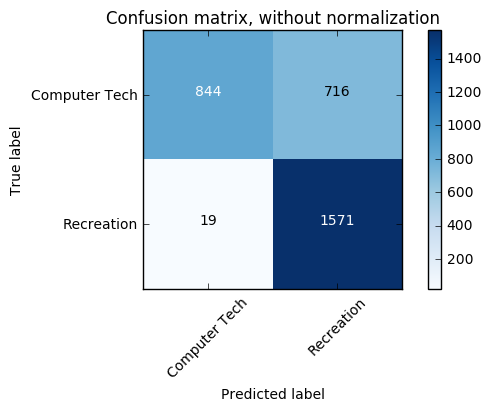
Problem(1)

In this part, similar to what we have done in Project 2, we first constructed a tokenizer to trim the text into pure words. Then we implemented TFxIDF vectors for the selected data. During this part, we also split the data into two parts, train and test data.

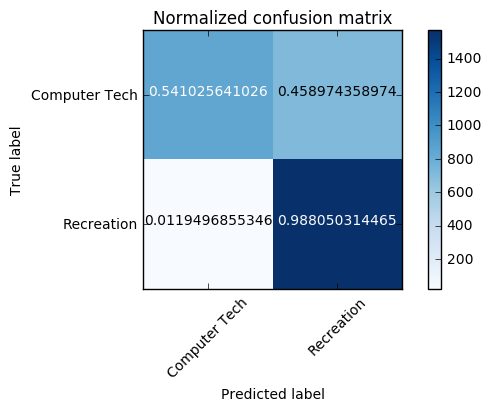
|  |  |  |
| --- | --- | --- |
| TFxIDF vector | # of documents | # of features |
| Train | 4732 | 9993 |
| Test | 3150 | 9993 |

Problem(2)

In this part, we implement K-means clustering with k = 2. First, we group subclasses into two main classes, ‘Computer Technology’ and ‘Recreation’. Similarly, we plot the confusion matrix of the result. The figure is show below.



Below is the normalized confusion matrix required.



We can easily tell that by only implementing k-means algorithms, we can’t reach to an almost diagonal matrix.

In order to make a concrete comparison of different clustering results, there are various measures of purity a given partitioning of the data points with respect to the ground truth. The measures we examine in this project are homogeneity score, completeness score, adjusted rand score and the adjusted mutual info score. Homogeneity is a measure of how purely clusters contain only data points that belong to a single class. On the other hand, a clustering result satisfies completeness if all of its clusters contain only data points that belong to a single class. Both of these scores span between 0 and 1; where 1 stands for perfect clustering. The Rand Index is similar to accuracy measure, which computes similarity between the clustering labels and ground truth labels. This method counts all pairs of points that both fall either in the same cluster and the same class or in different clusters and different classes. Finally, adjusted mutual information score measures mutual information between the cluster label distribution and the ground truth label distributions.

|  |  |
| --- | --- |
| Measures of Purity | Value |
| Homogeneity Score | 0.30711733196948748 |
| Completeness Score | 0.36252713800633013 |
| Adjusted Rand Score | 0.284250349488987 |
| Adjusted Mutual Info Score | 0.33367404362796677 |

Problem(3)

As is discussed in the Problem(2), high dimensional sparse TF-IDF vectors do not yield a good clustering performance. We have to find a better representation tailored to how the clustering algorithm works.

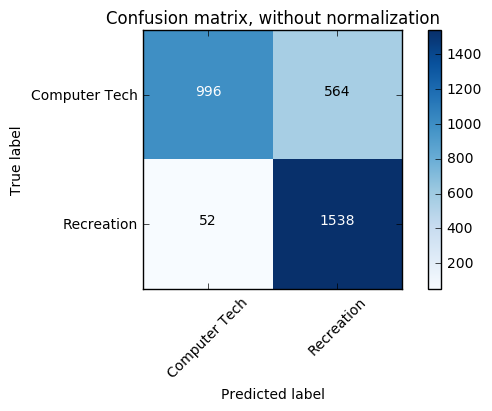
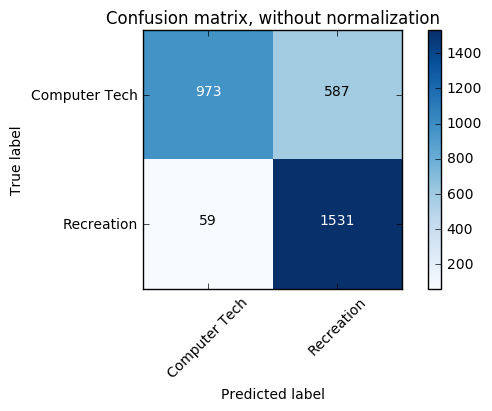
Therefore, in this part, we use Latent Semantic Indexing (LSI) and Non-negative Matrix Factorization (NMF). In order to get a good initial guess for an appropriate dimensionality to feed in the K-means algorithm, find the effective dimension of the data through inspection of the top singular values of the TF-IDF matrix and see how many of them are significant in reconstructing the matrix with the truncated SVD representation.

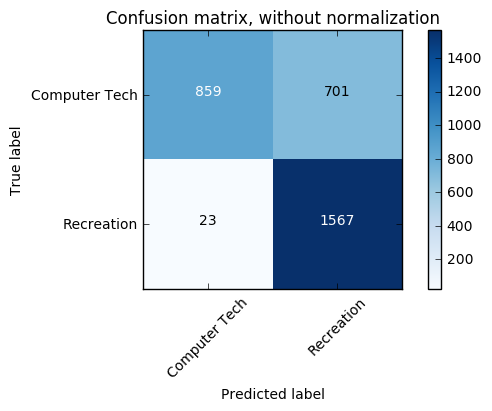
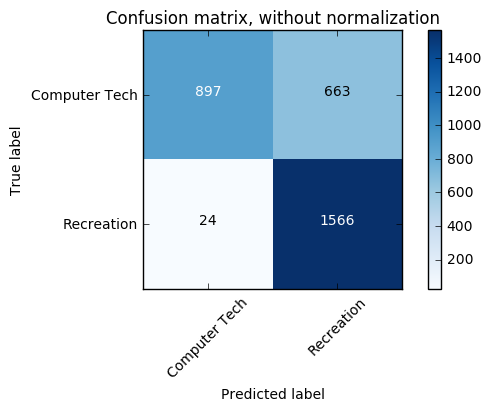
In order to make the clustering purity more satisfying, we apply normalization to the data as preprocessing.

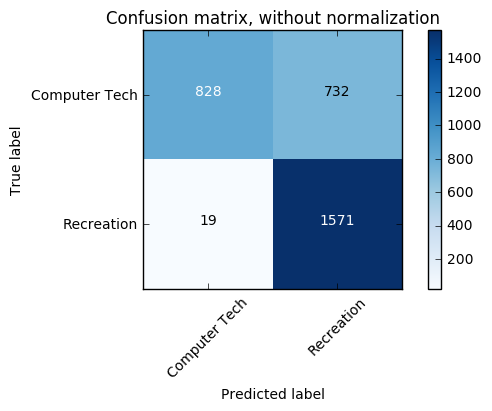
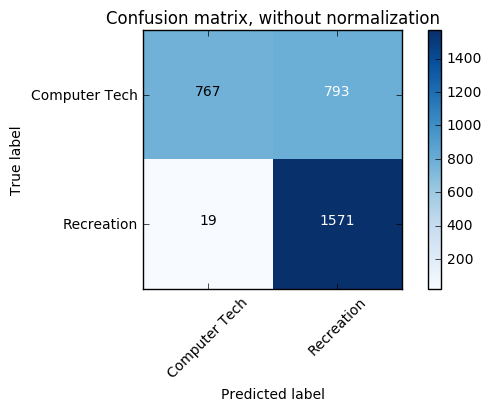
Firstly, we discuss LSI with the dimension starting from as low as 2 to 3 up to the effective dimension. The table below displays the error probability over number of components.

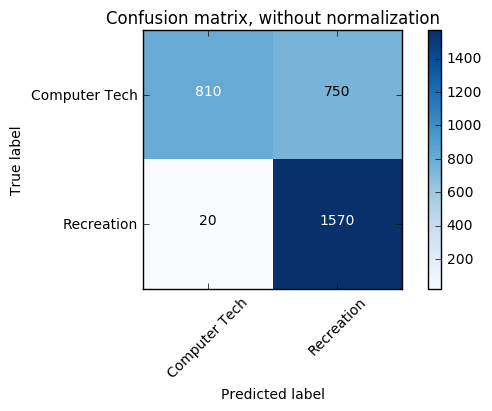
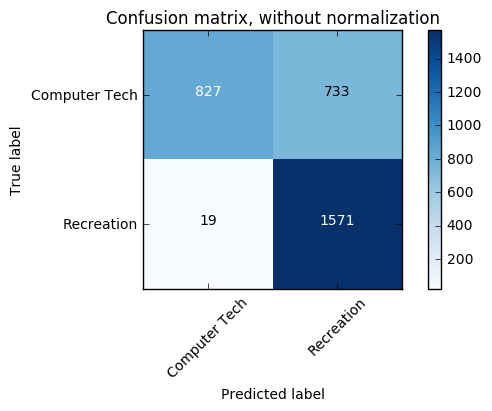
|  |  |
| --- | --- |
| N components | Error Probability (%) |
| 2 | 20.5 |
| 3 | 19.6 |
| 5 | 21.8 |
| 10 | 23.0 |
| 20 | 25.8 |
| 50 | 23.8 |
| 100 | 23.9 |
| 200 | 24.4 |

Below we plot the confusion matrices given n components under LSI.







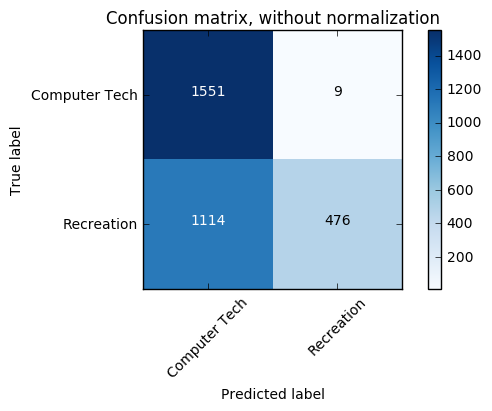
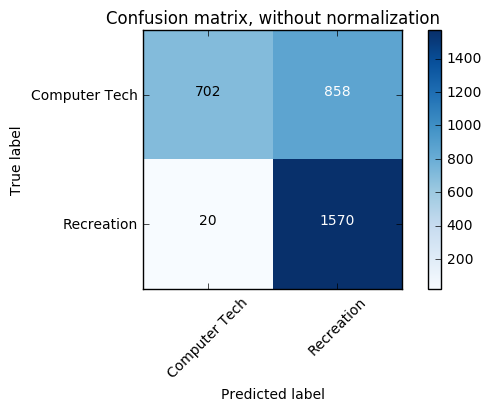


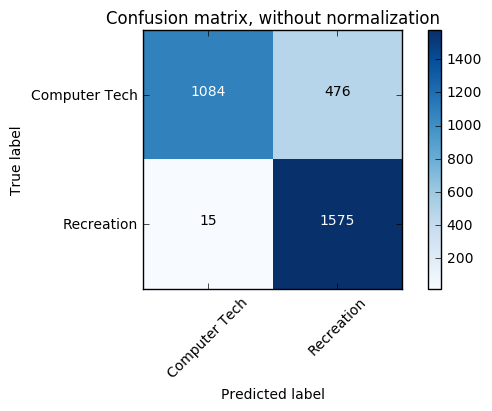
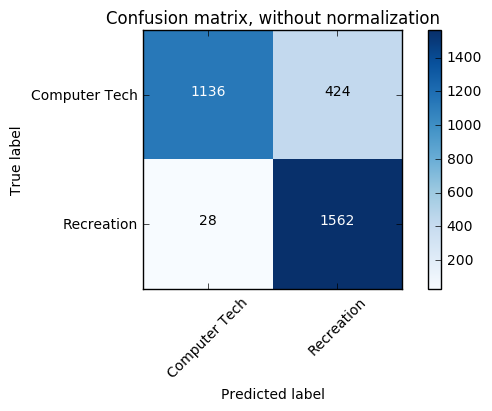
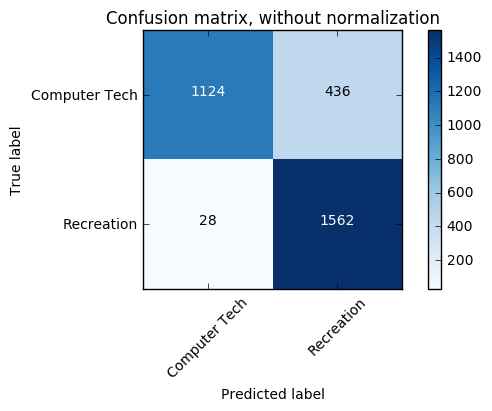
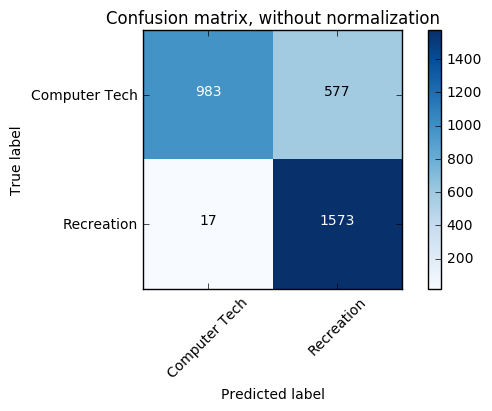
Hence, we conclude that the optimal value in Latent Semantic Indexing is 3.

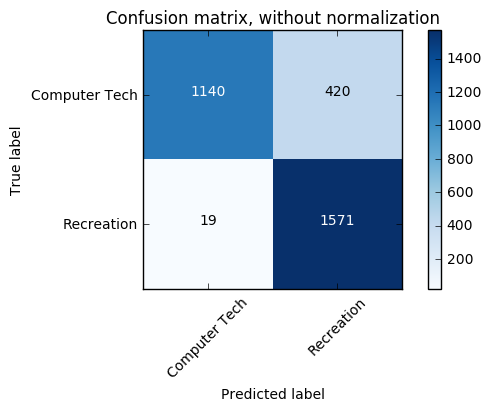
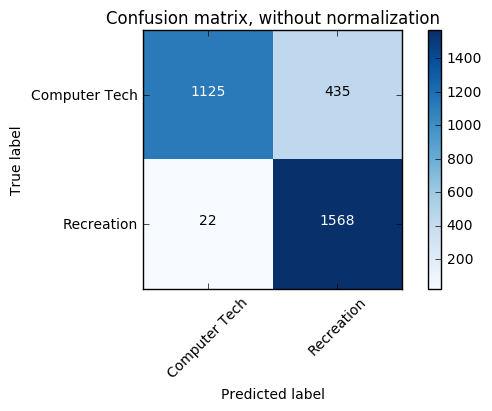
Next, we deal with the data in NMF. The table below displays the error probability over number of components.

|  |  |
| --- | --- |
| N components | Error Probability (%) |
| 20 | 27.9 |
| 50 | 35.7 |
| 100 | 18.9 |
| 200 | 14.7 |
| 300 | 14.3 |
| 400 | 15.6 |
| 500 | 14.5 |
| 600 | 13.9 |

Below we plot the confusion matrices given n components under NMF.







Hence, we conclude that the optimal value in NMF is 600.

Q: Can you justify why logarithm is a good candidate for your TFxIDF data?

A: The aspect emphasized is that the relevance of a term or a document does not increase proportionally with term frequency. Using a sub-linear function therefore helps dumped down this effect. To that extend the influence of very large or very small values (e.g. very rare words) is also amortized. Finally as most people intuitively perceive scoring functions to be somewhat additive using logarithms will make probability of different independent terms from P(A,B)=P(A)P(B) to look more like log(P(A,B))=log(P(A))+log(P(B)).

Below, we report the best final data representation we use, which is NMF in this case, with 600 components. We list the measures of purity mentioned before in the table below.

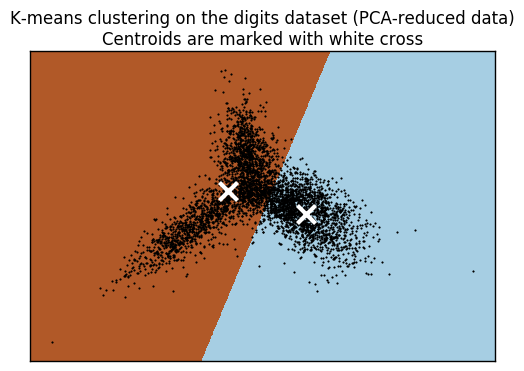
|  |  |
| --- | --- |
| Measures of Purity | Value |
| Homogeneity Score | 0.53613142177905266 |
| Completeness Score | 0.55258582267988761 |
| Adjusted Rand Score | 0.5591697286302122 |
| Adjusted Mutual Info Score | 0.53918340794056765 |

Problem(4)

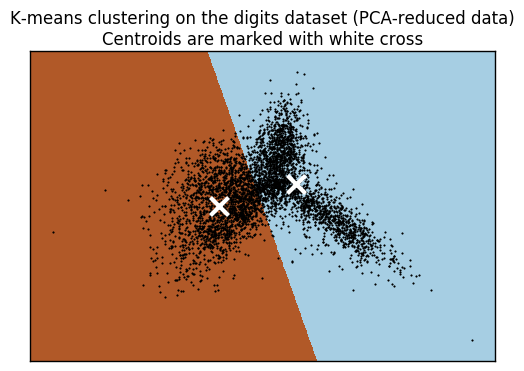
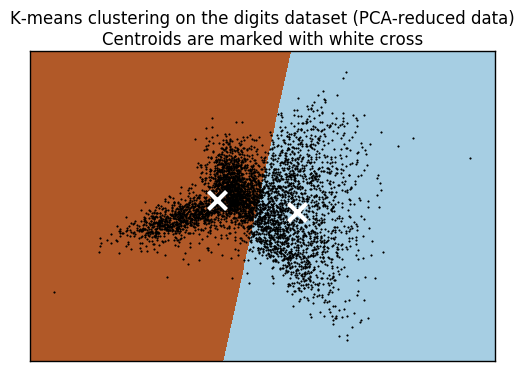
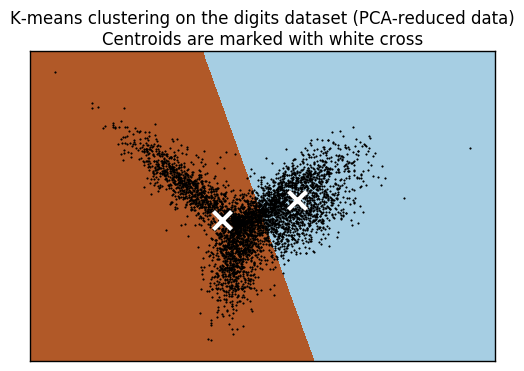
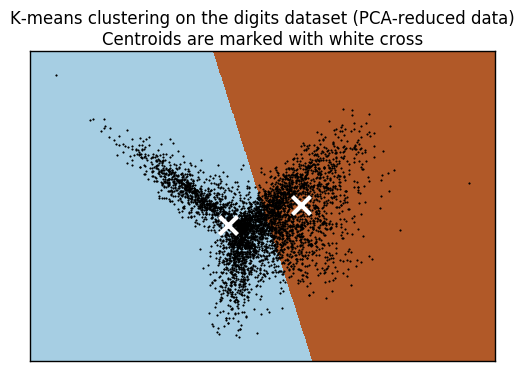
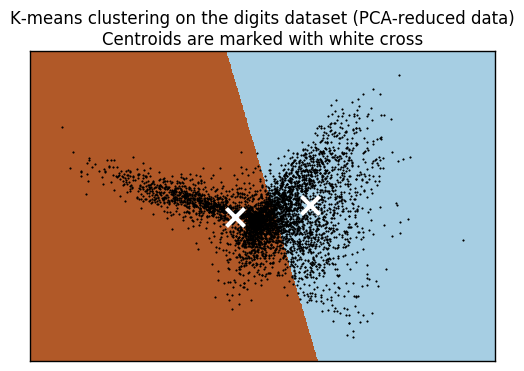
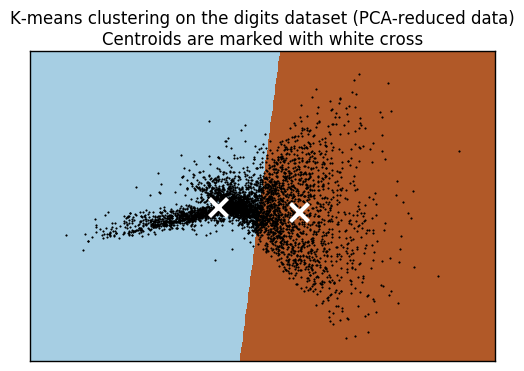
In this problem, to help understand the data more thoroughly, we visualize the performance of the clustering by projecting final data vectors onto 2 dimensions and color-coding the classes.

According to the package given from scikit-learn, we visualize the results on PCA-reduced data. Through this way, we project our data with a dimension of number of features onto two dimension.

Below we display several clustering figures given different number of components. In first case, the number of components is 600.



N = 600



N = 400

N = 300

N = 200

N = 100

N = 50

N = 20

Q: Can you justify why a non-linear transform is useful?

A: In this dataset, each features is independent with each other, hence, it indicates that there is no relationship between these features, hence, we are actually dealing with non-linear transformation. By using non-linear transformation, we can easily solve a non-linear problem as a linear (straight-line) problem.