# Machine Data and Learning

## Assignment-2: Bias-Variance Trade-off

Moida Praneeth Jain, 2022101093

## Task 1: Gradient Descent

### (a)

Gradient descent is an optimization algorithm used to fit a model to its training data.

Assume we have a dataset of the form

$$X = [x_1, x_2, ..., x_n]$$

$$Y = [y_1, y_2, ..., y_n]$$

Where $x_i$ are the independent variables and $y_i$ are the dependent variables

To find the line of best fit ($m$ is the slope and $c$ is the signed y-intercept)

$$y = mx + c$$

We first define a cost function such as Mean-Squared Error (MSE)

$$J = \text{cost} = \text{MSE} = \frac{1}{n} \sum_i (y_i - (mx_i + c))^2$$

We randomly choose initial values for $m$ and $c$, and iteratively move towards the correct values as follows:

First, find the gradient of the cost function $J$ with respect to $m$ and $c$

$$\frac{\partial J}{\partial m} = \frac{1}{n} \sum_i (2(y_i - (mx_i + c))(-x_i))$$

$$\frac{\partial J}{\partial m} = -\frac{2}{n} \sum_i x_i(y_i - (mx_i + c))$$

$$\frac{\partial J}{\partial c} = \frac{1}{n} \sum_i (2(y_i - (mx_i + c))(-1))$$

$$\frac{\partial J}{\partial c} = -\frac{2}{n} \sum_i (y_i - (mx_i + c))$$

Let us define $L$ to be the learning rate. It determines the step size in each iteration. A small value would increase accuracy but require more iterations, while a high value would decrease accuracy but require less iterations. Now, we update $m$ and $c$ in the opposite direction of the gradient, with the step size $L$

$$m = m - L \times \frac{\partial J}{\partial m}$$

$$c = c - L \times \frac{\partial J}{\partial c}$$

We continue this process until our loss function reaches an almost constant value. After we get our final $m$ and $c$, we have our fitted model.

### (b)

For a mutivariable model with q independent and one dependent variable, we have

$$y = \begin{pmatrix} x_1 & x_2 & x_3 & \dots & x_q \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \\ \beta_q \end{pmatrix} + c$$

$$y = \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_q x_q + c$$

The cost function is

$$J = \frac{1}{n} \sum_i \left( y_i - \left( \sum_{j=1}^{q} (\beta_i x_i) + c \right) \right)^2$$

The coefficients $(\beta)$ can be found simply by extending the single variable case.

$$\beta_i = \beta_i - L \frac{\partial J}{\partial \beta_i}$$

$$c = c - L \frac{\partial J}{\partial c}$$

We iteratively update all $\beta_i$ and $c$ until the cost function converges, and then we get our fitted model.