Information Engineering and Technology Faculty
GermanUniversity in Cairo

Machine learning Assignment 1&2 Report

Author: Youssef Tarek Abdelhady
Supervisor: Dr. Maggie Ezzat
Submission Date: 25 April, 2020

Contents

Assignment1

Linear Regression

Linear Regression is a supervised machine learning algorithm where a given data-set is with continuous output. The algorithm is used to predict output values with a continuous range when given the data input rather than trying to classify them into categories or classes, there are two approaches when using linear regression which are; **Single variable** or **Multivariable**. In the multivariable technique there is an aim to find the best hypothesis function (**h(x) → y**) by choosing the most suitable polynomial degree that result in a minimal cost function value **J**. Moreover, this goal is achieved using the **gradient descent** algorithm, where the theta values are continuously calculated using the gradient descent formula until convergence, convergence is reached when the optimal theta values are calculated.

The first assignment approach was to apply both single and multivariable linear regression techniques to the first data-set, and used to further predict house prices. The second assignment approach was to also apply the multivariable linear regression technique to another different data-set, but this time by also using the **Model-selection** technique, having an aim of enhancing our model's performance and prediction.

Model-Selection

Firstly, our data-set had two irrelevant columns that were removed (id and date), then we needed to see which features had the most impact on the calculated output prices, this was achieved using a correlation matrix that displays each feature's correlation with respect to the price, by this some features were further reduced (having a correlation less than 0.5) to enhance our model from falling within either the **under** or **over** fitting scenarios. By

applying this to our data-set, five features were chosen to be used in our machine learning model (bathrooms, sqft_living, grade, sqft_above, sqft_living15).

Furthermore, the data-set was further divided into 3 portions using the data_split function which are; **Training, Cross-validation and Testing**, that are 60%, 20%, 20% respectively. The data_split function uses a **random seed** each time it is called to divide the data, thus applying the **Random re-sampling** concept. In addition, the resulting three data portions (Training, CV and Testing) are then inserted to the **featureNormalize** to make the data values in the same range. The final step for our data to be ready was to concatenate a column of 1s that represents the **x0** term.

Training Phase

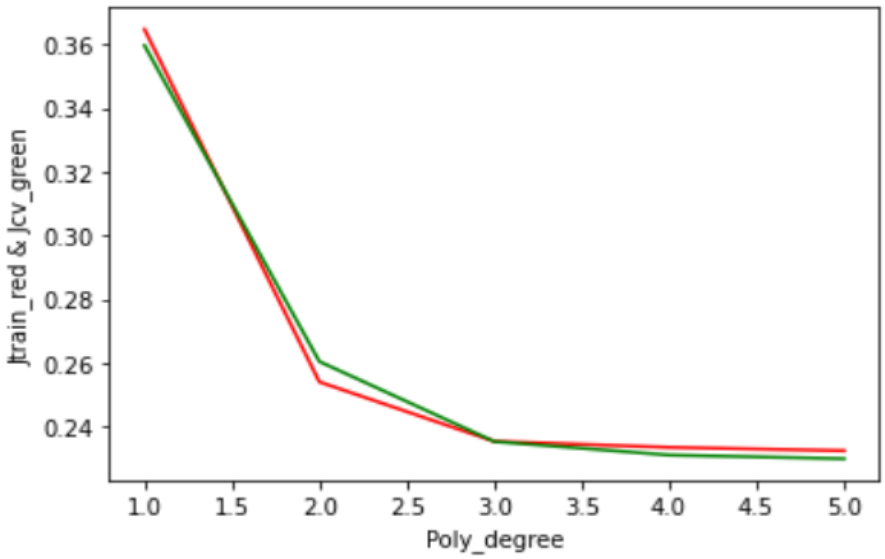
Using the training data portion, the gradient descent algorithm was used on the data using the **gradientdescentmulti** function, where iterations were applied to calculate the optimal theta values and minimal cost function value for each hypothesis degree (X power n), where it was calculated for hypothesis from degree **1 till 5.** After comparing the calculated cost function values, it was found that the cost function **J_train5** had the smallest value of 0.23247874230765692, thus choosing its corresponding theta values to be used.

Cross-validation Phase

The CV phase has an aim of choosing which hypothesis **degree** to use that would result in the minimal cost function value J_cv. By using the CV portion of the data to calculate J_cv, from J1 →J5 that were calculated using the **computeCostMulti** function. The resulting values showed that J5 had the minimal value of 0.22995553441512978 , which means that the hypothesis chosen will have a degree of 5.

Testing Phase

Finally using the Testing data to compute the cost J_test using the previously chosen hypothesis degree (from cv) and theta values from the training phase it was found that the **J_test** had a value of 0.22590417826714657 and as shown in the Plotting figure in the assignment that the chosen **degree** is the best to avoid over-fitting or under-fitting.
Plotting J_train and J_cv against the polynomial hypothesis degree.



Assignment2

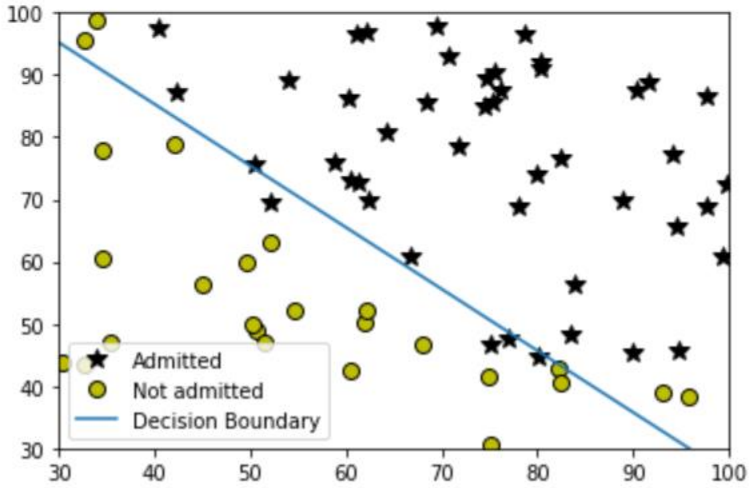
Logistic Regression

Logistic regression is a supervised learning **classification** algorithm, this algorithm is used in predicting the output data to belong to a certain class, this means that here the output isn't continuous, but discrete belonging to a certain class (e.g True or false). Logistic regression similar to linear

regression uses a hypothesis function **h(x)** and a cost function **J** that is needed to be minimized. Logistic regression uses the **sigmoid** function as its hypothesis to be used in the classification process.

In this assignment, firstly we applied the logistic regression technique to the first data-set containing exam scores and we classified the scores whether to be **Admitted** or **Not Admitted**. Moreover, this was done with an aim to test the algorithm's prediction capability, therefore the **data_split** function was used to split the data-set into two portions; **70% Training and 30% Testing,** then the training and testing portions were both split into **x** which is the i/p and **y** which is the o/p (x_train, y_train, x_test, y_test).

The next step was to concatenate 1s to both the x_train and x_test in order to represent the term x0. Moreover, applying the **gradient descent** algorithm to find the best thetas that minimize the cost function J and this was done using the **costFunction(initial_theta, x_train, y_train)** function through many iterations to find the minimal value of J. Furthermore, we can now plot the decision boundary that seperates the two classification classes (Admitted and Not Admitted) as shown.



Finally, the last step is the testing phase where we evaluate our model's prediction using the predict (theta,x_test), that resulted in a training accuracy of 90%.

The second assignment approach was to also apply logistic regression, but in addition to using the **Regularization technique** in order to enhance the model's performance

Logistic Regression with Regularization

In this approach we divide the data-set into three portions which are; **Training**, **Cross-validation** and **Testing** that are 60%, 20% and 20% respectively and this was done using the data_split function also.

The i/p x and o/p y were also separated into; **x_train_reg, x_cv_reg, x_test_reg, y_train_reg, y_cv_reg, y_test_reg.** Moreover, the 1s column that represents x0 was concatenated to x_train_reg, x_cv_reg, x_test_reg.

Most importantly, we use the regularizing term containing lambda.

Training Phase

In the training phase we use the training data portion in many iterations to calculate the cost function value and thetas using the gradient descent algorithm that also includes the **regularizing term** containing lambda and lambda is input with multiple values that are commonly used by engineers in the field. The loop calculates the cost function value using the **costFunctionReg** function using each lambda value **lambda_[i]** in order to find the best hypothesis **theta values** and degree that corresponds to this **lambda value**

Cross-validation Phase

In this phase we aim to choose the best **theta values** and corresponding **lambda,**thi s is done by the calculation of cost J_validate using the **costFunctionReg** function

, but this time using the cross-validation data portion and putting **lambda=0** (no re gularizing term), we then choose the best theta values that correspond to the lambd a value from the training phase, that **minimize the cross-validation error**. It was found that the J_validate value was 4.275176891922121.

Testing Phase

In the last phase we use the chosen theta values and lambda that minimized the cross-validation error and test the chosen values accuracy in predicting the output classification values y while using the **testing data portion**. It was found that the model's train accuracy has decreased to 70% , because the used data-set is small (contains only 100 records) this means that the cross validation and testing data size is only 20 records → leading to almost similar cost function values throughout the iterations , the solution is to use a bigger data-set in both records and number of features.