COL 774: Machine Learning

Assignment 2

**BRAJ RAJ NAGAR**

**2022AIB2682**

[**aib222682@scai.iitd.ac.in**](mailto:aib222682@scai.iitd.ac.in)

1. **Text Classification**

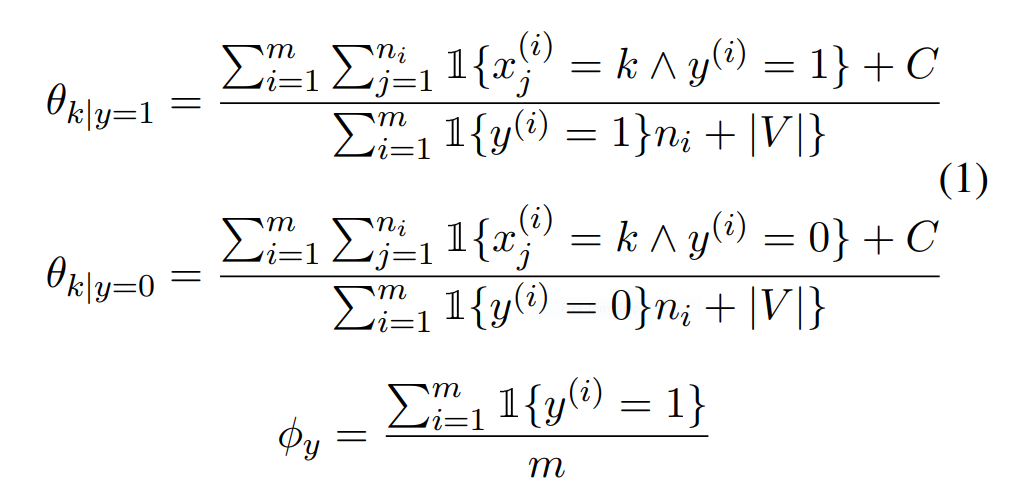
The Naive Bayes algorithm was applied in this question to classify various users' movie reviews.

The aim is to determine whether a user's review will be positive or negative given the review.

1. **Naive Bayes on unfiltered data:**

I used unfiltered data to create the model in the first section. The only pre-processing done to the evaluations is to take out punctuation and word separators.

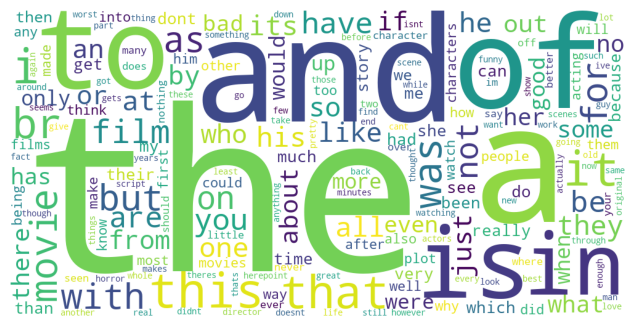
Laplace Smoothing is done by using **C = 1** to avoid any zero probabilities for the words that didn’t occur in the train set. The features *xj* of the model denote the identity of the *jth* word in the review of size *ni* where *i* is the *ith* training sample amongst *m* training samples. A feature *xj* can take any value between (1*; …..; |V|j*), where *|V|* is the size of the dictionary that we have built. There will be a parameter *θk* for all the words in the dictionary. The maximum likelihood estimates of the model parameters *θ* are as follows:



Train data accuracy on trained model: **91.068 %**

Test data accuracy on trained model: **79.58666666666666 %**

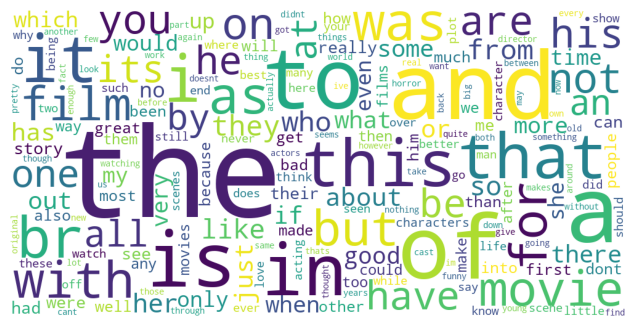
Unfiltered negative word cloud



Unfiltered positive word cloud



Unfiltered complete vocabulary word cloud



Test set accuracy obtained by randomly guessing one of the categories as the target class for each of the reviews (random prediction): **49.980000000000004 %**

Accuracy when each sample predicted as positive: **66.66666666666666 %**

My algorithm gives **29.606666666666655 %** more improvement than randomly guessing.

My algorithm gives **12.920000000000002 %** more improvement than positive prediction of each sample.

Confusion matrix for Part (a)

Confusion matrix for Part (b)

For randomly Guessing

For positive prediction

1. **And III.**

There are 10000 positive reviews and 5000 negative reviews in test data so by calculating percentage of diagonal entries we can draw observations:

Confusion matrix for Part (a) TP% = (7535/(7535+2465))\*100 = 75.35% and similarly TN% = (4403/(4403+597))\*100 = 88.06%

This means that the model is able to predict negative class more accurately compare to positive class. This also shows that the model is not very consistent to make correct predictions between the two classes.

Confusion matrix for Part (b) For randomly Guessing TP% = (5044/(5044+4956))\*100 = 50.44% and similarly TN% = (2453/(2453+2547))\*100 = 49.46% because class was randomly guessed so error in prediction very high but prediction accuracy is same for both the classes because samples were randomly guessed and there equal probability of occurring positive or negative.

For positive prediction obliviously all samples are predicted positive so accuracy for positive class is 100% and for negative class 0%. It shows this kind of model is not suitable for classification.

1. Performed stemming using nltk.stem library and removed the stop-words manually in the training as well as the test data.
2. Negative-word Cloud



Positive-word cloud



1. Test data accuracy**:84.70666666666666%**
2. Test data accuracy **without** stemming and stop word removal **: 79.58666666666666 %**

Test data accuracy **with** stemming and stop word removal: **84.70666666666666%**

1. Train data accuracy on Bi-grams model: **99.676%**

Test data accuracy on Bi-grams model: **84.1%**

Train data accuracy on modified tri-grams model: **99.992%**

Test data accuracy on modified tri-grams model: **83.1%**

No additional set of features doesn’t help as probability of occurrence of multiple word together is not significant as compare to single word occurrence probability.

Part (d) is the best performing model.

Confusion matrix for part (d)

Precision = (8797/(8797+1091)) = 0.8896

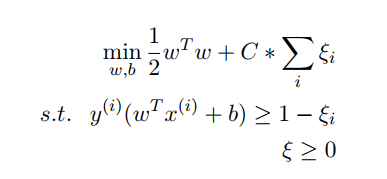
Recall = (8797/(8797+1023)) = 0.8958

F1-Score = 2\*( (0.8896\*0.8958) / (0.8896+0.8958) ) = 0.8926

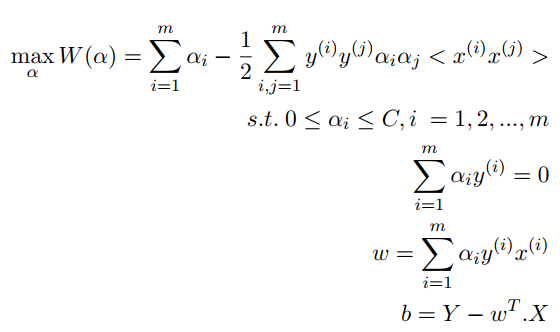
1. Part (d) matric and F1-Score is more suited for this king of database because in naïve base detection of a class is depend on key word frequency and stemming and stop-words removal improve the occurrence probability of key words.

1. **Binary Image Classification**

The soft margin formulation of the SVM is:



The Dual Formulation of the soft-margin SVM is:



1. No. of support vectors : **1180**

Percentage of training samples constitute the support vectors: **29.5%**

1. w: [[-0.39657675]

[-0.5563261 ]

[ 0.05663723]

...

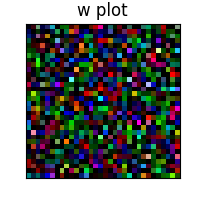
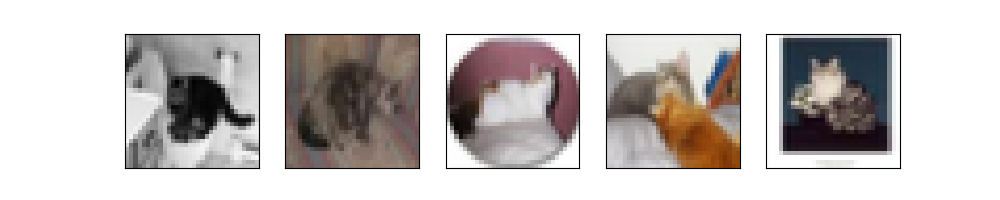
[ 0.09036578]

[ 0.29630958]

[-1.24973983]]

b: -0.1880411655766866

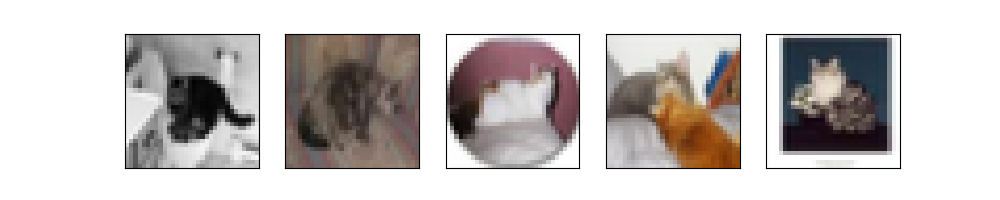
Test set accuracy: **64.0%**



Support vectors: **921**

Percentage of training samples constitute the support vectors: **23.025000000000002%**

Test set accuracy: **76.1%**



Test set accuracy in part (a): **64.0%**

Test set accuracy in this part: **76.1%**

So the Gaussian kernel gives more accurate results.

For Linear Kernel:

Support vectors in part(a) = 1180

Support vectors in this part = 2333

Matched support vectors: **1163**

For Gaussian Kernel:

Support vectors in part(b) = 921

Support vectors in this part = 3160

Matched support vectors: **816**

w: [-0.39636336 -0.55665332 0.05645169 ... 0.09092848 0.29656131

-1.24947302]

b: -0.18753543640156273

w and b obtain in part(a) and in this part are almost same.

Test set accuracy linear kernel: **64.05%**

Test set accuracy gaussian kernel: **72.65%**

Training time for CVXOPT linear kernel: **53.05776596069336sec**

Training time for CVXOPT gaussian kernel: **246.07178401947021sec**

Training time for linear kernel: **42.040398836135864sec**

Training time for gaussian kernel: **24.394362688064575sec**

**Libraries that are used in programs:**

numpy, matplotlib, wordcloud, math, os, sys, string, random, nltk(stemming), pickle, cvxopt, time, sklearn