CMPE 442/CS 542 Assignment #1

Due Friday, April 28.

Note: This assignment worth 10% of overall course grade.

Note: The assignment is to be done individually. You are going to submit the report and code separately. Prepare separate code files for each question. Please do not include codes in the report. Your report should be self-contained, i.e. explain what you are doing, put the resulting graphs, the hyperparameters that you use, and write your observations.

Note: You will find two additional ipynb files. numpy_stuff.ipynb gives basic introduction to numpy arrays and manipulations that can be done on them. It is given as a tutorial only, you are free to try different arrays and see the changes in the result. Assignment1_template.ipynb is where you are going to fill the missing parts. Data generation and plotting parts are given in the code.

In this assignment, you are going to work with synthetic data, i.e. the data that you generate yourselves. You will not separate this data into train and test set. Instead see the results on the training set itself.

1) **[25 pts]** In this part of assignment you are free to use python models (linear regression, polynomial features, etc.). Polynomial Regression:
   I. **[15 pts]** Generate synthetic data using a function $y = \sin(2\pi x) + \varepsilon$, where m=10 (number of training examples) and ε is a random noise. Example in python:

```
m=10;
X=np.random.rand(m,1)*2
y = np.sin(2*math.pi*X)+np.random.randn(m,1)
```

   Apply d-order polynomial model to the data set where d=0,1,3,9 (you can use PolynomialFeatures and LinearRegression class of Scikit). Plot on the same figure the data and the curves resulting from your fit for various d (0,1,3,9) values. Comment on the results for various d. How different values of d affect the curve/line that fit to the data? Which value for d you think is appropriate for this data?

   II. **[10 pts]** Increase the number of training examples to m=100. Repeat the steps done in (I). Comment briefly on what changed with the increase of m. What problems you encountered with large d in previous case? Is it resolved with m=100? Can you derive any conclusions from the results with increased number of samples?

2) **[25 pts]** Generate synthetic data using a function $y = 100 + 3x + \varepsilon$, where m=100 (number of training examples) and ε is a random noise. Example on python:

```
m=100;
X=np.random.rand(m,1)
y=100+3*X+np.random.randn(m,1);
```

I. **[15 pts]** Implement linear regression ($y = \boldsymbol{\theta}^T \boldsymbol{x}$) on this dataset using **batch gradient descent**. You are expected to write a separate function for linear regression that is expected to return thetas (parameters):
theta= linear_regression(X, y, iterNo, eta);
where, *X* is a matrix of features, *y* is target variables vector, *iterNo* is number of iterations and *eta* is learning rate. You can set eta=0.1, and iterNo=1000;

Plot on the same figure the data and the straight line resulting from your fit. (Remember to include the intercept term). **Give the function that was fit to your data (in other words give your hypothesis).**

II. **[10 pts]** Include to your linear_regression function statements that compute MSE for each iteration. Then update the return parameters so that the function also returns MSEs for every iteration. Plot the results for eta=0.1. Then try linear regression for eta=0.001, 0.01 and 0.5, while keeping iterNo fixed to 1000. Plot the results for every eta and also report on the hypothesis function that it learns for every eta. Discuss briefly on the results.

3) **[50 pts]** Locally Weighted Linear Regression
In this part of the assignment you are going to implement locally weighted linear regression. The difference of locally weighted linear regression from normal linear regression is that in the latter case all the weights (the $w^{(i)}$'s) were considered to be the same (i.e. all 1's). You will minimize the following function.

$$MSE(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} w^{(i)}\left(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)} - y^{(i)}\right)^2$$

Generate synthetic data using a function $y = \sin(2\pi x) + \varepsilon$, where m=100 (number of training examples) and ε is a random noise. Example in python:

```
m=100;
X=np.random.rand(m,1)*2
y = np.sin(2*math.pi*X)+np.random.randn(m,1)
```

I. **[30 pts]** Update the linear regression function that you implemented in (2) so that local weights are also considered in taking the gradient of a cost function. The function should have the following header:
theta= weighted_linear_regression(X, Y, iteration_cnt, eta, x, tau)

where x is a query point, tau is bandwidth parameter.
Use *eta=0.4* and *iterNo=100.*
When evaluating h(·) at a query point *x*, use weights

$$w^{(i)} = \exp\left(-\frac{(x^{(i)} - x)^2}{2\tau^2}\right)$$

With a bandwidth parameter $\tau = 0.1$. (Again, remember to include intercept term). Plot on the same figure the data and the curve resulting from your fit.
Note: The query samples are the samples coming from training set itself. Since you have 100 samples in your training set you will call weighted_linear_regression 100 times, each time selecting one of the samples from your training set.

II. **[20 pts]** Repeat (I) five times with $\tau = 0.001, 0.01, 0.3, 1 \; and \; 10$. Comment briefly on what happens to the fit when τ is too small or too large. Given the fixed parameters for learning rate and number of iterations as above, what is the best tau to fit your synthetic data?