1

CLUSTERING

Unit Structure:

1.1 OBJECTIVE

1. To understand the distance or similarity between two objects.
2. To understand the important role of data mining in business and big data analytics.
3. To learn and understand the K-means & K-Medoids algorithm.
4. To understand the implementation of partitioning and hierarchical algorithms.

1.2 INTRODUCTION

- Business intelligence (BI) enables organizations to study historical and live data, so they can shortly detect actionable insights for making planned decisions.
- Business intelligence tools allow the processing of large data sets over multiple sources and confer findings in visual formats that are easy to understand and share.
- Clustering is the process of examining, observing a collection of “points” and grouping the points into “clusters” according to some distance measure.
- The objective is that points in the same cluster have a small distance from each one another, while points in different clusters are at a large distance from one another.

- A Cluster look like was shown in Fig. 1, there are three clusters around three different road intersections, but two of the clusters blended into one another because they were not separated enough.
- Here we understand the methods for discovering clusters in data. We need to find where the data is very large, and/or where the space either is high-dimensional, or the space is not Euclidean at all.
- The two general approaches to clustering and the methods for dealing with clusters in a non- Euclidean space.

1.3 DISTANCE/SIMILARITY,

- Similarity or distance between two objects plays an essential role in many data mining tasks like classification and clustering which involves distance computation.
- The distance or similarity for integer type and ratio scaled data is well defined and understood. For continuous data, the most Similarity or distance between two objects plays an essential role in many data mining tasks like classification and clustering which involves distance computation.
- Using Minkowski distance we measure the continuous data, which is the most commonly used.
- These measures are independent of the nature of the underlying distribution of data. Many data-driven similarity compute like Mahalanobis distance have also been defined for continuous data.
- The similarity measures for categorical data are not well defined, as in our survey we could not find a generic mechanism for comparison of two values of a categorical attribute.
- For categorical data the domain values are discrete and have no ordering defined.
- An example of categorical attribute is color={red, green, blue} or shape={circle, rectangle, square }. Although it may be argued that categorical data are all different and must be looked at on a case-by-case basis, there are methodological generalities and similarities that we can find when dealing with them.
- A dataset suitable for clustering is a collection of points, which are objects belonging to some space.
- In General case, a space is just a universal set of points, from which the points in the dataset are drawn. However, we should be wise to the common case of a Euclidean space which has a number of important properties beneficial for clustering. In particular, a Euclidean space's points are vectors of real numbers.
- The length of the vector is equal to the number of dimensions of the space. The components of the vector are often called coordinates of the represented points.

- All spaces for which we can perform a clustering which have a distance measure, giving a distance between any two points in the space.
- The common Euclidean distance (square root of the sums of the squares of the differences between the coordinates of the points in each dimension) serves for all Euclidean spaces, although we also mentioned some other options for distance measures in Euclidean spaces, including the Manhattan distance (sum of the magnitudes of the differences in each dimension) and the L∞-distance (maximum magnitude of the difference in any dimension).

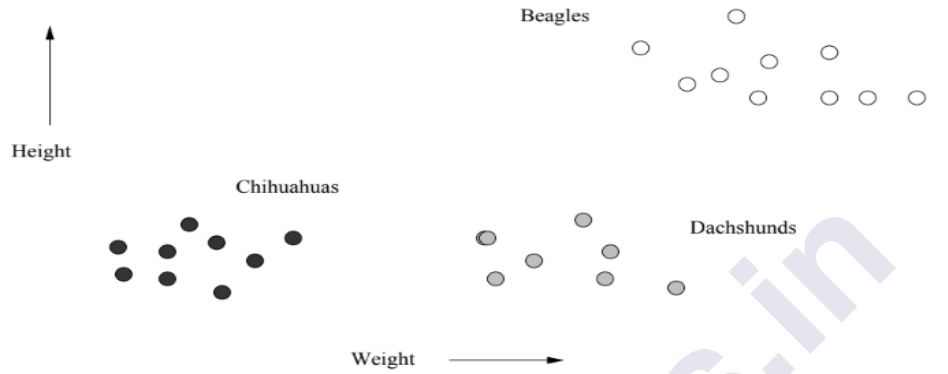


Fig.1. Heights and weights of dogs taken from three varieties

- For example, Fig. 1 shows height and weight measurements of dogs of several varieties. Without realizing which dog is of which variety, we can see just by observing the diagram that the dogs fall into three clusters, and those clusters emerge to correspond to three varieties. With the help of small amounts of data, any clustering algorithm will establish the right clusters, and simply plot the points.
- The distance measures for non-Euclidean spaces. These include the Jaccard distance, cosine distance, Hamming distance, and edit distance.
- Recall that the requirements for a function on pairs of points to be a distance measure are that
    - Distances are always counted as a nonnegative, and only the distance between a point and itself is 0.
    - Distance is symmetric
    - It isn't difficult in which order you consider the points when computing their distance.
    - Distance measures keep to the triangle inequality; the distance from x to y to z is never less than the distance going from x to z directly.

1.4 PARTITIONING ALGORITHM: K-MEANS; K-MEDOIDS,

K-means Algorithms

- The task is to categorize those items into groups. To achieve this, the k-Means algorithm is used. an unsupervised learning algorithm. 'K' in the name of the algorithm represents the number of groups or clusters to classify our items into.
- The algorithm will categorize the items into k groups or clusters of similarity. To calculate that similarity, the Euclidean distance as measurement.
- The algorithm works as follows:
    - First, we initialize k points, called means or cluster centroids, randomly.
    - We categorize each item to its closest mean and we update the mean's coordinates accordingly, which are the averages of the items categorized in that cluster so far.
    - We repeat the process for a given number of iterations and at the end, we have our clusters.

K-Means Basics

- A k-means algorithm is shown in Fig. 2 There are many ways to select the initial k points that represent the clusters.
- The core of the algorithm is the for-loop, in which we consider each point other than the k selected points and assign it to the closest cluster, where "closest" means closest to the centroid of the cluster.
- Note that the centroid of a cluster can migrate as points are assigned to it.
- Although, since only points near the cluster are likely to be assigned, the centroid tends not to move too much.

```
Initially choose k points that are likely to be in
different clusters;
Make these points the centroids of their clusters;
FOR each remaining point p DO
find the centroid to which p is closest;
Add p to the cluster of that centroid;
Adjust the centroid of that cluster to account for p;
END;
```

Fig. 2: Outline of k-means algorithms

- A discretionary step at the end is to fix the centroids of the clusters and to reassign each point, including the k initial points, to the k clusters.
- oftentimes, a point p will be assigned to the same cluster in which it was placed on the first pass.
- There are cases where the centroid of p's original cluster moved quite far from p after p was placed there, and p is assigned to a different cluster on the second pass.
- In fact, even some of the original k points could dissolve being reassigned.
- As these examples are unusual, we shall not dwell on the subject.

Initializing Clusters for K-Means

- There are two approaches.

1. Pick points that are as far away from one another as possible.
2. Cluster a sample of the data, perhaps hierarchically, so there are k clusters. Pick a point from each cluster, perhaps that point closest to the

Centroid of the cluster.

- The second approach requires little elaboration. For the first approach, there are variations. One good choice is:

```
Pick the first point at random;
WHILE there are fewer than k points DO
Add the point whose minimum distance from the selected
points is as large as possible;
END;
```

Example 1: Let us consider the twelve points of Fig. 2, which we reproduce here as Fig. 7.8.

- In the worst case, our initial choice of a point is near the centre, say (6, 8). The furthest point from (6,8) is (12,3), so that point is chosen next.

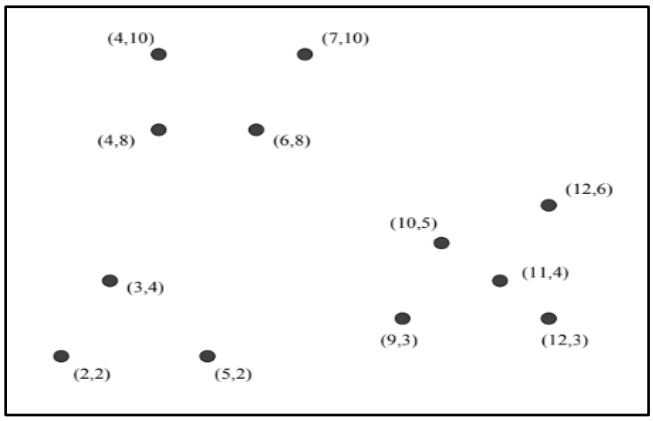


Fig. 3. Repeat of Fig. 2

- Among the remaining ten points, the one whose minimum distance to either (6,8) or (12,3) is a maximum is (2,2). That point has distance √52 = 7.21 from (6,8) and distance √ 101 = 10.05 to (12,3); thus its "score" is 7.21.
- We can check easily that any other point is less than distance 7.21 from at least one of (6,8) and (12,3). Our selection of three starting points is thus (6,8), (12,3), and (2,2).
- Notice that these three belong to different clusters. Had we started with a different point, say (10,5), we would get a different set of three initial points.
- In this case, the starting points would be (10,5), (2,2), and (4,10). Again, these points belong to the three different clusters.

Picking the Right Value of k

1. We may not know the correct value of k to use in a k-means clustering. However, if we can measure the quality of the clustering for various values of k, we can usually guess what the right value of k is.
2. If we take a measure of appropriateness for clusters, such as average radius or diameter, that value will grow slowly, as long as the number of clusters.
3. we assume remains at or above the true number of clusters. However, as soon as we try to form fewer clusters than there really are, the measure will rise precipitously. The idea is expressed by the diagram of Fig. 4

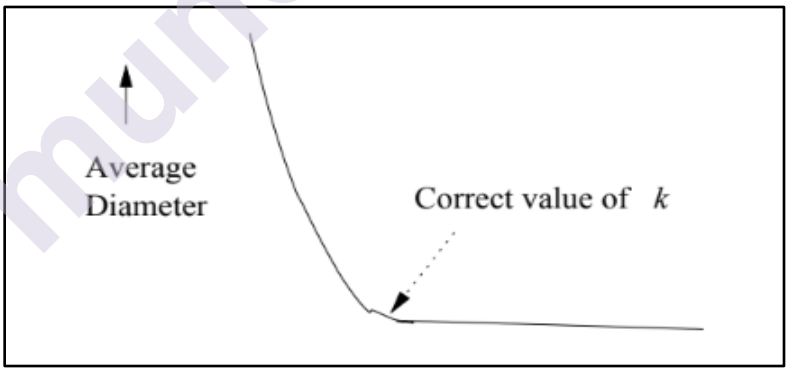


Fig. 4: Average diameter or another measure of diffuseness rises quickly as soon as the number of clusters falls below the true number present in the data

Partitioned Clustering - K-Means & K-Medoids

To achieve global optimality in partitioning based clustering it requires the continuous evaluation of all the possible partitions.

1) The k-means algorithm, In this algorithm, where each cluster is represented by the mean value of the objects in the cluster.

2) **The k-medoids algorithm**, in this algorithm, where each cluster is represented by one of the objects located near the centre of the cluster.

The heuristic clustering methods work well for finding spherical-shaped clusters in small to medium databases.

To find clusters with complex shapes and for clustering very large data sets, partitioning based methods need to be extended.

Partitioning Algorithms: Basic Concept

Partitioning method: Construct a partition of a database D of n objects into a set of k clusters.

Given a k, find a partition of k clusters that optimizes the chosen partitioning criterion

- **Global optimal:**
  - Continuous evaluation all partitions
- **Heuristic methods:**
  - k-means and k-medoids algorithms
- **k-means**:
  - Each cluster is represented by the center of the cluster
- **k-medoids or PAM (Partition around medoids):**
  - Each cluster is represented by one of the objects in the cluster

1. K-Means Algorithm

The k-means algorithm is implemented in 4 steps:

1. Partition objects into k nonempty subsets
2. Calculate seed points as the centroids of the clusters of the current partition. The centroid is the center (or mean point) of the cluster.
3. Assign each object to the cluster with the closest seed point.
4. Go to Step 2, stop when there are no more new assignments.

This criterion tries to make the resulting k clusters as compact and as separate as possible.

(a) (b) (c)

Fig. 5 Clusters

Suppose that there is a set of objects located in space as depicted in the rectangle. Let k =3; i.e. the user would like to cluster the object into three clusters. On the report of the algorithm, we arbitrarily choose three objects as the three initial cluster centers, where cluster centers are marked by a "+". All objects are distributed to a cluster based on the cluster center to which it is the nearest. Such distribution forms are circled by dotted curves.

Advantages of K-Means

- Relatively efficient: O(tkn), where n is # objects, k is # clusters, and t is # iterations. Normally, k, t << n.
- Each object is distributed to a cluster based on the cluster center to which it is the nearest.

Disadvantages of K-Means

- Applicable only when mean is defined, then what about categorical data?
- Need to specify k, the number of clusters, in advance.
- Unable to handle noisy data and outliers.
- Not suitable to discover clusters with non-convex shapes.

2. K-Medoids Clustering Algorithm

- A Medoid can be defined as that object of a cluster, whose average disparity to all the objects in the cluster is minimal; it mean that it is a most centrally located point in the given dataset.
- **K-Medoids:** Rather than taking the mean value of the object in a cluster as a reference point, medoids can be used, which is the most middle of located object in a cluster.
- The main plan of k-medoids clustering algorithms is to find k clusters in n objects by first arbitrarily finding a representative object (the medoid) for each cluster. \
- Each remaining object is clustered with the medoid to which it is most similar.
- The plan then iteratively replaces one of the medoids by one of the non-medoids as long as the quality of the resulting clustering is improved.
- This standard is estimated by using a cost function that measures the average dissimilarity between an object and the medoid of its cluster.
- To find whether a non-medoid object is "Oi" random is a good replacement for a current medoid "Oj", the following four cases are examined for each of the non-medoid objects "P".

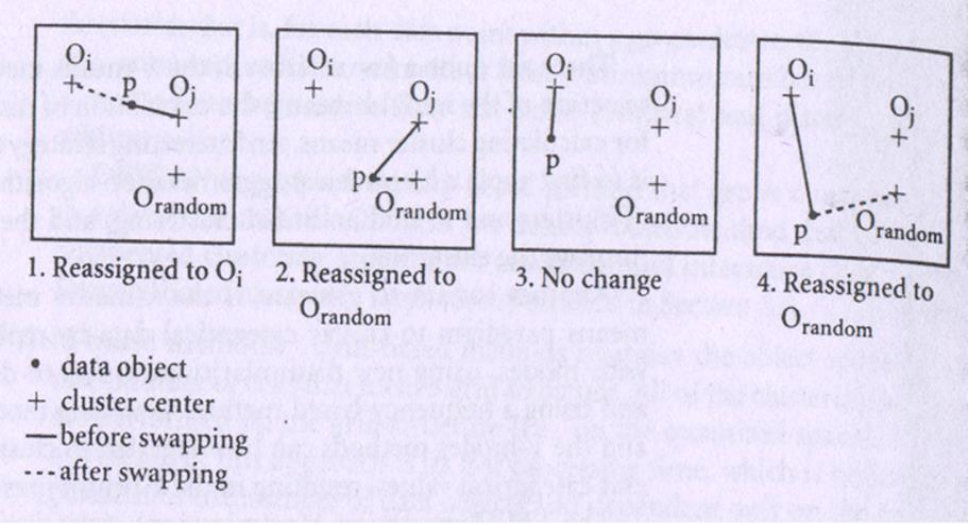


Fig.6 Home clustering Partitional Clustering - K-Means & K-Medoids

Case 1: "P" currently belongs to medoid "Oj", If "Oj" is replaced by "Orandom", as a medoid and "P" is closest to one of "Oi", it do not belong "j", then "P" is assigned to "Oi".

Case 2: "P" currently belongs to medoid "Oj". If "Oj" is replaced by "Orandom" as medoid and "P" is closest to "Orandom", then "P" is reassigned to "Orandom".

Case 3: "P" currently belongs to medoid "Oi", it does not belong "j". If "Oj" is replaced by "Orandom" as a medoid and "P" is still closest to "Oi", then the assignment does not change.

Case 4: "P" currently belongs to medoid "Oi", it does not belong to "j". If "Oj" is replaced by "Orandom" as a medoid and "P" is closest to "Orandom", then "P" is reassigned to "Orandom".

1.5 PARTITIONING ALGORITHM FOR LARGE DATA SET: CLARA; CLARANS,

- A typical k-medoids partitioning algorithm works effectively for small data sets, but does not scale well for large data sets. To distribute with larger data sets, a sampling-based method, called **CLARA** can be used.
- The concept behind CLARA is as follows:
  - Instead of taking the whole set of data into consideration, a small portion of the actual data is chosen as a representative of the data.
  - Medoids are then chosen from this sample using PAM.
  - Here PAM stands for “partition around medoids”. The algorithm is intended to find a sequence of objects called medoids that are centrally located in clusters.

- If the sample is selected in a fairly random manner, it should closely represent the original data set.
- The representative medoids chosen will likely be similar to those that would have been chosen from the whole data set.
- CLARA draws multiple samples of the data set, applies PAM on each sample, and returns its best clustering as the output.
- As expected, CLARA can deal with larger data sets than PAM. The complexity of each iteration now becomes $O(ks2 + k(n-k))$ where s is the size of the sample, k is the number of clusters, and n is the total number of objects.

- The potency of CLARA depends on the sample size. Notice that PAM searches for the best k medoids among a given data set, whereas CLARA searches for the best k medoids among the selected sample of the data set. CLARA is not able to find the best clustering if any of the best sampled medoids is not among the best k medoids.
- A best clustering based on sampling will not necessarily show a good clustering of the whole data set if the sample is biased.
- A k-medoids type algorithm called **CLARANS** (Clustering Large Applications based upon Randomized Search) was prospective, which combines the sampling technique with PAM. Unlike CLARA, CLARANS does not confine itself to any sample at any given time.
- CLARA has a fixed sample at each stage of the search, CLARANS draws a sample with some randomness in each step of the search. Conceptually, the clustering process can be viewed as a search through a graph, where each node is a potential solution (a set of k medoids).
- Two nodes are neighbours (that is, connected by an arc in the graph) if their sets differ by only one object. Each node can be assigned a cost that is defined by the total dissimilarity between every object and the medoid of its cluster.
- At every step, PAM studies all of the neighbours of the current node in its search for a minimum cost solution. The current node is then replaced by the neighbour with the largest descent in costs. Because CLARA works on a sample of the entire dataset, it examines fewer neighbours and restricts the search to subgraphs that are smaller than the original graph.
- While CLARA draws a sample of nodes at the beginning of a search, CLARANS dynamically draws a random sample of neighbours in each step of a search.
- The number of neighbours to be randomly sampled is restricted by a user specified parameter. In this way, CLARANS does not confine the search to a localized area. If a better neighbour is found or we can say having a lower error, CLARANS moves to the neighbour's node and the process starts again, or else, the current clustering produces a local minimum.

- If a local minimum is found, CLARANS starts with new randomly selected nodes in search for a new local minimum. Once a user-specified number of local minima has been found, the algorithm outputs, as a solution, the best local minimum, that is, the local minimum having the lowest cost.
- CLARANS has been experimentally shown to be more effective than both PAM and CLARA. It can be used to find the most "natural" number of clusters using a silhouette coefficient, a property of an object that specifies how much the object truly belongs to the cluster.
- CLARANS also enables the detection of outliers. However, the computational complexity of CLARANS is about O(n2), where n is the number of objects. Besides, its clustering quality is dependent on the sampling method used.
- The ability of CLARANS to deal with data objects that reside on disk can be further improved by focusing techniques that explore spatial data structures, such as R* trees.

1.6 HIERARCHICAL ALGORITHMS: AGGLOMERATIVE (AGNES); DIVISIVE (DIANA),

- A hierarchical clustering method is a grouping of data objects into a tree of clusters.
- Hierarchical clustering methods can be classified as either agglomerative or divisive, depending on whether the hierarchical decomposition is formed in a bottom-up also called merging or top-down like splitting fashion.
- In general, there are two types of hierarchical clustering methods:
    - Agglomerative hierarchical clustering:
        - This bottom-up plan starts by placing each object in its own cluster and then merges these atomic clusters into larger and larger clusters, until all of the objects are in a single cluster or until certain termination conditions are fulfilled. Most clustering methods are part of this category. They differ only in their definition of inter cluster similarity.
    - Divisive hierarchical clustering:
        - This top-down plan does the reverse of agglomerative hierarchical clustering by beginning with all objects in one cluster. It subdivides the cluster into smaller and smaller pieces, till each object forms a cluster on its own or until it satisfies certain termination conditions, such as a desired number of clusters is obtained or the diameter of each cluster is within a certain threshold.

- Example:
  - Agglomerative versus divisive hierarchical clustering. Figure 7 shows the application of AGNES (Agglomerative Nesting), an agglomerative hierarchical clustering method, and DIANA (Divisive Analysis), a divisive hierarchical clustering method, to a data set of five objects, fa, b, c, d, e.g. Initially, AGNES places each object into a cluster of its own. The clusters are then merged step-by-step according to some criterion. For example, clusters C1 and C2 may be merged if an object in C1 and an object in C2 form the minimum Euclidean distance between any two objects from different clusters. This is a single-linkage approach in that each cluster is represented by all of the objects in the cluster, and the similarity between two clusters is measured by the similarity of the closest pair of data points belonging to different clusters. The cluster merging process repeats until all of the objects are eventually merged to form one cluster.
  - In DIANA, all of the objects are used to form one initial cluster. The cluster is split according to some principle, such as the maximum Euclidean distance between the closest neighbouring objects in the cluster. The cluster splitting process repeats until, eventually, each new cluster contains only a single object.

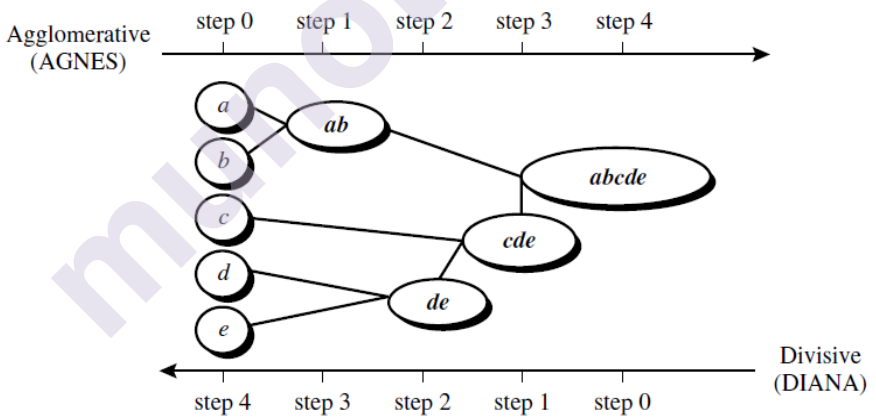


Fig. 7 Agglomerative and divisive hierarchical clustering on data objects {a, b, c, d, e}.

  - In either agglomerative or divisive hierarchical clustering, the user can specify the desired number of clusters as a termination condition.
  - A tree structure called a dendrogram is repeatedly used to act for the process of hierarchical clustering. It represents the objects that are grouped together step by step. Figure 8 shows a dendrogram for the five objects presented in Figure 7, where l = 0 shows the five objects as singleton clusters at level 0. At l = 1, objects a and b are grouped

together to form the first cluster, and they stay together at all subsequent levels. We can also use a vertical axis to show the similarity scale between clusters. For example, when the similarity of two groups of objects, fa, bg and fc, d, eg, is roughly 0.16, they are merged together to form a single cluster.

- Four widely used measures for distance between clusters are as follows, where |p-p'|is the distance between two objects or points, p and p0; mi is the mean for cluster,Ci; and ni is the number of objects in Ci.

$$\text{Minimum distance}: \quad d_{min}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} |p - p'|$$

$$\text{Maximum distance}: \quad d_{max}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} |p - p'|$$

$$\text{Mean distance}: \quad d_{mean}(C_i, C_j) = |m_i - m_j|$$

$$\text{Average distance}: \quad d_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} |p - p'|$$

- When an algorithm uses the minimum distance, dmin(Ci, Cj), to measure the distance between clusters, it is sometimes called a nearest-neighbour clustering algorithm. If the clustering process is terminated then the distance between nearest clusters exceeds an arbitrary threshold, it is called a single-linkage algorithm.
- If we view the data points as nodes of a graph, with edges forming a path between the nodes in a cluster, then the merging of two clusters, Ci and Cj, corresponds to adding an edge between the nearest pair of nodes in Ci and Cj.
- Edges linking clusters always present between distinct clusters, the resulting graph will generate a tree. Thus, an agglomerative hierarchical clustering algorithm that uses the minimum distance measure is also called a minimal spanning tree algorithm.

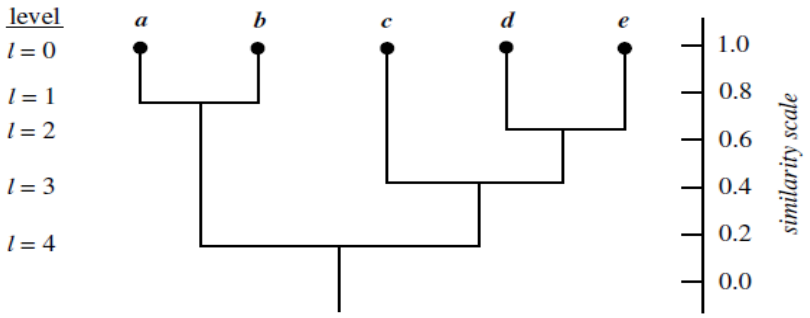


Fig. 8 Dendrogram representation for hierarchical clustering of data objects {a, b, c, d, e}.

1.7 DENSITY BASED CLUSTERING: DBSCAN, CLUSTERING IN NONEUCLIDEAN SPACES, CLUSTERING OR STREAMS AND PARALLELISM

- While discovering clusters with arbitrary shape, density-based clustering methods have been developed. These usually cluster as dense regions of objects in the data space that are separated by regions of low density. DBSCAN develops clusters according to density-based connectivity examinations. OPTICS increases DBSCAN to build a cluster ordering obtained from a wide range of parameter settings. DENCLUE clusters objects formulted on from a set of density distribution functions.

- DBSCAN (Density-Based Spatial Clustering of Applications with Noise) implemented using a density based clustering algorithm. The algorithm grows with sufficiently high density into clusters and discovers clusters of arbitrary shape in spatial databases with noise. It shows a cluster as a maximal set of density connected points.

Density-reachable:

- A point p is density-reachable from a point q wrt. Eps, MinPts if there is a chain of points p1, …, pn, p1 = q, pn = p such that pi+1 is directly density-reachable from pi

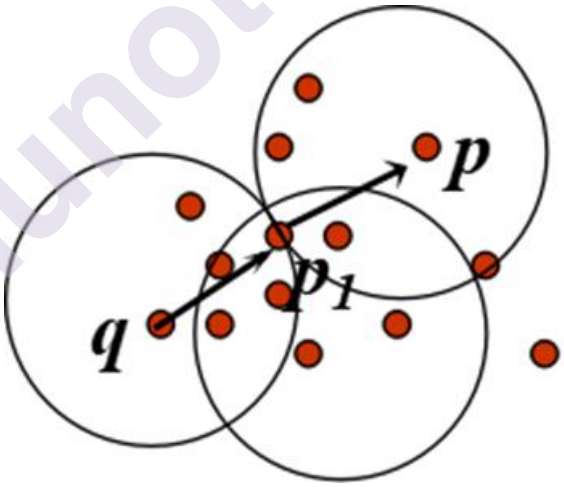


Fig. 9 Density-reachable

Density-connected

- A point p is density-connected to a point q wrt. Eps, MinPts if there is a point o such that both, p and q are density-reachable from o wrt. Eps and MinPts.

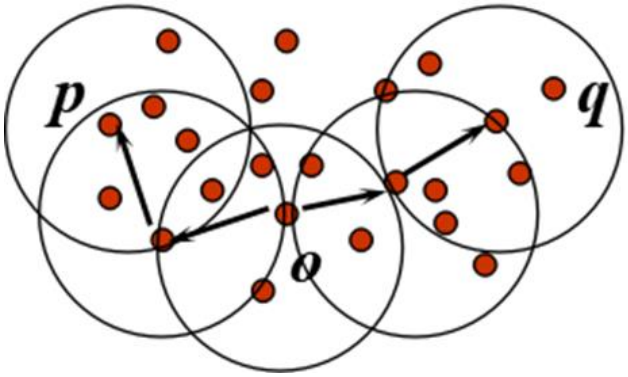


Fig. 10. Density-connected

Working of Density-Based Clustering

- Given a set of objects, D' we say that an object p is directly density-reachable from object q if p is within the ε-neighbourhood of q, and q is a core object.
- An object p is density-reachable from object q with respect to ε and MinPts in a set of objects, D' if there is a chain of objects p1,.,.,.pn, where p1 = q and pn = p such that pi+1 is directly density-reachable from pi with respect to e and MinPts, for 1/n, pi € D.
- An object p is density-connected to object q with respect to ε and MinPts in a set of objects, D', if there is an object o, belongs to D such that both p and q are density-reachable from o with respect to ε and MinPts.

Density-Based Clustering - Background

Two parameters:

- Eps: Maximum radius of the neighborhood.
- MinPts: Minimum number of points in an Eps-neighborhood of that point.

  NEps(p): {q belongs to D | dist(p,q) <= Eps}

  Directly density-reachable: A point p is directly density-reachable from a point q wrt. Eps, MinPts if

- p belongs to NEps(q)
- core point condition:|NEps (q)| >= MinPts

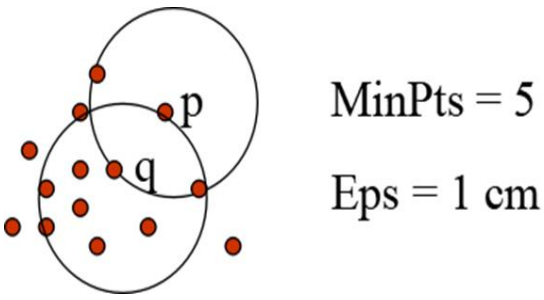


Fig. 11 Density-Based Clustering

Major features:

- It is used to discover clusters of arbitrary shape.
- It is also used to handle noise in the data clusters.
- It is a one scan method.
- It needs density parameters as a termination condition.

Density-Based Methods

DBSCAN: Ester, et al. (KDD'96)

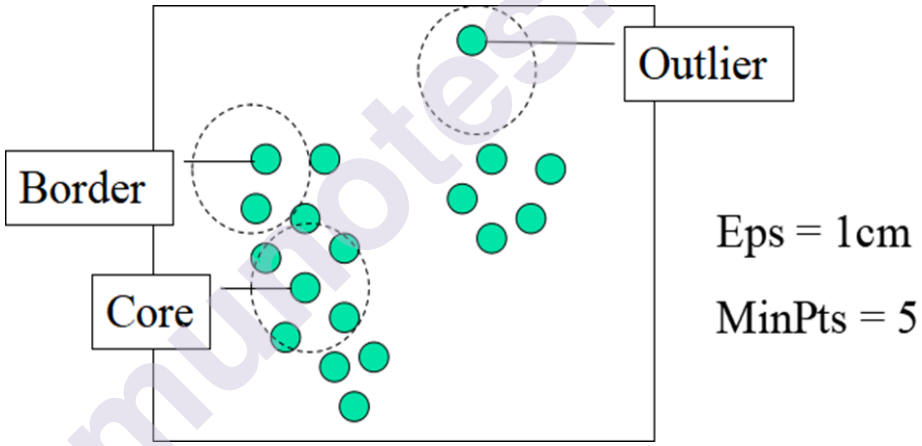
OPTICS: Ankerst, et al (SIGMOD'99)

DENCLUE: Hinneburg & D. Keim (KDD'98)

CLIQUE: Agrawal, et al. (SIGMOD'98)

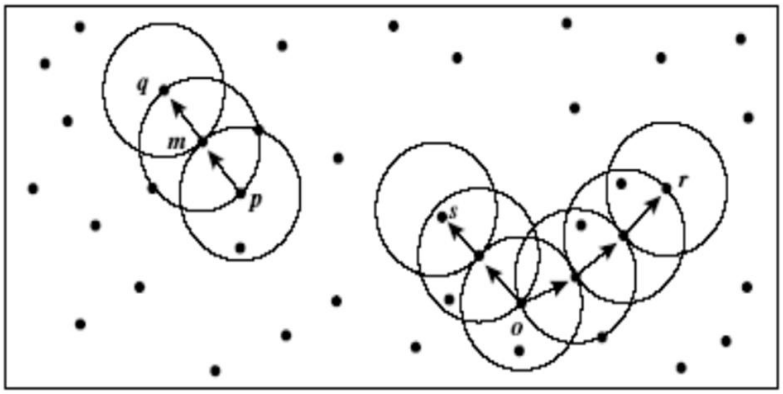
DBSCAN(Density-Based Spatial Clustering of Applications with Noise)

- It relies on a density-based notion of cluster: A cluster is defined as a maximal set of density-connected points.It discovers clusters of arbitrary shape in spatial databases with noise.



DBSCAN Algorithm

1. Arbitrary select a point p.
2. Retrieve all points density-reachable from p wrt Eps and MinPts.
3. If p is a core point, a cluster is formed.
4. If p is a border point, no points are density-reachable from p and DBSCAN visits the next point of the database.
5. Continue the process until all of the points have been processed.



- Example: let MinPts = 3.Of the labelled points, m, p, o, and r are core objects because each is in an ε-neighbourhood containing at least three points. q is directly density-reachable from m. m is directly density-reachable from p and vice versa. q is (indirectly) density-reachable from p because q is directly density-reachable from m and m is directly density-reachable from p. However, p is not density-reachable from q because q is not a core object. Similarly, r and s are density-reachable from o, and o is density-reachable from o, and o is density-reachable from R.

Hierarchical Clustering in Non-Euclidean Spaces

- When the space is non-Euclidean, we need to use some distance measure that is computed from points, such as Jaccard, cosine, or edit distance. That is, we cannot base distances on "location" of points.
- A problem arises when we need to represent a cluster, because we cannot replace a collection of points by their centroid.
- Given that we cannot combine points in a cluster when the space is non-Euclidean, our only choice is to pick one of the points of the cluster itself to represent the cluster. Ideally, this point is close to all the points of the cluster, so it in some sense lies in the "center."
- We call the representative point the clustroid. We can select the clustroid in various ways, each designed to, in some sense, minimize the distances between the clustroid and the other points in the cluster. Common choices include selecting as the clustroid the point that minimizes:
    1. The sum of the distances to the other points in the cluster.
    2. The maximum distance to another point in the cluster.
    3. The sum of the squares of the distances to the other points in the cluster.

Clustering for Streams and Parallelism

- The model we have in mind is one where there is a sliding window of N points, and we can ask for the centroids or clustroids of the best clusters formed from the last m of these points, for any $m \leq N$.

- This provides only a rough outline to suggest the possibilities, which depend on our assumptions about how clusters evolve in a stream.

1. The Stream-Computing Model

- The stream element is a point in some space. The sliding window consists of the most recent N points. Our goal is to precluster subsets of the points in the stream, so that we may quickly answer queries of the form "what are the clusters of the last m points?" for any m ≤ N. There are many variants of this query, depending on what we assume about what constitutes a cluster. For instance, we may use a k-means approach, where we are really asking that the last m points be partitioned into exactly k clusters.

2. A Stream-Clustering Algorithm

In this algorithm, we present a simplified version of an algorithm referred to as BDMO (B. Babcock, M. Datar, R. Motwani, and L. O'Callaghan). The true version of the algorithm involves much more complex structures, which are designed to provide performance guarantees in the worst case.

3. Initializing Buckets

Our smallest bucket size will be p, a power of 2. Thus, every p stream element, we create a new bucket, with the most recent p points. The timestamp for this bucket is the timestamp of the most recent point in the bucket. We may leave each point in a cluster by itself, or we may perform a clustering of these points according to whatever clustering strategy we have chosen.

4. Merging Buckets

After creating a new bucket, we need to review the sequence of buckets. First, if some bucket has a timestamp that is more than N time units prior to the current time, then nothing of that bucket is in the window, and we may drop it from the list. Second, we may have created three buckets of size p, in which case we must merge the oldest two of the three. The merger may create two buckets of size 2p, in which case we may have to merge buckets of increasing sizes, recursively.

1.8 SUMMARY

- A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. The process of grouping aset of physical or abstract objects into classes of similar objects is called clustering.

- Density-Based Clustering -> It is one of the clustering methods based on density (local cluster criterion), such as density-connected points.

- The quality of clustering can be assessed based on a measure of dissimilarity of objects, which can be computed for various types of data, including interval-scaled, binary, categorical, ordinal, and ratio-scaled variables, or combinations of these variable types. For nonmetric vector data, the cosine measure and the Tanimoto coefficient are often used in the assessment of similarity.

- A partitioning method first creates an initial set of k partitions, where parameter k is the number of partitions to construct. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another. Typical partitioning methods include k-means, k-medoids, CLARANS, and their improvements.

1.9 REFERENCE FOR FURTHER READING

- Mining of Massive Datasets, Anand Rajaraman and Jeffrey David Ullman, Cambridge University Press, 2012.
- Data Mining: Introductory and Advanced Topics, Margaret H. Dunham, Pearson, 2013.
- Data Mining:Concepts and Techniques,Second Edition, Jiawei Han,University of Illinois at Urbana-Champaign,Micheline Kamber

1.10 UNIT END EXERCISES

1. Explain with an example.

a. K-Means

b. K-Medoids

2. Explain the Partitioning Algorithm for a large data set, with example?

3. Explain the Hierarchical Algorithms: Agglomerative (AGNES), Divisive (DIANA).

❄❄❄❄❄❄❄

CLASSIFICATION

Unit Structure :

2.0 OBJECTIVE

1. To study and understand challenges of big data.
2. To study and understand distance-based algorithm (KNN).
3. To study and understand Trees based classifiers.
4. To study and understand statistical based classifiers.
5. To study and understand Bayesian networks.

2.1 INTRODUCTION TO BIG DATA

We all are surrounded by huge data. In every milli-seconds people upload or downloads the videos, images, audios from various devices. Using different social media people are sending text messages, updating facebook, multimedia messages, Whatsapp, Twitter, Use of Amazon, Flipkart, online advertising etc generates huge amount of data.

As a result of multiple processing machines have to generate and keep huge data too. Due to exponential growth of data, Data analysis is very much required task for day to day operations.

'Big Data' means huge volume, high velocity and variety of data.

2.1.1 Challenges of Big Data

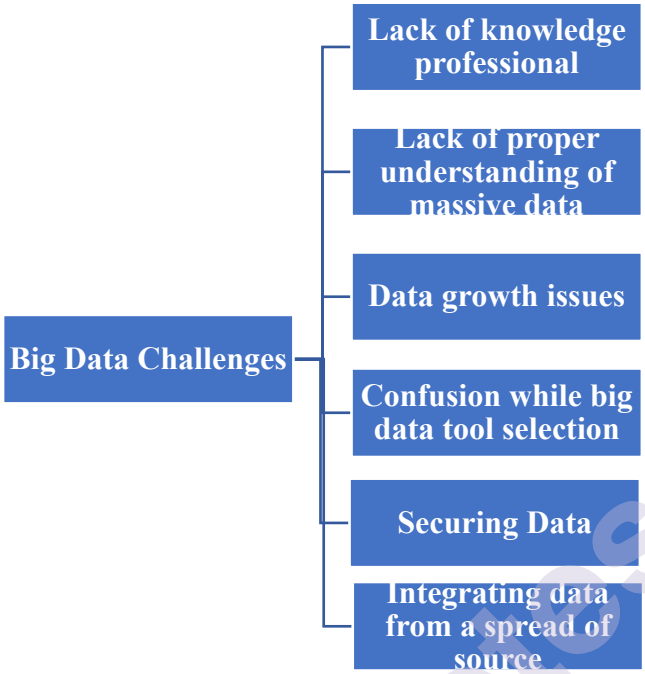


Figure 2.1 Big data challenges

Challenges include the best way of handling the numerous amounts of data that involves the process of storing, analysing the huge set of information on various data stores. There are various major challenges that come into the way while dealing with Big Data which need to be taken care of with Agility. Top 6 Big Data Challenges

1. Lack of knowledge Professionals

To run these modern technologies and large Data tools, companies need skilled data professionals. These professionals will include data scientists, data analysts, and data engineers to work with the tools and make sense of giant data sets. One of the Big Data Challenges that any Company face is a drag of lack of massive professionals. This is often because data handling tools have evolved rapidly, but in most cases, the professionals haven't. Actionable steps got to be taken to bridge this gap.

Solution

Companies are investing extra money in the recruitment of skilled professionals. They even have to supply training programs to the prevailing staff to urge the foremost out of them. Another important step taken by

organizations is purchasing knowledge analytics solutions powered by artificial intelligence/machine learning.

These Big Data Tools are often suggested by professionals who aren't data science experts but have the basic knowledge. This step helps companies to save lots of tons of cash for recruitment.

2. Lack of proper understanding of Massive Data

Companies fail in their Big Data initiatives, all thanks to insufficient understanding. Employees might not know what data is, its storage, processing, importance, and sources. Data professionals may know what's happening, but others might not have a transparent picture. For example, if employees don't understand the importance of knowledge storage, they could not keep the backup of sensitive data. They could not use databases properly for storage. As a result, when this important data is required, it can't be retrieved easily.

Solution

Big Data workshops and seminars must be held at companies for everybody. Military training programs must be arranged for all the workers handling data regularly and are a neighborhood of large Data projects. All levels of the organization must inculcate a basic understanding of knowledge concepts.

3. Data Growth Issues

One of the foremost pressing challenges of massive Data is storing these huge sets of knowledge properly. the quantity of knowledge being stored in data centers and databases of companies is increasing rapidly. As these data sets grow exponentially with time, it gets challenging to handle. Most of the info is unstructured and comes from documents, videos, audio, text files, and other sources. This suggests that you cannot find them in the database. Companies choose modern techniques to handle these large data sets, like compression, tiering, and deduplication. Compression is employed for reducing the number of bits within the data, thus reducing its overall size. Deduplication is the process of removing duplicate and unwanted data from a knowledge set. Data tiering allows companies to store data in several storage tiers. It ensures that the info is residing within the most appropriate space for storing. Data tiers are often public cloud, private cloud, and flash storage, counting on the info size and importance.

Companies also are choosing Big Data tools, like Hadoop, NoSQL, and other technologies.

4. Confusion while Big Data Tool selection

Companies often get confused while selecting the simplest tool for giant Data analysis and storage. Is HBase or Cassandra the simplest technology for data storage? Is Hadoop MapReduce ok, or will Spark be a far better option for data analytics and storage? These questions bother companies,

and sometimes they're unable to seek out the answers. They find themselves making poor decisions and selecting inappropriate technology. As a result, money, time, efforts, and work hours are wasted.

Solution

You'll either hire experienced professionals who know far more about these tools. Differently is to travel for giant Data consulting. Here, consultants will provide a recommendation of the simplest tools supporting your company's scenario. Supporting their advice, you'll compute a technique then select the simplest tool for you.

5. Integrating Data from a Spread of Sources

Data in a corporation comes from various sources, like social media pages, ERP applications, customer logs, financial reports, e-mails, presentations, and reports created by employees. Combining all this data to organize reports may be a challenging task. This is a neighborhood often neglected by firms. Data integration is crucial for analysis, reporting, and business intelligence, so it's perfect.

Solution

Companies need to solve their Data Integration problems by purchasing the proper tools. A number of the simplest data integration tools are mentioned below:

1. Talend Data Integration
2. Centerprise Data Integrator
3. ArcESB
4. IBM InfoSphere
5. Xplenty
6. Informatica PowerCenter
7. CloverDX
8. Microsoft SQL QlikView

6. Securing Data

Securing these huge sets of knowledge is one of the daunting challenges of massive Data. Often companies are so busy in understanding, storing, and analyzing their data sets that they push data security for later stages. This is often not a sensible move as unprotected data repositories can become breeding grounds for malicious hackers. Companies can lose up to $3.7 million for a stolen record or a knowledge breach.

Solution

Companies are recruiting more cybersecurity professionals to guard their data. Other steps taken for Securing Big Data include: Data encryption Data segregation Identity and access control Implementation of endpoint security Realtime security monitoring Use Big Data security tools, like IBM Guardian.

Big data challenges

In connection with the processing capacity issues, designing a big data architecture is a common challenge for users. Big data systems must be tailored to an organization's particular needs, a DIY undertaking that requires IT and data management teams to piece together a customized set of technologies and tools. Deploying and managing big data systems also require new skills compared to the ones that database administrators and developers focused on relational software typically possess.

Both of those issues can be eased by using a managed cloud service, but IT managers need to keep a close eye on cloud usage to make sure costs don't get out of hand. Also, migrating on-premises data sets and processing workloads to the cloud is often a complex process. Other challenges in managing big data systems include making the data accessible to data scientists and analysts, especially in distributed environments that include a mix of different platforms and data stores. To help analysts find relevant data, data management and analytics teams are increasingly building data catalogues, that incorporate metadata management and data lineage functions. The process of integrating sets of big data is often also complicated, particularly when data variety and velocity are factors.

Big data challenges with respect to classification

Big Data is currently defined using three data characteristics: volume, variety and velocity. It means that some point in time, when the volume, variety and velocity of the data are increased, the current techniques and technologies may not be able to handle storage and processing of the data. The first challenging problem rests on the current definition of Big Data; how to prove (or show) that the network traffic data satisfy the Big Data characteristics for Big Data classification. This is the first important task to address in order to make the Big Data analytics efficient and cost effective.

1. Network topology

The cardinality issue in Big Data analytics can be helped by contemporary computer technologies like public cloud and HDFS (Hadoop Distributed File System). They can be combined to create sizable, adaptable network architecture with storage infrastructure that can alter accordingly to the demands of Big Data processing. However, this integrated approach will provide a number of difficulties that must be successfully overcome.

2. Communication challenges

The communication cost is the primary concern in computer networking research and applications when compared to the processing cost of the data in this proposed topology. The difficulty here is to keep the cost of connectivity to a minimum while yet meeting the public cloud's increased storage and data needs for processing big data. The two main network characteristics that will impact communication between clients and the cloud server are bandwidth and latency. The timing requirements for the Big Data processing at HDFS and UILS will be negatively impacted by these issues and the difficulties associated with finding solutions. As a result, these issues and difficulties must be taken into consideration while developing machine learning approaches that make use of these technologies.

3. Security challenges

In general, cloud technology has a poor security system. Data manipulation in the public cloud is therefore inevitable and a major risk. It is difficult to find a reliable security system that can be used with public clouds like CCSS. An attacker can simply alter the data being sent between the HDFS and NTRS devices, the CCSS server, and other cloud computing components. The attacker can even fake the responses sent back and forth and use a DOS attack to bring down the server (CCSS). These issues can make it difficult to execute the big data analytics tool with the recommended network design.

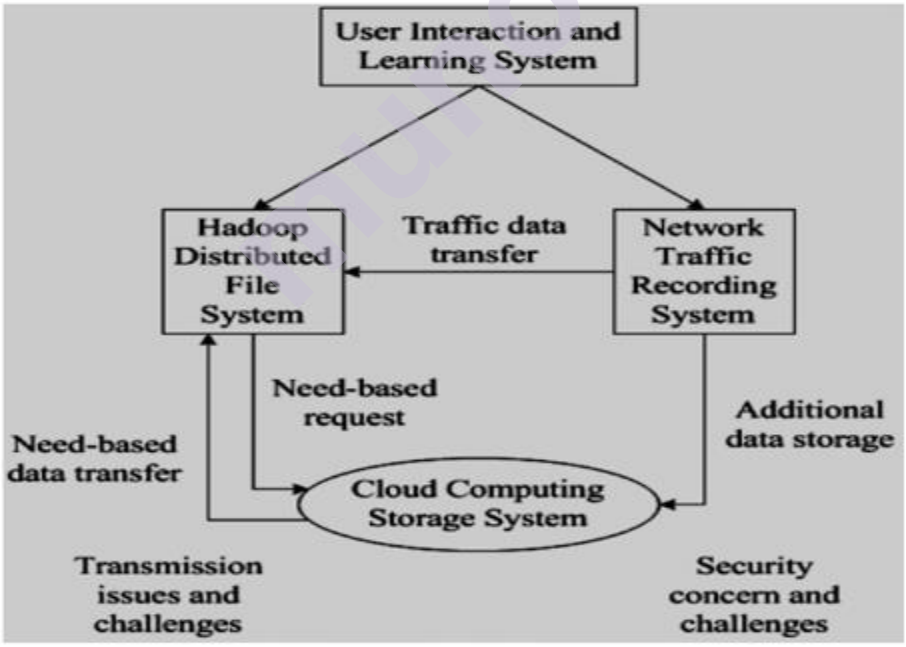


Figure 3.2 Network topology for Big Data Analytics

[Source: researchgate]

2.2 DISTANCE BASED ALGORITHM

Introduction

Distance based algorithms are nonparametric methods that can be used for classification. These algorithms classify objects by the dissimilarity between the two objects and as measured by distance functions.

Distance measures play an important role in machine learning.

A distance measure is an objective score that summarizes the relative difference between two objects in a problem domain.

Most commonly, the two objects are rows of data that describe a subject (such as a person, car, or house), or an event (such as a purchase, a claim, or a diagnosis).

Perhaps the most likely way you will encounter distance measures is when you are using a specific machine learning algorithm that uses distance measures at its core. The most famous algorithm of this type is the k-nearest neighbors algorithm, or KNN for short.

1. Distance Based Algorithm:

- Machine learning methods known as distance-based algorithms categorise queries by measuring the separations between them and a set of internally stored exemplars. The categorization given to the query is most strongly influenced by examples that are nearest to it. The closest neighbour algorithm and the nearest-hyperrectangle method are two particular distance-based algorithms that are thoroughly examined.

- The k-closest neighbour algorithm (kNN) is proved to perform better than the first nearest neighbour method only under specific circumstances. Moderate noise must be present in data sets. Training examples from each class must be located in clusters that permit an increase in k without affecting clusters from other classes. Investigated are various approaches to selecting the value of k for kNN. It was demonstrated that the best performance may be achieved using one-fold cross-validation on a small number of k-values. Additionally, it is demonstrated that, for optimal performance, the votes of a query's k-nearest neighbours should be given weights that are inversely proportional to their distances from the query.

2.2.1 K Nearest Neighbour (KNN)

K-NN or K Nearest Neighbour algorithm is a non-parametric (it does not make any assumption on underlying data), supervised learning classifier. It uses proximity to make classifications or predictions about grouping of individual data points. This algorithm assumes the similarity between new data and available data and put new data into categories that is most similar to available categories. K-NN is also called as lazy learner algorithm as it

does not learn from the training set immediately instead it stores the dataset and at the time of classification it performs actions on the dataset.

Working of K-NN

1. Select number 'k' of the neighbours.
2. Calculate Euclidean distance of 'k' number of neighbours.
3. Take these 'k' neighbours as per there calculated Euclidean distance
4. Among these 'k' neighbours, count the number of data point in each category.
5. Assign the new data points to that category for which the number of neighbour is maximum.

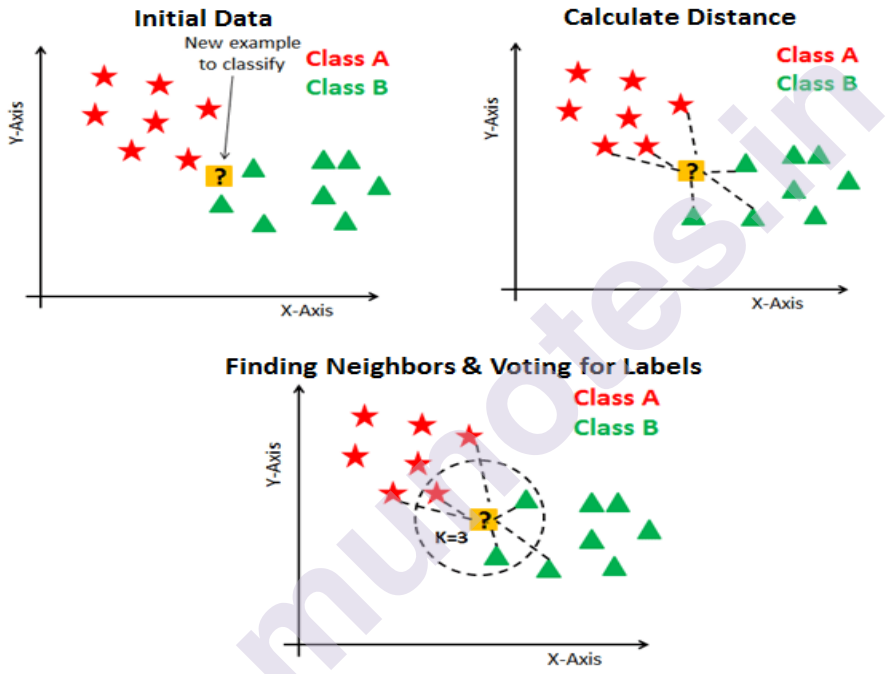


Figure 2.3: Finding Neighbors and Voting for labels

[Source: DataCamp]

- In numerous disciplines, it has been demonstrated that principal component analysis significantly reduces the number of pertinent dimensions. There are two suggested ways to determine the feature weights for a weighted Euclidean distance metric. In a number of fields, these techniques enhance the performance of kNN and NN.
- In a variety of fields, it is discovered that the nearest-hyperrectangle algorithm (NGE) provides predictions that are significantly inferior to those provided by kNN. The results of experiments conducted to comprehend this subpar performance resulted in the discovery of various enhancements to NGE. The most prominent of them is BNGE, a batch method that prevents the creation of overlapping

hyperrectangles from several classes. Even though it typically outperforms NGE, BNGE still falls short of kNN in a number of areas. As a result, a hybrid approach (KBNGE) is proposed that employs BNGE when the input space can be represented by a single hyperrectangle and kNN when it cannot.

- The main contributions of this dissertation are (a) a number of enhancements to already-existing distance-based algorithms, (b) a number of brand-new distance-based algorithms, and (c) a better understanding of the circumstances in which different distance-based algorithms are likely to perform well based on experimental evidence.

2.2.2 KD Tree:

A binary tree structure may be used to illustrate this partitioning process, as seen in figure 5-14. The two children of each node are the sub-regions that come from splitting on one of the dimensions, while the root node represents the entire space. Each node represents a region of the space. Since it represents a k-dimensional space, this data structure is referred to as a k-d tree (notice that this k is distinct from the k in k-NN). While in k-NN it's the number of nearest neighbours we use for inference, in k-d trees it's the dimension of our space. The same tree structure is used to establish where a point is located. We may move down the tree until we reach the leaf node that represents the region of the point by using the values of the point's components in each dimension, we can see in the figure

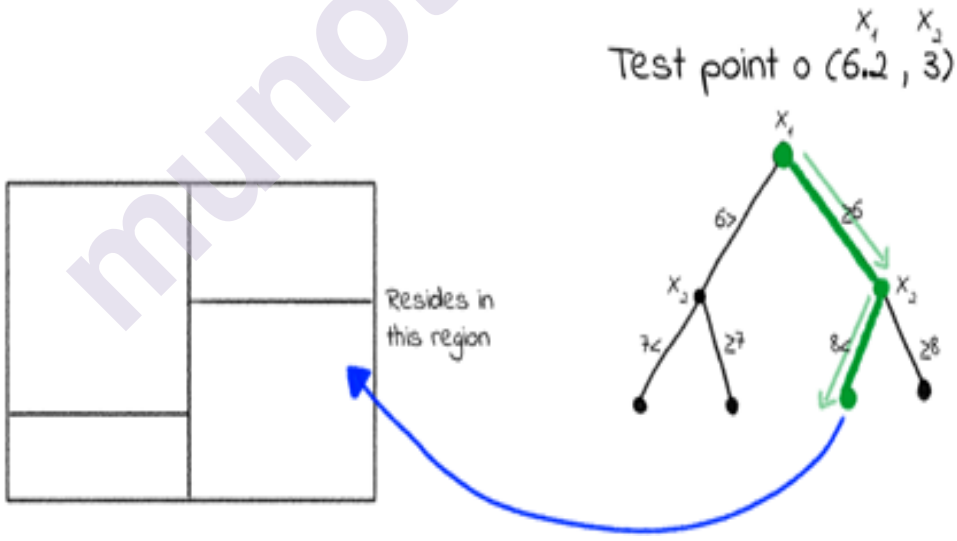


Figure 2.4 KD Tree

[Source: Manning publication]

2.2.3 Rules and Tree Based Classifiers

It is one type of classifier which makes the classes of decisions, with the help of "if….else" rules. These rules are easily understandable and with the help of rules we can generate descriptive model with these classifiers. These are static in nature.

The condition of "if" is known as antecedent and the else part i.e., predicted class of each different rule is known as consequent.

Defining IF-Then Rule

1. Rule Antecedent: It consist of "if condition" part of the rule. It is present in the Left hand side. Rule may have one or more attributes as conditions, with logic AND operator.

2. Rule Consequent: It is present in rule's right hand side. The rule consequent contains the class prediction.

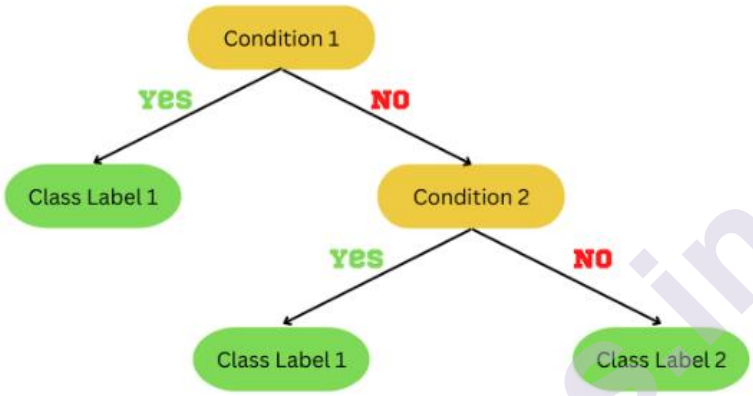


Figure 2.5: Rule Classifiers

[Source: codingninajs.com]

2.2.3.1 Assessment of Rule

There are two different factors based on the rules.

1. Coverage of rule: Number of records which satisfy the antecedent conditions of a particular rule.

   Number of records satisfying the rule = n1

   Total number of records = n

   We can calculate coverage of rule, coverage

$$= \frac{\textbf{Number of records satisfying the rule}}{\textbf{Total number of records}} = \frac{\textbf{n1}}{\textbf{n}}$$

2. Accuracy of rule: Number of records that satisfy the antecedent condition and meet the consequent values of a rule.

   The number of records satisfying the consequent values = n2

   Total number of records satisfying the rule = n1

   We can calculate accuracy of rule,

$$\text{Accuracy} = \frac{\textbf{Number of records satisfying the consequent}}{\textbf{Total number of records satisfying the rule}} = \frac{\textbf{n2}}{\textbf{n1}}$$

2.2.3.2 Properties of Rule-Based Classifiers

1. Rules may not be mutually exclusive in nature: In this we use ordered set of rules, With the help of ordering the rules, we can set the priority of orders. This priority orders set of rules is known as a decision list. So the class which has the highest priority rule is get selected for final class.

   One more solution for the same problem is that we can assigning votes for each class depending on their weights, in this case the set of rules are remain unordered.

2. Rules may not be exhaustive: There is no guarantee that the rule will cover the entire data entries. If any rule leaves some data entries then we can make the rules on non-exhaustive. Here we can define the default class, by using this default class we can assign all the data entries which are not covered by any rules to the default class. With the help of default class we can resolve the problem of non-exhausitivity.

2.2.3.3 Implementation of Rule-Based Classifier

There are two different methods for implementing the rule-based classification.

1. Direct Method: In this method the algorithm extract rules directly from the dataset. There are two different algorithms we are using for direct method of rule based classifier namely, 1R algorithm and Sequentia covering algorithm

1R algorithm: It is also known as Learn One Rule Algorithm. Simple classification rule is applied on it, so it is one of the simplest algorithms. We can make rules to test each and every attribute in the dataset. The rule is depend on the number of attributes in each class. By assigning the attribute value pair to each class and find the maximum frequency of each class. Select the record which has lowest error rate for each attribute.

Disadvantages of 1R algorithms are high chance of overfitting and noise sensitivity. To overcome these disadvantages, we use sequential covering algorithm.

Sequential Covering Algorithm: It is useful for rule base classification. It defines number of rules to cover the maximum data records of one class and no data records of other classes. In this algorithm rules are defined sequentially where after a rule is grown, sequentially all data records covered by that rule are eliminated, and this process keeps on repeating for the remaining records. This process automatically stopes when we reach to the stopping criteria.

If the accuracy of the rule is below the required level, then we can reject that rule and stop the execution.

Flow of sequential covering algorithm steps

Step 1: Original State

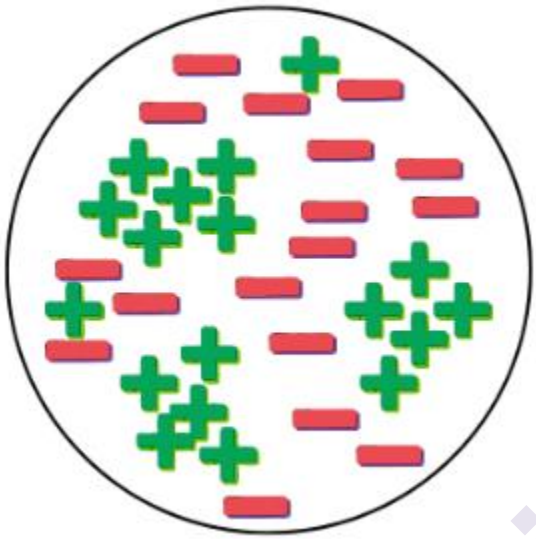


Figure 2.6: Original State

[Source: codingninjas.com]

Step 2 : Apply Rule Gowing (Rule 1)

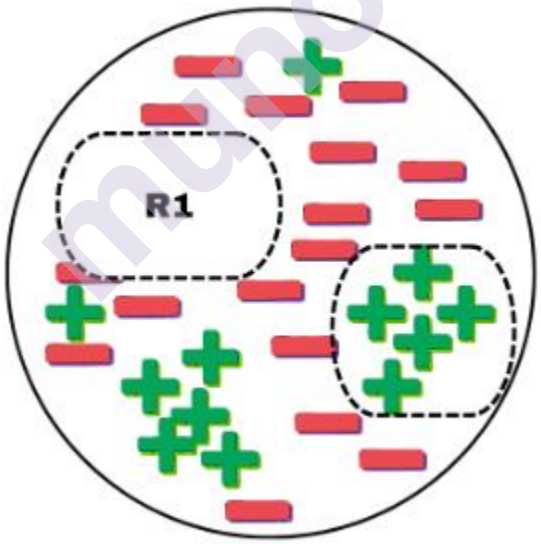


Figure 2.7: Rule 1

[source: codingninjas.com]

Step 3 : Instance Elimination (Removing Rule 1 and growing Rule 2)

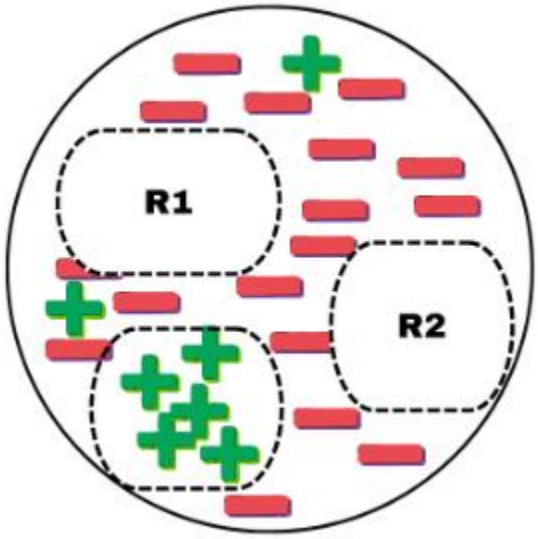


Figure 2.8: Instance Elimination

[Source: codingninjas.com]

Continue this process until we meet the stopping criteria.

2. Indirect Method

In this method we extract rules from different other classification models. Some of the common examples of indirect method are decision trees and neural networks.

For example

Converting the rules from decision tree to the rule set.

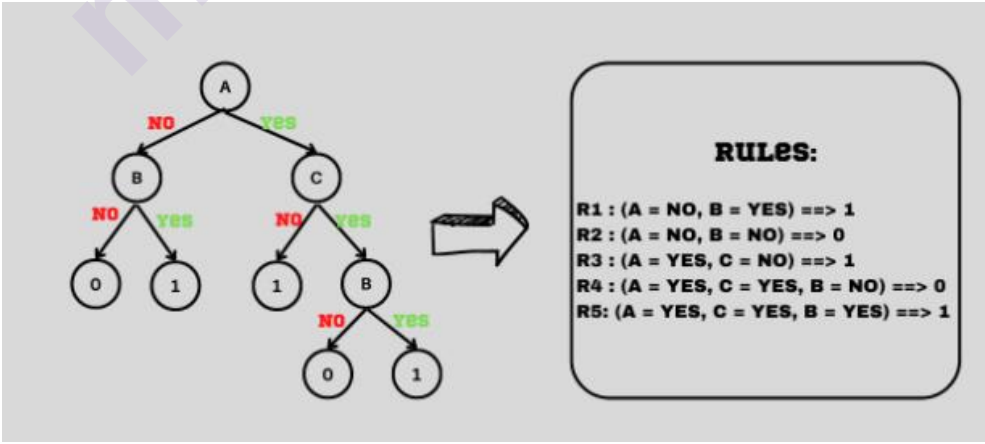


Figure 2.9: Indirect method

[Source: codeingninjas.com]

2.2.3.4 Advantages of rule-based classification

1. Easy to generate rule-based classification.
2. It is very easy to understand and highly expressive in nature.
3. It uses significantly less time and very quickly to assists in the classification of the new data.
4. It helps in handing redundant values during the classification.

2.2.3.5 Tree Based Classifiers

Tree based learning can be said to be one of the famous machine learning tools for classification problems. The simple example of a tree based classifier is decision tree, the goal of which is to split the sample into homogenous group of outcome variable based on series of comparisons.

There are 4 models in tree-based classifiers:

1. Decision tree
2. Bagged trees
3. Random forest
4. Boosted trees

1. Decision trees:

Decision tree is the basic building block of all tree-based classifiers. The decision tree classifier uses a series of decisions to determine the class label. In the case of binary decision trees, one raises yes/no questions. The root node and internal nodes of the tree are the decisions to be made, whereas the terminal nodes (leaves) are the class labels.

2. Bagged trees:

Bagging is short for bootstrap aggregating. As the name indicates, you first bootstrap then average over the results.

Bootstrap

It is a resampling method that generates new dataset from original dataset following the two rules

1. Sample with replacement.
2. Keep sample size the same.
3. Random Forest:

Two ideas are used in the random forest classifier:

- bagging
- random feature subsets for each tree

The second idea makes the individual trees less correlated to each other.

4. Boosted Trees:

The boosted trees classifier contains an iterative process to construct base classifiers. The idea is that you keep creating new classifiers that do well on the data that the old classifiers do not do well on. Typically the base classifiers are trees with depth one (also called stumps) so that data training can be less expensive.

2.2.4 Information Gain Theory:

- ❖ Information Gain, or IG for short, divides a dataset according to a certain value of a random variable to calculate the decrease in entropy or surprise.
- ❖ A higher information gain reflects a group or groups of samples with lower entropy, and so less surprise.
- ❖ You may remember that information quantifies an event's surprise in bits. Events with a lower likelihood contain more information than those with a greater probability. Entropy measures the amount of information contained in the probability distribution of a random variable. A distribution with unequal probabilities of occurrences has a higher entropy than one with a skewed distribution.
- ❖ We often refer to an event's "surprise" in information theory. Low probability occurrences have more information since they are more unexpected. The entropy of probability distributions with equally likely events is higher and they are more startling.
- Skewed Probability Distribution (unsurprising): Low entropy.
- Balanced Probability Distribution (surprising): High entropy.

2.3 STATISTICAL-BASED ALGORITHMS:

A mathematical expression of some characteristics of the patterns they discover in data is a statistical or data mining algorithm. Different algorithms offer various viewpoints on the pattern's overall complexity.

The following are two categories of statistical-based algorithms:

2.3.1 Regression: -

Analysing an output value that depends on input values is the focus of regression issues. The output values define the classifications, whereas the input values utilised for classification are obtained from the database. Although regression has numerous applications, including prediction, it may also be used to address classification issues. The most fundamental kind of regression is simple linear regression, which just contains one predictor and one prediction.

There are two different ways to implement classification using regression, which are as follows:

2. Regions based on class are used to split the data.

3. Formulas are developed to forecast the value of the output class.

Linear Regression:

- Linear regression is a type of supervised leaning algorithm of machine learning.
- It is used to find the best linear relationship that describes the data set.
- It assumes that there exists a linear relationship between a dependent variable and independent variable(s).
- The value of the dependent variable of a linear regression model is a continuous value i.e real numbers

Representation of Linear Regression Model

Linear regression model represents the linear relationship between a dependent variable and independent variables with the help of straight line.

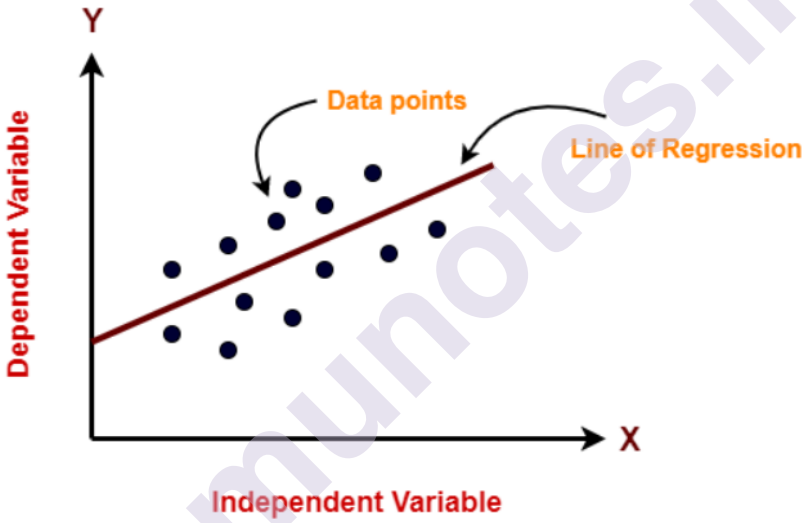


Figure 2.10: Linear Regression Representation

[Source: Javapoint]

The straight line shows the linear relationship between independent and dependent variables that fits the given data set, with best is called as regression line and it is also known as Best fit line.

Types of linear regression:

There are two types of linear regression

1. Simple Linear Regression

2. Multiple Linear Regression

1. Simple Linear Regression

In SLR (simple linear regression), the dependent variable depends only on a single independent variable.

For SLR the form of model is

$$\boldsymbol{Y = \beta_0 + \beta X}$$

Y = dependent variable

X = independent variable

β_0 and β_1 are the regression coefficients

β_0 *is the intercept or the bias that fixes offset to a line*

β_1 is the slope or weight that gives the impact factor on which X has on Y

There are 3 possibilities of cases

Case 1. $\boldsymbol{\beta_1 < 0}$

In this case it indicates that variable X has negative impact on Y.

If X increases, Y will decrease or vice-versa

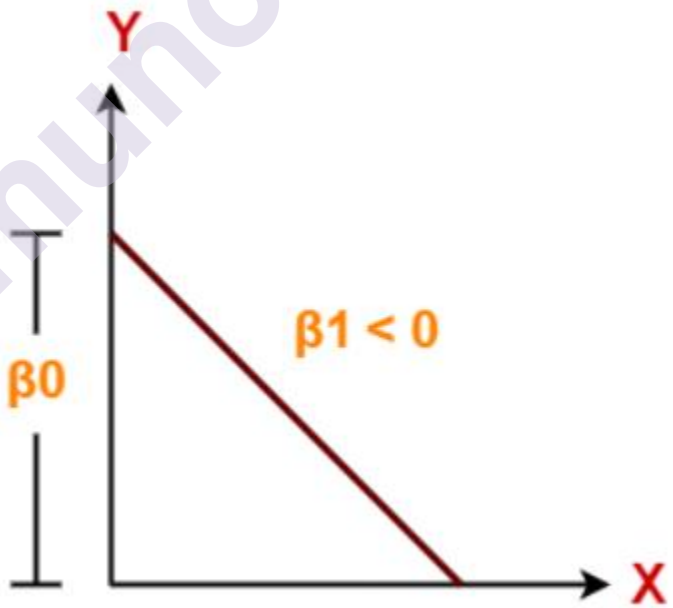


Figure 2.11: SLR Case 1. $\beta_1 < 0$

[Source: GateVidyalay]

Case 2. $\boldsymbol{\beta_1 = 0}$

In this case it shows that variable X has no impact on Y.

If X changes, then there will be no change in Y.

Figure: SLR Case 1. $\beta_1 < 0$

Figure: SLR Case 1. $\beta_1 < 0$

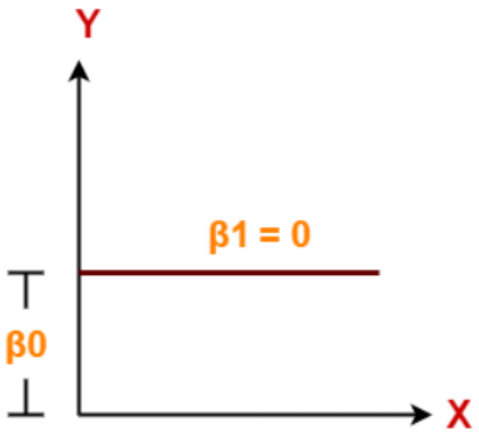


Figure 2.12: SLR Case 1. $\beta_1 = 0$

[Source: GateVidyalay]

Case 3. $\beta_1 > 0$

In this case it shows that the variable X has positive impact on Y.

If X increases, Y will also increase and vice-versa.

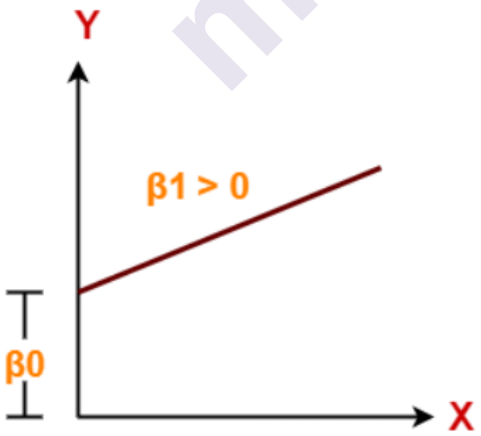


Figure 2.13 SLR Case 3. $B_1 > 0$

[Source: GateVidyalay]

Multiple Linear Regression (MLR):

In MLR, the dependent variable depends on more than one independent variables.

For Multiple Linear Regression, the form of the model is as follows:

$$\boldsymbol{Y = \beta_0 + \beta 1 \gamma = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n}$$

Y is dependent variable

X1, X2, …. Xn are independent variables

$\boldsymbol{\beta_0, \beta_1 \ldots\ldots \beta_n}$ are the regression coefficients

$\boldsymbol{\beta_j}$**(1<=k<=n)** is the slope/weight that specifies the factor by which X_j has an impact on Y.

2.3.2 Bayesian Classification: - Statistical classifiers are used to do the classification. Bayesian classification is based on the Bayes theorem. The efficacy and speed of Bayesian classifiers are quite high when used to huge databases.

2. Document Classification:

To make documents simpler to handle, search for, filter, or analyse, it is necessary to classify them into different categories or groups. In this context, a document is a piece of information with content pertaining to a certain category. Documents include things like emails, comments on products, bills, and document scans.

Text and visual classifications are the two main categories used in document classification tasks.

- The process of classifying texts involves identifying the kind, genre, or subject of the text in light of its content. Complex methods like Natural Language Processing (NLP) can be employed, depending on the objective, to evaluate words and phrases in context and comprehend their meanings (meaning). NLP is used, for instance, in sentiment analysis to identify the emotion or viewpoint represented in the text.

- Visual categorization employs image recognition and computer vision to concentrate on the visual organisation of documents. Both static photos and moving photographs can serve as visual representations. Here, we examine the pixels that make up an image, recognise the things depicted, and categorise them according to their actions or other distinctive characteristics.

2.3.3 Document classification

Document classification real-life use cases:

1. **Spam detection:** To determine if a document is spam, NLP-based classifiers can define spam terms and track their frequency in the text by analysing words in context.
2. **Opinion classification and social listening**: Businesses are interested in learning what their clients think of them. And one of the best methods is to use sentiment analysis to categorise social media comments and reviews according to their emotional content.
3. **Customer support ticket classification**: Customer service representatives often handle a lot of inquiries during the day. In this scenario, a ticket routing job may be implemented using an NLP-based system. The system defines the communication as "claims," "refunds," or "tech support" after analysing the language and forwards it to the appropriate department.
4. **Document scan classification:** Another difficulty in document categorization comes from working with paper-based documents. To extract written or typed material for further analysis, we must first scan them. Text and its arrangement must be discernible from photographs and scans in order for the technology needed for these objectives to make it possible to convert paper documents into digital formats and then categorise them.

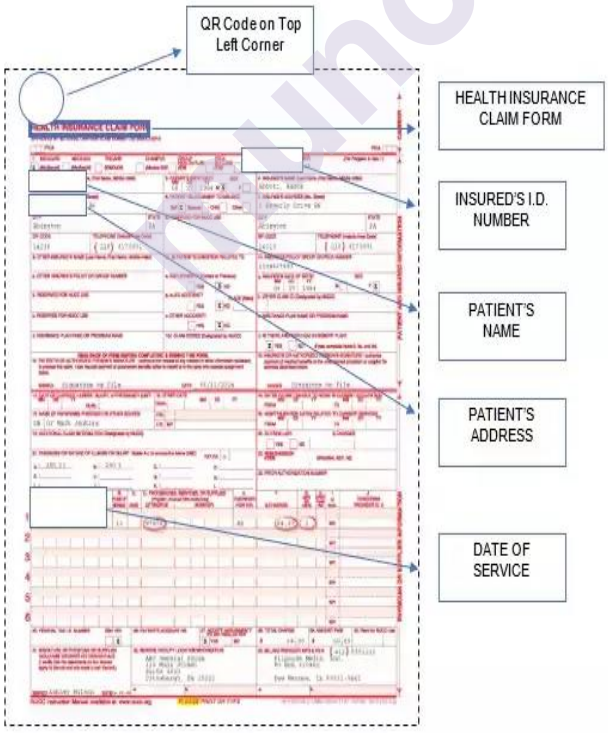


Figure 2.14 An example of document structure in healthcare insurance

[Source: affine.ai)

The insurance sector, which handles tens of thousands of claims each day, is still another such. However, due to the fact that the majority of claims are scans, the workflow is inefficient. It necessitates automatically removing the raw data from the claims and analysing it using NLP. For instance, software provider Wipro offers an NLP classifier to identify bogus insurance claims.

2.3.4 Bayesian Network

Bayesian network is a type of probabilistic graphical model that uses Bayesian interference for probability computations. A Bayesian network is a directed acyclic graph in which each edge corresponds to a conditional dependency and each node corresponds to unique random variable. Judia Pearl (1988) stated that Bayesian networks are graphical model that contains information about causal probability relationship between variables and are often used to aid in decision making. Bayesian network are ideal for taking an event that occurred and predict the likelihood that any one of several possible known cause was the contributing factor.

E.g. a Bayesian network could represent probabilistic relationship between disease and its symptoms. Given symptoms, the network can be used to compute probabilities of presence of various diseases.

Bayesian network models are used in Gene Regulatory Networks, Information Retrieval, Image Processing, Spam Filtering and Turbo Code.

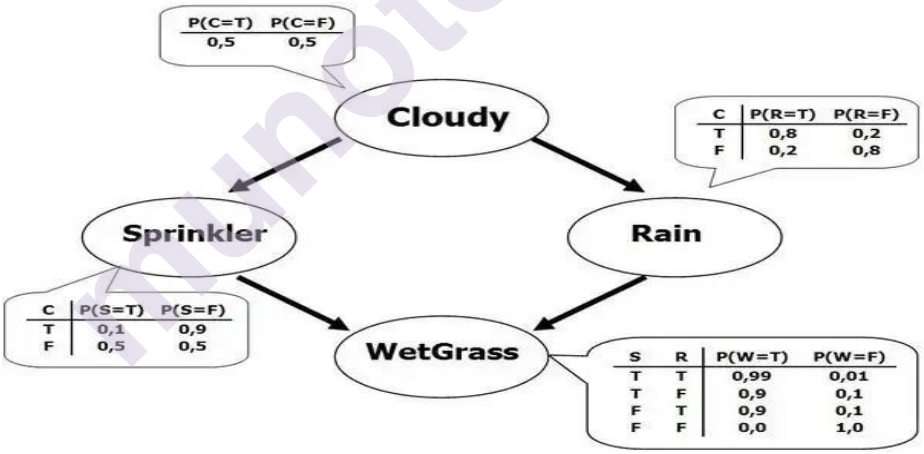


Figure 2.15: Bayesian network

[Source: Github]

SUMMARY

Multiple processing machines of various applications like Face book, Amazon, Flipkart, Whatsapp have to generate and keep huge data too. Due to exponential growth of data, Data analysis is very much required task for day-to-day operations. 'Big Data' means huge volume, high velocity and variety of data In this chapter we have discuss various challenges of Big data along with its solutions. K-NN or K Nearest Neighbour algorithm is a non-parametric (it does not make any assumption on underlying data),

supervised learning classifier. It uses proximity to make classifications or predictions about grouping of individual data points. K-NN is also called as lazy learner algorithm as it does not learn from the training set immediately instead it stores the dataset and at the time of classification it performs actions on the dataset. Bayesian network is a type of probabilistic graphical model that uses Bayesian interference for probability computations. It is a directed graph useful for the various probability computations.

QUESTIONS

Q.1) What is big data? Discuss challenges of big data.

Q.2) Elaborate the distance-based algorithms.

Q.3) Elaborate the statistical-based algorithms.

Q.4) Discuss KNN in detail.

Q.5) Explain in detail Rule-based classifier.

Q.6) What is Regression? Explain different types of regression.

Q.7) Explain Bayesian Network.

Q.8) Give the difference between Regression and Classification.

REFERENCES

1. Baldi, P., and S. Brunak. 1998. Bioinformatics: The Machine Learning Approach. Cambridge, MA: MIT Press.
2. Bilmes, J. A. 2006. “What HMMs Can Do.” IEICE Transactions on Information and Systems E89-D:869–891.
3. Mining of Massive Datasets, Anand Rajaraman and Jeffrey David Ullman, Cambridge University Press, 2012.
4. Data Mining: Introductory and Advanced Topics, Margaret H. Dunham, Pearson, 2013.
5. https://www.gatevidyalay.com/tag/machine-learning-handwritten-notes-pdf/
6. https://www.codingninjas.com/codestudio/library/rule-based-classification-in-data-mining

❄❄❄❄❄❄❄

SUPPORT VECTOR

Unit Structure :

3.0 OBJECTIVE

1) To study and understand Support Vector Machine.

2) To study and understand Confusion Matrix.

3) To study and understand Regression.

4) To study and understand Classification.

5) To study and understand Lift Curves and ROC Curves.

3.1 INTRODUCTION TO SUPPORT VECTOR MACHINE

Support vector machine is popular supervised learning algorithm used for classification and regression problems. It creates decision boundary also known as hyperplane that segregate n-dimensional space into different classes for easy entry for new data points in correct category. Support vectors are data points that are closest to hyperplane and it influence the orientation and position of the hyperplane. The hyperplane depends upon number of features i.e. If number of input feature is 3 then hyperplane becomes 2D plane. We can see that SVM are used in face detection, image classification, text categorization and many more.

Types of Support Vector Machine

Support vector machines are divided into 2 categories namely:

1. Linear: Linear SVM are used for linearly separable data, which means data set can be classified into 2 classes by using single straight line.

Such dataset are called linearly separable data and the mechanism is called linear SVM.

2. Non-linear: Non-linear SVM is used for data set which cannot be classified into 2 classes by using single straight line. Such dataset are called non-linearly separable data and the mechanism is called non-linear SVM.

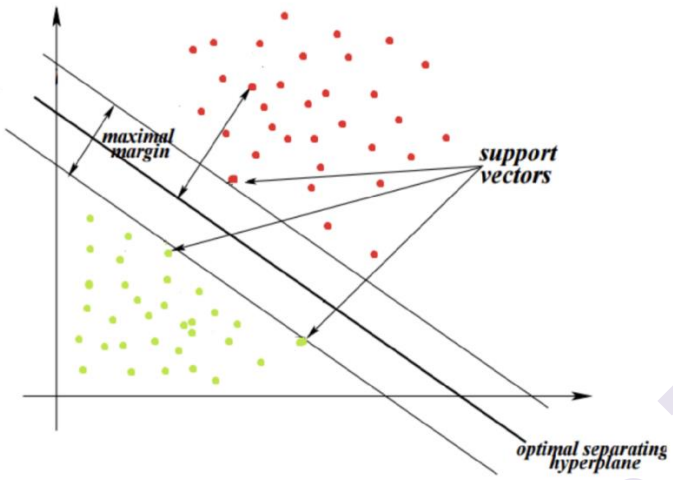


Figure 3.1 Support Vector Machine

3.2 EVALUATION:

There are different methods or techniques to find the performance of a classification problem. examples confusion matrix, costs, lift curves and ROC curves.

3.2.1 Confusion Matrix:

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an error matrix. Some features of Confusion matrix are given below:

For the 2 prediction classes of classifiers, the matrix is of 2*2 table, for 3 classes, it is 3*3 table, and so on.

The matrix is divided into two dimensions, that are predicted values and actual values along with the total number of predictions.

Predicted values are those values, which are predicted by the model, and actual values are the true values for the given observations.

It looks like the below table:

n = total predictions	Actual: No	Actual: Yes
Predicted: No	True Negative	False Positive
Predicted: Yes	False Negative	True Positive

Table 3.1 Prediction

Table shows the following cases:

1. True Negative: Model has given prediction No, and the real or actual value was also No.
2. True Positive: The model has predicted yes, and the actual value was also true.
3. False Negative: The model has predicted no, but the actual value was Yes, it is also called as Type-II error.
4. False Positive: The model has predicted Yes, but the actual value was No. It is also called a Type-I error.

Confusion Matrix Need in Machine learning

1. It evaluates the performance of the classification models, when they make predictions on test data, and tells how good our classification model is.
2. It not only tells the error made by the classifiers but also the type of errors such as it is either type-I or type-II error.
3. With the help of the confusion matrix, we can calculate the different parameters for the model, such as accuracy, precision, etc.

Example:

Suppose we are trying to create a model that can predict the result for the disease that is either a person has that disease or not. So, the confusion matrix for this is given as:

n = 100	Actual: No	Actual: Yes	
Predicted: No	TN: 65	FP: 3	68
Predicted: Yes	FN: 8	TP: 24	32
	73	27	

Table 3.2 Disease Prediction

From the above example, we can conclude that:

- The table is given for the two-class classifier, which has two predictions "Yes" and "NO." Here, Yes defines that patient has the disease, and No defines that patient does not has that disease.
- The classifier has made a total of 100 predictions. Out of 100 predictions, 89 are true predictions, and 11 are incorrect predictions.
- The model has given prediction "yes" for 32 times, and "No" for 68 times. Whereas the actual "Yes" was 27, and actual "No" was 73 times.

Calculations using Confusion Matrix:

We can perform various calculations for the model, such as the model's accuracy, using this matrix. These calculations are given below:

1. **Classification Accuracy:**

It is one of the important parameters to determine the accuracy of the classification problems. It defines how often the model predicts the correct output. It can be calculated as the ratio of the number of correct predictions made by the classifier to all number of predictions made by the classifiers. The formula is given below:

$$\textbf{Accuracy} = \frac{\textbf{TP+TN}}{\textbf{TP+FP+FN+TN}}$$

2. **Misclassification rate:**

It is also termed as Error rate, and it defines how often the model gives the wrong predictions. The value of error rate can be calculated as the number of incorrect predictions to all number of the predictions made by the classifier. The formula is given below:

$$\textbf{Error rate} = \frac{\textbf{FP} + \textbf{FN}}{\textbf{TP} + \textbf{FP} + \textbf{FN} + \textbf{TN}}$$

3. **Precision:** It can be defined as the number of correct outputs provided by the model or out of all positive classes that have predicted correctly by the model, how many of them were actually true. It can be calculated using the below formula:

$$\textbf{Precision} = \frac{\textbf{TP}}{\textbf{TP} + \textbf{FP}}$$

4. **Recall:** It is defined as the out of total positive classes, how our model predicted correctly. The recall must be as high as possible.

$$\textbf{Recall} = \frac{\textbf{TP}}{\textbf{TP+FN}}$$

F-measure:

If two models have low precision and high recall or vice versa, it is difficult to compare these models. So, for this purpose, we can use F-score. This score helps us to evaluate the recall and precision at the same time. The F-score is maximum if the recall is equal to the precision. It can be calculated using the below formula:

$$\mathbf{F-measure} = \frac{\mathbf{2*Recall*Precision}}{\mathbf{Recall+Precision}}$$

Important terms used in Confusion Matrix:

Null Error rate:

It defines how often our model would be incorrect if it always predicted the majority class. As per the accuracy paradox, it is said that "the best classifier has a higher error rate than the null error rate."

ROC Curve:

The ROC is a graph displaying a classifier's performance for all possible thresholds. The graph is plotted between the true positive rate (on the Y-axis) and the false Positive rate (on the x-axis).

3.2.2 Costs

Cost function:

A cost function is an important term that evaluates how well a machine learning model performs for a given dataset.

It calculates the difference between the expected value and predicted value and represents it as a single real number.

Cost function in Machine Learning

Machine Learning model should give a very high level of accuracy in order to perform well in a real world applications, but the most important term is how to calculate the accuracy of the model, where out model is good or bad while performing in the real world.

In that case, the use of cost function is important for machine learning parameter to correctly estimate the model whether it is good or bad.

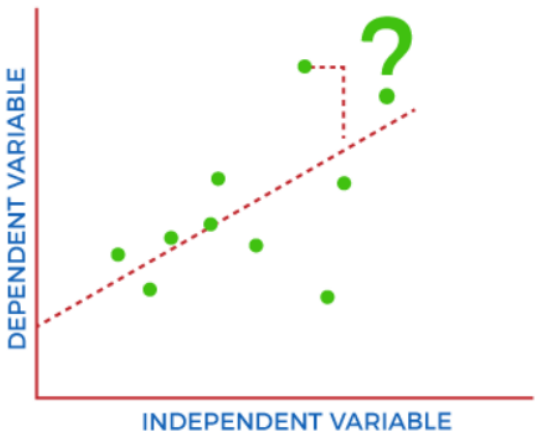


Figure 3.2: Cost function in machine learning

Cost function plays a very important role in machine learning that how well your model it shows. It estimates the relation-ship between input variables and the output variables. Variables are also known as parameters.

Cost function is a measure of how wrong the model is in estimating the relationship between X (input) and Y (output) parameter. A cost function is also known as loss function, and it can be estimated by iteratively running the model to compare estimated predictions against the known values of Y.

The aim of each machine learning model is to determine parameters or weights that can minimize the cost function.

Let us take an example and understand the use of cost function.

There are different accuracy parameters, then what is exact use of cost function for the machine learning model, we can understand it with an example of classification of data. Suppose we have a dataset which contains the weight of cats and dogs as well as height of cats and dogs. We have to classify them accordingly. The plotting of the records by using the scatter plot of these two features the figure shows:

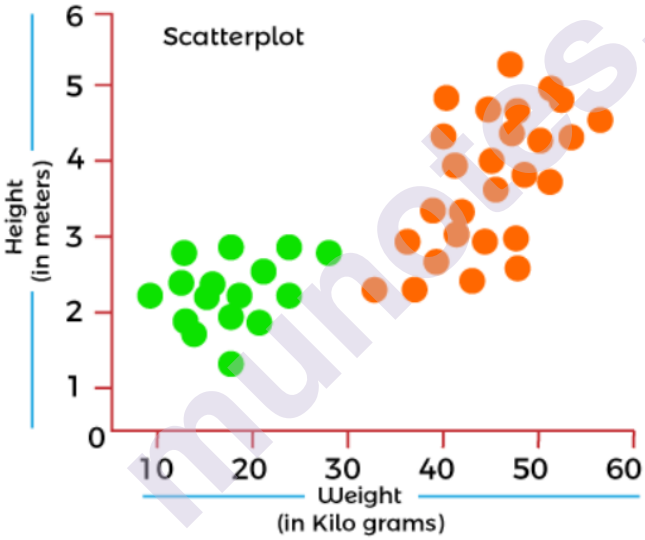


Figure 3.3: Green color dots are cats, and yellow color dots are dog.

There are possible solutions for the above classification problem.

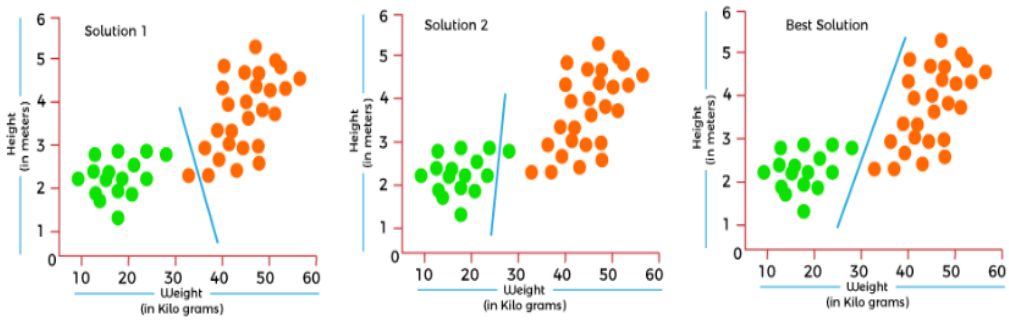


Figure 3.4: Possible solution for above classification problem

All three classifiers have high accuracy, but the third figure of solution is the best because it correctly classifiers each datapoints. For best classification is that it is in mid between both the classes, not close or not far to any of them.

By using cost function, we can get the optimal solution, It calculates the difference between the actual values and predicted values and measured how wrong was our model in the prediction, by minimizing the value of the cost function we always get the optimal solution.

For minimizing the cost function, we can use Gradient Descent.

3.2.3 Lift Curve:

For binary classification problem for identifying, we generally use the probability model and then we convert into the predictions for example if a certain patient has some disease using his or her health record the use of machine learning algorithms that we use generally and returns into a probability i.e. probability of the patient having the disease, where we can then converted into a prediction, whether or not the patient has such disease.

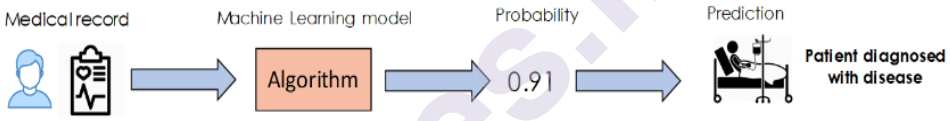


Figure 3.5: Pipeline of the previous explained scenario

The use of lift curve here is it returned probability to asses how our model is performing, and how well it is identifying the positive i.e. sick patients or negative i.e. healthy patients in our dataset. The main goal of our model is to classify patients accordingly.

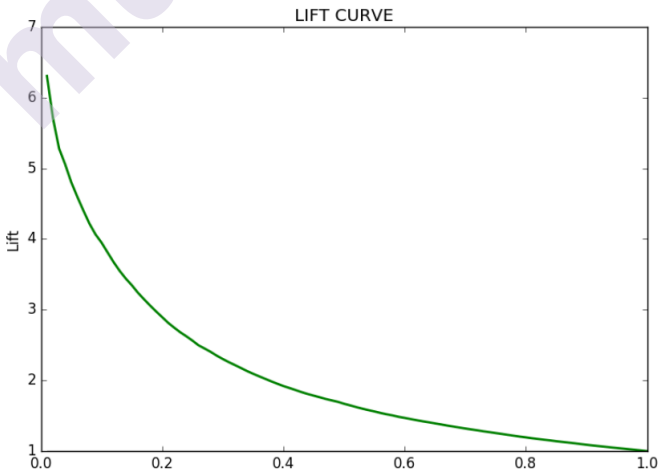


Figure 3.6: Lift Curve

X- axis is the proportion of our sample that corresponds to a certain lift, plotted on Y-axis. The Lift is easily calculated as the ratio of certain sample point, divided by the ratio of that whole dataset, which can also be viewed as the predictions that a random algorithm can make.

3.2.4 ROC Curves

ROC is Receiver Operator characteristic; it is useful for estimation of class probability.

Class Probability Estimation: It is a scoring classifier that gives the output as probability vectors. Consider we have two classes, the association of probability with one class is 1 minus the probability of the other class. It is also known as TP (True Positives).

To understand the probability estimation of y with the help of ROC curves. ROC curve denotes the performance of a binary classification model at various classification thresholds.

Roc curve is helpful in visualize the performance of a binary classifier. The AUC (Area under the curve) of the ROC curve indicates the ability of the binary classifier to show the difference between both the classes. With the help of all possible threshold probabilities, unlike other metrics that use a fixed threshold.

ROC plots 2 parameters namely :

1. TPR (True Positive Ratio)

$$TPR = \frac{TP}{TP + FN}$$

2. FPR (False Positive Ratio)

$$FPR = \frac{FP}{FP + TN}$$

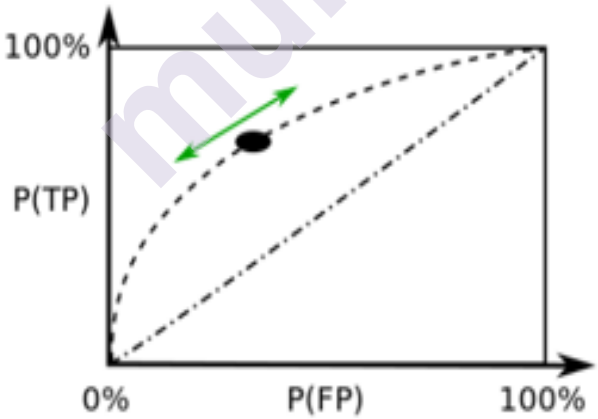


Figure 3.7: ROC Curve

It is a graph which shows the False Positives on the X-axis and True Positives on the Y-axisin the ROC curve.

AUC (Area Under the Curve) : It measures the 2dimentional area covering X-axis and Y-axis under the ROC curve from (0,0) to (1,1). It helps in

calculating the aggregate measure of performance across all classification thresholds. The range of AUC is from 0 to 1.

The prediction of a particular model is 100% wrong then the AUC value is 0.0 whereas the prediction of a model is 100% correct the AUC value of that model is 1.0.

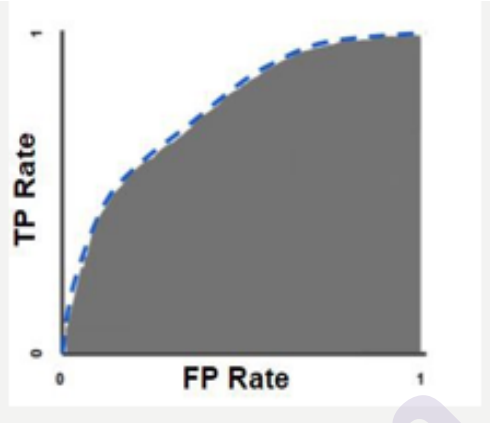


Figure 3.8: AUC (Area Under the Curve)

If the AUC value of a model is around 0.5 it means a very poor classifier whereas for very good classifier has the AUC value is around 1.

The figure shows the distribution of negatives (left side gaussian) and positives (right side gaussian) and the corresponding ROC curve below. The AUC value ranges from 0 and 1.

Evaluation of the classifier based on the AUC score as follows:

.90 – 1 = Excellent Classifier

.80 to .90 = Good Classifier

.70 to .80 = Fair Classifier

.60 to .70 = Poor Classifier

.50 to .60 = Fail Classifier

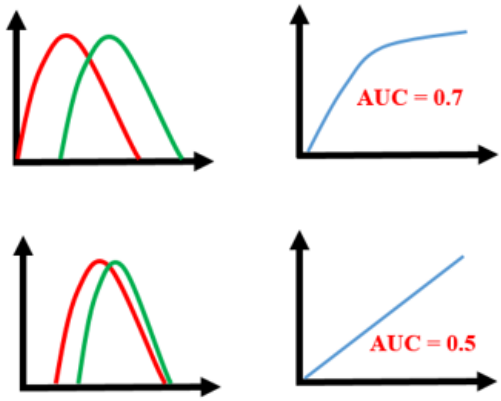


Figure 3.9: AUC Curves

Advantages of AUC Curve

It is useful in predicted probabilities which are not properly calibrated.

It is useful in metric even if the classes are imbalanced.

ROC and AUC

Figure 3.10 ROC and AUC [1]

This is the best possible ROC curve, it shows all positives above all negatives.

It has an AUV of 1.0

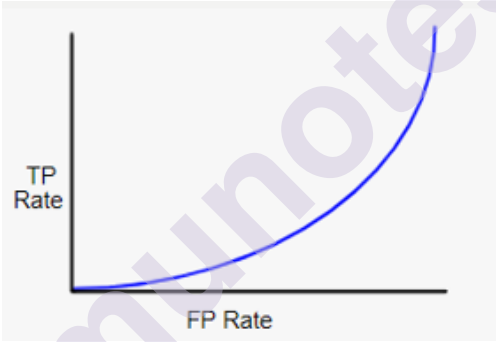


Figure 3.11 ROC and AUC [2]

This ROC curve has an AUC value between 0 and 0.5, The model performance is bad as the curve shows the there is a bug in data.

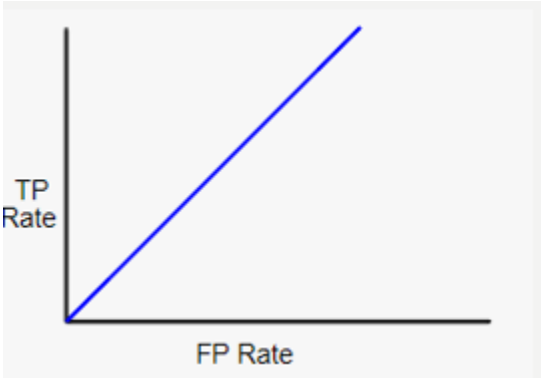


Figure 3.12 ROC and AUC [3]

This ROC curve has an AUC of 0.5 it shows 50% performance where it has the predictive ability is no better than random guessing.

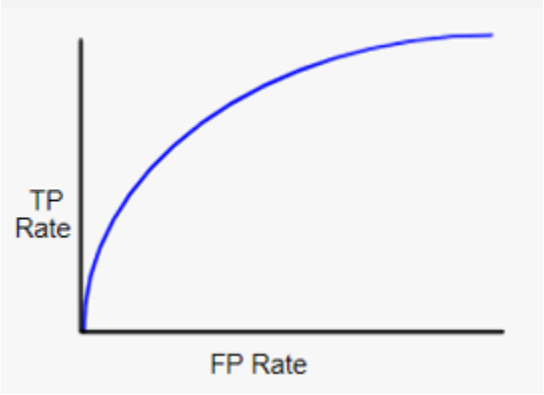


Figure 3.13 ROC and AUC [4]

This ROC Curve has an AUC between 0.5 and 1.0. Real world binary classification AUC values generally fall into this categorical range.

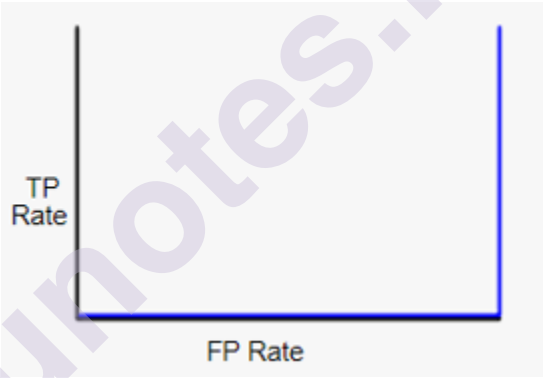


Figure 3.14 ROC and AUC [5]

This ROC Curve is the worst possible curve. It has an AUC of 0.0. It shows reverse every prediction (flip negatives to positives and positives to negatives).

REGRESSION/MODEL TREES:

Regression is statistical method of finding the relationship between the dependent and independent variables. How they are corelated to each other. Usually in graphical representation, dependent variable i.e. Y and a series of other variables (independent variables) i.e. X will determine the best fits of the given datapoints. With the help of regression we can make the predictions about the data.

3.3.1 Chi-Square Automatic Interaction Detection (CHAID)

CHAID technique was created by Gordon V. Kass in 1980. This algorithm is based on Chi-square statistic used for building decision tree to identify

optimal splits. A chi-square list yields a probability value as a result lying anywhere between 0 and 1. Chaid examines the cross tabulations between each input field and outcome and test for significance using chi-square independence test. If more than one of these relations is statistically significant Chaid will select the input field which is most significant (smallest p value). If input has more than 2 categories, these are compared and categories that show no difference in outcome are collapsed. For nominal input fields, any categories can be merged. For Ordinal set, only contiguous (next to each other) categories can be merged.

Advantages:

1. CHAID can generate non binary tree.

2. CHAID works for all types of input and accepts both case weight and frequency variables.

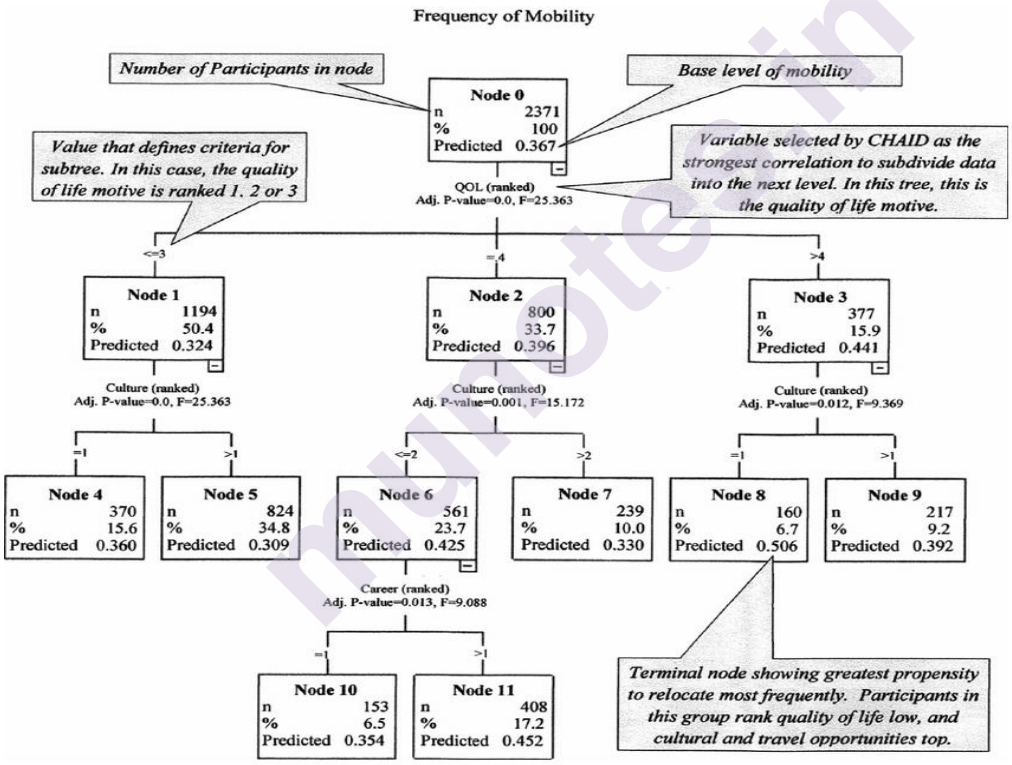


Figure 3.15 CHAID

3.3.2 Classification and regression tree (CART)

CART is a decision tree which can handle both classification and regression tasks. It was developed by Leo Breiman, Jerome Friedman, Richard Olshen and Charles Stone in 1984. It is a predictive algorithm which explains how the target variable's values can be predicted based on other matters. It is a decision tree where each task is split into predictor variable and each node has prediction for target variable at the end.

Working of CART:

1. Best split point of each input is obtained.
2. Based on the best split point of each input in step 1, the new "best" split point is identified.
3. Split chosen input according to new "best" split point.
4. Continue splitting until stopping criteria is met or further splitting is not possible.

Classification tree:

Classification tree is an algorithm where target variable is categorical. It is used to identify the class within which target variable is likely to fall. Classification trees are used for discrete values of dependent variables.

Regression tree:

Regression tree is an algorithm where target variable is continuous and it is used to predict its value. Regression trees are used for continuous values of dependent variables.

CART (Classification and Regression Tree) is a algorithm useful for predictions of data in machine learning.

SUMMARY

Support vector machine is popular supervised learning algorithm used for classification and regression problems. There are two types of support vector machine namely linear and non -linear. The use of cost function is to measure the performance of the model in machine learning. It calculates the difference between predicted values and the expected values. Classification tree is an algorithm where target variable is categorical. It is used to identify the class within which target variable is likely to fall. Classification trees are used for discrete values of dependent variables. Regression tree is an algorithm where target variable is continuous and it is used to predict its value. Regression trees are used for continuous values of dependent variables.CART (Classification and Regression Tree) is a algorithm useful for predictions of data in machine learning.

QUESTIONS

Q.1) What is support vector machine? Explain in detail different types of SVM.

Q.2) Explain Confusion matrix in detail.

Q.3) Write a note on

a) Precision b) Recall

Q.4) Discuss Cost function in detail.

Q.5) Elaborate ROC curve.

Q.6) Explain in detail CHAID.

Q.7) Explain in detail CART.

Q.8) Give the difference between Regression and Classification.

REFERENCES

1. Baldi, P., and S. Brunak. 1998. Bioinformatics: The Machine Learning Approach. Cambridge, MA: MIT Press.
2. Bilmes, J. A. 2006. "What HMMs Can Do." IEICE Transactions on Information and Systems E89-D:869–891.
3. Lecture Notes in Data Mining, Berry, Browne, World Scientific, 2009.

1. K-NN

- https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
- https://www.ibm.com/inen/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20algorithm%2C%20also%20known%20as%20KNN%20or,of%20an%20individual%20data%20point
- https://en.wikipedia.org/wiki/Euclidean_distance
- Image Reference: https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn

2. Support Vector Machine

- https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm
- https://www.geeksforgeeks.org/support-vector-machine-algorithm/
- https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
- Artificial Intelligence George F Luger 5th Edition
- Image Reference: https://www.marktechpost.com/2021/03/25/introduction-to-support-vector-machines-svms/?amp

ROC Curve

- https://howtolearnmachinelearning.com/articles/the-lift-curve-in-machine-learning/

3. CHAID

- https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/chaid/
- https://www.ibm.com/docs/en/cloud-paks/cp-data/4.6.x?topic=modeling-chaid-node
- https://towardsdatascience.com/clearly-explained-top-2-types-of-decision-trees-chaid-cart-8695e441e73e
- Image Reference: https://www.researchgate.net/figure/CHAID-analysis-of-predictors-for-frequency-of-mobility-Note-Cross-validation-risk_fig1_259435887

4. CART

- https://www.geeksforgeeks.org/cart-classification-and-regression-tree-in-machine-learning/
- https://www.analyticssteps.com/blogs/classification-and-regression-tree-cart-algorithm
- https://www.codingninjas.com/codestudio/library/classification-and-regression-tree-algorithm

5. Cost functions

https://www.javatpoint.com/cost-function-in-machine-learning

https://howtolearnmachinelearning.com/articles/the-lift-curve-in-machine-learning/

❄❄❄❄❄❄❄

4

DIMENSIONALITY REDUCTION

Unit Structure :

4.1 OBJECTIVE:

- We have examined matrices that represent social networks in previous chapters.
- In many of these matrix applications, the matrix can be summarized by finding "narrower" matrices that in some sense are close to the original.
- These narrow matrices have only a small number of rows or a small number of columns, and therefore can be used much more efficiently than can the original large matrix.
- In this chapter, we focus on techniques for symmetric matrices.

"The process of finding these narrow matrices is called dimensionality reduction."

- In this chapter we shall explore the idea of dimensionality reduction in more detail.
- We begin with a discussion of eigenvalues and their use in "principal component analysis" (PCA).
- We cover singular-value decomposition, a more powerful version of UV-decomposition.
- We will discuss CUR-decomposition, which is a variant of singular-value decomposition that keeps the matrices of the decomposition sparse if the original matrix is sparse.

4.2 INTRODUCTION TO EIGENVALUES AND EIGENVECTORS OF SYMMETRIC MATRICES:

4.2.1 Introduction:

- In this section, we shall define eigenvalues and eigenvectors of a symmetric matrix and show how to find them.
- A symmetric matrix is a matrix that is equal to its transpose.

- They contain three properties, including: **Real eigenvalues**, **eigenvectors** corresponding to the **eigenvalues** that are orthogonal and the matrix must be diagonalizable.
- A trivial example is the identity matrix.
- If a certain transformation or process, given by [A], acts on a particular non-zero vector X, such that [A][X] = λ[X], where λ is a constant, then **λ is called an eigenvalue and the corresponding vector X is called an eigenvector**.
- The transformed vector λ[X] has the same direction as X, although its length might differ.

4.2.2 Definitions:

Eigenvalue Definition:

- Eigenvalues are the special set of scalars associated with the system of linear equations.
- It is mostly used in matrix equations.
- 'Eigen' is a German word that means 'proper' or 'characteristic'.
- Therefore, the term eigenvalue can be termed as characteristic value, characteristic root, proper values or latent roots as well.
- **In simple words, the eigenvalue is a scalar that is used to transform the eigenvector.**

Let **A** be a square matrix.

Let **λ** be a constant and

x be a nonzero column vector with the same number of rows as A.

The basic equation is:

Ax = λx

The number or scalar value "λ" is an eigenvalue of A.

Then λ is an eigenvalue of A and

x is the corresponding eigenvector of A if Ax = λx.

- Then λ is an eigenvalue of M and e is the corresponding eigenvector of M if Me = λe.
- In Mathematics, an eigenvector corresponds to the real non zero eigenvalues which point in the direction stretched by the transformation whereas eigenvalue is considered as a factor by which it is stretched.

- In case, if the eigenvalue is negative, the direction of the transformation is negative.
- For every real matrix, there is an eigenvalue.
- Sometimes it might be complex.
- The existence of the eigenvalue for the complex matrices is equal to the fundamental theorem of algebra.

Eigen Vectors Definition:

- Eigenvectors are the vectors (non-zero) that do not change the direction when any linear transformation is applied.
- It changes by only a scalar factor.
- We can say, if A is a linear transformation from a vector space V and x is a vector in V, which is not a zero vector, then v is an eigenvector of A if A(X) is a scalar multiple of x.
- An Eigenspace of vector x consists of a set of all eigenvectors with the equivalent eigenvalue collectively with the zero vector.
- Though, the zero vector is not an eigenvector.
- Let us say A is an "n × n" matrix and λ is an eigenvalue of matrix A, then x, a non-zero vector, is called as eigenvector if it satisfies the given below expression;

  $Ax = \lambda x$
- x is an eigenvector of A corresponding to the eigenvalue, λ.
- **Note:**
  - There could be infinitely many Eigenvectors, corresponding to one eigenvalue.
  - For distinct eigenvalues, the eigenvectors are linearly dependent.
- **Properties of Eigenvalues**
  - Eigenvectors with Distinct Eigenvalues are Linearly Independent
  - Singular Matrices have Zero Eigenvalues
  - If A is a square matrix, then $\lambda = 0$ is not an eigenvalue of A
  - For a scalar multiple of a matrix: If A is a square matrix and λ is an eigenvalue of A. Then, aλ is an eigenvalue of aA.
  - **For Matrix powers:** If A is square matrix and λ is an eigenvalue of A and $n \geq 0$ is an integer, then λ^n is an eigenvalue of A^n.
  - **For polynomials of matrix:** If A is a square matrix, λ is an eigenvalue of A and p(x) is a polynomial in variable x, then p(λ) is the eigenvalue of matrix p(A).

- **Inverse Matrix:** If A is a square matrix, λ is an eigenvalue of A, then λ-1 is an eigenvalue of A-1
- **Transpose matrix:** If A is a square matrix, λ is an eigenvalue of A, then λ is an eigenvalue of At

- **Example:**

  Let M be the matrix :

  One of the eigenvectors of M is

$$\begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$$

  and its corresponding eigenvalue is 7.

  The equation

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix} = 7 \begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$$

  demonstrates the truth of this claim.

  Note that both sides are equal to

$$\begin{bmatrix} 7/\sqrt{5} \\ 14/\sqrt{5} \end{bmatrix}$$

  Also observe that the eigenvector is a unit vector,

  because $(1/\sqrt{5})^2+(2/\sqrt{5})^2 = 1/5+4/5 = 1$.

  and its corresponding eigenvalue is 7.

4.2.3 Computing Eigenvalues and Eigenvectors:

- We have seen how to find an eigenpair (an eigenvalue and its corresponding eigenvector) for a suitable matrix M in start with any unit vector v of the appropriate length and compute M^iv iteratively until it converges.
- When M is a stochastic matrix, the limiting vector is the principal eigenvector (the eigenvector with the largest eigenvalue), and its corresponding eigenvalue is 1.
- This method for finding the principal eigenvector, called **power iteration**.
- If the principal eigenvalue (eigenvalue associated with the principal eigenvector) is not 1, then as i grows, the ratio of $M^{i+1}v$ to M^iv approaches the principal eigenvalue while M^iv approaches a vector (probably not a unit vector) with the same direction as the principal eigenvector.

- **$O(n^3)$-running-time method**:
  - This method computes all the eigenpairs of a symmetric n × n matrix exactly.
  - The method starts by restating the equation that defines eigenpairs,

    $Me = \lambda e$ as $(M - \lambda I)e = 0$,

  Where,
  - I is the n × n identity matrix with 1's along the main diagonal and 0's elsewhere.
  - 0 is a vector of all 0's.
- A fact of linear algebra is that in order for $(M - \lambda I)e = 0$ to hold for a vector $e \neq 0$, the determinant of $M - \lambda I$ must be 0.
- Notice that $(M - \lambda I)$ looks almost like the matrix M, but if M has c in one of its diagonal elements,then $(M - \lambda I)$ has $c - \lambda$ there.
- While the determinant of an n × n matrix has n! terms, it can be computed in various ways in $O(n^3)$ time.
- An example is the method of "**pivotal condensation**."
- The determinant of $(M - \lambda I)$ is an nth-degree polynomial in λ, from which we can get the n values of λ that are the eigenvalues of M.
- For any such value,say c, we can then solve the equation Me = c e.
- There are n equations in n unknowns (the n components of e), but since there is no constant term in any equation, we can only solve for e to within a constant factor.
- However, using any solution, we can normalize it so the sum of the squares of the components is 1, thus obtaining the eigenvector that corresponds to eigenvalue c.

Note:

➔ A stochastic matrix is not generally symmetric.

➔ Symmetric matrices and stochastic matrices are two classes of matrices for which eigenpairs exist and can be exploited.

Example:

Let us find eigenpairs for the 2 x 2 matrix M where ,

M =

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$$

Then $M - \lambda I$ is

$$\begin{bmatrix} 3-\lambda & 2 \\ 2 & 6-\lambda \end{bmatrix}$$

The determinant of this matrix is $(3 - \lambda)(6 - \lambda) - 4$, which we must set to 0. The equation in λ to solve is thus $\lambda^2 - 9\lambda + 14 = 0$. The roots of this equation are $\lambda = 7$ and $\lambda = 2$; the first is the principal eigenvalue, since it is the larger. Let **e** be the vector of unknowns

$$\begin{bmatrix} x \\ y \end{bmatrix}$$

We must solve

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 7 \begin{bmatrix} x \\ y \end{bmatrix}$$

When we multiply the matrix and vector we get two equations

$$\begin{aligned} 3x+2y &= 7x \\ 2x+6y &= 7y \end{aligned}$$

Notice that both of these equations really say the same thing: $y = 2x$. Thus, a possible eigenvector is

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

But that vector is not a unit vector, since the sum of the squares of its components is 5, not 1. Thus to get the unit vector in the same direction, we divide each component by $\sqrt{5}$. That is, the principal eigenvector is

$$\begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$$

and its eigenvalue is 7. Note that this was the eigenpair we explored in Example 11.1.

For the second eigenpair, we repeat the above with eigenvalue 2 in place of 7. The equation involving the components of **e** is $x = -2y$, and the second eigenvector is

$$\begin{bmatrix} 2/\sqrt{5} \\ -1/\sqrt{5} \end{bmatrix}$$

Its corresponding eigenvalue is 2, of course. □

4.2.4 Finding Eigenpairs by Power Iteration

- In the previous section, we have seen that,

  column **vector v is an eigenvector of a square matrix A (of order n × n) if and only if Av = λv.**

- We can write this equation as Av = λIv,

  where I is the identity matrix of the same order n × n.

- The equation can be written as

  Av - λIv = O

  Here, O is a zero matrix.

  (A - λI) v = O ... (1)

- For this homogeneous system to have a non-trivial solution, the determinant should be equal to 0.

  |A - λI| = 0 ... (2)

  Steps to find the eigenvalues and eigenvectors as follows for any square matrix A:

- **To find the eigenvalues of A**,
  - solve the characteristic equation |A - λI| = 0 (equation (2)) for λ and all such values of λ would give the eigenvalues.
- **To find the eigenvectors of A**,
  - substitute each eigenvalue (i.e., the value of λ) in equation (1) (A - λI) v = O and
  - solve for v using the method of your choice.
- This would result in a system of homogeneous linear equations.

  Let us see how to find the eigenvectors of a 2 × 2 matrix and 3 × 3 matrix using these steps.

4.2.5 The Matrix of Eigenvectors

Suppose we have an n × n symmetric matrix M whose eigenvectors, viewed as column vectors, are e1, e2, . . . , en. Let E be the matrix whose ith column is e_i .

Then $EE^T = E^TE = I$.

The explanation is that the eigenvectors of a symmetric matrix are orthonormal.

That is, they are orthogonal unit vectors.

Example:

Take the matrix from,

$$M = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$$

and let us start with $\mathbf{x}_0$ a vector with 1 for both components. To compute $\mathbf{x}_1$, we multiply $M\mathbf{x}_0$ to get

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}$$

The Frobenius norm of the result is $\sqrt{5^2 + 8^2} = \sqrt{89} = 9.434$. We obtain $\mathbf{x}_1$ by dividing 5 and 8 by 9.434; that is:

$$\mathbf{x}_1 = \begin{bmatrix} 0.530 \\ 0.848 \end{bmatrix}$$

For the next iteration, we compute

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 0.530 \\ 0.848 \end{bmatrix} = \begin{bmatrix} 3.286 \\ 6.148 \end{bmatrix}$$

The Frobenius norm of the result is 6.971, so we divide to obtain

$$\mathbf{x}_2 = \begin{bmatrix} 0.471 \\ 0.882 \end{bmatrix}$$

We are converging toward a normal vector whose second component is twice the first. That is, the limiting value of the vector that we obtain by power iteration is the principal eigenvector:

$$\mathbf{x} = \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix}$$

Finally, we compute the principal eigenvalue by

$$\lambda = \mathbf{x}^{\mathrm{T}} M \mathbf{x} = \begin{bmatrix} 0.447 & 0.894 \end{bmatrix} \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix} = 6.993$$

Example :

We can compute:

$$M^* = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} - 6.993 \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix} \begin{bmatrix} 0.447 & 0.894 \end{bmatrix} =$$

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} - \begin{bmatrix} 1.397 & 2.795 \\ 2.795 & 5.589 \end{bmatrix} = \begin{bmatrix} 1.603 & -0.795 \\ -0.795 & 0.411 \end{bmatrix}$$

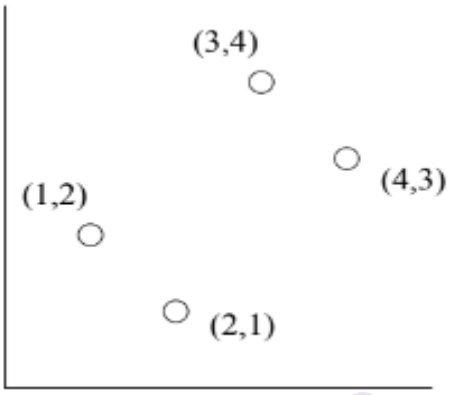
We may find the second eigenpair by processing the matrix above as we did the original matrix M. □

4.3 PRINCIPAL COMPONENT ANALYSIS:

- Principal-component analysis, or PCA, is a technique for taking a dataset consisting of a set of tuples representing points in a high-dimensional space and finding the directions along which the tuples line up best.
- The idea is to treat the set of tuples as a matrix M and find the eigenvectors for MMT or MTM.
- The matrix of these eigenvectors can be thought of as a rigid rotation in a high dimensional space.
- When you apply this transformation to the original data,
    - the axis corresponding to the principal eigenvector is the one along which the points are most "spread out,"
    - More precisely, this axis is the one along which the variance of the data is maximized.
- Put another way, the points can best be viewed as lying along this axis, with small deviations from this axis.
- Likewise, the axis corresponding to the second eigenvector (the eigenvector correspond-
- ing to the second-largest eigenvalue) is the axis along which the variance of distances from the first axis is greatest, and so on.
- PCA as a data-mining technique , high-dimensional data can be replaced by its projection onto the most important axes.
- These axes are the ones corresponding to the largest eigenvalues.
- Thus, the original data is approximated by data that has many fewer dimensions and that summarizes well the original data.
- **Principal-Component Analysis(PCA):**
    - This technique for dimensionality reduction views data consisting of a collection of points in a multidimensional space as a matrix, with rows corresponding to the points and columns to the dimensions.
    - The product of this matrix and its transpose has eigenpairs, and the principal eigenvector can be viewed as the direction in the space along which the points best line up.
    - The second eigenvector represents the direction in which deviations from the principal eigenvector are the greatest, and so on.
- **Dimensionality Reduction by PCA:**

○ By representing the matrix of points by a small number of its eigenvectors, we can approximate the data in a way that minimizes the root-mean-square error for the given number of columns in the representing matrix.

- **Example**
  - In this example, the data is two-dimensional, a number of dimensions that is too small to make PCA really useful.
  - Moreover, the data, shown in below Figure:



  - has only four points, and they are arranged in a simple pattern along the 45-degree line to make our calculations easy to follow.
  - That is, to anticipate the result, the points can best be viewed as lying along the axis that is at a 45-degree angle, with small deviations in the perpendicular direction.
  - To begin, let us represent the points by a matrix M with four rows – one for each point – and two columns, corresponding to the x-axis and y-axis.
  - This matrix is

$$M = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix}$$

  - Compute $M^T M$, which is

$$M^T M = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} = \begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix}$$

  - We may find the eigenvalues of the matrix above by solving the equation

$$(30 - \lambda)(30 - \lambda) - 28 \times 28 = 0$$

- **The solution is λ = 58 and λ = 2.**
- Now we must solve,

$$\begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 58 \begin{bmatrix} x \\ y \end{bmatrix}$$

- When we multiply out the matrix and vector we get two equations

$$\begin{aligned} 30x+28y &= 58x \\ 28x+30y &= 58y \end{aligned}$$

- Both equations tell us the same thing: x = y.
- Thus, the unit eigenvector corresponding to the principal eigenvalue 58 is

$$\begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

- For the second eigenvalue, 2, we perform the same process. Multiply out

$$\begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 2 \begin{bmatrix} x \\ y \end{bmatrix}$$

- to get the two equations

$$\begin{aligned} 30x+28y &= 2x \\ 28x+30y &= 2y \end{aligned}$$

- Both equations tell us the same thing: x = −y.
- Thus, the unit eigenvector corresponding to the principal eigenvalue 2 is

$$\begin{bmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

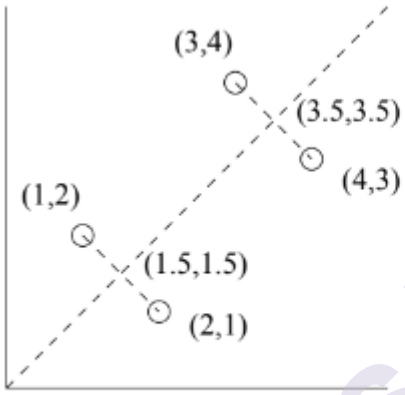
- **Now, let us construct E, the matrix of eigenvectors for the matrix M^TM.**
- Placing the principal eigenvector first, the matrix of eigenvectors is

$$E = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

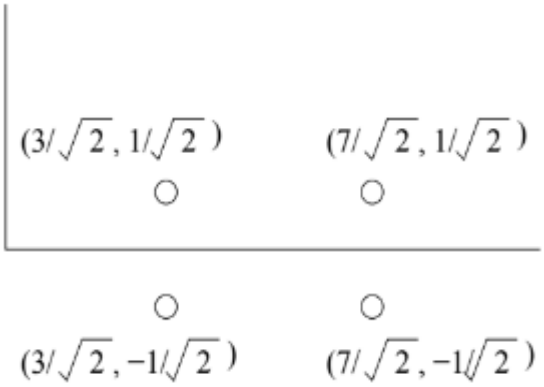
- The matrix above can be viewed as a rotation 45 degrees counterclockwise.
- Let us multiply the matrix M that represents each of the points of The product is,

$$ME = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 3/\sqrt{2} & 1/\sqrt{2} \\ 3/\sqrt{2} & -1/\sqrt{2} \\ 7/\sqrt{2} & 1/\sqrt{2} \\ 7/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

-



- l with the axes rotated 45 degrees counterclockwise
- We see the first point, [1, 2], has been transformed into the point
- $[3/\sqrt{2}, 1/\sqrt{2}]$
- To check this fact, notice that the point of projection for both the first and second points is [1.5, 1.5] in the original coordinate system,and the distance from the origin to this point is,
- $\sqrt{(1.5)^2 + (1.5)^2} = \sqrt{9/2} = 3/\sqrt{2}$
- The new y-axis is, of course, perpendicular to the dashed line.
- The first point is at distance 1/√2 above the new x-axis in the direction of the y-axis.
- That is, the distance between the points [1, 2] and [1.5, 1.5] is
- $\sqrt{(1-1.5)^2 + (2-1.5)^2} = \sqrt{(-1/2)^2 + (1/2)^2} = \sqrt{1/2} = 1/\sqrt{2}$
- Following Figure shows the four points in the rotated coordinate system,



- Both project onto the same point of the new x-axis, and that point is at distance $7/\sqrt{2}$ from the origin, while they are $1/\sqrt{2}$ above and below the new x-axis in the direction of the new y-axis.

Using Eigenvectors for Dimensionality Reduction

- If M is a matrix whose rows each represent a point in a Euclidean space with any number of dimensions, we can compute MTM and compute its eigenpairs.
- Let E be the matrix whose columns are the eigenvectors, ordered as largest eigenvalue first.
- Define the matrix L to have the eigenvalues of M^TM along the diagonal, largest first, and 0's in all other entries.
- Then, since $M^TMe = \lambda e = e\lambda$ for each eigenvector e and its corresponding eigenvalue λ, it follows that $M^TME = EL$.
- We observed that ME is the point of M transformed into a new coordinate space.
- In this space, the first axis (the one corresponding to the largest eigenvalue) is the most significant; formally, the variance of points along that axis is the greatest.
- The second axis, corresponding to the second eigenpair, is next most significant in the same sense, and the pattern continues for each of the eigenpairs.
- If we want to transform M to a space with fewer dimensions, then the choice that preserves the most significance is the one that uses the eigenvectors associated with the largest eigenvalues and ignores the other eigenvalues.
- That is, let Ek be the first k columns of E. Then MEk is a k-dimensional representation of M.

The Matrix of Distances

- In below example, we have,

$$MM^{\mathrm{T}} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{bmatrix} = \begin{bmatrix} 5 & 4 & 11 & 10 \\ 4 & 5 & 10 & 11 \\ 11 & 10 & 25 & 24 \\ 10 & 11 & 24 & 25 \end{bmatrix}$$

- Like M^TM, we see that MM^T is symmetric.
- The entry in the ith row and j^{th} column has a simple interpretation;
- It is the dot product of the vectors represented by the i^{th} and j^{th} points (rows of M).
- There is a strong relationship between the eigenvalues of M^TM and MM^T.
- Suppose e is an eigenvector of M^TM; that is,

$$M^{\mathrm{T}}M\mathbf{e} = \lambda\mathbf{e}$$

- Multiply both sides of this equation by M on the left. Then

$$MM^{\mathrm{T}}(M\mathbf{e}) = M\lambda\mathbf{e} = \lambda(M\mathbf{e})$$

- Thus, as long as Me is not the zero vector 0, it will be an eigenvector of MM^T and λ will be an eigenvalue of MM^T as well as of M^TM.
- The converse holds as well.
- That is, if e is an eigenvector of MM^T with corresponding eigenvalue λ, then start with $MM^Te = \lambda e$ and
- multiply on the left by M^T to conclude that $M^TM(M^Te) = \lambda(M^Te)$.
- Thus, if M^Te is not 0,then λ is also an eigenvalue of M^TM.
- We might wonder what happens when $M^Te = 0$.
- In that case, MM^Te is also 0, but e is not 0 because 0 cannot be an eigenvector.
- However, since $0 = \lambda e$, we conclude that $\lambda = 0$.
- We conclude that the eigenvalues of MM^T are the eigenvalues of M^TM plus additional 0's.
- If the dimension of MM^T were less than the dimension of M^TM, then the opposite would be true; the eigenvalues of M^TM would be those of MM^T plus additional 0's.

- **Example**
  - The eigenvalues of MM^T for our running example must include 58 and 2, because those are the eigenvalues of M^TM.
  - Since MM^T is a 4×4 matrix, it has two other eigenvalues, which must both be 0.
  - The matrix of eigenvectors corresponding to 58, 2, 0,and 0 is shown below Fig.

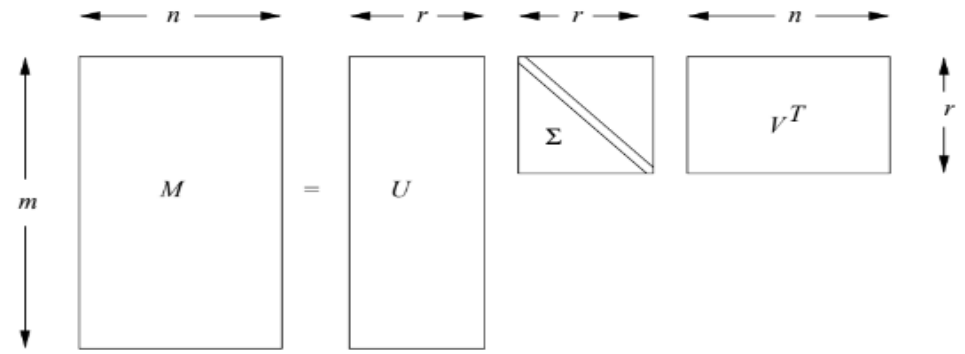
$$\begin{bmatrix} 3/\sqrt{116} & 1/2 & 7/\sqrt{116} & 1/2 \\ 3/\sqrt{116} & -1/2 & 7/\sqrt{116} & -1/2 \\ 7/\sqrt{116} & 1/2 & -3/\sqrt{116} & -1/2 \\ 7/\sqrt{116} & -1/2 & -3/\sqrt{116} & 1/2 \end{bmatrix}$$

4.4 SINGULAR-VALUE DECOMPOSITION:

- A second form of matrix analysis that leads to a low-dimensional representation of a high-dimensional matrix is this approach, called singular-value decomposition (SVD).
- SVD allows an exact representation of any matrix,
- It makes it easy to eliminate the less important parts of that representation to produce an approximate representation with any desired number of dimensions.

Definition of SVD

- Let M be an m × n matrix, and let the rank of M be r.
- The rank of a matrix is the largest number of rows (or equivalently columns) we can choose say a set of such rows or columns is independent).
- Then we can find matrices U, Σ, and V as shown in below figure with the following properties:



  - U is an m × r column-orthonormal matrix ;

- that is, each of its columns is a unit vector and the dot product of any two columns is 0.

- V is an n × r column-orthonormal matrix.
  - Note that we always use V in its transposed form, so it is the rows of V T that are orthonormal.
- Σ is a diagonal matrix; that is, all elements not on the main diagonal are 0.
  - The elements of Σ are called the singular values of M.

Example:

- Figure Gives a rank-2 matrix representing ratings of movies by users.

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

- In this contrived example there are two "concepts" underlying the movies: science-fiction and romance.
- All the boys rate only science-fiction,
- and all the girls rate only romance.
- It is this existence of two strictly adhered to concepts that gives the matrix a rank of 2.
- That is, we may pick one of the first four rows and one of the last three rows and observe that there is no nonzero linear sum of these rows that is 0.
- But we cannot pick three independent rows.
- For example, if we pick rows 1, 2, and 7, then three times the first minus the second, plus zero times the seventh is 0.
- We can make a similar observation about the columns.
- We may pick one of the first three columns and one of the last two columns, and they will be independent, but no set of three columns is independent.

- The decomposition of the matrix M from Figure into U, Σ, and V , with all elements correct to two significant digits, is shown in the following Figure.

$$\underset{M}{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix}} = \underset{U}{\begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix}} \underset{\Sigma}{\begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}} \underset{V^{\mathrm{T}}}{\begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}}$$

- Since the rank of M is 2, we can use r = 2 in the decomposition.

Interpretation of SVD

- The key to understanding what SVD offers is in viewing the r columns of U, Σ, and V as representing concepts that are hidden in the original matrix M.
- In Example

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

- one is “science fiction” and the other is “romance.”
- Let us think of the rows of M as people and the columns of M as movies.
- Then matrix U connects people to concepts.
- For example, the person Joe, who corresponds to row 1 of M in given Figure, likes only the concept science fiction.

$$\underset{U}{\begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix}}$$

- The value 0.14 in the first row and first column of U is smaller than some of the other entries in that column, because while Joe watches only science fiction, he doesn't rate those movies highly.
- The second column of the first row of U is 0, because Joe doesn't rate romance movies at all.
- The matrix V relates movies to concepts.

$$\begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}$$

$$V^{\mathrm{T}}$$

- The 0.58 in each of the first three columns of the first row of V
- T indicates that the first three movies – The Matrix,
- Alien, and Star Wars – each are of the science-fiction genre, while the 0's in the last two columns of the first row say that these movies do not partake of the concept romance at all. Likewise, the second row of V^T tells us that the movies Casablanca and Titanic are exclusively romances.
- Finally, the matrix Σ gives the strength of each of the concepts.

$$\begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}$$

$$\Sigma$$

- In our example, the strength of the science-fiction concept is 12.4, while the strength of the romance concept is 9.5.
- Intuitively, the science-fiction concept is stronger because the data provides more information about the movies of that genre and the people who like them.

Dimensionality Reduction Using SVD

- Suppose we want to represent a very large matrix M by its SVD components U, Σ, and V , but these matrices are also too large to store conveniently.
- The best way to reduce the dimensionality of the three matrices is to set the smallest of the singular values to zero.

- If we set the s smallest singular values to 0, then we can also eliminate the corresponding s columns of U and V .

Example :

The decomposition of the following figure has three singular values.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} =$$

M'

$$\begin{bmatrix} .13 & .02 & -.01 \\ .41 & .07 & -.03 \\ .55 & .09 & -.04 \\ .68 & .11 & -.05 \\ .15 & -.59 & .65 \\ .07 & -.73 & -.67 \\ .07 & -.29 & .32 \end{bmatrix} \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \\ .40 & -.80 & .40 & .09 & .09 \end{bmatrix}$$

U Σ V^{T}

- Suppose we want to reduce the number of dimensions to two.
- Then we set the smallest of the singular values, which is 1.3, to zero.
- The effect on the expression in given Figure is that the third column of U and the third row of V^T are:multiplied only by 0's when we perform the multiplication, so this row and this
- column may as well not be there.
- That is, the approximation to M′ obtained by using only the two largest singular values is that shown in below figure:

$$\begin{bmatrix} .13 & .02 \\ .41 & .07 \\ .55 & .09 \\ .68 & .11 \\ .15 & -.59 \\ .07 & -.73 \\ .07 & -.29 \end{bmatrix} \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \end{bmatrix}$$

$$= \begin{bmatrix} 0.93 & 0.95 & 0.93 & .014 & .014 \\ 2.93 & 2.99 & 2.93 & .000 & .000 \\ 3.92 & 4.01 & 3.92 & .026 & .026 \\ 4.84 & 4.96 & 4.84 & .040 & .040 \\ 0.37 & 1.21 & 0.37 & 4.04 & 4.04 \\ 0.35 & 0.65 & 0.35 & 4.87 & 4.87 \\ 0.16 & 0.57 & 0.16 & 1.98 & 1.98 \end{bmatrix}$$

- The entire difference is the result of making the last singular value be 0.
- However, in this simple example, much of the difference is due to rounding error caused by the fact that the decomposition of M′ was only correct to two significant digits.

Computing the SVD of a Matrix:

- The SVD of a matrix M is strongly connected to the eigenvalues of the symmetric matrices M^TM and MM^T.
- This relationship allows us to obtain the SVD of M from the eigenpairs of the latter two matrices.
- To begin the explanation, start with $M = U\Sigma V^T$, the expression for the SVD of M.
- Then,

$$M^T = (U\Sigma V^T)^T$$

$$= (V^T)^T \Sigma^T U^T$$

$$= V\Sigma^T U^T$$

- Since Σ is a diagonal matrix, transposing it has no effect.
- Thus, $M^T = V \Sigma U^T$.
- Now, $M^TM = V \Sigma U^T U \Sigma V^T$.
- Remember that U is an orthonormal matrix, so U^TU is the identity matrix of the appropriate size.
- That is,

$$M^T M = V\Sigma^2 V^T$$

- Multiply both sides of this equation on the right by V to get

$$M^T MV = V\Sigma^2 V^T V$$

- Since V is also an orthonormal matrix, we know that V^TV is the identity. Thus

$$M^T MV = V\Sigma^2$$

- Since Σ is a diagonal matrix, Σ^2 is also a diagonal matrix whose entry in the ith row and column is the square of the entry in the same position of Σ.
- Now, Equation $M^{\mathrm{T}}MV = V\Sigma^2$ should be familiar.
- It says that V is the matrix of eigenvectors of M^TM and Σ^2 is the diagonal matrix whose entries are the corresponding eigenvalues.
- Thus, the same algorithm that computes the eigenpairs for M^TM gives us the matrix V for the SVD of M itself.
- It also gives us the singular values for this SVD; just take the square roots of the eigenvalues for MTM.
- Only U remains to be computed, but it can be found in the same way we found V .
- Start with,

$$MM^{\mathrm{T}} = U\Sigma V^{\mathrm{T}}(U\Sigma V^{\mathrm{T}})^{\mathrm{T}} = U\Sigma V^{\mathrm{T}}V\Sigma U^{\mathrm{T}} = U\Sigma^2 U^{\mathrm{T}}$$

- Then by a series of manipulations analogous to the above, we learn that,

$$MM^{\mathrm{T}}U = U\Sigma^2$$

- That is, U is the matrix of eigenvectors of MM^T.

4.5 CUR DECOMPOSITION:

- In this section, we will see approach to decomposition, called CUR-decomposition.
- The merit of this approach lies in the fact that if M is sparse, then the two large matrices (called C and R for "columns" and "rows") analogous to U and V in SVD are also sparse.
- Only the matrix in the middle (analogous to Σ in SVD) is dense, but this matrix is small so the density does not hurt too much.
- Unlike SVD, which gives an exact decomposition as long as the parameter r is taken to be at least as great as the rank of the matrix M, CUR-decomposition is an approximation no matter how large we make r.
- There is a theory that guarantees convergence to M as r gets larger, but typically you have to make r so large to get, say within 1% that the method becomes impractical.
- A decomposition with a relatively small value of r has a good probability of being a useful and accurate decomposition.

Definition of CUR

- Let M be a matrix of m rows and n columns.
- Pick a target number of "concepts" r to be used in the decomposition.
- A CUR-decomposition of M is a randomly chosen set of r columns of M, which form the m × r matrix C, and a randomly chosen set of r rows of M, which form the r × n matrix R.
- There is also an r × r matrix U that is constructed from C and R as follows:
    - Let W be the r × r matrix that is the intersection of the chosen columns of C and the chosen rows of R.
        - That is, the element in row i and column j of W is the element of M whose column is the jth column of C and whose row is the ith row of R.
    - Compute the SVD of W;
        - say $W = X\Sigma Y^T$.
    - Compute Σ+, the Moore-Penrose pseudoinverse of the diagonal matrix Σ.
        - That is, if the ith diagonal element of Σ is σ 6= 0, then replace it by 1/σ.
        - But if the ith element is 0, leave it as 0.
    - Let $U = Y (\Sigma^+)^2 X^T$.

Choosing Rows and Columns Properly

- The measure of importance we must use is the square of the Frobenius norm, that is, the sum of the squares of the elements of the row or column.

$$f = \sum_{i,j} m_{i,j}^2$$

- Let the square of the Frobenius norm of M.
- Then each time we select a row, the probability pi with which we select row i is $\sum_j m_{ij}^2 / f$.
- Each time we select a column, the probability qj with which we select column j is $\sum_i m_{ij}^2 / f$.

Example:

- Let us reconsider the matrix M as shown below.

	Python	C++	Java	C	HTML
Sachin	1	1	1	0	0
Manish	3	3	3	0	0
Priya	4	4	4	0	0
Anita	5	5	5	0	0
Mahesh	0	0	0	4	4
Prakash	0	0	0	5	5
Sandesh	0	0	0	2	2

- The sum of the squares of the elements of M is 243.
- The three columns for the programming languages Python, C++ , and Java.
- Each have a squared Frobenius norm of $1^2 + 3^2 + 4^2 + 5^2 = 51$,
- so their probabilities are each 51/243 = .210.
- The remaining two columns each have a squared Frobenius norm of $4^2 + 5^2 + 2. = .45$, and therefore their probabilities are

  each 45/243 = .185.
- The seven rows of M have Frobenius norms of 3, 27, 48, 75, 32,50, and 8, respectively.
- Thus, their respective probabilities are .012, .111, .198, .309, .132, .206, and .033.

Constructing the Middle Matrix

- Finally, we must construct the matrix U that connects C and R in the decomposition.
- Recall that U is an r × r matrix.
- We start the construction of U with another matrix of the same size, which we call W.
- The entry in row i and column j of W is the entry of M whose row is the one from which we selected the ith row of R and whose column is the one from which we selected the j^{th} column of C.

Example :

Let us construct U from the matrix W that we constructed in

First, here is the SVD for W:

$$W = \begin{bmatrix} 0 & 5 \\ 5 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

That is, the three matrices on the right are X, Σ, and Y^T, respectively.

The matrix Σ has no zeros along the diagonal, so each element is replaced by its numerical inverse to get it's Moore-Penrose pseudoinverse:

$$\Sigma^+ = \begin{bmatrix} 1/5 & 0 \\ 0 & 1/5 \end{bmatrix}$$

Now X and Y are symmetric, so they are their own transposes.

Thus,

$$U = Y(\Sigma^+)^2 X^T$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1/5 & 0 \\ 0 & 1/5 \end{bmatrix}^2 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1/25 \\ 1/25 & 0 \end{bmatrix}$$

The Complete CUR Decomposition

- We now have a method to select randomly the three component matrices C, U, and R.
- Their product will approximate the original matrix M.
- As we mentioned at the beginning of the discussion, the approximation is only formally guaranteed to be close when very large numbers of rows and columns are selected.
- However, the intuition is that by selecting rows and columns that tend to have high “importance” (i.e., high Frobenius norm), we are extracting the most significant parts of the original matrix, even with a small number of rows and columns.
- As an example, let us see how well we do with the running example of this section.

Example:

- For our running example, the decomposition is shown in below figure :

$$CUR = \begin{bmatrix} 1.54 & 0 \\ 4.63 & 0 \\ 6.17 & 0 \\ 7.72 & 0 \\ 0 & 9.30 \\ 0 & 11.63 \\ 0 & 4.65 \end{bmatrix} \begin{bmatrix} 0 & 1/25 \\ 1/25 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 11.01 & 11.01 \\ 8.99 & 8.99 & 8.99 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0.55 & 0.55 & 0.55 & 0 & 0 \\ 1.67 & 1.67 & 1.67 & 0 & 0 \\ 2.22 & 2.22 & 2.22 & 0 & 0 \\ 2.78 & 2.78 & 2.78 & 0 & 0 \\ 0 & 0 & 0 & 4.10 & 4.10 \\ 0 & 0 & 0 & 5.12 & 5.12 \\ 0 & 0 & 0 & 2.05 & 2.05 \end{bmatrix}$$

=

- While there is considerable difference between this result and the original matrix M, especially in the science-fiction numbers, the values are in proportion to their originals.
- This example is much too small, and the selection of the small numbers of rows and columns was arbitrary rather than random, for us to expect close convergence of the CUR decomposition to the exact values.

4.6 EXERCISE

Answer the following:

1. Find the CUR-decomposition of the matrix of

	Python	C++	Java	C	HTML
Sachin	1	1	1	0	0
Manish	3	3	3	0	0
Priya	4	4	4	0	0
Anita	5	5	5	0	0
Mahesh	0	0	0	4	4
Prakash	0	0	0	5	5
Sandesh	0	0	0	2	2

when we pick two “random” rows and columns as follows:

(a) The columns for The Matrix and Alien and the rows for Jim and John.

(b) The columns for Alien and Star Wars and the rows for Jack and Jill.

(c) The columns for The Matrix and Titanic and the rows for Joe and Jane.

2. Find the CUR-decomposition of the matrix of

	Python	C++	Java	C	HTML
Sachin	1	1	1	0	0
Manish	3	3	3	0	0
Priya	4	4	4	0	0
Anita	5	5	5	0	0
Mahesh	0	0	0	4	4
Prakash	0	0	0	5	5
Sandesh	0	0	0	2	2

if the two "random" rows are both Jack and the two columns are Python and C++.

3. The SVD for the matrix:

$$M = \begin{bmatrix} 48 & 14 \\ 14 & -48 \end{bmatrix}$$

is

$$\begin{bmatrix} 48 & 14 \\ 14 & -48 \end{bmatrix} = \begin{bmatrix} 3/5 & 4/5 \\ 4/5 & -3/5 \end{bmatrix} \begin{bmatrix} 50 & 0 \\ 0 & 25 \end{bmatrix} \begin{bmatrix} 4/5 & -3/5 \\ 3/5 & 4/5 \end{bmatrix}$$

Find the Moore-Penrose pseudoinverse of M.

4. Use the SVD from Fig.

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix} \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}$$

M U Σ V^{T}

Suppose Leslie assigns rating 3 to Alien and rating 4 to Titanic, giving us a representation of Leslie in "movie space" of [0, 3, 0, 0, 4].

A. Find the representation of Leslie in concept space.

B. What does that representation predict about how well Leslie would like the other movies appearing in our example data?

5. Let M be the matrix of data points:

$$\begin{bmatrix} 1 & 1 \\ 2 & 4 \\ 3 & 9 \\ 4 & 16 \end{bmatrix}$$

(a) What are M^TM and MM^T?

(b) Compute the eigenpairs for M^TM.

6. Find the unit vector in the same direction as the vector [1, 2, 3].

REFERENCES:

Mining of Massive Datasets, Anand Rajaraman and Jeffrey David Ullman.pdf

❄❄❄❄❄❄❄

LINK ANALYSIS

Unit Structure :

5.0 OBJECTIVES

- To understand the concept of link analysis
- To get familiar with the computational fundamentals associated with page rank

5.1 INTRODUCTION

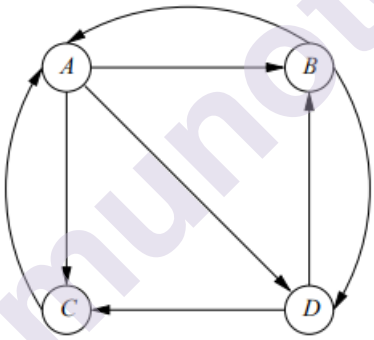
The advent of precise and efficient Web search through search engines like Google was one of the largest transformations in our lives in the decade that followed the turn of the century. Although not being the first search engine, Google was the first to successfully combat the spammers who had rendered search virtually ineffective. Also, Google's innovation included a complex technological advancement termed "PageRank."

But, the battle between people who want to utilize the Web for good and those who want to use it for personal gain never ends. Spammers created methods to alter a Web page's PageRank when it was identified as a crucial search engine ranking factor, a practice known as link spam. In reaction to such development, TrustRank and other methods were developed to stop spammers from targeting PageRank.

5.2 PAGERANK

Each page on the Web (or at least the section of the Web that has been crawled and its links found) is given a real number by the PageRank function. The idea is that a page is more "important" if its PageRank is higher. There isn't a single set algorithm for determining PageRank, and modifications on the fundamental concept can change the relative PageRank of any two pages. We start by outlining the fundamental, idealistic PageRank and then add modifications to it as needed to address some real-world issues with the Web's organizational structure.

If you imagine the Web as a directed graph with pages acting as the nodes, then a link from page p1 to page p2 will form an arc. An example of a small-scale Web with only four pages is shown in Figure 1. Each of the other three pages is linked from page A, but page B only has connections to pages A and D, page C only has a link to page A, and page D only has links to pages B and C.



A hypothetical example of the Web

Figure 1

Let's say a random user opens Figure 1 at page A. There are links to pages B, C, and D, so there is a 1/3 chance that the next user will land on each of those pages; there is no chance that they will land on page A. A random surfer at step two has a 50% chance of being at A, a 50% chance of being at D, and a 0% chance of being at either B or C.

In general, the Web's transition matrix can be used to represent what occurs to idly browsing individuals after one step. If there are n pages, this matrix M has n rows and n columns. If page j contains k outward arcs, and one of them is to page I the element m_{ij} in row I and column j has value 1/k. If not, m_{ij} equals 0.

5.3 EFFICIENT COMPUTATION OF PAGERANK

We must run a matrix-vector multiplication on the order of 50 times to calculate the PageRank for a sizable graph that represents the Web, until the vector is nearly unchanged after one iteration. We have two problems to solve:

1. The Web M's transition matrix is inadequate. Being represented by all of its components is therefore extremely ineffective. Instead, we wish to use the matrix's nonzero members to represent it.

2. We might not be utilizing MapReduce, or we might want to employ a combiner with the Map jobs to lessen the amount of data that needs to be transferred from the Map tasks to the Reduce tasks for efficiency reasons. The striping strategy is insufficient in this instance to prevent excessive disc usage.

5.3.1 Representing transition matrices

Due to the average Web page having only 10 out-links, the transition matrix is quite sparse. Only one entry out of every billion, let's say, ten billion pages in a graph, will not be zero and we require 16 bytes for each nonzero entry if we use 4-byte integers for the coordinates of an element and an 8-byte double-precision number for the value. In other words, the space required is linear with respect to the number of nonzero entries rather than quadratic with respect to the side of the matrix.

There is one more reduction we can perform for a Web transition matrix, though. Each nonzero element is defined as 1 divided by the page's out-degree if the nonzero entries are listed by column. As a result, a column can be represented by one integer for the out-degree and one integer for each nonzero entry, providing the row number where that entry is situated. So, to represent a transition matrix, we require slightly more than 4 bytes for each nonzero item.

5.3.2 PageRank iteration using MapReduce

An estimated PageRank vector v is used in one iteration of the PageRank algorithm to calculate the following estimate v ′ by

$$\mathbf{v}' = \beta M\mathbf{v} + (1-\beta)\mathbf{e}/n$$

Remember that n is the number of nodes that the transition matrix M represents in the graph, β is a constant that is marginally less than 1, and e is a vector of all 1s.

There isn't much going on here other than a matrix-vector multiplication if n is small enough for each Map job to have room in main memory to hold the entire vector v as well as the return vector v ′. The extra steps are to add (1-β)/n to each component of Mv and to multiply each component of Mv by constant β.

It is likely that v is much too big to fit in main memory given the scale of the Web today. We can efficiently run the MapReduce process using the striping technique by splitting M into vertical stripes and splitting v into equivalent horizontal stripes, using no more v at any given Map task than can fit in main memory.

5.3.3 Use of combiners to consolidate the result vector

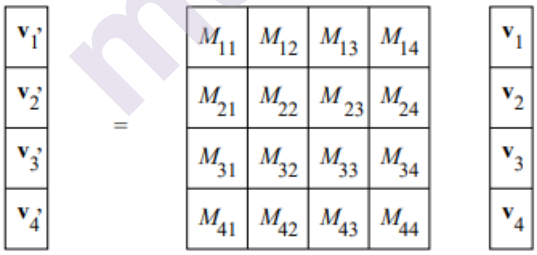
There are two reasons the procedure outlined above could not be adequate.

1. At the Map tasks, we could want to add words for v_i', the ith element of the output vector v. As the Reduce function only adds terms with a common key, this improvement is equivalent to employing a combiner.

2. We might execute the iteration step on a single machine or a group of machines instead of utilizing MapReduce at all.

Assume that we are partitioning a matrix and a vector that do not fit in main memory using the stripe method. Thus, all of the components of the result vector v ′ will include a vertical stripe from the matrix M and a horizontal stripe from the vector v. That vector won't fit in main memory either because it is the same length as v. Furthermore, because M is saved column-by-column for efficiency, a column may have an impact on any of the elements of v′. As a result, it is improbable that the component v_i' we need to add a term to will already be in main memory when we need to do so. So, for the majority of words, adding a page to the appropriate component will necessitate bringing it into main memory. It takes orders of magnitude too long for that condition, known as thrashing, to be possible.

An alternative approach divides the matrix into k^2 blocks while keeping the vectors divided into k stripes. Figure 2 depicts the division for the value of k = 4.



Partitioning a matrix into square blocks

Figure 2

This approach makes use of k^2 Map jobs. Each task is given one stripe of the vector v, which must be v_j, and one square of the matrix M, let's say M_{ij}. Observe that each vector stripe is sent to k distinct Map jobs; for each of the k possible values of i & v_j is delivered to the task handling M_{ij}. As a result, v is sent k times across the network. Each matrix piece is only sent out once, though.

The transmission cost is not significantly higher than the least feasible because the size of the matrix is anticipated to be many times that of the vector. Additionally, we save as data is transferred from the Map activities to the Reduce jobs because the Map tasks perform a significant amount of combining.

The benefit of this method is that while we process M_{ij}, we can keep both the jth stripe of v and the ith stripe of v′ in main memory. Notably, no other stripe of v ′ and all terms derived from M_{ij} and v_j contribute to v_i'.

5.4 TOPIC-SENSITIVE PAGERANK

There are various enhancements we can make to PageRank. We can give specific pages more weight based on their subject, for example. Random surfers' behavior is changed so that they prefer to land on a website that is known to cover the chosen topic as the technique for imposing this weighting.

5.4.1 Motivation for Topic-Sensitive PageRank

People have diverse interests, and occasionally different interests are indicated in queries using the same phrase. The most common example is the search term "jaguar," which may be used to refer to a variety of things, including an animal, a car, an older version of the MAC operating system, and even a video gaming console. A search engine can deliver more pertinent pages to a user if it can determine, for instance, that the user is interested in cars.

Ideally, each user would have a private PageRank vector that gives the relevance of each page to that user. For each of a billion users, it is not practical to keep a vector of length many billions, so we must find a simpler solution. One vector is created for each of a select few topics in the topic-sensitive PageRank approach, which tilts the PageRank in favor of pages related to that topic. While precision is undoubtedly lost, the advantage is that we just maintain a small vector for each user rather than a large vector for each user.

5.4.2 Using Topic-Sensitive PageRank

The following steps must be taken in order to include topic-sensitive PageRank into a search engine:

1. Choose the subjects for which we will build specialized PageRank vectors.
2. Choose a teleport set and use it to calculate the topic-sensitive PageRank vector for each of these topics.
3. Discover a method for identifying the topic or topics that are most pertinent to a specific search query.
4. When arranging the results of the search query, use the PageRank vectors for that topic or those topics.

Using the Open Directory's top-level subjects is one method of choosing the topic set. Various methods are viable, although at least some pages will likely require human classification. The third stage is most likely the most challenging, and numerous approaches have been suggested. Many options:

(a) Provide a menu from which the user can choose a topic.

(b) Deduce the topic(s) based on the words that appear in the Web pages that the user most recently searched for or in recent queries that the user issued.

(c) Infer the topic(s) from user information, such as their Facebook interests or their bookmarks.

5.5 Link Spam

When it became clear that Google's PageRank and other strategies rendered term spam ineffectual, spammers resorted to strategies intended to trick the PageRank algorithm into giving some pages a higher value than others. Link spam refers to all methods used to artificially raise a page's PageRank.

This section will look at numerous strategies for reducing the effectiveness of these spamming techniques, including TrustRank and measuring of spam bulk. We will start by looking at how spammers construct link spam.

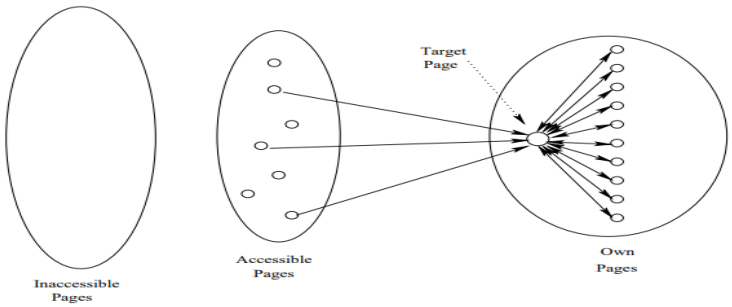
5.5.1 Architecture of a spam farm

A spam farm is a group of pages whose sole function is to raise the PageRank of another page or set of pages. The most basic type of spam farm is seen in Figure 3. The Web is divided into three sections from the perspective of the spammer:

1. Inaccessible pages: those which the spammer is unable to alter. This area makes up the bulk of the Web.

2. Accessible pages: those pages that, while not under the spammer's direct control, are susceptible to their effects.

3. Own pages: These are the pages that the spammer owns and is in charge of.



The Web from the point of view of the link spammer

Figure 3

The spam farm is made up of a few connections from the accessible pages to the spammer's pages as well as the spammer's own pages, which are organized in a unique style as seen on the right. The spam farm would be meaningless without some links from other sources because a normal search engine would not even scan it.

It can be unexpected that one can influence a page without owning it when it comes to the accessible pages. Nonetheless, there are many websites today, including blogs and newspapers, that allow visitors to leave comments. The spammer makes a lot of "I agree" comments in an effort to get as much PageRank flowing to his own pages from outside sources.

The spammer tries to place as much PageRank as they can on the target page in the spam farm, which is page t. The percentage of PageRank that is spread equally to all pages is accumulated by a large number of supporting pages, or m. The supporting pages also, to the extent that is possible, prevent the PageRank of t from being lost as some will be taxed away at each round. Take note that each supporting page only links to t, and that each supporting page has a link to t.

5.5.2 Analysis of a spam farm

Assume that the taxation parameter β which is normally about 0.85, is used to calculate PageRank. The portion of a page's PageRank that is distributed to its successors in the following round is known as β. Let's assume that there are n total pages on the Web, some of which are spam farms of the type depicted in Fig. 3 with a target page t and m supporting pages. Let x represent the portion of PageRank that the accessible pages have contributed. In other words, x is the product of the PageRank multiplied by the number of p's successors divided by the total number of accessible sites p with links to t. Let y represent the PageRank of t that is unknown. We will find the solution for y.

Each supporting page's PageRank is

$$\beta y/m + (1 - \beta)/n$$

The contribution from t is shown in the first phrase. Just the PageRank y of t is transferred to t's successors since βy is charged. The m supporting pages each receive an equal share of that PageRank. The second phrase refers to how much of 1 − β the PageRank, which is distributed evenly among all Web pages, goes to the supporting page.

Let's now calculate the target page's PageRank y. The three sources for PageRank are as follows:

1. An outside contribution, as we have presumed.

2. β times each supporting page's PageRank; that is,

$$\beta(\beta y/m + (1 - \beta)/n)$$

3. 1−β, the portion of the PageRank that belongs to t, expressed as (1-β)/n. This amount will be eliminated from the analysis because it is insignificant.
   From (1) and (2) above, we can therefore write

$$y = x + \beta m\left(\frac{\beta y}{m} + \frac{1-\beta}{n}\right) = x + \beta^2 y + \beta(1-\beta)\frac{m}{n}$$

We may solve the above equation for y, yielding

$$y = \frac{x}{1-\beta^2} + c\frac{m}{n}$$

where $c = \beta(1-\beta)/(1-\beta^2) = \beta/(1+\beta)$.

5.5.3 Combining link spam

Just as it was vital in the prior decade to get rid of word spam, it is now crucial for search engines to identify and delete link spam. Link spam can be approached in one of two ways. One is to search for structures like the spam farm in Fig. 3 where a page links to a huge number of other pages, all of which link back to the original page. Such structures are undoubtedly detected by search engines, which remove those pages from their index. Because of this, spammers create various structures that effectively achieve the same goal of grabbing PageRank for a target website or set of target pages. With the advent of the Internet, the world has become a much smaller place, and the world has become a much smaller place.

There is another method for getting rid of link spam, though, and it doesn't depend on finding the spam farms. Instead, a search engine can change its definition of PageRank to automatically reduce the rank of link-spam pages. We'll look at two distinct formulas:

1. TrustRank, a topic-sensitive PageRank variant created to devalue spammy pages.

2. Spam mass, a formula that determines which pages are most likely to contain spam and enables the search engine to either remove those pages or severely degrade their PageRank.

5.5.4 TrustRank

TrustRank is a topic-sensitive version of PageRank, where the "subject" is a collection of pages that are thought to be reliable (not spam). According to the hypothesis, it is unlikely for a trustworthy page to connect to a spam page, even though it might be simple to make a spam page link to a trustworthy one. A website featuring blogs or other options for spammers to establish links falls into the borderline category. Even if the pages' own content is quite dependable, as would be the case for a reputable newspaper that permitted people to add comments, these pages cannot be regarded as trustworthy.

We must create a suitable teleport set of trustworthy pages in order to apply TrustRank. There have been two methods tested:

1. Let people go over a collection of pages and pick which ones are reliable. For instance, based on the premise that while link spam might improve a page's rank from the bottom to the centre of the pack, it is virtually hard to give a spam page a PageRank towards the top of the list, we might choose the pages with the highest PageRank to analyse.

2. Choose a domain whose membership is restricted, presuming that spammers will find it difficult to obtain their sites into these domains. For instance, since university pages are unlikely to be spam farms, we could choose the.edu domain. One can select pick .mil, or .gov. The issue with these particular options is that they are nearly entirely US-based websites. We should include comparable international websites, such ac.il or edu.sg, to acquire a good distribution of reliable Web pages.

Today's search engines probably regularly use a second type of technique, making what we commonly refer to as PageRank actually a type of TrustRank.

5.5.5 Spam Mass

Using spam mass, we track the percentage of each page's PageRank that is attributable to spam. We accomplish this by calculating both the standard PageRank and the TrustRank using a selected set of reliable pages. Let's say page p has TrustRank t and PageRank r. Therefore (r-t)/r is the spam mass of p. A spam mass near to one indicates that the page is probably spam, whereas a negative or small positive spam mass indicates that the page is probably not spam. Without having to identify specific structures that spam farmers use, it is possible to remove pages with a high spam mass from the index of Web pages utilized by a search engine, greatly reducing link spam.

5.6 SUMMARY

Early search engines were prone to term spam, which is the injection of words into Web pages that falsely reflect what the page is about, making it impossible for them to produce relevant results.

Google used two strategies to combat word spam. The PageRank algorithm, which evaluates the relative relevance of Web pages, came first. The second was a tactic of not just believing what a page claimed about itself, but also what other pages said about it in or near their connections to that page.

Every page on the Web is given a real number, or its PageRank, using the PageRank algorithm. A page's importance or likelihood of providing an effective answer to a search query is indicated by its PageRank. The recursive equation "a page is important if important pages link to it" has a simple solution in PageRank.

5.7 LIST OF REFERENCES

1] Mining of Massive Datasets, Anand Rajaraman and Jeffrey David Ullman, Cambridge University Press, 2012.

2] Data Mining: Introductory and Advanced Topics, Margaret H. Dunham, Pearson, 2013.

3] Big Data for Dummies, J. Hurwitz, et al., Wiley, 2013.

4] Networks, Crowds, and Markets: Reasoning about a Highly Connected World, David Easley and Jon Kleinberg, Cambridge University Press, 2010.

5] Lecture Notes in Data Mining, Berry, Browne, World Scientific, 2009.

6] Data Mining: Concepts and Techniques third edition, Han and Kamber, Morgan Kaufmann, 2011.

7] Data Mining Practical Machine Learning Tools and Techniques, Ian H. Witten, Eibe Frank, The Morgan Kaufmann Series in Data Management Systems, 2005.

8] Big Data Analytics: From Strategic Planning to Enterprise Integration with Tools, Techniques, NoSQL and Graph, David Loshin, Morgan Kaufmann Publishers, 2013.

5.8 UNIT END EXERCISES

1] Explain the concept of PageRank.

2] What do you mean by Efficient Computation of PageRank?

3] How will you represent transition matrices?

4] Explain the PageRank iteration using MapReduce.

5] Describe the use of combiners to consolidate the result vector.

6] Write a note on Topic-Sensitive PageRank.

7] What is the motivation for Topic-Sensitive PageRank?

8] Explain how to use Topic-Sensitive PageRank.

9] Write a note on Link Spam.

10] Explain the architecture of a spam farm.

11] Write a note on Analysis of a spam farm.

12] How will you combine link spam?

13] Explain TrustRank.

14] Describe Spam Mass.

❄❄❄❄❄❄❄

6

RECOMMENDATION SYSTEMS

Unit Structure :

6.0 OBJECTIVES

- To understand the fundamentals of recommendation system
- To get familiar with the recommendation-based models and filtering strategies

6.1 INTRODUCTION

Predicting user responses to options is a key component of a wide range of Online applications. An option like this is referred to be a recommendation system. The most significant instances of these systems will be surveyed at the start of this chapter. Nonetheless, in order to highlight the issue, two excellent instances of recommendation systems are as follows:

1. Based on a prediction of reader interests, offering news stories to online newspaper readers.

2. Making suggestions to online shoppers about potential purchases based on their past purchasing patterns and/or product searches.

Systems that make recommendations employ a variety of technologies. These systems can be divided into two main categories.

- Content-based systems look at the recommended products' characteristics. If a Netflix subscriber, for instance, has watched a lot of cowboy movies, then suggest a film that is listed in the database as belonging to the "cowboy" category.
- Items are recommended by collaborative filtering algorithms based on user and/or item similarity metrics. The products that are suggested to a user are ones that similar users prefer.

6.2 A MODEL FOR RECOMMENDATION SYSTEMS

We present a recommendation system concept that is based on a utility matrix of user preferences.

6.2.1 The Utility Matrix

There are two sorts of entities in a recommendation-system application that we will refer to as users and items. People have preferences for particular goods, and it is necessary to extract these preferences from the data. The data is presented as a utility matrix, where each user-item pair is assigned a value that reflects what is known about the level of that user's preference for that particular item. Values originate from an ordered set, such as integers 1 through 5 that reflect the number of stars the user assigned that item in their rating. Assuming that the matrix is sparse, the majority of the entries are assumed to be "unknown." Unknown ratings imply that we are unaware of the user's explicit preferences for the product.

Example: An illustration of a utility matrix, including user ratings for movies on a scale of 1 to 5, is shown in Figure 1. Blanks signify instances where the user has not given the movie a rating. The titles of the films are TW for Twilight, HP1, HP2, and HP3 for Harry Potter I, II, and III, SW1, SW2, and SW3 for Star Wars episodes I, II, and III, respectively. A through D in capital letters stand in for the users.

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

A utility matrix representing ratings of movies on a 1–5 scale

Figure 1

The majority of user-movie combinations are blank, which indicates that the user has not given the film a rating. In reality, the matrix would be

considerably more sparse, with the average user rating covering only a tiny portion of the films that are really available.

A recommendation system's objective is to anticipate the utility matrix's gaps. Would user A, for instance, enjoy SW2? The little matrix in Fig.1 provides scant evidence. Our recommendation engine may be built to take into account a movie's producer, director, stars, or even the resemblance of their names. If so, we might then see the similarities between SW1 and SW2, and if A did not like SW1, they probably wouldn't like SW2 either. If we had access to additional data, we might also see that those who assessed both SW1 and SW2 tended to rank both similarly. Hence, we could draw the conclusion that A would evaluate SW2 poorly, same to how A evaluated SW1.

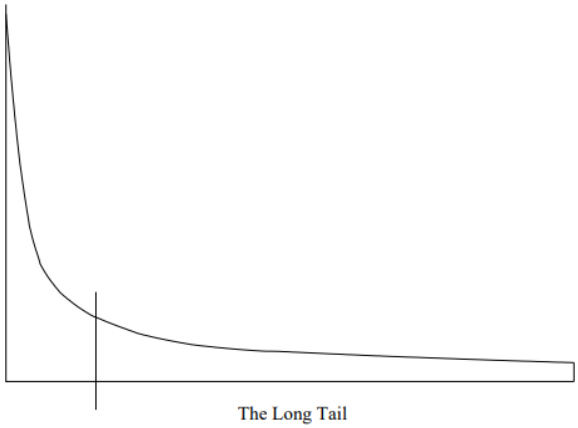
We should also be aware of a somewhat different objective that is appropriate in a variety of contexts. In a utility matrix, it is not required to predict every empty entry. Instead, all that is required is to identify a few entries in each row that are most likely to be high. The recommendation mechanism in most apps proposes a few items that the user should regard highly rather than providing them with a ranking of all available options. It might not even be necessary to locate every item with the greatest projected rating; rather, only a sizable portion of those with the highest ratings need to be located.

6.2.2 The Long Tail

Let's think about the long tail phenomenon that necessitates recommendation systems. A lack of resources is a characteristic of physical delivery systems. Brick-and-mortar retailers have limited shelf space and can only display a small portion of all available options to the customer. On the other side, online shops can give customers access to whatever that is available. As a result, while an actual bookstore might have few thousand books on the shelf, Amazon has millions. While online news sources offer thousands of stories daily, traditional newspapers can only print a few dozen.

In the real world, recommendations are relatively straightforward. The store cannot be customized for each individual customer, to start. In this way, only the total numbers determine what is made available. Normally, a bookstore would only display the best-selling titles, and a newspaper will only publish pieces it thinks will be read by the largest audience. In the first scenario, decisions are driven by sales numbers, but in the second scenario, editorial judgement is used.

The long tail phenomenon, which suggests a separation between the offline and online worlds, has been described in figure 2. Popularity is shown via the vertical axis (the number of times an item is chosen). On the horizontal axis, the items are arranged in ascending order of popularity. Whereas similar online institutions offer the whole range of things, including the tail as well as the popular items, physical institutions only offer the most popular items to the left of the vertical line.



The long tail: physical institutions can only provide what is popular, while on-line institutions can make everything available

Figure 2

Online businesses are compelled to make product recommendations to specific customers due to the long-tail phenomena. It is not possible to offer all available things to the consumer, the way physical institutions can. We cannot also assume that users are familiar with all of the products they may be interested in.

6.2.3 Applications of recommendation system

1. Product Recommendations: Online retailers utilize recommendation systems primarily for product recommendations. We have seen that Amazon and other online retailers make an effort to offer each returning customer a few ideas for things they might like to purchase. These recommendations are not random; rather, they are based on what comparable consumers have purchased or on other methods.

2. Netflix provides its consumers with suggestions for movies they might enjoy. These suggestions are based on user ratings, much like the ratings offered in the illustration utility matrix in Figure 1. Because it is so crucial to anticipate ratings correctly, Netflix offered a prize of $1 million to the first algorithm that could outperform its own recommendation system by 10%. After more than three years of competition, "Bellkor's Pragmatic Chaos," a group of researchers, ultimately took home the award in 2009.

3. News Articles: Based on the articles that users have previously read, news services have made an effort to identify items that will be of interest to them. The closeness could be explained by the similarity of key phrases in the texts or by articles that readers with similar reading preferences have read. The same guidelines apply when suggesting blogs among the millions of blogs available, YouTube videos, or other websites with regularly updated material.

6.3 CONTENT-BASED RECOMMENDATIONS

Systems that are content-based emphasize an item's attributes. By comparing how similar their properties are, we can gauge how similar two things are.

6.3.1 Item profiles

In a content-based system, we must create a profile for each item, which is a record or set of records that represents the object's key attributes. In straightforward situations, the item's features that are simple to find make up the profile. Think of a movie's characteristics that would be useful to a recommendation system, for instance

1. The film's cast of actors. Some moviegoers favor films starring their favorite actors.
2. The leader. Some viewers favor the films of particular directors.
3. The year the film was produced. Some moviegoers favor older films, while others only watch the most recent ones.
4. The movie's genre or broad category. Some prefer dramas or romances, while some viewers solely enjoy comedies.

There are numerous additional aspects of movies that can be utilized. The details, with the exception of the last one, genre, are easily accessible from movie synopses. The idea of genre is hazier. Yet, based on a list of frequently used phrases, movie reviewers typically assign a genre. For instance, the Internet Movie Database (IMDB) categorizes each film into one or more genres.

We can also collect features from many other groups of products, even when the data must initially be manually supplied. For instance, product descriptions frequently include qualities specific to that class of product (such as the TV's screen size and cabinet colour). Similar to movie descriptions, book descriptions include information on the author, the year the book was published, and the genre. Features such as artist, composer, and genre are featured on music goods including CDs and MP3 downloads.

6.3.2 Discovering features of documents

A recommendation system can be useful for many different types of publications. For instance, we are unable to read all of the news pieces that are published each day. A recommendation engine can propose articles on subjects that a user may be interested in, but how do we differentiate between subjects? Likewise, a collection of documents includes web pages. Can you suggest pages that a user might find interesting? Similarly, blogs could be suggested to users who are interested if we could categorize blogs by topics.

These types of documents don't frequently contain features that make information easy to access. The identification of terms that describe a

document's topic has shown to be a useful replacement. Start by removing stop words, which are among the several hundred most prevalent terms but rarely describe the subject of a document. Calculate the TF.IDF score for each of the remaining words in the text. The terms that best describe the document received the highest marks.

Thus, we can consider the n words with the highest TF.IDF scores to be the characteristics of a document. It is feasible to choose n so that it is constant across all documents or to set n to a defined portion of the document's words. We might alternatively decide to include in the feature set all words whose TF.IDF scores are greater than a particular cutoff.

Nowadays, word sets are used to represent documents. Intuitively, we anticipate that these words will convey the subjects or important points of the essay. For instance, we would anticipate the terms with the highest TF in a news piece. IDF score to take into account the names of the people mentioned in the article, the event's distinctive characteristics, and its location. There are numerous natural distance measures we may use to gauge how similar two publications are to one another:

1. To distinguish across word sets, we could use the Jaccard distance.

2. We might use the vector-based cosine distance between the sets.

Consider the collections of high-TF. IDF words as a vector with one component for each potential word in order to calculate the cosine distance in option (2). The vector contains 1 if the word is in the set and 0 otherwise. The infinite dimensionality of the vectors is irrelevant because there are only a finite number of words between two documents in each of their two sets. Both have almost entirely 0 components, therefore 0s have no bearing on the dot product's value. The lengths of the vectors are the square roots of the number of words in each set, and the dot product represents the size of the intersection of the two sets of words.

6.3.3 Obtaining item features from tags

As an illustration of how features have been acquired for items, let's look at a database of photographs. The issue with photos is that their data, which is usually an array of pixels, does not provide any valuable information about their characteristics. Simple pixel characteristics like the average amount of red in the image can be calculated, but few consumers specifically seek out or enjoy red images.

There have been several attempts to gather details about an item's qualities by asking people to tag the products with terms or phrases that best represent them. As a result, a photo with a lot of red may be labelled "Tiananmen Square," whereas a photo with a sunset in Malibu may be tagged "Tiananmen Square." The distinction cannot be identified by the image-analysis software that is currently available.

Tags can be used to describe the characteristics of almost any type of data. The website del.icio.us, later acquired by Yahoo!, allowed users to tag Web

pages and was one of the first attempts to categorize enormous volumes of data. This tagging was done with the intention of creating a new search method where users could submit a list of tags as their search query and have the system retrieve the marked web pages. The tags can also be utilized as a recommendation system, though. We can suggest further pages with the same tags if it is seen that a user frequently accesses or bookmarks pages with a particular set of tags.

The issue with using tags as a feature discovery strategy is that the process only works if users are willing to put in the effort to generate the tags, and there are enough tags that the occasional false positive won't significantly sway the system.

6.3.4 User profiles

In addition to creating vectors that describe things, we also need to build vectors that include the same elements that characterize user preferences. The utility matrix serves as a representation of the relationship between users and items. The nonblank matrix elements might simply be 1s signifying user purchases or another connection, or they could be random values representing a rating or level of affection that the user has for the item.

The user's preferences for particular goods can best be inferred from this data by combining the profiles of those things. The natural aggregate is the average of the elements of the vectors representing the item profiles for the items in which the utility matrix has 1 for that user if the utility matrix has only 1s.

6.4 COLLABORATIVE FILTERING

We will not consider modified methodology to recommendation. We concentrate on the similarity of the user ratings for two items rather than using the attributes of the items to compare their similarity. In other words, we use an item's column in the utility matrix instead of the item-profile vector for that item. Also, we represent users by their rows in the utility matrix rather than by creating a profile vector for each. If two users' vectors are comparable when compared using a distance metric such the Jaccard or cosine distance, then they are similar. Then, a recommendation is generated for user U by examining the users who are most like U in this regard and suggesting things that these users enjoy. Collaborative filtering refers to the technique of finding users who are similar to you and proposing things they would enjoy.

6.4.1 Measuring similarity

How to determine user or item similarity from their rows or columns in the utility matrix is the first issue we need to address. Now, as Fig. 3, we have a copy of Fig. 1. Although the sample size is too small to allow for any conclusive analysis, it will nonetheless highlight some of the difficulties in selecting a distance measure. Keep an eye out for users A and C in particular. They both gave high ratings to the same two films, despite having

seemingly differing views on them. A good distance measurement would place them fairly far away, as we might anticipate. Here are some alternatives to take into account.

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

The utility matrix introduced in Fig. 1

Figure 3

Jaccard distance: We might disregard the matrix's values and concentrate solely on the rated sets of objects. This measure would be appropriate if the utility matrix only included purchases. Unfortunately, the Jaccard distance loses crucial information when utilities are rated in more detail.

Cosine distance: Blanks can be regarded as a 0 value. This decision raises some red flags because it treats the absence of a rating as more indicative of disliking the film than of appreciating it.

Rounding the data: By rounding the numbers, we could attempt to get rid of the perception that movies with high and low ratings are alike.

Normalizing ratings: By deducting the average rating for that user from each rating, we can normalize ratings, which will convert low ratings into negative values and high ratings into positive ones. The users who had opposing viewpoints of the movies they had both seen can be thought of as being as far apart as is feasible if we then calculate the cosine distance between them. Those with comparable views on the rated films will, however, have a pretty tiny angle between them.

6.4.2 The duality of similarity

The utility matrix may be interpreted as providing information about users, items, or both. The symmetry is violated in practice in two different ways.

- To provide product recommendations, we can leverage user information. In other words, we can determine how many people are the most similar to a certain user. We can base our recommendations on the choices made by these comparable people; for example, we could suggest the products that the majority of them have bought or rated highly. There isn't any symmetry, though. Even if we locate pairings of related things, we still need to go one more step before we can suggest products to users.

- In terms of similarity, there is a distinction between the typical behavior of users and goods. Intuitively, things may usually be categorized in straightforward ways. For instance, music typically falls under a particular genre. For instance, a piece of music cannot be

both 1960s rock and 1700s baroque. On the other side, there are people who enjoy both 1960s rock and 1700s baroque music and who purchase copies of both. As a result, it is simpler to find objects that are similar because they belong to the same genre than it is to identify users who are similar because they love the same genre but also enjoy different genres that the other person does not.

Finding the n users (for some specified n) who are the most similar to U and averaging their ratings for item I, only including those among the n similar users who have rated I, is one method of forecasting the value of the utility matrix entry for user U and item I, as we mentioned in (1) above. In general, normalizing the matrix first is preferable. Subtract the average rating for things from the average rating for user I for each of the n users. Users who have rated I should average the discrepancy, and this average should then be added to the overall average rating provided by U.

Moreover, we can estimate the entry for user U and item I using item similarity. Identify the m items that are most comparable to I for certain m, then average the m items' ratings based on U's ratings. In terms of user-user similarity, we only take into account the m items that U has rated, and it is usually a good idea to first normalize the item ratings.

Be aware that finding only one entry in the utility matrix is not adequate, regardless of the method used to estimate the entries. We need to estimate every entry in the utility matrix row for user U in order to provide recommendations to them. Or, at the absolute least, we need to identify all or most of the entries in that row that are blank but have a high estimated value. There is a trade-off between using similar users or similar objects as a starting point.

- If we discover comparable users, we only need to repeat the procedure once for user U. We can estimate all of the blanks in the utility matrix for U from the set of comparable users. Before we can estimate the row for U if we work with similar things, we must compute similar items for virtually all items.

- On the other hand, due to the phenomenon mentioned above, where it is simpler to find products belonging to the same genre than it is to find consumers who exclusively like goods belonging to a single genre, item-item similarity frequently offers more trustworthy information.

6.4.3 Clustering users and items

Because the sparse utility matrix contains little information regarding user-item pairs, it is challenging to identify similarities between objects or users. Even if two products are in the same genre, very few customers are likely to have purchased or rated both. Likewise, two people who enjoy the same genre or genres may not have purchased any things together.

To avoid this pitfall, group users or/and things together. To perform a clustering of, let's say, objects, use any one of the distance metrics. We'll

see, though, that there may not be much of a motivation to try to cluster into a few clusters right once. As a first stage, a hierarchical strategy that leaves many clusters unmerged may be sufficient. We may, for instance, leave half the number of clusters as there are objects.

6.5 SUMMARY

Systems that make recommendations work with users and things. A utility matrix provides known details regarding how much a user enjoys a product. The primary challenge in making recommendations to users is predicting the values of the majority of entries, the majority of which are typically unknown, based on the values of the known elements.

By identifying related objects and the user's reaction to those, these systems try to anticipate how a user would react to a particular item. One type of recommendation system is content-based, and it determines how similar two items are by looking for features they have in common. Collaborative filtering is used in a second class of recommendation systems that compare individuals based on their item preferences and/or users who already like the items in question.

6.6 LIST OF REFERENCES

1] Mining of Massive Datasets, Anand Rajaraman and Jeffrey David Ullman, Cambridge University Press, 2012.

2] Data Mining: Introductory and Advanced Topics, Margaret H. Dunham, Pearson, 2013.

3] Big Data for Dummies, J. Hurwitz, et al., Wiley, 2013.

4] Networks, Crowds, and Markets: Reasoning about a Highly Connected World, David Easley and Jon Kleinberg, Cambridge University Press, 2010.

5] Lecture Notes in Data Mining, Berry, Browne, World Scientific, 2009.

6] Data Mining: Concepts and Techniques third edition, Han and Kamber, Morgan Kaufmann, 2011.

7] Data Mining Practical Machine Learning Tools and Techniques, Ian H. Witten, Eibe Frank, The Morgan Kaufmann Series in Data Management Systems, 2005.

8] Big Data Analytics: From Strategic Planning to Enterprise Integration with Tools, Techniques, NoSQL and Graph, David Loshin, Morgan Kaufmann Publishers, 2013.

6.7 UNIT END EXERCISES

1] Explain a Model for Recommendation Systems.

2] What is the Utility Matrix?

3] Explain: The Long Tail.

4] Illustrate the applications of recommendation system.

5] What is Content-Based Recommendations?

6] What do you mean by Item profiles?

7] How to discover features of documents?

8] Describe the concept of obtaining item features from tags.

9] What are User profiles?

10] Explain Collaborative Filtering.

11] Write a note on Measuring similarity.

12] Describe the duality of similarity.

13] Describe Clustering users and items.

❄❄❄❄❄❄❄

7

DIMENSIONALITY REDUCTION

Unit Structure :

7.0 OBJECTIVES

- To acquaint with the fundamentals of dimensionality reduction
- To get familiar with various components and decomposition methods involved in dimensionality reduction

7.1 INTRODUCTION

There are numerous data sources that can be visualized as a sizable matrix. Finding "narrower" matrices that, in some ways, are similar to the original matrix can be used to summarize the matrix in various situations. These little matrices can be used considerably more effectively than the original large matrix because they only have a small number of rows or columns. Dimensionality reduction is the process of locating these little matrices.

7.2 EIGEN VALUES AND EIGEN VECTORS OF SYMMETRIC MATRICES

The fundamentals of matrix algebra, such as multiplication, transposition, determinants, and solving linear equations, are assumed to be acquainted to you. We will define eigenvalues and eigenvectors of a symmetric matrix in this section and demonstrate how to locate them. Remember that if an element in row i and column j equals an element in row j and column i a matrix is symmetric.

7.2.1 Definitions

The matrix M should be square. Let M have the same number of rows as e, a nonzero column vector, and let be a constant. If Me = λe, then e is the appropriate eigenvector of M and λ is an eigenvalue of M.

It is also true that c e is an eigenvector of M with the same eigenvalue if e is an eigenvector of M and c is any constant. A vector's length is altered by multiplying it by a constant, but not its direction. We will therefore demand that each eigenvector be a unit vector, which means that the sum of the squares of the vector's components is 1. This will prevent confusion regarding the length. Even yet, we may still multiply by -1 without changing the sum of squares of the component parts, so the eigenvector is not entirely unique. As a result, we will typically demand that an eigenvector's first nonzero component be positive.

7.2.2 Computing Eigen values and Eigen vectors

Starting with any unit vector v of the right length and computing $M^i v$ iteratively until it converges is one method for obtaining an eigenpair (an eigenvalue and its corresponding eigenvector) for a suitable matrix M. The primary eigenvector, or the eigenvector with the biggest eigenvalue, is the limiting vector when M is a stochastic matrix, and its corresponding eigenvalue is 1. This process, known as power iteration, finds the principal eigenvector in most cases, but if the principal eigenvalue (the eigenvalue connected to the principal eigenvector) is not 1, then as i increases, the ratio of $M^{i+1}v$ to $M^i v$ approaches the principal eigenvalue while $M^i v$ approaches a vector (likely not a unit vector) with the same direction as the principal eigenvector.

Even though some eigenvalues may be identical in some instances, there will always be n eigenpairs. The first step in the procedure is to rewrite the equation Me = λe, which identifies eigenpairs, as (M − λI)e = 0, where

1. I is the nxn identity matrix, which has 1s along the major diagonal and 0s elsewhere.

2. A vector of all 0s is called 0

It is a fact of linear algebra that the determinant of M − λI must be 0 for (M − λI)e = 0 to hold for a vector e ≠ 0. You'll see that (M − λI) resembles the matrix M almost exactly, except if M has c in one of its diagonal

components, then (M − λI) also has c there. The approach of "pivotal condensation," for instance, can be used to compute the determinant of a nxn matrix in $O(n^3)$ time despite the fact that it has n! terms.

The nth-degree polynomial in λ that makes up the determinant of (M − λI) is where we may get the n values of λ that correspond to the eigenvalues of M. The equation Me = c e can then be solved for any such number, let's say c. There are n equations in n unknowns (the n components of e), but since no equation contains a constant term, we can only determine e to a constant factor of accuracy. The eigenvector that corresponds to eigenvalue c can be obtained by normalizing the data using any solution, though, so that the sum of the squares of the components equals 1.

7.2.3 Finding eigen pairs by power iteration

From among the different eigenvectors of the stochastic matrix of the Web, the PageRank vector was the major eigenvector, which was found by looking at the generalization process in that section. We begin by computing the main eigenvector using a slender generalization technique. The major eigenvector is then effectively removed by altering the matrix. The outcome is a new matrix whose major eigenvector is the second eigenvector of the original matrix, or the eigenvector with the second-largest eigenvalue. In this approach, each eigenvector is eliminated as it is discovered, and the principal eigenvector of the matrix that is left is then found using power iteration.

Let M represent the matrix whose eigenpairs we are interested in finding. Start with any vector that is not zero (x_0), and repeat:

$$\mathbf{x}_{k+1} := \frac{M\mathbf{x}_k}{\|M\mathbf{x}_k\|}$$

where $\|N\|$ for a matrix or vector N denotes the *Frobenius norm*; that is, the square root of the sum of the squares of the elements of N. We multiply the current vector $\mathbf{x}_k$ by the matrix M until convergence (i.e., $\|x_k - x_{k+1}\|$ is less than some small, chosen constant). Let $\mathbf{x}$ be $\mathbf{x}_k$ for that value of k at which convergence is obtained. Then $\mathbf{x}$ is (approximately) the principal eigenvector of M. To obtain the corresponding eigenvalue we simply compute $\lambda_1 = \mathbf{x}^{\mathrm{T}} M\mathbf{x}$, which is the equation $M\mathbf{x} = \lambda\mathbf{x}$ solved for λ, since $\mathbf{x}$ is a unit vector.

7.2.4 The matrix of eigen vectors

Assume we have a nxn symmetric matrix M with the eigenvectors e_1, e_2,..., e_n when viewed as column vectors. Let E be the matrix with e_i as the ith column. $EE^T = E^TE = I$ then. The orthonormal eigenvectors of a symmetric matrix provide the reason. Therefore, they are orthogonal unit vectors.

7.3 Principal Component Analysis

Using the principal-component analysis (PCA) method, one can determine the optimal directions along which a set of tuples representing points in a high-dimensional space line up. The goal is to identify the eigenvectors for

MM^T or M^TM by treating the collection of tuples as a matrix M. One way to conceptualize the matrix of these eigenvectors is as a rigid rotation in a high-dimensional space. The axis along which the points are most evenly distributed is the primary eigenvector when this transformation is applied to the original data. More specifically, this axis is the one along which the data's volatility is greatest. In other words, it is ideal to think of the points as lying along this axis with very minor deviations. The variation of distances from the first axis is greatest along the axis corresponding to the second eigenvector, which is the eigenvector corresponding to the second-largest eigenvalue, and so on.

PCA can be seen as a data-mining method. By projecting the high-dimensional data onto the most significant axes, it can be replaced. The axes that map to the biggest eigenvalues are those. As a result, data with far fewer dimensions and a good summary of the original data are used to approximate the original data.

7.3.1 Using eigen vectors for dimensionality reduction

We can deduce a general rule from the just-worked-out example. We can compute M^TM and its eigenpairs if M is a matrix whose rows each represent a point in a Euclidean space with any number of dimensions. The eigenvectors in the columns of the matrix E should be arranged in ascending order of largest eigenvalue. Assume that the matrix L has 0s in all other entries and the highest Possible eigenvalues along the diagonal. As a result, $M^TME = EL$ because $M^TMe = \lambda e = e\lambda$ for each eigenvector e and its corresponding eigenvalue.

We discovered that ME is a new coordinate space created by transforming the points of M. The first axis in this space is the most important one; officially, the variance of points along that axis is the highest. It corresponds to the largest eigenvalue. For each of the eigenpairs, the second axis, which corresponds to the second eigenpair, is the next most significant in the same way. The option that employs the eigenvectors associated with the largest eigenvalues and disregards the other eigenvalues is the one that keeps the most relevance when we want to downsize the dimensions of the space M.

Specifically, consider the first k columns of E as E_k. ME_k is then a representation of M in k dimensions.

7.3.2 The matrix of distances

Let's look at the MM^T eigenvalues. The matrix in our example M is larger than the one in M because M has more rows than columns; but, if M had more columns than rows, we would obtain a smaller matrix.

Consider an example:

$$MM^{\mathrm{T}} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{bmatrix} = \begin{bmatrix} 5 & 4 & 11 & 10 \\ 4 & 5 & 10 & 11 \\ 11 & 10 & 25 & 24 \\ 10 & 11 & 24 & 25 \end{bmatrix}$$

For further information, please visit the website. It is easy to understand the meaning of the entry in the ith row and jth column; it is the dot product of the vectors that are represented by the ith and jth points (rows of M). The eigenvalues of M^TM and MM^T are closely related to one another. Assume that e is an M^TM eigenvector.

$$M^TMe = \lambda e$$

Add M on the left and multiply both sides of this equation. Then

$$MM^T(Me) = M\lambda e = \lambda(Me)$$

Hence, Me will be an eigenvector of MM^T and λ will be an eigenvalue of MM^T as well as of M^TM as long as it is not the zero vector 0.

The opposite is also true. That is, starting with $MM^Te = e$ and multiplying on the left by M^T will lead to the conclusion that $M^TM(M^Te) = \lambda e$. If e is an eigenvector of MM^T with the corresponding eigenvalue (M^Te). As a result, λ is also an eigenvalue of M^TM if M^Te is not 0.

When $M^Te = 0$, we might be curious what occurs. As 0 cannot be an eigenvector, MM^Te is also 0 in this scenario, but e is not 0. But because $0 = e$, we deduce that $\lambda = 0$.

We come to the conclusion that the eigenvalues of MM^T are those of M^TM plus extra 0s. The eigenvalues of M^TM would be those of MM^T plus additional 0s if the dimension of MM^T were smaller than the dimension of M^TM, but the inverse would be true.

$$\begin{bmatrix} 3/\sqrt{116} & 1/2 & 7/\sqrt{116} & 1/2 \\ 3/\sqrt{116} & -1/2 & 7/\sqrt{116} & -1/2 \\ 7/\sqrt{116} & 1/2 & -3/\sqrt{116} & -1/2 \\ 7/\sqrt{116} & -1/2 & -3/\sqrt{116} & 1/2 \end{bmatrix}$$

Eigenvector matrix for MM^T

7.4 SINGULAR VALUE DECOMPOSITION (SVD)

SVD makes it simple to construct an approximation representation with any desired number of dimensions by easily removing the less significant

portions of the exact representation of any matrix. Obviously, the estimate will be less accurate the less dimensions we select.

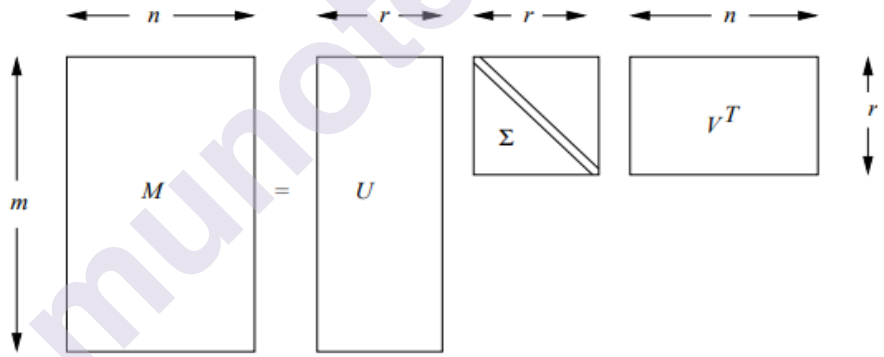
7.4.1 Definition of SVD

Let M be a mxn matrix with rank equal to r. Remember that the maximum number of rows (or, equivalently, columns) we can select for which no nonzero linear combination of the rows is the all-zero vector 0 this is what we mean when we say a set of such rows or columns is independent is the rank of the matrix. We can then locate matrices. Fig. 1 depicts U, Σ, and V with the following characteristics:

1. U is a m × r column-orthonormal matrix i.e., its every columns are unit vector and the dot product of any two columns is 0.

2. V is an orthonormal matrix with n r columns. The rows of V T are orthonormal because we always employ V in its transposed form.

3. Because Σ is a diagonal matrix, all elements that are not on the primary diagonal are equal to 0. The singular values of M refer to the components of Σ.



The form of a singular-value decomposition

Figure 1

7.4.2 Interpretation of SVD

It is important to think of the r columns of U, Σ, and V as standing in for notions that are concealed in the original matrix M if you want to fully grasp what the SVD delivers. Let's imagine that the M rows represent people, and the M columns represent movies. Then matrix U links individuals with ideas. For instance, Joe, who is represented by row 1 of M in Fig. 2, merely like the idea of science fiction. Because Joe exclusively watches science fiction films and doesn't think highly of them, the value 0.14 in the first row and first column of U is lower than some of the other entries there.

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

Ratings of movies by users

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix} \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}$$

M U Σ V^T

SVD for the matrix M

Figure 2

The matrix V links motion pictures to ideas. The first three films—The Matrix, Alien, and Star Wars are all science-fiction films, as indicated by the 0.58 in each of their first three columns of the first row of V^T, but the 0s in the last two columns of the first row indicate that they do not contain any idea romance. The second row of V^T similarly informs us that Casablanca and Titanic are solely romantic comedies.

The matrix Σ provides a summary of each concept's strength. In our example, the strength of the science-fiction concept is 12.4, whereas the strength of the romance concept is 9.5. Intuitively, the science-fiction idea is more compelling because the data offers more details about the genre's films and its fans.

Generally speaking, the notions won't be as clearly defined. Although Σ is always a diagonal matrix and will always include 0's off the diagonal, there will be less 0's in U and V. The entities represented by the rows and columns of M (corresponding to the people and movies in our example) will, to differing degrees, engage in a variety of different notions.

7.4.3 Dimensionality reduction using SVD

If we wish to store these matrices neatly, but we also want to represent a very big matrix M by its SVD components U, Σ, and V. Setting the lowest of the singular values to zero is the best strategy to decrease the three matrices' dimension. The corresponding columns of U and V can be removed if we set the smallest singular values to 0.

7.4.4 Computing the SVD of a matrix

The SVD of a matrix M is strongly connected to the eigenvalues of the symmetric matrices M^TM and MM^T. This relationship allows us to obtain the SVD of M from the eigenpairs of the latter two matrices. To begin the explanation, start with $M = U\Sigma V^T$, the expression for the SVD of M. Then

$$M^T = (U\Sigma V^T)^T = (V^T)^T\Sigma^T U^T = V\Sigma^T U^T$$

Since is a diagonal matrix, it is useless to transpose it. Thus, $M^T = V\ \Sigma U^T$.

M^TM now equals $V\ \Sigma\ U^T U\ \Sigma V^T$. Note that U is an orthonormal matrix, thus U TU is the identity matrix of the right dimension. This is,

$$M^TM = V\Sigma^2V^T$$

To obtain, multiply both sides of the right side of this equation by V

$$M^TMV = V\Sigma^2V^TV$$

Since V is also an orthonormal matrix, we know that V^TV is the identity. Thus

$$M^TMV = V\Sigma^2$$

Given that is a diagonal matrix, Σ^2 is also a diagonal matrix with a square entry in the same location as in the ith row and column. According to the equation above, V is the M^TM eigenvector matrix, and Σ^2 is the diagonal matrix, each of which entries corresponds to an eigenvalue.

As a result, the matrix V for the SVD of M itself is produced by the same process that calculates the eigenpairs for M^TM. Simply take the square roots of the eigenvalues for M^TM to get the singular values for this SVD.

Just U needs to be calculated, but it can be done using the same method we used to locate V. Begin with

$$MM^T - U\Sigma V^T(U\Sigma V^T)^T - U\Sigma V^TV\Sigma U^T - U\Sigma^2U^T$$

Then by a series of manipulations analogous to the above, we learn that

$$MM^TU = U\Sigma^2$$

That is, U is the matrix of eigenvectors of MM^T.

7.5 SUMMARY

The purpose of dimensionality reduction is to replace a large matrix with two or more smaller matrices that can roughly approximate the original matrix's size, typically by calculating their product.

A matrix may include numerous eigenvectors, and when the eigenvector is multiplied by the matrix, the outcome is always a multiple of the

eigenvector. The eigenvalue connected to this eigenvector is that constant. An eigenpair is made up of an eigenvector and its eigenvalue.

When used for dimensionality reduction, principal-component analysis treats data made up of a set of points in a multidimensional space as a matrix, with the points' rows representing the points and the dimensions' columns representing the dimensions. The primary eigenvector, or direction in space along which the points line up best, can be thought of as the product of the principal eigenpairs of this matrix and its transpose. The direction in which departures from the primary eigenvector are largest is represented by the second eigenvector, and so on.

7.6 LIST OF REFERENCES

1] Mining of Massive Datasets, Anand Rajaraman and Jeffrey David Ullman, Cambridge University Press, 2012.

2] Data Mining: Introductory and Advanced Topics, Margaret H. Dunham, Pearson, 2013.

3] Big Data for Dummies, J. Hurwitz, et al., Wiley, 2013.

4] Networks, Crowds, and Markets: Reasoning about a Highly Connected World, David Easley and Jon Kleinberg, Cambridge University Press, 2010.

5] Lecture Notes in Data Mining, Berry, Browne, World Scientific, 2009.

6] Data Mining: Concepts and Techniques third edition, Han and Kamber, Morgan Kaufmann, 2011.

7] Data Mining Practical Machine Learning Tools and Techniques, Ian H. Witten, Eibe Frank, The Morgan Kaufmann Series in Data Management Systems, 2005.

8] Big Data Analytics: From Strategic Planning to Enterprise Integration with Tools, Techniques, NoSQL and Graph, David Loshin, Morgan Kaufmann Publishers, 2013.

7.74 UNIT END EXERCISES

1] Explain the Eigen values and Eigen vectors of symmetric matrices.

2] Define Eigen values and Eigen vectors.

3] Explain the Computing process of Eigen values and Eigen vectors.

4] How you will find the eigen pairs by power iteration.

5] Describe the matrix of eigen vectors.

6] Explain the Principal Component Analysis.

7] Discuss how you will use eigen vectors for dimensionality reduction.

8] Explain the matrix of distances.

9] Write a note on Singular value decomposition.

10] Define SVD.

11] Explain the interpretation of SVD.

12] Describe the dimensionality reduction using SVD.

13] How to computing the SVD of a matrix.

❄❄❄❄❄❄❄❄