# MARKET BASKET ANALYSIS

## 1. ABSTRACT

Market Basket Analysis (MBA), is an important part of the analytical system in the retail organization to determine the placement of goods, designing sales promotion for different segments of customers to improve customer satisfaction and hence the profit of the supermarkets, Market Basket Analysis is well known activity of Association Rule Mining ultimately used for business intelligent decisions. Mining frequent item sets and hence deduce the rules to build classifiers with good accuracy is essential for efficient algorithm. The issues for leading supermarket are addressed here using frequent item set mining.

The project uses file as database. The frequent item sets are mined for database using Apriori Algorithm.

The project is beneficial for supermarket managers to determine the relationship between the items that are purchased by their customers.

**Keywords:** Market Basket Analysis, Association Rule Mining, Apriori Algorithm

## 2. INTRODUCTION

Market Basket Analysis is a data mining method focusing on discovering purchase patterns of the customers by extracting association or co-occurrences from a store's transactional data. For example, when a person checkout items in a supermarket all the details about their purchase goes into the transactional database D used in various industries besides supermarkets determined.

Association rule mining (ARM) identifies the association or relationship between a large set of data items and forms the base for market basket analysis. Association rule mining has been widely used in various industries besides supermarkets, such as mail order, telemarketing production, fraud detection of credit card and e-commerce.

One of the challenges for companies that have invested heavily in customer data collection is how to extract important information from vast customer database and product feature databases, in order to gain competitive advantage. Market Basket analysis has been intensively used in many companies as a means to discover product associations.

A retailer must know the needs of customers and adopt them. Market Basket Analysis is one possible way to find out which items can be put together.

Market basket analysis helps to identify the purchasing behaviour of the customer. By mining the data from the huge transaction database shop managers can study the behaviour or buying habits of the customer to increase the sale. In Market Basket Analysis, you look to see if there are combinations of products that frequently co-occur in a transaction.

# 3. LITERATURE REVIEW

Data Mining provides a lot of opportunities in the market sector. Decision making and understanding the behaviour of the customer has become vital and challenging problem for the organization in order to sustain in this competitive environment. The challenges that the organization faces is to extract the information from their vast customer databases, in order to gain competitive advantage.

This project states about the important goal in data mining is to reveal hidden knowledge from data and various algorithms have been proposed for, but the problem is that typically not all rules are interesting only small fraction of the generated rules would be of interest to any given users. Hence numerous methods such as confidence, support, and lift have been proposed to determine the best or most interesting rules.

However some algorithms are good at generating rules high in one measure but bad in other. Apriori was the first associative algorithm proposed and future development in association, classification, associative classification algorithms has used apriori as part of the technique. Apriori algorithm is a level-wise, breadth-first algorithm which counts transactions Apriori algorithm uses prior knowledge of frequent item set properties. Apriori uses an iterative approach known as a level-wise search, in which n-item sets are used to explore (n+1) - item sets.

To improve the efficiency of the level-wise generation of frequent item sets Apriori property is used here. Apriori property insists that all non-empty subsets of a frequent item set must also be frequent. This is made possible because of the anti-monotone property of support measure - the support for an item set never exceeds the support for its subsets.

A two-step process consists of join and prune actions are done iteratively It is one of the Data Mining Algorithm which is used to find the frequent items/item set from a given data repository.

The algorithm involves 2 steps

Pruning

Joining

The Apriori property is the important factor to be consider before proceeding with the algorithm. Apriori property states that If an item X is joined with item Y,

**Support (XUY) =min (Support(X), Support(Y))**

Basically when we are determining the strength of an association rule i.e. how strong the relationship is between the transaction of the items we measure through the use of the support and confidence. The support of an item is the number of transaction containing the item. Those items that do not meet the minimum support are excluded from the further processing. Support determines how often a rule is applicable to a given data set. Confidence is defined as the conditional probability that a transaction containing the LHS will also contain the RHS.

**Confidence(LHS>RHS>P(RHS/LHS)=P(RHS∩LHS)/P(LHS)=support(RHS∩LHS)/support(LHS)**

Confidence determines how frequently item in RHS appears in the transaction that contains LHS. While determining the rules we must measure these two components as it is very important to us. A rule that has very low support may occur simply by chance. Confidence on the other hand, measures the reliability of the inference made by the rule. CBA(Classification Based on Association) the first Associative Classification(AC), implements the famous Apriori algorithm in order to discover frequent rule items. The Apriori algorithm consists of three main steps.

a. Continuous attribute in the training data set gets discredited.
b. Frequent rule items discovery
c. Rule generation

CBA selects high confidence rules to represent the classifier. Finally, to predict a test case CBA applies the highest confidence rule whose body matches the test case. Experimental result designated that CBA drives higher quality classifiers with regards to accuracy that rule induction and decision tree classification approaches. This project states that there are lots of case studies about the a Market Basket Analysis and association Rules and existing data mining algorithms usage for market basket analysis but focuses on Apriori algorithm and concludes that the algorithm can be modified and it can be extended in the future work which also decrease the time complexity.

**Demerits**

a. It scans the database lot of times i.e. every time it runs it scans database every time, this results in shortage of memory to store the data.

b. The I/O load is not sufficient therefore it takes a lot of time to process and exhibits low efficiency.

c. The time complexity is very high.

**Solution to improve efficiency**

a. Group items into higher conceptual groups e.g. white and brown bread becomes “bread”.

b. Reduce the number of scan of the entire database.

# 4. SYSTEMANALYSIS

## 4.1. Problem Statement

Nowadays people buy daily goods from super market nearby. There are many supermarkets that provide goods to their customer. The problem many retailers face is the placement of the items. They are unaware of the purchasing habits of the customer so they don't know which items should be placed together in their store. With the help of this application shop managers can determine the strong relationships between the items which ultimately helps them to put products that co-occur together close to one another. Also decisions like which item to stock more, cross selling, up selling, store shelf arrangement are determined.

## 4.2. Apriori Algorithm

Association rule mining finds interesting associations and/or correlation relationships among large set of data items. Association rules shows attribute value conditions that occur frequently together in a given dataset. A typical and widely used example of association rule mining is Market Basket Analysis.

For example, data are collected from the supermarkets. Such market basket databases consist of a large number of transaction records. Each record lists all items bought by a customer on a single purchase transaction. Association rules provide information of this type in the form of "IF-THEN" statements. The rules are computed from the data, an association rule has two numbers that express the degree of uncertainty about the rule.

a. Support

b. Confidence

**a. Support**

The support of an item is the number of transaction containing the item. Those items that do not meet the minimum support are excluded from the further processing. Support determines how often a rule is applicable to a given data set.

**Support (XUY) =min (Support(X), Support(Y))**

**b. Confidence**

Confidence is defined as the conditional probability that a transaction containing the LHS will also contain the RHS.

**Confidence(LHS>RHS>P(RHS/LHS)=P(RHS∩LHS)/P(LHS)=support(RHS∩LHS)/support(LHS).**

Confidence determines how frequently item in RHS appears in the transaction that contains LHS. While determining the rules we must measure these two components as it is very important to us. A rule that has very low support may occur.

## 4.3. Algorithm

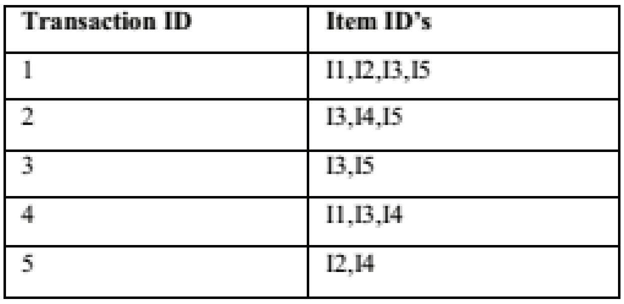
Steps:

**Pseudo code**

```
//Find all frequent item set
Apriori(database D of transaction, min_support){
F1={frequent 1-itemset}
K=2
While Fk-1≠ Empty Set
Ck=AprioriGeneration (Fk-1)//Generate candidate item sets.
For each transaction in the database D {
Ct=subset (Ck, t)
For each candidate c in Ct{

Count c++

}

Fk={c in Ck such that countc&gt;min_support}

K++

}

F=U K&gt;Fk

}

//prune the candidate item sets

Apriori generation (Fk-1) {

//Insert into Ck all combination of elements in Fk-1 obtained by self-joining item

sets in Fk-1

//Delete all item sets c in Ck such that some (K-1) subset of c is not in Lk-1

}

//find all subsets of candidate contained in t

Subset (Ck, t)

}
```

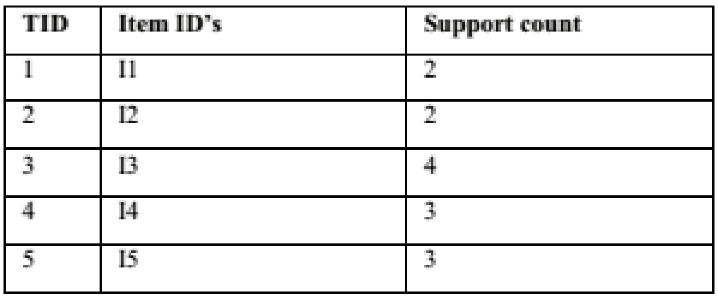
**Example:**

Assume that we are having 5 transactions in a Database i.e., D=5



| Transaction ID | Item ID's |
|---|---|
| 1 | I1,I2,I3,I5 |
| 2 | I3,I4,I5 |
| 3 | I3,I5 |
| 4 | I1,I3,I4 |
| 5 | I2,I4 |

**Step1-** Find frequent items in above transactions to count support of each item



| TID | Item ID's | Support count |
|---|---|---|
| 1 | I1 | 2 |
| 2 | I2 | 2 |
| 3 | I3 | 4 |
| 4 | I4 | 3 |
| 5 | I5 | 3 |

**Step 2-** Join the table with itself

| TID | Item ID's | Support Count |
|---|---|---|
| 1 | I1, I2 | 1 |
| 2 | I1, I3 | 2 |
| 3 | I1, I4 | 1 |
| 4 | I1, I5 | 1 |
| 5 | I2, I3 | 1 |
| 6 | I2, I4 | 1 |
| 7 | I2, I5 | 1 |
| 8 | I3, I4 | 2 |
| 9 | I3, I5 | 3 |
| 10 | I4, I5 | 1 |

**Step 3-** In this step we are pruning the above table on the basis of association rule which pursue equal or more than 2 support counts. But above table contains many item set having less than 2 support counts. So here pruning is required. After pruning the resultant table will be as

| TID | Item ID’s | Support Count |
|---|---|---|
| 1 | I1, I3 | 2 |
| 2 | I3, I4 | 2 |
| 3 | I3, I5 | 3 |

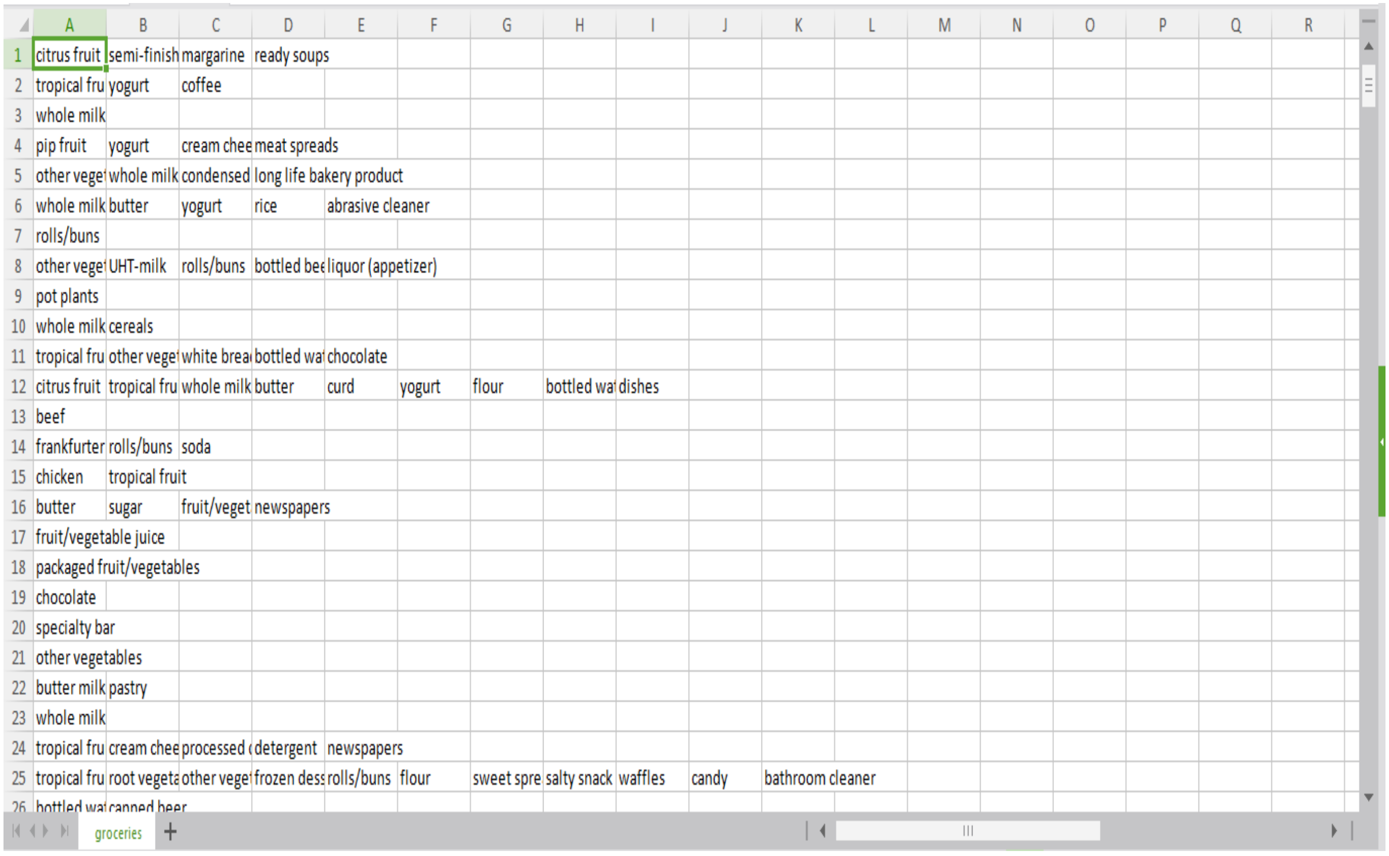
**Step 4-** Join the table with itself

| TID | Item ID’s | Support Count |
|---|---|---|
| 1 | I1, I3, I4 | 1 |
| 2 | I3, I4, I5 | 1 |

If we compare with support of the above table’s item set with minimum support (i.e., 2) then none of the transaction sets are qualified.

The result of applying APRIORI algorithm on above item sets with minimum support=2. So we get 3 frequent item sets as {I1, I3}, {I3, I4}, {I3, I5}.

## 4.4. Data Description



| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | citrus fruit | semi-finish | margarine | ready soups | | | | | | | | | | | | | | |
| 2 | tropical fru | yogurt | coffee | | | | | | | | | | | | | | | |
| 3 | whole milk | | | | | | | | | | | | | | | | | |
| 4 | pip fruit | yogurt | cream chee | meat spreads | | | | | | | | | | | | | | |
| 5 | other veget | whole milk | condensed | long life bakery product | | | | | | | | | | | | | | |
| 6 | whole milk | butter | yogurt | rice | abrasive cleaner | | | | | | | | | | | | | |
| 7 | rolls/buns | | | | | | | | | | | | | | | | | |
| 8 | other veget | UHT-milk | rolls/buns | bottled bee | liquor (appetizer) | | | | | | | | | | | | | |
| 9 | pot plants | | | | | | | | | | | | | | | | | |
| 10 | whole milk | cereals | | | | | | | | | | | | | | | | |
| 11 | tropical fru | other veget | white brea | bottled wat | chocolate | | | | | | | | | | | | | |
| 12 | citrus fruit | tropical fru | whole milk | butter | curd | yogurt | flour | bottled wat | dishes | | | | | | | | | |
| 13 | beef | | | | | | | | | | | | | | | | | |
| 14 | frankfurter | rolls/buns | soda | | | | | | | | | | | | | | | |
| 15 | chicken | tropical fruit | | | | | | | | | | | | | | | | |
| 16 | butter | sugar | fruit/veget | newspapers | | | | | | | | | | | | | | |
| 17 | fruit/vegetable juice | | | | | | | | | | | | | | | | | |
| 18 | packaged fruit/vegetables | | | | | | | | | | | | | | | | | |
| 19 | chocolate | | | | | | | | | | | | | | | | | |
| 20 | specialty bar | | | | | | | | | | | | | | | | | |
| 21 | other vegetables | | | | | | | | | | | | | | | | | |
| 22 | butter milk | pastry | | | | | | | | | | | | | | | | |
| 23 | whole milk | | | | | | | | | | | | | | | | | |
| 24 | tropical fru | cream chee | processed ( | detergent | newspapers | | | | | | | | | | | | | |
| 25 | tropical fru | root vegeta | other veget | frozen dess | rolls/buns | flour | sweet spre | salty snack | waffles | candy | bathroom cleaner | | | | | | | |
| 26 | bottled wat | canned beer | | | | | | | | | | | | | | | | |

**Fig 4.4 Description of Data**

The data set considered here is the Groceries data set downloaded from GITHUB.

The data set contains 1 month (30 days) of real-world point-of-sale transaction data from a typical local grocery outlet. The data set contains 9835 transactions and the items are aggregated to 169 different categories.

# 5. SYSTEM REQUIREMENTS AND SPECIFICATIONS

**Software Requirements**

| | |
|---|---|
| Environment | Windows |
| Database | Microsoft Excel |
| Software | PYCHARM |
| Language | PYTHON |

**Hardware Requirements**

| | |
|---|---|
| Disk space | 1GB |
| Processor | Any Intel or AMBx86-64 |
| Ram | Minimum 4GB |
| Graphics | No specific graphics card required |

**Non-Functional Requirements User Friendly**

The system will have an attractive and easy-to- use interface, so that users will not be confused or frustrated when using the system. The navigational systems are not too complex and are easy for non-technical user to understand.

**Flexibility**

The system should have the capability to take advantage of new technologies and resources. The system should be able to be implemented in the changing environment.

**Reliability**

Reliability is the extent to which a system can be expected to perform its intended function with required precision and accuracy. Thus, the system should be reliable in performing its functions and operation to the users. This also will convince the user that the system will make the correct respond and provide error handling ability.

**Usability**

Usability places the user in control. The application system must be easy to use. They can enhance and support rather than limit or restrict business process. Non-technical users should be able to handle the software with ease and in a straightforward manner.

**Efficiency**

This is one of the most important requirements of the system, where it should provide a good response time to all user requests. The system should not cause any delay in processing the user request or even in the midst of retrieving information.

**Maintainability and Expandability**

Maintainability maybe defined qualitatively as the ease with which software can be understood, corrected, adapted or enhanced. Expandability is the degree to which architectures, data or procedural design can be extended. This system can be expanded in the future.

# 6. SYSTEM MODELLING AND DESIGN

## 6.1. UML NOTATIONS

UML is a notation that resulted from the unification of Object Modelling Technique, Booch and OOSE. UML has been designed for a broad range of applications. Hence, it provides constructs for a broad range of systems and activities.UML is popular for its diagrammatic notations. We all know that UML is for visualizing, specifying, constructing and documenting the components of software and non-software systems. Hence, visualization is the most important part which needs to be understood and remembered.

UML notations are the most important elements in modelling. Efficient and appropriate use of notations is very important for making a complete and meaningful model. The model is useless, unless its purpose is depicted properly.

Hence, learning notations should be emphasized from the very beginning. Different notations are available for things and relationships. UML diagrams are made using the notations of things and relationships. Extensibility is another important feature which makes UML more powerful and flexible.

Graphical notations used in structural things are most widely used in UML. These are considered as the nouns of UML models. Following are the list of structural things.

- Classes
- Object
- Interface
- Collaboration
- Use case
- Active classes
- Components
- Nodes

### 6.1.1. Use Case Diagram

Use Case is used during requirements elimination and analysis to represent the functionality of the system. Use cases focus on the behaviour of the system from an external point of view.

A use case describes a function provided by system that yields a visible result for an actor. An actor describes any entity that interacts with the system.

The actors are outside the boundary of the system, whereas the use cases are inside the boundary of the system. Actors are represented with stick figures, use case with ovals, and the boundary of the system with a box enclosing use cases.

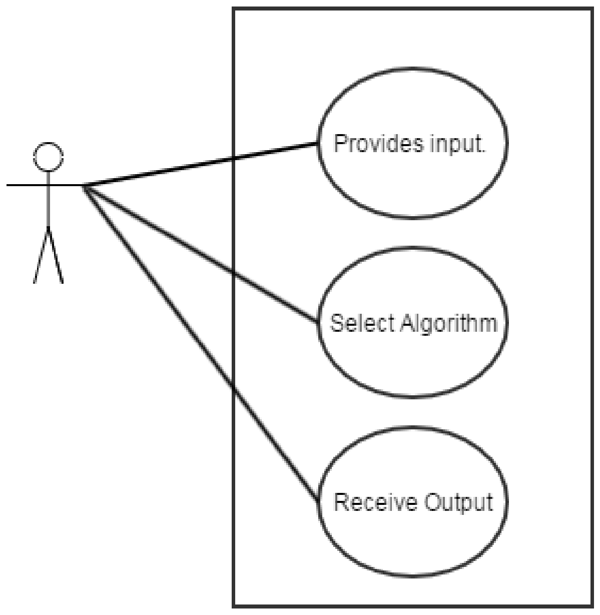


**Fig 6.1.Use Case Diagram**

## 6.1.2. Deployment Diagram

Deployment diagram is a structure diagram which shows architecture of the system as deployment (distribution) of software's artefacts to deployment targets. Artefact's represent concrete elements in the physical world that are the result of a development process.

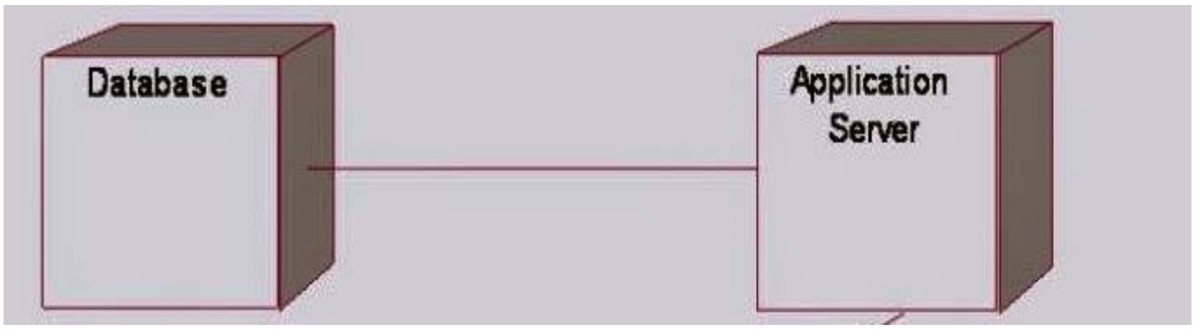


**Fig 6.2.Deployment Diagram**

## 6.1.3. Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function.

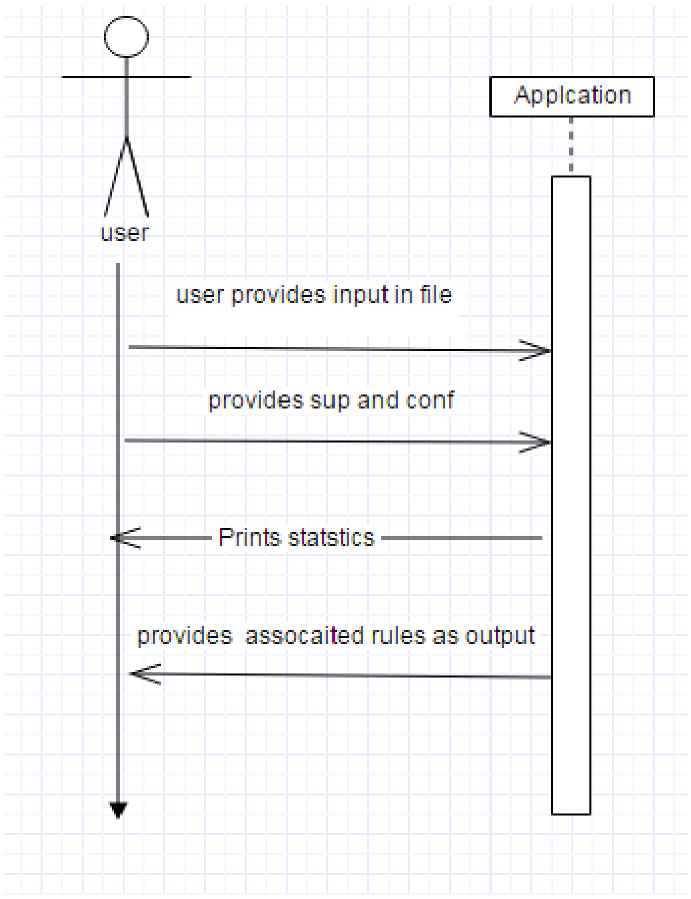

user
Applcation
user provides input in file
provides sup and conf
Prints statstics
provides assocaited rules as output

**Fig 6.3 Sequence Diagram**

### 6.1.4. Class Diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

The class diagram is the main building block of object-oriented modelling. It is used for general conceptual modelling of the systematic of the application, and for detailed modelling translating the models into programming code. Class diagrams can also be used for data modelling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

In the diagram, classes are represented with boxes that contain three compartments:

The top compartment contains the name of the class. It is printed in bold and centred, and the first letter is capitalized.

The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.

The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

In the design of a system, a number of classes are identified and grouped together in a class diagram that helps to determine the static relations between them. With detailed modelling, the classes of the conceptual design are often split into a number of subclasses.

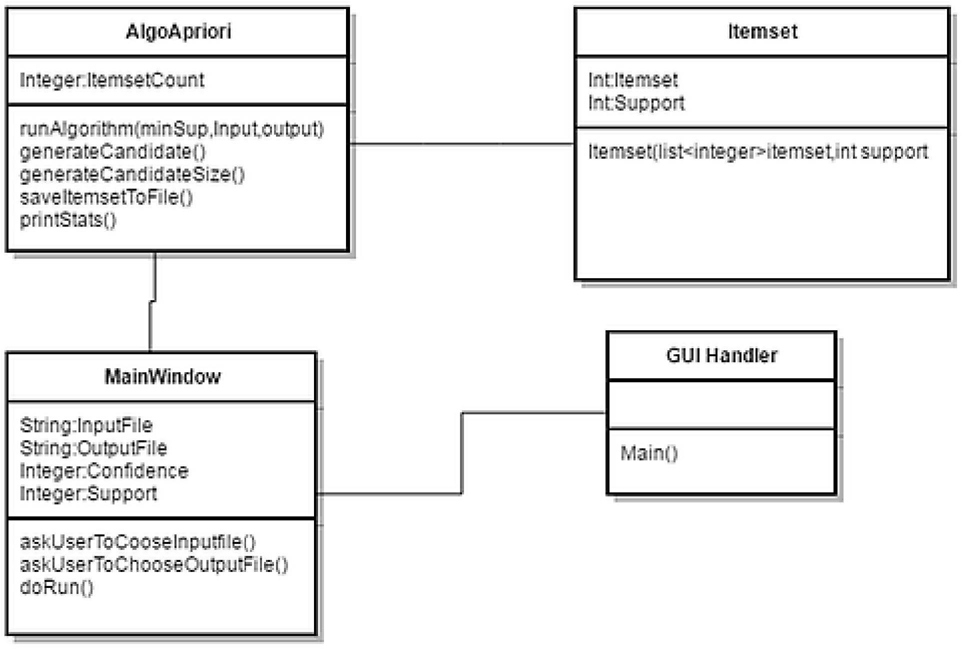


**Fig 6.4 Class Diagram**

# 7. FEASIBILITYSTUDY

## 7.1. Economic feasibility

Economic feasibility attempts 2 weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system. A simple economic analysis which gives the actual comparison of costs and benefits are much more meaningful in this case. In addition, this proves to be a useful point of reference to compare actual costs as the project progresses. The project invariably increases the accuracy of the classification of data. Thereby reducing the need for reclassification of data and this algorithm proves to be more efficient than the traditional algorithms and take less time to operate and produce the output.

## 7.2. Operational Feasibility

The project is operationally feasible as the user having basic knowledge about computer and Internet can use. Furthermore the project can be easily used if the computers have no internet access.

## 7.3. Technical Feasibility

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at this point in time, not too many detailed design of the system, making it difficult to access issues like performance, costs etc. A number of issues have to be considered while doing a technical analysis. Understand the different technologies involved in the proposed system before commencing the project we have to be very clear about what are the technologies that are to be required for the development of the new system. Find out whether the organization currently possesses the required technologies. Is the required technology available with the organization? The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

# 8. SYSTEM IMPLEMENTATION

## 8.1. About PYCHARM

With PyCharm you can develop applications in Python. In addition, in the Professional edition, one can develop Django, Flask and Pyramid applications. Also, it fully supports HTML (including HTML5), CSS, JavaScript, and XML: these languages are bundled in the IDE via plugins and are switched on for you by default. Support for the other languages and frameworks can also be added via plugins (go to Settings | Plugins or PyCharm | Preferences | Plugins for macOS users, to find out more or set them up during the first IDE launch).

Many programmers nowadays opt for Python to build software applications with concise, clean, and readable code base. They can even accelerate custom software application development by taking advantage of a number of integrated development environments (IDEs) for Python. PyCharm is one of the most widely used IDEs for Python programming language. At present, the Python IDE is being used by large enterprises like Twitter, Pinterest, HP, Symantec and Group on.

JetBrains has developed PyCharm as a cross-platform IDE for Python. In addition to supporting versions 2.x and 3.x of Python, PyCharm is also compatible with Windows, Linux, and macOS. At the same time, the tools and features provided by PyCharm help programmers to write a variety of software applications in Python quickly and efficiently. The developers can even customize the PyCharm UI according to their specific needs and preferences. Also, they can extend the IDE by choosing from over 50 plug-ins to meet complex project requirements.

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains . It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features, released under a proprietary license.

The beta version was released in July 2010, with the 1.0 arriving 3 months later. Version 2.0 was released on 13 December 2011, version 3.0 on 24 September 2013, and version 4.0 on November 19, 2014.

**Features of PYCHARM**

PyCharm is a Python IDE with complete set of tools for Python development. In addition, the IDE provides capabilities for professional Web development using the Django framework. Code faster and with more easily in a smart and configurable editor with code completion, snippets, code folding and split windows support.

- **Project Code Navigation** - Instantly navigate from one file to another, from method to its declaration or usages, and through classes hierarchy. Learn keyboard shortcuts to be even more productive
- **Code Analysis** - Take advantage of on-the-fly code syntax, error highlighting, intelligent inspections and one-click quick-fix suggestions to make code better
- **Python Refactoring** - Make project-wide code changes painlessly with rename, extract method/super class, introduce field/variable/constant, move and pull up/push down refactoring
- **Web Development with Django** - Even more rapid Web development with Django framework backed up with excellent HTML, CSS and JavaScript editors. Also with CoffeeScript, Mako and Jinja2 support
- **Google App Engine Support** - Develop applications for Google App Engine and delegate routine deployment tasks to the IDE. Choose between Python 2.5 or 2.7 runtime
- **Version Control Integration** - Check-in, check-out, view diffs, merge — all in the unified VCS user interface for Mercurial, Subversion, Git, Perforce and other SCMs
- **Graphical Debugger** - Fine-tune Python or Django applications and unit tests using a full-featured debugger with breakpoints, stepping, frames view, watches and evaluate expressions
- **Integrated Unit Testing** - Run a test file, a single test class, a method, or all tests in a folder. Observe results in graphical test runner with execution statistics
- **Customizable & Extensible** - Bundled Textmate, NetBeans, Eclipse & Emacs keyboard schemes, and Vi/Vim emulation plugin

## 8.2. About Microsoft Excel

Microsoft Excel is a spreadsheet program that is used to record and analyse numerical data. Think of a spreadsheet as a collection of columns and rows that form a table. Alphabetical letters are usually assigned to columns and numbers are usually assigned to rows. The point where a column and a row meet is called a cell. The address of a cell is given by the letter representing the column and the number representing a row.

We all deal with numbers in one way or the other. We all have daily expenses which we pay for from the monthly income that we earn. For one to spend wisely, they will need to know their income vs. expenditure. Microsoft Excel comes in handy when we want to record, analyze and store such numeric data.

Following are the few things which it can do for you.

- Number Crunching
- Charts and Graphs
- Store and Import Data
- Manipulating Text
- Templates/Dashboards
- Automation of Tasks

## 9. CODING

The following section gives the different sample segments of the source code. The code for the program was written in PYCHARM.

```
import tkinter.filedialog
from tkinter import *
import time
import itertools
from heapq import nlargest
import matplotlib.pyplot as plt;
plt.rcdefaults()
import numpy as np
import matplotlib.pyplot as plt

def ap(sts,z,qqq):
    l={}
    qww=0
    for x in sts:
        for y in x.split():
            if (y not in l):
                l[y]=1
            else:
                l[y]+=1
    list1={}
    for key in l:
        if l[key] > (z):
            list1[key] = l[key]
    print("list")
    print(list1)
    zz = nlargest(10, list1, key=list1.get)
    qww=qww+1
    p=[]
    q=[]
    with open("ttt.txt","w") as f:
        for val in zz:
            f.write(""+val+" : "+ str(list1.get(val))+"\n")
            #print(val, ":", list1.get(val))
            p.append(list1.get(val))
            q.append(val)
        objects = ('C', 'C++', 'Java', 'Python', 'Lisp', 'k', 'j', 'h', 'p', 'i')
        y_pos = np.arange(len(q))
        performance = [10, 8, 6, 4, 2, 1]
        if (qww==qqq):
            plt.bar(y_pos, p, align='center', alpha=1)
            plt.xticks(y_pos, q)
            plt.ylabel('Usage')
            plt.title('Frequent item sets')

            plt.show()
        else:
            pass
        print("sa")
        L = [list1]
        k = 0
        while (len(L[k]) > 0):
            Ck = {}
            unique = []
            for key in L[k]:
                for item in key.split(','):
                    if item not in unique:
                        unique.append(item)
            for key in itertools.combinations(unique, k + 2):
                if key not in Ck:
                    Ck[','.join(key)] = 0

            print(" saddd")
            for transactions in sts:
                for subset in set(itertools.permutations(transactions.split(), (k + 2))):
                    candidate = ','.join(subset)  # tuple to string
                    for keys in Ck:
                        if candidate == keys:
                            Ck[keys] += 1

            print("mmmm ")
            Lk = {}
            for key in Ck:
                if Ck[key] > (z):
                    Lk[key] = Ck[key]

            print(" nnnnnnmmm")
            print("L%s with count" % (k + 2))
            print(len(Lk))
            zzz = nlargest(4, Lk, key=Lk.get)
            qww=qww+1
            p = []
            q = []
            print("nnnn")
            for val in zzz:
                print(val, ":", Lk.get(val))
                p.append(Lk.get(val))
                q.append(val)
            objects = ('C & C++', 'C & Python', 'C & java', 'c++ and java')
            y_pos = np.arange(len(q))
            performance = [10, 8, 6, 4, 2, 1]

            '''plt.barh(y_pos, p, align='center', alpha=1)
```

```
        plt.yticks(y_pos, q)
        plt.xlabel('Usage')
        plt.title('Programming language usage')'''
        if (qww==qqq):
            plt.bar(y_pos, p, align='center', alpha=0.5)
            plt.xticks(y_pos, q)
            plt.ylabel('Usage')
            plt.title('Frequent item sets')

            plt.show()
        else:
            pass

        print(k + 2)
        statusText.set("These are the frequency of elements with element")
        print("sanjay ")

        L.append(Lk)

        k += 1
        f.close()


def button_go_callback():
    print("batch 16")
    confidence_threshold = float(entry_confidence.get())
    support_threshold = float(entry_support.get())
    print(confidence_threshold)

def button_browse_callback():
    filename = tkinter.filedialog.askopenfilename()
    # filename = tkFileDialog.askopenfilename()
    entry_filename.delete(0, END)
    entry_filename.insert(0, filename)


def button_viewinput_callback():
    input_file = entry_filename.get()
    text.delete(1.0, END)
    with open(input_file, 'r') as file_reader:
        for line in file_reader:
            text.insert(INSERT, line)
            text.insert(INSERT, "\n")
    statusText.set(" Transactions Displayed Successfully")

def go() :
    a = []
    b = ""
    print(type(b))
    inp = entry_filename.get()
    print(inp)
    with open(inp, "r") as s:
        for x in s:
            for y in x:
                if (y == " " or y == "\n"):
                    if (b not in a):
                        a.append(b)
                        b = ""
                    else:
                        b = ""
                        pass
                else:
                    b = b + y
        s.close()
    print(a)
    print(f"There are {len(a)} unique items in our dataset ")
    st = []
    with open(inp, "r") as s:
        count = 0
        for x in s:
            st.append(x)
            if count > 1010:
                break
            count += 1
        support_threshold = float(entry_support.get())
        countt=float(entry_confidence.get())
        print(support_threshold)
        ap(st, support_threshold,countt)
        s.close()
root = Tk()
frame = Frame(root)
frame.pack()

statusText = StringVar(root)
statusText.set(
    "Click Browse and select file, then select Confidence and Support Thresholds, " "\nClick
on View Input to view Input Transactions or Click on Go")

top_frame = Frame(frame)

label = Label(top_frame, text="CSV file: ", padx=5, pady=2)
label.pack(side=LEFT)
entry_filename = Entry(top_frame, bd=5, width=100)
entry_filename.pack(side=LEFT)
La = Label(top_frame, padx=3, pady=5, text="Batch 16 ")
La.pack(side=LEFT)
button_browse = Button(top_frame, text="Browse", command=button_browse_callback)
button_browse.pack(side=RIGHT)
```

```
top_frame.pack()
separator = Frame(frame, height=2, bd=1, relief=SUNKEN)
separator.pack(fill=X, padx=5, pady=5)

mid_frame = Frame(frame)

L1 = Label(mid_frame, padx=5, pady=5, text="count")
entry_confidence = Spinbox(mid_frame, width=5, from_=1, to=1000, format="%.2f",
increment=0.05, bd=5)
L1.pack(side=LEFT)
entry_confidence.pack(side=LEFT)

L2 = Label(mid_frame, padx=5, pady=5, text="Minimum Support :")
entry_support = Spinbox(mid_frame, width=5, from_=1, to=1000, format="%.2f",
increment=0.05, bd=5)
L2.pack(side=LEFT)
entry_support.pack(side=LEFT)

Lb = Label(mid_frame, padx=10, pady=5, text=" ")
Lb.pack(side=LEFT)

button_viewinput = Button(mid_frame, text="View Input",
command=button_viewinput_callback)
button_viewinput.pack(side=LEFT)

La = Label(mid_frame, padx=5, pady=5, text=" ")
La.pack(side=LEFT)

button_go = Button(mid_frame, text="Go", command=go)
button_go.pack(side=RIGHT)
mid_frame.pack()
separator = Frame(mid_frame, height=2, bd=1, relief=SUNKEN)
separator.pack(fill=X, padx=5, pady=5)

separator = Frame(frame, height=2, bd=1, relief=SUNKEN)
separator.pack(fill=X, padx=5, pady=5)

txt_frm = Frame(frame, width=600, height=600)
txt_frm.pack(fill="both", expand=True)
txt_frm.grid_propagate(False)
txt_frm.grid_rowconfigure(0, weight=1)
txt_frm.grid_columnconfigure(0, weight=1)
text = Text(txt_frm, borderwidth=3)
text.grid(row=0, column=0, sticky="nsew", padx=2, pady=2)
scrollb = Scrollbar(txt_frm, command=text.yview)
scrollb.grid(row=0, column=1, sticky='nsew')
text['yscrollcommand'] = scrollb.set
text.pack()
```

```
separator = Frame(frame, height=2, bd=1, relief=SUNKEN)
separator.pack(fill=X, padx=5, pady=5)

message = Label(frame, textvariable=statusText)
message.pack()


mainloop()


#if __name__ == '__main__':
 #   button_go_callback()
print("sas")
```

# 10. SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner.

Unit testing was performed to test correctness of different modules.

**Test case 1: Correctness of the output.**

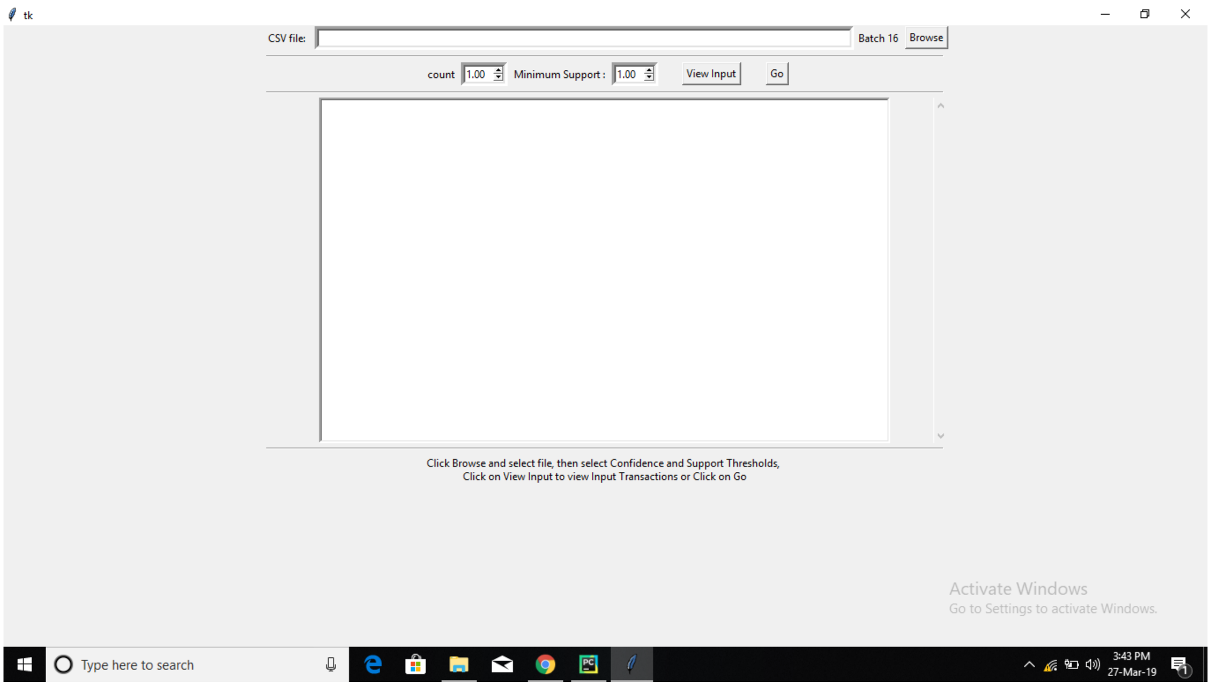
## 10.1. Testing correctness of the output:

In this test approach sample input was given to the system and different support and confidence was provided. At first large sample input were given to the system with different support and confidence value. Then few sample input were given to the system with different support and confidence.

Test case and output

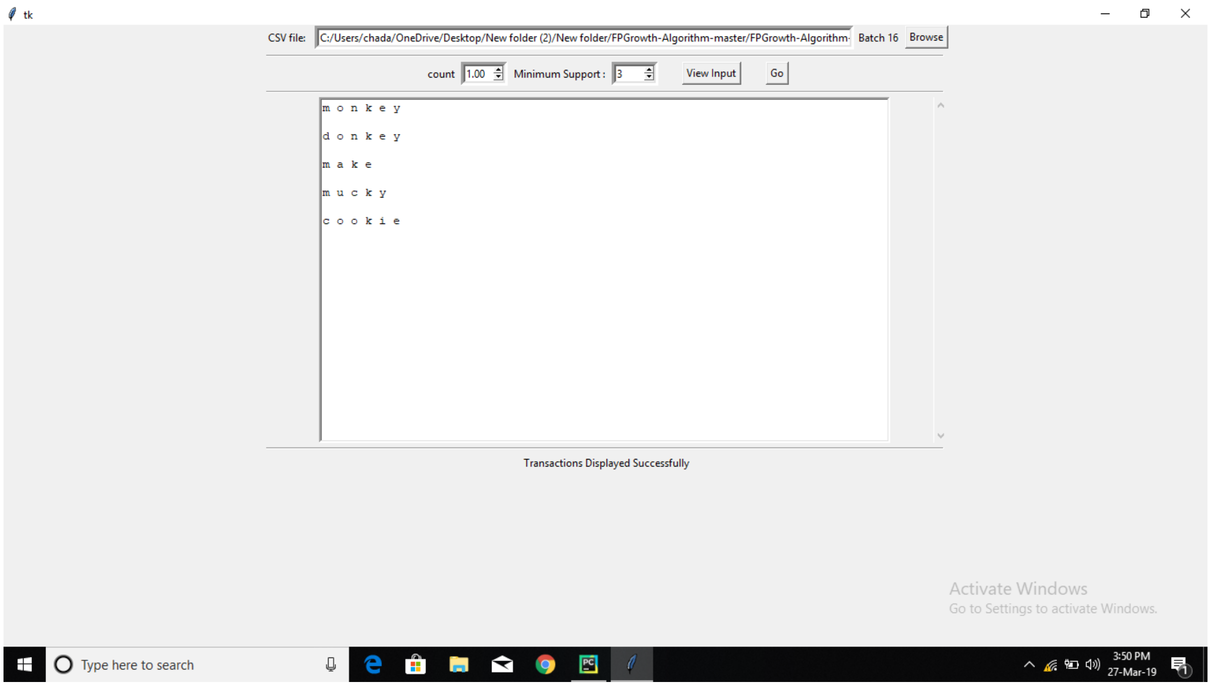
Test case 1:

A Dataset containing 5 transactions was taken as the sample input.



Click Browse and select file, then select Confidence and Support Thresholds,
Click on View Input to view Input Transactions or Click on Go

**Fig10.1 GUI Interface**



CSV file: C:/Users/chada/OneDrive/Desktop/New folder (2)/New folder/FPGrowth-Algorithm-master/FPGrowth-Algorithm

count 1.00 Minimum Support : 3

```
m o n k e y
d o n k e y
m a k e
m u c k y
c o o k i e
```

Transactions Displayed Successfully

**Fig10.2 Display of Transactions**

**Output:**

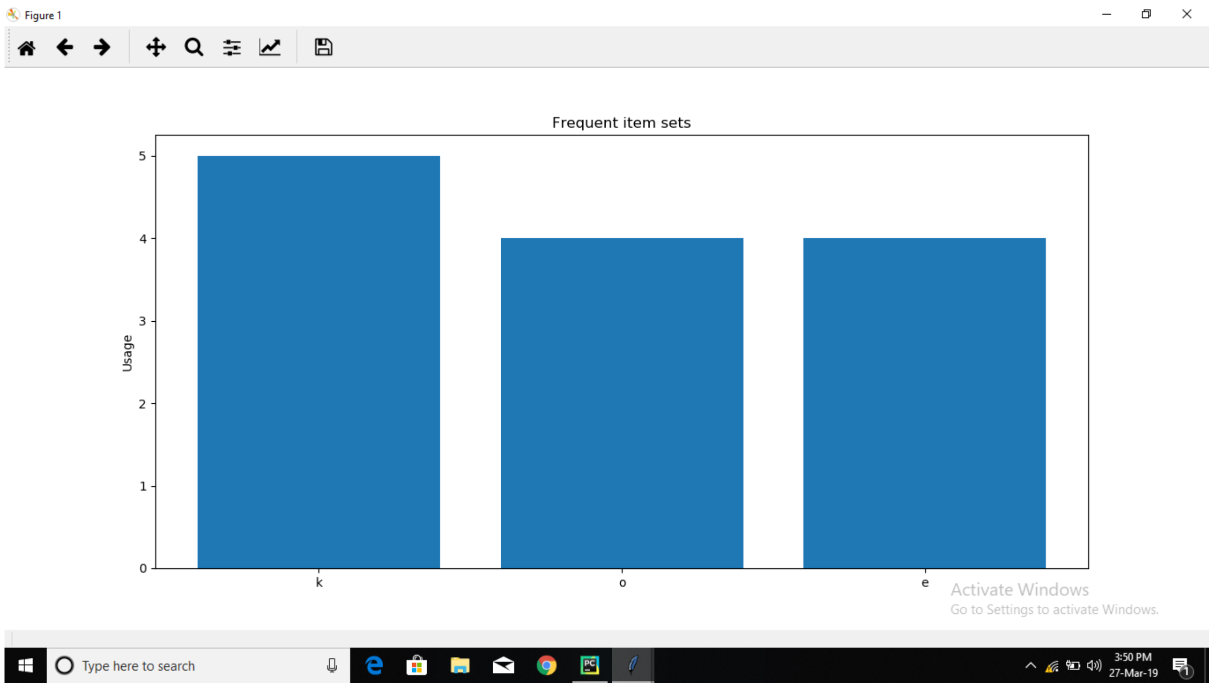


**Fig10.3 Frequent item sets containing 1 item**

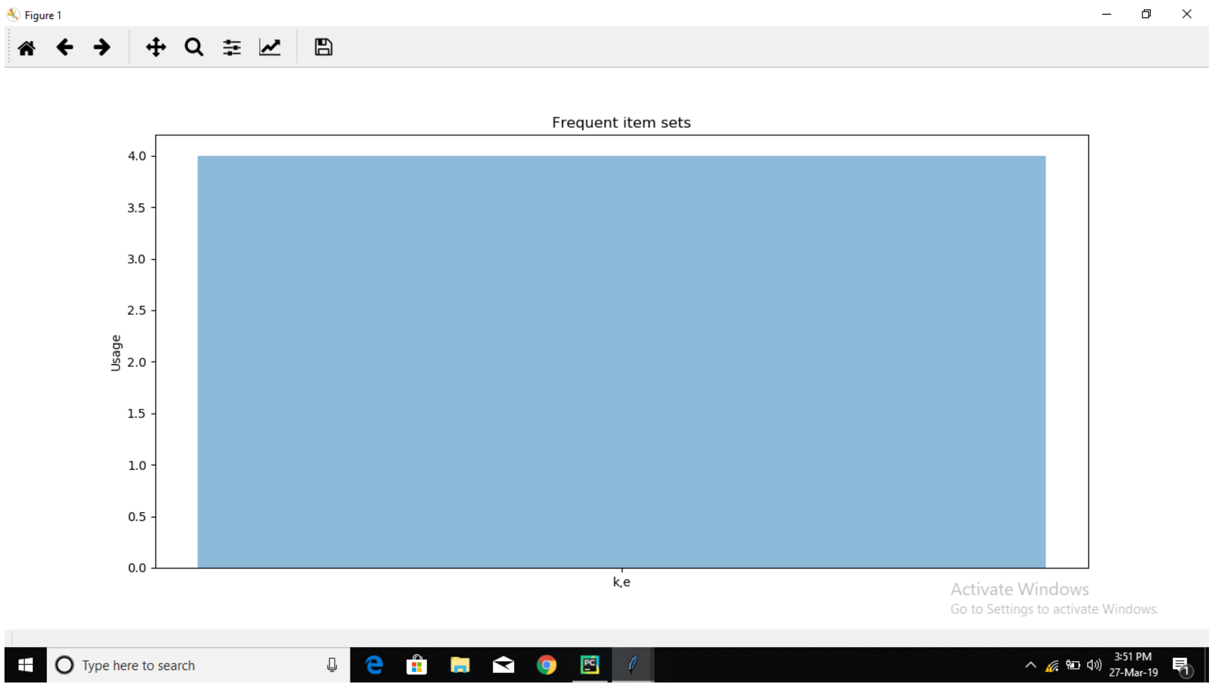
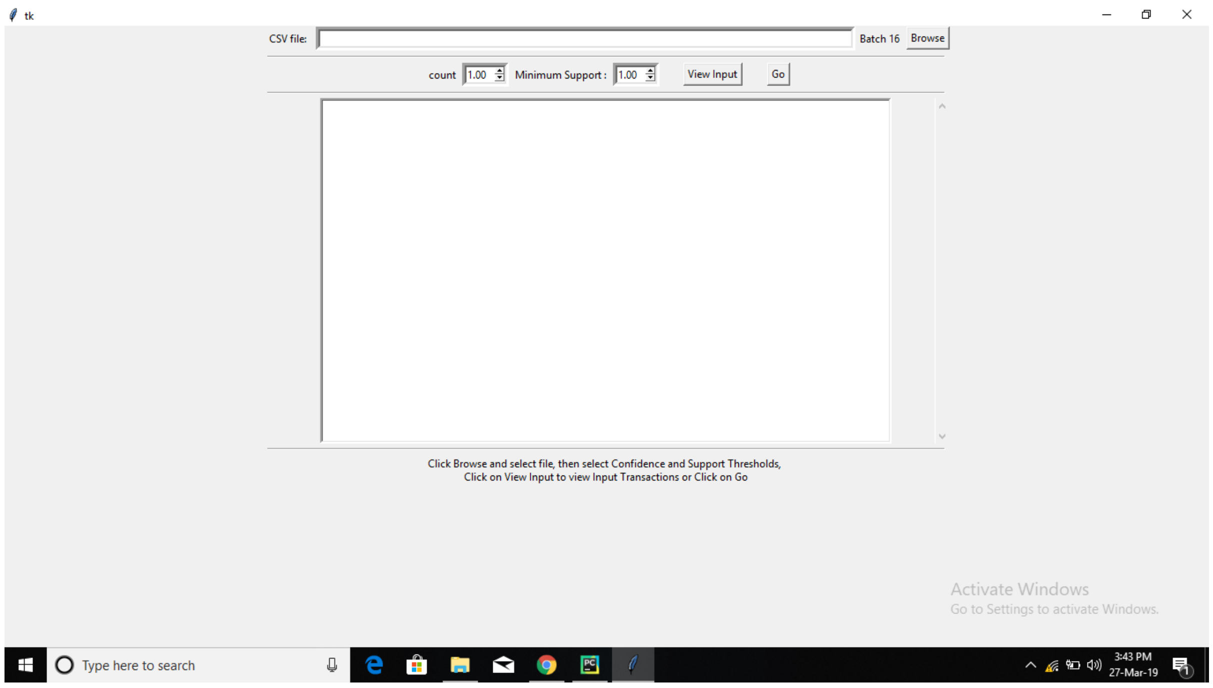


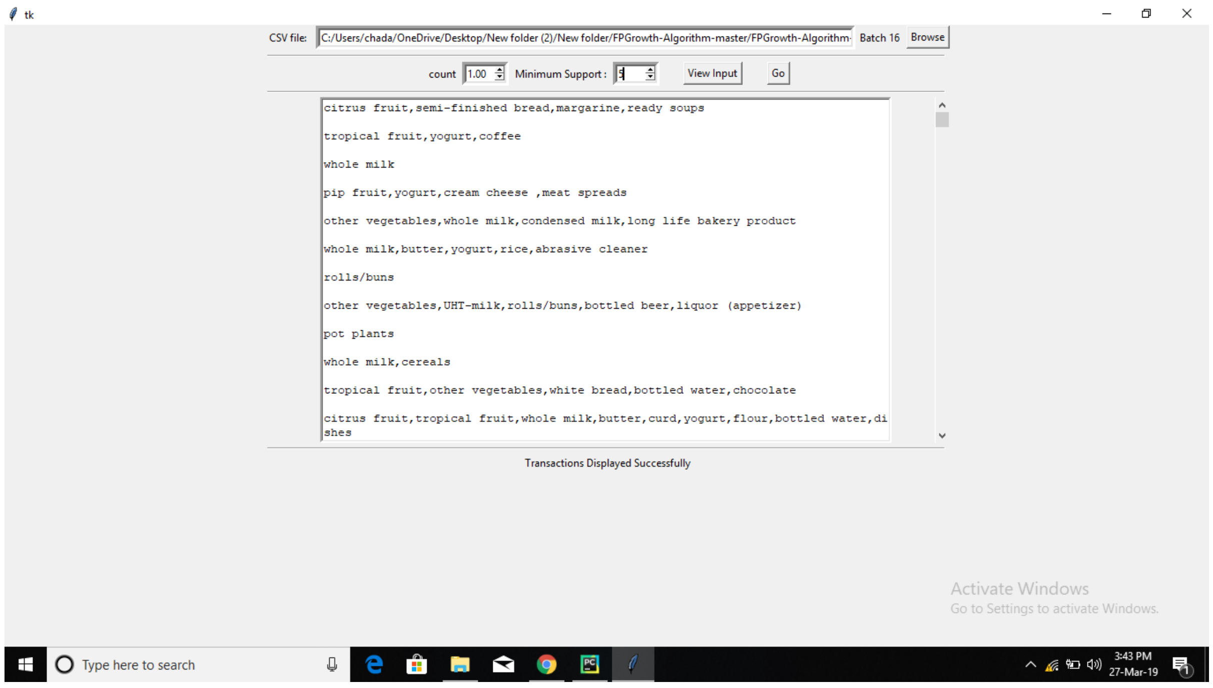
**Fig10.4 Frequent item sets containing 2 items**

# 11. OUTPUTS AND RESULTS

## GUI Interface



Click Browse and select file, then select Confidence and Support Thresholds,
Click on View Input to view Input Transactions or Click on Go



CSV file: C:/Users/chada/OneDrive/Desktop/New folder (2)/New folder/FPGrowth-Algorithm-master/FPGrowth-Algorithm-

count 1.00 Minimum Support : 5

```
citrus fruit,semi-finished bread,margarine,ready soups

tropical fruit,yogurt,coffee

whole milk

pip fruit,yogurt,cream cheese ,meat spreads

other vegetables,whole milk,condensed milk,long life bakery product

whole milk,butter,yogurt,rice,abrasive cleaner

rolls/buns

other vegetables,UHT-milk,rolls/buns,bottled beer,liquor (appetizer)

pot plants

whole milk,cereals

tropical fruit,other vegetables,white bread,bottled water,chocolate

citrus fruit,tropical fruit,whole milk,butter,curd,yogurt,flour,bottled water,di
shes
```

Transactions Displayed Successfully

**Fig 11.1 Data set with 9835 Transactions**

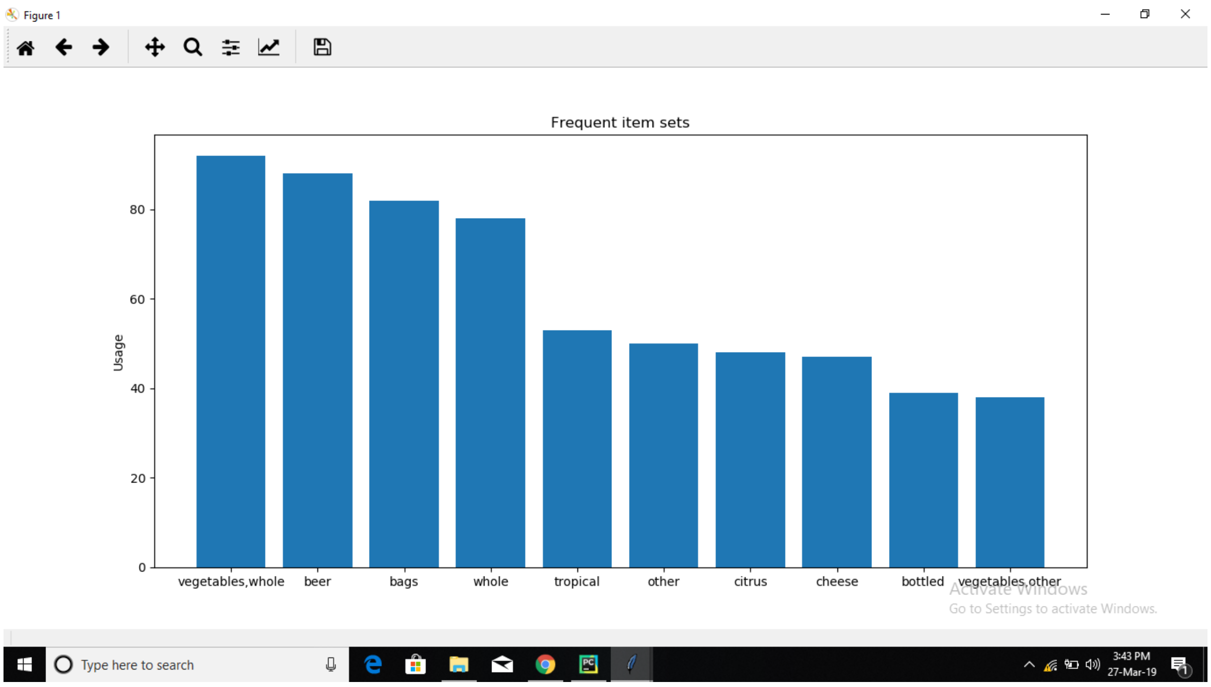


**Fig11.2 Frequent item sets containing single item**

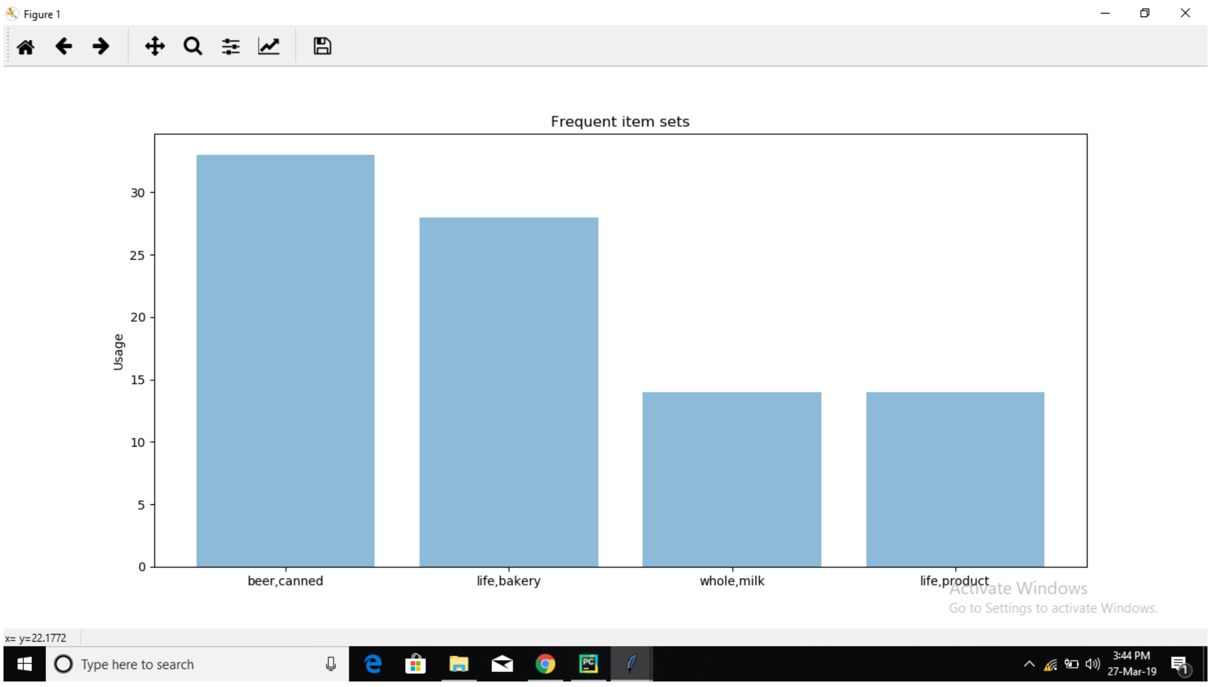


**Fig11.3 Frequent item sets containing 2 items**

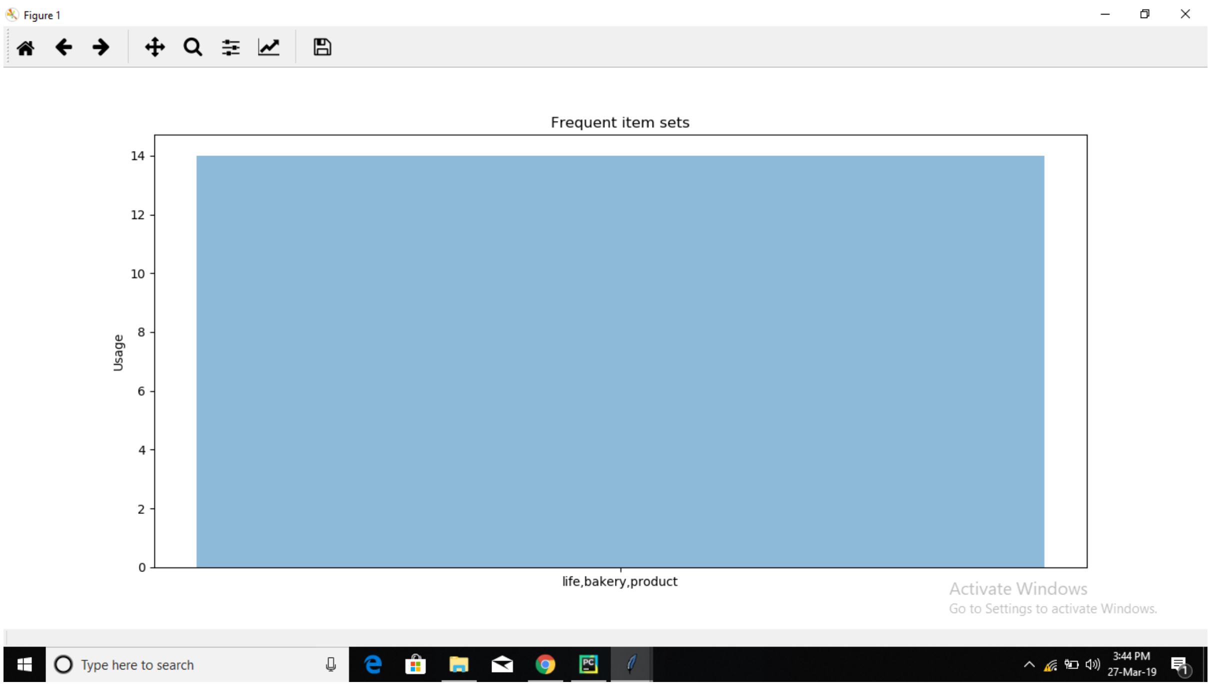


**Fig11.4 Frequent item sets containing 3 items**

# 12. CONCLUSION AND FUTURESCOPE

Due to exponential growth of computer hardware and system software technology there is large supply of powerful and cost effective computers. This technology provides a huge number of databases and information repositories available for transaction management information retrieval and data analysis. Physical analysis of this large amount of data is very difficult. This has lead to the necessity of data mining tools. Association rule mining and classification technique to find the related information in large databases is becoming very important in the current scenario. The large quantity of information collected through the set of association rules can be used not only for illustrating the relationships in the database, but also used for differentiating between different kinds of classes in a database.

The Apriori algorithm effectively generates highly informative frequent item sets and association rules for the data of the supermarket. The frequent data items are generated from the given input data and based on the frequent item stets strong association rules were generated.

This paper provides some of the existing data mining algorithms for market basket analysis. The analysis of existing algorithms suggests that the usage of association rule mining algorithms for market basket analysis will help in better classification of the huge amount of data. The apriori algorithm can be modified effectively to reduce the time complexity and enhance the accuracy.

## 13. REFERENCES

1. Chia-Chu Chiang. “Programming Parallel Apriori Algorithms for Mining Association Rules”, Proceedings of IEEE International Conference on System Science and Engineering, 2010, pp.593-598.
2. Du Ping., and Gao Yongping.“A New Improvement of Apriori Algorithm for Mining Association Rules”, Proceedings of IEEE International Conference on Computer Application and System Modelling, Vol.2, 2010, pp.529-532.
3. Marcin Gorawski., and Pawel Jureczek. “Using Apriori-like Algorithms for Spatio-Temporal Pattern Queries”, Proceedings of IEEE, International Multiconference on Computer Science and Information Technology, 2009, pp.43-48.
4. Mirela Pater., and Daniela, E.“Market-Basket Problem Solved With Depth First Multi-Level Apriori Mining Algorithm”, IEEE International Workshop on Soft Computing Applications,2009, pp: 133-138.
5. Xiaojun Wang.“Study of Data Mining based on Apriori Algorithm”, Proceedings of IEEE International Conference on Software Technology and Engineering, 2010, pp.400-403.
6. Michael Hahsler, Kurt Hornik, and Thomas Reutterer (2006) Implications of probabilistic data modeling for mining association rules. In M. Spiliopoulou, R. Kruse, C. Borgelt, A. Nuernberger, and W. Gaul, editors, From Data and Information Analysis to Knowledge Engineering, Studies in Classification, Data Analysis, and Knowledge Organization, pages 598–605. Springer-Verlag.