Problem Chosen	2023 MCM/ICM Summary Sheet	Team Control Number
C		2316007

Word, Subword, and Letter: Hierarchical Word Embedding for Predicting Wordle Results

Summary

Word-guessing games, such as the widely popular Wordle puzzle game offered by The New York Times, have gained a large following. The game requires players to guess a five-letter word in six or fewer attempts. Based on feedback from each attempt, players can adjust their guesses to get closer to the correct answer. To analyze the 355 results reported in the **Attachment**, we develop effective models to meet the needs of The New York Times.

First, we rethink the attributes of words in the context of Wordle and define the attributes of words at the levels of words, subwords, and letters, based on the difficulty of the game rules and word associations.

We analyze the changes in the number of reported results over time and establish an **ARIMA(p, d, q)** time series model to predict future changes in the number of reported results. We obtain a predicted range of the number of reported results on March 1, 2023, with a 95% confidence level. We use the **Spearman** correlation analysis method to explore the degree of correlation between word attributes and the proportion of players in the hard mode. The results show that there is no significant correlation between them while indicating a significantly correlated with time.

To predict the distribution of scores of a specified word on a particular date, we use **BERT** to embed each letter of the word and the word itself before training an LSTM model. The model has extraordinary performance, achieving only a 3% error on the validation set.

With the intention of grading the difficulty of **EERIE**, we originally use K-means clustering to divide the words from the Attachment into three categories. Then we summarize the main attribute of each category of words. Then, we use the 10-word attributes we constructed earlier to train a supervised **Random Forest** classifier, achieving a 73% accuracy rate in the three-categories classification task. Finally, we apply this model into determining that the difficulty level of the word **EERIE is "medium."** In addition, some interesting features of the Attachment are illustrated in our essay.

After validating the sensitivity of our models, we propose some **effective suggestions** to the Puzzle Editor of The New York Times to attract more players, such as adjusting question strategies. We recapitulate our conclusion and recommendations in a letter.

Keywords: ARIMA, BERT, LSTM, NLP, Random forest algorithm

Contents

1 Introduction

The Wordle word puzzle game offered by The New York Times has gained a large user base since its release. The game requires players to guess a five-letter word in six or fewer attempts, and based on feedback from each attempt, players can adjust their guessing direction to get closer to the correct answer. The system checks whether the guessed word is a real word after each attempt, and if it is not, the attempt will not be recorded. Nowadays, Wordle supports more than 60 languages and is attracting more and more users to join the game. To enrich players' game experience, they can play in "regular mode" or "hard mode." In the latter, once a correct letter is found in the word (indicated by a yellow or green block), the player must use these letters in the subsequent guessing attempts, thus limiting the player's choices. In order to deeply explore the interesting data features in the feedback report (the Attachment) of the Wordle game, we have established mathematical models in the following sections.

Section 2 presents the basic assumptions required to build the model, which are universal throughout the modeling process. Section 3 provides a description of all symbols used, and in Section 4, we define the attributes of words in the context of Wordle:First, we rethink the attributes of words in the context of Wordle, and define the attributes of words at three levels: word, subword, and letter, based on the difficulty of word association and in conjunction with the rules of the game.

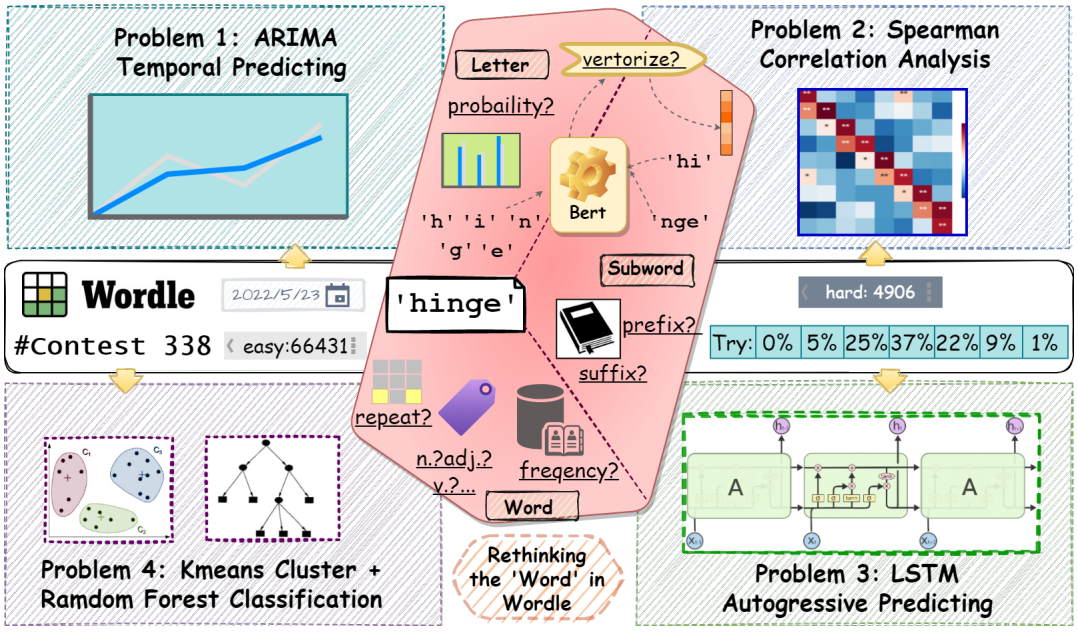


Figure 1: Our idea and motivation

In order to fulfill the task of analyzing the attachment information proposed by The New York Times, we analyze the change of the number of report results over time in Section 5 and establish an ARIMA(p,d,q) time series model to predict the future changes in the number of reports. At a 95% confidence level, we obtain the predicted range of the

number of report results on March 1, 2023. In Section 6, we use the Spearman correlation analysis method to analyze the correlation between various word attributes and the proportion of players in the hard mode game. The results show that there is no significant correlation between various word attributes and the proportion of players in the hard mode game. We find that the proportion of players in the difficult mode game is significantly correlated with time. In Section 7, in order to predict the distribution of scores for a specified vocabulary on a given date, we believe it necessary to use richer semantic features. Therefore, we use BERT to generate word embeddings for each letter and the word itself, training an LSTM model on this basis. On the validation set, we achieve a performance with only 3% error. In section 8, we first use K-means clustering to divide the words in the attachment into three categories according to their difficulty, and then study the significant attribute of each category of words. We then apply the 10 word attributes constructed in the previous section to train a supervised random forest classifier, which achieve an accuracy rate of 73% in the three-categories classification task. Finally, we use the model to determine that the category of the word "EERIE" is "medium". In Section 9, we briefly introduce other features of the data in the attachment. In Section 10, we verify the robustness of our model through three additional experiments, demonstrating that our model is relatively stable in terms of performance through ablation and perturbation experiments. Section 11 includes the conclusion of our solution. Finally, based on the results of our model analysis, we write a letter to the Wordle game editor of The New York Times, which is presented in Section 12.

2 Assumptions

For the convenience of modeling, we make the following main assumptions in this article. These assumptions apply throughout all the models and analyses, and any exceptions will be specified separately.

1. After excluding the outlier data, the data in the attachment truthfully reflects the historical results of the game.
2. The hints provided by the system after each player's guess can be helpful in guiding the player to the correct answer.
3. Players are not informed of the answers in advance before answering.
4. Players in the game do find it easier to associate more familiar words from different perspectives.

3 Notions

In this work, we use the nomenclature in Table 1 in the model construction. Other none-frequent-used symbols will be introduced once they are used.

Table 1: Notations used in this literature

Symbol	Definition	Type
W	word text	String
C_j	The j-th letter of the word, $j \in \{1,2,3,4,5\}$	Char
r_1	The percentage of 1 correct guess attempt for the i-th word	Int
r_2	The percentage of 2 correct guess attempt for the i-th word	Int
r_3	The percentage of 3 correct guess attempt for the i-th word	Int
r_4	The percentage of 4 correct guess attempt for the i-th word	Int
r_5	The percentage of 5 correct guess attempt for the i-th word	Int
r_6	The percentage of 6 correct guess attempt for the i-th word	Int
r_7	The percentage of 7 correct guess or more attempt for the i-th word	Int
rpt	Whether there are repeated letters in the word	Bool
sub	There are no subwords in the word	Bool
$freq_j$	The frequency of each letter , $j \in \{1,2,3,4,5\}$	Float
$freq_W$	How often words appear in the English corpus	Int
tag	Whether the part of speech is a noun	Bool
N	Number of reported results	Int
N_h	Number in hard mode	Int
$date$	Convert a date to an ordinal number	Int

4 Exploring Rich Semantics of Word Hierarchically in the Context of Wordle

4.1 Overview

In the context of Wordle, the difficulty of guessing a word is influenced by both the game rules and whether it's hard to be associated with in the player's mind. In order to obtain richer semantic features, we should explore the attributes of words in the Wordle context from different levels of the word.

4.2 Separate the word into letters

People tend to predict words that contain common letters. We define the letter frequency at each position as follows: (the 'Frequency' function returns the relative frequency of a letter based on the word frequency statistics data, with 'freq' as its alias)

$$freq_j = Freq(c_j) = Freqency(c_j),\ j \in \{1...5\}$$

4.3 Foscusing the words with subwords

Since people are likely to associate a word with meaningful prefixes and suffixes, therefore, we define the variable fix to represent whether a word has meaningful prefixes and

suffixes. The Fix function returns 1 if it has and 0 if it hasn't.

$$fix = Fix(W)$$

We use the rich vocabulary of BERT [3] to identify words that have no clear meaning. We use the variable "sub" to represent the number of subwords of the word:(where " length"is the length function and "Sub" is an alias):

$$sub = Sub(W) = length(BertTokenzier(W))$$

4.4 Regard the word as a whole

Above all, Words with repeated letters give players more yellow tiles. We define the variable "rpt" to represent whether a word has repeated letters, and the function to determine repetition is called "repeat", with "Rpt" as its alias.

$$rpt = Rpt(W) = repeat(W)$$

People tend to associate nouns more easily. We define the variable "tag" to represent whether a word is a noun. The function to determine whether a word is a noun is called "isN", "Tag" is the alias.

$$tag = Tag(W) = isN(W)$$

The more common a word is, the easier it is to be guessed. We define the variable "freq_w" to represent the frequency of a word in English. The function "Frequency" returns the frequency of a word in the frequency table.

$$Freq_W = Freqency(W)$$

4.5 Our definition of the word in the Wordle context

Therefore, by combining the rules of the game and people's tendencies to associate with different types of words, we have quantified the difficulty of guessing a word at different levels and enriched the semantics of the word in the context of Wordle. Therefore, the semantic attribute of a word under Wordle is represented by "Attr".

$$Attr = [Freqency(W, \{c_1...c_5\}), Tag(W), Rpt(W), Sub(W), Fix(W)] = [freq_1, ...freq_5, freq_W, tag, rpt, sub, fix]$$

4.6 Data preprocessing

By analyzing the data in the table, we found some special cases: some words are composed of four letters; the sum of the percentages of attempts with different numbers is far above 100%; the Number of reported results is 2000 (That values is far lower than other samples) etc. The dates with abnormal data include: 3.27, 4.29 , 11.26 and 11.30.

In order to ensure the accuracy of the model and avoid bias towards certain features, we have performed min-max normalization on the word attributes we have defined. For a variable m, with a maximum value of m_{max} and a minimum value of m_{min} in the dataset, the min-max normalization is defined as follows:

$$m_i = \frac{m_i - m_{min}}{m_{max} - m_{min}}$$

5 Problem I:

5.1 ARIMA model

In this section, we construct an ARIMA time series analysis model to examine the changes in the number of reported results over time and predict the range of reported results on March 1, 2023.

The full name of the ARIMA(p, d, q) model is the Autoregressive Integrated Moving Average model. "p" represents the number of autoregressive terms, "q" represents the number of moving average terms, and "d" represents the number of times that the time series is differenced to achieve stationarity. The term "ARIMA model" refers to a model that transforms a non-stationary time series into a stationary time series and then models the dependent variable as a regression against its lagged values and the current and lagged values of a random error term.

Initially, we process the missing data. As we remove erroneous data during data preprocessing, there are missing values at the deleted time points and the dates are no longer continuous. To address this, we fill in the missing data for the dates of 3.27, 4.29, 11.26, and 11.30 with the average values of neighboring points. Subsequently, we plot a scatterplot, which is presented in the figure below.

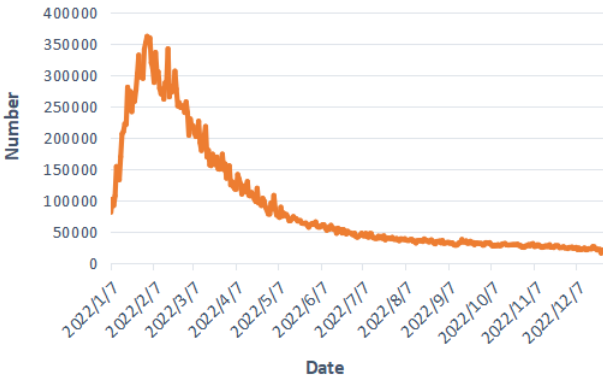


Figure 1: Date-Number scatterplot

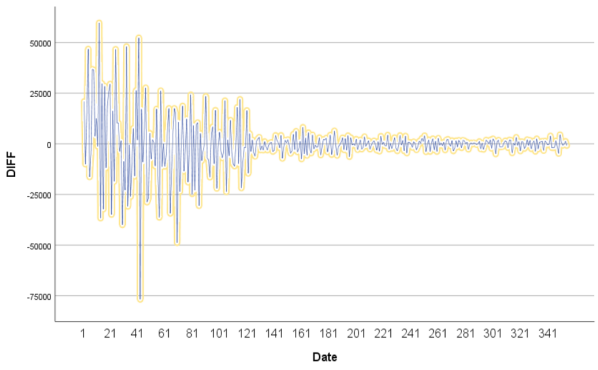


Figure 2: Number of reported results-time

By examining the relationship between the number of reported results and time in the figure above, we observe the following characteristics: the number of reported results shows a linear growth with time during the first 30 days of record keeping, followed by a continuous decline over the subsequent time period. The time series data exhibits random noise but does not demonstrate any discernible periodic or seasonal trends. Based on the observed time series plot, we can easily conclude that the mean value of the sequence is not constant, hence the series is considered non-stationary.

stationarity processing and testing

From a statistical perspective, stationarity refers to the property that the distribution of data remains unchanged when shifted over time. Thus, non-stationary data exhibits fluctuations due to trends so it requires transformation before analysis. In this study, we apply first-order differencing to transform the non-stationary sequence into a stationary one, thereby eliminating the trend in the sequence.

We conduct the Augmented Dickey-Fuller (ADF) unit root test on the sequence and find that the null hypothesis of a unit root can be rejected at the 0.05 significance level. Therefore, we conclude that the first-order differenced sequence is stationary and satisfies the requirement for time series analysis.

white noise test

We perform a Q-test to diagnose whether the residual sequence is white noise. The results are shown below.

Model statistics

Model	Number of predictors	Model Fit Statistics		Young Box Q(18)			Number of outliers
		Stationary R-squared	R-square	Statistics	DF	Significant	
ARIMA	0	.831	.984	27.865	16	.130	4

Figure 3: Q-test

P=0.13>0.05 so We accept the null hypothesis. The residual sequence is white noise. It indicates that the fluctuations of the residuals do not show any systematic patterns. Therefore, the residual sequence does not contain any predictable information, confirming the validity of our model.

Draw the ACF / PACF chart to find the optimal parameters

We calculate the autocorrelation coefficients (ACF) and partial autocorrelation coefficients (PACF) for the obtained stationary time series, plotting the ACF and PACF as follows:

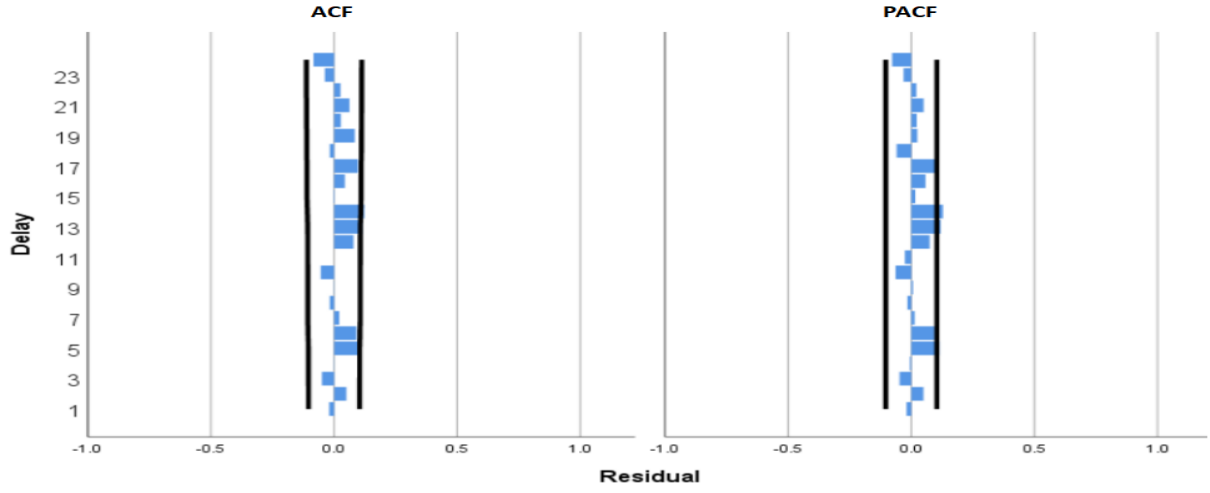


Figure 4: ACF and PACF

By judging the tailing and truncation characteristics of the ACF and PACF, we deter -mine the optimal order of the model to be p=0, q=7, and select the ARIMA(0,1,7) model.

5.2 Model Fitting and Results Analysis

We present the fitting results of the model, as shown in the figure below.

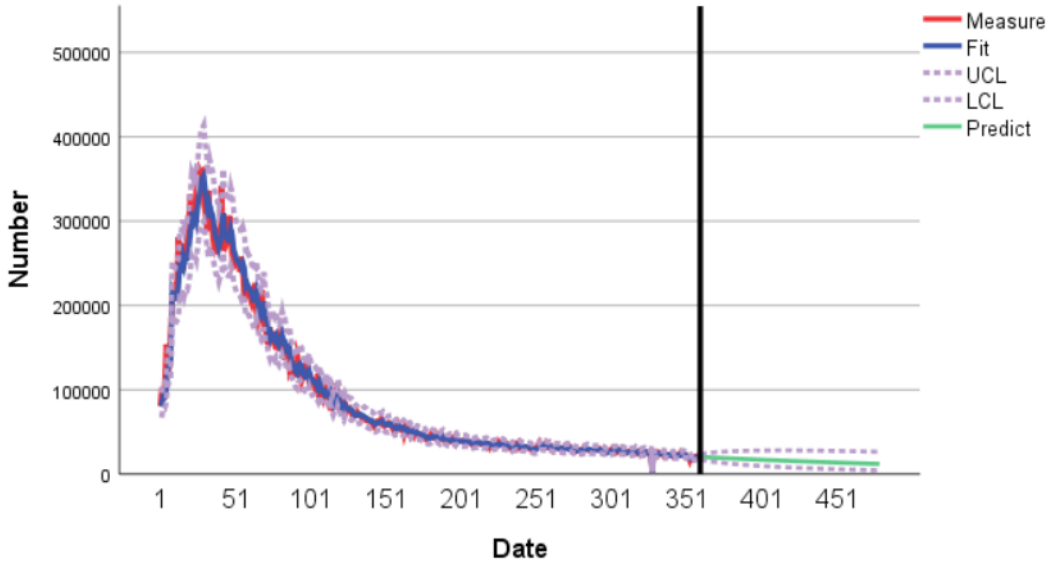


Figure 5: fitting results

It can be seen that the fitted value curve of the model has a high degree of overlap with t he observed value curve, indicating good fitting performance. The predicted value curve i s a flat line. The prediction accuracy can be inferred from several parameters: the stable R -squared and adjusted R-squared values are 0.831 and 0.984, respectively, which are close to 1; the normalized BIC value is only 18.799, indicating a good fitting degree of the model.Co mbining the analysis with the image, it can be observed that the report quantity has slig htly decreased since December 31, 2022, while still maintaining the original trend of the sequence.

$$y_t = y_{t-1} + 0.498\varepsilon_{t-1} - 0.111\varepsilon_{t-14}$$

The prediction equation obtained is as follows:

At a 95% confidence level, our predicted results for the report quantity range on March 1, 2023 are as follows:

$$[19458, 20523]$$

6 Problem II:

In this part, we use correlation analysis to investigate whether the attributes of words affect the proportion of players in difficult mode.

First, we perform a normality test on the data of each word attribute and the data of the proportion of players by drawing Q-Q plots. By observing whether the plotted graphs can be approximated to a straight line, we can determine whether the random variables are to be tested following a normal distribution. We find that some random variables failed the normality test, while others passed the test.

We plot a matrix scatterplot and a correlation coefficient matrix, as shown in the figure below.

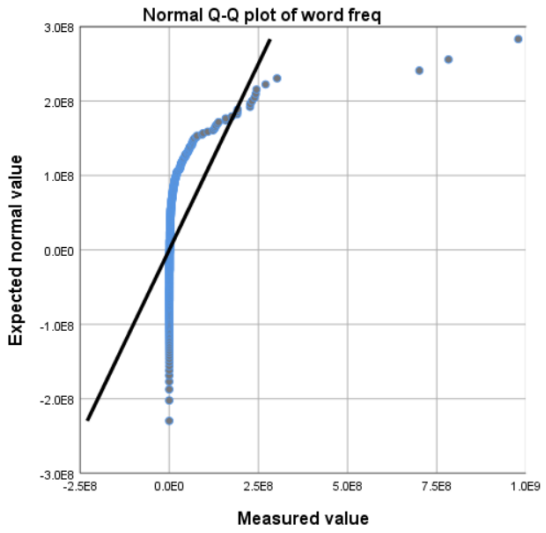


Figure 6: QQ plot

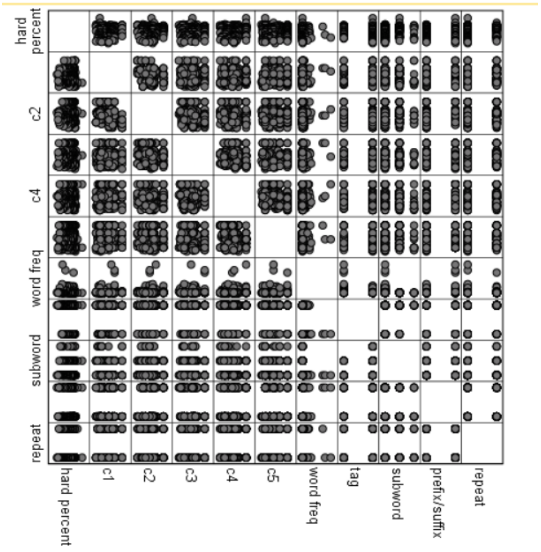


Figure 7: matrix scatterplot

Based on the scatter plot, we initially conclude that there is no linear relationship between the data.
Since the data do not follow a normal distribution, we used Spearman correlation analysis.
The calculation of the Spearman correlation coefficient is as follows:

$$r_s = 1 - \frac{6\sum_{i=1}^{N} d_i^2}{N\left(N^2 - 1\right)}$$

The positive and negative values of rs respectively represent positive and negative correlation, and the greater the absolute value, the stronger the correlation between the two variables. The results are shown in the following table:

Spearman correlation coefficient

		hard percent	c1	c2	c3	c4	c5	word freq	tag	subword	prefix/suffix	repeat
hard percent	correlation coefficient	1.000	.005	-.013	-.058	.022	-.016	-.095	-.023	.090	.057	.084
	Sig. (twin tails)	.	.929	.810	.278	.685	.763	.070	.661	.091	.282	.114

**. Correlation is significant at the 0.01 level (2-tailed).
*. Correlation is significant at the 0.05 level (2-tailed).

Figure 8: Spearman correlation

After hypothesis testing, we find that at the level of 0.05, there was no significant correlation between the attributes of each word and the proportion of people playing hard mode games.

Next, we attempt to analyze the reasons for the lack of significant correlation between the two sets of data. Through an analysis of game modes, we discover that players must first select a game mode before proceeding to guess words, and they cannot change the

game mode midway through the game. The properties of the words may affect the difficulty of guessing for the players, which may affect the percentage of guesses recorded on a given day, but it will not affect the selection of the game mode. We speculate that a player's past gaming experience may influence their choice of game mode in the future. Furthermore, we believe that the percentage of people selecting the difficult mode may be related to time. We have plotted a time series graph showing the percentage of players selecting the difficult mode over time.

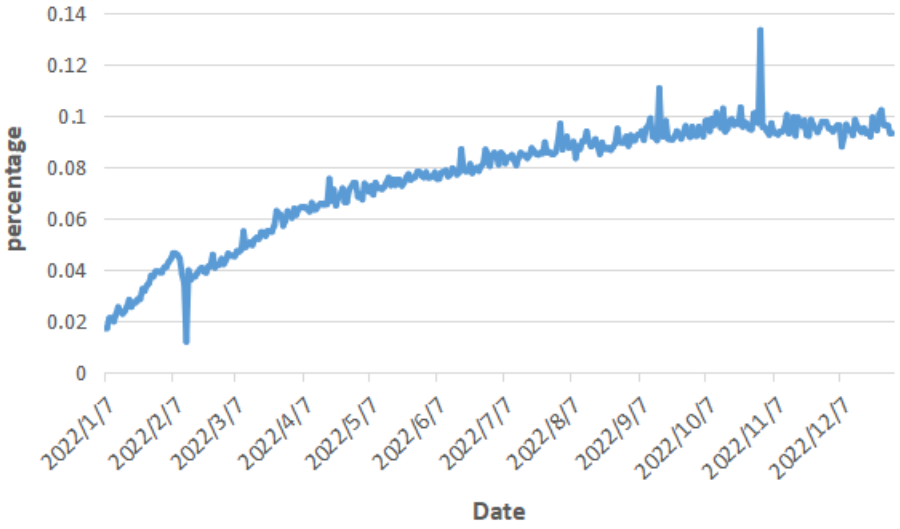


Figure 9: percentage-Date

The time series graph is a logarithmic curve. Over time, the percentage of players selecting the difficult mode initially increases rapidly and then gradually stabilizes. The graph indicates a significant correlation between time and the proportion of players selecting the difficult mode. Therefore, we conclude that time, rather than the properties of the words, has an effect on the proportion of players selecting the difficult mode.

7 Problem III:

7.1 We treat predicting the distribution of scores as the autoregressive predicting

The distribution of the prediction of scores is related to the features of the words. Similar to classical language models, we treat words as sequences of letters. At the same time, we treat the percentages of scores that we want to predict as a sequence. This allows the well-trained model to satisfy the property that the sum of the predicted percentages is Approximately equal to 1 during temporal inference. Therefore, our model can be defined as follows: f is our model, θ is the parameter of our model. Our modeling goal is to minimize the error between the output of our model predicted and the true label. Therefore,

we choose MSE loss as our loss function. Formulas are as follows:

$$\hat{r}_i = f(W, D, \{r_1...r_{i-1}\}; \theta), i \in \{1, 2...7\}$$
$$Loss = argmax \sum_{i=1}^{7} \frac{(f(W, D, \{r_1...r_{i-1}\}; \theta) - r_i)^2}{7}$$

On the other hand, we do not model the sequence by date because we conducted the correlation analysis between the attributes of the date and our define word attributes with percentages in different scores and we find that the correlation between the date and percentages in different scores are not significantly higher than the correlation between our defined word attributes and percentages in different scores.

The reason for not considering the number of game participants and the number of people who choose the difficult mode is that the model is uncertain about these variables when predicting the word "EERIE".

7.2 Using BERT embedding for richer semantics

Bert [3] is a milestone work in the field of natural language processing. By using t he de-sign of MLM (Masked Language Model) and successfully pre-training on a large c orpus,Bert has learned very rich representations, which enable it to universally express seman-tics for downstream tasks.

For our task, as abovementioned, whether the letters of the word are common and the structure of the word will affect the difficulty of Wordle guessing. Therefore, using Bert to embed each letter can make our model gain richer semantics, and using Bert for tokenization can increase the sensitivity of the model to the structure of the word. The process is shown in the following formula:

$$WordleEmbedding = Concat(Bert(c_1), Bert(c_2)...Bert(c_5), Bert(W))$$

7.3 Design of our LSTM model

Due to our small dataset, and the tendency of Transformers to overfit on small datasets, we choose a light recurrent neural network with fewer layers. Among the RNNs, LSTM with gate mechanism updates the internal state, allowing the sequence model to adaptively capture long term and short term information. Hence, we select LSTM as the frame of our model.

Specifically, as shown in the following formula, for the t-th input x_t in the sequence, the output and the memory of the (t-1)-th unit is h_{t-1} and c_{t-1}. The forget gate f_t, which learns to control how much information needs to be forgotten, which is parameterized by U_f, W_f, and b_f. The remaining information is kept as k_t.

$$f_t = \sigma\left(U_f h_{t-1} + W_f x_t + b_f\right)$$
$$k_t = c_{t-1} \odot f_t$$

For the input and memory of the previous unit, LSTM first generates a new candidate information g_t.

$$g_t = \tanh\left(U_g h_{t-1} + W_g x_t + b_g\right)$$

the input gate i_t learns to filter candidate information by learning its parameters U_i, W_i, and b_i. The filtered candidate information j_t is then incorporated into the new internal state c_t. This adaptive filtering mechanism enables the model to dynamically determine which information should be updated in the internal state, thereby allowing it to better capture relevant features of the input sequence.

$$
\begin{aligned}
i_t &= \sigma\left(U_i h_{t-1} + W_i x_t + b_i\right) \\
j_t &= g_t \odot i_t \\
c_t &= j_t + k_t
\end{aligned}
$$

Finally, the output gate o_t learns to generate the new hidden state h_t by learning its parameters U_o, W_o, and b_o, and outputs the hidden state based on the filtered candidate information in c_t. The new hidden state h_t is then passed along with c_t to the next unit in the sequence.

$$
\begin{aligned}
o_t &= \sigma\left(U_o h_{t-1} + W_o x_t + b_o\right) \\
h_t &= \tanh\left(c_t\right) \odot o_t
\end{aligned}
$$

For our problem, the performance of the LSTM model is not affected by the order in which we input different information, and it can use its adaptive long-term and short-term capture ability to keep the sum of the percentages of the score in a reasonable range when outputting. Therefore, our model is designed as shown in the following figure.

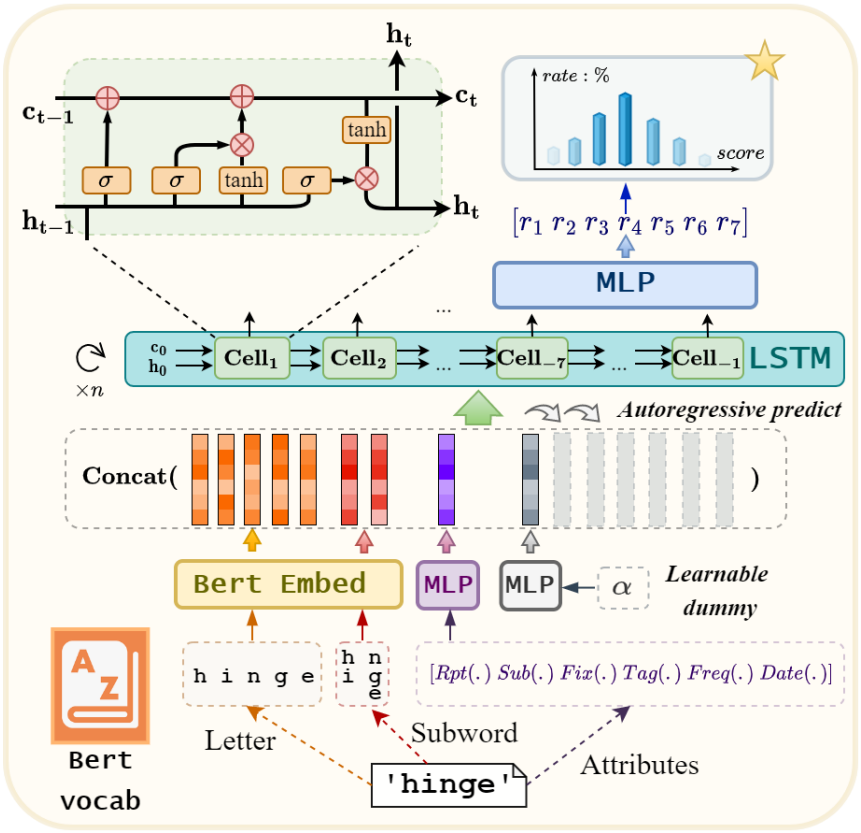


Figure 10: Our LSTM model

In addition to the letter and subword embeddings from Bert, our input also includes an 11-dimensional vector composed of the word attributes we defined and dates, which is projected to the same dim of word embeddings by a learnable MLP. For enabling the model to perform auto-regressive inference, we add a learnable padding parameter α, which is concatenated with the input after being projected by another learnable MLP. The inference process of our model can be represented as follows.

7.4 Train and evaluation of our model

For the training of our model, we split the dataset into a 9:1 ratio for training and validation, respectively. We use the loss function mentioned above and the AdamW optimization algorithm and train the model for 10 epochs with a learning rate of 0.0001. Considering that our dataset is relatively small, we set the batch size to 4 for the first 5 epochs, 2 for the 6th to 8th epochs, and 1 for the 9th to 10th epochs. To facilitate better convergence, we initialized the model parameters with Xavier initialization.

Regarding the evaluation metrics, we considered a vector Q composed of the absolute differences between the prediction and true labels for each percentage in the different scores, and the average value of this vector. We believe that this metric has an intuitive meaning that reflects the performance of the model. The definition is as follows (where "avg" represents the average of each element in the vector):

$$Q = [q_1, ..., q_7] = [\sum_i^N \frac{|\hat{r_1} - r_1|}{N}, ... \sum_i^N \frac{|\hat{r_7} - r_7|}{N}]$$
$$\bar{q} = avg(Q)$$

After training, it can be observed that the loss tends to converge after the 9th to 10th epochs.

The evaluation metrics on the validation set for the 10th epoch showed the minimum error as follows, indicating the good performance of our model. On average, the model had an absolute difference smallest error of only 0.7% for the xxx tries, and the largest error was 4% for a single try. Overall, the average difference was 3%, which is shown in Table 1.

7.5 Uncertainty of our model

When considering the impact of the number of players participating in the game and the number of players choosing the difficult mode, we extended the vector of word-related attributes and dates that we defined to include 13 dimensions from the original 11 dimensions. After retraining the model and observing the convergence of the loss function, we found that there was no significant performence improvement in our evaluation metrics, which is shown in Table 2.

Table 1: LSTM model's performance

avg error	1 try error	2 tries error	3 tries error	4 tries error	5try error	6 tries error	7tries error
2.3 %	0.7%	1.7%	3.9%	2.2%	3.4%	2.6%	1.3%

Table 2: Considering uncertainities

avg error	1 try error	2 tries error	3 tries error	4 tries error	5tries error	6 tries error	7tries error
2.2 %	0.76%	1.8%	3.7%	2.2%	3.3%	2.4%	1.3%

The reason is that the number of participants and the number of people selecting the hard mode may also have some correlation, which are the uncertainties of our model.

7.6 Predict the "EERIE"

As the model considering the number of players did not show significant performance improvement and our prediction of the number of players may have errors, we use a model that does not consider the number of players to predict the score of "EERIE" as follows. As seen in the performance on the validation set, we believe that our predicted results error no more than 4%, with an average error of 3% for each prediction of percentages of the score, which is shown in Table 3.

Table 3: Prediction of "EERIE"

1 try	2 tries	3 tries	4 tries	5tries	6 tries	7tries
1%	10%	30%	32%	19%	7%	1%

8 Problem IV:

For this problem, we divide the process into four steps for completion in a systematic manner using a structured approach.

8.1 Defining Word Difficulty

As a relatively abstract concept, there is no widely recognized metric for measuring the difficulty of words. Therefore, we chose to characterize the difficulty level of each word by considering both the number of attempts made by players and the proportion of players who selected the difficult mode. From the attached data, we can easily obtain a vector Mi for the i−th word:

$$Mi = [r_{1i}, r_{2i}, r_{3i}, r_{4i}, r_{5i}, r_{6i}, r_{7i}] \quad i \in (1, 356)$$

At the same time, we calculate a ratio vector Qi that represents the proportion of players who selected the difficult mode for a specific word, by dividing the number of players who selected the difficult mode for that word by the total number of players.

$$Qi = \frac{Nh_i}{N}, \quad i \in (1, 356)$$

By concatenating the vector Mi (355 x 10) and the vector Qi (355 x 1), we obtain a matrix M (355 x 11) that describes the difficulty level of all given words. In order to classify all the words according to their difficulty levels, we first need to define labels that describe "difficulty". Here, we choose to use the K-means algorithm to cluster the 355 samples and11 features of matrix M in an unsupervised manner, dividing all the samples into threeclasses that represent words with low, moderate, and high difficulty levels. The idea be-hind the K-means algorithm is very simple. Given a set of samples, the algorithm par-titions the samples into K clusters based on the distance between them. The algorithmtries to make the points within each cluster as close together as possible, while keepingthe distance between clusters as large as possible. In mathematical terms , assuming theclusters are denoted as (C1, C2, ..., Ck), our goal is to minimize the sum of squared errorsE:

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|_2^2$$

where μi is the mean vector of cluster Ci, sometimes referred to as the centroid, and can be expressed as:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

The resulting cluster centers are shown in the following table:

Table 2: cluster center

class	coordinates
1	[0.26 , 4.02 , 20.19 , 35.42 , 26.47 , 11.53 , 1.94 , 0.08]
2	[0.80 , 9.32 , 30.65 , 33.74 , 17.91 , 6.46 , 1.08 , 0.07]
3	[0.28 , 2.87 , 12.57 , 25.65 , 28.79 , 21.66 , 8.13 , 0.07]

As we can see from the table, the cluster centers of the three classes differ significantly for the same feature, indicating that the model can effectively distinguish between the

three classes. We can use t-SNE to visualize the clustering results after reducing the dimensionality of the data, as shown in the following figure:

As we can see from the figure, the three sets of points of different colors are clustered in different areas, indicating that the clustering effect is good.

The clustering results obtained are summarized in the following table:

Table 3: cluster result

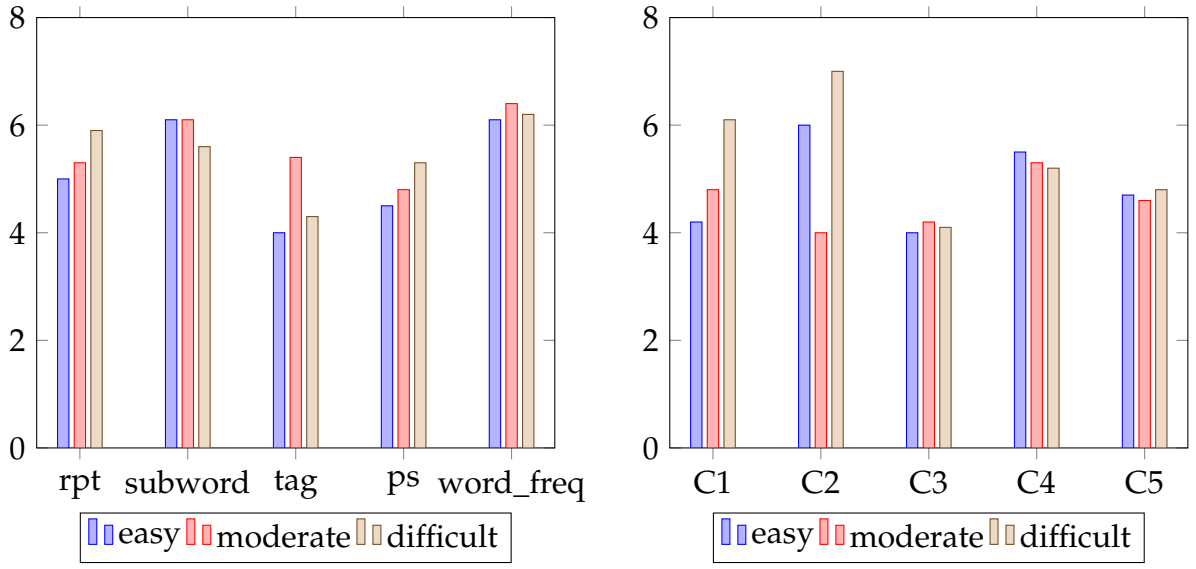
class1,class2,class3
154,133,68

Therefore, for the i−th word, Ci is used to represent its difficulty, where $Ci \in [0, 3)$.

8.2 Explore the indicators that affect the difficulty of words

For each category, we analyze the characteristics of the words in this category. For the different attributes of the words, the average value of the attribute values of the words in different difficulty categories within each attribute is calculated. Taking the attribute "rpt" (repeated letters) as an example, Ni represents the number of words in each difficulty category.we define Avgrpti as follows:

$$A_{vgrpt_i} = \sum r_{pt_i}/\mathrm{Ni} \quad i = 0, 1, 2$$



The line chart indicates that the *rpt* ,*ps* and *C*1 exhibit a monotonically increasing trend as the difficulty level increases, suggesting that for class2 (hard), these features have a more significant impact than other features. For class1 (moderate), *subword*5 and *word_freq* have significantly larger values than the other two classes, indicating that these

three features mainly affect the classification of moderate difficulty words. For easy words, i.e., class0, it is noticeable that the values of the features $C4$ are relatively large, showing that that feature has a significant impact on the classification of low-difficulty words.

8.3 Random forest (RF) classifier

After completing the task of determining the difficulty level of each word, the next step is to design a classifier model.

We chose to use a random forest model to build the classifier. Random forest (RF) classifier is an ensemble classifier that produces multiple decision trees,using a randomly selected subset of training samples and variables[1]

The basic learning algorithm of random forest, decision tree, is a common machine learning method. Generally, the decision tree follows the idea of selecting attributes: at each partition, samples of the same class should go to the same branch as much as possible, and samples of different classes should go to different branches as much as possible. That is, as the partitioning process proceeds, we hope that the samples contained in the branching nodes of the decision tree belong to the same category, and the purity of the nodes becomes higher and higher. For this problem, the decision tree model tends to place words with similar features in nearby or identical branches during training, while placing words with low similarity in different branches.

Given our data sample set X (a matrix of vectors), we use information entropy Ent(X) to measure the purity of the sample set, where p_{k}(k=1,2,...,|M|) represents the proportion of the k-th class samples in the sample set X. The smaller Ent(X), the higher the purity of X.

The algorithm is non-parametric and can efficiently deal with large, complicated datasets without imposing a complicated parametric structure. [2]

$$\mathrm{Ent}(D) = -\sum_{k=1}^{|\mathcal{Y}|} p_k \log_2 p_k$$

Algorithm 1 Decision tree's algorithm

Input: "Property set A,Training set D"
procedure:TreeGenerate(D,A)
Generate node
if *The samples in D all belong to the same category C* **then**
node is marked as a leaf node and its category is marked as the class with the largest number of samples in D **return**
end
Select the optimal partition attribute a* from a
for *Each value of a* a* ':* **do**
Generate a branch for node; Let Dv represent the subset of samples in D that values a* 'over a*
if *Dv is null* **then**
The branch node is marked as a leaf node, and its category is marked as the class with the most samples in D **return**
else
Take TreeGenerate(Dv, A "a.") as the branch node
end
end
Output: function TreeGenerate(D,A)

The random forest selected in this study employs bagging with CART decision trees as weak learners. Due to its randomness, random forest can significantly reduce model variance. Thus, it generally does not require additional pruning to achieve good generalization performance. The training process can be outlined as follows.

Algorithm 2 Random forest's procedure

Step1. Assuming the dataset is D and the label set is A, the decision tree to be constructed is denoted as tree.
Step2. By performing random sampling with replacement, the dataset and label set are sampled to obtain sampleD and sampleA respectively.
Step3. A training set subD is constructed by randomly selecting K features from the sampled data.
Step4. The decision tree is fitted using subD and sampleA.
Step5. All the trees are added to the model to form a forest.
Step6. The model is returned.

To evaluate the accuracy of the model, mean squared error (MSE) is selected as the loss function, and accuracy score is used as the accuracy metric in this experiment.

$$\begin{aligned} \mathrm{Loss}_{\mathrm{MSE}}(y,\hat{y}) &= \frac{1}{m}\arg\min_{(w,b)}\sum_{i=1}^{m}\left(f\left(x_i\right)-y_i\right)^2 \\ &= \frac{1}{m}\arg\min_{(w,b)}\sum_{i=1}^{m}\left(y_i-wx_i-b\right)^2 \end{aligned}$$

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Where TP represents the situation in which both prediction and label are positive exam-ples for each category; TN represents the situation in which both prediction and label arenegative examples; TP and TN are the situations in which the prediction is correct; FP isthe situation in which the prediction is positive but the actual table tag is positive; FN isthe situation in which the prediction is negative and the actual label is positive; FP andFN are the situations in which the prediction is wrong.

After multiple training iterations, the optimal random forest classifier model achieves an accuracy of 0.72 and an MSE of 0.44. We consider this performance acceptable for the three-class classification task. Furthermore, a bar chart of feature importances for the 10-word attributes can be obtained.

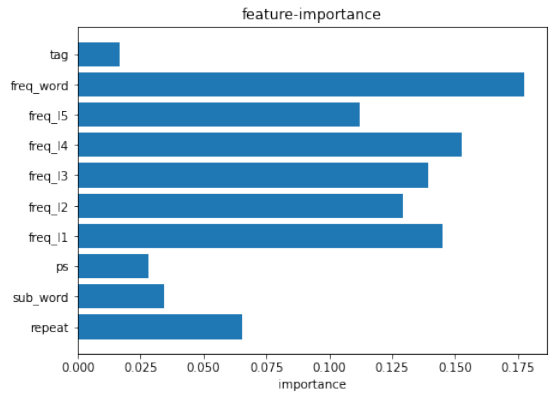


Figure 11: Q3classfier

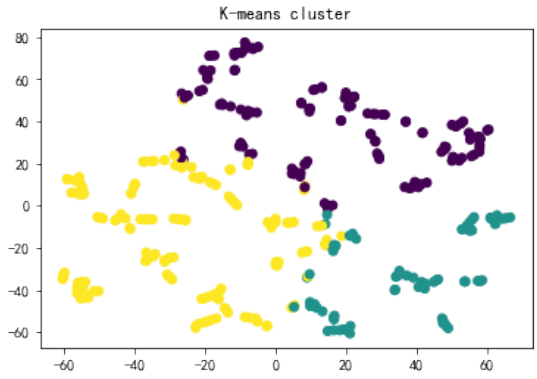


Figure 12: cluster

As shown in the chart, among the 355 words provided in the attachment, the *word_freq* of a word in the English corpus has the greatest impact on the word difficulty, considering various difficulty levels. The frequency of the first and fourth letters ($freq\_C1$ and $freq\_C4$) follows closely behind in terms of their impact. The *tag* and *ps* have the lowest impact on word difficulty.

8.4 How hard is EERIE?

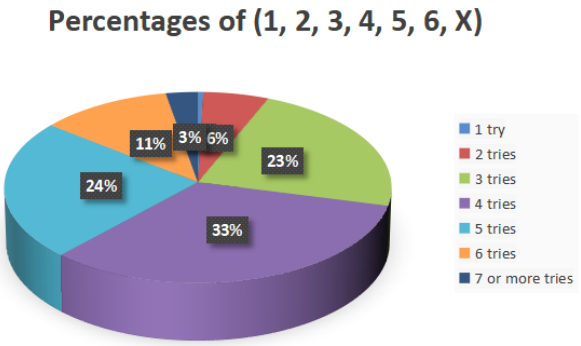
Using the trained random forest classifier model, the 10 attributes of the word EERIE are normalized according to the data provided in the attachment and represented as follows: [1 , 0 , 0 , 1 , 1 , 0.46 , 0.56 , 1 , 0.001 , 1] By inputting the vector into the aforementioned random forest classifier model, the result is "1", which indicates that the difficulty of the word is moderate. Based on the model accuracy mentioned earlier, we consider this classification result to be reliable.

9 Exploring More Features of Word in Wordle after Our Modeling

In the data table, we can see the distribution of the number of attempts made by users for each individual word. Furthermore, we want to understand the overall answer performance of users for all words in the vocabulary. We calculated the average number of tries for each column in the data table, and obtained the following results:

	Average
1 try	.46
2 tries	5.79
3 tries	22.65
4 tries	32.92
5 tries	23.71
6 tries	11.57
7 or more tries	2.80

(a) Image on the left



(b) Image on the right

Figure 13: Two images

We can visualize from the pie chart that on average, users can guess the words correctly in 3, 4, or 5 attempts for the entire word list.

10 Model Analysis

10.1 Strength and Weakness

```
The ARIMA model is simple; LSTM model has complex structure and
can take into account the temporal characteristics of data.Random
forest classifier has good classification effect, but unstable.
```

10.2 Ablation study for Bert Embedding

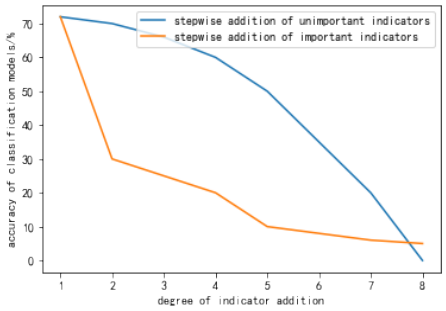
To verify whether the LSTM model using Bert word embeddings in Problem 3 is effective, we trained a multilayer perceptron with 2-6 layers and 11 neurons per hidden layer to predict the scores, as shown in the figure. The performance of the best model is shown below, with an error about twice that of our model. This indicates that using Bert's semantic representations does indeed enable the model to learn more features for the task.

Table 4: MLP for ablation study

avg error	1 try error	2 tries error	3 tries error	4 tries error	5tries error	6 tries error	7tries error
4.5 %	2.6%	3.8%	7.7%	4.2%	6.3%	4.4%	3.3%

10.3 Sensitivity analysis of our random forest model

We also conduct a perturbation experiment on our random forest model. In the positive experiment, we add appropriate normally distributed random values to each variable in ascending order of importance, to the important features that the random forest identified. In the negative experiment, we added the same randomly generated values in descending order of importance. The average results of the accuracy after 10 experiments as the percentage of variables perturbed are shown below:



It can be seen that our model is more sensitive to important features and more robust to non-important features, based on the average results of the accuracy decreasing with the percentage of perturbed variables after 10 experiments.

11 Conclusion

In this project, we conducted a detailed analysis of the given dataset and defined rich word attribute indicators combined with the game background. By building models such as ARIMA, LSTM, and Random Forest Classifier, we conducted multidimensional and in-depth mining of the relevant data of the Wordle game. We can predict the data we are interested in, such as future game players and the difficulty of guessing word attributes, and have a good model accuracy. Finally, I reported our model analysis results to the Wordle game editor at The New York Times and provided suggestions for their subsequent question-setting.

12 Our Letter for Designer of Wordle

letter

To: Puzzle Editor of the New York Times.
From: Team 2316007
Date: February 21th, 2023
Subject: Some suggestions about Wordle

We are very interested in Wordle Puzzle and it is our great honored to inform you of some results for your company after data analysis and modeling.

First, we focused on the changes in the daily number of players over time. We found that the number of players gradually increased in the first 30 days after the records started on January 7, 2022, and then gradually decreased. We proposed an ARIMA time series model to predict the number of players in the future. We predict that after December 31, 2022, the number of players will gradually stabilize at around 20,000. The number of players on March 1, 2023, will be between 19,500 and 20,500. We also found that the properties of the words do not affect the proportion of players who choose the difficult mode of the game and the total number of players. The proportion of players in the difficult mode is related to time. Overall, we believe that the number of players and the proportion of players in the difficult mode of Wordle game will not change significantly in the near future.

Next, we successfully built an LSTM model to predict scores for 1-7 tries. The model's prediction error is within 3%. With this model, you can get an approximate understanding of how users will perform on a chosen word for 1-7 tries.

Then, we built a model to evaluate the difficulty of words. We clustered 355 words into three difficulty levels using a clustering model. By analyzing the characteristics of word properties in each difficulty level, we found that the presence of repeated letters and subwords in a word significantly makes the puzzle easier. With our model, we can accurately judge the difficulty of a specified word, achieving an accuracy rate of 73%. You can use this model to judge the difficulty level of chosen words and adjust your word selection strategy accordingly.

References

[1] Belgiu, M., Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. ISPRS Journal of Photogrammetry and Remote Sensing, 114, 24-31.

[2] Song, Y. Y., Ying, L. U. (2015). Decision tree methods: applications for classification and prediction. Shanghai Archives of Psychiatry, 27(2), 130.

[3] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).