Predicting Video Game Sales

Jacob Miller

Background

Data Interpretation and Video Games Sales Prediction Using Machine Learning Algorithms

Dr.A.Manimuthu, Dr.U.Udhayakumar, A.Cathrine Loura, Dr.Peter Jose P, T D Gowri, Dr.L.Selvam, Dr.S.Roseline Mary

Published: March 8, 2023

Why is predicting video games sales important?

Well, one major reason is that the video game industry is MASSIVE

How massive?...

The video game industry outsales the music industry and the movie industry by a large margin!

Problem Statement

With video games increasing in popularity each year this has a led to a large amount of data collected on people, such as their likes and dislikes of certain games. Machine learning can be used to figure out sales predictions and upcoming business ventures in the video game industry. Having a way for developers, businesses, investors, and consumers to have accurate game sale trends can provide great marketing strategies, which can also be used to figure out if a new game will be considered a “hit” or not.

My Motivation

- My Bachelor degree is in Game Development so I thought it would be interesting to find a topic that relates to video games. I decided on forecasting video games because I thought it would be cool if I could implement an algorithm that would decide if a new game would be considered a "hit" or not.

So… What is considered a Hit game?

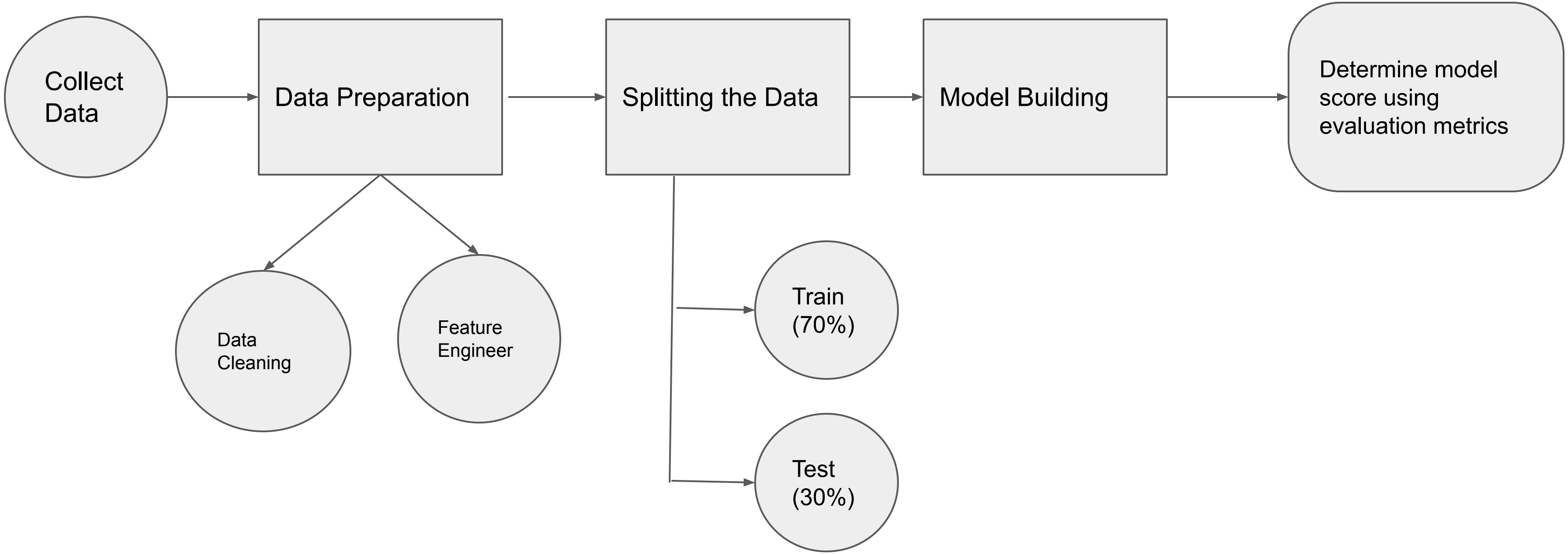
A game is considered a “Hit” if it has sold at least 1 million copies!

Summary of the Method

The method chosen in the paper was Multiple Regression and Random Forest, since they both provided the highest accuracy among all the tested algorithms.

- Some of the dataset features: Overall Sales, NA Sales, Platform, Genre, Year, Name, Publisher, Critic Rating, User Rating, etc.

Methodology



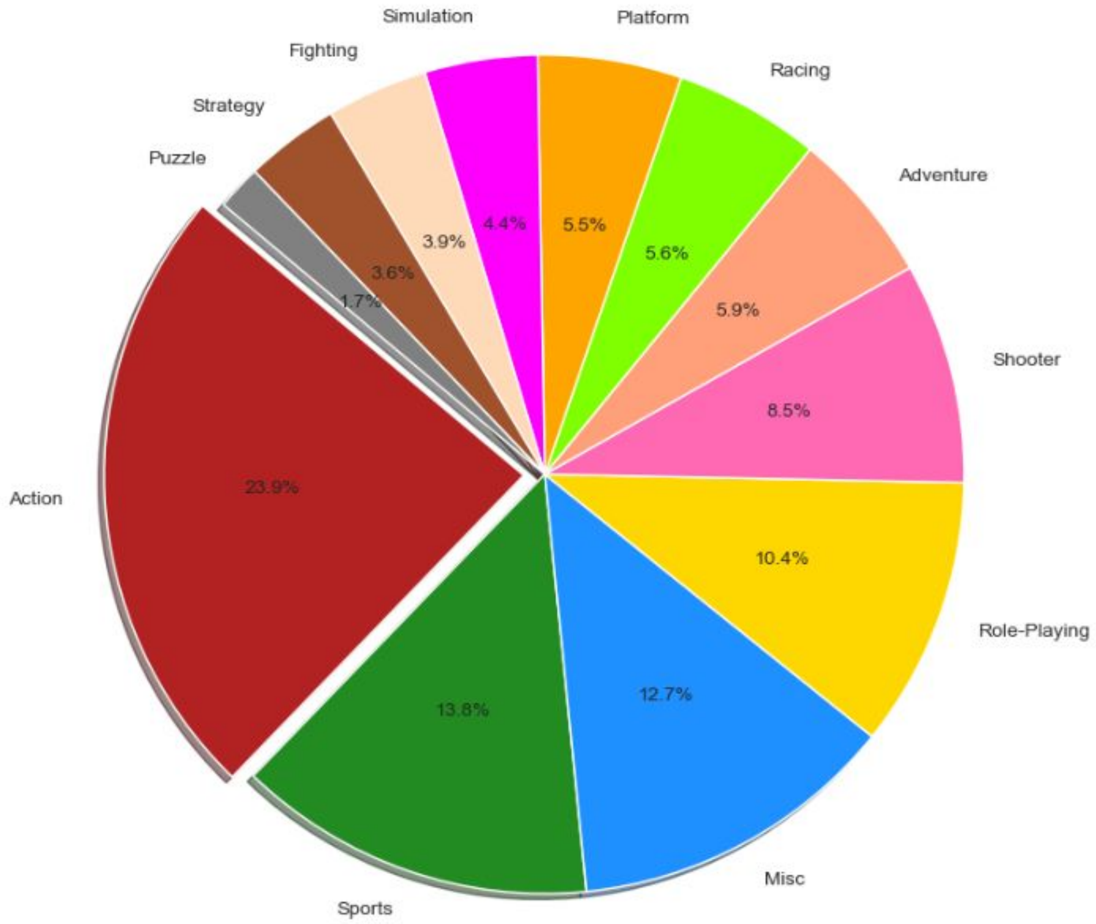
Paper Results

Regression Algorithms	**R-squared Score**
Random Forest Regressor	98%
Decision Tree Regressor	56%
Support Vector Regressor	64%
Multiple Regressor	99%

Calculating Most Played Genre

```
gen_amount = df_game['Genre'].value_counts()
colors = ("firebrick", "forestgreen","dodgerblue", "gold", "hotpink", "lightsalmon", "chartreuse", "ora
explode = (0.1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
gen_label = 'Action', 'Sports', 'Misc', 'Role-Playing', 'Shooter', 'Adventure', 'Racing', 'Platform', '
plt.pie(gen_amount, colors=colors, explode = explode, labels = gen_label,
autopct='%1.1f%%', shadow=True, startangle=140, radius = 2)
plt.title("Most Used Genres\n"+" \n"+" \n"+" ", fontsize = 40)
plt.show()
print('Genre Amount:')
print(" ")
df_game['Genre'].value_counts()
```

Most Used Genres



My implementation with the help of Kaggle

Predicting Hit Games (sales > 1 million)

```
df_names = df_game[['Name','Platform','Genre','Publisher','Year_of_Release','Critic_Score','Global_Sale
df_game = df_game.dropna().reset_index(drop=True)
df_hits = df_game[['Platform','Genre','Publisher','Year_of_Release','Critic_Score','Global_Sales']]
df_hits['Hit'] = df_hits['Global_Sales']
df_hits.drop('Global_Sales', axis=1, inplace=True)
```

```
def hit(sales):
    if sales >= 1:
        return 1
    else:
        return 0

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

Output Features

	Platform	Genre	Publisher	Year_of_Release	Critic_Score	Hit
0	Wii	Sports	Nintendo	2006.0	76.0	1
1	Wii	Racing	Nintendo	2008.0	82.0	1
2	Wii	Sports	Nintendo	2009.0	80.0	1
3	DS	Platform	Nintendo	2006.0	89.0	1
4	Wii	Misc	Nintendo	2006.0	58.0	1
5	Wii	Platform	Nintendo	2009.0	87.0	1
6	DS	Racing	Nintendo	2005.0	91.0	1
7	Wii	Sports	Nintendo	2007.0	80.0	1
8	X360	Misc	Microsoft Game Studios	2010.0	61.0	1
9	Wii	Sports	Nintendo	2009.0	80.0	1
10	PS3	Action	Take-Two Interactive	2013.0	97.0	1
11	PS2	Action	Take-Two Interactive	2004.0	95.0	1
12	DS	Misc	Nintendo	2005.0	77.0	1

Predicting with Random Forest Classification

```
rnd_forest = RandomForestClassifier(random_state=2).fit(X_train, y_train)
y_predict = rnd_forest.predict_proba(X_test)
print("Validation accuracy: ", sum(pd.DataFrame(y_predict).idxmax(axis=1).values == y_test)/len(y_test)
```

```
Validation accuracy:  0.8537938439513243
```

Predicting with Logistic Regression

```
log_reg = LogisticRegression().fit(X_train, y_train)
log_pred = log_reg.predict_proba(X_test)
print("Validation accuracy: ", sum(pd.DataFrame(log_pred).idxmax(axis=1).values
                                   == y_test)/len(y_test))
```

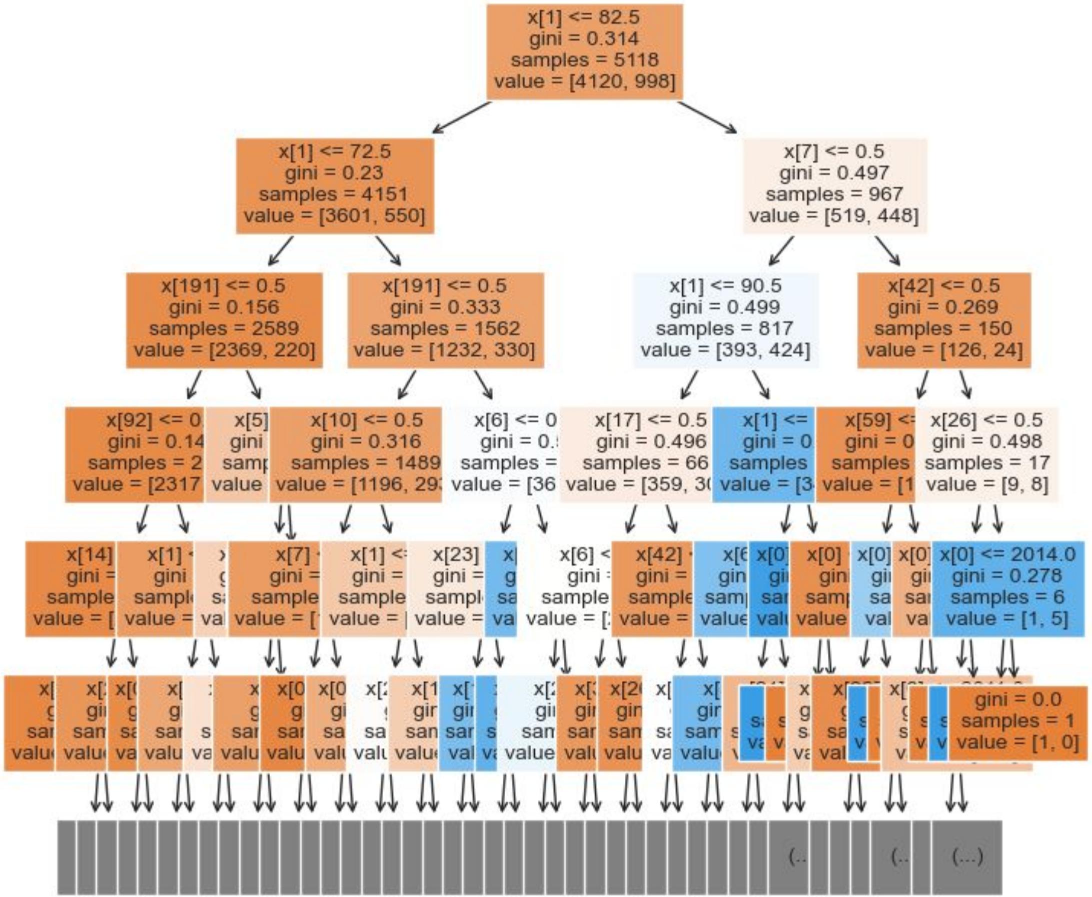
```
Validation accuracy:  0.85773085182534
```

Predicting with Decision Trees

```python
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier

X_train, X_test, y_train, y_test = train_test_split(X, y,  random_state=0)

model = DecisionTreeClassifier()

model.fit(X_train, y_train)

train_score = model.score(X_train, y_train)
test_score = model.score(X_test, y_test)
print("Train Accuracy: {}, Test Accuracy: {}".format(train_score, test_score))
```

```
Train Accuracy: 0.9969929836284664, Test Accuracy: 0.8191382765531062
```

```python
from sklearn.tree import plot_tree
plt.figure(figsize=(8,8))
#reeplot = plot_tree(model, filled=True)
treeplot = plot_tree(model, filled=True, max_depth = 5, fontsize = 10)
```



Results from 3 Tests

1. Logistic Regression: 85.8%

2. Random Forest: 85.3%

3. Decision Tree: 82%

Let’s now look at the probabilities of games that still have a chance of becoming a hit!

Calculating Potential Hits

```
not_hit = df_copy[df_copy['Hit'] == 0]
```

```
# Refernce: https://www.kaggle.com/code/ignacioch/predicting-vg-hits-1-million-sales-with-lr-rfc/notebo
not_hit_copy = not_hit
y = not_hit_copy['Hit'].values
not_hit_copy = not_hit_copy.drop(['Hit'], axis = 1)
X = not_hit_copy.values

pred = log_reg.predict_proba(X)
df_names = df_names[df_names['Global_Sales'] < 1] # < 1 means not a hit
df_names['Hit_Probability'] = pred[:,1]
```

```
df_names = df_names[df_names['Year_of_Release'] == 2016] # 2016
df_names.sort_values(['Hit_Probability'], ascending=[False], inplace = True)
df_names = df_names[['Name', 'Platform', 'Hit_Probability']]
```

10 Highest Probabilities of becoming a Hit (2016)

```
: df_names[:10].reset_index(drop=True)
```

:

	Name	Platform	Hit_Probability
0	Titanfall 2	PS4	0.824953
1	Dishonored 2	PS4	0.714073
2	Fast Racing Neo	WiiU	0.713176
3	Kirby: Planet Robobot	3DS	0.698390
4	BioShock The Collection	PS4	0.688588
5	Titanfall 2	XOne	0.681053
6	Plants vs. Zombies: Garden Warfare 2	PS4	0.653032
7	Deus Ex: Mankind Divided	PS4	0.645612
8	Dishonored 2	XOne	0.587360
9	Skylanders Imaginators	PS4	0.573406

10 Lowest Probabilities of becoming a Hit (2016)

```
df_names[:-11:-1].reset_index(drop=True)
```

	Name	Platform	Hit_Probability
0	Bus Simulator 16	PC	0.000816
1	RollerCoaster Tycoon World	PC	0.000894
2	Dino Dini's Kick Off Revival	PS4	0.001195
3	Homefront: The Revolution	PC	0.002020
4	The Technomancer	PC	0.002115
5	7 Days to Die	XOne	0.002459
6	Pro Cycling Manager 2016	PC	0.002903
7	Sherlock Holmes: The Devil's Daughter	PC	0.003011
8	Pro Evolution Soccer 2017	PC	0.003090
9	Agatha Christie: The ABC Murders	PC	0.003246

Results

Though most of the probabilities were pretty accurate, there obviously were some outliers. For example, 7 Days To Die only had a .002% chance of becoming a hit, but obviously that's wrong since it sold well over a million copies. However this did take multiple years to do so.

I question how a platform impacts how likely a game will become a hit. For example if your game is on the Xbox game-pass it will most likely have more sales compared to a similar game that is not on the game-pass.

Future Testing

I think having an up-to date dataset without any missing values could be very interesting to test with. If I had a dataset that was always updating you could make charts and tables in Tableau that showcase real-time game sale trends.

I also think taking the cost of the game into consideration for future testing could change some of the results. Games that cost a lot of money can have low sales if it is not marketed correctly.

Overall I had a lot of fun doing this project and gained some valuable information while doing so and would enjoy expanding on this in the future!

References

Main Paper: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4382007

Extra Paper: https://www.jetir.org/papers/JETIR1907H50.pdf

Kaggle:
https://www.kaggle.com/code/hamizanfirdaus/machine-learning-of-video-games-sales

Kaggle Reference:
https://www.kaggle.com/code/ignacioch/predicting-vg-hits-1-million-sales-with-lr-rfc/notebook?scriptVersionId=0