Machine Learning

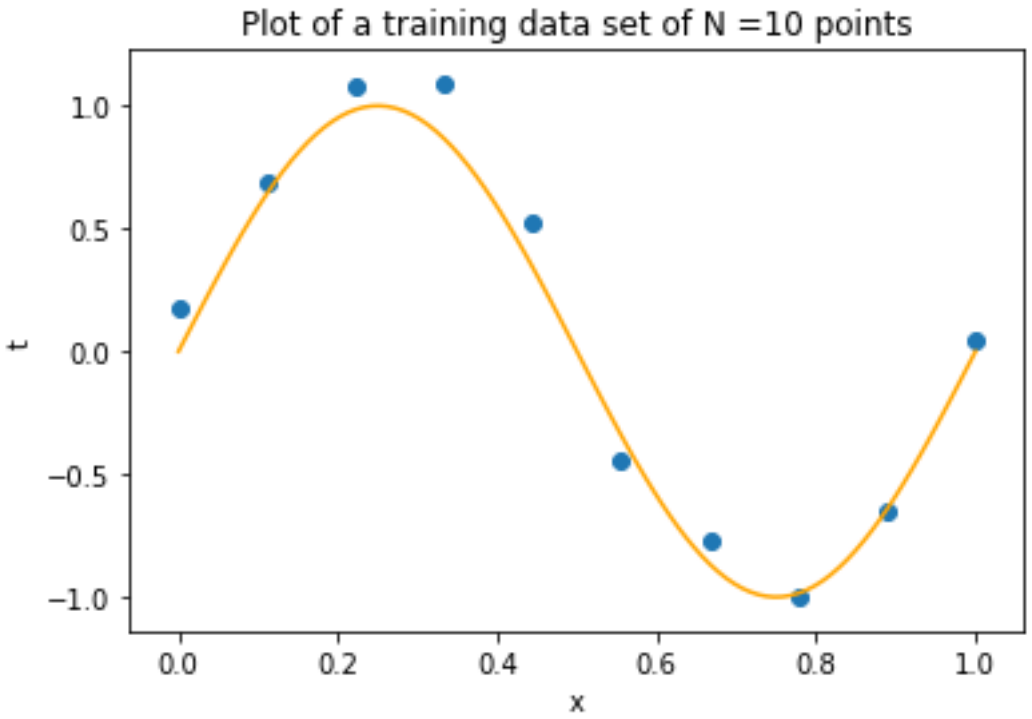
ASSIGNMENT I

Prepared by,
Debasis Mohanty (UEBA19001)

Polynomial Curve Fitting:

We begin by introducing a simple regression problem. Suppose we observe a real-valued input variable x and we wish to use this observation to predict the value of a real-valued target variable t. For the present purposes, it is instructive to consider an artificial example using synthetically generated data because we then know the precise process that generated the data for comparison against any learned model.

Now suppose that we are given a training set comprising N observations of x, written $x \equiv (x_1, \ldots, x_N)^T$, together with corresponding observations of the values of t, denoted $t \equiv (t_1, \ldots, t_N)^T$. The below figure shows a plot of a training set comprising N = 10 data points. The input data set x in the below figure was generated by choosing values of x_n, for n = 1, . . . , N, spaced uniformly in range [0, 1], and the target data set t was obtained by first computing the corresponding values of the function $\sin(2\pi x)$ and then adding a small level of random noise having a Gaussian distribution to each such point in order to obtain the corresponding value t_n. By generating data in this way, we are capturing a property of many real data sets, namely that they possess an underlying regularity, which we wish to learn, but that individual observations are corrupted by random noise. This noise might arise from intrinsically stochastic (i.e. random) processes such as radioactive decay but more typically is due to there being sources of variability that are themselves unobserved.



Our goal is to exploit this training set in order to make predictions of the value $\hat{t}$ of the target variable for some new value $\hat{x}$ of the input variable. As we shall see later, this involves implicitly trying to discover the underlying function $\sin(2\pi x)$. This is intrinsically a difficult problem as we must

generalize from a finite data set. Furthermore, the observed data are corrupted with noise, and so for a given $\hat{x}$ there is uncertainty as to the appropriate value for $\hat{t}$. Probability theory provides a framework for expressing such uncertainty in a precise and quantitative manner, and decision theory, allows us to exploit this probabilistic representation in order to make predictions that are optimal according to appropriate criteria.

For the moment, however, we shall proceed rather informally and consider a simple approach based on curve fitting. We shall fit the data using a polynomial function of the form where M is the order of the polynomial, and x^j denotes x raised to the power of j.

$$y(x, w) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

The polynomial coefficients $w_0, \ldots w_M$ are collectively denoted by the vector w. Note that, although the polynomial function $y(x, w)$ is a nonlinear function of x, it is a linear function of the coefficients w. Functions, such as the polynomial, which are linear in the unknown parameters have important properties and are called linear models.

The values of the coefficients will be determined by fitting the polynomial to the training data. This can be done by minimizing an error function that measures the misfit between the function $y(x, w)$, for any given value of w, and the training set data points. One simple choice of error function, which is widely used, is given by the sum of the squares of the errors between the predictions $y(x_n, w)$for each data point x_n and the corresponding target values t_n, so that we minimize

$$E(w) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, w) - t_n\}^2$$

where the factor of ½ is half of the mean of the squares of $\{y(x_n, w) - t_n\}$ or the difference between the predicted value and the actual value. This function is otherwise called the "Squared error function", or "Mean squared error". The mean is halved as a convenience for the computation of the gradient descent, as the derivative term of the square function will cancel out the ½ term. For the moment we simply note that it is a nonnegative quantity that would be zero if, and only if, the function $y(x, w)$ were to pass exactly through each training data point.

We can solve the curve fitting problem by choosing the value of w for which E(w) is as small as possible. Because the error function is a quadratic function of the coefficients w, its derivatives with respect to the coefficients will be linear in the elements of w, and so the minimization of the error function has a unique solution, denoted by w^*, which can be found in closed form. The resulting polynomial is given by the function $y(x, w^*)$.

There remains the problem of choosing the order M of the polynomial, and as we shall see this will turn out to be an example of an important concept called model comparison or model selection. In below figure we have shown four examples of the results of fitting polynomials having orders M = 1, 3, 5 and 9 to the data set shown in above figure.

We notice that the first order (M = 1) polynomials give rather poor fits to the data and consequently rather poor representations of the function sin(2 π x). The third order (M = 3) polynomial seems to give the best fit to the function sin(2 π x) of the examples shown in figure below followed by the fifth order (M = 5). When we go to a much higher order polynomial (M = 9), we obtain an excellent fit to the training data. In fact, the polynomial passes exactly through each data point and E(w^*) = 0. However, the fitted curve oscillates wildly and gives a very poor representation of the function sin(2 π x). This latter behaviour is known as over-fitting.