INTRODUCTION TO MACHINE LEARNING
Assignment 1
Linear Regression

Deadline: September 12th, 11:59 p.m. IST **Max points: 100**

Note: Only the first 100 points attempted by you will be evaluated.

1. **(Least Squares, one feature)** Suppose we want to predict the number of cold drink bottles sold in the Ashoka campus at Sonipat on a particular day (y), given the maximum temperature on that day (x). For this purpose, we collect data on maximum temperature and cold drink sales for n days. Let us label the temperatures as $x^{(1)}$, ..., $x^{(n)}$ and the cold drink sales as $y^{(1)}$, ..., $y^{(n)}$. Let $\bar{x}$ and $\bar{y}$ represent the means of $x^{(1)}$, ..., $x^{(n)}$ and $y^{(1)}$, ..., $y^{(n)}$ respectively. We want to build a linear predictor of the form
$$y = \theta_0 + \theta_1 x,$$
such that the loss function
$$J(\theta_0, \theta_1) = \frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - \theta_0 - \theta_1 x^{(i)})^2$$
is minimized. If solutions to this minimization problem are represented by $\hat{\theta}_0$ and $\hat{\theta}_1$, show that
$$\hat{\theta}_1 = \frac{\sum_{i=1}(x^{(i)} - \bar{x})(y^{(i)} - \bar{y})}{\sum_{i=1}(x^{(i)} - \bar{x})^2}, \ \hat{\theta}_0 = \bar{y} - \hat{\theta}_1\bar{x}.$$
What would the values $\hat{\theta}_0$ and $\hat{\theta}_1$ be, if all $x^{(i)}$s had the same value? If we were to use cold drink sales to predict maximum daily temperature, would the same linear predictor work? Justify your answer. **(30)**

2. **(Least Squares, one feature)** Prove the following results using the same notation as question 1.

   (a) $\bar{y} = \hat{\theta}_0 + \hat{\theta}_1\bar{x}$

   (b) $\sum_{i=1}^{n} \epsilon_i = 0$, where $\epsilon_i = \left(y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i\right)$

   (c) $\sum_{i=1}^{n} (y_i - \bar{y})\epsilon_i = 0$

   (d) $\sum_{i=1}^{n} x_i\epsilon_i = 0$

(20)

3. **(Logistic regression)** We have seen that in case of Logistic Regression, we use the hypothesis $h_\theta(\mathbf{x}) = \dfrac{e^{\theta^T\mathbf{x}}}{1+e^{\theta^T\mathbf{x}}}$. Also, the logit transform of a real number is given by $\text{logit}(t) = \log\left(\dfrac{t}{1-t}\right)$.

   (a) Show that $\text{logit}(h_\theta(\mathbf{x})) = \theta^T\mathbf{x}$.

   (b) Note that the logit transform linearizes the hypothesis function. If we had a single feature x, would Logistic Regression be equivalent to performing linear regression on $\text{logit}(x)$? Why/Why not?

   (c) Given the following values of x and y, first build a linear regression model of y on $\text{logit}(x)$, and then a logistic regression model of y on x. Report the differences that you see.
   x : 1.4, 6, 8.1, 9, 2.4, 0.7, 1.8, 8.8, 9.2, 5.6
   y : 0, 1, 1, 1, 0, 0, 0, 1, 1, 1

(30)

4. **(Programming Exercise)** Consider the Real Estate Valuation datset in `https://archive.ics.uci.edu/dataset/477/real+estate+valuation+data+set`. It contains 414 training examples with the following variables:

   - x_1, transaction date;
   - x_2, house age (years);
   - x_3, distance to the nearest MRT station (metres);
   - x_4, number of convenience stores in the living circle;
   - x_5, latitude;
   - x_6, longitude;
   - y, house price per unit area.

   Suppose we want to use linear regression to predict y with x_2, x_3 and x_4 as features. We will train the regression model on the first 325 training examples (this is the *training data*), and test our model on the remaining examples (*test data*). Let $\mathbf{X}$ be the matrix whose ith row is given by $\begin{bmatrix}1 & x_2^{(i)} & x_3^{(i)} & x_4^{(i)}\end{bmatrix}$, $i = 1, 2, ..., 325$. Using matrix inversion, evaluate $\hat{\theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$. Recall that this is the Least Squares minimizer of $J(\theta) = \dfrac{1}{2}\displaystyle\sum_{i=1}^{n}(y^{(i)} - \theta^T\mathbf{x}^{(i)})^2$, where $\mathbf{x}^{(i)} = \begin{bmatrix}1 & x_2^{(i)} & x_3^{(i)} & x_4^{(i)}\end{bmatrix}^T$.

   Create an implementation of this algorithm in Python. Note that you should not use any libraries except Pandas, Numpy, and Matplotlib. Your code should:

   - Load and reshape the dataset as necessary
   - Split the data into training and test data as instructed above
   - Use the matrix inversion method to implement linear regression

Note the time taken to reach this solution.

A measure of goodness of your model is the mean squared error, which is the mean of the squared difference between the values predicted by your model and the corresponding actual values. In your program, compute the mean squared error for training and test data sets. Are the two values far apart? Why/Why not? **(30)**

5. **(Programming Exercise)** Considering the same dataset and same features as question 4, use gradient descent to converge to the parameter values minimizing the sum of squared errors.

   With the same contraints as before, implement your solution in Python. Your code should:

   - Load and reshape the dataset as necessary
   - Split the data into training and test data as in question 4.
   - Initialize $\theta = \begin{bmatrix} 30 & 0 & 0 & 0 \end{bmatrix}$
   - Iteratively update θ till convergence (maximum difference between successive updates of θ is less than 0.01 for three consecutive iterations)

   Is the limiting value different from the least squares solution obtained in question 4? Also note the time taken to converge to this limiting value, and compare it to the time taken to solve the least squares problem using matrix inversion. Try experimenting with learning rates of $0.001, 0.003, 0.01$ and 0.1 to see how it affects your results, and note down your observations. **(30)**