

**Selected topics in Computer Engineering**

**Computer and Systems Engineering Dept.**

**Ain Shams University**

**Presented to: Dr. Gamal AbdElShafy**

**Presented by:**

**Hamed Abdelreheem Fathallah**

**Hassan Mohamed Hassan ElMarsafy**

**Mohammad Okasha Mohammad Hammam**

**Ahmad Abdelhalem Mohammad Kamal**

**Joseph Samir Edwar Shaker Farahat**

**Salma Tarek Hassan Mohammad**

# Abstract

This report describes all the work related to the project. The used algorithm, functions and development platform details are described, in addition to all the testing cases and results.

# Table of Contents

[Abstract 2](#_Toc39983777)

[Table of Contents 3](#_Toc39983778)

[Introduction 4](#_Toc39983779)

[Purpose 4](#_Toc39983780)

[List of definitions 4](#_Toc39983781)

[Overview 4](#_Toc39983782)

[Beneficiaries 5](#_Toc39983783)

[Project Aims and Objectives 5](#_Toc39983784)

[Detailed Project Description 6](#_Toc39983785)

[Project Phases 8](#_Toc39983786)

[project analysis 8](#_Toc39983787)

[Design 8](#_Toc39983788)

[Implementation 8](#_Toc39983789)

[Testing 8](#_Toc39983790)

[Maintenance 8](#_Toc39983791)

[Documenting 8](#_Toc39983792)

[System Architecture: 9](#_Toc39983793)

[Development environment 10](#_Toc39983794)

[Testing cases and Results 10](#_Toc39983795)

[Conclusion 11](#_Toc39983796)

# Introduction

## Purpose

Data Analysis is a very important concept that has several benefits on many aspects, such as management and marketing. Nowadays, all decision makers take decisions based on the relations between elements in their domain which can be found using data analysis. This results in increasing their profits and advancing their marketing strategies.

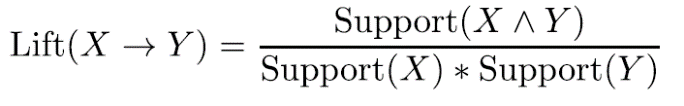
This project makes use of some data analysis techniques to find the relations in the given dataset.

## List of definitions

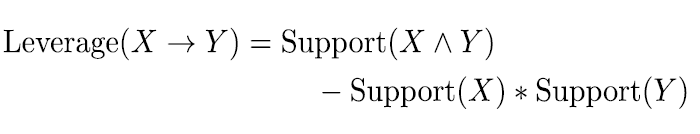
* Apriori Principle: If an itemset is frequent, then all of its subsets must also be frequent. The output of this algorithm/principle the set of all rules X --> Y with minimum support and confidence



* Inference rule: Describes the interesting relationships among variables in a large database, it usually takes the form of "When X observed, Y also observed".
* Confidence: for each inference rule X à Y, the confidence is the % of transactions that contain X, which also contain Y.
* Lift: measures how many times more often X and Y occur together than expected if they were statistically independent. It is a measure of how X and Y are really related rather than coincidentally happening together.



* Leverage: Leverage measures the difference in the probability of X and Y appearing together in the data set compared to what would be expected if X and Y were statistically independent.



* Itemset: A collection of one or more items
* k itemset: An itemset that contains k items
* Support count: Frequency of occurrence of an itemset
* Support: Fraction of transactions that contain an itemset
* Frequent Itemset: An itemset whose support is greater than or equal to a minimum support threshold

## Overview

This document will contain the full description of the implemented algorithm, the given dataset, the used libraries, the development platform, and the used test cases.

# Beneficiaries

Firstly, Institutions, companies, and decision makers are the primary beneficiaries from data analysis methods and techniques, as it is extremely vital in taking decisions to increase incoming profit and avoid financial loss.

Secondly, it has a necessary role regarding customers. For instance, it eases the process of taking decisions for customers too, it helps in enhancing the choice of the target customer, hence, it customizes recommendation systems based on the customers usual choices.

# Project Aims and Objectives

This project aims to understand the dataset of an insurance company and find all possible relations and association between all elements that make up the system, it does this by extracting all possible association rules depending on their support and confidence values. Also, it calculates the lift and leverage values of the extracted association rules.

The dataset we’re working on consists of 48 attributes and 5822 records, it includes attributes that describe the relation, financial status, education and background of the company’s customers.

# Detailed Project Description

We’ll discuss the implementation of the algorithm, in the displayed table, all implemented functions are illustrated.

|  |  |
| --- | --- |
| Function Declaration | Function Description |
| Level1(data, support) | **Parameters:**   * Data: the given data file * Support: the given minimum support   **Functionality:**   * Finds all the unique values in all attributes * Checks which of those attributes have a support value that exceeds the min. support   **Outputs:**   * List of all attributes that passed the min. support |
| Confidence(combinations, confidence) | **Parameters:**   * Combinations: combinations of attributes that have a support value >= the given support value * Confidence: given user value of confidence   **Functionality:**   * Makes rules out of the given combinations. * calculates the confidence for each one. * eliminates low confidence rules.   **Output:**   * Returns a list of rules |
| LL\_support(combination) | **Parameters:**   * Combination: list of attribute combinations that we need to calculate their support   **Functionality:**   * Calculate the support value   **Output:**   * Returns the support value |
| List\_combinations\_rules(singlerule) | **Parameters:**   * Singlerule: one combination of attributes   **Functionality:**   * The function takes the combination of attributes and generates all possible rules from it   **Output:**   * List of all rules |
| Calculate\_support(data, combinations\_arr, support) | **Parameters:**   * Data: the entire given data file * Combinations\_arr: combinations of attributes that can generate rules * Support: the given minimum support   **Functionality:**   * Checks combinations that exceed the minimum support   **Output:**   * Combinations of attributes with support >= min support * Attributes that are used in the next level |
| getCombinations(arr, n, r) | **Parameters:**   * Arr: array of the attributes that will be combined * N: size of the given array of attributes * R: size of combinations to be generated   **Functionality:**   * The main function that gets all combinations of size r in arr of size n. This function mainly uses combinationUtil()   **Output:**   * Returns all combinations of attributes |
| CombinationUtil(combinations , arr, data, start, end, index , r) | **Parameters:**   * Arr: Input Array * data: Temporary array to store current combination * start & end :Staring and Ending indexes in arr[] * index : Current index in data[] * r: Size of a combination   **Functionality:**   * Recursively traverses the array ‘arr’ to fill ‘combinations’ with all possible combinations of given size ‘r’   **Output:**   * None |
| apriori\_alg(data, support, confidence) | **Parameters:**   * Data: the entire given data file * Support: the given minimum support * Confidence: the given minimum confidence   **Functionality:**   * Analyzes the data to generate the inference rules   **Output:**   * Prints the inference rules |

# Project Phases

## project analysis

We’ve set several meetings to discuss the requirements, the programming language we will be using, the system architecture, how the tasks will be distributed on us, and how the data will be processed and handled.

## Design

In this phase, we tried to specify the system’s elements and units. Hence, we tried to imagine a suitable pipeline for our project. This pipeline was then divided into tasks on each team member.

## Implementation

After building the system’s pipeline and architecture, we started implementing the code using python programming language, specifically, to make the best use of Pandas and Numpy libraries. Implementation took approximately 6 days of continuous work.

## Testing

Testing is the most critical phase of the project, we initiated testing on several phases, where each team member would perform unit test on his code after finishing it. Lastly, a system test was held by all team members over a zoom meeting.

## Maintenance

After the testing phase, we discovered some bugs, so we had to start debugging our code and fixing those bugs.

## Documenting

After finishing all the main project’s phases, we started working on the documentation of the project. Documentation was broken down into several tasks on the team members where each member would handle a specific part of the report.

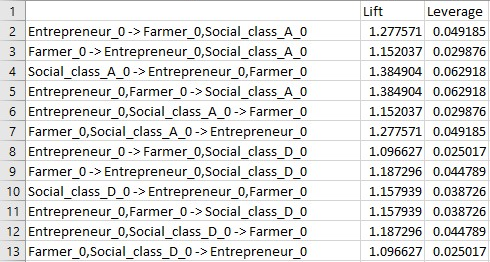
# System Architecture:

# Development environment

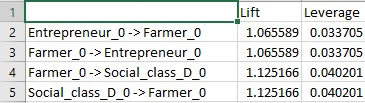
* For Language we used python.
* For the operating system: windows 10
* For the IDE, we use multiple IDEs:
  + Sublime Text 3.
  + Visual studio code.

# Testing cases and Results

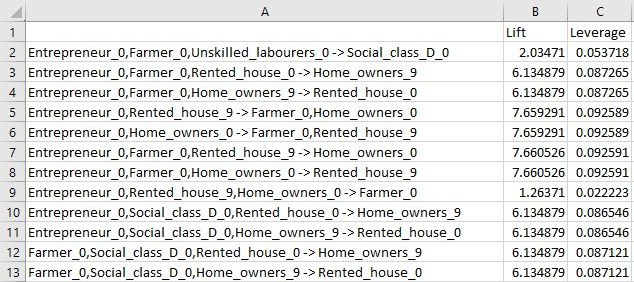
1. Support = 20% , confidence = 5%



1. Support = 35%, confidence = 9%



1. Support = 10 %, confidence = 90%



# Conclusion

In conclusion, we’ve practically discovered the advantages of using Apriori Algorithm in data analysis. We’ve learned that:

* This is the most simple and easy-to-understand algorithm among association rule learning algorithms
* he resulting rules are intuitive and easy to communicate to an end user
* The algorithm is exhaustive, so it finds all the rules with the specified support and confidence
* It doesn't require labeled data as it is fully unsupervised; as a result, you can use it in many different situations because unlabeled data is often more accessible

While, the cons of Apriori are:

* Requires many database scans
* Very slow

<https://www.youtube.com/watch?v=z7b3SYQuqJM&fbclid=IwAR2d3dYuWufsgk34fkC1eoKOrD8wTXWbD_eNlOk-z-0KoCQivmdFC1xueT0>