**Assignment 1**

483

**(a) Team Name:** XYZ

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Name** | **Email** | **% Contribution** |
| 1 | Steven Espin | sespin@csu.fullerto.edu | 33.33% |
| 2 | Navie Vurdien | nvurdien@csu.fullerton.edu | 33.33% |
| 3 | Hemanth Sivakumar | hemanth@csu.fullerton.edu | 33.33% |

**(b) Scikit learn**

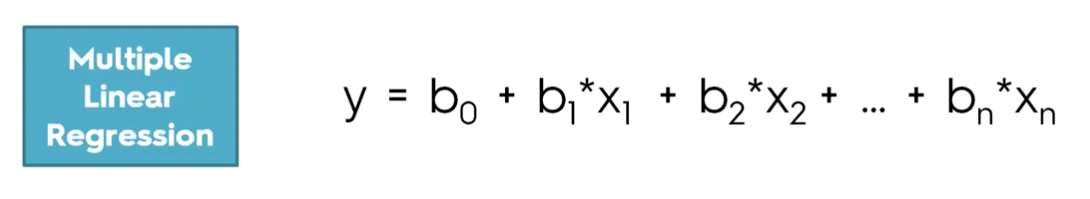
The software that we used was Python machine learning package, scikit learn, which could be found on their website at <https://www.python.org> . Python allowed us to import the dataset onto it which then gave us various commands to work with. We were able to plot each variable to accurately visualize if there was any relationship between each other.

**(c) Linear and Non-linear models**

Both linear and non-linear regression models were used in this project to try and predict one y (Idx) using multiple x variables (Tm, Pr, Th, Sv).

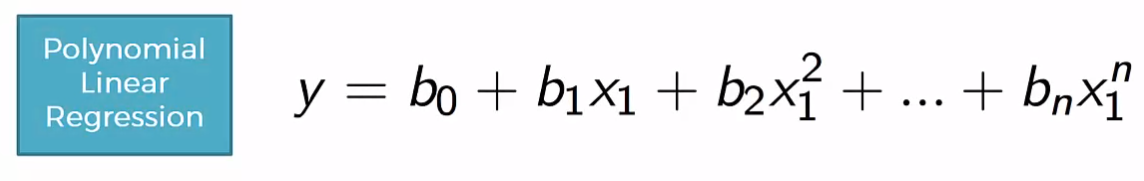
**Linear model**

A multiple linear regression model is used to predict y based on multiple x values. A linear method is useful in some cases, however may not fit our data as well as a non-linear approach.

****

**Non-linear Model (polynomial)**

A polynomial regression model was also used in this project, because a polynomial line can often fit data better. However, we must be careful of fitting training data too closely or overfitting.



**(i) Process**

**Step 1:** Decide on size of training and test set

After some research online and tests among ourselves, we decided to allocate 80% of the data to training and 20% to test our model. We felt that 20% was a fair number to allocate for tests considering the amount of data we were given.

**Step 2:** Split Training and Test Set

Next step was to split our data into training and test sets. We decided that picking our test cases at random was the best option to avoid getting data with similar values in the dataset. To accomplish this, we used a function in scikit learn called cross\_validation.

**Step 3:** Pick Degree of Regression

Final step was to pick a degree for our model and test our predictions. With a polynomial regression, we found that changing the degree had a big impact on our results.

**(ii) Analyzing Uncertainty**

We used mean absolute error and mean squared error to analyze the uncertainty of our models.

Mean Absolute Error - used to measure how close predictions are to eventual outcomes. It is computed by averaging out the absolute errors.

Mean Squared Error - similar to mean absolute error, however is used when error is squared. Error often squared to have simpler calculations later.

These are the values found for each of our models created. Note: Degree 1 is our linear model.

|  |  |  |
| --- | --- | --- |
| **Degree** | **Mean Absolute Error** | **Mean Squared Error** |
| **1** | **0.8200530876503422** | **1.1482282907374695** |
| **2** | **0.2507172572584658** | **0.09576087779729209** |
| **3** | **0.22759281221674374** | **0.07682939829237229** |
| **4** | **0.21174017572771014** | **0.06677175579225553** |
| **5** | **0.219494526663395** | **0.07080133819945196** |

**(iii) Best Model**

Based on our tests, we found that a **polynomial regression model of degree 4** was the best model to fit our data. The reason we believe this is because the mean absolute and squared error were the lowest on this model. This means that in general, our line is doing a better job at fitting our data.

In addition, if we were to keep going up higher in degree, the model will become prone to overfitting and become too complex for its own good. This is why we believe that a degree of 4 is a good medium of fitting our data well enough without overfitting.

In general, our model did a good job at predicting the values of our test cases. Below is a comparison of test case results (left) and our models predictions (right).

