Assignment 2

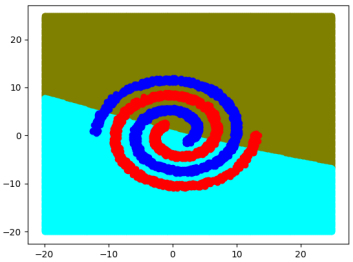
Archit Kumar, Avinav Sanyal, Sakhile Naga Koti Reddy

Friday 10$^{\text{th}}$ November, 2017

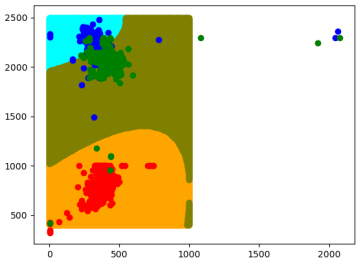
Classification

Classification is a process of grouping of objects according to their characterstics. The process can be supervised and unsupervised. In this assignment we are doing classification by , by dividing our dataset into training and test datasets, using training dataset to make a bayes classifier on Gaussian Mixture of Models. The Gaussian Mixture of Models is initiated by means calculated using k-means clustering. Expectation Maximisation is used in both the techniques.
The dataset provided to our team are :

1. **Data-set 1 :** 2-dimensional Non linearly separable artificial data
2. **Data-set 2a :** 2d Real Speech Data
3. **Data-set 2b :** Scene Image data
4. **Data-set 2c :** Cell Image



1 Non-linearly Separable data

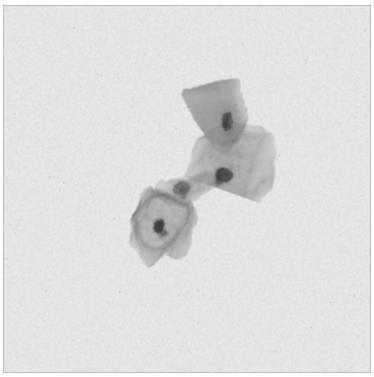


2a. Real-world Speech data

2b Arch

2b Forest Path

2b Highway



2c. Cervical Cytology Cell Data

Feature Extraction

The first step in this assignment is feature extraction: to take essential information out of the datasets and store it in a nice format.

Scene image : hist.py, kmeans-bovw.py, bovw.py

As directed in the assignment, the image is first divided into patches of 64*64 sq pixels and then for each patch RGB histogram is calculated and stored as 24-dimensional vector.The RGB value of a pixel is taken using Python Image Library. The pixel[x,y] returns [R, G, B] value of pixel (x,y).
For **Bag of Visual Words**, all the feature vectors are so calculated for all images and then clustered together into 64 clusters using k-means clustering technique. Then for each image a 64 bit array is maintained which stores count of each of these 64 cluster present in the image. This is called Bag of Visual Words(BoVW).
The BoVW can also be generated considering 32 or 16 clusters also, here as we have less train data, the representation is highly sparse leading to problems when various doing operations.

Cell Image : greyscale.py

We have greyscale images, which have only intensity values. We have divided the images into overlapping patches. The mean and variance for each patch is calculated and stored in a 2d vector. This results into a large amount of data, which will result in taking huge computation power to classify.

K-means

K-means clustering is an unsupervised learning algorithm, which is used when you have unlabeled data (i.e., data without defined categories or groups). The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity. The results of the K-means clustering algorithm are:
1. The centroids of the K clusters, which can be used to label new data.
2. Labels for the training data (each data point is assigned to a single cluster)
Rather than defining groups before looking at the data, clustering allows you to find and analyze the groups that have formed organically. The "Choosing K" section below describes how the number of groups can be determined. Each centroid of a cluster is a collection of feature values which define the resulting groups. Examining the centroid feature weights can be used to qualitatively interpret what kind of group each cluster represents.

Algorithm

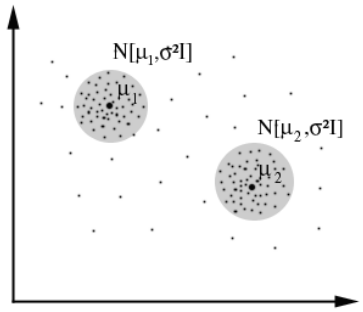
The means clustering algorithm uses iterative refinement to produce a final result. The algorithm inputs are the number of clusters and the data set. The data set is a collection of features for each data point. The algorithms starts with initial estimates for the centroids, which can either be randomly generated or randomly selected from the data set. The algorithm then iterates between two steps:
1. Data assigment step:
Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance.
2. Centroid update step:
In this step, the centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster.

The algorithm iterates between steps one and two until a stopping criteria is met (i.e., no data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached). This algorithm is guaranteed to converge to a result. The result may be a local optimum (i.e. not necessarily the best possible outcome), meaning that assessing more than one run of the algorithm with randomized starting centroids may give a better outcome. Time complexity of k-means algorithm
Let N be the number of points, D the number of dimensions, and K the number of centers. Suppose the algorithm runs I iterations to converge. The space complexity of K-means clustering algorithm is $O(N(D + K))$. Based on the number of distance calculations, the time complexity of K-means is $O(NKI)$.

Gaussian Mixture Models

In practice, each cluster can be mathematically represented by a parametric distribution, like a Gaussian (continuous) or a Poisson (discrete). The entire data set is therefore modelled by a mixture of these distributions. An individual distribution used to model a specific cluster is often referred to as a component distribution.
A mixture model with high likelihood tends to have the following traits:

- component distributions have high peaks (data in one cluster are tight)
- the mixture model covers the data well (dominant patterns in the data are captured by component distributions)



Clusters in Mixture Model

Algorithm

We apply Expectation Maximisation algorithm to get the parameters of different gaussians. The initial values of Means, covariance matrices and mixture coefficients are calculated using K-means to give a better start.

1. **E-step :** Responsibility term is calculated

$$\gamma_k(\mathbf{x}) \equiv p(k|\mathbf{x}) = \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

γ calculation

2. **M-step :** Values of Mean, Covariance matrice and Mixture coefficient are calculated.

$$\Sigma_j = \frac{\sum_{n=1}^{N} \gamma_j(x_n)(x_n - \mu_j)(x_n - \mu_j)^T}{\sum_{n=1}^{N} \gamma_j(x_n)}$$

σ

$$\mu_j = \frac{\sum_{n=1}^{N} \gamma_j(x_n) x_n}{\sum_{n=1}^{N} \gamma_j(x_n)}$$

μ

$$\pi_j = \frac{1}{N} \sum_{n=1}^{N} \gamma_j(x_n)$$

π

3. **Evaluation of Log likelihood :** If the value of log-likelihood converges then stop the EM algorithm, otherwise repeat steps 1 and 2.

$$\ln p(X \mid \mu, \Sigma, \pi) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k N(x_n \mid \mu_k, \Sigma_k) \right\}$$

Log likelihood

Problems Faced and their Solutions

Large determinant of covariance matrixes

Min-max normalisation:
Min-max normalisation is often known as feature scaling where the values of a numeric range of a feature of data, i.e. a property, are reduced to a scale between between 0 and 1. Therefore, in order to calculate z, i.e. the normalised value of a member of the set of observed values of x, we must employ the following formula:

$$z = \frac{x \text{ - } \min(x)}{\max(x) \text{ - } \min(x)}$$

Min Max Normalisation

Where min and max are the minimum and maximum values in x given its range.
We initially tried to normalise the histogram feature vectors using min-max normalisation.

GMM algorithm cons

According to scikit, "When one has insufficiently many points per mixture, estimating the covariance matrices becomes difficult, and the algorithm is known to diverge and find solutions with infinite likelihood unless one regularizes the covariances artificially", this is what we faced in the implementation.

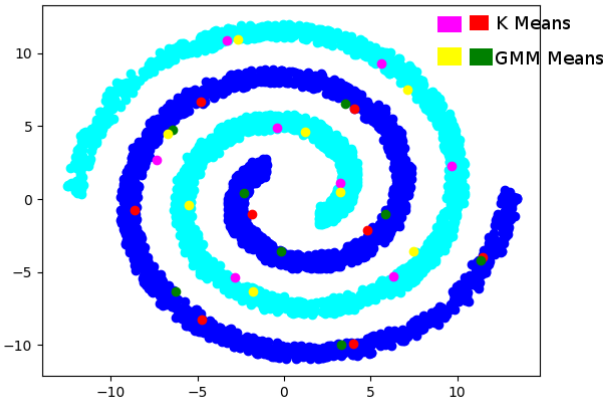
Too Sparse BoVW feature vector

Due to lack of training data the BoVW feature vectors were very sparse, hence when we applied our GMM on this, it gave many problems.
Singular Matrix problem: Solved by making non-diagonal elements zero and if the any of the diagonal elements is zero make it equal to some value (we experimented with average of diagonals and 0.1; taking 0.1 finally) of all the diagonal elements.

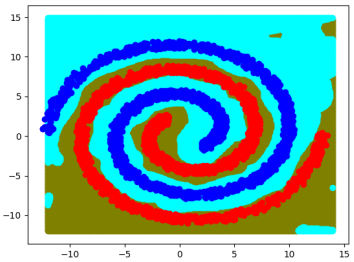
Results and Observations

Data Set 1

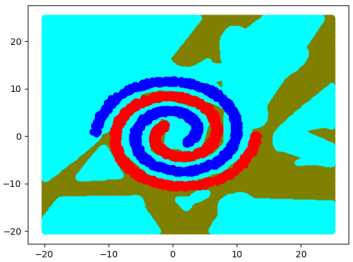
Non-linearly Separable data set We have two spirals in non-linearly separable data.



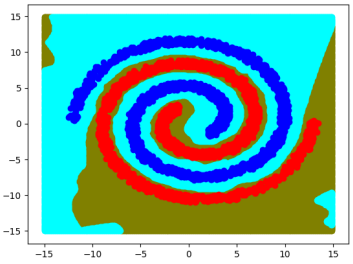
K Means Vs GMM Means



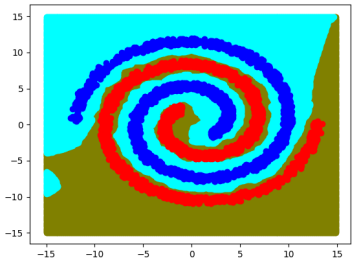
15 GMM iterations



20 GMM iterations



50 GMM iterations



100 GMM iterations

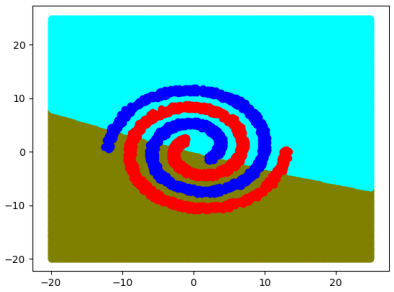


Figure 1: Decision region using Bayesian classification

For clusters=8

$$\begin{bmatrix} 597 & 15 \\ 0 & 612 \end{bmatrix}$$

Average accuracy : 98.77%

Table 1: Results

	Precision	Recall	F-measure
Class 1	0.975	1	0.987
Class 2	1	0.976	0.988
Average	0.988	0.988	0.988

For Clusters : 10

$$\begin{bmatrix} 612 & 0 \\ 0 & 612 \end{bmatrix}$$

Average accuracy : 100%

Table 2: Results

	Precision	Recall	F-measure
Class 1	1	1	1
Class 2	1	1	1
Average	1	1	1

For Clusters : 30

$$\begin{bmatrix} 612 & 0 \\ 0 & 612 \end{bmatrix}$$

Average accuracy : 100%

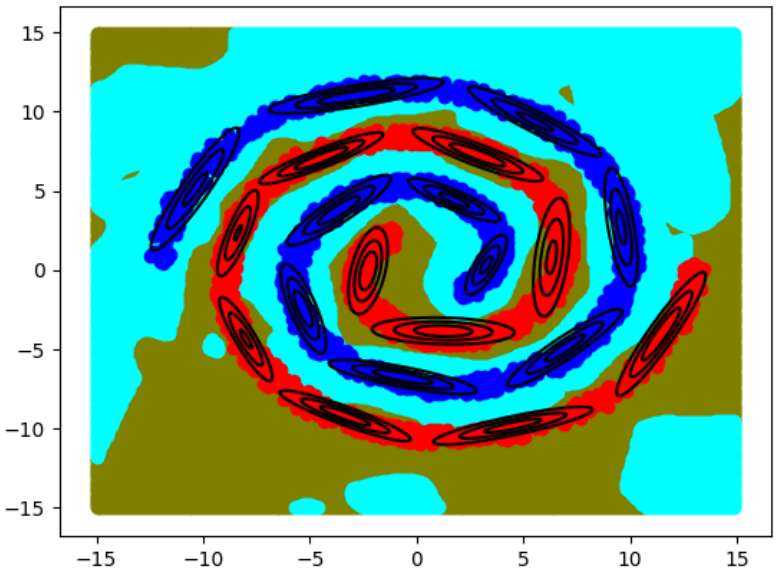
Table 3: Results

	Precision	Recall	F-measure
Class 1	1	1	1
Class 2	1	1	1
Average	1	1	1

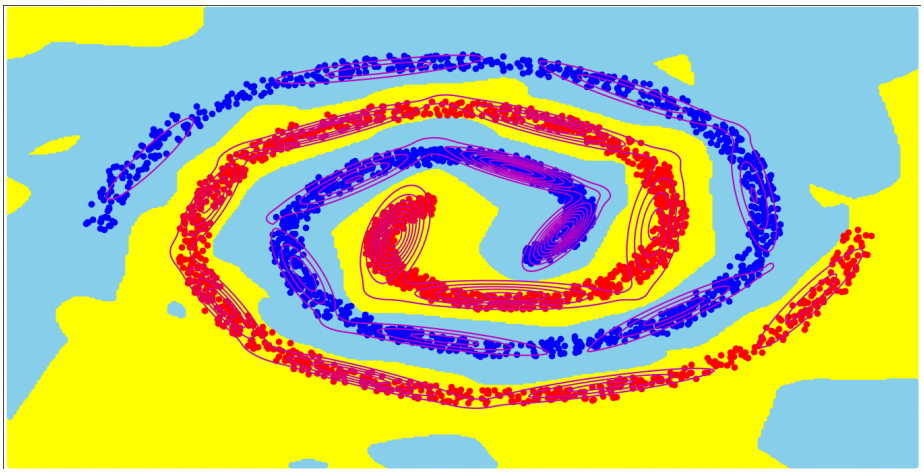
For more clusters the same Confusion Matrix is coming, this means increasing clusters than 10 does not change the accuracy much but if we see the Decision Region plot, GMM with 100 clusters is giving a much better clustering.

Comparision between Naive Bayes' and GMM

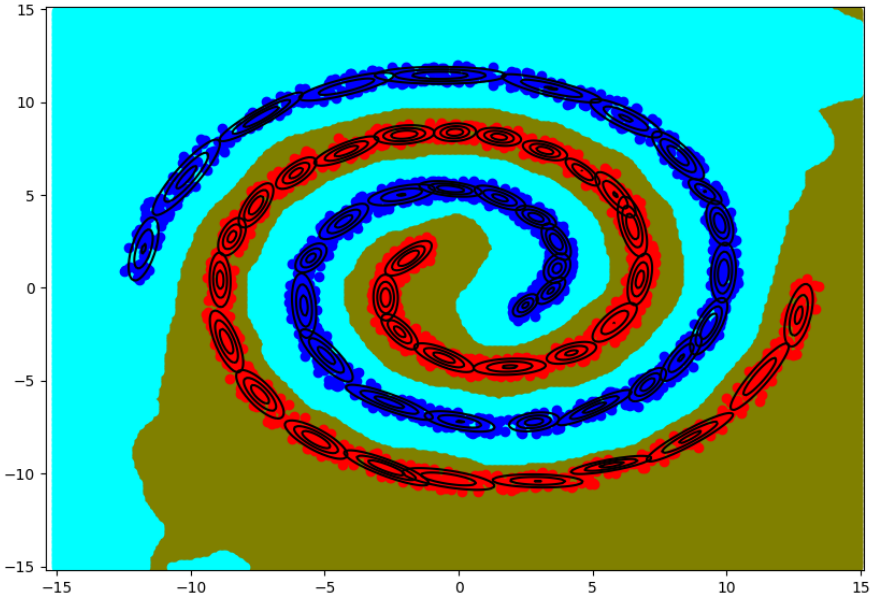
The average classification accuracy using GMM is 99.59%.
The classification accuracy using full covariance matrix naive Bayes' classifier is 63.15%.



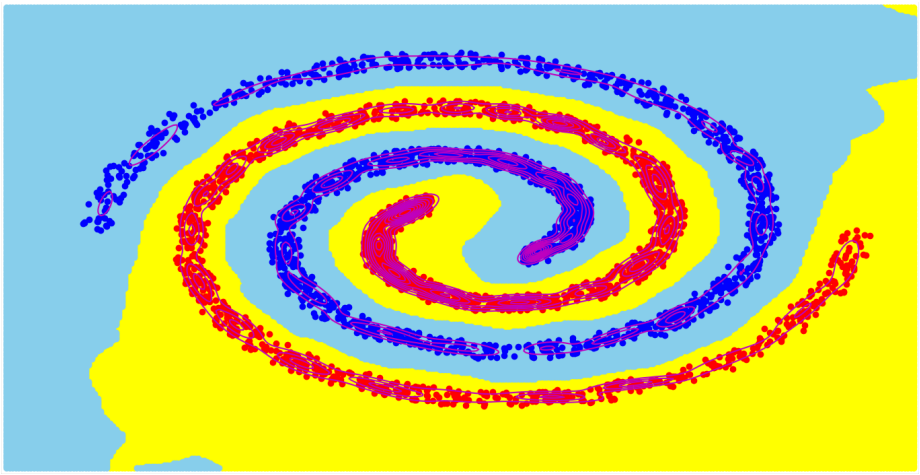
Contour of Clusters=10



Contour of GMM of clusters=10

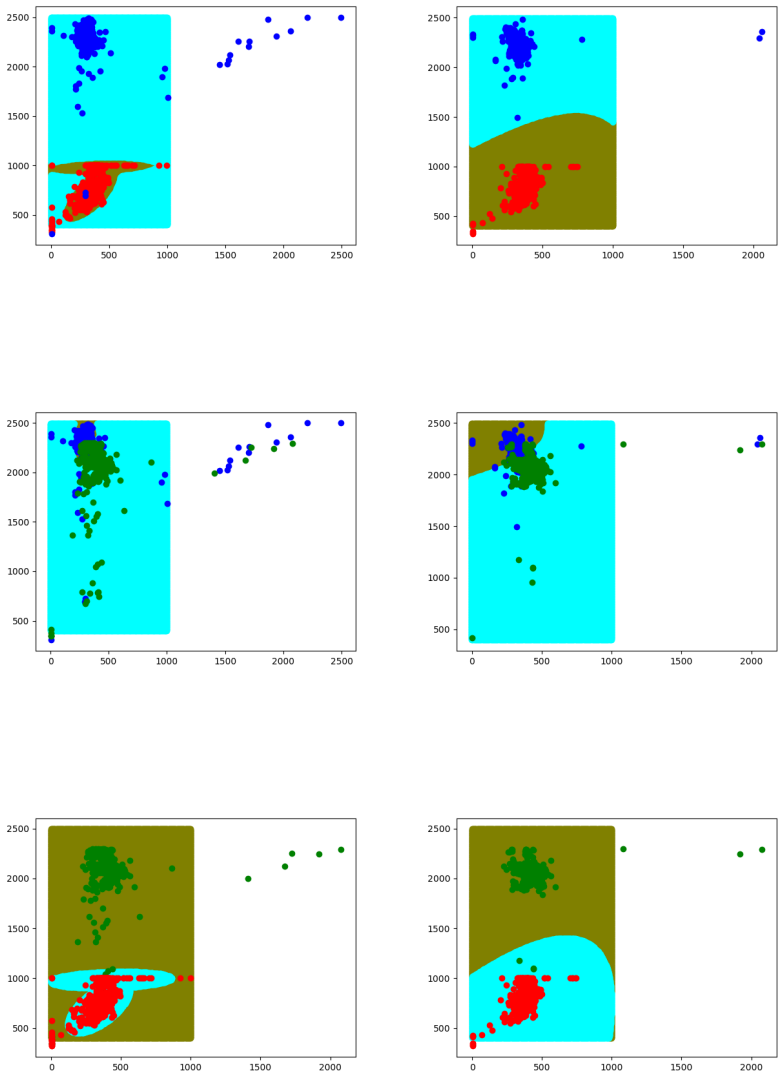


Contour of clusters=30



Contour of GMM of clusters=30

Data Set 2(a)



Footnote: The left side is using GMM classifier and the right is using Naive Bayes' classifier.
The points of the classes seem different because in assignment 1, training data was taken as top 75% of given data, whereas in assignment 2, we took the bottom 25% for the data.

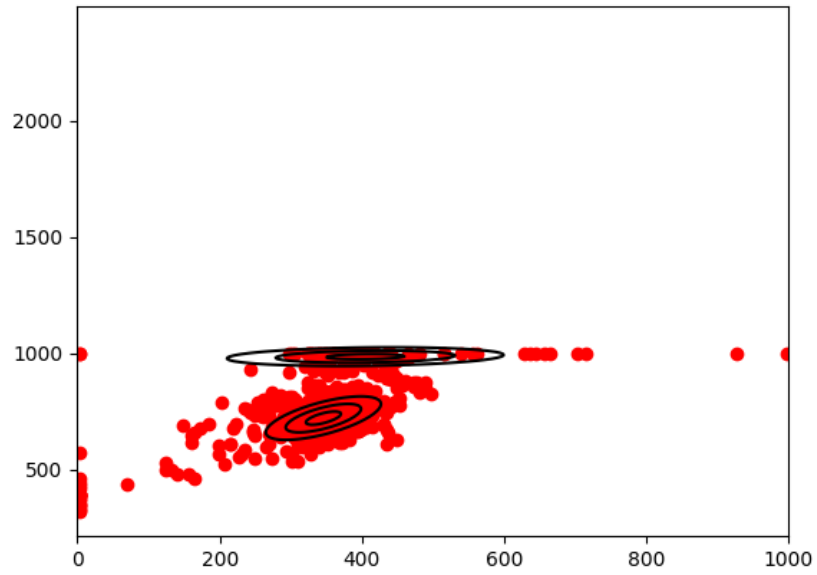
Clusters=2
Confusion matrix

$$\begin{bmatrix} 621 & 0 & 1 \\ 0 & 381 & 216 \\ 1 & 10 & 562 \end{bmatrix}$$

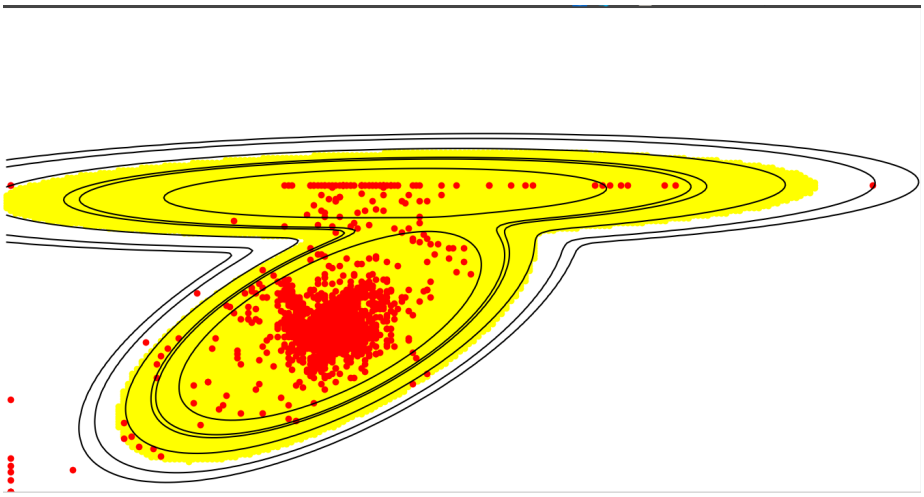
Average accuracy using GMM: 87.28%
Accuracy from assignment 1(taking full covariace matrix): 77.845%

Table 4: Results

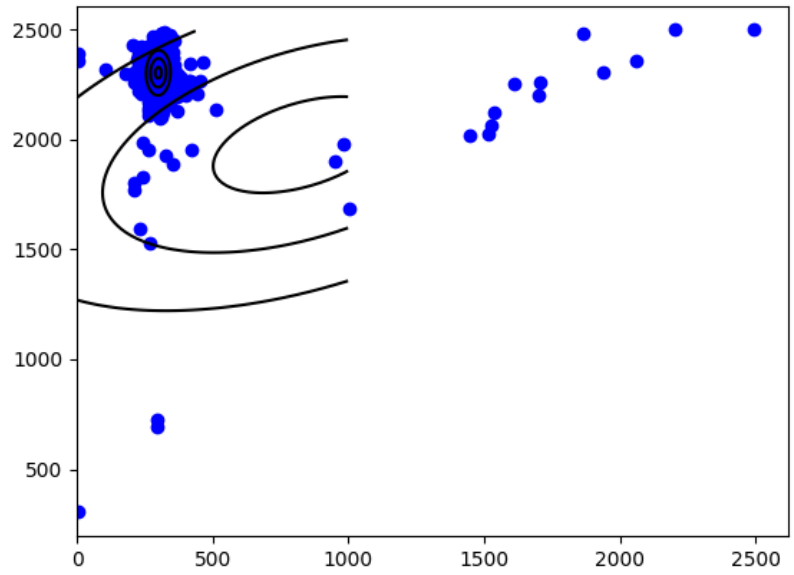
	Precision	Recall	F-measure
Class 1	0.998	0.998	0.998
Class 2	0.638	0.974	0.771
Class 3	0.981	0.721	0.831
Average	0.872	0.898	0.867



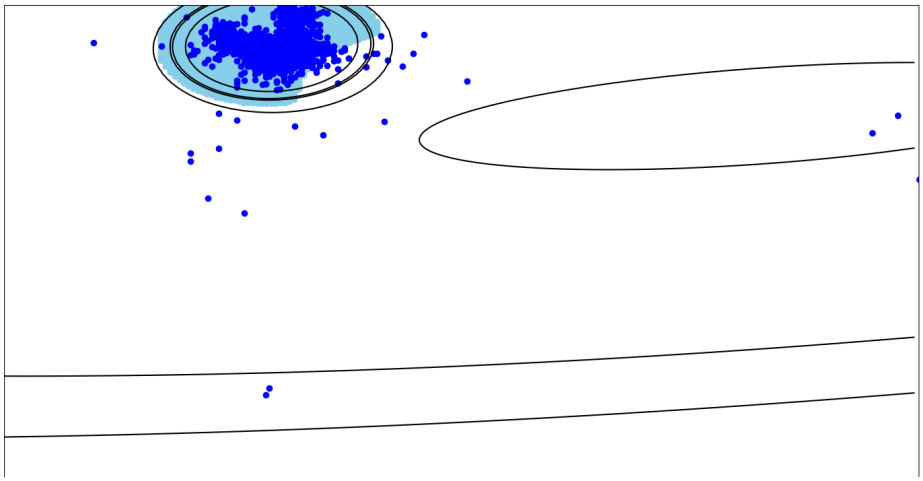
Contour of Clusters



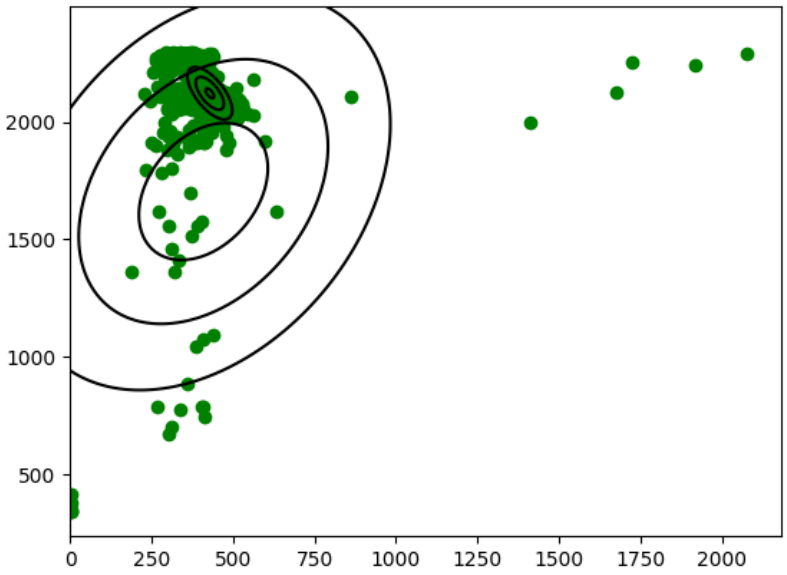
Contour of GMM



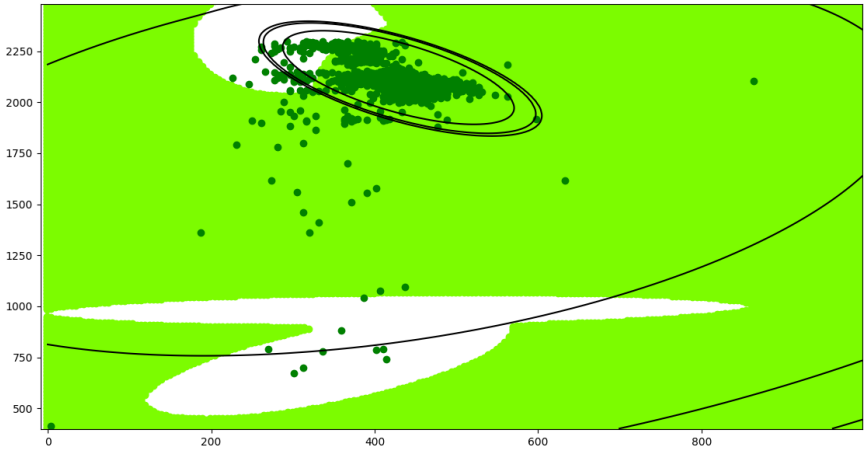
Contour of clusters



Contour of GMM of clusters



Contour of clusters



Contour of GMM of clusters

Data Set 2b

Colour histogram feature

- Consider 64 x 64 nonoverlapping patches on every images (from training and test sets). For example, if image size is 256 x 256, there there will be 16 number of 64 x 64 nonoverlapping patches.
- Extract 8-bin colour histogram from every colour channel(R, G and B) from a patch. It results in 3 8-dimentional feature vectors. Concatenate them to form 24-dimentional feature vectors.
- Similarly, extract 24-dimentional feature vector from every patch.
- Stack the 24-dimentional feature vectors corresponding to every patch in an image and save them as a file in the corresponding class folder.
- Thus an image is represented as set (collection) of 24-dimentional colour histogram vectors representation.
- Repeat the above steps to all the images in training and test sets of all the classes.

Colour histogram is computed as follows from a colour channel

- When the given image is read, it will be read as 3-dimentional matrix of pixel values. Each dimension is corresponding to a colour channel. The pixel values in each colour channel are in the range 0 to 255.
- For a colour channel,
  - Divide this range into 8 equal bins.
  - Count the number of pixels falling into each bins. This results in a vector of 8 values.
  - This is the 8-dimentional colour histogram (from a colour channel) feature vector.
- Do the same for other colour channels. Concatenate those three 8- dimensional colour histogram vectors to form 24-dimentional vector.

Scene Image data-set

For histogram feature vector

for clusters =[4,4,4]

$$\begin{bmatrix} 6 & 39 & 5 \\ 0 & 43 & 7 \\ 7 & 27 & 16 \end{bmatrix}$$

Average accuracy : 43.33%

Table 5: Results

	Precision	Recall	F-measure
Class 1	0.12	0.46	0.19
Class 2	0.86	0.39	0.54
Class 3	0.32	0.57	0.41
Average	0.43	0.47	0.38

Bag-of-visual-words (BoVW) feature using K-means clustering

- Take the 24-dimentional colour histogram feature vectors of all the training examples of all the classes.
- Group them into 64 clusters using K-means clustering algorithms.
- Now take an image, assign each 24-dimentional colour histogram feature vector to a cluster.
- Count the number of feature vectors assigned to each of the 64 clusters.
- This results in a 64-dimentional BoVW representation for that image.
- Repeat this for every images in training and test set.

Confusion matrices and their observations using different number of clusters

For clusters =[1,1,1]
We take all the images of same type in a class.

$$\begin{bmatrix} 16 & 17 & 17 \\ 5 & 43 & 2 \\ 11 & 6 & 33 \end{bmatrix}$$

Average accuracy : 59.42%

Table 6: Results

	Precision	Recall	F-measure
Class 1	0.32	0.5	0.78
Class 2	0.86	0.65	0.755
Class 3	0.66	0.635	0.647
Average	0.613	0.595	0.727

For clusters =[2,2,2]
We divide the images into two types in each class.

$$\begin{bmatrix} 18 & 21 & 11 \\ 2 & 48 & 0 \\ 11 & 10 & 29 \end{bmatrix}$$

Average accuracy : 63.33%

Table 7: Results

	Precision	Recall	F-measure
Class 1	0.3	0.581	0.396
Class 2	0.96	0.608	0.744
Class 3	0.58	0.725	0.644
Average	0.613	0.638	0.595

For clusters =[4,4,4] We divide the images into 4 classes.

$$\begin{bmatrix} 21 & 21 & 8 \\ 10 & 40 & 0 \\ 17 & 9 & 24 \end{bmatrix}$$

Average accuracy : 56.67%

Table 8: Results

	Precision	Recall	F-measure
Class 1	0.42	0.4375	0.429
Class 2	0.8	0.571	0.666
Class 3	0.48	0.75	0.585
Average	0.567	0.586	0.563

Observations

Among the three cluster representations, [2,2,2] has the highest accuracy. So, dividing the class into two different types of images than having either one or four different images in a class.

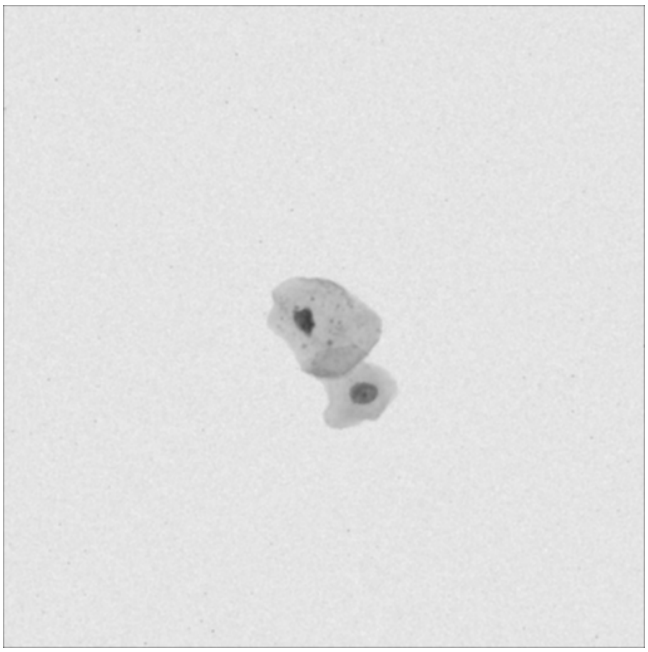
Observation between colour histogram representation and BoVW representation

Using the BoVW representation, we can classify more accurately than the colour histogram representation.

Data Set 2(c)

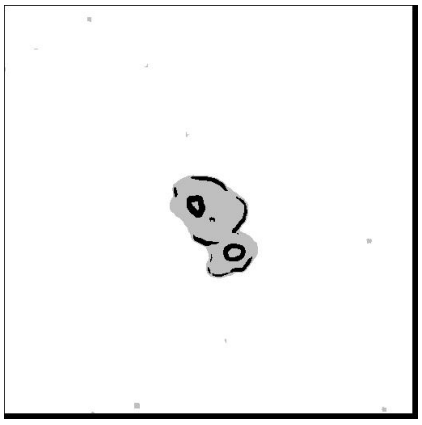
Features from images of Dataset 2(c)

- Consider 7 x 7 overlapping patches with a shift of 1 pixel on every training cell images.
- Compute mean and variance of intensities of pixels in the 7 x 7 patch.
- Thus a 7 x 7 patch is represented as 2-dimentional feature vector.
- In the similar way compute 2-dimentional feature vector from every patch from every training image.
- Stack all the 2-dimentional feature vectors in a file.
- For test images: Each test image is represented as a separate file of stacked 2-dimensional feature vectors.

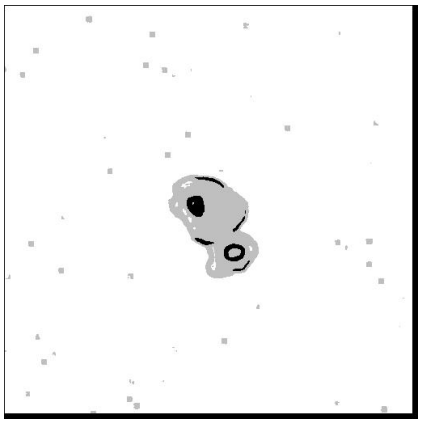


93.png

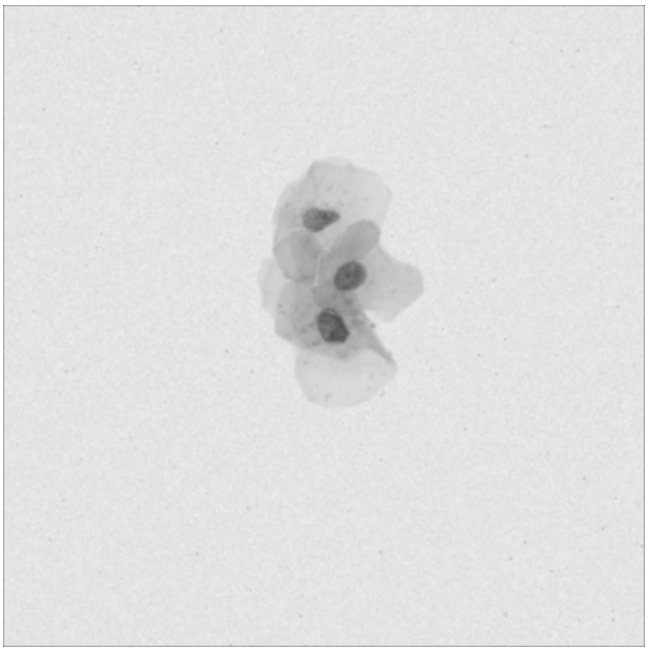
Number of clusters= 3
Those are the cell nucleus, cell body and outside surroundings of the cells.



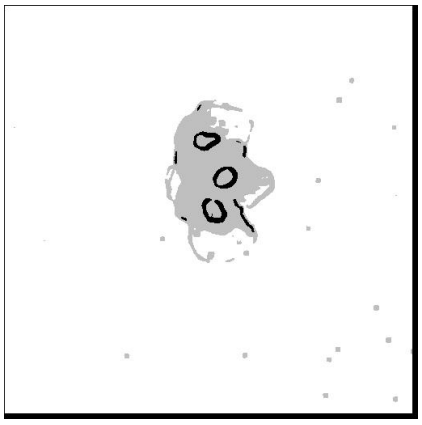
After 10 iterations of GMM



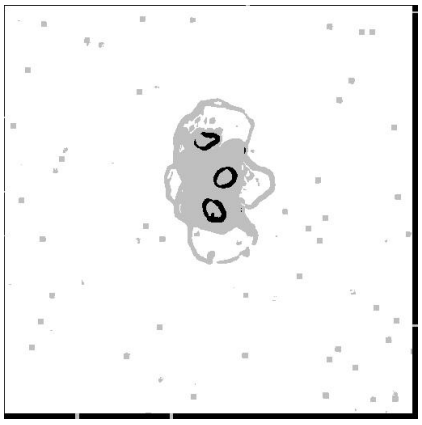
20 GMM iterations



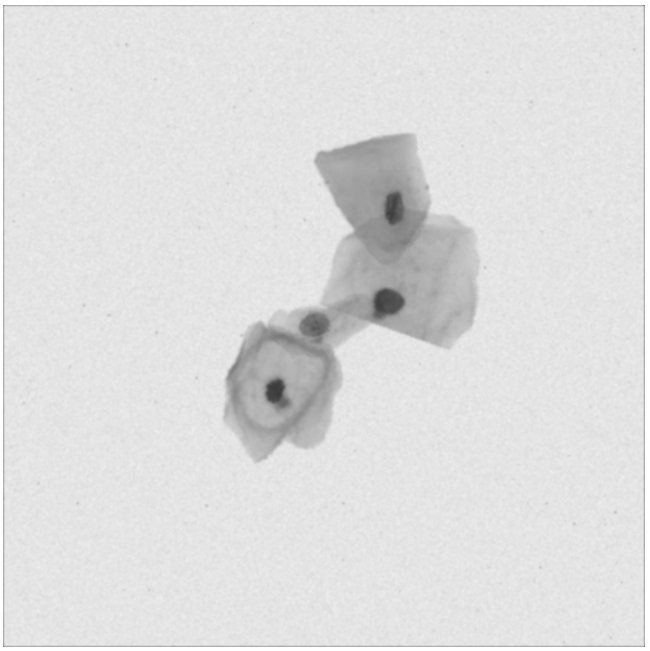
98.png



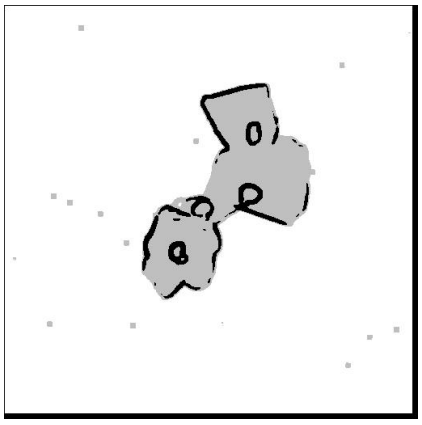
After 10 iterations of GMM



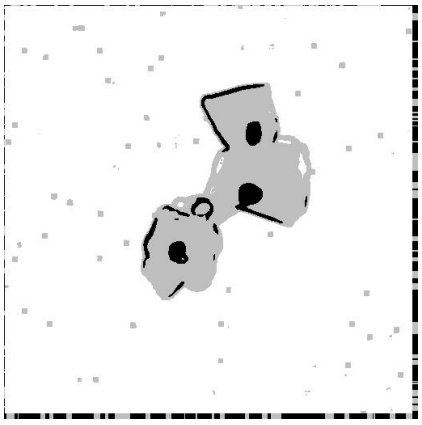
20 GMM iterations



103.png



10 GMM iterations



20 GMM iterations

Observations

In each iteration, the nucleus keeps on darkening. The cell membrane keeps becoming more visible. We stopped at 20 iterations as the construction of the nucleus and cell membrane was almost done and it was taking large amount of time.