predictive Analyses on commercial success of game

Submitted By

Sheikh Arafat Rahman Shovo

ID: 1915002003

Shafiul Alam

ID: 1915002033

A thesis report submitted in partial fulfillment of the requirements for

the degree of

Bachelor of Science in Computer Science Engineering from City University

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

CITY UNIVERSITY, DHAKA, BANGLADESH

FEBRUARY 2023

DECLARATION

This is to certify that the thesis titled "**predictive Analyses on commercial success of game**" is the result of our study in partial fulfillment of the B.Sc. Engineering degree under the supervision of Shaikh Shoriful Hibib, Assistant Professor, Department of Computer Science and Engineering (CSE), City University, Bangladesh. It is also hereby declared that this project or any part of it has not been submitted elsewhere for the award of any degree.

Signature of Author's

Sheikh Arafat Rahman Shovo
ID: 1915002003
B.Sc in CSE
City University, Dhaka, Bangladesh

Shafiul Alam
ID: 1915002032
B.Sc in CSE
City University, Dhaka, Bangladesh

Signature of Supervisor

Shaikh Shoriful Hibib
Designation
Dept. of Computer Science and Engineering
City University, Dhaka, Bangladesh

ACKNOWLEDGEMENT

First, we would like to express our heartfelt thanks to Almighty "ALLAH" for giving us the strength to complete this project as a partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering. We also want to express our gratitude and appreciation to our supervisor, Shaikh Shoriful Hibib, Assistant Professor, Department of Computer Science and Engineering of City University, and our former supervisor, Shahadat Hossain, Former Lecturer, Department of Computer Science and Engineering of City University, for their guidance and enthusiasm throughout the progress of this work.

Special thanks are due to our head of the department, Md Safaet Hossain, Department of Computer Science and Engineering of City University, for his valuable advice, fruitful suggestions, and coordination. We are also grateful to all the faculties of the Department of Computer Science and Engineering of City University, Bangladesh, for giving us the opportunity to complete the work and for their necessary support during the period.

This thesis work would not have been possible without the encouragement, logical help, and advice from our friends, and we are grateful to them. Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

The main goal of the thesis, titled "Predictive Analyses on the Commercial Success of Game," is to predict whether a game will be successful or not after being published by analyzing previous game sale data. Whether a game is considered a hit or not is determined by the number of game copies sold. Previous game sale data were collected from the website Kaggle ,Meta critics and Game spot. The attribute global sale was correlated with all other attributes from the dataset. This enabled the identification of trend lines and determination of the attributes that have the most impact on global sales. Prediction was carried out using Random Forest Classifier and Logistic Regression. One Hot Encoding was used as LR and RFC cannot work with string value.

Keywords: Video Game, Data Science, Data Analysis, Machine learning, Data visualization, LR, RFC, One HOT Encoding .

ABBREVIATION

LR: Logistic Regression

RFC: Random Forest Classifier

ROC: Receiver Operating Characteristic

TABLE OF CONTENTS

Name of The Content **Page**

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Introduction:

Predictive analysis on the commercial success of a game typically involves analyzing various features of the game, as well as external factors such as market trends and user demographics, to predict the likelihood of the game's success in terms of sales, user adoption, and other metrics. Some common features of a game that may be analyzed include Genre, Platform, Rating ,Publisher etc.

In addition to analyzing these features of the game, predictive analysis may also involve looking at external factors such as market trends, user demographics, and competing games to predict the game's potential success. For example, analyzing the sales and user adoption of similar games in the same genre or on the same platform can provide valuable insights into how well the game is likely to perform.

Overall, predictive analysis on the commercial success of a game can be a complex and multifaceted process that requires a deep understanding of the gaming industry and its trends, as well as advanced analytical and statistical techniques.

1.2 Problem Statement :

The gaming industry is rapidly growing, with thousands of games being released each passing year, but not all of them achieve success.

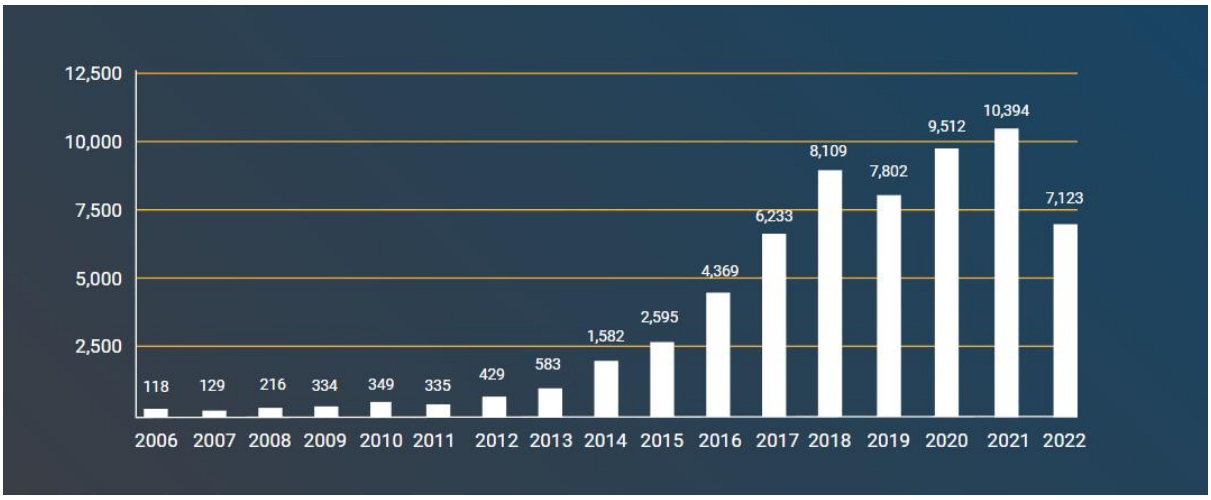


Fig 1.2.1: The number of games released from 2006 to 2022.

A study by EEDAR (Electronic Entertainment Design and Research) found that of the 7,000 games released on Steam between 2006 and 2016, only around 10% of games generated more than $50,000 in revenue. Another study by Statista found that in 2020, only 12.3% of video games released on the App Store were successful, defined as those that generated more than $1 million in revenue.

These figures suggest that achieving commercial success in the gaming industry can be challenging. The possibility of a game being a hit or flop depends on a lot of factors, and there is a lot of uncertainty involved. While developing and quality assuring a game, there can be no certainty that the game will be popular. That is why it is important to create a prediction model that can give developers an idea of the likelihood of a game being a hit. This way, they can make any necessary changes to the game before its release.

1.3 Motivation :

1. To provide publishers with a way to determine the likelihood of their game's success and inform them if any changes are needed.
2. Reduce the number of games that fail.

1.4 Objective :

1. Analyze sales data from previously released video games to identify clusters and determine industry trends.
2. Use the identified clusters to create a trend line for successful games. Develop a prediction model to forecast the probability of a game being successful.

CHAPTER 2

RELATED WORK

2.1 Related Work:

2.1.1 Thesis:

2.1.1.1 Machine Learning for Predicting Success of Video Games:

Author [1] applied machine learning techniques to predict the success of video games. It uses a dataset of over 6000 video games with various features like genre, platform, release date, and user ratings.

Contribution:

- Author [1] contributes to the field of video game development by providing a predictive model that can help game developers make informed decisions about the success of their games. It also highlights the importance of using machine learning techniques to analyze large datasets in the gaming industry.

Limitations:

- The thesis is that it only focuses on a specific dataset and may not be applicable to other datasets. The accuracy of the model may also be affected by factors that are not included in the dataset, such as marketing strategies, game quality, and player preferences. Additionally, the model may not account for the uniqueness of individual games and their success factors.

2.1.1.2 Predicting Video Game Sales in the European Market:

The authors[2] focused on game and console sales in Europe from March 12, 2005 to December 31, 2011. The authors used data about 2,450 games. The dataset contained 9 attributes and sales which they were attempting to predict. Simple regression models were fitted to predict weekly sales based on the first 2-6 weeks of sales. A prediction method for total sales was manually crafted and tested on all the data .

Contribution:

- Author[2] contributes to the understanding of the video game industry and its potential to generate revenue. By analyzing data from the European market, the thesis presents a model that predicts video game sales with a high degree of accuracy. The model is based on various factors such as platform, genre, and publisher, which allows for a more precise prediction of sales figures.

Limitations:

- limitation of the thesis is its focus on the European market, which may not be representative of the global market. Another limitation is that the model does not take into account external factors such as economic trends, social changes, or technological advancements, which could impact video game sales. Additionally, the thesis uses data from a specific time period (2008-2011), which may not be applicable to current market trends. Finally, the thesis does not address the impact of digital distribution on video game sales, which is becoming an increasingly important aspect of the industry.

2.1.1.3 Predicting Video Game Sales Using an Analysis of Internet Message Board Discussions:

The authors[3] aim of this thesis was to collect gaming forum posts and use this data to predict sales of video games. The data was collected from 2008 and 2009 from a major gaming message board. The author extracted mentions of each game and used the number of these mentions as well as sales from previous two weeks to predict sales in the upcoming weeks. The only evaluation metric used is Mean Absolute Error, making any conclusion of the results difficult.

Contribution:

- Author[3] contributes to the field of video game sales prediction by using a unique approach - analyzing internet message board discussions. This method offers an alternative way of analyzing consumer behavior and identifying market trends that may not be captured by traditional sales data. The study also demonstrates the potential of natural language processing techniques in predicting video game sales. The findings of this thesis may be useful to video game companies and market analysts looking for new ways to predict consumer behavior.

Limitations:

- limitation of this thesis is its reliance on internet message board discussions as the primary source of data. While message boards are a valuable source of information, they may not necessarily represent the entire consumer population or capture the opinions of non-online users. Additionally, message board discussions may be biased towards a specific demographic or genre of video games, which may limit the generalizability of the findings. Furthermore, the thesis does not account for other external factors that may influence video game sales, such as marketing campaigns, industry events, and economic factors. Overall, while this study provides valuable insights into video game sales prediction, it should be viewed as a complementary approach to existing methods rather than a replacement.

2.1.1.4 The Game Prophet: Predicting the success of Video Games:

The authors[4] proposes a predictive model, called the Game Prophet, to forecast the commercial success of video games using data analytics and machine learning techniques.

Contribution:

- Author[4] contributes to the field of video game development by offering a data-driven approach to predict the market demand and revenue potential of new video games. The Game Prophet model can assist game developers and publishers in making informed decisions about game design, marketing, and pricing strategies.

Limitations:

- Model's accuracy and applicability may depend on the quality and quantity of input data, which can be limited or biased. Moreover, the model's predictions may not account for unexpected market changes or subjective factors, such as user preferences and reviews, which can influence game success.

2.1.2 Conference Papers:

IEEE Conference on Computational Intelligence and Games1 stands out as a prominent source of papers dealing with applying machine learning methods in video games development. The most common topics include automatic content generation and agent planning. While the topic of predicting revenue is not present, there are papers utilizing machine learning methods to predict factors related to games' success, such as retention or player experience.

2.1.2.1 Predicting Retention in Sandbox Games with Tensor Factorization-based Representation Learning:

The authors[5] processed data about at what times players were playing and what they were doing within the game. The goal was to learn from 14 days of activity and predict if the players keep playing after the following 7 days. The study is heavily focused on spatio-temporal data, i.e. how players travel within the game and how to process this data. Ensemble methods were mostly used for evaluation, achieving precision of 81 % and recall of 75 % in the best case.

Contribution:

- The paper presents a machine learning approach for predicting player retention in sandbox games based on spatio-temporal data. The authors processed data on player activity within the game, including when players were playing and what they were doing. The study highlights the importance of spatio-temporal data in predicting player retention and provides insights into how this data can be processed for machine learning analysis. The authors used ensemble methods for evaluation and achieved high precision and recall rates in predicting player retention.

Limitations:

- While the study provides valuable insights into predicting player retention in sandbox games, there are some limitations to consider. Firstly, the study was conducted on a specific game, and the results may not be generalizable to other games or datasets. Secondly, the study only considers a 7-day retention period, which may not be sufficient for longer-term predictions. Additionally, the paper does not address the potential ethical implications of using player data for machine learning analysis, such as issues related to data privacy and consent. Therefore, future research should aim to address these limitations and provide more comprehensive guidelines for the use of machine learning in predicting player retention in video games.

2.1.2.2 Predicting Player Churn in Destiny: A Hidden Markov Models Approach to Predicting Player Departure in a Major Online Game:

The authors[6] used very detailed data about player activities in a major title, Destiny. The data span across 17 months and included the activities of 10,000 players. Similarly to the previous study, the goal was to predict whether a player quits the game after a certain time window, in this case 4 weeks. They focused on the use of multinomial Hidden Markov Model which returned the highest precision of 92 % with a relatively low recall of 43 % compared to other models

Contribution:

- The contribution of this paper is the application of a Hidden Markov Model approach to predict player churn in a major online game, which can have practical applications in game development and player retention strategies. The paper also showcases the usefulness of detailed player activity data in predicting player behavior.

Limitations:

- limitation of this study is that the dataset is limited to a single game, and the results may not generalize to other games. The study also did not consider external factors that may influence player churn, such as changes in the game's environment or the release of competing games.

2.1.2.3 Predicting Player Churn in Destiny: A Hidden Markov Models Approach to Predicting Player Departure in a Major Online Game:

The authors[7] describes learning of how players experience one game and making predictions about their experience in another game. The authors used statistical summarization of what players were doing and how well they were performing in two games. Players were then asked about their experience, namely engagement, frustration and challenge. The authors used two methods for the task of automatically mapping features between games, referred to as ”supervised feature mapping” and ”unsupervised transfer learning”. Both methods produced accuracies above 58 % and 55 %, respectively, achieving 83 % accuracy on one of the subtasks (predicting challenge). These results were comparable with manual mappings created by experts.

Contribution:

- The major contribution of this paper is the use of transfer learning to address the problem of limited data availability in predicting player experience across games. The TLSS method allows for leveraging knowledge learned from multiple games to improve the prediction performance in a new game. The paper also provides a detailed analysis of the proposed method and compares it with other state-of-the-art methods, showing its superior performance.

Limitations:

- limitation of the paper is that it only considers a limited number of games, and it is unclear how well the proposed approach would generalize to a larger and more diverse set of games. Additionally, the paper assumes that player experience can be measured and compared across different games, which may not always be the case. Overall, the paper presents a promising approach for predicting player experience across games and highlights the potential of transfer learning in this domain.

CHAPTER 3

METERIALS & METHODS

3.1 Methodology:

Step 1: Two game sale datasets were collected from Kaggle, and one was manually collected from Metacritic and Gamespot.

Step 2: These datasets were merged

Step 3: The most influential factors on global sale were identified by relating other attributes with global sale

Step 4: The rest of columns were dropped

Step 5: Any remaining row with null values were also dropped.

Step 6: String variables were represented as binary vectors by applying One Hot Encoding .

Step 7: The dataset was then split into train and test data

Step 8: Prediction models were created using the LR and RFC algorithms

Step 9: The models were trained using the train data, and their accuracy was tested using the test data.

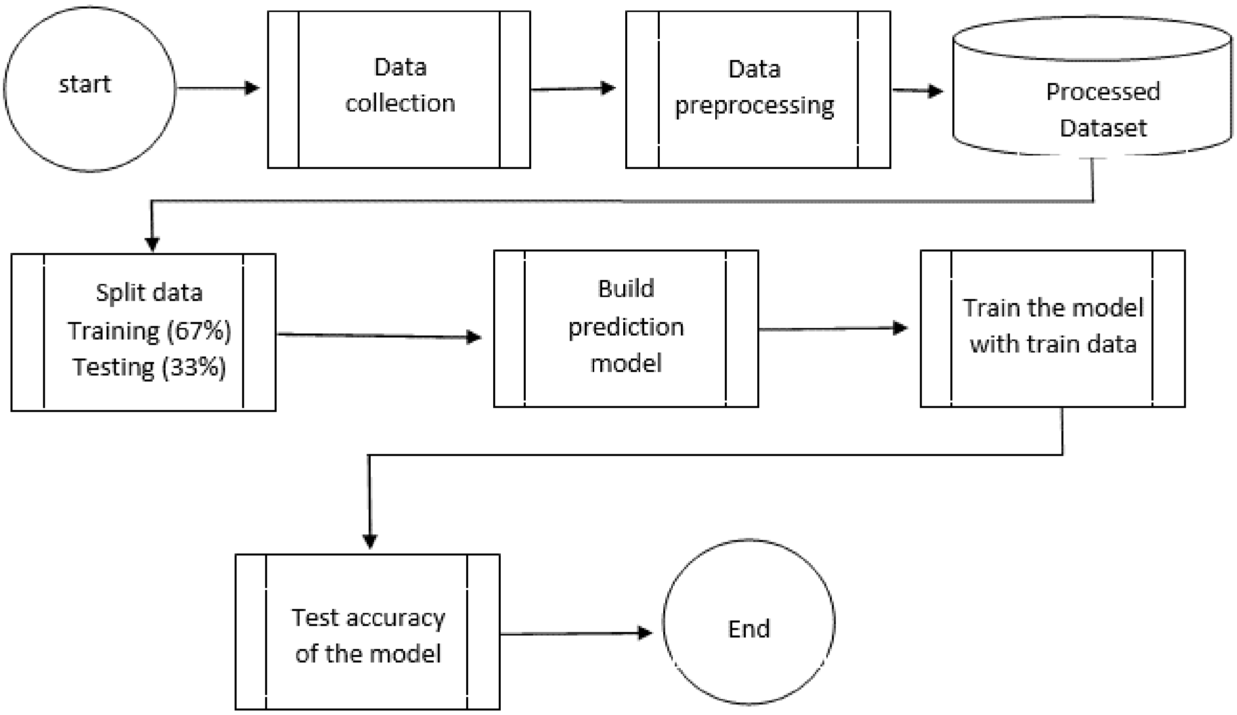


Fig 3.1.1: Workflow diagram

3.2 Materials:

3.2.1 USED Algorithms:

3.2.1.1 Logistic Regression:

Logistic Regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. It is a type of regression analysis used to predict the outcome of a categorical dependent variable based on one or more predictor variables.

The dependent variable in logistic regression is binary, meaning it can take one of two possible values, usually represented as 0 or 1. For example, in a medical study, the dependent variable might be whether a patient has a particular disease or not, where 0 represents "no disease" and 1 represents "disease."

The independent variables, also known as predictor variables, can be either continuous or categorical. They are used to explain the variation in the dependent variable.

The logistic regression model uses the logistic function to model the relationship between the independent variables and the probability of the dependent variable taking on the value of 1. The logistic function is an S-shaped curve that maps any real-valued number to a value between 0 and 1. The logistic function is defined as:

p(x) = 1 / (1 + e^{-z})

where p(x) is the probability of the dependent variable taking on the value of 1, e is the mathematical constant approximately equal to 2.71828, and z is the linear combination of the independent variables.

The logistic regression model estimates the parameters of the logistic function using a maximum likelihood estimation method. The resulting model can then be used to predict the probability of the dependent variable taking on the value of 1 for a given set of values of the independent variables.

Logistic regression can be extended to handle multiple independent variables and interactions between them. It can also be used for multi-class classification problems, where the dependent variable can take on more than two possible values.

Logistic regression is widely used in many fields, including medicine, social sciences, marketing, and finance, to predict the likelihood of an event occurring based on a set of predictor variables.

3.2.1.2 Random Forest Classifier :

Random forest classifier is a machine learning algorithm that is used for classification and regression tasks. It is a type of ensemble learning algorithm that combines multiple decision trees to make predictions. The algorithm is based on the concept of bagging, where a subset of data is selected randomly and a decision tree is trained on that subset. This process is repeated multiple times, and the predictions from each tree are combined to make the final prediction.

Some key features of the random forest classifier are:

- **Decision tree-based:** Random forest classifier is based on decision trees, which are used to make predictions by learning from data.

- **Ensemble learning:** The algorithm combines multiple decision trees to make predictions, which helps to reduce the risk of overfitting and improve accuracy.

- **Randomness:** The algorithm introduces randomness in the selection of data subsets and features to use for each decision tree, which helps to improve diversity among the trees and reduce correlation.

- **High accuracy:** Random forest classifier is known for its high accuracy in predicting outcomes, especially in complex problems where there are many features and interactions between them.

- **Robustness:** The algorithm is robust to missing values, outliers, and noisy data, making it a suitable choice for real-world problems.

Overall, random forest classifier is a powerful machine learning algorithm that can be used for a wide range of classification and regression tasks. Its ability to combine multiple decision trees and introduce randomness makes it a popular choice among data scientists and machine learning practitioners.

3.2.1.3 One Hot Encoding :

One hot encoding is a technique used in data processing and machine learning to convert categorical data into numerical data that can be used in algorithms. It is a binary representation of categorical data, where each category is represented by a vector of 0s and 1s.

The process of one hot encoding involves the following steps:

- Identify the categorical variables: Identify the categorical variables in the dataset that need to be encoded.

- Create a unique integer for each category: Assign a unique integer value to each category of the variable.

- Create a binary vector for each category: For each category, create a binary vector with a length equal to the number of unique categories. The vector should contain a 1 in the position corresponding to the unique integer assigned to the category, and 0s in all other positions.

- Concatenate the binary vectors: Concatenate the binary vectors for all categories to create a final one-hot encoded matrix.

For example, consider a categorical variable "fruit" with three categories: apple, banana, and orange. One hot encoding would create a binary vector for each category as follows:

Apple: [1, 0, 0]
Banana: [0, 1, 0]
Orange: [0, 0, 1]

The final one-hot encoded matrix would look like this:

[1, 0, 0, 0, 1, 0, 0, 0, 1]

One hot encoding is commonly used in machine learning algorithms that require numerical data, such as logistic regression, decision trees, and neural networks. It is also useful in data visualization and analysis, as it can help identify patterns and relationships in categorical data.

3.2.2 USED Language:

Python:

Python is a high-level, interpreted, and object-oriented programming language. It is designed to be easy to read and write with clean syntax and has gained widespread popularity due to its simplicity and versatility.

Here are some of the key features and characteristics of Python:

- **Simple and easy to learn:** Python has a clean syntax and is easy to read and write, making it an ideal language for beginners.

- **High-level language:** Python is a high-level programming language that provides a large library of built-in functions and modules that can be used to build complex applications.

- **Interpreted language:** Python

Python is structured in a simple way that makes it easy to understand and write code. It uses indentation to show the level of hierarchy in the code, and does not use curly braces or semicolons like other programming languages. Python code is divided into modules, packages, functions, and classes.

- **Modules:** A module is a file that contains Python code. It can contain functions, classes, and variables that can be used in other Python programs.

- **Packages:** A package is a collection of related modules. It allows for easy organization and sharing of Python code.

- **Functions:** Functions are a block of code that performs a specific task. They can take input parameters and return output values.

- **Classes:** Classes are a blueprint for creating objects in Python. They contain attributes (data) and methods (functions) that define the behavior of the object.

Python uses an interpreter to run code, meaning that it executes code line by line. The interpreter converts the source code into bytecode, which is then executed by the Python Virtual Machine (PVM). The PVM is responsible for managing memory, executing instructions, and handling exceptions.

Python supports dynamic typing, meaning that variables do not need to be declared before they are used, and their data type can change during runtime. Python also has automatic garbage collection, which means that it automatically frees up memory that is no longer being used.

In summary, Python's simple and organized structure, combined with its interpreted runtime, makes it a popular and easy-to-use programming language for a wide variety of applications.

CHAPTER 4

DATA

4.1 Dataset Description:

Video Game Sales with Ratings 2.0[8] contains 17417 game sale data (1.36Mb). PC Game Sales [9] contains 176 game sale data(15Kb). These datasets were collected from the Kaggle website. Dataset[10] is collected manually from critics score and game spot contains 108 game sale data(5Kb).

4.2 Data exploration And analysis:

4.2.1 Median sales (in millions of units) vs. critic scores:

The following three heatmaps show how game sales vary according to critic scores, which are split into six scoring groups. Additionally, each heatmap segments the data further by one of the following features: genre, ESRB rating, publisher.

Under each heatmap, we identify the categories where games sell best. This is done for okay, good, and great games, as defined by games with scores in the 70s, 80s, and 90s, respectively.

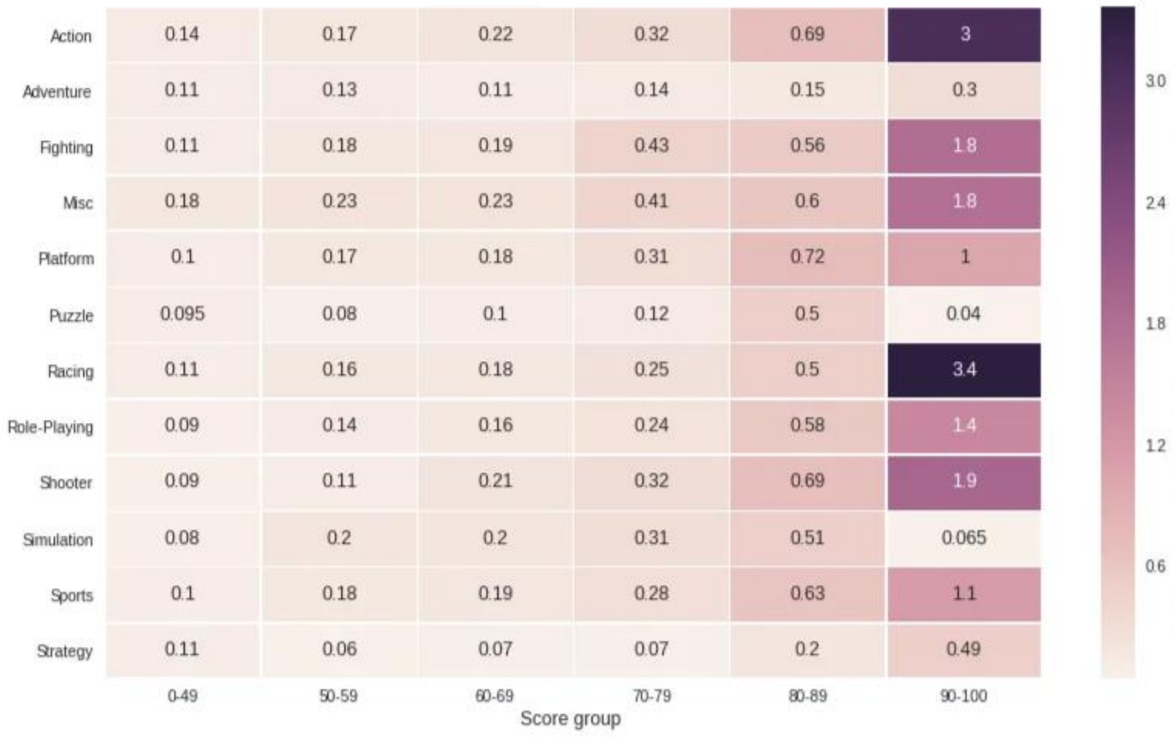


Fig 4.2.1.1: Genre vs critics Score by median sale

- Genres where great games sell best: Racing, Action
- Genres where good games sell best: Action, Shooter
- Genres where okay games sell best: Fighting

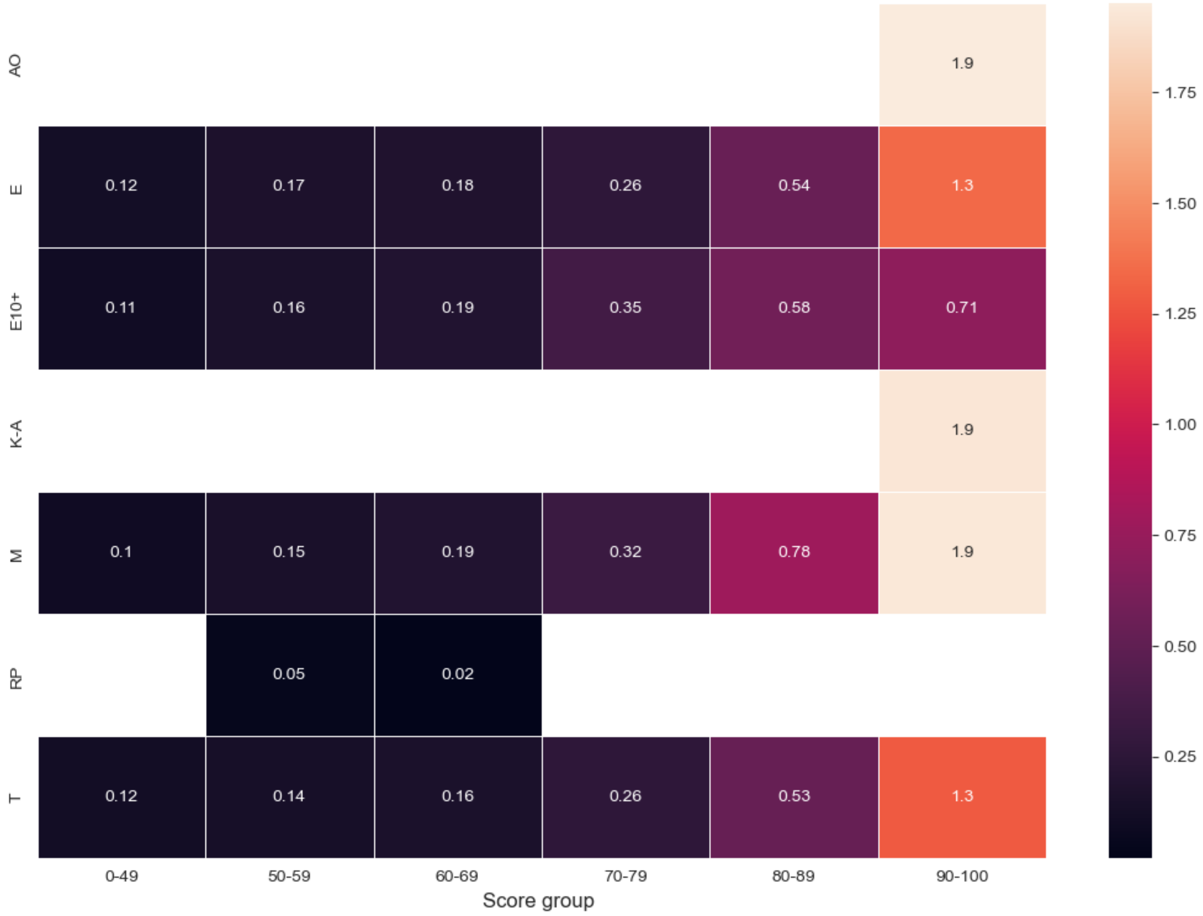


Fig 4.2.1.2: ESRB_RAting vs critics Score by median sale

- ESRB_RAting where great games sell best: AO, M,K-A
- ESRB_RAting where good games sell best: E,T
- ESRB_RAting where okay games sell best: E10+

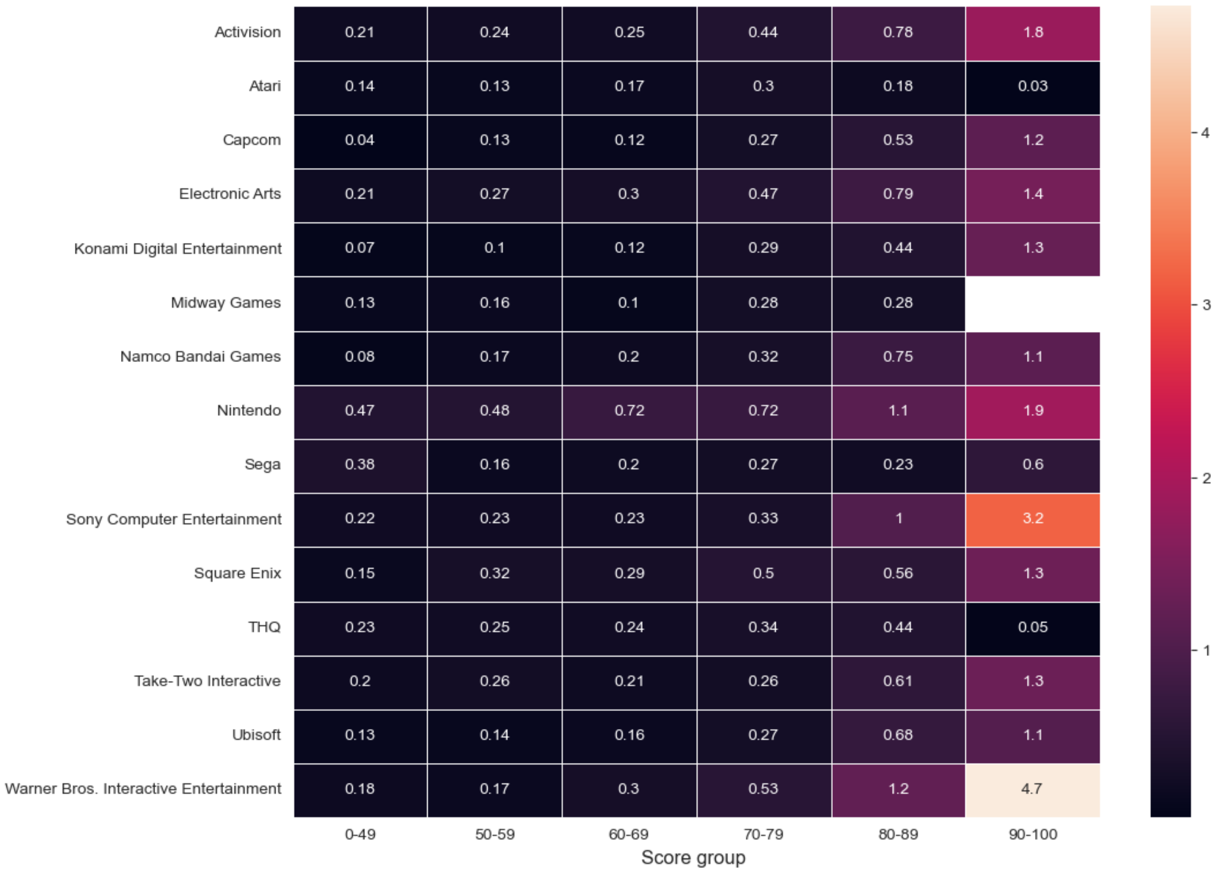


Fig 4.2.1.3: Publisher vs critics Score by median sale

- Publisher where great games sell best: Warner Bros Interactive Entertainment
- Publisher where good games sell best: Sony Computer Entertainment, Nintendo
- Publisher where okay games sell best: Electronic Art

4.2.2 Top values in the dataset

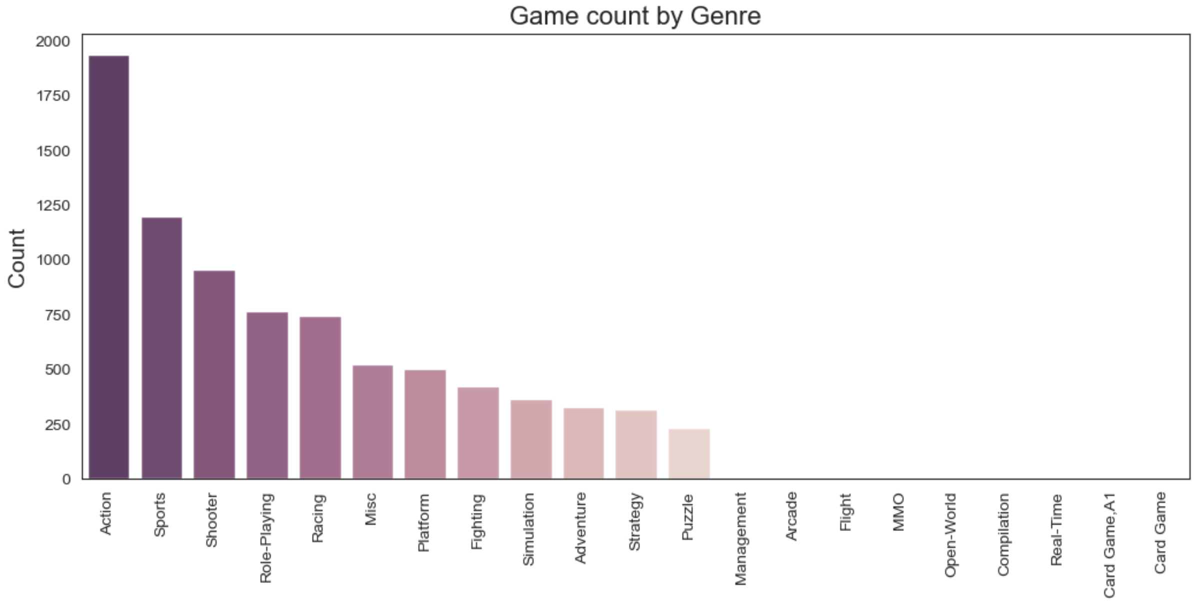


Fig 4.2.2.1: Game count by Genre

Genres with most games in dataset:

- Action
- Sports

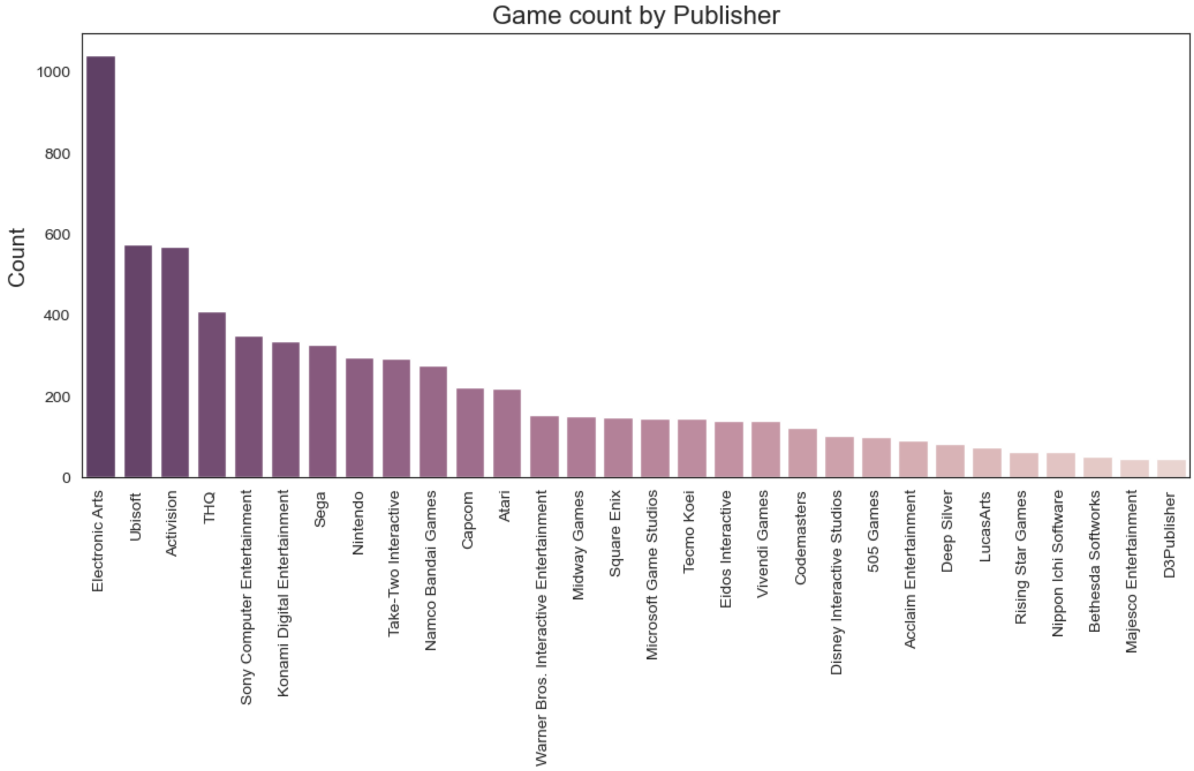


Fig 4.2.2.2: Game count by Publisher

Publishers with most games in dataset:

- Electronic Art
- Ubisoft
- Activision

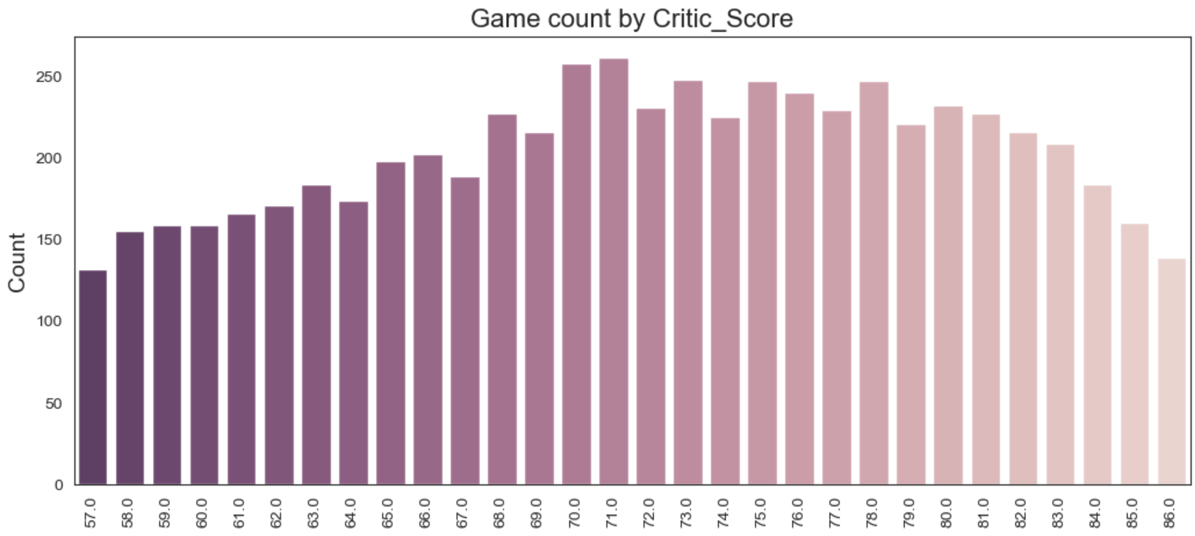


Fig 4.2.2.3: Game count by Critic Score

Critic Scores with most games in dataset are between 65 to 78

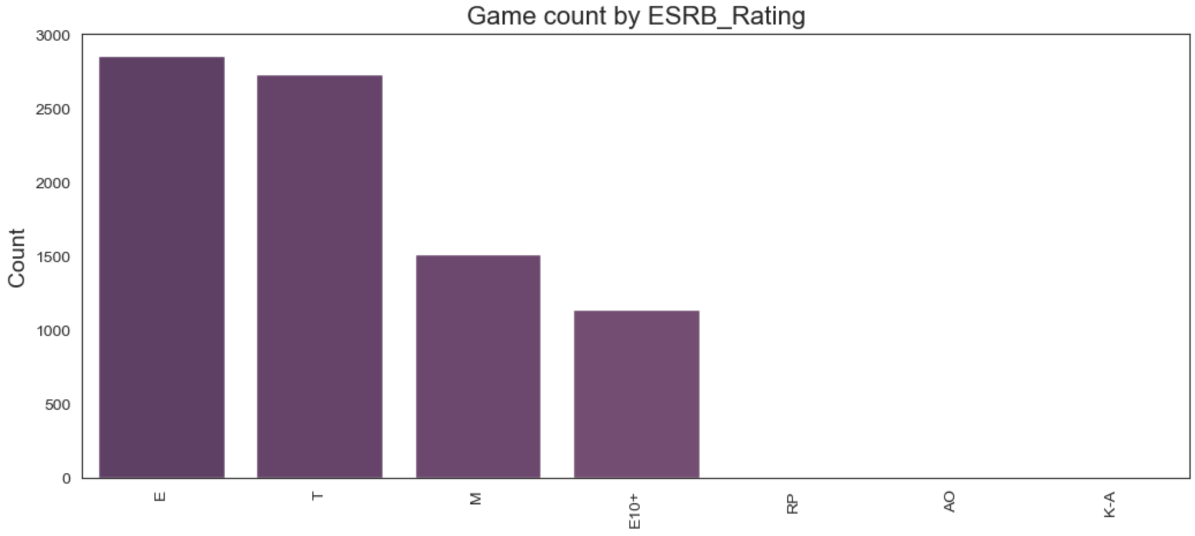


Fig 4.2.2.4: Game count by ESRB Rating

ESRB Ratings with most games in dataset are :

- E
- T

4.2.3 Dataset correlations

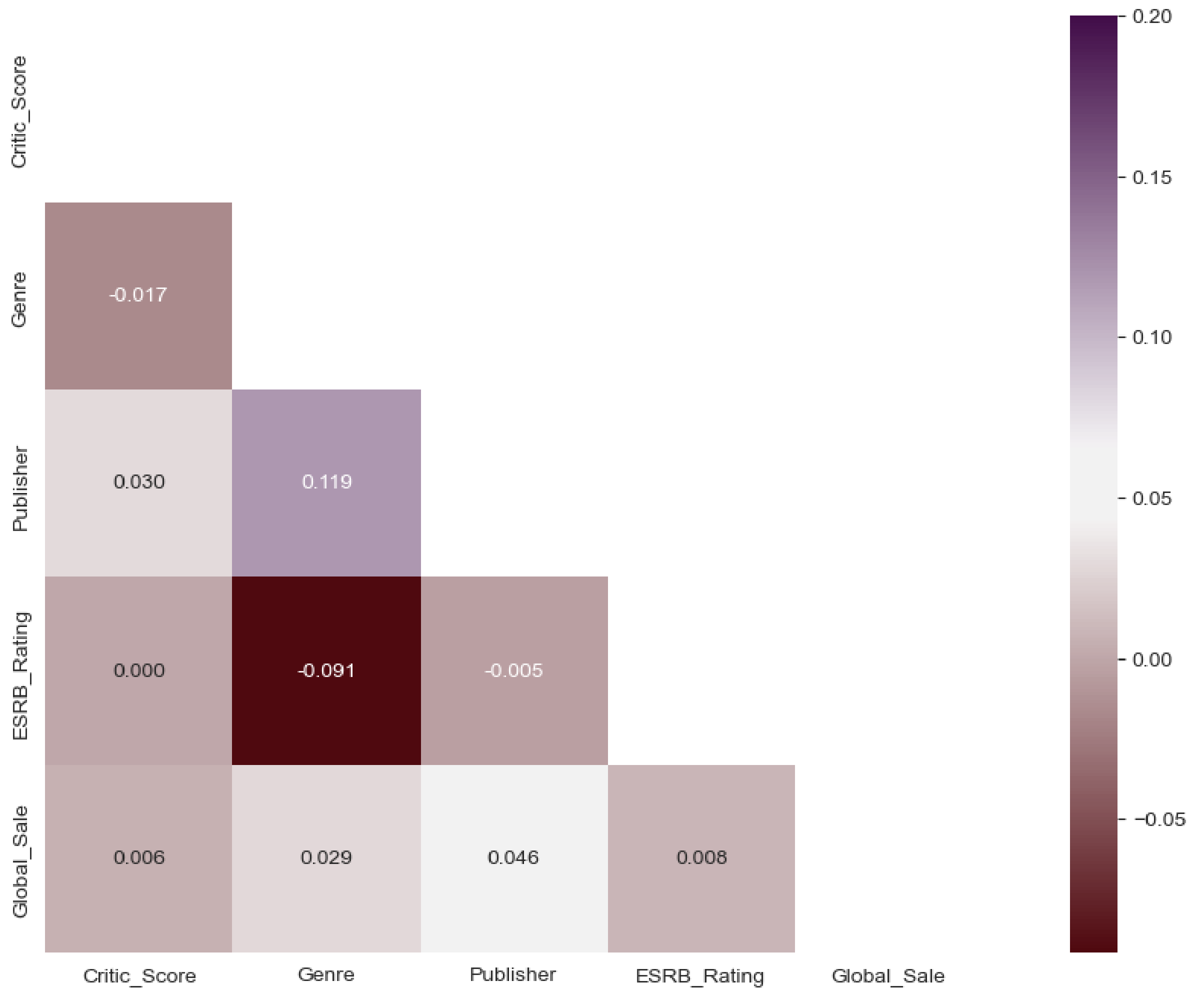


Fig 4.2.3.1: Dataset correlations for numeric and categorical variables

Strongest correlations:

- global sales to Publisher and ESRB rating
- Publisher to Genre

Critic scores to global sales

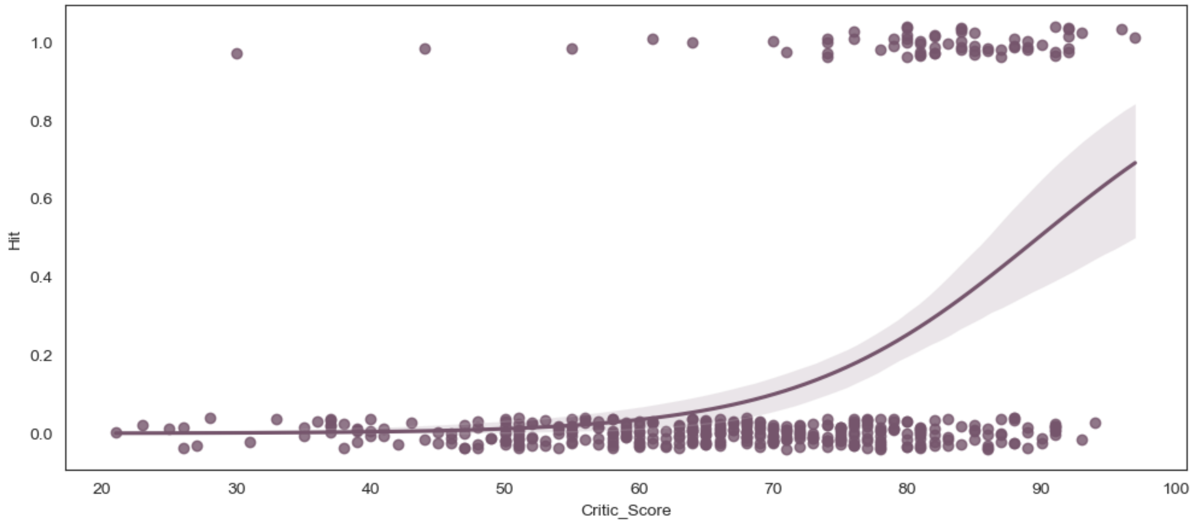


Fig 4.2.3.2: Critic scores vs global sales after defining hits as those with sales above 1 million units

As expected Critics Score , ESRB rating , publisher, Genre are important feature for global sale and we will take year of release science each years trend impacts global sale

CHAPTER 5

Implementation and Report

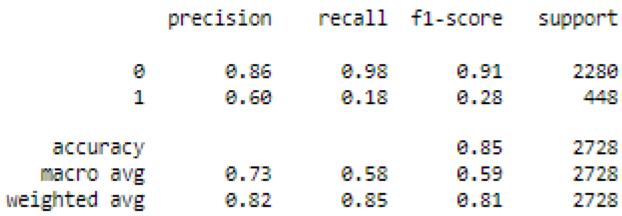
5.1 Logistic Regression:
5.1.1 implementaion

```
import pandas as pd
from pandas import get_dummies
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import
classification_report,confusion_matrix,roc_curve, roc_auc_score

###Reading Data
df = pd.read_csv('xceldata.csv', encoding="utf-8")

###  Removing null value
df = df.dropna(axis=0)

###  Declaring Hit
dfb =
df[['Name','ESRB_Rating','Genre','Publisher','Year_of_Release','Critic_Scor
e','Global_Sale']]
dfb = dfb.dropna().reset_index(drop=True)
df2 =
dfb[['ESRB_Rating','Genre','Publisher','Year_of_Release','Critic_Score','Gl
obal_Sale']]
df2['Hit'] = df2['Global_Sale']
df2.drop('Global_Sale', axis=1, inplace=True)
def hit(sales):
    if sales >= 1:
        return 1
    else:
        return 0

df2['Hit'] = df2['Hit'].apply(lambda x: hit(x))

###Generating Feature
df3=df2['Genre'].str.get_dummies(sep = ',')
df4=pd.concat([df2,df3], axis=1)
df4.drop('Genre', axis=1, inplace=True)
df5 = pd.get_dummies(df4)
df5.columns

###Spliting data set to train  and test  data
y = df5['Hit'].values
df5 = df5.drop(['Hit'],axis=1)
X = df5.values
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size=0.33,
random_state=2)
```

```
###testing prediction accuracy with test data
log_reg = LogisticRegression(max_iter=3000).fit(Xtrain, ytrain)
y_val_2 = log_reg.predict_proba(Xtest)
print("Logistic Regression Validation accuracy: ",
sum(pd.DataFrame(y_val_2).idxmax(axis=1).values==ytest)/len(ytest))
```

```
###testing if game  will  be success or not(game info is in test.csv)
new_data = pd.read_csv('test.csv', encoding="utf-8")
new_data
new_data =
new_data[['ESRB_Rating','Genre','Publisher','Year_of_Release','Critic_Score
']]
new= new_data['Genre'].str.get_dummies(sep = ',')
new_data = pd.concat([new_data,new], axis=1)
new_data.drop('Genre', axis=1, inplace=True)
new_data = pd.get_dummies(new_data)
new_data = new_data.reindex(columns=df5.columns, fill_value=0)

y_pred = log_reg.predict_proba(new_data.values)
y_pred
```

5.1.2 Classification Report:



```
              precision    recall  f1-score   support

           0       0.86      0.98      0.91      2280
           1       0.60      0.18      0.28       448

    accuracy                           0.85      2728
   macro avg       0.73      0.58      0.59      2728
weighted avg       0.82      0.85      0.81      2728
```

Table 5.1.2.1: Accuracy Report of LR

Model made by Logistic Regression has prediction accuracy of 85% and loss of 35%.

5.1.3 Confusion matrix:

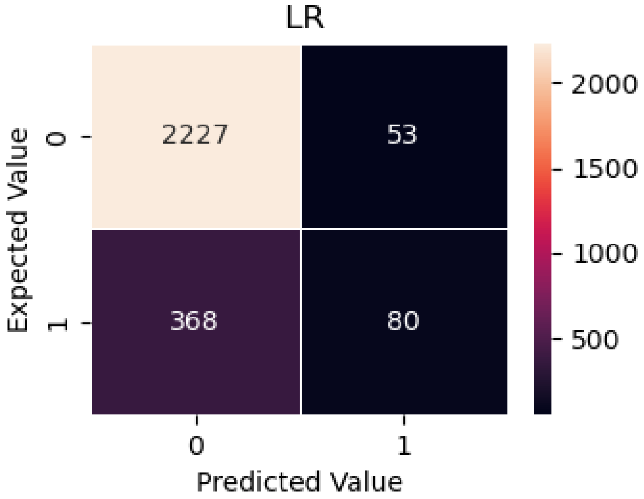


Fig 5.1.3.1: Confusion Matrix of LR

For LR :
The number of game will be hit predicted successfully(TP)=80
The number of game will be hit predicted unsuccessfully(FP)=53
The number of game will not be hit predicted successfully(TN)=2227
The number of game will not be hit predicted unsuccessfully(FN)=368

5.1.4 ROC Curve:

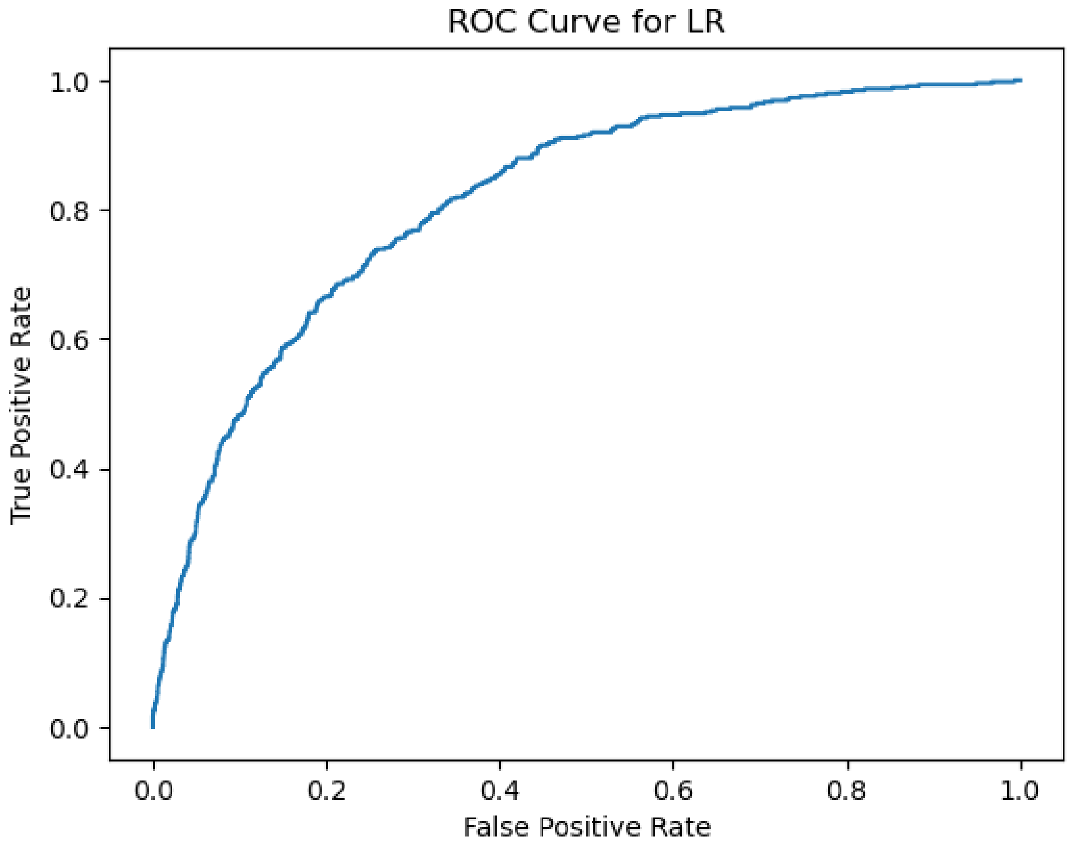


Fig 5.1.4.1: ROC curve of LR

For logistic regression True positive rate is higher than False positive rate

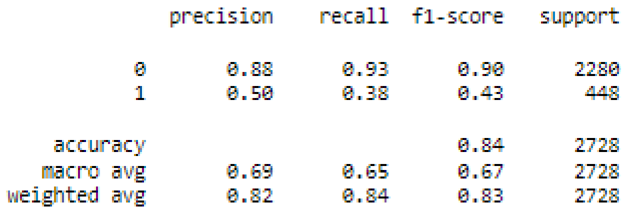
5.2 Random Forest Classifier:

5.2.1 implementation

```
import pandas as pd
from pandas import get_dummies
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import
classification_report,confusion_matrix,roc_curve, roc_auc_score

###Reading Data
df = pd.read_csv('xceldata.csv', encoding="utf-8")

###  Removing null value
df = df.dropna(axis=0)

###  Declaring Hit
dfb =
df[['Name','ESRB_Rating','Genre','Publisher','Year_of_Release','Critic_Scor
e','Global_Sale']]
```

```
dfb = dfb.dropna().reset_index(drop=True)
df2 =
dfb[['ESRB_Rating','Genre','Publisher','Year_of_Release','Critic_Score','Gl
obal_Sale']]
df2['Hit'] = df2['Global_Sale']
df2.drop('Global_Sale', axis=1, inplace=True)
def hit(sales):
    if sales >= 1:
        return 1
    else:
        return 0

df2['Hit'] = df2['Hit'].apply(lambda x: hit(x))

###Generating Feature
df3=df2['Genre'].str.get_dummies(sep = ',')
df4=pd.concat([df2,df3], axis=1)
df4.drop('Genre', axis=1, inplace=True)
df5 = pd.get_dummies(df4)
df5.columns

###Spliting data set to train  and test  data
y = df5['Hit'].values
df5 = df5.drop(['Hit'],axis=1)
X = df5.values
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size=0.33,
random_state=2)


###testing prediction accuracy with test data
radm = RandomForestClassifier(random_state=2).fit(Xtrain, ytrain)
y_val_1 = radm.predict_proba(Xtest)
print("Random Forest Validation accuracy: ",
sum(pd.DataFrame(y_val_1).idxmax(axis=1).values
                                   == ytest)/len(ytest))


###testing if game  will  be success or not(game info is in test.csv)
new_data = pd.read_csv('test.csv', encoding="utf-8")
new_data
new_data =
new_data[['ESRB_Rating','Genre','Publisher','Year_of_Release','Critic_Score
']]
new= new_data['Genre'].str.get_dummies(sep = ',')
new_data = pd.concat([new_data,new], axis=1)
new_data.drop('Genre', axis=1, inplace=True)
new_data = pd.get_dummies(new_data)
new_data = new_data.reindex(columns=df5.columns, fill_value=0)

y_pred = radm.predict_proba(new_data.values)
y_pred
```

5.2.2 Classification Report:



```
              precision    recall  f1-score   support

           0       0.88      0.93      0.90      2280
           1       0.50      0.38      0.43       448

    accuracy                           0.84      2728
   macro avg       0.69      0.65      0.67      2728
weighted avg       0.82      0.84      0.83      2728
```

Table 5.2.2.1: Accuracy Report of RFC

Model made by Random Forest Classification has prediction accuracy of 84% and loss of 47%.

5.2.3 Confusion matrix:

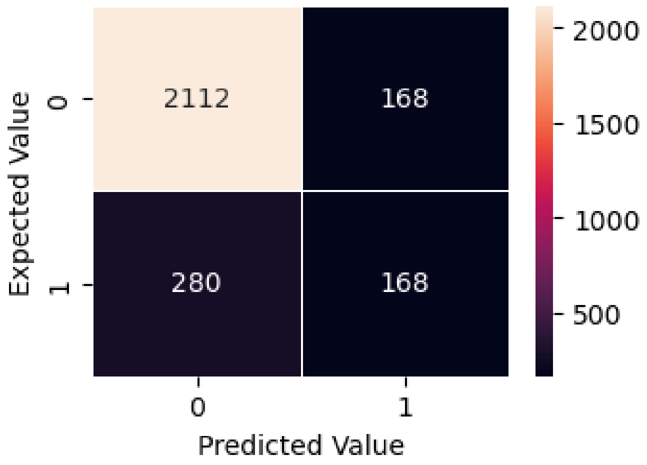


Fig 5.2.3.1: Confusion Matrix of RFC

For RFC:

The number of game will be hit predicted successfully(TP)=168
The number of game will be hit predicted unsuccessfully(FP)=168
The number of game will not be hit predicted successfully(TN)=2112
The number of game will not be hit predicted unsuccessfully(FN)=280

5.2.4 ROC Curve:

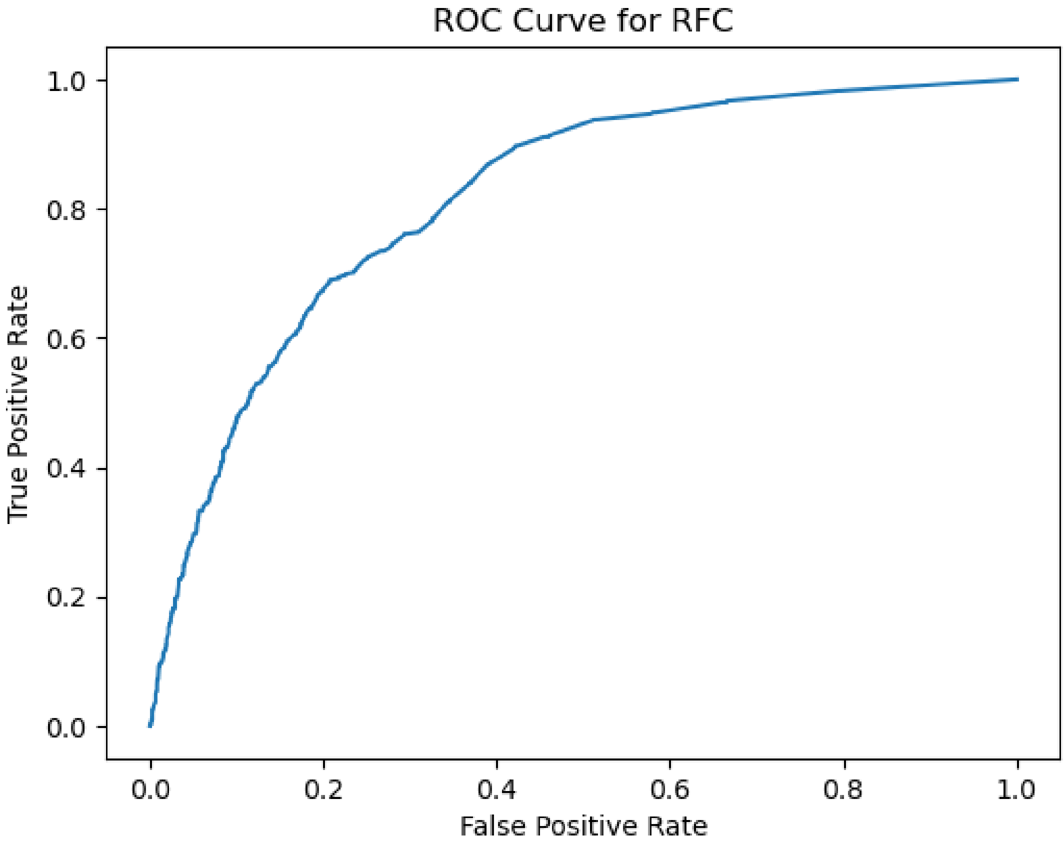


Fig 5.2.4.1: ROC curve of RFC

For RFC True positive rate is higher than False positive rate but its slightly worse than LR

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Result:

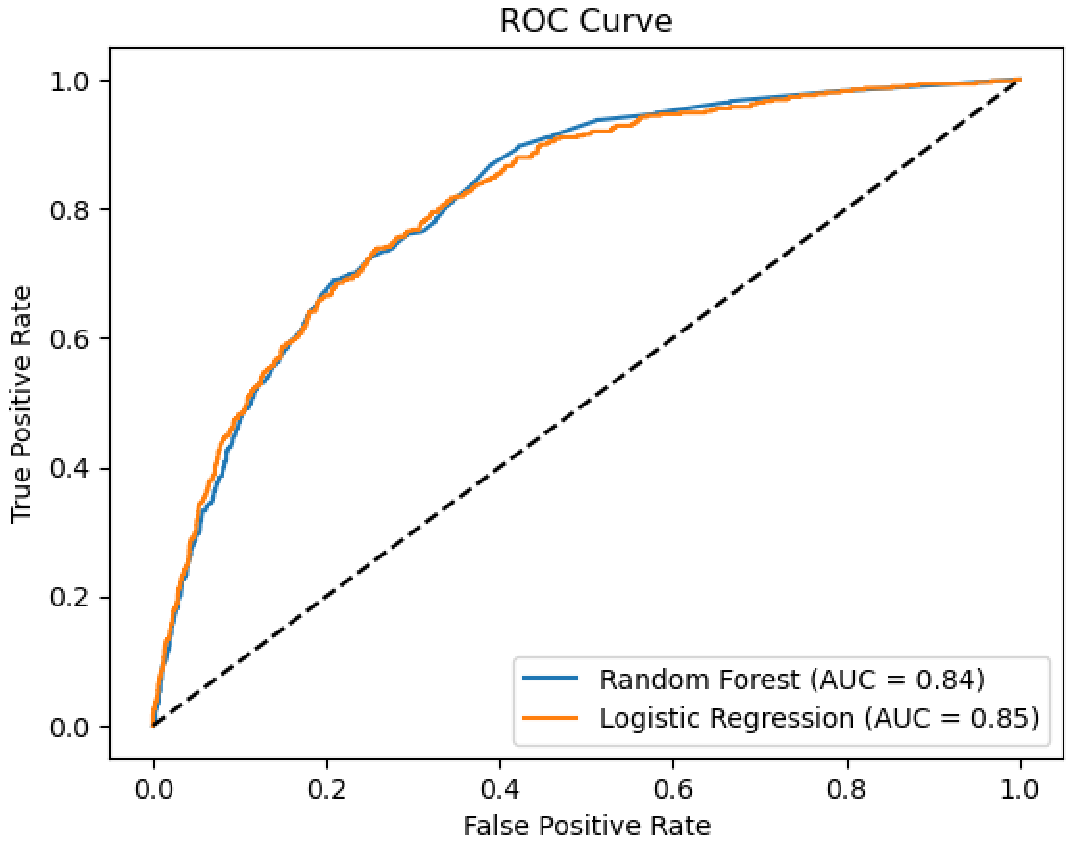


Fig 6.1.1: compressive ROC curve of LR and RFC

Model	Precession		Recall		F1 Score		Accuracy	Loss
	0	1	0	1	0	1		
LR	0.86	0.60	0.98	0.18	0.91	0.28	0.85	0.35
RFC	0.88	0.50	0.93	0.38	0.90	0.43	0.84	0.48

Table 6.1.2: performance comparison of LR and RFC

Analyzing the above results we can conclude that LR model provides us highire accuracy (85%) and loss (35%) compared to RFC with accuracy (85%) and loss (35%).

6.2 Discussion:

Sources such as Kaggle, Gamespot, and Metacritic were mainly used to collect the data for this study. Accurate game sales data is not always released by publishers in order to maintain their stock prices, and other important features such as marketing costs, developer information, or game mechanics are often kept secret to protect their company's privacy. If provided by the publisher, these features can improve the accuracy of our model.

CHAPTER 7

CONCLUSION

7.1 Conclusion:

An approach has been proposed to predict the probability of a game's success prior to its release by analyzing previous game sales data. More data sets with additional features can be included in the proposed models to enhance their accuracy. By using this model, the game's publisher will have a means to predict its success after release with a high degree of accuracy. This will assist in reducing the number of unsuccessful games.

References:

[1] Trneny, M. (2017). Machine learning for predicting success of video games. Masaryk University, Faculty of Informatics.

[2] BEAUJON, Walter Steven. Predicting Video Game Sales in the European Market. 2012. Available also from: https://www.few.vu.nl/nl/Images/werkstuk-beaujon_tcm243-264134.pdf.

[3] EHRENFELD, Steven Emil. Predicting Video Game Sales Using an Analysis of Internet Message Board Discussions. 2011. Available also from: https://sdsu-dspace.calstate.edu/bitstream/handle/10211.10/1073/Ehrenfeld_Steven.pdf. Master's thesis. San Diego State University

[4] GHATTAMANENI, Sriram; KOMARRAJU, Agastya kumar. The Game Prophet: Predicting the success of Video Games. 2012. Available also from: https://cepd.okstate.edu/files/Analytics_Ghatta.pdf.

[5] SIFA, Rafet; SRIKANTHY, Sridev; DRACHENZ, Anders; OJEDA, Cesar; BAUCKHAGE, Christian. Predicting Retention in Sandbox Games with Tensor Factorization-based Representation Learning. In: NOMIKOS, Petros M. (ed.). 2016 IEEE Conference on Computational Intelligence and Games. 2016, p. 142.

[6] TAMASSIA, Marco; RAFFEY, William; SIFAZ, Rafet; DRACHENX, Anders; ZAMBETTA, Fabio; HITCHENS, Michael. Predicting Player Churn in Destiny: A Hidden Markov Models Approach to Predicting Player Departure in a Major Online Game. In: NOMIKOS, Petros M. (ed.). 2016 IEEE Conference on Computational Intelligence and Games. 2016, p. 325.

[7] SHAKER, Noor; ABOU-ZLEIKHA, Mohamed. Transfer Learning for Cross-Game Prediction of Player Experience. In: NOMIKOS, Petros M. (ed.). 2016 IEEE Conference on Computational Intelligence and Games. 2016, p. 209.

[8]Video Game Sales with Ratings: https://www.kaggle.com/datasets/rush4ratio/video-game-sales-with-ratings

[9]PC Game Sales : https://www.kaggle.com/datasets/khaiid/most-selling-pc-games

[10]Manual Dataset : https://drive.google.com/file/d/1tvFanh-sjT7CiIkErtRh-ZNhdh5LFG1Z/view?usp=share_link