CSE 4554
Machine Learning Lab

Experiment No: 2

Name of the experiment: Visualization, Preprocessing, and Linear Regression Modeling of a Dataset

Hasan Mahmud, Ph.D.
Professor, Department of CSE, IUT
Md. Tanvir Hossain Saikat
Junior Lecturer, Department of CSE, IUT

September 22, 2024

Contents

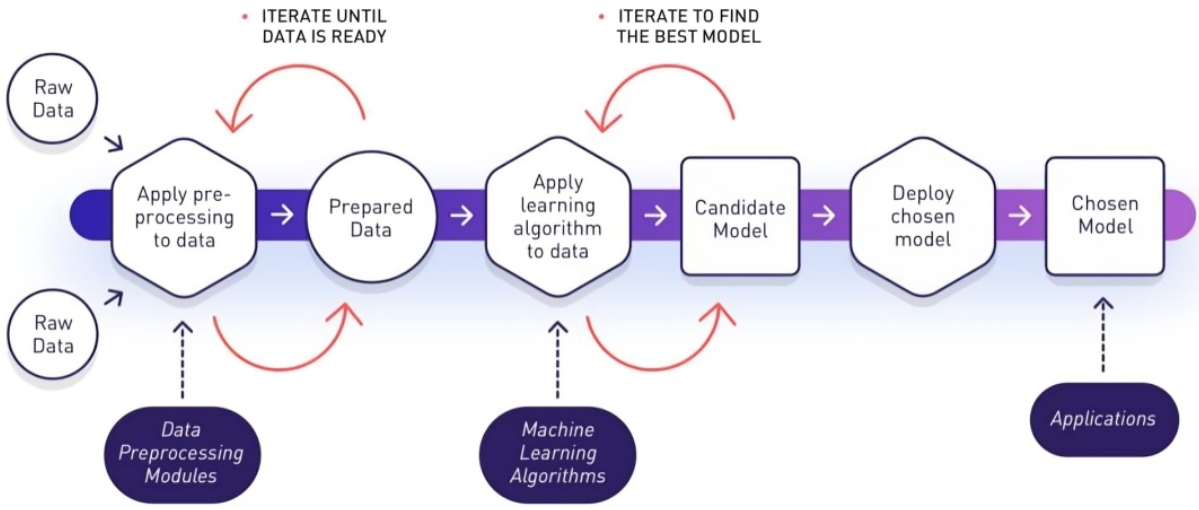
1 Objectives

- To know loading the dataset and split them into training and testing sets
- To know the basic Cross-Validation Task
- To use the Matplot Library for feature analysis and scores
- To know the use of Linear Regression
- To understand how to generate the regression line using numerical method

2 Problem Discussion

2.1 Preprocessing and Visualization

In this lesson you will be implementing the Machine Learning steps which paves the path of training a model. You will be pre-processing the dataset and preparing the dataset. You will also have to visualize different features of the dataset and divide them into training and testing sets.



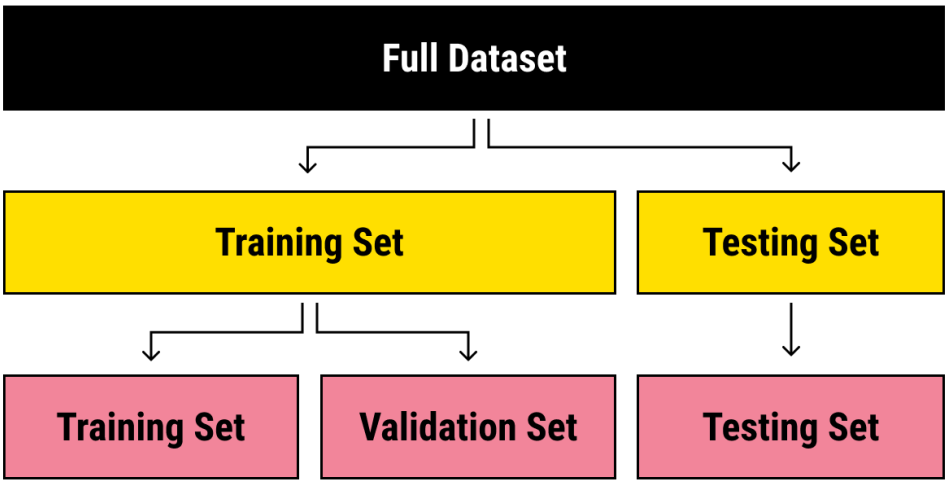
2.1.1 Train, Test and Validation sets

You don't want your model to over-learn from training data and perform poorly after being deployed in production. You need to have a mechanism to assess how well your model is generalizing. For this purpose, a testing dataset is usually separated from the data. Next, a validation dataset, while not strictly crucial, is quite helpful to avoid training your algorithm on the same type of data and to evaluate your model effectively.

1. **Training Dataset:** The sample of data used to fit the model. The actual dataset that we use to train the model (weights and biases in the case of a Neural Network). The model sees and learns from this data.
2. **Validation Dataset:** The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. The validation set is used to evaluate a given model, but this is for frequent

evaluation. We, as machine learning engineers, use this data to fine-tune the model hyperparameters. Hence the model occasionally sees this data, but never does it "Learn" from this. We use the validation set results, and update higher level hyperparameters. So the validation set affects a model, but only indirectly. The validation set is also known as the Dev set or the Development set. This makes sense since this dataset helps during the "development" stage of the model.

3. **Test Dataset:** The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset. The Test dataset provides the gold standard used to evaluate the model. It is only used once a model is completely trained(using the train and validation sets). The test set is generally what is used to evaluate competing models (For example on many Kaggle competitions, the validation set is released initially along with the training set and the actual test set is only released when the competition is about to close, and it is the result of the the model on the Test set that decides the winner). Many a times the validation set is used as the test set, but it is not good practice. The test set is generally well curated. It contains carefully sampled data that spans the various classes that the model would face, when used in the real world.



3 Linear Regression

At its most basic, **linear regression** means finding the best possible line to fit a group of data points that seem to have some kind of linear relationship. Linear Regression comes under the category of **supervised machine learning algorithms**. Regression problems try to predict results within a continuous output, i.e., they try to map input variables to some continuous function. Some examples of regression problems are:

1. Predicting the price of houses given the area, number of rooms, etc.
2. Calculating fare for a taxi depending on the distance, traffic, etc.

In any supervised learning problem, our goal is simple: Given a training set, we want to learn a function $h : X \to Y$ so that $h(x)$ is a good prediction for the corresponding value of y. Here, $h(x)$ is called the **hypothesis function**.

3.1 Define Hypothesis Function

For **linear regression**, our hypothesis is:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n$$

3.2 Define Cost Function

The **cost function** is defined from the error metric that we choose. There are several error metrics such as:

1. The **Mean Absolute Error (MAE)** is the simplest regression error metric to understand. We'll calculate the residual for every data point, taking only the absolute value of each so that negative and positive residuals do not cancel out. We then take the average of all these residuals:

   $$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m} |y_i - h_\theta(x_i)|$$

   where y_i is the actual value, $h_\theta(x_i)$ is the predicted value, and m is the number of data points.

2. The **Mean Squared Error (MSE)** is just like the MAE, but squares the difference before summing them all instead of using the absolute value. The equation for MSE is:

   $$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m} (y_i - h_\theta(x_i))^2$$

3. **R-squared** is a statistical measure of how close the data are to the fitted regression line. It is generally used to measure the accuracy of our linear model, mathematically:

   $$R^2 = 1 - \frac{\text{SSR}}{\text{SST}}$$

   where SSR is the sum of squares of residuals and SST is the total sum of squares.

The **cost function** is usually defined from the MSE as:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

3.3 Gradient Descent

Gradient descent in our context is an optimization algorithm that aims to adjust the parameters in order to minimize the cost function.

The main update step for gradient descent is:

Repeat{

$$\theta_j := \theta_j - \alpha \frac{1}{m}\sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)}$$

(simultaneously update θ_j for $j = 0, \ldots, n$)**}**

So we multiply the derivative of the cost function with the learning rate (α) and subtract it from the present value of the parameters (θ) to get the new updated parameters (θ).

3.4 Plot Cost Function vs Epoch

We can record the cost function at each update step (epoch) to plot this graph.

3.5 Using Numerical Method to Find the Regression Line

The equation of the **regression line** can be represented as:

$$h(x_i) = b_0 + b_1 x_i$$

Here,

- $h(x_i)$ represents the predicted response value for the i-th observation.
- b_0 and b_1 are **regression coefficients**, representing the y-intercept and slope of the regression line, respectively.

To create our model, we must "learn" or estimate the values of regression coefficients b_0 and b_1.

Once we've estimated these coefficients, we can use the model to predict responses!

We define the **squared error** or **cost function**, J, as:

$$J(b_0, b_1) = \frac{1}{2m} \sum_{i=1}^{m} (h(x_i) - y_i)^2$$

where m is the number of training examples, $h(x_i)$ is the predicted value for the i-th example, and y_i is the actual value for the i-th example.

Our task is to find the values of b_0 and b_1 for which $J(b_0, b_1)$ is minimum! Without going into the mathematical details, we can use the following equations to find the estimates:

$$b_1 = \frac{\sum_{i=1}^{m}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{m}(x_i - \bar{x})^2}$$

$$b_0 = \bar{y} - b_1 \bar{x}$$

where $\bar{x}$ is the mean of the input values and $\bar{y}$ is the mean of the output values.