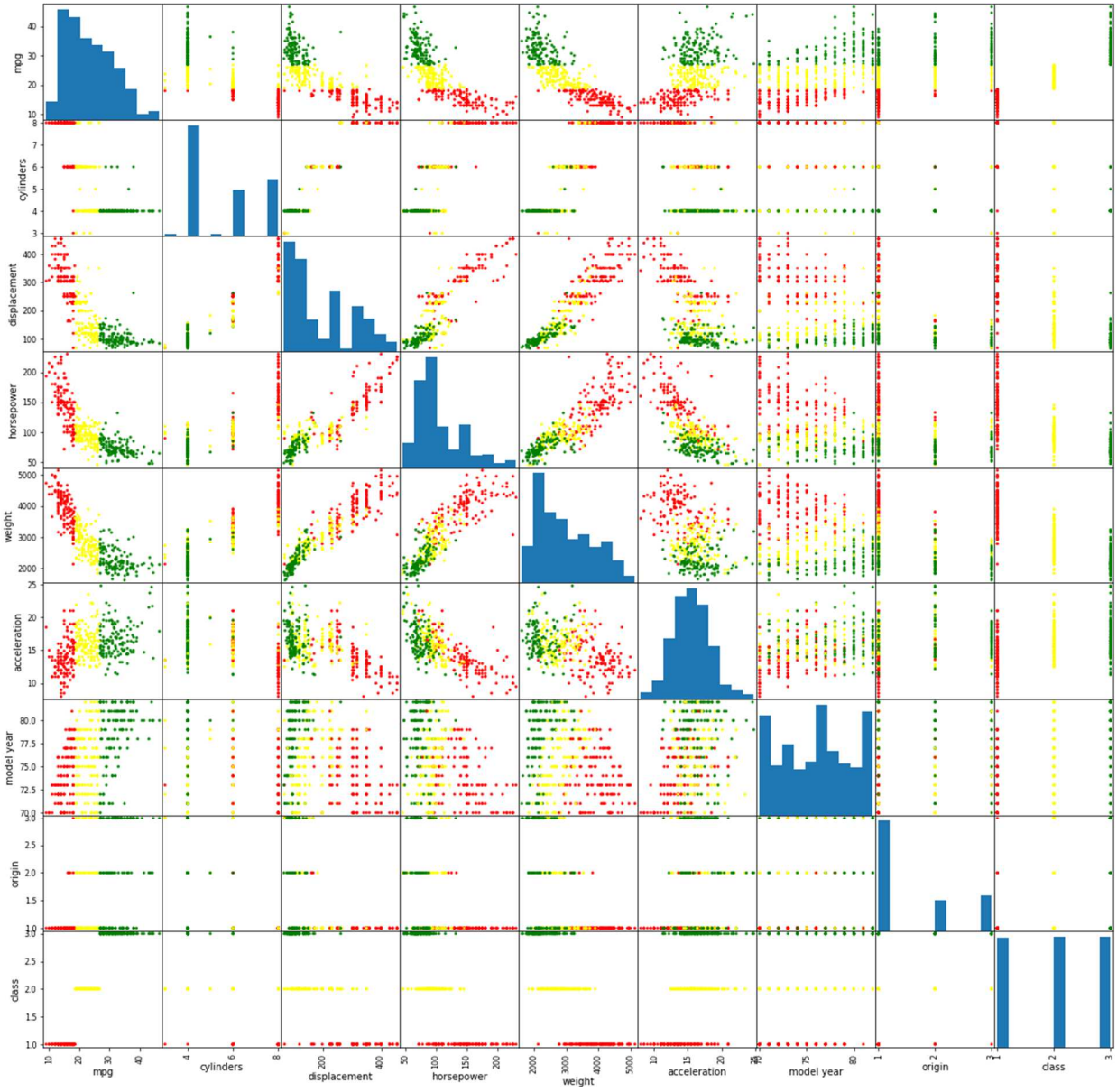
1. Assume that we want to classify the cars into 3 categories: low, medium and high mpg. Find what the threshold for each category should be, so that all samples are divided into three equally-sized bins.

   The correct thresholds for the classification of the auto-mpg data set can be found by first dividing the set into 3 subsets, finding the average of the minimum of the second subset and the maximum of the first subset, and then finding the average of the maximum of the second subset and the minimum of the third. These two averages will be the thresholds used to classify future mpg inputs.

```
In [22]: max1 = max(mpg_dfs[0]['mpg'])

In [23]: max2 = max(mpg_dfs[1]['mpg'])

In [24]: min1 = min(mpg_dfs[1]['mpg'])

In [25]: min2 = min(mpg_dfs[2]['mpg'])

In [26]: thresh1 = (max1 + min1) / 2

In [27]: thresh2 = (max2 + min2) / 2

In [28]: thresh1
Out[28]: 18.55

In [29]: thresh2
Out[29]: 26.9
```

2. Create a 2D scatterplot matrix, similar to that of Figure 1.4 in the ML book (K. Murphy, page 6; also available on the lecture 1 slides - the figure with the flowers). You may use any published code to perform this. Which pair from all pair-wise feature combinations is the most informative regarding the three mpg categories?



Green = High mpg

Yellow = Medium mpg

Red = Low mpg

Weight seems to be the best predictor of mpg with pair (mpg, weight) having a correlation of -0.8322.

(displacement, weight) has the highest correlation of any of the pairs (0.9329) and also appears to be an excellent predictor of mpg. Samples with the highest weight and displacement also are in the low mpg class (red), and samples with the lowest weight and displacement are in the high mpg class (green).

Report Continued Below

3. Write a linear regression solver that can accommodate polynomial basis functions on a single variable for prediction of MPG. Your code should use the Ordinary Least Squares (OLS) estimator (i.e. the Maximum-likelihood estimator). Code this from scratch. Its recommended to use a library (e.g. numpy) for basic linear algebra operations (addition, multiplication and inverse).

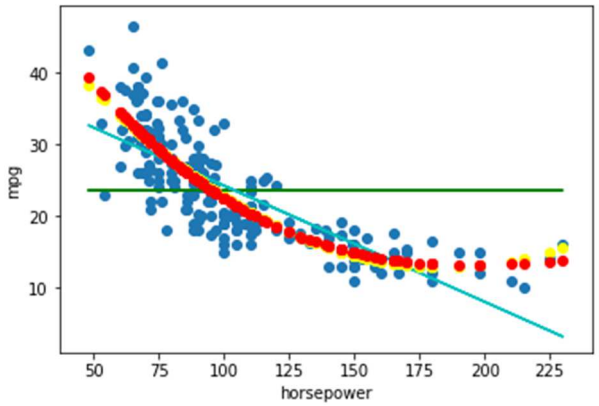
```
def LinearReg(order, x, y):

    ones = [1] * 392
    ones = np.array([ones]) #creates array of ones to be included in X matrix
    ones = ones.astype(float)

    #x = x.astype(float)       #converts arrays to float types so we can perform matrix operations
          #
    y = y.transpose()
    #y = y.astype(float)


    if order == 0:
        X = ones
        X = X.transpose()
        XT = X.transpose()
        w = OLS(X, XT, y)

    elif order == 1:
        X = np.append(ones, x, axis = 0)  #combine ones and x arrays to prpoduce X matrix
        X = X.transpose()  #transpose X to get N x 2 matrix
        XT = X.transpose()
        w = OLS(X, XT, y)

    elif order == 2:
        xsqr = np.square(x)
        X = np.append(ones, x, axis = 0)
        X = np.append(X, xsqr, axis = 0)
        X = X.transpose()
        XT = X.transpose()
        w = OLS(X, XT, y)

    elif order == 3:
        xcube = np.power(x, 3)
        xsqr = np.square(x)
        X = np.append(ones, x, axis = 0)
        X = np.append(X, xsqr, axis = 0)
        X = np.append(X, xcube, axis = 0)
        X = X.transpose()
        XT = X.transpose()
        w = OLS(X, XT, y)

    else:
        return print('error: order must be between 0 and 3')


    #calculate y-hat, the predicted equation for the line of best fit

    yhat = X @ w


    return yhat
```

```
def OLS(x, xt, y):
    w = xt @ x
    w = inv(w)
    w = w @ xt
    w = w @ y


    return w
```

4. Split the dataset in the first 200 samples for training and the rest 192 samples for testing. Use your solver to regress for 0th to 3rd order polynomial on a single independent variable (feature) each time by using mpg as the dependent variable. Report (a) the training and (b) the testing mean squared errors for each variable individually (except the "car name" string variable, so a total of 7 features that are independent variables). Plot the lines and data for the testing set, one plot per variable (so 4 lines in each plot, 7 plots total). Which polynomial order performs the best in the test set? Which is the most informative feature for mpg consumption in that case?

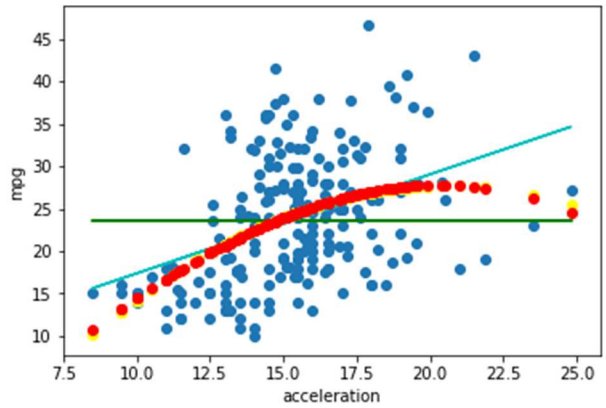
Plots and Training and Testing Errors of Univariate Linear Regression on each of the 7 features:

Green = 0^{th} order
Blue = 1^{st} order
Yellow = 2^{nd} order
Red = 3^{rd} order



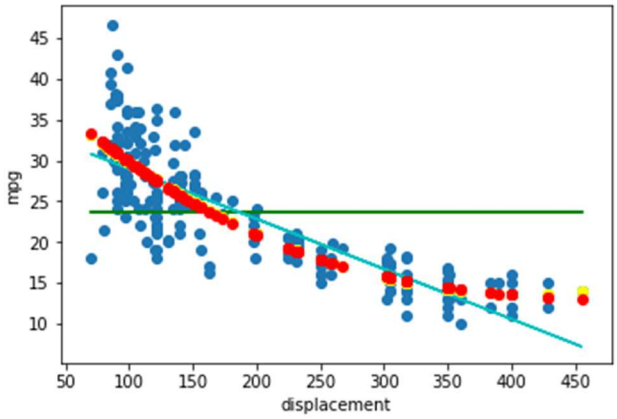
(horsepower, mpg)

	0^{th} Order	1^{st} Order	2^{nd} Order	3^{rd} Order
Training Error	57.2328	22.1191	17.1152	17.0227
Testing Error	63.7424	25.6961	21.0490	21.2175



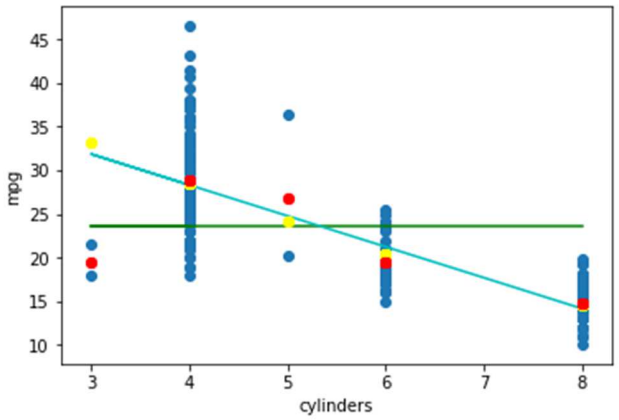
(acceleration, mpg)

	0th Order	1st Order	2nd Order	3rd Order
Training Error	57.2328	48.4851	46.7398	46.7239
Testing Error	63.7424	51.8287	51.8536	51.9746



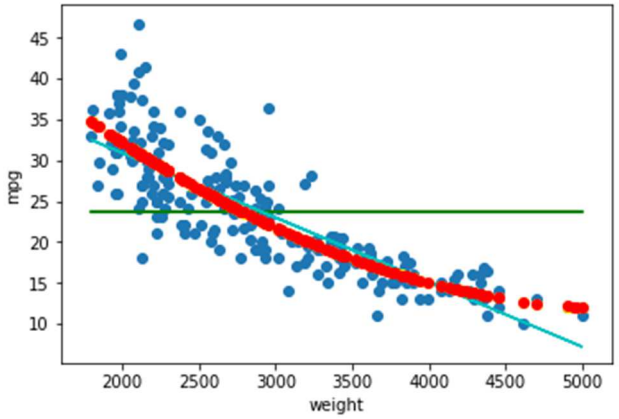
(displacement, mpg)

	0th Order	1st Order	2nd Order	3rd Order
Training Error	57.2328	19.9956	18.0231	18.0007
Testing Error	63.7424	22.4106	19.2573	19.1888



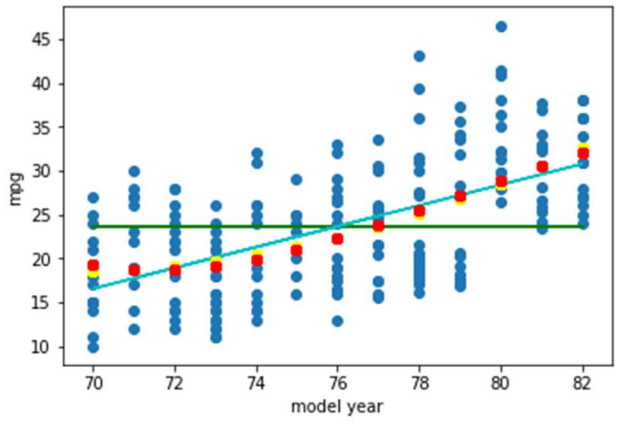
(cylinders, mpg)

	0th Order	1st Order	2nd Order	3rd Order
Training Error	57.2328	22.7113	22.5365	20.2738
Testing Error	63.7424	24.7657	24.5882	22.7283



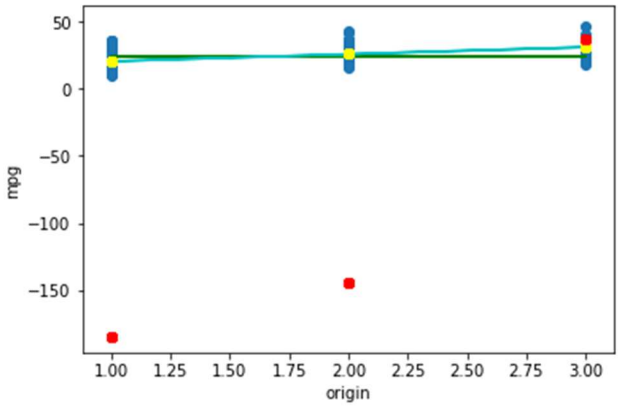
(weight, mpg)

	0th Order	1st Order	2nd Order	3rd Order
Training Error	57.2328	17.9719	16.5382	16.5381
Testing Error	63.7424	19.0143	17.5392	17.5382



(model year, mpg)

	0th Order	1st Order	2nd Order	3rd Order
Training Error	57.2328	37.6834	36.4132	36.1987
Testing Error	63.7424	43.4605	41.0016	41.2111



(origin, mpg)

	0th Order	1st Order	2nd Order	3rd Order
Training Error	57.2328	39.8072	39.6856	31371.89
Testing Error	63.7424	42.4975	41.6324	31234.68

5. Modify your solver to be able to handle second order polynomials of all 7 independent variables simultaneously (i.e. 15 terms). Regress with 0th, 1st and 2nd order and report (a) the training and (b) the testing mean squared error (MSE). Use the same 200/192 split.

Training and Testing Error of Multivariate Linear Regression

(horsepower, acceleration, displacement, cylinders, weight, model year, origin, mpg)

	0^{th} Order	1^{st} Order	2^{nd} Order
Training Error	57.2728	10.1801	7.0162
Testing Error	63.7424	12.0086	8.7600

6. Using logistic regression (1st order) for low/medium/high classification. Report the training/ testing classification precision (you might want to look how precision is defined and how it is calculated). You can use a library (e.g. scikit-learn) to perform logistic regression.

Precision Scores of Logistic Regression

	Horsepower	Acceleration	Displacement	Weight	Cylinders	Model Year	Origin
Training Precision Score	0.68305	0.48098	0.68339	0.70882	0.65553	0.45698	0.53696
Test Precision Score	0.75069	0.46819	0.71494	0.68861	0.65105	0.44312	0.56130

7. If a USA manufacturer (origin 1) had considered to introduce amodel in 1980 with the following characteristics: 6 cylinders, 350 cc displacement, 180 horsepower, 3700 lb weight, 9 *m*/*sec*2 acceleration, what is the MPG rating that we should have expected? In which mpg category (low,medium,high mpg) would it belong? Use second-order, multi-variate polynomial and logistic regression.

New Car Predictions

We should expect the new car to have an MPG rating of about 19.08 mpg based on the results of 2^{nd} order Multivariate Linear Regression.

It should fall under the low mpg class according to my Logistic Regression analysis.