In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
```

# Problem a

In [2]:

```
df = pd.read_csv('Auto.csv')
df_copy = df.copy()

# Eliminates the rows (instances) with '?' as a predictor value
df_copy['horsepower'] = pd.to_numeric(df_copy['horsepower'], errors='coerce')
df_copy = df_copy.dropna()
# df_copy['name'] = df_copy['name'].str.split(' ').str[0]

# Drop the name variable
df_copy = df_copy.drop('name', 1)

# Create 'mpg01' column, 1 => mpg > median, 0 => mpg < median
mpg_median = df_copy['mpg'].median()
df_copy.loc[df_copy['mpg'] >= mpg_median, 'mpg01'] = 1
df_copy.loc[df_copy['mpg'] < mpg_median, 'mpg01'] = 0
df_copy.head()
```

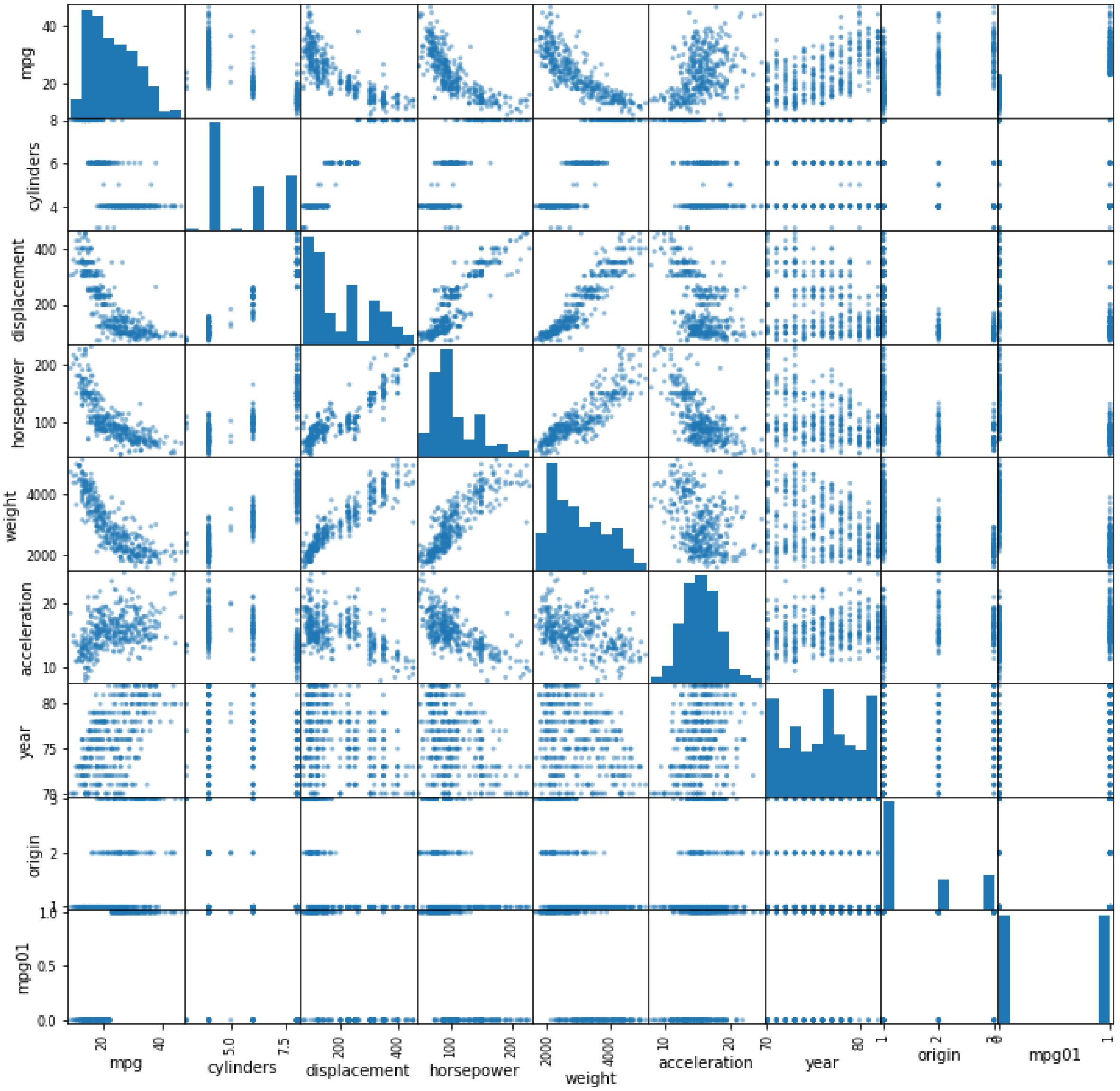
Out[2]:

| | mpg | cylinders | displacement | horsepower | weight | acceleration | year | origin | mpg01 |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 18.0 | 8 | 307.0 | 130.0 | 3504 | 12.0 | 70 | 1 | 0.0 |
| **1** | 15.0 | 8 | 350.0 | 165.0 | 3693 | 11.5 | 70 | 1 | 0.0 |
| **2** | 18.0 | 8 | 318.0 | 150.0 | 3436 | 11.0 | 70 | 1 | 0.0 |
| **3** | 16.0 | 8 | 304.0 | 150.0 | 3433 | 12.0 | 70 | 1 | 0.0 |
| **4** | 17.0 | 8 | 302.0 | 140.0 | 3449 | 10.5 | 70 | 1 | 0.0 |

# Problem b

In [3]:

```
# Creates Scatterplot Matrix with the quantitative variables
# columns = ['cylinders', 'displacement', 'horsepower', 'weight', 'acceleration', 'year', 'origin', 'mpg01']
pd.plotting.scatter_matrix(df_copy, figsize=(12,12)); # Semicolon used to suppress array output!
```



# Problem c

In [4]:

```
y_true = df_copy['mpg01']
X_var = df_copy[['displacement', 'horsepower', 'weight', 'year', 'origin']]

X_train, X_test, y_train, y_test = train_test_split(X_var, y_true, test_size=0.10, random_state=42)
```

# Problem d

In [5]:

```
y_true = df_copy['mpg01']
X_var = df_copy[['displacement', 'horsepower', 'weight', 'year', 'origin']]

# # Make variable transformations
# data = {'log-displacement': np.log(X_var['displacement']),
#          'log-horsepower': np.log(X_var['horsepower']),
#          'log-weight': np.log(X_var['weight']),
#          'year': X_var['year'],
#          'origin': X_var['origin'],
#        }
# transformed_data = pd.DataFrame(data)


X_train, X_test, y_train, y_test = train_test_split(X_var, y_true, test_size=
0.10, random_state=42)





# ----------------------- FITTING THE MODEL ---------------------------------
# Create the Linear Discriminant Analysis Classifier Model and fit it
LDA_classifier = LinearDiscriminantAnalysis()
LDA_classifier.fit(X_train, y_train)


# ----------------------- MAKING PREDICTIONS ---------------------------------
y_pred_test = LDA_classifier.predict(X_test)
num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_test, y_pred_test).ravel()
print("Number of True Positives: " + repr(num_TP))
print("Number of False Negatives: " + repr(num_FN))
print("Number of False Positives: " + repr(num_FP))
print("Number of True Negatives: " + repr(num_TN))
```

```
Number of True Positives: 20
Number of False Negatives: 0
Number of False Positives: 4
Number of True Negatives: 16
```

# Problem e

In [6]:

```
y_true = df_copy['mpg01']
X_var = df_copy[['displacement', 'horsepower', 'weight', 'year', 'origin']]

# # Make variable transformations
# data = {'log-displacement': np.log(X_var['displacement']),
#         'log-horsepower': np.log(X_var['horsepower']),
#         'log-weight': np.log(X_var['weight']),
#         'year': X_var['year'],
#         'origin': X_var['origin'],
#        }
# transformed_data = pd.DataFrame(data)


X_train, X_test, y_train, y_test = train_test_split(X_var, y_true, test_size=
0.10, random_state=42)





# ----------------------- FITTING THE MODEL ---------------------------------
# Create the Quadratic Discriminant Analysis Classifier Model and fit it
QDA_classifier = QuadraticDiscriminantAnalysis()
QDA_classifier.fit(X_train, y_train)


# ----------------------- MAKING PREDICTIONS ---------------------------------
y_pred_test = QDA_classifier.predict(X_test)
num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_test, y_pred_test).ravel()
print("Number of True Positives: " + repr(num_TP))
print("Number of False Negatives: " + repr(num_FN))
print("Number of False Positives: " + repr(num_FP))
print("Number of True Negatives: " + repr(num_TN))
```

```
Number of True Positives: 20
Number of False Negatives: 0
Number of False Positives: 3
Number of True Negatives: 17
```

# Problem f

In [7]:

```
y_true = df_copy['mpg01']
X_var = df_copy[['displacement', 'horsepower', 'weight', 'year', 'origin']]

# # Make variable transformations
# data = {'log-displacement': np.log(X_var['displacement']),
#          'log-horsepower': np.log(X_var['horsepower']),
#          'log-weight': np.log(X_var['weight']),
#          'year': X_var['year'],
#          'origin': X_var['origin'],
#         }
# transformed_data = pd.DataFrame(data)


X_train, X_test, y_train, y_test = train_test_split(X_var, y_true, test_size=
0.10, random_state=42)





# ----------------------- FITTING THE MODEL ---------------------------------
# Create the Logistic Classifier Model and fit it
Logit_classifier = LogisticRegression(random_state=0)
Logit_classifier.fit(X_train, y_train)


# ----------------------- MAKING PREDICTIONS ---------------------------------
y_pred_test = Logit_classifier.predict(X_test)
num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_test, y_pred_test).ravel()
print("Number of True Positives: " + repr(num_TP))
print("Number of False Negatives: " + repr(num_FN))
print("Number of False Positives: " + repr(num_FP))
print("Number of True Negatives: " + repr(num_TN))
```

```
Number of True Positives: 20
Number of False Negatives: 0
Number of False Positives: 1
Number of True Negatives: 19
```

# Problem g

In [83]:

```
y_true = df_copy['mpg01']
X_var = df_copy[['displacement', 'horsepower', 'weight', 'year', 'origin']]

# Make variable transformations
data = {'log-displacement': np.log(X_var['displacement']),
        'log-horsepower': np.log(X_var['horsepower']),
        'log-weight': np.log(X_var['weight']),
        'year': X_var['year'],
        'origin': X_var['origin'],
       }
transformed_data = pd.DataFrame(data)


X_train, X_test, y_train, y_test = train_test_split(X_var, y_true, test_size=
0.10, random_state=39)





# ----------------------- FITTING THE MODEL ---------------------------------
# Create the KNN Classifier Model and fit it
KNN_classifier = KNeighborsClassifier(n_neighbors=10)
KNN_classifier.fit(X_train, y_train)


# ----------------------- MAKING PREDICTIONS ---------------------------------
y_pred_test = KNN_classifier.predict(X_test)
num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_test, y_pred_test).ravel()
print("Number of True Positives: " + repr(num_TP))
print("Number of False Negatives: " + repr(num_FN))
print("Number of False Positives: " + repr(num_FP))
print("Number of True Negatives: " + repr(num_TN))
```

```
Number of True Positives: 18
Number of False Negatives: 2
Number of False Positives: 3
Number of True Negatives: 17
```

In [ ]:

In [ ]:

In [ ]: