# Logistic Regression

The following exercise is a logistic regression implementation with regularization from scratch without any usage of any libraries in pyspark. 170,000 text documents (this is 7.6 million lines of text in all), and a test/evaluation data set that consists of 18,700 text documents (almost exactly one million lines of text in all). All but around 6,000 of these text documents are Wikipedia pages; the remaining documents are descriptions of Australian court cases and rulings. Task is to build a classifier that can automatically figure out whether a text document is an Australian court case. We performed some data cleaning practices.

Created a term frequency (TF) for the 20K entries. Created relevant labels for Australian court cases (1) and wiki pages (0) as it’s a classification problem. Finally implemented logistic regression using gradient descent. The LLH function is used as the cost, and subsequently found the gradient. The gradient is also updated with the L2 regularization. The cost after 400 iterations is correctly decreasing with the learning. The weights are taken for the 20K words.

We find the regression coefficients of each word. Words with highest regression coefficients:

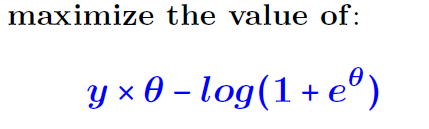
['that', 8), ('mr', 141), ('applicant', 448), ('application', 480), ('tribunal', 491)]

The table gives the word, index and the regression coefficients.

|  |  |  |
| --- | --- | --- |
| Word | Index | Regression Coefficients |
| that | 8 | 113.04 |
| mr | 141 | 112.34 |
| applicant | 448 | 79.20 |
| application | 480 | 61.65 |
| tribunal | 491 | 72.20 |

Similar to the training set created a TF for the testing set. Also, created the relevant labels for Australian court cases and Wiki pages. Using the formula below, predicted on the test data.

The negative and positive predictions are decided based on the higher value of them using this equation.



**Evaluation Metrics:**

True Positive: 360

False Positive: 8

False Negative: 17

True Negative: 18339

Precision: 0.9782  
Recall: 0.9549  
**F1-Score: 0.9664**

The results in the jupyter notebooks are incorrect due to insufficient training (less number of data points).