**REPORTS**

**NAÏVE BAYES**

* Implemented as naiveBayes.py.
* Log-scale used to avoid underflow.
* Words extracted using regex [A-Za-z0-9\’]+, which picks words containing uppercase letters and/or lowercase letters and/or numeric characters and/or apostrophe, and ignores punctuations and special characters.
* Input files have been treated as having latin-1 encoding.
* Output:

**\*\*\* Executing Naive Bayes with Stop Words intact in files...\*\*\***\*Calculating accuracy over HAM files in testing dataset...  
- Total number of files tested: 348  
- Number of files classified as Ham: 340  
- Number of files classified as Spam: 8  
- **Accuracy on Ham files in Training Dataset: 97.7%**

\*Calculating accuracy over SPAM files in testing dataset...  
- Total number of files tested: 130  
- Number of files classified as Ham: 20  
- Number of files classified as Spam: 110  
- **Accuracy on Spam files in Training Dataset: 84.62%**

**\*\*\* Executing Naive Bayes after removing Stop Words from files...\*\*\***  
Number of stop words removed from Ham: 117  
Number of stop words removed from Spam: 113  
\*Calculating accuracy over HAM files in testing dataset...  
- Total number of files tested: 348  
- Number of files classified as Ham: 338  
- Number of files classified as Spam: 10  
- **Accuracy on Ham files in Training Dataset: 97.13%**

\*Calculating accuracy over SPAM files in testing dataset...  
- Total number of files tested: 130  
- Number of files classified as Ham: 19  
- Number of files classified as Spam: 111  
- **Accuracy on Spam files in Training Dataset: 85.38%**

**Observation upon removing stop words**: Accuracy on Ham files decreased by 0.57%, and increased on Spam files by 0.76%. Cumulative accuracy decreased from [ (340 + 110) / 478 ] = **94.14%** MINUS [(338 + 111) / 478] = **93.93% => 0.21%**, so a marginal decrease overall. This leads to deduction that the number of stop words in the files are not too common and their conditional probabilities do not dominate classification probabilities.

**94.25%LOGISTIC REGRESSION**

* Implemented as logisticRegression.py.
* Words extracted using regex [A-Za-z0-9\’]+, which picks words containing uppercase letters and/or lowercase letters and/or numeric characters and/or apostrophe, and ignores punctuations and special characters.
* Input files have been treated as having latin-1 encoding.
* Learning rate set to 0.001 to ensure convergence, and number of iterations set to 100. Other values reduced accuracy on test dataset.
* Sample output:

**VALUES OF PARAMETERS:**

Regularization Lambda: 3

Learning Rate Alpha: 0.001

Number of Iterations: 100

**\*\*\*Executing Logistic Regression with Stop Words intact in files...\*\*\***

Training Feature Matrix constructed in 3.21 seconds

Starting training on training dataset...

Iterations: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

Training completed on training dataset, took 773.72 seconds.

Starting classification of test dataset...  
**Accuracy over Ham files: 94.83%**

**Accuracy over Spam files: 90.77%**

Classification of test dataset completed, took 1128.68 seconds.

Total time taken for Logistic Regression with Stop Words intact in files: 1905.61 seconds.

**\*\*\* Executing Logistic Regression after removing Stop Words from files...\*\*\***

Training Feature Matrix constructed in 5.76 seconds

Starting training on training dataset...

Iterations: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

Training completed on training dataset, took 702.41 seconds.

Starting classification of test dataset...  
**Accuracy over Ham files: 97.41%**

**Accuracy over Spam files: 90.77%**

Classification of test dataset completed, took 565.86 seconds.

Total time taken for Logistic Regression after removing Stop Words: 1274.03 seconds.

**Observation after removing stop words**: Accuracy on Ham files increased marginally, and stayed the same on Spam. Cumulative accuracy showed a slight increase. This leads to deduction that removal of stop words does not affect the uniqueness of documents on the basis of high probability terms.

* Outputs for different values of λ, without removing stop words and on removing stop words**:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S. No.** | **Lambda** | **Number of Iterations** | **Learning Rate** | **Stop Words Present** | | **Stop Words Removed** | |
| **Accuracy over Ham Files** | **Accuracy over Spam Files** | **Accuracy over Ham Files** | **Accuracy over Spam Files** |
| 1. | 3 | 100 | 0.001 | 94.83% | 90.77% | 97.41% | 90.77% |
| 2. | 5 | 100 | 0.001 | 94.54% | 90.77% | 97.41% | 90.0% |
| 3. | 10 | 100 | 0.001 | 94.25% | 91.54% | 97.13% | 90.0% |
| 4. | 50 | 100 | 0.001 | 95.69% | 76.15% | 94.25% | 93.08% |
| 5. | 100 | 100 | 0.001 | 97.7% | 39.23% | 89.08% | 93.85% |

End of document