CS60050 MACHINE LEARNING
ASSIGNMENT 02 — K-MEANS CLUSTERING

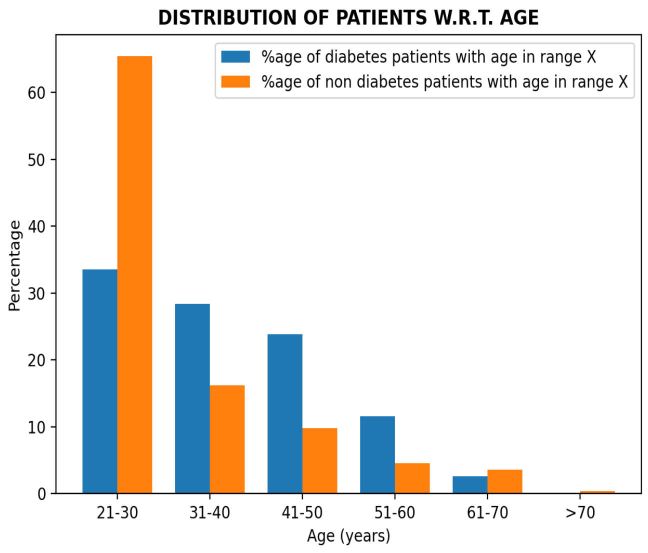
Group 23
Amrta Chaurasia (19EE10004)
Nakul Aggarwal (19CS10044)

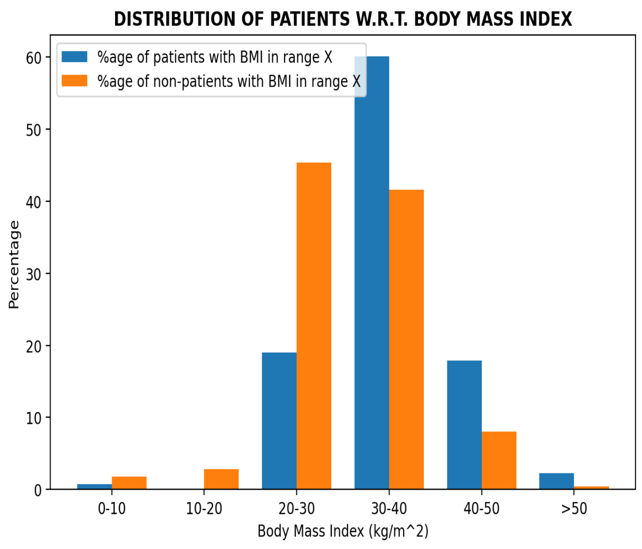
PREFACE

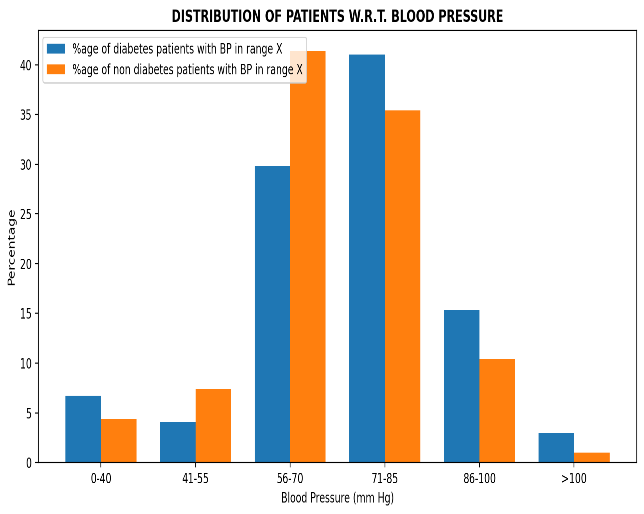
➢ **Dataset used: Pima Indians Diabetes Database (PIDD)**

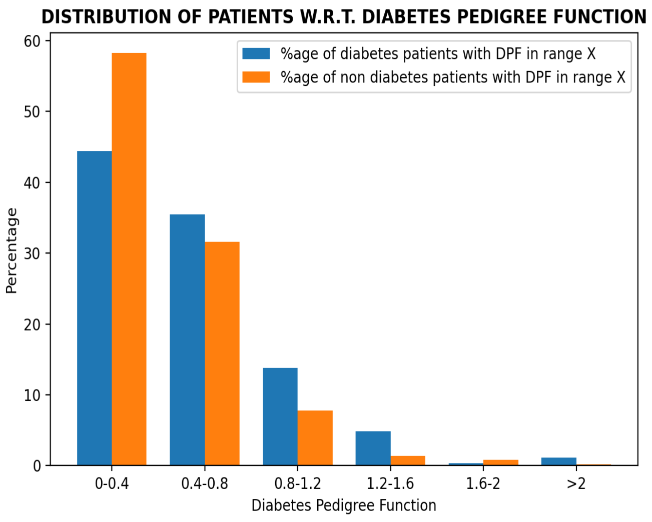
UNDERSTANDING PIDD DATASET

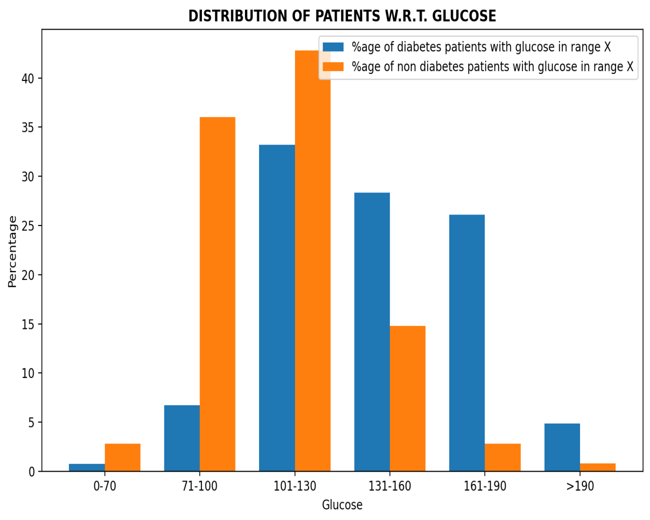
In data science and machine learning, visualization of datasets becomes of paramount importance as it gives a clear idea of what the information means by giving it a visual context through maps or graphs. This makes data more natural for the human mind to comprehend and therefore makes it easier to identify trends, patterns and outliers within large datasets. Accordingly, it helps the engineers to design their project pipeline and choose re-sampling and pre-processing techniques, if any, to rework their dataset.

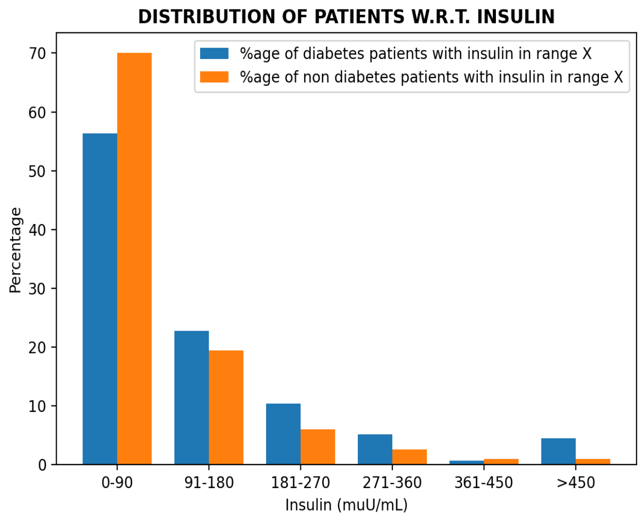


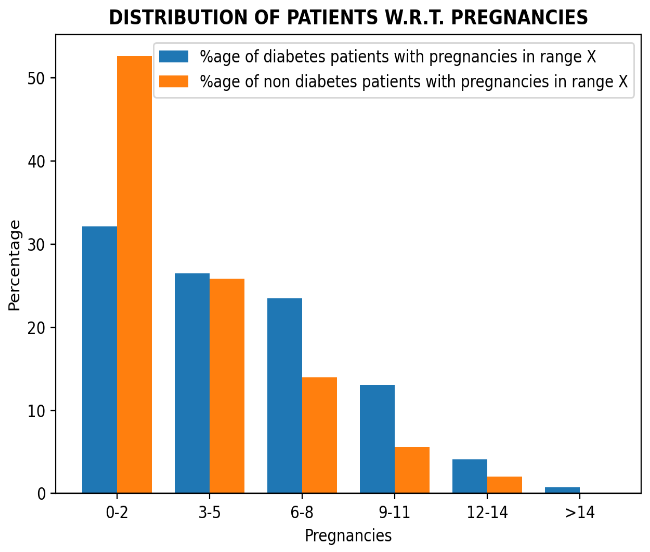


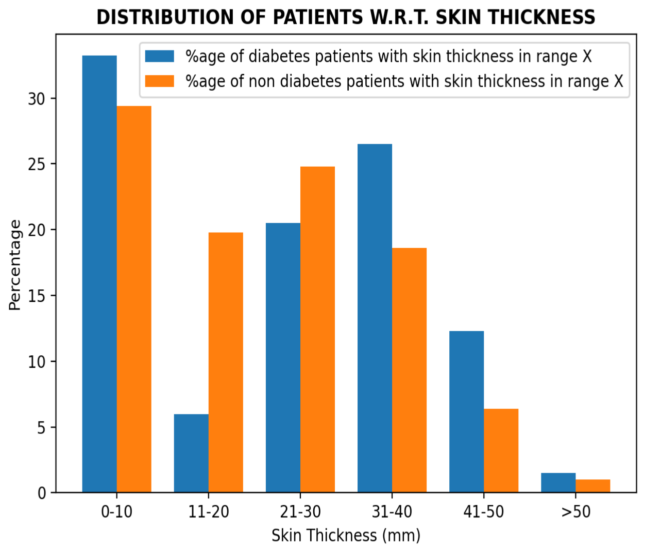












	CLASS DISTRIBUTION	
	COUNT	**PROPORTION**
POSITIVE	268	34.90 %
NEGATIVE	500	65.10 %

• Older people (age more than 50 years) are more likely to have diabetes than not.
• Higher body mass index and blood pressure are more common among the diabetes patients than the non-patients. Majority of the diabetes patients have BMI in the range of 30-40, that is considered as obsese, both among males and females.
• Diabetes patients are much more likely to have higher concentrations of glucose and insulin in their blood than the non-patients.
• Diabetes patients are more likely to have higher skin thickness (more than 30 mm) than the non-patients.

Similarly, numerous other important information can be deduced from the visualized data very easily. This also helps in getting an intuition about the diversity of the dataset, both with respect to the target values and the attributes values.

REWORKING PIDD DATASET

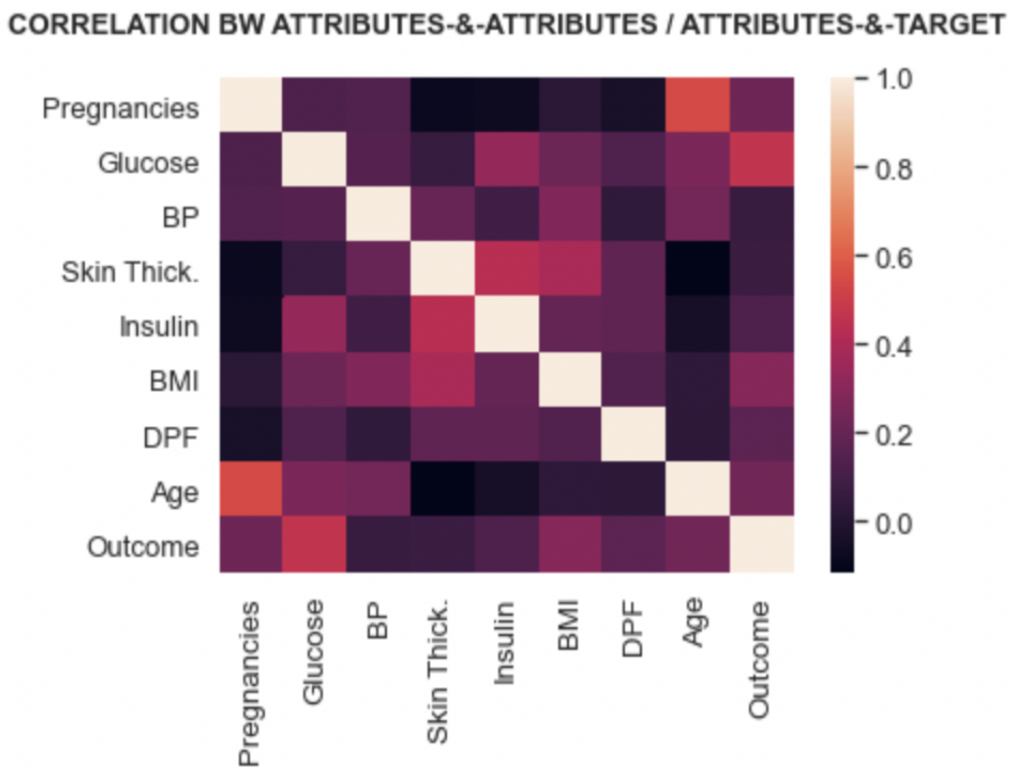
ATTRIBUTE	MIN VALUE	MAX VALUE	ATTRIBUTE	MIN VALUE	MAX VALUE
Age	21	81	*Glucose*	0	199
Body Mass Index	0.0	67.1	*Insulin*	0	846
Blood Pressure	0	122	*Pregnancies*	0	17
Diabetes Pedigree Function	0.078	2.420	*Skin Thickness*	0	99

Feature scaling in machine learning is one of the most critical steps during the pre-processing of data. Scaling can make a difference between a weak machine learning model and a better one. The major need of feature scaling is because the machine learning algorithms just see numbers — if there is a vast difference in the range, say few ranging in thousands and others in the tens, then it makes the underlying assumption that higher ranging numbers have superiority of some sort. So these more significant numbers start playing a more decisive role while training the model.

The $PIDD$ dataset has 8 attributes. As is clear from the table above, 3 attributes range in the hundreds, 4 attributes in tens and the remaining one attribute is always less than 10. So the 8 attributes belong to 3 different numerical scales and while training, the attributes *blood pressure*, *glucose* and *insulin* ranging in hundreds will overpower the others. This domination is more prominent in algorithms that involve computation of distances between the feature vectors, as is obviously required in k-means clustering. Feature scaling is essential for machine learning algorithms that calculate distances between data because if not scaled, the feature with a higher value range starts dominating when calculating distances. If one of the features has a broad range of values, the distance might majorly be governed by this particular feature. Therefore, the range of all features should be normalized so that each feature contributes approximately proportionately to the final distance.

We have used the **standard scaler** that assumes data is normally distributed within each feature and scales them such that the distribution is centered around 0, with a standard deviation of 1 (standardization). Centering and scaling happen independently on each feature by computing the relevant statistics on the data samples. So for any value x of an attribute att, the new scaled value will be $x_{new} = \frac{x-\mu_{att}}{\sigma_{att}}$, where μ_{att} and σ_{att} are respectively the mean and standard deviation of values of the attribute att observed in the dataset.

ATTRIBUTE	MEAN μ	S.D. σ	ATTRIBUTE	MEAN μ	S.D. σ
Age	33.24	11.75	*Glucose*	120.89	31.95
Body Mass Index	31.99	7.88	*Insulin*	79.80	115.17
Blood Pressure	69.10	19.34	*Pregnancies*	3.84	3.37
Diabetes Pedigree Function	0.47	0.33	*Skin Thickness*	20.54	15.94



The above heatmap illustrates the correlation between every pair of attributes and also between the attributes and the target value, $outcome$. It gives us insights into the linear correlation between various pairs of properties, and also which attributes are more relevant to the target value. For example, the correlation of $outcome$ with $glucose$ is higher than with any other attribute. $Skin\ thickness$ and $blood\ pressure$ have very less correlation with the $outcome$. Besides, the pair of attributes $insulin\ \&\ skin\ thickness$ and $age\ \&\ pregnancies$ also have significant correlation between them.

An important preprocessing task before k-$means\ clustering$ is re-sampling the attributes such that no two of them share a very high correlation coefficient. Highly correlated variables are not useful for ML classifications because they represent more or less the same characteristic of a class. So highly correlated variables are nothing but noise. Fortunately, though some pairs of attributes in our dataset are more correlated than the others, no pair of attributes has an absolute correlation coefficient of more than 0.55, that is surely tolerable. Therefore, no re-sampling/redundancy of attributes is needed to be taken care of.

PART 01

TASK SUMMARY

K-$Means\ clustering$ is performed on the PIDD dataset for k (cluster size) given by the user.

PROCEDURE

The value of k is taken as input from the user through a prompt. For the given k, the PIDD data samples are clustered into k clusters. The initial k centers for the clustering algorithm are derived using the k-means++ heuristic. The clustering of the data samples into k clusters is comprehended through the $contingency\ matrix$ and the $pair\ confusion\ matrix$.

BACKGROUND

CONTINGENCY MATRIX

Contingency matrix reports the intersection cardinality for every cluster-class pair. Since the PIDD dataset has 2 classes namely, $patient$ and $non\,patient$, for k clusters there will be $2k$ such pairs. Contingency matrices allow to examine the spread of each class across the constructed clusters and vice-versa. More generally, they are used in drawing a parallel between any two independent partitions of data samples, that might even be the partitions produced by two different runs of the clustering algorithm. Here, the two partitions we use to build the contingency matrix are the class-partition (ground truth values) as given in the dataset and the clustering-induced partition.

PAIR-CONFUSION MATRIX

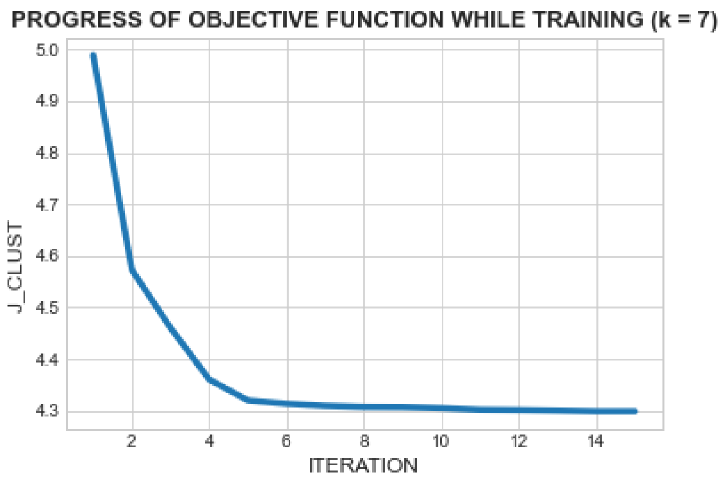
The pair-confusion matrix is a $2x2$ similarity matrix between two partitions computed by considering all pairs of samples and counting pairs that are assigned into the same or into different partitions under the two given partitions, one of which might be $true$ and the other $predicted$ partition. Here the two partitions can either both be two independent clustering-induced partitions or one of them can be the class-partition (ground truth values) as given in the dataset. Here, the two partitions we use to build the pair-confusion matrix are the class-partition (ground truth values) as given in the dataset that can be treated as the true partition and the clustering-induced partition that is treated as the predicted partition.

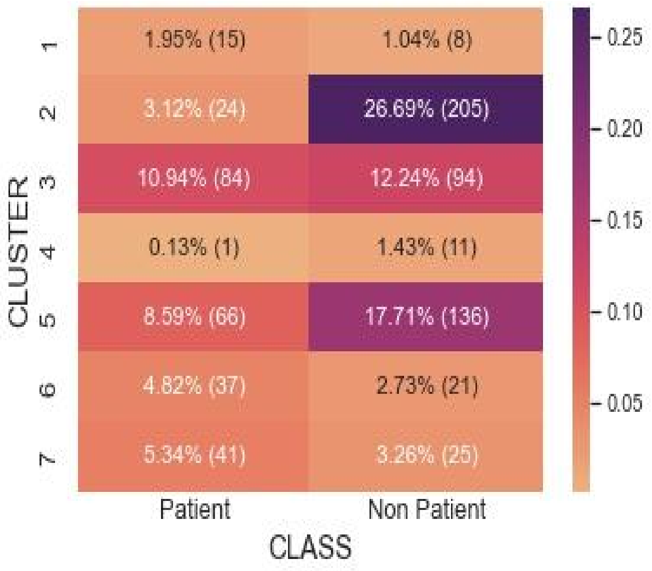
$$C = \begin{bmatrix} C_{00} & C_{01} \\ C_{10} & C_{11} \end{bmatrix}$$

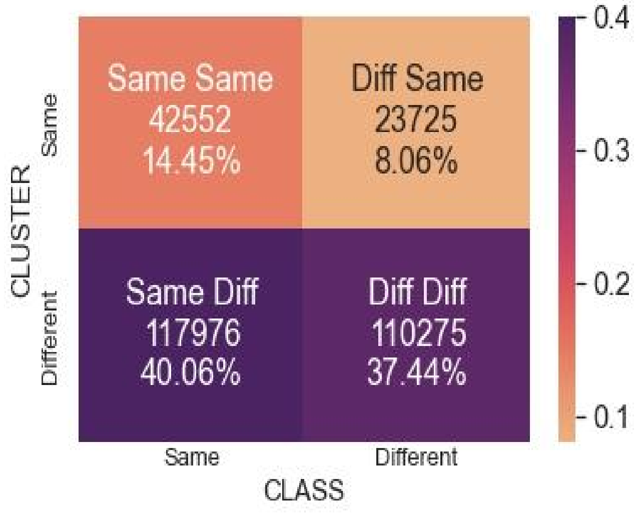
C_{00} : number of pairs belonging to the same class that also belong to the same cluster.
C_{01} : number of pairs belonging to different classes that belong to the same cluster.
C_{10} : number of pairs belonging to the same class that belong to different clusters.
C_{11} : number of pairs belonging to different classes that also belong to different clusters.

SAMPLE RUNS

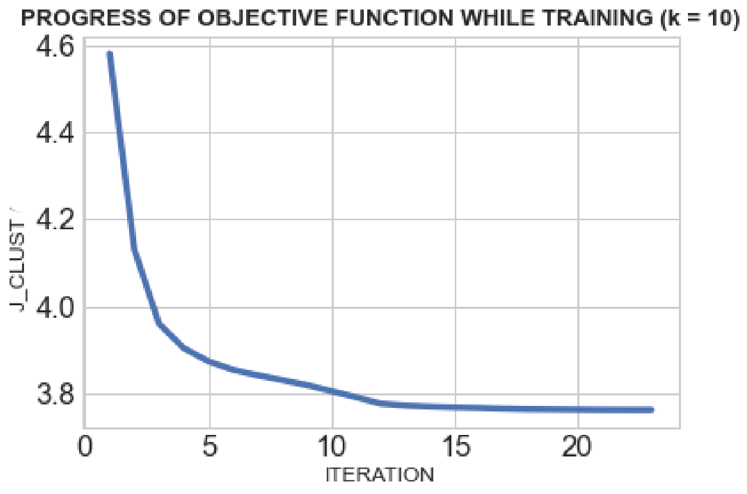
- $k = 7$

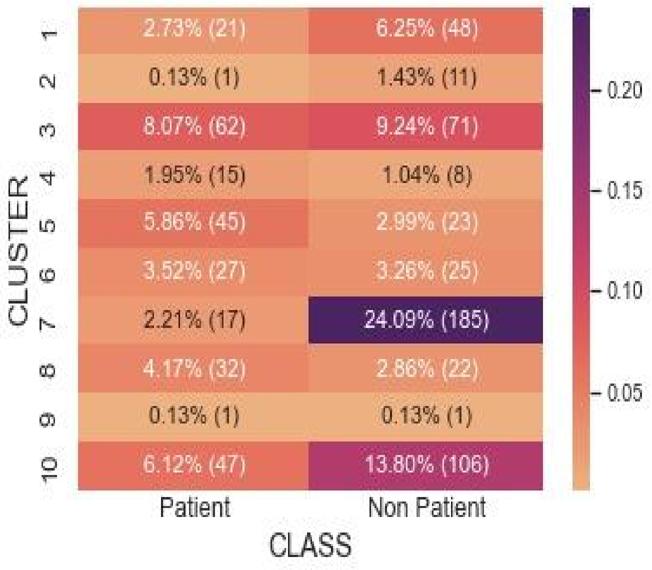


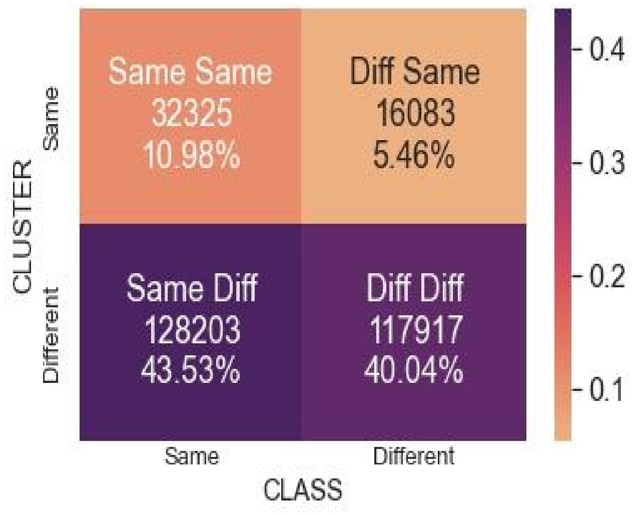

CLUSTER
1
2
3
4
5
6
7
1.95% (15)
1.04% (8)
3.12% (24)
26.69% (205)
10.94% (84)
12.24% (94)
0.13% (1)
1.43% (11)
8.59% (66)
17.71% (136)
4.82% (37)
2.73% (21)
5.34% (41)
3.26% (25)
Patient
Non Patient
CLASS
0.25
0.20
0.15
0.10
0.05


CLUSTER
Same
Different
Same Same
42552
14.45%
Diff Same
23725
8.06%
Same Diff
117976
40.06%
Diff Diff
110275
37.44%
Same
Different
CLASS
0.4
0.3
0.2
0.1

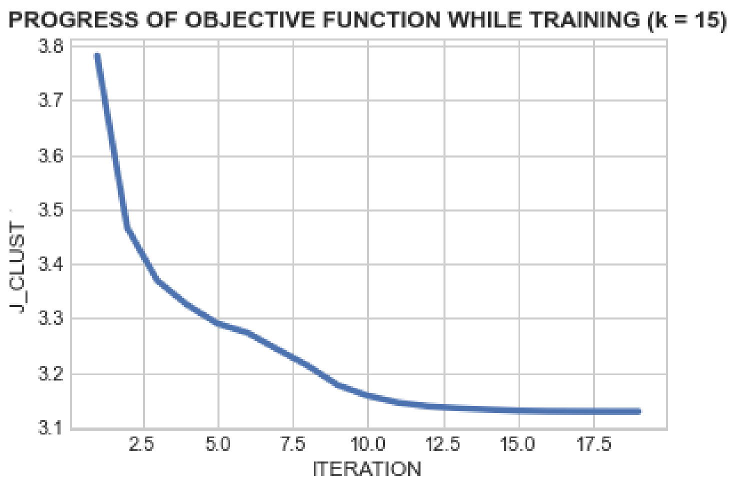
• $k = 10$

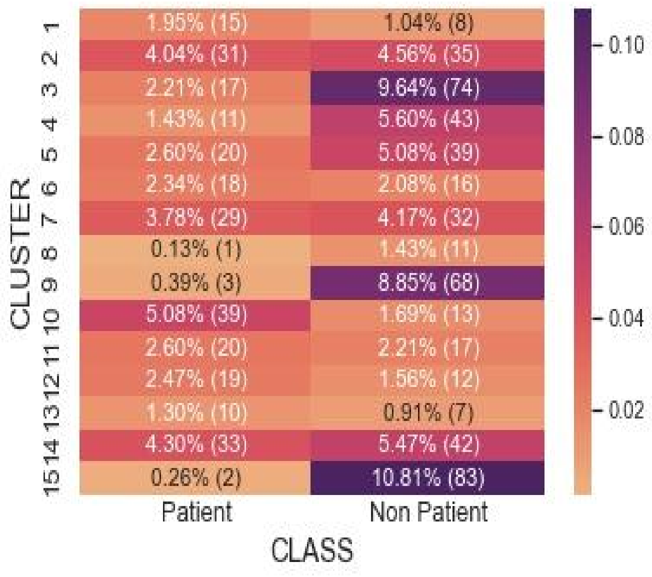

PROGRESS OF OBJECTIVE FUNCTION WHILE TRAINING (k = 10)
J_CLUST
4.6
4.4
4.2
4.0
3.8
0
5
10
15
20
ITERATION

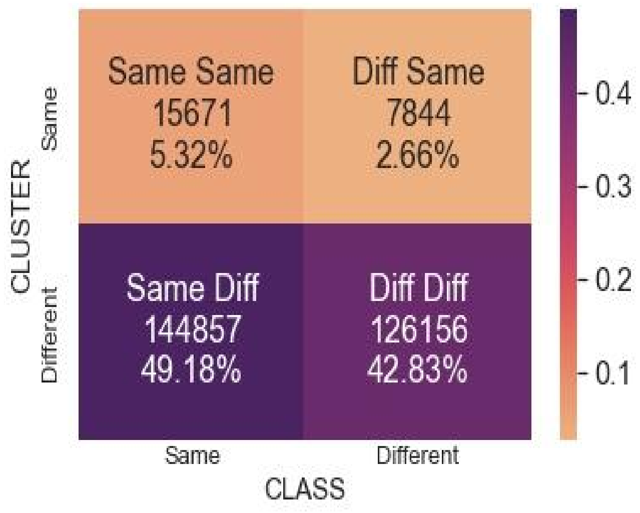

CLUSTER
1
2
3
4
5
6
7
8
9
10
2.73% (21)
6.25% (48)
0.13% (1)
1.43% (11)
8.07% (62)
9.24% (71)
1.95% (15)
1.04% (8)
5.86% (45)
2.99% (23)
3.52% (27)
3.26% (25)
2.21% (17)
24.09% (185)
4.17% (32)
2.86% (22)
0.13% (1)
0.13% (1)
6.12% (47)
13.80% (106)
Patient
Non Patient
CLASS
0.20
0.15
0.10
0.05


CLUSTER
Same
Different
Same Same
32325
10.98%
Diff Same
16083
5.46%
Same Diff
128203
43.53%
Diff Diff
117917
40.04%
Same
Different
CLASS
0.4
0.3
0.2
0.1

• $k = 15$







OBSERVATIONS

The value of the objective function J_{CLUST} is observed to decrease monotonically. The rate of decrease is high in the beginning and gradually slows down as it achieves the optimal value.

COMMENTS

The function that implements the $k\text{-}means\ clustering$ algorithm can work with 3 initialization settings to initialize the k cluster centers.

$RANDOM$: The k initial centers are chosen randomly from the N data samples.

$K\text{-}MEANS\ ++$: The $k\text{-}means\ ++$ heuristic is used to initialize the k cluster centers.

$CUSTOM$: The initial k cluster centers are passed as arguments to the function.

The default setting of the algorithm is $K\text{-}MEANS\ ++$ and all the results are obtained under this setting, unless explicitly mentioned otherwise. The reason behind choosing a heuristic-based initialization routine rather than random initialization will be clear in PART O4.

The objective function J_{CLUST} is the $mean\ squared\ error\ (MSE)$ between the data samples and the centers of the respective clusters they belong to.

The termination of the iterative clustering algorithm is conditioned on the change of J_{CLUST} after each iteration. If the change is less than 10^{-6} then the algorithm is terminated.

PART 02

TASK SUMMARY

Performance of the clustering obtained in the PART 01 is provided using available ground truth values and without using the ground truth values. Among the metrics that use the ground truth values RI, ARI, MI, NMI, $homogeneity$, $completeness$ and $V\,measure$ are evaluated. Among the metrics that do not use the ground truth values $silhouette\ coefficient$, $Calinski\ Harabasz\ Index$ and $Davies\ Bouldin\ Index$ are evaluated.

PROCEDURE

For the value of k given in the last part and hence the obtained clusters and cluster centers, the values of the 10 evaluation metrics are computed and printed, out of which 7 use the ground truth values as given in the dataset and the remaining 3 do not. A function is written for each of the evaluation metric that takes as input the cluster assignments and/or cluster centers and returns the value of the evaluation metric, rounded upto 4 decimal places.

BACKGROUND

RI (RAND INDEX)

Given the knowledge of the ground truth class assignments and clustering assignments of the same samples, the (adjusted or unadjusted) $rand\ index$ is a function that measures the similarity of the two assignments, ignoring permutations.

$$RI = \frac{a+b}{C_2^{n_{samples}}}$$

a : number of pairs of data samples that are in the same class and in the same cluster.
b : number of pairs of data samples that are in different classes and in different clusters.
$n_{samples}$: total number of samples in the dataset.
Higher the RI, better is the partition.

ARI (ADJUSTED RAND INDEX)

The $rand\ index$ does not ensure to obtain a value close to 0 for a random labelling. The $adjusted\ rand\ index$ accounts for this correction.

$$ARI = \frac{\Sigma_{ij} C_2^{n_{ij}} - [\Sigma_i C_2^{a_i} \,.\, \Sigma_j C_2^{b_j}] \,/\, C_2^{n}}{\frac{1}{2}[\Sigma_i C_2^{a_i} + \Sigma_j C_2^{b_j}] - [\Sigma_i C_2^{a_i} \,.\, \Sigma_j C_2^{b_j}] \,/\, C_2^{n}} \;;\; 1 \le i \le k \quad 1 \le j \le 2$$

n_{ij} : number of data samples belonging to the i^{th} cluster and the j^{th} class.
a_i : number of data samples belonging to the i^{th} cluster.
b_j : number of data samples belonging to the j^{th} class.
n : total number of samples in the dataset.
Higher the ARI, better is the partition.

HOMOGENEITY (h), COMPLETENESS (c), V-MEASURE (v)

$Homogeneity$ objectifies the notion of each cluster containing only members of a single class. $Completeness$ objectifies the notion of all members of a given class being assigned to the same cluster. $V\text{-}Measure$ evaluates the agreement of two independent assignments on the same dataset.

$$h = 1 - \frac{H(C|K)}{H(C)} \quad c = 1 - \frac{H(K|C)}{H(K)} \quad v = \frac{2\,.\,h\,.\,c}{h+c}$$

Higher the $h/c/v$, better is the partition.

DAVIES BOULDIN INDEX (DB)

If the ground truth labels are not known, the $Davies - Bouldin\ index$ can be used to evaluate the model, where a lower index relates to a model with better separation between the clusters. This index signifies the average similarity between clusters. Zero is the lowest possible score. Values closer to zero indicate a better partition.

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \qquad DB = \frac{1}{k}\sum_{i=1}^{k} max_{i \neq j}(R_{ij})$$

s_i : average distance between each point of cluster i and its center (also known as cluster diameter).
d_{ij} : distance between the centers of the clusters i and j.

Lower the DB, better is the partition.

SAMPLE RUNS

k	WITH GROUND TRUTH							W/O GROUND TRUTH		
	RI	ARI	MI	NMI	H	C	V	SC	CHI	DBI
7	0.519	0.083	0.130	0.079	0.139	0.055	0.079	0.167	109.220	1.472
10	0.510	0.076	0.139	0.074	0.149	0.049	0.074	0.169	94.923	1.398
15	0.482	0.036	0.184	0.079	0.197	0.049	0.079	0.130	83.718	1.610

H : $homogeneity$ C : $completeness$ V : $v\,measure$ SC : $silhouette\ coefficient$
CHI : $Calinski\ Harabasz\ Index$ DBI : $Davies\ Bouldin\ Index$

OBSERVATIONS

As the cluster size k is changed from one value to the other, some of the evaluation metrics are observed to get better and some are observed to get worse. Like for the external indices, when k is changed from 7 to 10, $homogeneity$ increases (becomes better) but $normalized\ mutual\ information$ reduces (becomes worse). This means that on changing k from 7 to 10, though each cluster becomes more affiliated to contain members of the same class (*more homogenous*), yet the class and the cluster assignments tend to agree less with one another (*less mutual information*).
Similarly for internal indices, when k is changed from 7 to 10, the $DB\ Index$ and $silhouette\ coefficient$ become better but the $CH\ Index$ becomes worse. In conclusion, different metrics/ indices can show different trends with varying cluster size k.

EXPLANATION

Different metrics/indices underline different properties or aspects of partitions induced by the clustering algorithm. When k is changed from one value to the other, some properties of these partitions might improve and others might deteriorate. This trade-off between several aspects of the clustering implies that *not all metrics may happen to achieve their best values at the same cluster size k.*

CONSEQUENCES

Determining the most optimal value of cluster size k involves choosing an internal index (like $silhouette\ coefficient$ or $Calinski\ Harabasz\ Index$) and evaluating it over a range of values of k. PART O2 verifies that since different internal indices evaluate different aspects related to a partition, all internal indices may not achieve an optimal value at the same cluster size. A direct consequence of this is that choosing different internal indices can result in different optimal values of k. This can be observed in PART O3.

PART 03

TASK SUMMARY

The most suitable value of cluster size, k is determined by varying k and evaluating internal indices (like $silhouette\ coefficient$ and $Calinski\ Harabasz\ Index$) on the obtained clusters and cluster centers.

PROCEDURE

The cluster size k is varied between the values low and $high$. This is done three times with different evaluation metrics to determine the optimal values of k in different cases.

$Silhouette\ Coefficient$: For each value of k between low and $high$, clustering is performed 3 times and the average of the silhouette coefficients obtained in each pass is recorded as the $average\ silhouette\ coefficient$ for that value of k. Finally $average\ silhouette\ coefficient\ v/s\ k$ graph is plotted and the optimal value of k at which the average value of the metric is the highest is reported.

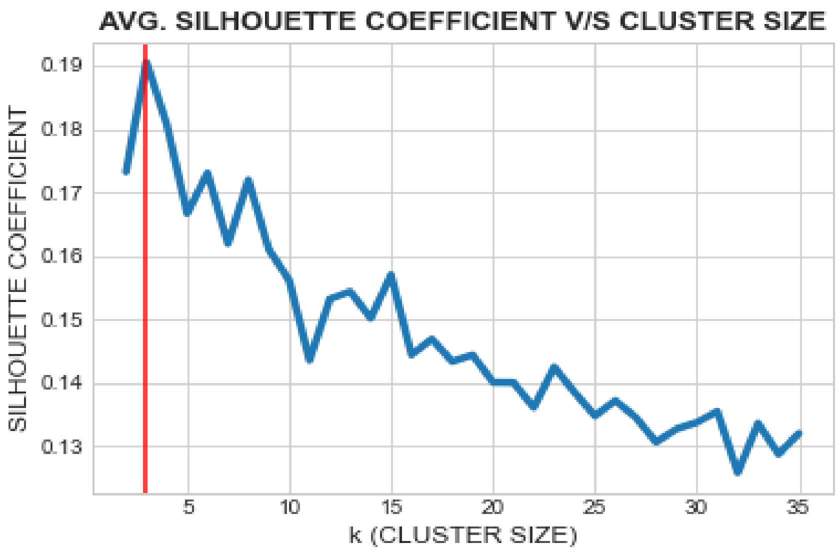
$Calinski\ Harabasz\ Index$: For each value of k between low and $high$, clustering is performed 3 times and the average of the Calinski Harabasz Indices obtained in each pass is recorded as the $average\ Calinski\ Harabasz\ Index$ for that value of k. Finally $average\ Calinski\ Harabasz\ Index\ v/s\ k$ graph is plotted and the optimal value of k at which the average value of the metric is the highest is reported.

$Wang's\ Method\ of\ Cross\text{-}Validation$: For each value of k between low and $high$, $Wang's\ Method\ of\ Cross\text{-}Validation$ was performed with 10 permutations and $|S_1|:|S_2|:|S_3| = 3:3:4$ split of the data samples and the $average\ disagreement\ ratio$ is recorded. Finally, a graph of $average\ disagreement\ ratio\ v/s\ k$ is plotted and the optimal value of k at which the average value of disagreement ratio is the least is reported.

RESULTS

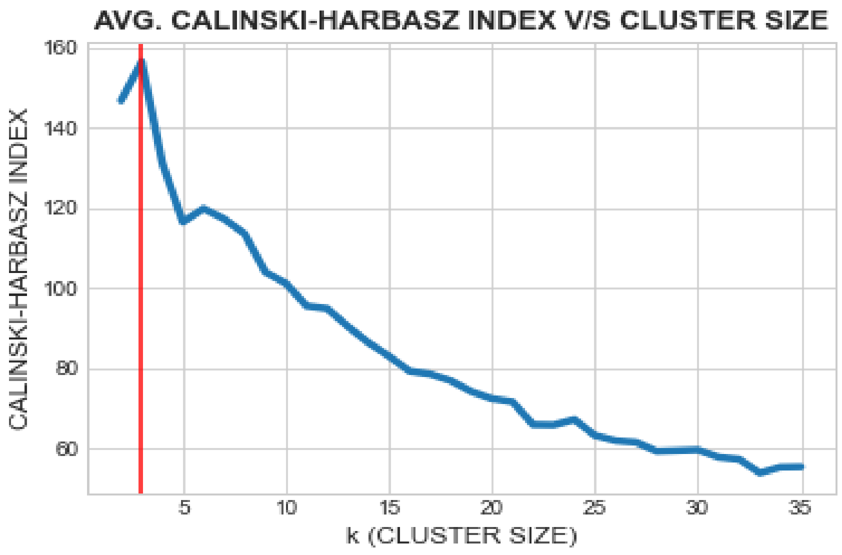
• **FINDING OPTIMAL k WITH $Silhouette\ Coefficient$**

k	AVG. SILHOUETTE COEFFICIENT	k	AVG. SILHOUETTE COEFFICIENT	k	AVG. SILHOUETTE COEFFICIENT
1	—	13	0.1544	25	0.1348
2	0.1732	14	0.1502	26	0.1372
3	**0.1905**	15	0.1570	27	0.1345
4	0.1807	16	0.1444	28	0.1306
5	0.1667	17	0.1469	29	0.1327
6	0.1731	18	0.1433	30	0.1337
7	0.1619	19	0.1444	31	0.1355
8	0.1719	20	0.1401	32	0.1258
9	0.1610	21	0.1400	33	0.1336
10	0.1561	22	0.1361	34	0.1287
11	0.1435	23	0.1425	35	0.1320
12	0.1532	24	0.1385		



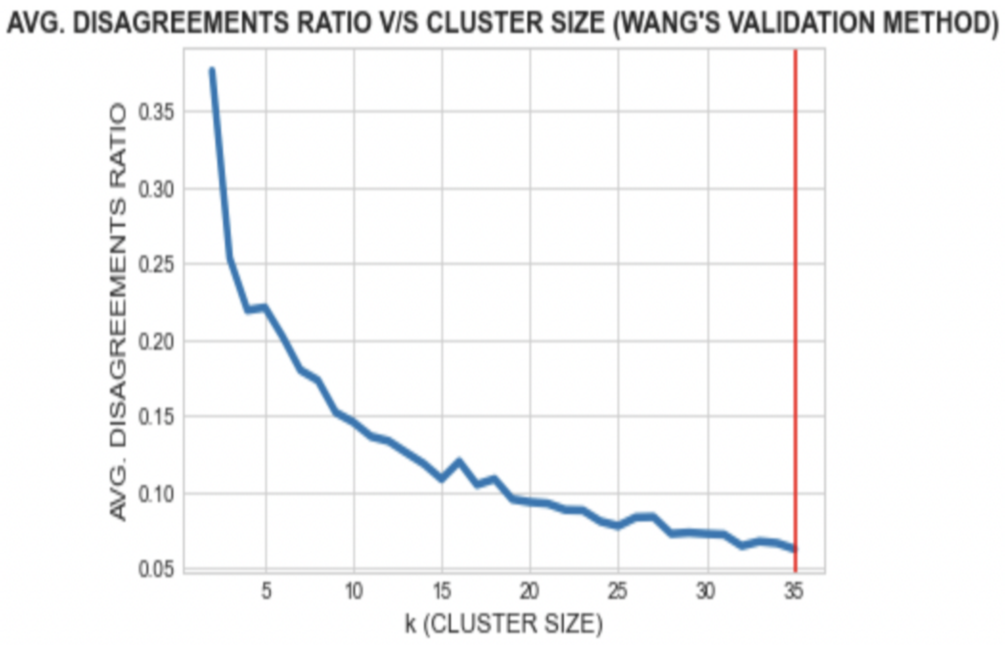
- **FINDING OPTIMAL k WITH** *Calinski Harabasz Index*

k	AVG. CH INDEX	k	AVG. CH INDEX	k	AVG. CH INDEX
1	—	13	90.5015	25	63.2858
2	146.7543	14	86.4328	26	62.0001
3	**156.4842**	15	83.0644	27	61.6018
4	131.1420	16	79.3063	28	59.3623
5	116.4756	17	78.5768	29	59.5282
6	119.8422	18	76.9571	30	59.6951
7	117.2645	19	74.2167	31	57.9014
8	113.5172	20	72.4651	32	57.3739
9	103.9890	21	71.6806	33	53.9401
10	101.2007	22	65.9792	34	55.3961
11	95.5564	23	65.9026	35	55.4742
12	94.9180	24	67.2659		



• **FINDING OPTIMAL k WITH** $Wang's\ Cross\text{-}Validation\ Method$

k	AVG. RATIO OF DISAGREEMENT	k	AVG. RATIO OF DISAGREEMENT	k	AVG. RATIO OF DISAGREEMENT
1	—	13	0.1260	25	0.0778
2	0.3766	14	0.1186	26	0.0835
3	0.2537	15	0.1086	27	0.0838
4	0.2195	16	0.1202	28	0.0727
5	0.2213	17	0.1049	29	0.0735
6	0.2021	18	0.1087	30	0.0727
7	0.1802	19	0.0954	31	0.0722
8	0.1734	20	0.0935	32	0.0647
9	0.1524	21	0.0926	33	0.0677
10	0.1460	22	0.0884	34	0.0666
11	0.1366	23	0.0881	**35**	**0.0627**
12	0.1336	24	0.0808		



CONCLUSION

• **FINDING OPTIMAL k WITH** $Silhouette\ Coefficient$

The highest $avg.\ silhouette\ coefficient$ was observed for $k = 3$ (when k was varied from 2 to 35), after which the metric decreases irregularly. So the most optimal value of cluster size k based on the $silhouette\ coefficient$ is 3.

• **FINDING OPTIMAL k WITH** $Calinski\ Harabasz\ Index$

The highest $avg.\ CH\ Index$ was observed for $k = 3$ (when k was varied from 2 to 35), after which the metric decreases almost monotonically. So the most optimal value of cluster size k based on the $CH\ Index$ is 3.

• **FINDING OPTIMAL k WITH $Wang's\ Cross\text{-}Validation\ Method$**

The least $average\ disagreements\ ratio$ was observed for $k = 35$ when k was varied from 2 to 35. The value decreases almost monotonically with increasing k. So the most optimal value of cluster size k in this interval based on $Wang's\ Cross\ Validation\ Method$ is 35.

As was inferred in the **CONSEQUENCES** of **PART 03**, different evaluation techniques can yield different optimal values of k. The trends in the second ($CH\ Index$) and third ($Wang's\ Method$) plots suggest a more consistent variation in the index with increasing k, whereas in the first plot ($silhouette\ coefficient$), the trends are more irregular.

Choice of optimal k value of 3 might be better than 35 because a smaller number of clusters is better to identify and interpret simpler similarities. For a larger number of clusters, it will become harder to interpret the characteristics of each cluster. Besides, since the data samples are originally grouped into 2 classes — $patient$ and $non\text{-}patient$, 3 is a more natural choice over 35!

COMMENTS

The algorithm was run at the $K\text{-}MEANS\ ++$ initialization setting for all the sub-parts. The value of k was varied from 2 to 35 for each of the internal indices.

PART 04

TASK SUMMARY

The deviation/stability of the clustering outcome with respect to different random initializations of k cluster centers is studied through the $Test\text{-}A$ as given in the assignment. In order to reduce the variation in the evaluation metric chosen in the $Test\text{-}A$, a heuristic-based initialization routine is selected and $Test\text{-}A$ is re-performed and the deviation/stability of the clustering outcome are reported and compared with the former results.

PROCEDURE

A function that implements the $Test\text{-}A$ as given in the assignment step-by-step and works for both the $random$ and the $heuristic$ settings is written. $Normalized\ Mutual\ Information$ (NMI) is chosen as the evaluation metric. $Standard\ deviation$ of the evaluation metric is computed to quantify the variability in the clustering outcome. The function takes an additional argument $setting$ as input that indicates whether the test has to be performed in a $random$ setting, where the initial cluster centers are assigned randomly, or in a $heuristic$ setting, where the initial cluster centers are determined using some heuristic-based initialization routine. The heuristic that we have chosen to initialize the k cluster centers under the $heuristic$ setting is k-means++.

The value of cluster size k is taken as input through a prompt. First $Test\text{-}A$ under $random$ and then under $heuristic$ setting is performed.

TEST-A ($Random\ Setting$)

1. Select k random points from the dataset.
2. Randomly split remaining data into $train$ and $test$ sets in $80:20$ ratio.
3. Run function $KMeans$ written in $PART\ 01$ under $CUSTOM$ setting with initial centers selected in step 1.
4. Label the data points in the $test$ set with the k cluster centers.
5. Evaluate NMI metric for the cluster labellings of the $test$ set.
6. Repeat steps 2-5 50 times and record the average value of NMI.
7. Repeat steps 1-6 50 times and report the standard deviation of the average NMI value.

TEST-A $(Heuristic\ Setting)$

1. Select a random $seed$ for the k-means++ heuristic from the dataset.
2. Randomly split remaining data into $train$ and $test$ sets in $80:20$ ratio.
3. Run function $KMeans$ written in $PART\ 01$ under $K\text{-}MEANS\ ++$ setting with $seed$ selected in step 1.
4. Label the data points in the $test$ set with the k cluster centers.
5. Evaluate NMI metric for the cluster labellings of the $test$ set.
6. Repeat steps 2-5 50 times and record the average value of NMI.
7. Repeat steps 1-6 for 50 random seeds and report the standard deviation of the average NMI value.

Repeat the tests for multiple arbitrary values of k and check if the trend is consistent, that is, if the variation in the clustering outcome is minimized in all the tested values of k or not.

RESULTS

k	$RANDOM\ SETTING$		$HEURISTIC\ SETTING$	
	$Mean\ (NMI_{avg})$	$StdDev\ (NMI_{avg})$	$Mean\ (NMI_{avg})$	$StdDev\ (NMI_{avg})$
4	0.0957234	0.0126940	0.0932570	0.0048182
7	0.0876546	0.0043408	0.0882053	0.0038775
10	0.0971058	0.0047453	0.0947282	0.0035170
13	0.1015221	0.0045867	0.0988716	0.0032091

OBSERVATIONS

• A variation in the clustering outcome is observed if the cluster centers are initialized with random points $(random\ setting)$ in different executions, as is indicated by a significant standard deviation in the average value of NMI over multiple executions.

• The variation $(standard\ deviation)$ in the average value of the evaluation metric NMI is reduced considerably (for all the tested values of k) when instead of randomly selecting the k initial centers, the centers are initialized through the k-means++ heuristic, seeded by a randomly chosen data point from the dataset.

• The increase in stability achieved by using a heuristic-based initialization might trade-off in the form a reduced mean value of the average NMI metric.

CONCLUSION

Using a heuristic like k-means++ for initialization of centers, rather than random initialization, can improve the stability of the clustering outcome of the same dataset over multiple runs of the algorithm.

COMMENTS

The results and conclusions of this part explain why K-MEANS++ is kept as the default initialization setting in the $KMeans$ function written in the $PART\ 01$.

APPENDIX

ABBREVIATIONS

%AGE	Percentage
ARI	Adjusted Rand Index
AVG	Average
BMI	Body Mass Index
BP	Blood Pressure
CHI	Calinski Harabasz Index
DBI	Davies Bouldin Index
DPF	Diabetes Pedigree Function
MI	Mutual Information
MSE	Mean Squared Error
NMI	Normalized Mutual Information
PIDD	Pima Indians Diabetes Database
RI	Rand Index
STDDEV	Standard Deviation

CODE EXECUTION

- requirements.txt

Specifies the python packages and modules required to run the code.
Ensure all the necessary dependencies of required versions and latest version of Python3 are available with the following command

```
>>> pip3 install -r requirements.txt
```

- diabetes.csv

Contains the dataset used in this assignment.

- GRP23_A02_IMPLEMENTATION.ipynb

Jupyter Notebook written in Python containing all the code relevant to the assignment.

Launch jupyter notebook through terminal using the command: `jupyter notebook`
Then open this notebook to view the code written and run to see the output results.

Note: Please make sure before running the notebook, the *PIMA Indians Diabetes Database* is present by the name of *diabetes.csv* and *GRP23_A02_IMPLEMENTATION.ipynb are* in the same directory.

(other details are present in README.txt)

REFERENCES

- Adjusted Rand Index (Wikipedia)
- Contingency Table
- Evaluating Clustering Performance
- Feature Scaling (Wikipedia)
- Heatmaps on Seaborn Python
- Matplotlib Python Documentation
- Numpy Python Documentation
- Pima Indian Diabetes Database