Ha

MODULE 3

UNSUPERVISED LEARNING, SUPPORT VECTOR MACHINES AND RESAMPLING

SYLLABUS

- Principal Component Analysis
- Clustering algorithms, practical issues in clustering
- Support vector classifiers and support vector machines
- Resampling methods: cross-validation and bootstrapping

IMPORTANT QUESTIONS

- Explain different types of clustering. What are the practical issues in clustering?
- What is Support Vector Machine? How classification is done using SVM?
- Explain different types of Resampling methods.
- What is SVM? Explain Different kernel tricks in SVM.
- Write a short note on Maximal Margin Hyperplanes. (MMH).
- Suppose that our task is to cluster data points into two clusters. Let the data points are {2, 4, 10, 12, 3, 20, 30, 11, 25}. Let 2 and 4 are initial cluster centroids. Apply Two rounds of k-means algorithm and find a set of clusters. Use Euclidean distance as the measure.
- Apply K-Means Clustering on data points {(1,1), (2,1), (2,3), (3,2), (4,3), (5,5)} to cluster them into two clusters. Initially select (2,1) and (2,3) as cluster centres.
- Discuss any two clustering techniques.

DIMENSIONALITY REDUCTION

In statistical and machine learning, dimensionality reduction or dimension reduction is the process of reducing the number of variables under consideration by obtaining a smaller set of principal variables. Dimensionality reduction may be implemented in two ways.

- ***Feature selection***

  In feature selection, we are interested in finding k of the total of n features that give us the most information and we discard the other (n−k) dimensions.

- ***Feature extraction***

  In feature extraction, we are interested in finding a new set of k features that are the combination of the original n features. These methods may be supervised or unsupervised depending on whether or not they use the output information. The best known and most widely used feature extraction methods are Principal Components Analysis (PCA) and Linear Discriminant Analysis (LDA), which are both linear projection methods, unsupervised and supervised respectively.

Why dimensionality reduction is useful

There are several reasons in reducing dimensionality.

- In most learning algorithms, the complexity depends on the number of input dimensions, d, as well as on the size of the data sample, N, and for reduced memory and computation, we are interested in reducing the dimensionality of the problem. Decreasing d also decreases the complexity of the inference algorithm during testing.
- When an input is decided to be unnecessary, we save the cost of extracting it.
- Simpler models are more robust on small datasets. Simpler models have less variance, that is, they vary less depending on the particulars of a sample, including noise, outliers, and so forth.
- When data can be explained with fewer features, we get a better idea about the process that underlies the data, which allows knowledge extraction.
- When data can be represented in a few dimensions without loss of information, it can be plotted and analyzed visually for structure and outliers.

PRINCIPAL COMPONENT ANALYSIS

- Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

- Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms without extraneous variables to process.

- So, to sum up, the idea of PCA is simple — reduce the number of variables of a data set, while preserving as much information as possible.

Steps involved in PCA:

1. Standardization of the data
2. Computing the covariance matrix
3. Calculating the eigenvectors and eigenvalues
4. Computing the Principal Components
5. Reducing the dimensions of the data set

Standardization of the data

- Standardization is all about scaling your data in such a way that all the variables and their values lie within a similar range.

- Let we have 2 variables in our data set, one has values ranging between 10-100 and the other has values between 1000-5000. It is obvious that the output calculated by using these predictor

- Variables is going to be biased since the variable with a larger range will have a more obvious impact on the outcome.

- Mathematically, standardization can be done by subtracting the mean and dividing by the standard deviation for each value of each variable. Once the standardization is done, all the variables will be transformed to the same scale.

$$z = \frac{value - mean}{standard\ deviation}$$

Computing the covariance matrix

- Mathematically, a covariance matrix is a *pxp* matrix, where *p* represents the dimensions of the data set.
- Each entry in the matrix represents the covariance of the corresponding variables.
- Consider a case where we have a 3-Dimensional data set with variables *a* and *b*, the covariance matrix is a 3*x* 3 matrix as shown below:

$$\begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix}$$

Covariance Matrix for 3-Dimensional Data

- Since the covariance of a variable with itself is its variance (Cov(a,a)=Var(a)), in the main diagonal (Top left to bottom right) we actually have the variances of each initial variable. And since the covariance is commutative (Cov(a,b)=Cov(b,a)), the entries of the covariance matrix are symmetric with respect to the main diagonal, which means that the upper and the lower triangular

portions are equal.

- The covariance matrix is not more than a table that summaries the correlations between all the possible pairs of variables.

Calculating the Eigenvectors and Eigenvalues

- Eigenvectors and eigenvalues are the linear algebra concepts that we need to compute from the covariance matrix in order to determine the *principal components* of the data.
- Eigen vectors and eigen values are the mathematical constructs that must be computed from the covariance matrix in order to determine the principal components of the data set.
- If your data set is of 5 dimensions, then 5 principal components are computed, the first principal component stores the maximum possible information and the second one stores the remaining maximum info and so on.
- For each dimension d we have eigen vector and eigen value associated with it. Reading together we have for each principal component we have a eigen vector and eigen value associated.

Computing the Principal Components

- Once we have computed the Eigenvectors and eigenvalues, all we have to do is order them in the descending order,
- The eigenvector with the highest eigenvalue is the most significant and thus forms the first principal component.
- The principal components of lesser significances can thus be removed in order to reduce the dimensions of the data.

PCA Algorithm

Step 1. Data

We consider a dataset having n features or variables denoted by $X_1, X_2, \ldots, X_n$. Let there be N examples. Let the values of the i-th feature X_i be $X_{i1}, X_{i2}, \ldots, X_{iN}$

Features	Example 1	Example 2	$\cdots$	Example N
X_1	X_{11}	X_{12}	$\cdots$	X_{1N}
X_2	X_{21}	X_{22}	$\cdots$	X_{2N}
$\vdots$				
X_i	X_{i1}	X_{i2}	$\cdots$	X_{iN}
$\vdots$				
X_n	X_{n1}	X_{n2}	$\cdots$	X_{nN}

Data for PCA algorithm

Step 2. Compute the means of the variables

We compute the mean $\bar{X}_i$ of the variable X_i:

$$\bar{X}_i = \frac{1}{N}(X_{i1} + X_{i2} + \cdots + X_{iN}).$$

Step 3. Calculate the covariance matrix

Consider the variables X_i and X_j (i and j need not be different). The covariance of the ordered pair (X_i, X_j) is defined as[1]

$$\text{Cov}\,(X_i, X_j) = \frac{1}{N-1}\sum_{k=1}^{N}(X_{ik} - \bar{X}_i)(X_{jk} - \bar{X}_j).$$

We calculate the following $n \times n$ matrix S called the covariance matrix of the data. The element in the i-th row j-th column is the covariance $\text{Cov}\,(X_i, X_j)$:

$$S = \begin{bmatrix} \text{Cov}\,(X_1, X_1) & \text{Cov}\,(X_1, X_2) & \cdots & \text{Cov}\,(X_1, X_n) \\ \text{Cov}\,(X_2, X_1) & \text{Cov}\,(X_2, X_2) & \cdots & \text{Cov}\,(X_2, X_n) \\ \vdots & & & \\ \text{Cov}\,(X_n, X_1) & \text{Cov}\,(X_n, X_2) & \cdots & \text{Cov}\,(X_n, X_n) \end{bmatrix}$$

Step 4. Calculate the eigenvalues and eigenvectors of the covariance matrix

Let S be the covariance matrix and let I be the identity matrix having the same dimension as the dimension of S.

i) Set up the equation:

$$\det(S - \lambda I) = 0.$$

This is a polynomial equation of degree n in λ. It has n real roots (some of the roots may be repeated) and these roots are the eigenvalues of S. We find the n roots $\lambda_1, \lambda_2, \ldots, \lambda_n$ of Eq. (4.2).

ii) If $\lambda = \lambda'$ is an eigenvalue, then the corresponding eigenvector is a vector

$$U = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}$$

such that

$$(S - \lambda' I)U = 0.$$

(This is a system of n homogeneous linear equations in $u_1, u_2, \ldots, u_n$ and it always has a nontrivial solution.) We next find a set of n orthogonal eigenvectors $U_1, U_2, \ldots, U_n$ such that U_i is an eigenvector corresponding to λ_i.[2]

iii) We now normalise the eigenvectors. Given any vector X we normalise it by dividing X by its length. The length (or, the norm) of the vector

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

is defined as

$$\|X\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}.$$

Given any eigenvector U, the corresponding normalised eigenvector is computed as

$$\frac{1}{\|U\|} U.$$

We compute the n normalised eigenvectors $e_1, e_2, \ldots, e_n$ by

$$e_i = \frac{1}{\|U_i\|} U_i, \quad i = 1, 2, \ldots, n.$$

Step 5. Derive new data set

Order the eigenvalues from highest to lowest. The unit eigenvector corresponding to the largest eigenvalue is the first principal component. The unit eigenvector corresponding to the next highest eigenvalue is the second principal component, and so on.

i) Let the eigenvalues in descending order be $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ and let the corresponding unit eigenvectors be $e_1, e_2, \ldots, e_n$.

ii) Choose a positive integer p such that $1 \leq p \leq n$.

iii) Choose the eigenvectors corresponding to the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_p$ and form the following $p \times n$ matrix (we write the eigenvectors as row vectors):

$$F = \begin{bmatrix} e_1^T \\ e_2^T \\ \vdots \\ e_p^T \end{bmatrix},$$

where T in the superscript denotes the transpose.

iv) We form the following $n \times N$ matrix:

$$X = \begin{bmatrix} X_{11} - \bar{X}_1 & X_{12} - \bar{X}_1 & \cdots & X_{1N} - \bar{X}_1 \\ X_{21} - \bar{X}_2 & X_{22} - \bar{X}_2 & \cdots & X_{2N} - \bar{X}_2 \\ \vdots & & & \\ X_{n1} - \bar{X}_n & X_{n2} - \bar{X}_n & \cdots & X_{nN} - \bar{X}_n \end{bmatrix}$$

v) Next compute the matrix:

$$X_{\text{new}} = FX.$$

Note that this is a $p \times N$ matrix. This gives us a dataset of N samples having p features.

Step 6. New dataset

The matrix X_{new} is the new dataset. Each row of this matrix represents the values of a feature. Since there are only p rows, the new dataset has only features.

Step 7. Conclusion

This is how the principal component analysis helps us in dimensional reduction of the dataset. Note that it is not possible to get back the original n-dimensional dataset from the new dataset.

CLUSTERING TECHNIQUES

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.

The aim of segmentation is to split up a set of customer observations into segments such that the homogeneity within a segment is maximized (cohesive) and the heterogeneity between segments is maximized (separated). Popular applications include:

- Understanding a customer population (e.g., targeted marketing or advertising [mass customization])
- Efficiently allocating marketing resources
- Differentiating between brands in a portfolio
- Identifying the most profitable customers
- Identifying shopping patterns (Recommendation engine)
- Identifying the need for new products
- Pattern detection

Clustering techniques can be categorized as either hierarchical or non-hierarchical (see Figure 1).

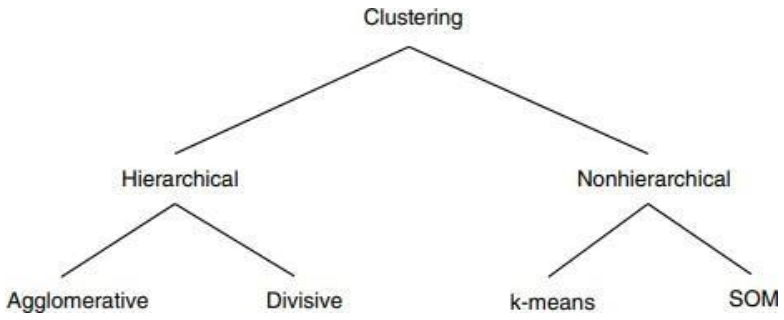


Figure 1: Hierarchical versus Non-hierarchical Clustering Techniques

HIERARCHICAL CLUSTERING

Hierarchical clustering (also called hierarchical cluster analysis or HCA) is a method of cluster analysis which seeks to build a hierarchy of clusters (or groups) in a given dataset. Thje hierarchical clustering produces clusters in which the clusters at each level of the hierarchy are created by merging clusters at the next lower level. At the lowest level, each cluster contains a single observation. At the highest level there is only one cluster containing all of the data. In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the dendrogram.

The decision regarding whether two clusters are to be merged or not is taken based on the measure of dissimilarity between the clusters. The distance between two clusters is usually taken as the measure of dissimilarity between the clusters.

- *Agglomerative*: Agglomerative is a bottom-up approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.
- *Divisive*: Divisive algorithm is the reverse of the agglomerative algorithm as it is a top-down approach. Divisive hierarchical clustering starts from the whole data set in one cluster, and then breaks this up in each time smaller clusters until one observation per cluster remains (right to left in Figure 2). Agglomerative clustering works the other way around, starting from all observations in one cluster and continuing to merge the ones that are most similar until all observations make up one big cluster (left to right in Figure 2).

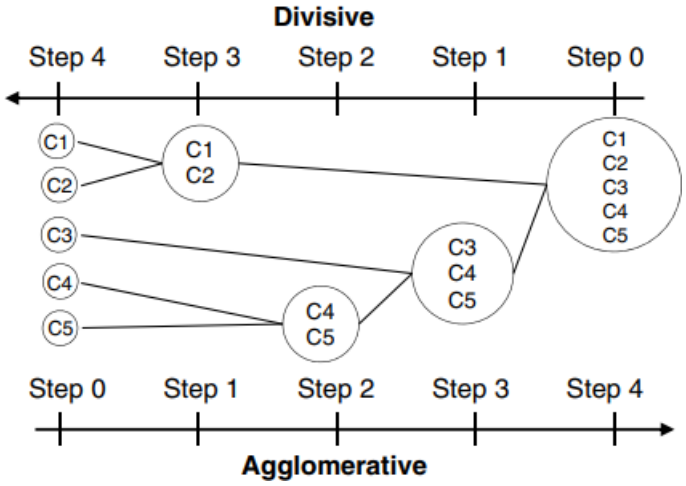


Figure 2: Divisive versus Agglomerative Hierarchical Clustering

In order to decide on the merger or splitting, a similarity rule is needed. Examples of popular similarity rules are the Euclidean distance and Manhattan (city block) distance (Table: 1).

Names	Formula
Euclidean distance	$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Squared Euclidean distance	$\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
Manhattan (or city block) distance	$\|a-b\|_1 = \sum_i \|a_i - b_i\|$
Maximum distance (or Chebyshev distance)	$\|a-b\|_\infty = \max_i \|a_i - b_i\|$

Table 1: Distance measures

There are several ways to measure the distance between clusters in order to decide the rules for clustering, and they are often called Linkage Methods (Figure 3). Some of the common linkage methods are:

- ***Complete-linkage***: the distance between two clusters is defined as the longest distance between two points in each cluster.
- ***Single-linkage***: the distance between two clusters is defined as the shortest distance between two points in each cluster. This linkage may be used to detect high values in your dataset which may be outliers as they will be merged at the end.
- ***Average-linkage***: the distance between two clusters is defined as the average distance between each point in one cluster to every point in the other cluster.
- ***Centroid-linkage***: finds the centroid of cluster 1 and centroid of cluster 2, and then calculates the distance between the two before merging.

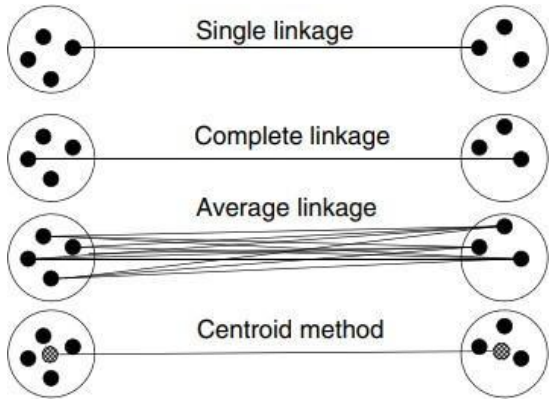


Figure 3: Calculating Distances between Clusters

Agglomerative Hierarchical clustering

In this technique, initially each data point is considered as an individual cluster. At each iteration, the similar clusters merge with other clusters until one cluster is formed. The basic algorithm of Agglomerative is straight forward.

1. Compute the proximity matrix
2. Let each data point be a cluster
3. Repeat: Merge the two closest clusters and update the proximity matrix
4. Until only a single cluster remains

Example

Problem 1

Given the dataset $\{a, b, c, d, e\}$ and the following distance matrix, construct a dendrogram by complete-linkage hierarchical clustering using the agglomerative method.

	a	b	c	d	e
a	0	9	3	6	11
b	9	0	7	5	10
c	3	7	0	9	2
d	6	5	9	0	8
e	11	10	2	8	0

The complete-linkage clustering uses the "maximum formula", that is, the following formula to compute the distance between two clusters A and B:

$$d(A, B) = \max\{d(x, y) : x \in A, y \in B\}$$

1. Dataset : $\{a, b, c, d, e\}$.

   Initial clustering (singleton sets) C_1: $\{a\}, \{b\}, \{c\}, \{d\}, \{e\}$.

2. The following table gives the distances between the various clusters in C_1:

	$\{a\}$	$\{b\}$	$\{c\}$	$\{d\}$	$\{e\}$
$\{a\}$	0	9	3	6	11
$\{b\}$	9	0	7	5	10
$\{c\}$	3	7	0	9	**2**
$\{d\}$	6	5	9	0	8
$\{e\}$	11	10	**2**	8	0

   In the above table, the minimum distance is the distance between the clusters $\{c\}$ and $\{e\}$. Also

$$d(\{c\}, \{e\}) = 2.$$

   We merge $\{c\}$ and $\{e\}$ to form the cluster $\{c, e\}$.

   The new set of clusters C_2: $\{a\}, \{b\}, \{d\}, \{c, e\}$.

3. Let us compute the distance of $\{c, e\}$ from other clusters.

   $d(\{c, e\}, \{a\}) = \max\{d(c, a), d(e, a)\} = \max\{3, 11\} = 11.$

   $d(\{c, e\}, \{b\}) = \max\{d(c, b), d(e, b)\} = \max\{7, 10\} = 10.$

   $d(\{c, e\}, \{d\}) = \max\{d(c, d), d(e, d)\} = \max\{9, 8\} = 9.$

   The following table gives the distances between the various clusters in C_2.

	$\{a\}$	$\{b\}$	$\{d\}$	$\{c,e\}$
$\{a\}$	0	9	6	11
$\{b\}$	9	0	**5**	10
$\{d\}$	6	**5**	0	9
$\{c,e\}$	11	10	9	0

In the above table, the minimum distance is the distance between the clusters $\{b\}$ and $\{d\}$. Also

$$d(\{b\},\{d\}) = 5.$$

We merge $\{b\}$ and $\{d\}$ to form the cluster $\{b,d\}$.

The new set of clusters C_3: $\{a\}$, $\{b,d\}$, $\{c,e\}$.

4. Let us compute the distance of $\{b,d\}$ from other clusters.

$d(\{b,d\},\{a\}) = \max\{d(b,a),d(d,a)\} = \max\{9,6\} = 9.$

$d(\{b,d\},\{c,e\}) = \max\{d(b,c),d(b,e),d(d,c),d(d,e)\} = \max\{7,10,9,8\} = 10.$

The following table gives the distances between the various clusters in C_3.

	$\{a\}$	$\{b,d\}$	$\{c,e\}$
$\{a\}$	0	**9**	11
$\{b,d\}$	**9**	0	10
$\{c,e\}$	11	10	0

In the above table, the minimum distance is the distance between the clusters $\{a\}$ and $\{b,d\}$. Also

$$d(\{a\},\{b,d\}) = 9.$$

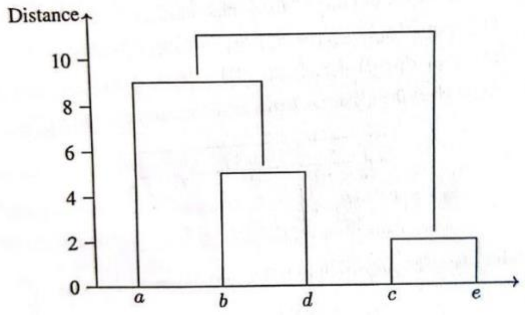
We merge $\{a\}$ and $\{b,d\}$ to form the cluster $\{a,b,d\}$.

The new set of clusters C_4: $\{a,b,d\}$, $\{c,e\}$

5. Only two clusters are left. We merge them form a single cluster containing all data points. We have

$$\begin{aligned} d(\{a,b,d\},\{c,e\}) &= \max\{d(a,c),d(a,e),d(b,c),d(b,e),d(d,c),d(d,e)\} \\ &= \max\{3,11,7,10,9,8\} \\ &= 11 \end{aligned}$$

Below figure shows the dendrogram for the above clustering.



Divisive clustering

Also known as a top-down approach. This algorithm also does not require to prespecify the number of clusters. Top-down clustering requires a method for splitting a cluster that contains the whole data and proceeds by splitting clusters recursively until individual data have been split into singleton clusters.

Advantages of hierarchal clustering

- It is easy to understand and implement.
- We don't have to pre-specify any particular number of clusters.
- Can obtain any desired number of clusters by cutting the Dendrogram at the proper level.
- They may correspond to meaningful classification.
- Easy to decide the number of clusters by merely looking at the Dendrogram.

Disadvantages of hierarchal clustering

- Hierarchical Clustering does not work well on vast amounts of data.
- All the approaches to calculate the similarity between clusters have their own disadvantages.
- Once a decision is made to combine two clusters, it cannot be undone.
- Different measures have problems with one or more of the following.
    - Sensitivity to noise and outliers.
    - Faces Difficulty when handling with different sizes of clusters.
    - It is breaking large clusters.
    - In this technique, the order of the data has an impact on the final results

NON-HIERARCHICAL CLUSTERING

This involves formation of new clusters by merging or splitting the clusters. It does not follow a tree like structure like hierarchical clustering. This technique groups the data in order to maximize or minimize some evaluation criteria. K- means clustering is an effective way of non-hierarchical clustering. In this method the partitions are made such that non-overlapping groups having no hierarchical relationships between themselves.

K-Means Clustering

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process. Working of k-means is shown in figure 3.

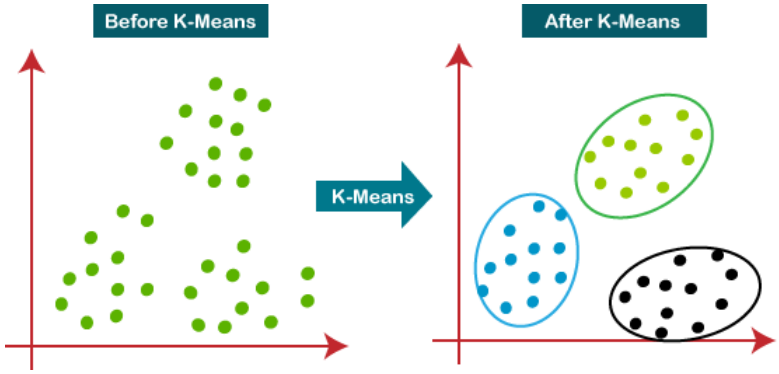


Figure 3: K-means clustering

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

The working of the K-Means algorithm is explained in the below steps:

- Step-1: Select the number K to decide the number of clusters.
- Step-2: Select random K points or centroids. (It can be other from the input dataset).
- Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.
- Step-4: Calculate the variance and place a new centroid of each cluster.
- Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

- Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.
- Step-7: The model is ready.

Advantages of k-means

- Relatively simple to implement.
- Scales to large data sets.
- k-Means may produce higher clusters than hierarchical clustering
- Guarantees convergence.
- Can warm-start the positions of centroids.
- Easily adapts to new examples.
- Generalizes to clusters of different shapes and sizes, such as elliptical

clusters. Dis-Advantages of k-means

- Choosing k manually.
- Difficult to predict K-Value.
- Being dependent on initial values.
- Clustering data of varying sizes and density.
- Different initial partitions can result in different final clusters.
- Scaling with number of dimensions
- Initial seeds have a strong impact on the final results
- Sensitive to scale
- The algorithm cannot be applied to categorical data.

Example

We illustrate the algorithm in the case where there are only two variables so that the data points and cluster centres can be geometrically represented by points in a coordinate plane. The distance between the points (x_1, x_2) and (y_1, y_2) will be calculated using the familiar distance formula of

Problem

Use k-means clustering algorithm to divide the following data into two clusters and also compute the the representative data points for the clusters.

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

Table 2 : Data for k-means algorithm example

Solution

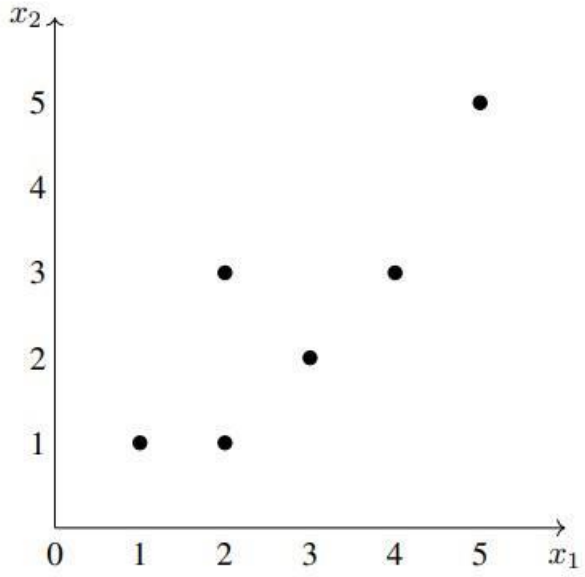


Figure : Scatter diagram of data

1. In the problem, the required number of clusters is 2 and we take $k = 2$.
2. We choose two points arbitrarily as the initial cluster centres. Let us choose arbitrarily

$$\vec{v}_1 = (2,1), \quad \vec{v}_2 = (2,3).$$

3. We compute the distances of the given data points from the cluster centers.

$\vec{x}_i$	Data point	Distance from $\vec{v}_1 = (2,1)$	Distance from $\vec{v}_2 = (2,3)$	Minimum distance	Assigned center
$\vec{x}_1$	$(1,1)$	1	2.24	1	$\vec{v}_1$
$\vec{x}_2$	$(2,1)$	0	2	0	$\vec{v}_1$
$\vec{x}_3$	$(2,3)$	2	0	0	$\vec{v}_2$
$\vec{x}_4$	$(3,2)$	1.41	1.41	0	$\vec{v}_1$
$\vec{x}_5$	$(4,3)$	2.82	2	2	$\vec{v}_2$
$\vec{x}_6$	$(5,5)$	5	3.61	3.61	$\vec{v}_2$

(The distances of $\vec{x}_4$ from $\vec{v}_1$ and $\vec{v}_2$ are equal. We have assigned $\vec{v}_1$ to $\vec{x}_4$ arbitrarily.)

This divides the data into two clusters as follows (see Figure 13.2):

Cluster 1: $\{\vec{x}_1, \vec{x}_2, \vec{x}_4\}$ represented by $\vec{v}_1$

Number of data points in Cluster 1: $c_1 = 3$.

Cluster 2 : $\{\vec{x}_3, \vec{x}_5, \vec{x}_6\}$ represented by $\vec{v}_2$

Number of data points in Cluster 2: $c_2 = 3$.

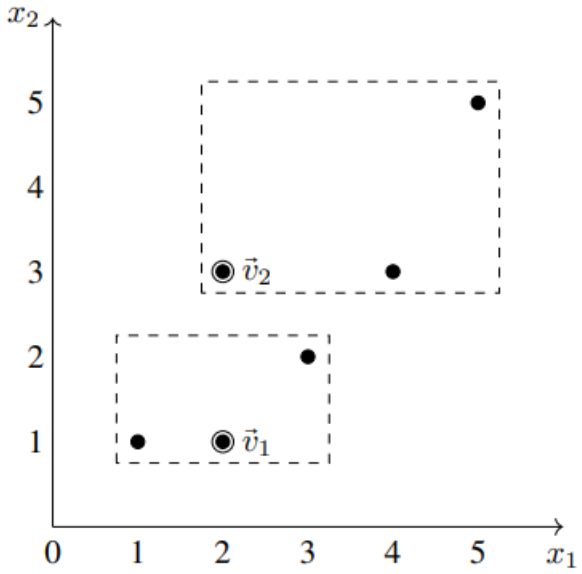


Figure : Initial choice of cluster centres and the resulting clusters

4. The cluster centres are recalculated as follows:

$$\begin{aligned}
\vec{v}_1 &= \frac{1}{c_1}(\vec{x}_1 + \vec{x}_2 + \vec{x}_4) \\
&= \frac{1}{3}(\vec{x}_1 + \vec{x}_2 + \vec{x}_4) \\
&= (2.00, 1.33) \\
\vec{v}_2 &= \frac{1}{c_2}(\vec{x}_3 + \vec{x}_5 + \vec{x}_6) \\
&= \frac{1}{3}(\vec{x}_3 + \vec{x}_5 + \vec{x}_6) \\
&= (3.67, 3.67)
\end{aligned}$$

5. We compute the distances of the given data points from the new cluster centers.

$\vec{x}_i$	Data point	Distance from $\vec{v}_1 = (2,1)$	Distance from $\vec{v}_2 = (2,3)$	Minimum distance	Assigned center
$\vec{x}_1$	$(1,1)$	1.05	3.77	1.05	$\vec{v}_1$
$\vec{x}_2$	$(2,1)$	0.33	3.14	0.33	$\vec{v}_1$
$\vec{x}_3$	$(2,3)$	1.67	1.80	1.67	$\vec{v}_1$
$\vec{x}_4$	$(3,2)$	1.20	1.80	1.20	$\vec{v}_1$
$\vec{x}_5$	$(4,3)$	2.60	0.75	0.75	$\vec{v}_2$
$\vec{x}_6$	$(5,5)$	4.74	1.89	1.89	$\vec{v}_2$

This divides the data into two clusters as follows (see Figure 13.4):

Cluster 1 : $\{\vec{x}_1, \vec{x}_2, \vec{x}_3, \vec{x}_4\}$ represented by $\vec{v}_1$

Number of data points in Cluster 1: $c_1 = 4$.

Cluster 2 : $\{\vec{x}_5, \vec{x}_6\}$ represented by $\vec{v}_2$

Number of data points in Cluster 1: $c_2 = 2$.

6. The cluster centres are recalculated as follows:

$$\begin{aligned}\vec{v}_1 &== \frac{1}{c_1}(\vec{x}_1 + \vec{x}_2 + \vec{x}_3 + \vec{x}_4) \\ &= \frac{1}{4}(\vec{x}_1 + \vec{x}_2 + \vec{x}_3 + \vec{x}_4) \\ &= (2.00, 1.33)\end{aligned}$$

$$\vec{v}_2 = \frac{1}{2}(\vec{x}_5 + \vec{x}_6) = (3.67, 3.67)$$

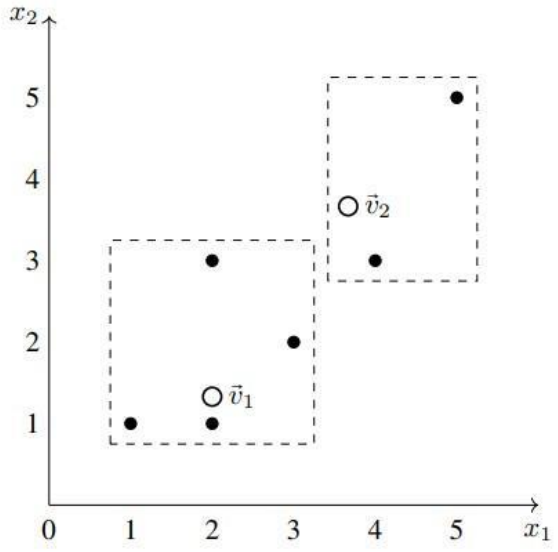


Figure : Cluster centres after first iteration and the corresponding clusters

7. We compute the distances of the given data points from the new cluster centers.

   4.609772 3.905125 2.692582 2.500000 1.118034 1.118034

$\vec{x}_i$	Data point	Distance from $\vec{v}_1 = (2,1)$	Distance from $\vec{v}_2 = (2,3)$	Minimum distance	Assigned center
$\vec{x}_1$	$(1,1)$	1.25	4.61	1.25	$\vec{v}_1$
$\vec{x}_2$	$(2,1)$	0.75	3.91	0.75	$\vec{v}_1$
$\vec{x}_3$	$(2,3)$	1.25	2.69	1.25	$\vec{v}_1$
$\vec{x}_4$	$(3,2)$	1.03	2.50	1.03	$\vec{v}_1$
$\vec{x}_5$	$(4,3)$	2.36	1.12	1.12	$\vec{v}_2$
$\vec{x}_6$	$(5,5)$	4.42	1.12	1.12	$\vec{v}_2$

This divides the data into two clusters as follows (see Figure **??**):

Cluster 1 : $\{\vec{x}_1, \vec{x}_2, \vec{x}_3, \vec{x}_4\}$ represented by $\vec{v}_1$

Number of data points in Cluster 1: $c_1 = 4$.

Cluster 2 : $\{\vec{x}_5, \vec{x}_6\}$ represented by $\vec{v}_2$

Number of data points in Cluster 1: $c_1 = 2$.

8. The cluster centres are recalculated as follows:

$$\begin{aligned}\vec{v}_1 &= \frac{1}{c_1}(\vec{x}_1 + \vec{x}_2 + \vec{x}_3 + \vec{x}_4) \\ &= \frac{1}{4}(\vec{x}_1 + \vec{x}_2 + \vec{x}_3 + \vec{x}_4) \\ &= (2.00, 1.75) \\ \vec{v}_2 &= \frac{1}{c_2}(\vec{x}_5 + \vec{x}_6) \\ &= \frac{1}{2}(\vec{x}_5 + \vec{x}_6) \\ &= (4.00, 4.50)\end{aligned}$$

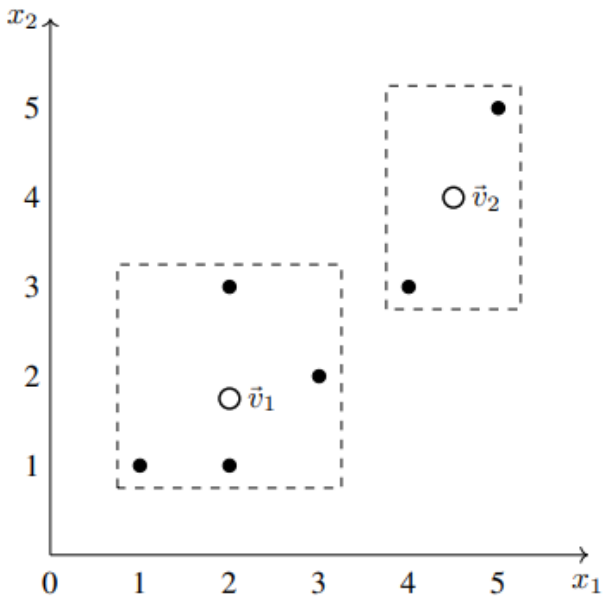


Figure : New cluster centres and the corresponding clusters

9. This divides the data into two clusters as follows

   Cluster 1 : $\{\vec{x}_1, \vec{x}_2, \vec{x}_3, \vec{x}_4\}$ represented by $\vec{v}_1$

   Cluster 2 : $\{\vec{x}_5, \vec{x}_6\}$ represented by $\vec{v}_2$

10. The cluster centres are recalculated as follows:

$$\vec{v}_1 = \frac{1}{4}(\vec{x}_1 + \vec{x}_2 + \vec{x}_3 + \vec{x}_4) = (2.00, 1.75)$$
$$\vec{v}_2 = \frac{1}{2}(\vec{x}_5 + \vec{x}_6) = (4.00, 4.50)$$

We note that these are identical to the cluster centres calculated in Step 8. So there will be no reassignment of data points to different clusters and hence the computations are stopped here.

11. Conclusion: The k means clustering algorithm with $k = 2$ applied to the dataset in Table yields the following clusters and the associated cluster centres:

Cluster 1 : $\{\vec{x}_1, \vec{x}_2, \vec{x}_3, \vec{x}_4\}$ represented by $\vec{v}_1 = (2.00, 1.75)$

Cluster 2 : $\{\vec{x}_5, \vec{x}_6\}$ represented by $\vec{v}_2 = (2.00, 4.75)$

Practical issues in clustering

- The cluster membership may change over time due to dynamic shifts in data.
  - Clustering struggles with time series data since you will have to pick a point in time to do your analysis. Data points could have belonged to any number of clusters at any time and doing one analysis at one point in time may be misleading.
- Handling outliers is difficult in cluster analysis.
  - Since most clustering algorithms use distance-based metrics, outliers in our datasets can completely change the clustering solution. The presence of just one outlier will cause cluster centroids to update, possibly moving points that should exist in one cluster to another.
- Clustering struggles with high-dimensional datasets.
  - High-dimensional data affects many machine learning algorithms, and clustering is no different. Clustering high-dimensional data has many challenges. These include the distance between points converging, the output becoming impossible to visualize, correlation skewing the location of the points, and the local feature relevance problem.
- Multiple correct answers for the same problem.
  - Many clustering algorithms will generate random centroids to start the computation. This methodology creates a scenario where different solutions can be found depending on how many clusters are chosen and where the centroids are initially placed. Since many local solutions can be found, the reproducibility of your analysis suffers.
- Evaluating a solution's correctness is problematic.
  - Since labelled data is missing in clustering algorithms, evaluating solutions becomes tough. This is because we have nothing to compare our clusters against. There are tactics to evaluate solutions from clusters (like the silhouette method) but utilizing distance in our metrics can cause problems.

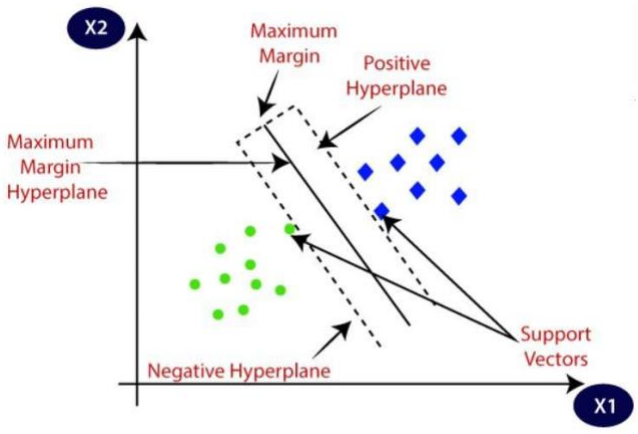
- Clustering computation can become complex and expensive.

- In situations where there are very large data sets or many dimensions, many clustering algorithms will fail to converge or come to a solution. For example, the time complexity of the K-means algorithm making it impossible to use as the number of rows (N) grows.

- Assumption of equal feature variance in distance measures.
  - Clustering algorithms that only utilize distance metrics assume that all features are equal in terms of relevance to the problem. This creates problems, as some dimensions are much more relevant to our specific situation than others, potentially weakening our analysis.
- Struggle with missing data (columns and points)
  - In clustering, missing values and dimensions can completely change the solution. While this problem exists in all subsets of machine learning, clustering is highly sensitive to it. This is due to the distance nature of clustering algorithms that recompute centroids based on available data.

SUPPORT VECTOR CLASSIFIERS AND SUPPORT VECTOR MACHINES

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.
- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
- SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.
- Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



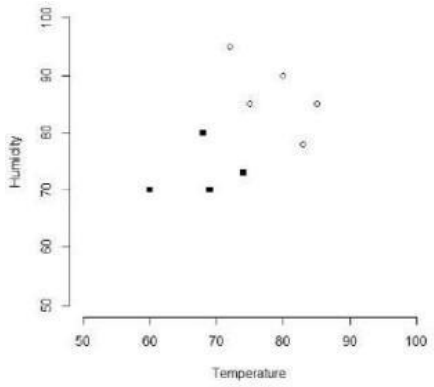
Hyperplane

- There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.
- The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.
- We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.
- The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a ***Support vector***.

- Consider the below example, we can draw many separating lines for classification.

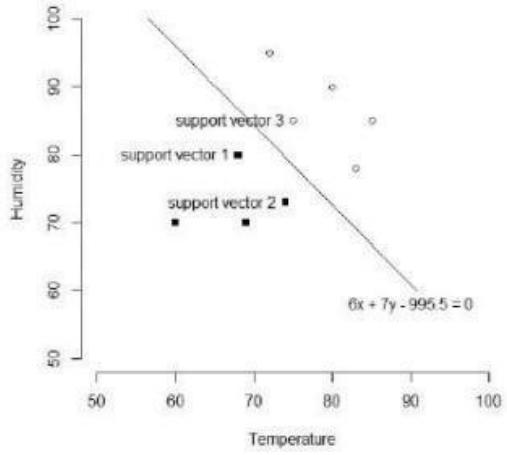
temperature	humidity	play
85	85	no
60	70	yes
80	90	no
72	95	no
68	80	yes
74	73	yes
69	70	yes
75	85	no
83	78	no

Table : Example data with two class labels



- The “best” separating line is the one with the maximum margin.
- The separating line with the maximum margin is called the “maximum margin line” or the “optimal separating line”. This line is also called the “***support vector machine***”
- The data points which are closest to the maximum margin line are called the “***support vectors***”.

ITT 306 DATA


Humidity
Temperature
support vector 3
support vector 1
support vector 2
6x + 7y - 995.5 = 0
50
60
70
80
90
100

Definition for hyperplane

For instance, in two dimensions, a hyperplane is a flat one-dimensional subspace—in other words, a line. In three dimensions, a hyperplane is a flat two-dimensional subspace—that is, a plane. In $p > 3$ dimensions, it can be hard to visualize a hyperplane, but the notion of a $(p-1)$-dimensional flat subspace still applies.

The mathematical definition of a hyperplane is quite simple. In two dimensions, a hyperplane is defined by the equation

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

Equation can be easily extended to the p-dimensional setting:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

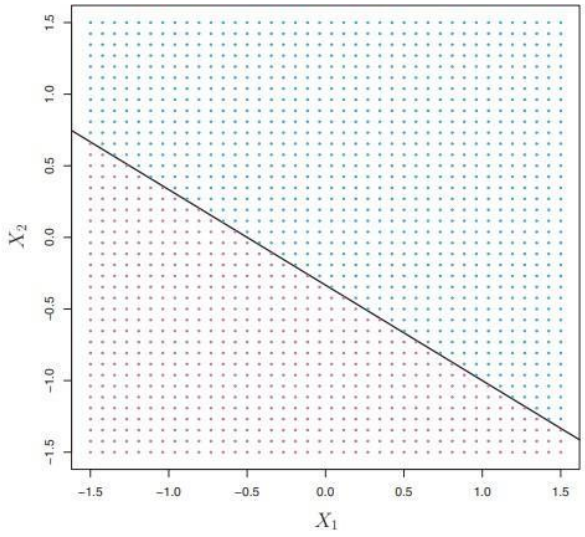
The above equation defines a p-dimensional hyperplane and equation satisfies then X lies on the hyperplane.

Now, suppose that X does not satisfy ; rather,

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0.$$

Then this tells us that X lies to one side of the hyperplane. On the other hand, if

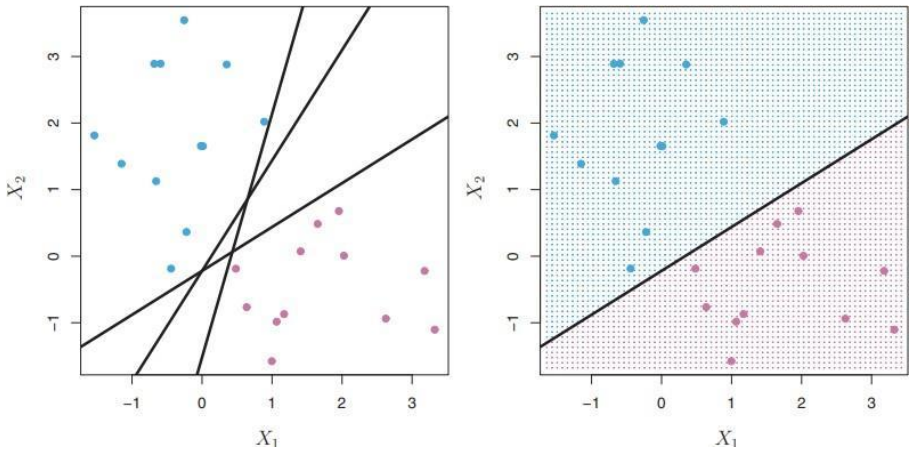
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0,$$



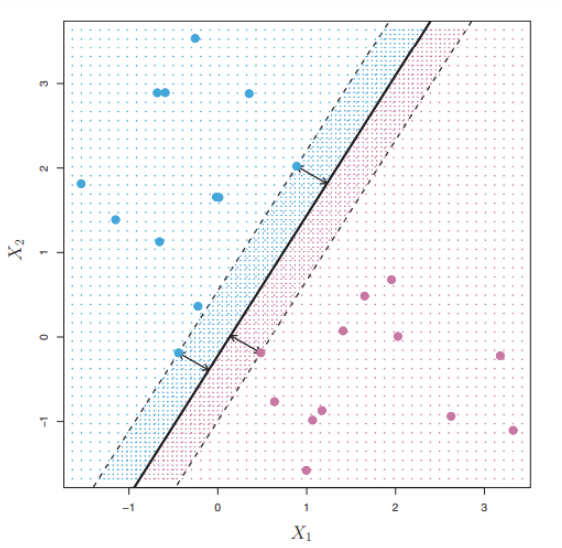
The hyperplane $1 + 2X_1 + 3X_2 = 0$ is shown. The blue region is the set of points for which $1 + 2X_1 + 3X_2 > 0$, and the purple region is the set of points for which $1 + 2X_1 + 3X_2 < 0$.

The Maximal Margin Classifier

In general, if our data can be perfectly separated using a hyperplane, then there will in fact exist an infinite number of such hyperplanes. This is because a given separating hyperplane can usually be shifted a tiny bit up or down, or rotated, without coming into contact with any of the observations as shown in below figure.



- The ***maximal margin hyperplane*** (optimal separating hyperplane), which is the separating hyperplane that optimal separating hyperplane is farthest from the training observations.
- That is, we can compute the (perpendicular) distance from each training observation to a given separating hyperplane; the smallest such distance is the minimal distance from the observations to the hyperplane and is known as the ***margin***.
- The maximal margin hyperplane is the separating hyperplane for which the margin is largest—that is, it is the hyperplane that has the farthest minimum distance to the training observations. We can then classify a test observation based on which side of the maximal margin hyperplane it lies. This is known as the ***maximal margin classifier***.



There are two classes of observations, shown in blue and in purple. The maximal margin hyperplane is shown as a solid line. The margin is the distance from the solid line to either of the dashed lines. The two blue points and the purple point that lie on the dashed lines are the support vectors, and the distance from those points to the hyperplane is indicated by arrows. The purple and blue grid indicates the decision rule made by a classifier based on this separating hyperplane.

- The three training observations are equidistant from the maximal margin hyperplane and lie along the dashed lines indicating the width of the margin.
- These three observations are known as support vectors.
- These are the points that are closest to the hyperplane. A separating line will be defined with the help of these data points.
- The generalization of the maximal margin classifier to the non-separable case is known as the support vector classifier.
- The support vector classifier, sometimes called a soft margin classifier, support vector classifier soft margin classifier does exactly this. Rather than seeking the largest possible margin so that every observation is not only on the correct side of the hyperplane but also on the correct side of the margin, we instead allow some observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane.
- The support vector machine (SVM) is an extension of the support vector support vector machine classifier that results from enlarging the feature space in a specific way, using kernels.
- Kernels or kernel methods (also called Kernel functions) are sets of different types of algorithms that are being used for pattern analysis.
- They are used to solve a non-linear problem by using a linear classifier. Kernels Methods are

employed in SVM (Support Vector Machines) which are used in classification and regression problems. The SVM uses what is called a “Kernel Trick” where the data is transformed and an optimal boundary is found for the possible outputs.

RESAMPLING METHODS

Resampling is the method that consists of drawing repeated samples from the original data samples. The method of Resampling is a nonparametric method of statistical inference. In other words, the method of resampling does not involve the utilization of the generic distribution tables (for example, normal distribution tables) in order to compute approximate p probability values.

Resampling involves the selection of randomized cases with replacement from the original data sample in such a manner that each number of the sample drawn has a number of cases that are similar to the original data sample. Due to replacement, the drawn number of samples that are used by the method of resampling consists of repetitive cases.

There are two major re-sampling methods available and are:

- Bootstrap
- Cross validation

Cross validation

- A resampling method that uses different portions of the data to test and train a model on different iterations.
- Prediction Problem
- It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.
- Used to test the effectiveness of a machine learning models.

Train-Test Split approach

- We randomly split the complete data into training and test sets.
- Then Perform the model training on the training set and use the test set for validation purpose.
- Training data is the initial dataset you use to teach a machine learning application to recognize patterns.
- You'll need a new dataset to validate the model because it already "knows" the training data.
- The goal of cross-validation is to test the model's ability to predict new data that was not used in estimating it.
- Derive a more accurate estimate of model prediction

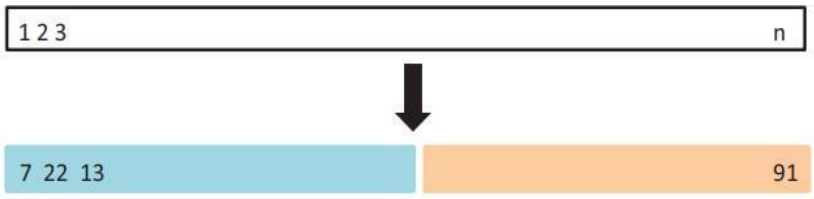
performance. The different approaches of cross validation are:

- Hold out method / validation approach
- Leave One Out Cross-Validation
- K-fold Cross validation

Hold out method / validation approach

- This is the simplest evaluation method and is widely used in Machine Learning projects. Here the entire dataset(population) is divided into 2 sets – train set and test set. The data can be divided into 70-30 or 60-40, 75-25 or 80-20, or even 50-50 depending on the use case. As a rule, the proportion of training data has to be larger than the test data.
- In the Hold out method, the test error rates are highly variable (high variance) and it totally depends on which observations end up in the training set and test set
- Only a part of the data is used to train the model (high bias) which is not a very good idea when data is not huge and this will lead to overestimation of test error.
- One of the major advantages of this method is that it is computationally inexpensive compared to other cross-validation techniques.

Example:



A schematic display of the validation set approach. A set of n observations are randomly split into a training set (shown in blue, containing observations 7, 22, and 13, among others) and a validation set (shown in beige, and containing observation 91, among others). The statistical learning method is fit on the training set, and its performance is evaluated on the validation set.

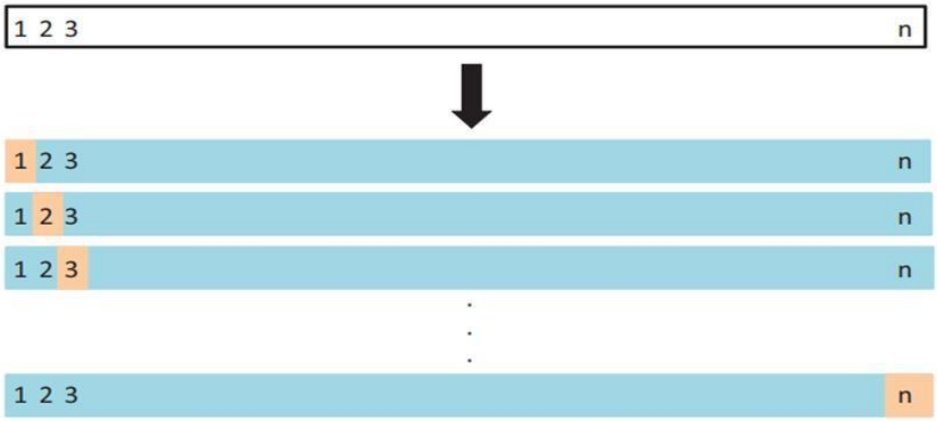
Drawbacks:

- The validation estimate of the test error rate can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set.
- In the validation approach, only a subset of the observations—those that are included in the training set rather than in the validation set—are used to fit the model. Since statistical methods tend to perform worse when trained on fewer observations, this suggests that the validation set error rate may tend to overestimate the test error rate for the model fit on the entire data set.

Leave One Out Cross-Validation

- In this method, we divide the data into train and test sets – but with a twist.
- Instead of dividing the data into 2 subsets, we select a single observation as test data, and everything else is labeled as training data and the model is trained.
- Now the 2nd observation is selected as test data and the model is trained on the remaining data.

- In the leave-one-out (LOO) cross-validation, we train our machine-learning model n times where n is to our dataset's size. Each time, only one sample is used as a test set while the rest are used to train our model.
- This process continues 'n' times and the average of all these iterations is calculated and estimated as the test set error.
- LOO is more appropriate since is used on small data sets, it will use more training samples in each iteration. That will enable our model to learn better representations.
- When it comes to test-error estimates, LOOCV gives unbiased estimates (low bias).
- LOOCV has an extremely high variance because we are averaging the output of n-models which are fitted on an almost identical set of observations, and their outputs are highly positively correlated with each other.
- This is computationally expensive as the model is run 'n' times to test every observation in the data.



A schematic display of LOOCV. A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige). The test error is then estimated by averaging the n resulting MSE's. The first training set contains all but observation 1, the second training set contains all but observation 2, and so forth.

K-fold Cross validation

- This approach involves randomly dividing the set of observations into k folds of nearly equal size.
- It ensures that every observation from the original dataset has the chance of appearing in training and test set.
- This is one among the best approach if we have a limited input data.
- The first fold is treated as a validation set and the model is fit on the remaining folds.
- The procedure is then repeated k times, where a different group each time is treated as the validation

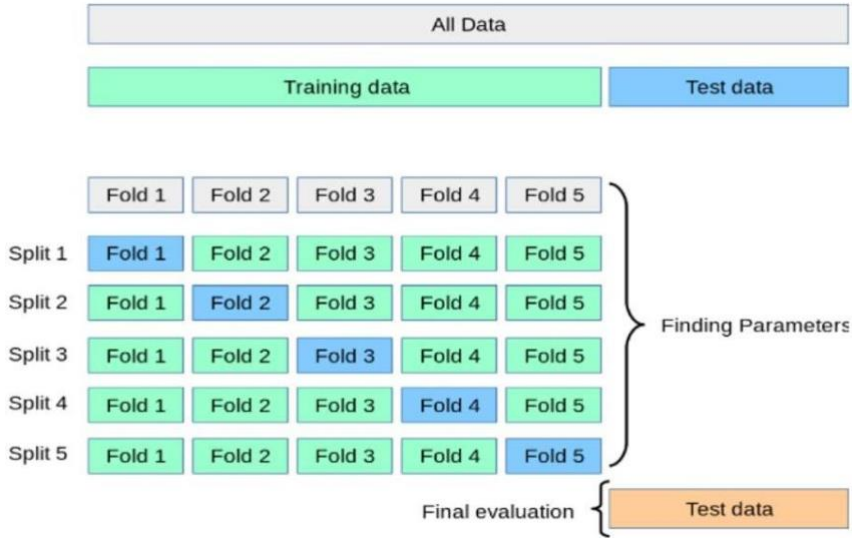


Figure 7: k-fold cross validation

Algorithm:

- The training set is split into *k* smaller sets.
- `A model is trained using k−1 of the folds as training data`
- The resulting model is validated on the remaining part of the data
- The performance measure reported by *k*-fold cross-validation is then the average of the values computed in the loop.

Bootstrapping

- Method that involves drawing of sample data repeatedly with replacement from a data source to estimate a population parameter.
- Sampling with replacement is considered.
- Data point in a drawn sample can reappear in future drawn samples as well.
- Method of estimating parameters for the population using samples.
- Example: For example, the average height of residents in a city, the count of red blood cells.
- Samples are constructed by drawing observations from a large data sample one at a time and returning them to the data sample after they have been chosen.
- This allows a given observation to be included in a given small sample more than once
- This approach to sampling is called sampling with replacement. (figure 8)

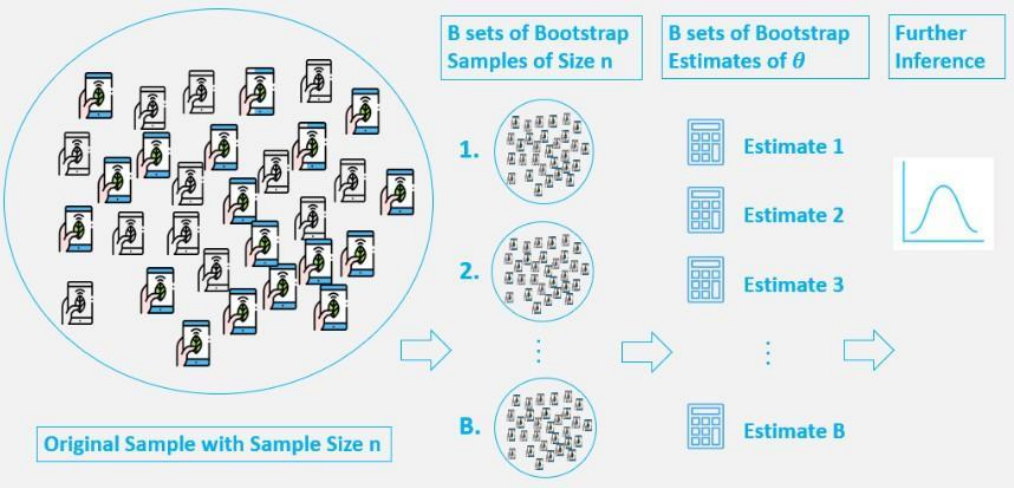


Figure 8: Steps involved in bootstrapping

Algorithm:

1. Choose a number of bootstrap samples to perform
2. Choose a sample size
3. For each bootstrap sample
   a) Draw a sample with replacement with the chosen size
   b) Fit a model on the data sample
   c) Estimate the skill of the model on the out-of-bag sample.
4. Calculate the mean of the sample of model skill estimates.

- In K-Fold CV, the no of folds k is less than the number of observations in the data (k<n) and we are averaging the outputs of k fitted models that are somewhat less correlated with each other since the overlap between the training sets in each model is smaller. This leads to low variance then LOOCV.
- The best part about this method is each data point gets to be in the test set exactly once and gets to be part of the training set k-1 times. As the number of folds k increases, the variance also decreases (low variance).
- This method leads to intermediate bias because each training set contains fewer observations (k-1)n/k than the Leave One Out method but more than the Hold Out method.
- K-fold Cross Validation is performed using k=5 or k=10 as these values have been empirically shown to yield test error estimates that neither have high bias nor high variance.
- The major disadvantage of this method is that the model has to be run from scratch k-times and is computationally expensive than the Hold Out method but better than the Leave One Out method.

Bias and Variance trade off

- K-Fold Cross Validation gives more accurate estimates than Leave One Out Cross-Validation. Whereas Hold One Out CV method usually leads to overestimates of the test error rate, because in this approach, only a portion of the data is used to train the machine learning model.
- When it comes to bias, the Leave One Out Method gives unbiased estimates because each training set contains n-1 observations (which is pretty much all of the data). K-Fold CV leads to an intermediate level of bias depending on the number of k-folds when compared to LOOCV but it's much lower when compared to the Hold Out Method.