FIS LAB 2 REPORT

SUBMITTED BY

ANKIT JAIN

FEATURE SELECTION:

The features selected for the lab projects were average number of words in a sentence. Name words from Sherlock movie series, answer words like said, replied, answered and replied. Count of semi colon in a paragraph and number of double quotes.

Each of them were calculated as a number of times they appeared in paragraphs on an average. Then, an appropriate split was decided according to the values noted down. Accordingly, the most appropriate and suitable features were the above mentioned. The average number of words for the author Herman was found to be more than author Arthur. Names from Sherlock were obviously a good fit as they are written by Arthur and the series itself being a huge hit amongst people and thus, making it a good classifying attribute. The number of semi-colons for Herman author were more than Arthur. The replying words were more in Arthur as his books had more number of dialogues when compared to Herman, who wrote more poems. The quotations were again used more by Arthur than Herman as there were more dialogues.

Logistic Regression:

Train:

For this part of the lab, the initial feature matrix values are filled by passing the paragraphs in the TrainingData.txt, used to train the data. The features as and when found fills up the feature matrix that tells us what paragraph has which features along with the known fact that who is the author of that paragraph. This is done for all the possible paragraphs in the training data file. The weights initially passed to the weigh calculation function is 1. Every time a new paragraph is read, a feature list within a list is passed and the corresponding initial weights which is 1 for the first iteration is taken, the dot product is taken using the numpy library as the dot product will be the equation of the line where the slope and the y-intercept are the 2 weights. This is done for all the paragraphs. The first time when we get a value for weights, accordingly we modify the initial weights as the model keeps learning with every iteration of a new paragraph. The final calculated value that is received is the final weights that we get in a list. This is then written to a new file as done in code. This process hence, completes the entire train data with logistic regression. A factor is also considered during our calculation, alpha, that is reduced every time to approximate the weights for updating of weights.

Predict:

Initially, when the user selects predict, the weights that were written in the new file has to be read in order to predict correctly and also the fact that if a person wants to predict a new paragraph, he/she need not train data always. They can directly give the command to predict and they will have a name of the author.

In this, a random paragraph can be predicted whether it is written by Arthur or Herman. When a new paragraph is taken into consideration, the paragraph is sent to the feature function to determine its features. Then, as we have the trained weights ready, a dot product of the feature list of new paragraph and final weights is done. The result is then passed to a sigmoid function that determines who the author of the paragraph is.

The accuracy was found by feeding random texts of paragraphs. Then taking the times the model predicted correctly on total into prediction consideration. Out of 30 random paragraphs, 17 were predicted correctly.

Decision Trees:

Train:

Decision tree is another way to classify data and predict who the author of a given text is. Initially, the decision tree function is made with the first step to calculate the first best attribute. This is done by passing the entire feature matrix to the function and calculating the entropy and hence the gain calculated. The attribute with the maximum gain is the best root feature. Once that is found, that attribute has a left sub-tree and a right sub-tree. The left sub-tree is the side where it is yes. We will then modify the matrix so that it doesn't take the 1 from the attribute selected in the matrix as we know we are on the left side of the tree. We need to modify the original matrix and then keep doing this until the entropy value is 0 or the length of the matrix is 0 as we basically reduce the matrix size by 1 every time we modify the matrix. (This is done in code but there are errors at this point. So, complete training not done and hence, prediction wasn't coded in file as well.) When we reach a node where we have null as our leaf node, we should save the decision on that node saying which author has written the paragraph.

Instruction to Run Code:

When running the code, select train or predict for logistic regression case. If train, it will return or display the final weights that are calculated. Then if you train or not, select predict, which will allow you to enter the file you want to predict upon. Fill the file with some text and press enter. It will predict and display if the author is Herman or Arthur.

Files Included:

- Lab2.py: python file containing code
- Test.txt: a random file with a text for prediction

- TrainingData.txt: file that has random paragraphs written by Arthur and Herman. 70 of them in total used to train.
- Weight.txt: file that has the final trained weights for logistic regression.