EE-439 Introduction to Machine Learning

Project Report

Projection of Hourly Load Demand to Calculate Net Metering

ZAIN MUJAHID (2016-EE-304)

January 7, 2020

EE-439 Introduction to Machine Learning

Project Report

Introduction:

This project takes Date, Time, PV and Temperature as input data and with the help of Machine Learning Algorithms predicts the hourly residential load demand, which is then further used to calculate the bill of that certain hour either it be bill (consumption of power) or the profit earned by selling the unused PV.

Dataset:

This project uses PV generation and load profile data of net zero energy homes in South Australia as past data. There are 24 readings taken for each day for a whole year making a total of 8760 instances of past data.

Hypothesis & Parameters:

The general hypothesis that we took is given below:

$$\text{Hypothesis:} \quad h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

The main objective is to find the following parameters:

$$\text{Parameters: } \theta_0, \theta_1, \ldots, \theta_n$$

Such as our final hypothesis is the best fit through data.

Cost Function:

We used the squared error cost function J(θ) as shown below:

Cost function:

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

The goal is to find values of parameters Theta which minimizes this cost function. For this purpose we have used gradient descent algorithm.

Gradient Descent:

We implemented the following algorithm to find parameters Theta by minimizing cost function as shown below:

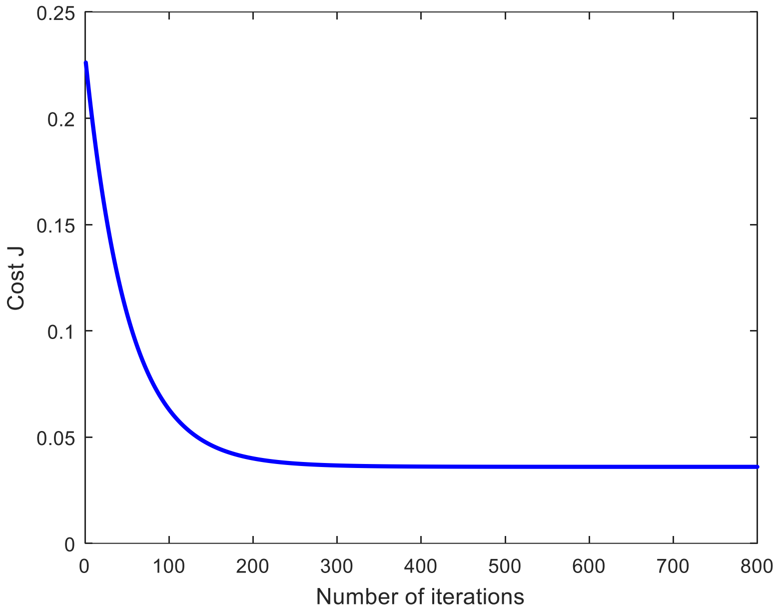
Repeat $\{$

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

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

$\}$

This algorithm first takes random values of parameters theta and keep on improving them iteratively by decreasing or increasing their values with **learning rate α.** In our case we chose $\alpha = 0.01$ with 800 iterations. To make sure that our gradient descent in not **overshooting** we plot a **convergence graph** which shows the progress of minimizing cost function. We get the following convergence graph for our project:



The flatness of this plot shows that the gradient descent have reached to a local minima of cost function. Note that by changing the initial values of vector theta may change the initial costs and final values of parameters theta as gradient descent may reach some other local minima of cost function but at the end the plot should converge like in the figure above. If this plot goes up with increasing iterations, try taking smaller value of learning rate α.

Normal Equations:

To cross check our final results we have used numerical methods to find the parameters theta from normal equations. And we predict Load by using these theta values and then checks how much it varies from the results of gradient descent. Following Matrix Equation is being used in our program:

$$\theta = (X^T X)^{-1} X^T y$$

Bill Calculation:

Once we have our values of parameters Theta which minimizes our cost function, we can then predict the hourly load by giving the input Date, Time, PV, Temperature to our hypothesis. This will predict the load demand in KWh of that specific hour.

Now if Load is greater than PV we can calculate the bill by simply multiplying their difference with the price of single unit of electricity. If PV is greater than the load then their difference multiplied with the unit selling price will give the profit of the specific hour.

References:

- PV generation and load profile data of net zero energy homes in South Australia
- Machine Learning by Stanford
- SCHEDULE OF ELECTRICITY TARIFF W.E.F 2019