Study on game data: visualization and modeling of game data ("All 55,000 Games on Steam Dataset")

Group Project of COMP5152

Advanced Data Analytics

Group number: 2

LUI Man Hon

YUEN Chi Ho

CHAN Cheuk Hei

PUN Sum Ho

YEUNG Wai Kin

MAK Hin Ming

The Hong Kong Polytechnic University

April 3, 2023

1. Abstract

Video gaming provides people a space to freely explore their personalities in a safe environment without anyone knowing them in the physical world. Especially in the past few years, it is considered that having social gathering events are not safe due to the COVID-19 pandemic. And because of the social distancing restriction, people are forced to stay and spend more time at home, such as watching Netflix, playing video games, and even working from home. As a result, some video game market leaders such as Nintendo and Tencent all recorded substantial revenue growth in the first half of 2020. Gaming companies like Nintendo reported that its Q1 2020 operating profit grew an incredible 428%[1].

Even though gaming companies are facing a slowdown in demand for video games from pandemic highs in the short term[2], it is expected that the economic contribution of the video game market will increase continuously in the long run[3]. As the value of the video game market is significant, creating an attractive game is the key to scrambling for a piece of the action in this market.

In the gaming industry, predicting potential user ratings for a new game on the video platform would be beneficial for game developers and publishers. It could provide some insights to the companies on the marketing and pricing strategies, as well as to determine the date of releasing the video game, based on the previous knowledge on what features and characteristics of the game are most appealing to users. Understanding the market trend on the attractive aspects of the game, the companies could set these as the selling points in the marketing campaigns. Additionally, the predicted user ratings could help the developers to determine the selling price to optimize their profit. If they expected the ratings would be high, they may consider setting a higher price for the game. Conversely, if the predicted ratings are lower, they may consider setting a lower price to increase the likelihood of users trying the game. Moreover, the expected popularity of the game may affect the decision on the release schedule of the game. If the predicted ratings are high, the publishers may want to release the game during a high-traffic time on the gaming platform to maximize exposure, and sales, and compete with other games. On the other hand, the publisher may consider releasing the game in a non-peak season to minimize the risk of negative reviews if they are not very confident with their game.

[1] COVID-19 Is Taking Gaming And ESports To The Next Level. Retrieved from: https://sourcing.hktdc.com/newsbites/covid-19-is-taking-gaming-and-esports-to-the-next-level/

[2] Gaming industry's fortunes fade as spending squeeze follows pandemic bump. Retrieved from: https://www.reuters.com/technology/gaming-industrys-fortunes-fade-spending-squeeze-follows-pandemic-bump-2022-08-10/

[3] Gaming is booming and is expected to keep growing. This chart tells you all you need to know. Retrieved from: https://www.weforum.org/agenda/2022/07/gaming-pandemic-lockdowns-pwc-growth/

2. Introduction

2.1 Motivation

Some empirical studies were conducted previously to study the important features of a successful video game. A paper published in 2010 concluded that some key elements of being a favorite video game. Some key features such as having good graphics and sound; complex story and different story outcomes; fast loading times; rare game items; function of customization; cooperation and social interaction, were mentioned in the paper[4]. The results mentioned in the previous study were based on the ratings from 421 video game players, which could be viewed as a qualitative method. It is commonly used especially when the analysis is difficult to quantify.

2.2 Objective

The result of the previous study was based on the traditional survey approach. In this study, different statistical methods are used to illustrate the features of the popular video game, as well as predict the user rating. There are two major objectives in this paper, including:

1. Explore the features of popular video games
2. Use the statistical models to predict potential user ratings

Definition of the popularity or user ratings: to measure the differences between the number of positive reviews and that of negative reviews for each video game.

2.3 Data introduction

The dataset ("All 55,000 Games on Steam (November 2022)") is retrieved from the Website in Kaggle. It consists of 22 data columns, which describe the features of the video games in the store page data on the online gaming platform in Steam, and over 55, 000 records.

3. Methodology

For exploring the features of popular video games, different descriptive statistics and graphs were used to visualize the important features of the popular video game. Apart from the features mentioned in the previous study, many features are included in the dataset. As such, statistical tools were used to identify the major components of a good video game. Statistical tools such as graphical representation, hypothesis testing, as well as some descriptive statistics were used to test the relationship between the popularity of the video game and different features.

For predicting potential user ratings, some statistical modeling methods were used to predict the popularity of video games. Statistical models, such as K-Nearest Neighbor and decision tree, were

[4] King, D.L., Delfabbro, P.H. & Griffiths, M.D. The Role of Structural Characteristics in Problematic Video Game Play: An Empirical Study. Int J Ment Health Addiction 9, 320–333 (2011). Retrieved from: https://link.springer.com/article/10.1007/s11469-010-9289-y

used to compare the accuracy of different modeling approaches. The comparison was then performed to evaluate the performance.

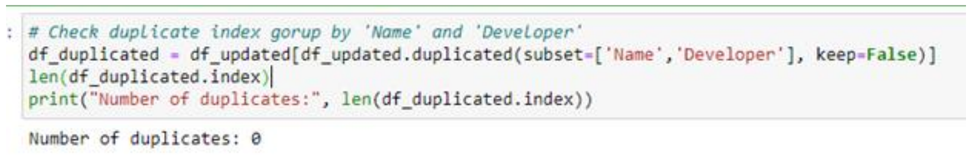
4. Data Preprocessing

4.1 Data Cleaning

Considering the raw data may contain missing values, inaccurate and irrelevant data, data cleaning was conducted. In the data preprocessing part, data cleaning was conducted to facilite the process of data analysis, visualization, and model training for machine learning. Data cleaning included checking the duplicate index, missing data handling, dropped irrelevant columns, and noisy data handling.

4.2 Duplicate Index

Duplicate records were identified and removed from the dataset to avoid redundancy. Python provides libraries and functions duplicated() that could be used to check duplicates from datasets. “Name” and “Developer” were used to identify which indexes were duplicated. The result showed that the number of duplicated Index is zero.



```
# Check duplicate index gorup by 'Name' and 'Developer'
df_duplicated = df_updated[df_updated.duplicated(subset=['Name','Developer'], keep=False)]
len(df_duplicated.index)
print("Number of duplicates:", len(df_duplicated.index))

Number of duplicates: 0
```

Figure 1 - Check duplicate Index

4.3 Identify the Missing Data

The Missing Data could degrade the accuracy of the analyses and machine-learning model results. Missing Data was identified so that these records can be imputed (meaning fill in reasonable values for the missing data). There is a way to simply check the summary of missing data in the dataset by using the isna() function. Moreover, The Dataprep library plotted the missing data spectrum for each column. The total number of missing Cells is 26910, the most missing data is the Website column, and the percentage of missing is 45.3%.

Columns	**Missing data**
Website	25217
Categories	970
Genre	161
Publisher	151
Tags	135
Developer	129
Release Date	99
Short Description	37
Languages	11

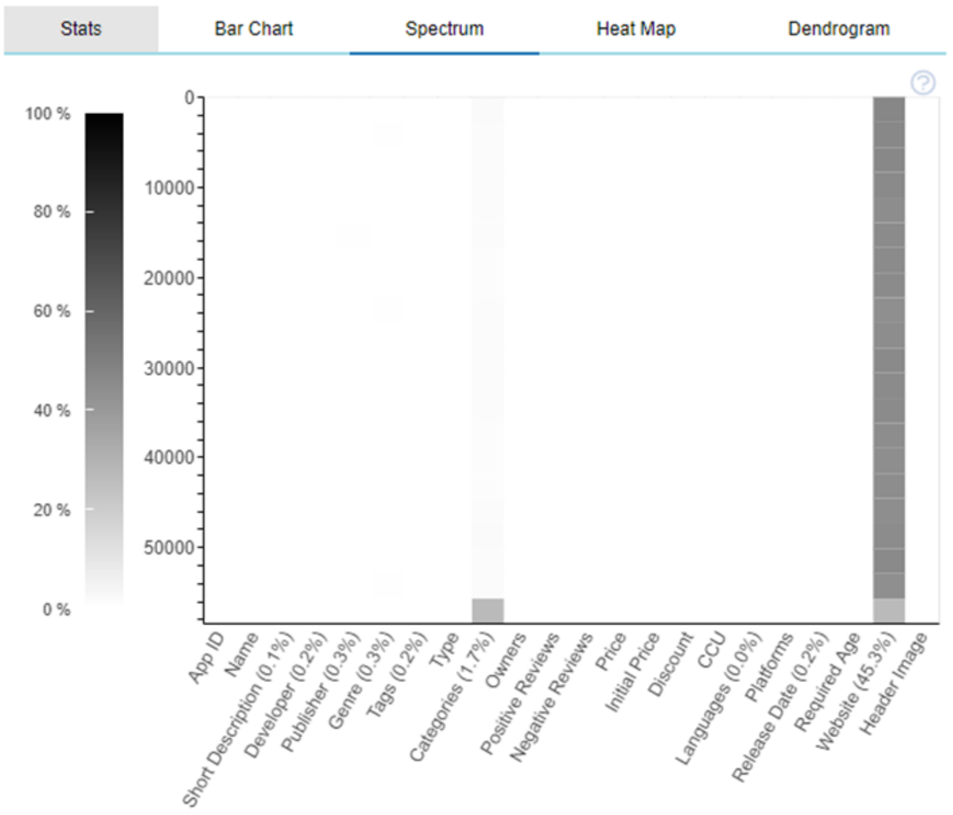


Figure 2 - Missing data spectrum

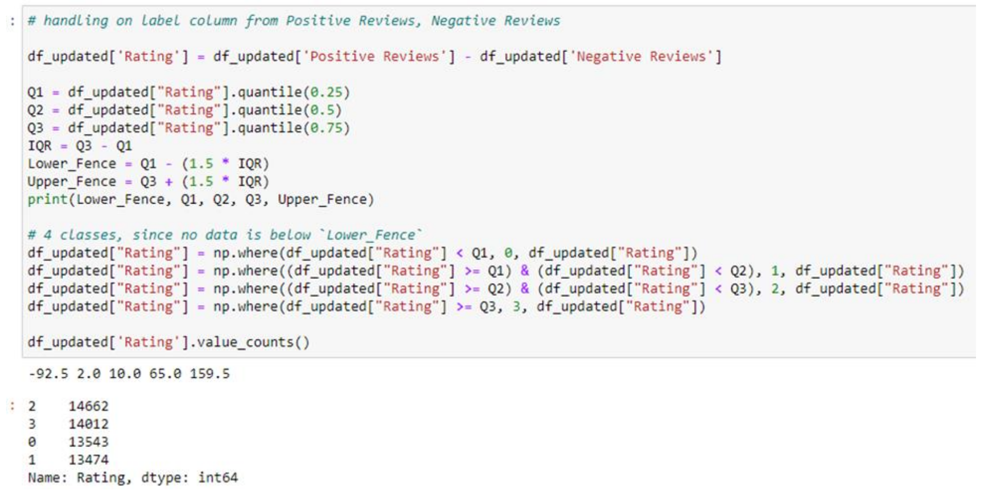
4.4 Missing data handling

The proportions of the missing data for most features are 0.4%, such as the Short Description, Developer, Publisher, Genre, and Tags. In addition, the above data is unique for each game. Considering the incorrect data will impact further analysis and model training, standard values "N/A" were used to replace the missing values.

For the data column website, the value of the website column is only the URL. It is considered not meaningful and representative values and accurately would not be affected in the analysis. To handle the data, the data type was changed to Boolean flag, convert to "1" if there is no missing value, or else as "0". This was performed the same as in Required Age. In addition, the extra feature Release Month was added to replace the Release Date as an integer, and 99 rows of missing data are dropped without any month provided.

4.5 Data Transformation

Standardized and normalized are commonly used techniques in data analysis to make the data more consistent and easier to analyze and improve the accuracy and performance of machine learning models. The one additional feature that "Rating" was used to standardize Positive Reviews and Negative Reviews; Rating was classified into four classes using Boxplot.



```
# handling on label column from Positive Reviews, Negative Reviews

df_updated['Rating'] = df_updated['Positive Reviews'] - df_updated['Negative Reviews']

Q1 = df_updated["Rating"].quantile(0.25)
Q2 = df_updated["Rating"].quantile(0.5)
Q3 = df_updated["Rating"].quantile(0.75)
IQR = Q3 - Q1
Lower_Fence = Q1 - (1.5 * IQR)
Upper_Fence = Q3 + (1.5 * IQR)
print(Lower_Fence, Q1, Q2, Q3, Upper_Fence)

# 4 classes, since no data is below `Lower_Fence`
df_updated["Rating"] = np.where(df_updated["Rating"] < Q1, 0, df_updated["Rating"])
df_updated["Rating"] = np.where((df_updated["Rating"] >= Q1) & (df_updated["Rating"] < Q2), 1, df_updated["Rating"])
df_updated["Rating"] = np.where((df_updated["Rating"] >= Q2) & (df_updated["Rating"] < Q3), 2, df_updated["Rating"])
df_updated["Rating"] = np.where(df_updated["Rating"] >= Q3, 3, df_updated["Rating"])

df_updated['Rating'].value_counts()
```

```
-92.5 2.0 10.0 65.0 159.5
```

```
2    14662
3    14012
0    13543
1    13474
Name: Rating, dtype: int64
```

Figure 3 - Rating feature

For the Categories, Genre, Languages, and Platforms features presented as array data, normalization is a way to deal with those data with different units or scales, changed to multiple columns showing whether it belongs to those features and falls within a specified range between 0 and 1. Moreover, the Discount feature also specified a range between 0 and 1 because only less than 5% of games have a discount. For Short Description, the length of text was treated as column value.

4.6 Dropped irrelevant

This project aimed to find out the optimal business decision to create a popular steam game. Some features are irrelevant to the Rating, including App ID, Name, Developer, Publisher, and Header Image. The above features were dropped before analysis and model training. The outlier row of type is not game, which was considered also irrelevant to the topics and was dropped.

5. Data Analysis and visualization

Finding an optimal business decision to create a popular steam game is the objective in the study. First, all the games were totaled grouped from different years, and it showed that PC games started to gain popularity in 2014, with a maximum of 2000 total. 2014 was a vast changing year for users because technology such as network and PC components proliferates. Moreover, professional gaming (or esports) was growing rapidly, with 32 million people watching the League of Legends game finals online. Then, it continued to increase to more than 8000 games in 2021, although there slightly decreased in 2018 and 2022.

5.1 General analysis

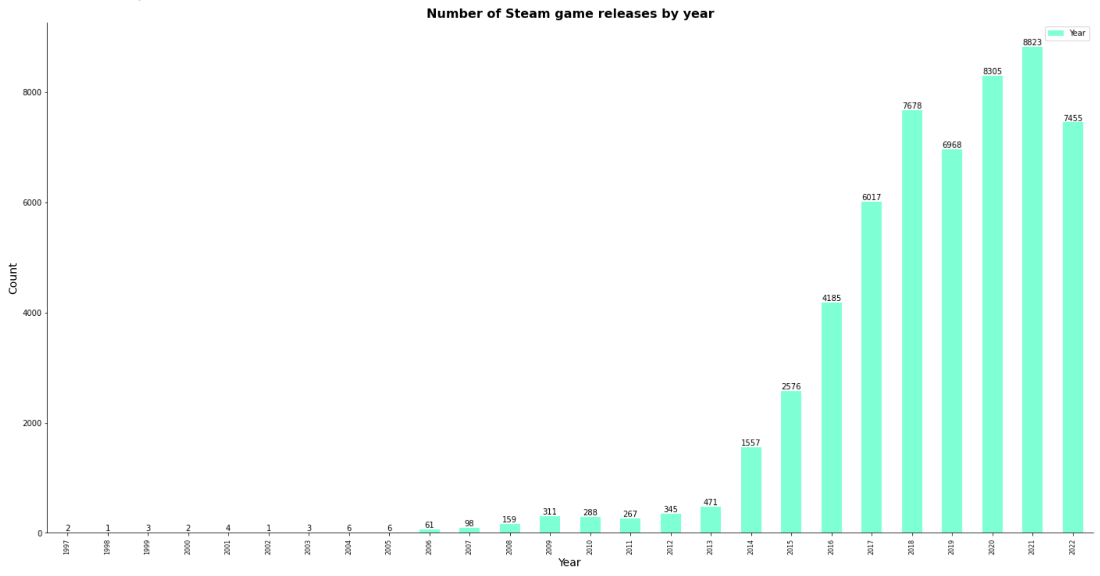


Figure 4 - Number of Steam game releases by year

To categorize the ratings, a new column called Rating was added (which was the subtraction of the values of the positive and negative reviews) from each row's value. The four groups that the quartiles generated were then mapped out using the boxplot approach. The lowest value up to Q1 is in the first group of values; Q1 to the median is in the second group; median to Q3 is in the third group; and for those higher than Q3 is in the fourth group. Next, the rating into four classes was divided.

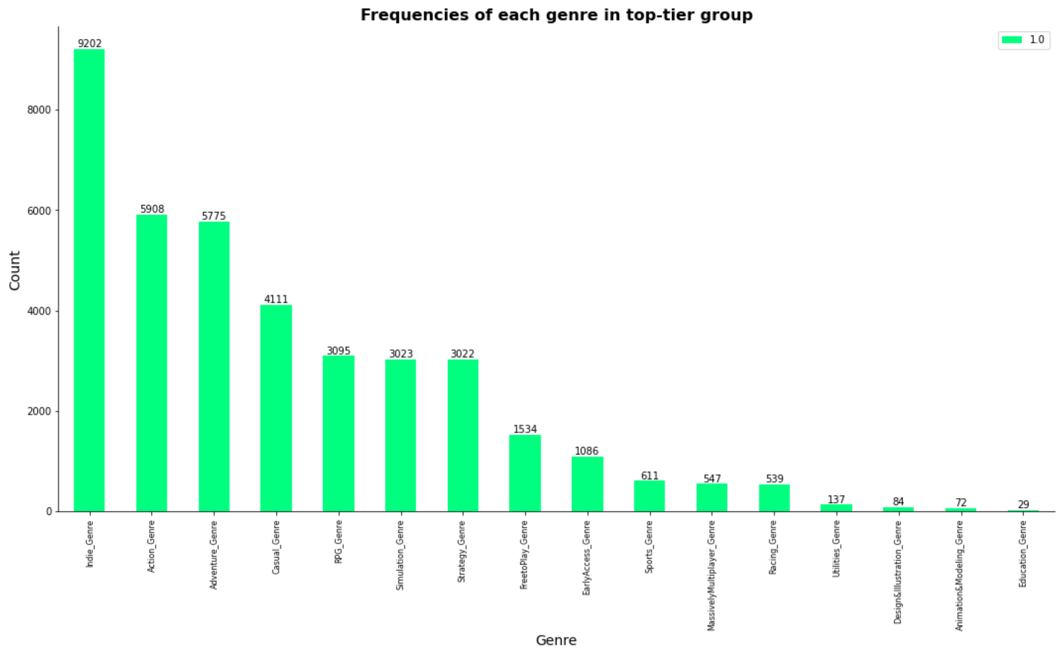


Figure 5 - Genre frequencies in upper-tier collection

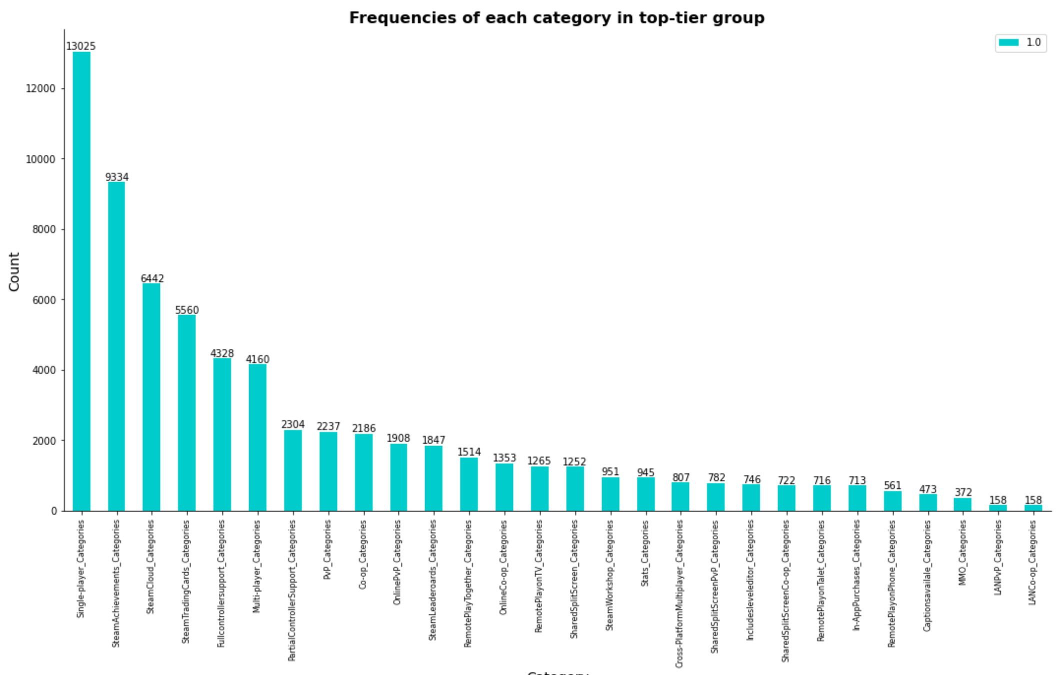


Figure 6 - Category frequencies in upper-tier collection

In addition, Upper Collections has 13963 records and assisted in deciding which genres, categories, platforms of well-liked games to include when producing the best steam games. The top 5 game genres are Indie 9202, Action 5908, Adventure 5775, Casual 4111, and RPG 4111. The top 5 game categories are Single Player Game 363, Steam Achievement 318, Steam Cloud 228, Full Game Controller Support 205, and 205 for Steam Trading Cards. Besides, mostly all the games in the Upper-tier collection are supported on the windows platform.

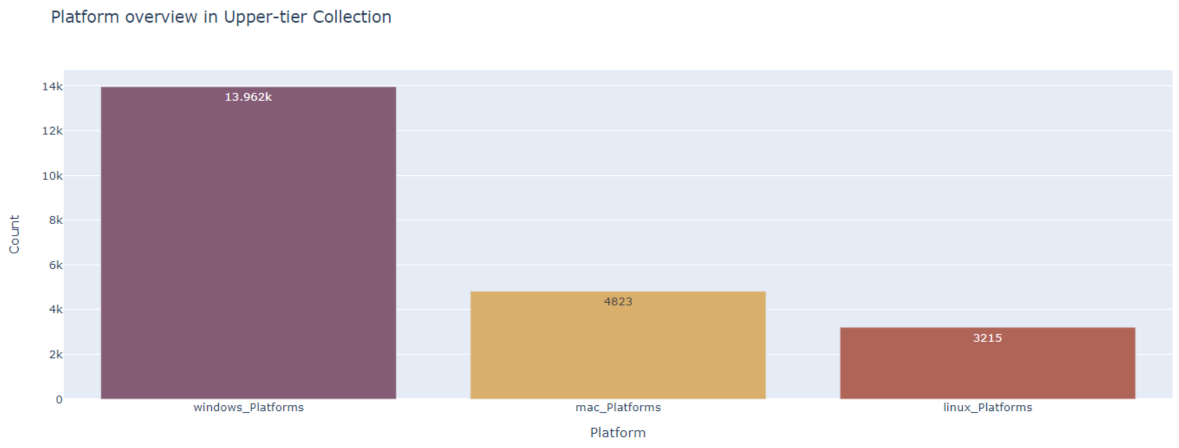


Figure 7 - Platform frequencies in upper-tier collection

5.2 Heatmap

A correlation heatmap showed the relationship between various variables in a dataset graphically. A correlation heatmap was used to display the intensity and direction of correlations between different features in the dataset.

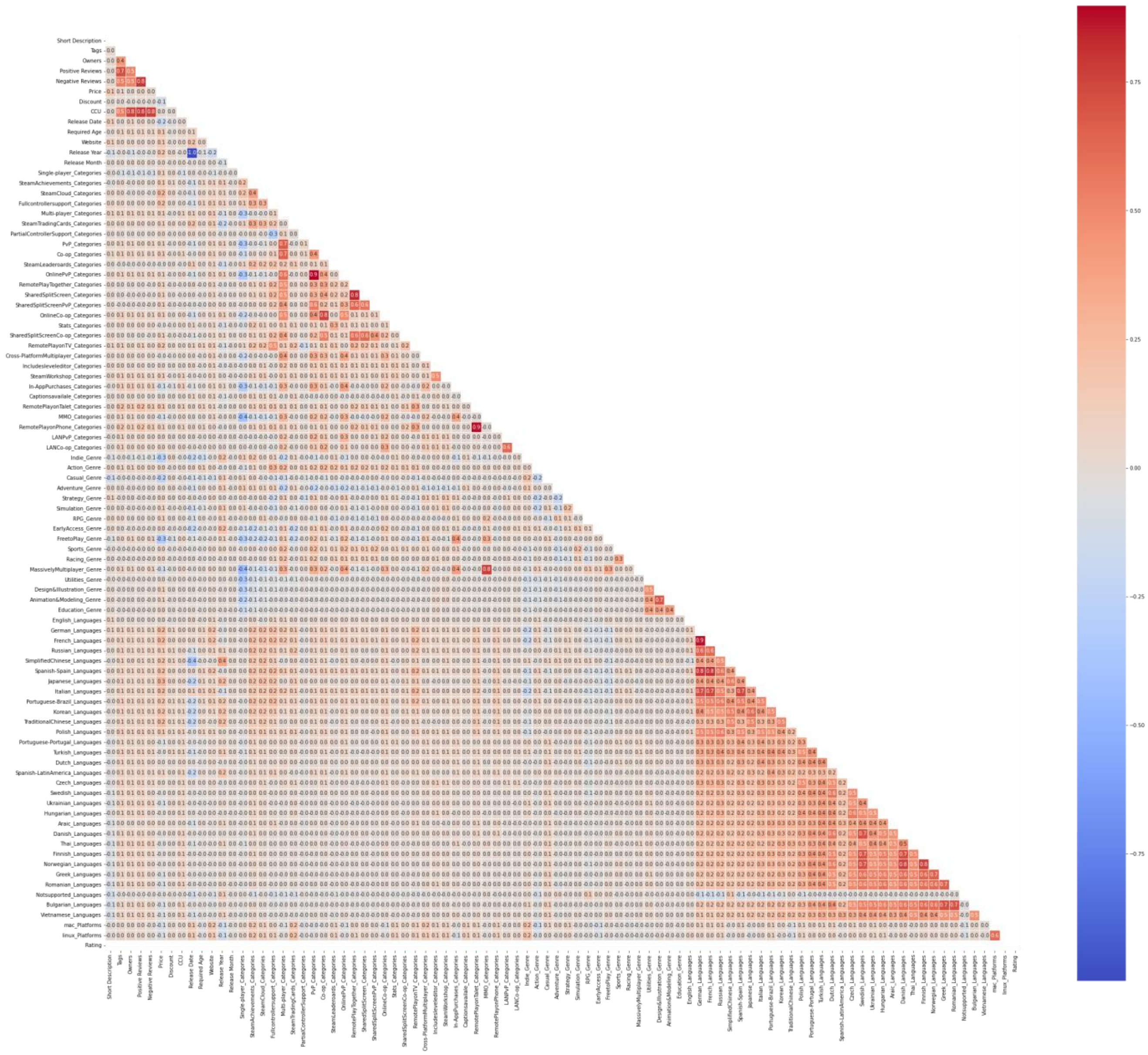
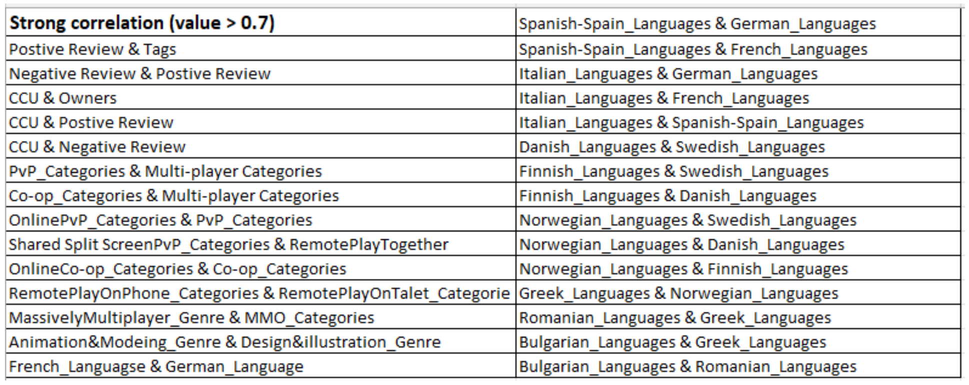


Figure 8 - Correlation Heatmap



Strong correlation (value > 0.7)	Spanish-Spain_Languages & German_Languages
Postive Review & Tags	Spanish-Spain_Languages & French_Languages
Negative Review & Postive Review	Italian_Languages & German_Languages
CCU & Owners	Italian_Languages & French_Languages
CCU & Postive Review	Italian_Languages & Spanish-Spain_Languages
CCU & Negative Review	Danish_Languages & Swedish_Languages
PvP_Categories & Multi-player Categories	Finnish_Languages & Swedish_Languages
Co-op_Categories & Multi-player Categories	Finnish_Languages & Danish_Languages
OnlinePvP_Categories & PvP_Categories	Norwegian_Languages & Swedish_Languages
Shared Split ScreenPvP_Categories & RemotePlayTogether	Norwegian_Languages & Danish_Languages
OnlineCo-op_Categories & Co-op_Categories	Norwegian_Languages & Finnish_Languages
RemotePlayOnPhone_Categories & RemotePlayOnTalet_Categorie	Greek_Languages & Norwegian_Languages
MassivelyMultiplayer_Genre & MMO_Categories	Romanian_Languages & Greek_Languages
Animation&Modeing_Genre & Design&illustration_Genre	Bulgarian_Languages & Greek_Languages
French_Languagse & German_Language	Bulgarian_Languages & Romanian_Languages

Figure 9 - Strong correlation features

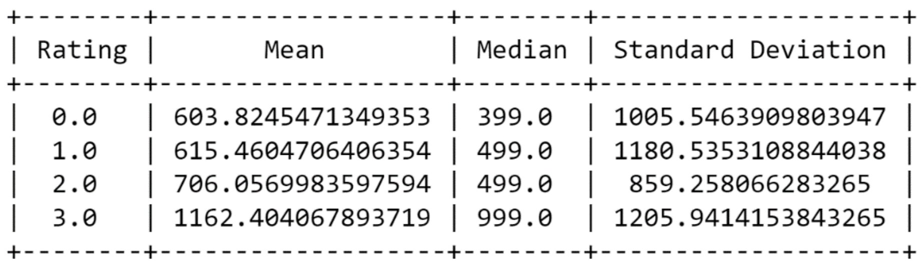
The table showed all features had a strong positive correlation with a value of greater than 0.7 which indicates the presence of multi-collinearity.

Besides, there are significant negative associations between variables like Release Year & Release Date.

5.3 Descriptive statistics

As the result of "rating" in the heatmap was reviewed, it was positively related to "Price" among all features. Therefore, following descriptive statistics on "Price" versus "Rating", was performed.

Basic statistics like mean, median, and standard deviation for price was calculated to understand the overall distribution of the data.



```
+--------+-------------------+--------+--------------------+
| Rating |        Mean       | Median | Standard Deviation |
+--------+-------------------+--------+--------------------+
|  0.0   | 603.8245471349353 | 399.0  | 1005.5463909803947 |
|  1.0   | 615.4604706406354 | 499.0  | 1180.5353108844038 |
|  2.0   | 706.0569983597594 | 499.0  |  859.258066283265  |
|  3.0   | 1162.404067893719 | 999.0  | 1205.9414153843265 |
+--------+-------------------+--------+--------------------+
```

Figure 10 – Distribution of Price

The result above echoed with the result of the heatmap. With higher Rating, the mean and median of the Price are increasing, showing that Price and Rating are positively related. Although a causal relationship could not be concluded, it gives an insight of setting pricing strategy for top-tier gaming.

Apart from the mean and the median, the standard deviation of rating 3 games, the top-tier gaming group, is the highest among all rating groups. With a high standard deviation, the prices of the games in this group, rating 3, are more spread out or diverse than in the other groups.

5.4 Distribution Frequency

To further study the distribution after calculating the standard deviation, distribution Frequency of Price in different rating gaming groups are shown below:

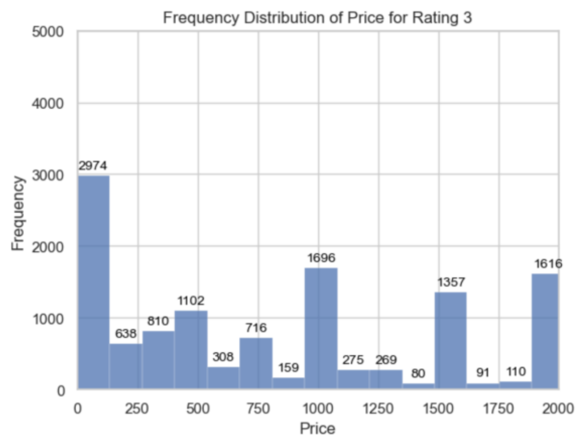


Figure 11 – Frequency Distribution of Price for Rating 3

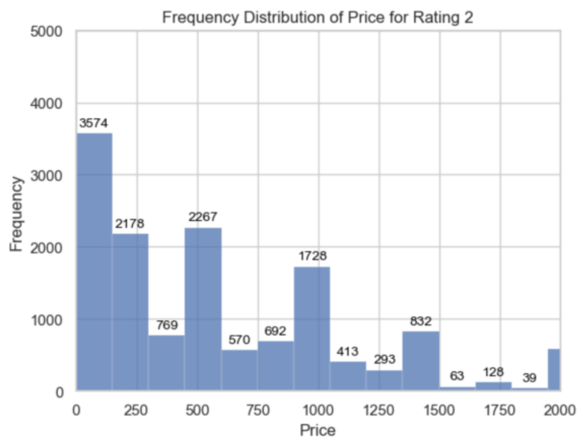


Figure 12 - Frequency Distribution of Price for Rating 2

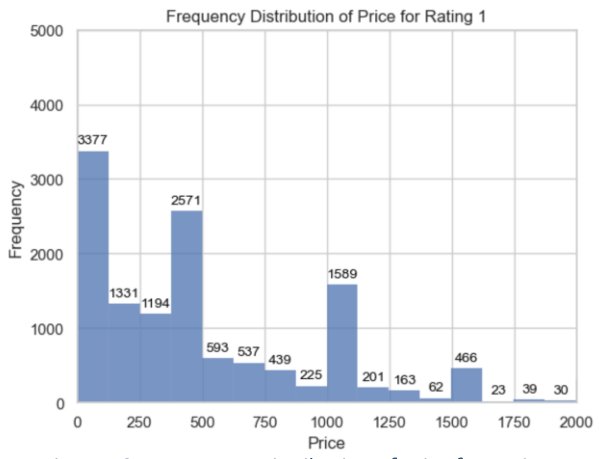


Figure 13 - Frequency Distribution of Price for Rating 1

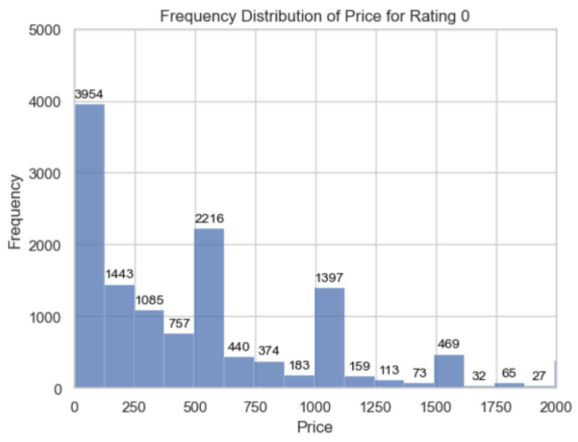


Figure 14 - Frequency Distribution of Price for Rating 0

One possible explanation for this could be that the higher-rated gaming products have more features, better graphics, or are otherwise more advanced than the products in the other groups. This could lead to greater variability in pricing, as different manufacturers or sellers may charge different prices based on factors such as production costs, marketing strategies, and competition.

Alternatively, the higher-rated gaming products are possibly more popular or in higher demand than the products in the other groups. This increased demand could lead to higher prices. However, it could result in greater variability in pricing as different sellers try to capture a share of the market by offering different prices or promotions.

5.5 Time-Series Analysis

A trend analysis of average price of games released each month is plotted below to observe changes in pricing strategies. It could be investigated to see if there's a general increase or decrease in prices, or if there are any noticeable fluctuations or patterns.

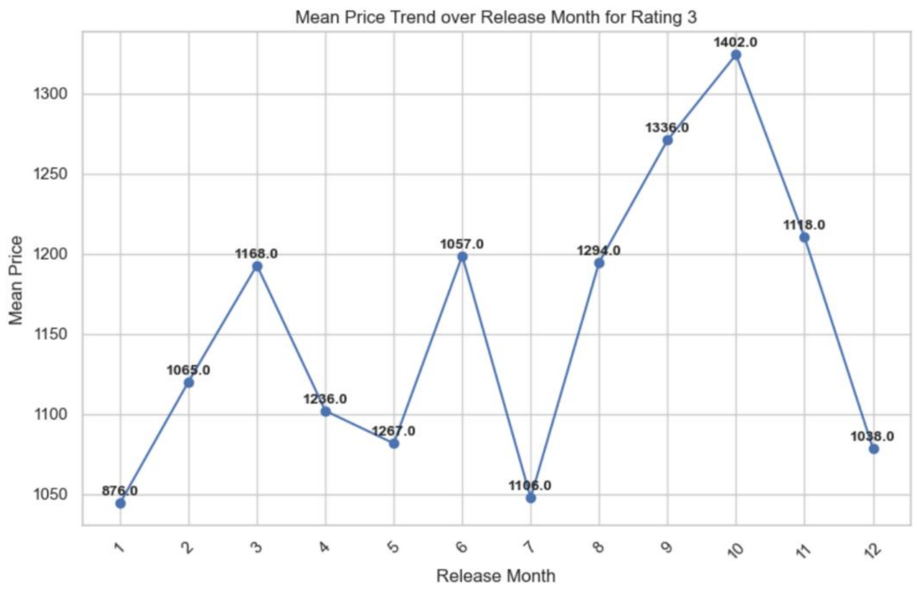


Figure 15 – Mean Price Trend in Month

Based on the analysis result of the dataset, an upward trend in the mean price of products with a rating of 3 has been observed, starting from July. The increase in the mean price continued until it reached its peak in October. This pattern suggests that there was a significant growth in the average price of products with a rating of 3 during this period.

There could be various potential reasons behind the observed uptrend in the mean price of products with a rating of 3 from July to October. Some of these reasons might include:

- Seasonal Demand: October is typically a busy month for game releases, as many publishers time their releases to coincide with the holiday season. As a result, more new and high-demand games may be released in October, which could drive up the average price of games on Steam.
- Sales and Promotions: Steam often runs sales and promotions throughout the year, with some of the biggest sales happening in the summer and winter. During these sales, games are often sold at a significant discount, which could bring down the average price of games in July. However, as these sales end and the discounts are no longer available, the average price of games may increase again.
- Publisher Pricing Strategies: Game publishers are strategically pricing their games to take advantage of the increased demand in the fall. They may increase the price of their games during this time to maximize profits, knowing that consumers are willing to pay more for new and popular games.

5.6 Boxplot

This boxplot shows that short description, release month have relatively higher Mean than other. 9 distributions such as Price, Discount, Required Age, Owners, CCU, Positive Reviews, Negative Reviews, Tags and Release Dates included potential outliers and had relatively low variation.

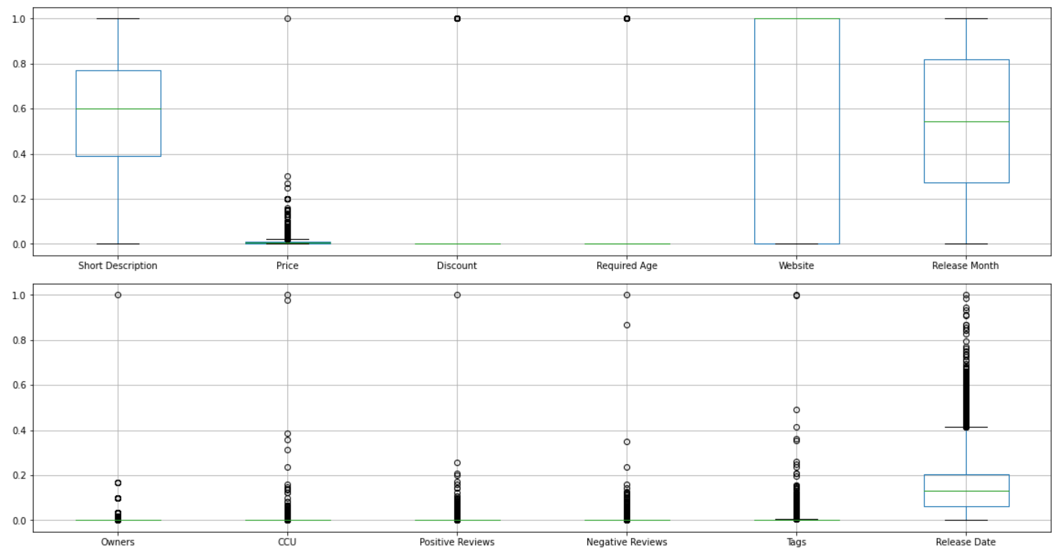


Figure 16 - Boxplot

This graph shows the price range for Steam games by release year in the Upper-tier collection. The average price in 2002 and 2022 is the same, but the number of games released in 2002 was less than in 2022. It is found that the average price is influenced by the total number of games released in different years and prices.

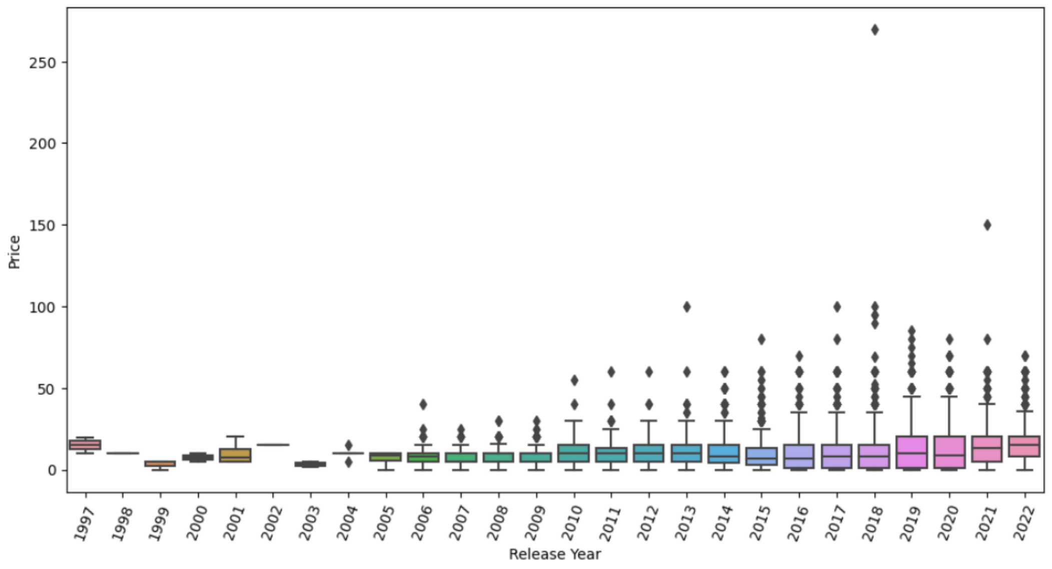


Figure 17 – mean price of various release year of steam games

5.7 Parallel Coordinates Plot

Parallel coordinate plot analysis was used to compare numerous variables and ascertain their relationships. For instance, they were comparing multiple goods with the same features for various models in the upper collection (resembling the game's short description length, price, tags, positive reviews, negative reviews, release month, and release date by rating). The top-level groups were utilized to understand how they link to prevent the parallel coordinates diagram from becoming very cluttered. As a result, it resolved the problem where very dense data was challenging to read.

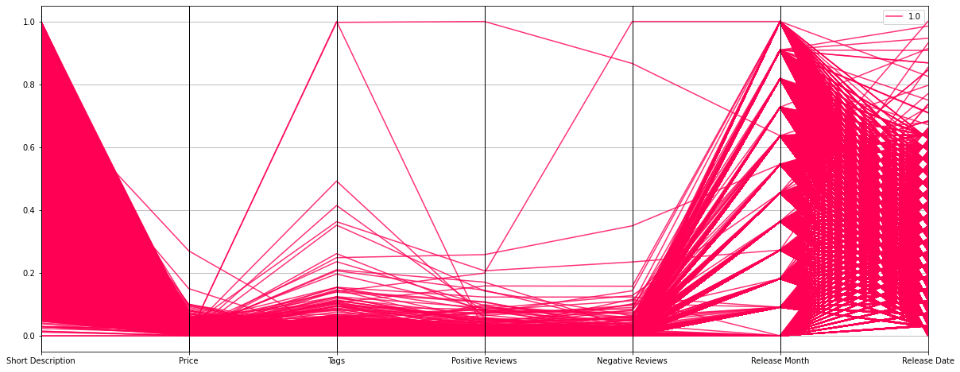
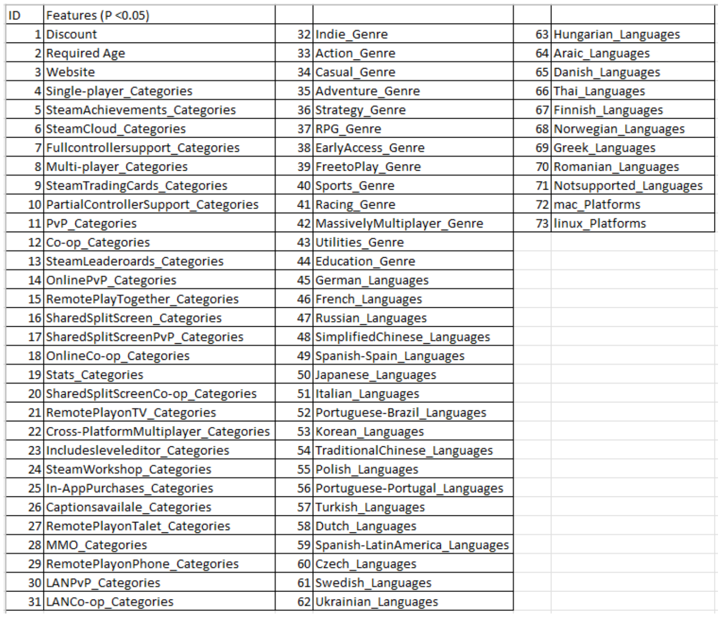


Figure 18 Parallel Coordinates Plot of upper-tier collection

5.8 Hypothesis Testing - T-test

The null hypothesis was either that the groups' means are equal or that there is no appreciable difference between the two groups' mean scores overall. The t-test was used to determine whether the statistical difference between the Rating means of each column was significant or not. Consequently, 73 columns rejected the null hypothesis because their p-value was less than 0.05, whereas six columns failed to do so since their p-value was more than 0.05.



ID	Features (P <0.05)				
1	Discount	32	Indie_Genre	63	Hungarian_Languages
2	Required Age	33	Action_Genre	64	Araic_Languages
3	Website	34	Casual_Genre	65	Danish_Languages
4	Single-player_Categories	35	Adventure_Genre	66	Thai_Languages
5	SteamAchievements_Categories	36	Strategy_Genre	67	Finnish_Languages
6	SteamCloud_Categories	37	RPG_Genre	68	Norwegian_Languages
7	Fullcontrollersupport_Categories	38	EarlyAccess_Genre	69	Greek_Languages
8	Multi-player_Categories	39	FreetoPlay_Genre	70	Romanian_Languages
9	SteamTradingCards_Categories	40	Sports_Genre	71	Notsupported_Languages
10	PartialControllerSupport_Categories	41	Racing_Genre	72	mac_Platforms
11	PvP_Categories	42	MassivelyMultiplayer_Genre	73	linux_Platforms
12	Co-op_Categories	43	Utilities_Genre		
13	SteamLeaderoards_Categories	44	Education_Genre		
14	OnlinePvP_Categories	45	German_Languages		
15	RemotePlayTogether_Categories	46	French_Languages		
16	SharedSplitScreen_Categories	47	Russian_Languages		
17	SharedSplitScreenPvP_Categories	48	SimplifiedChinese_Languages		
18	OnlineCo-op_Categories	49	Spanish-Spain_Languages		
19	Stats_Categories	50	Japanese_Languages		
20	SharedSplitScreenCo-op_Categories	51	Italian_Languages		
21	RemotePlayonTV_Categories	52	Portuguese-Brazil_Languages		
22	Cross-PlatformMultiplayer_Categories	53	Korean_Languages		
23	Includesleveleditor_Categories	54	TraditionalChinese_Languages		
24	SteamWorkshop_Categories	55	Polish_Languages		
25	In-AppPurchases_Categories	56	Portuguese-Portugal_Languages		
26	Captionsavailale_Categories	57	Turkish_Languages		
27	RemotePlayonTalet_Categories	58	Dutch_Languages		
28	MMO_Categories	59	Spanish-LatinAmerica_Languages		
29	RemotePlayonPhone_Categories	60	Czech_Languages		
30	LANPvP_Categories	61	Swedish_Languages		
31	LANCo-op_Categories	62	Ukrainian_Languages		

Figure 19 - Features of p-values greater than 0.05

5.9 Text Mining

Text mining is a way to extract valuable insights from the data for analyzing unstructured textual data. It aims to compare top-ranking and low-ranking games. The analysis focused on determining the frequency of the words 'Name', 'Short Description', and 'Tags'. Split the features to corpus one by one, preprocess text function used to filter the stop-words, numbers, regex and only display the 100 most frequency tokens. Finally, the results showed the word cloud and weight for each word. Results indicate that the top- and low-ranking rows had similar outcomes, except VR games being associated with negative reviews.

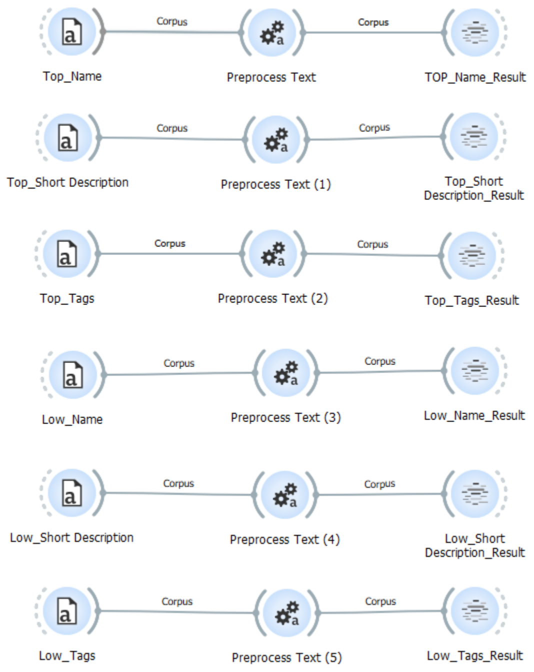


Figure 20 - Text Mining workflow in Orange

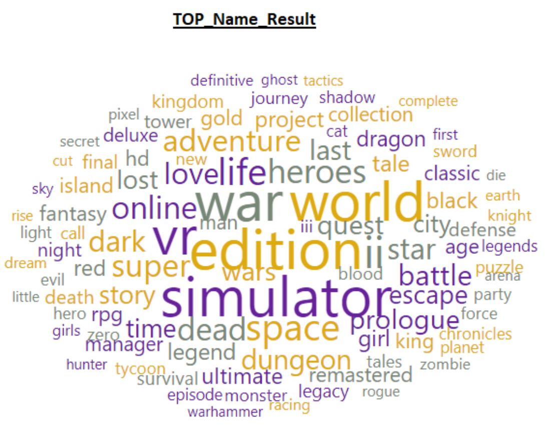


Figure 21 - Word Cloud frequencies of game names in upper-tier collection – Positive

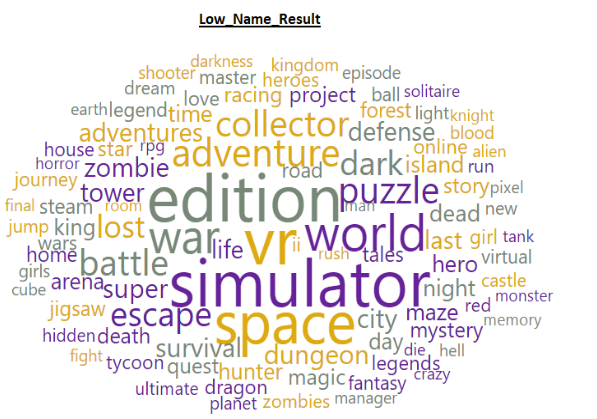


Figure 22 - Word Cloud frequencies of game names in lower-tier collection - Negative

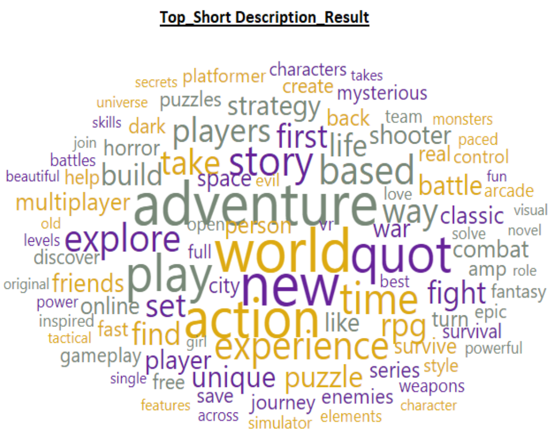


Figure 23 - Word Cloud frequencies of short description in Upper-tier collection – Positive

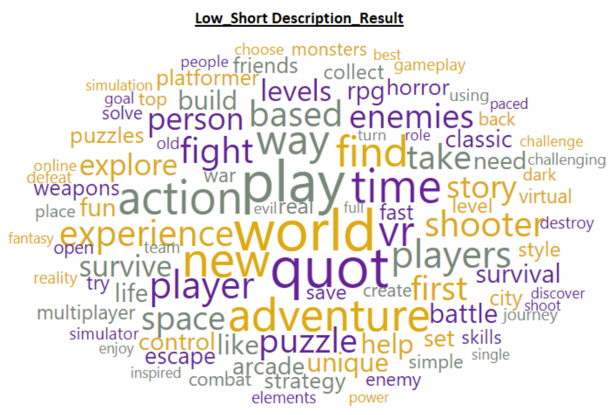


Figure 24 - Word Cloud frequencies of short description in lower-tier collection – Negative

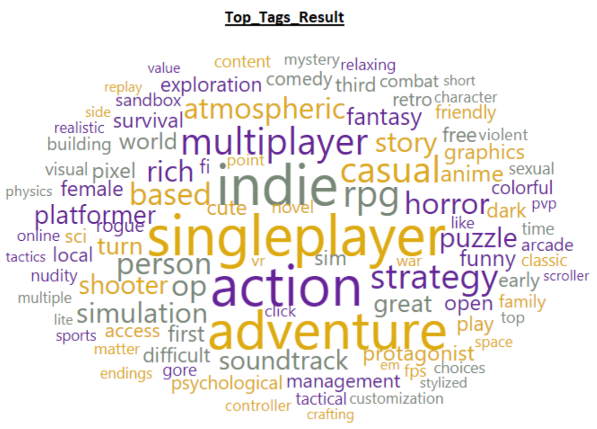


Figure 25 - Word Cloud frequencies of tags in Upper-tier collection - Positive

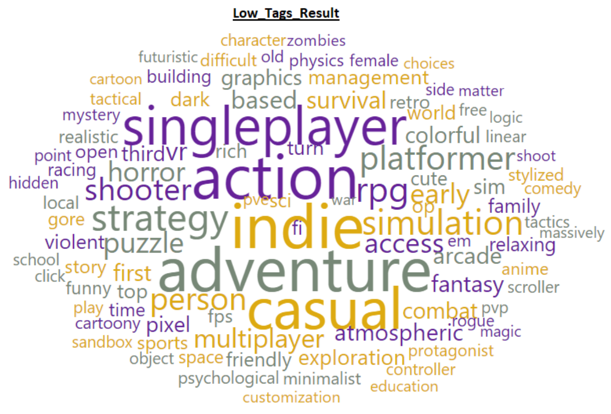


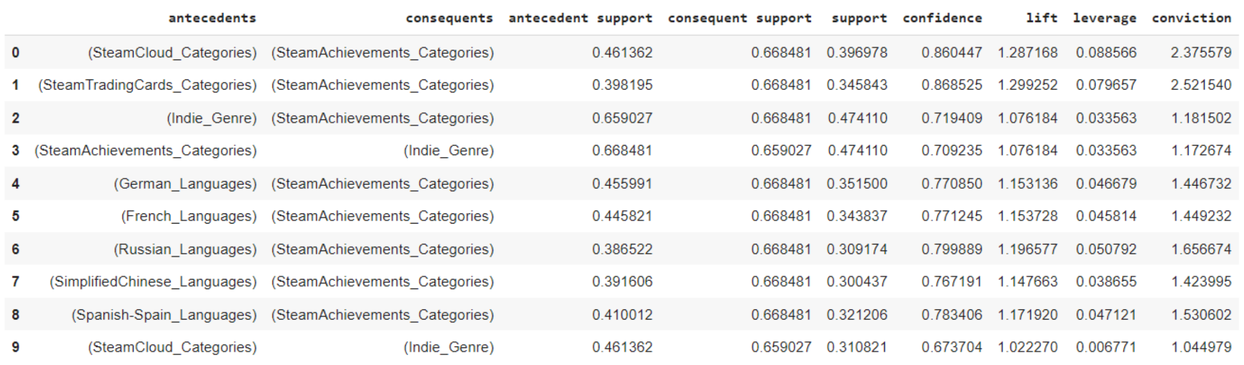
Figure 26 - Word Cloud frequencies of tags in lower-tier collection – Negative

5.10 Association Rule Mining - Apriori Algorithm

Association Rules have been applied to upper-tier collections, and this analysis is used for advice. Four features were used from the original data: Categories, Genres, Languages, and Platforms. We convert the features list to a One-Hot Encoded Boolean list. Then we set the min_support value with a threshold value of 30%. Besides, we can state that product Y has a purchase rate of 60% or higher when product X is purchased. The antecedent and consequent items are higher correlation which means this information can help us to make business decisions when we develop a new game.

Finding of examine the first index value (item0):

- 46% of games have Steam Cloud Category.
- 66% of games included Steam Achievements Category.
- 39% of users like playing games have Steam Cloud Category and Steam Achievements Category.
- 86% of games have Steam Cloud Category than Steam Achievements Category.
- The game that included Steam Cloud Category are 1.29 times more likely to Steam Achievements Category than expected by chance.
- There have strong positive relationship (high correlated) between the antecedents item and consequents item.



	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(SteamCloud_Categories)	(SteamAchievements_Categories)	0.461362	0.668481	0.396978	0.860447	1.287168	0.088566	2.375579
1	(SteamTradingCards_Categories)	(SteamAchievements_Categories)	0.398195	0.668481	0.345843	0.868525	1.299252	0.079657	2.521540
2	(Indie_Genre)	(SteamAchievements_Categories)	0.659027	0.668481	0.474110	0.719409	1.076184	0.033563	1.181502
3	(SteamAchievements_Categories)	(Indie_Genre)	0.668481	0.659027	0.474110	0.709235	1.076184	0.033563	1.172674
4	(German_Languages)	(SteamAchievements_Categories)	0.455991	0.668481	0.351500	0.770850	1.153136	0.046679	1.446732
5	(French_Languages)	(SteamAchievements_Categories)	0.445821	0.668481	0.343837	0.771245	1.153728	0.045814	1.449232
6	(Russian_Languages)	(SteamAchievements_Categories)	0.386522	0.668481	0.309174	0.799889	1.196577	0.050792	1.656674
7	(SimplifiedChinese_Languages)	(SteamAchievements_Categories)	0.391606	0.668481	0.300437	0.767191	1.147663	0.038655	1.423995
8	(Spanish-Spain_Languages)	(SteamAchievements_Categories)	0.410012	0.668481	0.321206	0.783406	1.171920	0.047121	1.530602
9	(SteamCloud_Categories)	(Indie_Genre)	0.461362	0.659027	0.310821	0.673704	1.022270	0.006771	1.044979

Figure 27 – Association rule result in Upper-tier collection

6. Model Training and Evaluation

6.1 Overview

Various machine learning algorithms were used for classification to predict the user rating classes (0: Very Bad; 1: Bad; 2: Good; 3: Very Good), which includes:

- Linear Support Vector Machine (SVM)
- K-Nearest Neighbor (KNN)
- Decision Tree
- Random Forest
- Logistic Regression
- Light Gradient Boosting Machine (LightGBM)
- eXtreme Gradient Boosting (XGBoost)
- Neural Network

Ensemble model (Voting Classifier) was used to make the final prediction. The 'hard' parameter was used to specify that the final model will use the mode of the predicted class labels (i.e., the class label that appears most frequently) as the final prediction.

6.2 Dataset Splitting

The dataset was split into training and testing datasets, with 80% of the data assigned to the training dataset and 20% of the data assigned to the testing dataset.

In the training dataset, 5-fold cross-validation was applied to get a cross-validation score. This was done to evaluate the performance of models on a variety of different subsets of the training data and reduce the risk of overfitting.

6.3 Feature Reduction

The dimension of dataset was reduced from 82 to 60 features by using PCA component explained variance 98% by using the ensemble model for testing.

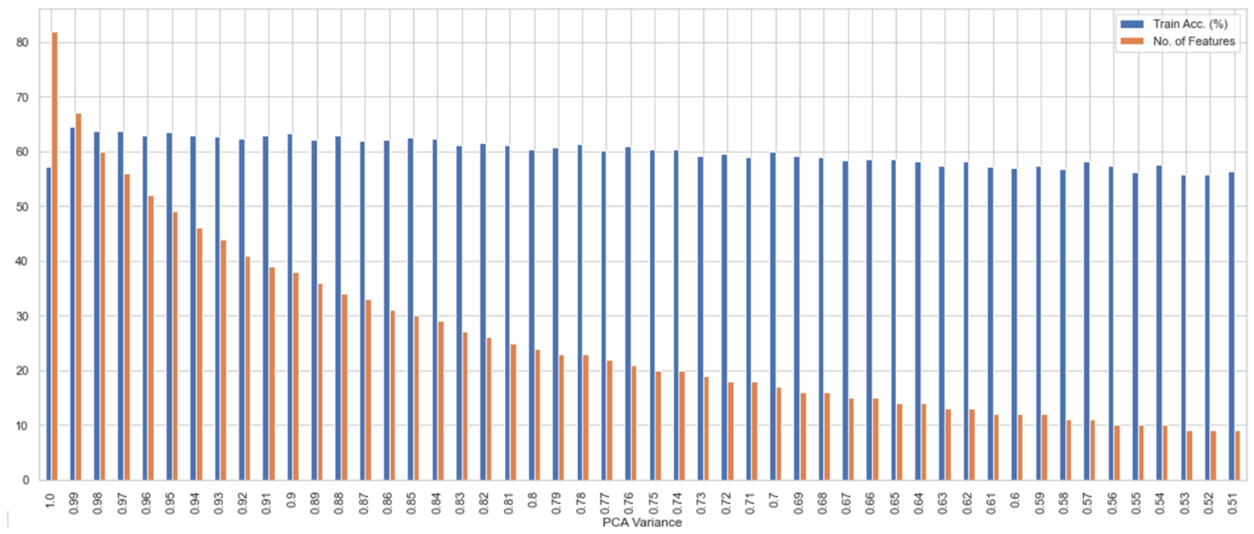
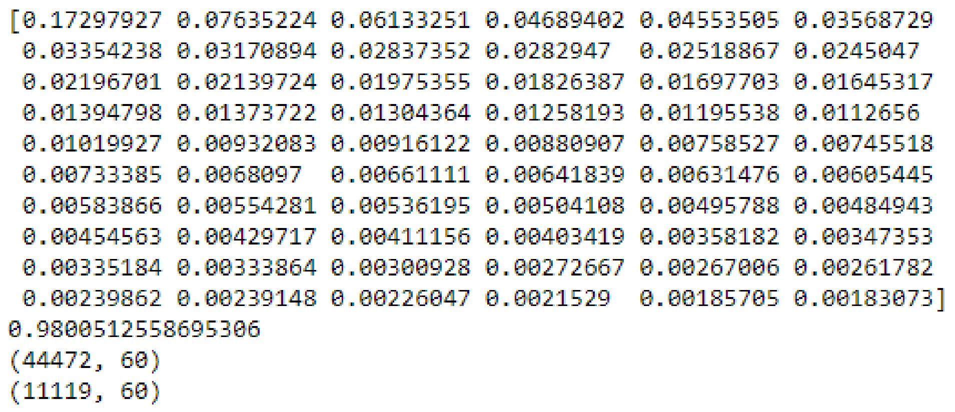


Figure 28 – Accuracies in different PCA variances

The distribution of the reduced 60 components was presented below.



```
[0.17297927 0.07635224 0.06133251 0.04689402 0.04553505 0.03568729
 0.03354238 0.03170894 0.02837352 0.0282947  0.02518867 0.0245047
 0.02196701 0.02139724 0.01975355 0.01826387 0.01697703 0.01645317
 0.01394798 0.01373722 0.01304364 0.01258193 0.01195538 0.0112656
 0.01019927 0.00932083 0.00916122 0.00880907 0.00758527 0.00745518
 0.00733385 0.0068097  0.00661111 0.00641839 0.00631476 0.00605445
 0.00583866 0.00554281 0.00536195 0.00504108 0.00495788 0.00484943
 0.00454563 0.00429717 0.00411156 0.00403419 0.00358182 0.00347353
 0.00335184 0.00333864 0.00300928 0.00272667 0.00267006 0.00261782
 0.00239862 0.00239148 0.00226047 0.0021529  0.00185705 0.00183073]
0.9800512558695306
(44472, 60)
(11119, 60)
```

Figure 29 – Distribution of PCA variance on 0.98

6.4 Statistical model

Model 1: k-NN

K was selected as 2 from elbow method.

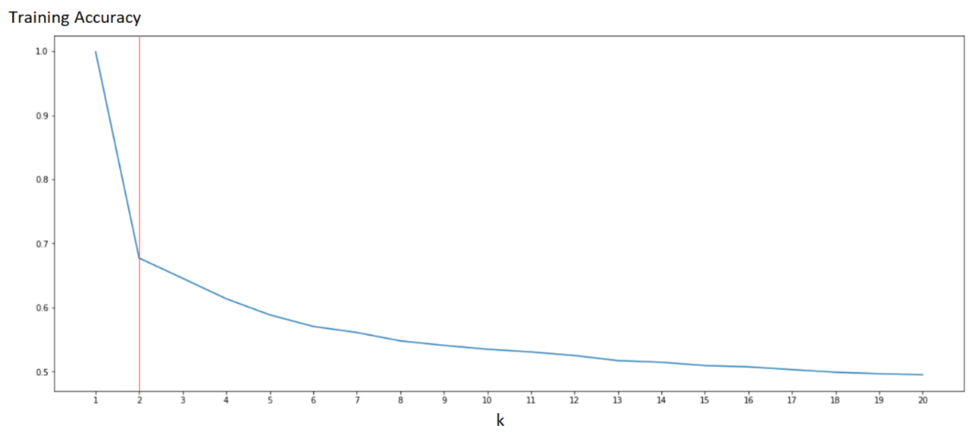


Figure 30 – Accuracies in k-NN

Permutation feature importance was used to identify the most important features for a k-nearest neighbor (k-NN) model. The analysis revealed that the top 3 important features, in descending order, are Release Month, Short Description, and Website.

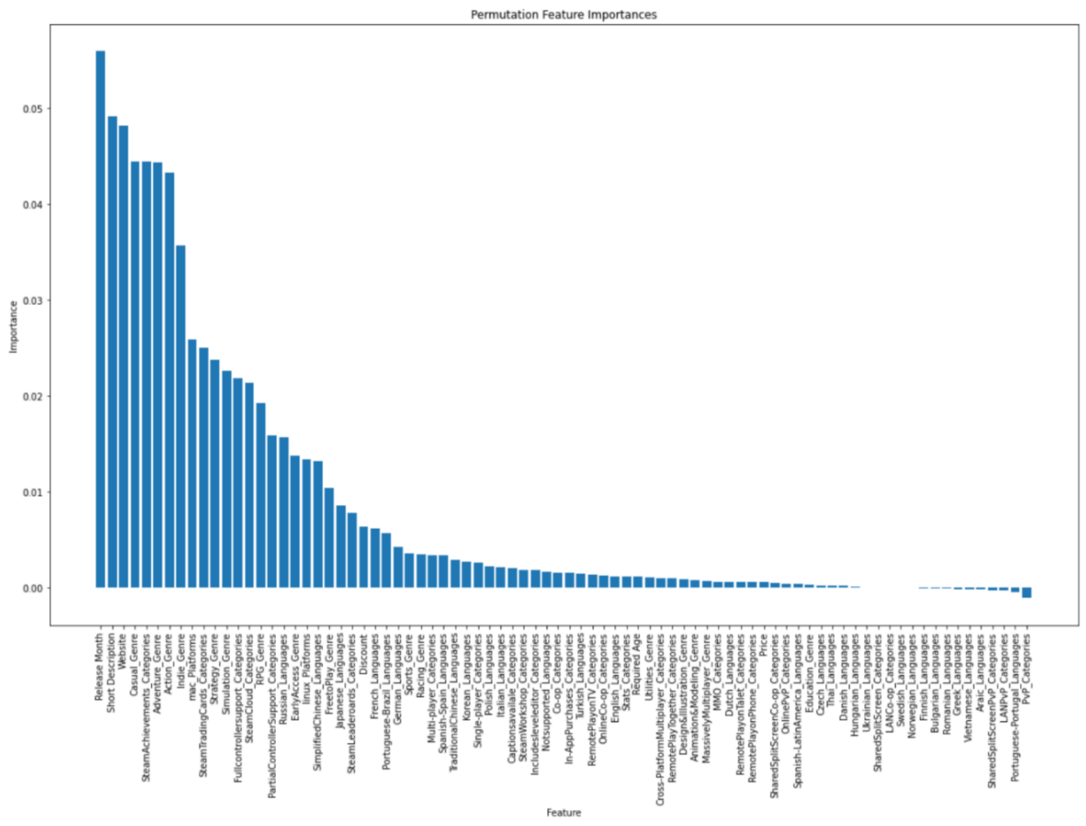
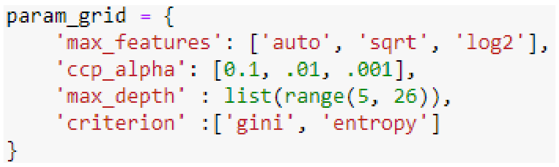


Figure 31 – Permutation Feature Importance of K-NN

Model 2: Decision Tree

Hyperparameter tuning for a decision tree classifier was performed by using the GridSearchCV to search for the best set of hyperparameters and optimized the model performance and avoided overfitting.

The hyperparameters to tune were defined in the `param_grid` dictionary as follows:



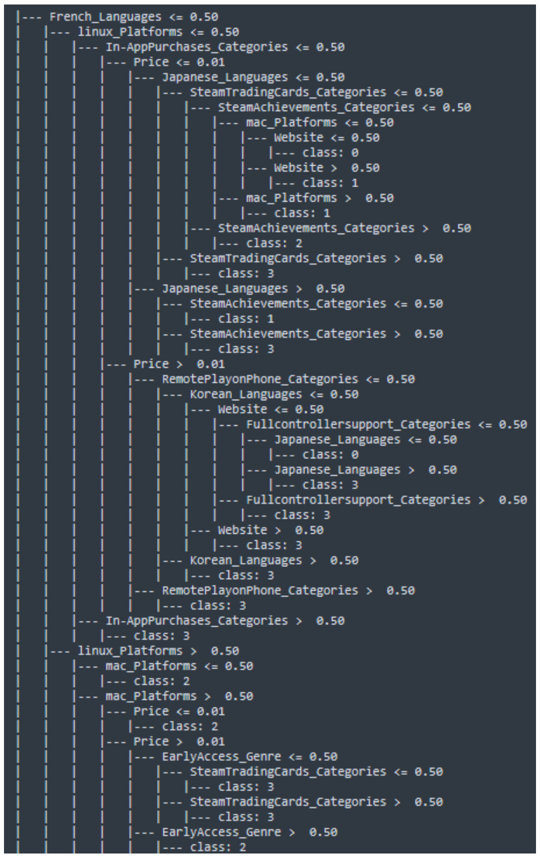
```
param_grid = {
    'max_features': ['auto', 'sqrt', 'log2'],
    'ccp_alpha': [0.1, .01, .001],
    'max_depth' : list(range(5, 26)),
    'criterion' :['gini', 'entropy']
}
```

Figure 32 – Hyperparameters of Decision Tree

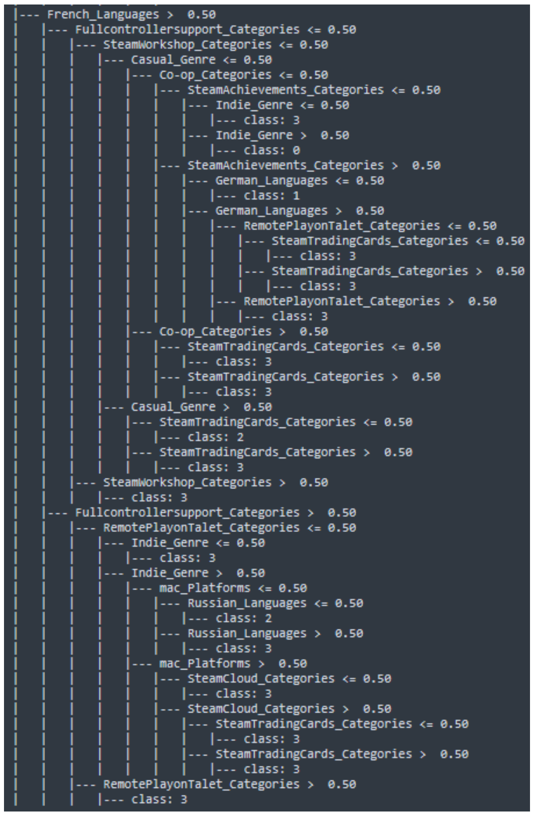
- `max_features`: The number of features to consider when looking for the best split.
- `ccp_alpha`: The complexity parameter used for pruning the decision tree.
- `max_depth`: The maximum depth of the decision tree.
- `criterion`: The impurity measure to use for splitting.

The result showed that the grid search performed 378 fits with 5-fold cross-validation, totaling 1890 fits. The best estimator found by the grid search had the hyperparameters `ccp_alpha`=0.001, `criterion`='entropy', `max_depth=`20, and `max_features`='sqrt'.

The rules of decision tree model were presented in a hierarchical format as below.



```
|--- French_Languages <= 0.50
|   |--- linux_Platforms <= 0.50
|   |   |--- In-AppPurchases_Categories <= 0.50
|   |   |   |--- Price <= 0.01
|   |   |   |   |--- Japanese_Languages <= 0.50
|   |   |   |   |   |--- SteamTradingCards_Categories <= 0.50
|   |   |   |   |   |   |--- SteamAchievements_Categories <= 0.50
|   |   |   |   |   |   |   |--- mac_Platforms <= 0.50
|   |   |   |   |   |   |   |   |--- Website <= 0.50
|   |   |   |   |   |   |   |   |   |--- class: 0
|   |   |   |   |   |   |   |   |--- Website >  0.50
|   |   |   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |   |   |--- mac_Platforms >  0.50
|   |   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |   |--- SteamAchievements_Categories >  0.50
|   |   |   |   |   |   |   |--- class: 2
|   |   |   |   |   |--- SteamTradingCards_Categories >  0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |   |--- Japanese_Languages >  0.50
|   |   |   |   |   |--- SteamAchievements_Categories <= 0.50
|   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |--- SteamAchievements_Categories >  0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |--- Price >  0.01
|   |   |   |   |--- RemotePlayonPhone_Categories <= 0.50
|   |   |   |   |   |--- Korean_Languages <= 0.50
|   |   |   |   |   |   |--- Website <= 0.50
|   |   |   |   |   |   |   |--- Fullcontrollersupport_Categories <= 0.50
|   |   |   |   |   |   |   |   |--- Japanese_Languages <= 0.50
|   |   |   |   |   |   |   |   |   |--- class: 0
|   |   |   |   |   |   |   |   |--- Japanese_Languages >  0.50
|   |   |   |   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |   |   |--- Fullcontrollersupport_Categories >  0.50
|   |   |   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |   |--- Website >  0.50
|   |   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |--- Korean_Languages >  0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |   |--- RemotePlayonPhone_Categories >  0.50
|   |   |   |   |   |--- class: 3
|   |   |--- In-AppPurchases_Categories >  0.50
|   |   |   |--- class: 3
|   |--- linux_Platforms >  0.50
|   |   |--- mac_Platforms <= 0.50
|   |   |   |--- class: 2
|   |   |--- mac_Platforms >  0.50
|   |   |   |--- Price <= 0.01
|   |   |   |   |--- class: 2
|   |   |   |--- Price >  0.01
|   |   |   |   |--- EarlyAccess_Genre <= 0.50
|   |   |   |   |   |--- SteamTradingCards_Categories <= 0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |--- SteamTradingCards_Categories >  0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |   |--- EarlyAccess_Genre >  0.50
|   |   |   |   |   |--- class: 2
```



```
|--- French_Languages >  0.50
|   |--- Fullcontrollersupport_Categories <= 0.50
|   |   |--- SteamWorkshop_Categories <= 0.50
|   |   |   |--- Casual_Genre <= 0.50
|   |   |   |   |--- Co-op_Categories <= 0.50
|   |   |   |   |   |--- SteamAchievements_Categories <= 0.50
|   |   |   |   |   |   |--- Indie_Genre <= 0.50
|   |   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |   |--- Indie_Genre >  0.50
|   |   |   |   |   |   |   |--- class: 0
|   |   |   |   |   |--- SteamAchievements_Categories >  0.50
|   |   |   |   |   |   |--- German_Languages <= 0.50
|   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |   |--- German_Languages >  0.50
|   |   |   |   |   |   |   |--- RemotePlayonTalet_Categories <= 0.50
|   |   |   |   |   |   |   |   |--- SteamTradingCards_Categories <= 0.50
|   |   |   |   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |   |   |   |--- SteamTradingCards_Categories >  0.50
|   |   |   |   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |   |   |--- RemotePlayonTalet_Categories >  0.50
|   |   |   |   |   |   |   |   |--- class: 3
|   |   |   |   |--- Co-op_Categories >  0.50
|   |   |   |   |   |--- SteamTradingCards_Categories <= 0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |--- SteamTradingCards_Categories >  0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |--- Casual_Genre >  0.50
|   |   |   |   |--- SteamTradingCards_Categories <= 0.50
|   |   |   |   |   |--- class: 2
|   |   |   |   |--- SteamTradingCards_Categories >  0.50
|   |   |   |   |   |--- class: 3
|   |   |--- SteamWorkshop_Categories >  0.50
|   |   |   |--- class: 3
|   |--- Fullcontrollersupport_Categories >  0.50
|   |   |--- RemotePlayonTalet_Categories <= 0.50
|   |   |   |--- Indie_Genre <= 0.50
|   |   |   |   |--- class: 3
|   |   |   |--- Indie_Genre >  0.50
|   |   |   |   |--- mac_Platforms <= 0.50
|   |   |   |   |   |--- Russian_Languages <= 0.50
|   |   |   |   |   |   |--- class: 2
|   |   |   |   |   |--- Russian_Languages >  0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |   |--- mac_Platforms >  0.50
|   |   |   |   |   |--- SteamCloud_Categories <= 0.50
|   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |--- SteamCloud_Categories >  0.50
|   |   |   |   |   |   |--- SteamTradingCards_Categories <= 0.50
|   |   |   |   |   |   |   |--- class: 3
|   |   |   |   |   |   |--- SteamTradingCards_Categories >  0.50
|   |   |   |   |   |   |   |--- class: 3
|   |   |--- RemotePlayonTalet_Categories >  0.50
|   |   |   |--- class: 3
```

Figure 33 – Hierarchical format of Decision Tree

The feature importance was extracted from a decision tree model using the feature_importances_ attribute. The analysis revealed that the top 3 important features, in descending order, were French_Languages, SteamTradingCards_Categories, and Price.

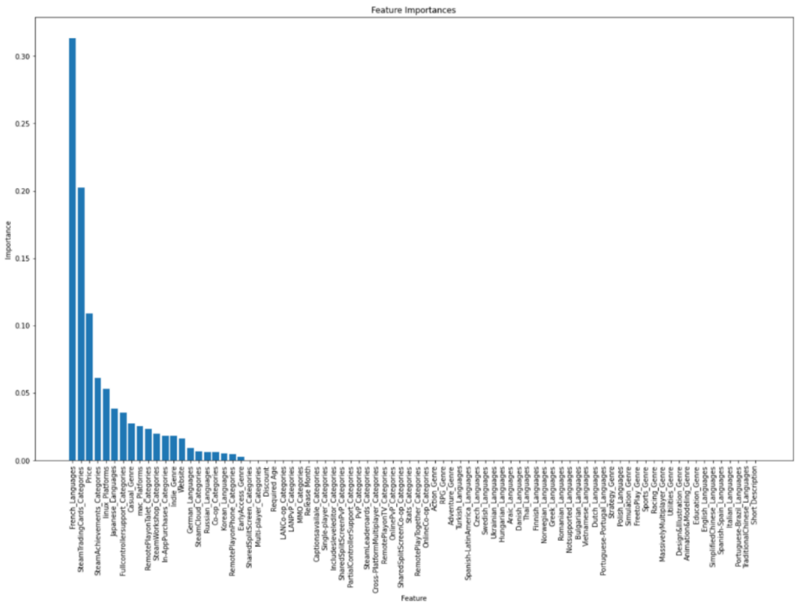
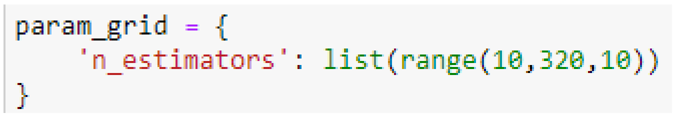


Figure 34 – Feature Importance of Decision Tree

Model 3: Random Forest

Hyperparameter tuning for a Random Forest classifier was performed by using the GridSearchCV. The hyperparameters used for the Random Forest classifier were selected by using the best estimator found in decision tree classifier. The values for `criterion`, `max_depth`, and `max_features` were obtained from the best estimator using `grid_search.best_params_`.



```
param_grid = {
    'n_estimators': list(range(10,320,10))
}
```

Figure 35 – Hyperparameters of Random Forest

The hyperparameter `n_estimators` specified the number of decision trees to be used in the Random Forest. The code defined a dictionary `param_grid` that contained the `n_estimators` hyperparameter with a range of values from 10 to 320 with a step size of 10.

The result showed that the grid search performed 31 fits with 5-fold cross-validation, totaling 155 fits. The best estimator found by the grid search had the hyperparameters `criterion`='entropy', `max_depth`=20, `max_features`='sqrt', and `n_estimators`=280.

The top 3 important features were SteamTradingCards_Categories, Price, and SteamCloud_Categories.

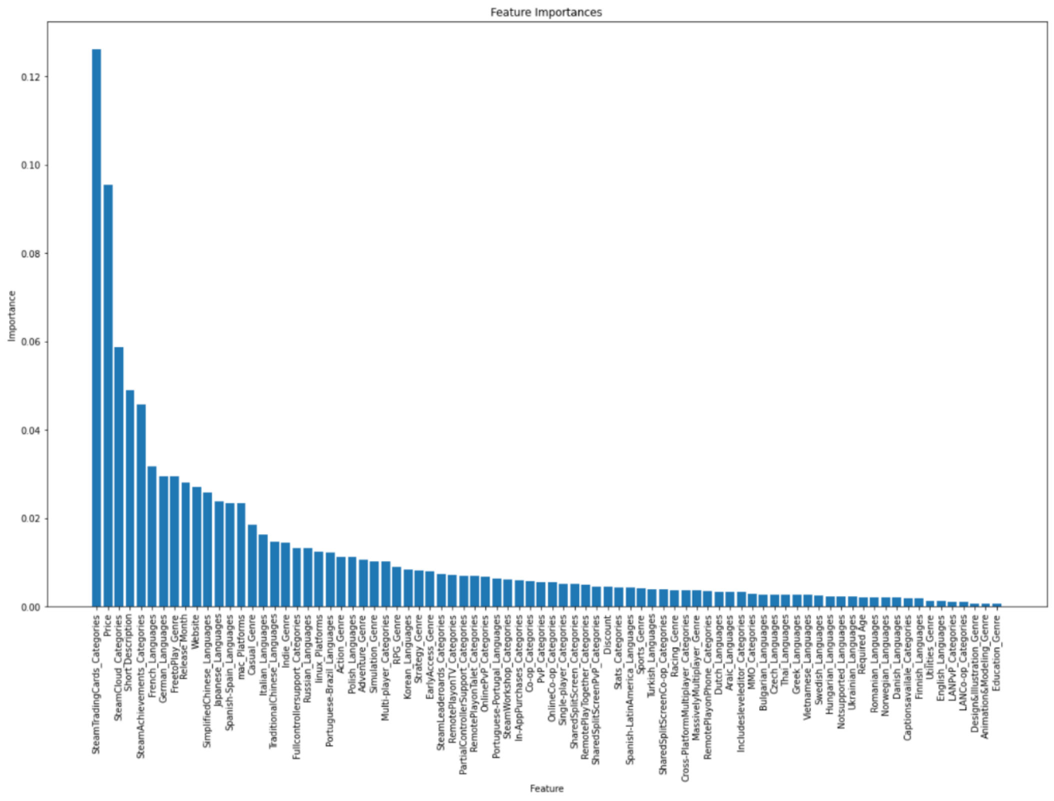


Figure 36 – Feature Importance of Random Forest

Model 4: SVM

The `dual` parameter in the LinearSVC() function was set to be False, which means that the primal formulation of the SVM problem was used instead of the dual formulation. This could be faster and more memory-efficient when the number of training samples is larger than the number of features.

```
svm.LinearSVC(dual=False)
```

Feature importance was extracted from a support vector machine (SVM) model by getting the coefficients of the model using the coef_ attribute. The analysis revealed that the top 3 important features, in descending order, were Price, FreetoPlay_Genre, and SteamCloud_Categories.

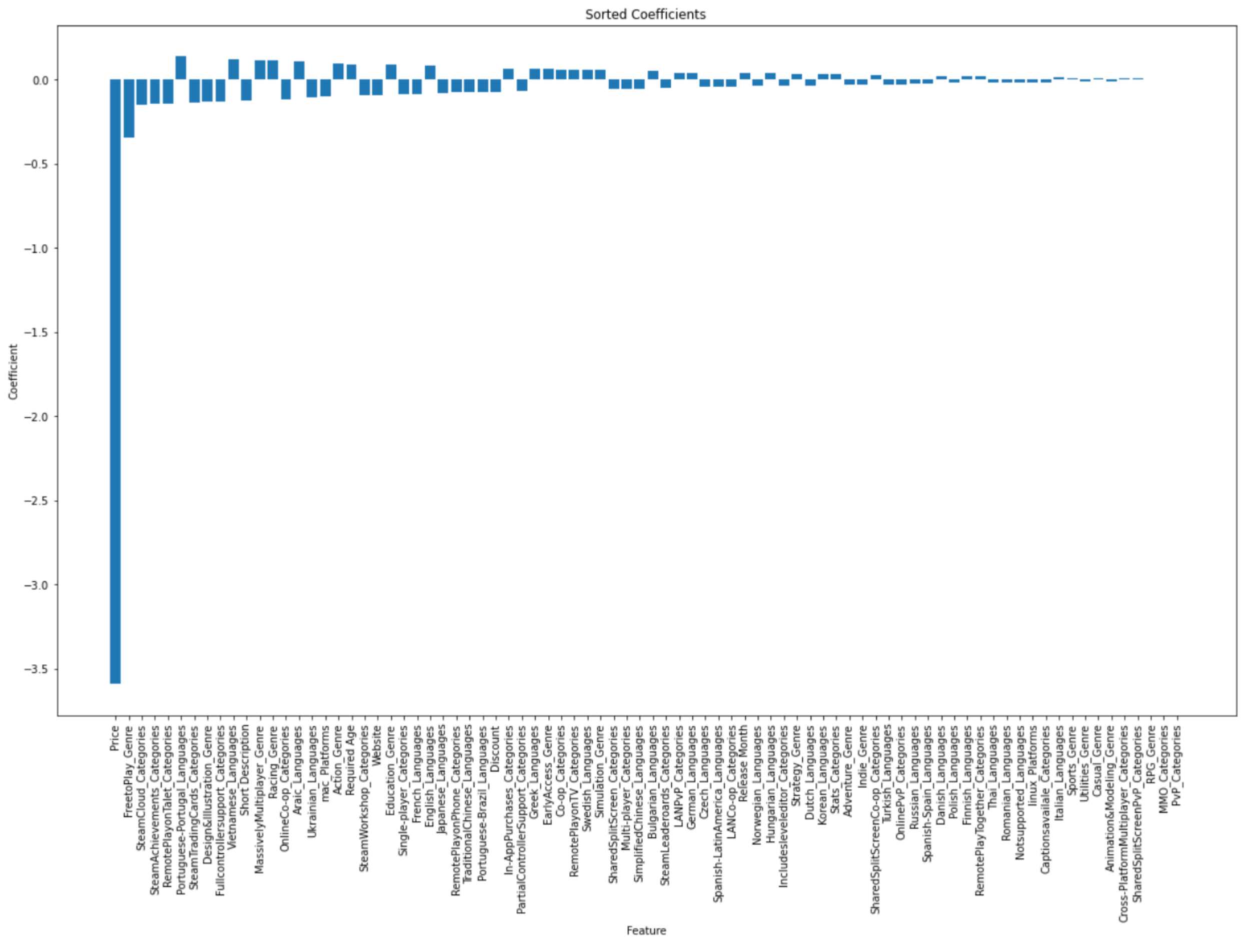


Figure 37 – Feature Coefficient of SVM

Model 5: Logistic Regression

Hyperparameters were specified in logistic regression model as below.

```
LogisticRegression(multi_class='multinomial', solver='lbfgs', max_iter=10000)
```

- `multi_class`='multinomial': Specifies the type of multi-class logistic regression model to use.
- `solver`='lbfgs': Specifies the optimization algorithm to use. The 'lbfgs' solver is a quasi-Newton method that works well for small to medium-sized datasets.
- `max_iter`=10000: Specifies the maximum number of iterations for the solver to converge. It is set to 10,000 to ensure convergence.

The top 3 important features were Price, FreetoPlay_Genre, and SteamCloud_Categories.

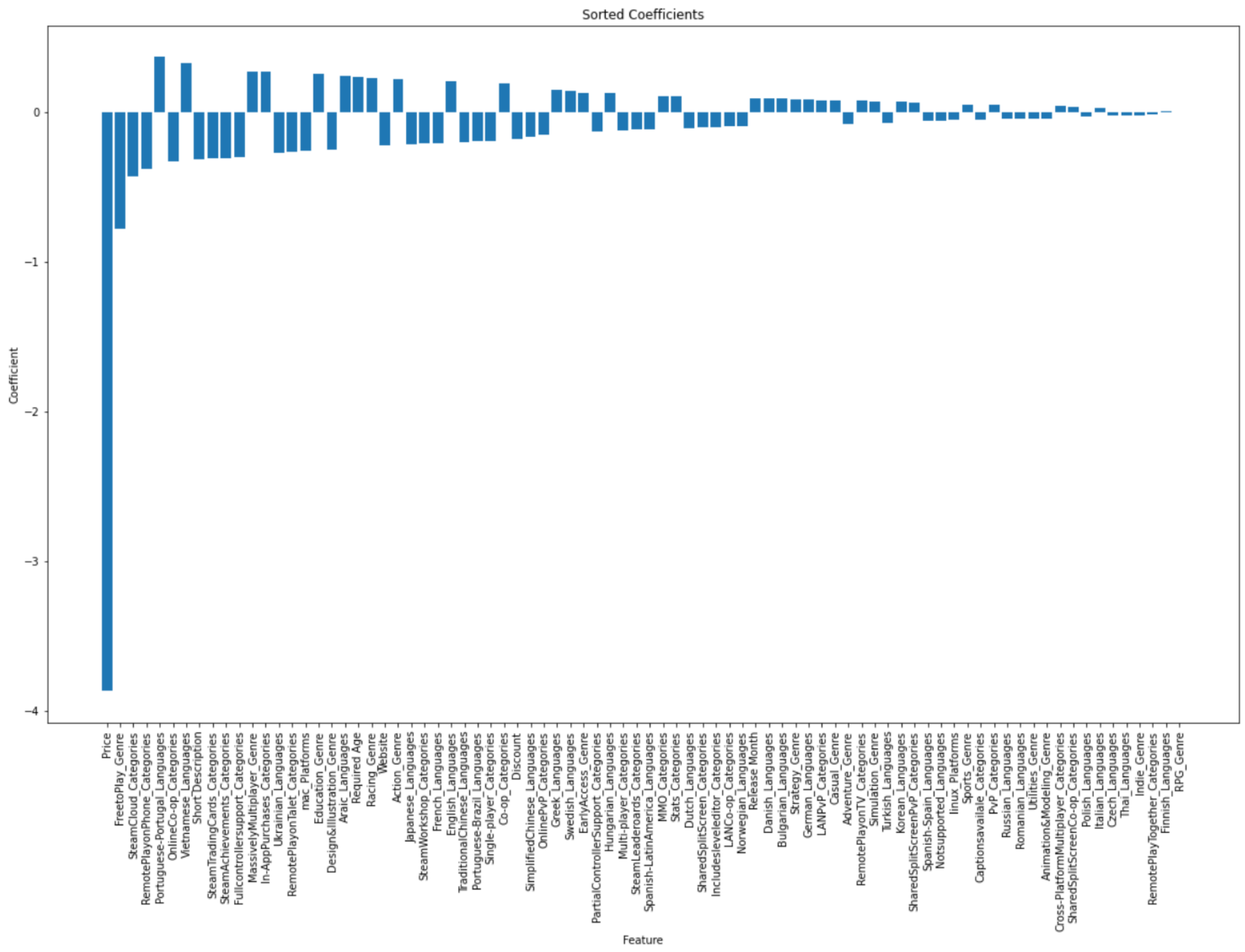


Figure 38 – Feature Coefficient of Logistic Regression

Model 6: LightGBM

A gradient boosting algorithm with tree-based learning algorithms was used.

The top 3 important features were Short Description, Price, and Release Month.

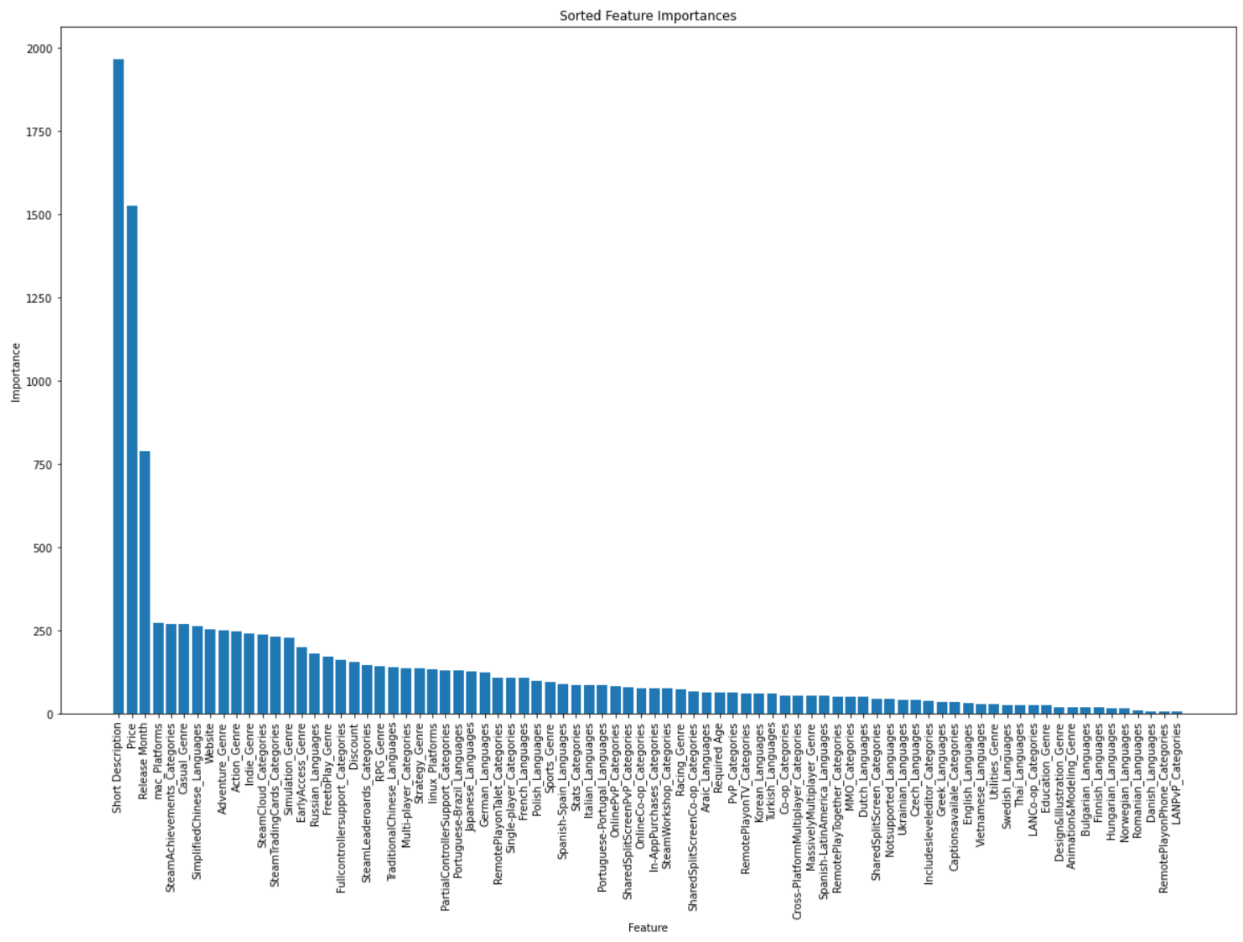


Figure 39 – Feature Importance of LightGBM

Model 7: XGBoost

Another gradient boosting algorithm with tree-based learning algorithms was used.

A log loss metric 'mlogloss' was used for multi-class classification problems. It measured the probability error between the predicted probabilities and the true labels. The lower the log loss value, the better the model's performance.

```
XGBClassifier(eval_metric='mlogloss')
```

The top 3 important features were SteamTradingCards_Categories, FreetoPlay_Genre, and SteamCloud_Categories.

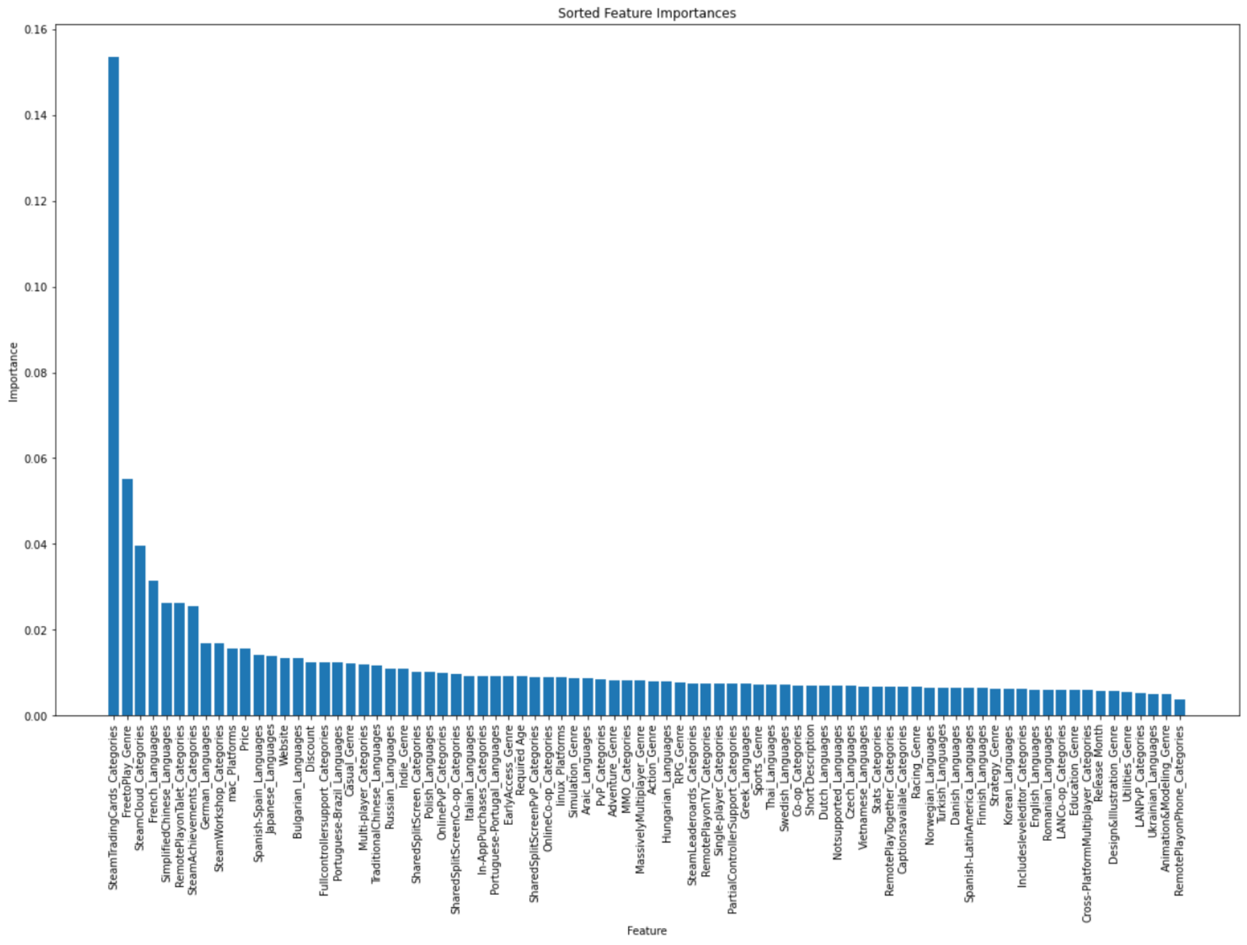
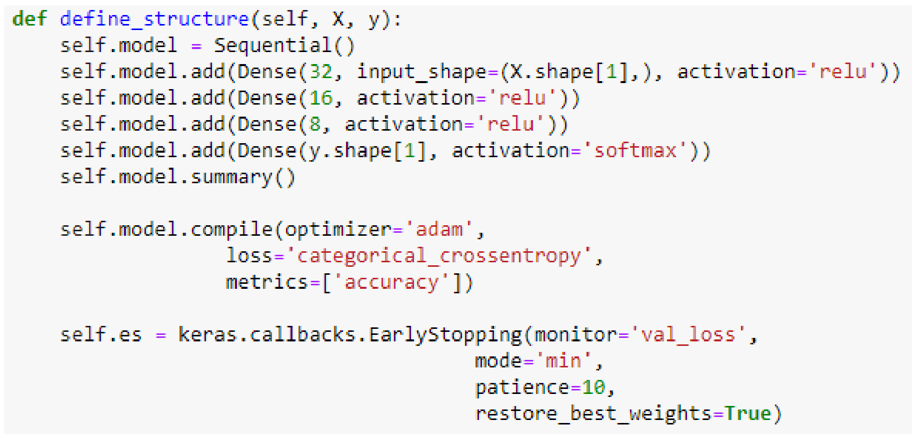


Figure 40 – Feature Importance of XGBoost

Model 8: Neural Networks

The architecture of this neural network model had four layers, including one input layer, two hidden layers, and one output layer. The model was compiled using Adam optimizer, categorical cross-entropy loss function, and accuracy metric. The input layer had 32 neurons with ReLU activation function. The second and third layers each had 16 and 8 neurons respectively, both with ReLU activation function. The output layer had 4 neurons which were equal to the number of classes in the output, with the softmax activation function. The early stopping callback was also defined, which was for preventing overfitting and improving the generalization performance of the model during training. The callback monitored the validation loss, waited for 10 epochs before stopping if the validation loss did not improve, and then restored the weights of the best model.



```python
def define_structure(self, X, y):
    self.model = Sequential()
    self.model.add(Dense(32, input_shape=(X.shape[1],), activation='relu'))
    self.model.add(Dense(16, activation='relu'))
    self.model.add(Dense(8, activation='relu'))
    self.model.add(Dense(y.shape[1], activation='softmax'))
    self.model.summary()

    self.model.compile(optimizer='adam',
                       loss='categorical_crossentropy',
                       metrics=['accuracy'])

    self.es = keras.callbacks.EarlyStopping(monitor='val_loss',
                                            mode='min',
                                            patience=10,
                                            restore_best_weights=True)
```

Figure 41 – Structure of Neural Networks

The model training of NN model stopped in 17th epoch by early stopping.

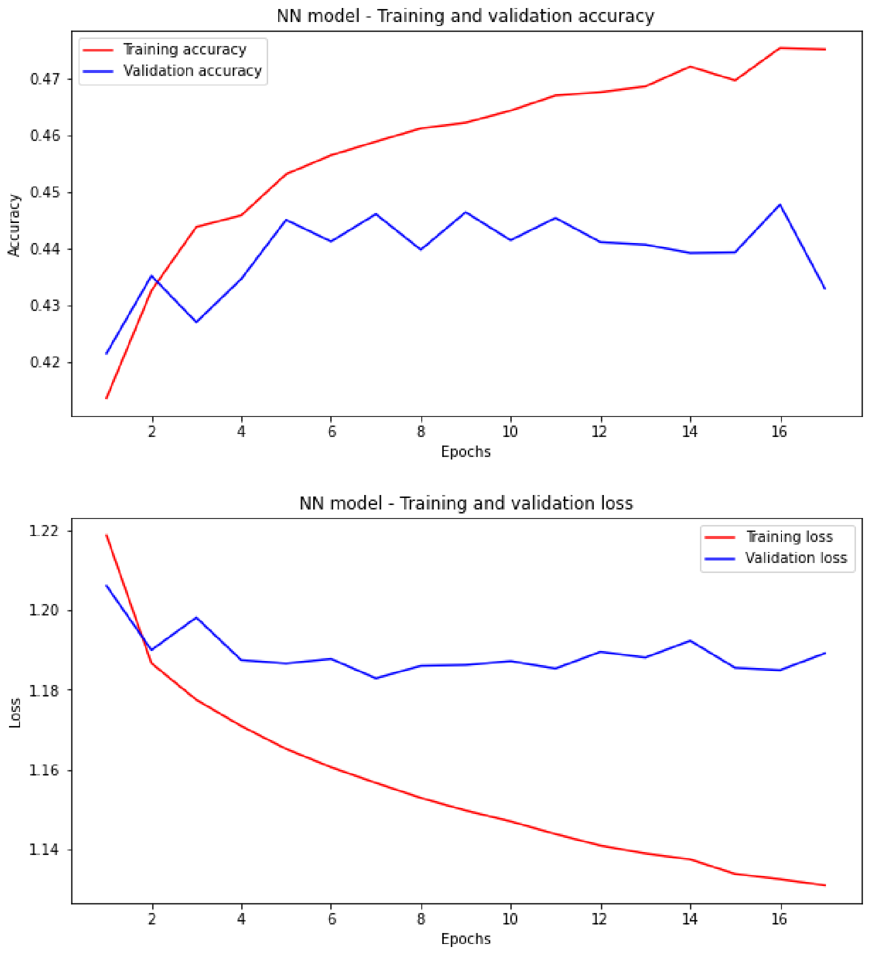
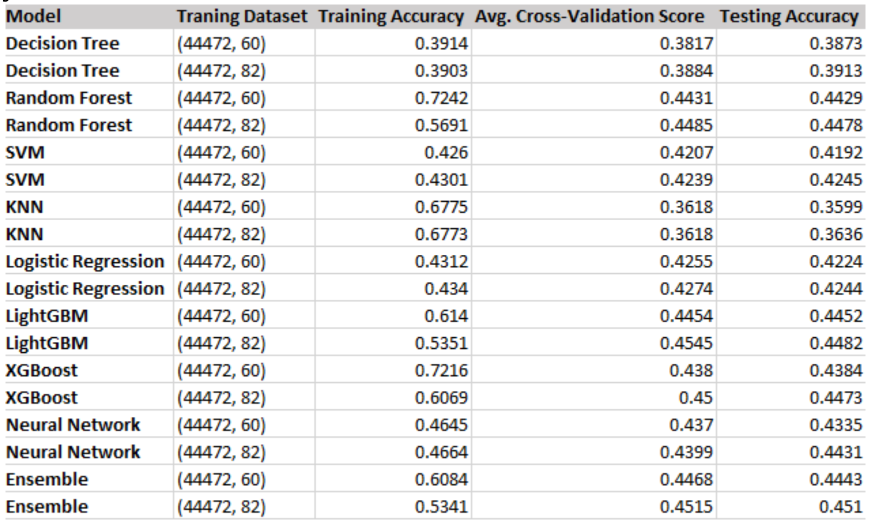


Figure 42 – Accuracy and loss in NN model training process

6.5 Accuracy of Models

Each model was trained on two datasets with different dimensions, consisting of 60 and 82 features. The metrics were reported for each model: training accuracy, average cross-validation score, and testing accuracy.



Model	Traning Dataset	Training Accuracy	Avg. Cross-Validation Score	Testing Accuracy
Decision Tree	(44472, 60)	0.3914	0.3817	0.3873
Decision Tree	(44472, 82)	0.3903	0.3884	0.3913
Random Forest	(44472, 60)	0.7242	0.4431	0.4429
Random Forest	(44472, 82)	0.5691	0.4485	0.4478
SVM	(44472, 60)	0.426	0.4207	0.4192
SVM	(44472, 82)	0.4301	0.4239	0.4245
KNN	(44472, 60)	0.6775	0.3618	0.3599
KNN	(44472, 82)	0.6773	0.3618	0.3636
Logistic Regression	(44472, 60)	0.4312	0.4255	0.4224
Logistic Regression	(44472, 82)	0.434	0.4274	0.4244
LightGBM	(44472, 60)	0.614	0.4454	0.4452
LightGBM	(44472, 82)	0.5351	0.4545	0.4482
XGBoost	(44472, 60)	0.7216	0.438	0.4384
XGBoost	(44472, 82)	0.6069	0.45	0.4473
Neural Network	(44472, 60)	0.4645	0.437	0.4335
Neural Network	(44472, 82)	0.4664	0.4399	0.4431
Ensemble	(44472, 60)	0.6084	0.4468	0.4443
Ensemble	(44472, 82)	0.5341	0.4515	0.451

Figure 43 – Accuracies of all models in table format

It was found that the cross-validation accuracy for some models was lower than the training accuracy. For instance, random forest and XGBoost model with 60 training features achieved a training accuracy of 0.72 and a cross-validation accuracy of 0.44. This discrepancy between the training and cross-validation accuracies suggested that some models may be overfitting to the training data.

Below is an overview of all accuracies.

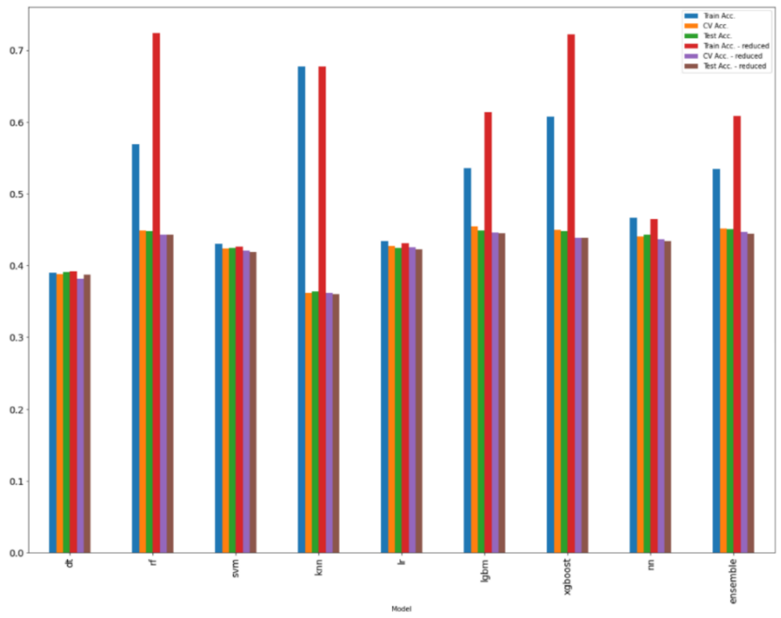
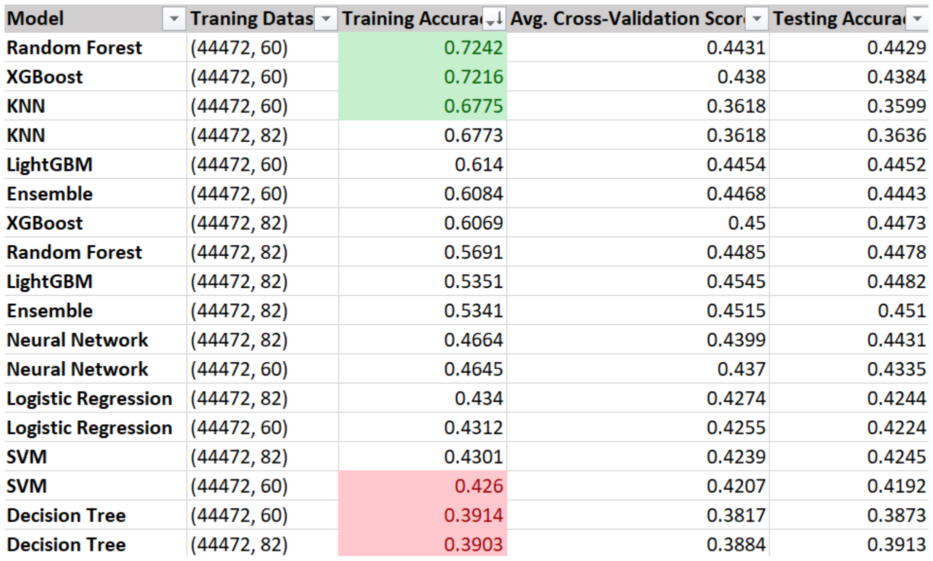


Figure 44 – Accuracies of all models in bar chart format

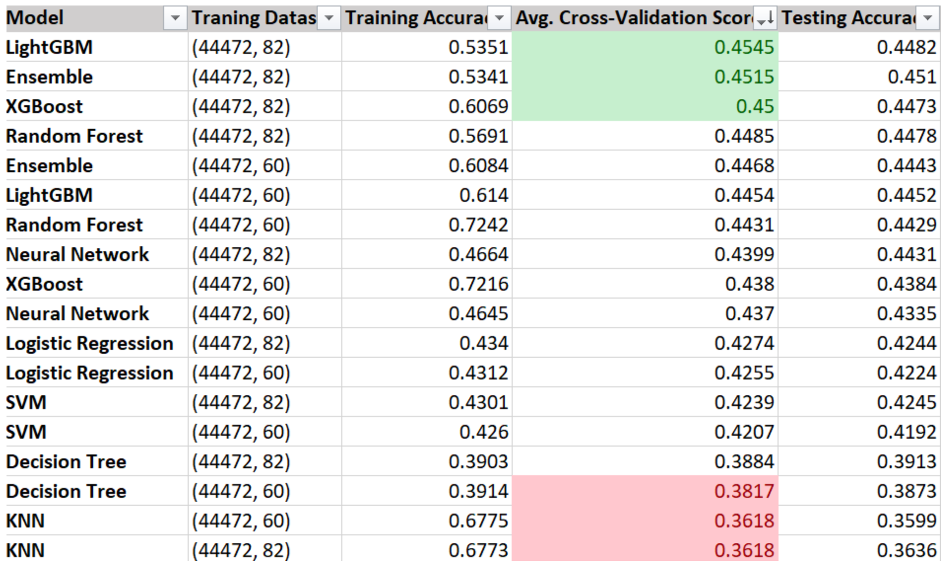
It was observed that the random forest, XGBoost, and KNN models with reduced 60 features achieved the highest training accuracy. However, it was important to note that these models did not exhibit similar testing accuracy. On the other hand, the SVM and decision tree models showed the lowest training accuracy.



Model	Traning Datas	Training Accura	Avg. Cross-Validation Scor	Testing Accura
Random Forest	(44472, 60)	0.7242	0.4431	0.4429
XGBoost	(44472, 60)	0.7216	0.438	0.4384
KNN	(44472, 60)	0.6775	0.3618	0.3599
KNN	(44472, 82)	0.6773	0.3618	0.3636
LightGBM	(44472, 60)	0.614	0.4454	0.4452
Ensemble	(44472, 60)	0.6084	0.4468	0.4443
XGBoost	(44472, 82)	0.6069	0.45	0.4473
Random Forest	(44472, 82)	0.5691	0.4485	0.4478
LightGBM	(44472, 82)	0.5351	0.4545	0.4482
Ensemble	(44472, 82)	0.5341	0.4515	0.451
Neural Network	(44472, 82)	0.4664	0.4399	0.4431
Neural Network	(44472, 60)	0.4645	0.437	0.4335
Logistic Regression	(44472, 82)	0.434	0.4274	0.4244
Logistic Regression	(44472, 60)	0.4312	0.4255	0.4224
SVM	(44472, 82)	0.4301	0.4239	0.4245
SVM	(44472, 60)	0.426	0.4207	0.4192
Decision Tree	(44472, 60)	0.3914	0.3817	0.3873
Decision Tree	(44472, 82)	0.3903	0.3884	0.3913

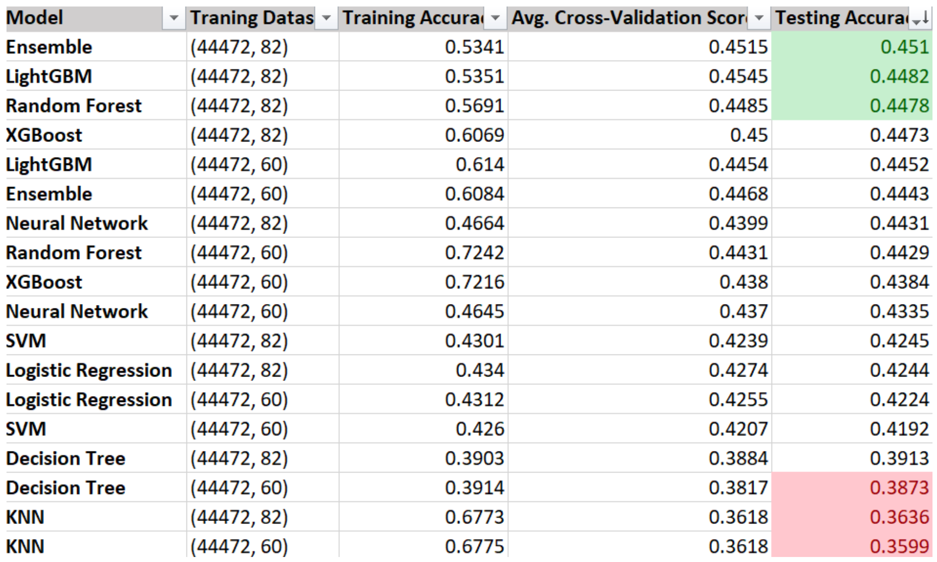
Figure 45 – Accuracies of all models sorted by training accuracy

Also, it was observed that the LightGBM, Ensemble, XGBoost, and random forest models with full 82 features achieved the highest average cross-validation score and testing accuracy. In contrast, the k-nn and decision tree models showed the lowest performance. Additionally, it was observed that some models with 82 features performed slightly better than those with reduced 60 features, as seen in the case of LightGBM, Ensemble, XGBoost, and random forest models.



Model	Traning Datas	Training Accura	Avg. Cross-Validation Scor	Testing Accura
LightGBM	(44472, 82)	0.5351	0.4545	0.4482
Ensemble	(44472, 82)	0.5341	0.4515	0.451
XGBoost	(44472, 82)	0.6069	0.45	0.4473
Random Forest	(44472, 82)	0.5691	0.4485	0.4478
Ensemble	(44472, 60)	0.6084	0.4468	0.4443
LightGBM	(44472, 60)	0.614	0.4454	0.4452
Random Forest	(44472, 60)	0.7242	0.4431	0.4429
Neural Network	(44472, 82)	0.4664	0.4399	0.4431
XGBoost	(44472, 60)	0.7216	0.438	0.4384
Neural Network	(44472, 60)	0.4645	0.437	0.4335
Logistic Regression	(44472, 82)	0.434	0.4274	0.4244
Logistic Regression	(44472, 60)	0.4312	0.4255	0.4224
SVM	(44472, 82)	0.4301	0.4239	0.4245
SVM	(44472, 60)	0.426	0.4207	0.4192
Decision Tree	(44472, 82)	0.3903	0.3884	0.3913
Decision Tree	(44472, 60)	0.3914	0.3817	0.3873
KNN	(44472, 60)	0.6775	0.3618	0.3599
KNN	(44472, 82)	0.6773	0.3618	0.3636

Figure 46 – Accuracies of all models sorted by average cross-validation score



Model	Traning Datas	Training Accura	Avg. Cross-Validation Scor	Testing Accura
Ensemble	(44472, 82)	0.5341	0.4515	0.451
LightGBM	(44472, 82)	0.5351	0.4545	0.4482
Random Forest	(44472, 82)	0.5691	0.4485	0.4478
XGBoost	(44472, 82)	0.6069	0.45	0.4473
LightGBM	(44472, 60)	0.614	0.4454	0.4452
Ensemble	(44472, 60)	0.6084	0.4468	0.4443
Neural Network	(44472, 82)	0.4664	0.4399	0.4431
Random Forest	(44472, 60)	0.7242	0.4431	0.4429
XGBoost	(44472, 60)	0.7216	0.438	0.4384
Neural Network	(44472, 60)	0.4645	0.437	0.4335
SVM	(44472, 82)	0.4301	0.4239	0.4245
Logistic Regression	(44472, 82)	0.434	0.4274	0.4244
Logistic Regression	(44472, 60)	0.4312	0.4255	0.4224
SVM	(44472, 60)	0.426	0.4207	0.4192
Decision Tree	(44472, 82)	0.3903	0.3884	0.3913
Decision Tree	(44472, 60)	0.3914	0.3817	0.3873
KNN	(44472, 82)	0.6773	0.3618	0.3636
KNN	(44472, 60)	0.6775	0.3618	0.3599

Figure 47 – Accuracies of all models sorted by testing accuracy

6.6 Confusion-matrix

From the confusion matrix, it was observed that the ensemble model exhibited better performance in predicting classes 0 and 3.

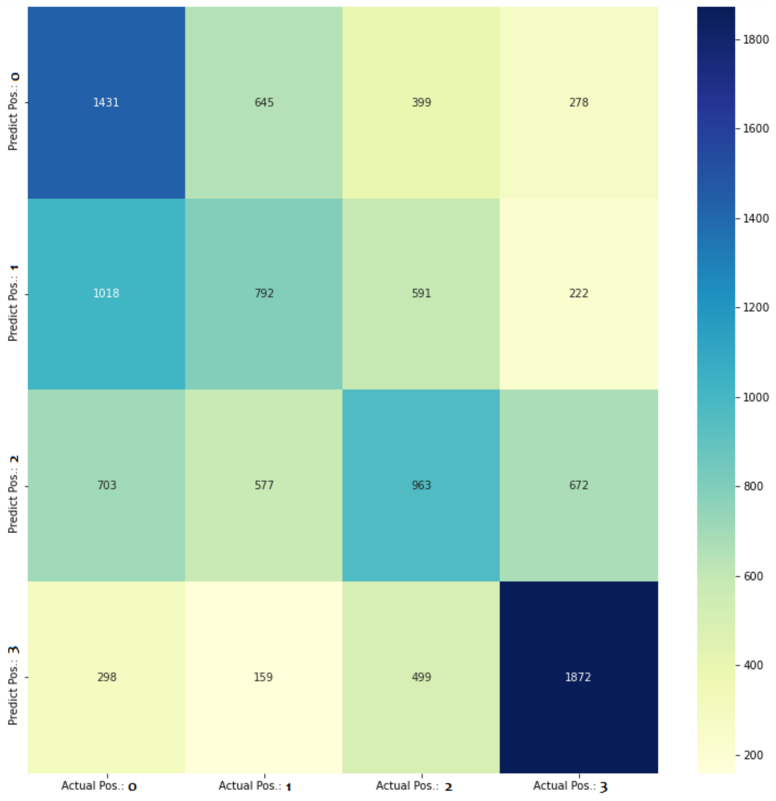


Figure 48 – Confusion matrix

ROC curves for a multi-class classification problem are shown below:

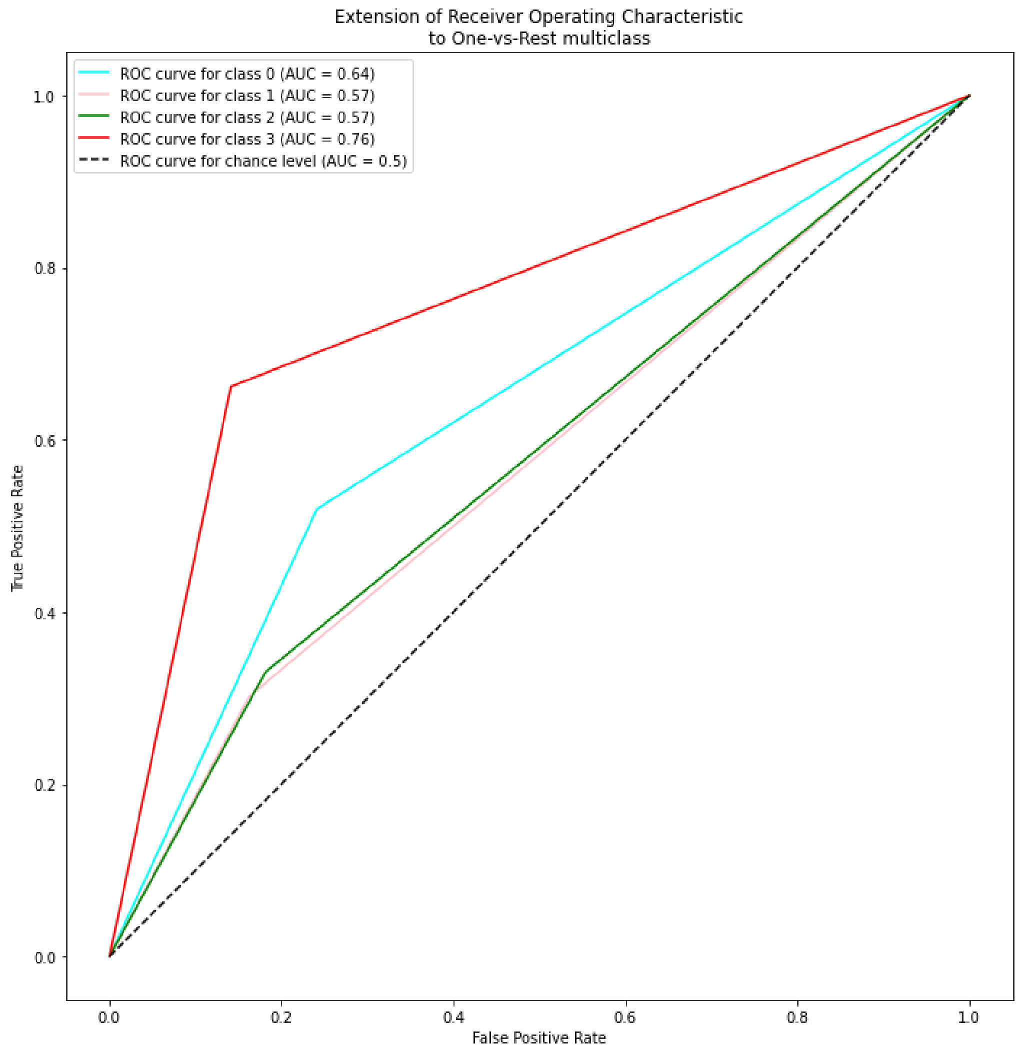
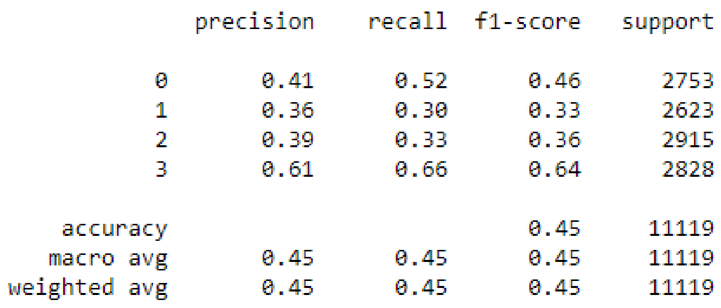


Figure 49 – ROC Curves

The final ensemble model performed better in distinguishing class 0 and class 3 from other classes, while had relatively poor performance in distinguishing class 1 and class 2 from other classes.

6.7 Classification Metrics

There was a classification report of the performance of the final ensemble model on a testing dataset.



```
              precision    recall  f1-score   support

           0       0.41      0.52      0.46      2753
           1       0.36      0.30      0.33      2623
           2       0.39      0.33      0.36      2915
           3       0.61      0.66      0.64      2828

    accuracy                           0.45     11119
   macro avg       0.45      0.45      0.45     11119
weighted avg       0.45      0.45      0.45     11119
```

Figure 50 – Classification Metrics

The accuracy on the testing dataset was 0.45. The precision of the model for each class label ranged from 0.36 to 0.61, indicating that the model's ability to correctly predict each class label varied. The recall of the model for each class label ranged from 0.30 to 0.66, indicating that the model may have missed predicting some instances for each class label.

The f1-score for each class label ranged from 0.33 to 0.64. The macro average of precision, recall, and f1-score was 0.45, indicating that the model's overall performance was average. The weighted average of precision, recall, and f1-score was also 0.45, indicating that the model performance was not influenced by the class imbalance in the testing dataset.

7. Results

The graphs showed some relationships between the popular Steam game and the corresponding feature. The result showed that indie games were more favorable than other types of games. Moreover, it was also observed that having the function of “Steam Achievement”, “Steam Cloud” and “Steam Trading Cards” in the game would be more favorable.

From the statistical model, a comparison of the different statistical models was performed. The result showed that the performance varies from different models. Among the methods used, random forest and XGBoost achieved the highest accuracy in the training data. The training accuracy for these two methods was 0.72, while the decision tree recorded the lowest training accuracy.

In general, the accuracy of the validation data and the testing data was around 0.4-0.45. In this study, it was aimed to use statistical models to predict potential user ratings and make a comparison between different methods. As such, other techniques, such as the combination of multi-models, may be required if the target is to achieve higher testing accuracy.

Nevertheless, the results of the models also gave some insight into the user ratings of the video game. Multiple models (such as decision trees, random forest, logistic regression, and LightGBM) suggested that price was a major factor in the user ratings of video games. Another price-related factor "Free to Play Genre" also showed as an important factor in some models (such as SVM and logistic regression). It may imply that the players would be satisfied with the game depending on the cost-performance ratio. As well as the quality is not too bad, players would give a positive review if the price of the game is low.

Another feature of having a positive review in a game is to participate in the scheme of "Steam Trading Cards". Participation in the scheme of Steam Trading Cards is up to the game's developers to decide. This scheme allows players to obtain some game badges. The badges are used to help the player to gain experience and levels. In other words, players prefer a game having the features of earning experience points and leveling up. This finding is aligned with the result of the previous empirical study.

The suitability of training models and their characteristics

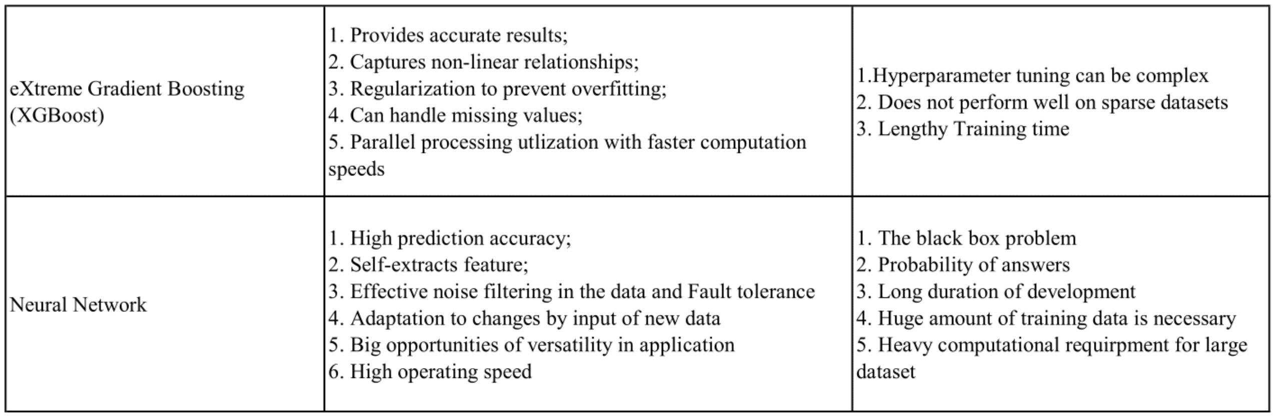
Different machine learning algorithms search for different trends and patterns. One algorithm is not the best across all data sets or for all use cases. The suitability of training models is dependent on the features of datasets, resources, and the determined goal. In this study we found that random forest and XGBoost achieved the highest accuracy. However, in other academic studies such as Osinsanwo et. al. in 2017[5], SVM and neural networks had the best performance regarding accuracy. Other researchers also have similar results[6,7]. However, the ranking of accuracy regarding SVM and neural network is relatively low in this study. In some sense, the features of the data sets may determine the suitability of the training models. Therefore, the characteristics of different training models may be considered in selection to adopt an appropriate algorithm for specific data sets. The below is a table comparing the pros and cons of the 8 used algorithms regarding classification in the study:

[5] F.Y, Osisanwo et al. "Supervised Machine Learning Algorithms: Classification and Comparison." International Journal of Computer Trends and Technology 48 (2017): 128-138.

[6] Flores, J.A., Malca, J.L., Saldarriaga, L.R., & Roman, C. (2017). Analysis and Comparison of Machine Learning Classification Models Applied to Credit Approval. *Symposium on Information Management and Big Data*.

[7] F.Y, O., AkinsolaJ.E., T., Awodele, O., HinmikaiyeJ., O., Olakanmi, O., & Akinjobi, J. (2017). Supervised Machine Learning Algorithms: Classification and Comparison. *International Journal of Computer Trends and Technology, 48,* 128-138.

Classification ML models	Pros	Cons
K-NN algorithm	1. Fairly intuitive and straightforward; 2. A non-parametric algorithm; 3. No training step and prediction formula; 4. Respond rapidly to input changes during real-time use ; 5. Adapts to multi-class situations; 6. Both for Classification and Regression Problems ; 7. One Hyper Parameter; 8. Multiple distance criteria are available for selection	1. Efficiency and performance degrade rapidly as the dataset grows; 2. Curse of Dimensionality; 3. Requires homogeneous features; 4. Optimal number of neighbors; 5. Unbalanced data poses issues; 6. Sensitivity to outliers; 7. No capability of dealing with missing value; 8. Heavy computational requirpment for large dataset
Support Vector Machine	1. Works adequately when there is a substantial difference across classes; 2. High-dimensional spaces are optimal; 3.Useful when the number of dimensions exceeds the number of samples; 4. Memory efficient memory utilization	1. Impractical for huge datasets; 2. Unable to handle overlapping classes; 3. Poor performance if additional noise is in the data set, such as overlapping target classes; 4. Poor performance if the number of features for each data point exceeds the number of training data samples
Decision Tree	1. Explainable and interpretable; 2. Can handle missing values; 3. Non-Parametric; 4. Versatility; 5. Non-Linearity;	1. Prone to overfitting; 2. Sensitive to outliers; 3. Feature Reduction & Data Resampling; 4. Optimization;
Random Forest	1. Reduces overfitting; 2. No change or scaling of variables is required; 3. An excellent model when working with a large number of data attributes; 4. Not much affected by outliers and accomplished by binning the variables; 5. High accuracy and a balanced bias-variance tradeoff	1. Training complexity can be high; 2. Not very interpretable; 3. Can be computationally costly; 4. Black box algorithm with very little control over what the model
Logistic Regression	1. One of the simplest machine learning algorithms; 2. Provision of inferences about the relevance of each feature and the direction of relationship; 3. Can be modified simply to incorporate new data with using stochastic gradient descent; 4. Production of calibrated probability in addition to classification outcomes; 5. Less susceptible to overfitting in low-dimensional datasets with a sufficient number of training samples 6. Highly effective when the dataset contains linearly separable features 7. Short training period comparing with sophisticated algorithms 8. The feasibility for extension to multi-class classification using a softmax classifier 9.The resulting weights are determined to be highly	1. Overfitting on high-dimensional datasets 2. Cannot tackle nonlinear problems 3. Hard to analyze complicated associations 4. Multicollinearity between independent variables must be minimal or moderate for logistic regression 5. Non-significant and non-pertinent characteristics in the model may result in erroneous 6. Sensitive to outliers and these may lead to inaccurate findings 7. A sizable dataset and sufficient training examples are necessary 8. Poor perform ance if the training data consist of matched or repeated measurements
Light Gradient Boosting Machine (LightGBM)	1. Faster training speed and higher efficiency; 2. Lower memory usage 3. Better accuracy than any other boosting algorithm 4. Compatibility with Large Datasets	1. Can overfit due to leaf-wise splitting and high sensitivity 2. Hyperparameter tuning can be complex



eXtreme Gradient Boosting (XGBoost)	1. Provides accurate results; 2. Captures non-linear relationships; 3. Regularization to prevent overfitting; 4. Can handle missing values; 5. Parallel processing utlization with faster computation speeds	1.Hyperparameter tuning can be complex 2. Does not perform well on sparse datasets 3. Lengthy Training time
Neural Network	1. High prediction accuracy; 2. Self-extracts feature; 3. Effective noise filtering in the data and Fault tolerance 4. Adaptation to changes by input of new data 5. Big opportunities of versatility in application 6. High operating speed	1. The black box problem 2. Probability of answers 3. Long duration of development 4. Huge amount of training data is necessary 5. Heavy computational requirpment for large dataset

Figure 51 – Pros and Cons of Models

8. Discussion

8.1 Limitation

8.1.1 Multicollinearity

This study shows that the problem of multicollinear exists in the dataset. From the heatmap diagram, it is observed that different languages highly correlate with each other. Indeed, it is expected to see this situation as the developer may often publish a game that includes multiple languages to fulfill the requirement of players in different countries. It could be easy to handle this problem by using some dimensional reduction techniques (such as PCA) or data combinations. On the other hand, it may also serve as a useful indicator to study the attitude of players in different places or the demand for video games from different places by using these indicators due to the data limitation. Therefore, it would be challenging if the study would like to incorporate this identifier into the data analysis.

8.1.2 Model Interpretation

Some results of the statistical methods used in this paper are difficult to interpret as meaningful results. Although the focus of the modeling part is comparing the performance of different modeling approaches, some methods are difficult to provide effective results in terms of interpretation. Some black box models, such as neural networks, are difficult to interpret the implication from the results. Apart from the black box models, data transformation by using PCA would also reduce the interpretation ability of the features.

8.2 Further Improvement

8.2.1 Use of “Irrelevant data”

Some data columns are dropped at the beginning based on the assumptions. As the dataset consists of a large amount of data, including text data and numeric data, some features such as Developer, Publisher, and Header Image are dropped from the dataset based on some preliminary assumptions to facilitate the data analysis. Useful information might be contained in those columns. A comprehensive study could be demonstrated if checking is performed for these assumed irrelevant data items.

8.2.2 Further data categorization

The dataset could be divided into different categories to serve further objectives and purposes. This paper aims to provide a general discussion of the video game dataset. As such, the whole dataset is used in data visualization as well as in statistical modeling. The data could also be split into multiple datasets to serve different purposes. For instance, splitting the data by the year may explain the impact on how social events (such as COVID) affect the players’ attitudes toward different game types.

9. Conclusion

In sum, graphical representation, descriptive statistics, and statistical models are presented in this paper based on the video game data retrieved in Kaggle. Before performing data analysis, data pre-processing, such as missing data handling and data transformation, is conducted to facilitate the work thereafter. Graphical representation and descriptive statistics provide some knowledge of the changes in the video game market on Steam in recent years. For the model, the accuracy comparison on predicting the user ratings for the video games of different statistical models is performed. Besides, the model also gives some insights into the features which would lead to a higher rating of the game.