DATA SCIENCE
PROJECT
Dizon, Pagaduan, Quindoza

I. Data Gathering

Our source of the dataset comes from the website Kaggle, which we can view a preview of the data set in tabular form and also view the description of the dataset, we chose Kaggle as well since it is a hub of many data science projects and has a wide variety of options and searchability for the right dataset we need. Aside from that, the website also states the size of the dataset and how many columns it has that eases our selection of dataset.

As for rows:

```
In [1]: import pandas as pd

In [16]: df = pd.read_csv('vgsales.csv')

In [17]: len(df)
Out[17]: 16598

In [ ]:
```

Or

```
In [20]: import pandas as pd
         import numpy as np

In [21]: video_game=pd.read_csv("vgsales.csv")

In [22]: video_game.shape
Out[22]: (16598, 11)
```

Since the dataset requires 10 columns and 2000 rows, and Kaggle does not show rows, we can view the number of rows a data set has through the following Python Code above which are:

`len(df)` `video_game.shape`

As seen, since our data set now passes the required rows and columns it is now viable for data cleansing discussed in the next chapter. As for this part, we can now say that our dataset has been obtained.

II. Data Cleansing

This part of our data science project discusses how we can make our dataset viable for data analysis and modelling. In order to do this we need to make sure that our dataset has the correct format, values, filters, and corrected tables if ever there are some that need to be merged. For our data set, we have found out that we have non values that need to be replaced with something else so that the dataset can be cleaner and accurate. Specifically the columns of the "video game publisher" and "year" have Nan Values and need to be replaced with "Unknown".

To do this we first bring up our columns and first 10 rows of the data, this gives us the preview of each column we need to check for Nan values.

```
In [20]: import pandas as pd
         import numpy as np

In [21]: video_game=pd.read_csv("vgsales.csv")

In [22]: video_game.shape
Out[22]: (16598, 11)

In [23]: video_game.head(10)
Out[23]:
```

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37
5	6	Tetris	GB	1989.0	Puzzle	Nintendo	23.20	2.26	4.22	0.58	30.26
6	7	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.38	9.23	6.50	2.90	30.01
7	8	Wii Play	Wii	2006.0	Misc	Nintendo	14.03	9.20	2.93	2.85	29.02
8	9	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.59	7.06	4.70	2.26	28.62
9	10	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.28	0.47	28.31

Now that we know our columns, we can use the code below to check if any of the columns have Nan values.

```
video_game['name of column'].isnull().values.any()
```

As we can notice we have ‘Publisher’ and ‘Year’ that has NaN values.

```
In [24]: video_game['Rank'].isnull().values.any()
Out[24]: False

In [25]: video_game['Name'].isnull().values.any()
Out[25]: False

In [26]: video_game['Platform'].isnull().values.any()
Out[26]: False

In [27]: #NaN Values Present
         video_game['Year'].isnull().values.any()
Out[27]: True

In [28]: video_game['Genre'].isnull().values.any()
Out[28]: False

In [29]: #NaN Values Present
         video_game['Publisher'].isnull().values.any()
Out[29]: True

In [30]: video_game['NA_Sales'].isnull().values.any()
Out[30]: False

In [31]: video_game['EU_Sales'].isnull().values.any()
Out[31]: False
```

Using video_game = pd.DataFrame(video_game).fillna("Unknown") We are able to fill in the NaN values we have found above with “Unknown” this resolves any conflict we can have in the future with missing values from the dataset.

```
In [35]: #fill NaN values with "Unknown"
         video_game = pd.DataFrame(video_game).fillna("Unknown")

In [36]: video_game['Publisher'].isnull().values.any()
Out[36]: False

In [37]: video_game['Year'].isnull().values.any()
Out[37]: False

In [38]: #No remaining NaN values present
         video_game.isnull().values.any()
Out[38]: False

In [ ]:
```

The following code above shows checking again the following columns of ‘Publisher’ and ‘Year’ for any NaN, as observed, the output is false and we have now resolved missing values, and have cleansed our data set.

III. Exploratory Data Analysis

Problem Statement: How can Game Companies maximize their sales?

Data Science Questions:

- Which genre is the most successful per each region?
- Which platform is the most successful per each region?
- What are each of the companies' top genres?

Narrative findings of data Inspection:

Since our problem statement revolves around how game companies can maximize sales, we can inspect the following columns and their relation with sales, for example companies can check the relationships between genre and sales per region, platform and sales per region and as well as top genres performers per company.

Column	Data Type
Genre	Categorical data
Year	Categorical data
Platform	Categorical data
Publisher	Nominal data
Other_Sales	Numerical data
JP_Sales	Numerical data
NA_Sales	Numerical data
EU_Sales	Numerical data

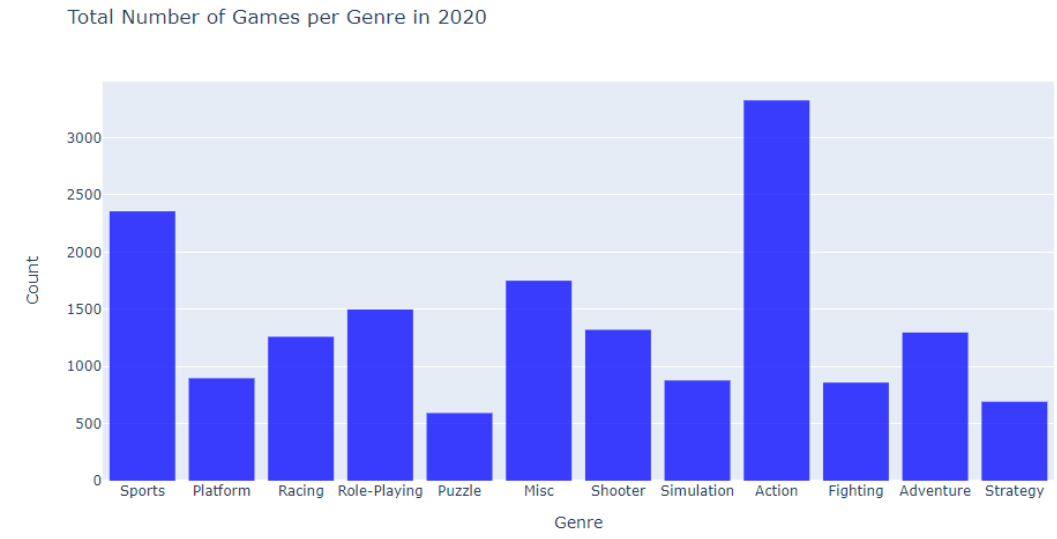
In this part we can also inspect if the following chosen data can be of importance in it's impact to sales so that game companies can get a better understanding of maximizing their sales, on the code in data visualization.

```
In [2]: vg=pd.read_csv("vgsales.csv")
        Max_year = vg["Year"].max()
        Max_year

Out[2]: 2020.0

In [3]: top_genres = go.Histogram(x=vg.Genre,marker=dict(color="blue",line=dict(color='blue', width=2)),opacity=0.75)
        top_genre_layout = go.Layout(
            title='Total Number of Games per Genre in 2020',
            xaxis=dict( title='Genre'),
            yaxis=dict( title='Count'))
        fig = go.Figure(data=[top_genres], layout=top_genre_layout)
        plot(fig)
```

On the above code, we are setting up our histogram to show the total number of games per genre in 2020. Having the X axis set for Genre and Y axis set for a number of games.

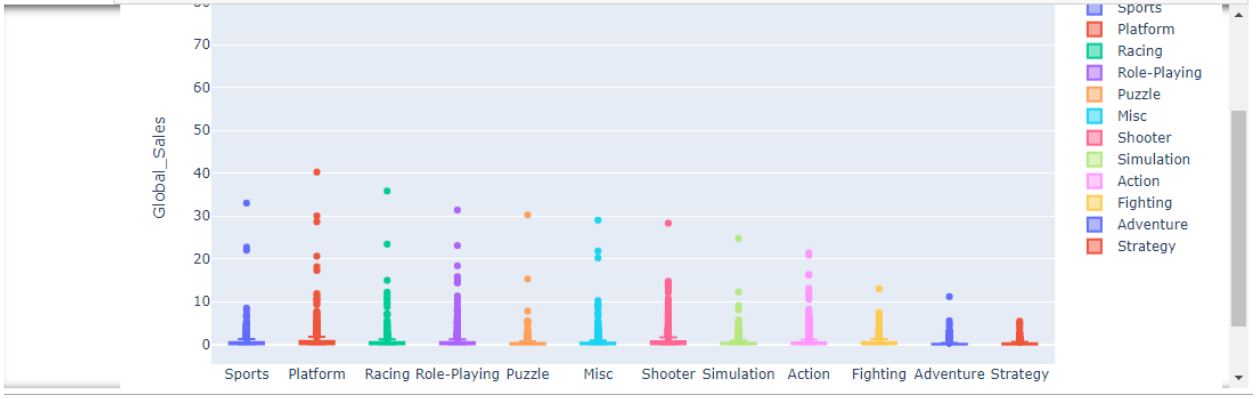


Based on the visualization, we can tell that genres are an important aspect in analyzing sales of games, for example we can see trends that most games released in 2020 were dominating the market with action games and Sports games.

This part emphasizes how genres in relation to sales can help companies form strategies in selecting what kind of game to produce, in the following code and data visualization below, the top genres in relation to global sales yield that top performing sales per genre are sports, platformers, racing, role -playing, puzzle, action and shooter.

```
In [5]: total_genre=pd.DataFrame(vg.groupby("Genre")[["NA_Sales","EU_Sales","JP_Sales","Other_Sales","Global_Sales"]].sum())
        total_genre.reset_index(level=0, inplace=True)
        genre_table=pd.DataFrame(vg["Genre"].value_counts())
        genre_table.reset_index(level=0, inplace=True)
        genre_table.rename(columns={"Genre": "Counts","index":"Genre"}, inplace=True)
        total_genre=pd.merge(total_genre,genre_table,on="Genre")
```

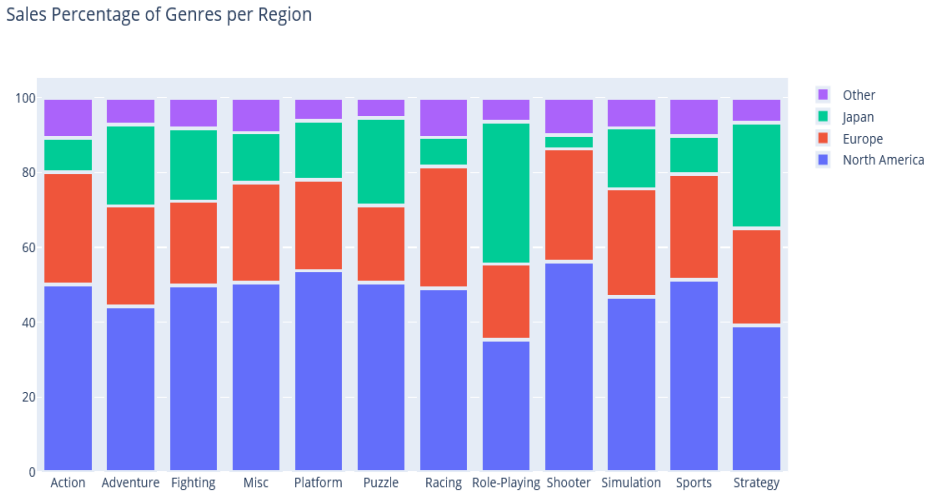
```
In [4]: fig1 = px.box(vg, x="Genre", y="Global_Sales", color="Genre", title='Global Sales Percentage per Genre')
        fig1.show()
```



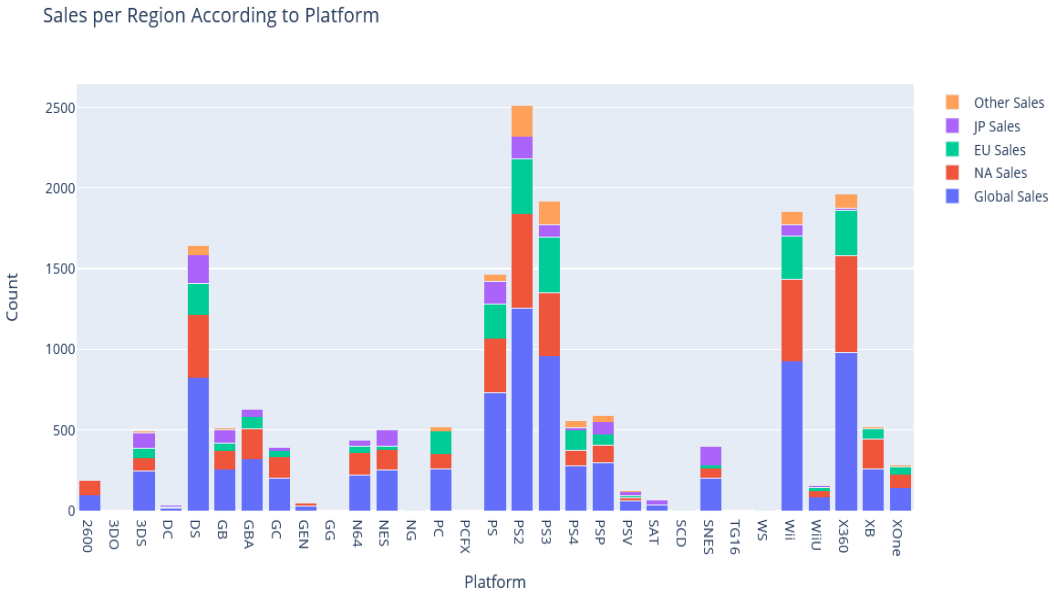
Variable correlations, as well as process in its testing:

The genre and the region are correlated with each other. Some genres, like Action, Adventure, Role-Playing and Racing, would sell better in certain regions, such as in North America, Europe and Japan. For example, shooter games sell especially well in North America as indicated by the 56.16M. However, they do not sell as well in Japan, with a value of 3.69M. This may indicate that video game companies should focus on selling shooter games in North America, while they could opt for other genres when considering selling games in Japan.

```
In [7]: data = [line1, line2, line3, line4]
layout2 = go.Layout(barmode='stack',
                    title='Sales Percentage of Genres per Region')
fig2 = go.Figure(data=data, layout=layout2)
plot(fig2)
```

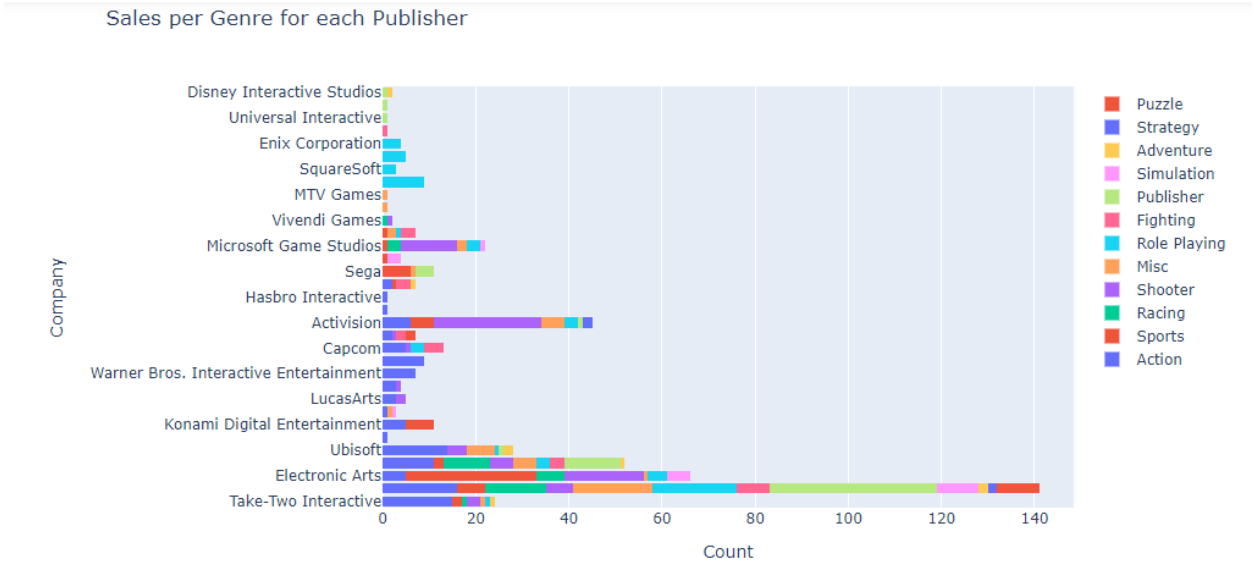


The platform and the region are correlated with each other. This is due to the fact that higher sales in a certain platform in a region may indicate the customers from such an area may prefer such a platform. For example, the PS2 sold well globally, with 1,255.64 in global sales. However, it did not sell as well in Japan, acquiring 139.2 in global sales. Considering this information, game companies could opt to avoid creating games exclusive to PS2, if they are planning to sell their products in Japan.

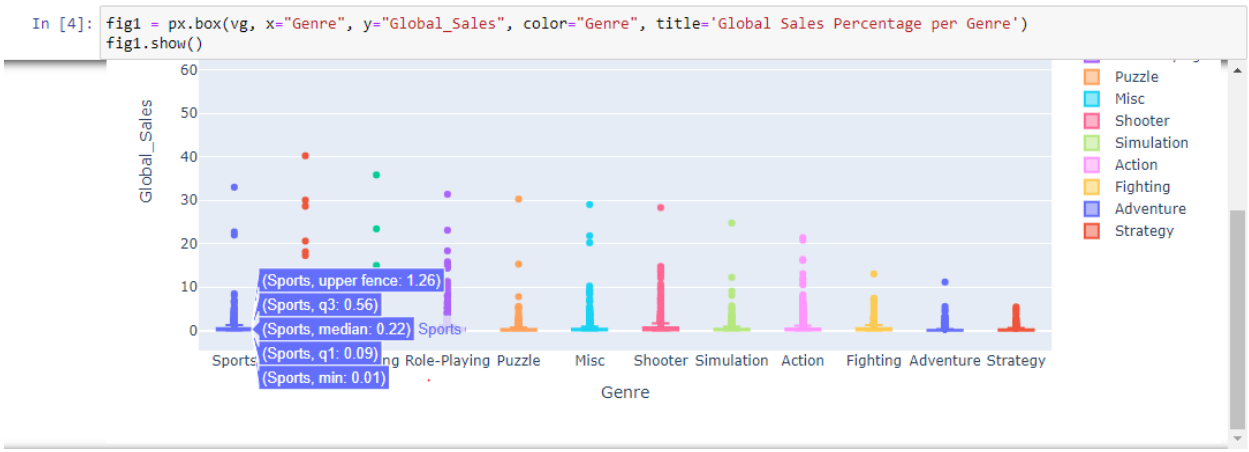


The most popular genres of each company could indicate which genres or type of video games they should focus on. More sales in a genre indicates that the game company is more adept in creating games of this kind, and they could create more games in the same genre.

Though less sales in a certain genre does not merely indicate the company should stop making games of this kind. Instead, the company could see the lack of sales in that genre as an opportunity to improve in said category, and as an indicator that they should improve their games there. An example of this is Microsoft game studios. A value of 12 in Shooter indicates the company creates better shooter games compared to their Simulation games, which acquired a value of 1.



In addition, to the explained correlations above, the implementation of descriptive statistics were also applied in the Notebook. As seen on the code below and data visualization, the following code indicates the average sales per region and then represented via formatted bar for the visuals. As such, we can see that the following box diagram indicates that there are averages such as the median of sport genre sales and the same format appears for the rest of the other genres as well.



```
In [6]: #Represent sales values as median
x=total_genre.Genre
NA=list(total_genre["NA_Sales"]/total_genre["Global_Sales"]*100)
EU=list(total_genre["EU_Sales"]/total_genre["Global_Sales"]*100)
JP=list(total_genre["JP_Sales"]/total_genre["Global_Sales"]*100)
Other=list(total_genre["Other_Sales"]/total_genre["Global_Sales"]*100)
#formats a bar which enumerates the median of sales per each region
line1 = go.Bar(
    x=x,
    y=NA,
    name="North America" ,
    marker=dict(line=dict(width=3)))
line2 = go.Bar(
    x=x,
    y=EU,
    marker=dict(line=dict(width=3)),
    name = "Europe",
    )
line3 = go.Bar(
    x=x,
    y=JP,
    marker=dict(line=dict(width=3)),
    name = "Japan",
    )
line4 = go.Bar(
    x=x,
    y=Other,
    marker=dict(line=dict(width=3)),
    name = "Other")
```

Process of Feature extraction:

```
In [20]: import pandas as pd
         import numpy as np

In [21]: video_game=pd.read_csv("vgsales.csv")

In [22]: video_game.shape
Out[22]: (16598, 11)

In [23]: video_game.head(10)
```

Out[23]:

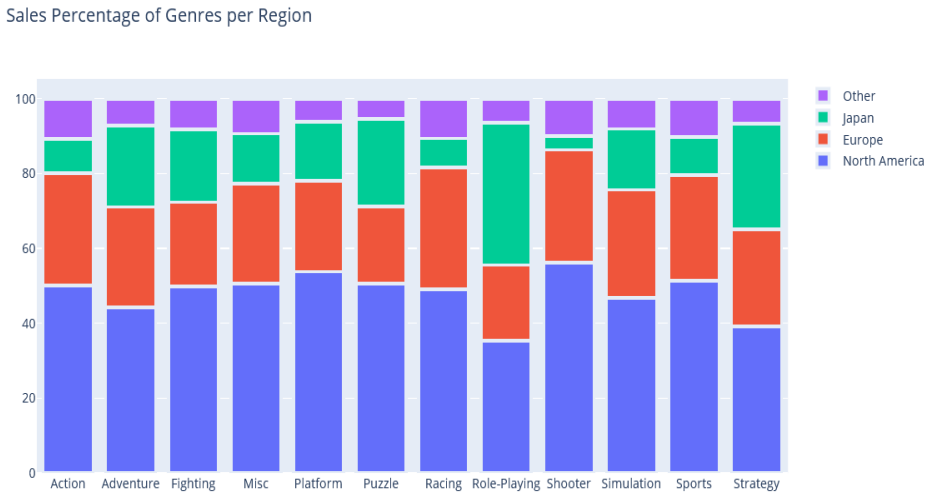
	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37
5	6	Tetris	GB	1989.0	Puzzle	Nintendo	23.20	2.26	4.22	0.58	30.26
6	7	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.38	9.23	6.50	2.90	30.01
7	8	Wii Play	Wii	2006.0	Misc	Nintendo	14.03	9.20	2.93	2.85	29.02
8	9	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.59	7.06	4.70	2.26	28.62
9	10	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.28	0.47	28.31

The table is already filtered based on the various data that describes the video game sales. Each column represents the game's name, the associated platform, the year it was released, its genre, the publisher, their sales in North America, Europe, Japan, Other countries, and lastly their global sale. The rows are arranged in descending order based on the global sales of each individual game so essentially, the table represents the top games based on their overall sales in the market.

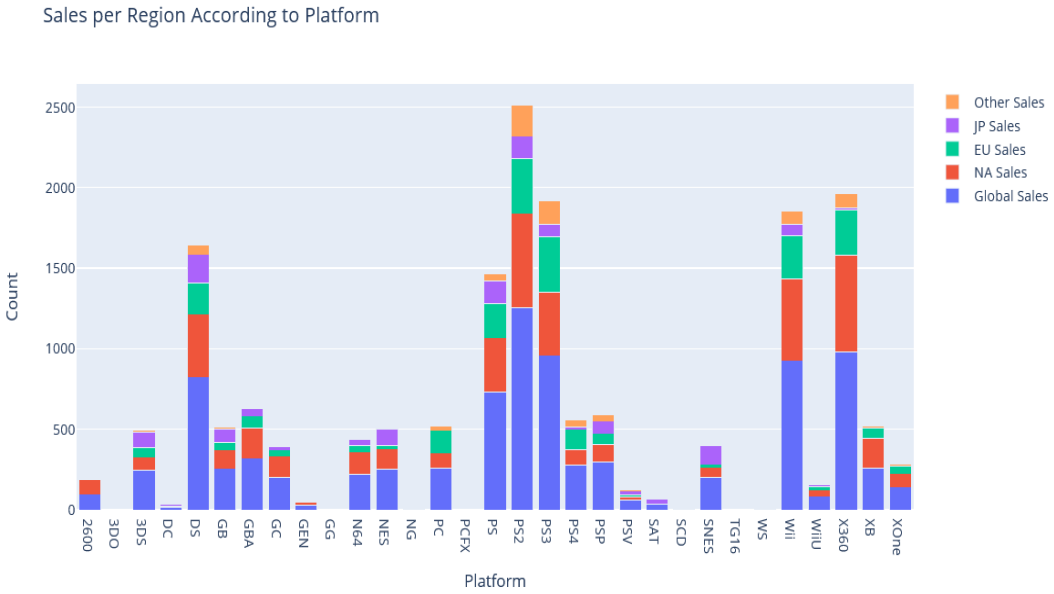
Data visualization:

Some genres sell better in certain regions than in others. By knowing and visualizing which genres sell better in other regions, video games companies have a better idea on what genres they should pursue in a certain region. This could also be an opportunity for video game companies to try to introduce or popularize certain genres in a region, especially one that the people there seldom see or encounter. By introducing something new or relatively unknown, the video game companies could be creating a wonderful business opportunity.

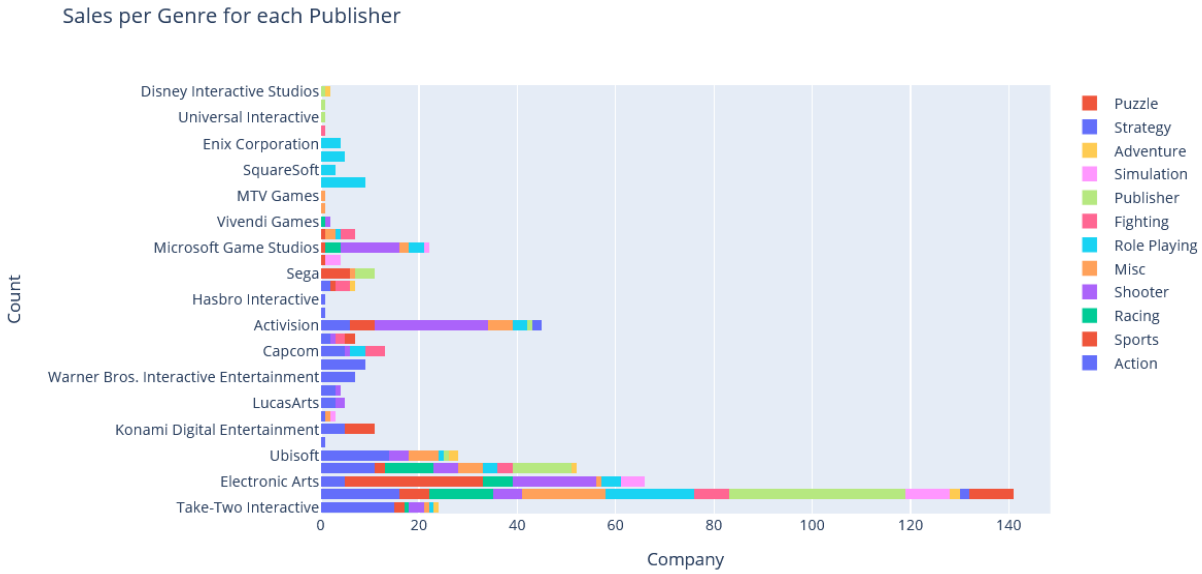
```
In [7]: data = [line1, line2, line3, line4]
        layout2 = go.Layout(barmode='stack',
                            title='Sales Percentage of Genres per Region')
        fig2 = go.Figure(data=data, layout=layout2)
        plot(fig2)
```



Regarding platforms and regions, higher sales in a certain platform in a certain region may indicate the popularity and preference of such a platform in such an area. This is important for video games companies because they could focus on creating games specifically for the mainstream platform, or try to introduce games in the less popular platforms as a possible business opportunity.



The most popular genres of each company is a possible indicator which genre of video games the company could opt to focus on. More sales in a certain genre could indicate that the videogame company should focus on such a category. By making more games of that type, they could sell more games. However, less sales in a certain genre could also be an opportunity for the videogame company to opt to improve their games in that genre.



IV. Data Modelling

Relevant Features

Based on the data science question presented in the previous chapter, we will be focusing as well on the presented problem statement, which is how can game companies maximize their sales. In which the related data science question is ***which genre is the most successful for each region.***

Based on the selected data science question that will be used for prediction, we will be focusing on the values that will show predictions revolving around it such as **Genre, NA_Sales, EU_Sales, JP_Sales, Other_Sales, and Global_Sales.**

The following code below shows the process of reducing the dimensionality of your dataset based on the following relevant values/features that are needed for our data modelling.

```
In [3]: vg=pd.read_csv("vgsales.csv")
        #Remove unnecessary columns
        vg.drop(columns = ['Year', 'Publisher', 'Platform', 'Rank'], inplace = True)
```

Process Discussion

For our Model training, we have opted for forecasting/prediction, to help aid companies on which kind of genre to make in order to maximize sales per region. To begin, columns were removed first so that they can be transferred into arrays as shown in the code below, this is done so that the algorithm can predict data based on the other columns that are available.

```
In [5]: #remove NA region sales columns
        X = vg.iloc[:, :].values
        X = np.delete(X, 2, 1)
        y = vg.iloc[:, 2:3].values
```

```
In [6]: #transfer data into different arrays for testing
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.33, random_state = 0)
        sales_training_set = X_train[:, 0]
        sales_test_set = X_test[:, 1]

        #remove 1st column
        X_train = X_train[:, 1:]
        X_test = X_test[:, 1:]
```

After that, conversion of the categorical data to numerical dummy values is conducted and as well as splitting the data set into training at test sets The next lines of code feature the use of gradient boosting regression algorithm for better prediction. Which are shown in the following codes below:

In [7]:
```
#converts all categorical data to numerical dummy values
ct = ColumnTransformer(transformers = [('encoder', OneHotEncoder(sparse=False, handle_unknown='ignore'), [0, 4])], remainder = 'passthrough')
X_train = ct.fit_transform(X_train)
X_test = ct.transform(X_test)
```

In [8]:
```
model = XGBRegressor(n_estimators = 500, learning_rate= 0.08)
model.fit(X_train, y_train)
```

Out[8]:
```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
             importance_type='gain', interaction_constraints='',
             learning_rate=0.08, max_delta_step=0, max_depth=6,
             min_child_weight=1, missing=nan, monotone_constraints='()',
             n_estimators=500, n_jobs=8, num_parallel_tree=1, random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
             tree_method='exact', validate_parameters=1, verbosity=None)
```

After applying the boosting regression algorithm, a table showing **Actual_Region_Sales** and the **Predicted_Region_Sales** is made.

In [9]:
```
#Predict values
y_predict = model.predict(X_test)

# comparing predicted and actual region sale
sales_test_set = sales_test_set.reshape(-1, 1)
y_predict = y_predict.reshape(-1, 1)
predict_table = np.concatenate([sales_test_set, y_predict, y_test], axis = 1)
predict_table = pd.DataFrame(predict_table, columns = ['Genre', 'Predicted_NA_Sales', 'Actual_NA_Sales'])
```

In [10]:
```
predict_table
```

Out[10]:

	Genre	Predicted_NA_Sales	Actual_NA_Sales
0	Shooter	1.007071	0.47
1	Simulation	0.11688	0.08
2	Action	0.11688	0.09
3	Racing	0.11944	0.08
4	Simulation	0.373314	0.0
...	...	...	...
5473	Action	0.088815	0.08
5474	Strategy	0.052173	0.13
5475	Puzzle	1.276912	1.63
5476	Action	0.041505	0.04
5477	Misc	0.054319	0.05

5478 rows × 3 columns

Since we have used regression, we cannot use precision, recall, and Fscores as they are for categorical data, with that reason we will be using we would be using R2_scoring and root mean squared error in which:

R2 score: closer to 1 = more accurate
Root mean square: less score = less error

In [11]:
```
score1 = r2_score(y_test, y_predict)
score2 = math.sqrt(mean_squared_error(y_test, y_predict))
print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")
```

```
R2 Score: 0.5112943388975608, Root Mean Square Error: 0.5672381635163921
```

The following is repeated for each region per **NA_Sales, JP_sales, EU_Sales,** and **Other_Sales**.

Libraries

```
In [1]: #Packages to install
        #  pip install xgboost
        #  pip install -U scikit-learn
```

To make use of the libraries, one has to install the following packages first. The first package is xgboost, which is necessary as an implementation of gradient boosted decision trees designed for speed and performance. The next package is scikit-learn. This contains a lot of efficient tools for machine learning and statistical modelling.

```
In [2]: import pandas as pd
        import numpy as np
```

Pandas is a powerful and flexible open source data analysis and manipulation tool. It is a software library, offering data structures and operations for manipulating numerical tables and time series. Numpy is a library used when working with arrays. It can also be used in the domains of linear algebra, fourier transform, and matrices.

```
from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn import preprocessing
```

The following libraries are used for splitting data into training and tests.

```
from xgboost import XGBRegressor
```

This is used for prediction. Specifically, it is used as a gradient boosting regression algorithm.

```
from sklearn.metrics import r2_score, mean_squared_error
import math
```

These are used to test the model scores.

Validation and Evaluation measurement

The goal of the research is to predict what genres will sell well in each region.

In [18]:
```
predict_table
```

Out[18]:

	Genre	Predicted_NA_Sales	Actual_NA_Sales
0	Shooter	1.007071	0.47
1	Simulation	0.11688	0.08
2	Action	0.11688	0.09
3	Racing	0.11944	0.08
4	Simulation	0.373314	0.0
...	...	...	...
5473	Action	0.088815	0.08
5474	Strategy	0.052173	0.13
5475	Puzzle	1.276912	1.63
5476	Action	0.041505	0.04
5477	Misc	0.054319	0.05

5478 rows × 3 columns

In [19]:
```
#Since we have used regression, we cannot use precision, recall, and Fscores as they are for categorical data
#we would be using R2_scoring and root mean squared error
score1 = r2_score(y_test, y_predict)
score2 = math.sqrt(mean_squared_error(y_test, y_predict))
#R2 score: closer to 1 = more accurate
#Root mean square: less score = less error
print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")
```

```
R2 Score: 0.5112943388975608, Root Mean Square Error: 0.5672381635163921
```

In North America, the predictions for the games were half accurate. The model predicted that Action, Shooter and Sports are the most popular genres. The R2 score is half as close to one, meaning it is more or less accurate. The root mean square error is not as reliable.

In [21]:
```
predict_table
```

Out[21]:

	Genre	Predicted_EU_Sales	Actual_EU_Sales
0	Shooter	0.529979	0.53
1	Simulation	0.059996	0.06
2	Action	0.059996	0.06
3	Racing	0.049997	0.05
4	Simulation	0.289996	0.29
...	...	...	...
5473	Action	0.000011	0.0
5474	Strategy	0.029999	0.03
5475	Puzzle	0.519985	0.52
5476	Action	0.000011	0.0
5477	Misc	0.019999	0.02

5478 rows × 3 columns

In [22]:
```
score1 = r2_score(y_test, y_predict)
score2 = math.sqrt(mean_squared_error(y_test, y_predict))
print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")
```

```
R2 Score: 0.9995879255689021, Root Mean Square Error: 0.00906111705440086
```

In Europe, the prediction is accurate. The model predicted that Action, Shooter and Sports are the most popular genres. The R2 score is closer to one, meaning it is more accurate. The root mean square error is low, meaning the chances of error are low as well.

In [27]:
```
predict_table
```

Out[27]:

	Genre	Predicted_Japan_Sales	Actual_Japan_Sales
0	Shooter	0.029996	0.03
1	Simulation	0.189985	0.19
2	Action	0.109991	0.11
3	Racing	0.000007	0.0
4	Simulation	0.000007	0.0
...	...	...	...
5473	Action	0.000007	0.0
5474	Strategy	0.000007	0.0
5475	Puzzle	0.349979	0.35
5476	Action	0.000007	0.0
5477	Misc	0.000007	0.0

5478 rows × 3 columns

In [28]:
```
score1 = r2_score(y_test, y_predict)
score2 = math.sqrt(mean_squared_error(y_test, y_predict))
print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")
```

```
R2 Score: 0.9955743720598962, Root Mean Square Error: 0.0194278898486434
```

In Japan, the prediction is accurate.The model predicted that Action, Role-playing and Sports are the most popular genres. The R2 score is closer to one, meaning it is more accurate. The root mean square error is low, meaning the chances of error are low as well.

```
In [30]: predict_table
```

Out[30]:

	Genre	Predicted_Other_Sales	Actual_Other_Sales
0	Shooter	0.219979	0.22
1	Simulation	0.019997	0.02
2	Action	0.019997	0.02
3	Racing	0.01	0.01
4	Simulation	0.019997	0.02
...	...	...	...
5473	Action	0.01	0.01
5474	Strategy	0.000012	0.0
5475	Puzzle	0.179982	0.18
5476	Action	0.000012	0.0
5477	Misc	0.000012	0.0

5478 rows × 3 columns

```
In [31]: score1 = r2_score(y_test, y_predict)
         score2 = math.sqrt(mean_squared_error(y_test, y_predict))
         print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")
```

```
R2 Score: 0.9859259102602533, Root Mean Square Error: 0.017493648295716355
```

In Other regions, the prediction is accurate. The model predicted that Action, Shooter and Sports are the most popular genres. The R2 score is closer to one, meaning it is more accurate. The root mean square error is low, meaning the chances of error are low as well.

Overall, the predictions are mostly correct, or close the actual values of the sales per region. The R2 scores are pretty close to one, while the root mean square errors are far from one. This shows the data produced by the model is reliable and trustworthy enough.

V. Data Evaluation

Results

The model enabled the researchers to determine which genres have the most potential in each region. For North America, the model predicted that Action, Shooter and Sports are the most successful genres. In Europe, Action, Shooter and Sports were also the most successful genres. In Japan, Role-playing, Action and Sports were predicted as the most successful genres, while in other regions, Action, Shooter and Sports were the most popular genres. The model allowed the researchers to determine that the most successful genres among all regions are Action, Shooter and Sports.

Considering these results, Action, Shooter and Sports are the genres with the most potential to be successful in all regions. Developing games oriented towards this genre has a high degree of success in all regions. Meanwhile, other genres have a less easy time being successful in regions. A game that belongs to these genres must be very appealing to the public to be successful, perhaps requiring more effort than their counterparts from the Action, Shooter or Sports genre.

Accuracy Test Results

The application of precision, recall and Fscores will not be applicable in this project since the said methods for showing accuracy are not applicable to categorical data. Hence, the usage of R2_scoring was applied on the project.

R2_scoring was able to produce results as well for each regional sales, for this project an R2 score and Root Mean Square yielded the accuracy for each region, having the R2 Score being closer to 1 meaning it is more accurate, while the root mean square being less meaning that there is less error in the accuracy per region.

For the results of the scoring of each region:

```
In [11]: score1 = r2_score(y_test, y_predict)
         score2 = math.sqrt(mean_squared_error(y_test, y_predict))
         print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")

         R2 Score: 0.5112943388975608, Root Mean Square Error: 0.5672381635163921
```

NA Sales yielded numbers revolving around 50% of accuracy.

```
In [14]: score1 = r2_score(y_test, y_predict)
         score2 = math.sqrt(mean_squared_error(y_test, y_predict))
         print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")

         R2 Score: 0.9995879255689021, Root Mean Square Error: 0.00906111705440086
```

EU Sales yielded numbers revolving around 90-99% of accuracy.

```
In [17]: score1 = r2_score(y_test, y_predict)
         score2 = math.sqrt(mean_squared_error(y_test, y_predict))
         print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")

         R2 Score: 0.9955743720598962, Root Mean Square Error: 0.0194278898486434
```

JP Sales yielded numbers revolving around 90-99% of accuracy.

```
In [20]: score1 = r2_score(y_test, y_predict)
         score2 = math.sqrt(mean_squared_error(y_test, y_predict))
         print(f"R2 Score: {score1}, Root Mean Square Error: {score2}")

         R2 Score: 0.9859259102602533, Root Mean Square Error: 0.017493648295716355
```

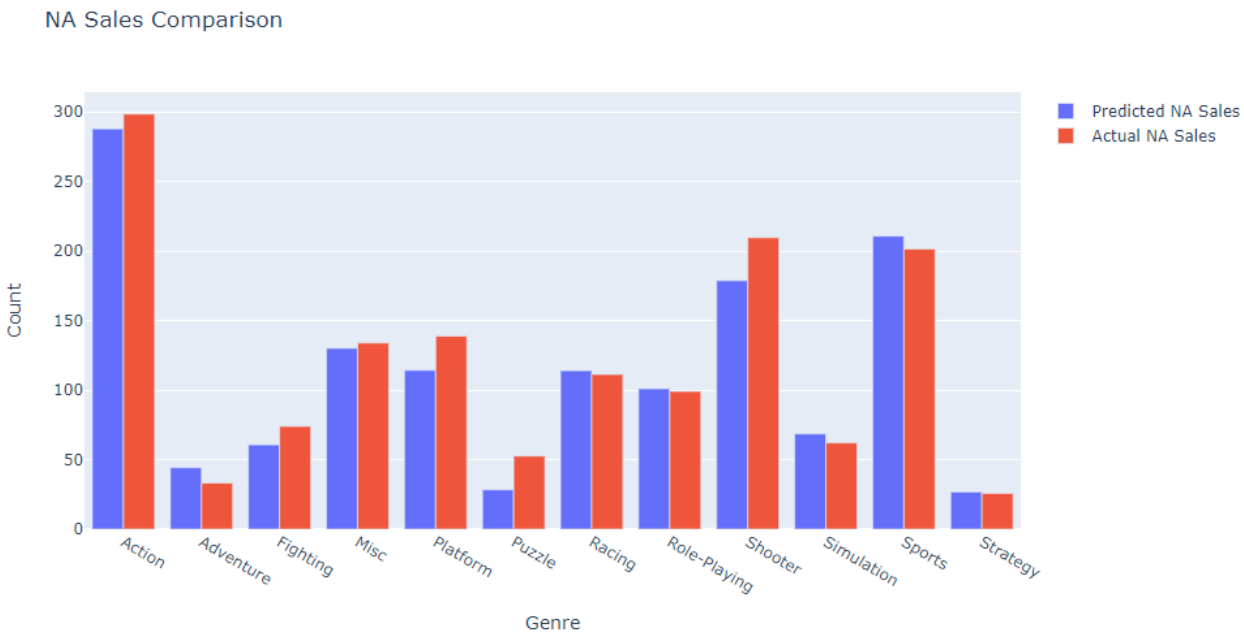
JP Sales yielded numbers revolving around 90-98% of accuracy.

Useful Findings

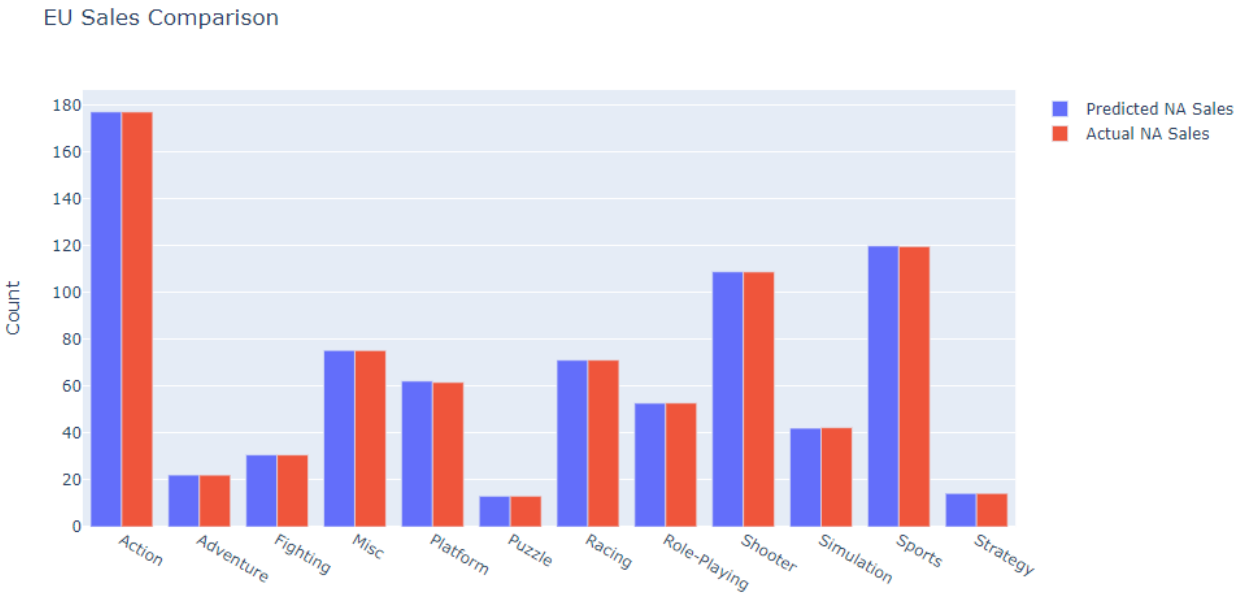
Our problem statement as mentioned in this paper asks ***"How Can Companies Maximize Their Sales?"*** With that said we were able to come up with the following data science questions.

- Which genre is the most successful per each region?
- Which platform is the most successful per each region?
- What are each of the companies' top genres?

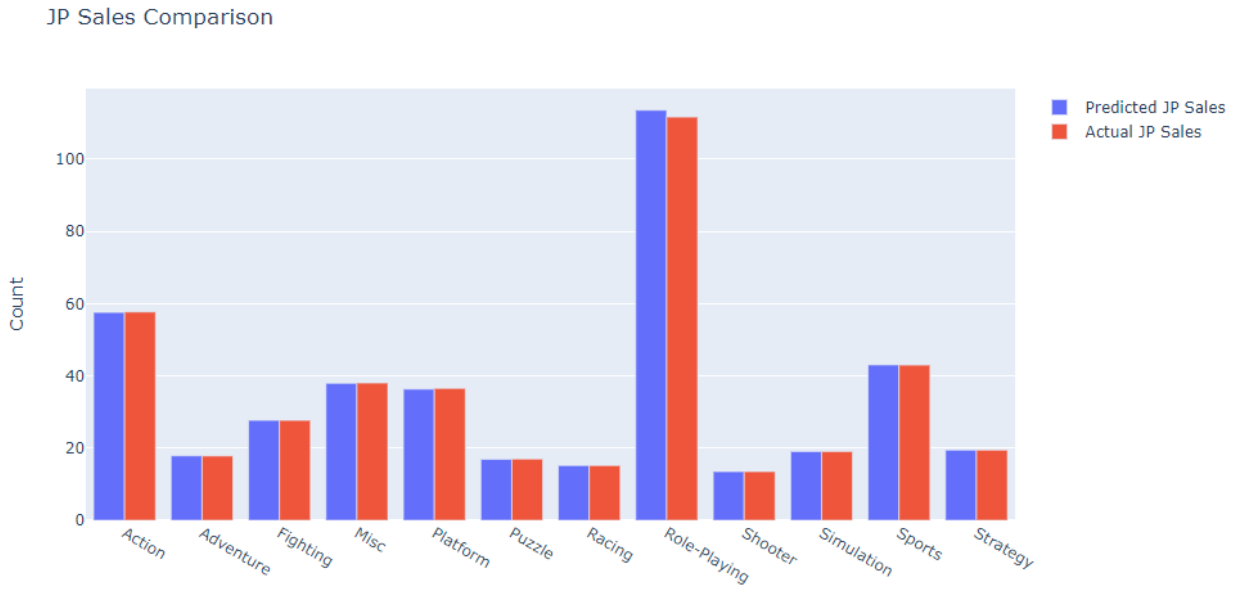
Out of the following questions we chose genres as the data science question to base our modelling and prediction for the problem statement as this also gives unseen future data that can be useful for companies. Having that said, the following are useful learnings for each visualization:



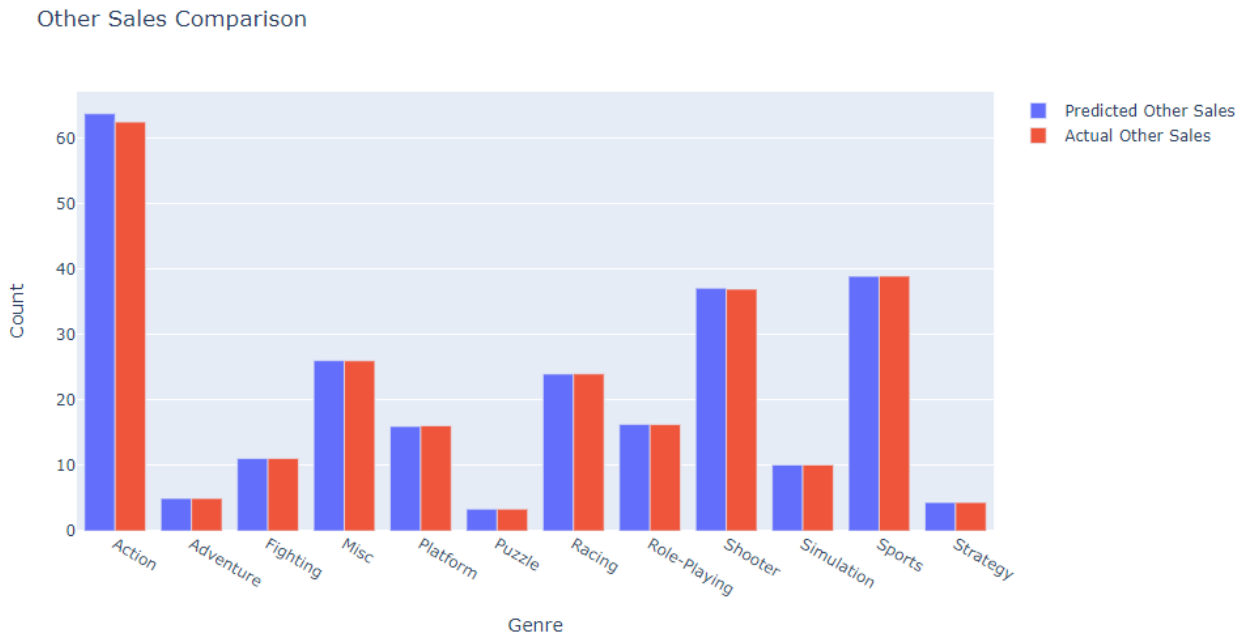
For NA Sales, we can observe that the top genres that could become of potential sales in that region are the following: Action, Shooter, Sports. However ,since their R2 scores provide only 50% accuracy, this cannot be fully trusted by game companies.



For EU Sales, we can observe that the top genres that could become of potential sales in that region are the following: **Action, Shooter and Sports,** which is similar to the NA Region.



For JP Sales, we can observe that the top genres that could become of potential sales in that region are the following: **Role - Playing, Action and Sports.**



For Other_Sales, we can observe that the top genres that could become of potential sales in that region are the following: **Action, Shooter and Sports.** It is observable that the same three are found on EU and NA Sales as well.

Scenarios where Useful Finding can be Helpful

The following genres of **Action, Shooter and Sports** seem to be present in almost all Regions Except for Japanese Sales where the **Role_Playing** Genre is a hit. From the following findings we can create scenarios in which they game companies can utilize the findings to maximize their sales:

- Game Companies can be wary that the following Genres of Action, Shooter and Sports will always yield a hit in sales in whatever region they choose to market their product.

- Role Playing Genres of games are a hit on sales in Japan, hence game companies that are planning to make such games can focus their target customer in Japan.

- Games that do not belong to either Action, Shooter and Sports will have a more difficult time compared to their counterparts in the more successful genres.

- The creation of games that belong to two or more successful genres can be tested to see if their chances of success is higher.

Final Learnings

The dataset shows the video game sales in various regions. The researchers chose this dataset to answer this problem: how can game companies maximize their sales? The researchers decided that by first identifying which genre and platform are the most successful per region and which genres are the top ones of each company, the game companies will have a better idea on what type of game to create that is appealing to the public and has a higher chance of being successful. Then, the researchers decided that if the model could predict which genres are the most successful per region, the video game companies would be able to have a clearer direction on what type of games to create. The researchers made use of various libraries to predict the genres. The model was successful in answering the data science question, predicting that Action, Shooter and Sports are the most successful genres among all the given regions in the dataset.

Determining the most successful genre is a difficult process. It took time and effort to produce the results discussed above, and it was a good learning opportunity to better examine and work with provided data.