Research on Predicting Wordle Word Attempt Counts and Word Difficulty Classification Based on Machine Learning and K-Means Clustering Techniques

Yanhui Liang [1, #], Junan Long [1, #], Chengyan Tan [2, #], Dejun Wang [1, #, *]

[1] College of Computer Science, South-Central Minzu University, Wuhan, China, 430074

[2] College of Management, South-Central Minzu University, Wuhan, China, 430074

* Corresponding Author Email: dejun@scuec.edu.cn

[#]These authors contributed equally.

Abstract. Wordle is a word-guessing mini-game that has gained tremendous popularity in recent years. As a result, there is a growing interest in analyzing Wordle's data to assist developers in problem-solving, predicting its popularity, and determining future directions. In this paper, we collected game data from Wordle users who shared their scores on Twitter between January 7, 2022, and December 31, 2022, using a Python program. We utilized machine learning and clustering techniques to develop models for predicting the number of word attempts and evaluating the game's difficulty grading. Subsequently, we conducted model testing using the word "EERIE" as an example to select the optimal model and verify its predictive accuracy. The research findings not only assist developers in enhancing user experience but also contribute to the broader field of game analytics, providing valuable insights for game design and player engagement. Ultimately, our study provides crucial data analysis support for the development of Wordle and reveals the potential and future directions of word-guessing games in the entertainment industry.

Keywords: Wordle, Machine learning, XGBoost, PCA, K-means.

1. Introduction

1.1. Background and motivation

In recent years, the popularity of word games has grown significantly [1], with Wordle emerging as a particularly captivating example. Wordle is a web-based word-guessing game that challenges players to guess a five-letter word within a limited number of attempts [2]. As the pace of life accelerates, such games provide players with a convenient way to engage in short, stimulating activities during their downtime. However, understanding the factors that contribute to user engagement and predicting game performance are crucial for ensuring the game's sustained success.

This study aims to address these concerns by employing machine learning and clustering techniques to analyze Wordle data. By focusing on predicting the number of word attempts and assessing the difficulty grading of the game, we can provide insights that will enable game developers to optimize the game's design and maintain user interest.

1.2. Overview of Wordle

Wordle is a daily word-guessing game where players attempt to uncover a five-letter word within six tries [3]. With each attempt, the game provides feedback on the accuracy of the guessed letters and their placement within the target word. The game's appeal lies in its simplicity, accessibility, and the ability to engage players in a challenging yet entertaining activity.

1.3. Objectives and scope of the study

The primary objectives of this study are as follows:

1) Predict the number of word attempts in the game using recursive chain multiple objective regression and machine learning algorithms, such as XGBoost, to gain insights into factors that influence player performance and engagement.

2) Assess the difficulty grading of words in Wordle by applying dimensionality reduction techniques like PCA and clustering methods, such as K-means clustering. Additionally, we aim to validate the difficulty grading predictions using a CNN model.65

Our study will provide valuable insights into the game's design and user engagement, enabling developers to make data-driven decisions that enhance the overall gaming experience for players.

1.4. Flow chart of the study

The flow chart of the study is shown in Fig. 1.

2. Literature Review

2.1. Related work on Wordle or similar word games

Wordle has garnered significant attention from both the public and the research community due to its simplicity and engaging gameplay. While there is limited literature specifically focused on Wordle, similar word-guessing games have been studied extensively, providing insights that may be applicable to Wordle.

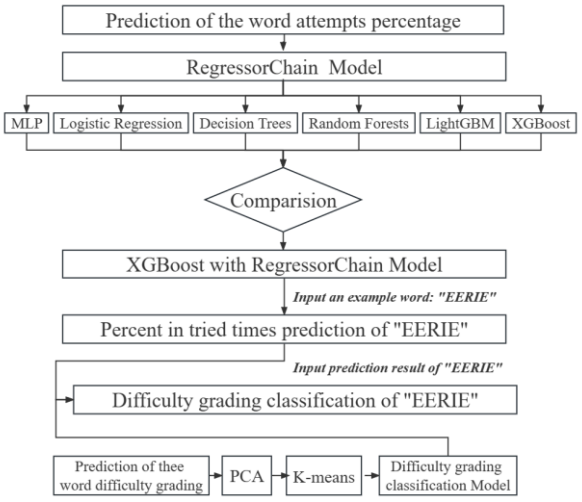

Fig. 1 Flow chart of the whole study

For example, Keim et al. investigated the cognitive processes involved in playing the word game Ghost, a game in which players take turns adding letters to a growing word fragment [4]. The study aimed to understand the strategies players use to guess words and adapt their gameplay based on the game state. In another study, Hanna et al. explored the use of computational models to predict human performance in the game Hangman, focusing on factors that influence the likelihood of guessing a word correctly [5].

These studies demonstrate the value of employing machine learning and clustering techniques to analyze word games and provide insights that can help optimize their design and user engagement. In this study, we aim to apply these techniques to Wordle, focusing on predicting the number of word attempts and assessing the difficulty grading of the game.

2.2. Application of Machine learning and clustering techniques in game analysis

In recent years, machine learning and clustering techniques have become increasingly popular tools for analyzing and enhancing various aspects of games. These techniques have been employed in several areas, including player behavior analysis, game outcome prediction, and game difficulty adjustment.

For instance, Drachen et al. utilized clustering techniques to identify distinct player behavior patterns in the online multiplayer game Tomb Raider: Underworld [6]. Another study by Periáñez et al. applied machine learning algorithms to predict player churn in the mobile game Puzzle & Dragons, enabling developers to target at-risk players and improve retention [7]. Furthermore, Hullett et al. employed machine learning techniques to adjust the difficulty of game levels in response to player performance, creating a more engaging and personalized experience [8].

3. Data Collection and Preprocessing

3.1. Data Source

In order to conduct our analysis, we collected game data from Wordle users who reported their scores on Twitter. The dataset was compiled by crawling user-generated tweets from January 7, 2022, to December 31, 2022. The collected data includes the following information for each entry, and saved as " Data Wordle.xlsx":

·Date
·Contest number
·Word
·Number of reported results
·Number in hard mode
·Percent in tried times (1 try, 2 tries, …, 7 or more)

3.2. Data Preprocessing

Before analyzing the data, we took the following preprocessing steps to ensure data quality and suitability for our study [9]:

1) Removing incorrect data

We eliminated entries with word lengths different from 5, as they contradicted Wordle's word mechanism.

2) Counting repeated letters

We used Python's Counter function to count letter occurrences in each word and saved the maximum count as a data feature.

3) Applying one-hot encoding

We used one-hot encoding to represent words as sparse features, which has benefits such as saving storage space and improving training efficiency.

4) Lexical annotation

We utilized the nltk package's pos_tag function to annotate the lexicality of each word, allowing us to analyze the impact of lexicality on Wordle data.

5) Data Normalization

we used the StandardScaler function in Python to normalize the prediction data. The StandardScaler function applies a normalization process, which is based on Formula 18. In this formula, μ represents the mean value of all sample data, and σ represents the standard deviation of all sample data.

$$x^*=(x-\mu)/\sigma \tag{1}$$

4. Model construction

4.1. Model For Prediction of The Word Attempts Percentage

We attempted to use the RegressorChain model in combination with multiple other machine learning models to compare their prediction error, performance on the training set, and generalization ability. We aimed to select the optimal model for prediction.

1) RegressorChain Model

The RegressorChain model is a method for constructing multi-target regression models by applying multiple single-target regression models to different regression targets [10]. In the regression chain, each single-target regression model is trained for a different regression target, while each regression target is influenced by the output of the previous regression target. This method combines different regression problems and considers their correlation and influence to improve the accuracy and robustness of the entire model.

In a RegressorChain, each target variable Y_i (i = 1, 2, ..., n) is predicted using a base regressor R_i. Here are general steps of the RegressorChain method using formulas:

Step 1: Train the first base regressor R_1 using input features X to predict the first target variable Y_1:

$$R_1 = train\ (X, Y_1) \quad (2)$$

Step 2: Train the second base regressor R_2 using input features X and the predicted Y_1 (Y_1_p) to predict the second target variable Y_2:

$$R_2 = train\ ([X, Y_1\_p], Y_2) \quad (3)$$

Step 3: Predict Y_2 using the trained R_2:

$$Y_2p = R_2.predict([X, Y_1\_p]) \quad (4)$$

Step 4: Repeat steps 3 and 4 for each target variable Y_i (i = 3, 4, ..., n) by training a base regressor R_i using input features X and the predictions of all previous target variables (Y_1_p, Y_2_p, ..., Y_{i-1}_p):

$$R_i = train\ ([X, Y_1\_p, Y_2\_p, ..., Y_{i-1}\_p], Y_i) \quad (5)$$

$$Y_i\_p = R_i.predict([X, Y_1\_p, Y_2\_p, ..., Y_{i-1}\_p]) \quad (6)$$

The final RegressorChain model consists of all trained base regressors R_1, R_2, ..., R_n. The basic principle of the RegressorChain model is illustrated in Fig.2.

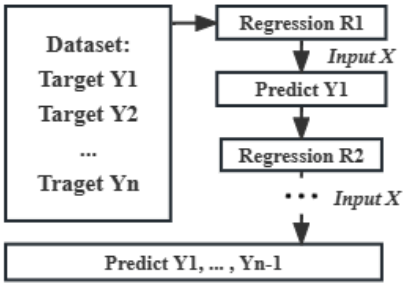

Fig. 2 Flow chart of RegressorChain Model

2) Machine Learning Models

In this study, we use the RegressorChain model to predict the number of word attempts. By applying different machine learning algorithms [11] (including Multi-Layer Perceptron (MLP) [12], Logistic Regression [13], Decision Trees [14], Random Forests [15], LightGBM [16], and XGBoost [17]) to the RegressorChain, we can consider the correlation between different regression targets while retaining the characteristics of individual models. Here are training and prediction formulas of different machine learning models that used in RegressorChain model, the former formula of each model is training, the latter is prediction.

For Multi-Layer Perceptron (MLP) Model:

$$R_i = train\_MLP(X, Y, hyperparameters) \quad (7)$$

$$Yp = predict\_MLP(X, R_i) \quad (8)$$

For Logistic Regression Model:

$$R_i = (X^TX + \lambda I)^{-1}X^TY \quad (9)$$

$$Yp = sigmoid\ (X * R_i) \quad 10$$

For Decision Trees Model:

$$R_i = train\_DecisionTree(X, Y, hyperparameters) \quad 11$$

$$Yp = predict\_DecisionTree(X, R_i) \quad 12$$

For Random Forests Model:

$$R_i = train\_\ RandomForest(X, Y, hyperparameters) \quad 13$$

$$Yp = predict\_\ RandomForest(X, R_i) \quad 14$$

For LightGBM Model:

$$R_i = train\_\ LightGBM(X, Y, hyperparameters) \quad 15$$

$$Yp = predict\_\ LightGBM(X, R_i) \quad 16$$

For XGBoost Model:

$$R_i = train\_\ XGBoost(X, Y, hyperparameters) \quad 17$$

$$Yp = predict\_\ XGBoost(X, R_i) \quad 18$$

4.2. Model For Prediction of The Word Difficulty Grading

1) PCA Principal Component Analysis [19]

Step 1: We need to input the data that has been normalized and transform it into a sample matrix X, according to the question, because the column of tried times data is 7, so we define the number of components p=7.

Step 2: We calculate the covariance matrix of the centralized sample matrix X:

$$C=(XX\text{^}T)/(i*j-1) \quad 19$$

Step 3: we compute the eigenvalues of the covariance matrix C and arranged in descending order, let λp as eigenvalues, and arrange them in the order of the smallest to the largest.

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda p \quad 20$$

Step 4: we compute the eigenvectors corresponding to the eigenvalues and take the eigenvectors corresponding to the first p' eigenvalues to form the projection matrix W.

$$W=w_1,w_2,\ldots,wn \quad 21$$

Step 5: we specify γ as the contribution margin, γk denotes the contribution margin of the *k* th principal component:

$$\gamma=\lambda pkk/(\sum\_(i=1)\text{^}p\ \lambda\_i\ ) \quad 22$$

By computing γ, we summarized each eigenvalue and draw the explained variance graph in Fig. 3.

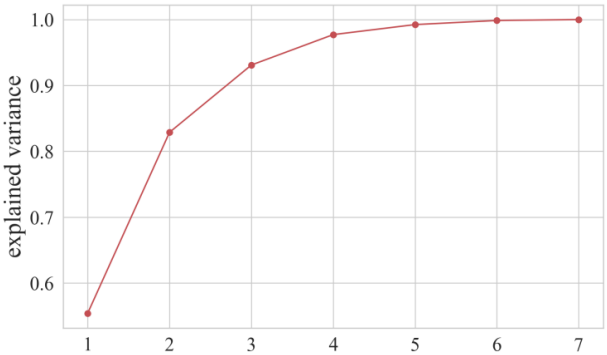

Fig. 3 Explained variance.

We get the number of principal components by determining the inflection points. In this figure, the 2nd, 3rd, and 4th present inflection points, so we set the number of principal components is 3.

Though we can confirm there is 3 principal components, but we can't exactly judge the difficulty of word, then we need to make K-means Clustering to get more details.

2) K-means Model Construction

Step 1: we use elbow method to determine K value, the algorithm is shown below where Ci is the ith cluster, p is the sample points in Ci, mi is the center of mass of Ci (the mean of all samples in Ci), and SSE (sum of the squared errors) is the clustering error of all samples, which represents how well the clustering works.

$$SSE=\sum_{i=1}^{K}\sum_{p\in C_i}|p-m_i|^2 \quad (23)$$

Step 2: we use sample contour method to test K again in order to make the value of K more convincing, assume all sample are all in cluster A and B, the algorithm shows in below. In following formulas, a(i) means the cohesiveness of sample points, b(i) means the minimum value of the sample mean distance of i from other clusters.

After calculating SSE, S, we derived a graph of Iner for SSE and Silhouette score for S, shown in Fig. 4.

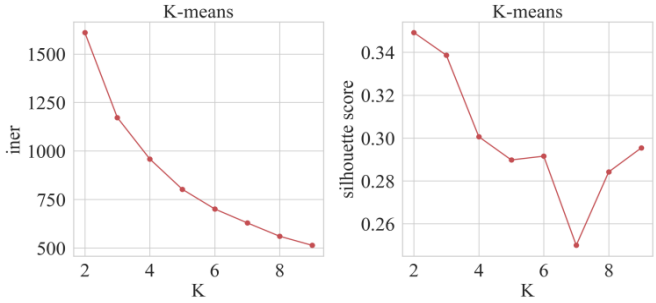

Fig. 4 Iner and Silhouette score

By combining the image and definition analysis, we believe that the value of K should be 3.

Step 3: we use K-means method to divide data into 3 clusters, then we calculate and visualize the clustering results on the graph by writing a Python program, the result figure as shown in Fig. 5. At the same time, we classified the different clusters automatically, and got the situation of different clusters is [0,1,2] = [184,95,80].

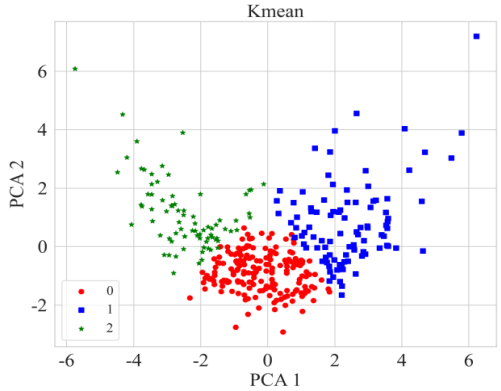

Fig. 5 K-means result.

5. Model test and results

5.1. Prediction Result of The Word Attempts Percentage

We input X (one-hot encoded letters corresponding to each different word) and Y (number of attempts) in order and set the training and test sets in an 8:2 ratio. We tested and calculated the

corresponding mean absolute percentage error of models established above, as shown in Fig. 6 and Fig. 7.

From the figures, it can be seen that the XGBoost model performs relatively stable on both the training and test sets compared to other machine learning models. Therefore, we believe that the combination of RegressorChain and XGBoost performs well in predicting the percentage of attempts. Afterwards, we selected **"EERIE"** as an example word for testing, and the percentage of attempts obtained is shown in TABLE I.

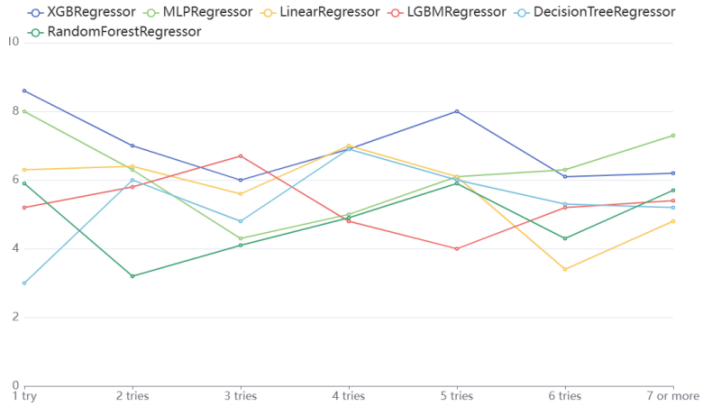

Fig. 6 Machine learning result for train sets

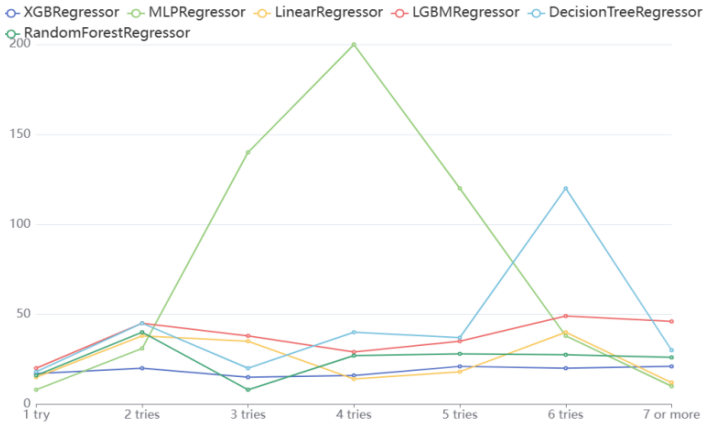

Fig. 7 Machine learning result for test sets

Tab. 1 Table Type Styles

Tried times	Predict percentage
1 try	0%
2 tries	2%
3 tries	19%
4 tries	43%
5 tries	24%
6 tries	11%
7 or more	1%

5.2. Prediction Result of The Word Difficulty Grading

We use the word **"EERIE"** as an example in prediction of the word attempts percentage, we also use it in this part to test our model. After input the one-hot code of **"EERIE"** into the model, we get information from the console, which is shown in Fig. 8.

```
Number of Sample: (array([0, 1, 2]), array([184,  95,  80], dtype=int64))
EERIE class [0]↵
```

Fig. 8 Classification result of "EERIE"

We can easily know **"EERIE"** is in Cluster 0. In order to make the classification of the word more visible and know how difficult it is, we use Seaborn boxplot based on Python to draw box diagram, shown in Fig. 9.

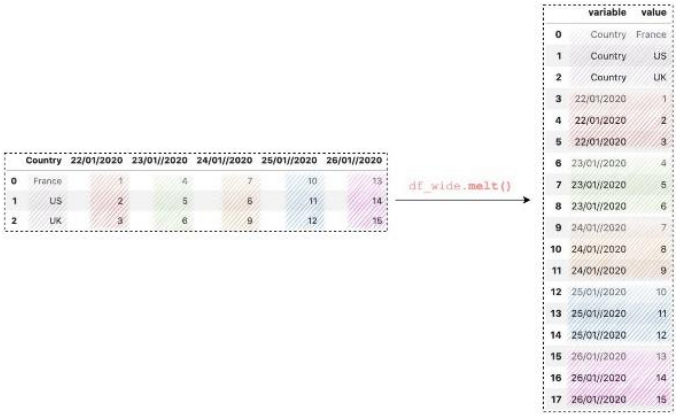

Fig. 9 Dimension transform

Firstly, we have to transform the dimension of the previous data [359*8] to [2513*3], combine the 7 columns with different number of attempts with the categories respectively, in order to transform the pattern to seaborn box painting.

After transformation, the data becomes score of tries and we can easily get the sore diagram, which is also follow normal distribution, shown in Fig. 10 and Fig. 11.

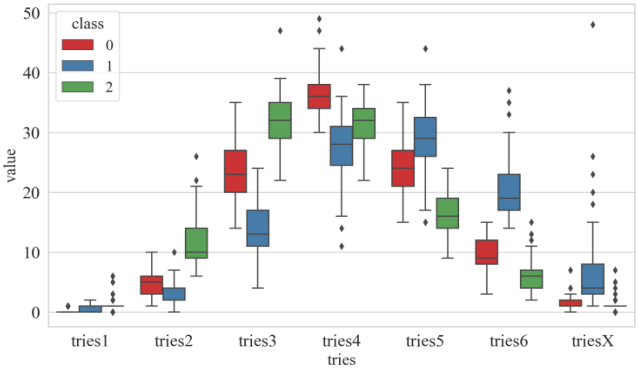

Fig. 10 Seaborn simulate result A

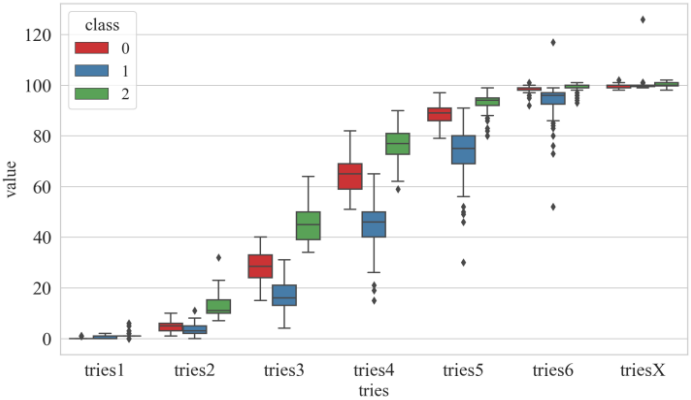

Fig. 11 Seaborn simulate result B.

According to the results of seaborn box plot, we can approximate group 0 as normal level, group 1 as difficult level, and group 2 as easy level. For "EERIE", it is **normal** level.

5.3. Result Examination

In order to further verify the correctness of classification result, we use machine learning algorithms for classification, employing LogisticRegression, DecisionTreeClassifier, RandomForestClassifier, LGBMClassifier, LightGBM, XGBClassifier, and MLPClassifier were trained on the data to derive the corresponding evaluation metric F-values, which were not satisfactory. Therefore, we visualized the training set accuracy, test set accuracy, and F-score, shown in Fig. 12.

Fig. 12 Evaluation of different machine learning model

Our data has 26 features and 359 data, so the dimension to be processed is [26,359]. Since the convolutional neural network CNN has a good use in image processing classification and the convolutional kernel has a good superiority in extracting features, we treat the input data as a one-channel grayscale image and use the convolutional neural network for classification.

We build the neural network using the model in the keras module. We build a total of three one-dimensional convolutional layers and then unfold the convolutional data, and finally build a fully connected layer with three output values corresponding to three difficulties (Difficult, Normal, Easy).

We choose to use tanh as the activation function because tanh has better symmetry and the tanh function centered on 0 is more conducive to improving training efficiency.

Since the derivative of the tanh function is 0 when the absolute value of the input data is large, it needs to be normalized before the data is input to the convolutional neural network. tanh function has better processing ability for the input normally distributed data.

The evaluation metrics of the obtained model results are training set F-value: 0.821, test set F-value: 0.780, this value is significantly better than the machine learning algorithm, so the CNN has a high confidence in the classification of this question.

Then the target word eerie is encoded with One-hot and trained into the neural network, and the result is [0.09,0.72,0.19], and the rating of the second category is significantly greater than the first and third category, so the word EERIE is classified as " normal".

Finally, we used the built model to predict the input data for 500 rounds, and the results are shown in Fig. 13.

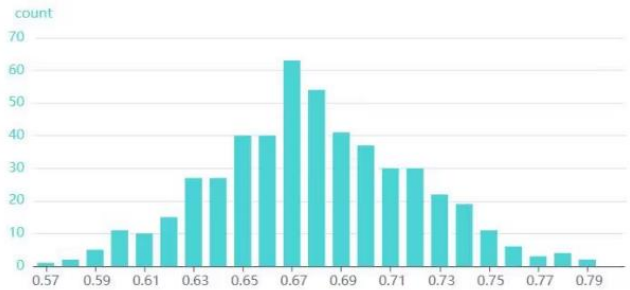

Fig. 13 Distribution of result by using CNN.

After 500 predictions, the prediction results are EERIE for "ordinary" category, and from the graph, the prediction of the possibility of "ordinary" are in about 70%, we predict 500 times the possibility of the second category as the final degree of certainty. The degree of certainty is 0.713.

6. Conclusion

6.1. Summary of the Study

In this study, we explored a novel method for predicting the number of words try times for Wordle players and for difficulty grading of word in Wordle. Firrstly, we collected a large amount of player interaction data of Wordle from the Twitter. In the try times prediction aspect, we adopted the RegressorChain model and experimented with various machine learning algorithms, including MLP, Logistic Regression, Decision Tree, Random Forest, LightGBM, and XGBoost. The results showed that the combination of XGBoost and RegressorChain provided the best prediction accuracy. For difficulty grading, we used a combination of PCA and K-means for clustering and difficulty grading, and finally we applied a CNN model to validate the classification accuracy.

6.2. Future Prospects

There are several directions for future work and potential improvements for this study:

1) Expand the dataset

To further enhance the performance and generalization of the model, it would be beneficial to collect more data from Wordle players with diverse levels and backgrounds.

2) Explore other features

Investigating other relevant features, such as player engagement or cognitive abilities, might provide additional insights for the predictive models and potentially improve their accuracy.

3) Evaluate other machine learning models

Although the combination of XGBoost and RegressorChain performed best in this study, it would be worth exploring and comparing other advanced machine learning and deep learning models, such as neural networks or transformers, to further optimize the predictive performance.

4) Personalized game recommendations

Based on the difficulty grading predictions, future work could involve developing a personalized game recommendation system tailored to the individual needs of players to optimize their gaming experience.

5) Investigate the impact of various gaming strategies

Different gaming strategies may have different effects on the performance of Wordle players. Future research could explore the relationship between gaming strategies and player progress, providing valuable insights for game developers and educators.

By addressing these potential improvements and expanding research in these directions, we believe our study can make significant contributions to providing effective and personalized gaming experiences for Wordle players worldwide.

References

[1] Anderson B J, Meyer J G. Finding the optimal human strategy for Wordle using maximum correct letter probabilities and reinforcement learning[J]. 2022.DOI:10.48550/arXiv.2202.00557.

[2] Littman, M. L., & Keim, G. A. (2022). Optimal Wordle Strategies. arXiv preprint arXiv:2202.00565.

[3] Feinman, J. (2022). Cracking the Wordle: The viral word game, explained. Vox. Retrieved from https://www.vox.com/22913342/wordle-explained-history-rules-strategy.

[4] Keim, D. A., Mansmann, F., Schneidewind, J., & Ziegler, H. (2017). Visual analytics: Scope and challenges. In Visual data mining (pp. 76-90). Springer, Berlin, Heidelberg.

[5] Hanna, J., & Richards, B. (2019). Investigating human performance in the game of Hangman. In Proceedings of the 41st Annual Conference of the Cognitive Science Society (pp. 1680-1686). Cognitive Science Society.

[6] Drachen, A., Sifa, R., Bauckhage, C., & Thurau, C. (2012). Guns, swords and data: Clustering of player behavior in computer games in the wild. In 2012 IEEE Conference on Computational Intelligence and Games (CIG) (pp. 163-170). IEEE.

[7] Periáñez, Á., Gao, Y., Chen, Z., & El-Nasr, M. S. (2016). Churn prediction in mobile social games: Towards a complete assessment using survival ensembles. International Journal of Computer Games Technology, 2016, 1-15.

[8] Hullett, K., Nagappan, N., Schuh, E., & Hopson, J. (2014). Empirical analysis of user data in game software development. In Proceedings of the 1st International Workshop on Games and Software Engineering (pp. 36-39). ACM.

[9] Alexandropoulos, S. A. N., Kotsiantis, S. B., & Vrahatis, M. N. (2019). Data preprocessing in predictive data mining. The Knowledge Engineering Review, 34, e1.

[10] Reyes, Oscar, and Sebastián Ventura. "Performing multi-target regression via a parameter sharing-based deep network." International journal of neural systems 29.09 (2019): 1950014.

[11] Weng, W., Wang, D. H., Chen, C. L., Wen, J., & Wu, S. X. (2020). Label specific features-based classifier chains for multi-label classification. IEEE Access, 8, 51265-51275.

[12] You, Jiaxuan, Zhitao Ying, and Jure Leskovec. "Design space for graph neural networks." Advances in Neural Information Processing Systems 33 (2020): 17009-17021.

[13] Vimal B. Application of Logistic Regression in Natural Language Processing[J]. International Journal of Engineering and Technical Research, 2020, V9(6). DOI:10.17577/IJERTV9IS060095.

[14] Blanc G, Lange J, Tan L Y. Top-down induction of decision trees: rigorous guarantees and inherent limitations.2019[2023-06-12]. DOI:10.48550/arXiv.1911.07375.

[15] Schonlau M, Zou R Y. The random forest algorithm for statistical learning [J]. Stata Journal, 2020, 20.DOI:10.1177/1536867X20909688.

[16] Yan, J., Xu, Y., Cheng, Q., Jiang, S., Wang, Q., Xiao, Y., ... & Wang, X. (2021). LightGBM: accelerated genomically designed crop breeding through ensemble learning. Genome Biology, 22, 1-24.

[17] Nalluri, M., Pentela, M., & Eluri, N. R. (2020). A Scalable Tree Boosting System: XG Boost. Int. J. Res. Stud. Sci. Eng. Technol, 7, 36-51.

[18] Benassi M, Garofalo S, Ambrosini F, et al.Using Two-Step Cluster Analysis and Latent Class Cluster Analysis to Classify the Cognitive Heterogeneity of Cross-Diagnostic Psychiatric Inpatients[J].Frontiers in Psychology, 2020, 11:1085.DOI:10.3389/fpsyg.2020.01085.

[19] Markel Rico-González.Training Design, Performance Analysis, and Talent Identification—A Systematic Review about the Most Relevant Variables through the Principal Component Analysis in Soccer, Basketball, and Rugby [J]. International Journal of Environmental Research and Public Health, 2021, 18.DOI:10.3390/ijerph18052642.