NAME: M. Siri Chandana

HT.NO:2203A51492

BATCH:12

AIML ASSIGNMENT-6

Question 1:
https://www.kaggle.com/datasets/iabhishekofficial/mobile-price-classification?select=train.csv
From the above data:
a) Read the data with pandas and find features and target variables
b) Normalize the data with min-max scaling
c) Split the data into train and test.

Question 2:
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. And provide the results like accuracy, precision recall and confusion
matrix.
https://www.kaggle.com/datasets/iabhishekofficial/mobile-price-classification?select=train.csv.

```
yp=model.predict(x_test_scaled)
```

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix
```

```
accuracy=accuracy_score(y_test,yp)
print(accuracy)
```

```
0.95
```

```
precision=precision_score(y_test,yp,average='weighted')
print(precision)
```

```
0.9499290394431786
```

```
recall=recall_score(y_test,yp,average='weighted')
print(recall)
```

```
0.95
```

```
1350    3
Name: price_range, Length: 1400, dtype: int64
```

```
from sklearn.preprocessing import StandardScaler
```

```
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

```
from sklearn.linear_model import LogisticRegression
```

```
model=LogisticRegression()
model.fit(x_train_scaled,y_train)
```

```
yp=model.predict(x_test_scaled)
```

```
[26] print(y_test)
```

```
423     0
1495    0
1618    2
1099    0
1307    0
       ..
14      0
282     1
952     3
1079    2
486     2
Name: price_range, Length: 600, dtype: int64
```

```
[27] print(y_train)
```

```
993     0
1156    3
615     2
703     3
1130    3
       ..
1016    0
165     3
7       0
219     3
1350    3
```

```
print(x_test)
```

```
      battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  \
423            1681     1          2.5         0   2       0          11
1495           1472     0          3.0         0   4       1          20
1618            502     0          0.8         0   7       0          52
1099           1697     0          0.5         0   0       1          60
1307            831     0          1.7         1   7       1          26
...             ...   ...          ...       ...  ..     ...         ...
14             1866     0          0.5         0  13       1          52
282            1839     1          1.2         0   9       1          54
952            1444     1          2.1         0   9       0          38
1079           1893     1          0.5         1   1       0          23
486            1089     1          0.9         1  12       1           2

      m_dep  mobile_wt  n_cores  pc  px_height  px_width   ram  sc_h  sc_w  \
423     0.4        158        2  13        195      1205  1122    12     6
1495    0.3        169        2   6        443       892   797     6     1
1618    1.0         82        6   8        281      1159  2666     5     4
1099    0.1         90        4   0         88      1046   441    15     1
1307    0.7        177        5  11        511       621  1704     6     5
...     ...        ...      ...  ..        ...       ...   ...   ...   ...
14      0.7        185        1  17        356       563   373    14     9
282     0.5        200        7  11        475      1493   927    19    10
952     0.4        104        7  16        624       917  3764    14     9
1079    0.1        179        8   3       1203      1432  1482    15     7
486     0.7        145        5  15        636      1259  2765    13    12
...
1079         17        0            1     0
486          10        1            0     1

[600 rows x 20 columns]
```

```
print(x_train)
```

```
      battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  \
993            686     1          0.5         0  11       0           3
1156          1732     0          0.8         0   2       0          61
615            880     0          0.5         1   1       0          44
703           1413     0          0.5         1   4       1          39
1130          1975     1          1.9         1   2       0          31
...            ...   ...          ...       ...  ..     ...         ...
1016           551     1          2.8         0   0       1          54
165            517     0          1.4         1   3       1          33
7             1954     0          0.5         1   0       0          24
219           1551     0          1.1         0   4       0          51
1350          1398     0          1.6         1   8       1          26

      m_dep  mobile_wt  n_cores  pc  px_height  px_width   ram  sc_h  sc_w  \
993     0.3         91        6  15       1109      1392   570     7     6
1156    0.3        172        5   3        201       656  3940    17    11
615     0.5        172        8  15        436      1302  3132     8     7
703     0.1        185        5  12       1039      1318  3878    19    16
1130    0.9        151        1  17        775      1607  3022    13     5
...     ...        ...      ...  ..        ...       ...   ...   ...   ...
1016    0.1        172        7  15        169      1916  1414     6     1
165     0.8        183        4   8        660       974  3704    17    16
7       0.8        187        4   0        512      1149   700    16     3
219     0.1         88        5   6       1738      1995  3844    11     8
1350    0.8        150        1  12        755      1284  3488    14     3
...
219           4        0            1     0
1350         11        1            1     0
```

```
from sklearn.preprocessing import MinMaxScaler
```

```
d=MinMaxScaler()
```

```
y=train['price_range']
x=train.drop('price_range',axis=1)
print(x)
print(y)
```

```
      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
 19  wifi           2000 non-null   int64
 20  price_range    2000 non-null   int64
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```

```
print(train.isnull().sum())
```

```
battery_power    0
blue             0
clock_speed      0
dual_sim         0
fc               0
four_g           0
int_memory       0
m_dep            0
mobile_wt        0
n_cores          0
pc               0
px_height        0
px_width         0
ram              0
sc_h             0
sc_w             0
talk_time        0
three_g          0
touch_screen     0
wifi             0
price_range      0
dtype: int64
```

```
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
...
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]
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```
train.info()
```

[5]

```
<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
```

```
import pandas as pd
```

```
train=pd.read_csv('/content/train.csv')
print(train.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
```

[49]
```
confusion=confusion_matrix(y_test,yp)
print(confusion)
```

```
[[159   4   0   0]
 [  6 140   4   0]
 [  0   3 128   8]
 [  0   0   5 143]]
```