**CSCE 623: Machine Learning**

**Spring 2020**

**HW1**

Your homework will be composed of an integrated written portion and Python programming component. You will produce a single jupyter notebook file (\*.ipynb). You will be using the Auto.csv dataset provided. In your answers to written questions, even if the question asks for a single number or other form of short answer (such as yes/no or which is better: A or B) you must provide supporting information for your answer to obtain full credit. Use Python to perform calculations or mathematical transformations, or provide python-generated graphs and figures or other evidence that explain how you determined the answer. Use both code cells and markup cells in your jupyter notebook. A shell is provided to get you started.

**Simple Linear Regression**

1. Load the “Auto.csv” dataset (note that missing values (e.g. “?”) must be handled – one suggestion is to remove unneeded data observations). Store the data in a pandas dataframe called “data”
2. Explore the dataset. Useful pandas functions include .info and .hist as well as scatter\_matrix in pandas.tools.plotting
   1. Display statistics of the dataset. How many numerical features/attributes are there? How many observations/datapoints?
   2. Display a histogram of each of the individual feature values. Describe these distributions in terms of descriptions from statistics (e.g. uniform, Gaussian, exponential, skewed, multi-modal)
   3. Choose a subset of at least 5 attributes you expect to have relationships and display a scatterplot of each of the pairings between each possible pair of these attributes. What pairs do you see with linear relationships? Non-linear? Which pairs have strong relationships and which appear to have weak relationships? Describe the phenomenon that you see in your plots.
3. Make a scatterplot (Horsepower vs mpg), Set the axes so that the origin (0,0) is included, as well as all of the datapoints. Label axes appropriately: “Horsepower”, “MPG”). On this Horsepower vs. MPG plot, assume that *β*0 is fixed at 40. Estimate what the slope *β*1 of the best fit line is for the dataset (eyeball an educated guess) given that *β*0 is fixed at 40. Report your eyeball estimate for *β*1 using a markdown cell in jupyter.
4. Using code, make a vector of possible *β*1 values that surround what you think the slope of the best fit line is (hint: use the linspace function in numpy). Display the vector of these numerical *β*1 values.
5. Make a python function “rss1d(beta0,beta1,x,y)” for computing cost: this function should compute residual sum of squared errors (RSS) for the dataset for a given *β*0 and *β*1. Then use this function to compute RSS for the fixed *β*0 under each version of *β*1 coefficients from step 4 and store these costs for each value of *β*1. You may find a loop might handy here.
6. Using your results from step 5, make a new plot of *β*1 value vs RSS cost. Your axes should be labeled as *β*1 on the x-axis and RSS on the y-axis). If possible, see if you can make the subscripted beta appear as math-style text in the x-axis label.
7. Answer these questions in your report: Describe the shape of the plot in step 6? Explain how using the plot, someone could find the best value of *β*1. Select the value of *β*1 you think will have the best fit (you may want to improve your estimate by exploring near it by adding additional values for *β*1 and repeat steps 3-6).
8. Determine the linear regression line formed when *β*0 is 40 and the value of *β*1 you computed in step 7. Make a new plot which displays a **red** linear regression line overlayed on a Horsepower vs. MPG scatterplot of the original dataset points
9. Review eqn 3.4 on page 62. In code, develop the closed-form *function* computeBetas(xVec, yVec) which accepts a vector of *x* values and a vector of *y* values and returns betas, which is a structure containing the values for the 2 coefficients *β*0 and *β*1
10. Compute *β*0 and *β*1 for the Auto dataset using the closed-form function you created in step 9.
11. How does the closed-form computed value of *β*1 compare with your estimate of *β*1 from step 6? Discuss in your report.
12. Make a new plot which displays a **green** linear regression line formed by the closed-form expression (from step 9 & 10) overlayed on a Horsepower vs. MPG scatterplot of the original dataset points.
13. Now use sklearn’s linear\_model function to fit a linear model from horsepower to mpg. What are the model’s coefficients, MSE & explained variance score?
14. Make a new plot which displays a **black** linear regression line formed by the sklearn linear model (from step 12) overlayed on a Horsepower vs. MPG scatterplot of the original dataset points.
15. Explore the residual errors from using the linear model to make predictions:
    1. Compute the residual errors in using the model to predict mpg from horsepower. Plot these residual errors as a function of horsepower using a scatterplot. Add a **red** horizontal line at y=0 to indicate the zero-error position.
    2. Describe the plot - particularly the trends. Do the errors appear well-distributed, or are there trends? If there are trends: describe the trends, explain what these trends indicate about the ability to predict mpg from horsepower using a linear model, and give at least one course of action you could take to make a better model.

Optional (not required … but good practice in developing your coding skills): build a structure containing possible values for *β*1 and *β*0 pairs. Compute the RSS over all beta pairs at each cell in the matrix on the horsepower vs. MPG data. Now build a contour and/or 3D plot of these RSS values as shown in the book Figure 3.2 on page 63 (the x and y axes are *β*1 and *β*0 and the z axis is RSS). Write code to determine the beta pair with the minimum RSS. Report the minimum value cost. On your contour/3D plot, add a point at the location of the *β*0, *β*1 coordinates which minimize the RSS.

**Helpful Tips**

You might find these python packages/imports useful:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

%matplotlib inline

from sklearn import datasets, linear\_model