CS5691: Pattern Recognition and Machine Learning Assignment 3

Anirud N CE21B014

1 Dataset

The dataset is a combination of 2 publically available dataset:

ENRON Dataset

- ENRON: Enron Dataset Website. This dataset consists of already pre processed emails in the form of .txt for each email.
  All the six folders in the website were used ie enron1, enron2,..enron6.

- Enron1 - Spam: 1500, Ham : 3672
- Enron2 - Spam: 1496 , Ham: 4361
- Enron3 - Spam: 1500, Ham : 4012
- Enron4 - Spam: 4500, Ham : 1500
- Enron5 - Spam: 3675 , Ham : 1500
- Enron6 - Spam: 4500 , Ham : 1500

So in total the Dataset has 17,171 Spam emails and 16145 Ham emails

Spam Assassin Public Mail Corpus Dataset

- Spam Assassin Public Mail Corpus Dataset Dataset Link
- It consists of 3779 Spam Emails and 2000 Ham emails
- Since this dataset was not pre-processed, and the file format was not in .txt, A separate Python script was written to convert this into the required form and pre-processing.

A subset of both datasets were chosen for training our models.
The resulting dataset has 16,625 Ham emails and 20,950 Spam Emails for training.
The dataset was split by 80-20 Rule, 80 % Training, and 20 % Testing data.

2 Algorithms Used:

Three Algorithms were used to train the spam classifier.

2.1 Naive Bayers:

Since this algorithm took a longer time to train, I used a subset of Enron dataset to train this.
Preprocessing was done to remove:

- Remove Punctuations
- Remove Stop words
- Tokenise and Lowercase

First the dictionary of all the words that occur in the emails is made. along with the number of times of occurance in spam and non spam.
Laplace smoothening was then used to handle exceptions, when model does not appear in the train set.

As shown in fig(1) the maximum likelihood estimators were calculated. Then, for the predictions, the conditional

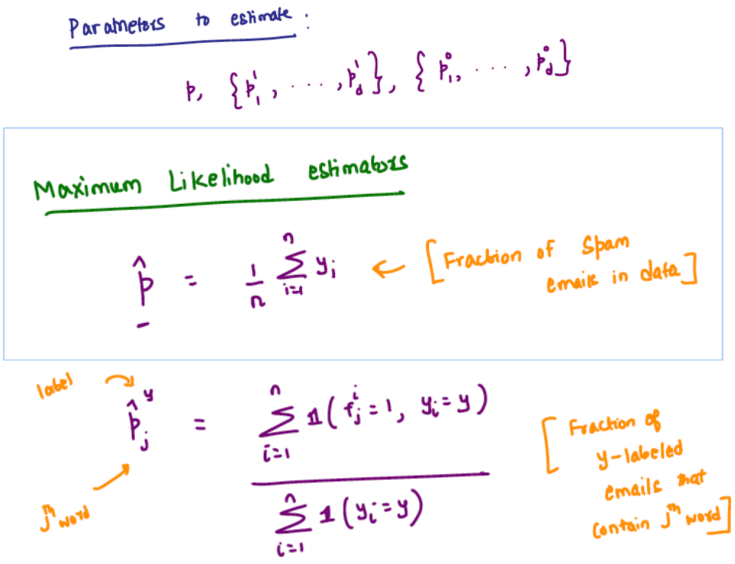

Figure 1: Compute the maximum likelihood estimators

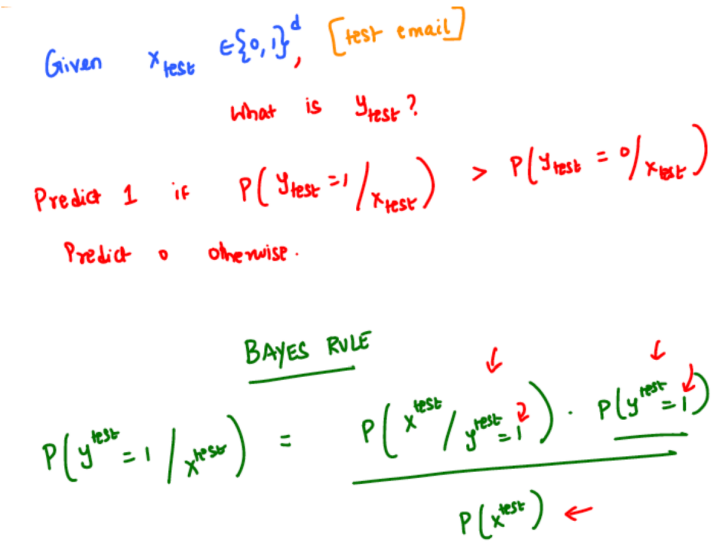

Figure 2: Predictions

probabilities were calculated and compared, as shown in Fig (2), to find the predictions. Fig(3) shows the equations to find the conditional probabilities
The model when tested on enron1 dataset gave an accuracy of **90.61%**

2.2 Perceptron

The Perceptron algorithm was used since the performance of Naive Bayers was not up to the mark. This was trained until 2000 epochs.
The Perceptron algorithm was slightly modified to the form of Gradient Descent, as discussed in class by a professor during the Loss functions lecture.

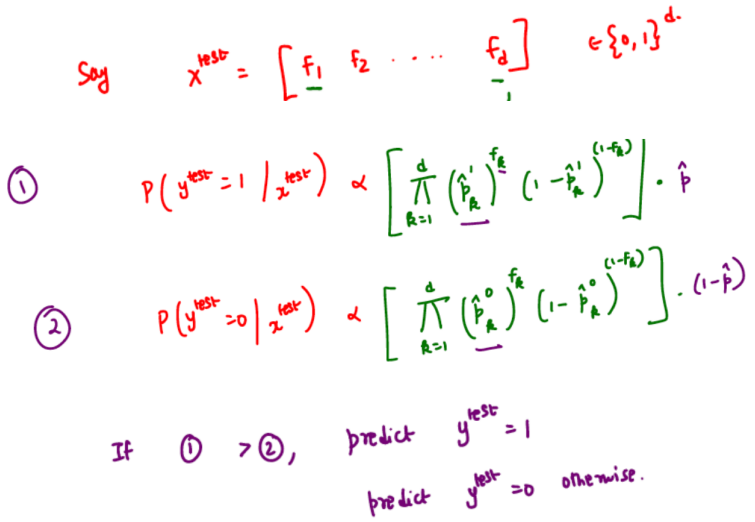

Figure 3: Equation to find probabilities

The Labels were set to 1 for spam and 0 for non-spam. The perceptron update equation is:

$$w_{t+1} = w_t + \alpha * (y_i - \hat{y}_i) * x_i \tag{1}$$

where w_{t+1} are the weights and y_i are the labels of x_i and $\hat{y}_i$ is the prediction made at time t.
α is the Learning rate.
x_i is the feature vector of each email.
Weights are initialized as 0 vectors.
The model was trained on the complete dataset of 35,000 emails.
We define the feature vector as follows:

- First, we find the count, i.e., the frequency of occurrence of each word in spam and ham mails.
- Now, we take only the top k words that have occurred over all the emails.
- These top k frequent words are our features of each data. The number of times these words occur in each email is stored as the feature vectors.

Upon performing hyperparameter tuning, α is set to 0.1. and k is set to 5000
At each epoch of training ,the performance of the model on the test set was observed and plotted.
Fig (4) shows the model's accuracy on test data during its training. As we can see, After 1000 epochs, the model's accuracy converges to **94.47%**.

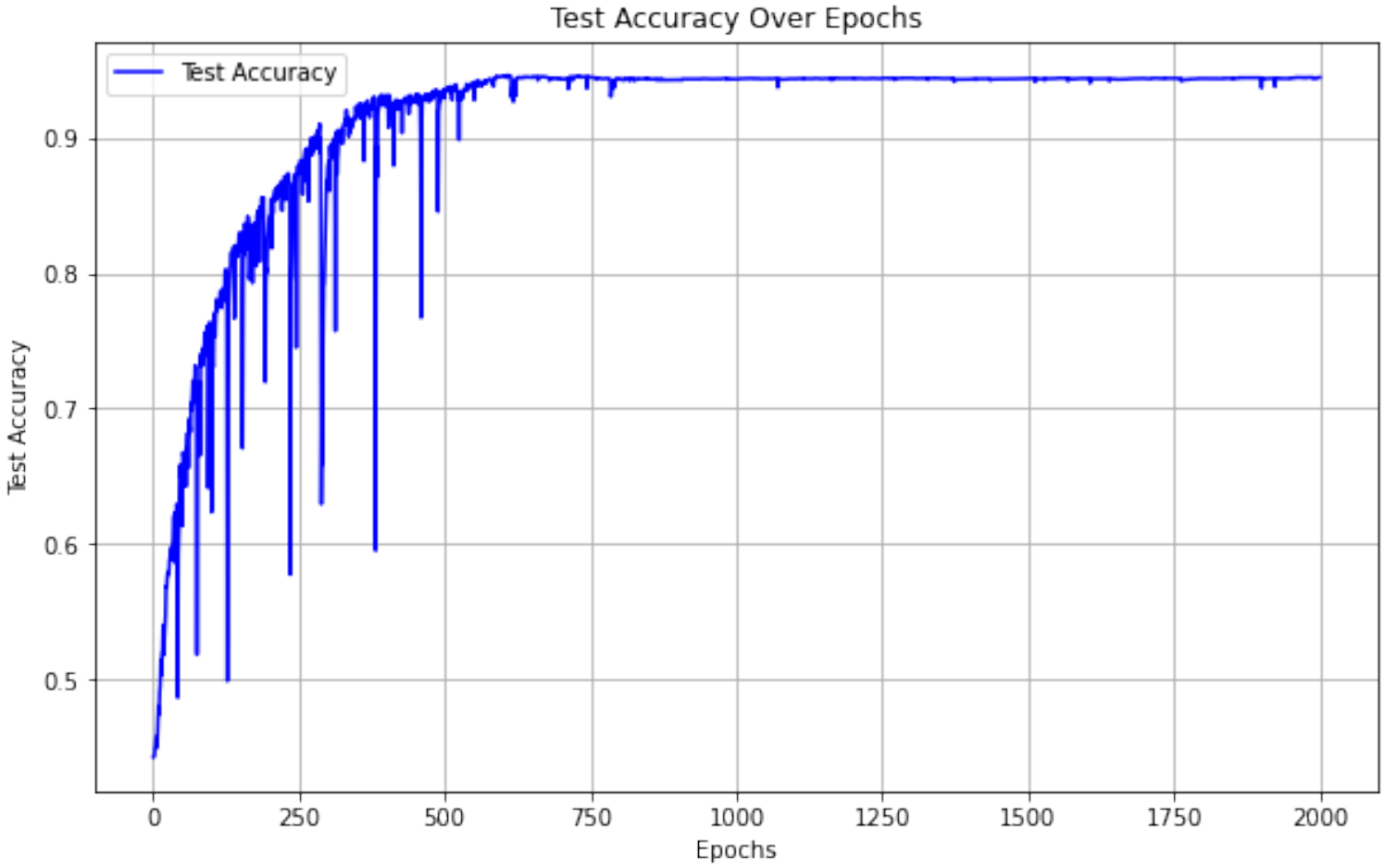

Figure 4: Plot of Test accuracy over epochs of training

2.3 Support Vector Machine

: For convenience, the data was converted to .csv format; This can be found in the "emails_to_csv.ipynb" file.
Preprocessing was done to remove:

- Remove Punctuations
- Remove Stop words
- Tokenise and Lowercase

The model was trained on the complete dataset of 35,000 emails.
Inbuilt SVM Libraries were used with the kernel as "Radial Basis Function Kernel."
The dataset was divided into 80% train and 20% test.
The model gave **93.47%** accuracy on the test dataset.

3 Inferences

- Naive bayers works poorly on the dataset
- Svm and Modified Perceptron work better; they have a good accuracy score.

4 Code files Guide:

The entire guide for the code files and how to run could be found in "Readme.txt"

Naive Bayers: It was trained on the dataset which is available as enron1 folder and tested in dataset enron2.
The weights for the model after training are stored in spam_data.json, ham_data.json.

The code for training the model can be found in "naive bayers_Preprocess_Train.ipynb"

To test the model on a data as given in the problem question as in a test folder, use the file :"naive bayers_Preprocess_Test.ipynb".
Please have all the weights as mentioned above in the same directory for the model to run and for the test data to be evaluated for the model and give results.

Perceptron : It was trained on the dataset which is available as dataset_fin folder in the directory and tested on the same.
The code for training the model can be found in "Perceptron_Train.ipynb"
The weights for this model after training are stored as "vocabulary.txt" and "perceptron_weights_2000.pkl"
To test the model on a data as given in the problem question as in a test folder, use the file :"Perceptron_Test.ipynb".
Please have all the weights as mentioned above in the same directory for the model to run and for the test data to be evaluated for the model and give results.

SVM: It was trained on the dataset which is available as dataset_fin folder in the directory and tested on the same.
The code for this could be found in "SVM_train.ipynb"