SUPERVISED CLUSTERING

Akshay Deshmukh - 3417

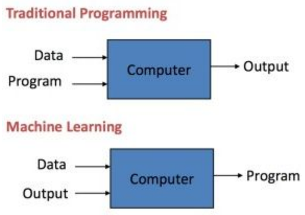
Pune Institute of Computer Technology

April 4, 2018

What is Machine Learning?

"A breakthrough in machine learning would be worth ten Microsoft's."
— Bill Gates, Former Chairman, Microsoft

- Machine Learning is getting computers to program themselves.
- If programming is automation, then machine learning is automating the process of automation.



Key Elements of Machine Learning

- Representation : Various machine learning techniques
- Evaluation : Metrics accuracy, loss, cost
- Optimization : Preprocessing, normalization

Types of Machine Learning

- Supervised Learning
- Unsupervised Learning
- Semi-supervised Learning
- Reinforcement Learning

What is Clustering?

- Clustering is the method of identifying similar groups of data in a data set.
- Entities in each group are comparatively more similar to entities of that group than those of the other groups.
- Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups.
- In simple words, the aim is to segregate groups with similar traits and assign them into clusters.

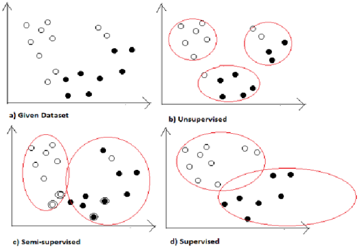
Applications of Clustering

Clustering has a large no. of applications spread across various domains. Some of the most popular applications of clustering are:

- Recommendation engines
- Social network analysis
- Search result grouping
- Anomaly detection

Types of Clustering

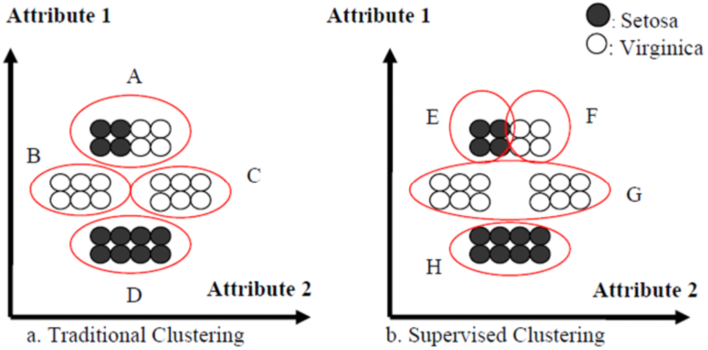
- **Unsupervised Clustering:** It is a technique using some functions, for example, that minimizes the distances in cluster to keep cluster in-tight.
- **Supervised Clustering:** It is a technique which is applied on classified examples with the objective of identifying cluster's that have high probability density with respect to single class.
- **Semi-Supervised Clustering:** It is applied to enhance an algorithm by using side information in clustering process.



WHY SUPERVISED CLUSTERING?

- Supervised Clustering is a type of Clustering. It assumes that the examples are classified and has a goal of determining clusters that have high probability function.
- Clustering is basically unsupervised to find objects belonging to same group.
- A supervised clustering algorithm which uses a fitness function which maximizes the purity of the clusters while keeping the number of clusters low would produce clusters.

WHY SUPERVISED CLUSTERING? (CONT.)



Fitness Function for Supervised Clustering

- Fitness function characterizes supervised clustering.
- The fitness function's used by supervised clustering are different from traditional clustering. It evaluates based on following two criteria:
  - Class Impurity
  - Number of clusters k

Fitness Function for Supervised Clustering (cont.)

- The fitness function which is used in the supervised clustering is as follows:

$$q(X) = Impurity(X) + \beta * Penalty(k) \tag{1}$$

where,

$$Impurity(X) = percentage\ of\ minority\ example\ of\ other\ classes \tag{2}$$

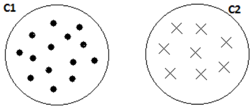
$$Penalty(k) = \begin{cases} sqrt((k-c)/n) & \text{if } k \geq c \\ 0, & \text{if } k < c \end{cases}$$

- With n being total number of examples and c being number of classes in dataset. The parameter β,

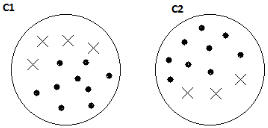
$$(0 < \beta <= 2.0) \tag{3}$$

determines the penalty that is associated with number of clusters $\boldsymbol{k}$. Higher values of β indicates larger penalties.

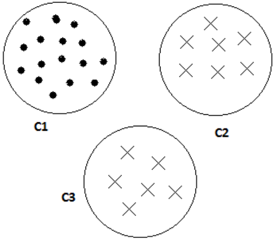
Fitness Function for Supervised Clustering (cont.)



Impurity = 0 , Penalty = 0



impurity(X) = 4 / 20 = 0.2 = 20%

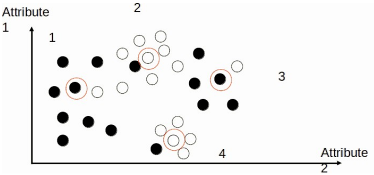


Impurity = 0

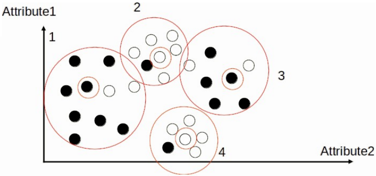
Representative-Based Algorithms

- The target of RBA is to find ***k*** representatives that fit the dataset properly.
- We will find distance of testing sample from these ***k*** representatives and assign to nearest representative.
- ***The goal is to find subset Pr of P such that the clustering obtained through it should minimize our q(x).***

ALGORITHMS IN SUPERVISED CLUSTERING (CONT.)



Initial Distribution



After Clustering

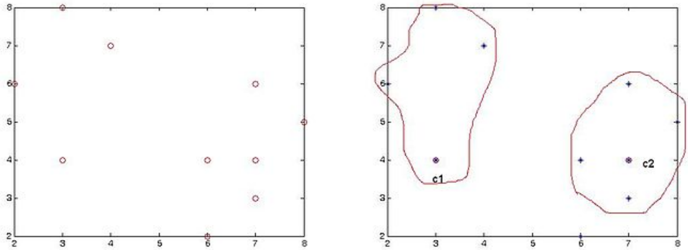
Partitioning Around Medoids (PAM)

- It is a clustering algorithm called as k-medoid which aims to find ***k*** represents objects among objects in the data set minimizing the fitness function as stated above.
- Initially a random is chosen but as move further more representatives are added.

$$Tightness(X) = 1 \mid objects \mid \Sigma\ distance(object, medoid(object)\ ) \tag{4}$$

- Where medoid(object) is the medoid representative of cluster that object belongs to. The number of clusters is input to this algorithm. PAM is again divided into two algorithms: SWAP and BUILD

Algorithms in Supervised Clustering (cont.)



Initial Distribution   After Clustering

Supervised Partitioning Around Medoids (SPAM)

- It is a slight variation of the algorithm PAM that uses the fitness function ***q(X)*** instead of ***Tightness(X)***.
- The number of clusters k is an input parameter to the algorithm. It consists of two sub-algorithms. 1)SBUILD 2)SSWAP.
- Intially members of the most frequent class in the data set as the first representative.
- After that, it repeatedly and greedily adds to the current set of representatives a non-representative object.
- In SSWAP for optimization the non-medoid is replaced with medoid to check wheather q(x) is minimized

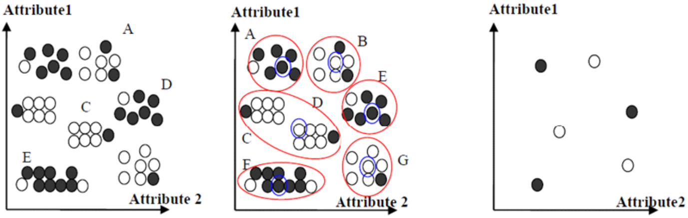
Top Down Splitting Algorithm (TDS)

- It is a very simple algorithm that aims at creating clusters quickly.
- It starts by assigning all data objects to a root cluster.
- Then the Meddoids with most first and second patterns are selected.
- If q(x) is minimized then it is selected otherwise rejected

Importance of Supervised Clustering

- Create background knowledge for a dataset
  - It shows how instances of a particular class distribute in the attribute space; this information is of value for "discovering" subclasses of particular classes.
  - Statistical summaries can be created for each cluster.
  - Meta attributes, such as various radius's, distances between representatives, etc. can be generated, and their usefulness for enhancing classifiers can be explored.

- Dataset compression and editing
  - The objective of dataset editing is to remove examples from a training set in order to enhance the accuracy of a classifier.



a. Dataset given

b. Dataset clustered using supervised clustering.

c. Dataset edited using supervised clustering

Literature Survey Table

Year	Conference/ Journal Paper	Author Name	Algorithms used	Salient features
2004	Supervised Clustering - Algorithms and benefits	Christoph F. Nidal Zeidat, and Zhenghong Zhao	PAM, SPAM, TDS	Supervised clustering and its features, Proposed algorithms for supervised clustering, Applications of Supervised Clustering
2017	Multi-label Classification Systems by the Use of Supervised Clustering	N.Rastin, M.Z.Jahromi ,M.Taheri	Supervised Clustering	Proposed Multilabel classifiation using Supervised Clustering, Intially performed using unsupervised clustering, Datasets used were From Mulan library
2001	Fuzzy Hypersphere Neural Network Classifier	U.V. Kulkarni ,T.R. Sontakke	Learning algorithm for Fuzzy Hypershpere	Proposed Fuzzy Hypersphere using Supervised Clustering, Concept of membership function for a cluster was introduced, Evaluation was done on Iris dataset with 93% accuracy, Gives 100% training accuracy
2017	Pruned Fuzzy Hypersphere Neural Network (PFHSNN) for Lung Cancer Classification	D. N. Sonar ,U. V. Kulkarni	Learning algorithm for fuzzy hypersphere with Pruning process.	Proposed Fuzzy with Pruning Process, Gives 100% training accuracy, Dataset was from JSRT database Japan

Fuzzy Hypersphere

- **Fuzzy Logic** is a technique which is basically based of degrees of truth rather than general true of false.
- For example, 38 of tallness or 64 of shortness.
- Fuzzy gives probabilistic results rather than binary results which can be useful.
- **Fuzzy hypersphere** is basically the n-dimensional space which maps to lowest abstraction level.
- In simple manner all the n dimensional features combine together to form a hypersphere.
- This hypersphere is created by using fuzzy logic which we know that it gives continuous values rather than discrete results.

Architecture of FHNN

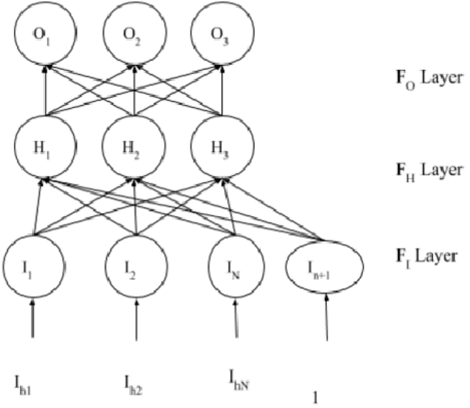


Figure: Architecture of FHNN

Notations and Symbols:-

k = 1, 2........k denotes the classes of samples
O = $O_1, O_2....O_n$ where n is no of centroid clusters formed
P = Total number of patterns of all classes
a_k = Total number of patterns belonging to class k
t_k = P - a_k denoting total number of patterns belonging to other than class k
I is the input pattern matrix
r = $r_1, r_2, ...r_n$ radii associated with centroid clusters

Algorithm FHNN()

Step 1: Compute the inter-class distance for every pattern in the class k A new matrix A^k is created which is of shape $a_k * t_k$. It will store interclass distance of every pattern in every class to that of other classes. Different types of distances are used to compute the distance. Mainly Manhanntan, Eucledian distances are used.

$$A^k = [dist(I_i - I_j)] \quad (5)$$

where, i = 1,2,......,a_k
j = 1,2,......,t_k

Step 2: Calculate intra-class distance for each class $\boldsymbol{k}$ A new matrix is used named B^k of shape $a_k * a_k$

$$B^k = [dist(I_i - I_j)] \text{ of size } a_k * a_k \quad (6)$$

where, i = 1,2,.......,a_k
j = 1,2,......,a_k

Step 3: Calculate min-interclass distance matrix for each class k

$$W^k = \min_{j=1}^{t_k} (A_{ij})^k \tag{7}$$

where, i = 1,2,.......,a_k
j = 1,2,.......,t_k

Step 4: Find the centroid that covers maximum patterns
Step 4.1: Initialize a count array of size $a_k * (a_k + 1)$
$pcount^k[a_k * (a_k + 1)]$

Consider each pattern in class k as initial centroid and its radius as minimum of interclass distance which we computed from W^K, that is,

$$tempr^k{}_i = w^k{}_i \tag{8}$$

Learning Algorithm (cont.)

Step 4.2: Check whether this centroid covers its intraclass neighboring points

for j <-1 to a_k do
    if B_j^k <= $tempr_i^k$ do
        increment $pcount^k$[i,a_k]
        mark as visited $pcount^k$[i,j]<-1
    end
end

Step 4.3: Choose that centroid which covers maximum patterns of its class

index <- maximum(pcount)
Add that centroid in resultant O matrix
increment pointer of O

Learning Algorithm (cont.)

Step 4.4: Update that centroid radius to its max-intraclass distance within its cluster

```
max1 <- 0
for j <- 1 to αk do
    if pcount^k[index,j] is 1 and max1 <= B^k[index,j] do
        max1 = B^k[index,j]
    end
end
```

Step 4.4.1:Check whether given centroid covers only himself

```
if maximum(pcount) = 0 do
    update its radius as half of minimum interclass
    distance, i.e., r[pointer]<-W^k[index]/2
else
    r[pointer]<-max1
end
```

Step 4.5: Associate label to that cluster the same as that of its centroid

Step 4.6: Mark all these points as visited which are covered in the cluster

for j<-1 to a_k do
    if $pcount^k$[index,j] =1 do
        mark as visited I^k[j]<-1
    end
end

Step 5: Decrement a_k by visited number of patterns in cluster

$$a_k \leftarrow a_k\text{-maximum(pcount)}$$

Step 6: Repeat the steps from 1 to 5 until a_k becomes zero and for each class k

Step 7:Halt

Initial Membership Function

In FHNN the membership function is defined as follows :

$$d(P_h, O_j, r_j) = f(dist, r_j) \tag{9}$$

where, P_h is h^{th} input pattern P = P_1, P_2,.....P_h,....P_n
O_j is j^{th} centroid of a cluster O=O_1,O_2,...O_h...O_n,
dist is distance between O_j and P_h,
r_j is radius of O_j.

$$f(dist, r_j) = \begin{cases} 1 & \text{, if } dist \leq r_j \\ r_j/dist & \text{, otherwise} \end{cases}$$

Modified Membership Function

The modified membership function is as follows:-

$$d(P_h, O, r) = f(D, r) \tag{10}$$

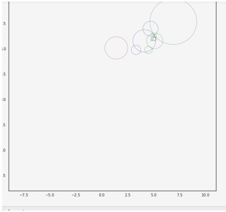
where, P_h is h^{th} input pattern P = P_1, P_2,.....P_h,....P_n ,
O_j is centroid array of a cluster O=O_1,O_2,...O_h...O_n,
D is distance array between

$$(P_h, O_j) \ \forall \ O_j \epsilon \ O \tag{11}$$

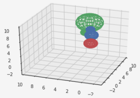
r is radius array for every centroid.

$$f(D, r) = max(r_j/D_j) \ \forall \ (r_j, D_j) \ \epsilon \ (r, D) \tag{12}$$

IRIS DATASET



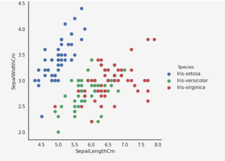

```
ax.view_init(azim=200)

plt.show()
```




```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt

def drawSphere(xCenter, yCenter, zCenter, r):
    #draw sphere
```

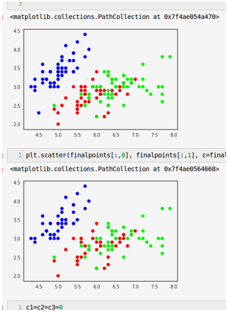
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
5	6	5.4	3.9	1.7	0.4	Iris-setosa
6	7	4.6	3.4	1.4	0.3	Iris-setosa
7	8	5.0	3.4	1.5	0.2	Iris-setosa
8	9	4.4	2.9	1.4	0.2	Iris-setosa
9	10	4.9	3.1	1.5	0.1	Iris-setosa

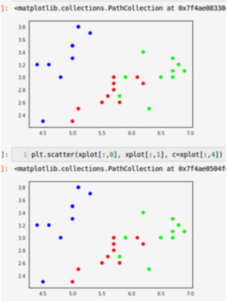
```
In [6]: dat['Species'].value_counts()          #finding count of each clas

Out[6]: Iris-virginica     50
        Iris-versicolor    50
        Iris-setosa        50
```



IMPLEMENTATIONS AND RESULTS

IRIS DATASET



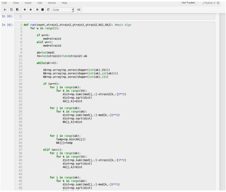


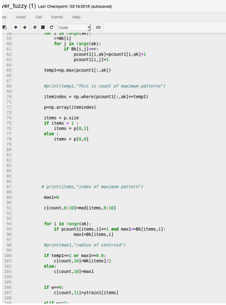
IRIS Dataset

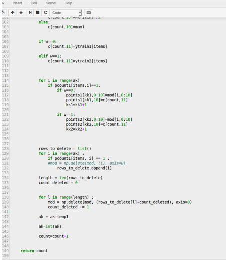
K-fold	Training Samples No.	Testing Samples No.	Accuracy
1	120	30	100%
2	120	30	96.66%
3	120	30	96.66%
4	120	30	93.33%
5	120	30	93.33%
Average = 96%			

Table: Iris Dataset results

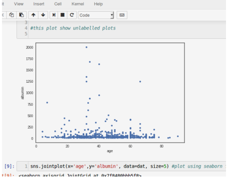
Liver DATASET

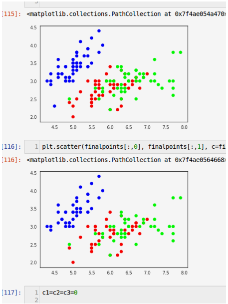


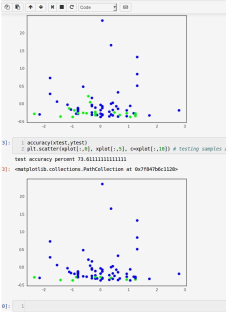




Liver DATASET







LIVER Dataset

K-fold	Training Samples No.	Testing Samples No.	Accuracy
1	504	70	73.611%
2	504	70	69.44%
3	504	70	68.04%
4	504	70	70.83%
5	504	70	69.44%
6	504	70	72.11%
Average = 70.65% ≈ 71%			

Table: Liver Dataset results

PIMA Dataset

K-fold	Training Samples No.	Testing Samples No.	Accuracy
1	572	192	75.52%
2	572	192	69.791%
3	572	192	68.75%
4	572	192	79.685%
Average = 73.43%			

Table: Pima Dataset results

GLASS Dataset

K-fold	Training Samples No.	Testing Samples No.	Accuracy
1	160	50	82%
2	160	50	74.3%
3	160	50	76.6%
4	160	50	73.8%
Average = 76.67%			

Table: Glass Dataset results

Thank You...