PV SRIRAM
18015537

Q1) Fischer's Linear Discriminants Analysis

Required to perform dimensionality reduction for given dataset.

$C_1 \Rightarrow$

x_1	x_2
1	2
2	3
3	3
4	5
5	5

$C_2 \Rightarrow$

x_1	x_2
1	0
2	1
3	1
3	2
5	3
6	5

M_1 (Mean) $= \frac{1}{n} \sum_{\substack{i=1 \\ x_i \in C_1}}^{n} x_i = \frac{1}{5} [15 \; 18]$

$= [3 \; 3.6]$

M_2 (Mean) $= \frac{1}{6} [20 \; 12]$

$= [3.3 \; 2]$

Scatter matrix $S = \sum_{j=1}^{n} (x_j - \bar{x})(x_j - \bar{x})^T$

($\bar{x}$: mean of x_j)

$\Rightarrow S_1 = \begin{bmatrix} 10 & 8 \\ 8 & 7.2 \end{bmatrix}$ $S_2 = \begin{bmatrix} 17.3 & 16 \\ 16 & 16 \end{bmatrix}$

Within class scatter $\Rightarrow S_w = S_1 + S_2$

$$S_w = \begin{bmatrix} 27.3 & 24 \\ 24 & 23.2 \end{bmatrix}$$

Since, S_w has full rank we don't solve for the eigenvalues, instead we have

$$\therefore S_w^{-1} = \frac{adj(S_w)}{|S_w|} = \begin{bmatrix} 0.40 & -0.42 \\ -0.42 & 0.48 \end{bmatrix}$$

The optimal line direction v is

$$v = S_w^{-1}(\mu_1 - \mu_2) = \begin{bmatrix} 0.40 & -0.42 \\ -0.42 & 0.48 \end{bmatrix} \begin{bmatrix} -0.3 \\ 1.6 \end{bmatrix}$$

$$= \begin{bmatrix} -0.79 \\ 0.89 \end{bmatrix}$$

Projection of initial 2D points on this line

$$Y_1 = v^t C_1^t = \begin{bmatrix} -0.79 & 0.89 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 3 & 5 & 5 \end{bmatrix}$$

$$\Rightarrow \boxed{\begin{bmatrix} 0.99 & 1.09 & 0.3 & 1.14 & 0.5 \end{bmatrix}}$$

$$Y_2 = v^t C_2^t \Rightarrow \begin{bmatrix} -0.79 & 0.89 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 & 3 & 5 & 6 \\ 0 & 1 & 1 & 2 & 3 & 5 \end{bmatrix}$$

$$\Rightarrow \boxed{\begin{bmatrix} -0.79 & -0.69 & -1.48 & -0.59 & -1.8 & 0.29 \end{bmatrix}}$$

Fischer LDA tries to find the projection weight vector W such that

$$J(W) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \quad \text{is maximum}$$

i.e, It tries to maximize the ratio between the Between class variance to the within-class variance. Qualitatively, it tries to find a large seperation between the projected class means while also giving a small variance within each class thereby minimizing the class overlap.

Principal Component Analysis

Mean of whole dataset = $\frac{1}{11}[35 \quad 30] = [3.18 \quad 2.72]$

Now, feature vectors are $x_i - \mu$ for all x_i in dataset

Features: $[(-2.18, -0.72),$
$(-1.18, 0.27),$
$\vdots$
$(2.82, 2.28)]$

Now, we get the covariance matrix of the features

$$C(X) = \frac{1}{|X|} \sum_{f_i \in features} (f_i - M_{f_i})(f_i - M_{f_i})^T$$

$$C(X) = \begin{bmatrix} 2.51 & 2.04 \\ 2.04 & 2.74 \end{bmatrix}$$

Now, we find the eigen values and vectors of the equation $CY = \lambda Y$; Eigen values : $|C - \lambda I| = 0$

$\lambda = 0.58, 4.68$; We chose the largest value of λ i.e 4.68 and consider the corresponding eigen vector $\begin{bmatrix} 0.94 \\ 1 \end{bmatrix}$ as the principal component.

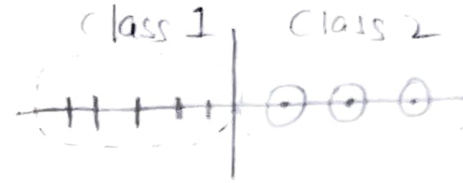
Now, projecting initial features on this vector we have $F_{n \times 2} \; Y_{2 \times 1} = \text{New Features}_{n \times 1}$

The id features obtained after PCA are :

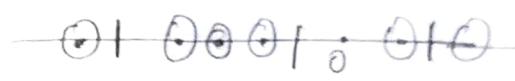
$[-2.79, -0.84, 0.10, 3.05, 4.0, -4.8, -2.8, -1.9, -0.9, 3.5]$

Performance Comparision

Fischer :



PCA :



On observing the points in the reduced dimension produced by both Fisher and PCA, we see that clearly Fisher is able to differentiate between the two classes. On the other hand, PCA does not reduce the dimension effectively and there is a mixup of class points as seen.

LDA Better than PCA

LDA is a supervised learning technique with the aim of reducing dimensions ensuring maximum class separability. As per the working, LDA gives us the axes that account for the most variance between the individual classes whereas PCA accounts for the most variance in the whole dataset. Since LDA is more class specific it performs better for classification as compared to PCA. Therefore Fisher LDA is better than PCA.

Q2) DBSCAN

DBSCAN is an Unsupervised ML algorithm that performs the clustering based on the density of the data points. This algorithm works on a parametric approach that uses two parameters

a) eps : It represents the radius of neighbors around data point

b) minPts : Minimum number of data points that we want in the neighbourhood.

Estimation of parameters

a) Min Pts:

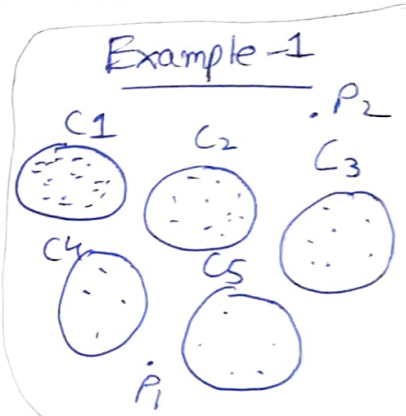
As a rule of thumb, a minimum minpts can be derived from the number of dimensions 'D' in the dataset as minPts $\geq D+1$. The low value of min Pts causes all points to be core. With minPts ≤ 2, the result will be the same as hierarchial clustering with single link metric, with the dendogram cut at height 'ε'. Therefore minpts ≥ 3. However large values are suitable for dataset with noise and will yield more significant cluster.

b) Eps:

If eps is too small, a large part of data will not be clustered, whereas for a too high value of eps clusters

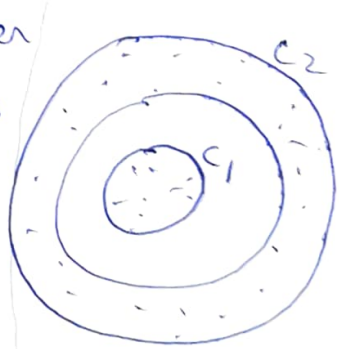
will merge and the majority of objects will be in the same cluster. In general, small values are preferable. The value of eps can be chosen by using a k-distance graph, plotting distance to the k = minpts − 1 nearest neighbour ordered from the largest to the smallest value. Good values of eps are where the plot shows an elbow. Alternatively an OPTICS plot can be used to choose eps.

DBSCAN cannot cluster datasets well with large difference in densities, since the minpts–eps combination cannot be chosen appropriately for all clusters.



In example-1, C_1, C_2, C_3 are very dense. C_4 is less dense while C_5 is sparse. P_1, P_2 are outliers. As different datapoints are located in different density regions, it is impossible to obtain all clusters simultaneously. Because different regions need different parameters.

Also, DBSCAN performs poorly in overlapping ~~clusters~~ regions.



We can see C_1 is surrounded by another cluster C_2. Points in C_2 represent a less dense region, while points in C_1 are in a high density region. So it is difficult to figure out optimal eps and minpts values.

03) Given a data set of 10 companies,

We are required to find the probability of a "small" and a "charged" company to be a "fraudulent" company.

i.e) $P(\text{fraud} \mid \text{size} = \text{small} \cap \text{charge} = \text{Yes})$

Bayes theorem $\Rightarrow P(A|B) = \dfrac{P(A \cap B)}{P(B)}$

$\Rightarrow P(\text{Fraud} \mid \text{small} \cap \text{Yes}) = \dfrac{P(F \cap (S \cap Y))}{P(S \cap Y)}$

$\Rightarrow \dfrac{P(F \cap S \cap Y)}{P(S \cap Y)}$

From table, we get, $P(F \cap S \cap Y) = \frac{1}{10}$

$P(S \cap Y) = \frac{2}{10}$

$\Rightarrow \frac{1}{10} / \frac{2}{10} = \boxed{\frac{1}{2}}$

Q4) Given,

$f(x) = x^2$

initial values of $x = \{13, 24, 8, 16\}$

Required to maximize $f(x)$ using Genetic algorithm over $\{0, 1, 2, \ldots 31\}$ over initial values of x.

String No.	x-value	Initial Population	$f(x) = x^2$	P(select) $= \frac{f_i}{\Sigma f_i}$	Expected count $= \frac{f_i}{\bar{f}}$	Actual count
1	13	01101	169	0.16	0.63	1
2	24	11000	576	0.54	2.16	2
3	8	01000	64	0.06	0.24	0
4	16	10000	256	0.24	0.96	1

$\Sigma f = 1065$

$\bar{f} = \frac{\Sigma f}{4} = 266$

For the initial population,

Sum, $\Sigma f = 1065$

Average, $\bar{f} = 266$

Max, $\max(f_i) = 576$

Mutation

We will replace the lowest number in the initial population i.e, 8 with the highest number i.e) 24.

For crossover, strings are randomly paired and mated. Then we select crossing sites randomly to perform crossover.

String No.	Mating pool after reproduction	Mate (Randomly selected)	Crossover site (Random)	New Population	x-value	$f(x) = x^2$
1	0110|1	2	4	01100	12	144
2	1100|0	1	4	11001	25	625
3	11|000	4	2	11000	24	576
4	10|000	3	2	10000	16	256

$\Sigma f = 1601$

$\bar{f} = \frac{\Sigma f}{4} \approx 400$

After one crossover and mutation:

Sum, $\Sigma f = 1601$

Average $\bar{f} = 400$

Max is 625

$\therefore f(x) = x^2$ has been maximized

Q5) Feature Selection

Feature selection is the process where you automatically or manually select those features which contribute most to the prediction variable or output. Having irrelevant features in data can decrease the accuracy of the models which learn irrelevant features.

Uses:

(i) It enables the Machine Learning algorithm to train faster

(ii) It reduces the complexity of a model and makes it easier to interpret.

(iii) It improves the accuracy of a model if the right subset is chosen.

(iv) It reduces overfitting

Selection strategies

Filter methods:

Filter methods pick up the intrinsic properties of the features measured via univariate statistics instead of cross validation performance. These methods are faster and less computationally expensive than wrapper methods. When dealing with high-dimensional data, it is computationally cheaper to use filter methods.

Set of all features → Selecting the best subset

↓

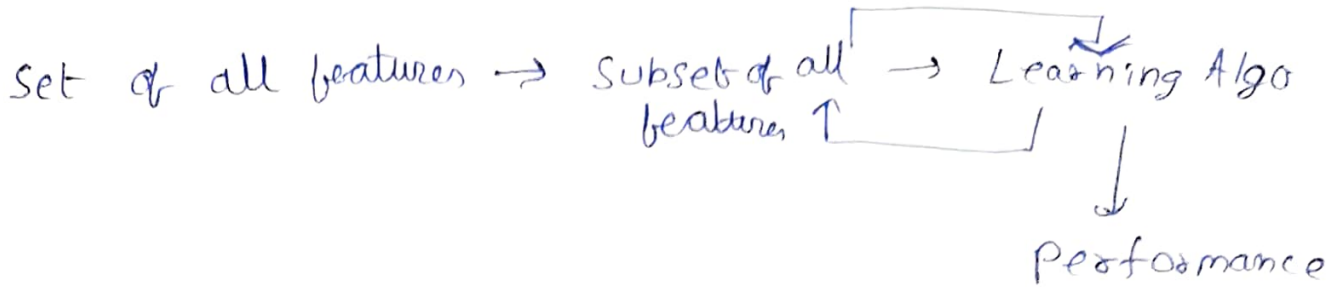
Performance ← Learning Algorithm

Techniques:

a) Information Gain
b) Chi-square Test
c) Fisher's Score
d) Correlation Coefficient
e) Variance Threshold
f) Mean Absolute Difference
g) Dispersion Ratio

Wrapper Methods

Wrappers require some method to search the space of all possible subsets of features, assessing their quality by learning and evaluating a classifier with that feature subset. The feature selection process is based on a specific machine learning algorithm that we are trying to fit on a given dataset. It follows a greedy search approach by evaluating all the possible combinations of features against the evaluation criterion. The wrapper methods usually result in better predictive accuracy than filter methods.

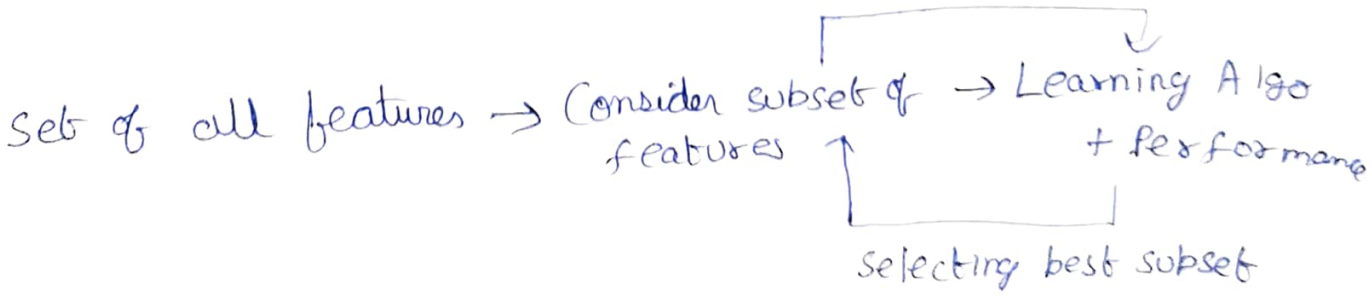


Techniques:

1) Forward selection
2) Backward elimination
3) Bi-directional elimination
4) Exhaustive selection
5) Recursive elimination

Embedded methods

These methods encompass the benefits of both the wrapper and filter methods, by including interactions of features but also maintaining reasonable computational cost. Embedded methods are iterative in the sense that takes care of each iteration of the model training process and carefully extracts those features which contribute the most to the training for a particular iteration



Techniques:

(i) Regularization

(ii) Tree based methods

When we are given a huge data sample containing 2^{20} features, we prefer to use filter methods as compared to wrapper and embedded methods. Filter methods are known to be faster and computationally less expensive than the other two. Since, wrapper methods search all possible subsets of features, it is not a good option for dealing with a huge number of features.

(Q6) K-medoid is more robust than - K-means, in terms of outliers. What this means is that medoids are less affected by outliers than centroids. However, there is a compromise in terms of complexity and efficiency as K-medoid algorithms are of $O(i k (n-k)^2)$ while K-means is of $O(i k n)$. A significant increase in complexity for performance.

However the given algorithm seems to solve the problem of complexities for performing local searches in place of universal searches done in the K-medoid algorithm. This obviously might result sometimes in a local optima, but increases the performance significantly based on our choice of subset size

Cost Function

While Square Mean error $\left(\frac{1}{n}\sum_{i=1}^{n}\|x - c_i\|^2\right)$ is a widely used cost function for its mathematical simplicity (smoothening of cost curve and easier gradient calculation), we will benefit by using a

simple Euclidean cost function because of the robustness it offers with outliers

A squared error ~~loss~~ cost penalises a cluster center highly compared to euclidean distance with outliers. This might result in a radical change of center location and might lead to misclassifi-cations. Therefore a subtler cost function like Euclidian distance wouldn't cause as much changes and thus, faster convergence

$$SED = \sum_{i=1}^{n} \sum_{j=1,\, x_i \in C_j}^{k} d(x_i, m_j)$$

Time Complexity

The algorithm includes,

1) K - Clusters
2) i iterations to update the medoids for each cluster
3) m * C comparisons for each iteration where m is the nearest neighbour size; C is cluster size.

$\Rightarrow O(i * k * m * C)$

Example,

Consider the dataset

D : { (2,10), (2,5), (8,4), (5,8), (7,5), (6,4), (1,2), (4,9) }

K-means

Consider, initial centroids : C1 (2,5) ; C2 (5,8) ;
C3 (7,5)

cost : $\|(x - c)\|^2$

X	Y	Dis C1	Dis C2	Dis C3
2	10	25	(13)	50
2	5	(—)	—	—
8	4	72	25	(2)
5	8	—	(—)	—
7	5	—	—	(—)
6	4	17	17	(2)
1	2	(10)	40	45
4	9	20	(2)	17

Updated Centroids: C1 (1.5, 3.5); C2 (3.67, 9);
C3 (7, 4.33)

X	Y	Dis C1	Dis C2	Dis C3
2	10	42.5	(3.79)	57.149
2	5	(2.5)	18.79	25.449
8	4	42.5	43.75	(1.1089)
5	8	32.5	(2.7689)	17.469
7	5	32.5	27.09	(0.449)
6	4	20.5	30.429	(1.109)
1	2	(2.5)	56.129	41.429
4	9	36.5	(0.109)	30.809

We can see that centroids will remain constant row onwards.

∴ Centoids are : C1 (1.5, 3.5)
C2 (3.67, 9)
C3 (7, 4.33)

Proposed method

Consider initial mediods : C1(2,5); C2(5,8); C3(7,5)

Consider m (nearest neighbours) = 1

Cost = $\Sigma\Sigma d(x, m)$

X	Y	Dis C1	Dis C2	Dis C3
2	10	5	3.605	7.071
2	5	—	—	—
8	4	6.083	5	1.414
5	8	—	—	—
7	5	—	—	—
6	4	4.123	4.123	1.414
1	2	3.1623	7.21	6.708
4	9	4.47	1.414	5

Updation of medoids,

1st Cluster :

Cost (2,5) → 3.1623

Cost (1,2) → 3.1623

∴ No update

2nd cluster :

Cost (5,8) → 5.019

Cost (4,9) → 3.65

∴ Update to (4,9)

3rd Cluster :

Cost (7,5) → 2.828

Cost (6,4) → 3.414

∴ No update

C1 (2,5) C2 (4,9) C3 (7,5)

X	Y	Dis C1	Dis C2	Dis C3
2	10	5	2.236	7.071
2	5	—	—	—
8	4	6.083	6.403	1.414
5	8	4.24	1.414	3.605
7	5	—	—	—
6	4	4.123	5.38	1.414
1	2	3.1623	7.615	6.708
4	9	—	—	—

We can see that there is no cluster change ⇒
No medoid change

∴ Medoids ⇒ C1 (2,5)
C2 (4,9)
C3 (7,5)

We can see that both these methods result in different medoids.

It is also observed that the variance of cost is higher in K-means and it is easily affected by outliers aswell.

Q7)

Given,

Cluster 1 : { (0,0), (0,1), (2,3) }

and

Cluster 2 : { (3,3), (3,4) }

Required to compute the silhouette for this clustering.

First we need to calculate the following metrics for each point p in each cluster:

a(p) → Average distance of point p to other points in its cluster,

b(p) → Smallest average distance of point p to all points in any other cluster

$$s(p) = \frac{b(p) - a(p)}{\max(a(p), b(p))}$$

By using these metrics, the silhouettes for each point are calculated in the table below:

Clusters	p	a(p)	b(p)	s(p)
Cluster 1	0, 0	(1+5)/2 = 3	(6+7)/2 = 6.5	3.5/6.5 = 0.538
	0, 1	(1+4)/2 = 2.5	(5+6)/2 = 5.5	3/5.5 = 0.545
	2, 3	(5+4)/2 = 4.5	(1+2)/2 = 1.5	−3/4.5 = −0.667
Cluster 2	3, 3	1/1 = 1	(6+5+1)/3 = 4	3/4 = 0.75
	3, 4	1/1 = 1	(7+6+2)/3 = 5	4/5 = 0.8

Interpreting the results

Observation: Cluster 2 is a good cluster. Cluster 1 isn't that good.

Reason: In Cluster 1, the point (2, 3) is having a negative silhouette of −0.667. This is because of the point (2, 3) is assigned to cluster 1, despite being closer to cluster 2. The remaining 4 points are assigned properly to their respective clusters. This is evident from their positive silhouette value

(Q8) Single Linkage

In single linkage hierarchical clustering, we merge in each step the two clusters whose two closest members have the smallest distance

Time complexity: $O(n^2)$

We compute all distances in $O(n^2)$. While doing this we also find the smallest distance for each data point and keep them in a next-best-merge array. In each of the $(n-1)$ merging steps, we then

find the smallest distance in the next best merge array. We merge the two identified clusters and update the distance matrix in $O(n)$. Finally, we update the next-best-merge array in $O(n)$ in each step. We can do the latter in $O(n)$ because if the best merge partner for k before merging i and j was either i or j, then after merging i and j the best merge partner for k is the merger of i and j.

Complete Linkage

In complete-link hierarchical clustering, we merge in each step the two clusters whose merger has the smallest diameter.

Time Complexity: $O(n^2 \log n)$

One $O(n^2 \log n)$ algorithm is to compute the n^2 distance metric and then sort the distances for each data point. After each merge iteration, the distance metric can be updated in $O(n)$. We pick the next pair to merge by finding the smallest distance that is still eligible for merging. If we do this by traversing the n sorted lists of distances, then by end of clustering, we done n^2 steps, Adding which gives $O(n^2 \log n)$

Average Linkage

The average linkage clustering is a method of calculating the distance between clusters in hierarchical cluster analysis. The linkage function specifying the distance between two clusters is computed as the average distance between objects from the first cluster and objects from second cluster.

Time Complexity : $O(n^2 \log n)$

We first compute all n^2 similarities for the singleton clusters and sort them for each cluster ($O(n^2 \log n)$). In each of $O(n)$ merge iterations, we identify the pair of clusters with the highest cohesion in $O(n)$; merge the pair; and update cluster centroids, gammas, and cohesions of the $O(n)$ possible mergers of the just created cluster with the remaining clusters. For each cluster, we also update the sorted list of merge candidates by deleting the two just merged clusters and inserting its cohesion with the just created cluster. Each iteration thus takes $O(n \log n)$. Overall TC is $O(n^2 \log n)$

Q9)

Given a simple linear regression model :-

$$y = \beta_0 + \beta_1 X + \varepsilon$$

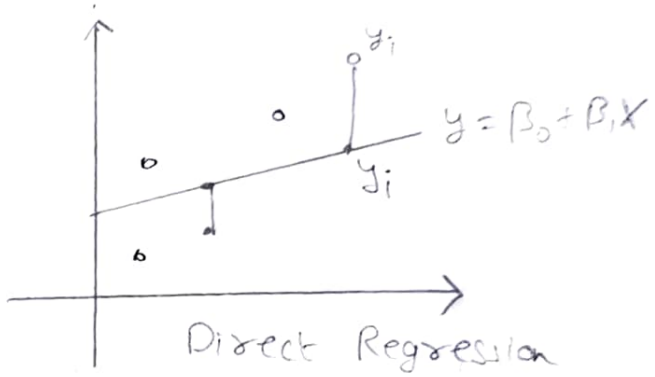
where y is the dependent variable, X is the independent or explanatory variable. The terms β_0 and β_1 are the parameters of the model. Specifically β_0 is the intercept term and β_1 the slope term. These are collectively the regression coefficients. ε is the unobservable error component that accounts for failure of data to lie on the straight line.

For determining the statistical model $y = \beta_0 + \beta_1 X + \varepsilon$, we will need to determine β_0 ; β_1. For knowing the values of these parameters, n pair of observations (x_i, y_i) on (x, y) are observed / collected.

For estimating the values of the parameters, various methods are available out of which methods of least square are the most popular.

Least Square estimation

The principle of LSE the parameters by minimizing the sum of the difference between observation and the line in the scatter diagram. In the direct regression method, the vertical difference between the observations and the line is taken for minimization, to obtain the parameters β_0, β_1.



Minimizing the sum of squares of error, w.r.t β_0, β_1

$$S(\beta_0, \beta_1) = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

→ Partial derivative of $S(\beta_0, \beta_1)$ w.r.t β_0 is

$$\frac{\partial S(\beta_0, \beta_1)}{\partial(\beta_0)} = -2 \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)$$

→ Partial derivative of $S(\beta_0, \beta_1)$ w.r.t β_1 is

$$\frac{\partial S(\beta_0, \beta_1)}{\partial(\beta_1)} = -2 \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i) x_i$$

equating these to 0, we get,

$$b_0 = \bar{y} - b_1 \bar{x} \quad ; \quad b_1 = \frac{S_{xy}}{S_{xx}}$$

where,

$$S_{xy} = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \; ; \; S_{xx} = \sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \; ; \quad \bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$$

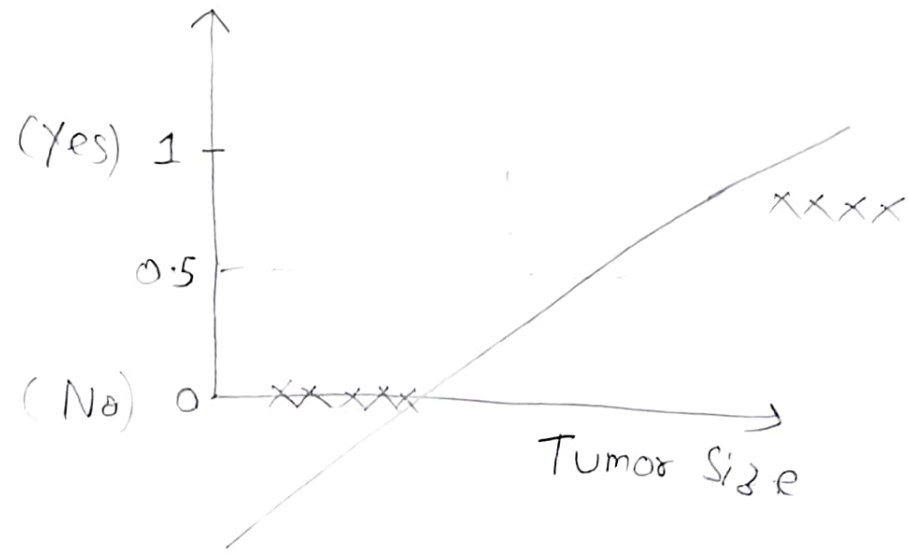
We can also verify that $S(\beta_0, \beta_1)$ has a global minimum at (b_0, b_1)

Now, the fitted line or the fitted linear regression model is $y = b_0 + b_1 x$, and $\hat{y}_i = b_0 + b_1 x_i$ are the predicted values.

Logistic Regression Model

In problems involving prediction of a discrete set of values typically called classes, we will need to model the probability of occurence of the class with respect to the ~~impact~~ input.

Suppose, using linear regression.



We see that for tumor size classification problem after setting the threshold to 0.5 the graph looks like the above one, with decision mapping function

$$h(f(x) = \beta_0 + \beta_1(x)$$

But $h(x)$ can be > 1 or < 0. We want discrete.

$$\Rightarrow 0 \leq h(x) \leq 1$$

∴ Given a binary output variable Y, we want to model conditional probability $P(Y|X)$ as a function of x.

⇒ linear func of x

⇒ $\log(P(x))$ also linear func of x

⇒ $\log(P)$ has an unbound range.

Firstly we have,

$$\text{Log}\left(\frac{P(x)}{1-P(x)}\right) = \beta_0 + x_i \beta_1$$

Solving for $P(x)$; we get $P(x_i; \beta_0 \beta_1) = \frac{1}{1+e^{-(\beta_0 + x_i \beta_1)}}$

Hence $P(x)$ is the estimated probability that $y = 1$ on input x. Hence, the logistic function for finding probability of y given x is as

$$P(Y|X) = \frac{1}{1+e^{-(\beta_0 + \beta_1 x)}}$$

Eg:

Data : $\{(2,3), (4,7), (6,5), (8,10)\}$

Assume, $y = a + bx$

$$a = \frac{\Sigma y \, \Sigma x^2 - \Sigma x \, \Sigma xy}{(\Sigma x^2) - (\Sigma x)^2}$$

$$b = \frac{n \Sigma xy - (\Sigma x)(\Sigma y)}{n \Sigma x^2 - (\Sigma x)^2}$$

From data, $\Sigma x = 20$, $\Sigma y = 25$, $\Sigma x^2 = 120$

$\Sigma xy = 144$, $n = 4$

$$\Rightarrow a = \frac{25 \times 120 - 20 \times 144}{4(120) - 400} = 1.5$$

$$b = \frac{4 \times 144 - 20 \times 25}{4 \times 120 - 400} = 0.95$$

$$\therefore \boxed{y = 1.5 + 0.95x}$$

---

10) Given data,

$\{ P_1(0.40, 0.53),\ P_2(0.22, 0.38),\ P_3(0.35, 0.32)$

$P_4(0.26, 0.19),\ P_5(0.08, 0.41),\ P_6(0.45, 0.3) \}$

Single-Link Technique

First we calculate pair wise dissimilarity among data using Euclidian distance ($\sqrt{\|x - x\|^2}$)

In this case $d(P_1, P_2) = \sqrt{|x_{P_1} - x_{P_2}|^2 + |y_{P_1} - y_{P_2}|^2}$

∴ Distance Matrix is

P_1	0					
P_2	0.24	0				
P_3	0.22	0.15	0			
P_4	0.37	0.20	0.15	0		
P_5	0.34	0.14	0.28	0.29	0	
P_6	0.23	0.25	0.11	0.22	0.39	0
	P_1	P_2	P_3	P_4	P_5	P_6

We identify clusters with shortest distance in the matrix and merge them together. Re compute the distance matrix until all the points cluster into a single cluster

⇒

P_1	0				
P_2	0.24	0			
(P_3, P_6)	0.22	0.15	0		
P_4	0.37	0.20	0.15	0	
P_5	0.34	0.14	0.28	0.29	0
	P_1	P_2	(P_3, P_6)	P_4	P_5

→

P_1	0			
(P_2, P_5)	0.24	0		
(P_3, P_6)	0.22	0.15	0	
P_4	0.37	0.20	0.15	0
	P_1	(P_2, P_5)	(P_3, P_6)	P_4

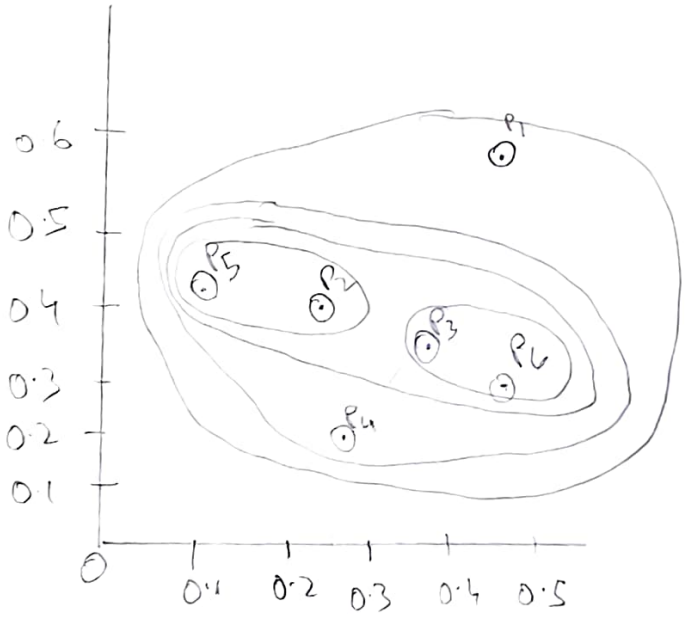
	P_1	(P_2,P_3,P_5,P_6)	P_4
P_1	0		
(P_2,P_3,P_5,P_6)	0.22	0	
P_4	0.37	0.15	0

→

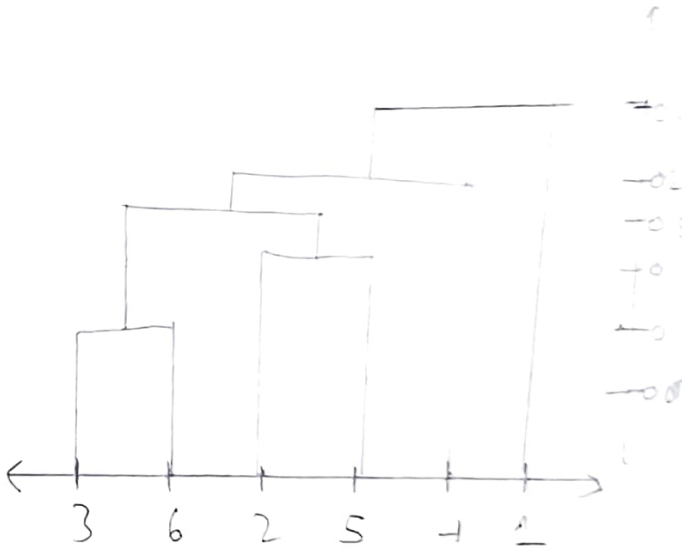
	P_1	(P_2,P_3,P_4,P_5,P_6)
P_1	0	
(P_2,P_3,P_4,P_5,P_6)	0.22	0

Now all points are clustered together

Clusters



Dendogram



Q11)

Given,

eps = 0.6

MinPts = 4

Data : {(1,2), (3,4), (2.5,4), (1.5,2.5), (3,5), (2.8,4.5), (2.5,4.5), (1.2,2.5), (1,3), (1,5), (1,2.5), (5,6), (4,3)}

[P_1 to P_{13} respectively]

Now, we need to find the neighbours of each of these points. (Distance $\leq$ eps). If the point has neighbours $\geq$ minpts $\Rightarrow$ Core. Else, if it lies in neighbourhood of core point $\Rightarrow$ Border. Else $\Rightarrow$ Noise

P_1 – P_8, P_{11}

P_2 – P_3, P_6

P_3 – P_2, P_6, P_7

P_4 – P_8, P_{11}

P_5 – P_6

P_6 – P_2, P_3, P_5, P_7

P_7 – P_3, P_6

P_8 – P_1, P_4, P_9, P_{11}

P_9 – P_8, P_{11}

P_{10} – NA

P_{11} – P_1, P_4, P_8, P_4

P_{12} – NA

P_{13} – NA

∴ Core points → P_6, P_8, P_{11}

Border points → $P_1, P_2, P_3, P_4, P_5, P_7, P_9$

Outliers → P_{10}, P_{12}, P_{13}

Now, every core point will be assigned to a new cluster unless some of the core points share neighbour points, They will be included in same cluster based on core point in their neighbourhood

Clusters

Cluster-1 : P_2, P_3, P_5, P_6, P_7 → (2.5, 4.5)

(3, 4) (2.5, 4) (3, 5) (3, 4.5)

Cluster-2 : $P_1, P_4, P_8, P_9, P_{11}$ → (1, 2.5)

(1, 2) (1.5, 2.5) (1.2, 2.5) (1, 3)

Terminologies :

1) Direct density reachable : Has a core point in its neighbourhood. Eg : $P_1(1, 2)$; $P_8(1.2, 2.5)$

2) Density reachable : Point is connected through a series of core points. Eg : $P_9(1, 3)$; $P_4(1.5, 2.5)$

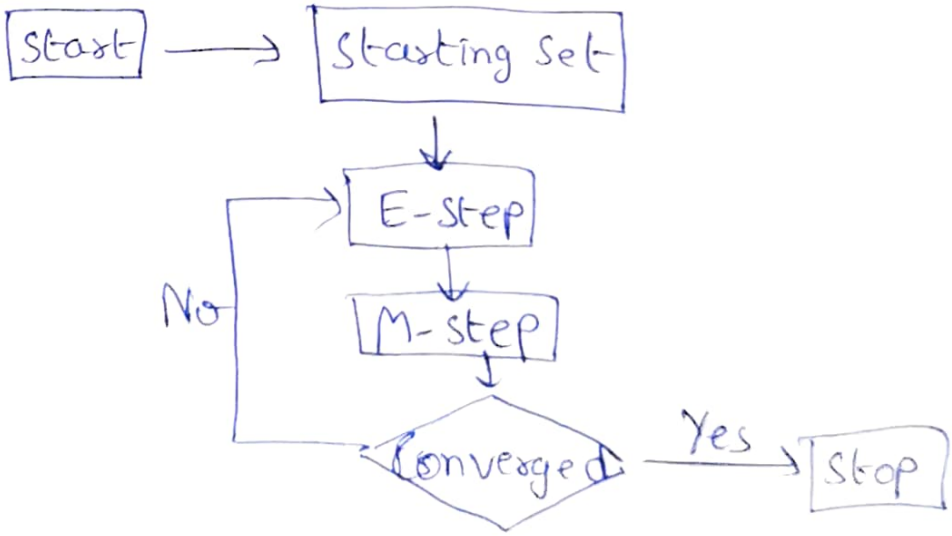
3) Density connected : Two points are called density connected if there is a core point which is density reachable from both points

12) Expectation-Maximization Algorithm

Expectation Maximization Algorithm is at the base of many unsupervised clustering algorithms. It is used also for latent variables (variables which aren't directly observable).

Algorithm:

* Consider random values as a set of starting parameters for μ, σ
* Expectation step (E-step): Using observed available data of dataset, estimate values of missing data.
* Maximization step (M-step): Complete data generated after expectation (E). Step is used in order to update parameters.
* Repeat E-step & M-step until convergence.



Let us take $X = \{x_1, x_2, \ldots x_n\}$

We have two clusters a & b and if we start with parameters of Gaussian (μ, σ^2) we can find

$$P(x_i|b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} e\left(-\frac{(x_i-\mu_b)^2}{2\sigma_b^2}\right)$$

$$P(b|x_i) = b_i = \frac{P(x_i|b)P(b)}{P(x_i|b)P(b) + P(x_i|a)P(a)}$$

$$\Rightarrow a_i = P(a|x_i) = 1 - b_i$$

Now we are gonna update μ & σ as,

$$\mu_b = \frac{b_1x_1 + b_2x_2 + \cdots + b_nx_n}{b_1 + b_2 + \cdots + b_n}$$

$$\sigma_b^2 = \frac{b_1(x_1-\mu_b)^2 + b_2(x_2-\mu_b)^2 + \cdots + b_n(x_n-\mu_b)^2}{b_1 + b_2 + \cdots + b_n}$$

Similarly we calculate μ_a, σ_a. We are going to repeat the above steps until μ and σ values are gonna converge.

priors: $P(b) = \frac{b_1 + b_2 + \cdots + b_n}{n}$ $\quad P(a) = 1 - P(b)$

1st iteration:

Prior Probability

Assuming $P(a) = P(b) = 0.5$, The 2 clusters are equally probable

$P(x_1|a) = 0.432$

$P(x_2|a) = 0.113$

$P(x_3|a) = 6.71 \times 10^{-3}$

$P(x_{10}|a) = 1.42 \times 10^{-29}$

$P(x_{11}|a) = 5.72 \times 10^{-36}$

$P(x_{12}|a) = 5.2 \times 10^{-43}$

$P(x_1|b) = 0.372$

$P(x_2|b) = 0.072$

$P(x_3|b) = 3.12 \times 10^{-3}$

$P(x_{10}|b) = 8.4 \times 10^{-31}$

$P(x_{11}|b) = 8.51 \times 10^{-37}$

$P(x_{12}|b) = 1.7 \times 10^{-39}$

$$P(a) = P(b) = 0.5$$

$$b_1 = \frac{0.432 \times 0.5}{0.432 \times 0.5 + 0.372 \times 0.5}$$

$b_2 = (0.61)(0.39)$

$b_3 = (0.677)(0.323)$

$b_{x=10} = (0.944)(0.056)$

$b_{x=11} = (0.957)(0.043)$

$b_{x=12} = (0.969)(0.031)$

New prior probability

$\sigma_a^2 = 3.48$

$\sigma_b = 1.54$

$P(a) = 0.7821$

$P(b) = 0.217$

$\mu_a = 7.53$

$\mu_b = 0.381$

2nd Iteration

$\sigma_a = 3.98$, $\mu_a = 7.53$, $\sigma_b = 1.54$, $\mu_b = 0.381$

$P(x_1|a) = 0.019$

$P(x_2|a) = 0.032$

$P(x_3|a) = 0.079$

$P(x_{10}|a) = 0.089$

$P(x_{11}|a) = 0.069$

$P(x_{12}|a) = 0.05$

$P(a) = 0.7821$

$P(b) = 0.217$

$P(x_1|b) = 0.07$

$P(x_2|b) = 3.1 \times 10^{-5}$

$P(x_3|b) = 1.42 \times 10^{-11}$

$P(x_{10}|b) = 0$

$P(x_{11}|b) = 0$

$P(x_{12}|b) = 0$

$a_1 = 0.493$ $a_2 = 0.99$ $a_3 = 0.99$

$a_{10} = 1$ $a_{11} = 1$ $a_{12} = 1$

$b_1 = 0.507$ $b_2 = 0.01$ $b_3 = 0.01$

$b_{10} = 0$ $b_{11} = 0$ $b_{12} = 0$

New,

$\mu_a = 6.70$ $\sigma_a = 4.45$

$\mu_b = 0.087$ $\sigma_b = 1.01$

Visualization

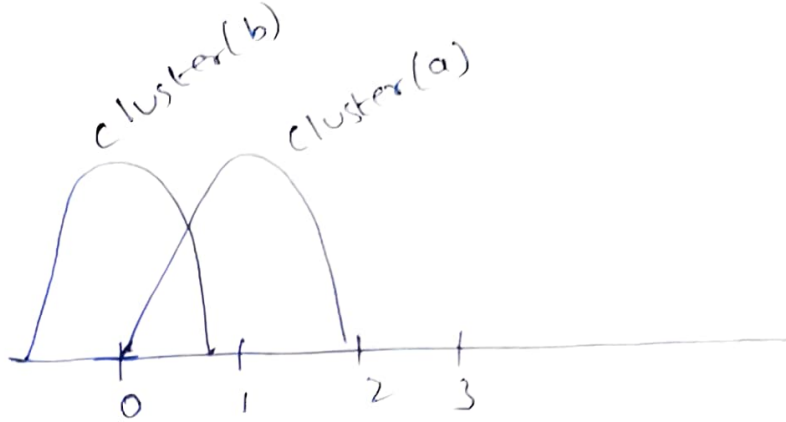
Initially

$\mu_1 = 0.6$

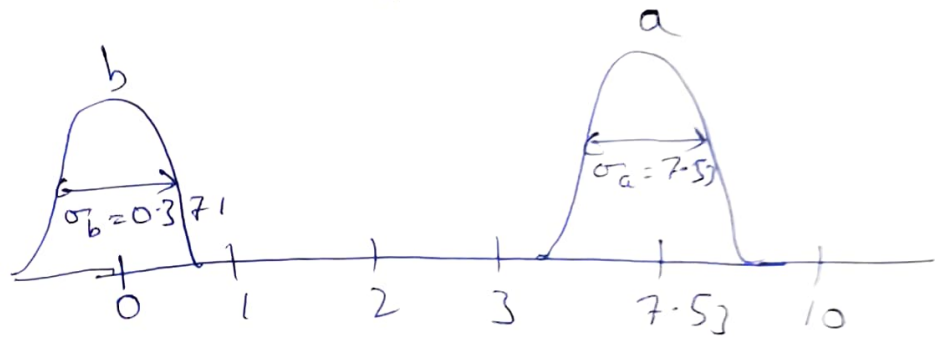
$\mu_2 = 0.4$

$\sigma_1 = 0.82$

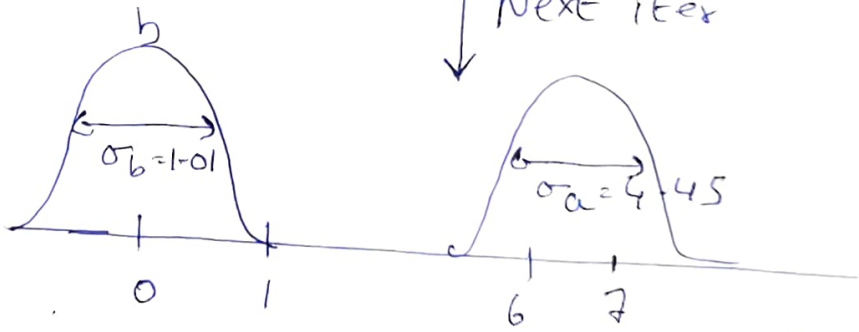
$\sigma_2 = 0.82$



After 2 iter



Next iter



∴ As seen in diagram, the cluster is improving by iteration

It will come to saturation after multiple iteration by changing appropriate μ and σ for both the probability distribution