Mobile Price Prediction

Description

- battery_power: Total energy a battery can store in one time measured in mAh.
- blue: Has bluetooth or not.
- clock_speed: speed at which microprocessor executes instructions.
- dual_sim: Has dual sim support or not.
- fc: Front Camera mega pixels.
- four_g: Has 4G or not.
- int_memory: Internal Memory in Gigabytes.
- m_dep: Mobile Depth in cm.
- mobile_wt: Weight of mobile phone.
- n_cores: Number of cores of processor.
- pc: Primary Camera mega pixels.
- px_height: Pixel Resolution Height.
- px_width: Pixel Resolution Width.
- ram: Random Access Memory in Mega Byte.
- sc_h: Screen Height of mobile in cm.
- sc_w: Screen Width of mobile in cm.
- talk_time: longest time that a single battery charge will last when you are.
- three_g: Has 3G or not.
- touch_screen: Has touch screen or not.
- wifi: Has wifi or not.
- price_range: This is the target variable with value of 0(low cost), 1(medium cost), 2(high cost) and 3(very high cost).

Context

- Bob has started his own mobile company. He wants to give tough fight to big companies like Apple,Samsung etc.

- He does not know how to estimate price of mobiles his company creates. In this competitive mobile phone market you cannot simply assume things. To solve this problem he collects sales data of mobile phones of various companies.

- Bob wants to find out some relation between features of a mobile phone(eg:- RAM,Internal Memory etc) and its selling price. But he is not so good at Machine Learning. So he needs your help to solve this problem.

- In this problem you do not have to predict actual price but a price range indicating how high the price is

Importing 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,GridSearchCV
from sklearn.metrics import
classification_report,accuracy_score,roc_curve,confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB

import warnings
warnings.filterwarnings('ignore')
```

Importing Dataset

```
df = pd.read_csv("mobile_data.csv")
df.head()
```

```
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory
m_dep  \
0            842     0          2.2         0   1       0           7
0.6
1           1021     1          0.5         1   0       1          53
0.7
2            563     1          0.5         1   2       1          41
0.9
3            615     1          2.5         0   0       0          10
0.8
4           1821     1          1.2         0  13       1          44
0.6

   mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w
talk_time  \
```

```
0        188         2  ...         20        756  2549     9     7
19
1        136         3  ...        905       1988  2631    17     3
7
2        145         5  ...       1263       1716  2603    11     2
9
3        131         6  ...       1216       1786  2769    16     8
11
4        141         2  ...       1208       1212  1411     8     2
15

   three_g  touch_screen  wifi  price_range
0        0             0     1            1
1        1             1     0            2
2        1             1     0            2
3        1             0     0            2
4        1             1     0            1

[5 rows x 21 columns]
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   battery_power  2000 non-null   int64  
 1   blue           2000 non-null   int64  
 2   clock_speed    2000 non-null   float64
 3   dual_sim       2000 non-null   int64  
 4   fc             2000 non-null   int64  
 5   four_g         2000 non-null   int64  
 6   int_memory     2000 non-null   int64  
 7   m_dep          2000 non-null   float64
 8   mobile_wt      2000 non-null   int64  
 9   n_cores        2000 non-null   int64  
 10  pc             2000 non-null   int64  
 11  px_height      2000 non-null   int64  
 12  px_width       2000 non-null   int64  
 13  ram            2000 non-null   int64  
 14  sc_h           2000 non-null   int64  
 15  sc_w           2000 non-null   int64  
 16  talk_time      2000 non-null   int64  
 17  three_g        2000 non-null   int64  
 18  touch_screen   2000 non-null   int64  
 19  wifi           2000 non-null   int64  
 20  price_range    2000 non-null   int64  
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```

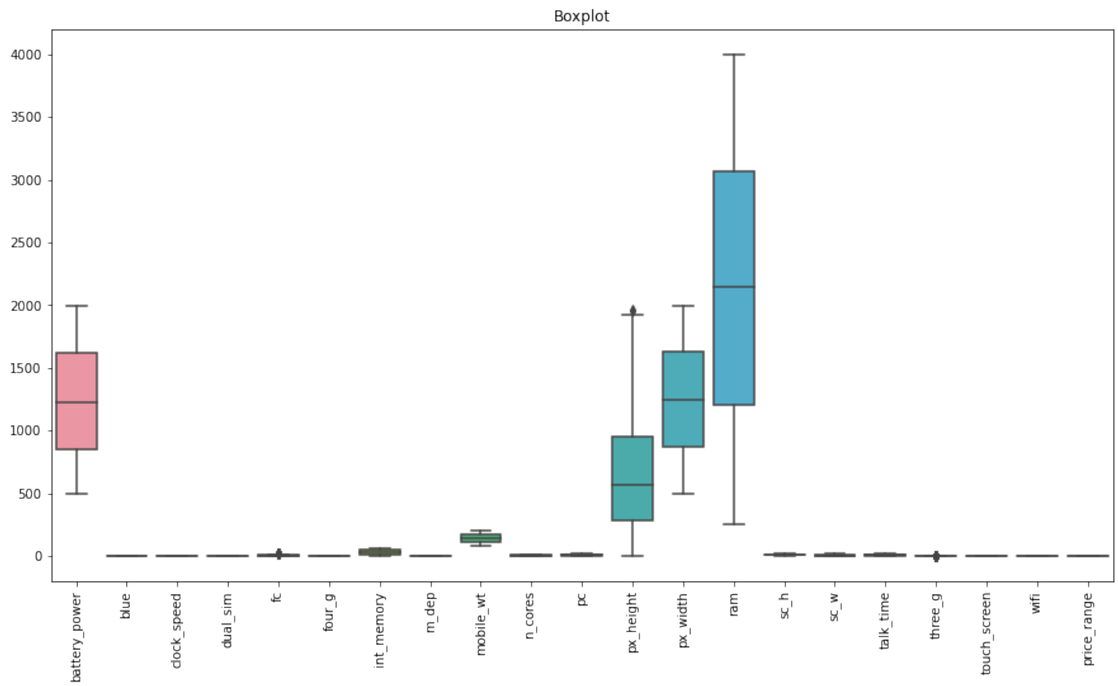
1. There are not any missing values in this dataset
2. Datatypes of all the features are correct

```
df.describe()

       battery_power       blue  clock_speed     dual_sim           fc  \
count    2000.000000  2000.0000  2000.000000  2000.000000  2000.000000

mean     1238.518500     0.4950     1.522250     0.509500     4.309500

std       439.418206     0.5001     0.816004     0.500035     4.341444

min       501.000000     0.0000     0.500000     0.000000     0.000000

25%       851.750000     0.0000     0.700000     0.000000     1.000000

50%      1226.000000     0.0000     1.500000     1.000000     3.000000

75%      1615.250000     1.0000     2.200000     1.000000     7.000000

max      1998.000000     1.0000     3.000000     1.000000    19.000000


            four_g   int_memory        m_dep    mobile_wt      n_cores  ...  \
count  2000.000000  2000.000000  2000.000000  2000.000000  2000.000000  ...
mean      0.521500    32.046500     0.501750   140.249000     4.520500  ...
std       0.499662    18.145715     0.288416    35.399655     2.287837  ...
min       0.000000     2.000000     0.100000    80.000000     1.000000  ...
25%       0.000000    16.000000     0.200000   109.000000     3.000000  ...
50%       1.000000    32.000000     0.500000   141.000000     4.000000  ...
75%       1.000000    48.000000     0.800000   170.000000     7.000000  ...
max       1.000000    64.000000     1.000000   200.000000     8.000000  ...

         px_height     px_width          ram         sc_h         sc_w  \
count  2000.000000  2000.000000  2000.000000  2000.000000  2000.000000

mean    645.108000  1251.515500  2124.213000    12.306500     5.767000

std     443.780811   432.199447  1084.732044     4.213245     4.356398
```

```
min       0.000000   500.000000   256.000000     5.000000     0.000000

25%     282.750000   874.750000  1207.500000     9.000000     2.000000

50%     564.000000  1247.000000  2146.500000    12.000000     5.000000

75%     947.250000  1633.000000  3064.500000    16.000000     9.000000

max    1960.000000  1998.000000  3998.000000    19.000000    18.000000


         talk_time     three_g  touch_screen        wifi
price_range
count  2000.000000  2000.000000   2000.000000  2000.000000
2000.000000
mean     11.011000     0.761500      0.503000     0.507000
1.500000
std       5.463955     0.426273      0.500116     0.500076
1.118314
min       2.000000     0.000000      0.000000     0.000000
0.000000
25%       6.000000     1.000000      0.000000     0.000000
0.750000
50%      11.000000     1.000000      1.000000     1.000000
1.500000
75%      16.000000     1.000000      1.000000     1.000000
2.250000
max      20.000000     1.000000      1.000000     1.000000
3.000000

[8 rows x 21 columns]
```

Data Cleaning

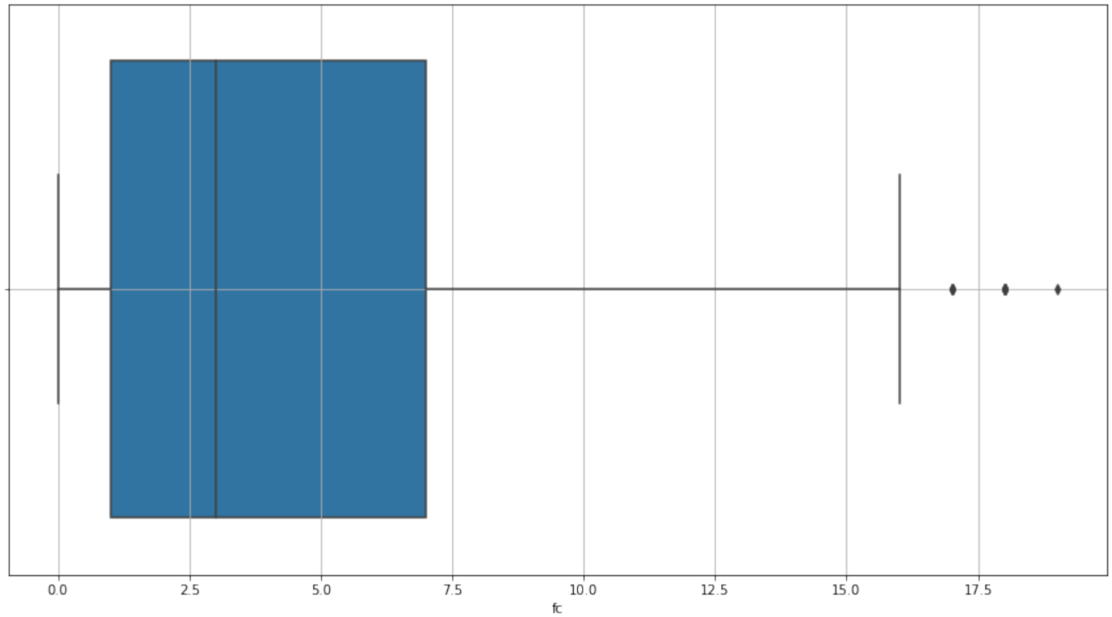
```
plt.figure(figsize=(15,8))
sns.boxplot(data=df)
plt.grid(False)
plt.xticks(rotation=90)
plt.title("Boxplot")
plt.show()
```



We can see there are outliers present in the data:

1. fc
2. px_height
3. three_g

```
# Plotting boxplot for 'fc' separately to spot outliers more precisely
plt.figure(figsize=(15,8))
sns.boxplot(df.fc)
plt.grid()
plt.show()
```




```
df.fc.describe()
```

```
count    2000.000000
mean        4.309500
std         4.341444
min         0.000000
25%         1.000000
50%         3.000000
75%         7.000000
max        19.000000
Name: fc, dtype: float64
```

Finding upper bound precisely to remove outliers correctly

```
# Upper Bound = Q3+1.5*(Q3-Q1)
# Lower Bound = Q1-1.5*(Q3-Q1)
7.000000+1.5*(7.000000-1.000000)
```

```
16.0
```

```
# Displaying outliers
df[df.fc > 16]
```

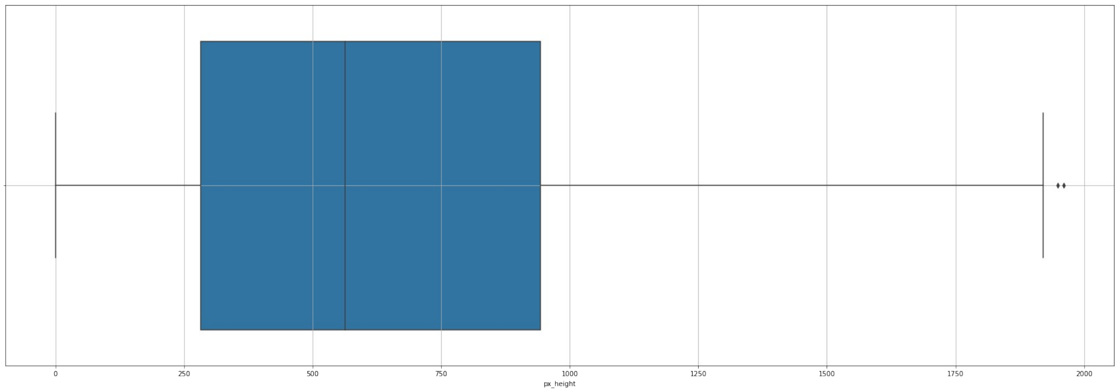
```
      battery_power  blue  clock_speed  dual_sim  fc  four_g
int_memory  \
95             1137     1          1.0         0  18       0
7
169            1569     0          2.8         1  17       0
44
226            1708     1          2.4         1  18       1
49
229            1689     0          1.8         0  17       0
```

```
24
300            1937     1          1.7         0  17       0
58
305            1348     0          2.0         0  18       0
52
372            1703     1          1.5         1  17       1
55
584             946     1          2.6         1  17       0
5
1387           1533     1          1.1         1  18       1
17
1406           1731     1          2.3         1  18       0
60
1416           1448     0          0.5         1  18       0
2
1549           1772     1          1.6         0  17       1
45
1554           1957     0          1.2         1  18       1
36
1693            695     0          0.5         0  18       1
12
1705           1290     1          1.4         1  19       1
35
1880           1720     0          1.6         0  18       1
2
1882            591     0          2.1         1  18       1
16
1888           1544     0          2.4         0  18       1
12

      m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h
sc_w  \
95      1.0        196        3  ...        942      1179  3616    13
5
169     0.3        110        4  ...         45      1942  1260     9
2
226     0.1        109        1  ...        233       517  3388     6
4
229     0.3        127        3  ...        954      1200  2766     7
2
300     0.6        189        1  ...       1728      1767  3321     5
4
305     0.3         98        3  ...       1869      1942   955    18
11
372     0.7        138        5  ...       1411      1711  2993     5
1
584     0.1        166        3  ...       1698      1771  3720    15
7
1387    0.3        160        4  ...       1054      1393  2520     8
2
```

```
1406    0.5         171         4  ...         142       1039  1220     9
3
1416    0.2         100         5  ...         846       1144   593     9
4
1549    0.5         159         2  ...         837       1405  1146     6
1
1554    0.8         151         2  ...        1194       1727  1115    16
2
1693    0.6         196         2  ...        1649       1829  2855    16
13
1705    0.3         110         4  ...         405        742   879    16
2
1880    0.8         188         5  ...         334        896  2522    10
5
1882    0.5         196         7  ...         952       1726   704    14
5
1888    0.1         186         7  ...         470        844   489     9
4

      talk_time  three_g  touch_screen  wifi  price_range
95           12        1             1     1            3
169          17        1             0     0            1
226          16        1             1     1            3
229           7        0             1     1            3
300          14        1             1     0            3
305           7        1             1     1            1
372          20        1             1     1            3
584           4        0             1     0            3
1387         11        1             0     1            2
1406         20        0             1     0            1
1416         18        1             1     1            0
1549         17        1             1     0            1
1554         18        1             0     1            1
1693          7        1             1     1            2
1705          8        1             0     0            0
1880          2        1             0     1            2
1882          4        1             1     1            0
1888          2        1             0     1            0

[18 rows x 21 columns]
```

```
# Getting indexes of outliers
index_fc = df[df.fc > 16].index

# Dropping outliers from the dataset
df.drop(index_fc,axis=0,inplace=True)

# Plotting boxplot for 'px_height' separately to spot outliers more
precisely
plt.figure(figsize=(30,10))
sns.boxplot(df.px_height)
```

```
plt.grid()
plt.show()
```




```
df.px_height.describe()
```

```
count    1982.000000
mean      642.509082
std       441.709410
min         0.000000
25%       282.000000
50%       562.500000
75%       943.500000
max      1960.000000
Name: px_height, dtype: float64
```

```
# Upper Bound = Q3+1.5*(Q3-Q1)
# Lower Bound = Q1-1.5*(Q3-Q1)
943.500000+1.5*(943.500000-282.000000)
```

```
1935.75
```

```
df[df.px_height > 1935.75]
```

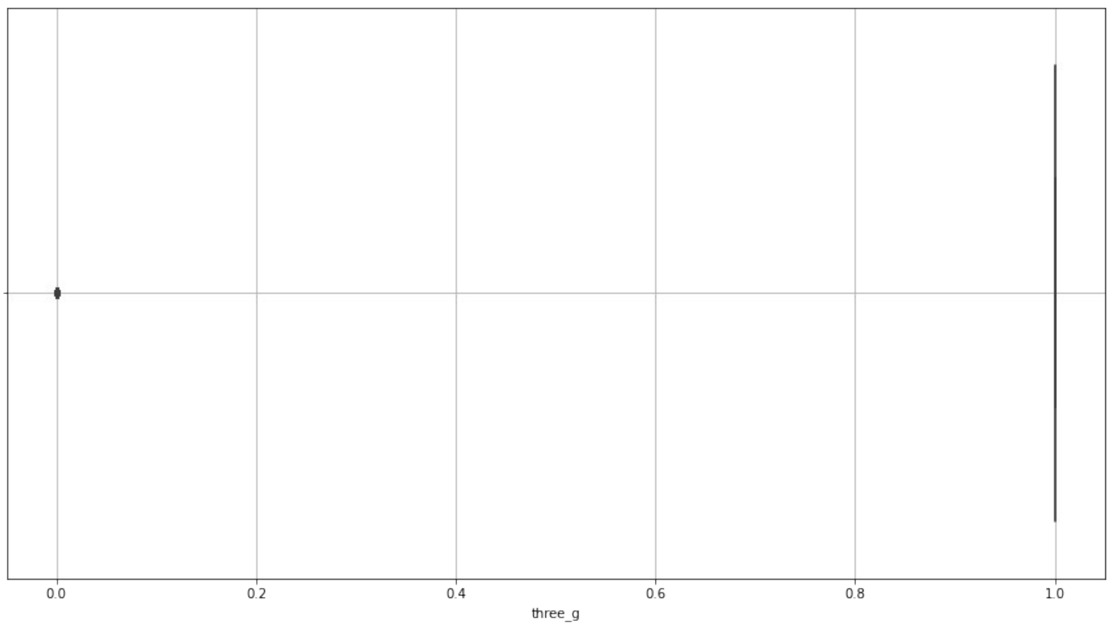
```
      battery_power  blue  clock_speed  dual_sim  fc  four_g  
int_memory  \
988            1413     1          0.5         1   4       1          
45   
1771           1230     1          1.6         0   0       1          
48   

      m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  
sc_w  \
988     0.4        104        5  ...       1949      1994  2973    17     
8   
1771    0.7        111        7  ...       1960      1963  1622    18    
17   

      talk_time  three_g  touch_screen  wifi  price_range  
988          15        1             0     1            3  
```

```
1771          16         1              1     1            2

[2 rows x 21 columns]
```

```python
index_px_height = df[df.px_height > 1935.75].index
```

```python
df.drop(index_px_height,inplace=True)
```

```python
# Plotting boxplot for 'three_g' separately to spot outliers more precisely
plt.figure(figsize=(15,8))
sns.boxplot(df.three_g)
plt.grid()
plt.show()
```



If we look at the data clearly we can see that these are not outliers so we are not going to remove them form our dataset

```python
df.corr()
```

```
               battery_power      blue  clock_speed  dual_sim
fc  \
battery_power       1.000000  0.009743     0.010136 -0.042558
0.020317
blue                0.009743  1.000000     0.021739  0.033543
0.004142
clock_speed         0.010136  0.021739     1.000000 -0.002017 -
0.006342
dual_sim           -0.042558  0.033543    -0.002017  1.000000 -
0.034041
fc                  0.020317  0.004142    -0.006342 -0.034041
1.000000
```

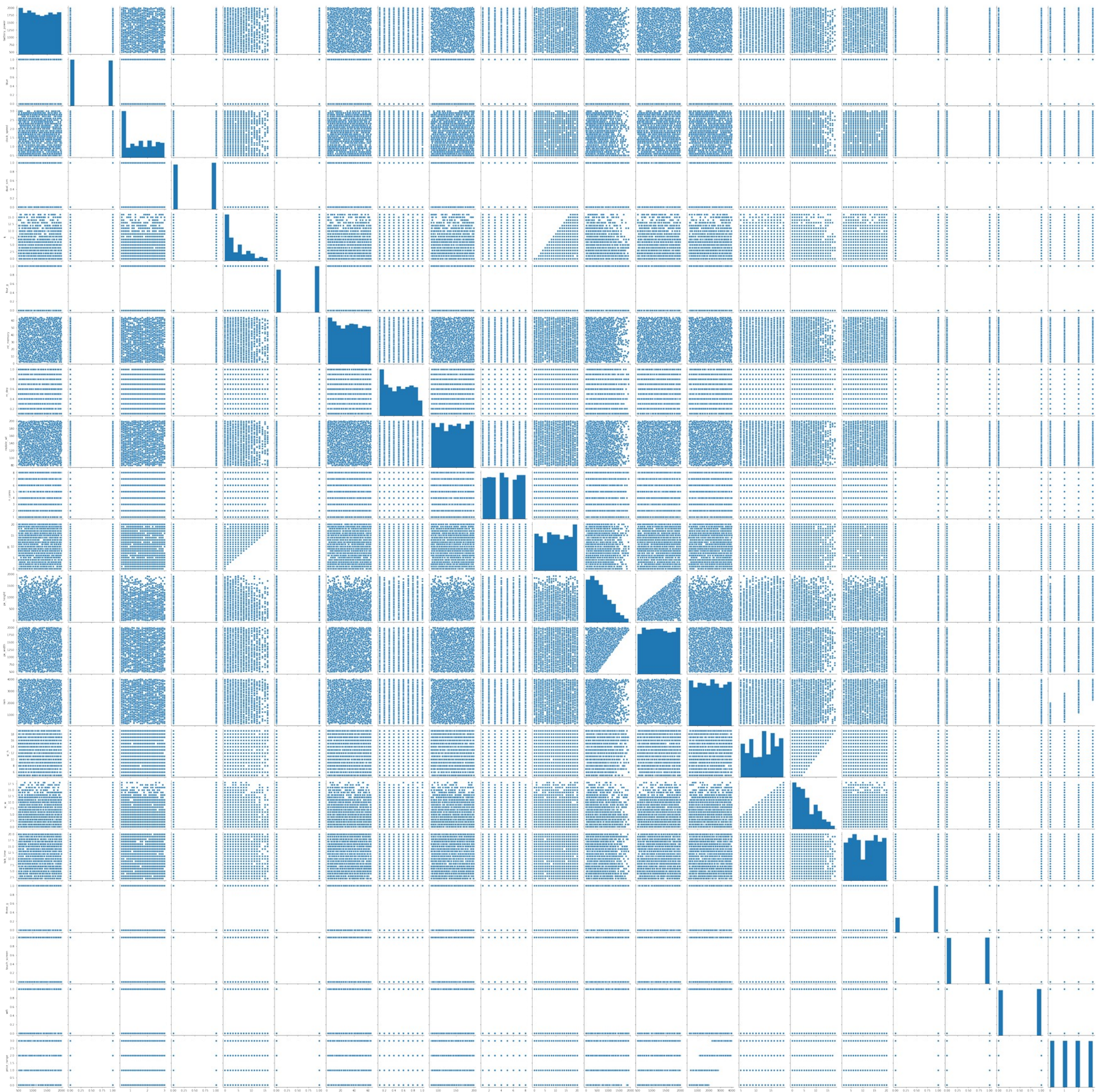
```
four_g             0.015559  0.012593    -0.041597  0.002280 -
0.019355
int_memory        -0.008197  0.037186     0.004461 -0.016975 -
0.025745
m_dep              0.034834  0.003585    -0.011400 -0.019390
0.004310
mobile_wt          0.002575 -0.008442     0.011939 -0.005955
0.014011
n_cores           -0.026451  0.038278    -0.005638 -0.025355 -
0.001971
pc                 0.025013 -0.009193    -0.009017 -0.019470
0.635564
px_height          0.014290 -0.010266    -0.011326 -0.018690 -
0.027777
px_width          -0.008205 -0.041741    -0.009050  0.014325 -
0.012505
ram               -0.000121  0.022024     0.004628  0.042491
0.019440
sc_h              -0.023784 -0.001204    -0.026876 -0.011811
0.000372
sc_w              -0.016533  0.001278    -0.005777 -0.012968 -
0.001778
talk_time          0.047909  0.009541    -0.010122 -0.043983 -
0.008136
three_g            0.010670 -0.029907    -0.044429 -0.013474 -
0.003121
touch_screen      -0.010004  0.008115     0.019023 -0.015209 -
0.024377
wifi              -0.009022 -0.019082    -0.021960  0.024064
0.011902
price_range        0.200763  0.015798    -0.006120  0.019016
0.021120

                 four_g  int_memory     m_dep  mobile_wt
n_cores  ...  \
battery_power  0.015559   -0.008197  0.034834   0.002575 -
0.026451  ...
blue           0.012593    0.037186  0.003585  -0.008442
0.038278  ...
clock_speed   -0.041597    0.004461 -0.011400   0.011939 -
0.005638  ...
dual_sim       0.002280   -0.016975 -0.019390  -0.005955 -
0.025355  ...
fc            -0.019355   -0.025745  0.004310   0.014011 -
0.001971  ...
four_g         1.000000    0.008995 -0.002771  -0.017901 -
0.031608  ...
int_memory     0.008995    1.000000  0.006426  -0.030009 -
0.026662  ...
m_dep         -0.002771    0.006426  1.000000   0.018595 -
```

```
0.003834  ...
mobile_wt      -0.017901   -0.030009  0.018595   1.000000 -
0.018240  ...
n_cores        -0.031608   -0.026662 -0.003834  -0.018240
1.000000  ...
pc             -0.005757   -0.030888  0.030108   0.013121
0.004900  ...
px_height      -0.021117    0.009328  0.024797  -0.000223 -
0.003893  ...
px_width        0.007998   -0.011010  0.022394  -0.000128
0.025602  ...
ram             0.008631    0.033712 -0.011402  -0.004555
0.008277  ...
sc_h            0.026550    0.039791 -0.027314  -0.032044 -
0.002913  ...
sc_w            0.036958    0.012055 -0.019880  -0.019790
0.024048  ...
talk_time      -0.046438   -0.010334  0.015418   0.010434
0.015609  ...
three_g         0.584754   -0.009660 -0.013723   0.001730 -
0.015022  ...
touch_screen    0.020094   -0.028687 -0.002823  -0.016019
0.026642  ...
wifi           -0.021650    0.010588 -0.029474  -0.000862 -
0.011145  ...
price_range     0.015906    0.043458 -0.000925  -0.031628
0.008307  ...

               px_height  px_width       ram      sc_h      sc_w
talk_time  \
battery_power   0.014290 -0.008205 -0.000121 -0.023784 -0.016533
0.047909
blue           -0.010266 -0.041741  0.022024 -0.001204  0.001278
0.009541
clock_speed    -0.011326 -0.009050  0.004628 -0.026876 -0.005777  -
0.010122
dual_sim       -0.018690  0.014325  0.042491 -0.011811 -0.012968  -
0.043983
fc             -0.027777 -0.012505  0.019440  0.000372 -0.001778  -
0.008136
four_g         -0.021117  0.007998  0.008631  0.026550  0.036958  -
0.046438
int_memory      0.009328 -0.011010  0.033712  0.039791  0.012055  -
0.010334
m_dep           0.024797  0.022394 -0.011402 -0.027314 -0.019880
0.015418
mobile_wt      -0.000223 -0.000128 -0.004555 -0.032044 -0.019790
0.010434
n_cores        -0.003893  0.025602  0.008277 -0.002913  0.024048
0.015609
```

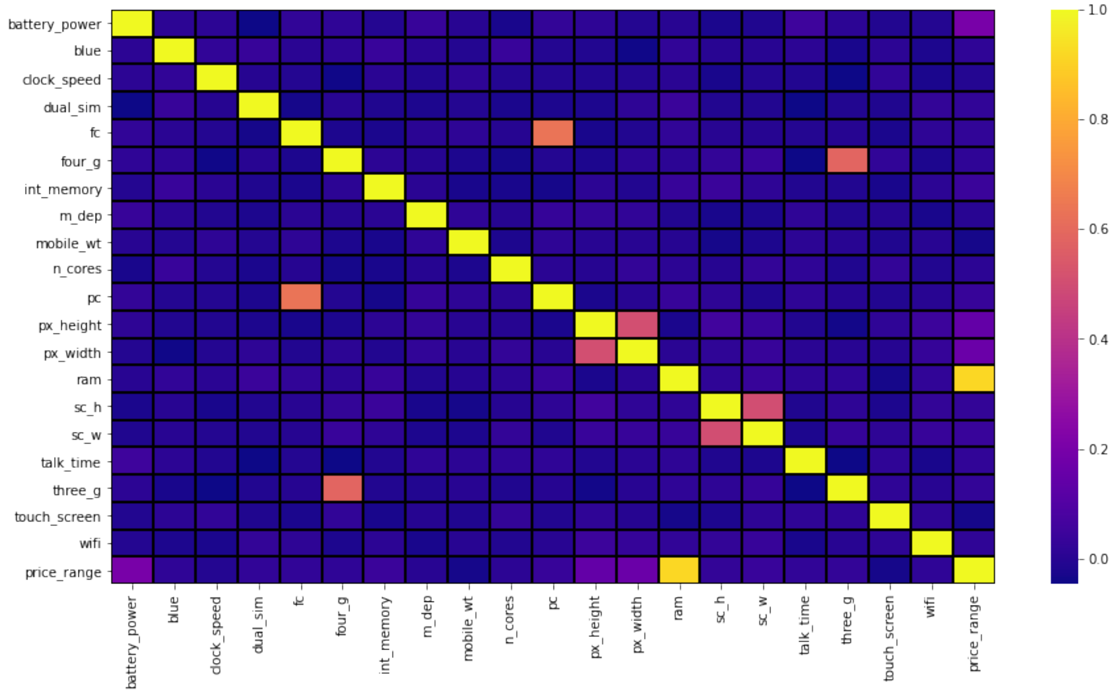
```
pc             -0.025150  0.001728  0.031317  0.011671 -0.017581
0.015342
px_height       1.000000  0.506294 -0.024568  0.055978  0.036888  -
0.011667
px_width        0.506294  1.000000  0.003795  0.018325  0.031277
0.004499
ram            -0.024568  0.003795  1.000000  0.017816  0.034949
0.011287
sc_h            0.055978  0.018325  0.017816  1.000000  0.504243  -
0.013949
sc_w            0.036888  0.031277  0.034949  0.504243  1.000000  -
0.020700
talk_time      -0.011667  0.004499  0.011287 -0.013949 -0.020700
1.000000
three_g        -0.034416 -0.001376  0.017870  0.011841  0.030629  -
0.044319
touch_screen    0.015557 -0.004146 -0.032817 -0.018110  0.010822
0.015270
wifi            0.046229  0.029086  0.022626  0.025284  0.033679  -
0.028608
price_range     0.144277  0.165132  0.917009  0.025641  0.038076
0.020582

                three_g  touch_screen      wifi  price_range
battery_power  0.010670     -0.010004 -0.009022     0.200763
blue          -0.029907      0.008115 -0.019082     0.015798
clock_speed   -0.044429      0.019023 -0.021960    -0.006120
dual_sim      -0.013474     -0.015209  0.024064     0.019016
fc            -0.003121     -0.024377  0.011902     0.021120
four_g         0.584754      0.020094 -0.021650     0.015906
int_memory    -0.009660     -0.028687  0.010588     0.043458
m_dep         -0.013723     -0.002823 -0.029474    -0.000925
mobile_wt      0.001730     -0.016019 -0.000862    -0.031628
n_cores       -0.015022      0.026642 -0.011145     0.008307
pc            -0.003256     -0.012834  0.001952     0.033871
px_height     -0.034416      0.015557  0.046229     0.144277
px_width      -0.001376     -0.004146  0.029086     0.165132
ram            0.017870     -0.032817  0.022626     0.917009
sc_h           0.011841     -0.018110  0.025284     0.025641
sc_w           0.030629      0.010822  0.033679     0.038076
talk_time     -0.044319      0.015270 -0.028608     0.020582
three_g        1.000000      0.015903  0.000933     0.025462
touch_screen   0.015903      1.000000  0.011081    -0.033888
wifi           0.000933      0.011081  1.000000     0.017192
price_range    0.025462     -0.033888  0.017192     1.000000

[21 rows x 21 columns]
```

```python
# Plotting pairplot to understand the data
plt.figure(figsize=(15,10))
sns.pairplot(data=df)
```

```
<seaborn.axisgrid.PairGrid at 0x1bbf3871e20>

<Figure size 1080x720 with 0 Axes>
```



```python
# With the help of heatmap it is easy to recognize different
correlation present in the data
plt.figure(figsize=(15,8))
sns.heatmap(df.corr(),annot=False,cmap='plasma',linewidths=1,linecolor
="black")
plt.show()
```



Following are the corelations present in the data

1. Between 'ram' and 'price_range' there is a 'Perfect Positive Correlation'
2. Between 'pc' and 'fc' and also between 'three_g' and 'four_g' there is 'Highly Positive Correlation'
3. Between ('battery_power', 'px_height', 'px_width' - 'price_range') respectively there is 'Low Positive Correlation'

Through above insights we can say that:

1. Higher the ram higher the price and also lower the ram lower the price vice versa
2. Same with pc-fc, if a mobile has higher front camera pixels then it is most likely possible that is has higher primary camera pixels and we can also conclude that if a mobile has four-g support then is is most likely possible that it may also have three-g support
3. There is slight positive change in price range with respect to battery power, pixel resolution height and pixel resolution width i.e. it is possible that all these three aspects can be the cause to increase in price of mobile

EDA

Separating Data

```
x = df.iloc[:,:-1]
x
```

```
      battery_power  blue  clock_speed  dual_sim  fc  four_g
int_memory  \
```

```
0               842     0          2.2          0   1       0
7
1              1021     1          0.5          1   0       1
53
2               563     1          0.5          1   2       1
41
3               615     1          2.5          0   0       0
10
4              1821     1          1.2          0  13       1
44
...             ...   ...          ...        ...  ..     ...           .
..
1995            794     1          0.5          1   0       1
2
1996           1965     1          2.6          1   0       0
39
1997           1911     0          0.9          1   1       1
36
1998           1512     0          0.9          0   4       1
46
1999            510     1          2.0          1   5       1
45

      m_dep  mobile_wt  n_cores  pc  px_height  px_width   ram  sc_h
sc_w  \
0       0.6        188        2   2         20       756  2549     9
7
1       0.7        136        3   6        905      1988  2631    17
3
2       0.9        145        5   6       1263      1716  2603    11
2
3       0.8        131        6   9       1216      1786  2769    16
8
4       0.6        141        2  14       1208      1212  1411     8
2
...     ...        ...      ...  ..        ...       ...   ...   ...
...
1995    0.8        106        6  14       1222      1890   668    13
4
1996    0.2        187        4   3        915      1965  2032    11
10
1997    0.7        108        8   3        868      1632  3057     9
1
1998    0.1        145        5   5        336       670   869    18
10
1999    0.9        168        6  16        483       754  3919    19
4

      talk_time  three_g  touch_screen  wifi
0            19        0             0     1
```

```
1              7        1             1     0
2              9        1             1     0
3             11        1             0     0
4             15        1             1     0
...          ...      ...           ...   ...
1995          19        1             1     0
1996          16        1             1     1
1997           5        1             1     0
1998          19        1             1     1
1999           2        1             1     1

[1980 rows x 20 columns]
```

```
y = df['price_range']
y
```

```
0       1
1       2
2       2
3       2
4       1
       ..
1995    0
1996    2
1997    3
1998    0
1999    3
Name: price_range, Length: 1980, dtype: int64
```

Splitting Data

```
xtrain, xtest, ytrain, ytest = 
train_test_split(x,y,test_size=0.3,random_state=1)
```

Model Building

```
def mymodel(model):
    '''
    -Function to train the model and then to predict
    -Also to get classification report and confusion matrix for the 
same
    '''
    model.fit(xtrain, ytrain)
    ypred = model.predict(xtest)
    print(f'Classification Report:\
n{classification_report(ytest,ypred)}\n')
    print(f'Confusion Matrix:\n{confusion_matrix(ytest,ypred)}')

# Creating objects of required models
knn = KNeighborsClassifier()
svm = SVC()
dt = DecisionTreeClassifier()
```

```
rf = RandomForestClassifier()
gnb = GaussianNB()
```

```
mymodel(knn)
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.95      0.96       124
           1       0.88      0.94      0.91       142
           2       0.92      0.86      0.89       168
           3       0.93      0.94      0.93       160

    accuracy                           0.92       594
   macro avg       0.92      0.92      0.92       594
weighted avg       0.92      0.92      0.92       594


Confusion Matrix:
[[118   6   0   0]
 [  5 134   3   0]
 [  0  12 144  12]
 [  0   0  10 150]]
```

```
mymodel(svm)
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.97      0.97       124
           1       0.91      0.96      0.94       142
           2       0.97      0.86      0.91       168
           3       0.92      0.99      0.95       160

    accuracy                           0.94       594
   macro avg       0.94      0.94      0.94       594
weighted avg       0.94      0.94      0.94       594


Confusion Matrix:
[[120   4   0   0]
 [  3 137   2   0]
 [  0  10 144  14]
 [  0   0   2 158]]
```

```
mymodel(dt)
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.93      0.90       124
```

```
           1       0.82      0.77      0.80       142
           2       0.78      0.78      0.78       168
           3       0.86      0.86      0.86       160

    accuracy                           0.83       594
   macro avg       0.83      0.84      0.83       594
weighted avg       0.83      0.83      0.83       594


Confusion Matrix:
[[115   9   0   0]
 [ 16 110  16   0]
 [  0  14 131  23]
 [  0   1  21 138]]

mymodel(rf)

Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.92      0.91       124
           1       0.76      0.82      0.79       142
           2       0.85      0.74      0.79       168
           3       0.89      0.93      0.91       160

    accuracy                           0.85       594
   macro avg       0.85      0.85      0.85       594
weighted avg       0.85      0.85      0.85       594


Confusion Matrix:
[[114  10   0   0]
 [ 13 117  12   0]
 [  0  25 125  18]
 [  0   1  10 149]]

mymodel(gnb)

Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.90      0.88       124
           1       0.67      0.66      0.66       142
           2       0.72      0.69      0.71       168
           3       0.89      0.92      0.90       160

    accuracy                           0.79       594
   macro avg       0.79      0.79      0.79       594
weighted avg       0.79      0.79      0.79       594
```

```
Confusion Matrix:
[[111  13   0   0]
 [ 16  94  32   0]
 [  0  33 116  19]
 [  0   1  12 147]]
```

SVM has performed best from all the other models so now will hypertune the SVM model and see if it improves the accuracy of a model

Hyperparameter Tunning

```
param_grid = {"kernel":['poly', 'rbf', 'sigmoid'],
 "C":[0.1,0.01,0.001,0.0001],
 "gamma":[0.1,0.01,0.001,0.0001]}

GS = GridSearchCV(svm, param_grid, verbose=3)
GS.fit(xtrain,ytrain)
ypred = GS.predict(xtest)
```

```
Fitting 5 folds for each of 48 candidates, totalling 240 fits
[CV] C=0.1, gamma=0.1, kernel=poly ...................................
[CV] ....... C=0.1, gamma=0.1, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=poly ...................................
[CV] ....... C=0.1, gamma=0.1, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=poly ...................................
[CV] ....... C=0.1, gamma=0.1, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=poly ...................................
[CV] ....... C=0.1, gamma=0.1, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=poly ...................................
[CV] ....... C=0.1, gamma=0.1, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.1, gamma=0.1, kernel=rbf ....................................

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.0s
remaining:    0.0s

[CV] ........ C=0.1, gamma=0.1, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.1, gamma=0.1, kernel=rbf ....................................
[CV] ........ C=0.1, gamma=0.1, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.1, gamma=0.1, kernel=rbf ....................................
[CV] ........ C=0.1, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.1, kernel=rbf ....................................
[CV] ........ C=0.1, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.1, kernel=rbf ....................................
[CV] ........ C=0.1, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.1, kernel=sigmoid ................................
[CV] .... C=0.1, gamma=0.1, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.1, gamma=0.1, kernel=sigmoid ................................
```

```
[CV] .... C=0.1, gamma=0.1, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.1, gamma=0.1, kernel=sigmoid .................................
[CV] .... C=0.1, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.1, kernel=sigmoid .................................
[CV] .... C=0.1, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.1, kernel=sigmoid .................................
[CV] .... C=0.1, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.01, kernel=poly ...................................
[CV] ...... C=0.1, gamma=0.01, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=poly ...................................
[CV] ...... C=0.1, gamma=0.01, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=poly ...................................
[CV] ...... C=0.1, gamma=0.01, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=poly ...................................
[CV] ...... C=0.1, gamma=0.01, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=poly ...................................
[CV] ...... C=0.1, gamma=0.01, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.1, gamma=0.01, kernel=rbf ....................................
[CV] ....... C=0.1, gamma=0.01, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.1, gamma=0.01, kernel=rbf ....................................
[CV] ....... C=0.1, gamma=0.01, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.1, gamma=0.01, kernel=rbf ....................................
[CV] ....... C=0.1, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.01, kernel=rbf ....................................
[CV] ....... C=0.1, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.01, kernel=rbf ....................................
[CV] ....... C=0.1, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.01, kernel=sigmoid ................................
[CV] ... C=0.1, gamma=0.01, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.1, gamma=0.01, kernel=sigmoid ................................
[CV] ... C=0.1, gamma=0.01, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.1, gamma=0.01, kernel=sigmoid ................................
[CV] ... C=0.1, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.01, kernel=sigmoid ................................
[CV] ... C=0.1, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.01, kernel=sigmoid ................................
[CV] ... C=0.1, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.001, kernel=poly ..................................
[CV] ..... C=0.1, gamma=0.001, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=poly ..................................
[CV] ..... C=0.1, gamma=0.001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=poly ..................................
[CV] ..... C=0.1, gamma=0.001, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=poly ..................................
[CV] ..... C=0.1, gamma=0.001, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=poly ..................................
[CV] ..... C=0.1, gamma=0.001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.1, gamma=0.001, kernel=rbf ...................................
[CV] ...... C=0.1, gamma=0.001, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.1, gamma=0.001, kernel=rbf ...................................
```

```
[CV] ...... C=0.1, gamma=0.001, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.1, gamma=0.001, kernel=rbf ..................................
[CV] ...... C=0.1, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.001, kernel=rbf ..................................
[CV] ...... C=0.1, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.001, kernel=rbf ..................................
[CV] ...... C=0.1, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.1, gamma=0.001, kernel=sigmoid ..............................
[CV] .. C=0.1, gamma=0.001, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.1, gamma=0.001, kernel=sigmoid ..............................
[CV] .. C=0.1, gamma=0.001, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.1, gamma=0.001, kernel=sigmoid ..............................
[CV] .. C=0.1, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.001, kernel=sigmoid ..............................
[CV] .. C=0.1, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.001, kernel=sigmoid ..............................
[CV] .. C=0.1, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=poly ................................
[CV] .... C=0.1, gamma=0.0001, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.1, gamma=0.0001, kernel=poly ................................
[CV] .... C=0.1, gamma=0.0001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.1, gamma=0.0001, kernel=poly ................................
[CV] .... C=0.1, gamma=0.0001, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.1, gamma=0.0001, kernel=poly ................................
[CV] .... C=0.1, gamma=0.0001, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.1, gamma=0.0001, kernel=poly ................................
[CV] .... C=0.1, gamma=0.0001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.1, gamma=0.0001, kernel=rbf .................................
[CV] ..... C=0.1, gamma=0.0001, kernel=rbf, score=0.270, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=rbf .................................
[CV] ..... C=0.1, gamma=0.0001, kernel=rbf, score=0.271, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=rbf .................................
[CV] ..... C=0.1, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=rbf .................................
[CV] ..... C=0.1, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=rbf .................................
[CV] ..... C=0.1, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=sigmoid .............................
[CV] . C=0.1, gamma=0.0001, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=sigmoid .............................
[CV] . C=0.1, gamma=0.0001, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=sigmoid .............................
[CV] . C=0.1, gamma=0.0001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=sigmoid .............................
[CV] . C=0.1, gamma=0.0001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.1, gamma=0.0001, kernel=sigmoid .............................
[CV] . C=0.1, gamma=0.0001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.1, kernel=poly ..................................
[CV] ...... C=0.01, gamma=0.1, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.01, gamma=0.1, kernel=poly ..................................
```

```
[CV] ...... C=0.01, gamma=0.1, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.01, gamma=0.1, kernel=poly ..................................
[CV] ...... C=0.01, gamma=0.1, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.01, gamma=0.1, kernel=poly ..................................
[CV] ...... C=0.01, gamma=0.1, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.01, gamma=0.1, kernel=poly ..................................
[CV] ...... C=0.01, gamma=0.1, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.01, gamma=0.1, kernel=rbf ...................................

[CV] ....... C=0.01, gamma=0.1, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.01, gamma=0.1, kernel=rbf ...................................
[CV] ....... C=0.01, gamma=0.1, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.01, gamma=0.1, kernel=rbf ...................................
[CV] ....... C=0.01, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.1, kernel=rbf ...................................
[CV] ....... C=0.01, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.1, kernel=rbf ...................................
[CV] ....... C=0.01, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.1, kernel=sigmoid ...............................
[CV] ... C=0.01, gamma=0.1, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.01, gamma=0.1, kernel=sigmoid ...............................
[CV] ... C=0.01, gamma=0.1, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.01, gamma=0.1, kernel=sigmoid ...............................
[CV] ... C=0.01, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.1, kernel=sigmoid ...............................
[CV] ... C=0.01, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.1, kernel=sigmoid ...............................
[CV] ... C=0.01, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.01, kernel=poly .................................
[CV] ..... C=0.01, gamma=0.01, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.01, gamma=0.01, kernel=poly .................................
[CV] ..... C=0.01, gamma=0.01, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.01, gamma=0.01, kernel=poly .................................
[CV] ..... C=0.01, gamma=0.01, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.01, gamma=0.01, kernel=poly .................................
[CV] ..... C=0.01, gamma=0.01, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.01, gamma=0.01, kernel=poly .................................
[CV] ..... C=0.01, gamma=0.01, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.01, gamma=0.01, kernel=rbf ..................................
[CV] ...... C=0.01, gamma=0.01, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.01, gamma=0.01, kernel=rbf ..................................
[CV] ...... C=0.01, gamma=0.01, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.01, gamma=0.01, kernel=rbf ..................................
[CV] ...... C=0.01, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.01, kernel=rbf ..................................
[CV] ...... C=0.01, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.01, kernel=rbf ..................................
[CV] ...... C=0.01, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.01, kernel=sigmoid ..............................
[CV] .. C=0.01, gamma=0.01, kernel=sigmoid, score=0.270, total=   0.1s
```

```
[CV] C=0.01, gamma=0.01, kernel=sigmoid ..............................
[CV] .. C=0.01, gamma=0.01, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.01, gamma=0.01, kernel=sigmoid ..............................
[CV] .. C=0.01, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.01, kernel=sigmoid ..............................
[CV] .. C=0.01, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.01, kernel=sigmoid ..............................
[CV] .. C=0.01, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.001, kernel=poly ................................
[CV] .... C=0.01, gamma=0.001, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.01, gamma=0.001, kernel=poly ................................
[CV] .... C=0.01, gamma=0.001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.01, gamma=0.001, kernel=poly ................................
[CV] .... C=0.01, gamma=0.001, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.01, gamma=0.001, kernel=poly ................................
[CV] .... C=0.01, gamma=0.001, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.01, gamma=0.001, kernel=poly ................................
[CV] .... C=0.01, gamma=0.001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.01, gamma=0.001, kernel=rbf .................................
[CV] ..... C=0.01, gamma=0.001, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.01, gamma=0.001, kernel=rbf .................................
[CV] ..... C=0.01, gamma=0.001, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.01, gamma=0.001, kernel=rbf .................................
[CV] ..... C=0.01, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.001, kernel=rbf .................................
[CV] ..... C=0.01, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.001, kernel=rbf .................................
[CV] ..... C=0.01, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.01, gamma=0.001, kernel=sigmoid .............................
[CV] . C=0.01, gamma=0.001, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.01, gamma=0.001, kernel=sigmoid .............................
[CV] . C=0.01, gamma=0.001, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.01, gamma=0.001, kernel=sigmoid .............................
[CV] . C=0.01, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.001, kernel=sigmoid .............................
[CV] . C=0.01, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.001, kernel=sigmoid .............................
[CV] . C=0.01, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=poly ...............................
[CV] ... C=0.01, gamma=0.0001, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.01, gamma=0.0001, kernel=poly ...............................
[CV] ... C=0.01, gamma=0.0001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.01, gamma=0.0001, kernel=poly ...............................
[CV] ... C=0.01, gamma=0.0001, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.01, gamma=0.0001, kernel=poly ...............................
[CV] ... C=0.01, gamma=0.0001, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.01, gamma=0.0001, kernel=poly ...............................
[CV] ... C=0.01, gamma=0.0001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.01, gamma=0.0001, kernel=rbf ................................
[CV] .... C=0.01, gamma=0.0001, kernel=rbf, score=0.270, total=   0.1s
```

```
[CV] C=0.01, gamma=0.0001, kernel=rbf ................................
[CV] .... C=0.01, gamma=0.0001, kernel=rbf, score=0.271, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=rbf ................................
[CV] .... C=0.01, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=rbf ................................
[CV] .... C=0.01, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=rbf ................................
[CV] .... C=0.01, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=sigmoid ............................
[CV]  C=0.01, gamma=0.0001, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=sigmoid ............................
[CV]  C=0.01, gamma=0.0001, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=sigmoid ............................
[CV]  C=0.01, gamma=0.0001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=sigmoid ............................
[CV]  C=0.01, gamma=0.0001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.01, gamma=0.0001, kernel=sigmoid ............................
[CV]  C=0.01, gamma=0.0001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.1, kernel=poly .................................
[CV] ..... C=0.001, gamma=0.1, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.001, gamma=0.1, kernel=poly .................................
[CV] ..... C=0.001, gamma=0.1, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.001, gamma=0.1, kernel=poly .................................
[CV] ..... C=0.001, gamma=0.1, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.001, gamma=0.1, kernel=poly .................................
[CV] ..... C=0.001, gamma=0.1, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.001, gamma=0.1, kernel=poly .................................
[CV] ..... C=0.001, gamma=0.1, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.001, gamma=0.1, kernel=rbf ..................................

[CV] ...... C=0.001, gamma=0.1, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.001, gamma=0.1, kernel=rbf ..................................
[CV] ...... C=0.001, gamma=0.1, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.001, gamma=0.1, kernel=rbf ..................................
[CV] ...... C=0.001, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.1, kernel=rbf ..................................
[CV] ...... C=0.001, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.1, kernel=rbf ..................................
[CV] ...... C=0.001, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.1, kernel=sigmoid ..............................
[CV] .. C=0.001, gamma=0.1, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.001, gamma=0.1, kernel=sigmoid ..............................
[CV] .. C=0.001, gamma=0.1, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.001, gamma=0.1, kernel=sigmoid ..............................
[CV] .. C=0.001, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.1, kernel=sigmoid ..............................
[CV] .. C=0.001, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.1, kernel=sigmoid ..............................
[CV] .. C=0.001, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.01, kernel=poly ................................
```

```
[CV] .... C=0.001, gamma=0.01, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.001, gamma=0.01, kernel=poly ................................
[CV] .... C=0.001, gamma=0.01, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.001, gamma=0.01, kernel=poly ................................
[CV] .... C=0.001, gamma=0.01, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.001, gamma=0.01, kernel=poly ................................
[CV] .... C=0.001, gamma=0.01, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.001, gamma=0.01, kernel=poly ................................
[CV] .... C=0.001, gamma=0.01, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.001, gamma=0.01, kernel=rbf .................................
[CV] ..... C=0.001, gamma=0.01, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.001, gamma=0.01, kernel=rbf .................................
[CV] ..... C=0.001, gamma=0.01, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.001, gamma=0.01, kernel=rbf .................................
[CV] ..... C=0.001, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.01, kernel=rbf .................................
[CV] ..... C=0.001, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.01, kernel=rbf .................................
[CV] ..... C=0.001, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.01, kernel=sigmoid .............................
[CV] . C=0.001, gamma=0.01, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.001, gamma=0.01, kernel=sigmoid .............................
[CV] . C=0.001, gamma=0.01, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.001, gamma=0.01, kernel=sigmoid .............................
[CV] . C=0.001, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.01, kernel=sigmoid .............................
[CV] . C=0.001, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.01, kernel=sigmoid .............................
[CV] . C=0.001, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.001, kernel=poly ...............................
[CV] ... C=0.001, gamma=0.001, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.001, gamma=0.001, kernel=poly ...............................
[CV] ... C=0.001, gamma=0.001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.001, gamma=0.001, kernel=poly ...............................
[CV] ... C=0.001, gamma=0.001, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.001, gamma=0.001, kernel=poly ...............................
[CV] ... C=0.001, gamma=0.001, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.001, gamma=0.001, kernel=poly ...............................
[CV] ... C=0.001, gamma=0.001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.001, gamma=0.001, kernel=rbf ................................
[CV] .... C=0.001, gamma=0.001, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.001, gamma=0.001, kernel=rbf ................................
[CV] .... C=0.001, gamma=0.001, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.001, gamma=0.001, kernel=rbf ................................
[CV] .... C=0.001, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.001, kernel=rbf ................................
[CV] .... C=0.001, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.001, kernel=rbf ................................
[CV] .... C=0.001, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.001, gamma=0.001, kernel=sigmoid ............................
```

```
[CV]  C=0.001, gamma=0.001, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.001, gamma=0.001, kernel=sigmoid ............................
[CV]  C=0.001, gamma=0.001, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.001, gamma=0.001, kernel=sigmoid ............................
[CV]  C=0.001, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.001, kernel=sigmoid ............................
[CV]  C=0.001, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.001, kernel=sigmoid ............................
[CV]  C=0.001, gamma=0.001, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.0001, kernel=poly ..............................
[CV] .. C=0.001, gamma=0.0001, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.001, gamma=0.0001, kernel=poly ..............................
[CV] .. C=0.001, gamma=0.0001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.001, gamma=0.0001, kernel=poly ..............................
[CV] .. C=0.001, gamma=0.0001, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.001, gamma=0.0001, kernel=poly ..............................
[CV] .. C=0.001, gamma=0.0001, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.001, gamma=0.0001, kernel=poly ..............................
[CV] .. C=0.001, gamma=0.0001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.001, gamma=0.0001, kernel=rbf ...............................
[CV] ... C=0.001, gamma=0.0001, kernel=rbf, score=0.270, total=   0.1s
[CV] C=0.001, gamma=0.0001, kernel=rbf ...............................
[CV] ... C=0.001, gamma=0.0001, kernel=rbf, score=0.271, total=   0.1s
[CV] C=0.001, gamma=0.0001, kernel=rbf ...............................
[CV] ... C=0.001, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.0001, kernel=rbf ...............................
[CV] ... C=0.001, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.0001, kernel=rbf ...............................
[CV] ... C=0.001, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.001, gamma=0.0001, kernel=sigmoid ...........................
[CV]  C=0.001, gamma=0.0001, kernel=sigmoid, score=0.270, total=
0.1s
[CV] C=0.001, gamma=0.0001, kernel=sigmoid ...........................
[CV]  C=0.001, gamma=0.0001, kernel=sigmoid, score=0.271, total=
0.1s
[CV] C=0.001, gamma=0.0001, kernel=sigmoid ...........................
[CV]  C=0.001, gamma=0.0001, kernel=sigmoid, score=0.267, total=
0.1s
[CV] C=0.001, gamma=0.0001, kernel=sigmoid ...........................
[CV]  C=0.001, gamma=0.0001, kernel=sigmoid, score=0.267, total=
0.1s
[CV] C=0.001, gamma=0.0001, kernel=sigmoid ...........................
[CV]  C=0.001, gamma=0.0001, kernel=sigmoid, score=0.267, total=
0.1s
[CV] C=0.0001, gamma=0.1, kernel=poly ................................
[CV] .... C=0.0001, gamma=0.1, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.0001, gamma=0.1, kernel=poly ................................
[CV] .... C=0.0001, gamma=0.1, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.0001, gamma=0.1, kernel=poly ................................
[CV] .... C=0.0001, gamma=0.1, kernel=poly, score=0.978, total=   0.0s
```

```
[CV] C=0.0001, gamma=0.1, kernel=poly .................................
[CV] .... C=0.0001, gamma=0.1, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.0001, gamma=0.1, kernel=poly .................................
[CV] .... C=0.0001, gamma=0.1, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.0001, gamma=0.1, kernel=rbf ..................................

[CV] ..... C=0.0001, gamma=0.1, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.0001, gamma=0.1, kernel=rbf ..................................
[CV] ..... C=0.0001, gamma=0.1, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.0001, gamma=0.1, kernel=rbf ..................................
[CV] ..... C=0.0001, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.0001, gamma=0.1, kernel=rbf ..................................
[CV] ..... C=0.0001, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.0001, gamma=0.1, kernel=rbf ..................................
[CV] ..... C=0.0001, gamma=0.1, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.0001, gamma=0.1, kernel=sigmoid ..............................
[CV] . C=0.0001, gamma=0.1, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.0001, gamma=0.1, kernel=sigmoid ..............................
[CV] . C=0.0001, gamma=0.1, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.0001, gamma=0.1, kernel=sigmoid ..............................
[CV] . C=0.0001, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.1, kernel=sigmoid ..............................
[CV] . C=0.0001, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.1, kernel=sigmoid ..............................
[CV] . C=0.0001, gamma=0.1, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.01, kernel=poly ................................
[CV] ... C=0.0001, gamma=0.01, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.0001, gamma=0.01, kernel=poly ................................
[CV] ... C=0.0001, gamma=0.01, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.0001, gamma=0.01, kernel=poly ................................
[CV] ... C=0.0001, gamma=0.01, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.0001, gamma=0.01, kernel=poly ................................
[CV] ... C=0.0001, gamma=0.01, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.0001, gamma=0.01, kernel=poly ................................
[CV] ... C=0.0001, gamma=0.01, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.0001, gamma=0.01, kernel=rbf .................................
[CV] .... C=0.0001, gamma=0.01, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.0001, gamma=0.01, kernel=rbf .................................
[CV] .... C=0.0001, gamma=0.01, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.0001, gamma=0.01, kernel=rbf .................................
[CV] .... C=0.0001, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.0001, gamma=0.01, kernel=rbf .................................
[CV] .... C=0.0001, gamma=0.01, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.0001, gamma=0.01, kernel=rbf .................................
[CV] .... C=0.0001, gamma=0.01, kernel=rbf, score=0.267, total=   0.3s
[CV] C=0.0001, gamma=0.01, kernel=sigmoid .............................
[CV]  C=0.0001, gamma=0.01, kernel=sigmoid, score=0.270, total=   0.1s
[CV] C=0.0001, gamma=0.01, kernel=sigmoid .............................
[CV]  C=0.0001, gamma=0.01, kernel=sigmoid, score=0.271, total=   0.1s
[CV] C=0.0001, gamma=0.01, kernel=sigmoid .............................
```

```
[CV]  C=0.0001, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.01, kernel=sigmoid ............................
[CV]  C=0.0001, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.01, kernel=sigmoid ............................
[CV]  C=0.0001, gamma=0.01, kernel=sigmoid, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.001, kernel=poly ..............................
[CV] .. C=0.0001, gamma=0.001, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.0001, gamma=0.001, kernel=poly ..............................
[CV] .. C=0.0001, gamma=0.001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.0001, gamma=0.001, kernel=poly ..............................
[CV] .. C=0.0001, gamma=0.001, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.0001, gamma=0.001, kernel=poly ..............................
[CV] .. C=0.0001, gamma=0.001, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.0001, gamma=0.001, kernel=poly ..............................
[CV] .. C=0.0001, gamma=0.001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.0001, gamma=0.001, kernel=rbf ...............................
[CV] ... C=0.0001, gamma=0.001, kernel=rbf, score=0.270, total=   0.2s
[CV] C=0.0001, gamma=0.001, kernel=rbf ...............................
[CV] ... C=0.0001, gamma=0.001, kernel=rbf, score=0.271, total=   0.2s
[CV] C=0.0001, gamma=0.001, kernel=rbf ...............................
[CV] ... C=0.0001, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.0001, gamma=0.001, kernel=rbf ...............................
[CV] ... C=0.0001, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.0001, gamma=0.001, kernel=rbf ...............................
[CV] ... C=0.0001, gamma=0.001, kernel=rbf, score=0.267, total=   0.2s
[CV] C=0.0001, gamma=0.001, kernel=sigmoid ...........................
[CV]  C=0.0001, gamma=0.001, kernel=sigmoid, score=0.270, total=
0.1s
[CV] C=0.0001, gamma=0.001, kernel=sigmoid ...........................
[CV]  C=0.0001, gamma=0.001, kernel=sigmoid, score=0.271, total=
0.1s
[CV] C=0.0001, gamma=0.001, kernel=sigmoid ...........................
[CV]  C=0.0001, gamma=0.001, kernel=sigmoid, score=0.267, total=
0.1s
[CV] C=0.0001, gamma=0.001, kernel=sigmoid ...........................
[CV]  C=0.0001, gamma=0.001, kernel=sigmoid, score=0.267, total=
0.1s
[CV] C=0.0001, gamma=0.001, kernel=sigmoid ...........................
[CV]  C=0.0001, gamma=0.001, kernel=sigmoid, score=0.267, total=
0.1s
[CV] C=0.0001, gamma=0.0001, kernel=poly .............................
[CV] . C=0.0001, gamma=0.0001, kernel=poly, score=0.950, total=   0.0s
[CV] C=0.0001, gamma=0.0001, kernel=poly .............................
[CV] . C=0.0001, gamma=0.0001, kernel=poly, score=0.971, total=   0.0s
[CV] C=0.0001, gamma=0.0001, kernel=poly .............................
[CV] . C=0.0001, gamma=0.0001, kernel=poly, score=0.978, total=   0.0s
[CV] C=0.0001, gamma=0.0001, kernel=poly .............................
[CV] . C=0.0001, gamma=0.0001, kernel=poly, score=0.968, total=   0.0s
[CV] C=0.0001, gamma=0.0001, kernel=poly .............................
[CV] . C=0.0001, gamma=0.0001, kernel=poly, score=0.971, total=   0.0s
```

```
[CV] C=0.0001, gamma=0.0001, kernel=rbf ..............................
[CV] .. C=0.0001, gamma=0.0001, kernel=rbf, score=0.270, total=   0.1s
[CV] C=0.0001, gamma=0.0001, kernel=rbf ..............................
[CV] .. C=0.0001, gamma=0.0001, kernel=rbf, score=0.271, total=   0.1s
[CV] C=0.0001, gamma=0.0001, kernel=rbf ..............................
[CV] .. C=0.0001, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.0001, kernel=rbf ..............................
[CV] .. C=0.0001, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.0001, kernel=rbf ..............................
[CV] .. C=0.0001, gamma=0.0001, kernel=rbf, score=0.267, total=   0.1s
[CV] C=0.0001, gamma=0.0001, kernel=sigmoid ..........................
[CV]  C=0.0001, gamma=0.0001, kernel=sigmoid, score=0.270, total=
0.1s
[CV] C=0.0001, gamma=0.0001, kernel=sigmoid ..........................
[CV]  C=0.0001, gamma=0.0001, kernel=sigmoid, score=0.271, total=
0.1s
[CV] C=0.0001, gamma=0.0001, kernel=sigmoid ..........................
[CV]  C=0.0001, gamma=0.0001, kernel=sigmoid, score=0.267, total=
0.1s
[CV] C=0.0001, gamma=0.0001, kernel=sigmoid ..........................
[CV]  C=0.0001, gamma=0.0001, kernel=sigmoid, score=0.267, total=
0.1s
[CV] C=0.0001, gamma=0.0001, kernel=sigmoid ..........................
[CV]  C=0.0001, gamma=0.0001, kernel=sigmoid, score=0.267, total=
0.1s

[Parallel(n_jobs=1)]: Done 240 out of 240 | elapsed:   22.3s finished
```

```python
# Finding the best parameters
GS.best_params_
```

```
{'C': 0.1, 'gamma': 0.1, 'kernel': 'poly'}
```

```python
print(f'Classification Report:\n{classification_report(ytest,ypred)}\
n')
print(f'Confusion Matrix:\n{confusion_matrix(ytest,ypred)}')
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.96      0.96       124
           1       0.93      0.94      0.94       142
           2       0.94      0.93      0.94       168
           3       0.96      0.96      0.96       160

    accuracy                           0.95       594
   macro avg       0.95      0.95      0.95       594
weighted avg       0.95      0.95      0.95       594


Confusion Matrix:
```

```
[[119   5   0   0]
 [  4 134   4   0]
 [  0   5 157   6]
 [  0   0   6 154]]
```

Conclusion

- After hypertunning the model we have seen increase in the accuracy of model
- Now bob can decide the price range of mobile phones for his company