```
# Lab 2: Evaluation Metrics
# 0. Imports, Data Loading, Metadata

# Install ucimlrepo (run in notebook cell if not installed)
# !pip install ucimlrepo

from ucimlrepo import fetch_ucirepo
import pandas as pd

# Fetch the Automobile dataset
automobile = fetch_ucirepo(id=10)

# DataFrames
X = automobile.data.features
y = automobile.data.targets

# Show metadata and variable info
print("Metadata:\n", automobile.metadata)
print("\nVariable Info:\n", automobile.variables)
```

```
Metadata:
 {'uci_id': 10, 'name': 'Automobile', 'repository_url': 'https://archive.ics.uci.edu/dataset/10/automobile', 'data_u

Variable Info:
                  name     role         type demographic  \
0               price  Feature   Continuous        None
1         highway-mpg  Feature   Continuous        None
2            city-mpg  Feature   Continuous        None
3            peak-rpm  Feature   Continuous        None
4          horsepower  Feature   Continuous        None
5   compression-ratio  Feature   Continuous        None
6              stroke  Feature   Continuous        None
7                bore  Feature   Continuous        None
8         fuel-system  Feature  Categorical        None
9         engine-size  Feature   Continuous        None
10   num-of-cylinders  Feature      Integer        None
11        engine-type  Feature  Categorical        None
12        curb-weight  Feature   Continuous        None
13             height  Feature   Continuous        None
14              width  Feature   Continuous        None
15             length  Feature   Continuous        None
16         wheel-base  Feature   Continuous        None
17    engine-location  Feature       Binary        None
18       drive-wheels  Feature  Categorical        None
19         body-style  Feature  Categorical        None
20        num-of-doors  Feature      Integer        None
21         aspiration  Feature       Binary        None
22          fuel-type  Feature       Binary        None
23               make  Feature  Categorical        None
24  normalized-losses  Feature   Continuous        None
25          symboling   Target      Integer        None

                                   description units missing_values
0                   continuous from 5118 to 45400  None            yes
1                        continuous from 16 to 54  None             no
2                        continuous from 13 to 49  None             no
3                    continuous from 4150 to 6600  None            yes
4                       continuous from 48 to 288  None            yes
5                         continuous from 7 to 23  None             no
6                    continuous from 2.07 to 4.17  None            yes
7                    continuous from 2.54 to 3.94  None            yes
8       1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi  None             no
9                       continuous from 61 to 326  None             no
10        eight, five, four, six, three, twelve, two  None             no
11            dohc, dohcv, l, ohc, ohcf, ohcv, rotor  None             no
12                   continuous from 1488 to 4066  None             no
13                   continuous from 47.8 to 59.8  None             no
14                   continuous from 60.3 to 72.3  None             no
15                 continuous from 141.1 to 208.1  None             no
16                     continuous from 86.6 120.9  None             no
17                                    front, rear  None             no
18                                  4wd, fwd, rwd  None             no
19     hardtop, wagon, sedan, hatchback, convertible  None             no
20                                      four, two  None            yes
21                                     std, turbo  None             no
22                                    diesel, gas  None             no
23  alfa-romero, audi, bmw, chevrolet, dodge, hond...  None             no
24                      continuous from 65 to 256  None            yes
```

```
# Lab 2: Evaluation Metrics
# 1. Data Preparation and Splitting

from sklearn.model_selection import train_test_split

# Split data (80% train, 20% test)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)
```

```
# Lab 2: Evaluation Metrics
# 2. Model Training (Random Forest Example)

from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
import numpy as np

# Identify categorical and numerical features
categorical_features = ['fuel-system', 'engine-type', 'num-of-cylinders', 'engine-location', 'drive-wheels', 'body-s
numerical_features = X_train.select_dtypes(include=np.number).columns.tolist()

# Create a column transformer for one-hot encoding categorical features and leaving numerical features as is
preprocessor = ColumnTransformer(
    transformers=[
        ('num', 'passthrough', numerical_features),
        ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_features)])

# Create a pipeline that first preprocesses the data and then trains the RandomForestClassifier
clf = Pipeline(steps=[('preprocessor', preprocessor),
                      ('classifier', RandomForestClassifier())])

# Train Random Forest classifier
clf.fit(X_train, y_train.values.ravel())
y_pred = clf.predict(X_test)
```

```
# Lab 2: Evaluation Metrics
# 3. Confusion Matrix

from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", cm)
```

```
Confusion Matrix:
 [[ 1  0  0  0  0  0]
 [ 0  5  0  0  0  0]
 [ 0  2 17  3  1  0]
 [ 0  0  1 10  0  0]
 [ 0  0  0  3  3  0]
 [ 0  0  0  0  0  6]]
```

```
# Lab 2: Evaluation Metrics
# 4. Accuracy Score

from sklearn.metrics import accuracy_score

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.8076923076923077
```

```
# Lab 2: Evaluation Metrics
# 5. Precision Score

from sklearn.metrics import precision_score

precision = precision_score(y_test, y_pred, average='weighted')
print("Precision:", precision)
```

```
Precision: 0.8397817460317459
```

```
# Lab 2: Evaluation Metrics
# 6. Recall Score

from sklearn.metrics import recall_score

recall = recall_score(y_test, y_pred, average='weighted')
print("Recall:", recall)
```

```
Recall: 0.8076923076923077
```

```
# Lab 2: Evaluation Metrics
# 7. F1 Score

from sklearn.metrics import f1_score

f1 = f1_score(y_test, y_pred, average='weighted')
print("F1-Score:", f1)
```

```
F1-Score: 0.807461260510041
```

```
# Lab 2: Evaluation Metrics
# 8. Classification Report

from sklearn.metrics import classification_report

print("Classification Report:\n", classification_report(y_test, y_pred))
```

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

          -2       1.00      1.00      1.00         1
          -1       0.71      1.00      0.83         5
           0       0.94      0.74      0.83        23
           1       0.62      0.91      0.74        11
           2       0.75      0.50      0.60         6
           3       1.00      1.00      1.00         6

    accuracy                           0.81        52
   macro avg       0.84      0.86      0.83        52
weighted avg       0.84      0.81      0.81        52
```

Start coding or generate with AI.