**Unit 8**

**Classification and Prediction**

There are two forms of data analysis that can be used for extracting models describing important classes or to predict future data trends. These two forms are as follows:

* Classification
* Prediction

Classification models predict categorical class labels; and prediction models predict continuous valued functions. For example, we can build a classification model to categorize bank loan applications as either safe or risky, or a prediction model to predict the expenditures in dollars of potential customers on computer equipment given their income and occupation.

**What is classification?**

Following are the examples of cases where the data analysis task is Classification:

* A bank loan officer wants to analyze the data in order to know which customer (loan applicant) is risky or which are safe.
* A marketing manager at a company needs to analyze a customer with a given profile, who will buy a new computer.

In both of the above examples, a model or classifier is constructed to predict the categorical labels. These labels are risky or safe for loan application data and yes or no for marketing data.

**What is prediction?**

Following are the examples of cases where the data analysis task is Prediction: Suppose the marketing manager needs to predict how much a given customer will spend during a sale at his company. In this example we are bothered to predict a numeric value. Therefore the data analysis task is an example of numeric prediction. In this case, a model or a predictor will be constructed that predicts a continuous-valued-function or ordered value. Regression analysis is a statistical methodology that is most often used for numeric prediction.

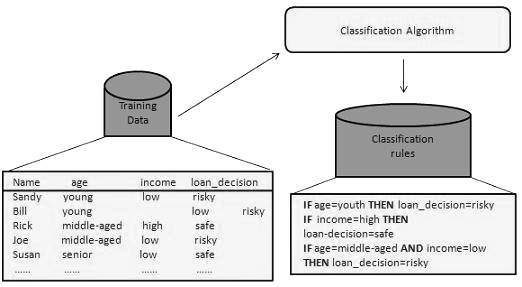
**How Does Classification Works?**

With the help of the bank loan application that we have discussed above, let us understand the working of classification. The Data Classification process includes two steps:

* Building the Classifier or Model
* Using Classifier for Classification

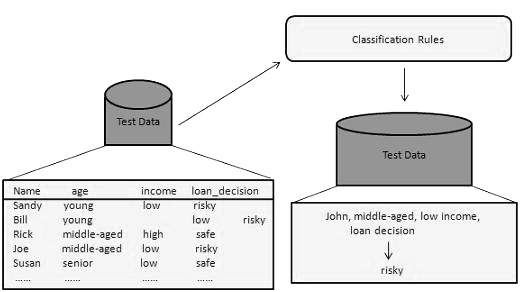
**Building the Classifier or Model**

* This step is the learning step or the learning phase.
* In this step the classification algorithms build the classifier.
* The classifier is built from the training set made up of database tuples and their associated class labels.
* Each tuple that constitutes the training set is referred to as a category or class. These tuples can also be referred to as sample, object or data points.



**Using Classifier for Classification**

In this step, the classifier is used for classification. Here the test data is used to estimate the accuracy of classification rules. The classification rules can be applied to the new data tuples if the accuracy is considered acceptable.



**Classification and Prediction Issues**

The major issue is preparing the data for Classification and Prediction. Preparing the data involves the following activities:

* **Data Cleaning** − Data cleaning involves removing the *noise and treatment of missing values*. The noise is removed by applying smoothing techniques and the problem of missing values is solved by replacing a missing value with most commonly occurring value for that attribute.
* **Relevance Analysis** − Database may also have the irrelevant attributes. Correlation analysis is used to know whether any two given attributes are related.
* **Data Transformation and reduction** − The data can be transformed by any of the following methods:
  + **Normalization** − Data is transformed using normalization. Normalization involves scaling all values for given attribute in order to make them fall within a small specified range. Normalization is used when in the learning step, the neural networks or the methods involving measurements are used.
  + **Generalization** − Data can also be transformed by generalizing it to the higher concept. For this purpose we can use the concept hierarchies.

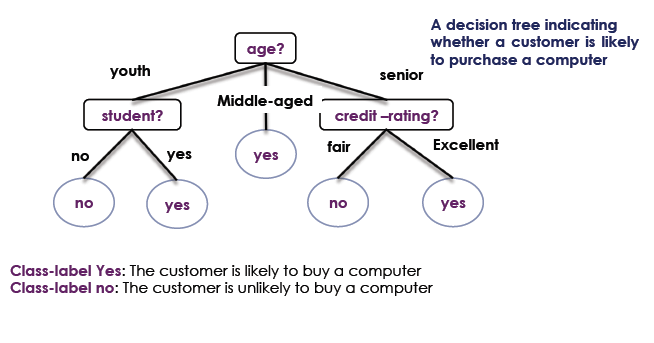
**Comparison of Classification and Prediction Methods**

Here are the criteria for comparing the methods of Classification and Prediction:

* **Accuracy** − Accuracy of classifier refers to the ability of classifier. It predicts the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.
* **Speed** − This refers to the computational cost in generating and using the classifier or predictor.
* **Robustness** − It refers to the ability of classifier or predictor to make correct predictions from given noisy data.
* **Scalability** − Scalability refers to the ability to construct the classifier or predictor efficiently; given large amount of data.
* **Interpretability** − It refers to what extent the classifier or predictor understands.

**Classification by Decision Tree Induction**

**Decision tree induction** is the learning of decision trees from class labeled training tuples. It is a decision tree is a flowchart-like tree structure where **internal nodes** (non leaf node) denotes a test on an attribute b**ranches** represent outcomes of tests **Leaf nodes** (terminal nodes) hold class labels and **Root node** is the topmost node.



The attributes of a tuple are tested against the decision tree. A path is traced from the root to a leaf node which holds the prediction for that tuple.

**Example**

**RID age income student credit-rating Class**

1  *youth high no fair ?*

*Test on age: youth*

*Test of student: no*

*Reach leaf node*

***Class NO:*** *the customer Is Unlikely to buy a computer*

**Why decision trees classifiers are so popular?**

* Why decision trees classifiers are so popular?
* The construction of a decision tree does not require any domain
* knowledge or parameter setting
* They can handle high dimensional data
* Intuitive representation that is easily understood by humans
* Learning and classification are simple and fast
* They have a good accuracy

**Algorithm for constructing Decision Tress**

Constructing a Decision tree uses **greedy algorithm.** Tree is constructed in a top-down recursive divide-and-conquer manner.

1. At start, all the training tuples are at the root
2. Tuples are partitioned recursively based on selected attributes
3. If all samples for a given node belong to the same class

* Label the class

1. If There are no remaining attributes for further partitioning

* **Majority voting** is employed for classifying the leaf

1. There are no samples left

* Label the class and terminate

1. Else

* Got to step 2

***Example***

**RID age student credit-rating Class: Buys-computer**

1 youth yes fair yes

2 youth yes fair yes

3 youth yes fair no

4 youth no fair no

5 middle-aged no excellent yes

6 senior yes fair no

7 senior yes excellent yes

***Solution***

***Step 1***



***Step 2***

****

***Step 3***

****

***Step 4***

****

***Step 5***

****

***Step 6***

****

**Prediction**

As already mentioned, numeric prediction is the task of predicting continuous values for given input. For example, we may wish to predict the salary of college graduates with 10 years of work experience, or the potential sales of a new product given its price. The most widely used approach for numeric prediction is regression. Regression analysis can be used to model the relationship between one or more *independent* or predictor variables and a *dependent* or response variable. In general, the values of the predictor variables are known. The response variable is what we want to predict.

**Linear Regression**

Straight-line regression analysis involves a response variable, *y*, and a single predictor variable, *x*. It is the simplest form of regression, and models *y* as a linear function of *x*.

That is,

*y* = *b*+*wx*

Where, the *b* and *w* are regression coefficients specifying the y-intercept and slope of the line respectively. These coefficients, *w* and *b*, can also be thought of as weights, so that we can equivalently write,

*y* = *w*0+*w*1*x*

Let *D* be a training set consisting of values of predictor variable, *x*, for some population and their associated values for response variable, *y*. The training set contains data points of the form(*x*1, *y*1), (*x*2, *y*2), (*x*j, *y*j). The regression coefficients can be estimated using this method with the following equations:

***Example***

Table given below shows a set of paired data where *x* is the number of years of work experience of a college graduate and *y* is the corresponding salary of the graduate. Predict the value of salary after 10 years of experience.

|  |  |
| --- | --- |
| Years of Experience  (x) | Salary  (y) |
| 1 | 20000 |
| 3 | 40000 |
| 6 | 58000 |
| 8 | 66000 |
| 9 | 70000 |
| 12 | 82000 |
| 13 | 85000 |
| 15 | 92000 |
| 17 | 100000 |
| 20 | 108000 |

***Solution***







Now, predict the value by using the equation

*y* = *w*0+*w*1*x=???????? (Calculate Value) use x=10*

**Introduction to Clustering**

The process of grouping a set of physical or abstract objects into classes of *similar* objects

is called clustering. 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. A cluster of data objects can be treated collectively as one group. Although classification is an effective means for distinguishing groups or classes of objects, it requires the often costly collection and labeling of a large set of training tuples or patterns, which the classifier uses to model each group.

Clustering is also called data segmentation in some applications because clustering partitions large data sets into groups according to their *similarity*. Clustering can also be used for outlier detection, where outliers (values that are “far away” from any cluster) may be more interesting than common cases. Applications of outlier detection include the detection of credit card fraud and the monitoring of criminal activities in electronic commerce. For example, exceptional cases in credit card transactions, such as very expensive and frequent purchases, may be of interest as possible fraudulent activity.

**Measure of Distance between Data Points**

During clustering of data objects, we need to find distance between data objects. The most popular distance measure is *Euclidean distance*, which is defined as:



Where, 

Another well-known metric is *Manhattan (or city block) distance*, defined as



*Minkowski distance* is a generalization of both Euclidean distance and Manhattan distance. It is defined as



Where *p* is a positive integer, such a distance is also called *Lp* norm, in some literature.

It represents the Manhattan distance when *p* = 1 (i.e., *L*1 norm) and Euclidean distance

when *p* = 2 (i.e., *L*2 norm).

**Categorization of Clustering Algorithms**

Many clustering algorithms exist in the literature. In general, the major clustering methods can be classified into the following categories.

* **Partitioning methods**: Given a database of *n* objects or data tuples, a partitioning method constructs *k* partitions of the data, where each partition represents a cluster and *k* <*n*. Given *k*, the number of partitions to construct, a partitioning method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another.
* **Hierarchical methods:** A hierarchical method creates a hierarchical decomposition of the given set of data objects. A hierarchical method can be classified as being either *agglomerative* or *divisive*. The *agglomerative approach*, also called the *bottom-up* approach, starts with each object forming a separate group. It successively merges the objects or groups that are close to one another, until all of the groups are merged into one (the topmost level of the hierarchy), or until a termination condition holds. The *divisive approach*, also called the *top-down* approach, starts with all of the objects in the same cluster. In each successive iteration, a cluster is split up into smaller clusters, until eventually each object is in one cluster, or until a termination condition holds.
* **Density-based methods:** Most partitioning methods cluster objects based on the distance between objects. Such methods can find only spherical-shaped clusters and encounter difficulty at discovering clusters of arbitrary shapes. Other clustering methods have been developed based on the notion of *density*. Their general idea is to continue growing the given cluster as long as the density (number of objects or data points) in the “neighborhood” exceeds some threshold.
* **Model-based methods:** Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model. EM is an algorithm that performs expectation-maximization analysis based on statistical modeling.

**K-means Clustering Algorithm**

k-means is  one of  the simplest unsupervised  learning  algorithms  that  solve  the well  known clustering problem. The procedure follows a simple and easy way to classify a given data set  through a certain number of  clusters (assume k clusters) fixed apriori. The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other.

**Algorithmic steps for k-means clustering**

Let  X = {x1,x2,x3,……..,xn} be the set of data points and V = {v1,v2,…….,vc} be the set of centers.

1. Randomly select *‘c’* cluster centers.
2. Calculate the distance between each data point and cluster centers.
3. Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers..
4. Recalculate the new cluster center using:  , where,*‘ci’* represents the number of data points in *ith* cluster.
5. Recalculate the distance between each data point and new obtained cluster centers.
6. If no data point was reassigned then stop, otherwise repeat from step 3.

**Advantages**

* Fast, robust and easier to understand.
* Gives best result when data set are distinct or well separated from each other.

**Disadvantages**

* The learning algorithm requires apriori specification of the number of  cluster centers.
* Randomly choosing of the cluster center cannot lead us to the fruitful result.
* Applicable only when mean is defined i.e. fails for categorical data.
* Algorithm fails for non-linear data set.

*Example*

Divide the data points {(1,1), ((2,1), (4,3), (5,4)} into two clusters.

***Solution***

Let p1=(1,1) p2=(2,1) p3=(4,3) p4=(5,4)

***Initial step***

Let c1=(1,1) and c2=(2,1) are two initial cluster centers.

***Iteration 1***

Calculate distance between clusters centers and each data points

Thus after first iteration

*Cluster 1= {p1}*

*Cluster 2={p2,p3,p4}*

Now, new cluster centers are:

c1=(1,1) and c2={(2+4+5)/3,(1+3+4)/3}=(11/3,8/3)

***Iteration 2***

Calculate distance between new cluster centers and each data points

Thus after second iteration

*Cluster 1= {p1,p2}*

*Cluster 2={p3,p4}*

Now, new cluster centers are:

c1={(1+2)/2,(1+1)/2}={3/2,1} and c2={(2+4+5)/3,(1+3+4)/3}=(11/3,8/3)

Repeat this process till no re-assignment of points to groups.

**K-medoid Clustering Algorithm**

A *medoid* can be defined as the object of a cluster whose average dissimilarity to all the objects in the cluster is minimal. i.e. it is a most centrally located point in the cluster. In contrast of K-means algorithm, K-medoid algorithm chooses data point as centers and works with arbitrary matrix of distances instead of *l2*.*K-medoid* is a classical partitioning technique of clustering that clusters the data set of *n* objects into *k* clusters known *a priori*. It is more robust to noise and outliers as compared to because it may minimize a sum of pair-wise dissimilarities instead of a sum of squared Euclidean distances.

Algorithms

The most common realization of *k*-medoid clustering is the **Partitioning around Medoid (PAM)** algorithm and is as follows:

1. Initialize: randomly select (without replacement) *k* of the *n* data points as the medoid
2. Associate each data point to the closest medoid.
3. While the cost of the configuration decreases:

* For each medoid *m*, for each non-medoid data point *o*:
* Swap *m* and *o*, re-compute the cost (sum of distances of points to their medoid)
* If the total cost of the configuration increased in the previous step, undo the swap

***Example***

Cluster the following data set of ten objects into two clusters i.e. *k* = 2. Data Points are {(1,3), (4,2), (6,2), (3,5), (4,1)}

***Solution***

Let p1=(1,3) p2=(4,5) p3=(6,3) p4=(3,4) p5=(2,1)

***Initial step***

Let m1=p1=(1,3) and m2=p4=(3,4) are two initial medoid

***Iteration 1***

Calculate distance between clusters centers and each data point

Thus after first iteration

*Cluster 1= {p1, p5}*

*Cluster 2={p2,p3,p4}*

Total Cost= {++}=10

***Iteration 2***

Lets swap m1 and p2

Now m1=p2= (4,5) and m2=p4=(3,4) are two medoid

Let p1=(1,3) p2=(4,5) p3=(6,3) p4=(3,4) p5=(2,1)

Calculate distance between new cluster centers and each data points

Thus after second iteration

*Cluster 1= {p2,p3}*

*Cluster 2={p1,p4,p5}*

Total Cost= {++}=3+3+4=10

Same as previous, thus no undo swap

Repeat this process till cost of configuration do not decrease