Problem Chosen	2023 MCM/ICM Summary Sheet	Team Control Number
C		2311332

Regression-Based Word Trials Distribution and Difficulty Classification Model

Summary

Question 1 is asking to develop a model to explain this variation and use the model to create a prediction interval for the number of reported results on March 1, 2023. Determine whether there are any attributes of the word affect the percentage of scores reported that were played in Hard Mode. For question 1, we fit the number of people in the report and the date with exponential regression, and add a confidence interval to obtain a prediction interval. At the same time, we calculated the correlation between the attributes of the word and the proportion of the number of people reporting in the hard mode, and thus came to the conclusion that none of the attributes we found has significant influence on the proportion of the number of people reporting in the hard mode. We think that this is reasonable, because we believe that the percentage of scores reported that were played in Hard Mode is proportional to the number of players who play the game in Hard Mode on that day. However, when choosing whether to play the game in Hard Mode or not, the players do not know anything about the word on that day, so the attributes of the word have no reason to affect this choice.

Question 2 is asking to develop a model to predict the distribution of the reported results. For question 2, we divide the attributes of words into three categories, and use time as one of the independent variables to establish a multiple linear regression model, and finally get a model that can predict the number of correct word attempts.

Question 3 is asking to develop and summarize a model to classify solution words by difficulty. Identify the attributes of a given word that are associated with each classification. For question 3, we use the number of correct word attempts as an indicator for dividing the difficulty of words, and use this indicator to divide the difficulty of words. For a new word, we only need to predict the distribution of the number of correct word attempts based on the model in question 2, and then obtain the difficulty based on the distribution, which means the solution is successful.

Finally, we synthesize the above three models and write a summary report.

Keywords: exponential regression fit; multiple linear regression

Contents

1 Introduction

1.1 Background

Wordle is a popular puzzle currently offered daily by the New York Times. Players try to solvethe puzzle by guessing a five-letter word in six tries or less, receiving feedback with every guess. For this version, each guess must be an actual word in English. Guesses that are not recognized as words by the contest are not allowed. Wordle continues to grow in popularity and versions of the game are now available in over 60 languages.

The New York Times website directions for Wordle state that the color of the tiles will change after you submit your word. A yellow tile indicates the letter in that tile is in the word, but it is in the wrong location. A green tile indicates that the letter in that tile is in the word and is in the correct location. A gray tile indicates that the letter in that tile is not included in the word at all.

Players can play in regular mode or "Hard Mode." Wordle's Hard Mode makes the game more difficult by requiring that once a player has found a correct letter in a word (the tile is yellow or green), those letters must be used in subsequent guesses. The example in Figure 1 was played in Hard Mode.

Many (but not all) users report their scores on Twitter. For this problem, MCM has generated a file of daily results for January 7, 2022 through December 31, 2022. This file includes the date, contest number, word of the day, the number of people reporting scores that day, the number of players on hard mode, and the percentage that guessed the word in one try, two tries, three tries, four tries, five tries, six tries, or could not solve the puzzle (indicated by X).

1.2 Questions

This contest requires solutions to several questions.

• Develop a model to explain this variation and use the model to create a prediction interval for the number of reported results on March 1, 2023. Determine whether there are any attributes of the word affect the percentage of scores reported that were played in Hard Mode.

• Develop a model 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. Give a specific example of the prediction for the word EERIE on March 1, 2023.

• Develop and summarize a model to classify solution words by difficulty. Identify the attributes of a given word that are associated with each classification.

1.3 Solutions

For question 1, we fit the number of people in the report and the date with exponential regression, and add a confidence interval to obtain a prediction interval. At the same time, we calculated the correlation between the attributes of the word and the proportion of the number of people reporting in the hard mode, and thus came to the conclusion that none of the attributes we found has significant influence on the proportion of the number of people reporting in the hard mode. We think that this is reasonable, because we believe that the percentage of scores reported that were played in Hard Mode is proportional to the number of players who play the game in Hard Mode on that day. However, when choosing whether to play the game in Hard Mode or not, the players do not know anything about the word on that day, so the attributes of the word have no reason to affect this choice.

For question 2, we divide the attributes of words into three categories, and use time as one of the independent variables to establish a multiple linear regression model, and finally get a model that can predict the number of correct word attempts. After testing, the credibility of the model higher degree.

For question 3, we use the number of correct word attempts as an indicator for dividing the difficulty of words, and use this indicator to divide the difficulty of words. For a new word, we only need to predict the distribution of the number of correct word attempts based on the model in question 2, and then obtain the difficulty based on the distribution, which means the solution is successful.

Finally, we synthesize the above three models and write a summary report.

2 Analysis of the Problem

2.1 Problem 1: The number of the reported results

Problem 1 requires a model to explain the variation of the number of reported results. And this model will be used to create a prediction interval for the number of reported results on March 1, 2023.

In addition, it would like to determine whether there are any attributes of the word affect the percentage of scores reported that were played in Hard Mode.

The first question essentially asks to build a regression model based on the independent variable **date** and the dependent variable **the number of the reported results**, so that the future situation can be predicted according to the model, and a reasonable prediction interval is given.

The second question is essentially asking the correlation between each **attribute** of the word and the **hardmode report percentage**.

2.2 Problem 2: The distribution of the reported results

Problem 2 requires to predict the associated percentages of (1, 2, 3, 4, 5, 6, X) for a future date, hence the most important thing is to find all the variables contributing to the percentages or distribution. We can mainly consider these variables, which is similar to the problem 1 but in addition, the time also needed to be considered.

$$\begin{cases} \text{1. daily frequency of a word} \\ \text{2. repetition of letters in a word} \\ \text{3. similarity between a word and other words} \\ \text{4. date} \end{cases}$$

According to the requirements of the topic, we can consider using multiple linear regression models. In regression analysis, if there are two or more independent variables, it is called multiple regression. In fact, a phenomenon is often associated with multiple factors, and the optimal combination of multiple independent variables to predict or estimate the dependent variable is more effective and more realistic than using only one independent variable to predict or estimate. Therefore, the practical significance of multiple linear regression is greater than that of single linear regression.

Once the regression model is derived, we use it to predict the distribution.

2.3 Problem 3: The classification of solution words by difficulty

Problem 3 requires us to develop and summarize a model to classify solution words by difficulty. According to the title, the difficulty of a word is mainly defined by the distribution of the number of answers, so it is considered to extract the elements in the distribution of the number of answers as an index of word difficulty. Then borrow the model for each attribute of the word in the second question to predict the difficulty of the new word.

3 Models and Results

3.1 Problem 1

3.1.1 Exponential regression fitting to predict the number of the reported results

Taking the date as x and the number of the reported results as y to make a $y-x$ relationship diagram, it can be concluded that the change trend of the number of reports is a sharp increase at

first, and then a significant decline after reaching the peak, and it shows a trend of slowing down. Therefore, we consider that the significant growth at the beginning is due to a large number of netizens were attracted by curiosity when the game was first launched, and after the peak, the rate of decline slowed down. The primitive relation is shown in **Figure 1**.

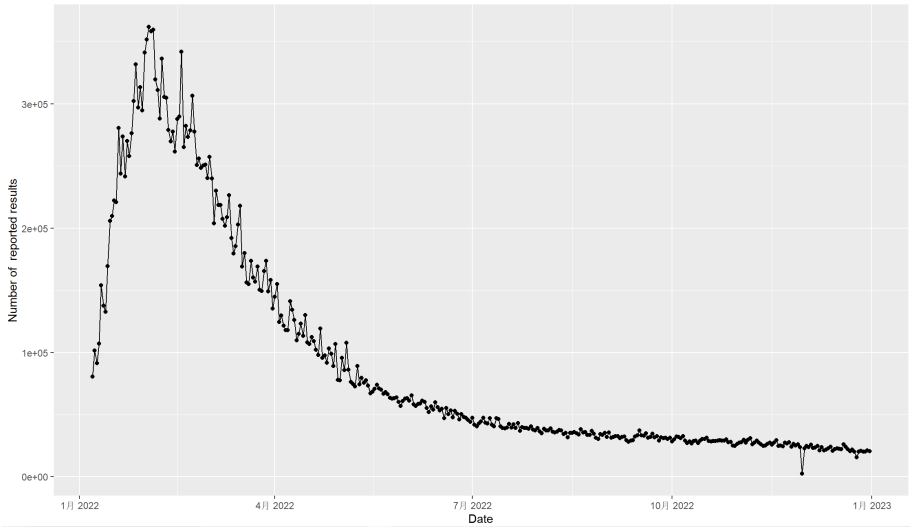

Figure 1: The mapping relation between the number of the reported results and the date

From this figure we see that there is an outlier when the contest number equals to 529, hence this data should be moved. Choose the data with the column number ≥ 115 and do exponential regression fitting.

When fitting a model, estimated parameter values and covariance matrices are usually available. The covariance matrix reflects the uncertainty in the estimated parameter values, and the standard error is a common measure of this uncertainty.

For a linear regression model, the estimated parameter values can be solved by the least squares method, and the covariance matrix can be calculated from the estimated residual sum of squares, explanatory variables, and sample size. In this case, for each parameter estimate, we can quantify its uncertainty by obtaining its standard error via the covariance matrix.

In this exponential fitting model, the $nls()$ function is used for fitting, which returns an object containing the model parameter estimates and the covariance matrix.

The function needed to be fitted is considered as:

$$y = \mathrm{e}^{a \cdot x + b} + c$$

After fitting we get the function fit, use the function to predic y for each x, then calculate the standard error at each x value. Standard error is an indicator used to describe the reliability of an estimator, and it is usually used to measure the difference between a sample statistic and a population parameter. In regression analysis, standard errors are often used to calculate confidence intervals

and significance tests. Details are as follows:

$$\begin{cases} \text{y\_new} = predict(fit) \\ \text{resid} = y - predict(fit) \\ \text{s} = \text{standard(resid)} \\ \text{mean\_resid} = \text{mean(resid)} \\ \text{standard\_error} = s * \sqrt{1 + 1/\text{length(x)} + \frac{(\text{x}-\text{mean(x)})^2}{\text{sum}((\text{x}-\text{mean(x)})^2)}} \\ \text{free} = \text{length(x)} - 2 \\ \text{t\_val} = \text{qt}(0.025, \text{free}) \\ \text{lower} = \text{y\_new} - \text{t\_val} * \text{se} \\ \text{upper} = \text{y\_new} + \text{t\_val} * \text{se} \end{cases}$$

and add the confidence interval to the new data frame, finally get the value and confidence interval at the point March 1, 2023, to get the prediction interval for the number of reported results.

3.1.2 Results of problem 1.1

The exponential regression fitting results to the data are as follows. Among them, with regard to the uncertainty measurement of parameters a, b, and c, for each parameter estimate, we obtain its standard error through the covariance matrix, thereby quantifying its uncertainty.

	Estimate	Std. Error	t value	Pr(>|t|)
a	-1.570e-02	5.356e-04	-29.32	<2e-16 ***
b	1.601e+01	1.722e-01	92.95	<2e-16 ***
c	2.311e+04	5.438e+02	42.49	<2e-16 ***

After this, normalize the raw data, calculate the standard error at each value of x, and use the qt() function to look up the value of t in the t-distribution table. We calculate the sample mean and degrees of freedom, and finally draw the confidence interval curve, which is shown in Figure 2.

The predicted interval for the number of the reported scores on March 1, 2023 is **[17302, 29975]**, which is a 95% confidence interval. The middle number of it is **23639**, which is the expected value.

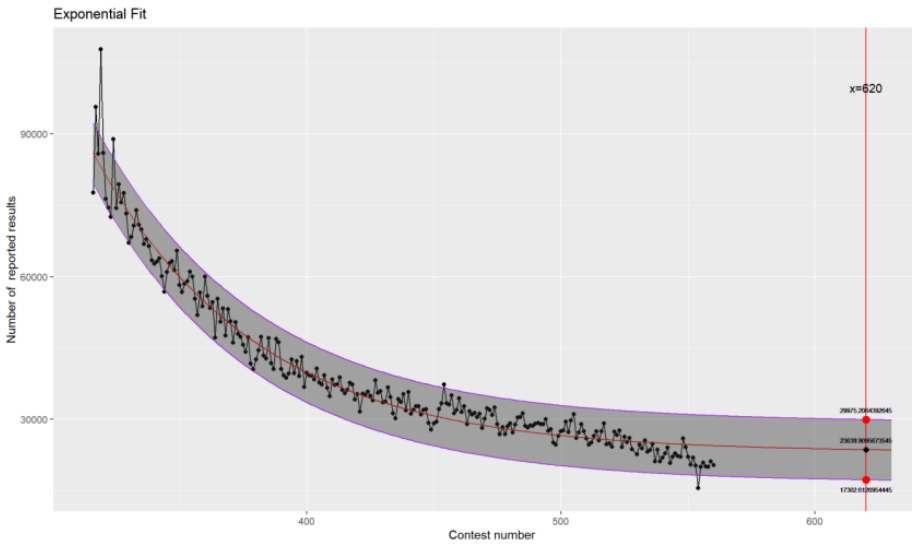

Figure 2: 95% confidence interval

3.1.3 Correlation test to explore attributes of words

Considering the specificity of words, we set the attributes of the words as:

$$\begin{cases} 1.\ \text{word frequency} \\ 2.\ \text{number of unique letters in the word} \\ 3.\ \text{similarity between the word and other words} \end{cases}$$

For the first attribute, the word frequency is an important variable in cognitive processing. Word of high-frequency are perceived and produced faster and more efficiently than words of low-frequency. At the same time, they are easier to recall but more difficult to recognize in episodic memory tasks. For the daily frequency of a word, we get the frequency data from the website SUBTLEXus .In Van Heuven, Mandera, Keuleers, & Brysbaert (QJEP, 2014) we proposed a new frequency measure, the Zipf scale, which is much easier to understand than the usual frequency measures. Zipf values range from 1 to 7, with the values 1-3 indicating low-frequency words (with frequencies of 1 per million words and lower) and the values 4-7 indicating high-frequency words (with frequencies of 10 per million words and higher). For each valid word, We explored the correlation between word frequency and the proportion of hard mode reports and found a small correlation **0.08641802**, which shows that the word frequency does not significantly affect the percentage of scores reported that were played in Hard Mode.

For the second attribute, the number of unique letters in a word may affect the result of the game since repetition may add to the difficulty by leading to more letters to choose from next time. For this attribute, we define a function called $uniqueness$, which shows how many unique letters in a word, e.g. uniqueness(happy) = 4, while uniqueness(tepid) = 5. Then we explored the corre-

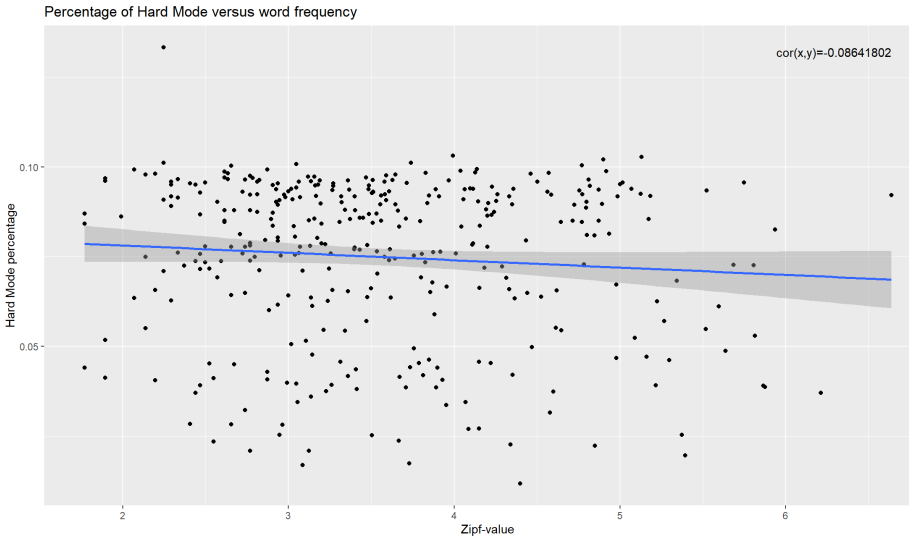

Figure 3: percentage of Hard Mode versus word frequency

lation between word uniqueness and the proportion of hard mode reports and found a correlation **0.07373057**, which shows that the number of unique letters in a word may not have significant effect on the percentage of scores reported that were played in Hard Mode.

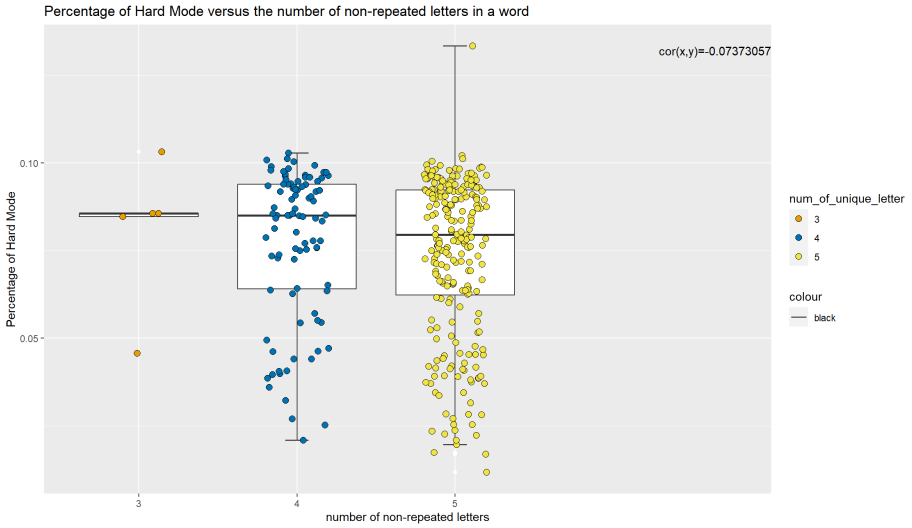

Figure 4: percentage of Hard Mode versus the number of non-repetited letters in a word

For the third attribute, we calculate one word's similarity to another one by function $num\_same()$, the result is the number of the letter at same position by other words by the sum of the number of the same letter at same position. For example, in a word set { happy, lucky, puppy}, the similarity between "happy" and "lucky" is $2^1 = 1$, and the similarity between "happy" and "puppy" is $2^3 = 8$, then in this set, "happy"'s similarity to other words is $1 + 8 = 9$. Build a letter set with all the five-letter words and calculate all words' similarity, then explored the correlation between word uniqueness and the proportion of hard mode reports and found a correlation **0.04626648**, which shows that the similarity between a word and other words may not have significant effect on the

percentage of scores reported that were played in Hard Mode.

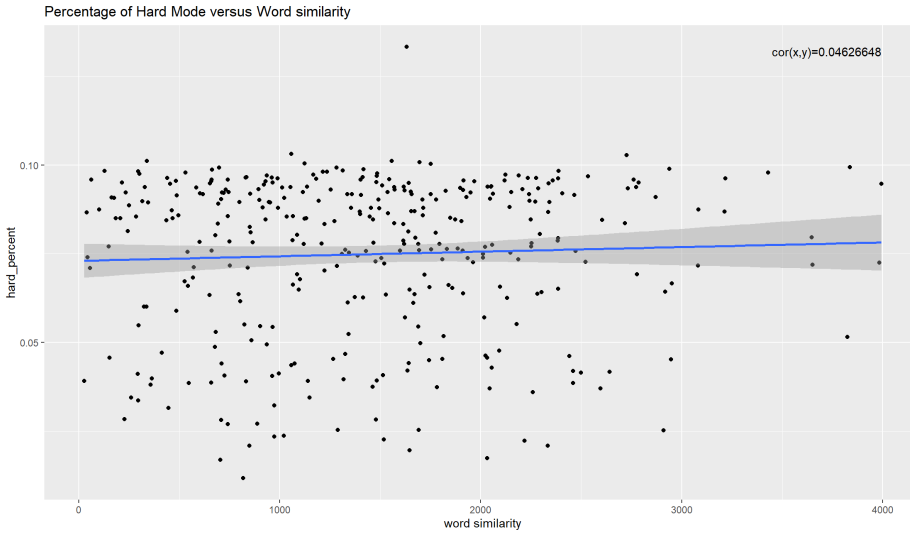

Figure 5: percentage of Hard Mode versus word similarity

3.1.4 Results of problem 1.2

In conclusion, the correlation measuring results are as follows. It shows that there is **no attribute of the word** affect the percentage of scores reported that were played in Hard Mode.

Content	Correlation coefficient
percentage of hard mode - word frequency	0.08641802
percentage of hard mode - number of unique letters	0.07373057
percentage of hard mode - similarity	0.04626648

In summary, we did not find any attributes of the word which significantly affect this percentage. We think that this is reasonable, because we believe that the percentage of scores reported that were played in Hard Mode is proportional to the number of players who play the game in Hard Mode on that day. However, when choosing whether to play the game in Hard Mode or not, the players do not know anything about the word on that day, so the attributes of the word have no reason to affect this choice.

3.2 Problem 2

3.2.1 Multiple linear regression forecasting model for distribution

First, we calculated the relationship between the distribution and time from one to six times, and performed a linear regression analysis of a single variable (contest number) on a multivariate

(a matrix composed of a ”distribution” distribution). In the table below, V2 refers to similarity. z refers to word frequenc. num_unique refer to number of unique letters in the word. time refers to contest number.

one	Estimate	Std. Error	t value	Pr(>|t|)
V2	-2.735e-05	4.924e-05	-0.555	0.57903
z	2.074e-01	4.165e-02	4.981	1.00e-06 ***
num_unique	3.240e-01	8.018e-02	4.041	6.56e-05 ***
time	-1.634e-03	3.766e-04	-4.339	1.89e-05 ***

two	Estimate	Std. Error	t value	Pr(>|t|)
V2	1.471e-04	2.379e-04	0.618	0.537
z	1.341e+00	2.012e-01	6.663	1.07e-10 ***
num_unique	2.794e+00	3.874e-01	7.213	3.53e-12 ***
time	4.376e-04	1.820e-03	0.240	0.810

three	Estimate	Std. Error	t value	Pr(>|t|)
V2	-4.685e-04	4.563e-04	-1.027	0.3053
z	1.916e+00	3.859e-01	4.966	1.08e-06 ***
num_unique	6.519e+00	7.430e-01	8.774	< 2e-16 ***
time	7.685e-03	3.490e-03	2.202	0.0283 *

four	Estimate	Std. Error	t value	Pr(>|t|)
V2	-0.0008076	0.0003388	-2.384	0.0177 *
z	-0.5329498	0.2865252	-1.860	0.0637
num_unique	0.9517038	0.5516177	1.725	0.0854
time	0.0114989	0.0025913	4.438	1.23e-05 ***

five	Estimate	Std. Error	t value	Pr(>|t|)
V2	2.455e-06	3.488e-04	0.007	0.9944
z	-1.716e+00	2.950e-01	-5.816	1.38e-08 ***
num_unique	-4.933e+00	5.679e-01	-8.687	< 2e-16 ***
time	-4.470e-03	2.668e-03	-1.676	0.0947

six	Estimate	Std. Error	t value	Pr(>|t|)
V2	0.0005504	0.0003745	1.470	0.142555
z	-1.1158347	0.3167114	-3.523	0.000484 ***
num_unique	-4.2614406	0.6097321	-6.989	1.45e-11 ***
time	-0.0118930	0.0028643	-4.152	4.16e-05 ***

more	Estimate	Std. Error	t value	Pr(>|t|)
V2	0.0004515	0.0002213	2.040	0.042133 *
z	-0.1589039	0.1871877	-0.849	0.396528
num_unique	-1.3096702	0.3603733	-3.634	0.000322 ***
time	-0.0031946	0.0016929	-1.887	0.059994

It was found that time and Pr(>|t|) of 1, 4, and 6 have a larger Signif., indicating that time is indeed one of the factors affecting the distribution. Time has also become a factor like, similarity, the number of unique letters in the word, and word frequency, so add time to the variables of the multiple linear regression model.

First, perform basic processing on the data, such as removing the data with a large offset and removing words whose length is not 5. After word processing, 348 valid words remain from 359 words.

The similarity, the number of unique letters in the word, word frequency, and time are used as independent variables, and the data frame composed of cbind(one,two,three,four,five,six,more) is used as dependent variables, and lm() linear regression and prediction are performed.

Then we calculating 95% confidence intervals for multiple linear regression models.

3.2.2 Results of problem 2

The linear regression results are as follows:

Attempts	1	2	3	4	5	6	7
Coefficient	-0.267	2.877	19.173	36.334	27.266	11.854	2.483

The result of solving for the word ”eerie” is as follows:

Attempts	1	2	3	4	5	6	7
Percentage	0	3	19	36	27	12	2

3.2.3 Validation

In order to verify the reliability of Model 2, we plotted the residuals of multiple linear regression.

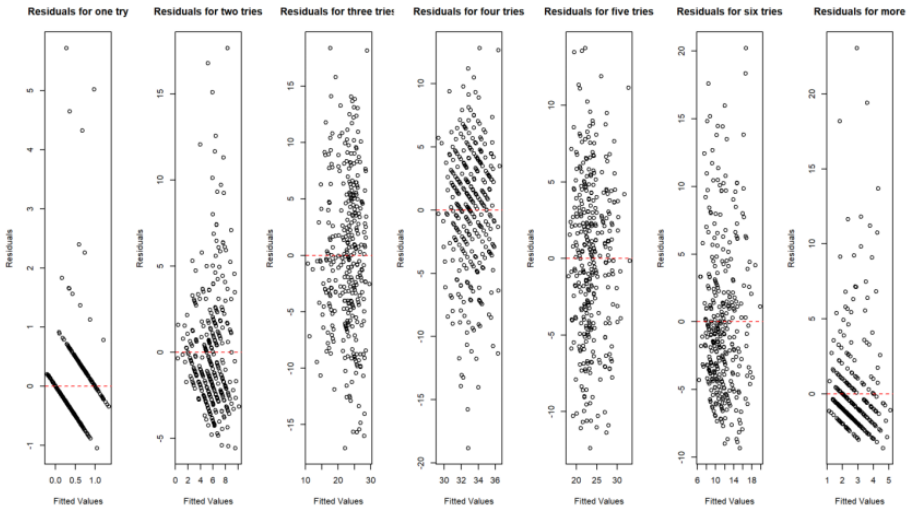

Figure 6: the residuals of multiple linear regression

According to the residual plot, our model has some systemmetic error and is not accurate enough.

3.3 Problem 3

3.3.1 Multiple Linear Regression Classification Model

Similarly, we preprocess the word data similar to the second question, and thus build a difficulty classification model. Arrange the processed data according to the average number of rounds, and divide the data into five equal parts. Specifically, for 348 rounds, the classification method of 69, 70, 70, 70, 69 can be used to obtain the numerical classification of difficulty.

After eliminating the time variable with the second question multiple linear regression model, insert the word to be predicted, calculate the distribution of the number of attempts, and test its validity. Then use this distribution data to obtain the average number of rounds, put it into the numerical classification model of difficulty, and obtain the difficulty of the word.

To Identify the attributes of a given word that are associated with each classification. We solve the correlation between the hard classification and the similarity between a word and other words, word frequency, and the number of unique letters in the word, and finally get the result.

3.3.2 Results of problem 3

Establish the average number of rounds E(x) to measure the difficulty, replace "X" by 7, arrange the processed data according to the average number of rounds, and divide the data into five equal parts, the results are as follows.

Difficulty	1	2	3	4	5
Interval	3.10-3.89	3.90-4.06	4.07-4.25	4.26-4.50	4.51-5.84

Use the multiple linear regression model of the second question, put the attributes of "EERIE" into the number of predicted rounds, and get the number of rounds as follows.

Attempts	0	1	2	3	4	5	6	more
Percentage	1	0	1	10	31	33	20	6

Calculate the average number of rounds on the distribution data of the number of attempts to obtain 4.83. According to the difficulty level, the final result is that the word "EERIE" has a difficulty of 5, which is the hardest.

To identify the attributes of a given word that are associated with each classification. We obtained the linear correlation coefficient between difficulty classification and similarity as 0.02437253, obtained the linear correlation coefficient between difficulty classification and word frequency as -0.2270678, and obtained the linear correlation coefficient between difficulty classification and the number of unique letters as -0.3817751.

Content	Correlation coefficient
hard classification - similarity	0.02437253
hard classification - word frequency	-0.2270678
hard classification - number of unique letters in the word	-0.3817751

This data shows that difficulty has little to do with similarity, but difficulty has a negative correlation with word frequency, and the number of unique letters in the word. The higher the word frequency, the simpler the word is. The more unique letters in the word, the simpler it is. The detailed figures for analyzing are shown in Figure 6, 7, 8.

4 Other Interesting Features of this Data Set

Success requires the average skewness and average kurtosis of the bureau number distribution data to be 0.3026415, -1.748139, respectively. For skewness, 0.3026415>0, indicating that the right

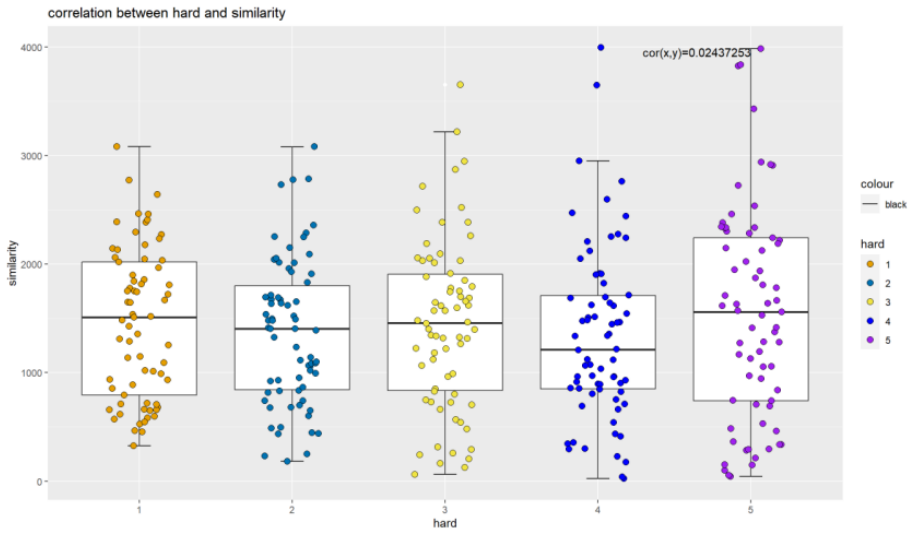

Figure 7: correlation between hard and similarity

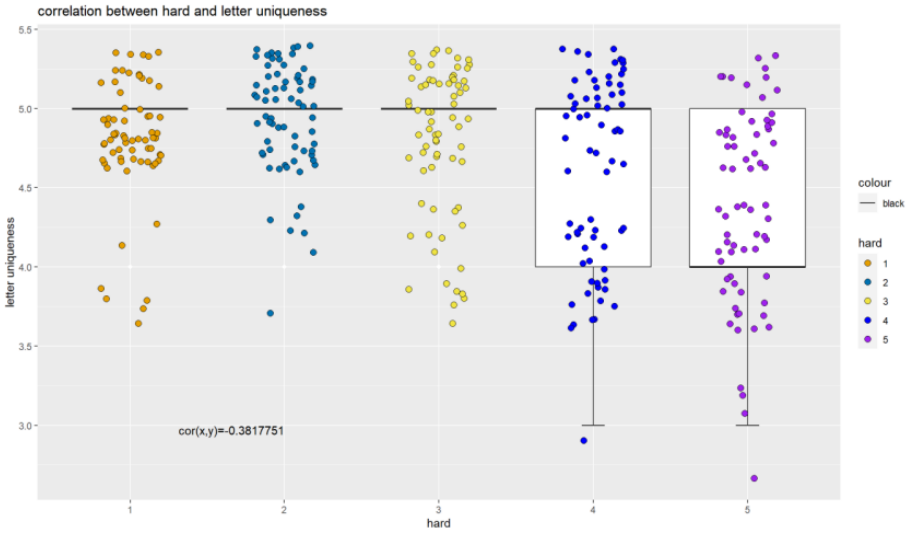

Figure 8: correlation between hard and letters uniqueness

tail is longer than the left, the data distribution is skewed to the right, and the mean is biased to the right of the data. Success requires that the game number distribution data be a positively skewed data distribution. For kurtosis, -1.748139<0, indicating that the kurtosis of the data is lower than that of the normal distribution, that is, the distribution of the data is relatively flat. This information can help us understand the distribution characteristics of the game winning rate data, and then adjust and optimize the winning rate prediction model. These are the interesting aspects of the data.

5 Summary Letter

Dear Puzzle Editor,

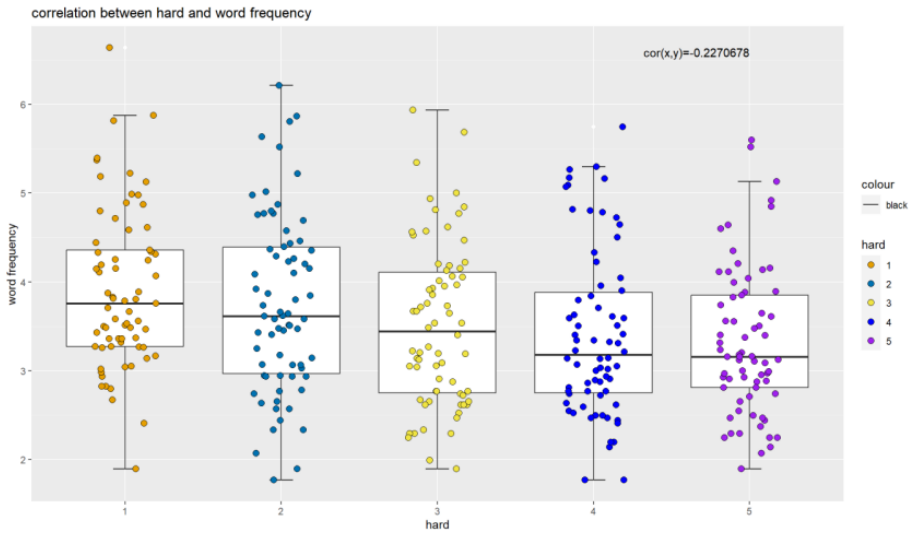

Figure 9: correlation between hard and word frequency

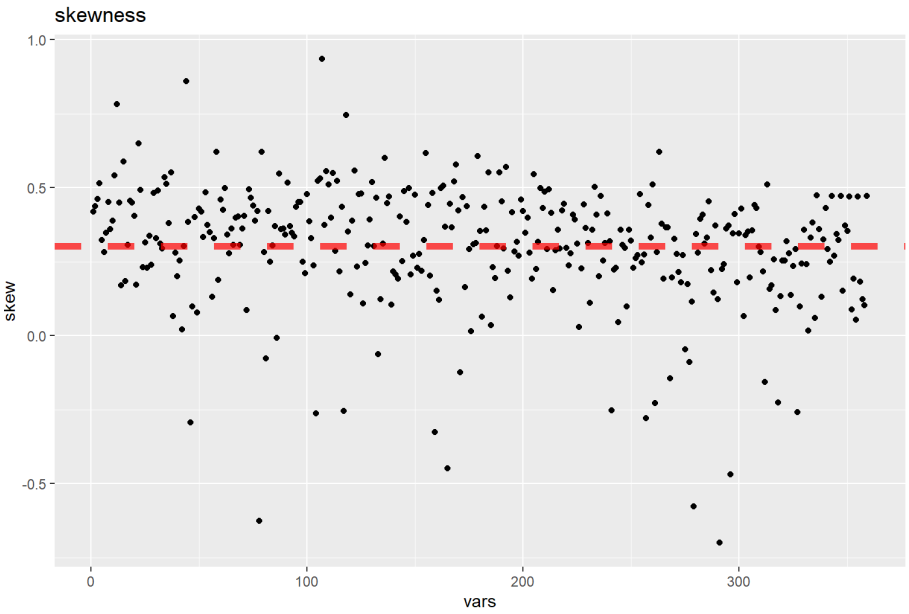

Figure 10: skewness

According to your requirements, here are the results given by us.

Firstly, you asked us to create a prediction interval for the number of reported results on March 1, 2023. We developed an exponential regression model of the form

$$y = \mathrm{e}^{a \cdot x + b} + c$$

to explain the daily variation of the decreasing period of the number of reported results. We used the data you gaved us to traine the parameters. The estimated value of each parameters are as follows:

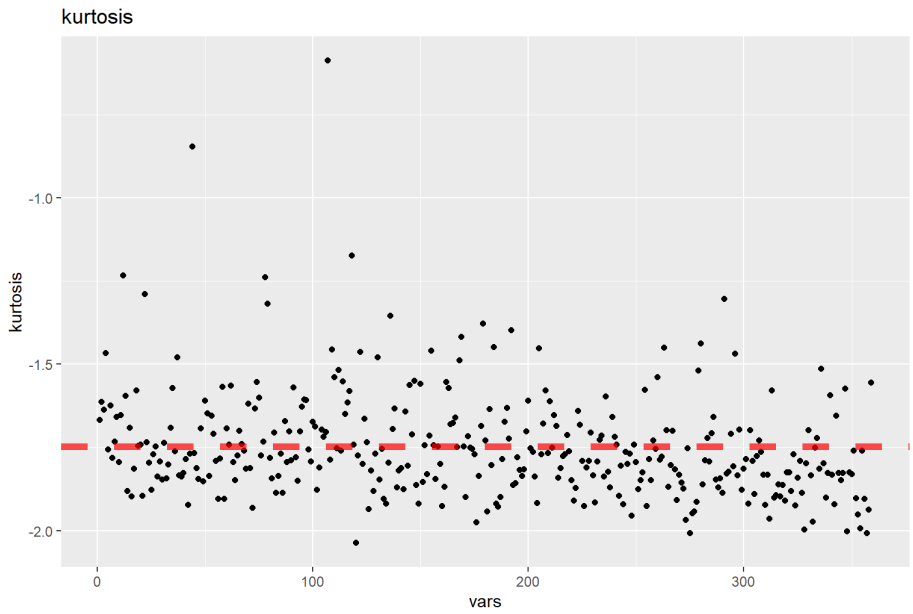

Figure 11: kurtosis

	Estimate	Std. Error	t value	Pr(>|t|)
a	-1.570e-02	5.356e-04	-29.32	<2e-16 ***
b	1.601e+01	1.722e-01	92.95	<2e-16 ***
c	2.311e+04	5.438e+02	42.49	<2e-16 ***

After this, we calculated the sample mean and degrees of freedom, and finally got the confidence interval. The predicted interval for the number of the reported scores on March 1, 2023 is **[17302, 29975]**, which is a 95% confidence interval. The middle number of it is **23639**, which is the expected value.

Then you asked us to find some attributes of the word affect the percentage of scores reported that were played in Hard Mode. We examined three attributes, which are the word frequency, the number of unique letters in the word, and the similarity of the word and other words. However, we find that all these three attributes have little correlation with the percentage of scores reported that were played in Hard Mode. Therefore, basically, we did not find any attributes of the word which significantly affect this percentage. We think that this is reasonable, because we believe that the percentage of scores reported that were played in Hard Mode is proportional to the number of players who play the game in Hard Mode on that day. However, when choosing whether to play the game in Hard Mode or not, the players do not know anything about the word on that day, so the attributes of the word have no reason to affect this choice.

After that, you asked us to develop a model to predict the distribution of the reported results. We developed a multiple linear regression model to predict the distribution. The variable we took into account are the word frequency, the number of unique letters in the word, the similarity of the

word and other words, and the date of the word being used for puzzle. The result of for the word "EERIE" on March 1, 2023 is as follows:

Attempts	1	2	3	4	5	6	X
Percentage	0	3	19	36	27	12	2

Then as you asked, we developed a model to classify solution words by difficulty. We arrange the processed data according to the average number of rounds, and divide the data into five equal parts. Specifically, for 348 rounds, the classification method of 69, 70, 70, 70, 69 can be used to obtain the numerical classification of difficulty. Establish the average number of rounds E(x) to measure the difficulty, replace "X" by 7, arrange the processed data according to the average number of rounds, and divide the data into five equal parts, the results are as follows.

Difficulty	1	2	3	4	5
Interval	3.10-3.89	3.90-4.06	4.07-4.25	4.26-4.50	4.51-5.84

Calculate the average number of rounds on the distribution data of the number of attempts to obtain 4.83. According to the difficulty level, the final result is that the word "EERIE" has a difficulty of 5, which is the hardest.

For other interesting features of this data set, we found that success requires the average skewness and average kurtosis of the bureau number distribution data to be 0.3026415, -1.748139, respectively. For skewness, 0.3026415>0, indicating that the right tail is longer than the left, the data distribution is skewed to the right, and the mean is biased to the right of the data. Success requires that the game number distribution data be a positively skewed data distribution. For kurtosis, -1.748139<0, indicating that the kurtosis of the data is lower than that of the normal distribution, that is, the distribution of the data is relatively flat. This information can help us understand the distribution characteristics of the game winning rate data, and then adjust and optimize the winning rate prediction model.

Above are all of our responses to your questions. Hope that they will be helpful for you.

Best,

Team 2311332

6 References

[1] Van Heuven, W. J., Mandera, P., Keuleers, E., & Brysbaert, M. (2014). "Subtlex-UK: A new and improved word frequency database for British English". Quarterly Journal of Experimental Psychology, 67(6), 1176-1190.

https://www.ugent.be/pp/experimentele-psychologie/en/research/documents/subtlexus/overview.htm

[2] 1eez. (2021). 103976 [Code repository]. GitHub. https://github.com/1eez/103976

Appendices

First appendix

Here are simulation programmes we used in our model as follow.

initial_number_data_plot.R:

Plot the exponential fit and confidence intervals from the first question, including predicted values

```
library(readxl)
library(ggplot2)
library(stats)
Problem_C_Data_Wordle <- read_excel("C:/Users/11872/Desktop/mcm_icm/2023_MCM-ICM_Problems/Problem_C_Data_Wordle.xlsx", skip = 1)
pp<-Problem_C_Data_Wordle[order(Problem_C_Data_Wordle$Date),]
ggplot(pp, aes(x=Date, y=`Number of  reported results`)) +
  geom_point() +geom_line()
```

plot_exp.R:

The initial scatter distribution chart drawing code in the first question

```
library(readxl)
library(ggplot2)
library(stats)
Problem_C_Data_Wordle <- read_excel("C:/Users/11872/Desktop/mcm_icm/2023_MCM-ICM_Problems/Problem_C_Data_Wordle.xlsx", skip = 1)
pp<-Problem_C_Data_Wordle[order(Problem_C_Data_Wordle$Date),]
pp2<-pp[-328,]
bb<-pp2[c(115:dim(pp2)[1]),]
x <- bb$`Contest number`
y <- bb$`Number of  reported results`
df <- data.frame(x = x, y = y)
nlc <- nls.control(maxiter = 1000,tol = 1e-2)
fit <- nls(y ~ (exp(a * x + b) + c), start = list(a = -0.001, b = 13,c=2000) , control = nlc,algorithm = "port")
summary(fit)
se <- sqrt(diag(vcov(fit)))
se
newdata <- data.frame(x = seq(min(df[,1]), 630), length.out = dim(df)[1])
y_hat <- predict(fit, newdata = newdata)
resid <- y - predict(fit)
s <- sd(resid)
mean_resid <- mean(resid)
se <- s * sqrt(1 + 1/length(x) + (x - mean(x))^2/sum((x - mean(x))^2))
free <- length(x) - 2
t_val <- qt(0.025, free)
lower <- y_hat - t_val * se
upper <- y_hat + t_val * se
newdata$y<-y_hat
newdata$lower <- lower
newdata$upper <- upper
```

```
p <- ggplot(df, aes(x, y)) +
  geom_point() +geom_line() +geom_line(data = newdata, aes(x = x, y = y),
      color = "red") +
  ggtitle("Exponential Fit") +
  xlab("Contest number") + ylab("Number of  reported results")
p <- p + geom_ribbon(data = newdata, aes(x = x, ymin = lower, ymax = upper),
    alpha = 0.4,size=0.5,color="purple")
p<-p+geom_vline(
  aes(xintercept = 620),
   color = "red")
p
new_observation <- data.frame(x = 620)
y_predict <- predict(fit, newdata = new_observation)
y_predict
lower_predict <- y_predict - t_val * se
upper_predict <- y_predict + t_val * se
lower_predict <-mean(lower_predict)
upper_predict<-mean(upper_predict)
p <- p + geom_point(aes(x = 620, y = upper_predict), color = "red", size = 3)
p <- p + geom_point(aes(x = 620, y = lower_predict), color = "red", size = 3)
p <- p + geom_point(aes(x = 620, y = y_predict), color = "black", size = 2)
p<-p+geom_text(aes(x = 620, y = upper_predict-2000, label = upper_predict),
    size = 2, color = "black")+
  geom_text(aes(x = 620, y = lower_predict+2000, label = lower_predict), size
      = 2, color = "black")+
  geom_text(aes(x = 620, y = y_predict+2000, label = y_predict), size = 2,
      color = "black")
p<-p+annotate("text", x = 620, y = max(df$y)-10000, label = "x=620", vjust =
    -0.5)
p
```

correlation.R:

Draw a correlation diagram for the second question

```
library(readxl)
library(ggplot2)
SUBTLEX_US_frequency_list_with_PoS_and_Zipf_information <- read_excel("C:/
    Users/11872/Desktop/mcm_icm/SUBTLEX-US frequency list with PoS and Zipf
    information.xlsx")
frequency_word=SUBTLEX_US_frequency_list_with_PoS_and_Zipf_information
Problem_C_Data_Wordle <- read_excel("C:/Users/11872/Desktop/mcm_icm/2023_MCM-
    ICM_Problems/Problem_C_Data_Wordle.xlsx", skip = 1)
table(Problem_C_Data_Wordle$Word %in% frequency_word$Word)
Problem_C_Data_Wordle$Word[!Problem_C_Data_Wordle$Word %in% frequency_word$
    Word]
valid_data<-Problem_C_Data_Wordle[Problem_C_Data_Wordle$Word %in% frequency_
    word$Word,]
table(frequency_word$Word %in% Problem_C_Data_Wordle$Word)
valid_frequency<-frequency_word[frequency_word$Word %in% Problem_C_Data_Wordle
    $Word,]
x
df<-merge(valid_data,valid_frequency, by.x = "Word", by.y = "Word", all.x =
    TRUE)
df<-subset(df,`Number in hard mode`/`Number of  reported results`<0.7)
x=df$`Zipf-value`
y=df$`Number in hard mode`/df$`Number of  reported results`
ggplot(df,aes(x=`Zipf-value`,y=`Number in hard mode`/`Number of  reported
    results`))+geom_point()+labs(title="Percentage of Hard Mode versus word
    frequency",y="Hard Mode percentage")+geom_smooth(method = "lm")+annotate("
    text", x = max(df$`Zipf-value`), y = max(y), label = "cor(x,y)=-0.08641802"
    , hjust = 1, vjust = 1)
cor(x,y)
cor.test(x,y)
```

```
length=nchar(Problem_C_Data_Wordle$Word)
table(length)
ggplot(df)+geom_point(aes(x=`Zipf-value`,y=`7 or more tries (X)`),color="red")
cor(df$`Zipf-value`,df$`7 or more tries (X)`)
cor(df$`Zipf-value`,df$`1 try`)
cor(df$`Zipf-value`,df$`2 tries`)
cor(df$`Zipf-value`,df$`3 tries`)
cor(df$`Zipf-value`,df$`4 tries`)
cor(df$`Zipf-value`,df$`5 tries`)
cor(df$`Zipf-value`,df$`6 tries`)
df$new1=df$`1 try`/(100-df$`7 or more tries (X)`)
df$new2=df$`2 tries`/(100-df$`7 or more tries (X)`)
df$new3=df$`3 tries`/(100-df$`7 or more tries (X)`)
df$new4=df$`4 tries`/(100-df$`7 or more tries (X)`)
df$new5=df$`5 tries`/(100-df$`7 or more tries (X)`)
df$new6=df$`6 tries`/(100-df$`7 or more tries (X)`)
df$average<-(df$new1*1+df$new2*2+df$new3*3+df$new4*4+df$new5*5+df$new6*6)
ggplot(df,aes(x=`Zipf-value`,y=`average`))+geom_point()+labs(title="Average
    tries of winners versus word frequency")+geom_smooth(method = "lm")+
    annotate("text", x = max(df$`Zipf-value`), y = max(df$average), label = "
    cor(x,y)=-0.08473747", hjust = 1, vjust = 1)
cov(df$`Zipf-value`,df$average)
ggplot(df,aes(x=`Zipf-value`,y=`7 or more tries (X)`))+geom_point()+labs(title
    ="`7 or more tries (X)` percentage versus word frequency")+geom_smooth(
    method = "lm")+annotate("text", x = max(df$`Zipf-value`), y = max(df$`7 or
    more tries (X)`), label = "cor(x,y)=-0.09161143", hjust = 1, vjust = 1)
cov(df$`Zipf-value`,df$`7 or more tries (X)`)
df<-df[nchar(df$Word)==5,]
a<-lapply(strsplit(df$Word,""),unique)
b<-lapply(a,length)
df$ununiqueness<-as.vector(unlist(b))
ggplot(df,aes(x=ununiqueness,y=average))+geom_point()+labs(title="Average
    tries of winners versus the number of non-repeated letters in a word ",x="
    number of non-repeated letters")
cor(df$ununiqueness,df$average)
cor(df$ununiqueness,df$`7 or more tries (X)`)
df$num_of_unique_letter<-as.factor(df$ununiqueness)
  stat_boxplot(geom = "errorbar",width=0.15,aes(color="black"))+ geom_boxplot(
      size=0.5,fill="white",outlier.fill="white",outlier.color="white")+ geom_
      jitter(aes(fill=num_of_unique_letter),width =0.2,shape = 21,size=2.5)+
      scale_fill_manual(values = c("#E69F00", "#0072B2","#F0E442"))+  scale_
      color_manual(values=c("black","black","black"))+labs(title="Average tries
       of winners versus the number of non-repeated letters in a word ",x="
      number of non-repeated letters")+
  annotate("text", x = max(df$ununiqueness), y = max(df$average), label = "cor
      (x,y)=-0.3896368", hjust = 1, vjust = 1)
df$hard_percent<-df$`Number in hard mode`/df$`Number of  reported results`
ggplot(df,aes(x=num_of_unique_letter,y=hard_percent,fill=num_of_unique_letter)
    )+ stat_boxplot(geom = "errorbar",width=0.15,aes(color="black"))+ geom_
    boxplot(size=0.5,fill="white",outlier.fill="white",outlier.color="white")+
    geom_jitter(aes(fill=num_of_unique_letter),width =0.2,shape = 21,size=2.5)+
     scale_fill_manual(values = c("#E69F00", "#0072B2","#F0E442"))+ scale_color
    _manual(values=c("black","black","black"))+labs(title="Percentage of Hard
    Mode versus the number of non-repeated letters in a word ",x="number of non
    -repeated letters",y="Percentage of Hard Mode")+annotate("text", x = max(df
    $ununiqueness), y = max(df$hard_percent), label = "cor(x,y)=-0.07373057",
    hjust = 1, vjust = 1)
cor(df$ununiqueness,df$hard_percent)
ggplot(df,aes(x=num_of_unique_letter,y=`7 or more tries (X)`,fill=num_of_
    unique_letter))+ stat_boxplot(geom = "errorbar",width=0.15,aes(color="black
    "))+ geom_boxplot(size=0.5,fill="white",outlier.fill="white",outlier.color=
    "white")+ geom_jitter(aes(fill=num_of_unique_letter),width =0.2,shape = 21,
    size=2.5)+ scale_fill_manual(values = c("#E69F00", "#0072B2","#F0E442"))+
    scale_color_manual(values=c("black","black","black"))+labs(title="`7 or
```

```
    more tries (X)` percentage versus the number of non-repeated letters in a
    word ",x="number of non-repeated letters",y="7 or more tries (X)")+annotate
    ("text", x = max(df$ununiqueness), y = max(df$`7 or more tries (X)`), label
     = "cor(x,y)=-0.1800359", hjust = 1, vjust = 1)
cor(df$ununiqueness,df$`7 or more tries (X)`)
```

similarity_correlation.R:

Draw similarity and hard mode percentages, the average number of rounds of winners, and the percentage of losers

```
output_1 <- read.csv("C:/Users/11872/Desktop/mcm_icm/output_1.txt", header=
    FALSE)
Problem_C_Data_Wordle <- read_excel("C:/Users/11872/Desktop/mcm_icm/2023_MCM-
    ICM_Problems/Problem_C_Data_Wordle.xlsx", skip = 1)
Problem_C_Data_Wordle$Word[!Problem_C_Data_Wordle$Word %in% output_1$V1]
valid_data<-Problem_C_Data_Wordle[Problem_C_Data_Wordle$Word %in% output_1$V1
    ,]
table(output_1$V1 %in% Problem_C_Data_Wordle$Word)
valid_repetition<-output_1[output_1$V1 %in% valid_data$Word,]
df<-merge(valid_data,valid_repetition, by.x = "Word", by.y = "V1", all.x =
    TRUE)
dim(df)
df<-subset(df,`Number in hard mode`/`Number of  reported results`<0.7)
x=df$V2
df$hard_percent<-df$`Number in hard mode`/df$`Number of  reported results`
y=df$`Number in hard mode`/df$`Number of  reported results`
ggplot(df,aes(x=V2,y=hard_percent))+geom_point()+labs(title="Percentage of
    Hard Mode versus Word similarity",x="word similarity")+geom_smooth(method =
     "lm")+annotate("text", x = max(df$V2), y = max(df$hard_percent), label = "
    cor(x,y)=0.04626648", hjust = 1, vjust = 1)
cor(x,y)
df$new1=df$`1 try`/(100-df$`7 or more tries (X)`)
df$new2=df$`2 tries`/(100-df$`7 or more tries (X)`)
df$new3=df$`3 tries`/(100-df$`7 or more tries (X)`)
df$new4=df$`4 tries`/(100-df$`7 or more tries (X)`)
df$new5=df$`5 tries`/(100-df$`7 or more tries (X)`)
df$new6=df$`6 tries`/(100-df$`7 or more tries (X)`)
df$average<-(df$new1*1+df$new2*2+df$new3*3+df$new4*4+df$new5*5+df$new6*6)
ggplot(df,aes(x='V2',y='average'))+geom_point()+labs(title="Average tries of
    winners versus Word similarity",x="word similarity")+geom_smooth(method = "
    lm")+annotate("text", x = max(df$V2), y = max(df$average), label = "cor(x,y
    )=0.03651125", hjust = 1, vjust = 1)
cor(df$`V2`,df$average)
ggplot(df,aes(x='V2',y='7 or more tries (X)'))+geom_point()+labs(title="`7 or
    more tries (X)` versus Word similarity",x="word similarity")+geom_smooth(
    method = "lm")+annotate("text", x = max(df$V2), y = max(df$`7 or more tries
     (X)`), label = "cor(x,y)=0.1114209", hjust = 1, vjust = 1)
cor(df$`V2`,df$`7 or more tries (X)`)
```

distribution_factor_correlation.R:

The correlation between the distribution and the three factors (common frequency, uniqueness of word letters, similarity)

```
library(readxl)
library(ggplot2)
SUBTLEX_US_frequency_list_with_PoS_and_Zipf_information <- read_excel("C:/
    Users/11872/Desktop/mcm_icm/SUBTLEX-US frequency list with PoS and Zipf
    information.xlsx")
frequency_word=SUBTLEX_US_frequency_list_with_PoS_and_Zipf_information
output_1 <- read.csv("C:/Users/11872/Desktop/mcm_icm/output_1.txt", header=
    FALSE)
```

```r
Problem_C_Data_Wordle <- read_excel("C:/Users/11872/Desktop/mcm_icm/2023_MCM-ICM_Problems/Problem_C_Data_Wordle.xlsx", skip = 1)
'trite' %in% frequency_word$Word
'trite' %in% output_1$V1
frequency_word$`Zipf-value`[frequency_word$Word=="trite"]
output_1$V2[output_1$V1=="trite"]
valid_data<-Problem_C_Data_Wordle[Problem_C_Data_Wordle$Word %in% output_1$V1,]
valid_similarity<-output_1[output_1$V1 %in% valid_data$Word,]
df<-merge(valid_data,valid_similarity, by.x = "Word", by.y = "V1", all.x = TRUE)
valid_data_2<-df[df$Word %in% frequency_word$Word,]
valid_frequency<-frequency_word[frequency_word$Word %in% df$Word,]
df<-merge(valid_data_2,valid_frequency, by.x = "Word", by.y = "Word", all.x = TRUE)
df<-df[nchar(df$Word)==5,]
a<-lapply(str_split(df$Word,""),unique)
b<-lapply(a,length)
df$num_unique<-as.vector(unlist(b))
df$z=df$`Zipf-value`
df$one<-df$`1 try`
df$two<-df$`2 tries`
df$three<-df$`3 tries`
df$four<-df$`4 tries`
df$five<-df$`5 tries`
df$six<-df$`6 tries`
df$more<-df$`7 or more tries (X)`
model1<-lm(cbind(one,two,three,four,five,six,more)~z,data=df)
summary(model1)
model2<-lm(cbind(one,two,three,four,five,six,more)~num_unique,data=df)
summary(model2)
model3<-lm(cbind(one,two,three,four,five,six,more)~V2,data=df)
summary(model3)
```

distribution_time_correlation.R:

The correlation of distribution in time

```r
library(readxl)
library(ggplot2)
Problem_C_Data_Wordle <- read_excel("C:/Users/11872/Desktop/mcm_icm/2023_MCM-ICM_Problems/Problem_C_Data_Wordle.xlsx", skip = 1)
df<-Problem_C_Data_Wordle
df$one<-df$`1 try`
df$two<-df$`2 tries`
df$three<-df$`3 tries`
df$four<-df$`4 tries`
df$five<-df$`5 tries`
df$six<-df$`6 tries`
df$more<-df$`7 or more tries (X)`
df$time<-df$`Contest number`
model1<-lm(cbind(one,two,three,four,five,six,more)~time,data=df)
summary(model1)
```

linear_predict_model_withtime.R

Prediction results when multiple linear regression counts time as the dependent variable

```r
library(readxl)
library(ggplot2)
library(stats)
SUBTLEX_US_frequency_list_with_PoS_and_Zipf_information <- read_excel("C:/Users/11872/Desktop/mcm_icm/SUBTLEX-US frequency list with PoS and Zipf
```

```
    information.xlsx")
frequency_word=SUBTLEX_US_frequency_list_with_PoS_and_Zipf_information
output_1 <- read.csv("C:/Users/11872/Desktop/mcm_icm/output_1.txt", header=
    FALSE)
Problem_C_Data_Wordle <- read_excel("C:/Users/11872/Desktop/mcm_icm/2023_MCM-
    ICM_Problems/Problem_C_Data_Wordle.xlsx", skip = 1)
'trite' %in% frequency_word$Word
'trite' %in% output_1$V1
frequency_word$`Zipf-value`[frequency_word$Word=="trite"]
output_1$V2[output_1$V1=="trite"]
valid_data<-Problem_C_Data_Wordle[Problem_C_Data_Wordle$Word %in% output_1$V1
    ,]
valid_similarity<-output_1[output_1$V1 %in% valid_data$Word,]
df<-merge(valid_data,valid_similarity, by.x = "Word", by.y = "V1", all.x =
    TRUE)
valid_data_2<-df[df$Word %in% frequency_word$Word,]
valid_frequency<-frequency_word[frequency_word$Word %in% df$Word,]
df<-merge(valid_data_2,valid_frequency, by.x = "Word", by.y = "Word", all.x =
    TRUE)
df<-df[nchar(df$Word)==5,]
a<-lapply(str_split(df$Word,""),unique)
b<-lapply(a,length)
df$num_unique<-as.vector(unlist(b))
df$z=df$`Zipf-value`
df$one<-df$`1 try`
df$two<-df$`2 tries`
df$three<-df$`3 tries`
df$four<-df$`4 tries`
df$five<-df$`5 tries`
df$six<-df$`6 tries`
df$more<-df$`7 or more tries (X)`
df$time<-df$`Contest number`
model1<-lm(`1 try` ~ V2+z+num_unique+time,data=df)
new_data<-data.frame(V2=462,z=2.904618,num_unique=4,time=620)
predicted_1 <- predict(model1, newdata = new_data)
predicted_1
model2<-lm(`2 tries` ~ V2+z+num_unique+time,data=df)
predicted_2 <- predict(model2, newdata = new_data)
predicted_2
model3<-lm(`3 tries` ~ V2+z+num_unique+time,data=df)
predicted_3 <- predict(model3, newdata = new_data)
predicted_3
model4<-lm(`4 tries` ~ V2+z+num_unique+time,data=df)
predicted_4 <- predict(model4, newdata = new_data)
predicted_4
model5<-lm(`5 tries` ~ V2+z+num_unique+time,data=df)
predicted_5 <- predict(model5, newdata = new_data)
predicted_5
model6<-lm(`6 tries` ~ V2+z+num_unique+time,data=df)
predicted_6 <- predict(model6, newdata = new_data)
predicted_6
model7<-lm(`7 or more tries (X)` ~ V2+z+num_unique+time,data=df)
predicted_7 <- predict(model7, newdata = new_data)
predicted_7
linear_result<-c(predicted_1,predicted_2,predicted_3,predicted_4,predicted_5,
    predicted_6,predicted_7)
round_linear_result<-round(linear_result)
linear_result
round_linear_result
sum(round_linear_result)
model<-lm(cbind(one,two,three,four,five,six,more) ~ V2+z+num_unique+time,data=
    df)
new_data<-data.frame(V2=462,z=2.904618,num_unique=4,time=620)
predicted <- predict(model, newdata = new_data)
```

```
predicted
result<-round(predicted)
result
summary(model)
df$pred <- predict(model)
df$resid <- df[,c("one","two","three","four","five","six","more")] - df$pred
a<-c("one try","two tries","three tries","four tries","five tries","six tries"
    ,"more")
par(mfrow = c(1, 7))
for (i in 1:7) {
  plot(df$pred[,i], residuals(model)[,i], main = paste("Residuals for", a[i]),
       xlab = "Fitted Values", ylab = "Residuals")
  abline(h = 0, col = "red", lty = 2)
}
pred1 <- predict.lm(model1, new_data, interval = "confidence", level = 0.95)
pred2 <- predict.lm(model2, new_data, interval = "confidence", level = 0.95)
pred3 <- predict.lm(model3, new_data, interval = "confidence", level = 0.95)
pred4 <- predict.lm(model4, new_data, interval = "confidence", level = 0.95)
pred5 <- predict.lm(model5, new_data, interval = "confidence", level = 0.95)
pred6 <- predict.lm(model6, new_data, interval = "confidence", level = 0.95)
pred7 <- predict.lm(model7, new_data, interval = "confidence", level = 0.95)
exact_interval<-rbind(pred1,pred2,pred3,pred4,pred5,pred6,pred7)
exact_interval
round(exact_interval)
```

mcm3.R

The third question is all the codes, including drawing

```
library(readxl)
library(ggplot2)
library(stats)
SUBTLEX_US_frequency_list_with_PoS_and_Zipf_information <- read_excel("C:/
    Users/11872/Desktop/mcm_icm/SUBTLEX-US frequency list with PoS and Zipf
    information.xlsx")
frequency_word=SUBTLEX_US_frequency_list_with_PoS_and_Zipf_information
output_1 <- read.csv("C:/Users/11872/Desktop/mcm_icm/output_1.txt", header=
    FALSE)
Problem_C_Data_Wordle <- read_excel("C:/Users/11872/Desktop/mcm_icm/2023_MCM-
    ICM_Problems/Problem_C_Data_Wordle.xlsx", skip = 1)
'trite' %in% frequency_word$Word
'trite' %in% output_1$V1
frequency_word$`Zipf-value`[frequency_word$Word=="trite"]
output_1$V2[output_1$V1=="trite"]
frequency_word[frequency_word$Word=="eerie",]$`Zipf-value`
valid_data<-Problem_C_Data_Wordle[Problem_C_Data_Wordle$Word %in% output_1$V1
    ,]
valid_similarity<-output_1[output_1$V1 %in% valid_data$Word,]
df<-merge(valid_data,valid_similarity, by.x = "Word", by.y = "V1", all.x =
    TRUE)
valid_data_2<-df[df$Word %in% frequency_word$Word,]
valid_frequency<-frequency_word[frequency_word$Word %in% df$Word,]
df<-merge(valid_data_2,valid_frequency, by.x = "Word", by.y = "Word", all.x =
    TRUE)
df<-df[nchar(df$Word)==5,]
a<-lapply(strsplit(df$Word,""),unique)
b<-lapply(a,length)
df$num_unique<-as.vector(unlist(b))
df$z=df$`Zipf-value`
df$one<-df$`1 try`
df$two<-df$`2 tries`
df$three<-df$`3 tries`
df$four<-df$`4 tries`
df$five<-df$`5 tries`
```

```r
df$six<-df$`6 tries`
df$more<-df$`7 or more tries (X)`
df$time<-df$`Contest number`
df$ave<-(df$one*1+df$two*2+df$three*3+df$four*4+df$five*5+df$six*6+df$more*7)/100
df<-df[order(df$ave),]
df$hard<-c(seq(1,1,length.out=69),seq(2,2,length.out=70),seq(3,3,length.out=70),seq(4,4,length.out=70),seq(5,5,length.out=69))
df$hard<-as.factor(df$hard)
model<-lm(cbind(one,two,three,four,five,six,more) ~ V2+z+num_unique,data=df)
new_data<-data.frame(V2=2937,z=3.246077,num_unique=3)
predicted <- predict(model, newdata = new_data)
result<-round(predicted)
sum(result)
result
avg_result<-(result[1]*1+result[2]*2+result[3]*3+result[4]*4+result[5]*5+result[6]*6+result[7]*7)/100
avg_result
df$ave[1]
df$ave[69]
df$ave[139]
df$ave[209]
df$ave[279]
df$ave[348]
```

generator.py

Preprocessing of mono-lexical data

```python
import csv
def num_same(item1, item2):
    res = 0
    add = 1
    for i in range(5):
        if (item1[i] == item2[i]):
            res += add
            add += 1
    return res
csvFile = open("EnWords.csv", "r")
reader = csv.reader(csvFile)
result = []
for item in reader:
    if (len(item[0]) == 5):
        result.append(item[0])
csvFile.close()
new_res = [ 0 for _ in range( len(result))]
for i in range(len(result)):
    for j in range(i+1, len(result)):
        new_res[i] += num_same(result[i], result[j])
print(len(result))
for i in range(len(result)):
    print(result[i], new_res[i])
```

Second appendix

Here are some files we used in our model as follow.

SUBTLEX-US frequency list with PoS and Zipf information.xlsx

common level source data

Enwords.xlsx

Glossary data