1. Introduction

The project aims for the experimental evaluation of the linear SVM's and K nearest neighbor algorithm on data set having many irrelevant features. The task is to learn which Reuters articles are on corporate acquisitions. To cater this requirement each feature represents the frequencies of the words appear in a document. The training and validation data set has 300 samples each having 20,000 features that include only 9947 actual features and rest of all are noise.

2. Filter Methods

The filter methods that can be directly used on the dataset is

a) Pearson Correlation Coefficient
b) Signal to Noise Ratio
c) T Test

Mutual Information and Chi-Square methods can be neglected because most of the features are 0 and some other features are having considerable values which could make the methods biased towards the high parity features and it well known that these methods are good only at nominal features which is not the case here. Whereas, the above stated three methods are well suited because the data set is sparse and they try to measure how clustered the data points are so that we could rank by pushing the noise to low rank end.

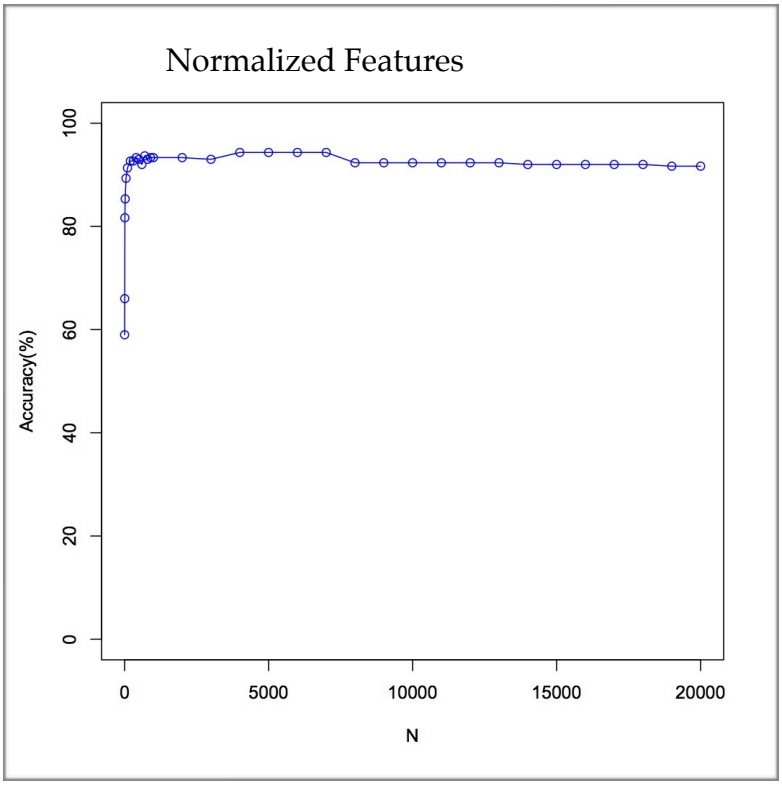
3. Filter Methods with Linear Kernel Support Vector Machines

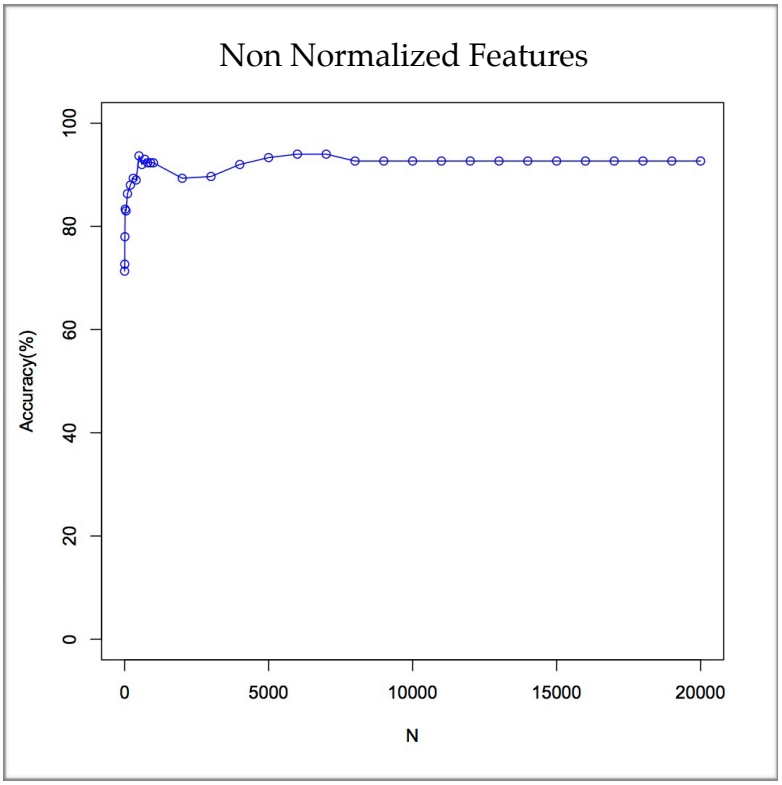
The three filter methods are implemented in java to compare and analyze which one works better on this data set. The algorithm's performance has been tested by taking multiple values for N = 1, 5, 10, 20, 50, 100, 200, 300, ..., 1000, 2000, ..., 20000 on samples of size 300 each for train and validation data .The features are ranked with Linear Kernel SVM having C=1 using SVM Light package.

3.1 Pearson Correlation Coefficient

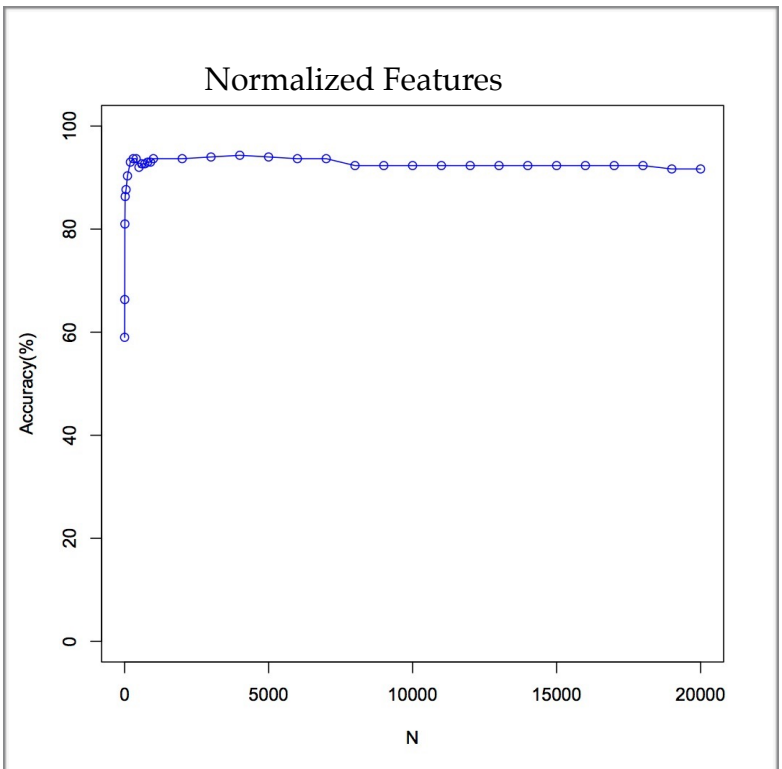
Its is observed that when the top ranked features are selected when N=1 till 10 the non normalized features did well when compared to normalized features but after N= 20 there is not a big difference between both when support vector machines are used and the same has been observed in S2Noise Ratio and T Test as shown below. The best performing N when Normalized features are used is 4000 (94.33%) and for non normalized features it is 6000 (94%) and it observed that curve stands flat between N as 10,000 and 20,000 for both normalized and non normalized features. This is because of the noise features that been added to the actual 9947 features and the average of the noised features would become 0 and this makes them stand at the low ranked end. So,

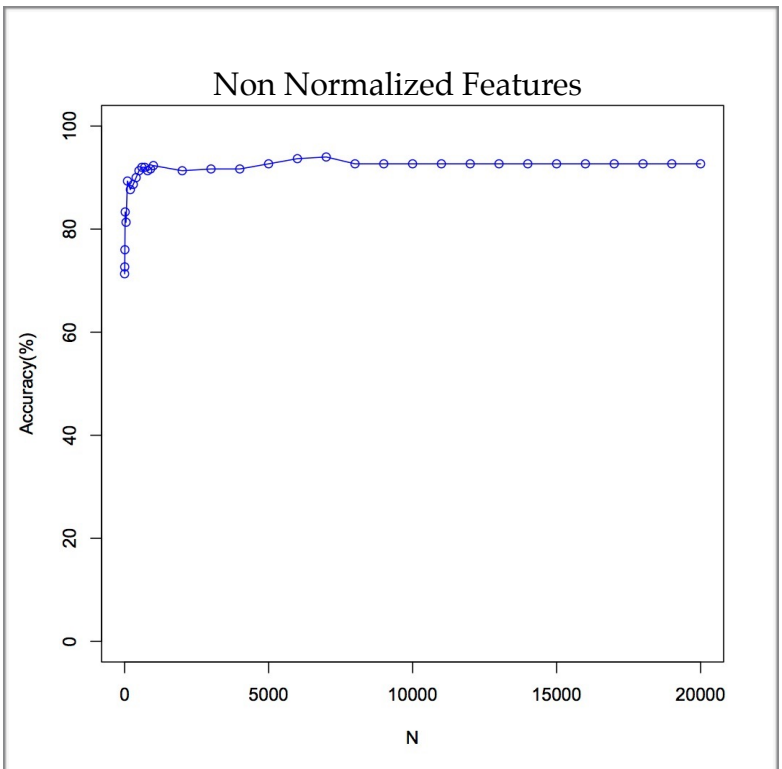
the accuracy has not been improved upon adding these noised features and the curves remains flat.





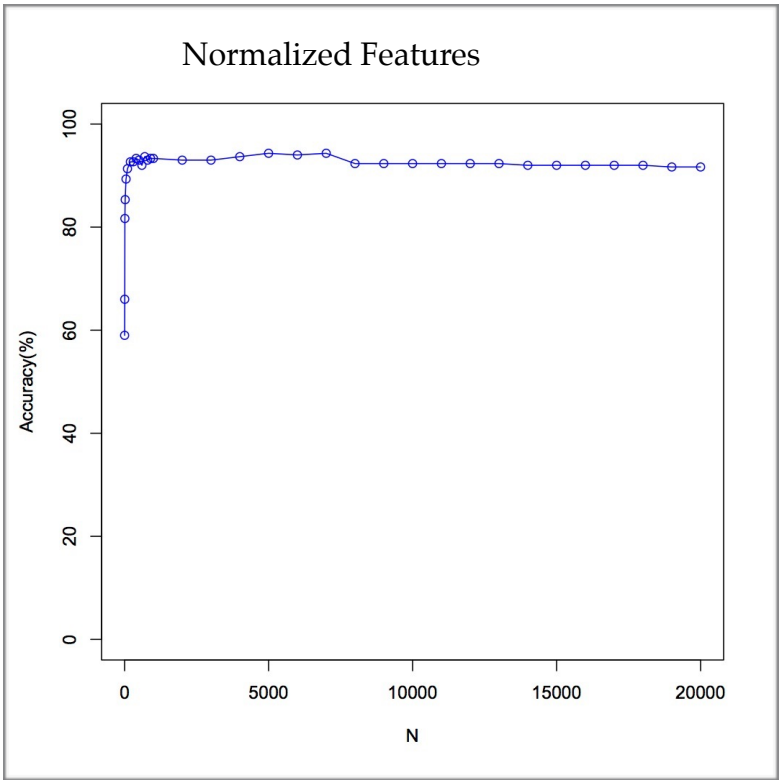
3.2 S2Noise Ratio

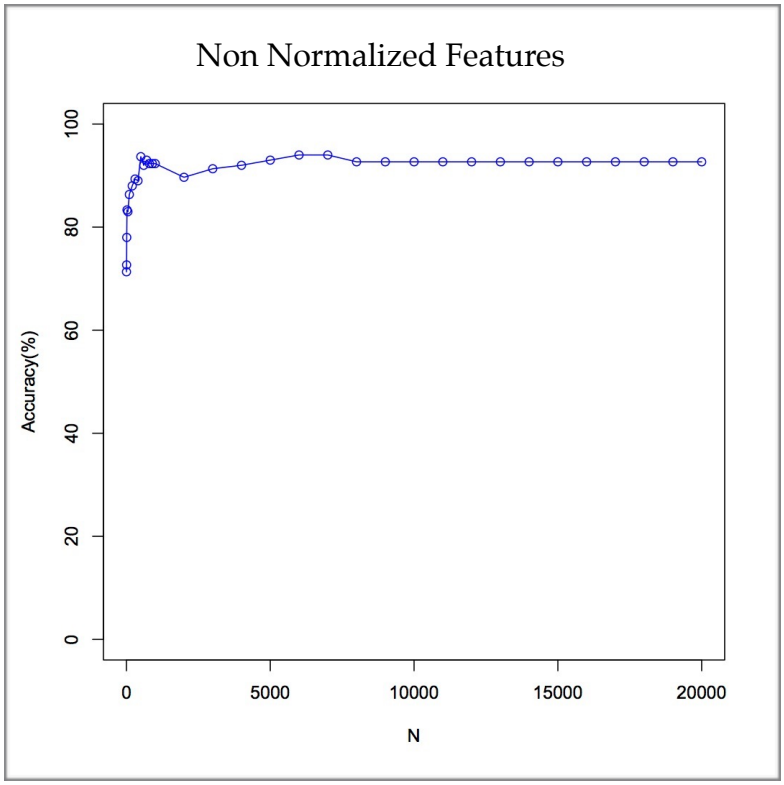




The best performing N when Normalized features are used is 4000 (94.33%) and for non normalized features is 6000 (94%) and it observed that curve stands flat for both normalized and non normalized features between 10,000 and 20,000.

3.3 T Test





The best performing N when Normalized features are used is 5000 (94.33%) and for non normalized features it is 6000 (94%) and it observed that curve stands flat for both normalized and non normalized features. Similar trends like to Pearson and S2Noise Ratio methods has been observed here.

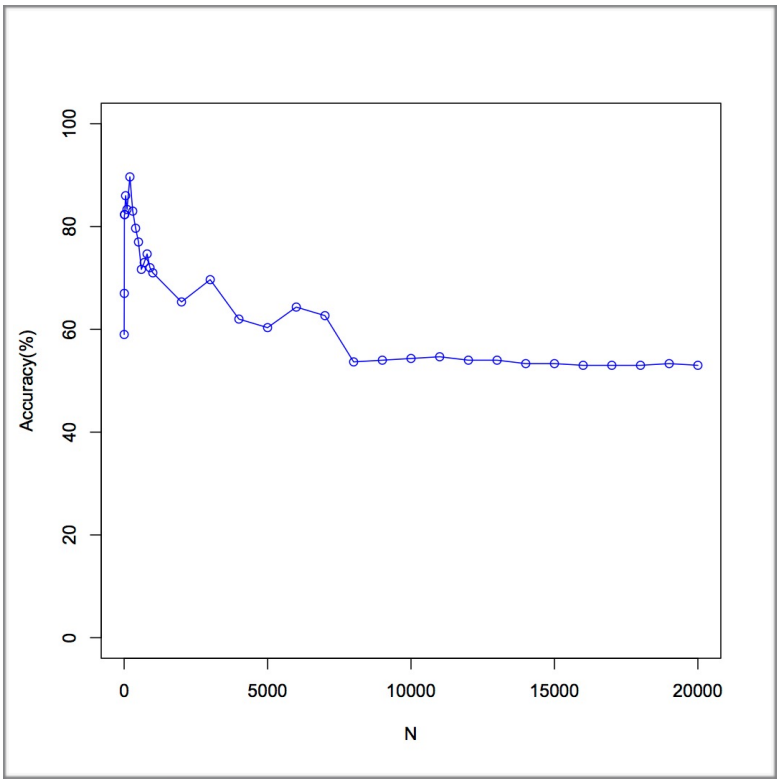
All the three filter methods performed equally good on the dexter data set in both normalized and non normalized feature cases. They yield accuracy of 94.33% with N= 4000 for Pearson and S2Noise and N=5000 for T Test on normalized features and accuracy of 94% for N=6000 in all the three methods. This clearly shows that Normalization doesn't have a much effect on dexter data set when Support vector machines are used.

4. Filter Methods with K Nearest Neighbor

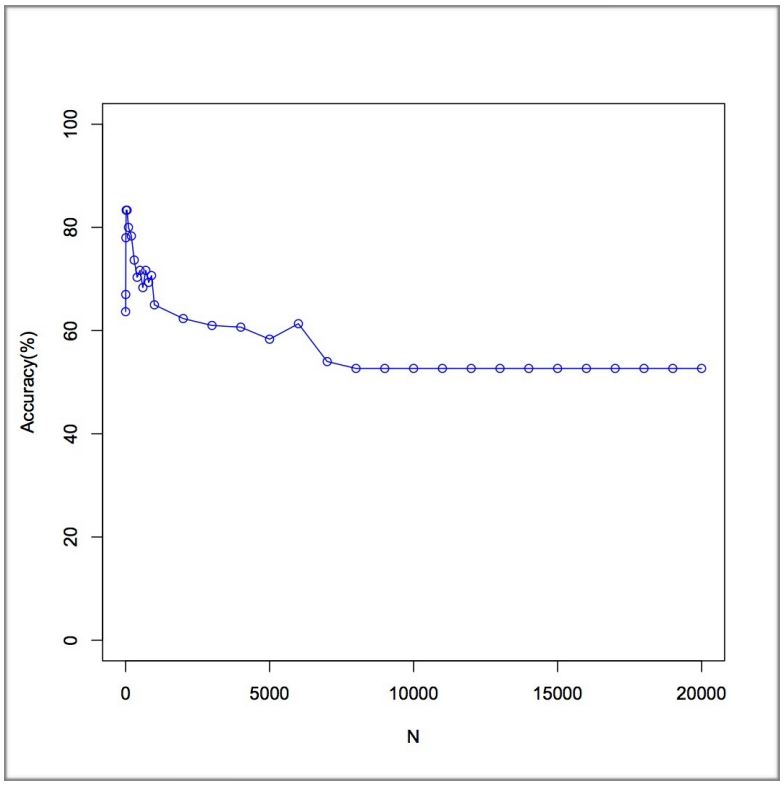
The above shortlisted three filter methods has been used to rank the features. The top N features has been selected with N = 1, 5, 10, 20, 50, 100, 200, 300, ..., 1000, 2000, ..., 20000 and tested with the validation data using K Nearest Neighbor algorithm with K as {1, 5, 10} to measure the accuracies for a given N. Below are the graphs are plotted for N values vs accuracy for the K-Filter-isNormalizedFeatures? combination.

4.1 Pearson Correlation Coefficient

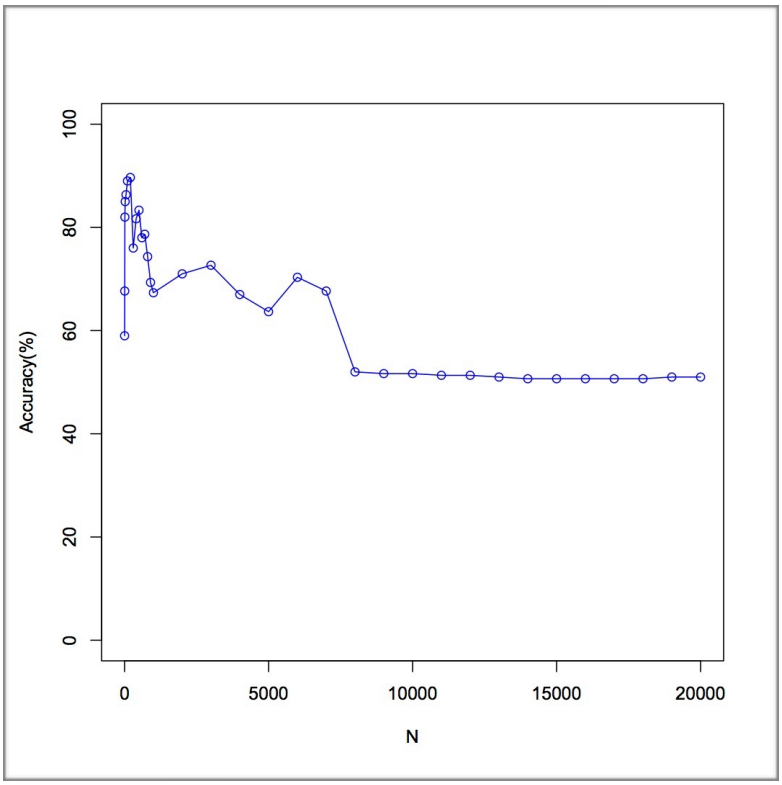
The left graph is plotted by normalizing the features and the right one is plotted using the actual features. Its observed that normalized features performed a bit well with best performed N as 200 (89.6%) when compared to non normalized features whose best is at N equals 20 (83.3%) and 50 (83.3%). Its also observed that the accuracies remain the same starting from N equals 10,000 till 20,000 for both normalized features and non normalized features. This is because the actual features are around 9947 and rest are all noise thats been added and upon ranking the noise(features) is been pushed to the low rank side as their feature averages is 0. So, When N Value has taken greater than 9947, here from 10,000 to 20,000 there is no changes in accuracies and the graph look flat as shown in the graphs below.



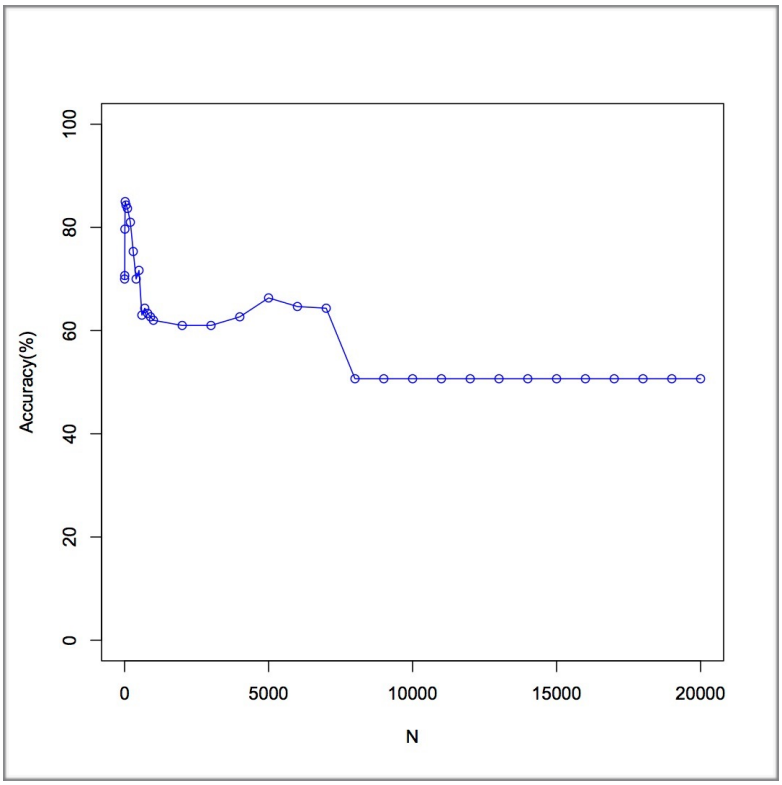
1-Pearson Correlation Coefficient -NormalizedFeatures



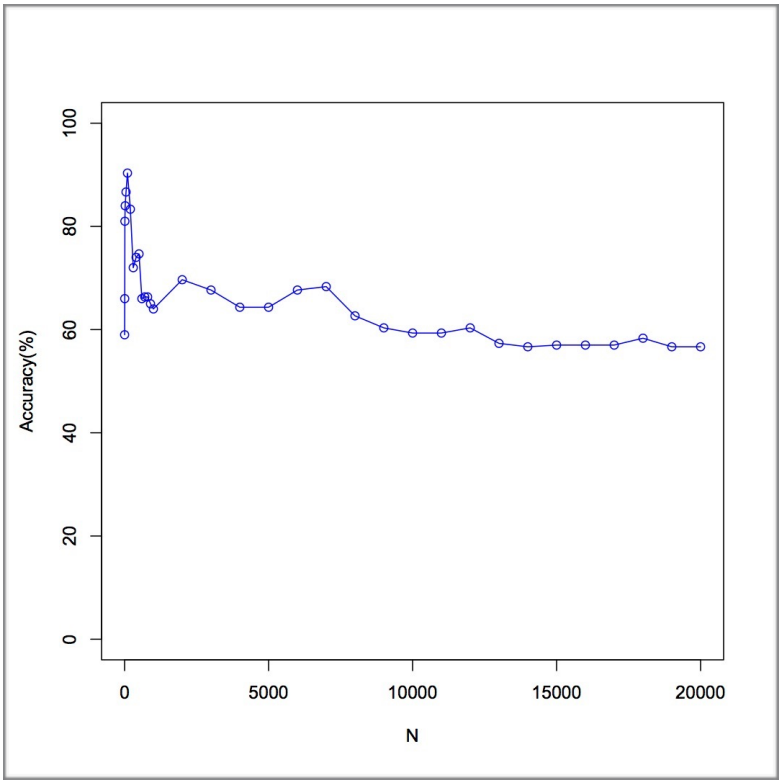
1-Pearson Correlation Coefficient -Features



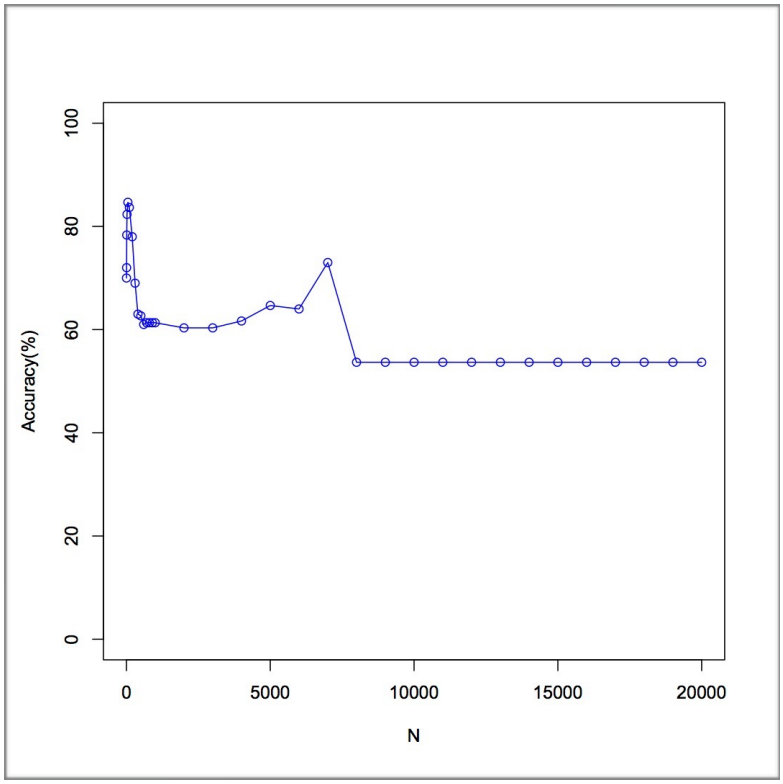
5-Pearson Correlation Coefficient -NormalizedFeatures



5-Pearson Correlation Coefficient -Features



10-Pearson Correlation Coefficient -NormalizedFeatures

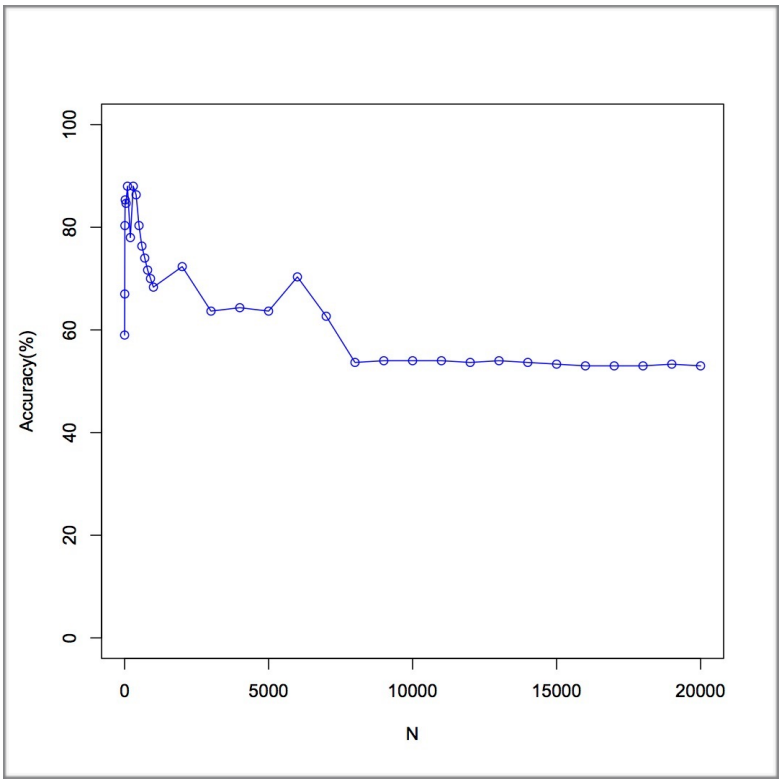


10-Pearson Correlation Coefficient -Features

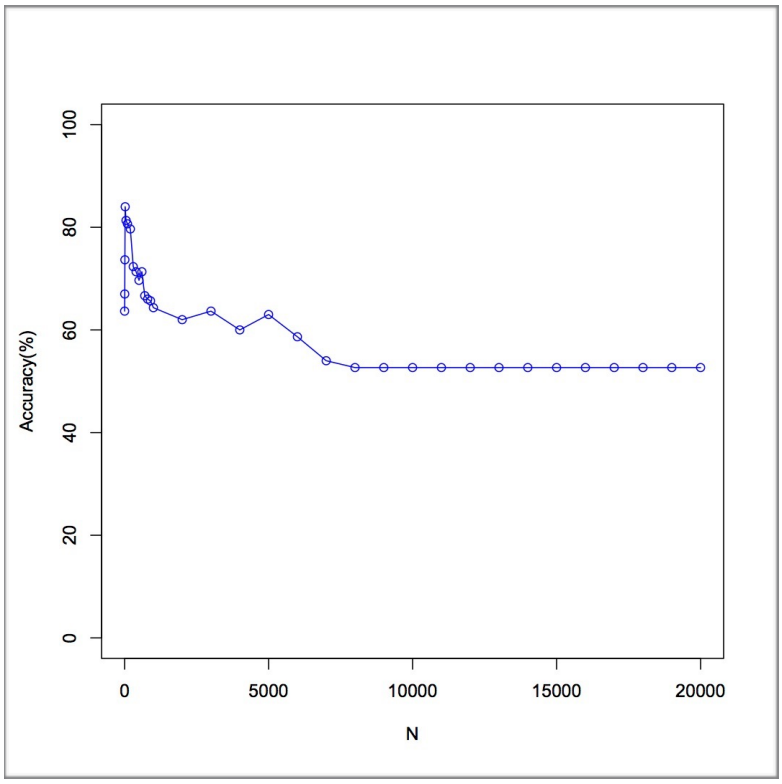
A similar trend has been observed when K values has been taken as 5 and 10. For K=5 the best performance is observed when N is 200 (89.66%) for normalized features and N is 20 (85%) for non normalized features. Similarly for K=10 the best performance is observed when N is 100(90.33%) for normalized features and N is 50 (84.66%) for non normalized features. The flat trend is observed for both K=5 and 10 when N is between 10,000 and 20,000 and the same explanation as above holds for this behavior.

4.2 S2Noise Ratio

Its is clear from the graphs below that changing the ranking scheme does not make changes to the flat trend that observed between N value 10,000 and 20,000 on dexter data set as the noised features would be pushed to the low rank end. Similar to Pearson, the normalized features did a bit well compared to non normalized features. For K=1 the best performance is observed when N is 100 and 300 (88%) for normalized features and N is 20 (84%) for non normalized features.

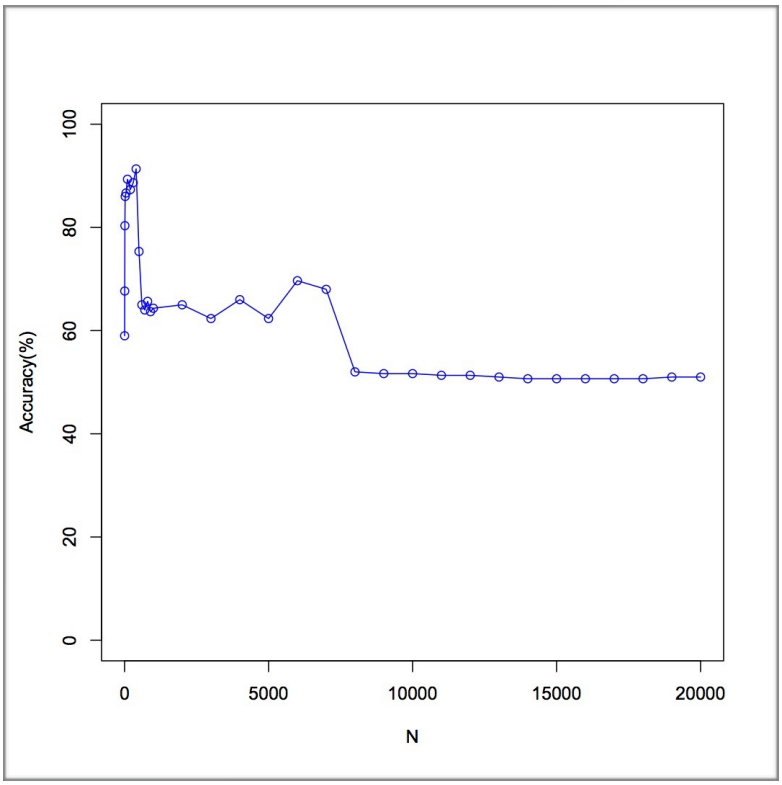


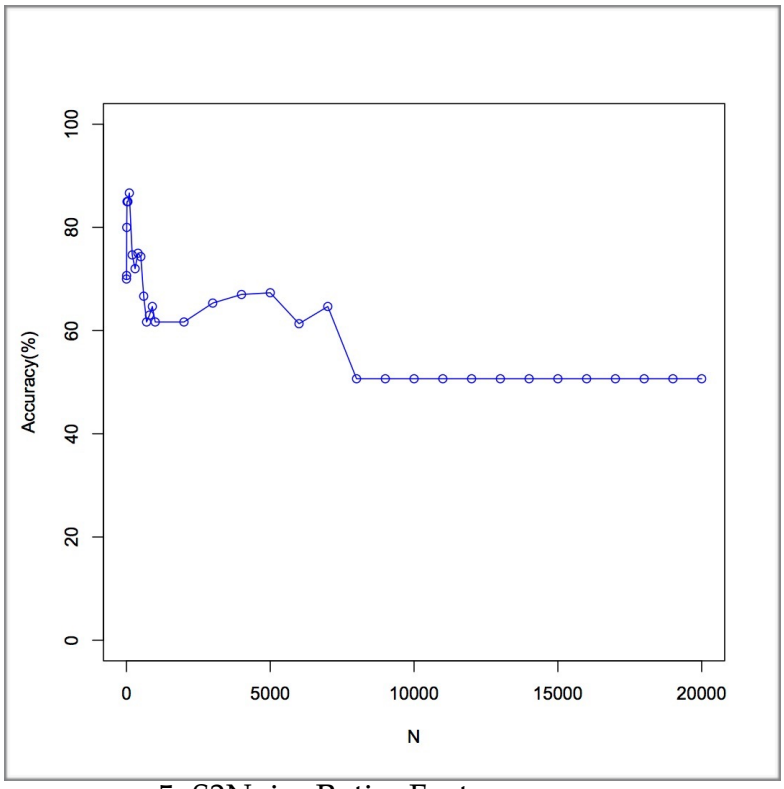
1- S2Noise Ratio - NormalizedFeatures



1- S2Noise Ratio- Features

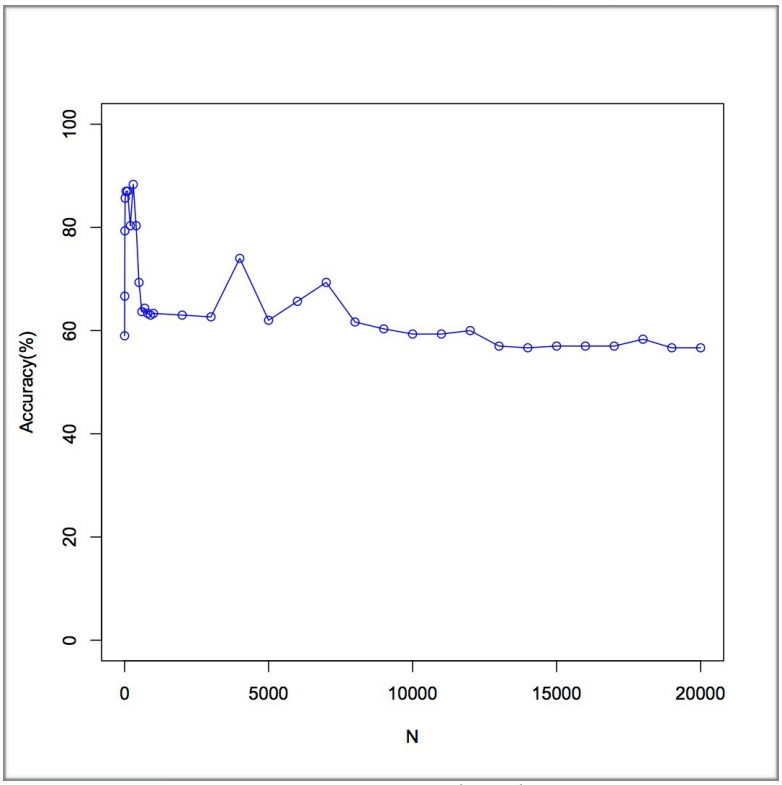
For K=5 the best performance is observed when N is 400 (91.33%) for normalized features and N is 100 (86.66%) for non normalized features.



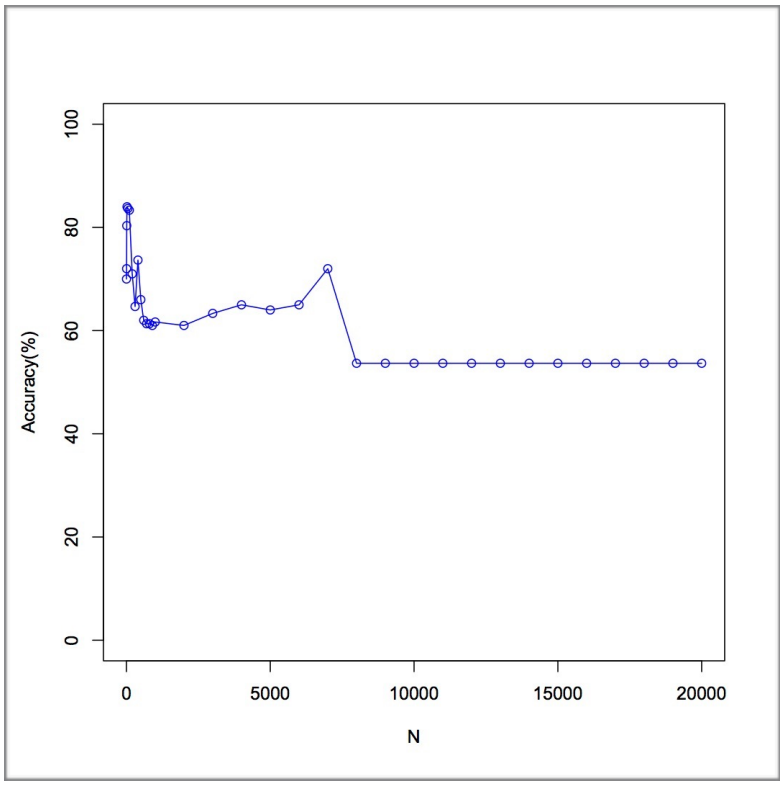


5- S2Noise Ratio- Features

For K=10 the best performance is observed when N is 50 and 100 (87%) for normalized features and N is 20(84%) for non normalized features.



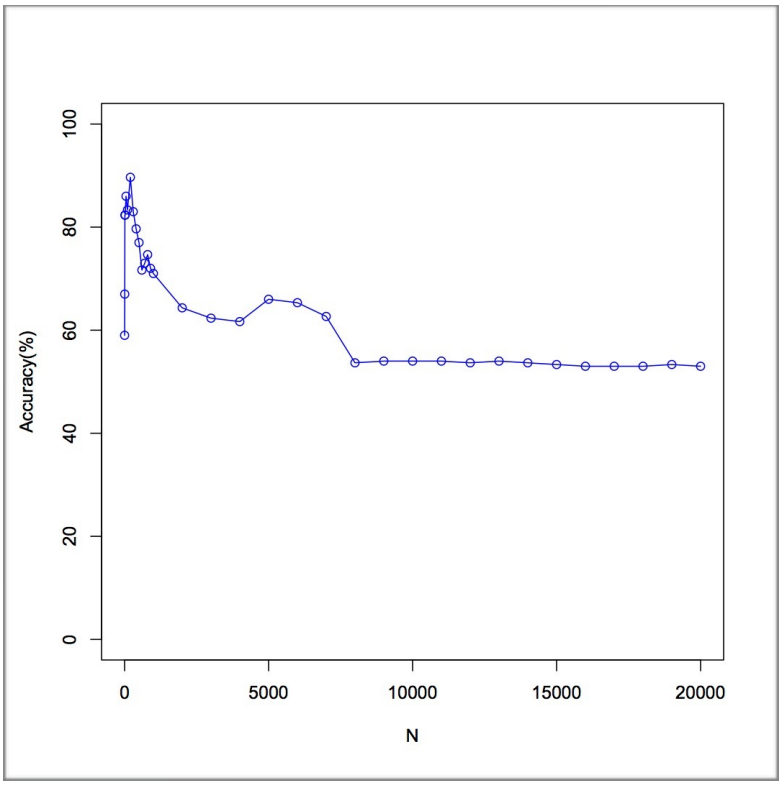
10- S2Noise Ratio - NormalizedFeatures



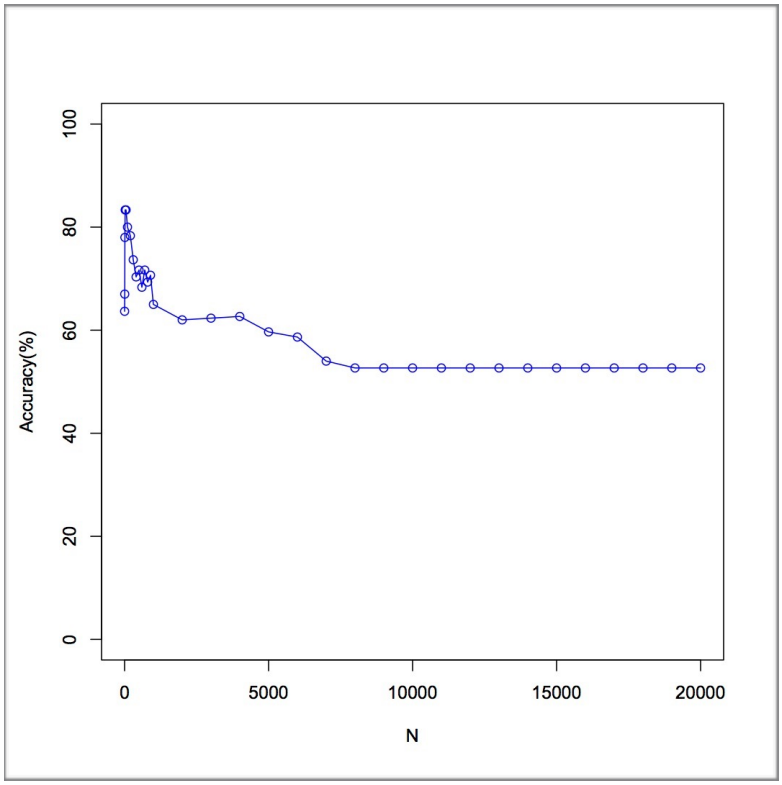
10- S2Noise Ratio- Features

4.3 T Test

T Test graphs also shows that the flat trend between N has 10,000 and 20,000 and the reason is same as explained for other filter methods. As similar to Pearson and S2Noise Ratio, T Test performed a bit well when normalized features are used than non normalized features. For K=1 the best performance is observed when N is 200 (89.66%) for normalized features and N is 20 (83.33%) and 50 (83.33%) for actual features.

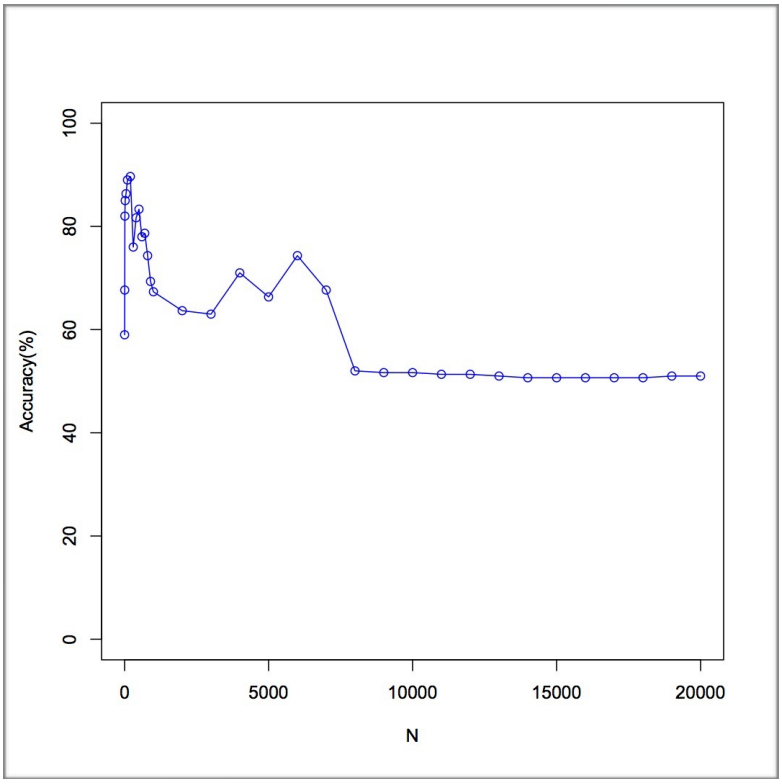


1- T Test - NormalizedFeatures

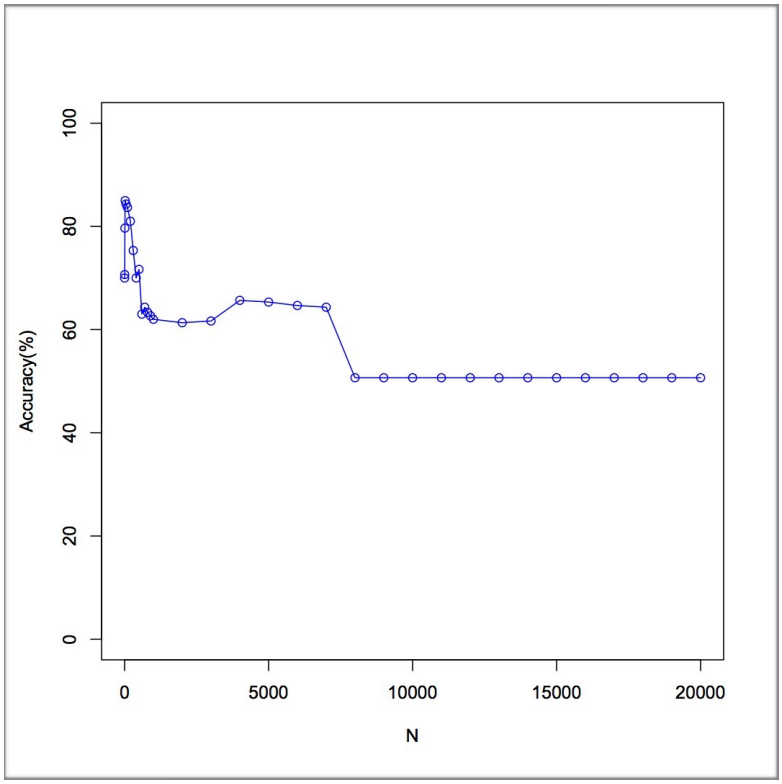


1- T Test - Features

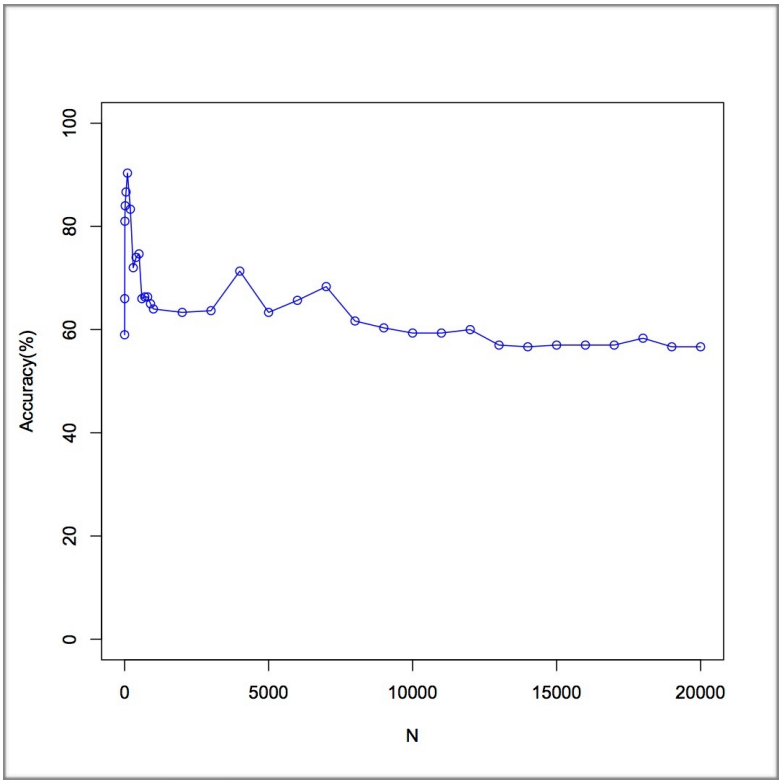
For K=5 the best performance is observed when N is 200 (89.66%) for normalized features and N is 20 (85%) for actual features. For K=10 the best performance is observed when N is 100 (90.33%) for normalized features and N is 50 (84.66%) for actual features.



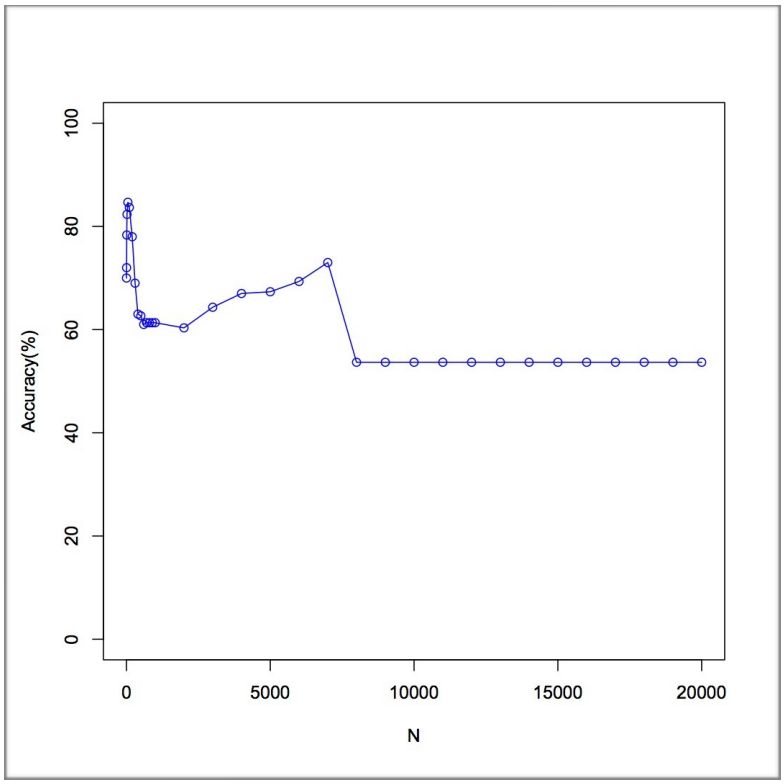
5- T Test - NormalizedFeatures



5- T Test - Features



10- T Test - NormalizedFeatures



10- T Test - Features

The best performing kNN-Filter-N for normalized features is 5 - S2Noise- 400 with a accuracy of 91.33% and for non normalized features it is 5 - S2Noise- 400 with an accuracy of 86.66%. Its does not mean that other combinations did not do well because the accuracies of those combinations are not far off from the best values.