```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import classification_report

# Read the Auto data
csv_pandas = pd.read_csv('auto.csv')

# Output the first few rows
print(csv_pandas.head(3))
print()

# Output the dimensions of the data
print(csv_pandas.shape)
print()

# Data exploration with code
print(csv_pandas[['mpg', 'weight', 'year']].describe())
print()

# Explore data types
print(csv_pandas.dtypes)
print()

csv_pandas['cylinders'] = csv_pandas['cylinders'].astype('category')
csv_pandas['origin'] = pd.Categorical(csv_pandas['origin'])

print(csv_pandas.dtypes)
print()

# Deal with NAs
csv_pandas.dropna(inplace=True)
print(csv_pandas.shape)
print()

# Modify columns
csv_pandas['mpg_high'] = np.where(csv_pandas['mpg'] > csv_pandas['mpg'].mean(), 1, 0)
csv_pandas.drop(columns=['mpg', 'name'], inplace=True)
print(csv_pandas.head(3))
print()

# Data exploration with graphs
sns.catplot(x='mpg_high', kind='count', data=csv_pandas)
sns.relplot(x='horsepower', y='weight', style='mpg_high', data=csv_pandas)
sns.boxplot(x='mpg_high', y='weight', data=csv_pandas)

# Train/test split
X = csv_pandas.drop(columns=['mpg_high'])
y = csv_pandas['mpg_high']
training_for_x, testing_for_x, training_for_y, testing_for_y = train_test_split(X, y, test_
```

```
print(training_for_x.shape, testing_for_x.shape)

# Logistic Regression
log_reg = LogisticRegression(solver='lbfgs', max_iter=1000)
log_reg.fit(training_for_x, training_for_y)

predicted_y = log_reg.predict(testing_for_x)

print(classification_report(testing_for_y, predicted_y))

# Decision Tree
dtc = DecisionTreeClassifier()
dtc.fit(training_for_x, training_for_y)

predicted_y = dtc.predict(testing_for_x)

print(classification_report(testing_for_y, predicted_y))

# Neural Network
neuralNetwork = MLPClassifier(hidden_layer_sizes=(16, 8), max_iter=1000)
neuralNetwork.fit(training_for_x, training_for_y)

predicted_y = neuralNetwork.predict(testing_for_x)

print(classification_report(testing_for_y, predicted_y, zero_division=0))

neuralNetwork2 = MLPClassifier(hidden_layer_sizes=(8, 4), max_iter=100)
neuralNetwork2.fit(training_for_x, training_for_y)

predicted_y = neuralNetwork2.predict(testing_for_x)

print(classification_report(testing_for_y, predicted_y))

# Analysis

# a. which algorithm performed better?
#   Algorithm 1
# b. compare accuracy, recall and precision metrics by class
#   Algorithm 1 had better metrics
# c. give your analysis of why the better-performing algorithm might have outperformed the
#   Due to more iterations, algorithm 1 outperformed
# d. write a couple of sentences comparing your experiences using R versus sklearn. Feel fr
#   Sklearn is easier than R but R is more readable
```

```
    mpg  cylinders  displacement  horsepower  weight  acceleration  year  \
0  18.0          8         307.0         130    3504          12.0  70.0
1  15.0          8         350.0         165    3693          11.5  70.0
2  18.0          8         318.0         150    3436          11.0  70.0

   origin                       name
0       1  chevrolet chevelle malibu
1       1          buick skylark 320
2       1         plymouth satellite

(392, 9)

              mpg       weight        year
count  392.000000   392.000000  390.000000
mean    23.445918  2977.584184   76.010256
std      7.805007   849.402560    3.668093
min      9.000000  1613.000000   70.000000
25%     17.000000  2225.250000   73.000000
50%     22.750000  2803.500000   76.000000
75%     29.000000  3614.750000   79.000000
max     46.600000  5140.000000   82.000000

mpg             float64
cylinders         int64
displacement    float64
horsepower        int64
weight            int64
acceleration    float64
year            float64
origin            int64
name             object
dtype: object

mpg              float64
cylinders       category
displacement     float64
horsepower         int64
weight             int64
acceleration     float64
year             float64
origin          category
name              object
dtype: object

(389, 9)

  cylinders  displacement  horsepower  weight  acceleration  year origin  \
0         8         307.0         130    3504          12.0  70.0      1
1         8         350.0         165    3693          11.5  70.0      1
2         8         318.0         150    3436          11.0  70.0      1

   mpg_high
0         0
1         0
2         0

(311, 7) (78, 7)
              precision    recall  f1-score   support

           0       0.98      0.82      0.89        50
```

```
           1       0.75      0.96      0.84        28

    accuracy                           0.87        78
   macro avg       0.86      0.89      0.87        78
weighted avg       0.89      0.87      0.87        78

              precision    recall  f1-score   support

           0       0.94      0.92      0.93        50
           1       0.86      0.89      0.88        28

    accuracy                           0.91        78
   macro avg       0.90      0.91      0.90        78
weighted avg       0.91      0.91      0.91        78

              precision    recall  f1-score   support

           0       0.91      0.84      0.87        50
           1       0.75      0.86      0.80        28

    accuracy                           0.85        78
   macro avg       0.83      0.85      0.84        78
weighted avg       0.85      0.85      0.85        78

              precision    recall  f1-score   support

           0       0.00      0.00      0.00        50
           1       0.36      1.00      0.53        28

    accuracy                           0.36        78
   macro avg       0.18      0.50      0.26        78
weighted avg       0.13      0.36      0.19        78

/usr/local/lib/python3.9/dist-packages/sklearn/metrics/_classification.py:1344: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.9/dist-packages/sklearn/metrics/_classification.py:1344: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.9/dist-packages/sklearn/metrics/_classification.py:1344: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
```