Default title text

```
# @title Default title text
import pandas as pd

# Read the data
data = pd.read_csv("train.csv")

# Display the first few rows of the data
print(data.head())

# Identify features (independent variables) and target variable (dependent variable)
features = data.drop('price_range', axis=1)  # Features are all columns except 'price_range'
target = data['price_range']  # Target variable is 'price_range'

print("Features:")
print(features.head())
print("Target:")
print(target.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]
Features:
   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  pc  px_height  px_width   ram  sc_h  sc_w  talk_time  \
0        188        2   2         20       756  2549     9     7         19
1        136        3   6        905      1988  2631    17     3          7
2        145        5   6       1263      1716  2603    11     2          9
3        131        6   9       1216      1786  2769    16     8         11
4        141        2  14       1208      1212  1411     8     2         15

   three_g  touch_screen  wifi
0        0             0     1
1        1             1     0
2        1             1     0
3        1             0     0
4        1             1     0
Target:
0    1
1    2
2    2
3    2
4    1
Name: price_range, dtype: int64
```

New Section

```
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

# Step 1: Load and preprocess the data
data = pd.read_csv("train.csv")
features = data.drop('price_range', axis=1)
target = data['price_range']
scaler = MinMaxScaler()
normalized_features = scaler.fit_transform(features)

# Step 2: Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(normalized_features, target, test_size=0.2, random_state=42)

# Step 3: Train the logistic regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Step 4: Evaluate the model
y_pred = model.predict(X_test)

# Calculate evaluation metrics
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 evaluation metrics
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("Confusion Matrix:")
print(conf_matrix)
```

```
Accuracy: 0.94
Precision: 0.9395870523979205
Recall: 0.94
Confusion Matrix:
[[105   0   0   0]
 [  1  86   4   0]
 [  0   9  77   6]
 [  0   0   4 108]]
```