

Name-Priyansu Panda


Branch-CSE,Year-2nd

College-GITA Autonomous College ,
Bhubaneswar

Course-Data science

Github-<https://github.com/PriyansuSonu>

Major project-1

 Major project 1.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at January 12

Comment Share Settings Profile

+ Code + Text

Connect Editing

```
#MACHINE LEARNING - SUPERVISED LEARNING - REGRESSION - LINEAR REGRESSION
#Univariate/Single - 1 column as input , 1 column as output
#Multivariate/Multiple - multiple column as input , 1 column as output
#Dataset - /content/diamonds price.csv
#Diamond carat, Prices of quality diamond

#1.Take the data and create dataframe
import pandas as pd
df = pd.read_csv('/content/diamonds price.csv')
df

#IMAGINARY STORY - IMAGINE a REAL ESTATE/PROPERTY BROKER COMES TO YOU AND GIVES YOU THE BELOW DATASET and says
#CREATE A MODEL FOR ME ,WHICH COULD PREDICT THE PROPERTY PRICES ,BASED ON THE DATA I GIVE
```

	carat	price
0	0.20	326
1	0.21	326
2	0.22	327
3	0.22	334
4	0.23	335
5	0.24	340
6	0.26	344
7	0.29	345
8	0.30	345
9	0.31	352
10	0.32	354



+ Code + Text

Connect

Editing

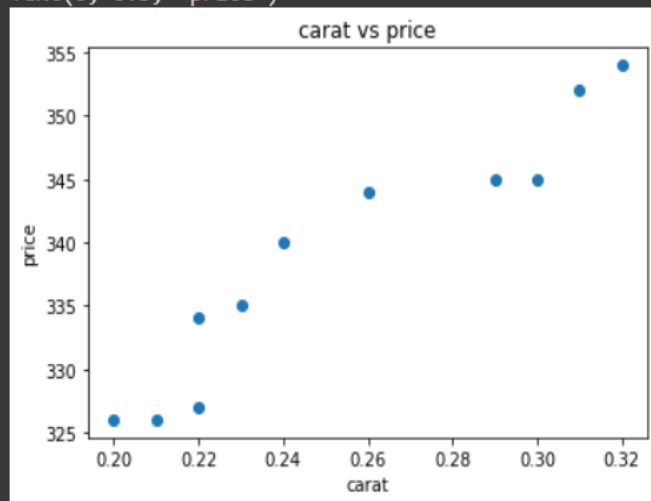


```
[ ] 9 0.31 352
    10 0.32 354
```

```
[ ] #2.We are not performing step no 2 ,for this dataset
```

```
[ ] #3.DATA VISUALISATION - CREATION of GRAPHS
import matplotlib.pyplot as plt
#plt.scatter(x-axis,y-axis)
plt.scatter(df['carat'],df['price'])
plt.title('carat vs price')
plt.xlabel('carat')
plt.ylabel('price')
```

```
Text(0, 0.5, 'price')
```



```
#INPUT(x) - AREA
#OUTPUT - PRICES
```



+ Code + Text

Connect ▾

Editing



```
#INPUT(x) - AREA
#OUTPUT - PRICES
#4.DIVIDE THE DATA INTO INPUT and OUTPUT
#INPUT(x) is always 2 dimensional,OUTPUT(y) is always 1 dimensional array
#df.iloc[row slicing,column slicing]
#In df.iloc ,if the column's place has a ':',then array is 2 dimensional
x = df.iloc[0:11,0:1].values
x
#.values converts the dataframe into an array
```

```
array([[0.2 ],
       [0.21],
       [0.22],
       [0.22],
       [0.23],
       [0.24],
       [0.26],
       [0.29],
       [0.3 ],
       [0.31],
       [0.32]])
```

+ Code

+ Text

```
[ ] #In df.iloc ,if the column's place does not has a ':',then array is 1 dimensiona
y = df.iloc[0:11,1].values #df.iloc[:,1].values
#If I want to select all rows or all cols , I can write only :
y
```

```
array([326, 326, 327, 334, 335, 340, 344, 345, 345, 352, 354])
```

```
[ ] #5.TRAIN and TEST VARIABLES
#Here we are not performing step no 5 , due to limited data
```

```
[ ] #6.NORMALIZATION/SCALING
```



+ Code + Text

Connect ▾

Editing



```
[ ] #6.NORMALIZATION/SCALING
#NORMALIZATION/SCALING IS ONLY DONE FOR MULTIVARIATE DATASETS
#HERE for UNIVARIATE data , Step no 6 is not required
```

```
[ ] #7.Run classifier,REGRESSOR or clusterer(APPLYING a suitable ALGORITHM)
#sklearn.linear_model - package(collection of libraries),LinearRegression - Library
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

```
▶ #8.FIT the MODEL(Mapping/Plotting the inputs with the outputs in the library)
#LinearRegression.fit(x,y)
model.fit(x,y) #We are mapping the values of x and y in the LinearRegression library
```

```
↳ LinearRegression()
```

```
[ ] #9.PREDICT THE OUTPUT
y_pred = model.predict(x) #Using the input values,we predict the output
y_pred #PREDICTED OUTPUT VALUES
```

```
array([326.84393064, 329.05587669, 331.26782274, 331.26782274,
       333.47976879, 335.69171484, 340.11560694, 346.75144509,
       348.96339114, 351.17533719, 353.38728324])
```

```
[ ] y #ACTUAL OUTPUT VALUES
```

```
array([326, 326, 327, 334, 335, 340, 344, 345, 345, 352, 354])
```

```
[ ] #CONCLUSION: We have to compare the corresponding values of y and y_pred.
#So when we compare , we come to know , that there is a huge difference
#This huge difference does not mean that our model has predicted wrong
#It only means , that our model is NOT LINEAR/LESS LINEAR
```



+ Code + Text

Connect | Editing

```
[ ] #CONCLUSION: We have to compare the corresponding values of y and y_pred.  
#So when we compare , we come to know , that there is a huge difference  
#This huge difference does not mean that our model has predicted wrong  
#It only means , that our model is NOT LINEAR/LESS LINEAR  
#LINEARITY of a model ,depends on the NATURE of the data as well as the SIZE of the data.
```

```
[ ] #INDIVIDUAL PREDICTION  
#I want to know the price for 2000 sqft  
model.predict([[.30]])  
  
array([348.96339114])
```

```
 #CROSS VERIFICATION  
#y = mx + C #Equation of a Straight line  
#m - Slope  
#C - Constant/y-intercept  
#y - dependant variable  
#x - independant variable  
#From excel m = 221.19460501, C = 282.6050095339114
```

```
[ ] m = model.coef_ #slope(m)  
m  
  
array([221.19460501])
```

```
[ ] C = model.intercept_ # constant/y-intercept  
C  
  
282.6050096339114
```

```
[ ] #y = mx +C
```



+ Code + Text

Connect ▾

Editing

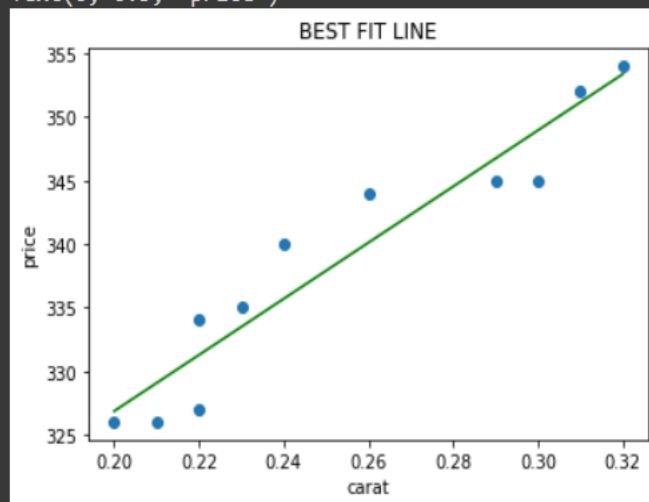


```
[ ] #y = mx +C  
    m*2000 + C
```


```
array([442671.8150289])
```

```
#FINAL VISUALISATION - BEST FIT LINE  
plt.scatter(x,y) #ACTUAL VALUES  
plt.plot(x,y_pred,c = 'green') #PREDICTED VALUES  
plt.title('BEST FIT LINE')  
plt.xlabel('carat')  
plt.ylabel('price')
```

```
Text(0, 0.5, 'price')
```



Major project-2

 Major project 2.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 12:02 AM

Comment Share Settings Profile

RAM Disk Editing

↑ ↓ ↻ ⌨ ⚙ 📄 🗑 ⋮

+ Code + Text

▶

```
#1 Take the data and create
import pandas as pd
df = pd.read_csv('/content/best-selling-gameboy.csv')
df
```

	Game	Developer(s)	Publisher	Platform	Release date	Sales
0	Pokémon Red / Green / Blue / Yellow	Game Freak	Nintendo	Game Boy	1996-02-27	46020000
1	Tetris	Nintendo R&D1	Nintendo	Game Boy	1989-06-14	35000000
2	Pokémon Gold / Silver / Crystal	Game Freak	Nintendo	Game Boy Color	1999-11-21	29490000
3	Super Mario Land	Nintendo R&D1	Nintendo	Game Boy	1989-04-21	18140000
4	Super Mario Land 2: 6 Golden Coins	Nintendo R&D1	Nintendo	Game Boy	1992-10-21	11180000
...
61	Space Invaders	Taito	Nintendo	Game Boy	1994-01-01	1000000
62	The Smurfs	Bit Managers	Infogrames	Game Boy	1994-01-01	1000000
63	Street Fighter II	Sun L	Nintendo	Game Boy	1995-08-11	1000000
64	Game & Watch Gallery	Tose	Nintendo	Game Boy	1997-02-01	1000000
65	James Bond 007	Saffire	Nintendo	Game Boy	1998-01-29	1000000

66 rows × 6 columns

< >

[]

df.shape

(66, 6)

+ Code + Text

✓ RAM
Disk

Editing

```
[ ] df.size #total no. of elements
```

```
396
```

```
[ ] df.info() #complete information
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 66 entries, 0 to 65  
Data columns (total 6 columns):  
#   Column      Non-Null Count  Dtype  
---  ---  
0   Game        66 non-null    object  
1   Developer(s) 66 non-null    object  
2   Publisher    66 non-null    object  
3   Platform     66 non-null    object  
4   Release date 66 non-null    object  
5   Sales        66 non-null    int64  
dtypes: int64(1), object(5)  
memory usage: 3.2+ KB
```

```
[ ] df.nunique()
```

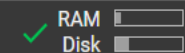
```
Game        66  
Developer(s) 31  
Publisher     8  
Platform      2  
Release date 61  
Sales        47  
dtype: int64
```

```
[ ] df['Platform'].value_counts()
```

```
Game Boy        48  
Game Boy Color  18
```



+ Code + Text

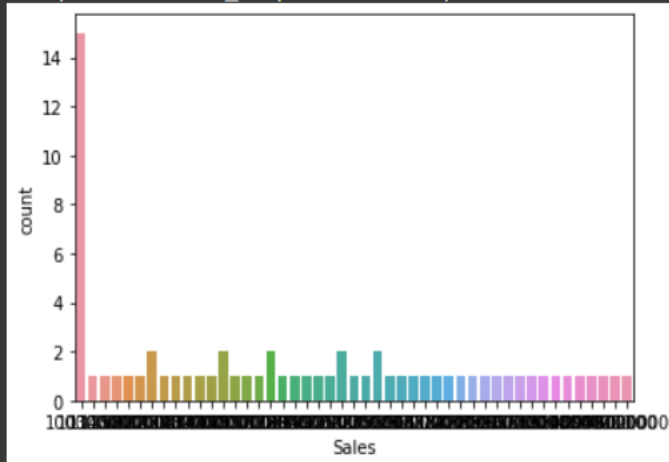


Editing



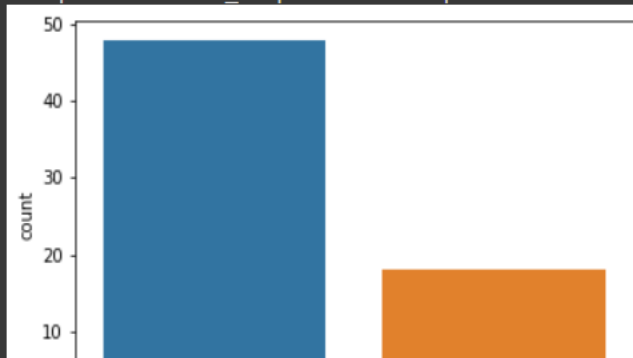
```
import seaborn as sns#import the library
sns.countplot(x = 'Sales',data = df)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f62d7be9eb0>



```
[ ] import seaborn as sns#import the library
sns.countplot(x = 'Platform',data = df)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f62df3f3af0>

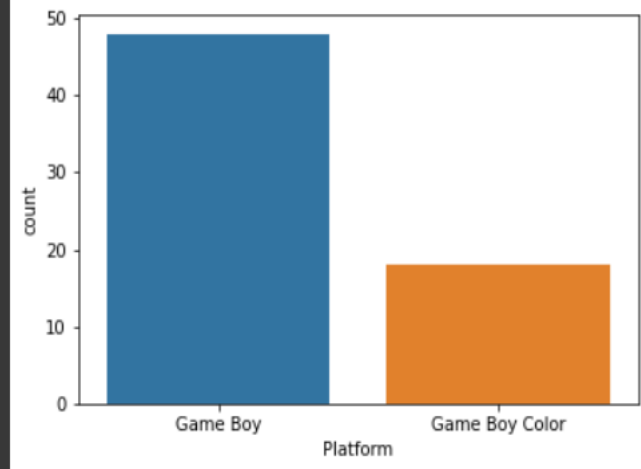


+ Code + Text

✓ RAM Disk Editing ^

```
import seaborn as sns#import the library
sns.countplot(x = 'Platform',data = df)
```

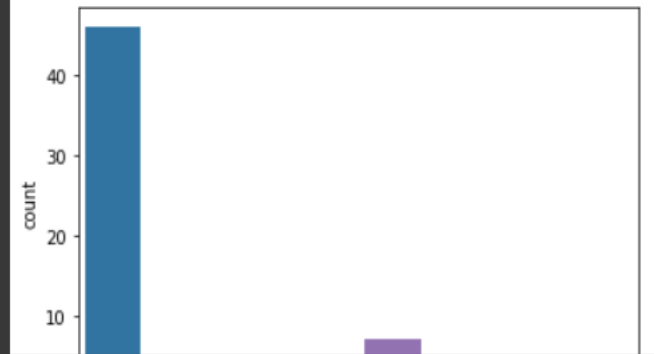
<matplotlib.axes._subplots.AxesSubplot at 0x7f62df3f3af0>



+ Code + Text

```
[ ] import seaborn as sns#import the library
sns.countplot(x = 'Publisher',data = df)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f62d7596af0>

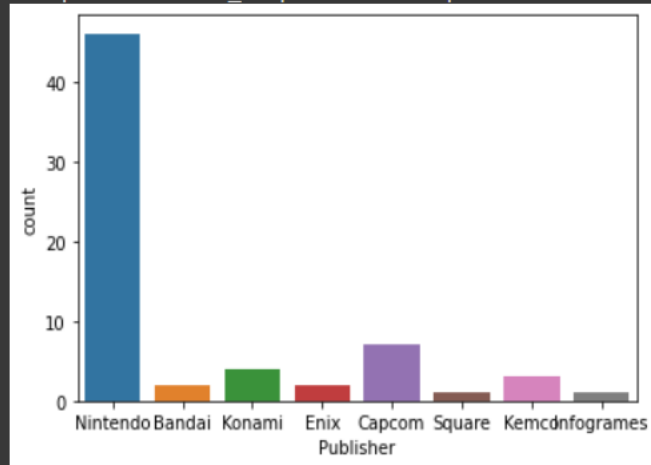


+ Code + Text

✓ RAM Disk Editing ^

```
import seaborn as sns#import the library
sns.countplot(x = 'Publisher',data = df)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f62d7596af0>



```
[ ] import numpy as np
    np.max(df['Sales'])
```

46020000

```
[ ] import numpy as np
    np.min(df['Sales'])
```

1000000

```
[ ] # statistical summary of the dataset
    print("Statistical summary of the dataset:")
    print(df.describe())
```

+ Code + Text

RAM Disk Editing ^

```
[ ] # statistical summary of the dataset
print("Statistical summary of the dataset:")
print(df.describe())
```

Statistical summary of the dataset:


	Sales
count	6.600000e+01
mean	4.010402e+06
std	7.801533e+06
min	1.000000e+06
25%	1.142500e+06
50%	1.610000e+06
75%	3.052500e+06
max	4.602000e+07

```
[ ] df['Developer(s)'].value_counts()
```

Nintendo R&D1	12
Rare	5
Tose	5
Konami	4
Game Freak	3
Kemco	3
HAL Laboratory	3
Nintendo EAD	3
Minakuchi Engineering	3
Capcom	2
Bandai	2
Nintendo EAD and Pax Softnica	2
Camelot Software Planning	1
Sun L	1
Bit Managers	1
Taito	1
Make Software	1
NMS Software	1
Japan System House	1


+ Code + Text

✓ RAM  Disk  Editing 

 `df['Developer(s)'].value_counts()`

```

Nintendo R&D1      12
Rare               5
Tose              5
Konami            4
Game Freak        3
Kemco             3
HAL Laboratory    3
Nintendo EAD      3
Minakuchi Engineering 3
Capcom            2
Bandai            2
Nintendo EAD and Pax Softnica 2
Camelot Software Planning 1
Sun L             1
Bit Managers      1
Taito             1
Make Software     1
NMS Software      1
Japan System House 1
Now Production    1
Jupiter           1
Flagship          1
Nintendo R&D2      1
Nintendo R&D1 and Tose 1
Intelligent Systems 1
Square            1
Hudson Soft and Creatures Inc. 1
Bullet-Proof Software 1
Nintendo R&D1 and Intelligent Systems 1
Nintendo          1
Saffire           1
Name: Developer(s), dtype: int64
```

 `[] df.head()`



+ Code

+ Text

✓ RAM

Disk

Editing

```
[ ] Nintendo 1
    Saffire 1
    Name: Developer(s), dtype: int64
```

```
[ ] df.head()
```

	Game	Developer(s)	Publisher	Platform	Release date	Sales
0	Pokémon Red / Green / Blue / Yellow	Game Freak	Nintendo	Game Boy	1996-02-27	46020000
1	Tetris	Nintendo R&D1	Nintendo	Game Boy	1989-06-14	35000000
2	Pokémon Gold / Silver / Crystal	Game Freak	Nintendo	Game Boy Color	1999-11-21	29490000
3	Super Mario Land	Nintendo R&D1	Nintendo	Game Boy	1989-04-21	18140000
4	Super Mario Land 2: 6 Golden Coins	Nintendo R&D1	Nintendo	Game Boy	1992-10-21	11180000

```
[ ]
```