Time Series Prediction

Viraj Savaliya
Electrical and Computer Engineering
Arizona State University
Tempe, Arizona, USA
Viraj.Savaliya@asu.edu

***Abstract*—The goal of the report is to investigate linear regression model for time series data, predicting a next sample value in time based on the dataset using tap delay lines. In this assignment the time sample is predicted using linear and non-linear features for the hypothesis equation and the results are compared based on how fitting the model is for different tap delays and different features and the mean square error obtained in each case.**

Keywords—Time Series, linear regression, gradient descent, Tap delay

I. Introduction

In the current decade is known as the decade for Machine Learning. In every sector there is tons and tons of data which can be processed and used for getting valuable information or making useful predictions about the foreseeable future values. Machine learning has caught attention in every sector like engineering, business, trading, medicine, energy, agriculture, etc. Having powerful computers available nowadays to process very complex and large amount of computations aids to the demand and resource requirement of Machine Learning. Data Science and Machine learning are driving all the sectors in their development and have become and essential part of the industry. Having the knowledge and understanding the basics of the computations behind the Machine learning models is important before using the tools and to progress in the field. Linear regression is one of the important parts of all of this.

Regression is basically developing relationship between various variables. It means the future value or the next sample in data is dependent on various independent features which contribute to predict the value of one quantity or the next data sample. Regression can be termed in simple words as an analysis done to find an equation that maps the various features to other feature of interest sufficiently. It is a mathematical equation about the relationship of the various variables for a feature of interest. We can observe how a particular data sample is influenced by number of parameters or features. Linear regression is a simple and widely used regression method to model these relationships in data and interpreting results.

Times series data is a set of values which are dependent on time or possess temporal nature. In a times series prediction, the value for that set of data at some time ahead from now i.e. in future is to be predicted from values known at a given time. For example, if the time series equation is given by *Y(t)*, then we need to predict some time sample of future *Y(t+h)* using the values available at time t. The value in the future can be dependent on various values from the past and can also have linear and non-linear relationship amongst them.

II. Linear Regression Training Model

A basic linear regression model can be described as to fit a line passing through a set of datapoints and representing the dataset very closely using a straight line. The equation of line is given as:

$$y = m * x + c \qquad (1)$$

where, m = slope of the line
c = y – intercept of the line

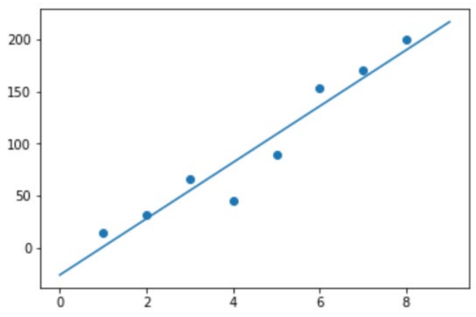

Fig 1. Straight line relation between points

In the above figure, the straight line best describes the relationship between the dataset points and hence linear regression can be done to represent the relationship. By finding the value of slope(m) and the y-intercept(c) of the line we can find the line most closely passing through all data points. The closeness of the line can be represented by cost function and the equation of the line is basically a hypothesis equation we need to predict.

A. Model Parameters

The linear regression model hypothesis and the cost function can be given by,

$$h_\theta(x) = \theta_0 + \theta_1 * x \qquad (2)$$

$$J(\theta_0, \theta_1) = \frac{1}{2*m} * \sum_{i=1}^{m} (h_\theta(x)^i - y^i)^2 \qquad (3)$$

Where,

θ_0, θ_1 : Parameters of the model

$h_\theta(x)$: Hypothesis

$J(\theta_0, \theta_1)$: Cost function

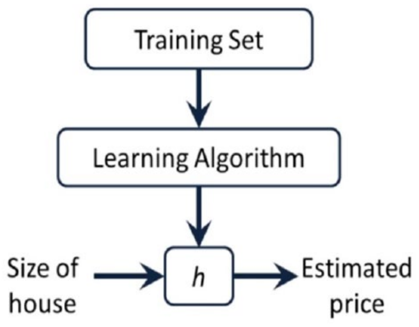

Fig 2. Hypothesis/ Prediction model[1]

The goal over here is to minimize the cost function, which is basically to reduce the mean square error of the hypothesis value from the actual value. The hypothesis model is given some input which is the features and a training data set along with a leaning model to predict the next sample data. In figure 2 the size of the house is the feature provided and the estimated price is the next sample we need based on the prediction model. We need to assume some initial parameter values to start with and then keep updating the parameters in every iteration using a small positive learning rate. This is the gradient descent algorithm method where you need to reach the minimum value or the lowest point of the curve where the parameters will converge to a constant minimum value depicting that the equation obtained using this converged value represents the dataset most closely and has lowest mean square error possible. Gradient descent is an iterative optimization algorithm to reduce the loss and make accurate predictions. To find the value of the parameters we need to differentiate the cost function w.r.t each parameter in order to find its minimum value and then update it over some iterations using the learning rate. The equation for each parameter in gradient descent convergence can be given as follows:

$$\theta_0(gradient) = \frac{\partial}{\partial\theta_0} J\,(\theta_0, \theta_1)$$

$$\theta_1(gradient) = \frac{\partial}{\partial\theta_1} J\,(\theta_0, \theta_1) \qquad (4)$$

After, differentiating the cost function w.r.t the parameters the equation is as follows:

$$\theta_0(gradient) == \frac{1}{m} * \sum_{i=1}^{m}(h_\theta\,(x)^i - y^i)$$

$$\theta_0(gradient) == \frac{1}{m} * \sum_{i=1}^{m}(h_\theta\,(x)^i - y^i) * x^i \qquad (5)$$

Now, the updating rules for θ_0 *and* θ_1 over number of iterations using the gradient values and α as the learning rate can be given by:

$$\theta_0 = \theta_0 - \alpha * \theta_0(gradient)$$

$$\theta_1 = \theta_1 - \alpha * \theta_1(gradient) \qquad (6)$$

B. Algorithm Description

All the parameters and the equations involved for performing linear regression using gradient descent are given by equations (1) to (6). Now to start with the algorithm, first we need to read the entire dataset and then store it. This is done as the training of the model is offline and there is predefined dataset provided for training. Then we pass the x and y values as input to our training model. The linear regression model tries to generate a straight line which fits the entire input data points and generalize it. The training model initializes some value for Parameters to start with and then using the cost function which is the mean square error it keeps on updating the parameter value using a learning rate. The aim of the model is to find the minima of the cost function which is the minimum values of the parameters for which the model will converge and the value of the parameters remain almost same after the minima for any iterations thereafter. For iterations to reach the minima a learning rate is used which is basically a jump made to quickly reach the minima value. It is a control parameter to adjust the weights in respect to the loss gradient. The value of this leaning rate should be small. If we choose a very small number, then it will take longer time to reach minima as there will be tiny updates. If the learning rate is too high, then there will be undesirable divergent behavior in the cost function and maybe the values will keep jumping higher and higher and never reach minima i.e. never converge. Once the converged parameter values are available then we have the correct hypothesis equation based on the entire dataset i.e. we have the equation of the straight line which we can use to predict the next sample values we want.

III. PSEUDO CODE

In the given assignment, time series data prediction is done for P data samples where P = 124. The data prediction is done using a tap delay line T, where T is either 2, 5, 8 or 10. The data sample to be predicted is $y(n + T + 1)$ and it is predicted based on $\{\, y(n), y(n+1), y(n+2), \dots\dots\dots.., y(n+T)\}$. Hence, the entire dataset is broken down into small datasets for each value of y based on the tap T value and n value for predicting a single data sample. Here the next sample in time is predicted from a tapped dataset. Also, the features of the model are varied, and linear and non-linear features are used for the hypothesis equation. The equations for the experiment having linear and non-linear features in the hypothesis equation are as follows:

$$h_\theta\,(x) = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2 + \theta_3 * x_3 \qquad (7)$$

$$h_\theta\,(x) = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2^2 + \theta_3 * x_3^3 \qquad (8)$$

$$h_\theta\,(x) = \theta_0 + \theta_1 * x_1 + \theta_2 * \log(x_2) + \theta_3 * \sin x_3 \qquad (9)$$

For Tap more than 2, the above equations have linear features for θ after θ_3 i.e. equation 8 can be written in general form as:

$$h_\theta\,(x) = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2^2 + \theta_3 * x_3^3 + \sum \theta_n * x_n^n$$

The pseudo code for the given assignment for a model using different tap delay lines is as follows:

```
1.  Import libraries (sklearn, matplotlib, pandas,
    numpy)

2.  Read dataset and store in an array

3.  Generate X value array for the dataset

4.  Initialize the Tap delay line array for T = 2,
    5, 8, 10

5.  For T in Tap delay line:

6.      Y_val = Initialize array for T value and
    dataset

7.      X_val = Compute array based on equations

8.      linear_regression_Model.fit(X_val, Y_val)

9.      Y_prediction = Compute prediction value
    array based on model

10. Plot graph of the Y_Prediction using
    matplotlib
```

IV. Results

The results obtained for all the equations 7, 8 and 9 and for different Tap delays are provided in the figures below:

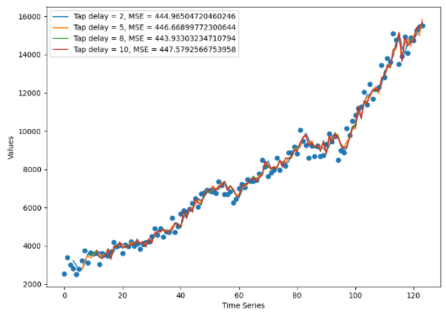

Fig 3. Equation 7 hypothesis prediction

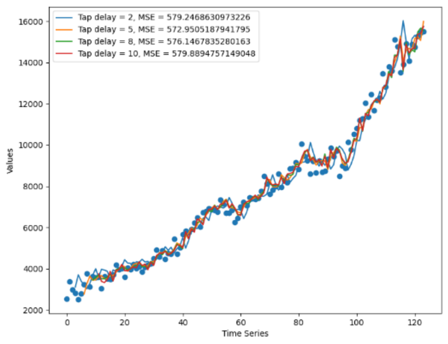

Fig 4. Equation 8 hypothesis prediction

All the above results in Figure 3, 4 and 5 were obtained using the Linear Regression from Sckit-learn library for Machine Learning in Python. A linear regression model for the basic hypothesis given in Equation 2 was also implemented from scratch without using any libraries and the cost function convergence over the number of iterations was observed for 0.001 as the learning rate as shown in Figure 6.

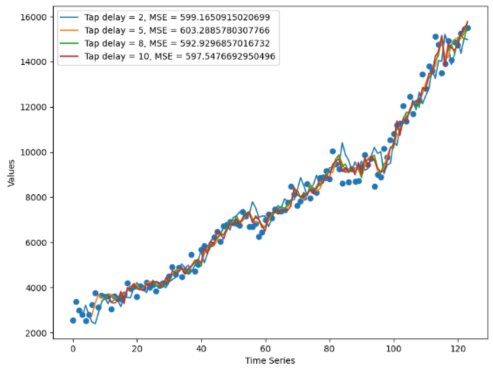

Fig 5. Equation 9 hypothesis prediction

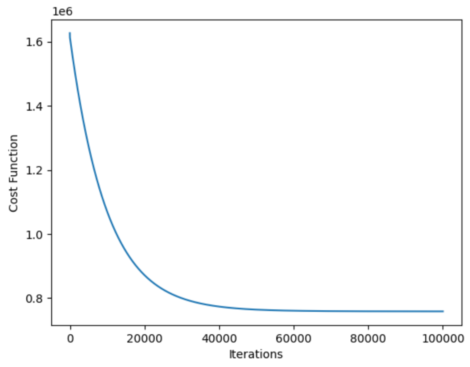

Fig 6. Cost function convergence over iterations

V. Conclusion

In the assignment, times series data prediction using a linear regression model for linear and non-linear features and for 2, 5, 8 and 10 Tap delay lines was observed. From the results of the assignment it can be clearly seen that the hypothesis equation with only linear features has the least root mean square error as compared to the equations with non-linear features. Also, there is no generic trend observed for root mean square error for different tap delay values for the given set of equations considered in the assignment. The tap delay line of 8 has the least mean square error observed and hence, it most closely follows the dataset compared to other. Out of the three equations used the first equation with only linear features is the best fit model for the given dataset as it has the least MSE out of all the equations. We can also see that the cost function converges after a given number of iterations as it reaches a minima for a set of parameters.

References

[1] EEE511: ANC, Univarite Linear Regression.pdf, by Dr. Jennie Si, Fall 2020

[2] Joseph J. Bautista; Linear Regression using Gradient Descent; URL: https://towardsdatascience.com/linear-regression-using-gradient-descent-in-10-lines-of-code-642f995339c0

[3] Mirko Stojilkovic; Linear Regression in Python; URL: https://realpython.com/linear-regression-in-python/

[4] Sklearn Linear Regression, URL: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html