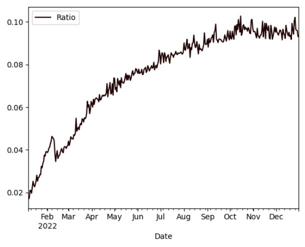

Figure 9: Line plot of Ratio after filtering

However, we did note that even after Hampel filter, for data in column Num2, there was still a sharp turn, and therefore, we used the mean value of the adjacent 4 values to compute a new representation value for it. From the figure below, we found that after correcting the error data, the sharp turn was represented by a moderate turn.

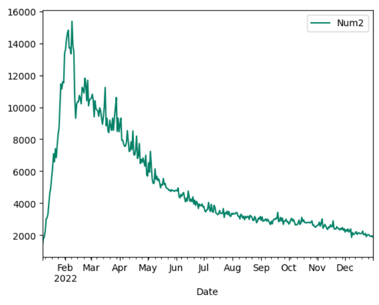

Figure 10: Line plot of Num2 after deducing the deviation value

We also plot the box plot for Num1, Num2 and Ratio. We can find that after the data cleaning process, the deviation of Num1, Num2 and Ratio are acceptable.

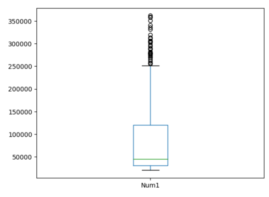

Figure 11: Box plot of Num1 after filtering

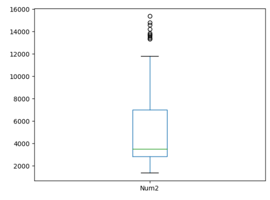

Figure 12: Box plot of Num2 after filtering

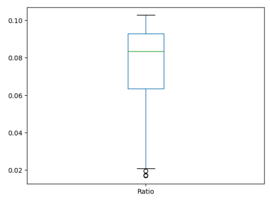

Figure 13: Box plot of Ratio after filtering

We also plot the simple line plot for all possible outcomes, and from the figures we can find that there are no significant data deviation, and therefore we do not apply Hampel filter for those data. Now, we have finished cleaning our data and it's time to use data visualization to explore patterns.

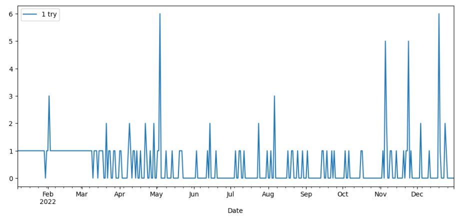

Figure 14: Line plot of 1 try

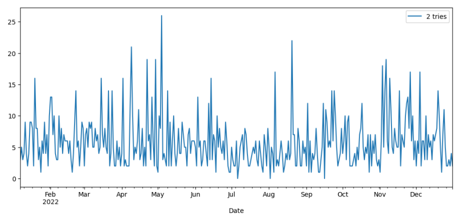

Figure 15: Line plot of 2 tries

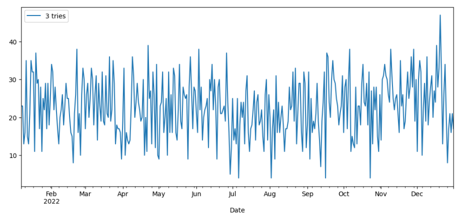

Figure 16: Line plot of 3 tries

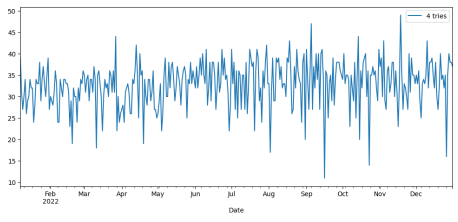

Figure 17: Line plot of 4 tries

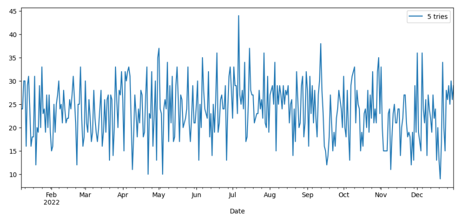

Figure 18: Line plot of 5 tries

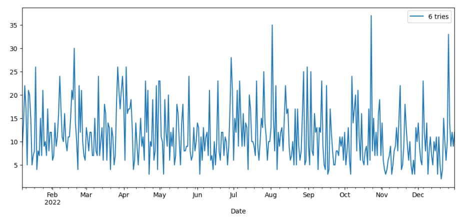

Figure 19: Line plot of 6 tries

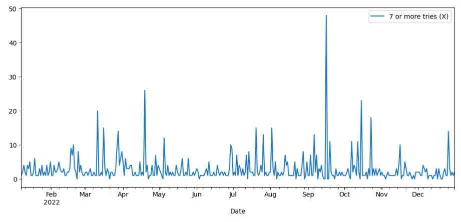

Figure 20: Line plot of 7 or more tries

3.2 Exploratory Data Analysis

We use heat map to show the correlation between different columns. In the heat map, we can find that the correlation between Num1, Num2 and number of tries are low. Therefore, the main reason which affects the distribution of different tries is not Num1 and Num2, but perhaps the difficulty of word and the ratio of hard mode intuitively. Additionally, there are negative correlations between Num1 and Ratio, Num2 and Ratio, and there are also positive correlation between Num1 and Num2.

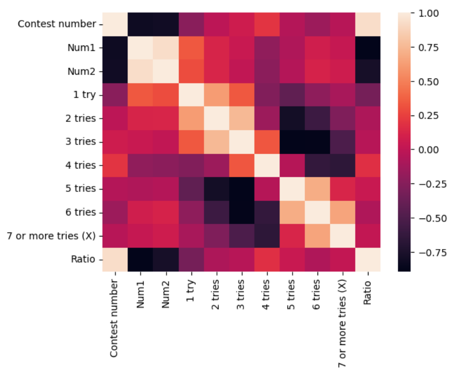

Figure 21: Heat Map of data frame

We first use scatter plot with regression analysis to explore the relation between Num1, Num2 and their ratio. We find that there is a positive correlation between Num1 and Num2, and there is a negative correlation between Num1 and Ratio, Num2 and Ratio, which also corresponds to our previous heat map analysis.

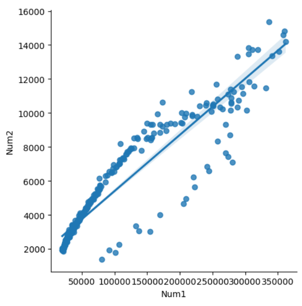

Figure 22: Regression plot between Num1 and Num2

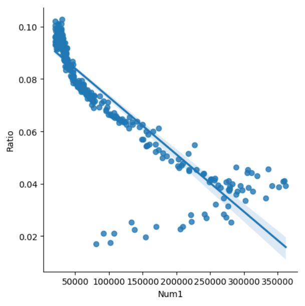

Figure 23: Regression plot between Num1 and Ratio

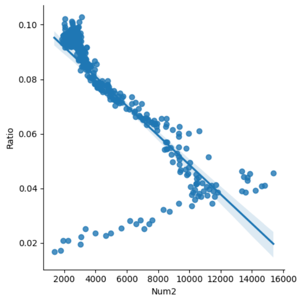

Figure 24: Regression plot between Num2 and Ratio

But we still didn't figure out what caused the variation of Num1 and Num2, we tried to plot them on a single figure against each other. In the previous Figure.7 and Figure.8 we found that the overall variation of both figures are similar, they all increase before Mar 2022, and then decrease. However, in Figure.25, we found that the amplitude of their variation is significantly different and the amplitude of variation for Num1 is much greater than that of Num2.

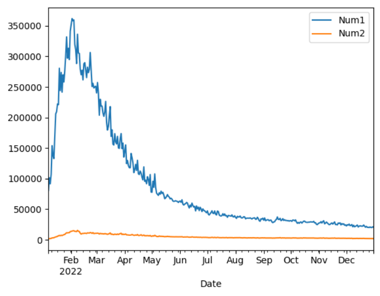

Figure 25: Plot Num1 and Num2 in a single graph

From Figure.9 the filtered ratio, We found that despite the fact that the variations of ratio of Num1 and Num2 are different, the ratio of Num2 divided by Num1 is always increasing, no matter Num2 is increasing or not. We have also noticed that the speed of increasing is decreasing. There are two ways for us to predict the variation of the ratio. We can simply predict the ratio by the ratio itself, or we can predict the ratio by predicting

Num1 and Num2. We decided to try both methods and see whether there are significant differences.

4 Word Difficulty Estimation Model

4.1 Word Attributes

4.1.1 Word Frequency

To measure word frequency, we first downloaded the dataset containing all the words from Wordle and dropped the noise value and columns. The remaining columns are the word list and data like Fre-Hal and Mean-RT. The index we choose is the I-Zscore, which is the main lexical decision latency for each word. We use Fre-Hal and Mean-RT to compute the index. However, we changed a little bit of the algorithm to make the scale larger from -1 to 1 instead of 0 to 1 to enlarge the differences between each word. After having the I-Zscore for each word, we filter the words with five letters with their I-Zscores repextively and output a csv file to save.

4.1.2 Letter Frequency

A dataset containing all the five-letter words that have appearred in Wordle from the Art of Problem Solving has been downloaded to calculate the frequency of each letter. We found out that 'e' is the most frequently appearing letter, with 10.67% as its percentage, while 'j' is the lowest one with percentage of 0.23%. For each word consisting of five letters, we refer to the table for calculation. We multiply the frequency of each letter. These numbers are later manipulated to be used as one of the metrics to classify the difficulty of the word. It is obvious that in this part, words with letters that occur more frequently would gain higher scores.

Letter	appearing times	frequency	Letter	appearing times	frequency
a	979	8.47	n	575	4.97
b	281	2.43	o	755	6.53
c	477	4.13	p	367	3.17
d	393	3.40	q	29	0.25
e	1233	10.67	r	899	7.78
f	230	1.99	s	669	5.79
g	311	2.70	t	729	6.31
h	389	3.37	u	467	4.04
i	671	5.80	v	153	1.32
j	27	0.23	w	195	1.69
k	210	1.82	x	37	0.32
l	719	6.22	y	425	3.68
m	316	2.73	z	40	0.35

Table 1: The appearance times of each letter and its frequency in %.

4.1.3 Letter Repetition

Our group uses a simple strategy to weigh the influence of repetition to the difficulty of the word. For a five-letter word, the possibility of repetition is 1, 2 and 3. Thus, it is convenient to give each time of repetition a score to measure the influence. when there exists maximum of two repetition for each character in the word, we assign 1 as the repetition value. Similarly, for three repetition among each character in the word, we assign 2 as the repetition value. For the word with no repetition occurring among characters, we recognize its value as 0 by default.

4.2 Word Difficulty Calculation

After obtaining the measuring indices of three features respectively, we export the csv file and transform to excel to measure the difficulty of each word. The general method is to assign weight to each feature score and sum them up to get the comprehensive score of word difficulty. For the letter frequency, our group first takes the (1/4)th power of frequency score of each word since the original scale and standard deviation is huge. Next, we add the negative sign in the front because higher letter frequency score means relatively easier for people to come up with the word. Since our goal is that higher difficulty scores indicate that the word is harder to guess, we generally intend to take the reverse of letter frequency. For the repetition value, we simply multiply the value by 5 as its weight. The reason accounting for the operation is that, considering relatively weak effects of repetition to the final word difficulty, the weight shall be smaller than weights assigned to other two feature scores. As to the last feature - word frequency, we first add two to the word frequency score to make each value positive. Then, we use the similar strategy to assign a weight of 15 to multiply the word frequency score because word frequency shows greater impact on word difficulty. Eventually, we get the difficulty score of each word by the formula

$$\text{difficulty score for each word} = (2 + x) \times 15 - \sqrt[4]{y} + z \times 5 \tag{1}$$

x = word frequency score for the word y = letter frequency score for the word z = character repetition value for the word

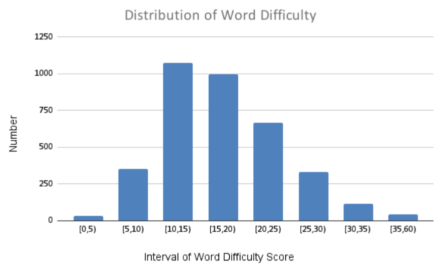

Figure 26: Heat Map of data frame

4.3 Conclusion

We make analysis of the data to show relevant statistics. The minimum score is 1.40 and the maximum is 55.06. The average difficulty score is 17 with the standard deviation of 6.65. From the Figure 26, the distribution of difficulty score is similar between right skewed distribution and normal distribution. Thus, for the word "EERIE", its word frequency score is 0.13, letter frequency score is 138299 and repetition score is 2. Thus, by the formula,

$$\text{difficulty score of "EERIE} = (2 + 0.13) \times 15 - \sqrt[4]{138299} + 2 \times 5 = 22 \tag{2}$$

We could find that The difficulty of word "ERRIE" is higher than the average and more precisely higher than 76 percent of words. But from the standard deviation, the word difficulty index of "EERIE" is relatively close to the central. Therefore, we could reach the result that the word "EERIE" is moderately difficult according to our model.

5 Univariate Time Series Forecasting Model

During the process of data exploration, we found that there is no obvious relationship between the difficulty of the word and the percentage of users choosing the hard mode. This is reasonable because the users do not know the word before the game has finished, let only its specific difficulty. To test our hypothesis, we decided to use the **Granger Casuality Test**. The test is designed to test whether one stationary time series can provide forecasting information to another. For our test, we have used Variable-lag Granger causality method (VL-Granger) and transfer entropy (VL-Transfer Entropy), with word difficulty levels and ratio of hard mode players as the two series. The test results rejects the hypothesis of correlations, and we also gained a negative -0.017 showing there was no predictive effect of word difficulty on ratio of hard mode players. Additionally, as for the reported number of all players, there is also no correlated data for model parameter. And thus we decided to use univariate time series forecasting model to predict the reported number of players and the number of hard mode players, for the sake of predicting the percentage of hard mode players among all players.

5.1 Autoregressive Integrated Moving Average (ARIMA)

An autoregressive integrated moving average, or ARIMA, is a statistical analysis model that uses time series data to either better understand the data set or to predict future trends. Considering time series data, an ARIMA model is given by

$$\left(1-\sum_{i=1}^{p'}\alpha_i L^i\right)X_t=\left(1+\sum_{i=1}^{q}\theta_i L^i\right)\varepsilon_t \tag{3}$$

Assume now that the polynomial

$$\left(1-\textstyle\sum_{i=1}^{p'}\alpha_i L^i\right) \tag{4}$$

has a unit root, a factor (1-L) of multiplicity d. Then it can be rewritten as:

$$\left(1-\sum_{i=1}^{p'}\alpha_i L^i\right)=\left(1-\sum_{i=1}^{p'-d}\varphi_i L^i\right)(1-L)^d \tag{5}$$

An ARIMA(p,d,q) process expresses this polynomial factorisation property with $p = p' - d$, and is defined as:

$$\left(1-\sum_{i=1}^{p}\varphi_i L^i\right)(1-L)^d X_t=\left(1+\sum_{i=1}^{q}\theta_i L^i\right)\varepsilon_t \tag{6}$$

The ARIMA model requires our time series to be stationary. There are many ways to determine the stationarity, such as line plot, ploting their moving statistics or use ADF or KPSS test. We apply all of them to Num1, Num2, and Ratio in order to get a more accurate results.

ADF test comes from DF test (Dickey-Fuller Test). It is a test for unit root and the t test for the hypothesis

$$\Delta y_t=\beta y_{t-1}+e_t \tag{7}$$

The ADF test improves the Dickey-Fuller test equation to include high order regressive process in the model.

$$\Delta y_t=c_0+c_1 t+\beta y_{t-1}+\sum_{i=1}^{p}\gamma_i\Delta y_{t-i}+e_t \tag{8}$$

The KPSS test assesses the null hypothesis that a univariate time series is stationary against the alternative that it is a nonstationary unit root process. The test uses the structural model

$$\begin{aligned} y_t &= c_t + \delta t + u_{1t} \\ c_t &= c_{t-1} + u_{2t} \end{aligned} \tag{9}$$

δ is the trend coefficient (see the Trend argument)
u_{1t} is a stationary process
u_{2t} is an independent and identically distributed process with mean 0 and variance σ_2

5.1.1 Determine the Stationarity of Time Series Data

From the previous picture of Num1, Num2 and Ratio, we can guess that they are not stationary. Then We calculate and plot the rolling mean and rolling standard deviation for Num1, Num2 and Ratio. Apparently, the rolling mean and rolling deviation vary significantly and thus the time series for Num1, Num2 and Ratio are not stationary process.

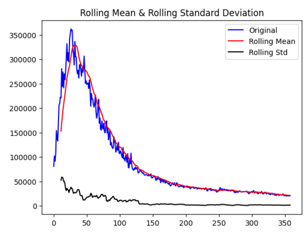

Figure 27: Rolling mean and standard deviation of Num1

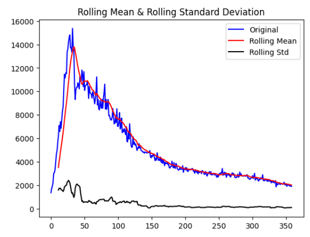

Figure 28: Rolling mean and standard deviation of Num2

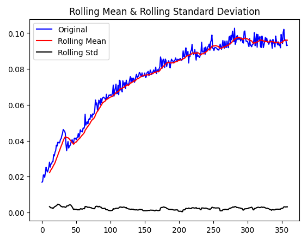

Figure 29: Rolling mean and standard deviation of Ratio

5.1.2 Make Time Series Stationary Using Differencing

Then we use difference methods to make their time series in a stationary way. The following figures show the line plot of Num1, Num2 and Ratio after differencing. We can find that their moving average and moving standard deviation become relatively stable. However, to make it more accurate, we use both ADF test and KPSS test to determine the order of differencing and when to stop differencing.

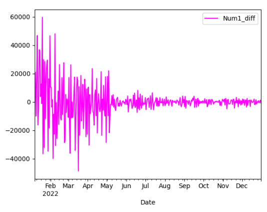

Figure 30: Line plot of Num1 after one differencing

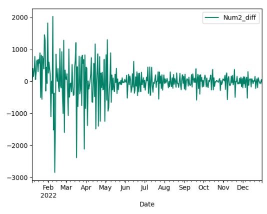

Figure 31: Line plot of Num2 after one differencing

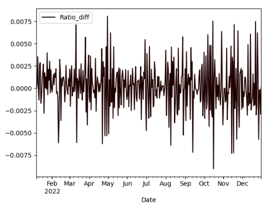

Figure 32: Line plot of Ratio after one differencing

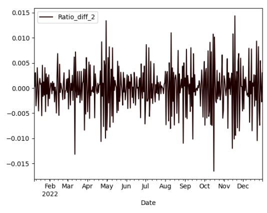

Figure 33: Line plot of Ratio after two differencing

For ADF test, we can determine whether a time series is stationary by comparing the test statistics with critical values and the p-value with 0.05. If the test statistics is less than the corresponding critical value and p-value is less than 0.05, we can reject the null hypothesis, which says the time series is non-stationary.

Similarly, for KPSS test, we can determine whether a time series is stationary by comparing the test statistics with critical values and the p-value with 0.05. If the test statistics is less than the corresponding critical value and p-value is larger than 0.05, we can reject the null hypothesis, which says the time series is non-stationary.

Num1	ADF		KPSS					
	test statistics	p-value	test statistics	p-value	cv 10%	cv 5%	cv 2.5%	cv 1%
before	-3.33	0.0137	2.23	0.01	0.347	0.463	0.574	0.739
after	-3.98	0.0015	0.17	0.1	0.347	0.463	0.574	0.739

Table 2: Num 1 test data.

Num2	ADF		KPSS					
	test statistics	p-value	test statistics	p-value	cv 10%	cv 5%	cv 2.5%	cv 1%
before	-1.82	0.3704	2.24	0.01	0.347	0.463	0.574	0.739
after	-4.22	0.0006	0.33	0.1	0.347	0.463	0.574	0.739

Table 3: Num 2 test data.

Ratio	ADF		KPSS					
	test statistics	p-value	test statistics	p-value	cv 10%	cv 5%	cv 2.5%	cv 1%
before	-3.73	0.0037	2.79	0.01	0.347	0.463	0.574	0.739
diff 1	-16.39	2.6991	0.83	0.01	0.347	0.463	0.574	0.739
diff 2	-9.24	1.5831	0.04	0.1	0.347	0.463	0.574	0.739

Table 4: Ratio test data.

The above tables show the test statistics of ADF and KPSS tests for Num1, Num2 and Ratio. We find that after diffferencing once, Num1 and Num2 become stationary for both tests and after differencing twice, Ratio become stationary for both tests. We will use the new stationary time series for later model training.

5.1.3 Determine the parameters for ARIMA Model

We use the ACF and PACF plot to determine the parameters for our ARIMA model. Autocorrelation is the correlation between two values in a time series. In other words, the time series data correlate with themselves. The autocorrelation function (ACF) assesses the correlation between observations in a time series for a set of lags. The partial autocorrelation function is similar to the ACF except that it displays only the correlation between two observations that the shorter lags between those observations do not explain. The partial autocorrelation function (PACF) is more useful during the specification process for an autoregressive model.

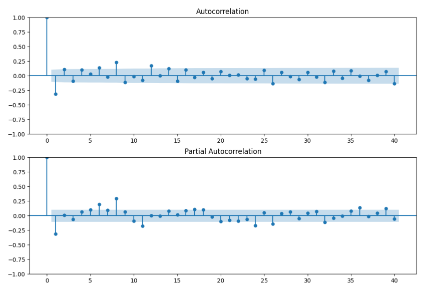

Figure 34: ACF and PACF of Num1 after one differencing

For the PACF plot of Num1, the first lag is significantly out of the limit and the second one is also out of the significant limit but it is not that far so we can select the order of the p as 1. For the ACF plot of Num1, 2 of the lags are out of the significance limit, but the second one is not that far so we can say that the optimal value of our q is 1. Since Num1 use first differencing to become stationary, d is 1.

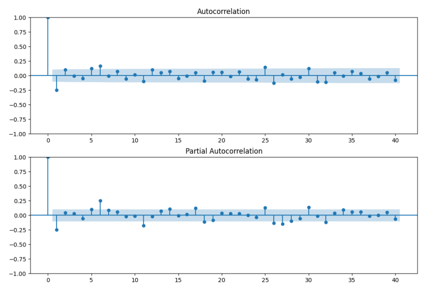

Figure 35: ACF and PACF of Num2 after one differencing

Similarly, for the PACF plot of Num2, the first lag is significantly out of the limit and the second one is also out of the significant limit but it is not that far so we can select the order of the p as 1. For the ACF plot of Num2, 2 of the lags are out of the significance limit, but the second one is not that far so we can say that the optimal value of our q is 1. Since Num2 use once differencing to become stationary, d is 1.

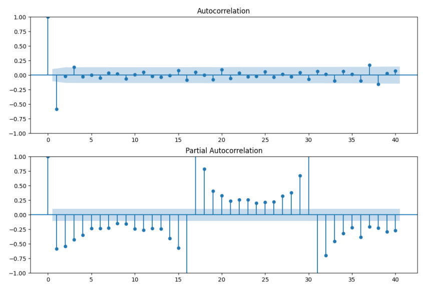

Figure 36: ACF and PACF of Ratio after two differencing

For the PACF plot, we can find that the first lag is significantly out of the limit and we can select the order of the p as 1. For the ACF plot of Ratio, 2 of the lags are out of the significance limit so we can say that the optimal value of our q is 2. Since Ratio use twice differencing to become stationary, d is 2.

5.1.4 Predict the Reported Number of Results and Percentage of Hard Mode

Then we apply the ARIMA model for Num1, Num2 and Ratio respectively. We divide the provided data set into training set and test set. We use the training set to train our model and the test set to test the accuracy of our model. The following figure show the line plot of real data against predicting data. We find that the prediction is quite accurate.

By prediction, the number of reported results on March 1st is 19548 and the number of hard mode is 1836. The Mean Square Error (MSE) for Num1 prediction is 1721.575 and the Normalized Mean Square Error (NMSE) is 0.005, which is calculated using the prediction value and actual value of test sets. The Mean Square Error (MSE) for Num2 prediction is 161.333 and the Normalized Mean Square Error (NMSE) is 0.001.

Therefore, the estimated percentage of hard mode P can be calculated by

$$P = \frac{1836}{19548} \times 100\% = 9.393\% \tag{10}$$

Additionally, we also use ARIMA model to directly predict the value of the percentage of hard mode, with MSE 0.003 and NMSE 0.03, which is 9.422%.

Therefore, we estimate that the percentage of hard mode on March 1st is in in the interval [9.3%, 9.5%]

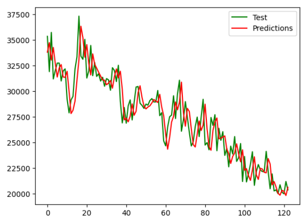

Figure 37: ARIMA forecasting of Num1

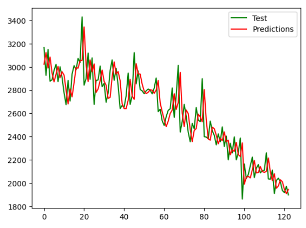

Figure 38: ARIMA forecasting of Num2

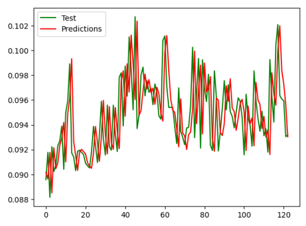

Figure 39: ARIMA forecasting of Ratio

6 Multivariate Time Series Forecasting Model

We want to predict the distribution of different tries on March 1st. Thinking intuitively,the distribution should be related to the percentage of hard mode because the hard mode makes the game harder and thus it's mode difficult to finish the game in less tries. Additionally, it should also relate to the difficulty of solution word because those esoteric word will definitely cost more tries. Therefore, we decided to use multivariate time series forecasting methods to take into account several different time series to predict the distribution.

6.1 Vector Autoregression (VAR)

Vector autoregression (VAR) is a statistical model used to capture the relationship between different time series data. VAR models generalize the single-variable (univariate) autoregressive model by allowing for multivariate time series. Like other time series forecasting models, the VAR model also requires that the input time series are stationary and thus we have to determine the stationarity of time series.

The structure is that each variable is a linear function of past lags of itself and past lags of the other variables. As an example suppose that we measure three different time series variables, denoted by $x_{t,1}$, $x_{t,2}$, and $x_{t,3}$. The vector autoregressive model of order 1 , denoted as VAR(1), is as follows:

$$x_{t,1} = \alpha_1 + \phi_{11}x_{t-1,1} + \phi_{12}x_{t-1,2} + \phi_{13}x_{t-1,3} + w_{t,1} \tag{11}$$

$$x_{t,2} = \alpha_2 + \phi_{21}x_{t-1,1} + \phi_{22}x_{t-1,2} + \phi_{23}x_{t-1,3} + w_{t,2} \tag{12}$$

$$x_{t,3} = \alpha_3 + \phi_{31}x_{t-1,1} + \phi_{32}x_{t-1,2} + \phi_{33}x_{t-1,3} + w_{t,3} \tag{13}$$

Each variable is a linear function of the lag 1 values for all variables in the set.

In a VAR(2) model, the lag 2 values for all variables are added to the right sides of the equations, In the case of three x-variables (or time series) there would be six predictors on the right side of each equation, three lag 1 terms and three lag 2 terms.

In general, for a VAR(p) model, the first p lags of each variable in the system would be used as regression predictors for each variable.

6.1.1 Stationarity of Time Series Data

From the previous data analysis, we believe that the total reported number of results, the number of hard mode, their percentage and word difficulty along with the distribution of all of the other tries may influence the distribution of different tries. And therefore, we use the above 11 columns to train our VAR model.

From the last section, we also know that the series of Num1, Num2 and their ratios are not stationary. In fact, Num1 and Num2 are stationary after one time of diffrencing and their ratio is stationary after two times of differencing. Therefore, we use the differenced series for model training instead of themselves. As for the time series for word difficulty, we checked its stationarity using ADF test and it turned out to be stationary, which also coresponds to the intuition. We also test the time series for different tries to make sure they are stationary under ADF test.

6.1.2 Predict the Distribution of Different Tries

We divided the above series into two parts, the training set and the test set. The training set is used to train our model and the test set is used to evaluate the accuracy of our model. By prediction, the percentage of reaching success on the first try is 0.0475%, on the second tries is 5.6561%, 23.9448% for the third try, 35.4033% for the fourth try, 22.8811% for the fifth try, 9.2509% for the sixth try, and the percentage of 7 or more tries is 2.4153%.

The prediction for the percentage of different tries is shown in the following figures.

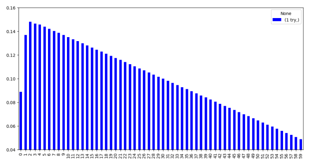
Figure 40: Prediction of percentage of 1 try in %

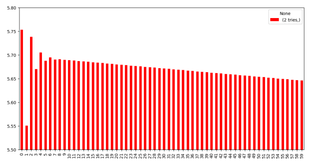
Figure 41: Prediction of percentage of 2 tries in %

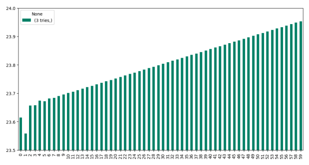
Figure 42: Prediction of percentage of 3 tries in %

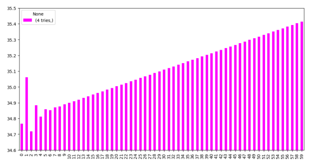
Figure 43: Prediction of percentage of 4 tries in %

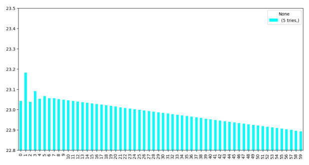
Figure 44: Prediction of percentage of 5 tries in %

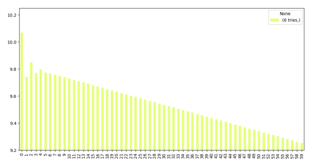
Figure 45: Prediction of percentage of 6 tries in %

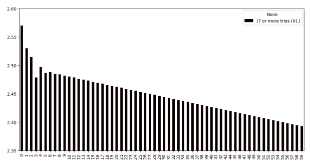
Figure 46: Prediction of percentage of 7 or more tries in %

The stacked bar graph is shown below.

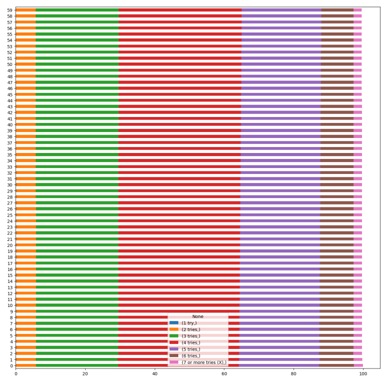

Figure 47: Stacked bar plot of predictions

6.1.3 Analysis of Model Accuracy by Mean Square Error

The RMSE for 1 try is 1.293, the RMSE for 2 tries is 4.778, the RMSE for 3 tries is 8.943, the RMSE for 4 tries is 5.616, the RMSE for 5 tries is 6.663, the RMSE for 6 tries is 6.310, the RMSE for 7 or more tries is 3.081.

7 Strengths and Weaknesses

7.1 Strengths

1. Our model successfully answers the question of the problem statement, predicting the number of reported results, the percent of hard mode and the different distributions. Meanwhile, we successfully construct a word difficulty model which can distinguish word by difficulty and also corresponds to our intuition.

2. From the calculation of Mean Square Error and Normalized Mean Square Error, our prediction using univariate time series forecasting for the number of reported results and the percent of hard mode are relatively accurate.

3. We use many statistical tests to evaluate the correctness of whether we select the right model and time series. For example, we use Granger Casuality Test to test the casuality between different time sseries. And we use ADF and KPSS test to test the stationarity of time series. Finally, we use ACF and PACF measures to determine the parameters for ARIMA model.

7.2 Weaknesses

1. Due to time limit, we are not able to implement different forecasting methods and cross compare their accuracy. We also didn't further optimize our current model.

2. For the multivariate time series forecasting problem, Vector Autoregression might not be the optimal choice because the forecasting results is not satisfying. If time permits, we would use deep learning model such as Long Short Term Memory to forecast it again and see whether there are better results.

References

[1] COMAP 2023. Problem C Data Set. 2023

[2] COMAP 2023. Problem C Problem Statement. 2023

[3] Hobijn, B., Franses, P. H., & Ooms, M. (2004). Generalizations of the KPSS-test for stationarity. Statistica Neerlandica, 58(4), 483-502.

[4] Koustubhk. (2020, October 12). Word difficulty prediction. Kaggle. Retrieved February 20, 2023, from https://www.kaggle.com/datasets/kkhandekar/word-difficulty

[5] Seth, A. (2007). Granger causality. Scholarpedia, 2(7), 1667.

[6] TeXeR. (n.d.). Wordle "word of the day". Art of Problem Solving. Retrieved February 20, 2023, from https://artofproblemsolving.com/texer/vxeinejf

Appendices

Appendix A Letter to the Puzzle Editor of the New York Times

DATE: February 2023
TO: New York Times
FROM: Modeling Team 2300879
SUBJECT: Predicting Model for Wordle

Our team was assigned to create a model that would predict the possible number of players of Wordle on March 1st, 2023. We were also asked to predict the proportion of hard-mode players, the distribution of times of try, as well as to create a model to categorize the word difficulty. We were given the player dataset of Jan 7, 2022 to Dec 31, 2022, including information such as the distribution of number of tries, player numbers etc..

In the process of data cleaning, we noticed that there were some outliers in the data and used Hampel Filter to replace them with more representative values. After doing this, we did the line plot and confirmed that there were no significant data deviation. Then, in the process of data exploration, we focused on the possible correlation between Num1 (number of players), Num2(number of players that chose hard mode), and their ratio. We want to predict the ratio in two ways: by both predicting Num1 and Num2 and do the division and directly predicting the ratio.

We have three metrics to classify the difficulty of words, which are word frequency, letter frequency, and character repetition. Basically, words with lower usage frequency, contain letters with lower usage frequency, and contain more letter repetition would be classified as more difficult. While the former two metrics are easier to understand, the third metric is set up due to the premise of the game. If there are more repeating letters in one word, the total number of differing letters would be less. Therefore, it would be harder for a user to guess the word arbitrarily. We then assign different weights to each of the metrics and calculate the score.

We used the Granger Casuality Test to decide word difficulty has no forecasting effects on the percentage of users that chose the hard mode. Tested with ADF and KPSS tests, we used Differencing to make our data stationary and then used autoregressive integrated moving average(ARIMA) to predict the future trends of Num1, Num2, and the ratio. Then we used the vector autoregression (VAR) to predict the distribution of different tries.

To summarize, we have tried several different methods to make our data have predictable effects, and use some time series analysis models to predict future scenarios. There are still weaknesses of our paper due to the time limit, but we are happy to further discuss the possible improvement of our model. Feel free to let us know if you have any questions.

Best,
Modeling Team 2300879

Appendix B Python Code

```
# split into train and test sets
X = series.values
size = int(len(X) * 0.66)
train, test = X[0:size], X[size:len(X)]
history = [x for x in train]
predictions = list()

# walk-forward validation
for t in range(len(test) + 60):
 model = ARIMA(history, order=(2,1,1))
 model_fit = model.fit()
 output = model_fit.forecast()
 yhat = output[0]
 predictions.append(yhat)
 if t < len(test):
        obs = test[t]
 else:
    obs = predictions[len(predictions) - 2]
 history.append(obs)
 print('predicted=%f, expected=%f' % (yhat, obs))

# plot forecasts against actual outcomes
plt.plot(test, color = 'green', label = 'Test')
plt.plot(predictions, color='red', label = 'Predictions')
plt.legend()
plt.show()

data = df.drop(['Word', 'Date', 'Num1', 'Num2', 'Ratio', 'Ratio_diff'], axis = 1)
cols = data.columns

#creating the train and validation set
train = data[:int(0.8*(len(data)))]
valid = data[int(0.8*(len(data))):]

#fit the model
from statsmodels.tsa.vector_ar.var_model import VAR
model = VAR(endog=train)
model_fit = model.fit()

# make prediction on validation
prediction = model_fit.forecast(y = model_fit.endog, steps=len(valid))
#converting predictions to dataframe
pred = pd.DataFrame(index=range(0,len(prediction)),columns=[cols])
for j in range(0,12):
    for i in range(0, len(prediction)):
       pred.iloc[i][j] = prediction[i][j]

#check rmse
for i in cols:
    print('rmse value for', i, 'is : ', sqrt(mean_squared_error(pred[i], valid[i])))

#make final predictions
model = VAR(endog=data)
model_fit = model.fit()
yhat = model_fit.forecast(y = model_fit.endog, steps=60)
pred1 = pd.DataFrame(index=range(0,len(yhat)),columns=[cols])
for j in range(0,12):
    for i in range(0, len(yhat)):
       pred1.iloc[i][j] = yhat[i][j]
print(yhat)
```