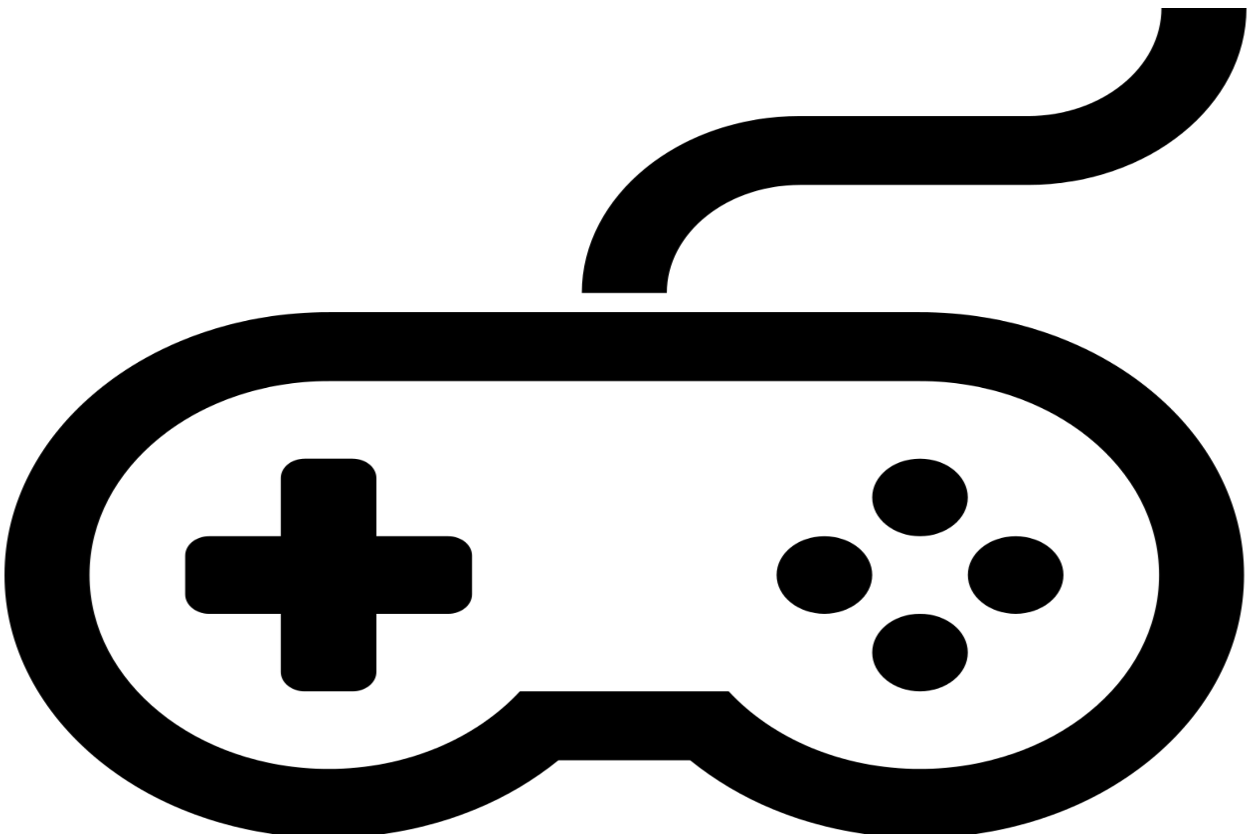
Live in Your World. Play In Ours



Video Game Sales

Final Report

Nisha Pepsi Selvarajan

Introduction:

In this project, we propose to build a model that can predict video games sales based on features from dataset. Emphasis is placed on video game publishers like play station which will helpi them predict which games will be best sellers before they are released. The data we used identifies games based on genre, publisher, platform, etc. giving us multiple factors useful for predicting a game's success.

The main work that we have done includes: analyzing the features of data set via data-visualization, processing the data set, using four regression model to predict the model. Firstly, we read in data from the data set and explore the data, getting a brief recognition of features in this data set. After that, we modified the data set in following ways including renaming, processing the missing values and integrating significant features.

Then, we select 4 different machine learning models as candidates including linear regression, ridge regression, random forest regression and KNN. We use them to make predictions and evaluate their performance to decide which models are appropriate to be used for further modification.

Data Source

The original dataset has games ranging from 1980 to 2020 with 11,493 different game titles. There are 579 publishers with 31 platforms. Games are broken down into 12 unique categories as follows: Sports, Platform, Racing, Role-playing, Puzzle, Misc, Shooter, Simulation, Action, Fighting, Adventure,and Strategy. The dataset was taken off of Kaggle, but originates from VGChartz, a business intelligence and research firm. Additional data has been provided by Metacritic which has critics' scores, user scores, developer name, and rating for recommended maturity of player. The shape of dataset is **(16719, 16)**

Glimpse of the data:

Name	Platfor	Year_of	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sal	Global_Sa	Critic_Sco	Critic_Cou	User_Scor	User_Cou	Develope	Rating
Wii Sports	Wii	2006	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76	51	8	322	Nintendo	E
Super Mai	NES	1985	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24						
Mario Kar	Wii	2008	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	82	73	8.3	709	Nintendo	E
Wii Sports	Wii	2009	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	80	73	8	192	Nintendo	E
Pokemon	GB	1996	Role-Playi	Nintendo	11.27	8.89	10.22	1	31.37						
Tetris	GB	1989	Puzzle	Nintendo	23.2	2.26	4.22	0.58	30.26						
New Supe	DS	2006	Platform	Nintendo	11.28	9.14	6.5	2.88	29.8	89	65	8.5	431	Nintendo	E
Wii Play	Wii	2006	Misc	Nintendo	13.96	9.18	2.93	2.84	28.92	58	41	6.6	129	Nintendo	E
New Supe	Wii	2009	Platform	Nintendo	14.44	6.94	4.7	2.24	28.32	87	80	8.4	594	Nintendo	E
Duck Hunt	NES	1984	Shooter	Nintendo	26.93	0.63	0.28	0.47	28.31						
Nintendo	DS	2005	Simulatio	Nintendo	9.05	10.95	1.93	2.74	24.67						
Mario Kar	DS	2005	Racing	Nintendo	9.71	7.47	4.13	1.9	23.21	91	64	8.6	464	Nintendo	E
Pokemon	GB	1999	Role-Playi	Nintendo	9	6.18	7.2	0.71	23.1						
Wii Fit	Wii	2007	Sports	Nintendo	8.92	8.03	3.6	2.15	22.7	80	63	7.7	146	Nintendo	E
Kinect Ad	X360	2010	Misc	Microsoft	15	4.89	0.24	1.69	21.81	61	45	6.3	106	Good Scie	E
Wii Fit Plu	Wii	2009	Sports	Nintendo	9.01	8.49	2.53	1.77	21.79	80	33	7.4	52	Nintendo	E
Grand The	PS3	2013	Action	Take-Two	7.02	9.09	0.98	3.96	21.04	97	50	8.2	3994	Rockstar N	M
Grand The	PS2	2004	Action	Take-Two	9.43	0.4	0.41	10.57	20.81	95	80	9	1588	Rockstar N	M
Super Mai	SNES	1990	Platform	Nintendo	12.78	3.75	3.54	0.55	20.61						
Brain Age	DS	2005	Misc	Nintendo	4.74	9.2	4.16	2.04	20.15	77	58	7.9	50	Nintendo	E
Pokemon	DS	2006	Role-Playi	Nintendo	6.38	4.46	6.04	1.36	18.25						

Variable Summaries:

Variable Name	Var Type	Summary
Name	String	Lists name of video game
Platform	String	31 distinct platform names in abbreviated form (i.e. Wii, GB)
YearofRelease	String	Lists year game was released from 1980 to 2020
Genre	String	12 distinct genres (i.e. Sports, Racing, Puzzle)
Publisher	String	580 distinct publishers (i.e. Nintendo, Microsoft, Sony)
NA_Sales	Numeric	North American sales for each game in millions of dollars Min: 0 Max: $41.36 million Mean: $.263 million
EU_Sales	Numeric	European sales for each game in millions of dollars

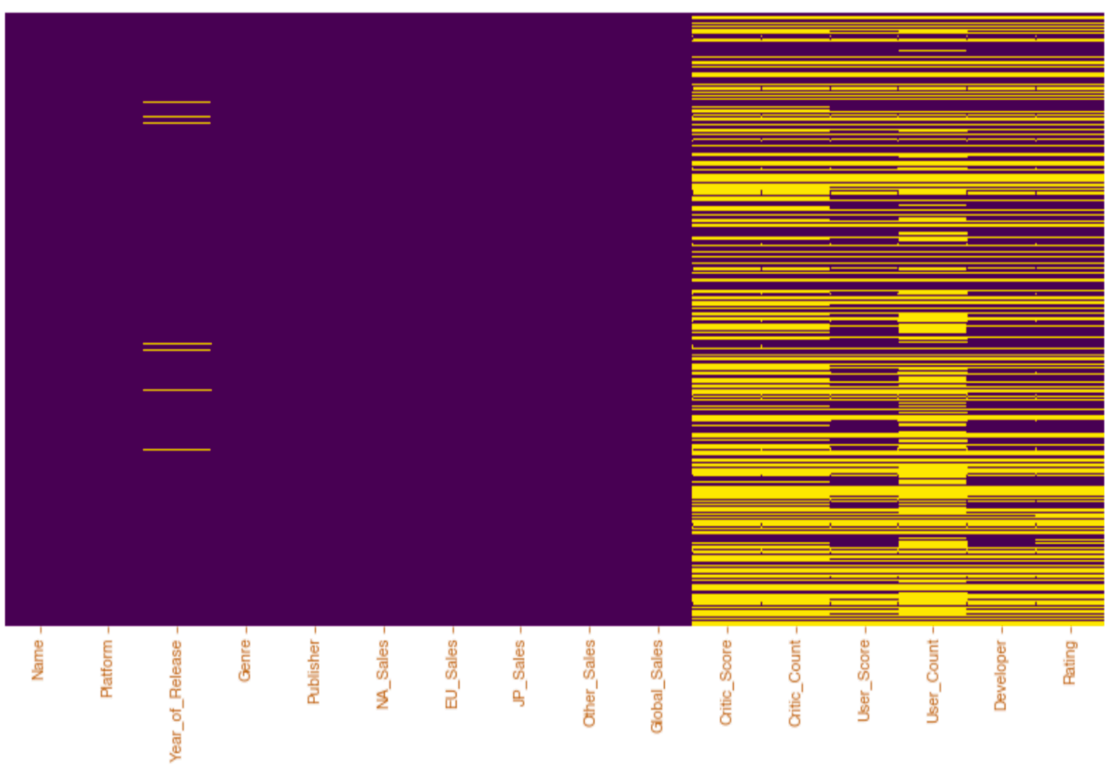
		Min: 0 Max: $28.96 million Mean: $.145 million
JP_Sales	Numeric	Japan sales for each game in millions of dollars Min: 0 Max: $10.22 million Mean: $.078 million
Other_Sales	Numeric	Sales for regions not included in North America, Europe, or Japan. In millions of dollars. Min: 0 Max: $10.57 million Mean: $.047 million
Global_Sales	Numeric	Total sales in millions of dollars Min: $.01 Max: $82.53 million Mean: $.534 million
Critic_Score	Numeric	Score from 0-100 based on critic reviews where higher scores indicate more favorable reviews *Missing 51% Min: 13 Max: 98 Mean: 68.963
Critic_Count	Numeric	Number of critic reviews used to form critic score *Missing 51% Min: 3 Max: 113 Mean: 26.361
User_Score	Numeric	Score from 0-10 based on user reviews *Missing 55% Min: 0 Max: 9.7 Mean: 7.125
User_Count	Numeric	Number of user reviews used to form User_Score *Missing 55% Min: 4 Max: 10665 Mean: 162.23
Developer	String	1696 distinct developer names *Missing 40% (i.e. Nintendo, Game Arts)
Rating	String	8 distinct ratings (E for everyone, M for mature) *Missing 40%

Data Cleaning

1. **Changing data type.** From the overall review of our data set, we can see that the data are in different data type. Some of them are numerical data, like user score, user counts, critic score and critic counts. While some of them are text data, like Name of game, publisher and platform. We must change the data type for our further work.

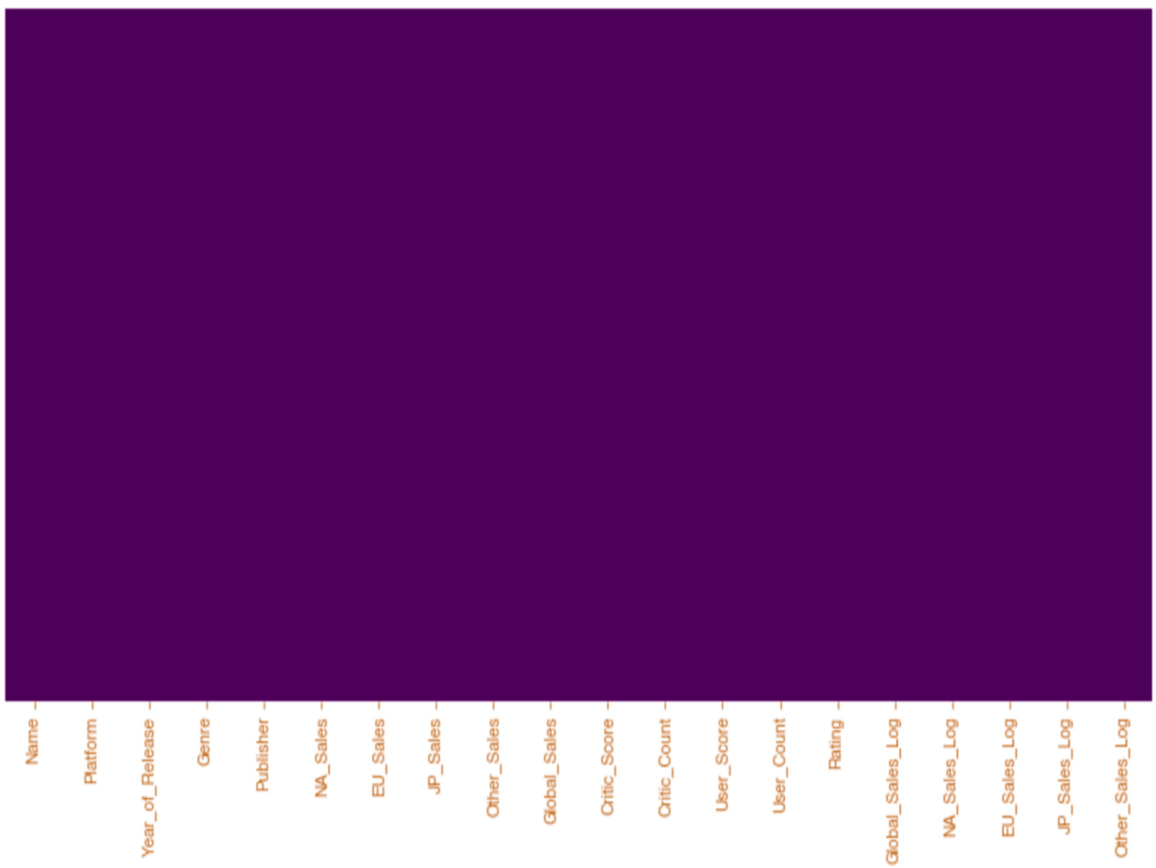
```
Name                          object
click to scroll output; double click to hide
Year_of_Release              float64
Genre                         object
Publisher                     object
NA_Sales                     float64
EU_Sales                     float64
JP_Sales                     float64
Other_Sales                  float64
Global_Sales                 float64
Critic_Score                 float64
Critic_Count                 float64
User_Score                    object
User_Count                   float64
Developer                     object
Rating                        object
dtype: object
```

2. **Processing the missing data**. We can see that lots of game do not have the feature critic score and user score, which will make a vital impact on our project. According to this poor data set, we must fill the missing values with some rational data. Below figure shows missing data in the corresponding data columns.



- *Textual data:* If the original data type is text, we need to fill the missing data with appropriate value or general text "TBD/ Unknown".
- *Numerical data* If the original data type is numerical, we need to fill the missing data with this column's mean value.
- *Outlier*: Outlier is an observation that lies an abnormal distance from other values in a random sample from a population. there are outliers in sales columns. They might be useful for training as they indicate bestseller games, but for now we are going to remove them and maybe add them later.

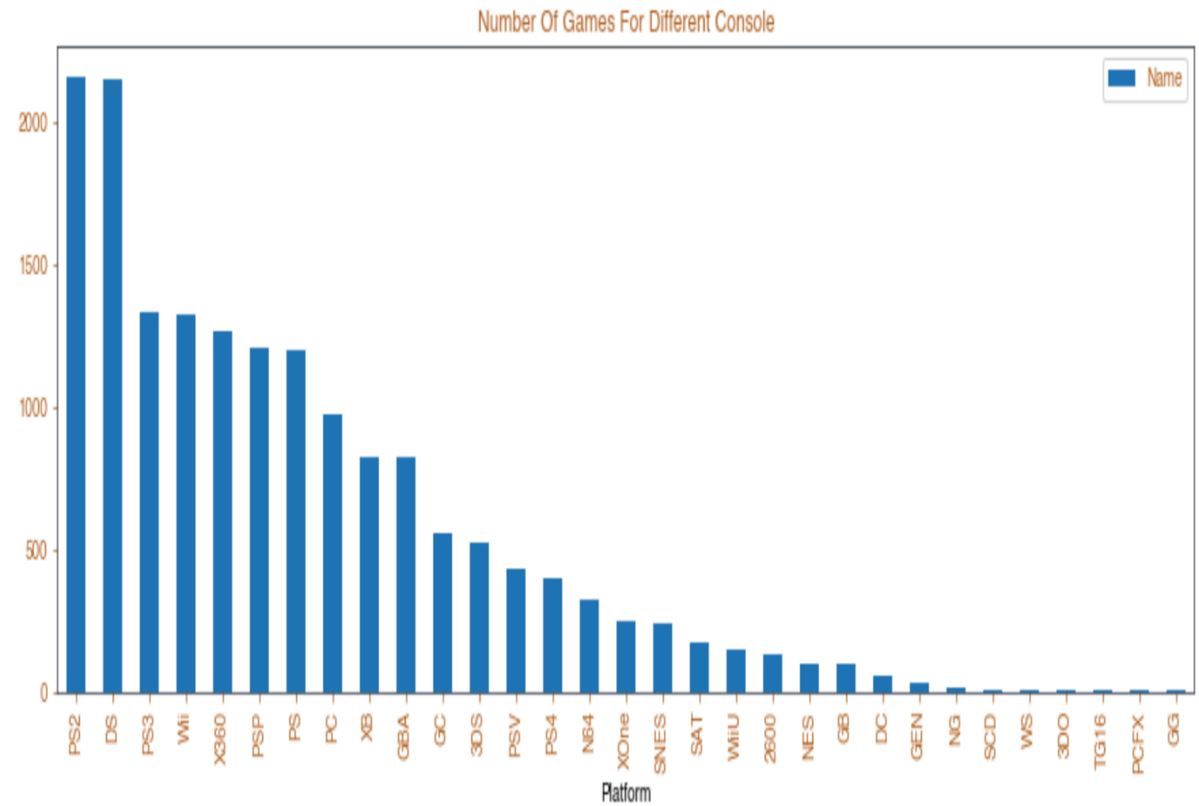
Below figure shows the data frames after data clean up.



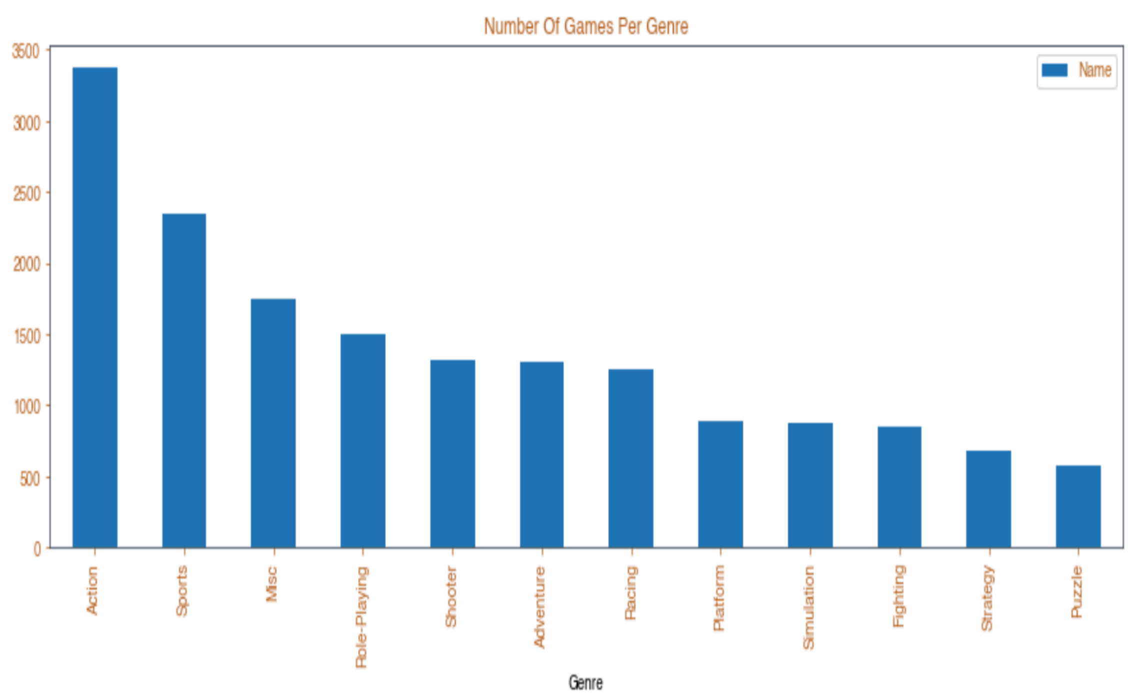
Data Analysis:

Below are few of the analytic questions which the dataset can answer

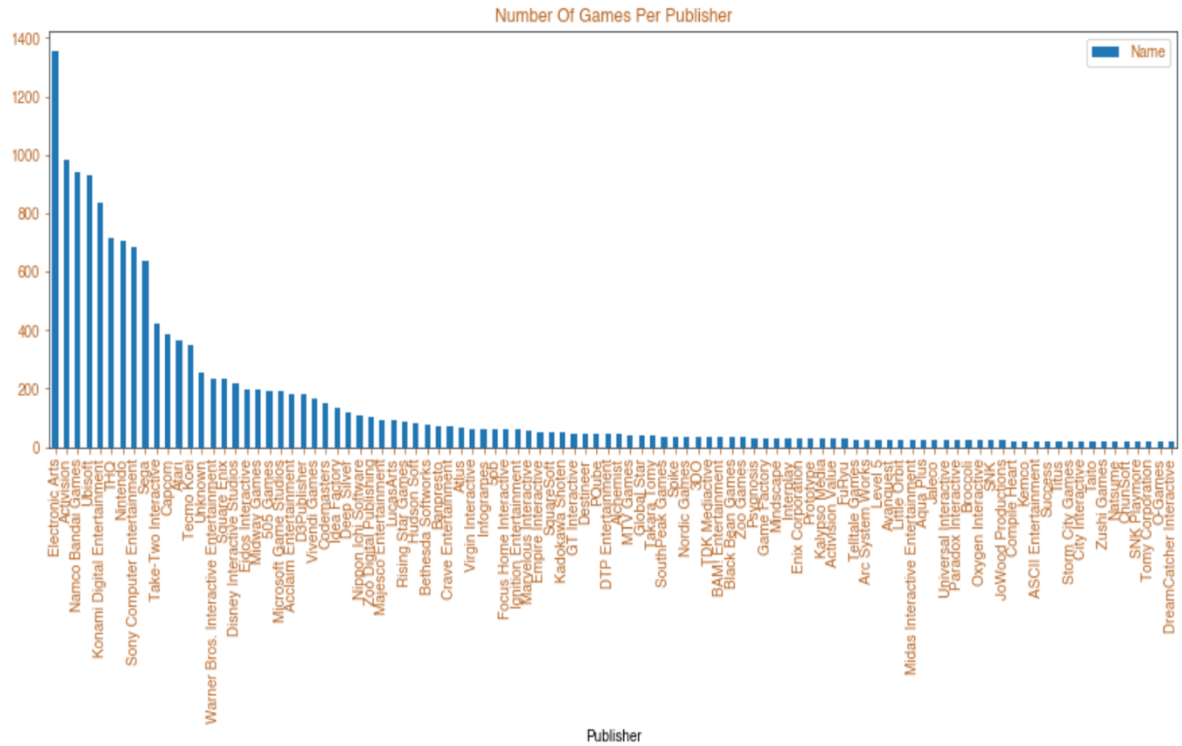
NUMBER OF DIFFERENT GAMES PER CONSOLE:



NUMBER OF DIFFERENT GAMES PER GENRE



NUMBER OF DIFFERENT GAMES PER PUBLISHER



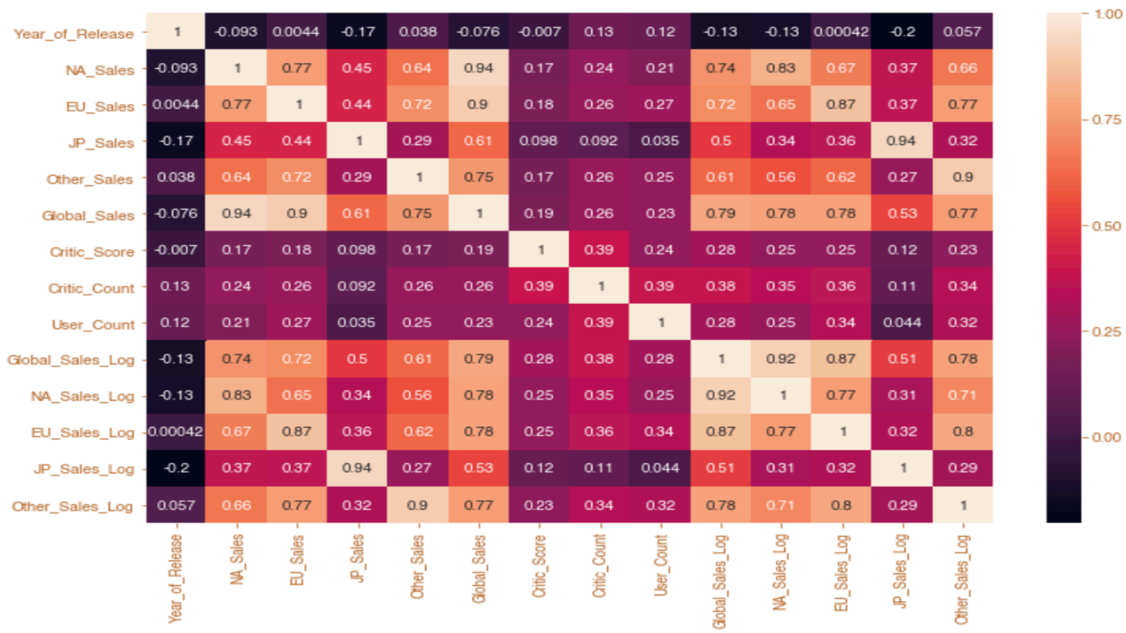
Data Correlation: Is a way to understand the relationship between multiple variables and attributes in your dataset. Using Correlation, you can get some insights such as

- One or multiple attributes depend on another attribute or a cause for another attribute.
- One or multiple attributes are associated with other attributes.

SO, WHY IS CORRELATION USEFUL?

- Correlation can help in predicting one attribute from another (Great way to impute missing values).
- Correlation can (sometimes) indicate the presence of a causal relationship.
- Correlation is used as a basic quantity for many modelling techniques

Below figure shows correlation for different columns, in the dataset.



Requirements:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
from pandas import Series
import seaborn as sns
import numpy as np
from sklearn.metrics import precision_recall_fscore_support
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import Ridge
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, f1_score, accuracy_score, confusion_matrix
from sklearn import svm
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
from sklearn.metrics import auc
from sklearn.model_selection import learning_curve
from sklearn.neighbors import KNeighborsClassifier
```

First ML Problem Statement:

1. How Global Sales Gets affected with Critic_Score_x', 'Critic_Count_x', 'User_Score_x', 'User_Count_x', 'year_after_release_x'

   X: Critic_Score_x', 'Critic_Count_x', 'User_Score_x', 'User_Count_x','year_after_release_x'
   Y: Global_Sales_Log

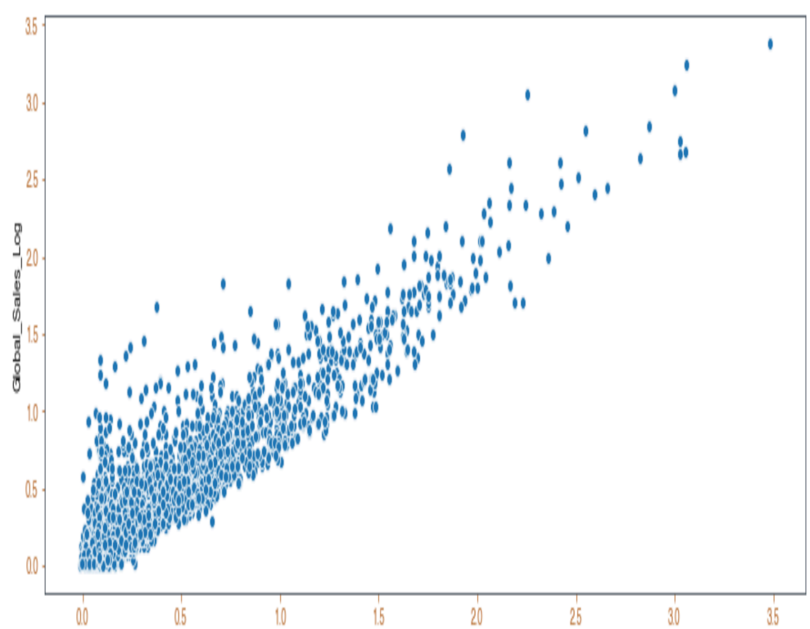
MODELING:

We use 1/2 of all data in the data set as training data and the left 1/2 data as testing data. we will evaluate the model performance by Mean Absolute Error(MAE). At the meanwhile, we will make plot graph of each models for visualized assessment of their performance.

LINEAR REGRESSION:

mean_absolute_error	0.09902835743695935
mean_squared_error	0.02286022063308966
accuracy	0.8523184660252421

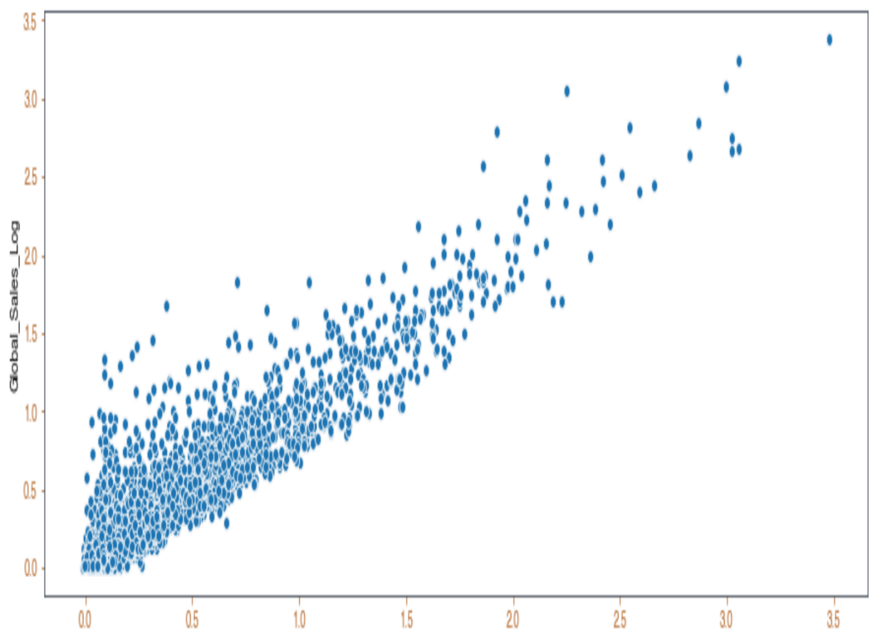
SCATTER PLOT:



RIDGE REGRESSION:

mean_absolute_error	0.09908128080265304
mean_squared_error	0.022863384030535606
Accuracy	0.8522980298538295

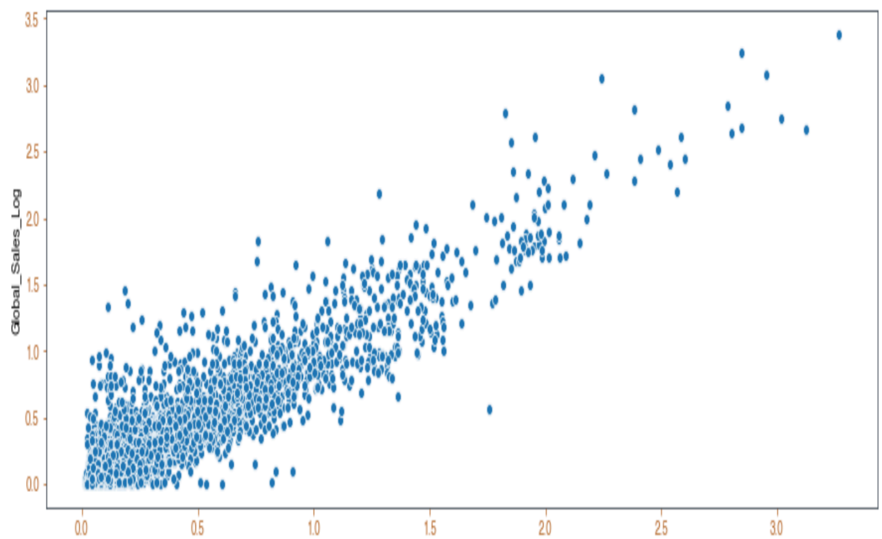
SCATTER PLOT:



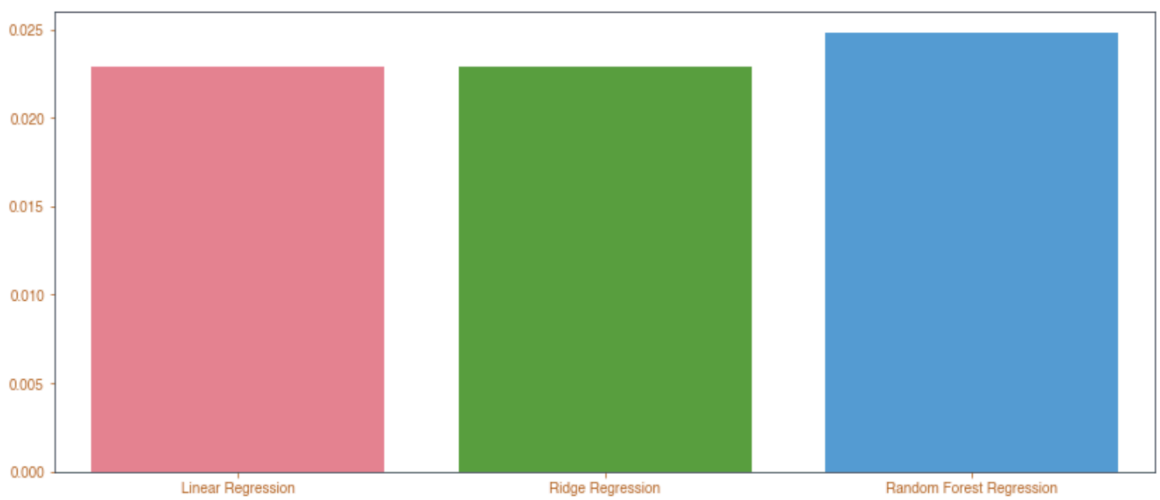
RANDOM FOREST REGRESSION:

mean_absolute_error	0.09884309984717227
mean_squared_error	0.025108675278109 16
Accuracy	0.8379337860864112

SCATTER PLOT:



Evaluate different models based on MSE:



Second ML Problem Statement:

2. Does a game hit gets affected by *Year_of_Release Critic_Score*
X: Year_of_Release ,Critic_Score
Y: Hit

Modeling:

We use 1/2 of all data in the data set as training data and the left 1/2 data as testing data. we will evaluate the model performance by Mean Absolute Error(MAE).

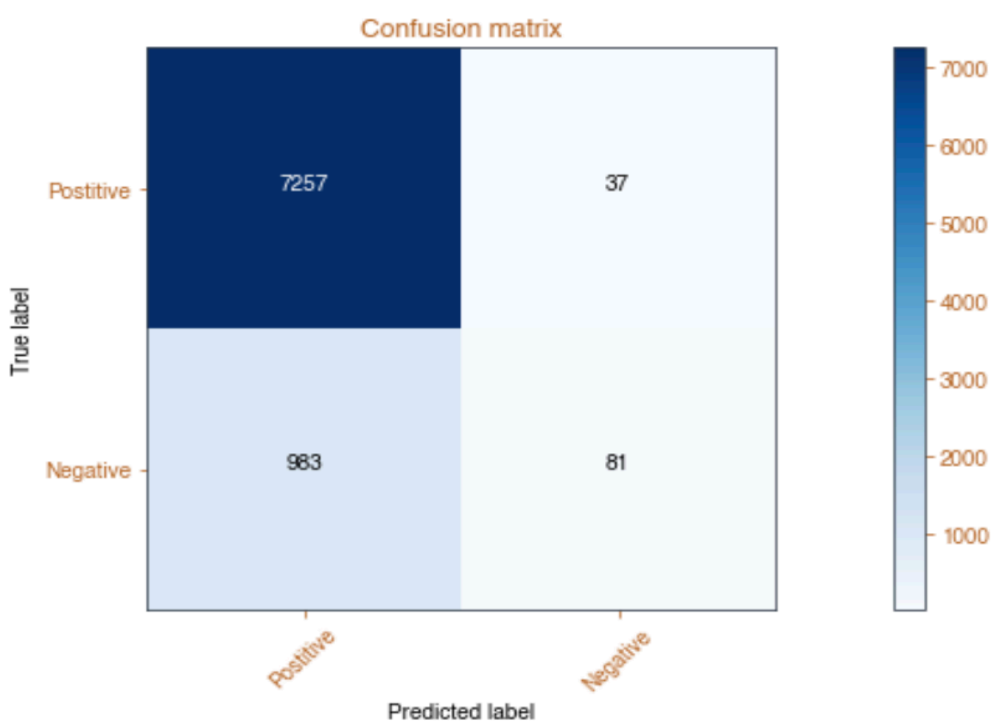
Random Forest Classifier:

mean_absolute_error	0.18015534953645704
mean_squared_error	0.18015534953645704
variance	0.20697724448893373
accuracy	0.8198446504635429

Logistic Regression:

mean_absolute_error	0.16612377850162866
mean_squared_error	0.16612377850162866
variance_score	0.014201230193749192
Accuracy	0.8338762214983714
Regression on training set	0.8466549736908043
Regression Score on test set	0.8338762214983714

Confusion Matrix:



Classification Report:

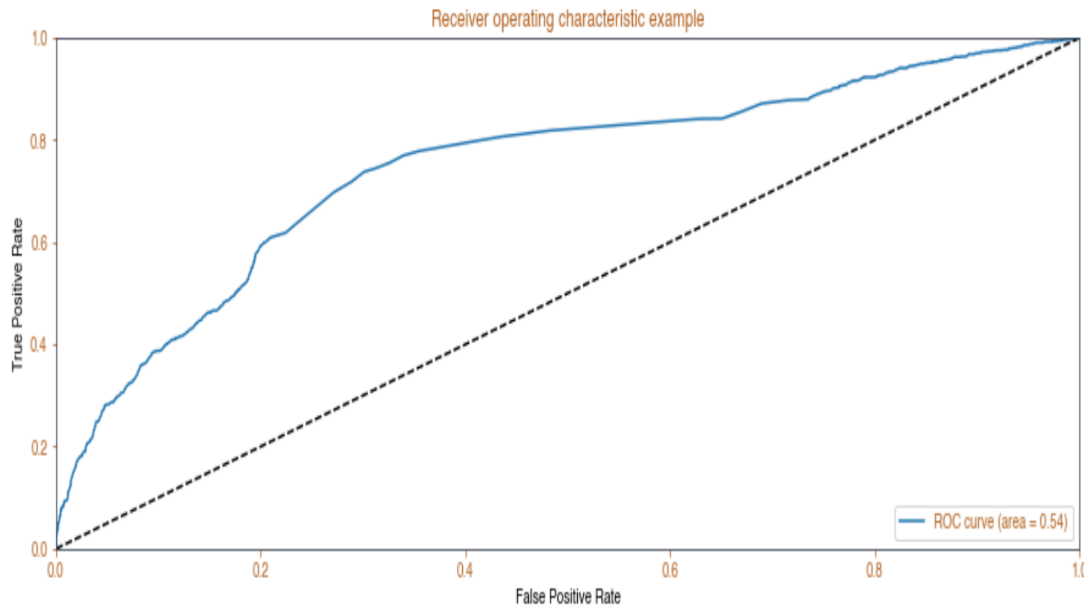
Receiver Operating Characteristic curve

This type of graph is called a *Receiver Operating Characteristic curve* (or ROC curve.) It is a plot of the true positive rate against the false positive rate for the different possible cutpoints of a diagnostic test.

An ROC curve demonstrates several things:

- It shows the tradeoff between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity).

- The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test.
- The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test.



Third ML Problem Statement:

1. How does the game rated as "RATING EVERYONE", changes with Year_of_Release Sales Platform

X: 'Year_of_Release' ,'Sales','Platform'
Y: 'Global_Sales_Log'

MODELING:

We use 1/2 of all data in the data set as training data and the left 1/2 data as testing data. we will evaluate the model performance by Mean Absolute Error(MAE). At the meanwhile, we will make plot graph of each models for visualized assessment of their performance.

Logistic Regression

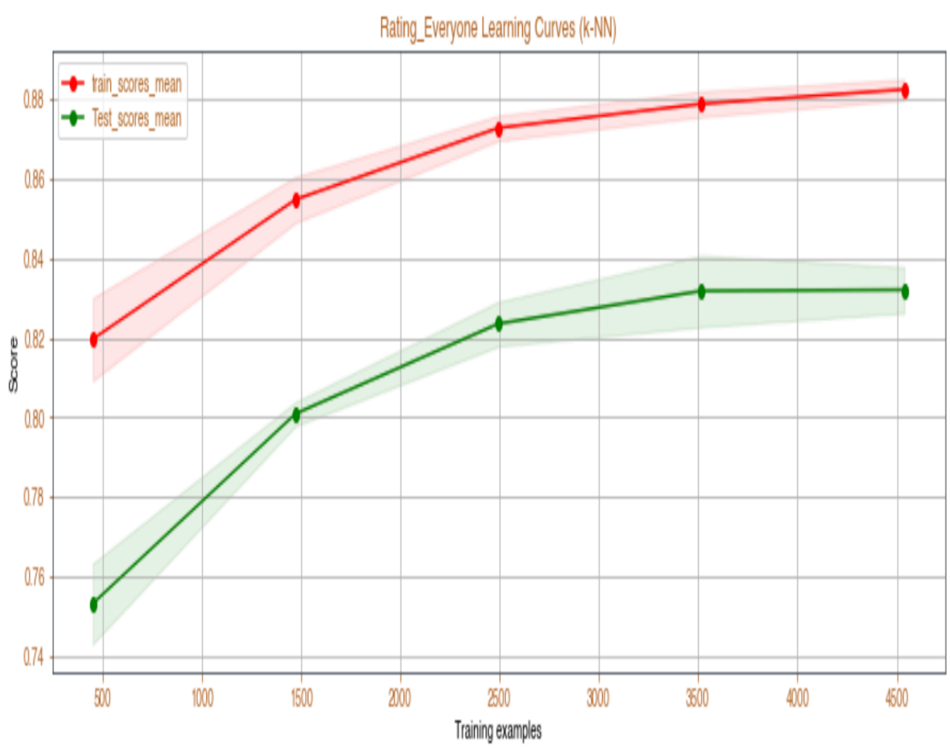
mean_absolute_error	0.16662490603858682

mean_squared_error	0.16662490603858682
variance	0.011810211222773925
accuracy	0.8333750939614132

K-Neighbour Classifier

mean_absolute_error	0.1623653219744425
mean_squared_error	0.1623653219744425
variance	0.03927400129078129
accuracy	0.8376346780255575

K-Neighbor Classifier Learning Curve Test & Train Score Mean:

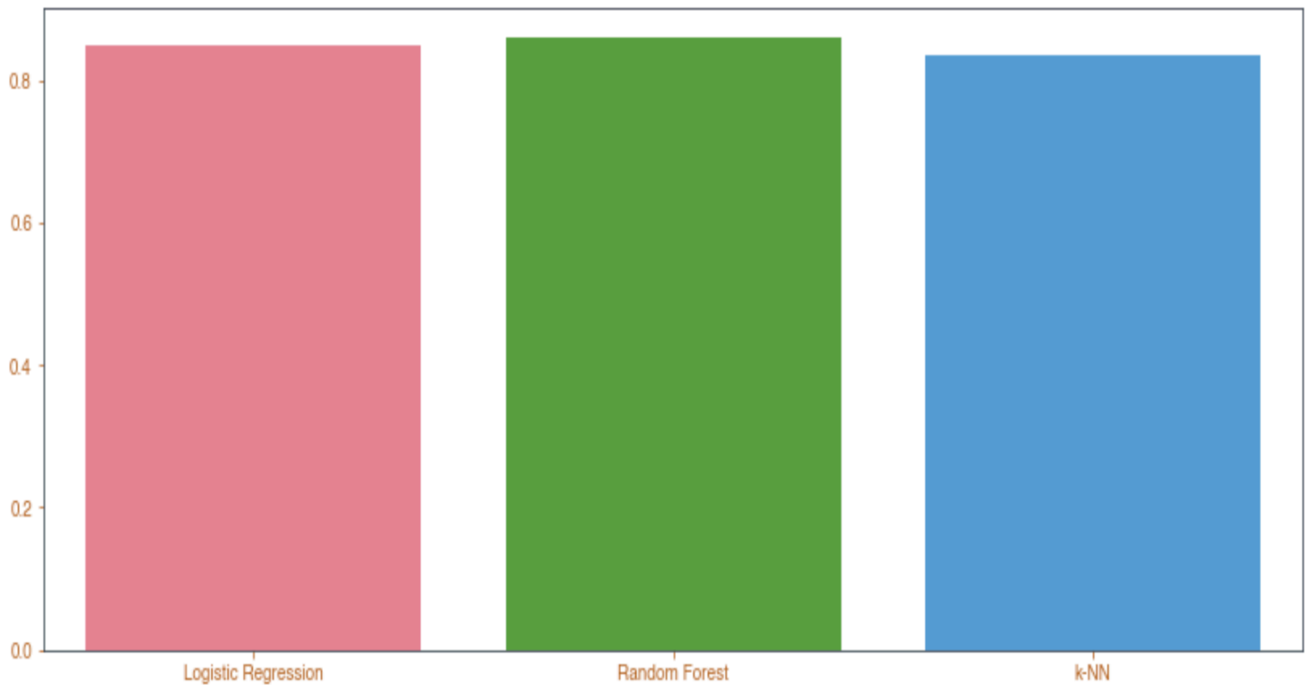


RANDOM Forest Classifier

mean_absolute_error	0.18466549736908044
mean_squared_error	0.18466549736908044

variance	-0.24269676733883894
accuracy	0.8153345026309196

Evaluate different models based on Accuracy:



CODE:

Attached PDF file, with all codes and output.