# DATA MINING

## ASSIGNMENT 2

**C Rajmohan** rajmohan.c@csa.iisc.ernet.in

040400104113110164

---

Question 1:

### Implementation of Spectral Clustering using Ng and Jordan Algorithm

### Task:

Implement spectral clustering Ng and Jordan algorithm and check it on the given data. Partitional clustering algorithms like the K-means produce good clusters when the data has isotropic or spherical clusters, K-means algorithm is not suited when the clusters are non-isotropic; specifically when the clusters are chain-like (elongated in a direction) or concentric (where the clusters have roughly the same centroid; imagine two circles or spheres with the same center and different radii and the points in each cluster on the respective circle or sphere). Spectral clustering algorithms are well suited to deal with such data sets.

### Algorithm:

Consider the data points $S = \{s_1, s_2, \ldots..,s_n\}$ in $R^m$ that has to be clustered into k clusters.

The algorithm is as follows.

1. Form the affinity matrix $A \in R^{n \times n}$ defined by
   $A_{ij} = \exp(-||s_i - s_j||^2 / 2\sigma^2)$ if $i \neq j$
   $A_{ii} = 0$ otherwise

2. Define D to be the diagonal matrix whose (i,i)th element is the sum of A's i-th row and construct the matrix
   $L = D^{-1/2} A D^{-1/2}$

   Matrix L is called normalized Laplacian matrix.

3. Find k largest eigen vectors $x^1, x^2, \ldots x^k$ of L (chosen to be orthogonal to each other in case of repeated eigen values), and form the matrix
   $X = [x^1\ x^2\ \ldots x^k] \in R^{n \times k}$

4. Form the matrix Y from X by renormalizing each of X's rows to have unit length.
   $Y_{ij} = X_{ij} / (\sum_j X_{ij}^2)^{1/2}$

5. Treating each row of Y as a point in $R^k$, cluster them into k-clusters via k-means algorithm.
6. Finally, assign the original point $s_i$ to cluster j if and only if row i of the matrix Y was assigned to cluster j.

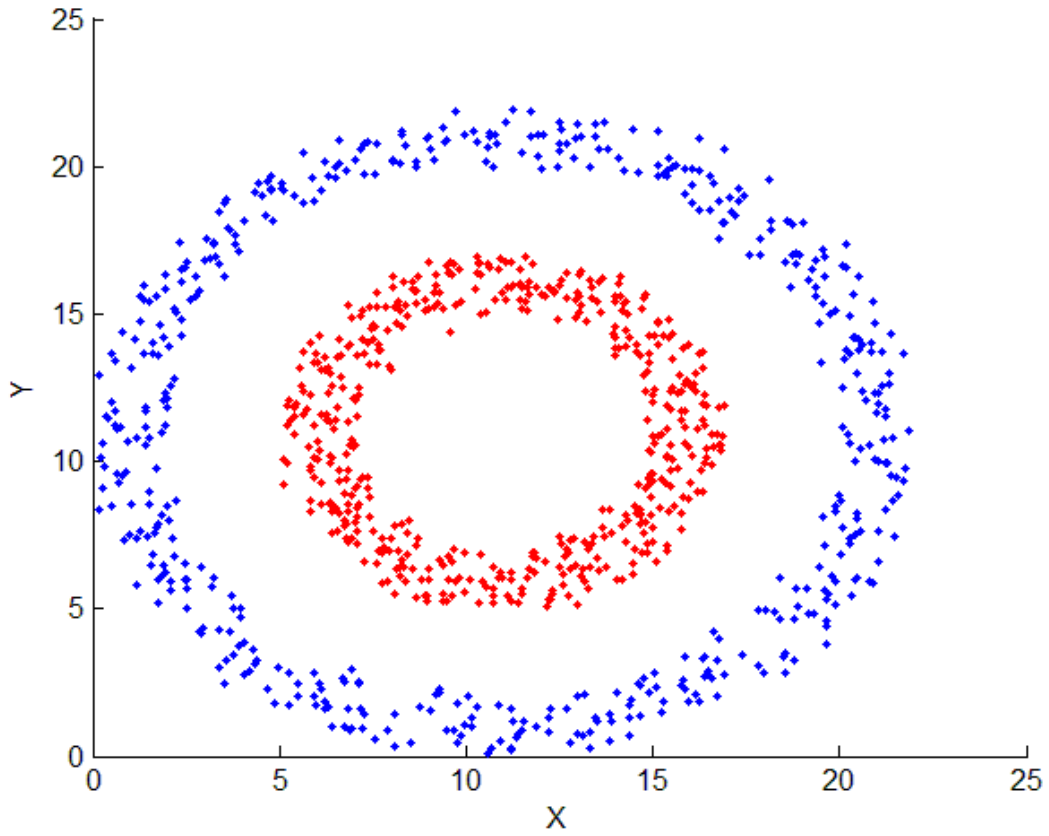
Here the scaling parameter $\sigma^2$ controls how rapidly the affinity $A_{ij}$ falls off with the distance between $s_i$ and $s_j$.

How to run:

1. Import data file to matlab using the following command
   points = importdata('filename.csv') ;
2. Run spectral_clustering.m matlab script file.
3. Check out the clustering results. If the data is 2D or 3D then a visual plot is also shown with colors to differentiate different clusters.

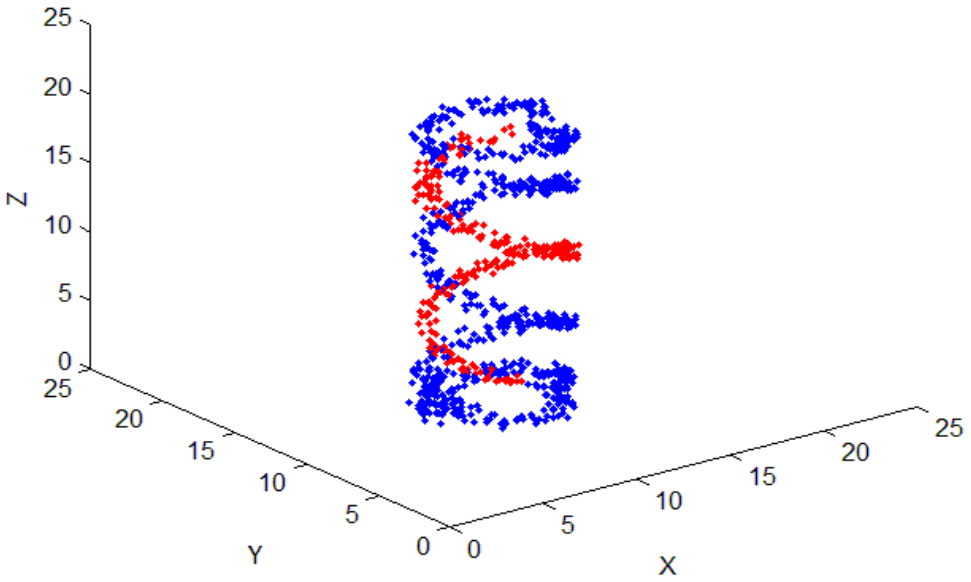
## ConcCircle

Concentric Circle: Set of 1000, two dimensional points forming concentric circles. (ConcCircle.csv)

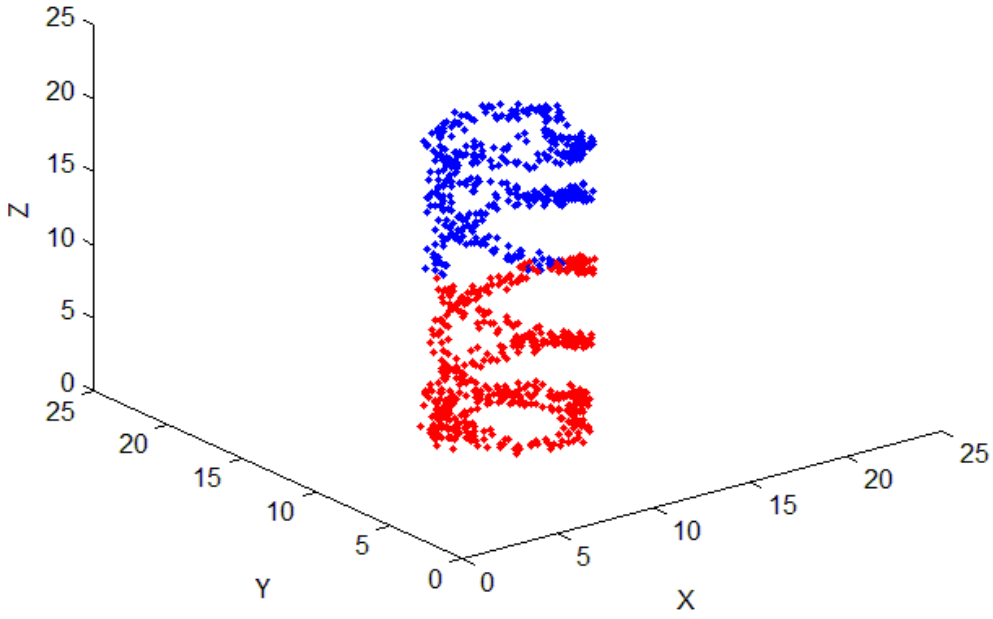


## Double Helix

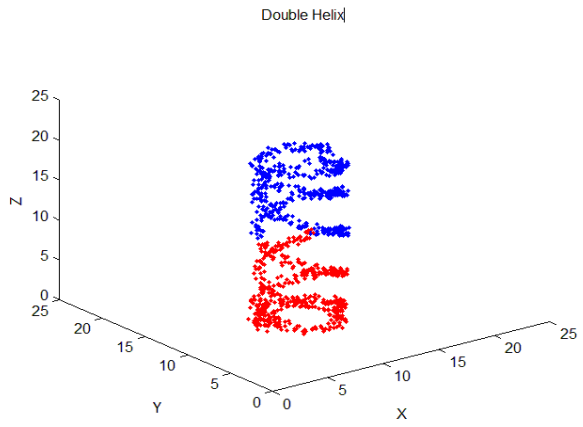
Double Helix: Set of 1000, three dimensional points forming Double Helix. (DoubleHelix.csv)



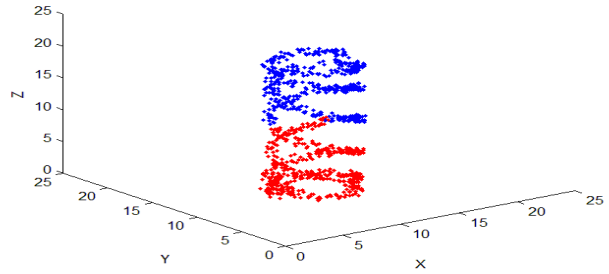

(Sigma = 0.1)




(Sigma = 0.25)




(Sigma = 0.5)




(Sigma = 0.75)

**Les Miserables**: coappearance weighted network of characters in the novel Les Miserables (LesMiserables.csv)

Given: Coappearnce weighted matrix

Construct similarity matrix(symmetric) from given coappearance matrix and apply Ng and Jordan spectral clustering algorithm.

dataset=importdata('lesMiserables.csv');

points = dataset.data;

spectral_clustering

With k=2, number of data points in cluster 1 is 44 and cluster 2 is 33.

**Political blogs**: A directed network of hyperlinks between weblogs on US politics (polBlog.csv)

Import the data into matlab

Form a similarity matrix which is symmetric.

Apply Ng and Jordan Spectral clustering algorithm to create ‘k’clusters. The clusters reflect network hyperlinks between the blogs.

With K=4 the no. of points in each clusters are 145, 112, 659 and 574

```
dataset=importdata('polBlog.csv');

H = dataset.data;

A = H+transpose(H);

spectral_cluster
```

**Tomcat Dependency:** Dependency among java class files in Apache Tomcat. (TomcatDependency.csv)

1. Import data into matlab.
2. Find similarity matrix which is symmetric by taking classes as rows and its dependency class as columns and each entry inside matrix says how much the dependency is.
3. Apply Ng and Jordan spectral clustering algorithm
4. Observe the clustering results. Classes which are having similar dependencies that is which are of same package are clustered together. It shows classes of which package are dependent on which other package classes.

**Spectral_clustering.m**

```
% ‘data’contains the data points to be clustered.
k = 2;  % number of clusters to be created
A = zeros(n,n);
D = zeros(n,n);
Norm_Laplacian = zeros(n,n);
Y = zeros(n,k);
sigma = 0.1;

% Form the affinity matrix
for i = 1: n
    for j = 1 : n
        if( i ~= j)
            sval = ((norm(data(i,:) - data(j,:)))^2);
            A(i,j) = exp(-sval/(2*(sigma*sigma)));
        else
            A(i,j) = 0;
        end
    end
end

% Find the diagonal matrix D whose (i,i)th element is the sum of A’s
i-th row
for i=1:size(A,1)
    D(i,i) = sum(A(i,:));
```

```
end

% compute the normalized laplacian matrix
% Norm_Laplacian = D^(-1/2) * L * D^(-1/2);
for i=1:size(A,1)
    for j=1:size(A,2)
        Norm_Laplacian(i,j) = (A(i,j) / (sqrt(D(i,i)) * sqrt(D(j,j))));
    end
end

% Perform the eigen value decomposition
[eig_vectors,eig_values] = eig(Norm_Laplacian);

% Select k largest eigen vectors
principal_eig_vect = eig_vectors(:,(size(eig_vectors,1)-(k-1)):
size(eig_vectors,1));

% construct the normalized matrix Y from the obtained eigen vectors
for i=1:size(principal_eig_vect,1)
    sum1 = sqrt(sum(principal_eig_vect(i,:).^2));
    Y(i,:) = principal_eig_vect(i,:) ./ sum1;
end

% perform kmeans clustering on the matrix Y creating K Clusters
[IDX,C] = kmeans(Y,k);

% Set axis dimensions for visual plot
axis([0,10,0,10]);

% plot the clustered data points
for i=1:size(IDX,1)
    if IDX(i,1) == 1
        plot(data(i,1),data(i,2),'r.');     % Cluster 1
    elseif IDX(i,1) == 2
        plot(data(i,1),data(i,2),'b.');     % Cluster 2
    end
end
```

## Question 2:

Implement Multi-nomial Naïve Bayes and Multi-variate Bernoulli classifier and apply it to newsnet 20-newsgroups dataset.

**Bernoulli document model:** a document is represented by a feature vector with binary elements taking value 1 if the corresponding word is present in the document and 0 if the word is not present.

**Multinomial document model:** a document is represented by a feature vector with integer elements whose value is the frequency of that word in the document.

When classifying a test document, the Bernoulli model uses binary occurrence information, ignoring the number of occurrences, whereas the multinomial model keeps track of multiple occurrences.

**Multinomial Naive Bayes and Bernoulli model:**

Naïve bayes classifier estimates class conditional probability by assuming that the attributes are conditionally independent, given the class label 'c'. This can be stated as

$$P(D / C = c) = \prod_{i=1}^{l} P(Wi/C = c)$$

Where each attribute D = {d1,d2,…dn} consists of l attributes.

In our case, each row of document term matrix represents a document and attributes of the documents are frequency of words occurring in each document which are represented as columns. We first need to compute conditional probability of each word in documents of a given training data class label c.

To classify a test record then we can compute posterior probabilities for each class C. Then we need to choose the class that maximizes this probability.

$$P(C / D) = [P(C) \prod_{i=1}^{l} P(Wi/C = c) ] \div P (D)$$

Since P(D) is fixed for every document, it is sufficient to choose the class that maximizes the numerator term $P(C) \prod_{i=1}^{l} P(Wi/C = c)$.

In bernoulli model, instead of considering the frequency of occurrence of words, consider the occurrence of words as binary information. This model also considers the probabilities of words which are not present in a document while classifying the document.

**20 newsgroups dataset:**

The 20 Newsgroups data set is a collection of approximately 20,000 newsgroup documents, partitioned nearly evenly across 20 different newsgroups each corresponding to a different topic. Some of the newsgroups are very closely related to each other while other are not.

**Implementation:**

1. Import the data files into matlab which includes vocabulary list, training labels, test labels, and mainly training data and test data.

2. Let alpha = 1/|vocabulary|. Here alpha is used as normalization value to avoid zero probabilities appearing in calculations which could make the entire result zero. Hence this normalization term is added with all probabilities of word frequencies in documents.

3. Calculate probability of each class label (p(C)) using training data and its labels.

4. Calculate the conditional probabilities of each word given class label. Note that naïve bayes classifier considers frequency of words present in each training document while calculating class conditional probabilities whereas Bernoulli model considers only occurrence or non-occurrence of words.

5. Now consider each test document and find the probabilities of the document belonging to each class. Note that this step is implemented differently for naïve bayes classifier and Bernoulli classifier. Bernoulli classfier considers absence of words as well.

6. Classify the document‘d’ as belonging to some cluster $C_i$ if $p(C_i/d)$ is greater than $p(C_j/d)$ for all j: 1 to 20 and $j \neq i$. Do the same for each test document.

7. Produce a confusion matrix showing how the algorithm has classified the documents by showing what the actual class are and predicted class are for each document.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | 249 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 3 | 3 | 24 | 2 | 3 | 4 | 26 |
| **2** | 0 | 286 | 13 | 14 | 9 | 22 | 4 | 1 | 1 | 0 | 1 | 11 | 8 | 6 | 10 | 1 | 2 | 0 | 0 | 0 |
| **3** | 1 | 33 | 204 | 57 | 19 | 21 | 4 | 2 | 3 | 0 | 0 | 12 | 5 | 10 | 8 | 3 | 1 | 0 | 5 | 3 |
| **4** | 0 | 11 | 30 | 277 | 20 | 1 | 10 | 2 | 1 | 0 | 1 | 4 | 32 | 1 | 2 | 0 | 0 | 0 | 0 | 0 |
| **5** | 0 | 17 | 13 | 30 | 269 | 0 | 12 | 2 | 2 | 0 | 0 | 3 | 21 | 8 | 4 | 0 | 1 | 0 | 1 | 0 |
| **6** | 0 | 54 | 16 | 6 | 3 | 285 | 1 | 1 | 3 | 0 | 0 | 5 | 3 | 6 | 4 | 0 | 1 | 1 | 1 | 0 |
| **7** | 0 | 7 | 5 | 32 | 16 | 1 | 270 | 17 | 8 | 1 | 2 | 0 | 7 | 4 | 6 | 0 | 2 | 1 | 2 | 1 |
| **8** | 0 | 3 | 1 | 2 | 0 | 0 | 14 | 331 | 17 | 0 | 0 | 1 | 13 | 0 | 4 | 2 | 0 | 0 | 6 | 1 |
| **9** | 0 | 1 | 0 | 1 | 0 | 0 | 2 | 27 | 360 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| **10** | 0 | 0 | 0 | 1 | 1 | 0 | 2 | 1 | 2 | 352 | 17 | 0 | 1 | 3 | 3 | 5 | 2 | 1 | 5 | 1 |
| **11** | 2 | 0 | 1 | 0 | 0 | 0 | 2 | 1 | 2 | 4 | 383 | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 1 | 0 |
| **12** | 0 | 3 | 0 | 3 | 4 | 1 | 0 | 0 | 0 | 1 | 1 | 362 | 2 | 2 | 2 | 0 | 9 | 0 | 5 | 0 |
| **13** | 3 | 20 | 4 | 25 | 7 | 4 | 8 | 11 | 6 | 0 | 0 | 21 | 264 | 9 | 7 | 1 | 3 | 0 | 0 | 0 |
| **14** | 5 | 7 | 0 | 3 | 0 | 0 | 3 | 5 | 4 | 1 | 0 | 1 | 8 | 320 | 8 | 7 | 6 | 5 | 8 | 2 |
| **15** | 0 | 8 | 0 | 1 | 0 | 3 | 1 | 0 | 1 | 0 | 1 | 4 | 6 | 5 | 343 | 3 | 2 | 1 | 12 | 1 |
| **16** | 11 | 2 | 0 | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 362 | 0 | 1 | 2 | 15 |
| **17** | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 1 | 1 | 0 | 4 | 0 | 5 | 2 | 1 | 303 | 5 | 23 | 13 |
| **18** | 12 | 1 | 0 | 1 | 0 | 0 | 1 | 2 | 0 | 2 | 0 | 2 | 1 | 0 | 0 | 6 | 3 | 326 | 18 | 1 |
| **19** | 6 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 10 | 6 | 2 | 63 | 6 | 196 | 13 |
| **20** | 39 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 2 | 6 | 27 | 10 | 3 | 7 | 151 |

## (Confusion Matrix of Naïve Bayes Classification)

Classes are,

1. alt.atheism
2. comp.graphics
3. comp.os.ms-windows.misc
4. comp.sys.ibm.pc.hardware
5. comp.sys.mac.hardware
6. comp.windows.x
7. misc.forsale
8. rec.autos
9. rec.motorcycles
10. rec.sport.baseball
11. rec.sport.hockey
12. sci.crypt
13. sci.electronics
14. sci.med
15. sci.space
16. soc.religion.christian
17. talk.politics.guns
18. talk.politics.mideast
19. talk.politics.misc
20. talk.religion.misc

(1st column represents original class and first row indicate predicted class)

Naïve bayes classification of all the 20 newsgroups produces an overall performance of 78.52% correct classification with 1/|vocabulary| as alpha. If we use 0.0667 as alpha we can get as much as 81% overall accuracy but this may not be genaralized for any test set.

Note that as few classes are closely related, they affect the classification accuracy. For example comp.graphics, comp.windows.x and 3. comp.os.ms-windows.misc are closely related. If we consider only those documents which are not from closely related classes, then naïve bayes classifier produces upto 98% accuracy.

**naivebayes.m**

```
% Read vocabulary and training, test labels data from files into matlab
newsgroup_names=textread('data/newsgrouplabels.txt','%s');
words=textread('data/vocabulary.txt','%s');
labels_train=textread('data/train.label','%d');
labels_test=textread('data/test.label','%d');

% read training data
[docid wordid counts]=textread('data/train.data','%d %d %d');
data_train=sparse(docid, wordid, counts, numel(labels_train),
numel(words));

% read test data
[docid wordid counts]=textread('data/test.data','%d %d %d');
data_test=sparse(docid, wordid, counts, numel(labels_test), numel(words));

clearvars docid;clearvars wordid;clearvars counts;

alph=1/nume1(words);

% initialize the matrices
p_W_giv_C=zeros(numel(newsgroup_names),size(data_train,2))+alph;
pC=zeros(numel(newsgroup_names),1);

% find class probabilities for(i=1:numel(newsgroup_names))
    pC(i)=sum(labels_train==i);
end
pC=pC./sum(pC);

% find conditional probabilities of each word given class
for(i=1:numel(newsgroup_names))
p_W_giv_C(i,:)= p_W_giv_C(i,:)+sum(data_train(find(labels_train==i),:),1);
p_W_giv_C(i,:)= p_W_giv_C(i,:)./sum(p_W_giv_C(i,:));
end

doc_prob_mat=log(p_W_giv_C)*data_test'+repmat(log(pC),1,size(data_test,1));

[~,classes]=max(doc_prob_mat);
```

```
% find confusion matrix
C=confusionmat(labels_test,classes,'order',1:20);
%print overall performance
crct = sum(diag(C));
accuracy = crct/sum(sum(C));
```

**Bernoulli model:**

Bernoulli model classification of all the 20 newsgroups test documents produces an overall performance of 73.36% correct classification with $1/10^5$ as alpha. If we use 0.5 as alpha we get almost 64% overall accuracy.

Note again that as few classes are closely related, they affect the classification accuracy. If we consider only those documents which are not from closely related classes, then Bernoulli model classifiers also produce very high accuracy classification similar to naïve bayes classifier.

Bernoulli model works well for short documents but for long documents considering frequency of words is better hence mutinomial naïve bayes works well. In Bernoulli model non-occurring words affect the probabilities whereas in naïve bayes model it is not the case.

## Confusion matrix for Bernoulli:

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | 238 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 2 | 4 | 31 | 2 | 8 | 7 | 22 |
| **2** | 0 | 231 | 6 | 10 | 5 | 44 | 1 | 1 | 0 | 0 | 2 | 37 | 4 | 13 | 16 | 3 | 5 | 3 | 8 | 0 |
| **3** | 1 | 17 | 156 | 35 | 12 | 45 | 1 | 1 | 1 | 0 | 0 | 60 | 4 | 14 | 15 | 4 | 5 | 3 | 15 | 2 |
| **4** | 0 | 9 | 27 | 264 | 13 | 5 | 4 | 1 | 0 | 0 | 1 | 24 | 23 | 4 | 8 | 3 | 1 | 2 | 3 | 0 |
| **5** | 1 | 17 | 9 | 32 | 215 | 5 | 3 | 1 | 0 | 0 | 0 | 25 | 15 | 23 | 14 | 1 | 7 | 3 | 11 | 1 |
| **6** | 0 | 20 | 8 | 4 | 0 | 305 | 0 | 1 | 0 | 0 | 0 | 30 | 1 | 6 | 7 | 0 | 3 | 3 | 2 | 0 |
| **7** | 0 | 12 | 8 | 40 | 13 | 3 | 177 | 16 | 5 | 0 | 2 | 13 | 20 | 16 | 15 | 4 | 8 | 13 | 14 | 3 |
| **8** | 0 | 2 | 1 | 0 | 0 | 0 | 3 | 314 | 11 | 0 | 0 | 7 | 4 | 4 | 5 | 3 | 15 | 4 | 22 | 0 |
| **9** | 1 | 1 | 0 | 0 | 0 | 0 | 2 | 24 | 343 | 0 | 0 | 3 | 2 | 2 | 2 | 0 | 8 | 4 | 5 | 0 |
| **10** | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 308 | 18 | 4 | 1 | 5 | 5 | 7 | 11 | 5 | 30 | 0 |
| **11** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 377 | 2 | 0 | 0 | 0 | 0 | 5 | 3 | 8 | 1 |
| **12** | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 366 | 0 | 1 | 3 | 1 | 9 | 3 | 8 | 1 |
| **13** | 4 | 9 | 1 | 11 | 1 | 2 | 4 | 9 | 7 | 0 | 0 | 66 | 221 | 19 | 23 | 3 | 5 | 2 | 5 | 1 |
| **14** | 4 | 5 | 0 | 1 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 4 | 3 | 308 | 8 | 14 | 11 | 11 | 19 | 1 |
| **15** | 0 | 3 | 0 | 1 | 0 | 3 | 0 | 1 | 1 | 0 | 1 | 5 | 3 | 4 | 337 | 3 | 3 | 5 | 22 | 0 |
| **16** | 6 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 362 | 1 | 7 | 5 | 13 |
| **17** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 3 | 0 | 6 | 1 | 3 | 300 | 6 | 32 | 11 |
| **18** | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 6 | 2 | 350 | 12 | 0 |
| **19** | 9 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 5 | 6 | 2 | 53 | 17 | 202 | 9 |
| **20** | 31 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 3 | 0 | 1 | 8 | 41 | 12 | 8 | 14 | 132 |

1. alt.atheism
2. comp.graphics
3. comp.os.ms-windows.misc
4. comp.sys.ibm.pc.hardware
5. comp.sys.mac.hardware
6. comp.windows.x
7. misc.forsale
8. rec.autos
9. rec.motorcycles
10. rec.sport.baseball
11. rec.sport.hockey
12. sci.crypt
13. sci.electronics
14. sci.med
15. sci.space
16. soc.religion.christian
17. talk.politics.guns
18. talk.politics.mideast
19. talk.politics.misc
20. talk.religion.misc

## bernoulli.m

```
% Read vocabulary and training, test labels data from files into matlab
words=textread('data/vocabulary.txt','%s');
newsgroup_names=textread('data/newsgrouplabels.txt','%s');
labels_train=textread('data/train.label','%d');
labels_test=textread('data/test.label','%d');

% read training data
[docid wordid counts]=textread('data/train.data','%d %d %d');
data_train=sparse(docid, wordid, counts, numel(labels_train),
numel(words));

% read test data
[docid wordid counts]=textread('data/test.data','%d %d %d');
data_test=sparse(docid, wordid, counts, numel(labels_test), numel(words));

clearvars docid;clearvars wordid;clearvars counts;

alph=1/2;

p_W_giv_C=zeros(numel(newsgroup_names),size(data_train,2))+alph;
pC=zeros(numel(newsgroup_names),1);

% find probability of each class
for(i=1:numel(newsgroup_names))
    pC(i)=sum(labels_train==i);
end
pC=pC./sum(pC);

% find probability of each word given a class
for(i=1:numel(newsgroup_names))

p_W_giv_C(i,:)= p_W_giv_C(i,:)+sum(data_train(find(labels_train==i),:) ~=
0,1);
p_W_giv_C(i,:)=p_W_giv_C(i,:)./size(data_train(find(labels_train==i),:),1);

end

x = data_test(:,:) ~= 0;
doc_probs_mat = log(p_W_giv_C)* x' + log( 1 - p_W_giv_C) * (~x)';
doc_probs_mat = docprobs + repmat(log(pC),1,size(data_test,1));

[~,classes]=max(docprobs);

% find confusion matrix
C=confusionmat(labels_test,classes,'order',1:20);

% print overall performance
crct = sum(diag(C));
accuracy = crct/sum(sum(C));
```