MAJOR PROJECT

NAME – **ADITYA PRATIK NAYAK**

COLLEGE – Indian Institute of Technology, Bhubaneswar

BRANCH – B.Tech. in Mechanical Engineering and M.Tech. in Mechanical Systems Design

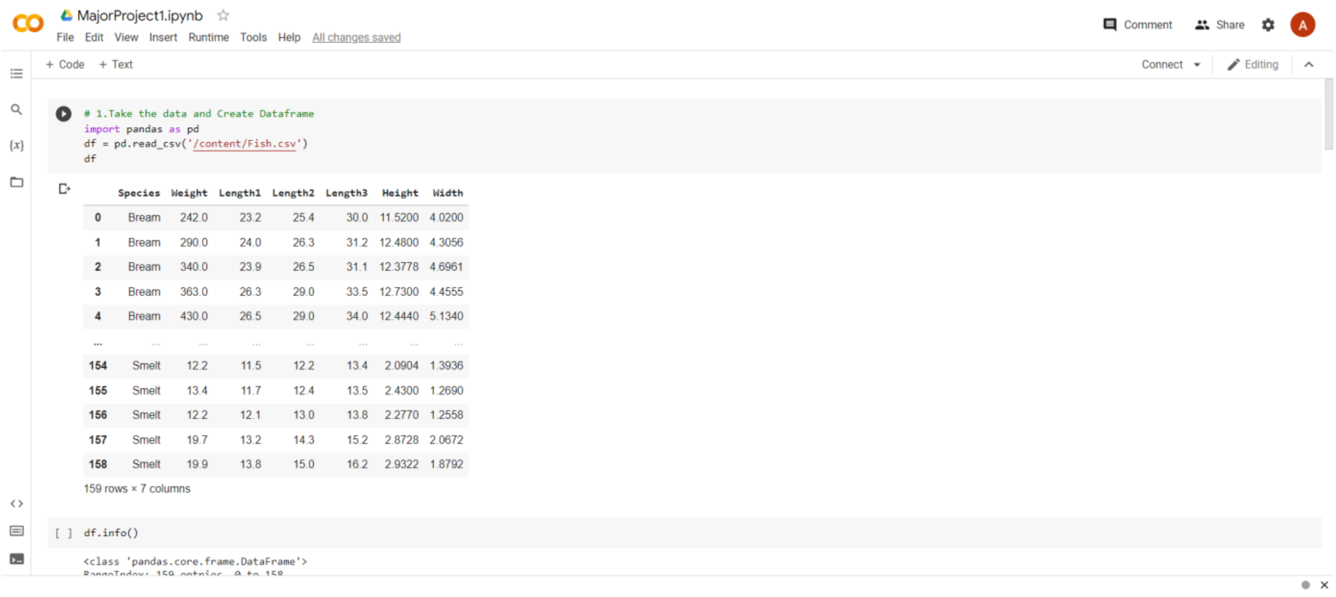
YEAR – 2nd Year

MY GITHUB ACCOUNT - https://github.com/ADITYA2647

MAJOR PROJECT 1

Link To Download the Dataset Required for the Major Project 1:

https://1drv.ms/x/s!Aj2Pc9HLMsmEeYjoBeGhykJL5_c?e=2wh0kT

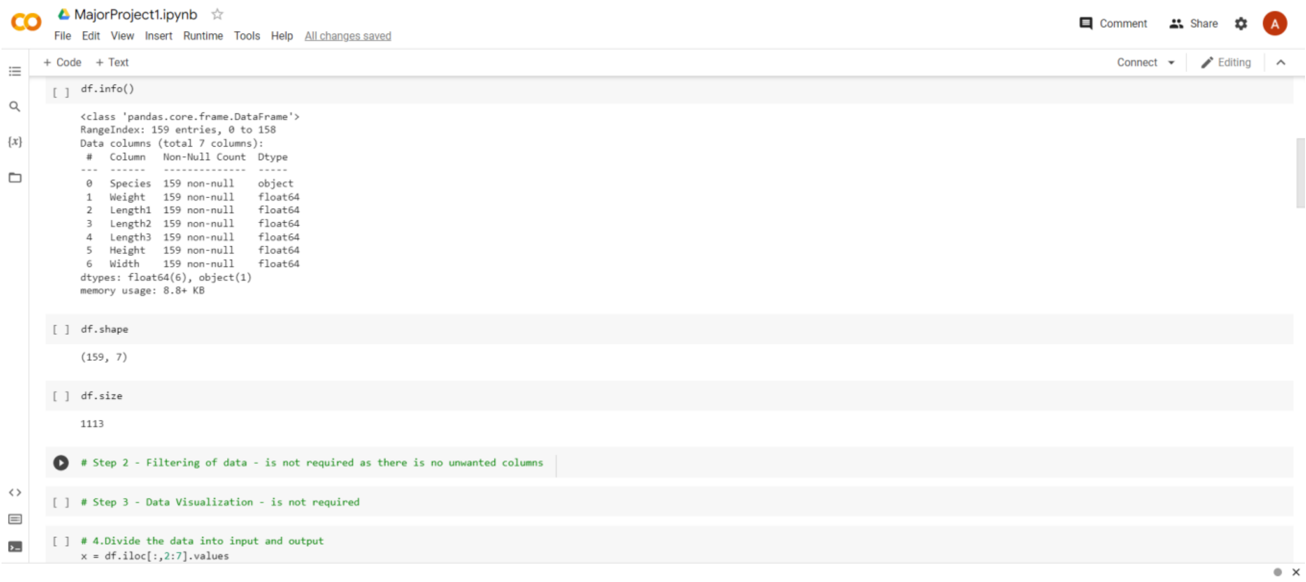


```
# 1.Take the data and Create Dataframe
import pandas as pd
df = pd.read_csv('/content/Fish.csv')
df
```

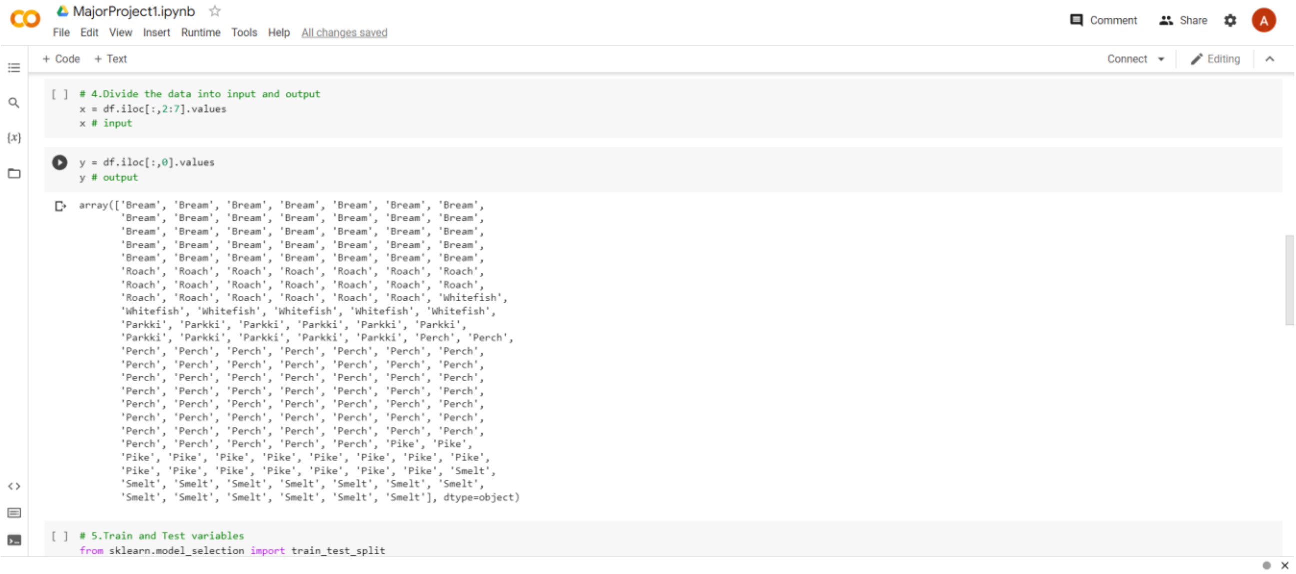
	Species	Weight	Length1	Length2	Length3	Height	Width
0	Bream	242.0	23.2	25.4	30.0	11.5200	4.0200
1	Bream	290.0	24.0	26.3	31.2	12.4800	4.3056
2	Bream	340.0	23.9	26.5	31.1	12.3778	4.6961
3	Bream	363.0	26.3	29.0	33.5	12.7300	4.4555
4	Bream	430.0	26.5	29.0	34.0	12.4440	5.1340
...	...	...	...	...	...	...	...
154	Smelt	12.2	11.5	12.2	13.4	2.0904	1.3936
155	Smelt	13.4	11.7	12.4	13.5	2.4300	1.2690
156	Smelt	12.2	12.1	13.0	13.8	2.2770	1.2558
157	Smelt	19.7	13.2	14.3	15.2	2.8728	2.0672
158	Smelt	19.9	13.8	15.0	16.2	2.9322	1.8792

159 rows × 7 columns

```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159 entries, 0 to 158
```



```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159 entries, 0 to 158
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   Species  159 non-null    object 
 1   Weight   159 non-null    float64
 2   Length1  159 non-null    float64
 3   Length2  159 non-null    float64
 4   Length3  159 non-null    float64
 5   Height   159 non-null    float64
 6   Width    159 non-null    float64
dtypes: float64(6), object(1)
memory usage: 8.8+ KB

[ ] df.shape

(159, 7)

[ ] df.size

1113

# Step 2 - Filtering of data - is not required as there is no unwanted columns

[ ] # Step 3 - Data Visualization - is not required

[ ] # 4.Divide the data into input and output
x = df.iloc[:,2:7].values
```

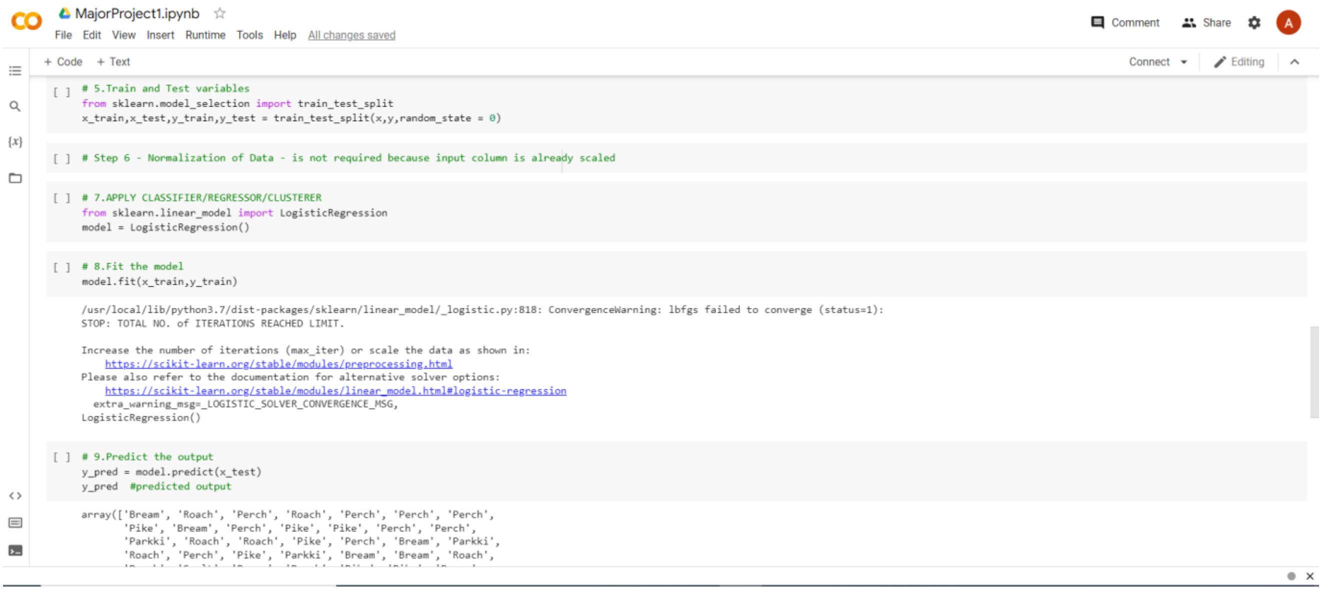


```
# 4.Divide the data into input and output
x = df.iloc[:,2:7].values
x # input
```

```
y = df.iloc[:,0].values
y # output
```

```
array(['Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream',
       'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream',
       'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream',
       'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream',
       'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream', 'Bream',
       'Roach', 'Roach', 'Roach', 'Roach', 'Roach', 'Roach', 'Roach',
       'Roach', 'Roach', 'Roach', 'Roach', 'Roach', 'Roach', 'Roach',
       'Roach', 'Roach', 'Roach', 'Roach', 'Roach', 'Roach', 'Whitefish',
       'Whitefish', 'Whitefish', 'Whitefish', 'Whitefish', 'Whitefish',
       'Parkki', 'Parkki', 'Parkki', 'Parkki', 'Parkki', 'Parkki',
       'Parkki', 'Parkki', 'Parkki', 'Parkki', 'Parkki', 'Perch', 'Perch',
       'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch',
       'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch',
       'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch',
       'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch',
       'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch',
       'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch',
       'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Perch',
       'Perch', 'Perch', 'Perch', 'Perch', 'Perch', 'Pike', 'Pike',
       'Pike', 'Pike', 'Pike', 'Pike', 'Pike', 'Pike', 'Pike', 'Pike',
       'Pike', 'Pike', 'Pike', 'Pike', 'Pike', 'Pike', 'Pike', 'Smelt',
       'Smelt', 'Smelt', 'Smelt', 'Smelt', 'Smelt', 'Smelt', 'Smelt',
       'Smelt', 'Smelt', 'Smelt', 'Smelt', 'Smelt', 'Smelt'], dtype=object)
```

```
# 5.Train and Test variables
from sklearn.model_selection import train_test_split
```



```
# 5.Train and Test variables
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,random_state = 0)
```

```
# Step 6 - Normalization of Data - is not required because input column is already scaled
```

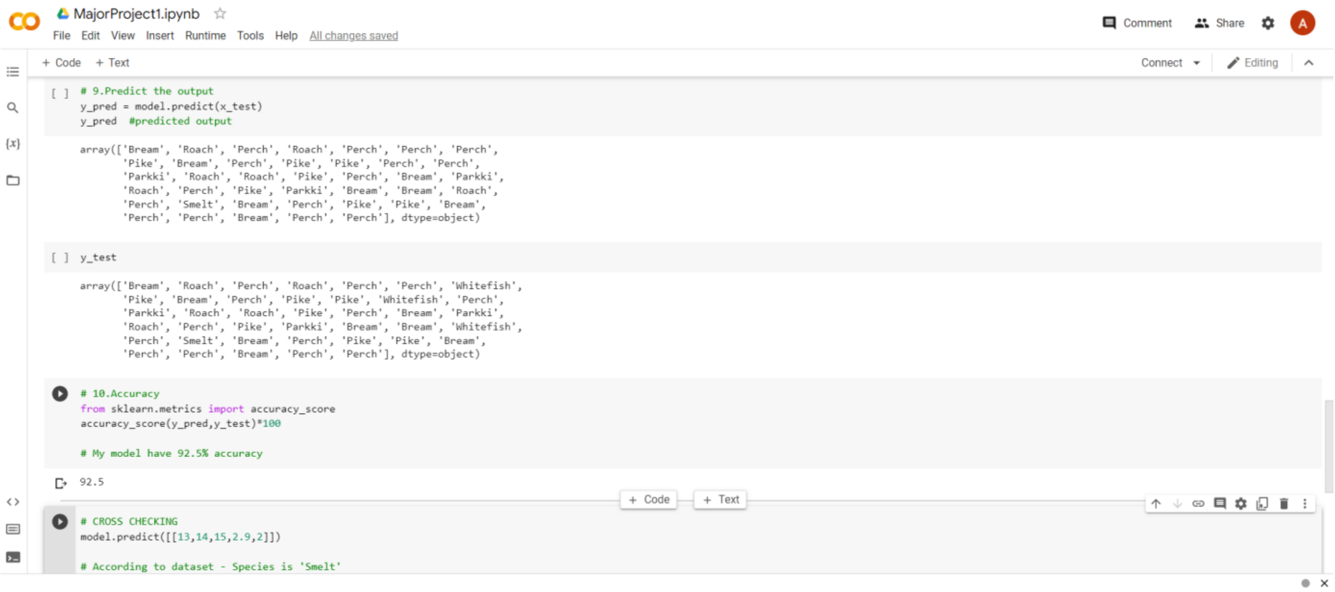
```
# 7.APPLY CLASSIFIER/REGRESSOR/CLUSTERER
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

```
# 8.Fit the model
model.fit(x_train,y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
LogisticRegression()
```

```
# 9.Predict the output
y_pred = model.predict(x_test)
y_pred  #predicted output
```

```
array(['Bream', 'Roach', 'Perch', 'Roach', 'Perch', 'Perch', 'Perch',
       'Pike', 'Bream', 'Perch', 'Pike', 'Pike', 'Perch', 'Perch',
       'Parkki', 'Roach', 'Roach', 'Pike', 'Perch', 'Bream', 'Parkki',
       'Roach', 'Perch', 'Pike', 'Parkki', 'Bream', 'Bream', 'Roach',
```



```
# 9.Predict the output
y_pred = model.predict(x_test)
y_pred  #predicted output
```

```
array(['Bream', 'Roach', 'Perch', 'Roach', 'Perch', 'Perch', 'Perch',
       'Pike', 'Bream', 'Perch', 'Pike', 'Pike', 'Perch', 'Perch',
       'Parkki', 'Roach', 'Roach', 'Pike', 'Perch', 'Bream', 'Parkki',
       'Roach', 'Perch', 'Pike', 'Parkki', 'Bream', 'Bream', 'Roach',
       'Perch', 'Smelt', 'Bream', 'Perch', 'Pike', 'Pike', 'Bream',
       'Perch', 'Perch', 'Bream', 'Perch', 'Perch'], dtype=object)
```

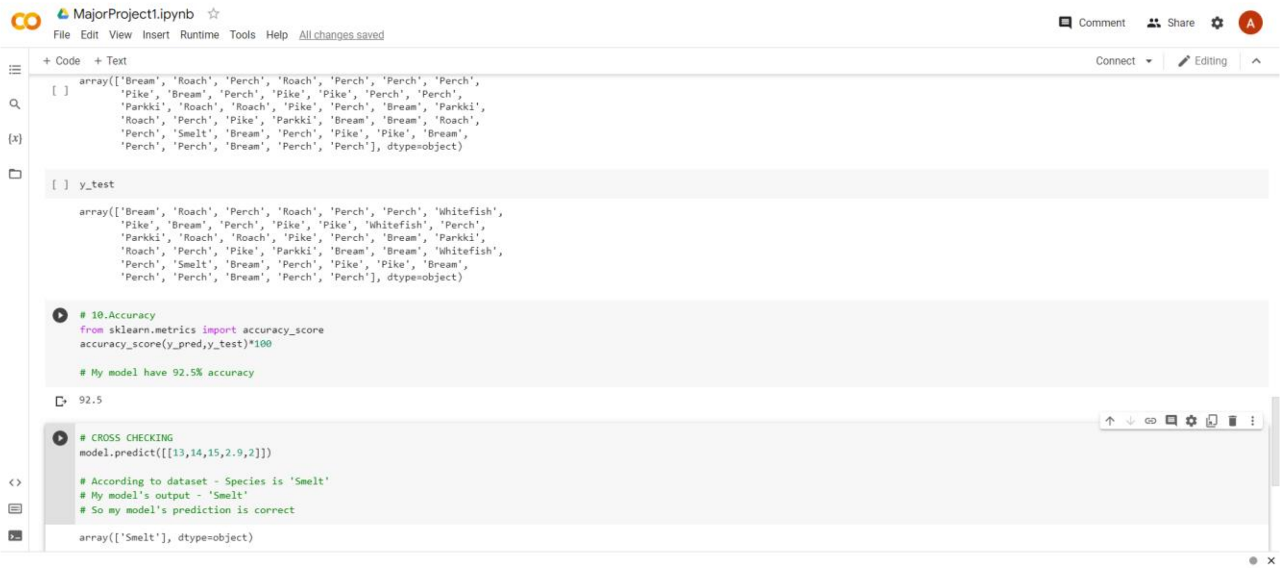
```
y_test
```

```
array(['Bream', 'Roach', 'Perch', 'Roach', 'Perch', 'Perch', 'Whitefish',
       'Pike', 'Bream', 'Perch', 'Pike', 'Pike', 'Whitefish', 'Perch',
       'Parkki', 'Roach', 'Roach', 'Pike', 'Perch', 'Bream', 'Parkki',
       'Roach', 'Perch', 'Pike', 'Parkki', 'Bream', 'Bream', 'Whitefish',
       'Perch', 'Smelt', 'Bream', 'Perch', 'Pike', 'Pike', 'Bream',
       'Perch', 'Perch', 'Bream', 'Perch', 'Perch'], dtype=object)
```

```
# 10.Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_pred,y_test)*100

# My model have 92.5% accuracy
```

```
92.5
```

```
# CROSS CHECKING
model.predict([[13,14,15,2.9,2]])

# According to dataset - Species is 'Smelt'
```



```
array(['Bream', 'Roach', 'Perch', 'Roach', 'Perch', 'Perch', 'Perch',
       'Pike', 'Bream', 'Perch', 'Pike', 'Pike', 'Perch', 'Perch',
       'Parkki', 'Roach', 'Roach', 'Pike', 'Perch', 'Bream', 'Parkki',
       'Roach', 'Perch', 'Pike', 'Parkki', 'Bream', 'Bream', 'Roach',
       'Perch', 'Smelt', 'Bream', 'Perch', 'Pike', 'Pike', 'Bream',
       'Perch', 'Perch', 'Bream', 'Perch', 'Perch'], dtype=object)
```

```
y_test
```

```
array(['Bream', 'Roach', 'Perch', 'Roach', 'Perch', 'Perch', 'Whitefish',
       'Pike', 'Bream', 'Perch', 'Pike', 'Pike', 'Whitefish', 'Perch',
       'Parkki', 'Roach', 'Roach', 'Pike', 'Perch', 'Bream', 'Parkki',
       'Roach', 'Perch', 'Pike', 'Parkki', 'Bream', 'Bream', 'Whitefish',
       'Perch', 'Smelt', 'Bream', 'Perch', 'Pike', 'Pike', 'Bream',
       'Perch', 'Perch', 'Bream', 'Perch', 'Perch'], dtype=object)
```

```
# 10.Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_pred,y_test)*100

# My model have 92.5% accuracy
```

```
92.5
```

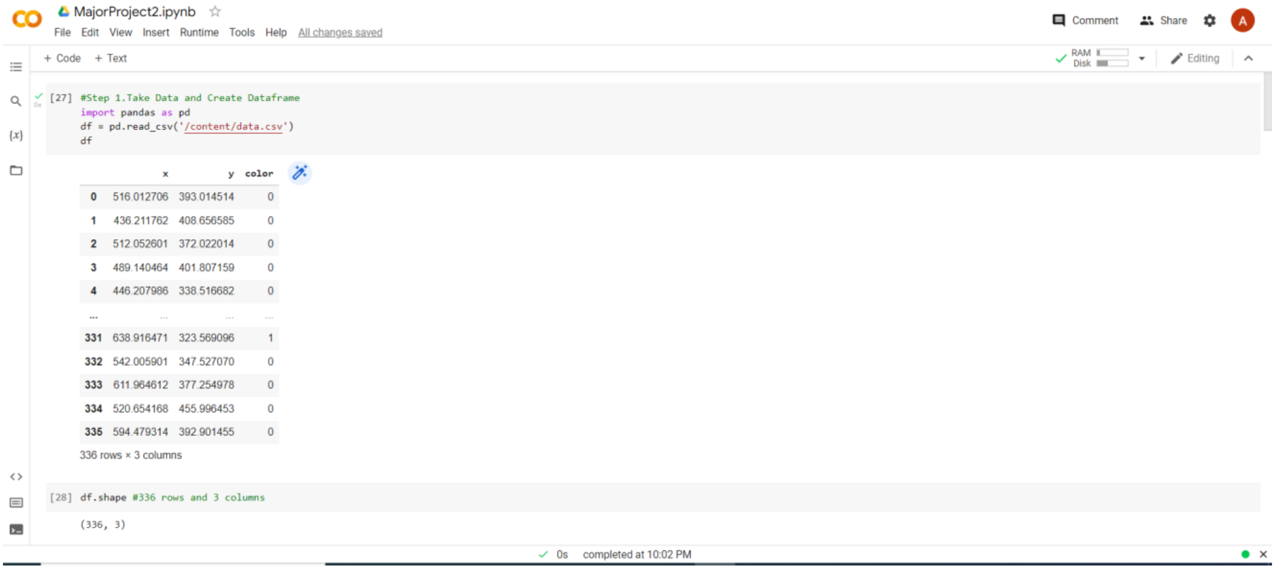
```
# CROSS CHECKING
model.predict([[13,14,15,2.9,2]])

# According to dataset - Species is 'Smelt'
# My model's output - 'Smelt'
# So my model's prediction is correct
```

```
array(['Smelt'], dtype=object)
```

MAJOR PROJECT 2

Link To Download the Dataset Required for the Major Project 2:

https://1drv.ms/x/s!Aj2Pc9HLMsmEfJ_A2auEhly6ND4?e=nxfoET



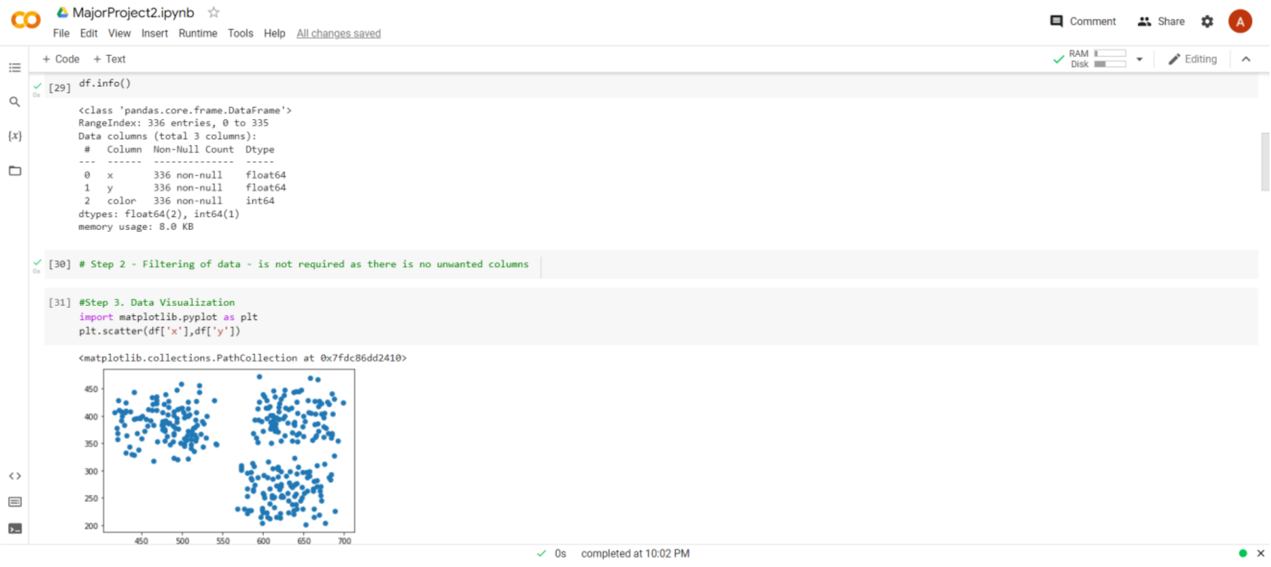
```
[27] #Step 1.Take Data and Create Dataframe
     import pandas as pd
     df = pd.read_csv('/content/data.csv')
     df
```

	x	y	color
0	516.012706	393.014514	0
1	436.211762	408.656585	0
2	512.052601	372.022014	0
3	489.140464	401.807159	0
4	446.207986	338.516682	0
...	...	...	...
331	638.916471	323.569096	1
332	542.005901	347.527070	0
333	611.964612	377.254978	0
334	520.654168	455.996453	0
335	594.479314	392.901455	0

336 rows × 3 columns

```
[28] df.shape #336 rows and 3 columns

     (336, 3)
```

✓ 0s completed at 10:02 PM

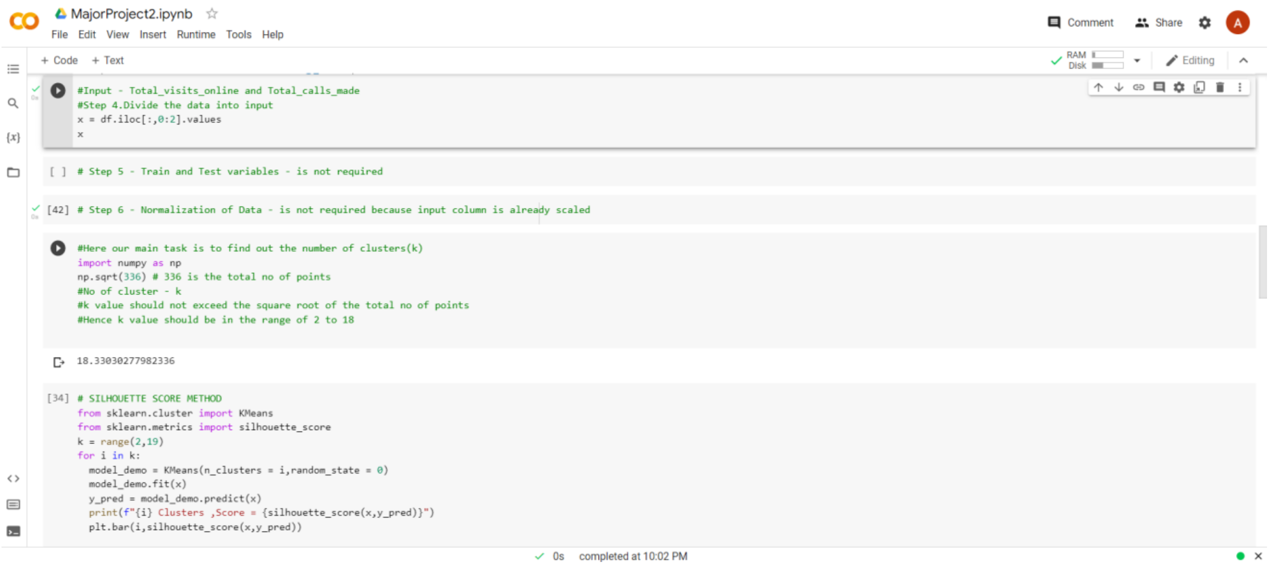


```
[29] df.info()

     <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 336 entries, 0 to 335
     Data columns (total 3 columns):
      #   Column  Non-Null Count  Dtype
     ---  ------  --------------  -----
      0   x       336 non-null    float64
      1   y       336 non-null    float64
      2   color   336 non-null    int64
     dtypes: float64(2), int64(1)
     memory usage: 8.0 KB
```

```
[30] # Step 2 - Filtering of data - is not required as there is no unwanted columns
```

```
[31] #Step 3. Data Visualization
     import matplotlib.pyplot as plt
     plt.scatter(df['x'],df['y'])

     <matplotlib.collections.PathCollection at 0x7fdc86dd2410>
```

✓ 0s completed at 10:02 PM



```
#Input - Total_visits_online and Total_calls_made
#Step 4.Divide the data into input
x = df.iloc[:,0:2].values
x
```

```
[ ] # Step 5 - Train and Test variables - is not required
```

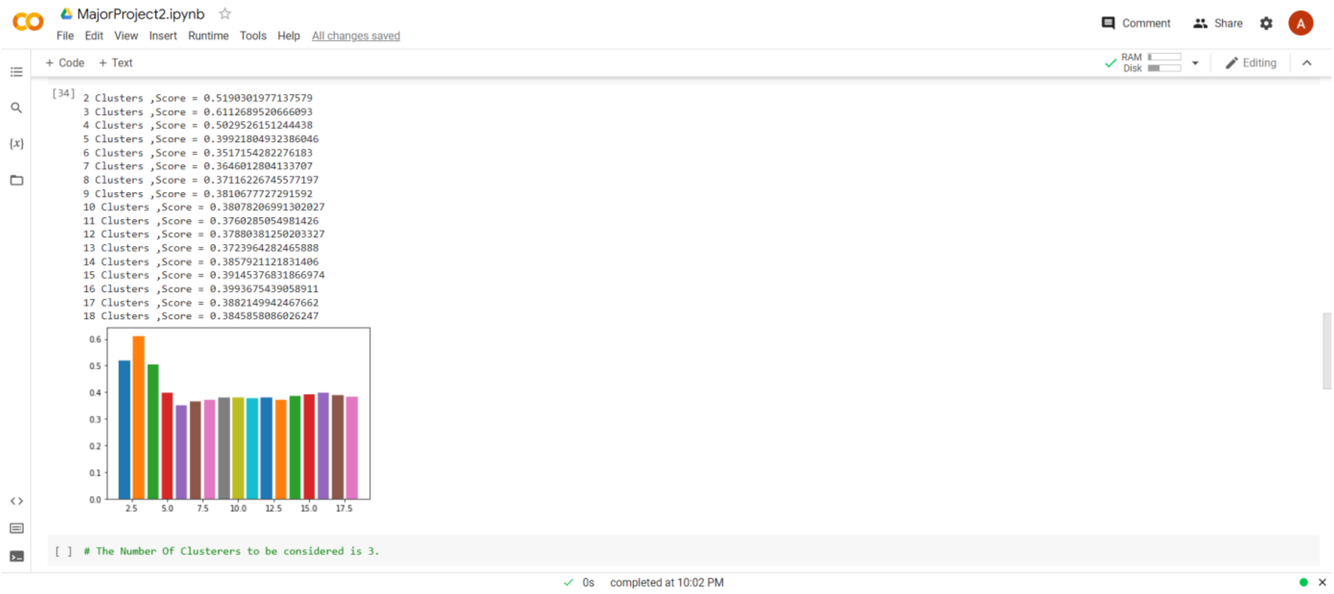
```
[42] # Step 6 - Normalization of Data - is not required because input column is already scaled
```

```
#Here our main task is to find out the number of clusters(k)
import numpy as np
np.sqrt(336) # 336 is the total no of points
#No of cluster - k
#k value should not exceed the square root of the total no of points
#Hence k value should be in the range of 2 to 18
```

```
18.33030277982336
```

```
[34] # SILHOUETTE SCORE METHOD
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
k = range(2,19)
for i in k:
  model_demo = KMeans(n_clusters = i,random_state = 0)
  model_demo.fit(x)
  y_pred = model_demo.predict(x)
  print(f"{i} Clusters ,Score = {silhouette_score(x,y_pred)}")
  plt.bar(i,silhouette_score(x,y_pred))
```

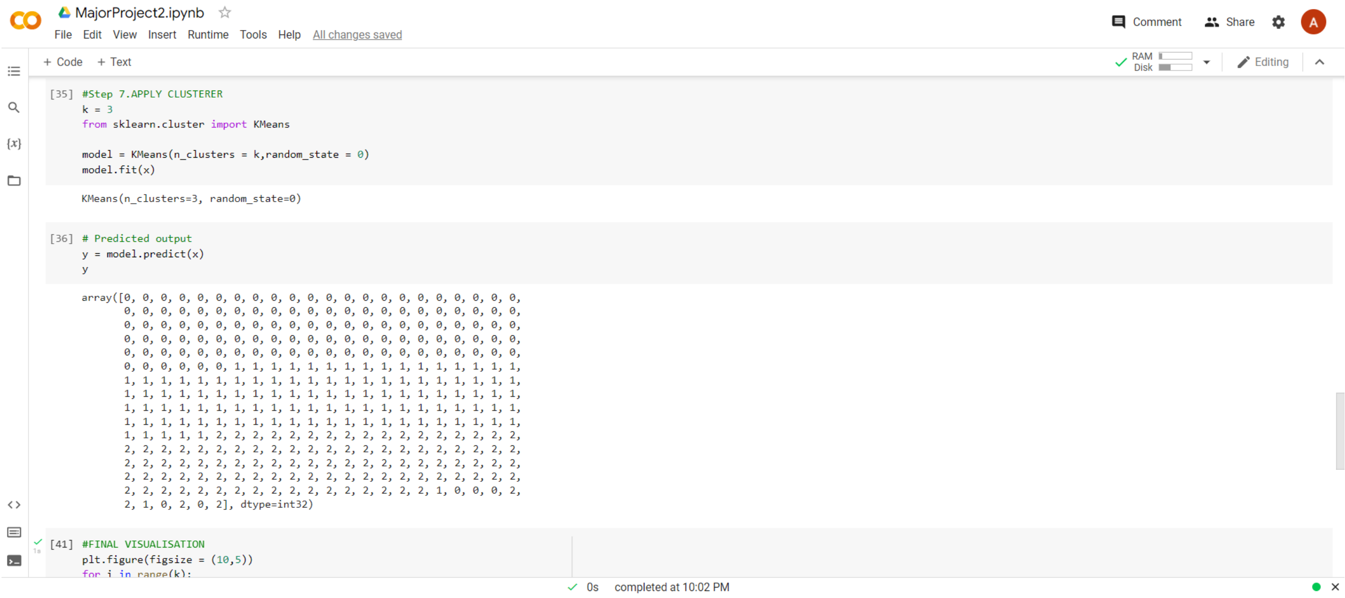
✓ 0s completed at 10:02 PM



MajorProject2.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing

```
[34] 2 Clusters ,Score = 0.5190301977137579
3 Clusters ,Score = 0.6112689520666093
4 Clusters ,Score = 0.5029526151244438
5 Clusters ,Score = 0.39921804932386046
6 Clusters ,Score = 0.3517154282276183
7 Clusters ,Score = 0.3646012804133707
8 Clusters ,Score = 0.37116226745577197
9 Clusters ,Score = 0.3810677727291592
10 Clusters ,Score = 0.38078206991302027
11 Clusters ,Score = 0.3760285054981426
12 Clusters ,Score = 0.37880381250203327
13 Clusters ,Score = 0.3723964282465888
14 Clusters ,Score = 0.3857921121831406
15 Clusters ,Score = 0.39145376831866974
16 Clusters ,Score = 0.3993675439058911
17 Clusters ,Score = 0.3882149942467662
18 Clusters ,Score = 0.3845858086026247
```

```
[ ] # The Number Of Clusterers to be considered is 3.
```

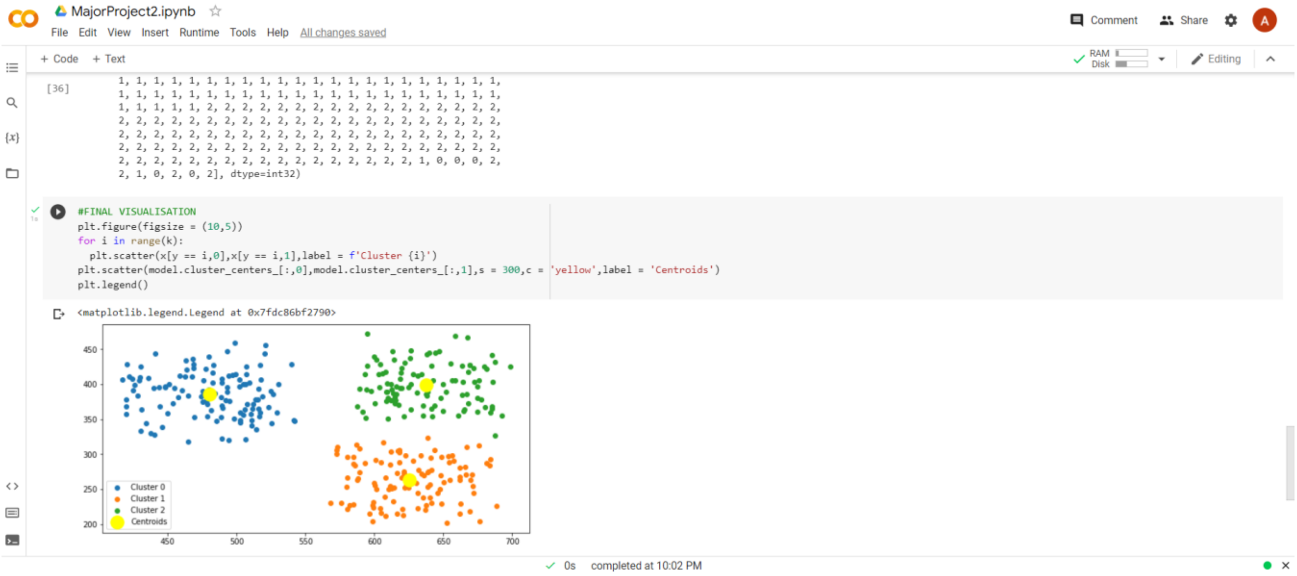


```
[35] #Step 7.APPLY CLUSTERER
     k = 3
     from sklearn.cluster import KMeans

     model = KMeans(n_clusters = k,random_state = 0)
     model.fit(x)

     KMeans(n_clusters=3, random_state=0)
```

```
[36] # Predicted output
     y = model.predict(x)
     y

     array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
            1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
            1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
            1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
            1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
            1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
            2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
            2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
            2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
            2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 0, 0, 0, 2,
            2, 1, 0, 2, 0, 2], dtype=int32)
```

```
[41] #FINAL VISUALISATION
     plt.figure(figsize = (10,5))
     for i in range(k):
```

✓ 0s completed at 10:02 PM



```
[36]   1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 0, 0, 0, 2,
       2, 1, 0, 2, 0, 2], dtype=int32)
```

```
#FINAL VISUALISATION
plt.figure(figsize = (10,5))
for i in range(k):
  plt.scatter(x[y == i,0],x[y == i,1],label = f'Cluster {i}')
plt.scatter(model.cluster_centers_[:,0],model.cluster_centers_[:,1],s = 300,c = 'yellow',label = 'Centroids')
plt.legend()

<matplotlib.legend.Legend at 0x7fdc86bf2790>
```

✓ 0s completed at 10:02 PM

>>>> THANK YOU <<<<