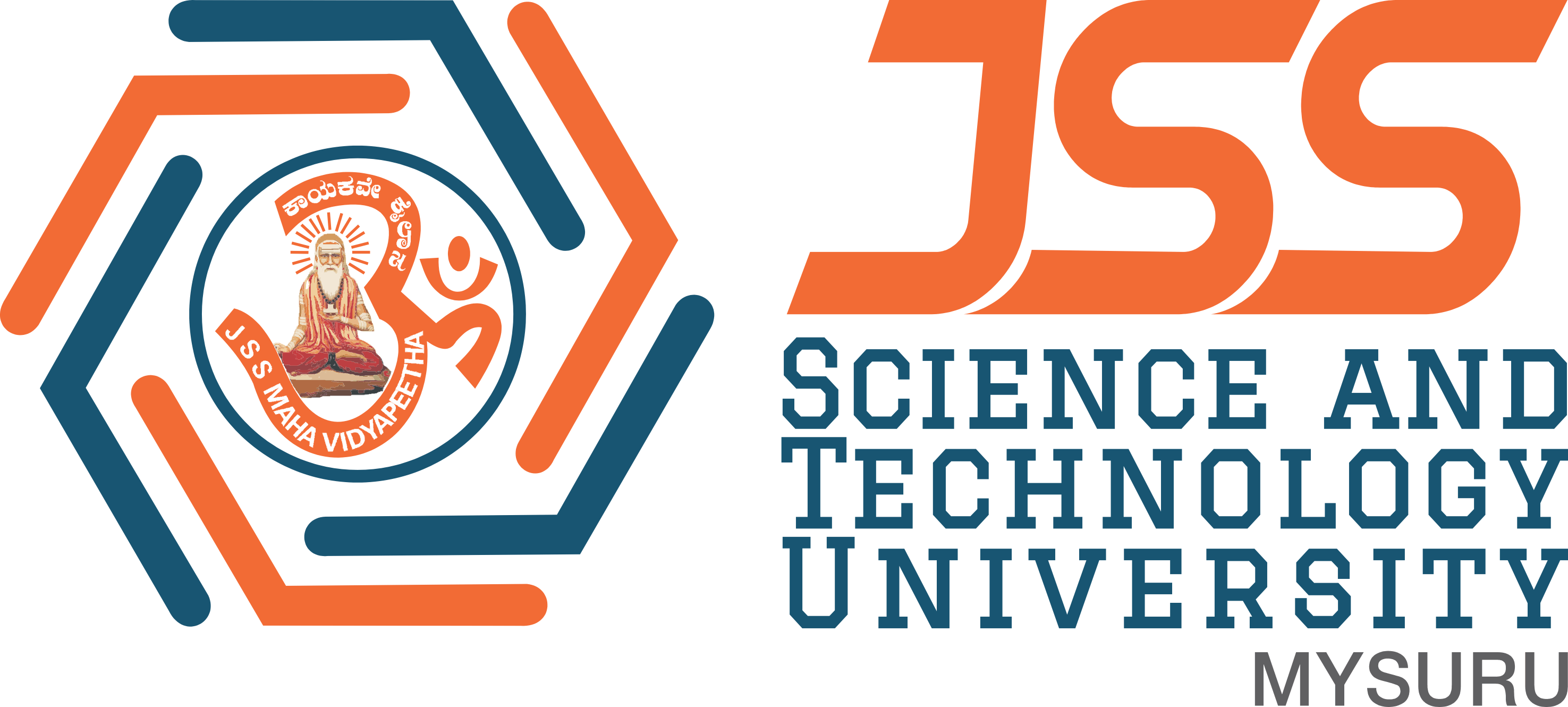
**JSS MAHAVIDYAPEETHA**

**JSS SCIENCE AND TECHNOLOGY UNIVERSITY**

JSS Technical Institutions Campus, Mysuru – 570006



**“Implementation of Data Preprocessing and Apriori Algorithm using the Electronics Goods Kaggle Dataset”**

Mini project report submitted in partial fulfillment of curriculum prescribed for the Data Warehouse and Data Mining (CS610) course for the award of the degree of

#### BACHELOR OF ENGINEERING

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

*by*

# **Vishal Chincholi Dhanush B L**

**(01JST17CS178) (01JST18CS404)**

# **Chandan Kumar M Meraj Ahmed**

**(01JST18CS402) (01JST17CS086)**

*Under the Guidance of*

#### Dr. Trisiladevi C. Nagavi

Assistant Professor,

Dept.of CS & E, JSS STU Mysore

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

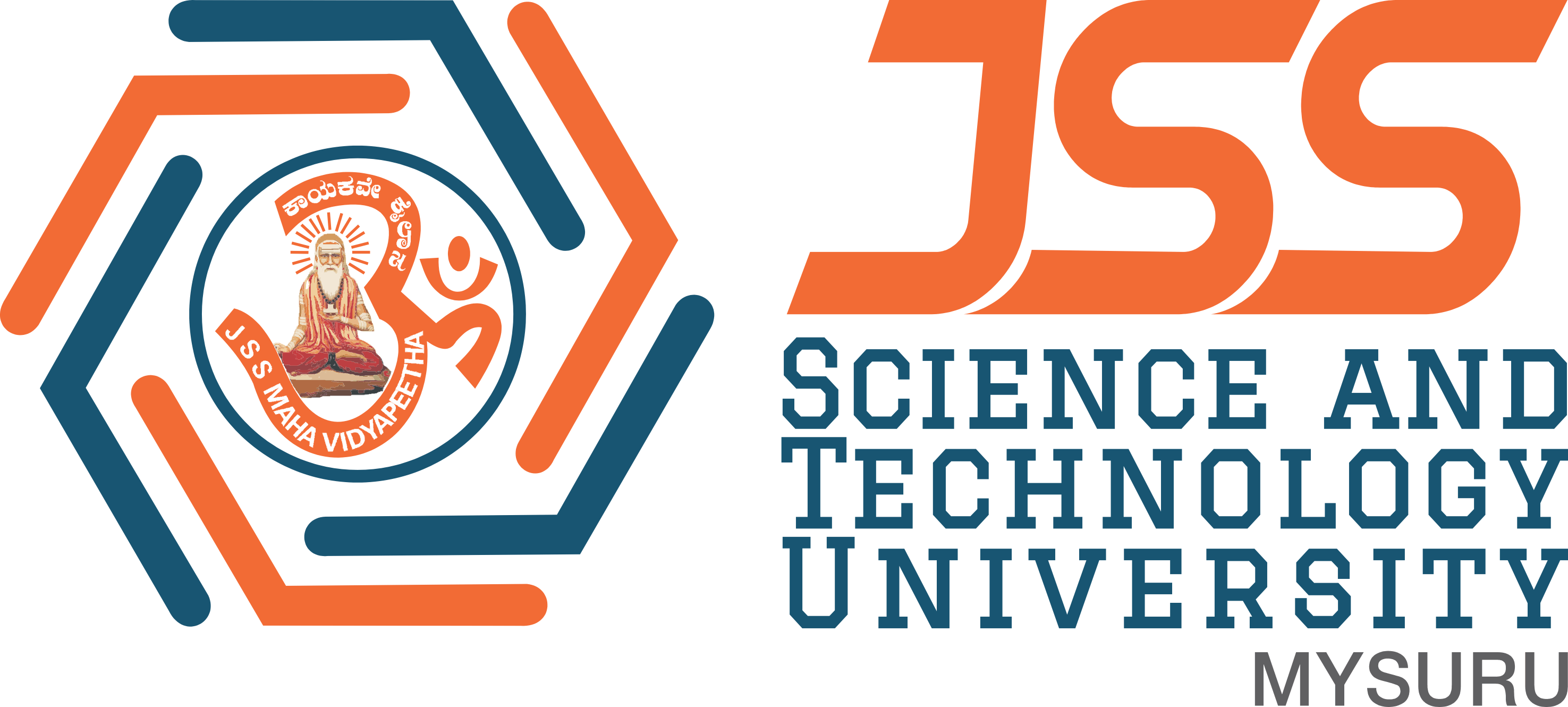
**2020**

|  |  |  |
| --- | --- | --- |
|  |  |  |

**JSS MAHAVIDYAPEETHA**

**JSS SCIENCE AND TECHNOLOGY UNIVERSITY**

JSS Technical Institutions Campus, Mysuru – 570006



**CERTIFICATE**

This is to certify that the work entitled “Implementation of Data Preprocessing and Apriori Algorithm using the Electronics Goods Kaggle Dataset” is a bonafide work carried out by Vishal C, Meraj A, Chandan Kumar M, Dhanush B L in partial fulfillment of the award of the degree of Bachelor of Engineering in Computer Science and Engineering of JSS Science and Technology, Mysuru during the year 2020. It is certified that all corrections / suggestions indicated during CIE have been incorporated in the report. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work prescribed for the Data Warehosue and Data Mining (CS610) course.

## Course In Charge and Guide

**Dr. Trisiladevi C. Nagavi**

Assistant Professor

Dept. of CS & E

JSS STU Mysore

**Place:** Mysore **Date : 07/05/2020**

**DECLARATION**

We the undersigned solemnly declare that this project report is based on our own work carried out during the course of our study under the supervision of **Dr. Trisiladevi C. Nagavi.** We assert the statements made and conclusions drawn are an outcome of my research work. I further certify that

1. The work contained in the report is original and has been done by us under the general supervision of our supervisor.
2. The work has not been submitted to any other Institution for any other degree/diploma/certificate in this university or any other University of India or abroad.
3. We have followed the guidelines provided by the university in writing the report.
4. Whenever we have used materials (data, theoretical analysis, and text) from other sources, we have given due credit to them in the text of the report and giving their details in the references.

# Vishal Chincholi Name: Dhanush B L

Roll No: 63 Roll No: 70

USN No:01JST17CS178USN No:01JST18CS404

# Name: Chandan Kumar MName: Meraj Ahmed

Roll No: 69 Roll No: 34

USN No:01JST18CS402USN No:01JST17CS086

**ABSTRACT**

Now a day's Data mining has a lot of e-Commerce applications. The key problem is how to find useful hidden patterns for better business applications in the retail sector. For the solution of these problems, The Apriori algorithm is one of the most popular data mining approach for finding frequent item sets from a transaction dataset and derive association rules. Rules are the discovered knowledge from the data base. Finding frequent item set (item sets with frequency larger than or equal to a user specified minimum support) is not trivial because of its combinatorial explosion. Once frequent item sets are obtained, it is straightforward to generate association rules with confidence larger than or equal to a user specified minimum confidence. This mini project illustrates apriori algorithm on a kaggle database and finds the association rules on different confidence value.

**ACKNOWLEDGEMENT**

We would like to express our deep gratitude to Professor Dr. Trisiladevi C. Nagavi, for their patient guidance, enthusiastic encouragement and useful critiques of this research work. We also wish to thank our parents for their support and encouragement throughout this tough time of COVID-19.

**PROJECT TEAM DETAILS WITH PHOTOGRAPH**



# Vishal Chincholi Name: Dhanush B L

Roll No: 63 Roll No: 70

USN No:01JST17CS178USN No:01JST18CS404

# Name: Chandan Kumar MName: Meraj Ahmed

Roll No: 69 Roll No: 34

USN No:01JST18CS402USN No:01JST17CS086

**TABLE OF CONTENTS**

**CHAPTER 1: INTRODUCTION**

* 1. **Introduction to the problem domain**

**Association rule mining** is a technique to identify underlying relations between different items. Take an example of a Super Market where customers can buy variety of items. Usually, there is a pattern in what the customers buy. For instance, mothers with babies buy baby products such as milk and diapers. Damsels may buy makeup items whereas bachelors may buy beers and chips etc. In short, transactions involve a pattern. More profit can be generated if the relationship between the items purchased in different transactions can be identified.

For instance, if item A and B are bought together more frequently then several steps can be taken to increase the profit. For example:

A and B can be placed together so that when a customer buys one of the product he doesn't have to go far away to buy the other product.

People who buy one of the products can be targeted through an advertisement campaign to buy the other.

Collective discounts can be offered on these products if the customer buys both of them.

Both A and B can be packaged together.

* 1. **Aim/Statement of problem**

Preprocess the electronics and vegetables dataset and apply Apriori algorithm to mine the frequent item set. Define the association rules from the frequent item dataset.

* 1. **Objectives of Project Work**
* The objective of using Apriori algorithm is to find frequent itemsets and association between different itemsets, that is, association rule. Apriori is Easy implementation.
* The algorithm applies information from previous steps to produce the frequent itemsets. Apriori is the most uncomplicated algorithm, which is employed for mining of repetitive patterns from the transaction database.
* We have aimed to execute the Apriori algorithm for adequate study work.
* The benefits of using Apriori algorithm are using large item set property. Easily parallelized, simply and easy to implement. It is an efficient algorithm for finding all frequent itemsets.
  1. **Applications**

1. **Market Basket Analysis**

E-commerce giants (like Amazon, eBay) are using the Apriori algorithm to extract data insights on which products are likely to be purchased together. For example, a retailer might use Apriori to predict that people who buy sugar and flour are likely to buy eggs to bake a cake.



1. **Detecting Adverse Drug Reactions**

Apriori algorithm is used for association analysis on healthcare data like-the drugs taken by patients, characteristics of each patient, adverse ill-effects patients experience, initial diagnosis, etc. This analysis produces association rules that help identify the combination of patient characteristics and medications that lead to adverse side effects of the drugs.



1. **Auto-Complete Applications**

Google auto-complete is another popular application of Apriori wherein - when the user types a word, the search engine looks for other associated words that people usually type after a specific word.



1. **Education Field:** Extracting association rules in data mining of admitted students through characteristics and specialties.
2. **Education Field:** Extracting association rules in data mining of admitted students through characteristics and specialties.
3. **Forestry:** Analysis of probability and intensity of forest fire with the forest fire data.

**CHAPTER 2: TOOLS AND TECHNOLOGY USED**

* 1. **Tools**

**Jupyter Notebook**

Project Jupyter is a nonprofit organization created to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages". Spun-off from IPython in 2014 by Fernando Pérez, Project Jupyter supports execution environments in several dozen languages. Project Jupyter's name is a reference to the three core programming languages supported by Jupyter, which are Julia, Python and R, and also a homage to Galileo's notebooks recording the discovery of the moons of Jupiter. Project Jupyter has developed and supported the interactive computing products Jupyter Notebook, JupyterHub, and JupyterLab, the next-generation version of Jupyter Notebook.

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

A Jupyter Notebook can be converted to a number of open standard output formats (HTML, presentation slides, LaTeX, PDF, ReStructuredText, Markdown, Python) through "Download As" in the web interface, via the nbconvert library or "jupyter nbconvert" command line interface in a shell. To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer which can take a URL to any publicly available notebook document, convert it to HTML on the fly and display it to the user.

Jupyter Notebook provides a browser-based REPL built upon a number of popular open-source libraries:

IPython

ØMQ

Tornado (web server)

jQuery

Bootstrap (front-end framework)

MathJax

* 1. **Technology used**
     1. **Python**

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python was conceived in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system capable of collecting reference cycles. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3.

The Python 2 language was officially discontinued in 2020 (first planned for 2015), and "Python 2.7.18 is the last Python 2.7 release and therefore the last Python 2 release." No more security patches or other improvements will be released for it. With Python 2's end-of-life, only Python 3.5.x and later are supported.

Python interpreters are available for many operating systems. A global community of programmers develops and maintains CPython, an open source reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

Python's large standard library, commonly cited as one of its greatest strengths, provides tools suited to many tasks. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary-precision decimals, manipulating regular expressions, and unit testing.

Some parts of the standard library are covered by specifications (for example, the Web Server Gateway Interface (WSGI) implementation wsgiref follows PEP 333), but most modules are not. They are specified by their code, internal documentation, and test suites. However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

As of November 2019, the Python Package Index (PyPI), the official repository for third-party Python software, contains over 200,000 packages with a wide range of functionality, including:

Automation

Data analytics

Databases

Documentation

Graphical user interfaces

Image processing

Machine learning

Mobile App

Multimedia

Networking

Scientific computing

System administration

Test frameworks

Text processing

Web frameworks

Web scraping

* + 1. **Libraries**

1. **Pandas**

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

**Features:**

DataFrame object for data manipulation with integrated indexing.

Tools for reading and writing data between in-memory data structures and different file formats.

Data alignment and integrated handling of missing data.

Reshaping and pivoting of data sets.

Label-based slicing, fancy indexing, and subsetting of large data sets.

Data structure column insertion and deletion.

Group by engine allowing split-apply-combine operations on data sets.

Data set merging and joining.

Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.

Time series-functionality: Date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging.

Provides data filtration.

The library is highly optimized for performance, with critical code paths written in Cython or C.

1. **Numpy**

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

**Features:**

NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available. SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

1. **Itertools**

Python’s Itertool is a module that provides various functions that work on iterators to produce complex iterators. This module works as a fast, memory-efficient tool that is used either by themselves or in combination to form iterator algebra.

For example, let’s suppose there are two lists and you want to multiply their elements. There can be several ways of achieving this. One can be using the naive approach i.e by iterating through the elements of both the list simultaneously and multiply them. And another approach can be using the map function i.e by passing the mul operator as a first parameter to the map function and Lists as the second and third parameter to this function. Let’s see the time taken by each approach.

Different types of iterators provided by this module are:

Infinite iterators

Combinatoric iterators

Terminating iterators

**CHAPTER 3: SYSTEM DESIGN AND IMPLEMENTATION**

**System Design**

Apriori is an algorithm for frequent item set mining and association rule learning over relational databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent item sets determined by Apriori can be used to determine association rules which highlight general trends in the database and has applications in domains such as market basket analysis.

Apriori is **designed** to operate on databases containing transactions (for example, collections of items bought by customers, or details of a website frequentation or IP addresses). Other algorithms are designed for finding association rules in data having no transactions (Winepi and Minepi), or having no timestamps (DNA sequencing). Each transaction is seen as a set of items (an itemset). Given a threshold C, the Apriori algorithm identifies the item sets which are subsets of at least C transactions in the database.

Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

Apriori uses breadth-first search and a Hash tree structure to count candidate item sets efficiently. It generates candidate item sets of length k from item sets of length k-1. Then it prunes the candidates which have an infrequent sub pattern. According to the downward closure lemma, the candidate set contains all frequent k-length item sets. After that, it scans the transaction database to determine frequent item sets among the candidates.

Frequent itemset or pattern mining is broadly used because of its wide applications in mining association rules, correlations and graph patterns constraint that is based on frequent patterns, sequential patterns, and many other data mining tasks. It was the first algorithm that was proposed for frequent itemset mining. This algorithm uses two steps “join” and “prune” to reduce the search space. It is an iterative approach to discover the most frequent itemsets.

The probability that item I is not frequent is if:

P(I) < minimum support threshold, then I is not frequent.

P (I+A) < minimum support threshold, then I+A is not frequent, where A also belongs to itemset.

If an itemset set has value less than minimum support then all of its supersets will also fall below min support, and thus can be ignored. This property is called the Antimonotone property.

**Implementation**

Apriori algorithm is a sequence of steps to be followed to find the most frequent itemset in the given database. This data mining technique follows the join and the prune steps iteratively until the most frequent itemset is achieved. A minimum support threshold is given in the problem or it is assumed by the user.

#1) In the first iteration of the algorithm, each item is taken as a 1-itemsets candidate. The algorithm will count the occurrences of each item.

#2) Let there be some minimum support, min\_sup. The set of 1 – itemsets whose occurrence is satisfying the min sup are determined. Only those candidates which count more than or equal to min\_sup, are taken ahead for the next iteration and the others are pruned.

#3) Next, 2-itemset frequent items with min\_sup are discovered. For this in the join step, the 2-itemset is generated by forming a group of 2 by combining items with itself.

#4) The 2-itemset candidates are pruned using min-sup threshold value. Now the table will have 2 –itemsets with min-sup only.

#5) The next iteration will form 3 –itemsets using join and prune step. This iteration will follow antimonotone property where the subsets of 3-itemsets, that is the 2 –itemset subsets of each group fall in min\_sup. If all 2-itemset subsets are frequent then the superset will be frequent otherwise it is pruned.

#6) Next step will follow making 4-itemset by joining 3-itemset with itself and pruning if its subset does not meet the min\_sup criteria. The algorithm is stopped when the most frequent itemset is achieved.

**CHAPTER 4: SYSTEM TESTING AND RESULT ANALYSIS**

**CHAPTER 5: CONCLUSION AND FUTURE WORK**

* 1. **Conclusion**

Apriori algorithm is an efficient algorithm that scans the database only once.It reduces the size of the itemsets in the database considerably providing a good performance. Thus, data mining helps consumers and industries better in the decision-making process.

This mini project addresses the necessity of finding frequent patterns in the data mining. Problems related to frequent pattern mining have been analyzed and found a better solution for it. Basically Apriori algorithm is used to find the frequent patterns available in the database. Initially the mini project work starts with the proposed AprioriAllHybrid algorithm for mining a frequent pattern which performs better when compared to Apriori algorithm and it have been proved by taking various size of database.

The association rule generated by Apriori algorithm is optimized using genetic algorithm and then a parallel algorithm has been proposed which is efficient and proved with some sample data, but the disadvantage is cost. Finally partition algorithm have been proposed, in this approach control will move to the particular partition instead of scanning the entire database.

Massive experimentation work was performed for evaluating and comparing Apriori, AprioriAllHybrid and Partition algorithms. This mini project concludes that the proposed partition algorithm approach consistently performs well than rest of the algorithm and will support many future researches in many ways.

* 1. **Future Work and Enhancements**

Different data items have different formats, so there is no universal solution for this problem. The future enhancements will focus on,

* At present, this algorithm takes only table as input, which contains only text, and this can be improved by allowing any type of inputs.

* Association rule generated can be effectively optimized using GA or some other techniques, so that only strong rules will be obtained which improves the effectiveness of the algorithm.

Thus the algorithm can be enhanced in different aspects, which helps to improve the performance of the algorithm. By incorporating all the mentioned enhancements in the proposed system, an efficient, effective and intelligent method for mining frequent patterns can be developed.

**Many methods are available for improving the efficiency of the algorithm:**

**Hash-Based Technique:** This method uses a hash-based structure called a hash table for generating the k-itemsets and its corresponding count. It uses a hash function for generating the table.

**Transaction Reduction:** This method reduces the number of transactions scanning in iterations. The transactions which do not contain frequent items are marked or removed.

**Partitioning:** This method requires only two database scans to mine the frequent itemsets. It says that for any itemset to be potentially frequent in the database, it should be frequent in at least one of the partitions of the database.

**Sampling:** This method picks a random sample S from Database D and then searches for frequent itemset in S. It may be possible to lose a global frequent itemset. This can be reduced by lowering the min\_sup.

**Dynamic Itemset Counting:** This technique can add new candidate itemsets at any marked start point of the database during the scanning of the database.

**REFERENCES**