Lecture 5

Classification, Clustering and Association Rules Mining

MSBA Python Bootcamp – July 2022

Facilitator: A/P TAN Wee Kek

Email: tanwk@comp.nus.edu.sg :: **Tel**: 6516 6731 :: **Office**: COM3-02-35

Learning Objectives

- At the end of this lecture, you should understand:
  - Appreciate the limitation of regression models.
  - Convert a regression problem to classification problem.
  - Understand how decision tree classifier works.
  - How to perform decision tree classification.
  - Understand how k-means clustering works.
  - How to perform k-means clustering.
  - Understand what is association rules mining.
  - How to perform association rules mining.

Limitation of Linear Regression Models

- Regression analysis is useful but suffers from an important limitation.
- In linear regression models, the numerical dependent variable must be continuous:
  - The dependent variable can take on any value, or at least close to continuous.
  - In some data analytics scenarios, the dependent variable may not be continuous.
  - In other scenarios, it may be unnecessary to make a point prediction.
- It is possible to convert a regression problem into a classification problem.

Parametric versus Non-parametric

- Linear regression is parametric:
  - Assumes that sample data comes from a population that can be adequately modelled by a probability distribution that has a fixed set of parameters.
  - Assumptions can greatly simplify the learning process, but can also limit what can be learned.
- **Parametric** ML algorithms:
  - Algorithms that simplify the function to a known form.
- **Non-parametric** ML algorithms:
  - Algorithms that do not make strong assumptions about the form of the mapping function.
  - Free to learn any functional form from the training data.

Parametric versus Non-parametric (cont.)

- Non-parametric ML methods are good when:
  - You have a lot of data and no prior knowledge.
  - You do not want to worry too much about choosing just the right features.
- Classification algorithms include both parametric and non-parametric:
  - Parametric – Logistic Regression, Linear Discriminant Analysis, Perceptron, Naive Bayes, Simple Neural Networks
  - Non-parametric – k-Nearest Neighbors, Decision Trees, Support Vector Machines

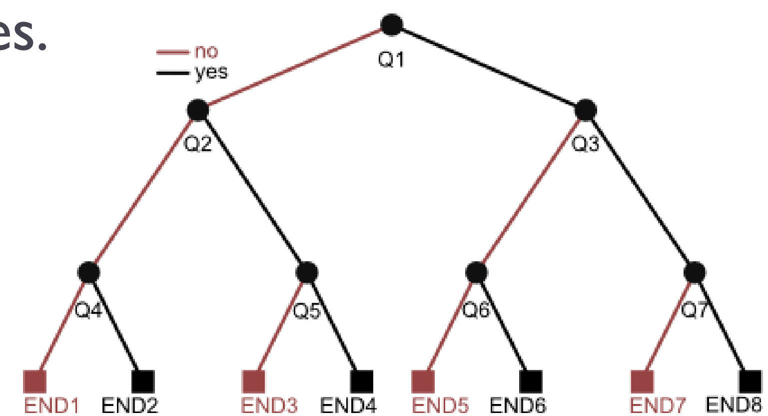
Data Mining Goes to Hollywood

- Data mining scenario – Predicting the box-office receipt (i.e., financial success) of a particular movie.
- Problem:
  - Traditional approach:
    - Frames it as a forecasting (or regression) problem.
    - Attempts to predict the point estimate of a movie's box-office receipt.
  - Sharda and Delen's (2006) approach:
    - Convert the regression problem into a multinomial classification problem.
    - Classify a movie based on its box-office receipts into one of nine categories, ranging from "flop" to "blockbuster".
    - Use variables representing different characteristics of a movie to train various classification models.

Classification with Decision Tree Classifier

Decision Trees

- The best known and most widely used learning methods in data mining applications.
- Reasons for its popularity include:
  - Conceptual simplicity.
  - Ease of usage.
  - Computational speed.
  - Robustness with respect to missing data and outliers.
  - Interpretability of the generated rules.



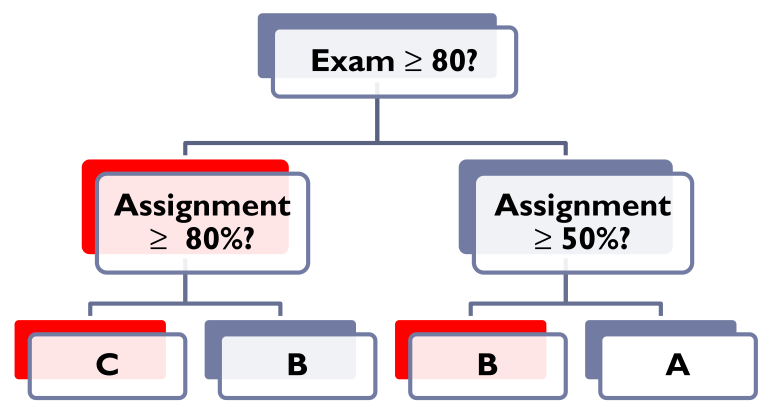
Decision Trees (cont.)

- The **development of a decision tree** involves <u>recursive, heuristic, top-down induction</u>:
  1. <u>Initialization phase</u> – All observations are placed in the root of the tree. The root is placed in the active node list L.
  2. If the list L is empty, stop the procedure. Otherwise, node $J \in L$ is selected, removed from the list and used as the node for analysis.
  3. The optimal rule to split the observations in J is then determined, based on an appropriate preset criterion:
     - If J does not need to be split, node J becomes a leaf, target class is assigned according to <u>majority</u> class of observations.
     - Otherwise, split node J, its children are added to the list.
     - Go to Step 2.

Components of Decision Trees

- <u>Components</u> of the top-down induction of decision trees:
  - **Splitting rules** – Optimal way to split a node (i.e., assigning observations to child nodes) and for creating child nodes.
  - **Stopping criteria** – If the node should be split or not. If not, this node becomes a leaf of the tree.
  - **Pruning criteria** – Avoid excessive growth of the tree (pre-pruning) during tree generation phase, and reduce the number of nodes after the tree has been generated (post-pruning).

A simple decision tree:
- Left branch for "No"
- Right branch for "Yes"



Example of a Decision Tree

- Given the dataset:

Observation #	Income	Credit Rating	Loan Risk
0	23	High	High
1	17	Low	High
2	43	Low	High
3	68	High	Low
4	32	Moderate	Low
5	20	High	High

- The task is to predict Loan-Risk.
- We will be using the <u>univariate binary splitting</u> approach.

Example of a Decision Tree (cont.)

- Given the data set D, we start building the tree by creating a root node.
- If this node is sufficiently "pure", then we stop.
- If we do stop building the tree at this step, we use the majority class to classify/predict.
- In this example, we classify all patterns as having Loan-Risk = "High".
- Correctly classify 4 out of 6 input samples to achieve classification accuracy of: $(4/6)\times 100\% = 66.67\%$
- This node is split according to impurity measures:
  - Gini Index (used by CART)
  - Entropy (used by ID3, C4.5, C5)

Loan-Risk = High
Acc = 66.67%

Using Gini Index

- CART (Classification and Regression Trees) uses the **Gini index** to measure the impurity of a dataset:
  - Gini index for the observations in node q is:

$$Gini(q) = 1 - \sum_{h=1}^{H} p_h^2$$

where

q is the node that contains Q examples from H classes

p_h is a relative frequency of class h in node q

  - In our dataset, there are 2 classes High and Low, $H = 2$.

$$p_{High} = \frac{4}{4+2} = \frac{2}{3} \qquad p_{Low} = \frac{2}{4+2} = \frac{1}{3}$$

$$Gini(q) = 1 - \left(\frac{2}{3} \times \frac{2}{3}\right) - \left(\frac{1}{3} \times \frac{1}{3}\right) = \frac{4}{9} = 0.4444$$

Using Gini Index (cont.)

- Should Income be used as the variable to split the root node?
- Income is a variable with continuous values.
- Sort the data according to Income values:

	Observation #	Income	Credit Rating	Loan Risk
	1	17	Low	High
	5	20	High	High
Split 1	0	23	High	High
Split 2	4	32	Moderate	Low
Split 3	2	43	Low	High
	3	68	High	Low

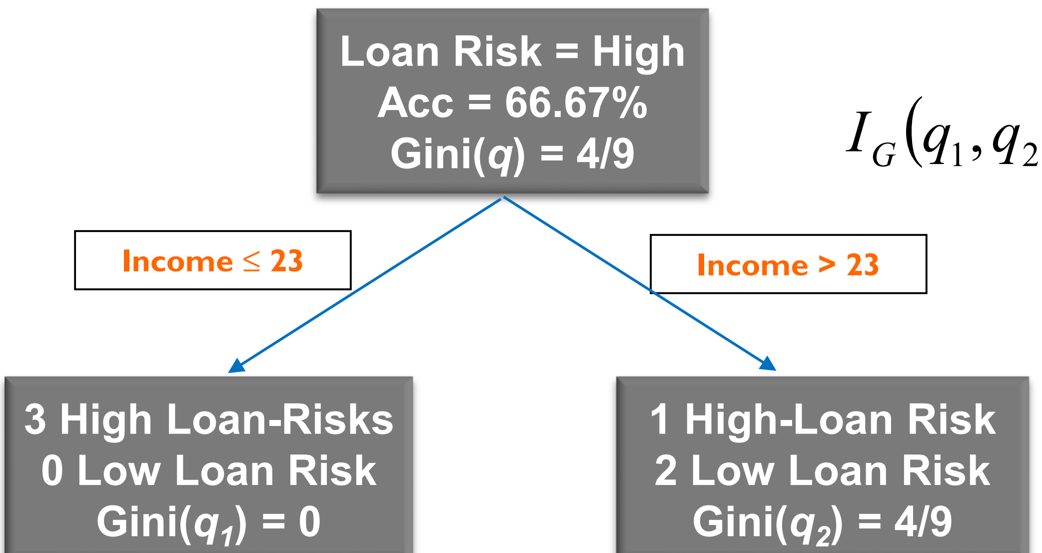
Using Gini Index (cont.)

- We consider 3 possible splits when there are changes in the value of Loan-Risk.
  - Case 1 – Split condition Income ≤ 23 versus Income > 23

Impurity after split:

$$I(q_1, q_2, \ldots q_k) = \sum_{k=1}^{K} \frac{Q_k}{Q} I(q_k)$$

$$I_G(q_1, q_2) = \left(\underbrace{\frac{3}{6} \times 0}_{I_G(q_1)}\right) + \left(\underbrace{\frac{3}{6} \times \frac{4}{9}}_{I_G(q_2)}\right) = \frac{2}{9} = 0.2222$$



Using Gini Index (cont.)

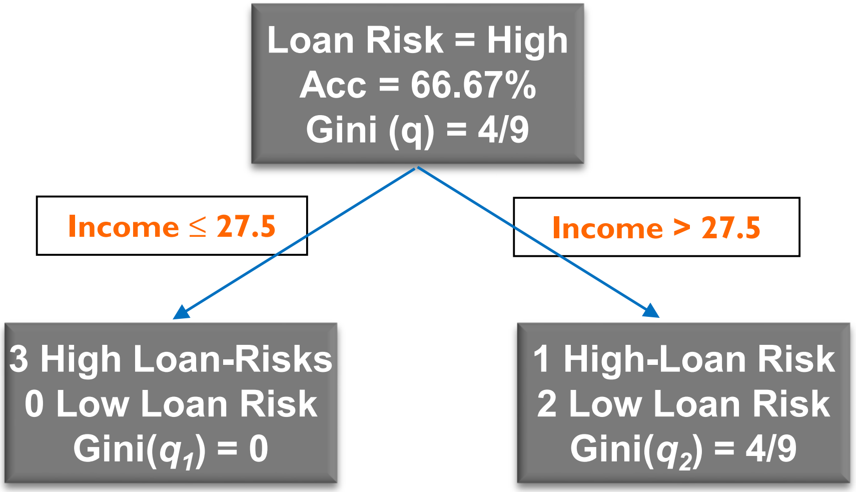
- Case 2 – Split condition Income ≤ 32 versus Income > 32:

$$I_G(q_1, q_2) = \left(\frac{4}{6} \times \frac{3}{8}\right) + \left(\frac{2}{6} \times \frac{1}{2}\right) = \frac{5}{12} = 0.41667$$

- Case 3 – Split condition Income ≤ 43 versus Income > 43:

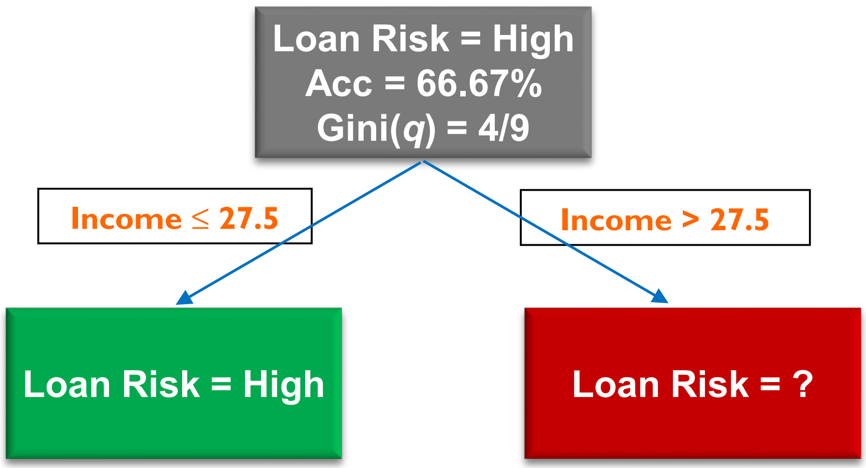
$$I_G(q_1, q_2) = \left(\frac{5}{6} \times \frac{8}{25}\right) + \left(\frac{1}{6} \times 0\right) = \frac{4}{15} = 0.26667$$

- Case 1 is the best.
- Instead of splitting between Income ≤ 23 versus Income > 23, the midpoint is selected as actual splitting point: (23 + 32)/2.



Using Gini Index (cont.)

- Apply the tree generating method recursively to nodes that are still not "pure".



- Develop a subtree by examining the variable Credit-Rating.
- Credit-Rating is a discrete variable with ordinal values, i.e., they can be ordered in a meaningful sequence.

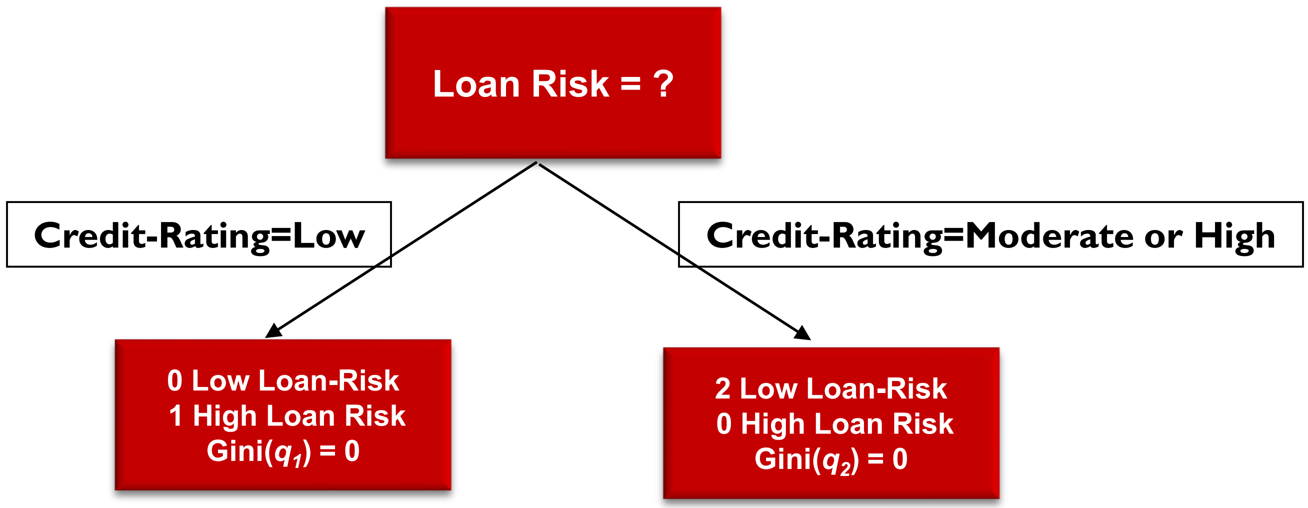
Using Gini Index (cont.)

- Possible values are $\{\text{Low}, \text{Moderate}, \text{High}\}$.
- Check for best split:
  - Case 1 – Low versus (Moderate or High)
  - Case 2 – (Low or Moderate) versus High
- Compute the Gini index for splitting the node:

Loan Risk = ?

Using Gini Index (cont.)

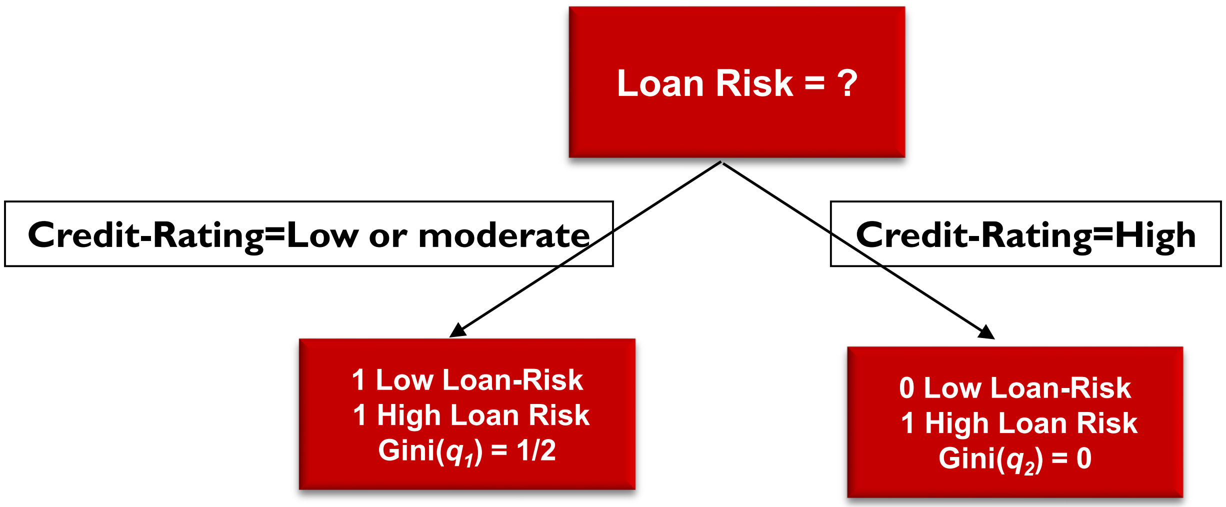
- Case 1 – Split Credit-Rating = Low versus Credit-Rating = Moderate or High:



$$I_G(q_1, q_2) = \left(\frac{1}{3} \times 0\right) + \left(\frac{2}{3} \times 0\right) = 0$$

Using Gini Index (cont.)

- Case 2 – Split Credit-Rating = Low or Moderate versus Credit-Rating = High:

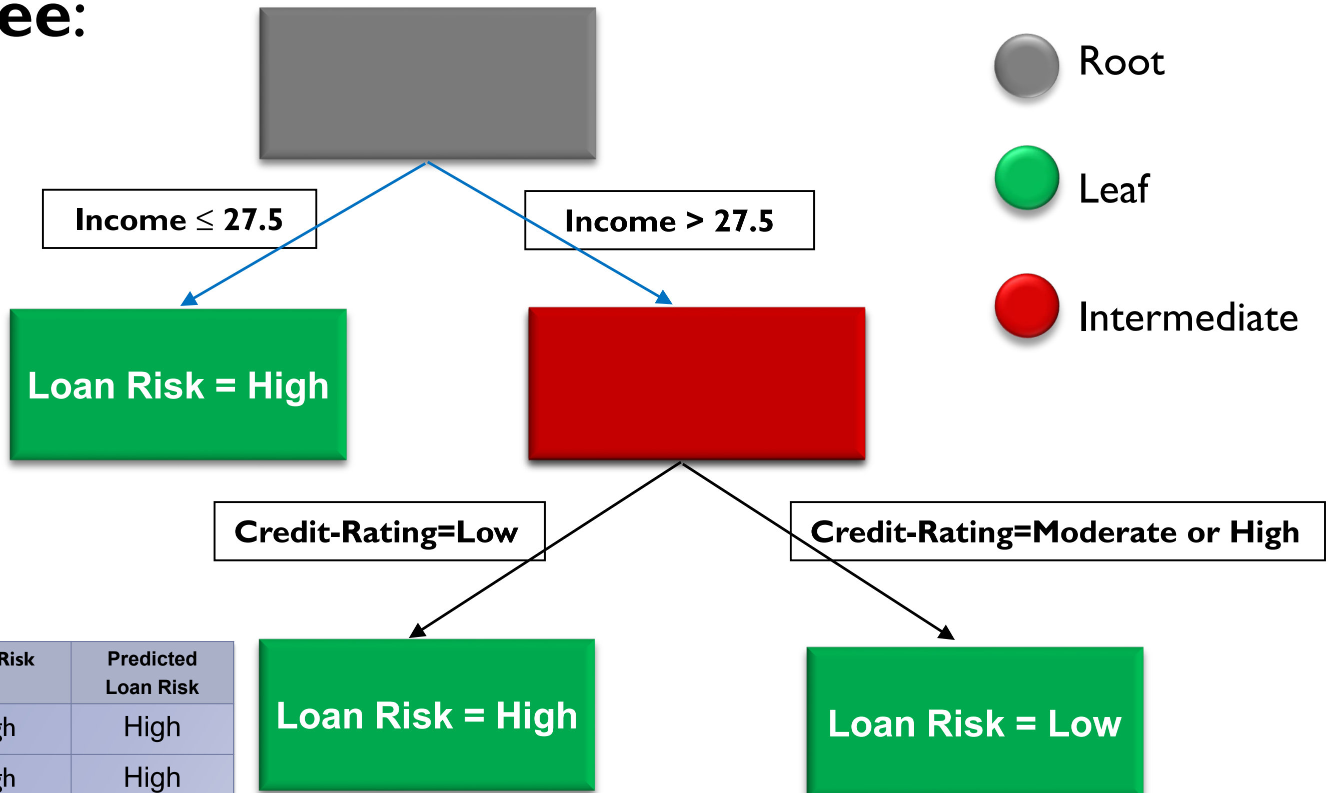


$$I_G(q_1, q_2) = \left(\frac{2}{3} \times \frac{1}{2}\right) + \left(\frac{1}{3} \times 0\right) = \frac{1}{3}$$

- Case 2 split is not as good as Case 1 split.

Using Gini Index (cont.)

- **Complete tree:**



Observation #	Income	Credit Rating	Loan Risk	Predicted Loan Risk
0	23	High	High	High
1	17	Low	High	High
2	43	Low	High	High
3	68	High	Low	Low
4	32	Moderate	Low	Low
5	20	High	High	High

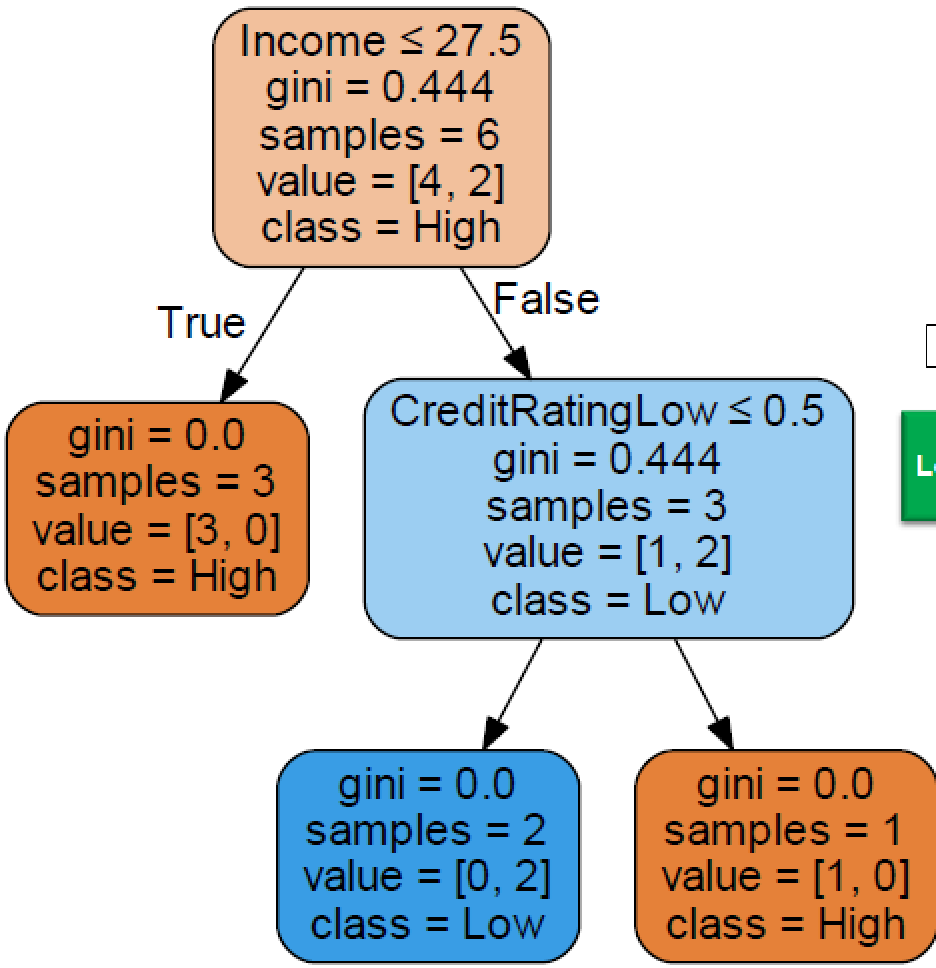
Using Gini Index (cont.)

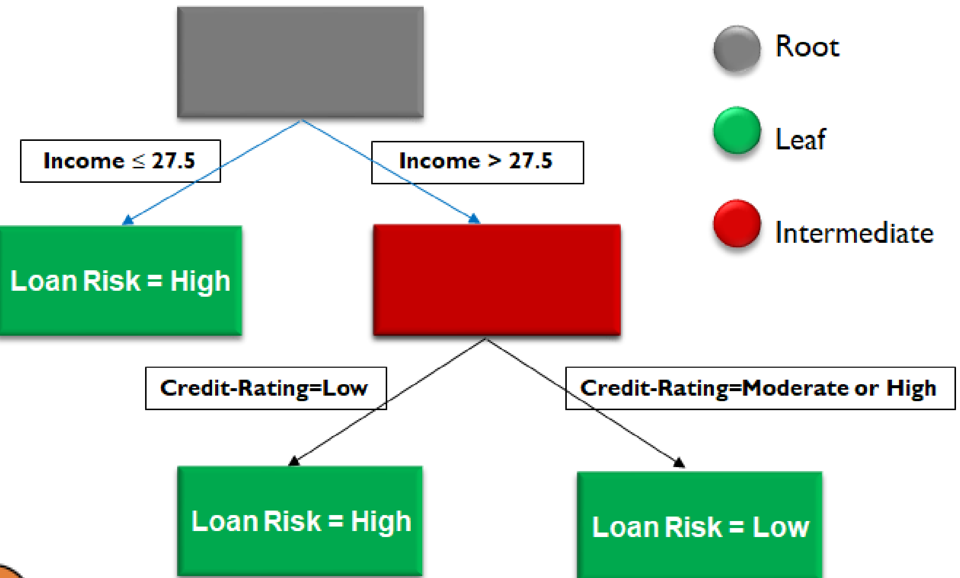
- The tree achieves 100% accuracy on the training data set.
- It may overfit the training data instances.
- Trees may be simplified by **pruning**:
  - Removing nodes or branches to improve the accuracy on the test samples.
- Tree growing could be terminated when the number of instances in the node is less than a pre-specified number.
- Notice we have built a binary tree where every non-leaf nodes have 2 branches.

Decision Tree in Scikit Lean

- We can perform decision tree classification using Scikit Learn's `tree.DecisionTreeClassifier`.
- However, this class cannot process categorical independent variables and thus we need to recode CreditRating:
  - Use one hot encoding or one-of-K scheme.
  - LoanRisk has three levels – Low, Moderate and High.
  - So we will create three binary variables – CreditRatingLow, CreditRatingModerate and CreditRatingHigh.
  - For each observation, only exactly one of these three variables will be set to 1.
- Refer to sample source file `src01` for the example.

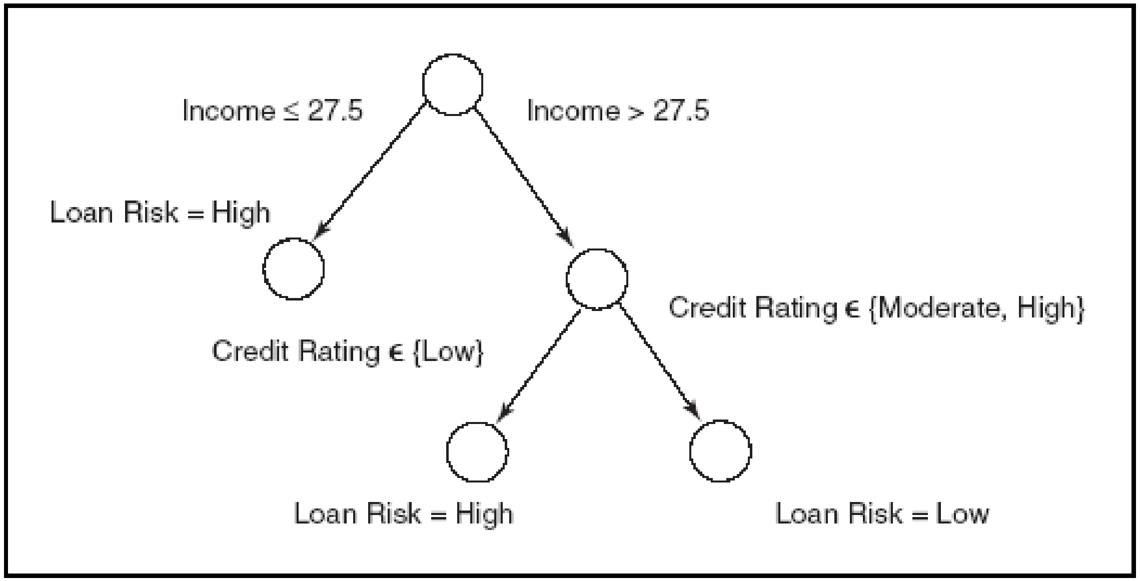
Decision Tree in Scikit Lean (cont.)





Classification Rule Generation

- Trace each path from the root node to a leaf node to generate a rule:



If Income ≤ 27.5, then Loan-Risk = High

Else if Income > 27.5 and Credit-Rating=Low, then Loan-Risk = High

Else if Income > 27.5 and Credit-Rating= Moderate or High, then Loan-Risk = Low

Practical Exercise: PE07-01

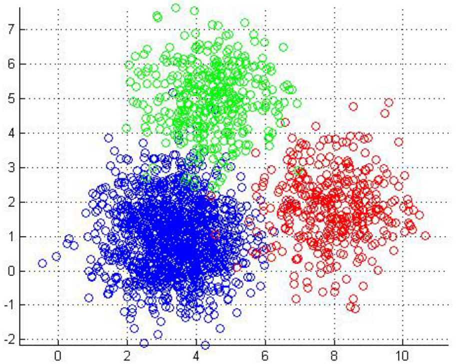
Clustering with K-Means

Overview of Clustering

- **Clusters** are homogeneous groups of observations.
- To measure similarity between pairs of observations, a distance metric must be defined.
- **Clustering** is an unsupervised learning process.
- Focus of our discussions will be on:
  - Features of clustering models.
  - A partition method: ***K*-means**.
  - Quality indicators for clustering methods.

Clustering Methods

- **Aim** – To subdivide the records of a dataset into homogeneous groups of observations called clusters.
- Observations in a cluster are similar to one another and are dissimilar from observations in other clusters.



- Purpose of clustering:
  - As a tool which could provide meaningful interpretation of the phenomenon of interest:
    - Example – Grouping consumers based on their purchase behavior may reveal the existence of a market niche.

Clustering Methods (cont.)

- As a preliminary phase of a data mining project that will be followed by other methodologies within each cluster:
  - Example:
    - Clustering is done before classification.
    - In retention analysis, distinct classification models may be developed for various clusters to improve the accuracy in spotting customers with high probability of churning.
- As a way to highlight outliers and identify an observation that might represent its own cluster.

Taxonomy of Clustering Methods

- Based on the logic used for deriving the clusters.
- **Partition methods**:
  - Develop a subdivision of the given dataset into a predetermined number *K* of non-empty subsets.
  - They are usually applied to small or medium sized data sets.
- **Hierarchical methods**:
  - Carry out multiple subdivisions into subsets.
  - Based on a tree structure and characterized by different homogeneity thresholds within each cluster and inhomogeneity threshold between distinct clusters.
  - No predetermined number of clusters is required.

Affinity Measures

- Clustering models are typically based on a measure of similarity between observations.
- The measure can typically be obtained by defining an appropriate notion of distance between each pair of observations.
- There are many popular metrics depending on the type of variables being analyzed.

Affinity Measures (cont.)

- Given a dataset $\mathbb{D}$ having m observations $\mathrm{X}_1, \mathrm{X}_2, \mathrm{X}_3, \ldots \mathrm{X}_m$ each described by *n*-dimensional variables, we compute the **distance** matrix D:

$$\mathrm{D} = \left[d_{ik} \right] = \begin{bmatrix} 0 & d_{12} & \cdots & d_{1,m-1} & d_{1m} \\ & 0 & \cdots & d_{2,m-1} & d_{2m} \\ & & \cdots & \vdots & \vdots \\ & & & 0 & d_{m-1,m} \\ & & & & 0 \end{bmatrix}$$

where d_{ik} is the distance between observations X_i and X_k.

$$d_{ik} = \mathrm{dist}(\mathrm{X}_i, \mathrm{X}_k) = \mathrm{dist}(\mathrm{X}_k, \mathrm{X}_i) \text{ for } i, k = 1, 2, \ldots, m$$

D is a symmetric $m \times m$ matrix with zero diagonal.

Affinity Measures (cont.)

- **Similarity measure** can be obtained by letting:

$$s_{ik} = \frac{1}{1 + d_{ik}} \quad \text{or} \quad s_{ik} = \frac{d_{\max} - d_{ik}}{d_{\max}}$$

where $d_{\max} = \max_{i,k} d_{ik}$ is the max value of D .

Affinity Measures for Numerical Variables

- If all n variables of the observations $\mathrm{X}_1, \mathrm{X}_2, \mathrm{X}_3, \ldots \mathrm{X}_m$ are numerical, the distance between X_i and X_k can be computed in four ways.
- **Euclidean distance** (or 2 norm):

$$\mathrm{dist}(\mathrm{X}_i, \mathrm{X}_k) = \sqrt{\sum_{j=1}^{n} (x_{ij} - x_{kj})^2} = \sqrt{(x_{i1} - x_{k1})^2 + (x_{i2} - x_{k2})^2 + \ldots + (x_{in} - x_{kn})^2}$$

- **Manhattan distance** (or 1 norm):

$$\mathrm{dist}(\mathrm{X}_i, \mathrm{X}_k) = \sum_{j=1}^{n} |x_{ij} - x_{kj}| = |x_{i1} - x_{k1}| + |x_{i2} - x_{k2}| + \ldots + |x_{in} - x_{kn}|$$

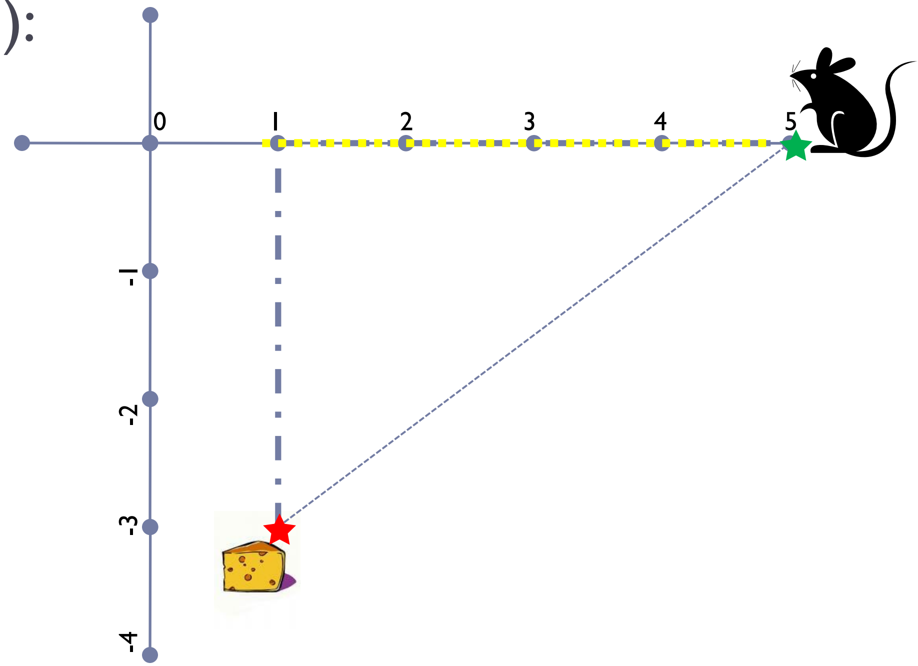
Affinity Measures for Numerical Variables (cont.)

- **Example:** $x_1 = (5,0)$ and $x_2 = (1,-3)$
  - Euclidean distance (or 2 norm):

$$\mathrm{dist}(x_1, x_2) = \sqrt{(5-1)^2 + (0-(-3))^2}$$
$$= \sqrt{16+9} = 5$$

  - Manhattan distance (or 1 norm):

$$\mathrm{dist}(x_1, x_2) = |5-1| + |0-(-3)|$$
$$= 4+3 = 7$$



Partition Methods

- Given a dataset $\mathbb{D}$, each represented by a vector in *n*-dimensional space, construct a collection of subsets $C = \{C_1, C_2, \ldots, C_K\}$ where $K \leq m$.
- K is the number of clusters and is generally predetermined.
- Clusters generated are usually exhaustive and mutually exclusive – Each observation belongs to only one cluster.
- Partition methods are iterative:
  - Assign m observations to the K clusters.
  - Then iteratively reallocate to improve overall quality of clusters.

Partition Methods (cont.)

- **Criteria for quality**:
  - Degree of homogeneity of observations in the same clusters.
  - Degree of heterogeneity with respect to observations in other clusters.
- The methods terminate when during the same iteration no reallocation occurs, i.e., clusters are stable.

K-means Algorithm

1. Initialize: choose K observations arbitrarily as the **centroids** of the clusters.
2. Assign each observation to a cluster with the nearest centroid.
3. If no observation is assigned to different cluster with respect to previous iteration, stop.
4. For each cluster, the new centroid is computed as the mean of the values belonging to that cluster. Go to Step 2.

K-means Algorithm (cont.)

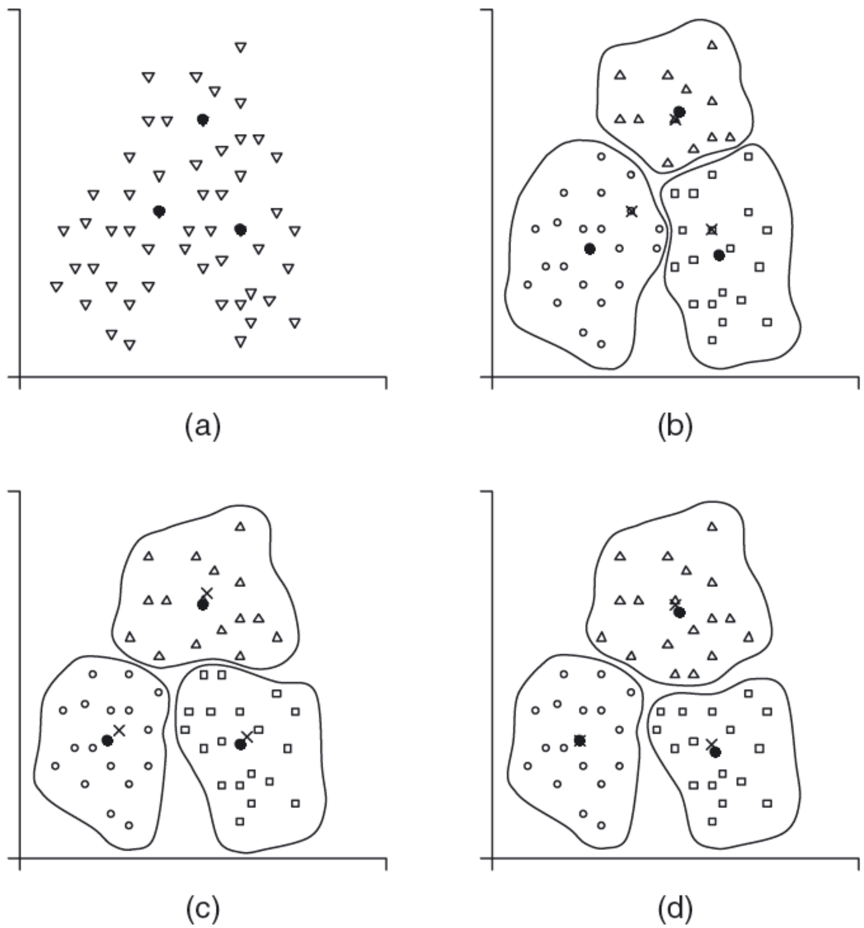


Figure 12.2 An example of application of the K-means algorithm

Source: Vercellis (2009), pp. 304

K-means Algorithm (cont.)

- Given a cluster C_h , $h = 1, 2, \ldots, K$, the **centroid** of the cluster is the point z_h having coordinates equal to the mean value of each variable in the observations belonging to that cluster:

$$z_{hj} = \frac{\sum_{\mathrm{X}_i \in C_h} x_{ij}}{\mathrm{card}\{C_h\}}$$

where $\mathrm{card}\{C_h\}$ is the number of observations in cluster C_h .

K-means Algorithm (cont.)

- Example – Suppose we have 2-dimensional data with the variables $\{\text{Weight}, \text{Height}\}$:
  - In Cluster 1, the observations are: $\{65,168\}, \{69,172\}$.
  - In Cluster 2, the observations are: $\{50,165\}, \{58,158\}, \{54,157\}$.
  - The centroids are:
    - Cluster 1:

$$z_1 = \{z_{11}, z_{12}\} = \left\{\frac{65+69}{2}, \frac{168+172}{2}\right\} = \{67,170\}$$

    - Cluster 2:

$$z_2 = \{z_{21}, z_{22}\} = \left\{\frac{50+58+54}{3}, \frac{165+158+157}{3}\right\} = \{54,160\}$$

Clustering Example – *K*-means

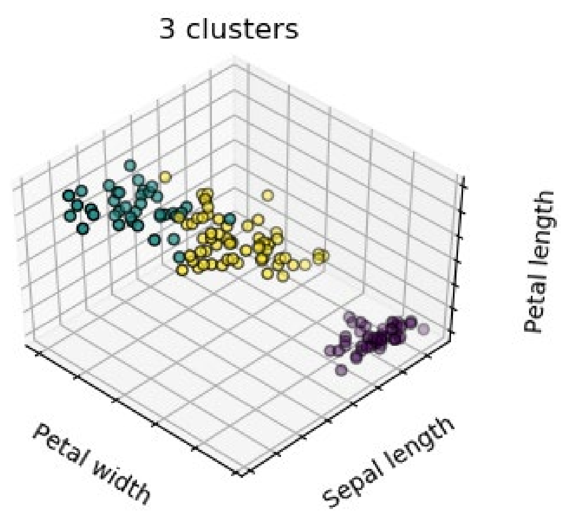
- Iris classification problem:
  - 3 classes – Setosa, Versicolor and Virginica.

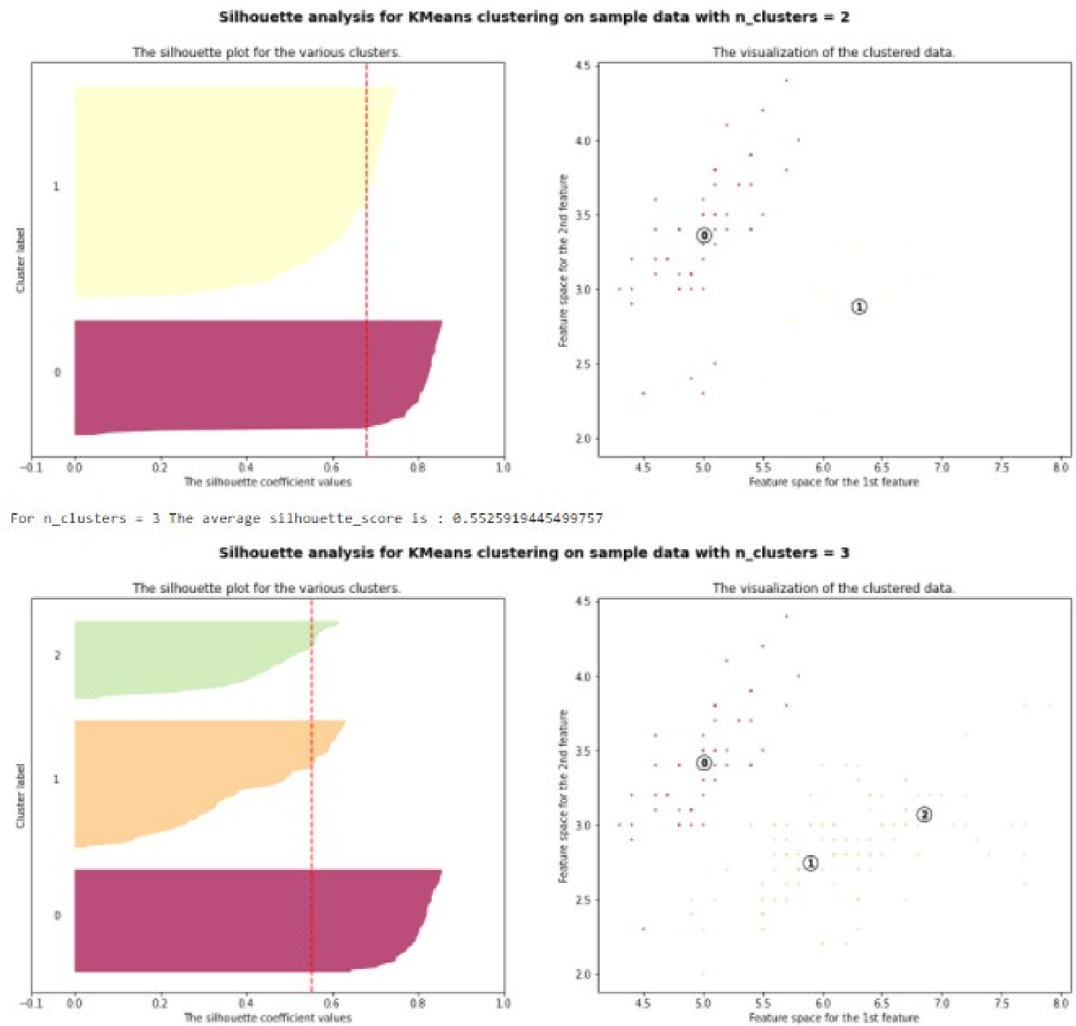


  - 4 variables – Sepal length, sepal width, petal length and petal width.
- We use *K*-means clustering with *K*=3:
  - Silhouette Score = 0.5526 (positive and close to 1.0 is better)
- Refer to sample source file `src02` for the example.

Clustering Example – *K*-means (cont.)

- We can generate the silhouette diagrams for *K*=2 and *K*=3 for comparison:
  - See the sample script `src03`.





Clustering Example – *K*-means (cont.)

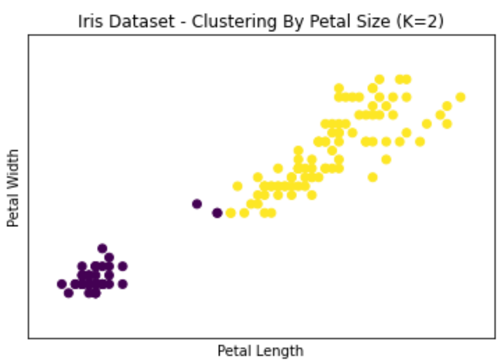
- To identify the distinguishing characteristics of observations in each cluster:
  - We can compute the within-cluster means and standard deviations of the independent variables.
  - Plot scatter plots of the observations using the required independent variables.
  - See the sample script src04.

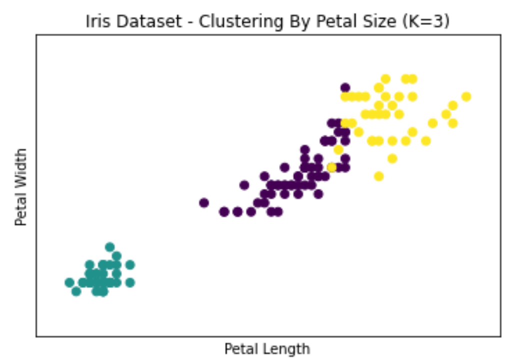
```
Cluster sepal_length   sepal_width    petal_length   petal_width
------------------------------------------------------------------
0       5.006 (0.343)  3.360 (0.440)  1.562 (0.440)  0.289 (0.212)
1       6.301 (0.634)  2.887 (0.327)  4.959 (0.780)  1.696 (0.416)

==================================================================

Cluster sepal_length   sepal_width    petal_length   petal_width
------------------------------------------------------------------
0       5.902 (0.466)  2.748 (0.296)  4.394 (0.509)  1.434 (0.297)
1       5.006 (0.352)  3.418 (0.381)  1.464 (0.174)  0.244 (0.107)
2       6.850 (0.494)  3.074 (0.290)  5.742 (0.489)  2.071 (0.280)

==================================================================
```



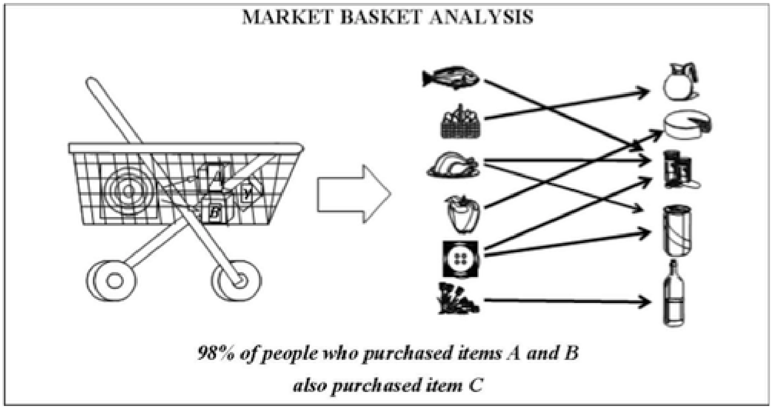


Practical Exercise: PE07-02

Association Rules Mining

Overview of Association Rules

- **Association rules** is a class of unsupervised learning models.
- Aim of association rules is to identify regular patterns and recurrences within a large set of transactions.
- Fairly simple and intuitive.
- Frequently used to investigate:
  - Sales transactions in **market basket analysis**.
  - Navigation paths within websites.



Structure of Association Rules

- Given two propositions Y and Z, which may be true or false, we can state in general terms that a **rule** is an implication of the type $Y \Rightarrow Z$ with the following meaning:
  - If Y is true then Z is also true.
  - A rule is called **probabilistic** if the validity of Z is associated with a probability p.
  - That is, if Y is true then Z is also true with probability p.
  - The notation $\Rightarrow$ read as "material implication":
    - $A \Rightarrow B$ means if A is true then B is also true;
    - if A is false then nothing is said about B.

Representation of Association Rules

- Let $O = \{o_1, o_2, \ldots, o_n\}$ be a set of n **objects**.
- A generic subset $L \subseteq O$ is called an **itemset**.
- An itemset that contains k objects is called a ***k*-itemset**.
- A **transaction** represents a generic itemset recorded in a database in conjunction with an activity or cycle of activities.
- The dataset D is composed of a list of m transactions T_i, each associated with a unique identifier denoted by t_i.
  - Market basket analysis – The <u>objects</u> represent items from the retailer and each <u>transaction</u> corresponds to items listed in a sales receipt.

Representation of Association Rules (cont.)

- Web mining – The objects represent the web pages in a website and each transaction corresponds to the list of web pages visited by a user during one session.
- Example on market basket analysis:

Table 11.1 Example of a dataset consisting of transactions defined over the set of objects $\mathcal{O} = \{a, b, c, d, e\} = \{\text{bread, milk, cereals, coffee, tea}\}$

identifier t_i	transaction T_i
001	$\{a, c\}$
002	$\{a, b, d\}$
003	$\{b, d\}$
004	$\{b, d\}$
005	$\{a, b, c\}$
006	$\{b, c\}$
007	$\{a, c\}$
008	$\{a, b, e\}$
009	$\{a, b, c, e\}$
010	$\{a, e\}$

- This example is for market basket analysis.
- In this example, $t_1 = 001$ and $T_1 = \{a, c\} = \{\text{bread, cereals}\}$.
- Similarly, $t_3 = 003$ and the corresponding $T_3 = \{b, d\} = \{\text{milk, coffee}\}$.

Source: Vercellis (2009), pp. 279

Representation of Association Rules (cont.)

- A dataset of transactions can be represented by a two-dimensional matrix $\mathbf{X}$:
  - The n objects of the set O correspond to the columns of the matrix.
  - The m transactions T_i are the rows.
  - The generic element of $\mathbf{X}$ is defined as:

$$x_{ij} = \begin{cases} 1 & \text{if object } o_j \text{ belongs to transaction } T_i, \\ 0 & \text{otherwise.} \end{cases}$$

Representation of Association Rules (cont.)

- Same example on market basket analysis:

Table 11.2 Matrix **X** for the example of Table 11.1

identifier t_i	a	b	c	d	e
001	1	0	1	0	0
002	1	1	0	1	0
003	0	1	0	1	0
004	0	1	0	1	0
005	1	1	1	0	0
006	0	1	1	0	0
007	1	0	1	0	0
008	1	1	0	0	1
009	1	1	1	0	1
010	1	0	0	0	1

Source: Vercellis (2009), pp. 280

- Recall that $T_1 = \{\text{bread}, \text{cereals}\} = \{a, c\}$
- And $T_3 = \{\text{milk}, \text{coffee}\} = \{b, d\}$

Representation of Association Rules (cont.)

- The representation could be <u>generalized</u>:
  - Assuming that each object o_j appearing in a transaction T_i is associated with a number f_{ij} .
  - f_{ij} represents the frequency in which o_j appears in T_i .
  - Possible to fully describe multiple sales of a given item in a single transaction.
- Let $L \subseteq O$ be a given set of objects, then transaction T is said to **contain** the set L if $L \subseteq T$.
  - In the market basket analysis example, the 2-itemset $L = \{a, c\}$ is <u>contained</u> in the transaction with identifier $t_i = 005$.
  - But it is <u>not contained</u> in $t_i = 006$.

005	$\{a, b, c\}$
006	$\{b, c\}$

Representation of Association Rules (cont.)

- The **empirical frequency** $f(L)$ of an itemset L is defined as the number of transactions T_i existing in the dataset D that contain the set L:

$$f(L) = \mathrm{card}\{T_i : L \subseteq T_i, i = 1,2,\ldots,m\}$$

- For a large sample (i.e., as m increases), the ratio $f(L)/m$ approximate the **probability** $\Pr(L)$ of occurrence of itemset L:
  - That is, the probability that L is contained in a new transaction T recorded in the database.
  - In the market basket analysis example:
    - The set of objects $L = \{a, c\}$ has a frequency $f(L) = 4$.
    - Probability of occurrence is estimated as $\Pr(L) = 4/10 = 0.4$.

identifier t_i	transaction T_i
001	$\{a, c\}$
002	$\{a, b, d\}$
003	$\{b, d\}$
004	$\{b, d\}$
005	$\{a, b, c\}$
006	$\{b, c\}$
007	$\{a, c\}$
008	$\{a, b, e\}$
009	$\{a, b, c, e\}$

Single-dimension Association Rules

- Given two items $L \subset O$ and $H \subset O$ such that $L \cap H = \phi$ and a transaction T, the **association rule** is a probabilistic implication denoted by $L \Rightarrow H$ with the following meaning:
  - If L is contained in T, then H is also contained in T with a given probability p.
  - p is termed the **confidence** of the rule in D and defined as:

$$p = \text{conf}\{L \Rightarrow H\} = \frac{f(L \cup H)}{f(L)}$$

  - The set L is called the **antecedent** or **body** of the rule.
  - H is the **consequent** or **head**.

Single-dimension Association Rules (cont.)

- The confidence of the rule indicates the proportion of transactions containing the set H among those that include L.
- This refers to the **inferential reliability** of the rule.

- As the number of m transactions increases, the confidence approximates the conditional probability that H belongs to a transaction T given that L does belong to T:

$$\Pr\{H \subseteq T \mid L \subseteq T\} = \frac{\Pr\{\{H \subseteq T\} \cap \{L \subseteq T\}\}}{\Pr\{L \subseteq T\}}$$

- Higher confidence thus corresponds to greater probability that itemset H exists in a transaction that also contains the itemset L.

Single-dimension Association Rules (cont.)

- The rule $L \Rightarrow H$ is said to have a **support** s in D if the proportion of transactions containing both L and H is equal to s:

$$s = \text{supp}\{L \Rightarrow H\} = \frac{f(L \cup H)}{m}$$

  - The support of the rule expresses the proportion of transactions containing both the body and head of the rule.
  - Measures the frequency with which an antecedent-consequent pair appears together in the transactions of a dataset.
  - A low support suggests that a rule may have occurred occasionally, of little interest to decision maker and is typically discarded.

Single-dimension Association Rules (cont.)

- As m increases, the <u>support</u> approximates the <u>probability</u> that both L and H are contained in some future transactions.
- In the market basket analysis example:
  - Given the itemsets $L = \{a, c\}$ and $H = \{b\}$ for the rule $L \Rightarrow H$.
  - We have:

$$p = \text{conf}\{L \Rightarrow H\} = \frac{f(L \cup H)}{f(L)} = \frac{2}{4} = \frac{1}{2} = 0.5$$

$$s = \text{supp}\{L \Rightarrow H\} = \frac{f(L \cup H)}{m} = \frac{2}{10} = 0.2$$

identifier t_i	a	b	c	d	e
001	1	0	1	0	0
002	1	1	0	1	0
003	0	1	0	1	0
004	0	1	0	1	0
005	1	1	1	0	0
006	0	1	1	0	0
007	1	0	1	0	0
008	1	1	0	0	1
009	1	1	1	0	1
010	1	0	0	0	1

Strong Association Rules

- Once a dataset D of m transactions has been assigned:
  - Determine minimum threshold value $s_{\min}$ for the support.
  - Determine minimum threshold value $p_{\min}$ for the confidence.
- All **strong association rules** should be determined, characterized by:
  - A support $s \geq s_{\min}$; and
  - A confidence $p \geq p_{\min}$.

Apriori Algorithm

- The **Apriori algorithm** is a more efficient method of extracting strong rules:
  - In the first phase, the algorithm generates the frequent itemsets in a systematic way, without exploring the space of all candidates:
    - The aim of generating frequent itemsets is to extract all sets of objects whose relative frequency is greater than the assigned minimum support s_{min}.
  - In the second phase, it extracts the strong rule.

Example – Market Basket Analysis

- Scikit Learn does not support the Apriori algorithm.
- So we will use Mlxtend (machine learning extensions):

```
import pandas as pd

from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

from IPython.core.display import HTML

import util



display(HTML("<style>pre { white-space: pre !important; }</style>")),
util.set_default_pandas_options()
```

```
df = pd.read_csv('../data/mba.csv', index_col=0)
df
```

```
frequent_itemsets = apriori(df, min_support=0.2, use_colnames=True)
frequent_itemsets
```

```
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=0.5)
rules
```

src05

Example – Market Basket Analysis (cont.)

identifier	a	b	c	d	e
1	1	0	1	0	0
2	1	1	0	1	0
3	0	1	0	1	0
4	0	1	0	1	0
5	1	1	1	0	0
6	0	1	1	0	0
7	1	0	1	0	0
8	1	1	0	0	1
9	1	1	1	0	1
10	1	0	0	0	1

Original transactions dataset

	support	itemsets
0	0.7	(a)
1	0.7	(b)
2	0.5	(c)
3	0.3	(d)
4	0.3	(e)
5	0.4	(b, a)
6	0.4	(c, a)
7	0.3	(a, e)
8	0.3	(c, b)
9	0.3	(b, d)
10	0.2	(b, e)
11	0.2	(c, b, a)
12	0.2	(b, a, e)

Frequent itemsets

Example – Market Basket Analysis (cont.)

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(b)	(a)	0.7	0.7	0.4	0.571429	0.816327	-0.09	0.700000
1	(a)	(b)	0.7	0.7	0.4	0.571429	0.816327	-0.09	0.700000
2	(c)	(a)	0.5	0.7	0.4	0.800000	1.142857	0.05	1.500000
3	(a)	(c)	0.7	0.5	0.4	0.571429	1.142857	0.05	1.166667
4	(a)	(e)	0.7	0.3	0.3	0.428571	1.428571	0.09	1.225000
5	(e)	(a)	0.3	0.7	0.3	1.000000	1.428571	0.09	inf
6	(c)	(b)	0.5	0.7	0.3	0.600000	0.857143	-0.05	0.750000
7	(b)	(c)	0.7	0.5	0.3	0.428571	0.857143	-0.05	0.875000
8	(b)	(d)	0.7	0.3	0.3	0.428571	1.428571	0.09	1.225000
9	(d)	(b)	0.3	0.7	0.3	1.000000	1.428571	0.09	inf
10	(b)	(e)	0.7	0.3	0.2	0.285714	0.952381	-0.01	0.980000
11	(e)	(b)	0.3	0.7	0.2	0.666667	0.952381	-0.01	0.900000
12	(c, b)	(a)	0.3	0.7	0.2	0.666667	0.952381	-0.01	0.900000
13	(c, a)	(b)	0.4	0.7	0.2	0.500000	0.714286	-0.08	0.600000
14	(b, a)	(c)	0.4	0.5	0.2	0.500000	1.000000	0.00	1.000000
15	(c)	(b, a)	0.5	0.4	0.2	0.400000	1.000000	0.00	1.000000
16	(b)	(c, a)	0.7	0.4	0.2	0.285714	0.714286	-0.08	0.840000
17	(a)	(c, b)	0.7	0.3	0.2	0.285714	0.952381	-0.01	0.980000
18	(b, a)	(e)	0.4	0.3	0.2	0.500000	1.666667	0.08	1.400000
19	(b, e)	(a)	0.2	0.7	0.2	1.000000	1.428571	0.06	inf
20	(a, e)	(b)	0.3	0.7	0.2	0.666667	0.952381	-0.01	0.900000
21	(b)	(a, e)	0.7	0.3	0.2	0.285714	0.952381	-0.01	0.980000
22	(a)	(b, e)	0.7	0.2	0.2	0.285714	1.428571	0.06	1.120000
23	(e)	(b, a)	0.3	0.4	0.2	0.666667	1.666667	0.08	1.800000

Practical Exercise: PE08-01 to PE08-02