Overview.

The gaming industry is certainly one of the thriving industries of the modern age and one of those that are most influenced by the advancement in technology. With the availability of technologies like AR/VR in consumer products like gaming consoles and even smartphones, the gaming sector shows great potential. In this hackathon, you as a data scientist must use your analytical skills to predict the sales of video games depending on given factors. Given are **8 distinguishing factors** that can influence the sales of a video game.

Data Description:-

Train.csv – 3506 observations.
Test.csv – 1503 observations.
Sample Submission – Sample format for the submission.
Target Variable: SalesInMillions

```
#Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings(action='ignore')
```

```
# Read the csv files
input = pd.read_csv("/content/Train.csv")
```

```
#print all columns to understand the dataset
input.head()
```

	ID	CONSOLE	YEAR	CATEGORY	PUBLISHER	RATING	CRITICS_POINTS	USER_POINTS	SalesInMillions
0	2860	ds	2008	role-playing	Nintendo	E	2.833333	0.303704	1.779257
1	731	wii	2012	simulation	Konami Digital Entertainment	E10+	13.200000	1.640000	0.215050
2	495	pc	2019	shooter	Activision	M	4.562500	0.006410	0.534402
3	2641	ps2	2002	sports	Electronic Arts	E	4.181818	0.326923	1.383964
4	811	ps3	2013	action	Activision	M	2.259259	0.032579	0.082671

Data cleaning

```
input.isnull().sum()
```

```
ID                0
CONSOLE           0
YEAR              0
CATEGORY          0
PUBLISHER         0
RATING            0
CRITICS_POINTS    0
USER_POINTS       0
SalesInMillions   0
dtype: int64
```

There are no null values in the dataset. So we can move to the next step of removing unnecessary columns.

From dataset, we can observe that except `id` column, all the other columns play a significant role in final sales of videogames. So it can be dropped.

```
input = input.drop(columns=['ID'])
train, test = train_test_split(input, test_size=0.2, random_state=42, shuffle=True)
```

Descriptive Statistics

Descriptive Statistics

```
train.shape, test.shape
```

```
((2804, 8), (702, 8))
```

```
train.nunique()
```

```
CONSOLE              17
YEAR                 23
CATEGORY             12
PUBLISHER           184
RATING                6
CRITICS_POINTS     1499
USER_POINTS        1875
SalesInMillions    2804
dtype: int64
```

```
#If you are seeing the output below for the first time visit this link
#to understand what the values in each of this rows(mean, std, min, max) actually
```

```
train.describe()
```

	YEAR	CRITICS_POINTS	USER_POINTS	SalesInMillions
count	2804.000000	2804.000000	2804.000000	2804.000000
mean	2008.982168	3.748742	0.403144	2.184942
std	4.286690	3.101958	0.455677	2.578479
min	1997.000000	0.568966	0.000341	0.001524
25%	2006.000000	1.735220	0.063171	0.952236
50%	2009.000000	2.745968	0.229331	1.863315
75%	2012.000000	4.555556	0.600000	2.807032
max	2019.000000	23.250000	2.325000	84.226041

From above table, my first insight is I can create bar charts of **console, year**, **category** and **ratings** columns easily. For other columns I might have to go for some other visual representation since the the number of unique values is high.

EDA

I am first opting for auto EDA packages like pandas-profiling for generating visualisations and there corresponding reports.

```
!pip install https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
```

```
Collecting https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
  Using cached https://github.com/pandas-profiling/pandas-profiling/archive/master.zip (17.8 MB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: scipy<1.12,>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (1.11.3)
Requirement already satisfied: pandas!=1.4.0,<2.1,>1.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (1.5.
Requirement already satisfied: matplotlib<=3.7.3,>=3.2 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (3.7.
Requirement already satisfied: pydantic>=2 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (2.4.2)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (6.0.1)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (3.1.2)
Requirement already satisfied: visions[type_image_path]==0.7.5 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev
Requirement already satisfied: numpy<1.26,>=1.16.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (1.23.5)
Requirement already satisfied: htmlmin==0.1.12 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (0.1.12)
Requirement already satisfied: phik<0.13,>=0.11.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (0.12.3)
Requirement already satisfied: requests<3,>=2.24.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (2.31.0)
Requirement already satisfied: tqdm<5,>=4.48.2 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (4.66.1)
Requirement already satisfied: seaborn<0.13,>=0.10.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (0.12.2
Requirement already satisfied: multimethod<2,>=1.4 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (1.10)
Requirement already satisfied: statsmodels<1,>=0.13.2 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (0.14.
Requirement already satisfied: typeguard<5,>=4.1.2 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (4.1.5)
Requirement already satisfied: imagehash==4.3.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (4.3.1)
Requirement already satisfied: wordcloud>=1.9.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (1.9.2)
Requirement already satisfied: dacite>=1.8 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (1.8.1)
Requirement already satisfied: numba<0.59.0,>=0.56.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (0.56.4
Requirement already satisfied: PyWavelets in /usr/local/lib/python3.10/dist-packages (from imagehash==4.3.1->ydata-profiling==0.0.dev0)
```

```
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (from imagehash==4.3.1->ydata-profiling==0.0.dev0) (9.4
Requirement already satisfied: attrs>=19.3.0 in /usr/local/lib/python3.10/dist-packages (from visions[type_image_path]==0.7.5->ydata-prc
Requirement already satisfied: networkx>=2.4 in /usr/local/lib/python3.10/dist-packages (from visions[type_image_path]==0.7.5->ydata-prc
Requirement already satisfied: tangled-up-in-unicode>=0.0.4 in /usr/local/lib/python3.10/dist-packages (from visions[type_image_path]==0
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2<3.2,>=2.11.1->ydata-profiling==0.
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<=3.7.3,>=3.2->ydata-profilin
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib<=3.7.3,>=3.2->ydata-profiling==0
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib<=3.7.3,>=3.2->ydata-profili
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<=3.7.3,>=3.2->ydata-profili
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib<=3.7.3,>=3.2->ydata-profiling
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<=3.7.3,>=3.2->ydata-profilin
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib<=3.7.3,>=3.2->ydata-prof
Requirement already satisfied: llvmlite<0.40,>=0.39.0dev0 in /usr/local/lib/python3.10/dist-packages (from numba<0.59.0,>=0.56.0->ydata-
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from numba<0.59.0,>=0.56.0->ydata-profiling==0.0.d
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas!=1.4.0,<2.1,>1.1->ydata-profiling==0
Requirement already satisfied: joblib>=0.14.1 in /usr/local/lib/python3.10/dist-packages (from phik<0.13,>=0.11.1->ydata-profiling==0.0.
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic>=2->ydata-profiling==0.0
Requirement already satisfied: pydantic-core==2.10.1 in /usr/local/lib/python3.10/dist-packages (from pydantic>=2->ydata-profiling==0.0.
Requirement already satisfied: typing-extensions>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from pydantic>=2->ydata-profiling==0
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.24.0->ydata-prof
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.24.0->ydata-profiling==0.0.d
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.24.0->ydata-profiling=
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.24.0->ydata-profiling=
Requirement already satisfied: patsy>=0.5.2 in /usr/local/lib/python3.10/dist-packages (from statsmodels<1,>=0.13.2->ydata-profiling==0.
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.2->statsmodels<1,>=0.13.2->ydata-profilin
```

```
from pandas_profiling import ProfileReport
report = ProfileReport(train, title="Report", html={'style': {'full_width':True}}, explorative=True, missing_diagrams={'bar': True})
```

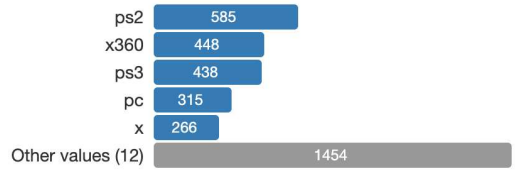
```
report.to_notebook_iframe()
```

```
#Save the report in file
report.to_file("pandas_profiling_report.html")
```

```
Export report to file: 100%                    1/1 [00:02<00:00, 2.31s/it]
```

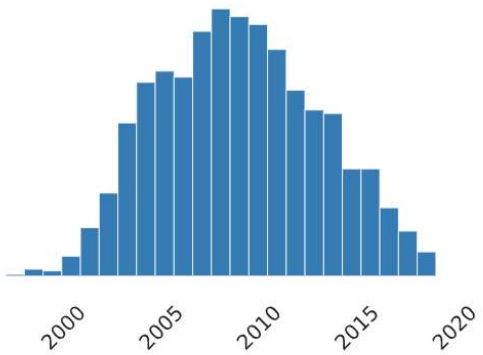
From the above reports we can gain following insights:-

- Console column graph:



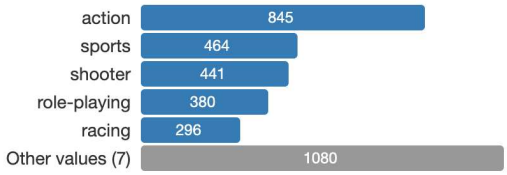
The sales of **PS2** were the highest in the data set

- Years Column graph:



The sales were highest between the period **2005-2010**.

- Game category column graph:



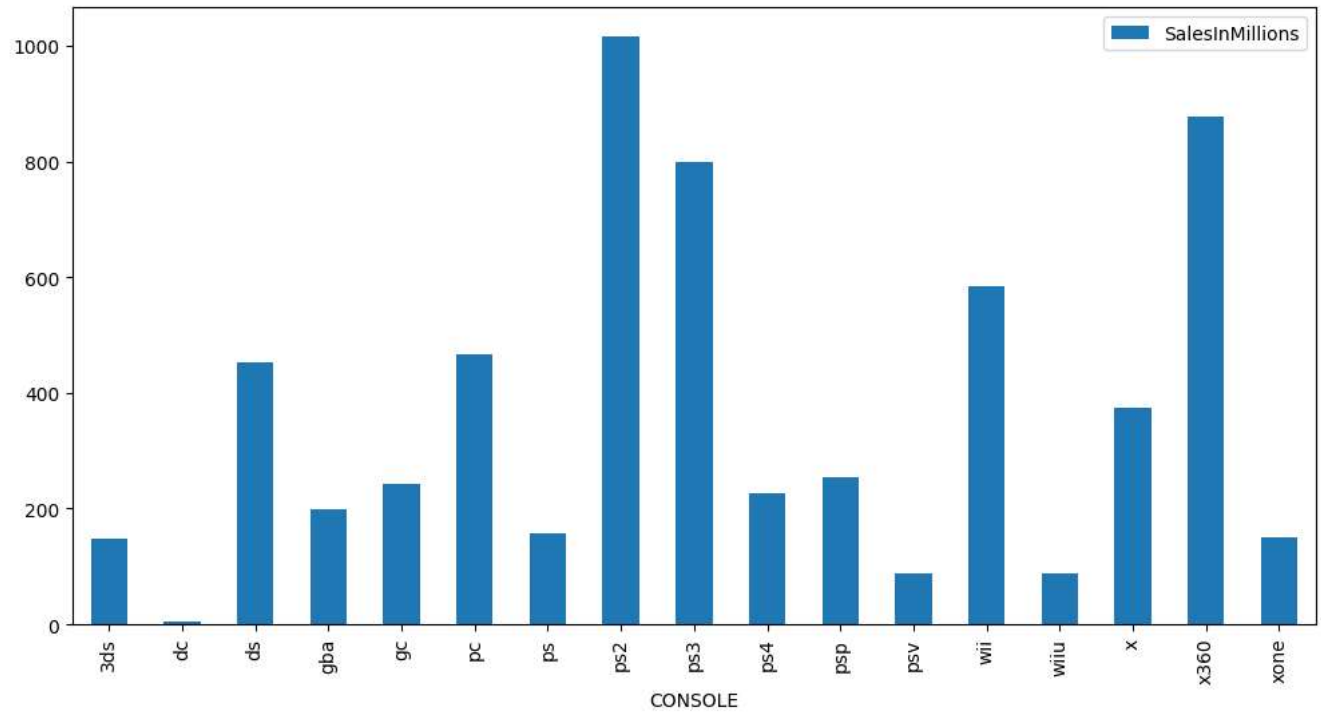
Action category games are most popular

Now let's compare individual columns with target(SalesInMillions) column to gain a few more insights into the data.

```
#Sales of games that happened corresponding to each console.
df = pd.DataFrame(train.groupby(['CONSOLE']).agg({'SalesInMillions': 'sum'}))
```

```
df.plot.bar(figsize=(12, 6))
```

```
<Axes: xlabel='CONSOLE'>
```

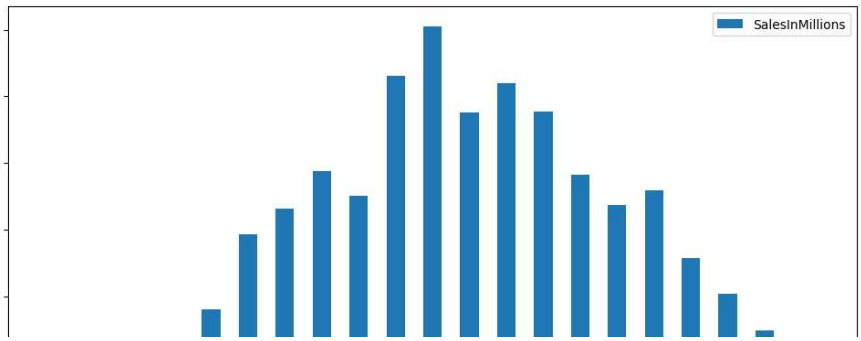


Insight: From the above graph we can see that sales were highest for PS3 platform followed by Xbox360

```
df = pd.DataFrame(train.groupby(['YEAR']).agg({'SalesInMillions': 'sum'}))
```

```
df.plot.bar(figsize=(12, 6))
```

```
<Axes: xlabel='YEAR'>
```

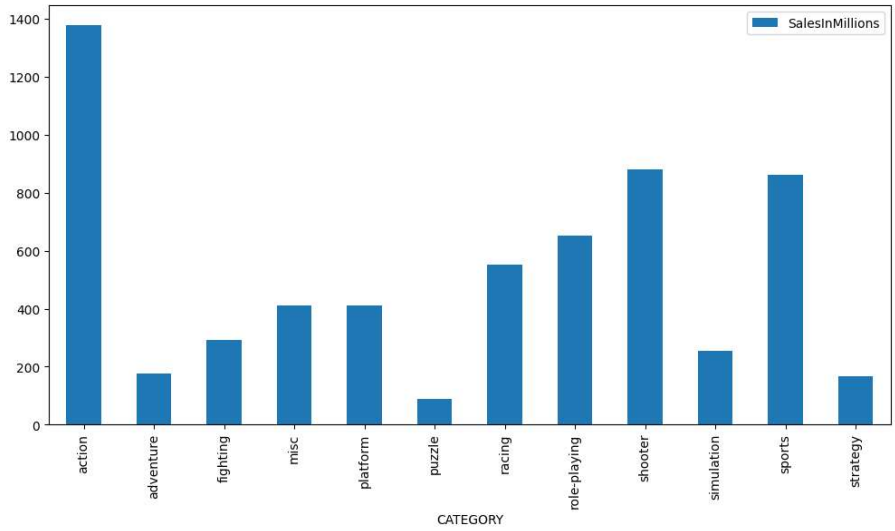


Insight: From the above graph we can see that sales were highest in the year 2010

```
df = pd.DataFrame(train.groupby(['CATEGORY']).agg({'SalesInMillions': 'sum'}))
```

```
df.plot.bar(figsize=(12, 6))
```

```
<Axes: xlabel='CATEGORY'>
```



Insight: From the above graph we can see that sales were highest for action genre

Model training

```
!pip install catboost
```

```
Collecting catboost
  Downloading catboost-1.2.2-cp310-cp310-manylinux2014_x86_64.whl (98.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 98.7/98.7 MB 3.1 MB/s eta 0:00:00
Requirement already satisfied: graphviz in /usr/local/lib/python3.10/dist-packages (from catboost) (0.20.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from catboost) (3.7.1)
Requirement already satisfied: numpy>=1.16.0 in /usr/local/lib/python3.10/dist-packages (from catboost) (1.23.5)
Requirement already satisfied: pandas>=0.24 in /usr/local/lib/python3.10/dist-packages (from catboost) (1.5.3)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from catboost) (1.11.3)
Requirement already satisfied: plotly in /usr/local/lib/python3.10/dist-packages (from catboost) (5.15.0)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from catboost) (1.16.0)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.24->catboost) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.24->catboost) (2023.3.post1)
```

```
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (1.1.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (4.43.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (23.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (3.1.1)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly->catboost) (8.2.3)
Installing collected packages: catboost
Successfully installed catboost-1.2.2
```

```
import catboost as cat
cat_feat = ['CONSOLE','CATEGORY', 'PUBLISHER', 'RATING']
features = list(set(train.columns)-set(['SalesInMillions']))
target = 'SalesInMillions'
model = cat.CatBoostRegressor(random_state=100,cat_features=cat_feat,verbose=0)
model.fit(train[features],train[target])
```

```
<catboost.core.CatBoostRegressor at 0x79a951746aa0>
```

Model Accuracy

```
y_true= pd.DataFrame(data=test[target], columns=['SalesInMillions'])
test_temp = test.drop(columns=[target])
```

```
y_pred = model.predict(test_temp[features])
```

```
from sklearn.metrics import mean_squared_error
from math import sqrt

rmse = sqrt(mean_squared_error(y_true, y_pred))
print(rmse)
```

```
1.5540129994547134
```

```
import pickle
filename = 'finalized_model.sav'
```

```
pickle.dump(model, open(filename, 'wb'))
```

```
loaded_model = pickle.load(open(filename, 'rb'))
```

```
test_temp[features].head(1)
```

	CATEGORY	YEAR	USER_POINTS	CONSOLE	RATING	PUBLISHER	CRITICS_POINTS
3272	shooter	2015	0.009848	ps3	M	Take-Two Interactive	2.806452

```
loaded_model.predict(test_temp[features].head(1))
```

```
array([2.97171105])
```