Sentiment Analysis: LDA and Predicting Genshin Impact's Ratings using Reviews

LSE Candidate ID: 45672

April 2023

Abstract

This study explores the use of natural language processing (NLP) to analyze reviews and predict ratings. Specifically, I conduct a case study on the popular game Genshin Impact using data from Apple Store and Google Play reviews. I employ Latent Dirichlet Allocation (LDA) for topic modeling and analyze the key topics discussed in reviews to identify factors that Genshin Impact is beloved by players. The results demonstrate that Genshin Impact's stunning graphics, well-designed characters, rewarding mechanics, etc. are among the factors contributing to its widespread likability. Furthermore, I apply several machine learning algorithms, including K-Nearest Neighbors, Multinomial Naive Bayes, RandomForest, and SVM, to predict ratings based on reviews and compare their performance. The results show that SVM achieves the best performance, with an accuracy of around 0.66. These findings highlight the potential of NLP and machine learning techniques for analyzing app reviews and providing valuable insights to developers.

1 Introduction

Genshin Impact is a popular open-world action role-playing game developed by miHoYo. It was released in September 2020 for multiple platforms, including PC, PlayStation, and mobile devices (Wikipedia 2023). The game is set in the fantasy world of Teyvat, where players explore the open world, battle enemies, and collect various items and characters. This action role-playing game (RPG) genre has captured the world's attention and made it the biggest international launch in Chinese game history. The reviews from players can directly show players' thoughts on the game, and ratings can reflect the overall performance of the game. In this project, the data I used are the reviews of Genshin Impact under Google Play and Apple Store. LDA is used to do topic modeling and machine learning methods are used to predict the ratings [1]. This study aims to answer the following questions:

1. Which words are more likely to cause players to give a lower rating, and which words are more likely to cause players to give a higher rating?
2. What topics are mainly discussed in the reviews? Is there any difference between reviews under Google Play and the reviews under Apple Store?
3. What are the possible reasons that make Genshin Impact so popular?
4. Are there any models that can be used to predict the ratings based on reviews? Which model performs well in predicting? Why?

2 Motivation

Reviews play a significant role in our daily lives, as people often express their emotions and opinions through reviews after using new products or trying new things. While there has been extensive research on analyzing reviews of products and services (Guo, Barnes, and Jia 2017), there is not as much research on game reviews. However, analyzing game reviews can be valuable because there is often a gap between the opinions of game developers and players. Game reviews provide players with valuable information, and companies can also use online reviews to gauge feedback on their games. Due to the large number of reviews and the amount of information they contain, topic modeling can be used to extract key insights. This enables players to quickly extract important information without having to go through every review, and it also provides companies with another source of information to inform their product decisions.

Additionally, app ratings indicate the performance of the app. High ratings indicate better performance, which can convince new users to download the app. Therefore, I am also interested in exploring the relationship between reviews and ratings. Previous studies have shown that some machine learning methods such as logistic regression, random forest, etc. perform well on text reviews classification (Pranckevičius and Marcinkevičius 2017). Inspired by this, I will apply some machine learning models to predict the ratings using the text of the reviews on Genshin Impact.

3 Data

3.1 Data Scraping

1. The code for my project: https://github.com/jtyhhsh/Genshin_Impact_Analysis

The data was scraped from Google Play and Apple Store on April 7th using the *google_play_scraper* and *app_store_scraper* packages in Python. By specifying the *app_name* as genshin-impact, the language as English, and sorting the reviews by newest first, I obtained the 4000 latest Apple Store reviews and 4000 latest Google Play reviews of Genshin Impact, totaling 8000 reviews. Data is saved into 2 CSV files, one for reviews on Apple Store, and one for reviews on Google Play. Each CSV file comprises 4000 records which have two columns namely 'content which has the reviews, and 'score' which shows the rating the user gave (range: 1-5). A snapshot of raw data is shown in Figure 1.

	content	score
0	Hoyovorse did you make genshin or am I imagining this game?	5
1	This game is so cool but it is too big	5
2	This game is awesome. This game is perfect for people who like adventure and exploring. It is also kind of challenging and gets harder as you leve...	5
3	Cool	3
4	Great game,but sometimes bugs don't let me enter it says wait for the developer update with the bug fixes thingie	4
5	Reminds me of Zelda but more...	5
6	Couldnt even start playing. After the cutscene with traveller and paimon, I couldnt even move! Then the sound of the ringing kept replaying and re...	2
7	nice game	5
8	I like the game the characters, the lore but my only problem is everytime i use ayakas auto attacks it glitches	2
9	Love it so much I don't know how this is free to play!!!! Really fun to play, but there's a problem. I delete some stuff so I can install the stuf...	4

Figure 1: Snapshot of Raw Data under Analysis

3.2 Data Pre-processing

The text data is pre-processed by the following steps: firstly, all strings are converted to lowercase; then, all punctuation marks, numbers, and stop words are removed from the strings. Finally, stemming is applied to obtain the root form of each word. Prior to pre-processing, I found the most frequent words in 8000 reviews are: "the", "and", "I", "to", "a", etc (shown in Figure 2). These words are typically meaningless for analysis. However, after pre-processing the data, the most frequent words become "game", "play", "charact", "get", etc. which are useful words for further analysis (shown in Figure 3).

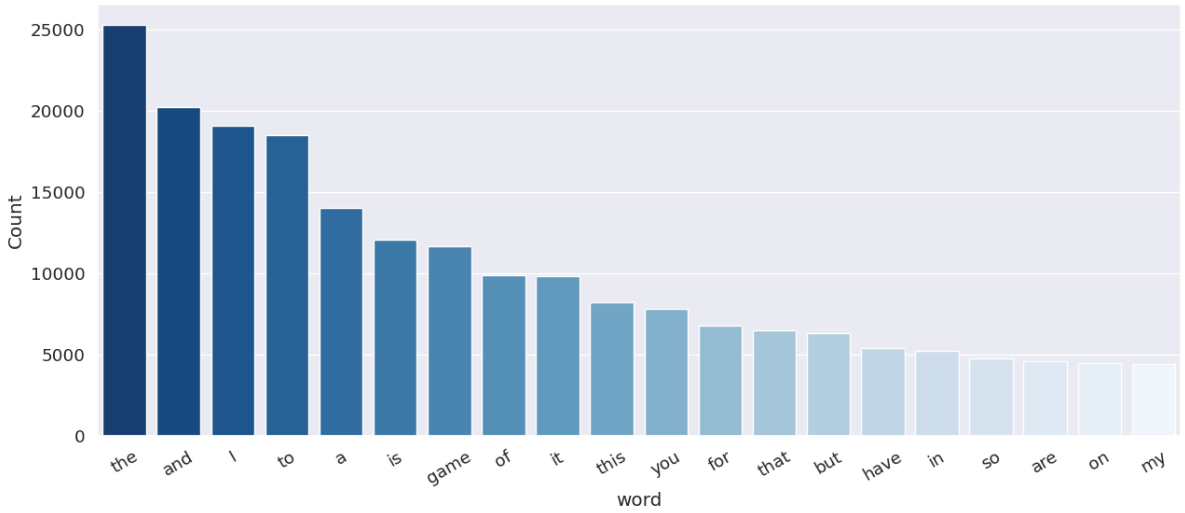

Figure 2: Top 20 Most Frequently Occurring Words before Data Pre-processing

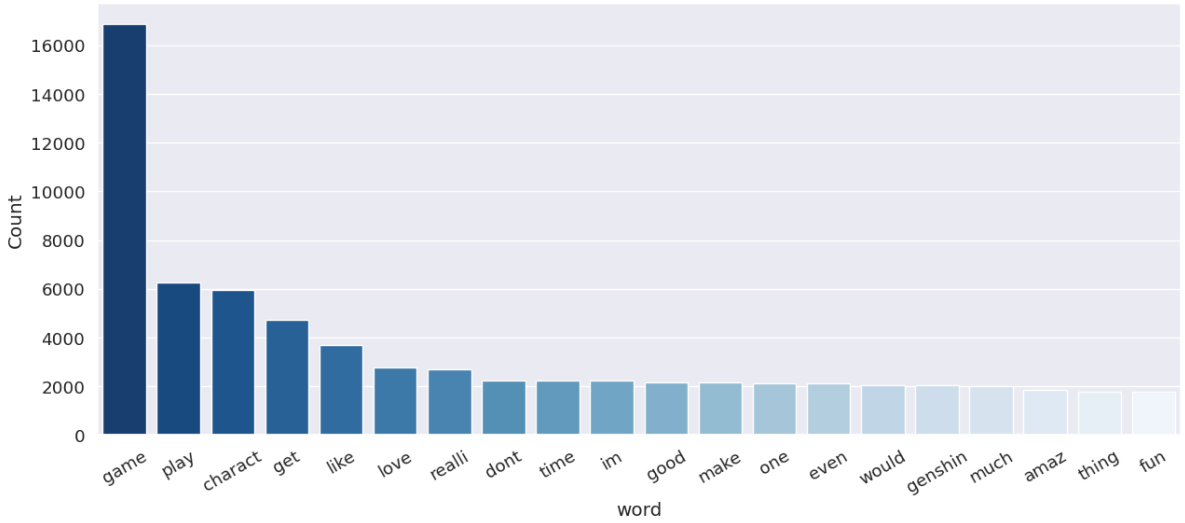

Figure 3: Top 20 Most Frequently Occurring Words after Data Pre-processing

After pre-processing the data, I created WordCloud plots to visualize the frequent words in negative and positive reviews. Reviews with a rating of < 3 are considered negative, while those with a rating of $>= 3$ are considered positive. From Figure 4 and Figure 5, I observe that there are many common high-frequency words between the reviews from Google Play and those from Apple Store, such as "game", "charact", and "good". For Google Play reviews, negative reviews contain words like "bug", "trash", "space", "terribl", etc., while positive reviews include words like "love", "fun", and "nice". For reviews on Apple Store, negative reviews commonly contain words like "reward", "money", "tone", "system", etc., while positive reviews frequently feature words like "great", "graphic", and "free".

Upon comparing these results, I observe that negative reviews from the Apple Store seem to be more neutral than those from Google Play. Moreover, some negative reviews that appeared from Android users on Google Play are related to storage space issues, such as the words "space", "full", and "storag", whereas this problem is not as frequently mentioned by iOS users.

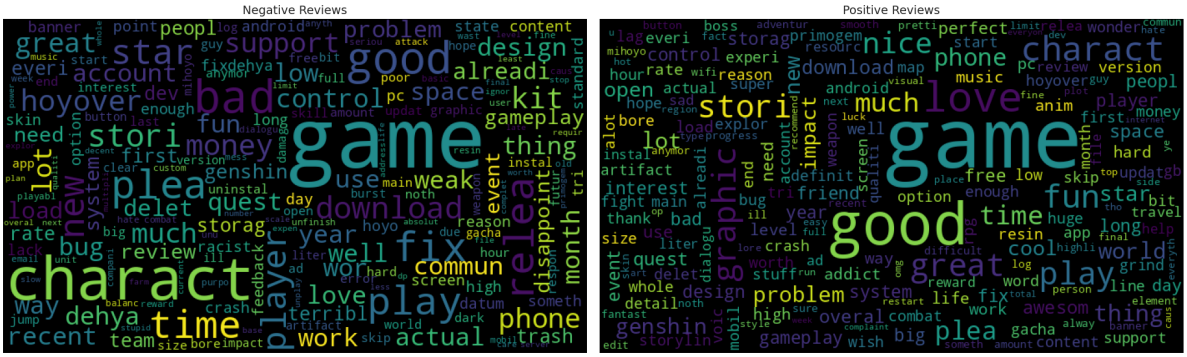

Figure 4: WordCloud for Reviews on Google Play

Figure 5: WordCloud for Reviews on Apple Store

3.3 Rating Distribution

Before proceeding with further analysis, I created plots to show the distribution of ratings in the reviews, as seen in Figure 6 and Figure 7. Through the plots, we can see that the proportion of five-star reviews is the highest, no matter whether it is in Google Play or Apple Store. Also, it seems there are more negative reviews (rating < 3) and fewer positive reviews (rating $>= 3$) on Google Play compared to the reviews on Apple Store.

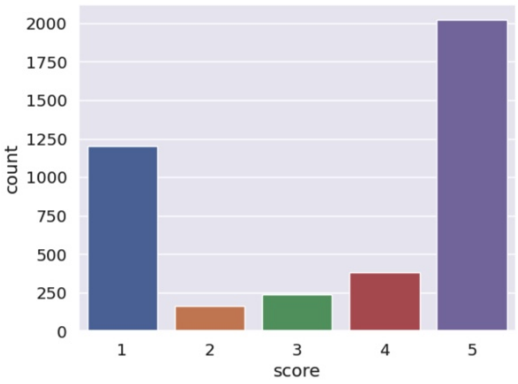

Figure 6: Google Play Rating Distribution

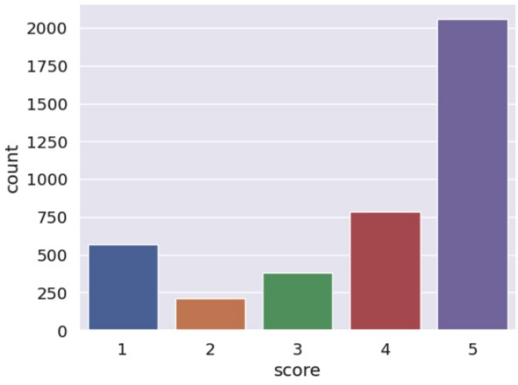

Figure 7: Apple Store Rating Distribution

4 Methods

4.1 Topic Modeling: LDA

Latent Dirichlet Allocation (LDA) is a commonly used method for topic modeling that enables the discovery of hidden topics within a collection of documents. This method has been successfully used for discovering topics of interest on Steam (Yu et al. 2021).

In LDA, each document is represented as a mixture of topics, and each topic is represented as a probability distribution over words. LDA assumes that documents are composed of a few dominant topics, and each of these topics is characterized by a unique set of words. The main idea of LDA has shown in Figure 8. LDA, aided by visualization tools like LDAvis (Sievert and Shirley 2014), provides a powerful analysis of the words within documents.

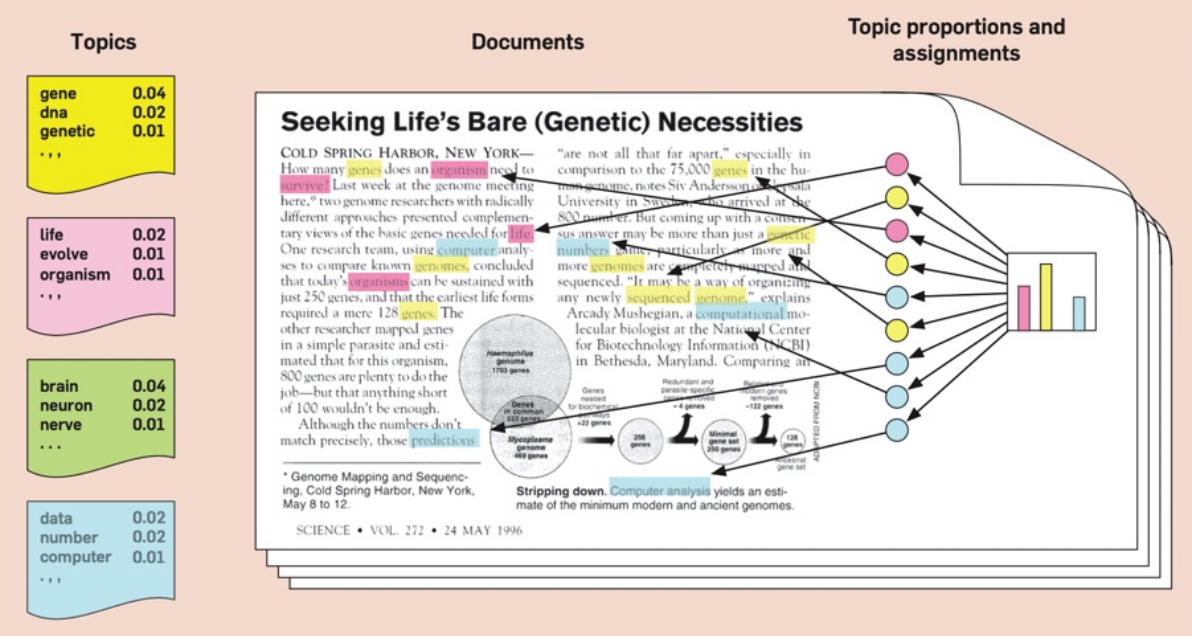

Figure 8: The intuitions behind LDA
(Blei 2012)

To implement the LDA method, I use Gensim library in Python. First, I create a dictionary to hold all unique terms and assign an index. Then convert the list of reviews into a Document Term Matrix as a corpus, which is a mapping of (word index, word frequency). Next, I build an LDA model by using the corpus. In addition to the corpus and dictionary, a K value also needs to be chosen, which is the number of topics. We use the coherence score to measure how interpretable the topics are to humans. In this project, the topics are represented as the top N words with the highest probability of belonging to that particular topic, and the coherence score measure how similar these words are to each other. To compute the coherence score, I use the *model.get_coherence*() directly from *Gensim* in Python.

4.2 Supervised Learning

In this project, I implement some machine learning models, including KNN, Multinomial Naive Bayes, Random Forest, Multinomial Logistics Regression, and SVM, to predict the ratings. Before building these models, I use $TfidfVectorizer$ in Python to convert the clean text data into a matrix of TF-IDF features and use K-Fold Cross Validation (set K = 5) during the training process to tune the parameters.

4.2.1 K-nearest neighbors (KNN)

K-nearest neighbors (KNN) is a widely used algorithm for both classification and regression tasks. KNN employs a majority voting principle based on the classes of its k-nearest neighbors. One of the main advantages of KNN is its simplicity and ease of implementation (Guo et al. 2003). It does not require any training phase, and the computational cost is relatively low. However, KNN has some limitations. It may not work well with high-dimensional data or imbalanced datasets, and it can be sensitive to the choice of distance metric and the number of neighbors. In this project, I implement KNN as our baseline model with K = 5 by using $sklearn.neighbors.KNeighborsClassifier$ in Python.

4.2.2 Multinomial Naïve Bayes

Multinomial Naïve Bayes is a classification algorithm based on Bayes' theorem, which assumes that the presence of a particular feature in a class is independent of the presence of other features. It is commonly used in Natural Language Processing (Jiang et al. 2016). Multinomial Naïve Bayes estimates the probability of each class given the document's features and assigns the class with the highest probability as the prediction. The Naïve Bayes Classifier can be written as follows. MAP here means "maximum a posteriori", i.e., the most likely class.

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c|d) \tag{1}$$

By Bayes' theorem, we have:

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c)P(c)}{d} = \underset{c \in C}{\operatorname{argmax}} P(d|c)P(c) = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, ..., x_n|c)P(c) \tag{2}$$

where d is document, c is the class label, and document d can be represented as features $x_1..x_n$

4.2.3 Random Forest

The random forest algorithm is widely used in machine learning. It combines multiple decision trees to obtain a single outcome. Each decision tree is a model that comprises a set of hierarchical "questions," and the data is divided continuously based on a specific parameter (Quinlan 1996). Random forest has gained popularity in recent years due to its ability to perform well in scenarios where there are numerous variables and few observations (Biau and Scornet 2016).

4.2.4 Multinomial Logistic Regression

Multinomial Logistic Regression uses separate logistic regression models for each category to estimate the probability of each category. The class with the highest probability is predicted. Compared to ordinal logistic regression, multinomial logistic regression is more flexible as it does not make as many strong assumptions about the data structure, such as normality, linearity, or homoscedasticity (Kwak and Clayton-Matthews 2002).

4.2.5 Support Vector Machines (SVM)

Support Vector Machines (SVM) is a classification algorithm that aims to find a hyperplane in an n-dimensional space that maximizes the separation of data points to their respective classes (Noble 2006). For multiclass classification, a hyperplane is needed to separate a class and all other classes at once. SVM is widely used since it is effective in handling high-dimensional data and could effectively capture complex relationships between features and classes by using kernel functions.

5 Results

5.1 Topic Modeling

Before building the LDA model, it is important to choose a proper K value. As I mentioned before in the Method section, the coherence score can be used to choose the optimal K value. In general, we pick the model that gave the highest coherence value before flattening out or a major drop. By looking through Figure 9, it suggests the optimal K value for the LDA model based on reviews from Google Play is 10, since after K = 10, the curve flattens out. We also notice that when K = 2, it gets the

highest topic coherence score. Therefore, I built two LDA models for the reviews on Google Play with K = 2 and K = 10. Similarly, Figure 10 suggests the optimal K value for the reviews from Apple Store is 14. Hence, I built an LDA model with K =14 using the reviews from Apple Store.

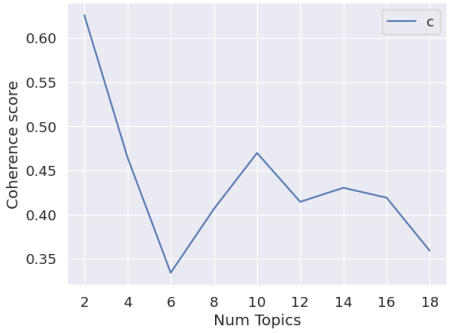

Figure 9: Topic Coherence (Google Play)

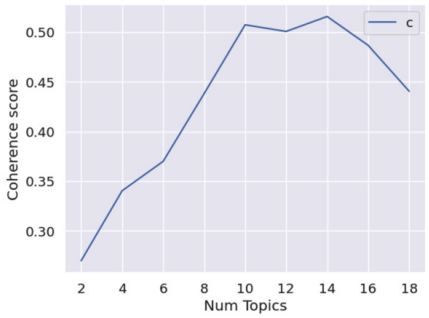

Figure 10: Topic Coherence (Apple Store)

5.1.1 Reviews on Google Play

1. LDA with K = 2: Through the result (shown in Figure 11), the data shown displays 2 different topics where each topic is a combination of keywords and each keyword has a certain weight that puts on the topic; The higher the number the more important the word is to the topic. For example, in Topic 0 word "charact" has a weight of 0.067 within the topic which in comparison to the second word "bad" with a score of 0.029 has a double amount of weight. Also, Topic 0 seems to be about a negative gaming experience, because it has the terms: bad, fix, money. In contrast, Topic 1 seems to be about a positive gaming experience, since it has the terms: good, love, fun, great.

```
[(0,
  '0.067*"charact" + 0.029*"bad" + 0.022*"new" + 0.021*"star" + 0.019*"relea" +
0.013*"fix" + 0.013*"player" + 0.010*"well" + 0.009*"money" + 0.009*"kit"'),
 (1,
  '0.164*"game" + 0.055*"good" + 0.026*"love" + 0.021*"play" + 0.017*"fun" + 0.015*"stori"
+ 0.015*"great" + 0.013*"time" + 0.013*"much" + 0.013*"plea"')]
```

Figure 11: LDA with K = 2

2. LDA with K = 10: Through the result (shown in Figure 12), Most topics talks about their positive gaming experience. Topics 1, 2, 3, 6, 8, and 9 have reviews expressing their love for the game, and topics 0, 4, and 5 focus on the game's shortcomings. Through these topics, we could summarize the advantages and disadvantages of Genshin Impact.

Based on the results of these two models, we can summarize the advantages and disadvantages of Genshin Impact.

- Advantages:
  - Beautiful graphics and design
  - Interesting and diverse characters
  - Fun gameplay and rewards mechanics
  - Exciting storyline and lore
  - Good balance between free-to-play and pay-to-win elements

  - Addictive and engaging overall experience

- Disadvantages:
  - Bugs and glitches can be frustrating
  - New content releases infrequently or underwhelmingly
  - Too reliant on luck-based gacha mechanics
  - Disappointed with the endgame content
  - Large storage space requirement
  - Spend money to progress

```
[(0,
  '0.096*"bad" + 0.072*"new" + 0.030*"charact" + 0.026*"rate" + 0.025*"dehya" +
0.025*"player" + 0.023*"content" + 0.021*"wor" + 0.020*"version" + 0.020*"bit"'),
 (1,
  '0.052*"op" + 0.041*"experi" + 0.033*"balanc" + 0.029*"monster" + 0.026*"job" +
0.026*"excit" + 0.022*"gem" + 0.018*"primo" + 0.018*"tbh" + 0.015*"charect"'),
 (2,
  '0.195*"plea" + 0.151*"nice" + 0.067*"kit" + 0.029*"hour" + 0.024*"gb" + 0.023*"least" +
0.022*"reward" + 0.016*"import" + 0.014*"art" + 0.014*"increa"'),
 (3,
  '0.404*"love" + 0.028*"guy" + 0.021*"full" + 0.017*"recommend" + 0.016*"commun" +
0.016*"sometim" + 0.015*"dialogu" + 0.014*"tho" + 0.013*"attack" + 0.012*"ye"'),
 (4,
  '0.045*"grind" + 0.045*"awesom" + 0.041*"updat" + 0.041*"bore" + 0.041*"day" +
0.035*"bug" + 0.031*"fight" + 0.029*"uninstal" + 0.026*"probabl" + 0.022*"hoyo"'),
 (5,
  '0.053*"year" + 0.035*"month" + 0.033*"main" + 0.031*"actual" + 0.030*"account" +
0.030*"end" + 0.028*"combat" + 0.026*"size" + 0.024*"disappoint" + 0.023*"screen"'),
 (6,
  '0.282*"game" + 0.050*"charact" + 0.036*"play" + 0.027*"stori" + 0.026*"great" +
0.022*"time" + 0.022*"much" + 0.020*"thing" + 0.018*"star" + 0.015*"phone"'),
 (7,
  '0.076*"genshin" + 0.069*"impact" + 0.056*"control" + 0.048*"support" + 0.040*"use" +
0.036*"everi" + 0.031*"low" + 0.026*"android" + 0.026*"reason" + 0.025*"option"'),
 (8,
  '0.142*"graphic" + 0.069*"fix" + 0.057*"high" + 0.044*"design" + 0.044*"interest" +
0.041*"load" + 0.028*"explor" + 0.026*"dark" + 0.024*"lag" + 0.024*"perfect"'),
 (9,
  '0.329*"good" + 0.099*"fun" + 0.056*"relea" + 0.027*"storag" + 0.026*"hoyover" +
0.023*"gameplay" + 0.022*"cool" + 0.020*"hard" + 0.019*"recent" + 0.016*"addict"')]
```

Figure 12: LDA with K = 10

5.1.2 Reviews on Apple Play

LDA with K = 14: The result is shown in Figure 13. In this case, we found that most topics are stating instead of expressing emotions, i.e., the attitude seems neutral. Topics 4, 9, and 10 slightly express negative attitudes to the game, while topics 6 and 8 are expressing their love for the game. The advantages and disadvantages mentioned through these topics can be summarized as the following:

- Advantages:
  - Well-designed characters
  - Good reward system
  - Frequently updated events

```
[(0,
  '0.122*"custom" + 0.107*"choo" + 0.086*"dog" + 0.065*"interact" + 0.051*"femal" + 0.050*"step" + 0.044*"male" + 0.035*"boy" + 0.033*"ca
t" + 0.024*"card"'),
 (1,
  '0.000*"lifespan" + 0.000*"comfort" + 0.000*"e" + 0.000*"tabi" + 0.000*"rampag" + 0.000*"snap" + 0.000*"audienc" + 0.000*"degenar" + 0.0
00*"innov" + 0.000*"lunch"'),
 (2,
  '0.115*"monster" + 0.080*"aspect" + 0.061*"twin" + 0.059*"water" + 0.055*"cloth" + 0.049*"hair" + 0.046*"defeat" + 0.040*"troubl" + 0.03
8*"fli" + 0.036*"glider"'),
 (3,
  '0.171*"fight" + 0.139*"boss" + 0.077*"mission" + 0.065*"sometim" + 0.052*"attack" + 0.050*"action" + 0.029*"entertain" + 0.023*"quick"
+ 0.023*"view" + 0.022*"extrem"'),
 (4,
  '0.100*"money" + 0.075*"star" + 0.066*"system" + 0.065*"free" + 0.062*"gacha" + 0.047*"charact" + 0.045*"wish" + 0.036*"weapon" + 0.033
*"rate" + 0.029*"gem"'),
 (5,
  '0.278*"quest" + 0.050*"coop" + 0.034*"side" + 0.033*"inazuma" + 0.031*"mode" + 0.031*"chest" + 0.030*"liyu" + 0.030*"job" + 0.026*"comm
iss" + 0.024*"stress"'),
 (6,
  '0.152*"game" + 0.025*"charact" + 0.024*"play" + 0.021*"good" + 0.020*"time" + 0.017*"love" + 0.016*"thing" + 0.015*"great" + 0.015*"fu
n" + 0.014*"much"'),
 (7,
  '0.161*"skin" + 0.089*"region" + 0.078*"dark" + 0.066*"black" + 0.064*"charact" + 0.053*"tone" + 0.052*"diver" + 0.045*"white" + 0.036
*"color" + 0.027*"pale"'),
 (8,
  '0.199*"reward" + 0.094*"player" + 0.075*"commun" + 0.058*"mihoyo" + 0.048*"event" + 0.043*"compani" + 0.042*"spend" + 0.039*"year" + 0.
030*"money" + 0.030*"base"'),
 (9,
  '0.135*"glitch" + 0.135*"dollar" + 0.071*"order" + 0.055*"repre" + 0.046*"consid" + 0.039*"optim" + 0.033*"menu" + 0.031*"deep" + 0.028
*"charg" + 0.022*"pure"'),
 (10,
  '0.120*"download" + 0.118*"phone" + 0.068*"crash" + 0.054*"space" + 0.051*"app" + 0.049*"problem" + 0.041*"fix" + 0.038*"load" + 0.037
*"screen" + 0.034*"pc"'),
 (11,
  '0.198*"resin" + 0.081*"artifact" + 0.066*"iphon" + 0.061*"level" + 0.052*"domain" + 0.050*"farm" + 0.040*"rank" + 0.032*"cap" + 0.031
*"hous" + 0.029*"boss"'),
 (12,
  '0.000*"lifespan" + 0.000*"comfort" + 0.000*"e" + 0.000*"tabi" + 0.000*"rampag" + 0.000*"snap" + 0.000*"audienc" + 0.000*"degenar" + 0.0
00*"innov" + 0.000*"lunch"'),
 (13,
  '0.221*"power" + 0.031*"tutori" + 0.022*"abyss" + 0.000*"hydro" + 0.000*"mid" + 0.000*"lage" + 0.000*"task" + 0.000*"altern" + 0.000*"du
pe" + 0.000*"laugh"')]
```

Figure 13: LDA on Apple Store reviews

- Disadvantages:
  - Bad gacha system
  - Waste much money
  - Glitches and crashes in the game
  - Large storage space requirement

To better understand the model, I visualized the model by using *pyLDAvis* in Python. This package provides two methods of displaying results: the Intertopic Distance Map and the bar chart showcasing the Top 30 Most Salient Terms. The Intertopic Distance Map is a visualization technique that arranges the data in a two-dimensional space, highlighting the most useful words and identifying the topics they belong to. The topics with high prevalence typically do not overlap, with a few exceptions. Prevalence here refers to the frequency with which words within a topic are used in the model. For example, in Figure 14, we observe that topics 1 and 4 overlap, indicating that they share some common words. By clicking on the bubble, as shown in Figure 14, we can view the top 30 relevant terms within Topic 9.

5.2 Rating Prediction

5.2.1 Test Dataset

I randomly split our data into 80% training and 20% testing sets. There are 8000*20% = 1600 reviews in the test set. Out of these, there are 356 reviews with a true rating of 1, 78 reviews with a rating of 2, 123 reviews with a rating of 3, 231 reviews with a rating of 4, and 812 reviews with a rating of 5.

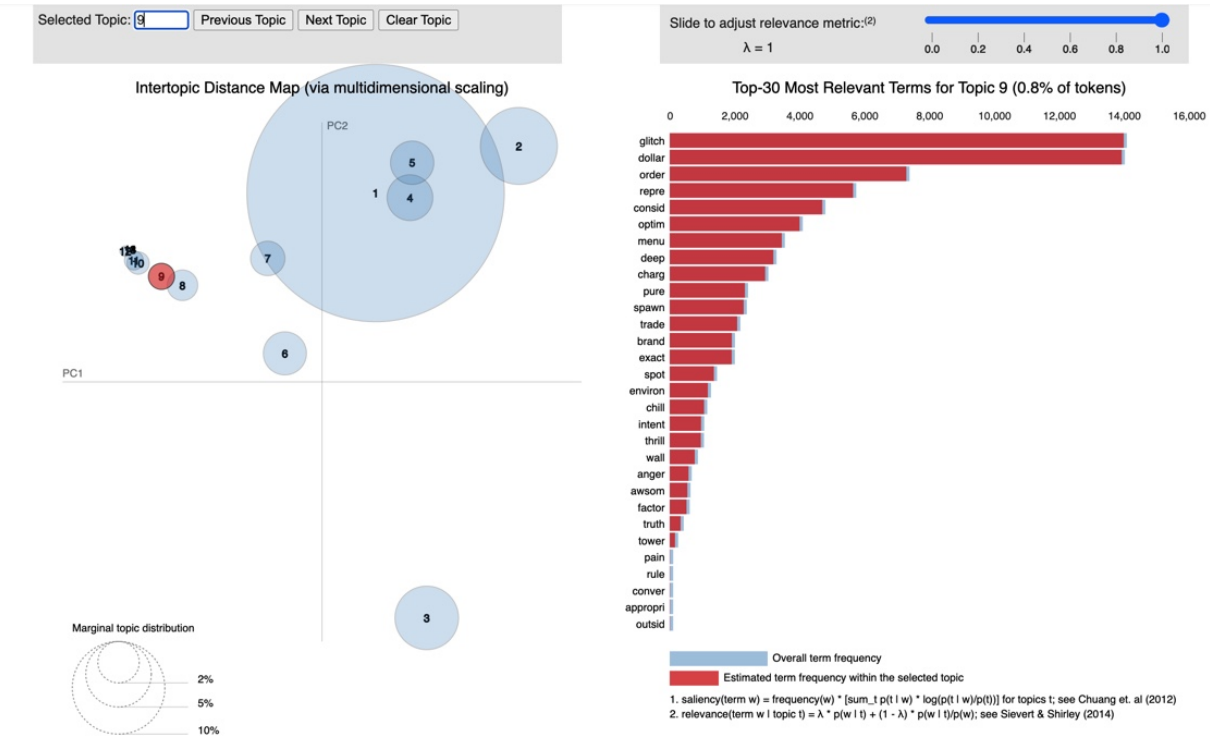

Figure 14: Example of LDAvis

5.2.2 Evaluation Metrics

In this project, I choose to use accuracy, receiver operating characteristic curve (ROC) and Area under the ROC Curve (AUC) to evaluate the performance of the models. Accuracy is a commonly used method to evaluate performance, and it is defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

However, accuracy is not a perfect measure for imbalanced data because it can not distinguish between the numbers of correctly classified instances for different classes. To deal with this issue, the ROC curve and AUC are introduced. ROC curve is a graph showing the performance of a classification model at all classification thresholds (Fawcett 2006). This curve plots the true positive rate (TPR) against the false positive rate (FPR), which are defined as follows:

$$TPR = \frac{TP}{TP + FN} \tag{4}$$

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

5.2.3 Model Performance

The results for the machine learning models are shown in Table 1. Through the table, we can see that SVM gets the best performance, with accuracy = 0.660625, while KNN gets the worst performance with accuracy = 0.551094.

	Accuracy
KNN	0.551094
MultinomialNB	0.603750
RandomForest	0.643437
Logistic	0.659531
SVM	0.660625

Table 1: Result Comparsion

Logistic Regression, Random Forest, and SVM perform relatively well may due to their ability to do feature selection and capture non-linear relationships. Logistic regression uses regularization, Random Forest uses feature importance scores, and SVM uses the kernel trick to identify relevant features, which improves algorithm performance by reducing irrelevant features. Also, Random Forest can capture non-linear relationships by combining multiple decision trees, and SVM achieves this by transforming data into a higher-dimensional space. Additionally, all these three methods are robust to noise and outliers, and regularization prevents overfitting.

However, KNN and MultinomialNB do not perform well due to the curse of dimensionality, imbalanced data, and their inability to capture non-linear relationships. KNN's performance may decrease with high-dimensional text data due to the distance between nearest neighbors becoming less significant as the number of features expands. Imbalanced data can skew KNN's local density and assign too much importance to the dominant class. MultinomialNB assumes conditional independence between features and the class, which may not hold in imbalanced data. Both KNN and MultinomialNB struggle to capture complex, non-linear relationships and lack regularization, leading to overfitting and poor generalization performance.

We then obtained the confusion matrix, computed AUC values for each class, and plotted ROC curves. The confusion matrix and ROC curves form the two best-performing models (Logistic and SVM) are shown in Figure 15, Figure 16, Figure 17, and Figure 18.

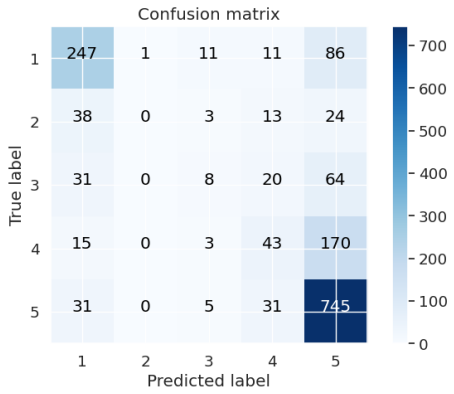

Figure 15: Confusion Matrix for Logistic

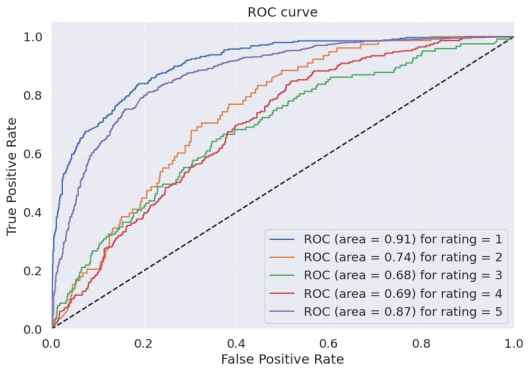

Figure 16: ROC for Logistic Regression

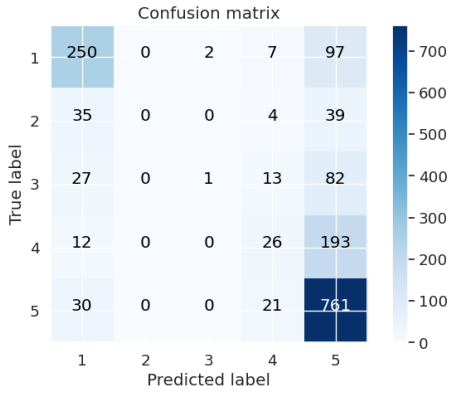

Figure 17: Confusion Matrix for SVM

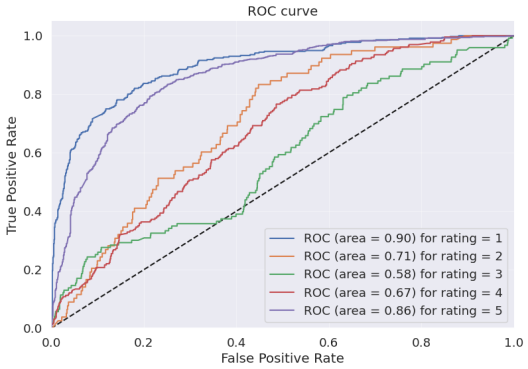

Figure 18: ROC for SVM

Through these plots, it is clearly observable that the classifiers predict 1-star and 5-star reviews with much higher accuracy than predicting the reviews with 2/3/4 stars. The AUC values for predicting 1-star reviews and 5-star reviews achieve around 0.9 (for both models), while the AUC values for 2/3/4 star reviews are around 0.7. This may be because the number of reviews with 1-star and 5-stars is much larger than from the other class, the model may be biased towards the majority class and may have difficulty accurately classifying the minority classes.

6 Conclusion

After analyzing the Genshin Impact reviews, we were able to answer the questions posed in the introduction. We found that the words "love", "fun", "nice", "great", etc. were more likely to cause players to give a higher rating, while words like "bug", "trash", "space", "terrible", and "money", were more likely to cause players to give a lower rating.

We discovered that the main topics discussed in the reviews were related to gameplay, characters, and storyline. And there was no significant difference between the reviews under Google Play and the reviews under Apple Store in terms of topic distribution, but the attitudes toward reviews under Apple Store seem more neutral. By using LDA, we found the possible reasons that Genshin Impact was widely liked including the stunning graphics, good rewards mechanics, frequent updates, an engaging storyline, well-designed characters, etc.

The machine learning models including KNN, Multinomial Naive Bayes, etc. are built to predict the ratings based on the reviews. The results showed that SVM, Logistic Regression, and Random Forest perform well. This might be because of their ability to perform feature selection and capture non-linear relationships between the features and the rating.

In conclusion, our analysis sheds light on some factors that Genshin Impact is successful, and the models we built can be useful in predicting the ratings based on reviews. Further work is needed to explore more advanced natural language processing techniques and deep learning models for even more accurate predictions.

References

Biau, Gérard, and Erwan Scornet. 2016. "A random forest guided tour." *Test* 25:197–227.

Blei, David M. 2012. "Probabilistic topic models." *Communications of the ACM* 55 (4): 77–84.

Fawcett, Tom. 2006. "An introduction to ROC analysis." *Pattern recognition letters* 27 (8): 861–874.

Guo, Gongde, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. 2003. "KNN model-based approach in classification." In *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings,* 986–996. Springer.

Guo, Yue, Stuart J Barnes, and Qiong Jia. 2017. "Mining meaning from online ratings and reviews: Tourist satisfaction analysis using latent dirichlet allocation." *Tourism management* 59:467–483.

Jiang, Liangxiao, Shasha Wang, Chaoqun Li, and Lungan Zhang. 2016. "Structure extended multinomial naive Bayes." *Information Sciences* 329:346–356.

Kwak, Chanyeong, and Alan Clayton-Matthews. 2002. "Multinomial logistic regression." *Nursing research* 51 (6): 404–410.

Noble, William S. 2006. "What is a support vector machine?" *Nature biotechnology* 24 (12): 1565–1567.

Pranckevičius, Tomas, and Virginijus Marcinkevičius. 2017. "Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification." *Baltic Journal of Modern Computing* 5 (2): 221.

Quinlan, J. Ross. 1996. "Learning decision tree classifiers." *ACM Computing Surveys (CSUR)* 28 (1): 71–72.

Sievert, Carson, and Kenneth Shirley. 2014. "LDAvis: A method for visualizing and interpreting topics." In *Proceedings of the workshop on interactive language learning, visualization, and interfaces,* 63–70.

Wikipedia. 2023. *Genshin Impact — Wikipedia, The Free Encyclopedia.* http://en.wikipedia.org/w/index.php?title=Genshin%20Impact&oldid=1150945166. [Online; accessed 21-April-2023].

Yu, Yang, Ba-Hung Nguyen, Fangyu Yu, and Van-Nam Huynh. 2021. "Discovering topics of interest on Steam community using an LDA approach." In *Advances in the Human Side of Service Engineering: Proceedings of the AHFE 2021 Virtual Conference on The Human Side of Service Engineering, July 25-29, 2021, USA,* 510–517. Springer.