

Practical-1

Aim:-Introduction to data mining and tool and its installation.

Data Mining is defined as extracting information from huge sets of data. In other words, we can say that data mining is the procedure of mining knowledge from data. The information or knowledge extracted so can be used for any of the following applications –

- Market Analysis
- Fraud Detection
- Customer Retention
- Production Control
- Science Exploration

Data Mining Techniques

1. Classification:

This analysis is used to retrieve important and relevant information about data, and metadata. This data mining method helps to classify data in different classes.

2. Clustering:

Clustering analysis is a data mining technique to identify data that are like each other. This process helps to understand the differences and similarities between the data.

3. Regression:

Regression analysis is the data mining method of identifying and analyzing the relationship between variables. It is used to identify the likelihood of a specific variable, given the presence of other variables.

4. Association Rules:

This data mining technique helps to find the association between two or more Items. It discovers a hidden pattern in the data set.

5. Outer detection:

This type of data mining technique refers to observation of data items in the dataset which do not match an expected pattern or expected behavior. This technique can be used in a variety of domains, such as intrusion, detection, fraud or fault detection, etc. Outer detection is also called Outlier Analysis or Outlier mining.

6. Sequential Patterns:

This data mining technique helps to discover or identify similar patterns or trends in transaction data for certain period.

7. Prediction:

Prediction has used a combination of the other data mining techniques like trends, sequential patterns, clustering, classification, etc. It analyzes past events or instances in a right sequence for predicting a future event.

Data Mining Tools

R-language:

R language is an open source tool for statistical computing and graphics. R has a wide variety of statistical, classical statistical tests, time-series analysis, classification and graphical techniques. It offers effective data handling and storage facility.

Oracle Data Mining:

Oracle Data Mining popularly knowns as ODM is a module of the Oracle Advanced Analytics Database. This Data mining tool allows data analysts to generate detailed insights and makes predictions. It helps predict customer behaviour, develops customer profiles, identifies cross-selling opportunities.

Installation

Oracle Data Miner

Oracle Data Miner is a graphical user interface for Oracle Data Mining. Oracle Data Miner's easy-to-use wizards guide you through the data preparation, data mining, model evaluation, and model scoring process.

Oracle Data Miner is supported on Windows 2000, Windows XP Professional Edition, and Linux.

Installing Oracle Data Miner on Windows

To install Oracle Data Miner on a Microsoft Windows platform, take these steps:

1. Download Oracle Data Miner 10.2 from the Oracle Web site.
2. Open the ZIP file and extract all files to an empty directory (such as `C:\ODMINER`). Be sure to use folder names so that the files retain their original organization in subfolders.
3. Create a Windows shortcut to `BIN\ODMINERW.EXE`, and drag the shortcut to your Windows desktop for easy access.

Installing Oracle Data Miner on Linux

To install Oracle Data Miner on a Linux or Unix platform, take these steps:

- Download Oracle Data Miner from the Oracle Website.
- Open the ZIP file and extract all files to an empty directory. The following command creates a directory named `odminer` and inflates the files into it.

```
unzip odminer.zip -d odminer
```

- Grant execution permission to `bin/odminer` with a command such as this:

```
chmod +x odminer/bin/odminer
```

- Run the `odminer` executable from the `bin` subdirectory

```
chmod odminer/bin
```

```
odminer
```

Installing Oracle Client

- Logon as Administrator.
- From the Client installation directory, run `SETUP.EXE`.

Oracle Universal Installer opens and displays the Welcome page. Click Next.

- On the Select Installation Type page, choose Administrator and click Next.
- On the Specify Home Details page, provide the path of a home directory for the Oracle Client installation.
- On the Product-Specific Prerequisite page, verify that all checks succeeded. If any checks failed, then you must correct the problem before proceeding.
- On the Summary page, click the Install button.
- The Installer displays the progress of the installation. When you click Next, the Configuration Assistants page is displayed.
- Oracle Net Configuration Assistant starts and displays the Welcome page. Choose Perform Typical Configuration.
- When the Oracle Net Configuration process is complete, click Finish.

Practical:-2

Aim:- Introduction to data warehousing.

A data warehousing is defined as a technique for collecting and managing data from varied sources to provide meaningful business insights. It is a blend of technologies and components which aids the strategic use of data. It is electronic storage of a large amount of information by a business which is designed for query and analysis instead of transaction processing. It is a process of transforming data into information and making it available to users in a timely manner to make a difference.

How Datawarehouse works?

A Data Warehouse works as a central repository where information arrives from one or more data sources. Data flows into a data warehouse from the transactional system and other relational databases.

Data may be: Structured, Semi-structured, Unstructured data

The data is processed, transformed, and ingested so that users can access the processed data in the Data Warehouse through Business Intelligence tools, SQL clients, and spreadsheets. A data warehouse merges information coming from different sources into one comprehensive database.

By merging all of this information in one place, an organization can analyze its customers more holistically. This helps to ensure that it has considered all the information available. Data warehousing makes data mining possible. Data mining is looking for patterns in the data that may lead to higher sales and profits.

Types of Data Warehouse:

1. Enterprise Data Warehouse:

Enterprise Data Warehouse is a centralized warehouse. It provides decision support service across the enterprise. It offers a unified approach for organizing and representing data. It also provide the ability to classify data according to the subject and give access according to those divisions.

2. Operational Data Store:

Operational Data Store, which is also called ODS, are nothing but data store required when neither Data warehouse nor OLTP systems support organizations reporting needs. In ODS, Data warehouse is refreshed in real time. Hence, it is widely preferred for routine activities like storing records of the Employees.

3. Data Mart:

A data mart is a subset of the data warehouse. It specially designed for a particular line of business, such as sales, finance, sales or finance. In an independent data mart, data can collect directly from sources.

Components of Data warehouse:

Load manager: Load manager is also called the front component. It performs with all the operations associated with the extraction and load of data into the warehouse. These operations include transformations to prepare the data for entering into the Data warehouse.

Warehouse Manager: Warehouse manager performs operations associated with the management of the data in the warehouse. It performs operations like analysis of data to ensure consistency, creation of indexes and views, generation of denormalization and aggregations, transformation and merging of source data and archiving and baking-up data.

Query Manager: Query manager is also known as backend component. It performs all the operation operations related to the management of user queries. The operations of this Data warehouse components are direct queries to the appropriate tables for scheduling the execution of queries.

End-user access tools: This is categorized into five different groups like 1. Data Reporting 2. Query Tools 3. Application development tools 4. EIS tools, 5. OLAP tools and data mining tools.

Data warehouse characteristics:

•Subject-oriented

A data warehouse is always a subject oriented as it delivers information about a theme instead of organization's current operations. It can be achieved on specific theme. That means the data warehousing process is proposed to handle with a specific theme which is more defined. These themes can be sales, distributions, marketing etc.

•Integrated

A data warehouse is built by integrating data from various sources of data such that a mainframe and a relational database. In addition, it must have reliable naming conventions, format and codes. Integration of data warehouse benefits in effective analysis of data. Reliability in naming conventions, column scaling, encoding structure etc. should be confirmed.

•Time-Variant

In this data is maintained via different intervals of time such as weekly, monthly, or annually etc. It finds various time limit which are structured between the large datasets and are held in online transaction process (OLTP). The time limits for data warehouse is wide-ranged than that of operational systems.

•Non-Volatile

Data is not erased or deleted when new data is inserted. It includes the mammoth quantity of data that is inserted into modification between the selected quantity on logical business. It evaluates the analysis within the technologies of warehouse.

Practical:- 3

Aim:-KDD Process in Data Mining

Data Mining also known as Knowledge Discovery in Databases, refers to the nontrivial extraction of implicit, previously unknown and potentially useful information from data stored in databases.

1. **Data Cleaning:** Data cleaning is defined as removal of noisy and irrelevant data from collection.
 - Cleaning in case of *Missing values*.
 - Cleaning *noisy* data, where noise is a random or variance error.
 - Cleaning with *Data discrepancy detection* and *Data transformation tools*.
2. **Data Integration:** Data integration is defined as heterogeneous data from multiple sources combined in a common source(DataWarehouse).
 - Data integration using *Data Migration tools*.
 - Data integration using *Data Synchronization tools*.
 - Data integration using *ETL*(Extract-Load-Transformation) process.
3. **Data Selection:** Data selection is defined as the process where data relevant to the analysis is decided and retrieved from the data collection.
 - Data selection using *Neural network*.
 - Data selection using *Decision Trees*.
 - Data selection using *Naive bayes*.
 - Data selection using *Clustering, Regression*, etc.
4. **Data Transformation:** Data Transformation is defined as the process of transforming data into appropriate form required by mining procedure. Data Transformation is a two step process:
 - *Data Mapping:* Assigning elements from source base to destination to capture transformations.
 - *Code generation:* Creation of the actual transformation program.
5. **Data Mining:** Data mining is defined as clever techniques that are applied to extract patterns potentially useful.
 - Transforms task relevant data into *patterns*.
 - Decides purpose of model using *classification* or *characterization*.
6. **Pattern Evaluation:** Pattern Evaluation is defined as as identifying strictly increasing patterns representing knowledge based on given measures.
 - Find *interestingness score* of each pattern.
 - Uses *summarization* and *Visualization* to make data understandable by user.
7. **Knowledge representation:** Knowledge representation is defined as technique which utilizes visualization tools to represent data mining results.
 - Generate *reports*.
 - Generate *tables*.
 - Generate *discriminant rules, classification rules, characterization rules*, etc.

Practical:-4

Aim:-Demonstration of Association rule process on dataset using apriori algorithm.

Apriori algorithm is given by R. Agrawal and R. Srikant in 1994 for finding frequent itemsets in a dataset for boolean association rule. Name of the algorithm is Apriori because it uses prior knowledge of frequent itemset properties. We apply an iterative approach or level-wise search where k-frequent itemsets are used to find k+1 itemsets.

To improve the efficiency of level-wise generation of frequent itemsets, an important property is used called *Apriori property* which helps by reducing the search space.

Apriori Property:

All non-empty subset of frequent itemset must be frequent. The key concept of Apriori algorithm is its anti-monotonicity of support measure. Apriori assumes that

All subsets of a frequent itemset must be frequent (Apriori property).
If an itemset is infrequent, all its supersets will be infrequent.

Before we start understanding the algorithm, go through some definitions which are explained in my previous post.

Consider the following dataset and we will find frequent itemsets and generate association rules for them.

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

minimum support count is 2
minimum confidence is 60%

Step-1: K=1

(I) Create a table containing support count of each item present in dataset – Called **C1(candidate set)**

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

(II) compare candidate set item's support count with minimum support count(here $\text{min_support}=2$ if support_count of candidate set items is less than min_support then remove those items). This gives us itemset L1.

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

Step-2: K=2

- Generate candidate set C2 using L1 (this is called join step). Condition of joining L_{k-1} and L_{k-1} is that it should have (K-2) elements in common.
- Check all subsets of an itemset are frequent or not and if not frequent remove that itemset.(Example subset of {I1, I2} are {I1}, {I2} they are frequent. Check for each itemset)
- Now find support count of these itemsets by searching in dataset.

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

(II) compare candidate (C2) support count with minimum support count(here $\text{min_support}=2$ if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L2.

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

Step-3:

- Generate candidate set C3 using L2 (join step). Condition of joining L_{k-1} and L_{k-1} is that it should have (K-2) elements in common. So here, for L2, first

element should match.

So itemset generated by joining L2 is {I1, I2, I3} {I1, I2, I5} {I1, I3, I5} {I2, I3, I4} {I2, I4, I5} {I2, I3, I5}

- Check if all subsets of these itemsets are frequent or not and if not, then remove that itemset. (Here subset of {I1, I2, I3} are {I1, I2}, {I2, I3}, {I1, I3} which are frequent. For {I2, I3, I4}, subset {I3, I4} is not frequent so remove it. Similarly check for every itemset)
- find support count of these remaining itemset by searching in dataset.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

(II) Compare candidate (C3) support count with minimum support count (here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L3.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

Step-4:

- Generate candidate set C4 using L3 (join step). Condition of joining L_{k-1} and L_{k-1} ($K=4$) is that, they should have ($K-2$) elements in common. So here, for L3, first 2 elements (items) should match.
- Check all subsets of these itemsets are frequent or not (Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contains {I1, I3, I5}, which is not frequent). So no itemset in C4
- We stop here because no frequent itemsets are found further

Thus, we have discovered all the frequent item-sets. Now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule.

Confidence –

A confidence of 60% means that 60% of the customers, who purchased milk and bread also bought butter.

$$\text{Confidence (A} \rightarrow \text{B)} = \text{Support_count (A} \cup \text{B)} / \text{Support_count (A)}$$

So here, by taking an example of any frequent itemset, we will show the rule generation.

Itemset {I1, I2, I3} //from L3

SO rules can be

$$[I1 \wedge I2] \Rightarrow [I3] \text{ //confidence} = \text{sup}(I1 \wedge I2 \wedge I3) / \text{sup}(I1 \wedge I2) = 2/4 * 100 = 50\%$$

$$[I1 \wedge I3] \Rightarrow [I2] \text{ //confidence} = \text{sup}(I1 \wedge I2 \wedge I3) / \text{sup}(I1 \wedge I3) = 2/4 * 100 = 50\%$$

$$[I2 \wedge I3] \Rightarrow [I1] \text{ //confidence} = \text{sup}(I1 \wedge I2 \wedge I3) / \text{sup}(I2 \wedge I3) = 2/4 * 100 = 50\%$$

$$[I1] \Rightarrow [I2 \wedge I3] \text{ //confidence} = \text{sup}(I1 \wedge I2 \wedge I3) / \text{sup}(I1) = 2/6 * 100 = 33\%$$

$[I2] \Rightarrow [I1 \wedge I3] \text{ //confidence} = \frac{\text{sup}(I1 \wedge I2 \wedge I3)}{\text{sup}(I2)} = \frac{2}{7} * 100 = 28\%$
 $[I3] \Rightarrow [I1 \wedge I2] \text{ //confidence} = \frac{\text{sup}(I1 \wedge I2 \wedge I3)}{\text{sup}(I3)} = \frac{2}{6} * 100 = 33\%$

So if minimum confidence is 50%, then first 3 rules can be considered as strong association rules.

Practical:-5:

Aim:- Demonstration of classification rule process on dataset using id3 algorithm.

ID3 stands for Iterative Dichotomiser 3. It is a classification algorithm that follows a greedy approach by selecting a best attribute that yields maximum Information Gain(IG) or minimum Entropy(H).

What are the steps in ID3 algorithm?

1. Calculate entropy for dataset.
2. For each attribute/feature
 - a. Calculate entropy for all its categorical values.
 - b. Calculate information gain for the feature.
3. Find the feature with maximum information gain.
4. Repeat it until we get the desired tree.

Use ID3 algorithm on a data

Here, dataset is of binary classes(yes and no), where 9 out of 14 are "yes" and 5 out of 14 are "no".

Complete entropy of dataset is –

$$\begin{aligned} H(S) &= - p(\text{yes}) * \log_2(p(\text{yes})) - p(\text{no}) * \log_2(p(\text{no})) \\ &= - (9/14) * \log_2(9/14) - (5/14) * \log_2(5/14) \\ &= - (-0.41) - (-0.53) \\ &= 0.94 \end{aligned}$$

For each attribute of the dataset, let's follow the step-2 of pseudocode : -

First Attribute - Outlook

Categorical values - sunny, overcast and rain

$$H(\text{Outlook}=\text{sunny}) = -(2/5)*\log(2/5)-(3/5)*\log(3/5)=0.971$$

$$H(\text{Outlook}=\text{rain}) = -(3/5)*\log(3/5)-(2/5)*\log(2/5)=0.971$$

$$H(\text{Outlook}=\text{overcast}) = -(4/4)*\log(4/4)-0 = 0$$

Average Entropy Information for Outlook -

$$\begin{aligned} I(\text{Outlook}) &= p(\text{sunny}) * H(\text{Outlook}=\text{sunny}) + p(\text{rain}) * H(\text{Outlook}=\text{rain}) + p(\text{overcast}) * \\ &H(\text{Outlook}=\text{overcast}) \\ &= (5/14)*0.971 + (5/14)*0.971 + (4/14)*0 \\ &= 0.693 \end{aligned}$$

$$\begin{aligned} \text{Information Gain} &= H(S) - I(\text{Outlook}) \\ &= 0.94 - 0.693 \\ &= 0.247 \end{aligned}$$

Second Attribute - Temperature

Categorical values - hot, mild, cool

$$H(\text{Temperature}=\text{hot}) = -(2/4)*\log(2/4)-(2/4)*\log(2/4) = 1$$

$$H(\text{Temperature}=\text{cool}) = -(3/4)*\log(3/4)-(1/4)*\log(1/4) = 0.811$$

$$H(\text{Temperature}=\text{mild}) = -(4/6)*\log(4/6)-(2/6)*\log(2/6) = 0.9179$$

Average Entropy Information for Temperature -

$$\begin{aligned} I(\text{Temperature}) &= p(\text{hot})*H(\text{Temperature}=\text{hot}) + p(\text{mild})*H(\text{Temperature}=\text{mild}) + \\ &p(\text{cool})*H(\text{Temperature}=\text{cool}) \\ &= (4/14)*1 + (6/14)*0.9179 + (4/14)*0.811 \\ &= 0.9108 \end{aligned}$$

$$\begin{aligned} \text{Information Gain} &= H(S) - I(\text{Temperature}) \\ &= 0.94 - 0.9108 \\ &= 0.0292 \end{aligned}$$

Third Attribute - Humidity

Categorical values - high, normal

$$H(\text{Humidity}=\text{high}) = -(3/7)*\log(3/7)-(4/7)*\log(4/7) = 0.983$$

$$H(\text{Humidity}=\text{normal}) = -(6/7)*\log(6/7)-(1/7)*\log(1/7) = 0.591$$

Average Entropy Information for Humidity -

$$\begin{aligned} I(\text{Humidity}) &= p(\text{high})*H(\text{Humidity}=\text{high}) + p(\text{normal})*H(\text{Humidity}=\text{normal}) \\ &= (7/14)*0.983 + (7/14)*0.591 \\ &= 0.787 \end{aligned}$$

$$\begin{aligned} \text{Information Gain} &= H(S) - I(\text{Humidity}) \\ &= 0.94 - 0.787 \\ &= 0.153 \end{aligned}$$

Fourth Attribute - Wind

Categorical values - weak, strong

$$H(\text{Wind}=\text{weak}) = -(6/8)*\log(6/8)-(2/8)*\log(2/8) = 0.811$$

$$H(\text{Wind}=\text{strong}) = -(3/6)*\log(3/6)-(3/6)*\log(3/6) = 1$$

Average Entropy Information for Wind -

$$\begin{aligned} I(\text{Wind}) &= p(\text{weak})*H(\text{Wind}=\text{weak}) + p(\text{strong})*H(\text{Wind}=\text{strong}) \\ &= (8/14)*0.811 + (6/14)*1 \\ &= 0.892 \end{aligned}$$

$$\begin{aligned} \text{Information Gain} &= H(S) - I(\text{Wind}) \\ &= 0.94 - 0.892 \\ &= 0.048 \end{aligned}$$

Now, finding the best attribute for splitting the data with Outlook=Sunny values { Dataset rows = [1, 2, 8, 9, 11]}.

Complete entropy of Sunny is -

$$\begin{aligned}
H(S) &= -p(\text{yes}) * \log_2(p(\text{yes})) - p(\text{no}) * \log_2(p(\text{no})) \\
&= - (2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) \\
&= 0.971
\end{aligned}$$

First Attribute - Temperature

Categorical values - hot, mild, cool

$$H(\text{Sunny}, \text{Temperature}=\text{hot}) = -0 - (2/2) * \log(2/2) = 0$$

$$H(\text{Sunny}, \text{Temperature}=\text{cool}) = -(1) * \log(1) - 0 = 0$$

$$H(\text{Sunny}, \text{Temperature}=\text{mild}) = -(1/2) * \log(1/2) - (1/2) * \log(1/2) = 1$$

Average Entropy Information for Temperature -

$$\begin{aligned}
I(\text{Sunny}, \text{Temperature}) &= p(\text{Sunny}, \text{hot}) * H(\text{Sunny}, \text{Temperature}=\text{hot}) + p(\text{Sunny}, \\
&\text{mild}) * H(\text{Sunny}, \text{Temperature}=\text{mild}) + p(\text{Sunny}, \text{cool}) * H(\text{Sunny}, \text{Temperature}=\text{cool}) \\
&= (2/5) * 0 + (1/5) * 0 + (2/5) * 1 \\
&= 0.4
\end{aligned}$$

$$\begin{aligned}
\text{Information Gain} &= H(\text{Sunny}) - I(\text{Sunny}, \text{Temperature}) \\
&= 0.971 - 0.4 \\
&= 0.571
\end{aligned}$$

Second Attribute - Humidity

Categorical values - high, normal

$$H(\text{Sunny}, \text{Humidity}=\text{high}) = -0 - (3/3) * \log(3/3) = 0$$

$$H(\text{Sunny}, \text{Humidity}=\text{normal}) = -(2/2) * \log(2/2) - 0 = 0$$

Average Entropy Information for Humidity -

$$\begin{aligned}
I(\text{Sunny}, \text{Humidity}) &= p(\text{Sunny}, \text{high}) * H(\text{Sunny}, \text{Humidity}=\text{high}) + p(\text{Sunny}, \\
&\text{normal}) * H(\text{Sunny}, \text{Humidity}=\text{normal}) \\
&= (3/5) * 0 + (2/5) * 0 \\
&= 0
\end{aligned}$$

$$\begin{aligned}
\text{Information Gain} &= H(\text{Sunny}) - I(\text{Sunny}, \text{Humidity}) \\
&= 0.971 - 0 \\
&= 0.971
\end{aligned}$$

Third Attribute - Wind

Categorical values - weak, strong

$$H(\text{Sunny}, \text{Wind}=\text{weak}) = -(1/3) * \log(1/3) - (2/3) * \log(2/3) = 0.918$$

$$H(\text{Sunny}, \text{Wind}=\text{strong}) = -(1/2) * \log(1/2) - (1/2) * \log(1/2) = 1$$

Average Entropy Information for Wind -

$$\begin{aligned}
I(\text{Sunny}, \text{Wind}) &= p(\text{Sunny}, \text{weak}) * H(\text{Sunny}, \text{Wind}=\text{weak}) + p(\text{Sunny}, \text{strong}) * H(\text{Sunny}, \\
&\text{Wind}=\text{strong}) \\
&= (3/5) * 0.918 + (2/5) * 1 \\
&= 0.9508
\end{aligned}$$

$$\begin{aligned}
\text{Information Gain} &= H(\text{Sunny}) - I(\text{Sunny}, \text{Wind}) \\
&= 0.971 - 0.9508
\end{aligned}$$

$$= 0.0202$$

Here, when Outlook = Sunny and Humidity = High, it is a pure class of category "no". And When Outlook = Sunny and Humidity = Normal, it is again a pure class of category "yes". Therefore, we don't need to do further calculations.

Now, finding the best attribute for splitting the data with Outlook=Sunny values { Dataset rows = [4, 5, 6, 10, 14]}.

Complete entropy of Rain is -

$$\begin{aligned} H(S) &= -p(\text{yes}) * \log_2(p(\text{yes})) - p(\text{no}) * \log_2(p(\text{no})) \\ &= - (3/5) * \log(3/5) - (2/5) * \log(2/5) \\ &= 0.971 \end{aligned}$$

First Attribute - Temperature

Categorical values - mild, cool

$$H(\text{Rain}, \text{Temperature}=\text{cool}) = -(1/2)*\log(1/2) - (1/2)*\log(1/2) = 1$$

$$H(\text{Rain}, \text{Temperature}=\text{mild}) = -(2/3)*\log(2/3) - (1/3)*\log(1/3) = 0.918$$

Average Entropy Information for Temperature -

$$\begin{aligned} I(\text{Rain}, \text{Temperature}) &= p(\text{Rain}, \text{mild}) * H(\text{Rain}, \text{Temperature}=\text{mild}) + p(\text{Rain}, \text{cool}) * H(\text{Rain}, \text{Temperature}=\text{cool}) \\ &= (2/5) * 1 + (3/5) * 0.918 \\ &= 0.9508 \end{aligned}$$

$$\begin{aligned} \text{Information Gain} &= H(\text{Rain}) - I(\text{Rain}, \text{Temperature}) \\ &= 0.971 - 0.9508 \\ &= 0.0202 \end{aligned}$$

Second Attribute - Wind

Categorical values - weak, strong

$$H(\text{Wind}=\text{weak}) = -(3/3)*\log(3/3) - 0 = 0$$

$$H(\text{Wind}=\text{strong}) = 0 - (2/2)*\log(2/2) = 0$$

Average Entropy Information for Wind -

$$\begin{aligned} I(\text{Wind}) &= p(\text{Rain}, \text{weak}) * H(\text{Rain}, \text{Wind}=\text{weak}) + p(\text{Rain}, \text{strong}) * H(\text{Rain}, \text{Wind}=\text{strong}) \\ &= (3/5) * 0 + (2/5) * 0 \\ &= 0 \end{aligned}$$

$$\begin{aligned} \text{Information Gain} &= H(\text{Rain}) - I(\text{Rain}, \text{Wind}) \\ &= 0.971 - 0 \\ &= 0.971 \end{aligned}$$

Here, when Outlook = Rain and Wind = Strong, it is a pure class of category "no". And When Outlook = Rain and Wind = Weak, it is again a pure class of category "yes". And this is our final desired tree for the given dataset.

Practical:-6

Aim:-Demonstration of clustering rule process on dataset using simple k-means.

k-means creates k groups from a set of objects so that the members of a group are more similar. It's a popular cluster analysis technique for exploring a dataset.

K-means Clustering Method:

If k is given, the K-means algorithm can be executed in the following steps:

- Partition of objects into k non-empty subsets
- Identifying the cluster centroids (mean point) of the current partition.
- Assigning each point to a specific cluster
- Compute the distances from each point and allot points to the cluster where the distance from the centroid is minimum.
- After re-allotting the points, find the centroid of the new cluster formed.

This is how the algorithm works:

1. K centroids are created randomly (based on the predefined value of K).
2. K-means allocates every data point in the dataset to the nearest centroid (minimizing Euclidean distances between them), meaning that a data point is considered to be in a particular cluster if it is closer to that cluster's centroid than any other centroid.
3. Then K-means recalculates the centroids by taking the mean of all data points assigned to that centroid's cluster, hence reducing the total intra-cluster variance in relation to the previous step. The "means" in the K-means refers to averaging the data and finding the new centroid.
4. The algorithm iterates between steps 2 and 3 until some criteria is met (e.g. the sum of distances between the data points and their corresponding centroid is minimized, a maximum number of iterations is reached, no changes in centroids value or no data points change clusters).

Implementation of k-means

When it comes to popularity among clustering algorithms, *k-means* is the one. The approach behind this simple algorithm is just about some iterations and updating clusters as per distance measures that are computed repeatedly. k is the number of clusters that are to be formed. This content represents the implementation of ***K-Means algorithm*** from the scratch using *numpy, pandas and plotly*.

Imports that are required for implementing:

```
import pandas as pd
import numpy as np
import plotly.offline as plt
import plotly.graph_objs as go
```

Data set that is taken here consists of 788 data points and have 7 shapes that can be identified visually.

Data points initially before clustering

One notable thing about this algorithm is that, it changes the cluster groups whenever it is re-evaluated. Hence, a data point may become a part of different cluster after the program is executed again.

Initially, two dataframes are created that will be used as set of **current means and previous means**. Before the iteration, one dataframe from these two contains randomly selected means from the targeted data set itself.

```
k_means = (data.sample(k, replace=False)) # store current means
k_means2 = pd.DataFrame()                # store previous means
clusters = pd.DataFrame()                 # store distances
```

Here, first statement shows assigning set of random series from the data set itself to the variable named *k_means*, using the method *sample()* whose return type is dataframe. Another variable is *k_means2* that is currently an empty dataframe which will eventually store previous means. Along with these, a new dataframe named **clusters** is created for storing the calculated distances.

- Second step is to create a parent loop that iterates until **current and previous means** (i.e., *k_means* and *k_means2*) are equal.
- Next, calculate distances of all the data points from each mean stored in *k_means* and put it in dataframe named **clusters**.
- A new column named '**MDCluster**' will be appended to the existing dataframe that denotes the **label of the cluster** to which all the data points are assigned. This assignment is done by fetching index (which will act as label) of minimum value from the **clusters** dataframe.
- After setting each data point to specific cluster using '**MDCluster**'. It's time to calculate new means of updated clusters and store older means in *k_means2*. These new means will be allocated to *k_means* and calculated repeatedly from '**MDCluster**' using groupby and aggregate function.

```
while not k_means2.equals(k_means):
```

```
    # distance matrix (euclidean distance)
    cluster_count = 0
    for idx, k_mean in k_means.iterrows():
```



```

clusters[cluster_count] = (data[k_means.columns] -
                           np.array(k_mean)).pow(2).sum(1).pow(0.5)
cluster_count += 1

# update cluster
data['MDCluster'] = clusters.idxmin(axis=1)

# store previous cluster
k_means2 = k_means
k_means = pd.DataFrame()
k_means_frame = data.groupby('MDCluster').agg(np.mean)

k_means[k_means_frame.columns] =
    k_means_frame[k_means_frame.columns]

```

Changes in the dataframe regarding clustering labels in '**MDCluster**' in each iteration of updating means are as follows:

Shows data points in different clusters in iterations

If these intermediate clusters formed by changing means are plotted, then they would seem like:

Iterative steps and cluster reformations

Final step is to plot all the points using **plotly** library and see how the clusters are formed. Here, along with the data points, centres of each cluster are also plotted as bold black dots. Following code snippet shows how the points are plotted:

```

# plotting
data_graph = [go.Scatter(
    x=data['V1'],
    y=data['V2'].where(data['MDCluster'] == c),
    mode='markers',
    name='Cluster: ' + str(c))
    for c in range(k)]
data_graph.append(
    go.Scatter(
        x=k_means['V1'],
        y=k_means['V2'],
        mode='markers',
        marker=dict(
            size=10,
            color='#000000',
        ),
        name='Centroids of Clusters'
    )
)

plt.plot(data_graph, filename='../output_files/cluster.html')

```

Practical:-7

Aim:-Installation and working of Big Data and Cloud tools.

Cassandra and why to use it?

Apache **Cassandra** is a highly scalable, high-performance distributed database designed to handle large amounts of data across many commodity (cheap) servers, providing high availability with no single point of failure. It is a type of NoSQL database that is considered a key-value store. It's query much similar to SQL.

- Outgrown your regular relational database i.e petabytes, zettabytes size
- Need of fast insert/update/selection of data that is scalable and fault-tolerant.

Steps to setup Cassandra on Window Machine locally.

1. Cassandra need JDK to run. First need to install JDK on the PC.

- Go To Oracle (SignUp required) or Filehippo to download JDK 1.8 from which you find suitable.
- Run the Install as it is.
- Configure JDK path As:
- Copy the JDK where is installed it's bin directory. Mine is 64-bit found inside "C:\Program Files\Java\jdk1.8.0_181\bin" and place on Environment Variable as new Env_name : "JAVA_HOME" later we use for cassandra.
- Note: Progra~1 = 'Program Files' for 64-bit
Progra~2 = 'Program Files(x86)' for 32-bit in Environment path.

Environment variable set in JAVA_HOME

2. Go to Apache Cassandra Download Page. And Download the latest version. The latest version at that time is cassandra-3.11.4

- Unzip it and place all files inside sub folder into "C:\Program Files\apache-cassandra-3.11.4"
- Open CMD inside "C:\Program Files\apache-cassandra-3.11.4\bin" and type cassandra then the output as shown below.

JAVA_HOME should set error

- If you got above error then Edit "cassandra.bat" Add JAVA_HOME as:

```
set JAVA_HOME="C:\Progra~1\Java\jdk1.8.0_181"
```

- Save the cassandra.bat then run "cassandra.bat -f" with CMD Run as Administrator. This time, error should resolve if your edit successful.

3. Need Python2.7 to run Cassandra Query shell `cqlsh` .

- Download Python2.7 latest version and extract inside the “C:\Program Files\apache-cassandra-3.11.4\bin” during installing. Or simply copy after installing python2.7 all files inside “bin” directory of cassandra-3.11.4. This is a easy way to go.
- Directory of cassandra-3.11.4\bin with python all files

4. Finally run the Cassandra Server as “cassandra.bat -f” with CMD Run as Administrator from the bin directory. Following screen should see on successful start.

- Don’t close it, keep running.
- Then, open other CMD go over the bin directory of cassandra installed. To run `cqlsh` by type `cqlsh` . Following output should get.

Let’s try to create the table in it:

First create the namespace where data holds.

```
cqlsh> create keyspace test
```

```
cqlsh> Use test;
```

```
cqlsh:test>CREATE TABLE emp(
emp_id int PRIMARY KEY,
emp_name text,
emp_city text,
emp_sal varint,
emp_phone varint
);
```

```
For Verification: cqlsh:test> select * from emp;
```

Practical:-8

Aim:-Understanding files formats supported by the tool (Rapid Miner).

RapidMiner Studio supports a wide range of file formats, databases, and cloud services, either natively or via extensions.

Supported databases:

SQL: RapidMiner Studio supports all relational database systems offering a fully compliant JDBC driver, including:

- AccessDB
- HSQLDB
- Microsoft SQL Server (JTDS / Microsoft)
- MySQL
- Oracle
- PostgreSQL
- Sybase

NoSQL: RapidMiner Studio can connect to a variety of NoSQL databases via extensions.

- Cassandra
- MongoDB
- Solr
- Splunk (*read only*)

Supported cloud services: RapidMiner Studio supports the following cloud services:

- Amazon S3
- Azure Blob Storage
- Azure Data Lake Storage
- Dropbox
- Google Cloud Storage
- Salesforce

Supported file types: RapidMiner Studio reads and / or writes the following file types:

- ACCDB - Microsoft Access database
- ARFF - Weka file format
- CSV - Comma Separated Value
- DBF - dBASE Database file format (read only)
- HYPER - Tableau file format
- MDB - Microsoft Access database
- SAS - SAS file format up to v9.2 (read only)
- SAV - IBM SPSS file format (read only)
- TDE - Tableau file format
- XLS/XLSX - Microsoft Excel spreadsheet
- XML - Extensible Markup Language
- XRFF - Weka file format

Practical:-9:

Aim:-Demonstration of Data preprocessing.

Data preprocessing is a data mining technique which is used to transform the raw data in a useful and efficient format.

Steps Involved in Data Preprocessing:

1. Data Cleaning:

The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc.

(a) Missing Data:

This situation arises when some data is missing in the data. It can be handled in various ways. Some of them are:

- 1. Ignore the tuples:**

This approach is suitable only when the dataset we have is quite large and multiple values are missing within a tuple.

- 2. Fill the Missing values:**

There are various ways to do this task. You can choose to fill the missing values manually, by attribute mean or the most probable value.

(b) Noisy Data:

Noisy data is a meaningless data that can't be interpreted by machines. It can be generated due to faulty data collection, data entry errors etc. It can be handled in following ways :

- 3. Binning Method:**

This method works on sorted data in order to smooth it. The whole data is divided into segments of equal size and then various methods are performed to complete the task. Each segmented is handled separately. One can replace all data in a segment by its mean or boundary values can be used to complete the task.

- 4. Regression:**

Here data can be made smooth by fitting it to a regression function. The regression used may be linear (having one independent variable) or multiple (having multiple independent variables).

- 5. Clustering:**

This approach groups the similar data in a cluster. The outliers may be undetected or it will fall outside the clusters.

2. Data Transformation:

This step is taken in order to transform the data in appropriate forms suitable for mining process. This involves following ways:

- 1. Normalization:**

It is done in order to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0)

2. **Attribute Selection:**

In this strategy, new attributes are constructed from the given set of attributes to help the mining process.

3. **Discretization:**

This is done to replace the raw values of numeric attribute by interval levels or conceptual levels.

4. **Concept Hierarchy Generation:**

Here attributes are converted from level to higher level in hierarchy. For Example- The attribute “city” can be converted to “country”.

3. Data Reduction:

Since data mining is a technique that is used to handle huge amount of data. While working with huge volume of data, analysis became harder in such cases. In order to get rid of this, we use data reduction technique. It aims to increase the storage efficiency and reduce data storage and analysis costs.

The various steps to data reduction are:

1. **Data Cube Aggregation:**

Aggregation operation is applied to data for the construction of the data cube.

2. **Attribute Subset Selection:**

The highly relevant attributes should be used, rest all can be discarded. For performing attribute selection, one can use level of significance and p- value of the attribute. The attribute having p-value greater than significance level can be discarded.

3. **Numerosity Reduction:**

This enables to store the model of data instead of whole data, for example: Regression Models.

Practical:-10

Aim:-Introduction to Data mining tool (Tanagra).

TANAGRA is a free DATA MINING software for academic and research purposes. It proposes several data mining methods from exploratory data analysis, statistical learning, machine learning and databases area.

Tanagra is a free suite of machine learning software for research and academic purposes developed by Ricco Rakotomalala at the Lumière University Lyon 2, France. Tanagra supports several standard data mining tasks such as: Visualization, Descriptive statistics, Instance selection, feature selection, feature construction, regression, factor analysis, clustering, classification and association rule learning.

Tanagra is an academic project. It is widely used in French-speaking universities. Tanagra is frequently used in real studies and in software comparison papers.

Tanagra works similarly to current data mining tools. The user can design visually a data mining process in a diagram. Each node is a statistical or machine learning technique, the connection between two nodes represents the data transfer. But unlike the majority of tools which are based on the workflow paradigm, Tanagra is very simplified. The treatments are represented in a tree diagram. The results are displayed in an HTML format. This makes it is easy to export the outputs in order to visualize the results in a browser. It is also possible to copy the result tables to a spreadsheet.

Tanagra makes a good compromise between statistical approaches (e.g. parametric and nonparametric statistical tests), multivariate analysis methods (e.g. factor analysis, correspondence analysis, cluster analysis, regression) and machine learning techniques (e.g. neural network, support vector machine, decision trees, random forest).

TANAGRA is a free open source data mining software for academic and research purposes. It proposes several data mining methods from exploratory data analysis, statistical learning, machine learning and databases area. TANAGRA is more powerful, it contains some supervised learning but also other paradigms such as clustering, factorial analysis, parametric and nonparametric statistics, association rule, feature selection and construction algorithms. The main purpose of Tanagra project is to give researchers and students an easy-to-use **data mining software**, conforming to the present norms of the software development in this domain (especially in the design of its GUI and the way to use it), and allowing to analyse either real or synthetic data.

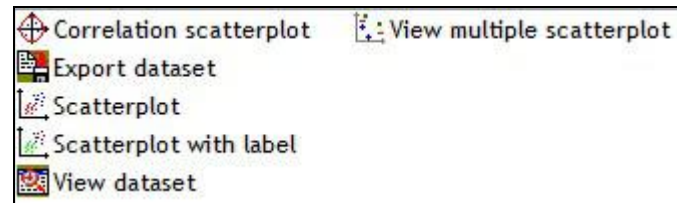
Features:

- Offers easy to use data mining software for researcher and students.
- It allows the user to add their data mining methods.

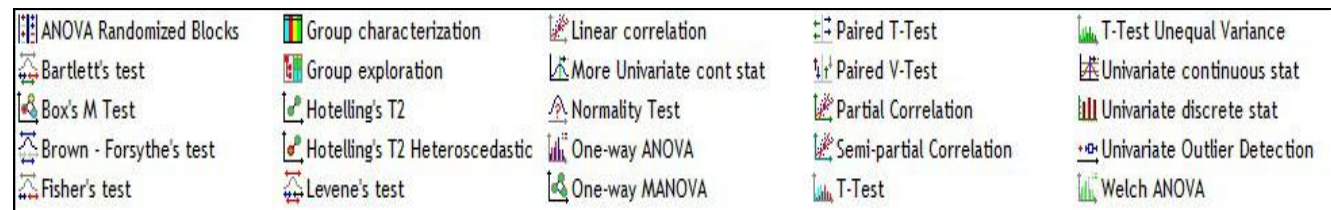
Component Categories

TANAGRA's has the following categories of components.

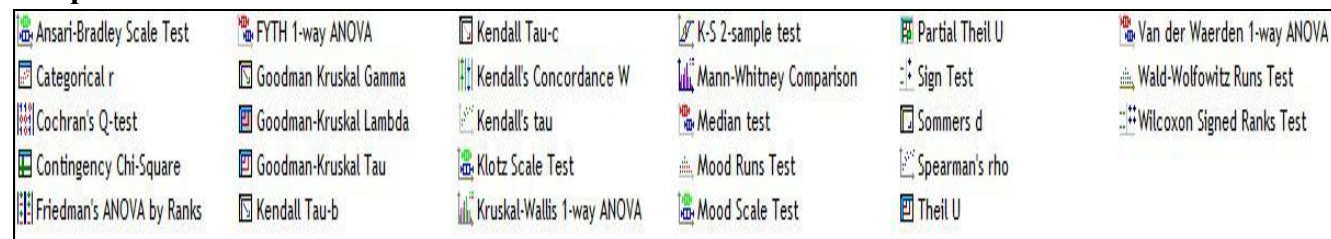
Data visualisation



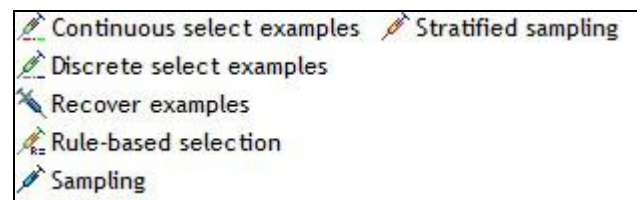
Statistics



Nonparametric statistics



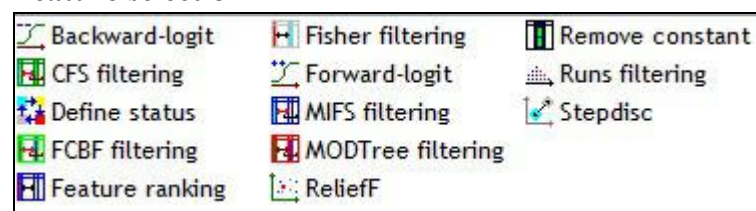
Instance selection



Feature construction



Feature selection



Regression

Backward Elimination Reg	Multiple linear regression
C-RT Regression tree	Nu SVR
DfBetas	Outlier Detection
Epsilon SVR	Regression Assessment
Forward Entry Regression	Regression tree

Factorial analysis

Canonical Discriminant Analysis	Principal Component Analysis
Correspondence Analysis	
Factor rotation	
Multiple Correspondence Analysis	
NIPALS	

PLS

PLS Conf. Interval
PLS Factorial
PLS Regression
PLS Selection
PLSR

Clustering

CT	K-Means	VARHCA
CTP	Kohonen-SOM	VARKMeans
EM-Clustering	LVQ	
EM-Selection	Neighborhood Graph	
HAC	VARCLUS	

Spv learning

Binary logistic regression	CS-MC4	Linear discriminant analysis	PLS-DA	Rule Induction
C4.5	C-SVC	Log-Reg TRIRLS	PLS-LDA	SVM
C-PLS	Decision List	Multilayer perceptron	Prototype-NN	
C-RT	ID3	Multinomial Logistic Regression	Radial basis function	
CS-CRT	K-NN	Naive bayes	Rnd Tree	







Meta-spv learning

Arcing [Arc-x4]	MultiCost
Bagging	Supervised Learning
Boosting	
Cost Sensitive Bagging	
Cost Sensitive Learning	

Spv learning assessment

Bias-variance decomposition	Logistic Regression Residuals
Bootstrap	Test
Cross-validation	Train-test
Hosmer Lemeshow Test	
Leave-One-Out	

Scoring

	Lift curve		↓ Scoring
	Posterior Prob		
	Precision-Recall curve		
	Reliability Diagram		
	Roc curve		

Association

	A priori		Spv Assoc Tree
	A priori MR		
	A priori PT		
	Assoc Outlier		
	Spv Assoc Rule		

Table of Contents

1. Introduction to data mining and tool and its installation.
2. Introduction to data warehousing.
3. KDD Process in Data Mining.
4. Demonstration of Association rule process on dataset using apriori algorithm.
5. Demonstration of classification rule process on dataset using id3 algorithm.
6. Demonstration of clustering rule process on dataset using simple k-means.
7. Installation and working of Big Data and Cloud tools.
8. Understanding files formats supported by the tool (Rapid Miner).
9. Demonstration of preprocessing.
10. Introduction to Data mining tool (Tanagra).