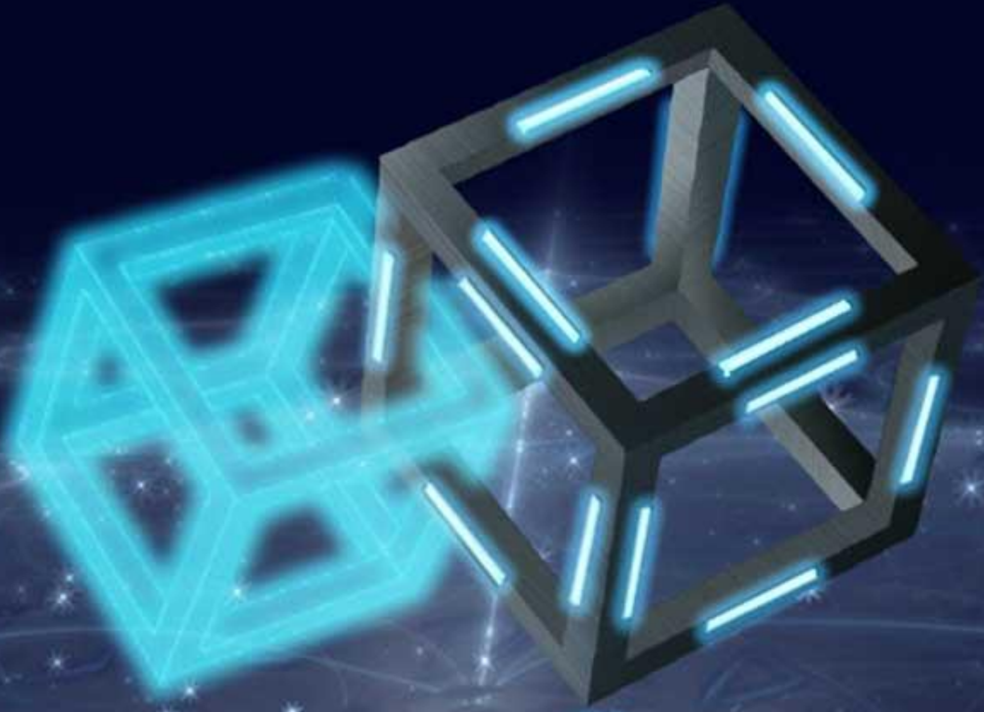
ACHDIYAT KUSUMA FINAL PROJECT

Video_Games_Sales.csv



WELCOME

FINAL PROJECT PRESENTATION

Final Project / Video_Games_Sales.csv - Executive Summary

Business Background

The data we were provided suggest that the company currently have approximately 16.700 data. Which contains many feature involving Game such as : Name, Platform, Release Date, Genre, Publisher, Sales, Critics, Users, Developers and Ratings

Problems Statements

There has been a steady decline on Global Sales which is the total value of Sales since 2010 until 2016 (according to the data provided)

Objective

Find out which feature to be used to deter the declining Sales

Proposed Solutions

Use Machine Learning (ML) to predict which feature of the games have the potential to bring the sales up and which is not. Then, provide special treatment accordingly

Result:

Analysis Results:

- ❖ The Genre Action and Sport bears the most Sales and possibly the most promising genre to be treated specially
- ❖ From the publisher's side, the most the bear Sales are EA, Activision and Ubisoft
- ❖ Sales from NA are significantly the best

ML Result :

the evaluation using Linear Regression showed 97% of R Squared and 0.0016 MSE score accuracy in the prediction model.

Business Benefit

- ✓ Solved the problem of decreasing profit
- ✓ It could serve as a baseline to product treatment
- ✓ Serves analyzed data of User interest, behavior and preference of games
- ✓ Give hints of the current trend of games which can be useful for future marketing plan.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16719 entries, 0 to 16718
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Name             16717 non-null  object
 1   Platform         16719 non-null  object
 2   Year_of_Release  16450 non-null  float64
 3   Genre            16717 non-null  object
 4   Publisher        16665 non-null  object
 5   NA_Sales         16719 non-null  float64
 6   EU_Sales         16719 non-null  float64
 7   JP_Sales         16719 non-null  float64
 8   Other_Sales      16719 non-null  float64
 9   Global_Sales     16719 non-null  float64
 10  Critic_Score     8137 non-null   float64
 11  Critic_Count     8137 non-null   float64
 12  User_Score       10015 non-null  object
 13  User_Count       7590 non-null   float64
 14  Developer        10096 non-null  object
 15  Rating           9950 non-null   object
dtypes: float64(9), object(7)
memory usage: 2.0+ MB
```

```
[92] # Menampilkan jumlah baris dan kolom pada datafr
     print(df.shape)
```

```
(16719, 16)
```

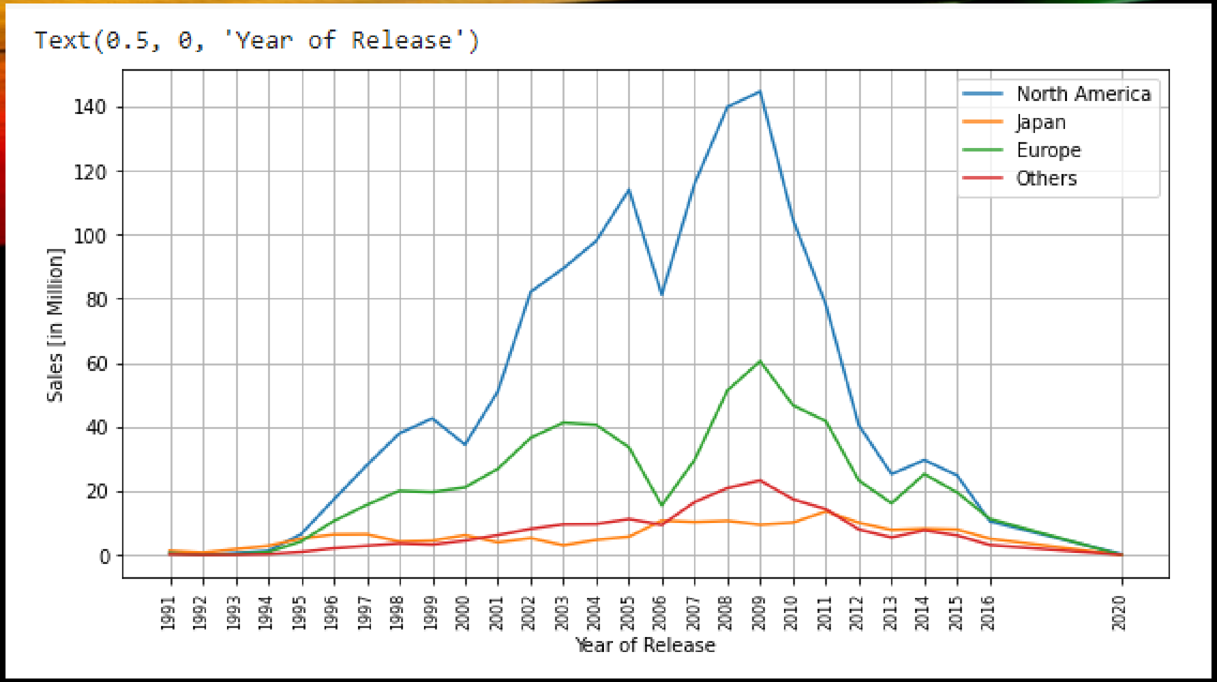
```
replaced_value_df.info()
```

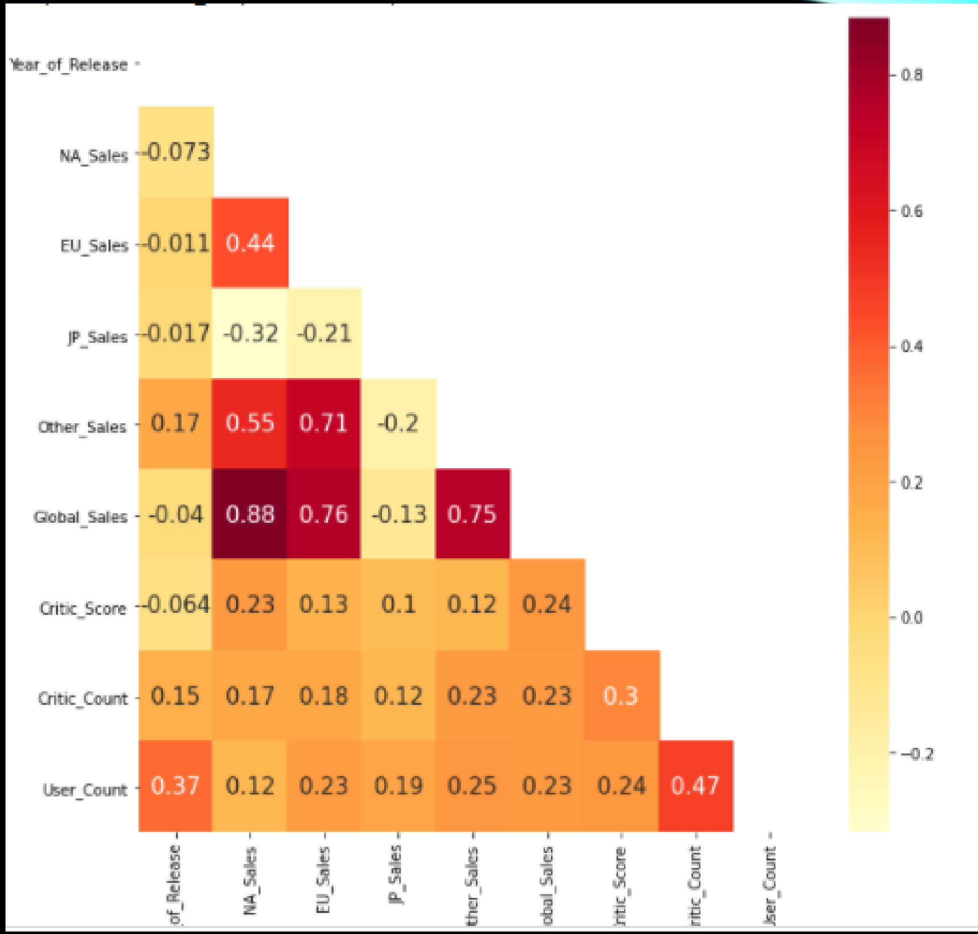
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7365 entries, 1366 to 10826
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Name             7365 non-null   object
 1   Platform         7365 non-null   object
 2   Year_of_Release  7365 non-null   float64
 3   Genre            7365 non-null   object
 4   Publisher        7365 non-null   object
 5   NA_Sales         7365 non-null   float64
 6   EU_Sales         7365 non-null   float64
 7   JP_Sales         7365 non-null   float64
 8   Other_Sales      7365 non-null   float64
 9   Global_Sales     7365 non-null   float64
 10  Critic_Score     7365 non-null   float64
 11  Critic_Count     7365 non-null   float64
 12  User_Score       7365 non-null   object
 13  User_Count       7365 non-null   float64
 14  Developer        7365 non-null   object
 15  Rating           7365 non-null   object
dtypes: float64(9), object(7)
memory usage: 978.2+ KB
```

```
[148] print(replaced_value_df.shape)
```

```
(7365, 16)
```

The data we used is surprisingly very dirty with lots of null value. So we decided to to do outlier first then proceed to handle missing value.
It’s more efficient this way because we believe that missing value would stretch the numerical data by a lot thus affecting the outlier result.
And we’ve also tested if we handle missing value first it would mean loosing too much data






```
final_feature = ['Year_of_Release', 'Platform', 'Genre', 'NA_Sales', 'EU_Sales', 'Other_Sales', 'Global_Sales', 'Critic_Score', 'Critic_Count', 'User_Count']
```

After giving a look at the Graph and Heatmap above we can conclude that JP_Sales is not a significant Variable thus i chose to remove it from final feature in order to further improve my analysis

```
# Evaluasi Model dengan Mean Square Error (MSE) dan R squared
print("MSE :", metrics.mean_squared_error(y_test,y_test_pred))
print("R squared :", metrics.r2_score(y_test,y_test_pred))
```

```
MSE : 3.15843608838748e-05
R squared : 0.9994666376262938
```

With
JP_Sales

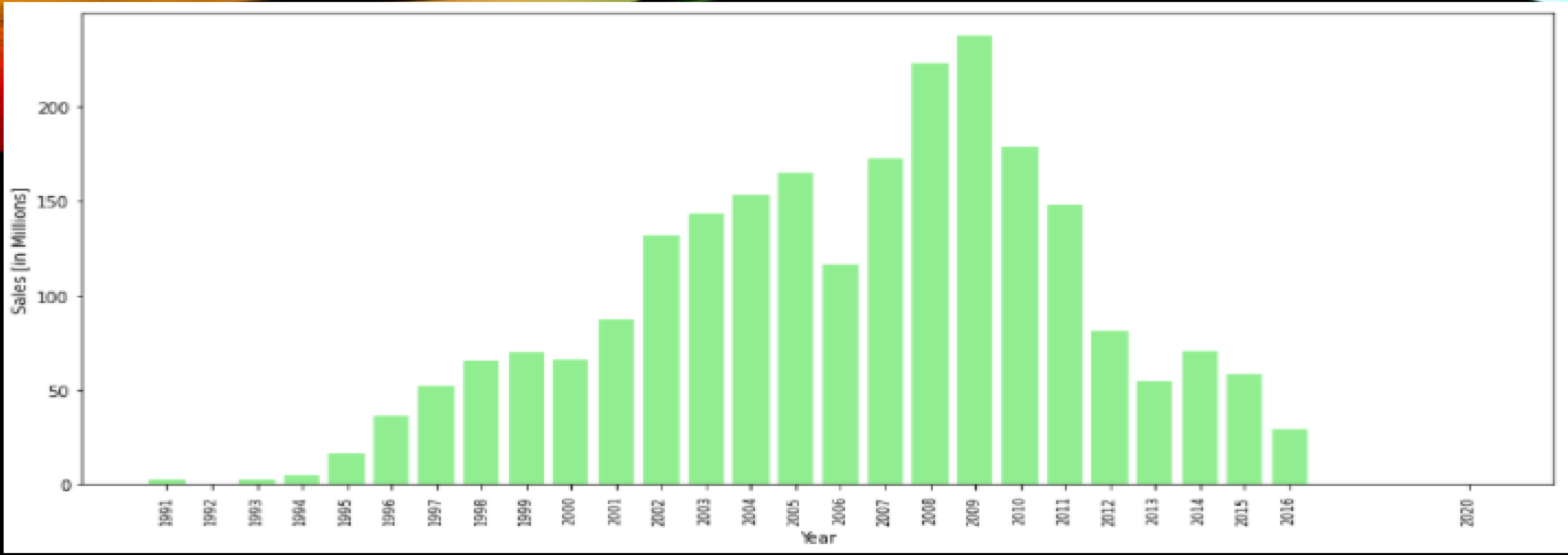
```
[228] # Evaluasi Model dengan Mean Square Error (MSE) dan R squared
      print("MSE :", metrics.mean_squared_error(y_test,y_test_pred))
      print("R squared :", metrics.r2_score(y_test,y_test_pred))
```

```
MSE : 0.001680154702264934
R squared : 0.9716273726896582
```

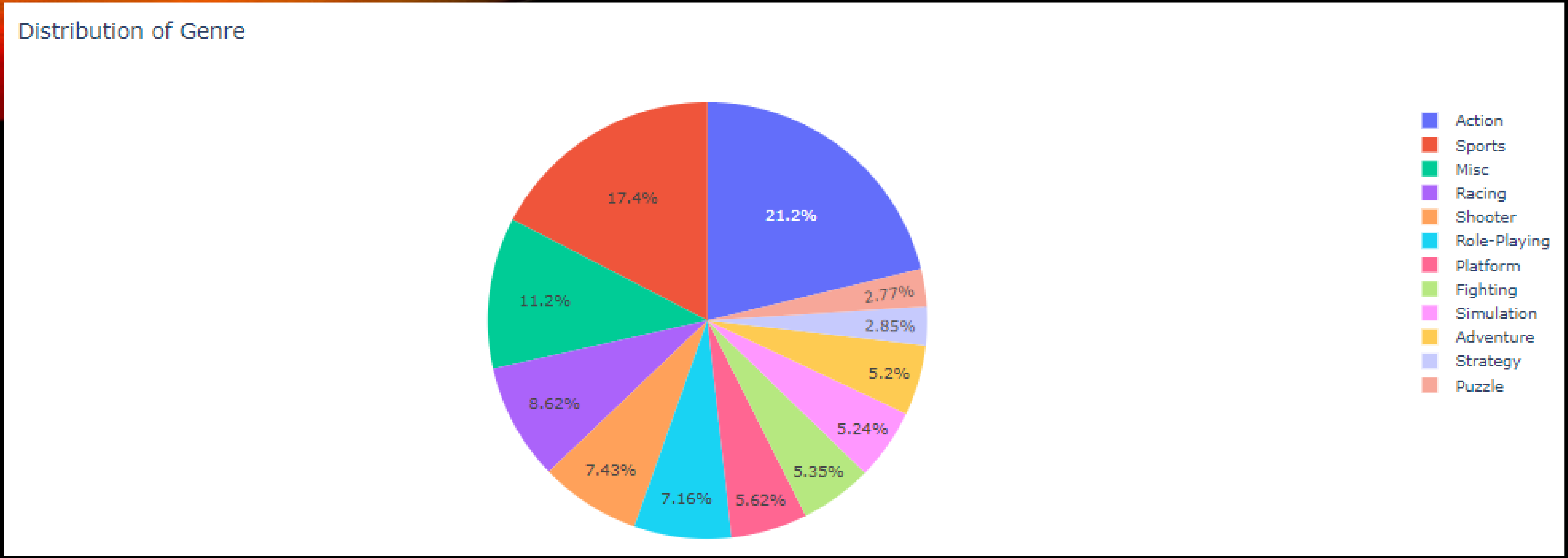
Without
JP_Sales

After removing JP_Sales the MSE got significanty better even at the cost of slightly lower R squared score.

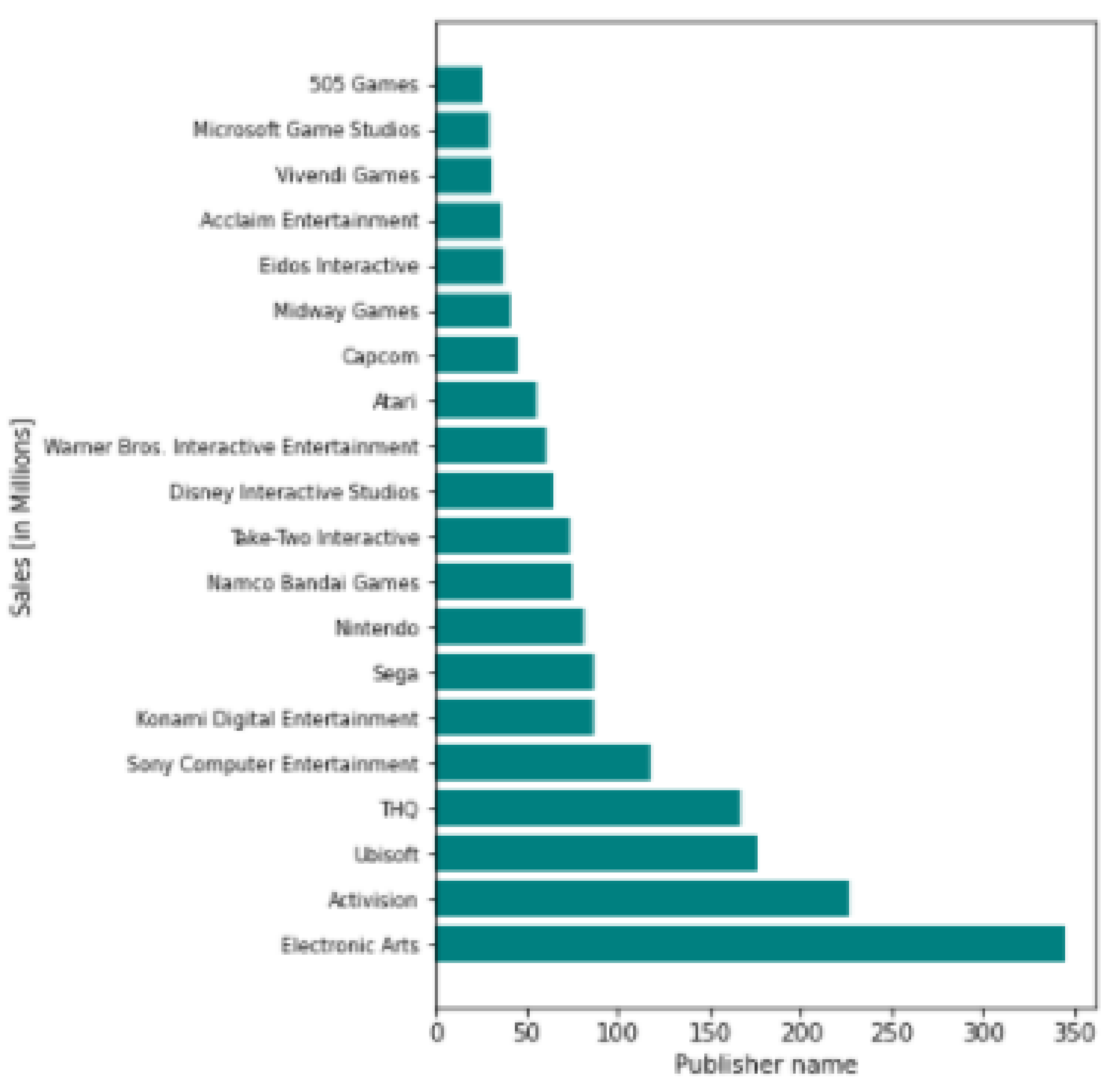
In my opinion this model is more reliable and accurate



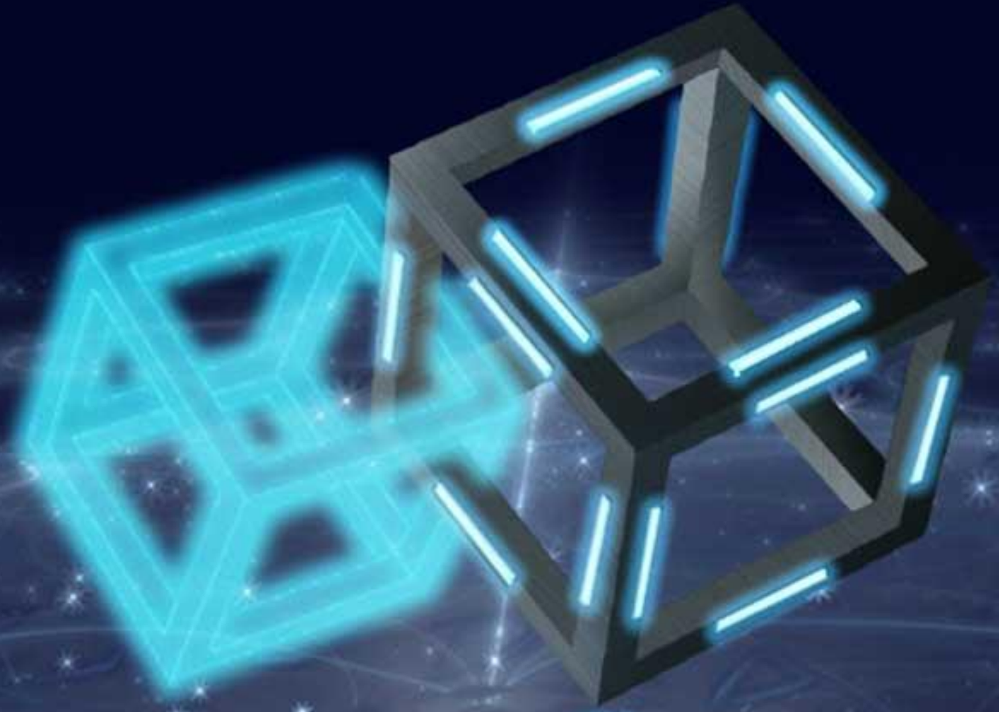
Above is the graph explaining the declining in profit started on 2010 which lasted until the end of the data provided in 2016



The graph above describe the Distribution of Genre provided by the data. In which Action and Sport Games dominates the majority of sales made.



Shows the Game Publisher which bears the most sales and profits the most compared to other which is the EA



THANK YOU

FOR YOUR ATTENTION AND PARTICIPATION