ML ASSIGNMENT-3

NAME:SATHWIKA.P

ROLLNO:2203A51384

```
import pandas as pd

# Read the data from the provided Kaggle link
data = pd.read_csv("/content/train.csv")

# Identify features (X) and target variable (y)
X = data.drop(columns=["price_range"])  #
Features y = data["price_range"]  # Target
variable print("Features:") print(X)
print("\nTarget Variable:") print(y)
```

```
    Features:
          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

[2000 rows x 20 columns]

Target Variable:
0       1
1       2
2       2
3       2
4       1        ..
1995    0
1996    2
1997    3
1998    0
1999    3
Name: price_range, Length: 2000, dtype: int64
```

```
import pandas as pd

# Read the data from the provided Kaggle

link data = pd.read_csv("/content/train.csv")

from sklearn.preprocessing import

MinMaxScaler

scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)
print("Data after Min-Max scaling:")
print(X_scaled)
```

```
Data after Min-Max scaling:
[[0.22778891 0.         0.68       ... 0.         0.         1.        ]
 [0.34736139 1.         0.         ... 1.         1.         0.        ]
[0.04141617 1.         0.         ... 1.         1.
0.        ]  ...
 [0.94188377 0.         0.16       ... 1.         1.         0.        ]
 [0.6753507  0.         0.16       ... 1.         1.         1.        ]
 [0.00601202 1.         0.6        ... 1.         1.         1.        ]]
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
print(X_train, X_test, y_train, y_test )
```

```
[[0.9498998  0.         0.         ... 1.         1.         1.       ]
 [0.08817635 1.         0.68       ... 1.         1.         1.       ]
 [0.49098196 0.         0.16       ... 1.         0.         1.       ]
 ...
 [0.88710755 1.         0.         ... 1.         0.         1.       ]
 [0.95257181 0.         0.16       ... 0.         1.         1.       ]
 [0.08951236 1.         0.04       ... 1.         0.         0.       ]] [[0.76486306 0.         0.8
 [0.45490982 0.         0.         ... 1.         0.         0.       ]
 [0.98263193 0.         0.96       ... 1.         1.         0.       ]
 ...
 [0.36272545 0.         0.52       ... 1.         1.         1.       ]
 [0.15163661 0.         0.88       ... 1.         1.         1.       ]
 [0.45691383 0.         0.56       ... 1.         1.         1.       ]] 968     1
240     2
819     0
692     3
420     1
       ..
1130    3
1294    0
860     2
1459    3
1126    1
Name: price_range, Length: 1600, dtype: int64 1860    0
353     2
1333    1
905     3
1289    1
       ..
965     3
1284    2
1739    1
261     1
535     2
Name: price_range, Length: 400, dtype: int64
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix
# Train a logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Predict prices on the test set
y_pred = model.predict(X_test)

# Evaluate the model accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred,
average="weighted") recall = recall_score(y_test, y_pred,
average="weighted") conf_matrix = confusion_matrix(y_test,
y_pred) print(f"Accuracy: {accuracy:.2f}") print(f"Precision:
{precision:.2f}") print(f"Recall: {recall:.2f}")
print("Confusion Matrix:") print(conf_matrix)
```

```
Accuracy: 0.63
Precision: 0.64
Recall: 0.63
Confusion Matrix:
[[79 25  1  0]
 [17 46 20  8]
 [ 0 17 46 29]
 [ 0  1 29 82]]
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning:
lbfg 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
n_iter_i = _check_optimize_result(
```