Assignment 1

- Problem Statement : For an organization of your choice, choose a set of business processes. Design star / snowflake schemas for analyzing these processes. Create a fact constellation schema by combining them. Extract data from different data sources, apply suitable transformations and load into destination tables using an ETL tool. For Example: Business Origination: Sales, Order, Marketing Process.
- Multi-dimensional Schema :
    - Used to model data warehouse systems
    - To address unique needs of very large databases for analytical purposes
- Types of Schemas :
    - Star
        - Simplest type
        - Structure resembles a star
        - Center of the star has 1 fact table and many dimension tables
        - Also called Star join and optimized for querying large data sets
        - Fact Table : Comprises of the measurements, metrics or facts of a business process
        - Dimension Table : A structure that categorizes facts and measures in order to enable users to answer business questions. Commonly used dimensions are people, products, place and time.
    - Snowflake
        - Extension of Star Schema
        - Additional dimensions
        - Dimension tables are normalized which splits data into additional tables
    - Constellation/Galaxy Schema
        - Contains 2 fact tables that share dimension tables
        - Collection of stars hence galaxy

Assignment 2

- Problem Statement : Consider a suitable dataset. For clustering of data instances in different groups, apply different clustering techniques (minimum 2). Visualize the clusters using suitable tools.
- Clustering :
    - Grouping a set of similar objects together
    - Main task of explorative data mining & statistical analysis
    - Used in ML, Pattern recognition, image analysis etc
- K means clustering
    - Clusters data into k groups where k is predefined
    - Algorithm
        - Select k random points as cluster centers
        - Assign points to cluster centers according to their Euclidean distances
        - Calculate the centroid in each cluster to get new center
        - Repeat the above steps until all points are assigned the best cluster
    - Use elbow method to find the most optimum k
    - Elbow Method :
        - Vary the k from 1-10
        - For each k calculate the WCSS (Within-Cluster Sum of Square)
        - WCSS is the sum of squared distance between each point and the centroid in a cluster
        - When we plot the WCSS with the K value, the plot looks like an Elbow. As the number of clusters increases, the WCSS value will start to decrease.
        - WCSS value is largest when K = 1. W
        - Then we analyze the graph we can see that the graph will rapidly change at a point and thus creating an elbow shape.
        - From this point, the graph starts to move almost parallel to the X-axis. The K value corresponding to this point is the optimal K value or an optimal number of clusters.

Assignment 3

- Problem Statement : Apply a-priori algorithm to find frequently occurring items from given data and generate strong association rules using support and confidence thresholds. For Example: Market Basket Analysis
- Accumulation of vast quantities of data
- Data Mining / Knowledge Discovery Database used to find anomalies, correlations, patterns and trends to predict outcomes
- Apriori Algorithm :
    - Used for mining frequent item sets and relevant association rules
    - Devised to operate on a database containing lot of transactions
    - Terms in Apriori :
        - min_length : min no of items we want in our rules
        - Support: Number of transactions containing a particular item divided by total number of transactions.
        - Confidence: Likelihood that B is bought if A is also bought.
        - Confidence(A-->B) = (Transactions containing both A and B)/(Transactions containing A)
        - Lift: increase in ratio of sale of B when A is sold.
        - Life = Confidence(A-->B)/Support(B)

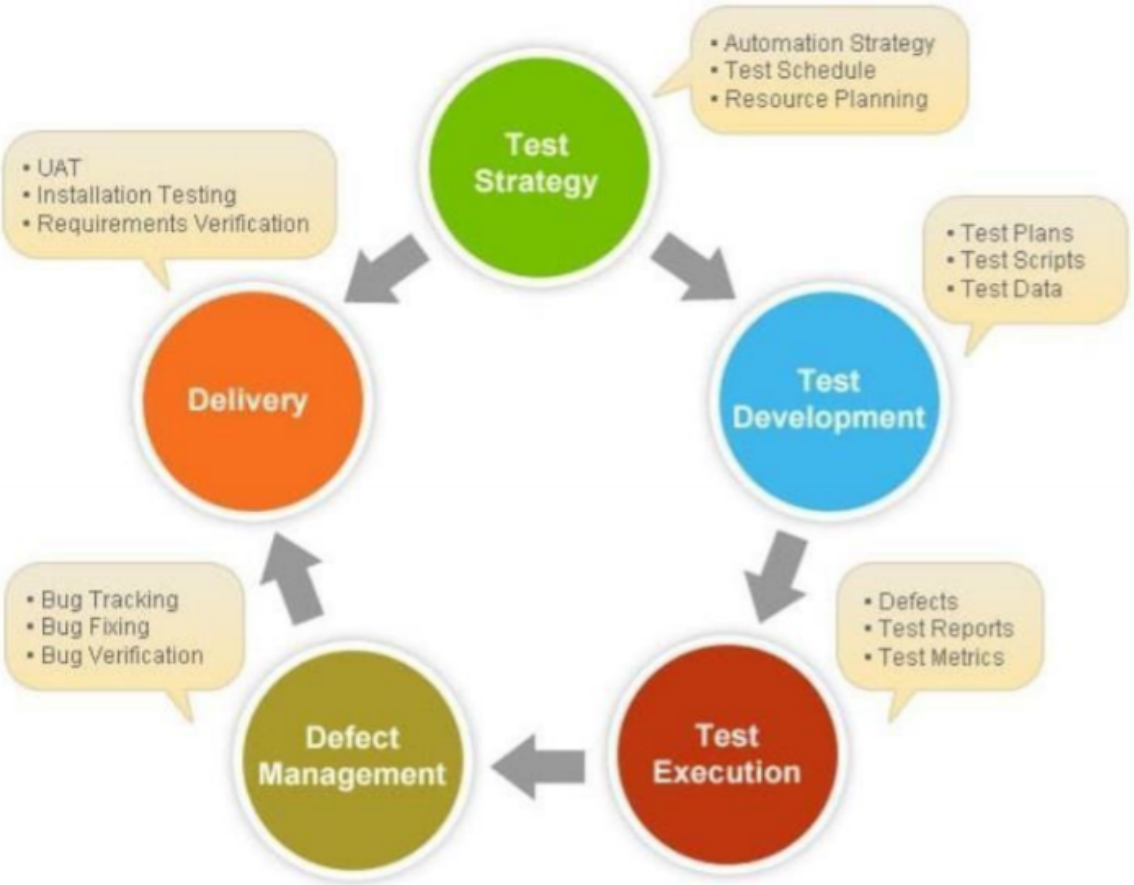
Assignment 4

- Problem Statement : Consider a suitable text dataset. Remove stop words, apply stemming and feature selection techniques to represent documents as vectors. Classify documents and evaluate precision, recall
- Feature selection :
    - Process of selecting a subset of the terms occurring in training set and using only this subset as features in text classification
    - Makes training and applying a classifier more efficient by decreasing the size of the effective vocabulary
    - Increases classification accuracy by eliminating noise features
- Stemming :
    - Reduce inflectional forms and derived words to a common base form
    - cleaning : removing @ # links
    - **Stopwords** - is,where, when, etc - propositions
    - **Stemmer** - stems the morphological part
    - **Countvectorizer** - count/frequency of each word
    - **Lemmatization** - dictionary based stemming

Testing

- Testing Methodology :
    - Exploratory (Black Box Testing) : Purpose is to make sure critical defects are removed before testing next levels of testing can start
    - Functional (White Box Testing) : Performed to check the functions of the application. Input is fed and output from the app is validated
    - User Acceptance Test (Integration) : Focuses on validating the business logic. Allows end user to complete one final review of the system prior to deployment

Process followed



-
- Testing Tools
    - TestNG is a testing framework inspired from JUnit and NUnit but with new functionalities which make it more power and easier to use
        - Annotations
        - Run your tests in arbitrarily big thread pools with various policies available (all methods in their own thread, one thread per test class,

        - Test that your code is multithread safe.
        - Flexible test configuration.
        - Support for data-driven testing (with @DataProvider).
        - Support for parameters.
        - Powerful execution model (no more TestSuite).
        - Supported by a variety of tools and plug-ins (Eclipse, IDEA, Maven)
        - Embeds BeanShell for further flexibility.
        - Default JDK functions for runtime and logging (no dependencies).
        - Dependent methods for application server testing.
- Selenium
    - Open source tool that automates web browsers
    - Single interface to run test scripts in multiple languages like Ruby, Java, NodeJS, Python, Perl, PHP, C#
    - Browser driver executes these scripts on a browser-instance on the device
    - Selenium WebDriver :
        - Also known as Selenium 2.0, WebDriver executes test scripts through browser- specific drivers. It consists of:
            - API : Application Programming Interface. Ports test scripts you write in Ruby, Java, Python, or C# to Selenese (Selenium's own scripting language), through bindings.
            - Library : Houses the API and language-specific bindings. Although plenty of third party bindings exist to support different programming languages, the core client-side bindings supported by the main project are: Selenium Java (as selenium jar files), Selenium Ruby, Selenium dotnet (or Selenium C#, available as .dll files), Selenium Python, and Selenium JavaScript (Node).
            - Driver : Executable module that opens up a browser instance and runs the test script. Browser-specific—for instance, Google develops and maintains Chromedriver for Selenium to support automation on Chromium/Chrome.
            - Framework : Support libraries for integration with natural or programming language test frameworks, like Selenium with Cucumber or Selenium with TestNG.
- JUnit
    - Unit testing framework for Java language
    - Originates from family of unit testing frameworks collectively known as xUnit
    - Features of JUnit :
        - JUnit is an open source framework, which is used for writing and running tests.
        - Provides annotations to identify test methods.
        - Provides assertions for testing expected results.
        - Provides test runners for running tests.
        - JUnit tests allow you to write codes faster, which increases quality.
        - JUnit is elegantly simple. It is less complex and takes less time.

        - JUnit tests can be run automatically and they check their own results and provide immediate feedback.
        - JUnit tests can be organized into test suites containing test cases and even other test suites.
        - JUnit shows test progress in a bar that is green if the test is running smoothly, and it turns red when a test fails
- Unit Test Case
    - Part of code which ensures another part of code (method) works as expected
    - Characterized by a known input and an expected output, which is worked out before the test is executed
    - This known input should test a precondition and the expected output should test a post condition