CS 3481 Fundamentals of Data Science

WANG Ruochen

Table of contents

1. DATA

1.1. General

A data[1] set can often be viewed as a **collection** of data objects.

Data objects are described by a number of **attributes**[2] that capture the basic characteristics of an object.

1.2. Attributes

There are four types of attributes:

- Nominal

  eye color, gender.

- Ordinal

  grades.

- Interval

  calendar dates.

- Ratio

  length.

Another way to distinguish between attributes is by the number of values they can take:

- Discrete

  Has a finite or countably infinite set of values.

- Continuous

  real numbers.

1.3. Types of data sets

Four types of data sets:

- Record data
- Transaction or market basket data
- Data matrix
- Sparse data matrix

1. also named as record, point, vector, pattern, event, case, sample, observation or entity.
2. also named as variable, characteristic, field, feature or dimension

1.4. Data quality

- Precision

  Closeness of repeated measurements to one another.

  Measured by the standard deviation of a set of value.
- Bias

  A systematic variation of measurements.

  Measured by the difference between .

  - The mean of the set of values.

  - The known value of the quantity being measured.
- Noise

  Random component of a measurement error.
- Outliers

1.4.1. Missing values

Strategies for dealing with missing data:

- Eliminate data objects
- Estimate missing values

  if attribute is discrete, then the most commonly occurred one.

  if attribute is continuous, then average.

1.5. Data preprocessing

Five techniques for data preprocessing:

- Aggregation
- Sampling
- Dimensionality reduction
- Discretization and binarization
- Normalization

1.5.1. Aggregation

motivations for aggregation:

- The smaller dataset after aggregation require less memory and processing time
- Aggregation can also provide a high-level view of data
- Aggregate quantities, such as averages or totals, have less variability than the individual objects.

Disadvantages: potential loss of interesting details.

1.5.2. Sampling

Advantages:

- reduce the data size to a point where a better, but more computationally expensive algorithm can be used.

The simplest type of sampling is uniform random sampling. There is an equal probability of selecting any particular item.
Usually do *Sampling without replacement* instead of *Sampling with replacement*[3].

1.5.3. Dimensionality reduction

It is usually more difficult to analyze high-dimensional data (**curse of dimensionality**)
The **curse of dimensionality** refers to the phenomenon that many types of data analysis become significantly harder as the number of dimensions increases.

Advantages:

- eliminate irrelevant features and reduce noise.
- lead to a more understandable model which involves fewer attributes.
- allow the data to be more easily visualized.
- The amount of time and memory required for processing the dasta is reduced.

Two techniques for dimensionality reduction:

- Feature transformation
- Feature subset selection

Feature transformation
Principal Component Analysis(PCA).

3. Faster

- Linear combinations of the original attributes.
- Capture the maximum amount of variation in the data

Feature subset selection
This approach is effective if redundant and irrelevant features are present.

Three standard approaches to feature selection

- Embedded approaches
- Filter approaches
- Wrapper approaches

1.5.4. Discretization and binarization

Transform a continuous attribute into a discrete attribute (**discretization**)
Transform into one or more binary attributes(**binarization**)
Discretization
Two subtasks:

- Decide how many possible discrete values to have
- Determining how to map the values of the continuous attribute to these discrete values

1.5.5. Normalization

The goal of normalization or standardization is to make an entire set of values have a particular property.

Normalization is necessary to avoid the case where a variable with large values dominates the result of the calculation.

Ex 1. :

$$x' = \frac{x - \bar{x}}{\sigma_x}$$

Ex 2. :

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

1.6. Similarity and dissimilarity

1.6.1. Nominal

Dissimilarity is defined as :

0 if the attribute values match.

1 otherwise.

1.6.2. Ordinal

The values of the ordinal attribute are often mapped to successive integers. The dissimilarity can be defined by taking the absolute difference between these integers.

1.6.3. Interval/Ratio

For interval or ratio attributes, the natural measure of dissimilarity between two objects is the absolute difference of their values.

1.7. Distance

1.7.1. Euclidean distance

$$d(\boldsymbol{x},\boldsymbol{y}) = \sqrt{\sum_{u=1}^{n} (x_u - y_u)^2}$$

where n is the number of dimensions, $x_u$andy_u are, respectively, the u-th attributes of **x** and **y**.

The Euclidean distance measure is generalized by the Minkowshi distance metric as follows:

$$d(\boldsymbol{x},\boldsymbol{y}) = \left(\sum_{u=1}^{n} | x_u - y_y|^h \right)^{\frac{1}{h}}$$

Three most common examples of Minkowshi distances are:

- h=1: City block distance (L_1 norm)
- h=2: Euclidean distance(L_2 norm)
- h=∞: Supremum distance($L_{\max}$ or L_∞ norm), which is the maximum difference between any attribute of the objects.

1.7.2. General

A distance measure has some well-known properties:

- Positivity
- Symmetry
- Triangle inequality

1.8. Statistics

1.8.1. Relative frequency and the mode

$$\text{relative frequency}(a_s) = \frac{\text{number of objects with attribute value } a_s}{m}$$

The mode of a discrete attribute is the value that has the highest relative frequency.

1.8.2. Mean

$$\text{mean}(x) = \bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i$$

1.8.3. Range

$$\text{range}(x) = x_{\text{max}} - x_{\text{min}}$$

1.8.4. Variance

$$\text{variance}(x) = \sigma_x^2 = \frac{1}{m-1}\sum_{i=1}^{m} (x_i - \bar{x})^2$$

The standard deviation, which is the square root of the variance, is denoted as σ_x

1.8.5. Multivariate summary statistics

$$\text{covariance}(x_u, x_v) = \frac{1}{m-1}\sum_{i=1}^{m} (x_{\text{iu}} - \bar{x})(x_{\text{iv}} - \bar{x})$$

The correlation coefficient r_{uv}:

$$r_{\text{uv}} = \frac{\text{covariance}(x_u, x_v)}{\sigma_u \sigma_v}$$

The range of r_{uv} is from -1 to 1

1.9. Data Visualization

- Histogram
- Scatter plot

2. Introduction to Data Science

Six step:

1. Data Sources
2. Data Warehouse
3. Data Exploration
4. Data Mining
5. Data Presentation
6. Making Decisions

2.1. Knowledge Discovery in Databases(KDD)

Data mining is an integral part of knowledge discovery in databases(KDD)
KDD is the overall process of converting raw data into useful information.
Five Steps:

1. Data Cleaning and integration
2. Data Selection and transformation
3. Data mining
4. Pattern evaluation
5. Knowledge presentation

2.2. Challenges

- Scalability
- High dimensionality
- Heterogeneous and complex data

2.2.1. Scalability

- Employing special search strategies in search problems
- Implementing novel data structures to access individual records in an efficient manner
- Using sampling
- Developing parallel and distributed algorithms

2.2.2. High dimensionality

- Traditional data analysis techniques that were developed for low-dimensional data often do not work well for such high-dimensional data

- The Computational complexity increases rapidly as the dimensionality increases.

2.3. Data mining tasks

Four core data mining tasks:

- ○ Predictive modeling
  - Classification
  - Regression
- ○ Association analysis
- ○ Cluster analysis
- ○ Anomaly detection

3. Decision Tree

Classification is the task of assigning objects to one of several predefined categories.

The input data for a classification task is a collection of records. Each record, also know as an instance or example, is characterized by a tuple $(\boldsymbol{x}, y)$

- **x** is the attribute set
- y is the class label, also known as category or target attribute.

A classification model is useful for the following purposes:

- Descriptive modeling
- Predictive modeling

The model generated by a leaning algorithm should

- Fit the input data well
- Correctly predict the class labels of records it has never seen before.

3.1. Confusion matrix

Each entry a_{ij} in this table denotes the number of records from class i predicted to be of class j.

		Predicted Class	
		Class=1	Class=0
Actual Class	Class=1	a_{11}	a_{10}
	Class=0	a_{01}	a_{00}

Table 1.

$$\text{Accuracy} = \frac{a_{11} + a_{00}}{a_{11} + a_{10} + a_{01} + a_{00}}$$

$$\text{Error Rate} = \frac{a_{10} + a_{01}}{a_{11} + a_{10} + a_{01} + a_{00}}$$

3.2. Decision Tree

Decision Tree has three types of nodes:

- A root node that has no incoming edges.
- Internal nodes, each of which has exactly one incoming edge and a number of outgoing edges.

- Leaf or terminal nodes, each of which has exactly one incoming edge and no outgoing edges.

In decision tree, each leaf node is assigned a class label. The non-terminal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics.

3.2.1. Decision tree construction

Usually employ a greedy strategy that makes a serise of locally optimal decisions about which attribute to use for partitioning the data.

Algorithm 1

If all the records in U_s belong to the same class c_k, then s is a leaf node labeled as c_k.
If U_s contains records that belong to more than one class:

- An attribute test condition is selected to partition the records into smaller subsets
- A child node is created for each outcome of the test condition.
- The records in U_s are distributed to the children based on the outcomes.

The algorithm is then recursively applied to each child node.

Decision trees that are too large are susceptible to a phenomenon known as overfitting.

A tree pruning step can be performed to reduce the size of the decision tree.

Pruning helps by trimming the tree branches in a way that improves the generalization error.

3.2.2. Attribute test

Binary attributes

The test condition for a binary attribute generates two possible outcomes.

Nominal attributes

A nominal attribute can produce binary or multi-way splits.

There are 2^{S-1} ways of creating a binary partition of S attribute values.

For a multi-way split, the number of outcomes depends on the number of distinct values for the corresponding attribute.

Ordinal attributes

Ordinal attributes can also produce binary or multi-way splits.

Ordinal attributes can be grouped as long as the grouping does not violate the order property of the attribute values.

Continuous attributes
The test condition can be expressed as a comparison test $x \leqslant T$ or $x > T$.

3.3. Information Theory

The information content of a message depends on

- The size of this message set
- The frequency with which each possible message occurs.

The amount of information in a message with occurence probability p is defined as $-\log_2 p$.
Suppose we are given

- A set of messages, $C = \{c_1, c_2, \ldots, c_k\}$
- The occurence probability $p(c_k)$ of each c_k

We define the entropy I as the expected information content of a message in C:

$$I = -\sum_{k=1}^{K} p(c_k) \log_2 p(c_k)$$

The entropy is measured in bits.

If we select attribute P, with S values, this will parition U into the subset $\{U_1, U_2, \ldots, U_S\}$. The average degree of uncertainty after selecting P is:

$$\bar{I}(P) = \sum_{s=1}^{S} \frac{|U_s|}{|U|} I(U_S)$$

The information gain associated with attribute P is computed as follows:

$$\text{gain}(P) = I(U) - \bar{I}(P)$$

3.4. Impurity measures

3.4.1. Gini index

$$G = 1 - \sum_{k=1}^{K} p(c_k)^2$$

3.4.2. Classification error rate

$$E = 1 - \max_k p(c_k)$$

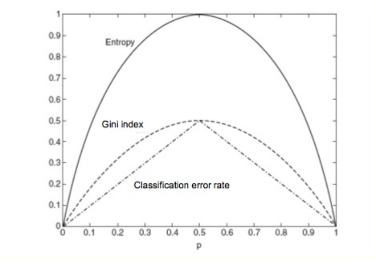

Figure 1.

3.5. Gain ratio

$$\text{Gain Ratio} = \frac{\text{Gain}(P)}{\text{Split Info}}$$

where

$$\text{Split Info} = -\sum_{s=1}^{S} \frac{|U_s|}{|U|} \log_2 \frac{|U_s|}{|U|}$$

3.6. Oblique decision tree

Oblique decision tree allows test conditions that involve more than one attribute.

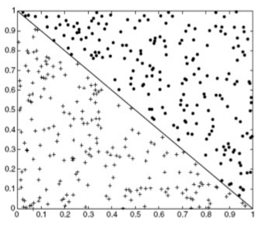

Figure 2.

4. CLASSIFICATION ERROR

The errors committed by a classification model are generally divided into two types:

- Training errors
- Generalization errors

Training error is also known as resubstitution error or apparent error.

Generalization error is the expected error of the model on previously unseen records.

A good classification model should :

- Fit the training data well. (low training error)
- Accurately classify records it has never seen before. (low generalization error)

Both error rates are large when the size of the deicision tree is very small[4].

Underfitting occurs because the model cannot learn the true structure of the data.

When the tree becomes too large, [5]

- The training error rate continues to decrease.
- However, the test error rate begins to increase.

4.1. Overfitting

The performance of the model on the training set does not generalize well to the test examples.

4.2. Generalization error estimation

Approaches to estimate the generalization error:

- Resubstitution estimate
- Estimates incorporating model complexity
- Using a validation set

4.2.1. Resubstitution estimate

The resubstitution estimate approach assumes that the training set is a good representation fo the overall data.

4. This is know as model underfitting.

5. This is know as model overfitting.

4.2.2. Estimates incorporating model complexity

The error rate e_c of the decision tree can be estimated as follows:

$$e_c = \frac{\sum_{l=1}^{L} [e(n_l) + \zeta(n_l)]}{\sum_{l=1}^{L} m(n_l)}$$

- L be the number of leaf nodes.
- n_l be the l-th leaf nodes.
- $m(n_l)$ be the number of training records classified by node n_l
- $e(n_l)$ be the number of misclassified records by node n_l
- $\zeta(n_l)$ be a penalty term associated with the node n_l.

4.2.3. Using a validation set

This approach provides a better way for estimating how well the model performs on previously unseen records.
However, less data are available for training.

4.3. Handling overfitting in decision tree

There are two approaches for avoiding model overfitting in decision tree:

- Pre-pruning
- Post-pruning

4.3.1. Pre-pruning

Advantage: It avoids generating overly complex sub-trees that overfit the training data.
However, it is difficult to choose the right threshold for the change in impurity measure.

4.3.2. Post-pruning

In this approach, the decision tree is initially grown to its maximum size. This is followed by a tree pruning step, which trims the fully grown tree.
Advantages:

- Post-pruning tends to give better results than pre-pruning because it makes pruning decisions based on a fully grown tree.
- Pre-pruning can suffer from premature termination of the tree growing process.

Disadvantage:

- The additional computations for growing the full tree may be wasted when some of the sub-trees are pruned.

4.4. Classifier evaluation

4.4.1. Hold-out method

Limitation:

1. Fewer examples are available for training
2. Model may be highly dependent on the composition of the training and test sets.

4.4.2. Cross validation

- The k-fold cross validation method generalizes this approach by segmenting the data into k equal-sized partitions.
- During each run
  - One of the partitions is chosen for testing.
  - The rest of them are used for training.
- This procedure is repeated k times so that each partition is used for testing exactly once.
- The estimated error is obtained by averaging the errors on the test sets for all k runs.

5. Nearest Neighbor and Bayesian Classifiers

5.1. Nearest Neighbor Classifier

When the nearest neighbors have more than one label, the data point is assigned to the majority class of its neighbors.

- If k is too small, the presence of noise in the training data may affect the generalization performance.
- If k is too large, the classifier may misclassify the test instance, since the nearest neighbors may now include points that are far away from the instance.

5.2. Bayesian Classifier

Bayesian classifiers are required to model the probabilistic relationships between the attributes and the class variable.

5.2.1. Bayes Theorem

Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be a pair of random variables.
The joint and conditional probabilities for X and Y are related in the following way:

$$P(X,Y) = P(Y|X)P(X) = P(X|Y)P(Y)$$

Rearranging the previous expression results in the Bayes theorem

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

5.2.2. Bayes Theorem for Classification

Bayes theorem can be applied for classification by interpreting the attributes $\boldsymbol{X}$ and the class variable $\boldsymbol{Y}$ as random variables.
Their probabilistic relationship can be captured using $P(Y|\boldsymbol{X})$.
This conditional probability is known as the posterior probability for Y.
On the other hand, the probability P(Y) is known as the prior probability for Y.

The Bayes theorem for classification is expressed in the following form:

$$P(Y|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|Y)P(Y)}{P(\boldsymbol{X})}$$

where $\boldsymbol{X}$ denotes multiple attributes.

The posterior probability $P(Y|\boldsymbol{X})$ is expressed in term of:

- The prior probability P(Y)
- The class-conditional probability $P(\boldsymbol{X}|Y)$
- The evidence $P(\boldsymbol{X})$

When comparing the posterior probabilities for different Y values, $P(\boldsymbol{X})$ is always constant and can be ignored.

The prior probaility can be estimated from the training set by computing the training records that belong to each class. To estimate the class-conditional probabilities $P(\boldsymbol{X}|Y)$, we consider the naive Bayes classification approach.

5.2.3. Naive Bayes Classifier

A naive Bayes classifier estimates the class-conditional probabilities by assuming that the attribhutes are conditionally independent, given the class label c.

The conditional independence assumption can be expressed as follows:

$$P(\boldsymbol{X}|Y=c)=\prod_{u=1}^{n}P(X_u|Y=c)$$

where $\boldsymbol{X}=\{X_1,X_2,...,X_n\}$ consists of n attributes.

To classify a test record $\boldsymbol{X}$, the naive Bayes classifier computes the posterior probability for each class Y:

$$P(Y|\boldsymbol{X})=\frac{\left[\prod_{u=1}^{n}P(X_u|Y)\right]P(Y)}{P(\boldsymbol{X})}$$

Since $P(\boldsymbol{X})$ is fixed for every Y, it is sufficient to choose the class that maximizes the numerator term.

5.2.4. Categorical Attributes

For a categorical attribute X_u, the conditional probability $P(X_u=x_u|Y=c)$ can be easily estimated. It is just the fraction of training instances in class c that takes on a particular attribute value x_u.

5.2.5. Continuous Attributes

There are two ways to estimate the class-conditional probabilities for continuous attributes in naive Bayes classifiers:

- Discretize each continuous attribute and then replace the continuous attribute value with its corresponding discrete interval.
- Assume a certain form of probability distribution for the continuous attribute.

A Gaussian distribution is usually chosen to represent the class-conditional probability for continuous attributes.

For each class c_k, the class conditional probability for attribute X_u is:

$$P(X_u = x_u | Y = c_k) \approx \frac{\varepsilon}{\sqrt{2\pi}\sigma_{u,k}} e^{\left[-\frac{\left(xu - \bar{x}_{u,k}\right)^2}{2\sigma^2_{u,k}}\right]}$$

The distribution is characterized by two parameters:

- Mean $\bar{x}$
- Variance σ

ϵ is a small constant.

- $\bar{x}_{u,k}$ is the sample mean of X_u for all training records belonging to the class c_k
- $\sigma^2_{u,k}$ is the sample variance of X_u of such training records.

6. Cluster Analysis

6.1. General

6.1.1. Main objective

- The objects within a group be similar to one another
- They are different from the object in the other groups

6.1.2. Different types of clusterings

- Partitional vs. hierarchical
- Exclusive vs. fuzzy
- Complete vs. partial

6.1.3. Partitional vs. hierarchical

A partitional clustering is a division of the set of data objects into subsets(clusters).

A hierarchical clustering is a set of nested clusters that are organized as a tree.

6.1.4. Exclusive vs. fuzzy

In an exclusive clustering, each object is assigned to a single cluster.

In a fuzzy clustering, every object belongs to every cluster with a membership weight that is between:

- 0 (absolutely does not belong)
- 1 (absolutely belongs)

This approach is useful for avoiding the arbitrariness of assigning an object to only one cluster when it is close to several.

A fuzzy clustering can be converted to an exclusive clustering by assigning each object to the cluster in which its membership value is the highest.

6.1.5. Complete vs. partial

A complete clustering assigns every object to a cluster.

A partial clustering does not asssign every object to a cluster.

The motivation of partial clustering is that some objects in a data set may not belong to wekk-defined groups. Instead, they may represent noise or outliers.

6.2. K-means

K-means is a prototype-based clustering technique which creates a one-level paritioning of the data objects.

```
Algorithm 2
1. Select K point as inital centroids
2. Repeat:
   a. Form K clusters by assigning each point to its cloest centroid.
   b. Re-compute the centroid of each cluster.
3. Until centroids do not change.
```

6.2.1. Distance measure

Euclidean(L_2) distance is often used for this purpose.
The goal of clustering is typicaly expressed by an objective function.
For our objective function, which measures the quality of a clustering solution, we can use the sum of the squared error(SSE).

We calculate the Euclidean distance of each data point to its associated centroid. We then compute the total sum of the squared distances, which is also known as the sum of the squared error(SSE). A small value of SSE means that the prototypes (centroids) of this clustering are good representations of the points in their clusters.

The SSE is deined as follows:

$$SSE = \sum_{i=1}^{K} \sum_{\boldsymbol{x} \in C_i} d(\boldsymbol{x}, \boldsymbol{c}_i)^2$$

In this equation:

- $\boldsymbol{x}$ is a data object
- C_i is the i-th cluster
- $\boldsymbol{c}_i$ is the centroid of cluster C_i
- d is the Euclidean(L_2) distance between two objects.

It can be shown that the mean of the data points in the cluster minimizes the SSE of the cluster. The centroid (mean) of the i-th cluster is defined as :

$$\boldsymbol{c}_i = \frac{1}{m_i} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$$

In this equation, m_i is the number of objects in the i-th cluster.

Steps 2a and 2b of the K-means algorihm (algorithm 2) attempt to minimize the SSE. Step 2a forms clusters by assigning each point to its nearest centroid, which minimizes the SSE for the given set of centroids. Step 2b recomputes the centroids so as to further minimize the SSE.

6.2.2. Choosing initial centroids

A common approach is to choose the inital centroids randomly. However, randomly selected initial centroids may be poor choices.

One technique is to perform multiple runs. Each run uses a different set of initial centroids. We then choose the set of clusters with the minimum SSE.

6.2.3. Outliers

To identify the outliers, we can keep track of the contribution of each point to the SSE. We then eliminate those points with unusually high contributions to the SSE.

6.2.4. Post-processing

Two post-processing strategies that decrease the SSE by increasing the number of clusters are:

- Split a cluster
- Introduce a new cluster centroid

Two post-processing strategies that decrease the number of clusters, while trying to minimize the increase in SSE, are :

- Disperse a cluster
- Merge two clusters

6.2.5. Bisecting K-means

Bisecting K-means algorithm is an extension of the basic K-means algorithm.

Algorithm 3
1. To obtain K clusters, split the set of all points into two cluster. 2. Select one of these clusters to split. 3. Continue the process until K clusters have been produced.

We can choose the one which, when split, leads to the largest decrease in SSE after updating the centroids.

We often refine the resulting clusters by using their centroids as the initial centroids for the basic K-means algorithm.

6.2.6. Limitations of K-means

The algorithm has difficulty in detecting clusters with non-spherical shapes or widely different sizes or densities.

This because K-means is designed to look for globular clusters of similar sizes and densities, or clusters that are well separated.

6.3. Hierarchical clustering

A hierarchical clustering is a set of nested clusters that are organized as a tree. There are two basic approaches for generating a hierarchical clustering:

- Agglomerative(凝结)
- Divisive

In agglomerative hierarchical clustering, we start with the points as individual clusters. At each step, we merge the closest pair of clusters. This requires defining a **notion of cluster distance**.

In divisive hierarchical clustering, we start with one, all-inclusive cluster. At each step, we split a cluster. This process continues until only singleton clusters of individual points remain. In this case, we need to decide:1. which cluster to split at each step 2. how to do the splitting.

A hierarchical clustering is often displayed graphically using a tree-like diagram called the **dendrogram**.

The dendrogram displays:

- the cluster-subcluster relationships
- the order in which the clusters are merged (agglomerative) or split(divisive)

The basic agglomerative hierarchical clustering algorithm is summerized as follows:

- Compute the distance matrix
- Repeat
  - Merge the cloest two clusters
  - Update the distance matrix to reflect the distance between the new cluster and the original clusters.
- Until only one cluster remains

Different definitions of cluster distance leads to different versions of hierarchical clustering:

- Single link or MIN
- Complete link or MAX
- Group average

6.3.1. Single link

The distance between two clusters is defined as the **minimum** of the distance between any two points, with one of the points in the first cluster, and the other point in the second cluster.

This technique is good at handling clusters with non-elliptical shapes.

6.3.2. Complete link

The distance between two clusters is defined as the **maximum** of the distance between any two points, with one of the points in the first cluster, and the other point in the second cluster.

Complete link tends to produce clusters with globular shapes.

6.3.3. Group average

The distance between two clusters is defined as the **average distance** across all pairs of points, with one point in each pair belonging to the first cluster, and the other point of the pair belonging to the second cluster.

This is an intermediate approach between the single and compelte link approaches.

$$d(C_i, C_j) = \frac{\sum_{\boldsymbol{x} \in C_i, \boldsymbol{y} \in C_j} d(\boldsymbol{x}, \boldsymbol{y})}{m_i m_j}$$

6.3.4. Key issues

Hierarchical clustering is effective when the underlying application requires the creation of a multi-level structure.

However, they are expensive in terms of their computational and storage requirements.

In addition, once a decision is made to merge two clusters, it cannot be undone at a later time.

6.3.5. DBSCAN

Density-based clustering locates regions of high density that are separated from one another by regions of low density.

DBSCAN is a simple and effective density-based clustering algorithm.

In DBSCAN, we need to estimate the **density** for a particular point in the data set.

This is performed by counting the number of points within or at a specified radius, Eps, of that point. The count includes the point itself.

A point is classified into:

- In the interior of a dense region (a core point)
- At the edge of a dense region (a border point)
- In a sparsely occupied region (a noise point)

Core points are in the interior of a density-based cluster.

7. Association analysis

7.1. Market basket transactions

Each row in the table coresponds to a transaction, which contains:

- A unique identifier labeled TID
- A set of items bought by a given customer.

Association analysis is useful for discovering interesting relationship hidden in large data sets.

The uncovered relationships can be represented in the form of association rules or set of frequent items.

There are two key issues that need to be addressed when applying association analysis to market basket data.

- First, discovering patterns from a large transaction data set can be computationally expensive.
- Second, some of the discovered patterns are potentially spurious because they may happen simply be chance.

Market basket data can be represented in a binary format.

The presence of an item in a treansaction is often considered more important than its absence.

7.2. Itemset and support count

Let $I = \{i_1, i_2, ..., i_d\}$ be the set of all items in a market basket data.

Let $T = \{t_1, t_2, ..., j_N\}$ be the set of all transactions.

Each transaction t_i contains a subset of items chosen from I.

A collection of zero or items is called itemset. If an itemset contains k items, it is called a k-itemset. The null (or empty) set is an itemset that does not contain any items.

A transaction t_j is said to contain an item set X if X is a subset of t_j.

The support count of an itemset is the number of transactions which contain that particular itemset.

The support count $\sigma(X)$ for an itemset X can be stated as :

$\sigma(X) = |\{t_i| X \subseteq t_i, t_i \in T\}|$

7.3. Association rule

An association rule is an implication expression of the form $X \rightarrow Z$.

X and Z are disjoint itemsets.

The strength of an association rule can be measured in terms of its support and confidence.

7.4. Support and confidence

Support determines how often a rule is applicable to a given data set.

Confidence determines how frequently items in Z appear in transactions that contain X.

$$\text{Support}, s(X \to Z) = \frac{\sigma(X \cup Z)}{N}$$

$$\text{Confidence}, c(X \to Z) = \frac{\sigma(X \cup Z)}{\sigma(X)}$$

A rule that has very low support occur simply by chance.

A low support rule is likely to be uninteresting from a business perspective.

For this reason, support is often used to eliminate uninteresting rules.

Confidence measures the reliability of the inference made by a rule.

For a given rule $X \to Z$, the higher the confidence, the more likely it is for Z to be present in transactions that contain X.

Confidence also provides an estimate of the conditional probability of Z given X.

The inference made by an association rule does not necessarily imply causality.

Instead, it suggests a strong **co-occurrence relationship** between items in the antecedent and consequent of the rule.

Causality, on the other hand, requires **knowledge about cause and effects** in the data.

7.5. Association rule mining

The association rule mining problem can be formally stated as follows:

- Given a set of transactions T, find all the rules having support $\geqslant$ minsup and confience $\geqslant$ minconf.
- minsup and minconf are the coresponding support and confidence threholds.

A brute force approach is prohibitively expensive.

A common strategy is to decompose the problem into two major subtasks:

- Frequent itemset generation
    - The objective of this step is to find all the itemsets that satisfy the minsup threhold.
    - These itemsets are called frequent itemsets.

- Rule generation
    - The objective of this step is to extract all the high-confidence rules from the frequent itemsets found in the previous step.
    - These rules are called strong rules.

7.6. Frequent itemset generation

There are several ways to reduce the computational complexity of frequent itemset generation:

- Reduce the number of candidate itemsets
    - The Apriori principle is an effective way to eliminate some of the candidate itemsets without determining their support values
- Reduce the number of comparisons
    - We can reduce the number of comparisons by using more advanced data structures.

7.7. The Apriori principle

If an itemset if frequent, then all of its subsets must also be frequent.

Conversely, if an itemset is infrequent, then all of its supersets must be infrequent.

This strategy of trimming the search space based on the support measure is known as support-based pruning.

Such a pruning strategy is made possible by the anti-monotone property of the support measure.

7.7.1. Apriori algorithm

Apriori is the first algorithm that uses support-based pruning to control the exponential growth of candidate itemsets.

Let C_k denote the set of candidate k-itemsets.

Let F_k denote the set of frequent k-itemsets.

The algorithm initially makes a single pass over the data set to determine the support count of each item.

Upon completion of this step, the set of all frequent 1-itemsets, F_1, will be known.

Next, the algorithm will generate new candidate k-itemsets.

These itemsets are generated using the frequent (k-1)-itemsets found in the previous stage.

To determine the support counts of the candidates, the algorithm needs to make an additional pass over the data set.

After determining their support counts, the algorithm eliminates all candidate itemsets whose support counts are less than the threshold.

The algorithm terminates when there are no new frequent itemsets generated.

The frequent itemset generation part of the Apriori algorithm has two important characteristics.

- First, it is a level-wise algorithm
    - It traverses the itemset lattice one level at a time, from frequent 1-itemsets to the maximum size of frequent itemsets.
- Second, it employs a generate-and-test strategy for finding frequent itemsets.
    - At each iteration, new candidate itemsets are generated from the frequent itemsets found in the previous stage.
    - After a pruning process, the support count of each remaining candidate is then determined and tested against the threhold.

7.7.2. Candidate generation and pruning

Candidate itemsets are generated by performing the following two operations:

- Candidate generation
    - This operation generates new candidate k-itemsets based on the frequent (k-1)-itemsets found in the previous stage.
- Candidate pruning
    - This operation eliminates some of the candidate k-itemsets.

There are a number of requirements for an effective candidate generation procedure.

- First, it should avoid generating too many unvessary candidates
    - A candidate itemset is unnecessary if at least one of its subsets is infrequent.
    - Such a candidate is guaranteed to be infrequent according to the anti-monotone property of support.
- It must ensure that the candidate set is complete.
    - In other words, no frequent itemsets are left out by the candidate generation procedure.
    - To ensure completeness, the set of candidate itemsets must subsume the set of all frequent itemsets, i.e., $\forall k$: $F_k \subseteq C_k$
- It should not generate the same candidate itemset more than once.
    - Generation of duplicate candidates leads to wasted computations.

Candidate generation procedure used in the Apriori algorithm:

- Find all pairs of frequent (k-1)-itemsets where their first k-2 items are identical.

- Each pair is merged to form a candidate k-itemset.
- Let $A=\{a_1, a_2, \ldots, a_{k-1}\}$ and $B=\{b_1, b_2, \ldots, b_{k-1}\}$ be a pair of frequent (k-1)-itemsets.
- A and B are merged if they satisfy the following conditions:
  - $a_i = b_i$ (for $i = 1, 2, \ldots, k-2$) and $a_{k-1} \neq b_{k-1}$

Consider a candidate k-itemset, $X=\{i_1, i_2, \ldots, i_k\}$:

- The algorithm determines whether all of the subsets, $X - \{j_i\}, j = 1, 2, \ldots, k$, are frequent.
- If one of them is infrequent, then X is immediately pruned.
- This approach can effectively reduce the number of candidate itemsets considered during support counting.

We do not have to examine all k subsets of a given candidate itemset. This is because the subsets used to generate a candidate are know to be frequent. We only need to check the remaining subsets during candidate pruning.

7.7.3. Rule generation

Each frequent k-itemset Y can produce up to $2^k - 2$ association rules.

We ignore rules that have empty antecedents and consequents, e.g. $\varnothing \rightarrow Y$ or $Y \rightarrow \varnothing$.

An association rule can be extracted by partitioning the itemset Y into two non-empty subsets.

Specifically, the subsets X and Y-X should form a rule $X \rightarrow Y$ that satisfies the confidence threshold.

All such rules must have already met the support threshold since they are gnerated from a frequent itemset.

Suppose a rule $X \rightarrow Y - X$ does not satisfy the confidence threshold. Then any rule $X' \rightarrow Y - X'$, where X' is a subset of X, will not satisfy the confidence threshold.

The Apriori algorithm uses a level-wise approach for generating association rules. Each level corresponds to the number of items that belong to the rule consequent. Initially, all the high-confidence rules that have only one item in the rule consequent are extracted. These rules are then used to generate new candidate rules.

7.8. Compact representation of frequent itemsets

It is useful to identify a small representative set of frequent itemsets from which all other frequent itemsets can be derived.

Two representations are:

- Maximal frequent itemsets.

- Closed frequent itemsets.

7.8.1. Maximal frequent itemsets

A maximal frequent itemset is defined as a frequent itemset for which none of its immediate supersets are frequent.

Maximal frequent itemsets effectively provide a compact representation of frequent itemsets.

They form the smallest set of itemsets from which all frequent itemsets can be derived.

Maximal frequent itemsets do not contain the support information of their subsets. An additional pass over the data set is required to determine the support counts of the non-maximal frequent itemsets.

7.8.2. Closed frequent itemsets

An itemset X is closed if none of its immediate supersets has exactly the same support count as X.

Put another way, X is not closed if at least one of its immediate supersets has the same support count as X.

An itemset is a closed frequent itemset if

- It is closed
- Its support is greater than or equal to minsup.

We can use the closed frequent itemsets to determine the support counts for the non-closed frequent itemsets.

All maximal frequent itemsets are closed. This is because, by definition, the support count of a maximal frequent itemset cannot be the same as that of any of its immediate supersets.