AIML ASSIGNMENT

NAME:N.KARTHIKEYA
HTNO.:2203A51813
BATCH-08

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler

# a) Read the data with pandas and find features and target variables
data = pd.read_csv("/train.csv")

# Print column names to verify
print("Column names:", data.columns)

# Ensure target_column_name is set to the correct column name
target_column_name = 'battery_power'  # Replace with the actual name of
your target column

# Check if target_column_name exists in the DataFrame
if target_column_name not in data.columns:
    raise KeyError(f"'{target_column_name}' not found in the dataset.
Please check the target column name.")

features = data.drop(columns=[target_column_name])
target = data[target_column_name]

# b) Normalize the data with min-max scaling
scaler = MinMaxScaler()
scaled_features = scaler.fit_transform(features)
scaled_data = pd.DataFrame(scaled_features, columns=features.columns)

# c) Split the data into train and test
X_train, X_test, y_train, y_test = train_test_split(scaled_data,
target, test_size=0.2, random_state=42)

# Optionally, you can print the shapes of train and test data to verify
print("Shape of X_train:", X_train.shape)
print("Shape of X_test:", X_test.shape)
print("Shape of y_train:", y_train.shape)
print("Shape of y_test:", y_test.shape)
```

output:

```
Column names: Index(['battery_power', 'blue', 'clock_speed',
'dual_sim', 'fc', 'four_g',
       'int_memory', 'm_dep', 'mobile_wt', 'n_cores', 'pc',
'px_height',
```

```
       'px_width', 'ram', 'sc_h', 'sc_w', 'talk_time', 'three_g',
       'touch_screen', 'wifi', 'price_range'],
      dtype='object')
Shape of X_train: (1600, 20)
Shape of X_test: (400, 20)
Shape of y_train: (1600,)
Shape of y_test: (400,)
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score,
recall_score, confusion_matrix
data = pd.read_csv('/train.csv')
X = data.drop(columns=['price_range'])
y = data['price_range']
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
model = LogisticRegression(max_iter=1000)
print(model.fit(X_train, y_train))
y_pred = model.predict(X_test)
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("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("Confusion Matrix:")
print(conf_matrix)
```

output:

LogisticRegression(max_iter=1000)
Accuracy: 0.73
Precision: 0.7365416673696998
Recall: 0.73
Confusion Matrix:
[[88 17 0 0]
[6 62 18 5]
[0 18 52 22]
[0 0 22 90]]
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning:
lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.