General Sir John Kotelawala Defence University

Faculty of Management, Social Sciences and Humanities

Department of Languages

BSc in Applied Data Science Communication

Fundamentals of data mining / LB 2114

Group Assignment

P Laksia – D/ADC/23/0013

PRM Perera – D/ADC/23/0031

MM Jayasinghe – D/ADC/23/0035

RYN Sanduprabha – D/ADC/23/0046

Fundamentals of data mining / LB 2114

Year 2: Semester 1

Assignment number 2

30.04.2024

Content

Task 01

Market Basket Analysis

1.1. Introduction

Each shopper, customer, consumer or buyer have essential and desired products that they wish to purchase according to their consumer preference and the price of the good demanded. In this competitive environment, market competition is a constant battle of strategy and tactics between businesses of the same or similar industry, where they should focus on customer centric strategies, satisfying customer preferences while increasing their profit, sales and revenue enabling them to survive in the market and the competitive business environment.

In today's digitalized world, with the introduction of the internet, technology and globalization, businesses and firms transitioned from labour intensive towards more technical and digital intensive. In today's business world, the strategies implemented are often based on the analysis of customer data regarding their purchase where they discover and extract useful information, develop and implement strategies according to them.

Market basket analysis is a strategic data mining method which is well known in the marketing sector which is used by retailers to increase profits, sales and revenue by understanding the purchasing order of customers by analysing the data sets such as the purchase history reveals the most likely items that are purchased and which items that are likely to be purchased together.

Through the recognition of these patterns, it allows the retailers to make informed decisions on regards to inventory management, concoct effective marketing and pricing strategies which improves customer satisfaction thereby, improving the name and value of the business, increasing the sales, profits and the business's revenue while enabling them to survive in the market.

1.2. Dataset

The market basket analysis is a multivariate dataset which consists of 788 rows and 39 columns.

The 39 variables are as follows;

1. **Date:** represents the date of the items purchased by consumers. Data type – numerical.
2. **All-purpose:** an item purchased by customers. Data type – categorical.
3. **Aluminum foil:** an item purchased by customers. Data type – categorical.
4. **Bagels:** an item purchased by customers. Data type – categorical.
5. **Beef:** an item purchased by customers. Data type – categorical.
6. **Butter:** an item purchased by customers. Data type – categorical.
7. **Cereal:** an item purchased by customers. Data type – categorical.
8. **Cheese:** an item purchased by customers. Data type – categorical.
9. **Coffee/ tea:** an item purchased by customers. Data type – categorical.
10. **Dinner rolls:** an item purchased by customers. Data type – categorical.
11. **Dishwashing liquid/ detergent:** an item purchased by customers. Data type – categorical.
12. **Eggs:** an item purchased by customers. Data type – categorical.
13. **Flour:** an item purchased by customers. Data type – categorical.
14. **Fruits:** an item purchased by customers. Data type – categorical.
15. **Hand soap:** an item purchased by customers. Data type – categorical.
16. **Ice cream:** an item purchased by customers. Data type – categorical.
17. **Individual meals:** an item purchased by customers. Data type – categorical.
18. **Juice:** an item purchased by customers. Data type – categorical.
19. **Ketchup:** an item purchased by customers. Data type – categorical.
20. **Laundry detergent:** an item purchased by customers. Data type – categorical.
21. **Lunch meat:** an item purchased by customers. Data type – categorical.
22. **Milk:** an item purchased by customers. Data type – categorical.
23. **Mixes:** an item purchased by customers. Data type – categorical.
24. **Paper towels:** an item purchased by customers. Data type – categorical.
25. **Pasta:** an item purchased by customers. Data type – categorical.
26. **Pork:** an item purchased by customers. Data type – categorical.
27. **Poultry:** an item purchased by customers. Data type – categorical.
28. **Sandwich bags:** an item purchased by customers. Data type – categorical.
29. **Sandwich loaves:** an item purchased by customers. Data type – categorical.
30. **Shampoo:** an item purchased by customers. Data type – categorical.
31. **Soap:** an item purchased by customers. Data type – categorical.
32. **Soda:** an item purchased by customers. Data type – categorical.
33. **Spaghetti sauce:** an item purchased by customers. Data type – categorical.
34. **Sugar:** an item purchased by customers. Data type – categorical.
35. **Toilet paper:** an item purchased by customers. Data type – categorical.
36. **Tortillas:** an item purchased by customers. Data type – categorical.

37. **Vegetables:** an item purchased by customers. Data type – categorical.
38. **Waffles:** an item purchased by customers. Data type – categorical.
39. **Yogurt:** an item purchased by customers. Data type – categorical.

The dataset consists of 38 categorical attributes and 1 numerical attribute.

There are no missing values.

Data preprocessing technique is applied here.

Only association rule mining is used in the dataset.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	Date	all-purpose	aluminum foil	bagels	beef	butter	cereals	cheese	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour	fruits	hand soap	ice cream	individual meals	juice	ketchup	laundry detergent	lunch meat	milk	mixes	paper towels	pasta
2	1/1/2000	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	No
3	2/1/2000	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
4	3/1/2000	Yes	No	No	Yes	No	No	No	No	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes
5	4/1/2000	No	No	No	No	No	No	Yes	Yes	No	Yes	No	No	No	No	Yes	No	Yes	No	No	No	No	No	Yes	No
6	5/1/2000	No	No	Yes	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	No	Yes
7	6/1/2000	No	No	Yes	No	Yes	No	No	No	No	No	No	Yes	No	Yes	No	No	Yes	No	Yes	Yes	No	No	No	Yes
8	7/1/2000	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
9	8/1/2000	No	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	No	No	No	Yes	No	Yes	Yes
10	9/1/2000	No	Yes	Yes	Yes	Yes	No	No	Yes	No	No	Yes	Yes	No	No	No	Yes	Yes	No	Yes	No	No	No	No	No
11	10/1/2000	No	Yes	No	Yes	Yes	No	Yes	Yes	No	Yes	No	No	No	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes	Yes	No
12	11/1/2000	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
13	12/1/2000	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No	No	Yes	No	Yes	Yes	No
14	13/01/2000	Yes	No	Yes	No	Yes	Yes	Yes	No	Yes	No	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
15	14/01/2000	No	No	Yes	No	No	No	Yes	No	No	No	No	Yes	No	No	No	No	No	No	No	No	No	No	No	No
16	15/01/2000	Yes	Yes	No	Yes	No	Yes	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
17	16/01/2000	No	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No
18	17/01/2000	Yes	Yes	No	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
19	18/01/2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
20	19/01/2000	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No	No
21	20/01/2000	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes
22	21/01/2000	Yes	Yes	No	Yes	Yes	No	No	No	Yes	No	No	No	No	No	Yes	No	Yes	No	Yes	No	Yes	Yes	Yes	Yes
23	22/01/2000	No	No	Yes	No	Yes	Yes	No	Yes	No	Yes	No	Yes	Yes	Yes	Yes	No	Yes	No	No	Yes	Yes	No	No	No
24	23/01/2000	Yes	Yes	Yes	Yes	No	No	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
25	24/01/2000	No	No	Yes	No	Yes	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No	No	Yes
26	25/01/2000	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No	No	No	Yes	Yes	No	No	Yes	No	Yes	No	No	Yes	Yes
27	26/01/2000	No	No	Yes	No	No	Yes	No	Yes	Yes	Yes	No	Yes	Yes	No	No	No	Yes	No	Yes	Yes	Yes	No	Yes	Yes
28	27/01/2000	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No
29	28/01/2000	No	No	No	Yes	No	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
30	29/01/2000	No	No	No	Yes	Yes	No	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	No
31	30/01/2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
32	31/01/2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes
33	1/2/2000	Yes	Yes	No	Yes	No	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	No
34	2/2/2000	No	No	No	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes
35	3/2/2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Figure 01

The purpose of the dataset is to identify the purchasing intention of customers through the identification of patterns of the items that are most purchased and the items that are mostly purchased together.

This dataset can be utilized by all businesses and retailers alike to identify the consumer purchase patterns and intentions, understand them and devise various marketing, pricing strategies which improves consumer satisfaction while increasing profits, sales and revenue.

1.3. Explanation and preparation of dataset

The dataset utilized for the association rule mining is about the purchasing details of consumers which consists of 788 data records.

Even though, the dataset is a clean dataset where no missing values were found. Preparatory tasks were carried out on the dataset.

Data preprocessing technique was utilized where raw data was transformed into a useful understandable format to perform the data analysis of the dataset.

Preprocessing of the dataset:

1. Download the dataset into Excel as a csv file.

PurchaseID	Date	V1	V2	V3	V4	V5	V6	V7	V8	V9
1	1/1/2000	yogurt	pork	sandwich bags	lunch meat	all- purpose	flour	soda	butter	vegetables
2	1/1/2000	toilet paper	shampoo	hand soap	waffles	vegetables	cheeses	mixes	milk	sandwich bags
3	1/2/2000	soda	pork	soap	ice cream	toilet paper	dinner rolls	hand soap	spaghetti sauce	milk
4	1/2/2000	cereals	juice	lunch meat	soda	toilet paper	all- purpose			
5	1/2/2000	sandwich loaves	pasta	tortillas	mixes	hand soap	toilet paper	vegetables	vegetables	paper towels
6	1/2/2000	laundry detergent	toilet paper	eggs	toilet paper	vegetables	bagels	dishwashing liquid/detergent	cereals	paper towels
7	1/3/2000	individual meals	paper towels	tortillas	vegetables	milk	ice cream	juice	dishwashing liquid/detergent	soap
8	1/4/2000	ice cream	juice	paper towels	waffles	soda	cheeses	poultry	toilet paper	vegetables
9	1/4/2000	juice	poultry	coffee/tea	coffee/tea	dishwashing liquid/detergent				
10	1/5/2000	ketchup	coffee/tea	toilet paper	pork	flour	milk	soda	dishwashing liquid/detergent	eggs
11	1/5/2000	sandwich loaves	ice cream	soda	bagels	dishwashing liquid/detergent	eggs	sugar	waffles	individual meals
12	1/6/2000	pork	tortillas	pork	shampoo	lunch meat	pasta	juice	bagels	vegetables
13	1/7/2000	sugar	fruits	all- purpose	aluminum foil	laundry detergent	individual meals	flour	pork	shampoo
14	1/7/2000	fruits	dinner rolls	individual meals	shampoo	ketchup	cereals	sandwich bags	laundry detergent	vegetables
15	1/7/2000	individual meals	ice cream	cereals	paper towels	bagels	mixes	lunch meat	juice	toilet paper
16	1/8/2000	sugar	sandwich bags	flour	juice	milk	paper towels	cereals	sandwich bags	pasta
17	1/8/2000	milk	hand soap	pasta	individual meals	spaghetti sauce	cereals	sandwich loaves	hand soap	individual meals
18	1/8/2000	sandwich bags	toilet paper	bagels	shampoo	coffee/tea				
19	1/9/2000	individual meals	laundry detergent	coffee/tea	eggs	aluminum foil	beef	juice	flour	sugar
20	1/10/2000	shampoo	dishwashing liquid/detergent	yogurt	juice	sugar	soap	sandwich loaves	butter	sandwich loaves
21	1/11/2000	waffles	fruits	all- purpose	pork	juice	bagels	mixes		
22	1/11/2000	cheeses	vegetables	cereals	sugar	bagels	soda			
23	1/11/2000	vegetables	vegetables	aluminum foil	bagels	vegetables	shampoo	shampoo	vegetables	dishwashing liquid/detergent
24	1/11/2000	fruits	all- purpose	pasta	cheeses	juice	sandwich bags	vegetables	sandwich loaves	coffee/tea
25	1/11/2000	bagels	sugar	pork	sandwich loaves	tortillas	ice cream	all- purpose	yogurt	fruits
26	1/12/2000	fruits	sandwich loaves	vegetables	coffee/tea	aluminum foil	vegetables	shampoo	cereals	dinner rolls
27	1/13/2000	laundry detergent	pork	pasta	cheeses	fruits	sugar	lunch meat	laundry detergent	paper towels
28	1/13/2000	pork	bagels	poultry	pasta	butter	all- purpose	pasta	shampoo	sugar
29	1/13/2000	pasta	butter	sandwich loaves	spaghetti sauce	juice	dinner rolls	all- purpose	pork	yogurt

Figure 02

[miles-away/market-basket-analysis | Workspace | data.world]

2. Sort the data in each column according to their ascending order.

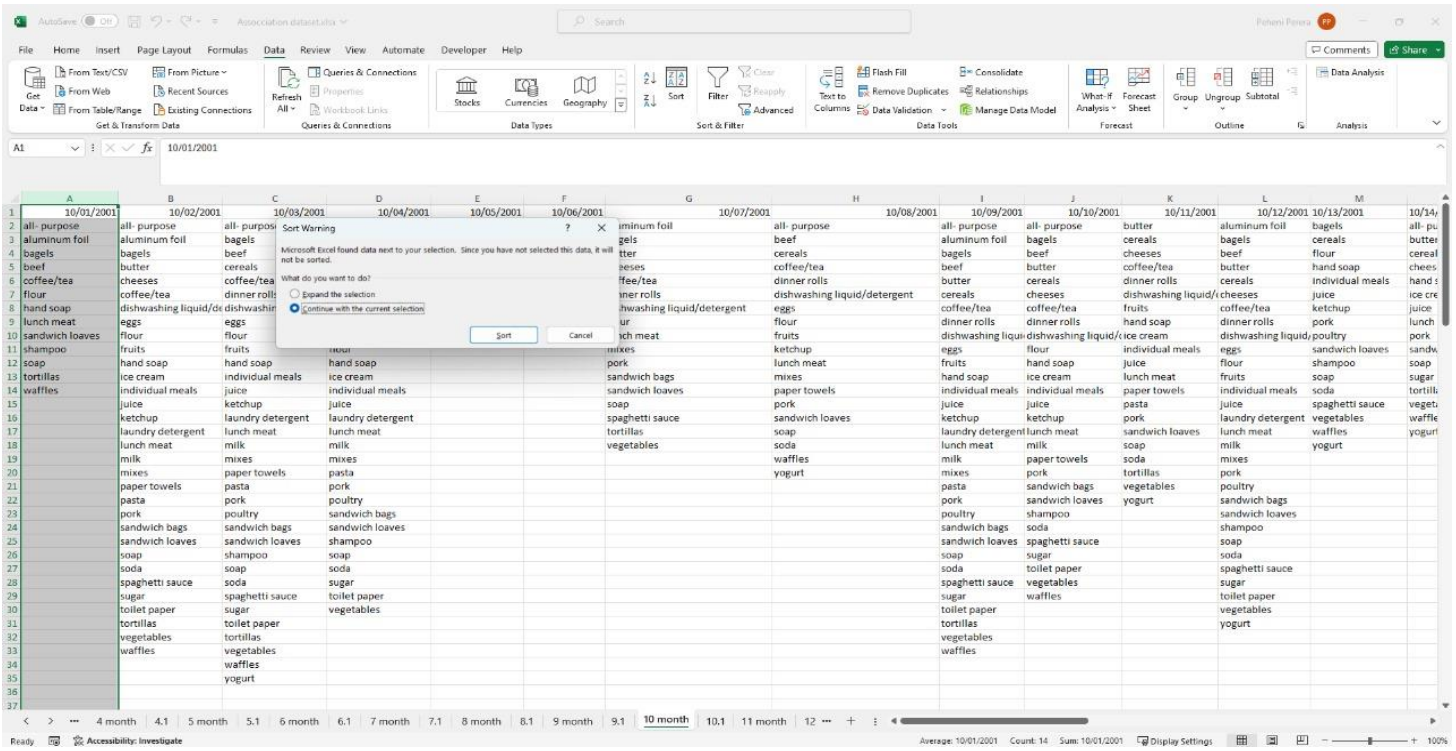

Figure 03

3. Remove duplicates from each column of the dataset.

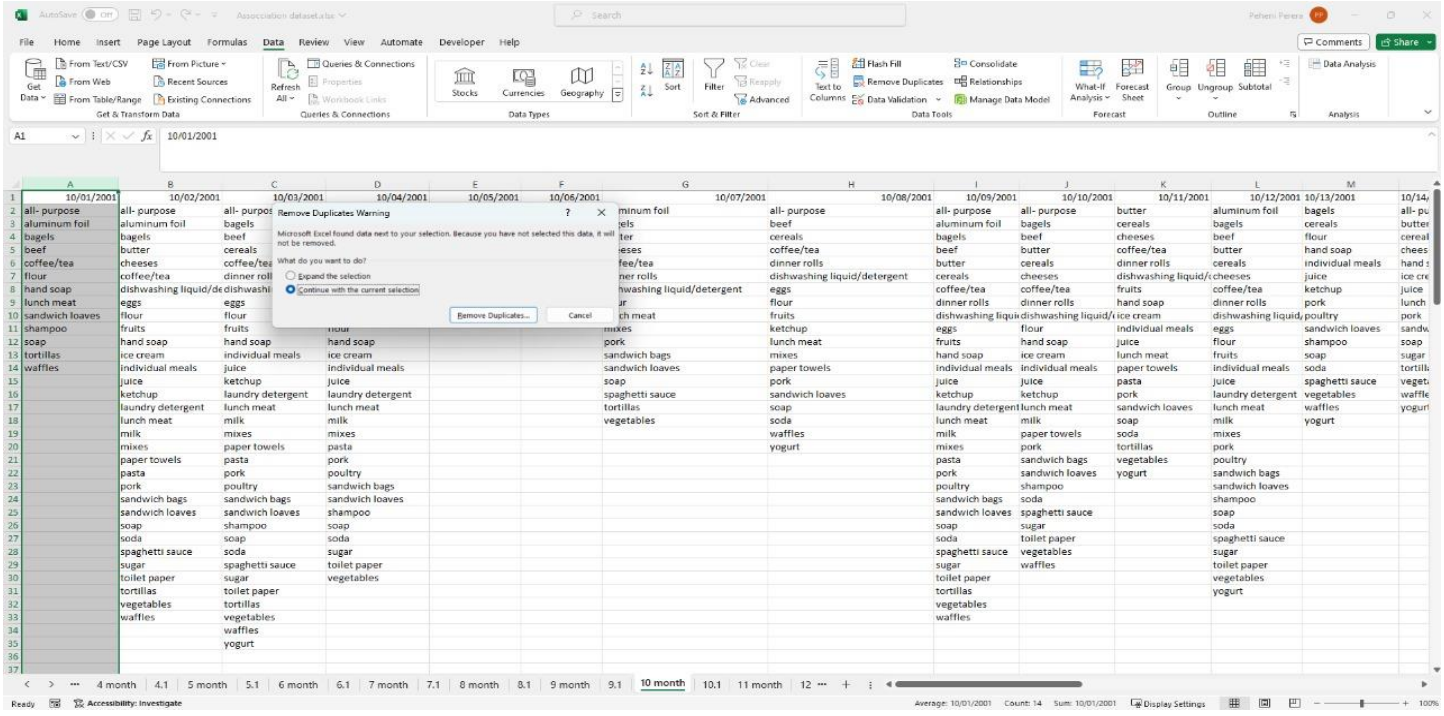

Figure 04 (a)

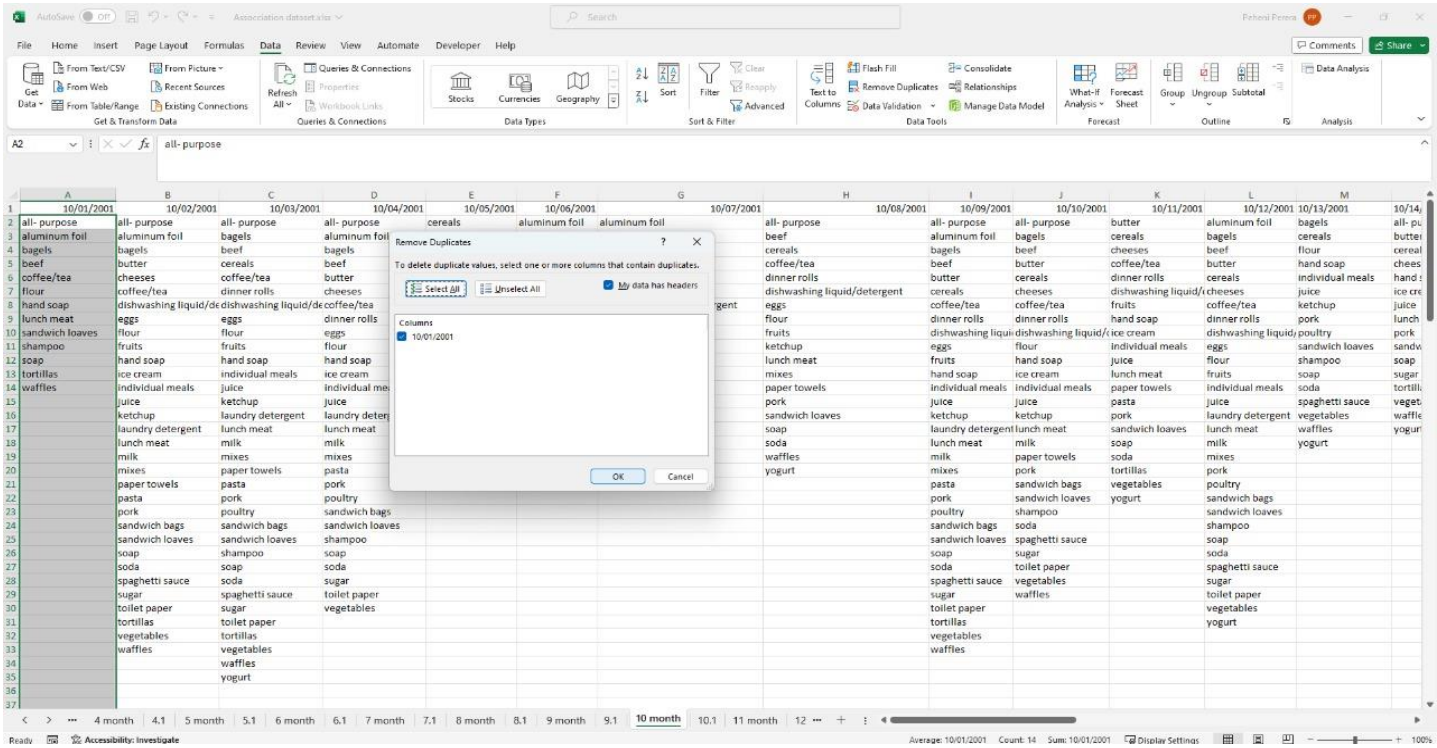

Figure 04 (b)

4. Transpose the order of the columns and the rows of the dataset.

Date	all-purpose	aluminum foil	bagels	beef	butter	cereals	cheese	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour
09/01/2001	all- purpose	aluminum foil	butter	cereals	cheeses	dinner rolls	dishwashing liquid/detergent	eggs	flour	fruits	individual meals	ketchup
09/02/2001	bagels	dinner rolls	dishwashing liquid/detergent	fruits	ketchup	poultry	shampoo					
09/03/2001	bagels	beef	lunch meat	shampoo	vegetables	waffles						
09/04/2001	all- purpose	aluminum foil	bagels	beef	butter	cereals	cheeses	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour
09/05/2001	dinner rolls	lunch meat	mixes	spaghetti sauce	vegetables							
09/06/2001	all- purpose	aluminum foil	bagels	beef	cereals	cheeses	dinner rolls	dishwashing liquid/detergent	eggs	flour	fruits	hand so
09/07/2001	bagels	beef	cereals	dishwashing liquid/detergent	flour	fruits	hand soap	individual meals	juice	ketchup	laundry detergen	lunch m
09/08/2001	aluminum foil	bagels	butter	dinner rolls	eggs	flour	fruits	hand soap	ice cream	juice	ketchup	milk
09/09/2001	all- purpose	aluminum foil	bagels	cereals	cheeses	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour	individual meals	juice
09/10/2001	butter	lunch meat	shampoo	soap	sugar	toilet paper						
09/11/2001	all- purpose	aluminum foil	beef	cereals	cheeses	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour	fruits	hand so
09/12/2001	aluminum foil	bagels	beef	butter	cereals	cheeses	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour	hand so
9/13/2001	all- purpose	butter	cereals	dinner rolls	dishwashing liquid/detergent	flour	fruits	hand soap	ice cream	individual meals	laundry detergen	lunch m
9/14/2001	all- purpose	aluminum foil	bagels	beef	butter	dinner rolls	flour	fruits	juice	ketchup	lunch meat	mixes
9/15/2001	all- purpose	aluminum foil	beef	butter	cereals	dinner rolls	eggs	flour	hand soap	ice cream	individual meals	laundry
9/16/2001	all- purpose	bagels	beef	butter	cereals	cheeses	coffee/tea	dishwashing liquid/detergent	eggs	fruits	ice cream	individu
9/17/2001	all- purpose	aluminum foil	coffee/tea	dinner rolls	hand soap	shampoo	vegetables					
9/18/2001	bagels	eggs	individual meals	juice	ketchup	laundry detergent	mixes	pasta	soap	vegetables		
9/19/2001	all- purpose	aluminum foil	bagels	cheeses	dinner rolls	eggs	laundry detergent	poultry	sandwich bags	shampoo	soap	soda
9/20/2001	beef	cereals	cheeses	coffee/tea	fruits	ice cream	individual meals	juice	ketchup	laundry detergent	lunch meat	milk
9/21/2001	all- purpose	bagels	beef	butter	cereals	cheeses	dishwashing liquid/detergent	eggs	flour	fruits	individual meals	juice
9/22/2001	aluminum foil	bagels	beef	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour	hand soap	ice cream	individual meals	ketchup
9/23/2001	cheeses	flour	ketchup	milk	sandwich bags	sandwich loaves	soda	spaghetti sauce	sugar	vegetables		
9/24/2001	aluminum foil	bagels	cheeses	coffee/tea	dinner rolls	flour	individual meals	ketchup	milk	pork	spaghetti sauce	tortillas
9/25/2001	bagels	cereals	cheeses	coffee/tea	eggs	flour	fruits	hand soap	ice cream	individual meals	juice	laundry
9/26/2001	all- purpose	aluminum foil	bagels	beef	butter	cereals	cheeses	coffee/tea	dishwashing liquid/detergent	eggs	flour	fruits
9/27/2001	all- purpose	aluminum foil	cereals	coffee/tea	eggs	individual meals	juice	paper towels	pork	sandwich bags	sandwich loaves	shampo
9/28/2001	all- purpose	aluminum foil	beef	butter	flour	individual meals	ketchup	lunch meat	pasta	poultry	sandwich loaves	soap
9/29/2001	aluminum foil	bagels	beef	butter	cheeses	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour	hand soap	ice crea
9/30/2001	all- purpose	aluminum foil	butter	cereals	cheeses	coffee/tea	dishwashing liquid/detergent	eggs	flour	hand soap	ice cream	individu

Figure 05

5. Replace the characters attributes of the dataset with “Yes” or “No”.

Date	all-purpose	aluminum foil	bagels	beef	butter	cereals	cheese	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour	fruits	hand soap	ice cream	indiv
01/09/2001	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	No	Yes
02/09/2001	No	No	Yes	No	No	No	No	No	Yes	Yes	No	No	Yes	No	No	No
03/09/2001	No	No	Yes	Yes	No	No	No	No	No	No	No	No	No	No	No	No
04/09/2001	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
05/09/2001	No	No	No	No	No	No	No	No	Yes	No	No	No	No	No	No	No
06/09/2001	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
07/09/2001	No	No	Yes	Yes	No	Yes	No	No	No	Yes	No	Yes	Yes	Yes	No	Yes
08/09/2001	No	Yes	Yes	No	Yes	No	No	No	Yes	No	Yes	Yes	Yes	Yes	Yes	No
09/09/2001	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	Yes
10/09/2001	No	No	No	No	Yes	No	No	No	No	No	No	No	No	No	No	No
11/09/2001	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
12/09/2001	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
13/09/2001	Yes	No	No	No	Yes	Yes	No	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
14/09/2001	Yes	Yes	Yes	Yes	Yes	No	No	No	Yes	No	No	Yes	Yes	No	No	No
15/09/2001	Yes	Yes	No	Yes	Yes	Yes	No	No	Yes	No	Yes	Yes	No	Yes	Yes	Yes
16/09/2001	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	No	Yes	Yes
17/09/2001	Yes	Yes	No	No	No	No	No	Yes	Yes	No	No	No	No	Yes	No	No
18/09/2001	No	No	Yes	No	No	No	No	No	No	No	Yes	No	No	No	No	Yes
19/09/2001	Yes	Yes	Yes	No	No	No	Yes	No	Yes	No	Yes	No	No	No	No	No
20/09/2001	No	No	No	Yes	No	Yes	Yes	Yes	No	No	No	No	Yes	No	Yes	Yes
21/09/2001	Yes	No	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	No	Yes
22/09/2001	No	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
23/09/2001	No	No	No	No	No	No	Yes	No	No	No	No	Yes	No	No	No	No
24/09/2001	No	Yes	Yes	No	No	No	Yes	Yes	Yes	No	No	Yes	No	No	Yes	Yes
25/09/2001	No	No	Yes	No	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes
26/09/2001	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
27/09/2001	Yes	Yes	No	No	No	Yes	No	Yes	No	No	Yes	No	No	No	No	Yes
28/09/2001	Yes	Yes	No	Yes	Yes	No	No	No	No	No	No	Yes	No	No	No	Yes
29/09/2001	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
30/09/2001	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes

Figure 06

6. The final dataset which is used for the data analysis.

Date	all-purpose	aluminum foil	bagels	beef	butter	cereals	cheese	coffee/tea	dinner rolls	dishwashing liquid/detergent	eggs	flour	fruits	hand soap	ice cream	individual meals	juice	ketchup	laundry detergent	lunch meat	milk	mixes	paper towels	pasta
1/1/2000	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	No
2/1/2000	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
3/1/2000	Yes	No	No	Yes	No	No	No	No	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes
4/1/2000	No	No	No	No	No	No	Yes	Yes	No	Yes	No	No	No	No	Yes	No	Yes	No	No	No	No	No	Yes	No
5/1/2000	No	No	Yes	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	No	Yes
6/1/2000	No	No	Yes	No	Yes	No	No	No	No	No	No	Yes	No	Yes	No	No	Yes	No	Yes	Yes	No	No	No	Yes
7/1/2000	Yes	Yes	Yes	No	Yes	Yes	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
8/1/2000	No	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	No	No	No	Yes	No	Yes	Yes
9/1/2000	No	Yes	Yes	Yes	Yes	No	No	Yes	No	No	Yes	Yes	No	No	No	Yes	Yes	No	Yes	No	No	No	No	No
10/1/2000	No	Yes	No	Yes	Yes	No	Yes	Yes	No	Yes	No	No	No	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes	Yes	No
11/1/2000	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
12/1/2000	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No	No	Yes	No	Yes	Yes	No
13/01/2000	Yes	No	Yes	No	Yes	Yes	Yes	No	Yes	No	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
14/01/2000	No	No	Yes	No	No	No	Yes	No	No	No	No	Yes	No	No	No	No	No	No	No	No	No	No	No	No
15/01/2000	Yes	Yes	No	Yes	No	Yes	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
16/01/2000	No	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No
17/01/2000	Yes	Yes	No	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
18/01/2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
19/01/2000	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No	No
20/01/2000	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes
21/01/2000	Yes	Yes	No	Yes	Yes	No	No	No	Yes	No	No	No	No	No	Yes	No	Yes	No	Yes	No	Yes	Yes	Yes	Yes
22/01/2000	No	No	Yes	No	Yes	Yes	No	Yes	No	Yes	No	Yes	Yes	Yes	Yes	No	Yes	No	No	Yes	Yes	No	No	No
23/01/2000	Yes	Yes	Yes	Yes	No	No	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
24/01/2000	No	No	Yes	No	Yes	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No	No	Yes
25/01/2000	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No	Yes	Yes	No	No	Yes	No	Yes	No	No	Yes	Yes
26/01/2000	No	No	Yes	No	No	Yes	No	Yes	Yes	Yes	No	Yes	Yes	No	No	No	Yes	No	Yes	Yes	Yes	No	Yes	Yes
27/01/2000	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No
28/01/2000	No	No	No	Yes	No	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
29/01/2000	No	No	No	Yes	Yes	No	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	No
30/01/2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
31/01/2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes
1/2/2000	Yes	Yes	No	Yes	No	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	No
2/2/2000	No	No	No	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes
3/2/2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Figure 01

1.4. Data mining

The data mining method utilized for the dataset is association rule mining using R programming language.

Apriori algorithm is the technique used for the association rule mining of the dataset.

Using the apriori algorithm, we find the associated support and the associated confidence of the dataset where we generate and derive the association rules of the dataset.

Visualization tools used:

1. barplot to explore the dataset.

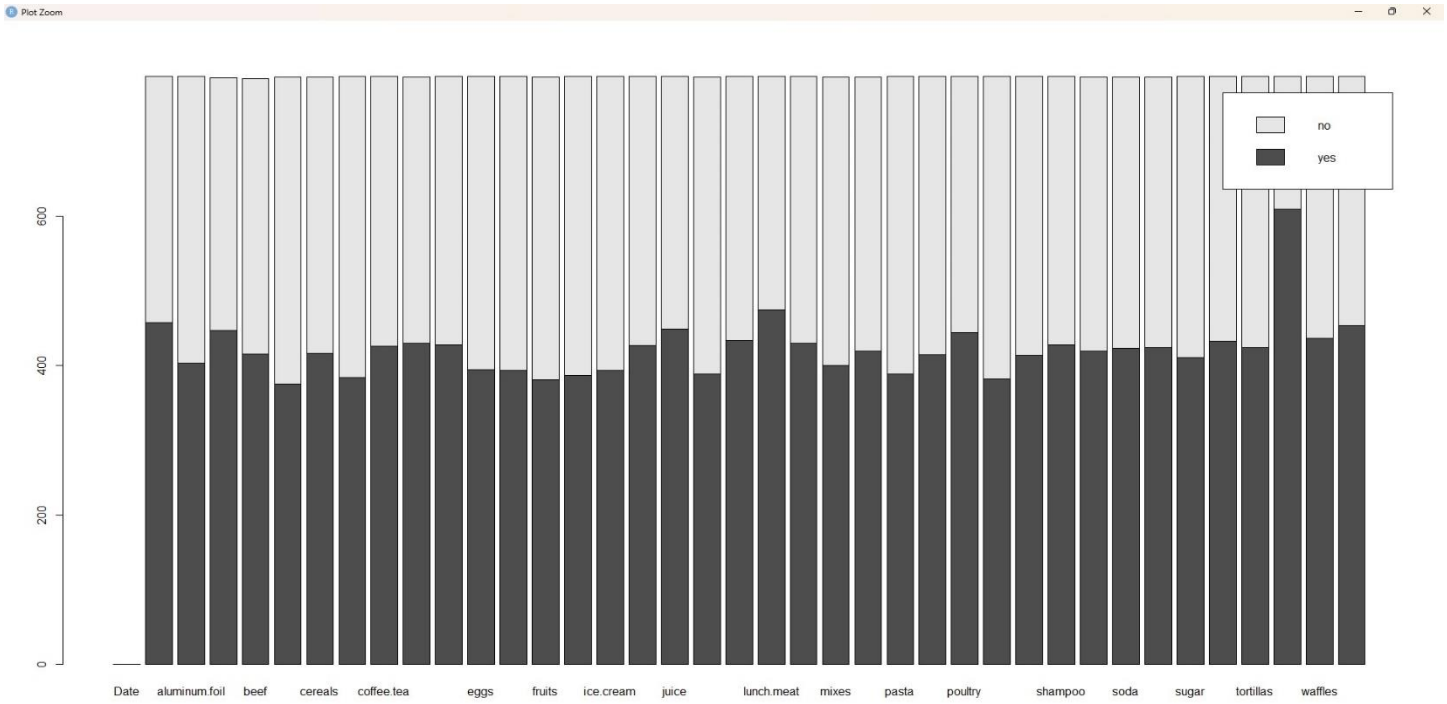

Figure 07 (a)

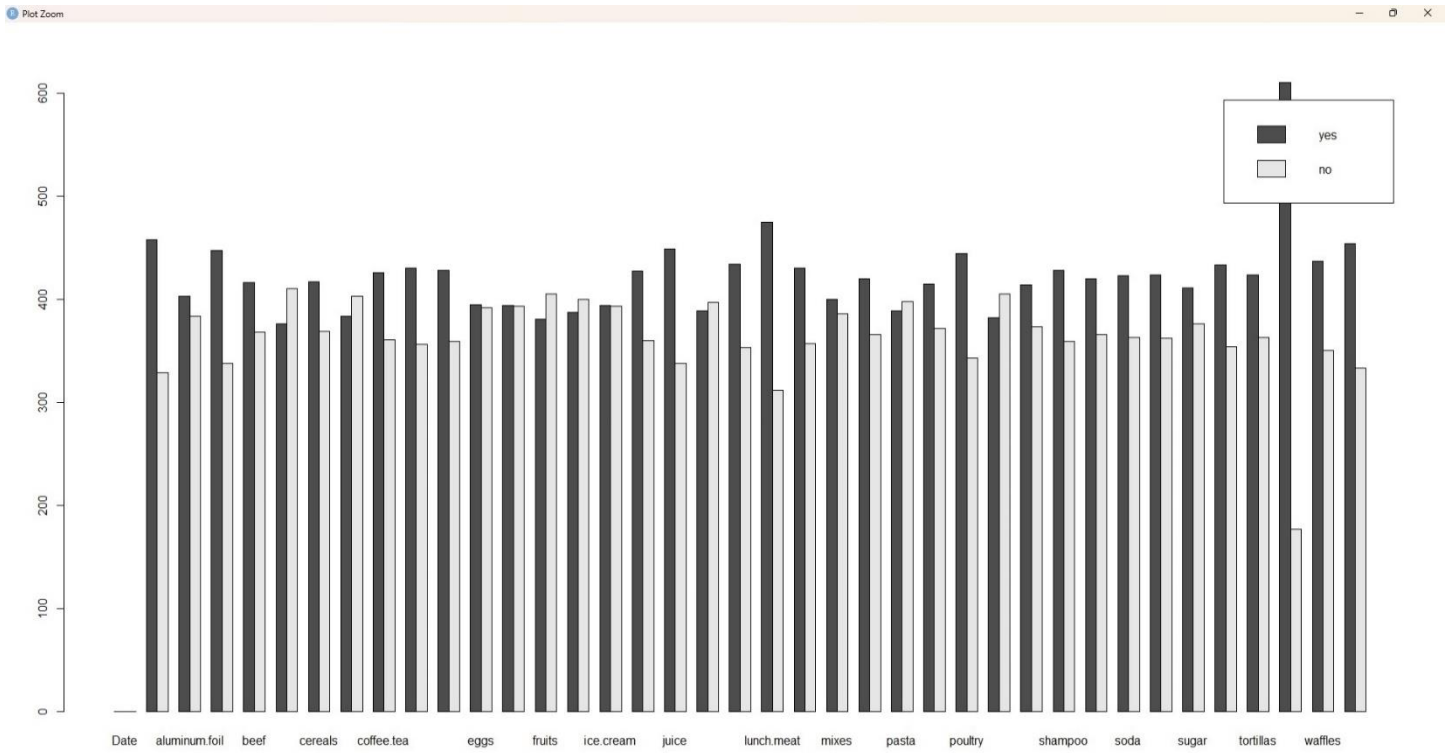

Figure 07 (b)

2. plot to;
    i. plot rules.

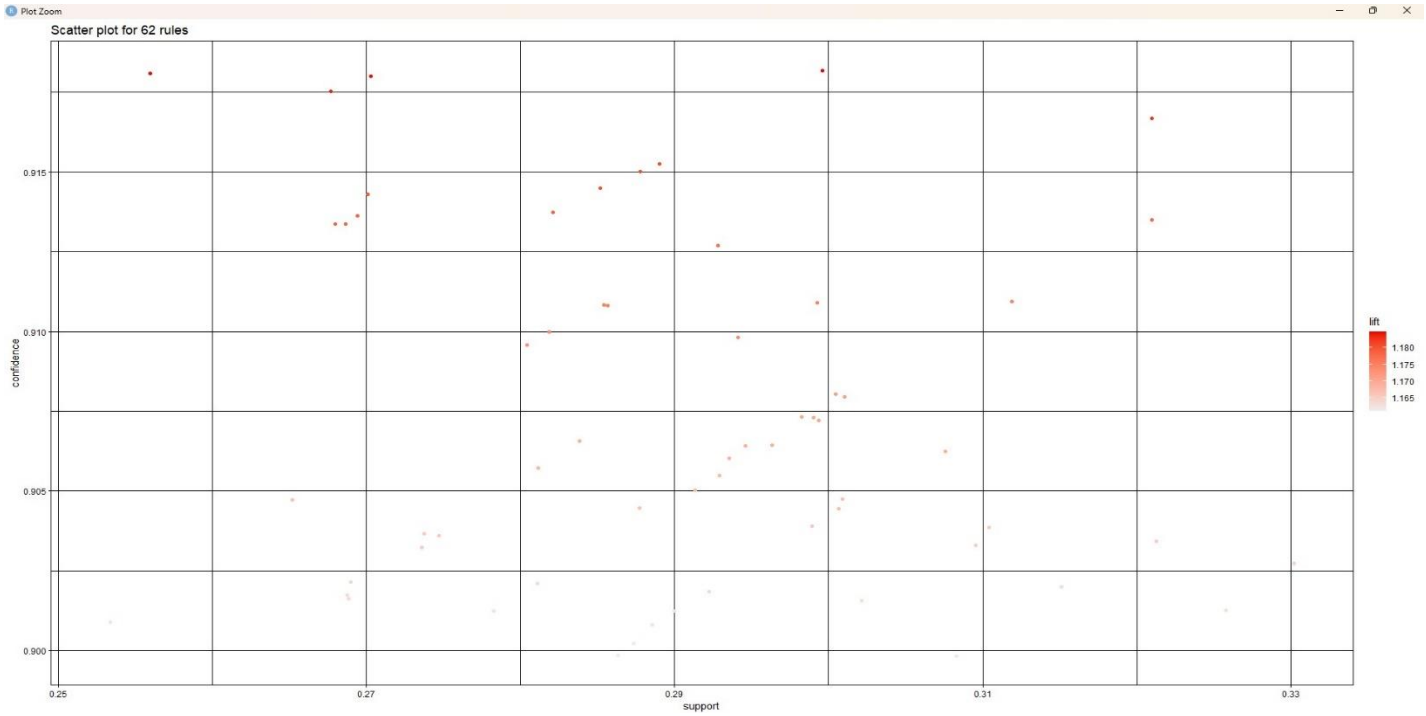

Figure 08

ii. plot rules in groups.

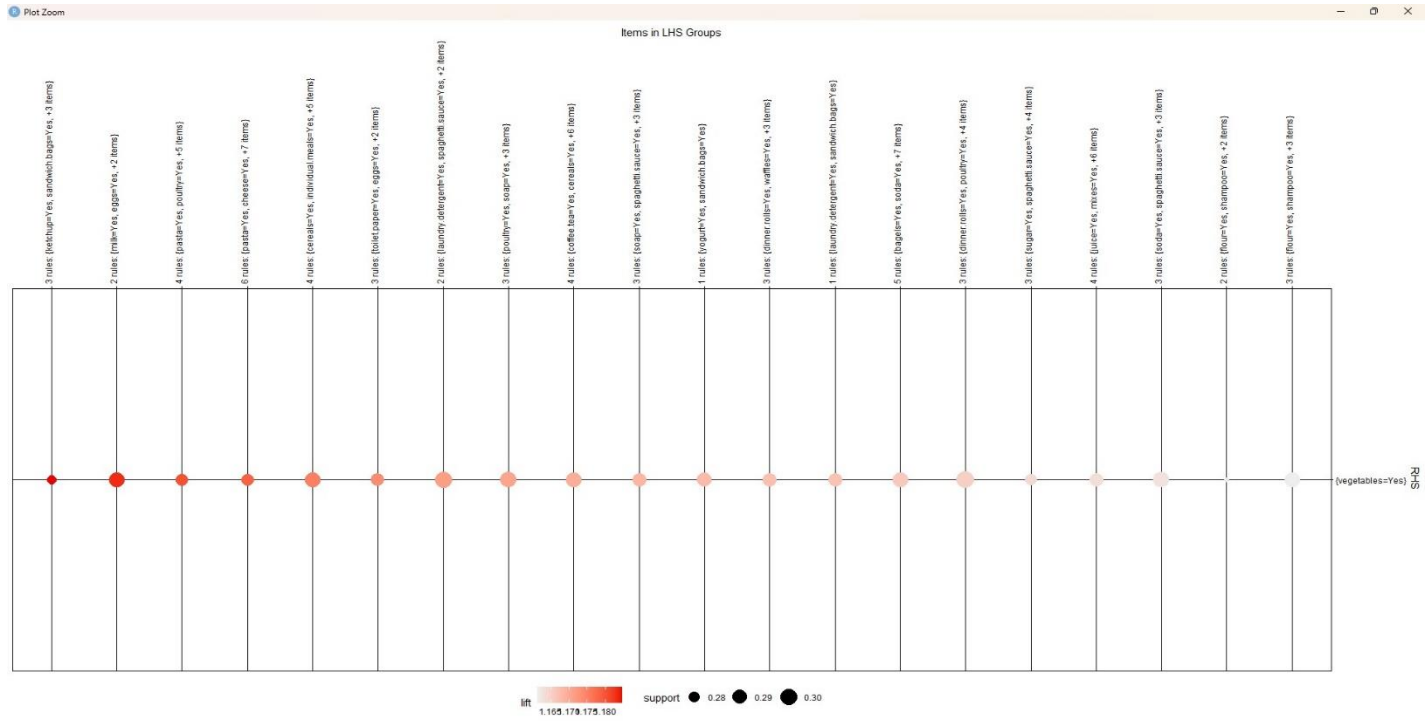

Figure 09

iii. display the support, confidence and lift with a scatterplot matrix.

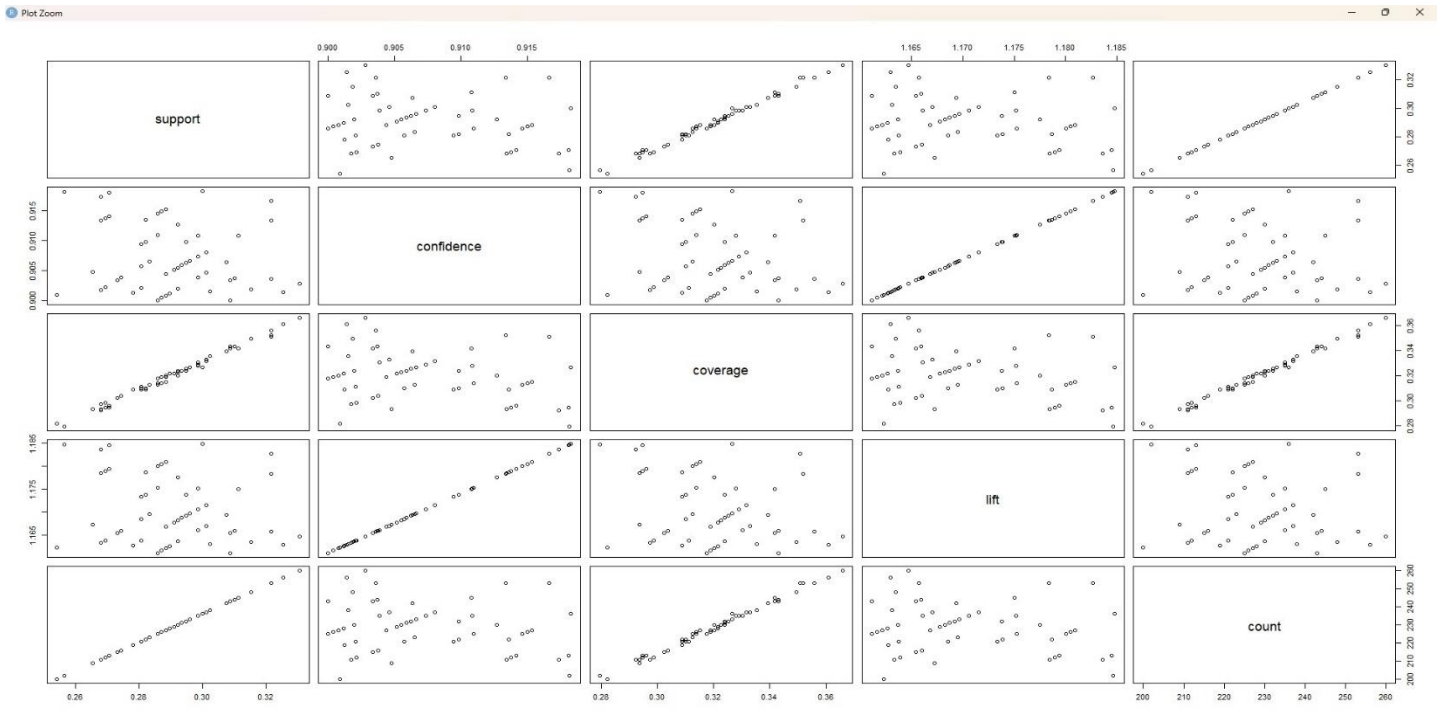

Figure 10

3. scatterplot to plot the association rules.

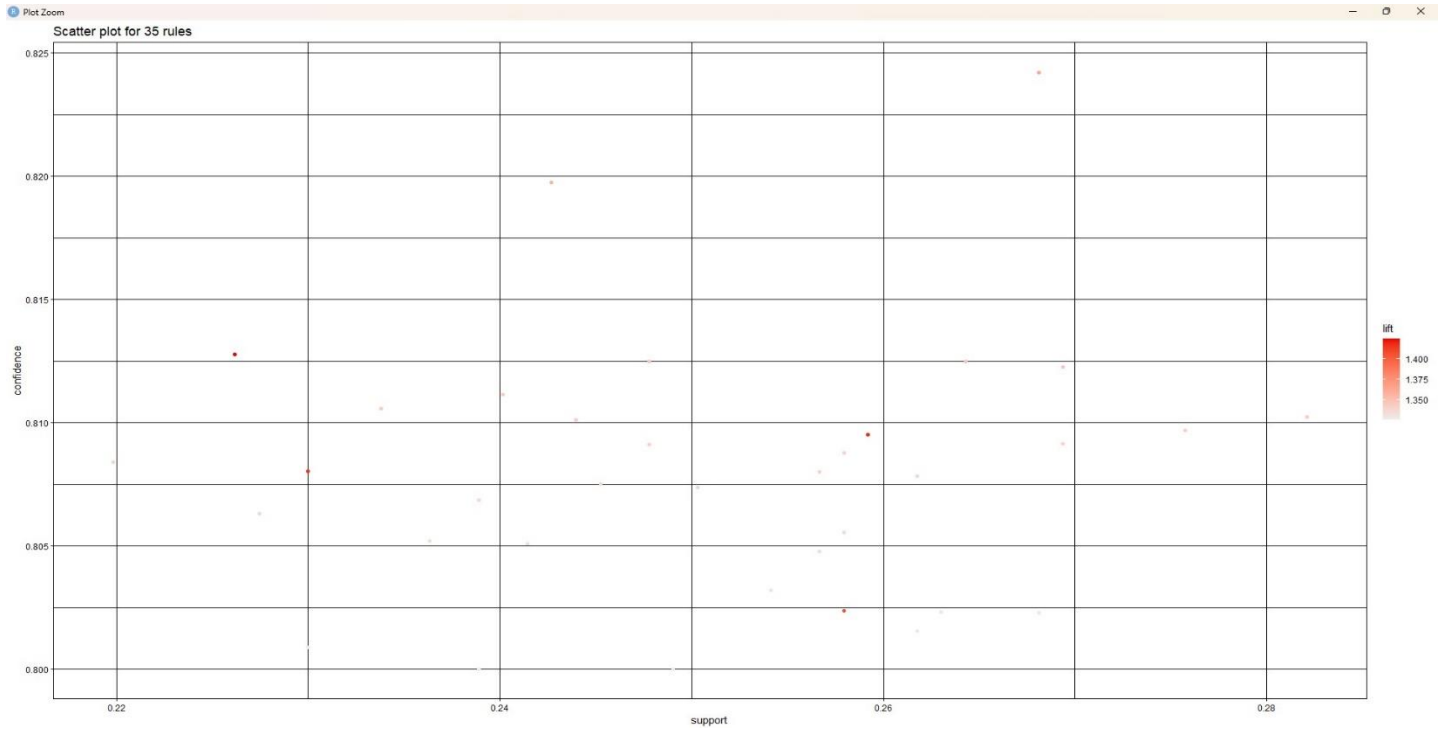

Figure 11 (a)

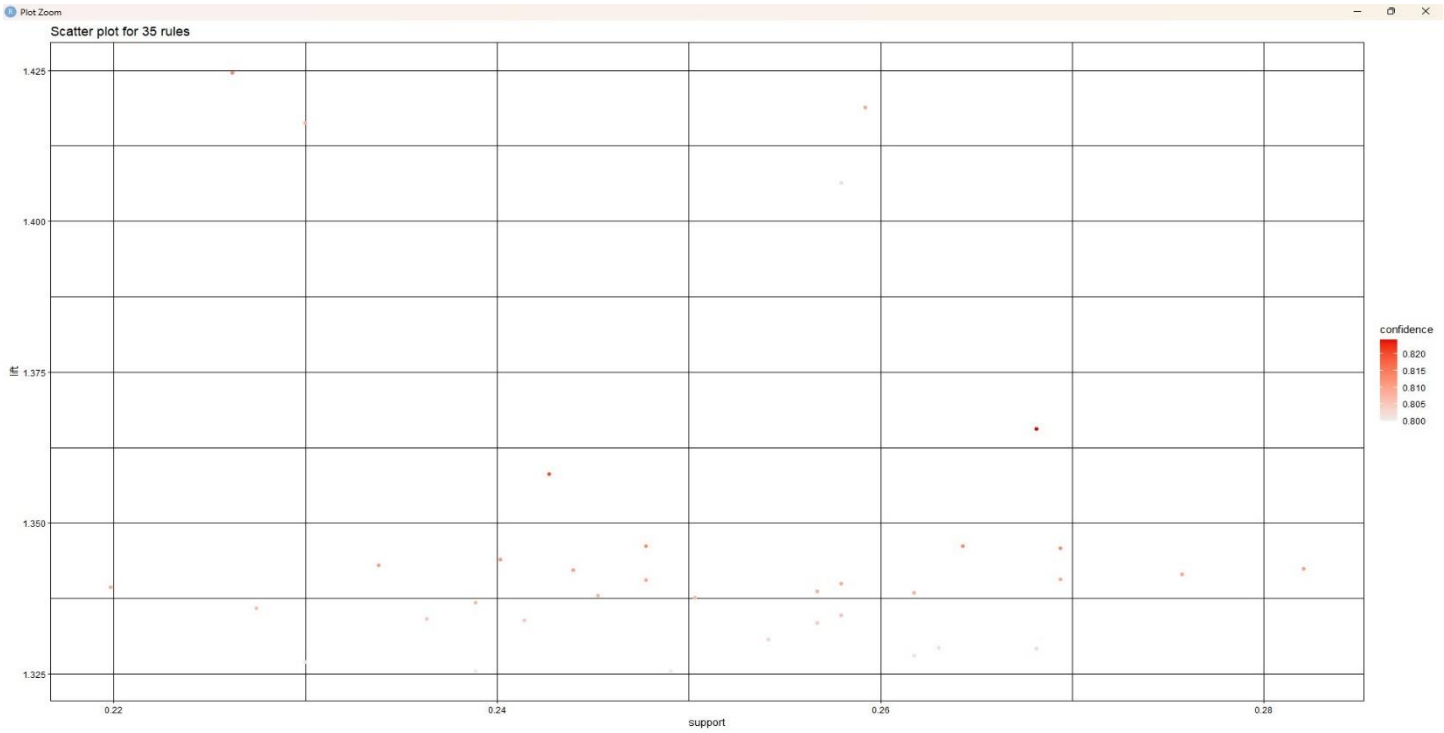

Figure 11 (b)

1.5. Implementation in R

- Install packages to R.
- The R packages installed are as follows:
  1. **arules** – it offers the infrastructure for representing, manipulating, and analyzing transaction data and patterns using frequent itemsets and association rules.
  2. **arulesViz** – it is the extension of the arules package in R, which is specifically designed to provide various visualization techniques for association rules and frequent item sets in the dataset.
- Importing, viewing and reading the dataset file.

```
# 1 - Read the data file

marketbasket = read.csv("Associationfinal.csv", header = TRUE, colClasses = "factor")
```

Figure 12

- Inspecting the dataset. For our convenience, checking the head (which shows the 1st 6 rows of the dataset), the tail (which shows the last 6 rows of the dataset), the str (string) and the summary of the dataset.

```
# 2 - Inspect the dataset

names(marketbasket)
head(marketbasket)
tail(marketbasket)
summary(marketbasket)
str(marketbasket)
```

Figure 13

```
> names(marketbasket)
 [1] "Date"                        "all.purpose"                 "aluminum.foil"               "bagels"
 [5] "beef"                        "butter"                      "cereals"                     "cheese"
 [9] "coffee.tea"                  "dinner.rolls"                "dishwashing.liquid.detergent" "eggs"
[13] "flour"                       "fruits"                      "hand.soap"                   "ice.cream"
[17] "individual.meals"            "juice"                       "ketchup"                     "laundry.detergent"
[21] "lunch.meat"                  "milk"                        "mixes"                       "paper.towels"
[25] "pasta"                       "pork"                        "poultry"                     "sandwich.bags"
[29] "sandwich.loaves"             "shampoo"                     "soap"                        "soda"
[33] "spaghetti.sauce"             "sugar"                       "toilet.paper"                "tortillas"
[37] "vegetables"                  "waffles"                     "yogurt"
> head(marketbasket)
       Date all.purpose aluminum.foil bagels beef butter cereals cheese coffee.tea dinner.rolls dishwashing.liquid.detergent eggs flour
1 01/01/2000         Yes           Yes     No  Yes    Yes     Yes    Yes         No          Yes                            Yes   No   Yes
2 02/01/2000         Yes           Yes    Yes   No    Yes     Yes    Yes        Yes          Yes                            Yes  Yes   Yes
3 03/01/2000         Yes            No     No  Yes     No      No     No         No           No                            Yes  Yes    No
4 04/01/2000          No            No     No   No     No      No    Yes        Yes           No                            Yes   No    No
5 05/01/2000          No            No    Yes  Yes     No      No    Yes        Yes          Yes                            Yes  Yes   Yes
6 06/01/2000          No            No    Yes   No    Yes      No     No         No           No                             No   No   Yes
  fruits hand.soap ice.cream individual.meals juice ketchup laundry.detergent lunch.meat milk mixes paper.towels pasta pork poultry
1     No       Yes       Yes              Yes    No      No               Yes        Yes  Yes   Yes           No    No  Yes      No
2     No       Yes       Yes              Yes   Yes     Yes               Yes        Yes  Yes   Yes          Yes   Yes  Yes     Yes
3    Yes        No       Yes              Yes   Yes     Yes                No         No  Yes   Yes          Yes   Yes   No      No
4     No        No       Yes               No   Yes      No                No         No   No    No          Yes    No   No     Yes
5    Yes       Yes       Yes              Yes   Yes     Yes                No        Yes  Yes    No           No   Yes  Yes     Yes
6     No       Yes        No              No    Yes      No               Yes        Yes   No    No           No   Yes  Yes      No
  sandwich.bags sandwich.loaves shampoo soap soda spaghetti.sauce sugar toilet.paper tortillas vegetables waffles yogurt
1           Yes              No     Yes  Yes  Yes              No    No          Yes       Yes        Yes     Yes    Yes
2            No             Yes     Yes  Yes  Yes             Yes    No          Yes       Yes        Yes     Yes    Yes
3           Yes              No      No  Yes   No             Yes    No          Yes       Yes        Yes      No    Yes
4            No              No      No   No  Yes              No    No          Yes        No        Yes     Yes     No
5            No             Yes     Yes  Yes  Yes              No   Yes          Yes       Yes        Yes     Yes     No
6            No              No     Yes   No   No             Yes   Yes           No       Yes        Yes     Yes    Yes
```

Figure 14 (a)

```
> tail(marketbasket)
       Date all.purpose aluminum.foil bagels beef butter cereals cheese coffee.tea dinner.rolls dishwashing.liquid.detergent eggs flour
782 21/02/2002          No           Yes    Yes  Yes    Yes      No    Yes         No          Yes                            Yes  Yes    No
783 22/02/2002         Yes           No      No  Yes     No     Yes    Yes        Yes          Yes                            Yes   No    No
784 23/02/2002          No           No      No   No     No     Yes    Yes        Yes          Yes                            Yes  Yes    No
785 24/02/2002          No           No     Yes  Yes     No      No     No         No          Yes                            Yes  Yes    No
786 25/02/2002         Yes           No      No  Yes     No      No    Yes         No           No                             No   No    No
787 26/02/2002          No           No      No   No     No      No     No         No           No                             No   No    No
    fruits hand.soap ice.cream individual.meals juice ketchup laundry.detergent lunch.meat milk mixes paper.towels pasta pork poultry
782    Yes       Yes       Yes               No   Yes     Yes               Yes        Yes  Yes    No           No    No  Yes      Yes
783    Yes       Yes       Yes               No   Yes      No                No         No  No    Yes          Yes   Yes  Yes      Yes
784     No       Yes        No              Yes    No      No                No        Yes  Yes   Yes           No   Yes   No      Yes
785     No       Yes        No              Yes   Yes      No                No        Yes   No    No          Yes    No  Yes      Yes
786     No        No        No               No    No     Yes                No         No  Yes    No          Yes    No   No      Yes
787     No        No        No               No    No      No               Yes         No   No    No           No    No   No       No
    sandwich.bags sandwich.loaves shampoo soap soda spaghetti.sauce sugar toilet.paper tortillas vegetables waffles yogurt
782           Yes             Yes     Yes  Yes   No              Yes   Yes          Yes       Yes        Yes      No     No
783           Yes              No     Yes  Yes   No               No   Yes          Yes        No         No      No     No
784            No              No      No  Yes   No              Yes    No           No        No        Yes      No     No
785            No              No     Yes  Yes   No               No   Yes           No       Yes        Yes      No    Yes
786           Yes              No      No  Yes  Yes              Yes   Yes          Yes        No        Yes     Yes    Yes
787            No              No     Yes   No  Yes               No    No           No        No        Yes      No     No
> summary(marketbasket)
         Date     all.purpose aluminum.foil bagels     beef       butter    cereals   cheese    coffee.tea dinner.rolls
 01/01/2000:  1   No :329     No :384         :  2      :  3   No   :410      :  1   No :403   No :361       :  1
 01/01/2001:  1   Yes:458     Yes:403       No :338   No :368   Yes  :376   No :369   Yes:384   Yes:426     No :356
 01/01/2002:  1                           Yes:447   Yes:416   YesNo:  1   Yes:417                       Yes:430
 01/02/2000:  1
 01/02/2001:  1
 01/02/2002:  1
 (other)   :781
 dishwashing.liquid.detergent  eggs     flour     fruits    hand.soap ice.cream individual.meals juice     ketchup   laundry.detergent
 No :359                      No :392   No :393      :  1   No :400   No :393   No :360          No :338      :  1   No :353
 Yes:428                      Yes:395   Yes:394   No :405   Yes:387   Yes:394   Yes:427          Yes:449   No :397   Yes:434
                                        Yes:381                                                          Yes:389


 lunch.meat  milk     mixes     paper.towels pasta      pork     poultry   sandwich.bags sandwich.loaves shampoo     soap      soda
 No :312    No :357      :  1      :  1      No :398   No :372   No :343   No :405       No :373         No :359   No  :366      :  1
 Yes:475    Yes:430   No :386   No :366      Yes:389   Yes:415   Yes:444   Yes:382       Yes:414         Yes:428   soda:  1   No :363
                      Yes:400   Yes:420                                                                  Yes :420   Yes:423



 spaghetti.sauce sugar     toilet.paper tortillas vegetables waffles   yogurt
 No   :362       No :376   No :354      No :363   No :177    No :350   No :333
 Yes  :424       Yes:411   Yes:433      Yes:424   Yes:610    Yes:437   Yes:454
 YesNo:  1
```

Figure 14 (b)

```
> str(marketbasket)
'data.frame':   787 obs. of  39 variables:
 $ Date                          : Factor w/ 787 levels "01/01/2000","01/01/2001",..: 1 27 53 79 105 131 157 183 209 235 ...
 $ all.purpose                   : Factor w/ 2 levels "No","Yes": 2 2 2 1 1 1 2 1 1 1 ...
 $ aluminum.foil                 : Factor w/ 2 levels "No","Yes": 2 2 1 1 1 1 2 1 2 2 ...
 $ bagels                        : Factor w/ 3 levels "","No","Yes": 2 3 2 2 3 3 3 3 3 2 ...
 $ beef                          : Factor w/ 3 levels "","No","Yes": 3 2 3 2 3 2 2 3 3 3 ...
 $ butter                        : Factor w/ 3 levels "No","Yes","YesNo": 2 2 1 1 1 2 2 1 2 2 ...
 $ cereals                       : Factor w/ 3 levels "","No","Yes": 3 3 2 2 2 2 3 3 2 2 ...
 $ cheese                        : Factor w/ 2 levels "No","Yes": 2 2 1 2 2 1 2 2 1 2 ...
 $ coffee.tea                    : Factor w/ 2 levels "No","Yes": 1 2 1 2 2 1 1 2 2 2 ...
 $ dinner.rolls                  : Factor w/ 3 levels "","No","Yes": 3 3 2 2 3 2 3 3 2 2 ...
 $ dishwashing.liquid.detergent: Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 1 2 1 2 ...
 $ eggs                          : Factor w/ 2 levels "No","Yes": 1 2 2 1 2 1 1 1 2 1 ...
 $ flour                         : Factor w/ 2 levels "No","Yes": 2 2 1 1 2 2 2 2 2 1 ...
 $ fruits                        : Factor w/ 3 levels "","No","Yes": 2 2 3 2 3 2 3 2 2 2 ...
 $ hand.soap                     : Factor w/ 2 levels "No","Yes": 2 2 1 1 2 2 2 2 1 2 ...
 $ ice.cream                     : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 2 2 1 1 ...
 $ individual.meals              : Factor w/ 2 levels "No","Yes": 2 2 2 1 2 1 2 2 2 2 ...
 $ juice                         : Factor w/ 2 levels "No","Yes": 1 2 2 2 2 2 2 2 2 2 ...
 $ ketchup                       : Factor w/ 3 levels "","No","Yes": 2 3 3 2 3 3 2 3 2 2 3 ...
 $ laundry.detergent             : Factor w/ 2 levels "No","Yes": 2 2 1 1 1 2 2 1 2 2 ...
 $ lunch.meat                    : Factor w/ 2 levels "No","Yes": 2 2 1 1 2 2 2 1 1 1 ...
 $ milk                          : Factor w/ 2 levels "No","Yes": 2 2 2 1 2 1 1 2 1 1 ...
 $ mixes                         : Factor w/ 3 levels "","No","Yes": 3 3 3 2 2 2 3 2 2 3 ...
 $ paper.towels                  : Factor w/ 3 levels "","No","Yes": 2 3 3 3 2 2 3 3 2 3 ...
 $ pasta                         : Factor w/ 2 levels "No","Yes": 1 2 2 1 2 2 1 2 1 1 ...
 $ pork                          : Factor w/ 2 levels "No","Yes": 2 2 1 1 2 2 2 1 1 1 ...
 $ poultry                       : Factor w/ 2 levels "No","Yes": 1 2 1 2 2 1 1 1 1 2 ...
 $ sandwich.bags                 : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 1 2 2 1 1 ...
 $ sandwich.loaves               : Factor w/ 2 levels "No","Yes": 1 2 1 1 2 1 2 2 1 2 ...
 $ shampoo                       : Factor w/ 2 levels "No","Yes": 2 2 1 1 2 2 2 2 1 2 ...
 $ soap                          : Factor w/ 3 levels "No","soda","Yes": 3 3 3 1 3 1 3 3 1 3 ...
 $ soda                          : Factor w/ 3 levels "","No","Yes": 3 3 2 3 3 2 2 3 2 3 ...
 $ spaghetti.sauce               : Factor w/ 3 levels "No","Yes","YesNo": 1 2 2 1 1 2 2 2 2 1 ...
 $ sugar                         : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 2 2 2 2 2 ...
 $ toilet.paper                  : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 2 2 1 2 ...
 $ tortillas                     : Factor w/ 2 levels "No","Yes": 2 2 2 1 2 2 2 1 1 2 ...
 $ vegetables                    : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 1 2 2 ...
 $ waffles                       : Factor w/ 2 levels "No","Yes": 2 2 1 2 2 2 1 1 2 1 ...
 $ yogurt                        : Factor w/ 2 levels "No","Yes": 2 2 2 1 1 2 1 1 1 2 ...
>
```

Figure 14 (c)

- Checking the dimensions of the dataset.

```
13  # 3 - Check the dimension of the dataset
14
15  dim(marketbasket)
```

Figure 15

```
> dim(marketbasket)
[1] 787  39
```

Figure 16

- Plotting and exploring the dataset.

```
17  # 4 - Plot and explore the dataset
18
19  yes = colSums(marketbasket == "Yes")
20  yes
21
22  no = colSums(marketbasket == "No")
23  no
24
25  purchased = rbind(yes,no)
26  purchased
27
28  barplot(purchased, legend = rownames(purchased))
29  barplot(purchased, beside = T, legend =  rownames(purchased))
```

Figure 17

```
> yes = colSums(marketbasket == "Yes")
> yes
                      Date                all.purpose               aluminum.foil                     bagels
                         0                        458                        403                        447
                      beef                     butter                   cereals                    cheese
                       416                        376                        417                        384
               coffee.tea               dinner.rolls dishwashing.liquid.detergent                      eggs
                       426                        430                        428                        395
                     flour                     fruits                 hand.soap                 ice.cream
                       394                        381                        387                        394
          individual.meals                      juice                    ketchup           laundry.detergent
                       427                        449                        389                        434
                lunch.meat                      milk                     mixes                paper.towels
                       475                        430                        400                        420
                     pasta                      pork                   poultry                sandwich.bags
                       389                        415                        444                        382
           sandwich.loaves                   shampoo                      soap                       soda
                       414                        428                        420                        423
           spaghetti.sauce                     sugar             toilet.paper                   tortillas
                       424                        411                        433                        424
                vegetables                   waffles                     yogurt
                       610                        437                        454
> no = colSums(marketbasket == "No")
> no
                      Date                all.purpose               aluminum.foil                     bagels
                         0                        329                        384                        338
                      beef                     butter                   cereals                    cheese
                       368                        410                        369                        403
               coffee.tea               dinner.rolls dishwashing.liquid.detergent                      eggs
                       361                        356                        359                        392
                     flour                     fruits                 hand.soap                 ice.cream
                       393                        405                        400                        393
          individual.meals                      juice                    ketchup           laundry.detergent
                       360                        338                        397                        353
                lunch.meat                      milk                     mixes                paper.towels
                       312                        357                        386                        366
                     pasta                      pork                   poultry                sandwich.bags
                       398                        372                        343                        405
           sandwich.loaves                   shampoo                      soap                       soda
                       373                        359                        366                        363
           spaghetti.sauce                     sugar             toilet.paper                   tortillas
                       362                        376                        354                        363
                vegetables                   waffles                     yogurt
                       177                        350                        333
```

Figure 18 (a)

```
> purchased = rbind(yes,no)
> purchased
    Date all.purpose aluminum.foil bagels beef butter cereals cheese coffee.tea dinner.rolls dishwashing.liquid.detergent eggs flour fruits
yes    0         458           403    447  416    376     417    384        426          430                          428  395   394    381
no     0         329           384    338  368    410     369    403        361          356                          359  392   393    405
    hand.soap ice.cream individual.meals juice ketchup laundry.detergent lunch.meat milk mixes paper.towels pasta pork poultry sandwich.bags
yes       387       394              427   449     389               434        475  430   400          420   389  415     444            382
no        400       393              360   338     397               353        312  357   386          366   398  372     343            405
    sandwich.loaves shampoo soap soda spaghetti.sauce sugar toilet.paper tortillas vegetables waffles yogurt
yes             414     428  420  423             424   411          433       424        610     437    454
no              373     359  366  363             362   376          354       363        177     350    333
> barplot(purchased, legend = rownames(purchased))
> barplot(purchased, beside = T, legend =  rownames(purchased))
>
```

Figure 18 (b)

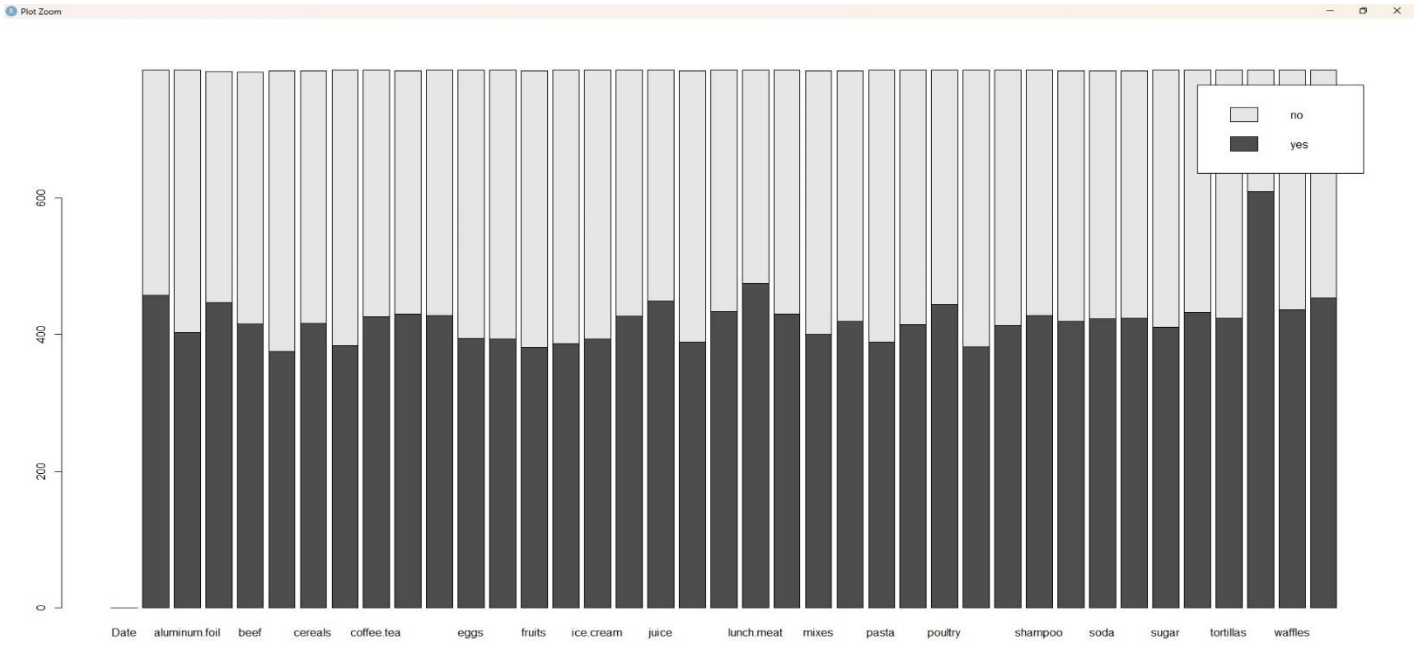

Figure 07 (a)

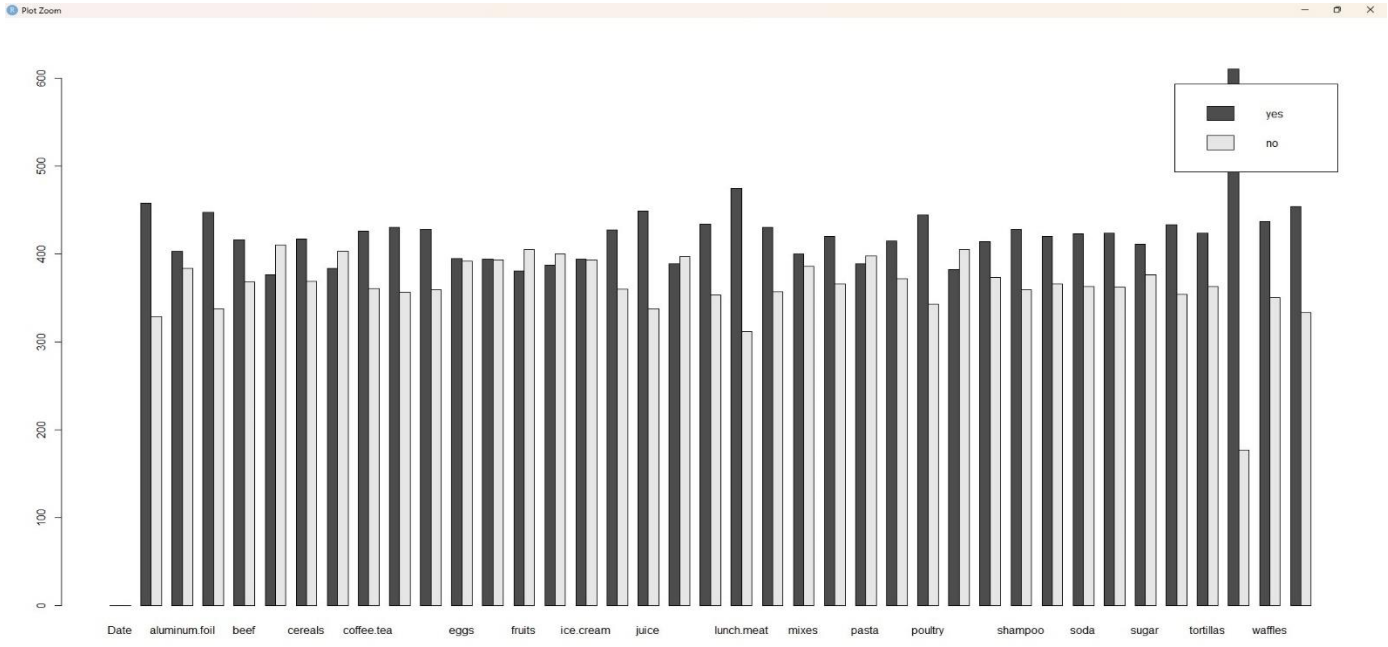

Figure 07 (b)

- Installing “arules” package.

```
31  # 5 - Install "arules" package
32
33  install.packages("arules")
34  library(arules)
35
```

Figure 19

- Creating the association rule.

```
36  # 6 - Create association rules
37
38  rules = apriori(marketbasket)
```

Figure 20

```
> rules = apriori(marketbasket)
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.8    0.1    1 none FALSE            TRUE       5     0.1      1     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 78

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[875 item(s), 787 transaction(s)] done [0.01s].
sorting and recoding items ... [76 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 done [8.72s].
writing ... [29339832 rule(s)] done [25.63s].
creating S4 object  ... done [20.80s].
Warning message:
In apriori(marketbasket) :
  Mining stopped (time limit reached). Only patterns up to a length of 8 returned!
>
```

Figure 21

- Getting the summary of the rules and inspecting them.

```
42  # 7 - Getting the summary of these rules
43
44  summary(rules)
45
46  # 8 - Inspect the rules
47
48  inspect(rules)
```

Figure 22

```
> summary(rules)
set of 29339832 rules

rule length distribution (lhs + rhs):sizes
       2        3        4        5        6        7        8
      37      973    27528   912521  9280875 15290519  3827379

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.000   6.000   7.000   6.749   7.000   8.000

summary of quality measures:
    support         confidence        coverage            lift           count
 Min.   :0.1004   Min.   :0.8000   Min.   :0.1004   Min.   :1.032   Min.   : 79.00
 1st Qu.:0.1017   1st Qu.:0.8318   1st Qu.:0.1169   1st Qu.:1.514   1st Qu.: 80.00
 Median :0.1055   Median :0.8596   Median :0.1233   Median :1.585   Median : 83.00
 Mean   :0.1085   Mean   :0.8645   Mean   :0.1258   Mean   :1.573   Mean   : 85.37
 3rd Qu.:0.1118   3rd Qu.:0.8901   3rd Qu.:0.1309   3rd Qu.:1.657   3rd Qu.: 88.00
 Max.   :0.4956   Max.   :1.0000   Max.   :0.6036   Max.   :2.023   Max.   :390.00

mining info:
         data ntransactions support confidence                          call
 marketbasket           787     0.1        0.8 apriori(data = marketbasket)
> |
```

Figure 23

- Reducing the number of rules to a smaller quantity. (62 rules)

```
50  # 9 - The rules are reduced to a smaller number of rules
51
52  rules = apriori(marketbasket, parameter = list(minlen = 2, maxlen = 3, conf = 0.9))
53
54  ## This has reduced the rules to 62
```

Figure 24

```
> rules = apriori(marketbasket, parameter = list(minlen = 2, maxlen = 3, conf = 0.9))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.9    0.1    1 none FALSE            TRUE       5     0.1      2      3  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 78

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[875 item(s), 787 transaction(s)] done [0.03s].
sorting and recoding items ... [76 item(s)] done [0.00s].
creating transaction tree ... done [0.01s].
checking subsets of size 1 2 3 done [0.02s].
writing ... [62 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
Warning message:
In apriori(marketbasket, parameter = list(minlen = 2, maxlen = 3,  :
  Mining stopped (maxlen reached). Only patterns up to a length of 3 returned!
> |
```

Figure 25

- Getting the summary of the rules, inspecting them and getting the summary of the dataset.

```
56  # 10 - Getting the summary of these rules
57
58  summary(rules)
59
60  # 11 - Inspect the rules
61
62  inspect(rules)
63
64  # 12 - Get the summary of the dataset
65
66  summary(marketbasket)
```

Figure 26

```
> summary(rules)
set of 62 rules

rule length distribution (lhs + rhs):sizes
 3
62

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      3       3       3       3       3       3

summary of quality measures:
    support         confidence        coverage           lift           count
 Min.   :0.2541   Min.   :0.9000   Min.   :0.2795   Min.   :1.161   Min.   :200.0
 1st Qu.:0.2789   1st Qu.:0.9029   1st Qu.:0.3088   1st Qu.:1.165   1st Qu.:219.5
 Median :0.2891   Median :0.9061   Median :0.3202   Median :1.169   Median :227.5
 Mean   :0.2901   Mean   :0.9073   Mean   :0.3198   Mean   :1.171   Mean   :228.3
 3rd Qu.:0.3008   3rd Qu.:0.9109   3rd Qu.:0.3313   3rd Qu.:1.175   3rd Qu.:236.8
 Max.   :0.3304   Max.   :0.9183   Max.   :0.3659   Max.   :1.185   Max.   :260.0

mining info:
         data ntransactions support confidence                                                                  call
 marketbasket          787     0.1        0.9 apriori(data = marketbasket, parameter = list(minlen = 2, maxlen = 3, conf = 0.9))
> |
```

Figure 27 (a)

```
> inspect(rules)
     lhs                                                rhs                 support   confidence coverage  lift     count
[1]  {cheese=Yes, sandwich.bags=Yes}                 => {vegetables=Yes} 0.2681067 0.9134199  0.2935197 1.178461 211
[2]  {pasta=Yes, sandwich.bags=Yes}                  => {vegetables=Yes} 0.2706480 0.9141631  0.2960610 1.179420 213
[3]  {ketchup=Yes, sandwich.bags=Yes}                => {vegetables=Yes} 0.2566709 0.9181818  0.2795426 1.184605 202
[4]  {flour=Yes, sandwich.bags=Yes}                  => {vegetables=Yes} 0.2541296 0.9009009  0.2820839 1.162310 200
[5]  {eggs=Yes, sandwich.bags=Yes}                   => {vegetables=Yes} 0.2681067 0.9173913  0.2922490 1.183585 211
[6]  {mixes=Yes, sandwich.bags=Yes}                  => {vegetables=Yes} 0.2681067 0.9017094  0.2973316 1.163353 211
[7]  {aluminum.foil=Yes, sandwich.bags=Yes}          => {vegetables=Yes} 0.2706480 0.9181034  0.2947903 1.184504 213
[8]  {sandwich.bags=Yes, soap=Yes}                   => {vegetables=Yes} 0.2960610 0.9066148  0.3265565 1.169682 233
[9]  {sandwich.bags=Yes, soda=Yes}                   => {vegetables=Yes} 0.2897078 0.9011858  0.3214740 1.162677 228
[10] {sandwich.bags=Yes, spaghetti.sauce=Yes}        => {vegetables=Yes} 0.2820839 0.9135802  0.3087675 1.178668 222
[11] {individual.meals=Yes, sandwich.bags=Yes}       => {vegetables=Yes} 0.2808132 0.9020408  0.3113088 1.163781 221
[12] {laundry.detergent=Yes, sandwich.bags=Yes}      => {vegetables=Yes} 0.2884371 0.9043825  0.3189327 1.166802 227
[13] {poultry=Yes, sandwich.bags=Yes}                => {vegetables=Yes} 0.2986023 0.9073359  0.3290978 1.170612 235
[14] {sandwich.bags=Yes, yogurt=Yes}                 => {vegetables=Yes} 0.2909784 0.9051383  0.3214740 1.167777 229
[15] {cheese=Yes, eggs=Yes}                          => {vegetables=Yes} 0.2693774 0.9137931  0.2947903 1.178943 212
[16] {cheese=Yes, soap=Yes}                          => {vegetables=Yes} 0.2808132 0.9057377  0.3100381 1.168550 221
[17] {cheese=Yes, soda=Yes}                          => {vegetables=Yes} 0.2782719 0.9012346  0.3087675 1.162740 219
[18] {cheese=Yes, individual.meals=Yes}              => {vegetables=Yes} 0.2655654 0.9047619  0.2935197 1.167291 209
[19] {cheese=Yes, dinner.rolls=Yes}                  => {vegetables=Yes} 0.2731893 0.9033613  0.3024142 1.165484 215
[20] {cheese=Yes, laundry.detergent=Yes}             => {vegetables=Yes} 0.2744600 0.9037657  0.3036849 1.166006 216
[21] {cheese=Yes, waffles=Yes}                       => {vegetables=Yes} 0.2681067 0.9017094  0.2973316 1.163353 211
[22] {cheese=Yes, yogurt=Yes}                        => {vegetables=Yes} 0.2871665 0.9149798  0.3138501 1.180474 226
[23] {eggs=Yes, pasta=Yes}                           => {vegetables=Yes} 0.2744600 0.9037657  0.3036849 1.166006 216
[24] {aluminum.foil=Yes, pasta=Yes}                  => {vegetables=Yes} 0.2681067 0.9134199  0.2935197 1.178461 211
[25] {individual.meals=Yes, pasta=Yes}               => {vegetables=Yes} 0.2858958 0.9146341  0.3125794 1.180028 225
[26] {laundry.detergent=Yes, pasta=Yes}              => {vegetables=Yes} 0.2922490 0.9126984  0.3202033 1.177531 230
[27] {pasta=Yes, poultry=Yes}                        => {vegetables=Yes} 0.2947903 0.9062500  0.3252859 1.169211 232
[28] {pasta=Yes, yogurt=Yes}                         => {vegetables=Yes} 0.2986023 0.9108527  0.3278272 1.175149 235
[29] {aluminum.foil=Yes, flour=Yes}                  => {vegetables=Yes} 0.2871665 0.9003984  0.3189327 1.161662 226
[30] {aluminum.foil=Yes, eggs=Yes}                   => {vegetables=Yes} 0.2820839 0.9098361  0.3100381 1.173838 222
[31] {eggs=Yes, spaghetti.sauce=Yes}                 => {vegetables=Yes} 0.2922490 0.9019608  0.3240152 1.163677 230
```

Figure 27 (b)

```
[32] {coffee.tea=Yes, eggs=Yes}                  => {vegetables=Yes} 0.2833545 0.9065041  0.3125794 1.169539 223
[33] {eggs=Yes, waffles=Yes}                     => {vegetables=Yes} 0.2808132 0.9094650  0.3087675 1.173359 221
[34] {eggs=Yes, poultry=Yes}                     => {vegetables=Yes} 0.2884371 0.9153226  0.3151207 1.180916 227
[35] {eggs=Yes, yogurt=Yes}                      => {vegetables=Yes} 0.2998729 0.9182879  0.3265565 1.184742 236
[36] {aluminum.foil=Yes, sugar=Yes}              => {vegetables=Yes} 0.2693774 0.9021277  0.2986023 1.163893 212
[37] {aluminum.foil=Yes, cereals=Yes}            => {vegetables=Yes} 0.2858958 0.9000000  0.3176620 1.161148 225
[38] {aluminum.foil=Yes, spaghetti.sauce=Yes}    => {vegetables=Yes} 0.2935197 0.9058824  0.3240152 1.168737 231
[39] {aluminum.foil=Yes, individual.meals=Yes}   => {vegetables=Yes} 0.2858958 0.9109312  0.3138501 1.175251 225
[40] {aluminum.foil=Yes, shampoo=Yes}            => {vegetables=Yes} 0.2884371 0.9007937  0.3202033 1.162171 227
[41] {aluminum.foil=Yes, toilet.paper=Yes}       => {vegetables=Yes} 0.2947903 0.9098039  0.3240152 1.173796 232
[42] {aluminum.foil=Yes, laundry.detergent=Yes}  => {vegetables=Yes} 0.3011436 0.9080460  0.3316391 1.171528 237
[43] {aluminum.foil=Yes, poultry=Yes}            => {vegetables=Yes} 0.2986023 0.9073359  0.3290978 1.170612 235
[44] {aluminum.foil=Yes, bagels=Yes}             => {vegetables=Yes} 0.2986023 0.9038462  0.3303685 1.166110 235
[45] {aluminum.foil=Yes, juice=Yes}              => {vegetables=Yes} 0.3024142 0.9015152  0.3354511 1.163102 238
[46] {aluminum.foil=Yes, yogurt=Yes}             => {vegetables=Yes} 0.2922490 0.9055118  0.3227446 1.168259 230
[47] {cereals=Yes, laundry.detergent=Yes}        => {vegetables=Yes} 0.3074968 0.9063670  0.3392630 1.169362 242
[48] {cereals=Yes, waffles=Yes}                  => {vegetables=Yes} 0.2858958 0.9109312  0.3138501 1.175251 225
[49] {paper.towels=Yes, yogurt=Yes}              => {vegetables=Yes} 0.3151207 0.9018182  0.3494282 1.163493 248
[50] {individual.meals=Yes, soap=Yes}            => {vegetables=Yes} 0.3113088 0.9107807  0.3418043 1.175056 245
[51] {soap=Yes, waffles=Yes}                     => {vegetables=Yes} 0.2986023 0.9073359  0.3290978 1.170612 235
[52] {soap=Yes, yogurt=Yes}                      => {vegetables=Yes} 0.3214740 0.9133574  0.3519695 1.178381 253
[53] {individual.meals=Yes, soda=Yes}            => {vegetables=Yes} 0.3100381 0.9037037  0.3430750 1.165926 244
[54] {soda=Yes, yogurt=Yes}                      => {vegetables=Yes} 0.3214740 0.9035714  0.3557814 1.165755 253
[55] {individual.meals=Yes, spaghetti.sauce=Yes} => {vegetables=Yes} 0.3011436 0.9080460  0.3316391 1.171528 237
[56] {spaghetti.sauce=Yes, waffles=Yes}          => {vegetables=Yes} 0.3011436 0.9045802  0.3329098 1.167057 237
[57] {spaghetti.sauce=Yes, yogurt=Yes}           => {vegetables=Yes} 0.3252859 0.9014085  0.3608640 1.162965 256
[58] {individual.meals=Yes, poultry=Yes}         => {vegetables=Yes} 0.3087675 0.9033457  0.3418043 1.165464 243
[59] {shampoo=Yes, yogurt=Yes}                   => {vegetables=Yes} 0.3087675 0.9000000  0.3430750 1.161148 243
[60] {milk=Yes, yogurt=Yes}                      => {vegetables=Yes} 0.3214740 0.9166667  0.3506989 1.182650 253
[61] {dinner.rolls=Yes, waffles=Yes}             => {vegetables=Yes} 0.3011436 0.9045802  0.3329098 1.167057 237
[62] {laundry.detergent=Yes, yogurt=Yes}         => {vegetables=Yes} 0.3303685 0.9027778  0.3659466 1.164731 260
> |
```

Figure 27 (c)

```
> summary(marketbasket)
        Date     all.purpose aluminum.foil bagels     beef       butter    cereals   cheese    coffee.tea dinner.rolls
 01/01/2000:  1   No :329     No :384          :  2      :  3   No   :410      :  1   No :403   No :361       :  1
 01/01/2001:  1   Yes:458     Yes:403       No :338   No :368   Yes  :376   No :369   Yes:384   Yes:426    No :356
 01/01/2002:  1                             Yes:447   Yes:416   YesNo:  1   Yes:417                        Yes:430
 01/02/2000:  1
 01/02/2001:  1
 01/02/2002:  1
 (other)   :781
 dishwashing.liquid.detergent  eggs     flour     fruits    hand.soap ice.cream individual.meals juice     ketchup   laundry.detergent
 No :359                       No :392   No :393      :  1   No :400   No :393   No :360          No :338      :  1   No :353
 Yes:428                       Yes:395   Yes:394   No :405   Yes:387   Yes:394   Yes:427          Yes:449   No :397   Yes:434
                                         Yes:381                                                           Yes:389


 lunch.meat  milk     mixes     paper.towels pasta      pork     poultry   sandwich.bags sandwich.loaves shampoo     soap      soda
 No :312    No :357      :  1      :  1      No :398   No :372   No :343   No :405       No :373         No :359   No  :366      :  1
 Yes:475    Yes:430   No :386   No :366      Yes:389   Yes:415   Yes:444   Yes:382       Yes:414         Yes:428   soda:  1   No :363
                      Yes:400   Yes:420                                                                            Yes :420   Yes:423


 spaghetti.sauce sugar     toilet.paper tortillas vegetables waffles   yogurt
 No   :362       No :376   No :354      No :363   No :177    No :350   No :333
 Yes  :424       Yes:411   Yes:433      Yes:424   Yes:610    Yes:437   Yes:454
 YesNo:  1
```

Figure 27 (d)

- Plotting to find the most purchased item.

```
68  # 13 - Plotting to find the most purchased item
69
70  barplot(purchased, beside = TRUE, legend = rownames(purchased))
71  rules <- apriori(marketbasket,
72                   parameter = list(minlen=2, maxlen=3,conf = 0.9),
73                   appearance= list(rhs=c("vegetables=Yes"),default="lhs"))
74  summary(rules)
```

Figure 28

```
> barplot(purchased, beside = TRUE, legend = rownames(purchased))
> rules <- apriori(marketbasket,
+                  parameter = list(minlen=2, maxlen=3,conf = 0.9),
+                  appearance= list(rhs=c("vegetables=Yes"),default="lhs"))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.9    0.1    1 none FALSE            TRUE       5     0.1      2      3  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 78

set item appearances ...[1 item(s)] done [0.00s].
set transactions ...[875 item(s), 787 transaction(s)] done [0.01s].
sorting and recoding items ... [76 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.01s].
writing ... [62 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
Warning message:
In apriori(marketbasket, parameter = list(minlen = 2, maxlen = 3,  :
  Mining stopped (maxlen reached). Only patterns up to a length of 3 returned!
> summary(rules)
set of 62 rules

rule length distribution (lhs + rhs):sizes
 3
62

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      3       3       3       3       3       3

summary of quality measures:
    support         confidence        coverage          lift           count
 Min.   :0.2541   Min.   :0.9000   Min.   :0.2795   Min.   :1.161   Min.   :200.0
 1st Qu.:0.2789   1st Qu.:0.9029   1st Qu.:0.3088   1st Qu.:1.165   1st Qu.:219.5
 Median :0.2891   Median :0.9061   Median :0.3202   Median :1.169   Median :227.5
 Mean   :0.2901   Mean   :0.9073   Mean   :0.3198   Mean   :1.171   Mean   :228.3
 3rd Qu.:0.3008   3rd Qu.:0.9109   3rd Qu.:0.3313   3rd Qu.:1.175   3rd Qu.:236.8
 Max.   :0.3304   Max.   :0.9183   Max.   :0.3659   Max.   :1.185   Max.   :260.0

mining info:
         data ntransactions support confidence
 marketbasket           787     0.1        0.9
                                                                                                                                                call
 apriori(data = marketbasket, parameter = list(minlen = 2, maxlen = 3, conf = 0.9), appearance = list(rhs = c("vegetables=Yes"), default = "lhs"))
>
```

Figure 29

- Inspecting the rules.

```
76  # 14 - Inspect the rules
77
78  inspect(rules)
79
```

Figure 30

```
> inspect(rules)
     lhs                                                rhs              support   confidence coverage  lift     count
[1]  {cheese=Yes, sandwich.bags=Yes}                 => {vegetables=Yes} 0.2681067 0.9134199  0.2935197 1.178461 211
[2]  {ketchup=Yes, sandwich.bags=Yes}                => {vegetables=Yes} 0.2566709 0.9181818  0.2795426 1.184605 202
[3]  {pasta=Yes, sandwich.bags=Yes}                  => {vegetables=Yes} 0.2706480 0.9141631  0.2960610 1.179420 213
[4]  {flour=Yes, sandwich.bags=Yes}                  => {vegetables=Yes} 0.2541296 0.9009009  0.2820839 1.162310 200
[5]  {eggs=Yes, sandwich.bags=Yes}                   => {vegetables=Yes} 0.2681067 0.9173913  0.2922490 1.183585 211
[6]  {mixes=Yes, sandwich.bags=Yes}                  => {vegetables=Yes} 0.2681067 0.9017094  0.2973316 1.163353 211
[7]  {aluminum.foil=Yes, sandwich.bags=Yes}          => {vegetables=Yes} 0.2706480 0.9181034  0.2947903 1.184504 213
[8]  {sandwich.bags=Yes, soap=Yes}                   => {vegetables=Yes} 0.2960610 0.9066148  0.3265565 1.169682 233
[9]  {sandwich.bags=Yes, soda=Yes}                   => {vegetables=Yes} 0.2897078 0.9011858  0.3214740 1.162677 228
[10] {sandwich.bags=Yes, spaghetti.sauce=Yes}        => {vegetables=Yes} 0.2820839 0.9135802  0.3087675 1.178668 222
[11] {individual.meals=Yes, sandwich.bags=Yes}       => {vegetables=Yes} 0.2808132 0.9020408  0.3113088 1.163781 221
[12] {laundry.detergent=Yes, sandwich.bags=Yes}      => {vegetables=Yes} 0.2884371 0.9043825  0.3189327 1.166802 227
[13] {poultry=Yes, sandwich.bags=Yes}                => {vegetables=Yes} 0.2986023 0.9073359  0.3290978 1.170612 235
[14] {sandwich.bags=Yes, yogurt=Yes}                 => {vegetables=Yes} 0.2909784 0.9051383  0.3214740 1.167777 229
[15] {cheese=Yes, eggs=Yes}                          => {vegetables=Yes} 0.2693774 0.9137931  0.2947903 1.178943 212
[16] {cheese=Yes, soap=Yes}                          => {vegetables=Yes} 0.2808132 0.9057377  0.3100381 1.168550 221
[17] {cheese=Yes, soda=Yes}                          => {vegetables=Yes} 0.2782719 0.9012346  0.3087675 1.162740 219
[18] {cheese=Yes, individual.meals=Yes}              => {vegetables=Yes} 0.2655654 0.9047619  0.2935197 1.167291 209
[19] {cheese=Yes, dinner.rolls=Yes}                  => {vegetables=Yes} 0.2731893 0.9033613  0.3024142 1.165484 215
[20] {cheese=Yes, laundry.detergent=Yes}             => {vegetables=Yes} 0.2744600 0.9037657  0.3036849 1.166006 216
[21] {cheese=Yes, waffles=Yes}                       => {vegetables=Yes} 0.2681067 0.9017094  0.2973316 1.163353 211
[22] {cheese=Yes, yogurt=Yes}                        => {vegetables=Yes} 0.2871665 0.9149798  0.3138501 1.180474 226
[23] {eggs=Yes, pasta=Yes}                           => {vegetables=Yes} 0.2744600 0.9037657  0.3036849 1.166006 216
[24] {aluminum.foil=Yes, pasta=Yes}                  => {vegetables=Yes} 0.2681067 0.9134199  0.2935197 1.178461 211
[25] {individual.meals=Yes, pasta=Yes}               => {vegetables=Yes} 0.2858958 0.9146341  0.3125794 1.180028 225
[26] {laundry.detergent=Yes, pasta=Yes}              => {vegetables=Yes} 0.2922490 0.9126984  0.3202033 1.177531 230
[27] {pasta=Yes, poultry=Yes}                        => {vegetables=Yes} 0.2947903 0.9062500  0.3252859 1.169211 232
[28] {pasta=Yes, yogurt=Yes}                         => {vegetables=Yes} 0.2986023 0.9108527  0.3278272 1.175149 235
[29] {aluminum.foil=Yes, flour=Yes}                  => {vegetables=Yes} 0.2871665 0.9003984  0.3189327 1.161662 226
[30] {aluminum.foil=Yes, eggs=Yes}                   => {vegetables=Yes} 0.2820839 0.9098361  0.3100381 1.173838 222
[31] {eggs=Yes, spaghetti.sauce=Yes}                 => {vegetables=Yes} 0.2922490 0.9019608  0.3240152 1.163677 230
```

Figure 31 (a)

```
[32] {coffee.tea=Yes, eggs=Yes}                  => {vegetables=Yes} 0.2833545 0.9065041  0.3125794 1.169539 223
[33] {eggs=Yes, waffles=Yes}                     => {vegetables=Yes} 0.2808132 0.9094650  0.3087675 1.173359 221
[34] {eggs=Yes, poultry=Yes}                     => {vegetables=Yes} 0.2884371 0.9153226  0.3151207 1.180916 227
[35] {eggs=Yes, yogurt=Yes}                      => {vegetables=Yes} 0.2998729 0.9182879  0.3265565 1.184742 236
[36] {aluminum.foil=Yes, sugar=Yes}              => {vegetables=Yes} 0.2693774 0.9021277  0.2986023 1.163893 212
[37] {aluminum.foil=Yes, cereals=Yes}            => {vegetables=Yes} 0.2858958 0.9000000  0.3176620 1.161148 225
[38] {aluminum.foil=Yes, spaghetti.sauce=Yes}    => {vegetables=Yes} 0.2935197 0.9058824  0.3240152 1.168737 231
[39] {aluminum.foil=Yes, individual.meals=Yes}   => {vegetables=Yes} 0.2858958 0.9109312  0.3138501 1.175251 225
[40] {aluminum.foil=Yes, shampoo=Yes}            => {vegetables=Yes} 0.2884371 0.9007937  0.3202033 1.162171 227
[41] {aluminum.foil=Yes, toilet.paper=Yes}       => {vegetables=Yes} 0.2947903 0.9098039  0.3240152 1.173796 232
[42] {aluminum.foil=Yes, laundry.detergent=Yes}  => {vegetables=Yes} 0.3011436 0.9080460  0.3316391 1.171528 237
[43] {aluminum.foil=Yes, poultry=Yes}            => {vegetables=Yes} 0.2986023 0.9073359  0.3290978 1.170612 235
[44] {aluminum.foil=Yes, bagels=Yes}             => {vegetables=Yes} 0.2986023 0.9038462  0.3303685 1.166110 235
[45] {aluminum.foil=Yes, juice=Yes}              => {vegetables=Yes} 0.3024142 0.9015152  0.3354511 1.163102 238
[46] {aluminum.foil=Yes, yogurt=Yes}             => {vegetables=Yes} 0.2922490 0.9055118  0.3227446 1.168259 230
[47] {cereals=Yes, laundry.detergent=Yes}        => {vegetables=Yes} 0.3074968 0.9063670  0.3392630 1.169362 242
[48] {cereals=Yes, waffles=Yes}                  => {vegetables=Yes} 0.2858958 0.9109312  0.3138501 1.175251 225
[49] {individual.meals=Yes, soap=Yes}            => {vegetables=Yes} 0.3113088 0.9107807  0.3418043 1.175056 245
[50] {soap=Yes, waffles=Yes}                     => {vegetables=Yes} 0.2986023 0.9073359  0.3290978 1.170612 235
[51] {soap=Yes, yogurt=Yes}                      => {vegetables=Yes} 0.3214740 0.9133574  0.3519695 1.178381 253
[52] {paper.towels=Yes, yogurt=Yes}              => {vegetables=Yes} 0.3151207 0.9018182  0.3494282 1.163493 248
[53] {individual.meals=Yes, soda=Yes}            => {vegetables=Yes} 0.3100381 0.9037037  0.3430750 1.165926 244
[54] {soda=Yes, yogurt=Yes}                      => {vegetables=Yes} 0.3214740 0.9035714  0.3557814 1.165755 253
[55] {individual.meals=Yes, spaghetti.sauce=Yes} => {vegetables=Yes} 0.3011436 0.9080460  0.3316391 1.171528 237
[56] {spaghetti.sauce=Yes, waffles=Yes}          => {vegetables=Yes} 0.3011436 0.9045802  0.3329098 1.167057 237
[57] {spaghetti.sauce=Yes, yogurt=Yes}           => {vegetables=Yes} 0.3252859 0.9014085  0.3608640 1.162965 256
[58] {individual.meals=Yes, poultry=Yes}         => {vegetables=Yes} 0.3087675 0.9033457  0.3418043 1.165464 243
[59] {shampoo=Yes, yogurt=Yes}                   => {vegetables=Yes} 0.3087675 0.9000000  0.3430750 1.161148 243
[60] {dinner.rolls=Yes, waffles=Yes}             => {vegetables=Yes} 0.3011436 0.9045802  0.3329098 1.167057 237
[61] {milk=Yes, yogurt=Yes}                      => {vegetables=Yes} 0.3214740 0.9166667  0.3506989 1.182650 253
[62] {laundry.detergent=Yes, yogurt=Yes}         => {vegetables=Yes} 0.3303685 0.9027778  0.3659466 1.164731 260
> |
```

Figure 31 (b)

- Installing the arulesViz package.

```
80  # 15 - To visualize these rules we are using the "arulesViz" package
81
82  install.packages("arulesviz")
83  library(arulesviz)
84
```

Figure 32

- Plotting the rules.

```
85  # 16 - Plot the rules
86
87  plot(rules)
```

Figure 33

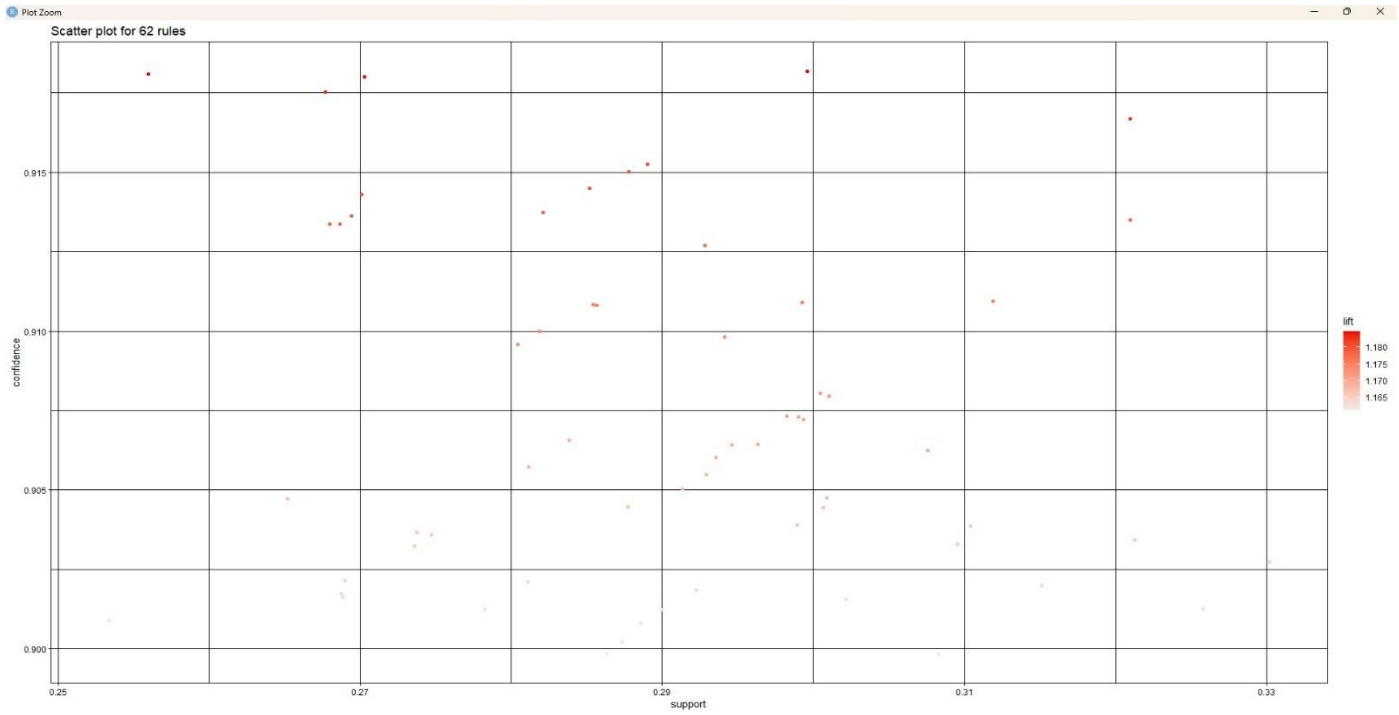

Figure 08

- Plotting the rules in groups.

```
89  # 17 - Plotting the rules in groups
90
91  plot(rules,method = "grouped")
92
```

Figure 34

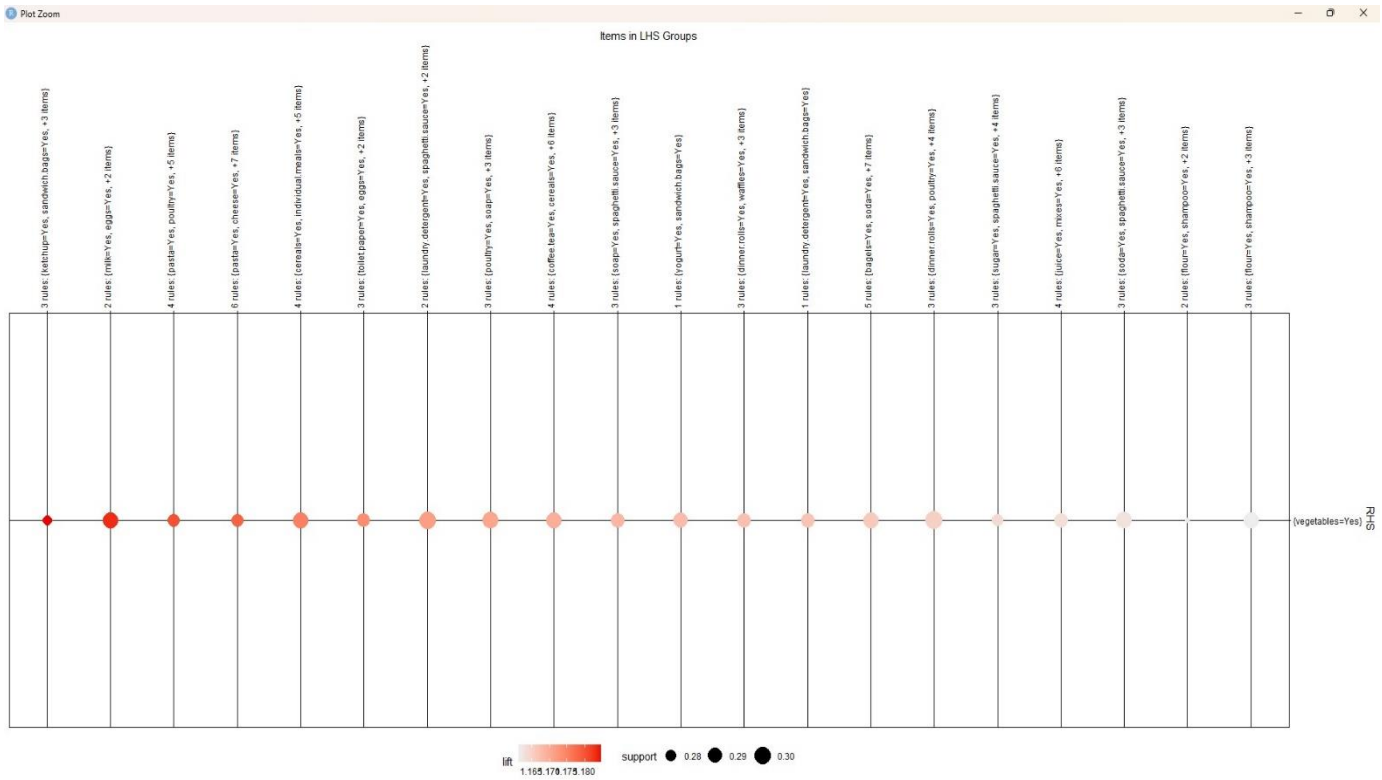

Figure 09

- Displaying the support, confidence and the lift using a scatterplot matrix.

```
93  # 18 - Displays the support, confidence, and lift with a scatterplot matrix
94
95  plot(rules@quality)
```

Figure 35

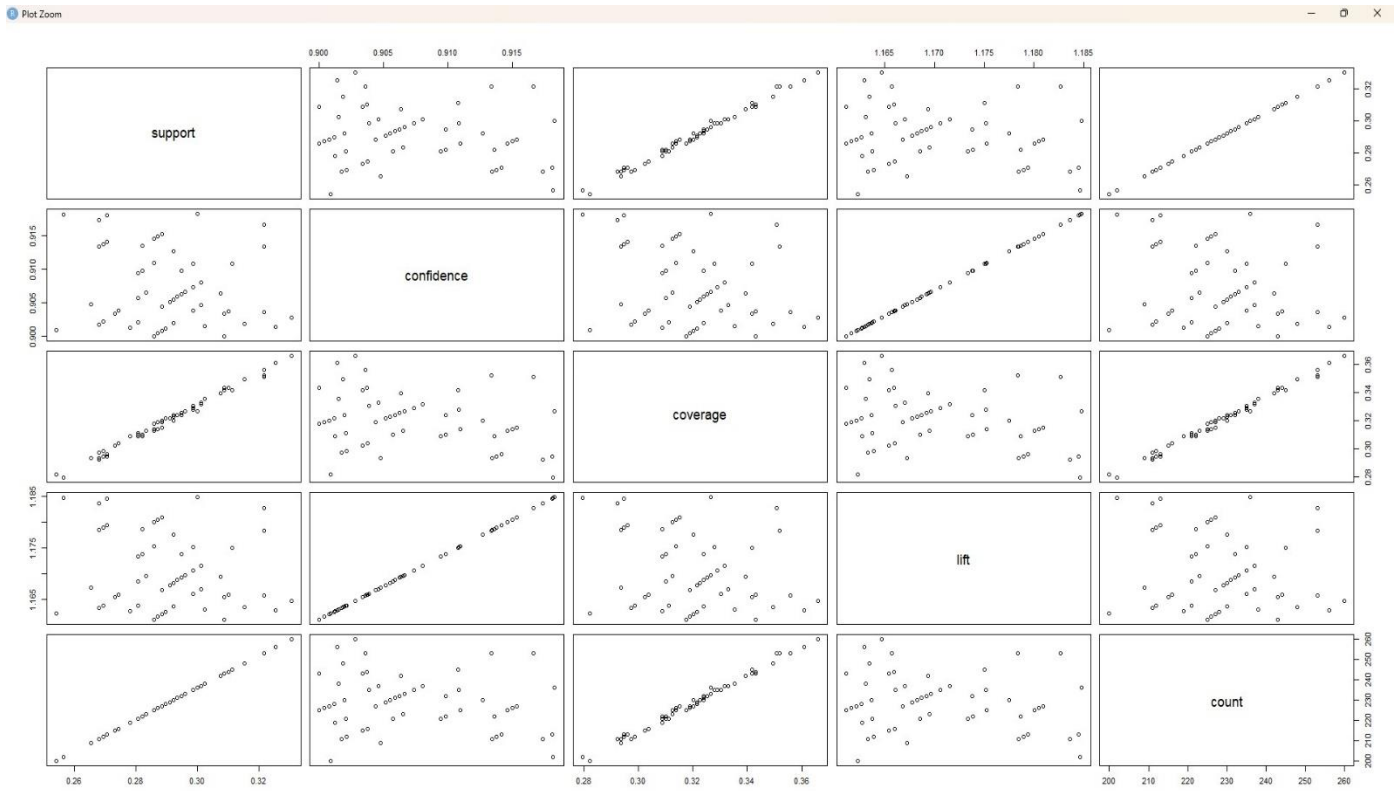

Figure 10

- Creating an interactive scatterplot for association rules.

```
# 19 - This is used to create interactive scatter plot for association rules

rules3 = apriori(marketbasket,
                 parameter = list(minlen=2,maxlen=3, conf = 0.8),
                 appearance =list(rhs=c("lunch.meat=Yes","all.purpose=Yes","juice=Yes")
                                  ,default="lhs") )
summary(rules3)

plot(rules3)

plot(rules3, measure = c("support", "lift"), shading = "confidence")
```

Figure 36

```
> rules3 = apriori(marketbasket,
+                  parameter = list(minlen=2,maxlen=3, conf = 0.8),
+                  appearance =list(rhs=c("lunch.meat=Yes","all.purpose=Yes","juice=Yes")
+                                   ,default="lhs") )
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.8    0.1    1 none FALSE            TRUE       5     0.1      2      3  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 78

set item appearances ...[3 item(s)] done [0.00s].
set transactions ...[875 item(s), 787 transaction(s)] done [0.01s].
sorting and recoding items ... [76 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.01s].
writing ... [35 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
Warning message:
In apriori(marketbasket, parameter = list(minlen = 2, maxlen = 3,  :
  Mining stopped (maxlen reached). Only patterns up to a length of 3 returned!
> summary(rules3)
set of 35 rules

rule length distribution (lhs + rhs):sizes
 3
35

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      3       3       3       3       3       3

summary of quality measures:
    support         confidence        coverage           lift           count
 Min.   :0.2198   Min.   :0.8000   Min.   :0.2719   Min.   :1.325   Min.   :173.0
 1st Qu.:0.2395   1st Qu.:0.8049   1st Qu.:0.2961   1st Qu.:1.334   1st Qu.:188.5
 Median :0.2503   Median :0.8080   Median :0.3113   Median :1.339   Median :197.0
 Mean   :0.2507   Mean   :0.8079   Mean   :0.3103   Mean   :1.347   Mean   :197.3
 3rd Qu.:0.2618   3rd Qu.:0.8102   3rd Qu.:0.3247   3rd Qu.:1.345   3rd Qu.:206.0
 Max.   :0.2821   Max.   :0.8242   Max.   :0.3482   Max.   :1.425   Max.   :222.0

mining info:
         data ntransactions support confidence
 marketbasket           787     0.1        0.8

call
 apriori(data = marketbasket, parameter = list(minlen = 2, maxlen = 3, conf = 0.8), appearance = list(rhs = c("lunch.meat=Yes", "all.purpose=Yes",
"juice=Yes"), default = "lhs"))
>
```

Figure 37

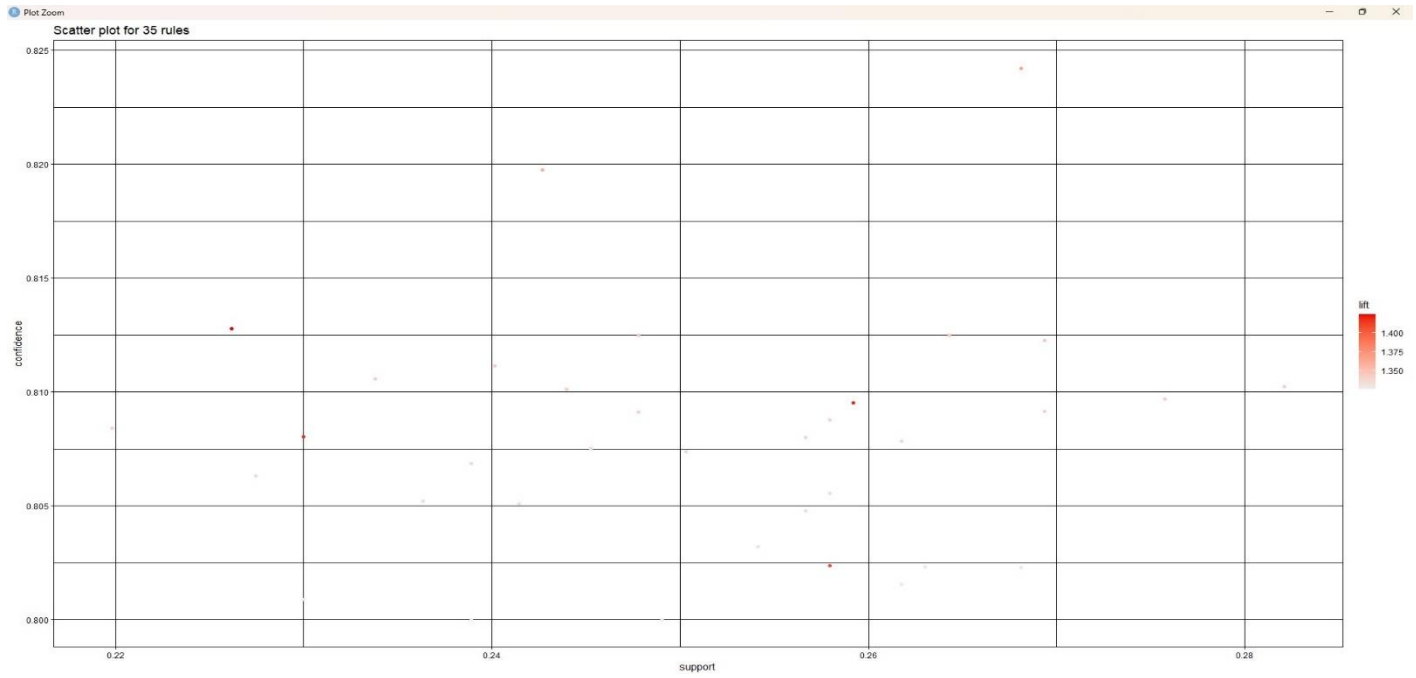

Figure 11 (a)

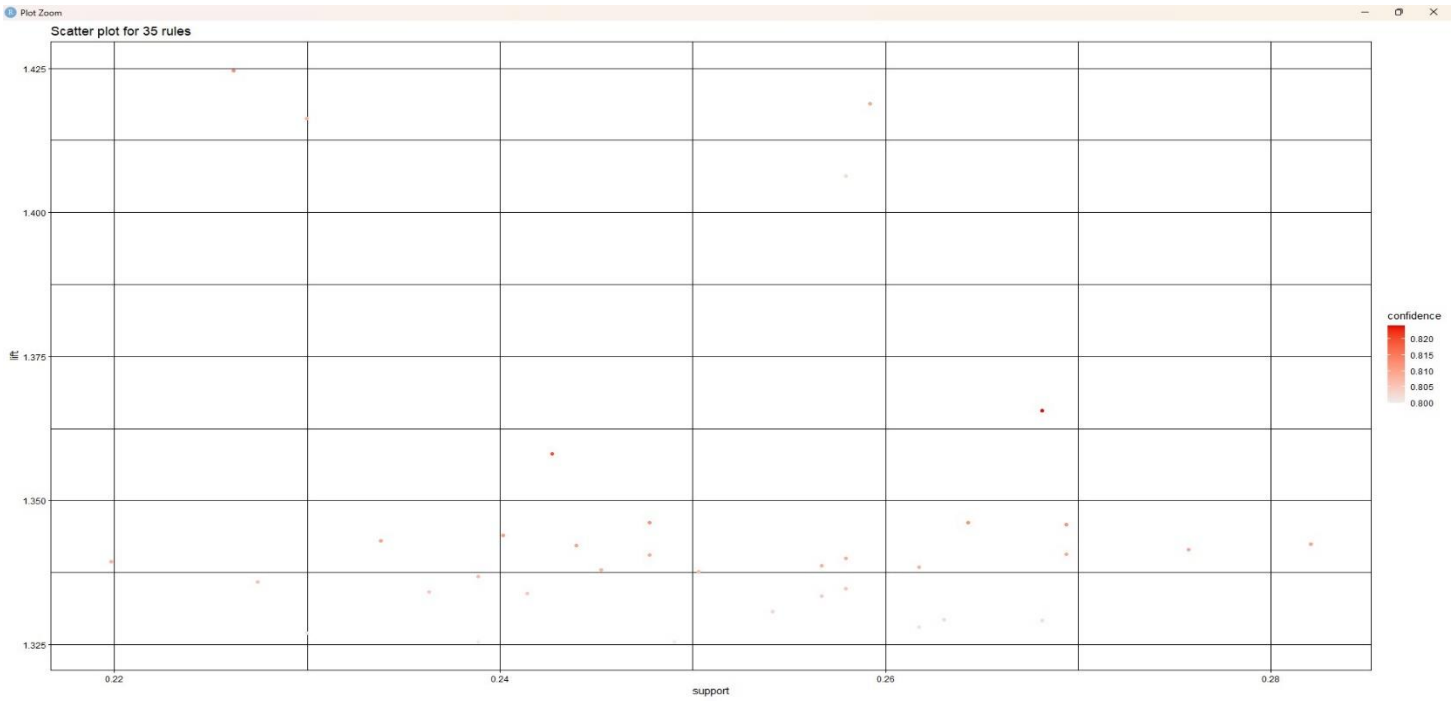

Figure 11 (b)

- Gets the rules only purchase items “Yes” on both the left hand and right hand sides while getting the summary and inspecting the rules.

```
# 20 - This code gets the rules only purchase items "YES" on left hand side and right hand side.

rules2 <- apriori(marketbasket,
                  parameter = list(minlen=2, maxlen=3,conf = 0.88),
                  appearance =list(rhs=c("vegetables=Yes"),
                                   lhs=c("all.purpose=Yes",
                                         "bagels=Yes",
                                         "juice=Yes",
                                         "lunch.meat=Yes",
                                         "poultry=Yes",
                                         "toilet.paper=Yes",
                                         "yogurt=Yes"),
                                   default="none"))
summary(rules2)

inspect(rules2)
```

Figure 38

```
> rules2 <- apriori(marketbasket,
+                   parameter = list(minlen=2, maxlen=3,conf = 0.88),
+                   appearance =list(rhs=c("vegetables=Yes"),
+                                    lhs=c("all.purpose=Yes",
+                                          "bagels=Yes",
+                                          "juice=Yes",
+                                          "lunch.meat=Yes",
+                                          "poultry=Yes",
+                                          "toilet.paper=Yes",
+                                          "yogurt=Yes"),
+                                    default="none"))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
       0.88    0.1    1 none FALSE            TRUE       5     0.1      2      3  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 78

set item appearances ...[8 item(s)] done [0.00s].
set transactions ...[8 item(s), 787 transaction(s)] done [0.00s].
sorting and recoding items ... [8 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [3 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
Warning message:
In apriori(marketbasket, parameter = list(minlen = 2, maxlen = 3,  :
  Mining stopped (maxlen reached). Only patterns up to a length of 3 returned!
```

Figure 39 (a)

```
> summary(rules2)
set of 3 rules

rule length distribution (lhs + rhs):sizes
3
3

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      3       3       3       3       3       3

summary of quality measures:
    support         confidence        coverage          lift           count
 Min.   :0.3024   Min.   :0.8815   Min.   :0.3431   Min.   :1.137   Min.   :238.0
 1st Qu.:0.3145   1st Qu.:0.8854   1st Qu.:0.3551   1st Qu.:1.142   1st Qu.:247.5
 Median :0.3266   Median :0.8893   Median :0.3672   Median :1.147   Median :257.0
 Mean   :0.3198   Mean   :0.8881   Mean   :0.3600   Mean   :1.146   Mean   :251.7
 3rd Qu.:0.3285   3rd Qu.:0.8914   3rd Qu.:0.3685   3rd Qu.:1.150   3rd Qu.:258.5
 Max.   :0.3304   Max.   :0.8935   Max.   :0.3698   Max.   :1.153   Max.   :260.0

mining info:
        data ntransactions support confidence
 marketbasket          787     0.1       0.88

call
 apriori(data = marketbasket, parameter = list(minlen = 2, maxlen = 3, conf = 0.88), appearance = list(rhs = c("vegetables=Yes"), lhs = c("all.pur
pose=Yes", "bagels=Yes", "juice=Yes", "lunch.meat=Yes", "poultry=Yes", "toilet.paper=Yes", "yogurt=Yes"), default = "none"))
> inspect(rules2)
    lhs                                rhs                 support   confidence coverage  lift     count
[1] {poultry=Yes, toilet.paper=Yes} => {vegetables=Yes} 0.3024142 0.8814815  0.3430750 1.137256 238
[2] {poultry=Yes, yogurt=Yes}       => {vegetables=Yes} 0.3265565 0.8892734  0.3672173 1.147308 257
[3] {juice=Yes, yogurt=Yes}         => {vegetables=Yes} 0.3303685 0.8934708  0.3697586 1.152724 260
>
```

Figure 39 (b)

1.6.Results analysis and discussion

The association rule technique allows us to formulate products within baskets in binary form regardless of the quantity, where the relations are occasionally represented in a table of binary relationships. It is used to uncover hidden relationships and patterns between different items in large datasets.

Association rule mining algorithms typically employ a two-step process: generating candidate rules and then filtering them based on user-defined minimum support and confidence values where according to the dataset, originally, there were 29, 339, 832 rules for a confidence of 0.8 (according to ***Figure 20***), where the confidence was increased to 0.9 which resulted in a reduction in the number of rules to 62 rules (according to ***Figure 24***). To get the final scatterplot, which displays the relationship between the support, confidence and the lift of the dataset, we reduce the confidence to 0.8 where we get the number of rules as 35 (according to ***Figure 37***). Ultimately, increasing the confidence level to 0.88, we get the final number of rules of 3 which are;

1. {poultry = Yes, toilet paper = Yes} => {vegetables = Yes} with the confidence of 0.881
2. {poultry = Yes, yogurt = Yes} => {vegetables = Yes} with the confidence of 0.889
3. {juice = Yes, yogurt = Yes} => {vegetables = Yes} with the confidence of 0.893

Accordingly, we can identify that the first rule has a confidence of 0.881, the second rule has a confidence of 0.889, and the third rule has a confidence of 0.893.

Since, the third rule has the highest confidence of 0.893. Therefore, the third rule is considered to be the best.

1.7. Conclusion

This dataset is about the market basket analysis which consists of 788 data records of the customers purchasing order. The dataset is a multivariate dataset which consists of 788 rows and 39 columns.

The association rule mining technique is used in this dataset.

Accordingly, the number of rules is reduced with the increase and/ or decrease of the confidence level where ultimately, it got reduced to three rules from 29, 339, 832 rules.

Finally, declaring that the third rule is the best since its confidence value is the highest than the rest.

Through the recognition of these patterns, it allows the retailers to make informed decisions on regards to inventory management, concoct effective marketing and pricing strategies which improves customer satisfaction thereby, improving the name and value of the business, increasing the sales, profits and the business's revenue while enabling them to survive in the market.

Task 02

Forest Fire Prediction

2.1. Introduction

A forest fire or a wild fire is an uncontrolled fire that burns wildlands, vegetations, forests, grasslands, savannas and other types of eco systems which is not bound by a specific continent, environment or ecosystem. They can be borne due to natural occurrence such as due to lighting strike or due to human intervention such as due to human made spark – fire.

Forest fires are influenced by factors such as weather, vegetation type, geography, and human activity. Understanding the complex relationship between these variables is crucial for effective prediction and to propose and implement prevention methods in order to reduce their disastrous effects on ecosystems, human lives and infrastructure.

Logistic regression is a powerful statistical framework for predicting binary outcomes. In this study, we use logistic regression analysis to create a predictive model for determining the chance of forest fire occurrence using environmental variables and historical fire data.

This model allows us to evaluate the contribution of each predictor variable to the chance of a fire outbreak, providing important insights into the causes of forest fires which can be used by forest management authorities, emergency responders, and policymakers as an effective decision making on regards to resource allocation ultimately reducing the impact of forest fires.

2.2. Dataset

The forest fire prediction dataset is a multivariate dataset which consists of 245 rows and 15 columns where 14 columns contain feature variables which are considered as the independent variables while 1 column is a target variable which is considered as a dependant variable.

The 15 variables are as follows:

1. **Day:** represents the day the forest fire occurred. Feature variable. Data type – Integer.
2. **Region:** represents the region the forest fire occurred. Feature variable. Data type – Categorical.
3. **Month:** represents the month the forest fire occurred. (From June to September). Feature variable. Data type – Integer.
4. **Year:** represents the year the forest fire occurred (2012). Feature variable. Data type – Integer.
5. **Temperature:** represents the temperature in Celsius degrees: 24 to 42. Feature variable. Data type – Integer.
6. **RH:** represents the Relative Humidity in %: 21 to 90. Feature variable. Data type – Integer.
7. **Ws:** represents the Wind speed in km/h: 6 to 29. Feature variable. Data type – Integer.
8. **Rain:** represents the rain fall outside in mm/m^2: 0 to 16.8. Feature variable. Data type – Continuous.
9. **FFMC:** represents the Fine Fuel Moisture Code index from the FWI (Fire Weather Index) system: 28.6 to 92.5. Feature variable. Data type – Continuous.
10. **DMC:** represents the Duff Moisture Code index from the FWI (Fire Weather Index) system: 1.1 to 65.9. Feature variable. Data type – Continuous.
11. **DC:** represents the Drought Code index from the FWI (Fire Weather Index) system: 7 to 220.4. Feature variable. Data type – Continuous.
12. **ISI:** represents the Initial Spread Index from the FWI (Fire Weather Index) system:0 to 18.5. Feature variable. Data type – Continuous.
13. **BUI:** represents the Buildup Index from the FWI (Fire Weather Index) system:1.1 to 68. Feature variable. Data type – Continuous.
14. **FWI:** represents the Fire Weather Index system: 0 to 31.1. Feature variable. Data type – Continuous.
15. **Classes:** represents two classes namely: “Fire” as 1 and “Not Fire” as 0. Target variable. Data type –Integer.

The target variable, “Classes” has an output attribute which is class.

There are no missing values.

Only logistic regression is used in the dataset.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	day	region	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
2	1	Bejaia	6	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	0
3	2	Bejaia	6	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	0
4	3	Bejaia	6	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0
5	4	Bejaia	6	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	0
6	5	Bejaia	6	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	0
7	6	Bejaia	6	2012	31	67	14	0	82.6	5.8	22.2	3.1	7	2.5	1
8	7	Bejaia	6	2012	33	54	13	0	88.2	9.9	30.5	6.4	10.9	7.2	1
9	8	Bejaia	6	2012	30	73	15	0	86.6	12.1	38.3	5.6	13.5	7.1	1
10	9	Bejaia	6	2012	25	88	13	0.2	52.9	7.9	38.8	0.4	10.5	0.3	0
11	10	Bejaia	6	2012	28	79	12	0	73.2	9.5	46.3	1.3	12.6	0.9	0
12	11	Bejaia	6	2012	31	65	14	0	84.5	12.5	54.3	4	15.8	5.6	1
13	12	Bejaia	6	2012	26	81	19	0	84	13.8	61.4	4.8	17.7	7.1	1
14	13	Bejaia	6	2012	27	84	21	1.2	50	6.7	17	0.5	6.7	0.2	0
15	14	Bejaia	6	2012	30	78	20	0.5	59	4.6	7.8	1	4.4	0.4	0
16	15	Bejaia	6	2012	28	80	17	3.1	49.4	3	7.4	0.4	3	0.1	0
17	16	Bejaia	6	2012	29	89	13	0.7	36.1	1.7	7.6	0	2.2	0	0
18	17	Bejaia	6	2012	30	89	16	0.6	37.3	1.1	7.8	0	1.6	0	0
19	18	Bejaia	6	2012	31	78	14	0.3	56.9	1.9	8	0.7	2.4	0.2	0
20	19	Bejaia	6	2012	31	55	16	0.1	79.9	4.5	16	2.5	5.3	1.4	0
21	20	Bejaia	6	2012	30	80	16	0.4	59.8	3.4	27.1	0.9	5.1	0.4	0
22	21	Bejaia	6	2012	30	78	14	0	81	6.3	31.6	2.6	8.4	2.2	1
23	22	Bejaia	6	2012	31	67	17	0.1	79.1	7	39.5	2.4	9.7	2.3	0
24	23	Bejaia	6	2012	32	62	18	0.1	81.4	8.2	47.7	3.3	11.5	3.8	1
25	24	Bejaia	6	2012	32	66	17	0	85.9	11.2	55.8	5.6	14.9	7.5	1
26	25	Bejaia	6	2012	31	64	15	0	86.7	14.2	63.8	5.7	18.3	8.4	1
27	26	Bejaia	6	2012	31	64	18	0	86.8	17.8	71.8	6.7	21.6	10.6	1
28	27	Bejaia	6	2012	34	53	18	0	89	21.6	80.3	9.2	25.8	15	1
29	28	Bejaia	6	2012	32	55	14	0	89.1	25.5	88.5	7.6	29.7	13.9	1
30	29	Bejaia	6	2012	32	47	13	0.3	79.9	18.4	84.4	2.2	23.8	3.9	0

Figure 40

https://archive.ics.uci.edu/dataset/547/algerian+forest+fires+dataset

The purpose of this dataset is to predict the chance of a fire outbreak; whether a forest fire occurs or not. This dataset about the forest fire which was occurred during 2012 in the Northern Algeria from the month of June till September.

The dataset consists of dependant variable which is "classes" where class 0 is "not fire" while class 1 is "fire". and the independent variables which are "Date", "Temperature", "Relative Humidity", "Wind speed", "Rain", "Fine Fuel Moisture Code", "Duff Moisture Code", "Drought Code", "Initial Spread Index", "Buildup Index" and "Fire Weather Index".

For the analysis of this dataset, the independent variables are utilized to predict the dependent variable; ultimately predicting whether a forest fire outbreak would occur or not. Which can be used to make effective decisions regarding resource allocation ultimately reducing the risk of forest fire outbreaks.

2.3. Explanation and preparation of dataset

This dataset contains information about the Algerian Forest Fires.

The dataset includes 244 instances that regroup the data of two regions of Algeria, the Béjaïa region located in the northeast of Algeria and the Sidi Bel-Abbes region located in the northwest of Algeria. Each region consists of 122 instances.

The period in which the dataset is collected is from June to September in the year 2012.

The dataset includes 14 independent variables and 1 dependent variable which is "Class". The independent variables are; "Temperature", "RH", "Ws" and "DMC" are some of them.

Logistics regression is used to predict the chance of a fire outbreak: whether a forest fire occurs or not.

Even though, the dataset is a clean dataset where no missing values were found. Preparatory tasks were carried out on the dataset.

Data preprocessing technique was utilized to further clarify the dataset for the ease of the data analysis process.

Preprocessing of the dataset:

1. Download the dataset into Excel as a csv file.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Bejaia Region Dataset													
2	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
3	1	6	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire
4	2	6	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire
5	3	6	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire
6	4	6	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire
7	5	6	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire
8	6	6	2012	31	67	14	0	82.6	5.8	22.2	3.1	7	2.5	fire
9	7	6	2012	33	54	13	0	88.2	9.9	30.5	6.4	10.9	7.2	fire
10	8	6	2012	30	73	15	0	86.6	12.1	38.3	5.6	13.5	7.1	fire
11	9	6	2012	25	88	13	0.2	52.9	7.9	38.8	0.4	10.5	0.3	not fire
12	10	6	2012	28	79	12	0	73.2	9.5	46.3	1.3	12.6	0.9	not fire
13	11	6	2012	31	65	14	0	84.5	12.5	54.3	4	15.8	5.6	fire
14	12	6	2012	26	81	19	0	84	13.8	61.4	4.8	17.7	7.1	fire
15	13	6	2012	27	84	21	1.2	50	6.7	17	0.5	6.7	0.2	not fire
16	14	6	2012	30	78	20	0.5	59	4.6	7.8	1	4.4	0.4	not fire
17	15	6	2012	28	80	17	3.1	49.4	3	7.4	0.4	3	0.1	not fire
18	16	6	2012	29	89	13	0.7	36.1	1.7	7.6	0	2.2	0	not fire
19	17	6	2012	30	89	16	0.6	37.3	1.1	7.8	0	1.6	0	not fire
20	18	6	2012	31	78	14	0.3	56.9	1.9	8	0.7	2.4	0.2	not fire
21	19	6	2012	31	55	16	0.1	79.9	4.5	16	2.5	5.3	1.4	not fire
22	20	6	2012	30	80	16	0.4	59.8	3.4	27.1	0.9	5.1	0.4	not fire
23	21	6	2012	30	78	14	0	81	6.3	31.6	2.6	8.4	2.2	fire
24	22	6	2012	31	67	17	0.1	79.1	7	39.5	2.4	9.7	2.3	not fire
25	23	6	2012	32	62	18	0.1	81.4	8.2	47.7	3.3	11.5	3.8	fire
26	24	6	2012	32	66	17	0	85.9	11.2	55.8	5.6	14.9	7.5	fire
27	25	6	2012	31	64	15	0	86.7	14.2	63.8	5.7	18.3	8.4	fire
28	26	6	2012	31	64	18	0	86.8	17.8	71.8	6.7	21.6	10.6	fire

Figure 41

2. Adding a separate column called “Region” which will represent the regions where the forest fire occurred. Change the data type of the Classes column from categorical to integer where is 1 = “Fire” and 0 = “Not Fire”.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	day	region	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
2	1	Bejaia	6	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	0
3	2	Bejaia	6	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	0
4	3	Bejaia	6	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0
5	4	Bejaia	6	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	0
6	5	Bejaia	6	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	0
7	6	Bejaia	6	2012	31	67	14	0	82.6	5.8	22.2	3.1	7	2.5	1
8	7	Bejaia	6	2012	33	54	13	0	88.2	9.9	30.5	6.4	10.9	7.2	1
9	8	Bejaia	6	2012	30	73	15	0	86.6	12.1	38.3	5.6	13.5	7.1	1
10	9	Bejaia	6	2012	25	88	13	0.2	52.9	7.9	38.8	0.4	10.5	0.3	0
11	10	Bejaia	6	2012	28	79	12	0	73.2	9.5	46.3	1.3	12.6	0.9	0
12	11	Bejaia	6	2012	31	65	14	0	84.5	12.5	54.3	4	15.8	5.6	1
13	12	Bejaia	6	2012	26	81	19	0	84	13.8	61.4	4.8	17.7	7.1	1
14	13	Bejaia	6	2012	27	84	21	1.2	50	6.7	17	0.5	6.7	0.2	0
15	14	Bejaia	6	2012	30	78	20	0.5	59	4.6	7.8	1	4.4	0.4	0
16	15	Bejaia	6	2012	28	80	17	3.1	49.4	3	7.4	0.4	3	0.1	0
17	16	Bejaia	6	2012	29	89	13	0.7	36.1	1.7	7.6	0	2.2	0	0
18	17	Bejaia	6	2012	30	89	16	0.6	37.3	1.1	7.8	0	1.6	0	0
19	18	Bejaia	6	2012	31	78	14	0.3	56.9	1.9	8	0.7	2.4	0.2	0
20	19	Bejaia	6	2012	31	55	16	0.1	79.9	4.5	16	2.5	5.3	1.4	0
21	20	Bejaia	6	2012	30	80	16	0.4	59.8	3.4	27.1	0.9	5.1	0.4	0
22	21	Bejaia	6	2012	30	78	14	0	81	6.3	31.6	2.6	8.4	2.2	1
23	22	Bejaia	6	2012	31	67	17	0.1	79.1	7	39.5	2.4	9.7	2.3	0
24	23	Bejaia	6	2012	32	62	18	0.1	81.4	8.2	47.7	3.3	11.5	3.8	1
25	24	Bejaia	6	2012	32	66	17	0	85.9	11.2	55.8	5.6	14.9	7.5	1
26	25	Bejaia	6	2012	31	64	15	0	86.7	14.2	63.8	5.7	18.3	8.4	1
27	26	Bejaia	6	2012	31	64	18	0	86.8	17.8	71.8	6.7	21.6	10.6	1
28	27	Bejaia	6	2012	34	53	18	0	89	21.6	80.3	9.2	25.8	15	1
29	28	Bejaia	6	2012	32	55	14	0	89.1	25.5	88.5	7.6	29.7	13.9	1
30	29	Bejaia	6	2012	32	47	13	0.3	79.9	18.4	84.4	2.2	23.8	3.9	0

Figure 40

2.4. Data mining

The data mining technique utilized for the dataset is the logistic regression using R programming language.

Logistic regression is the technique used for the regression analysis of the dataset.

Using the logistic regression model, we split the dataset where 80% is used to train the dataset and 20% of the dataset is used to test the dataset using a random sampling technique where the model will be able to understand, evaluate and predict the dataset.

Visualization tools used:

1. corrplot is used to create Correlogram between the variables of the dataset.

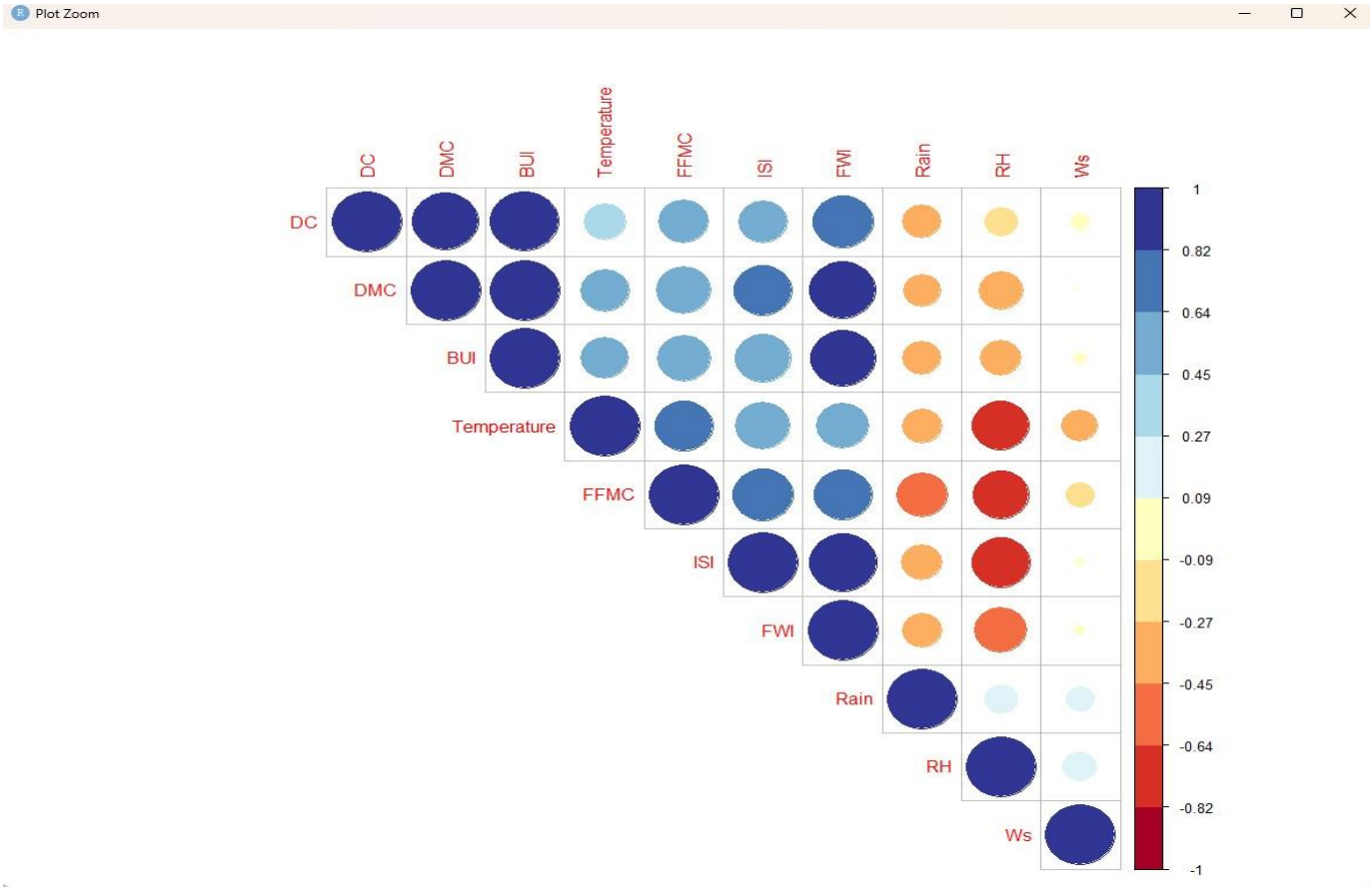

Figure 42

2. ggpairs is used to create pairwise plots between the variables.

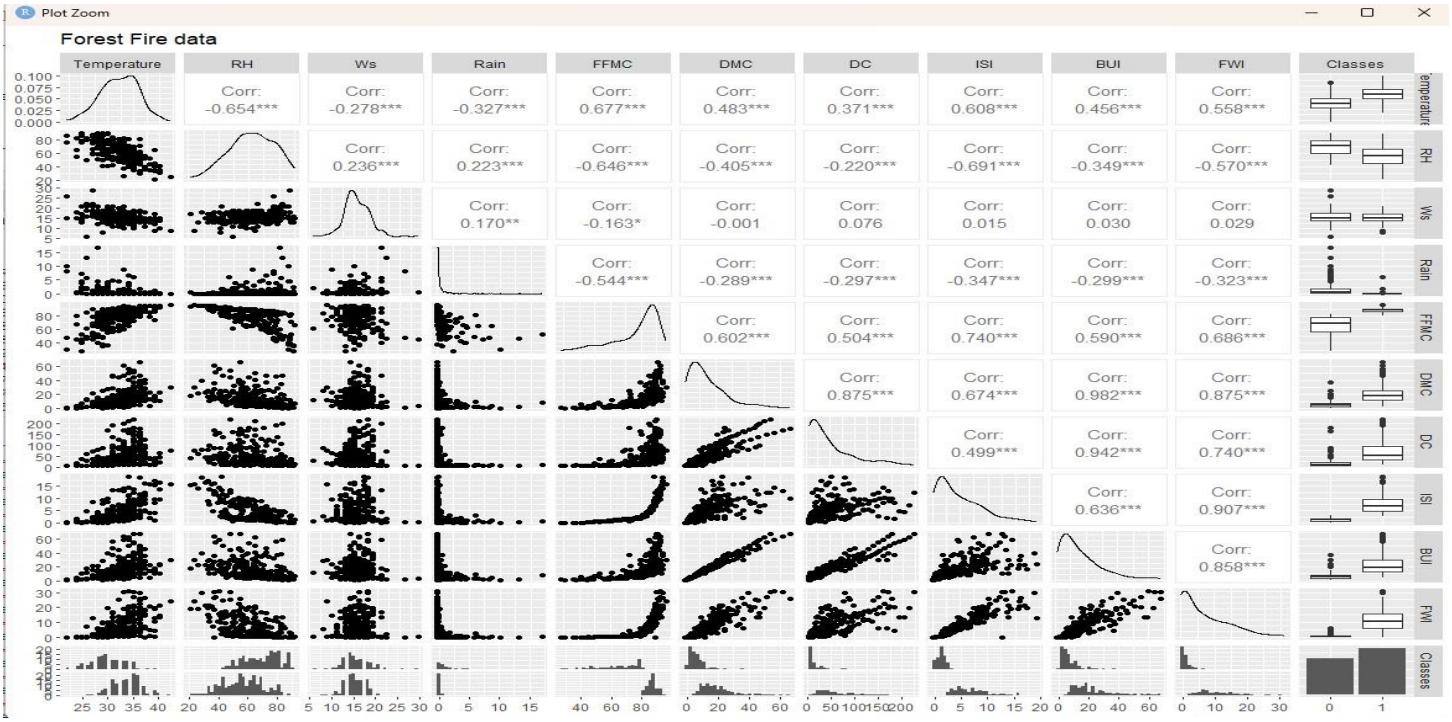

Figure 43

3. ggpairs is used to create a map to visualize the relationship between variables.

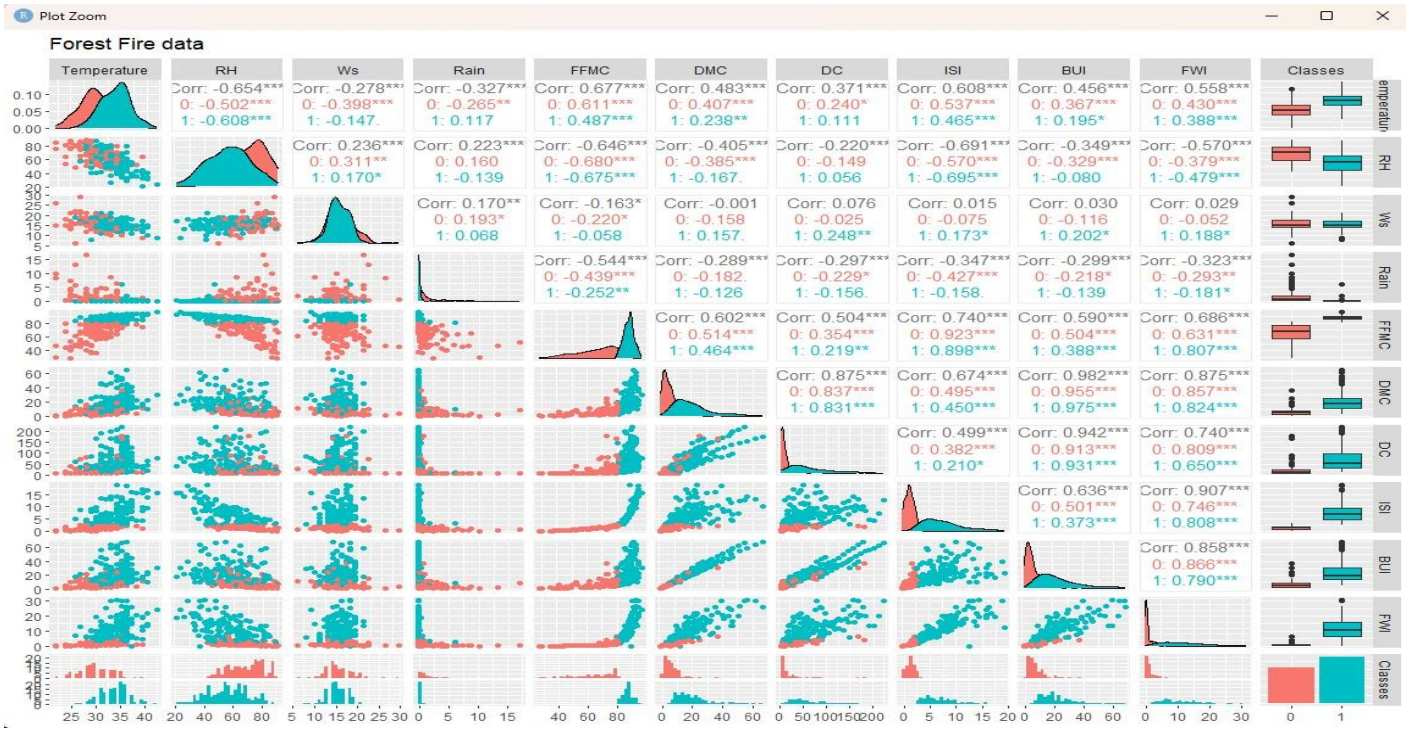

Figure 44

4. ggscatmat is used to create visuals from numerical variables.

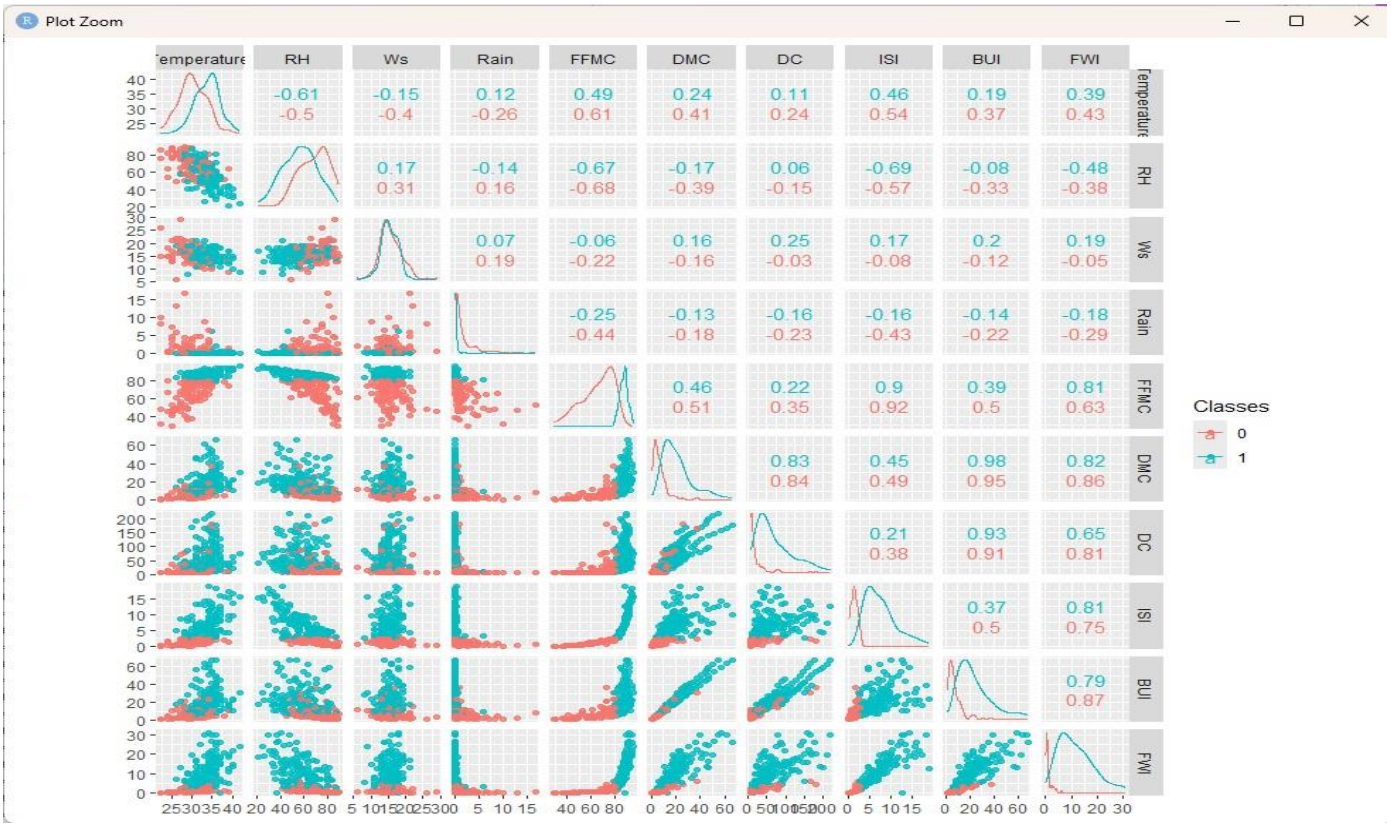

Figure 45

5. Building a linear regression model with “Temperature” and “Rain” using ggplot.

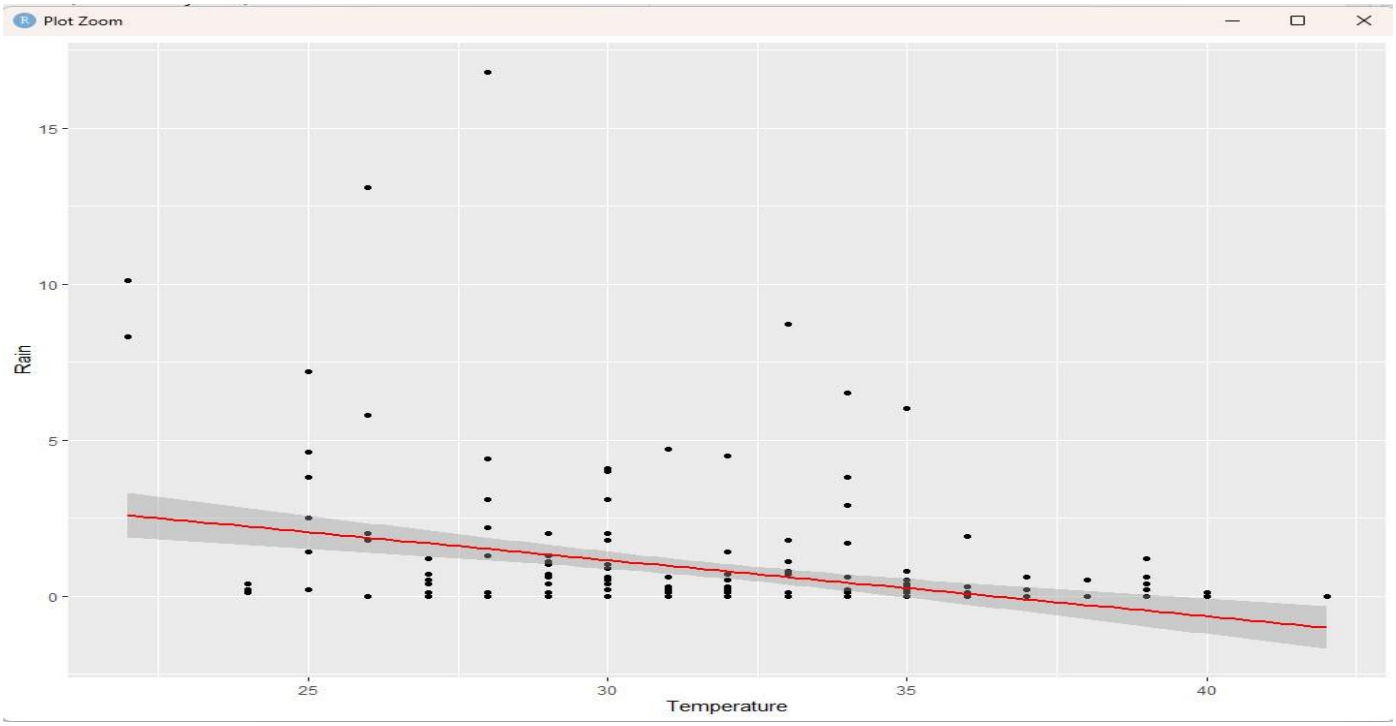

Figure 46

6. Plotting a linear regression model with “Temperature” and “Wind speed” using ggplot.

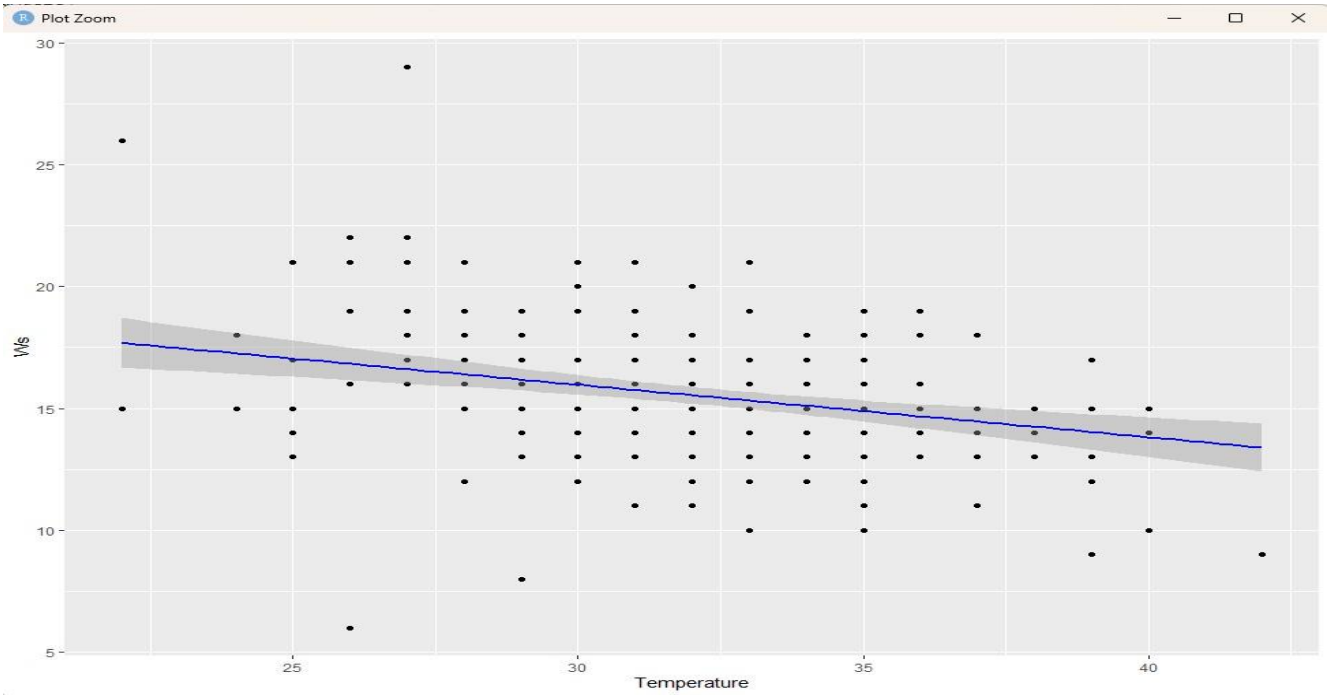

Figure 47

7. Plotting the linear regression model using ggplot.

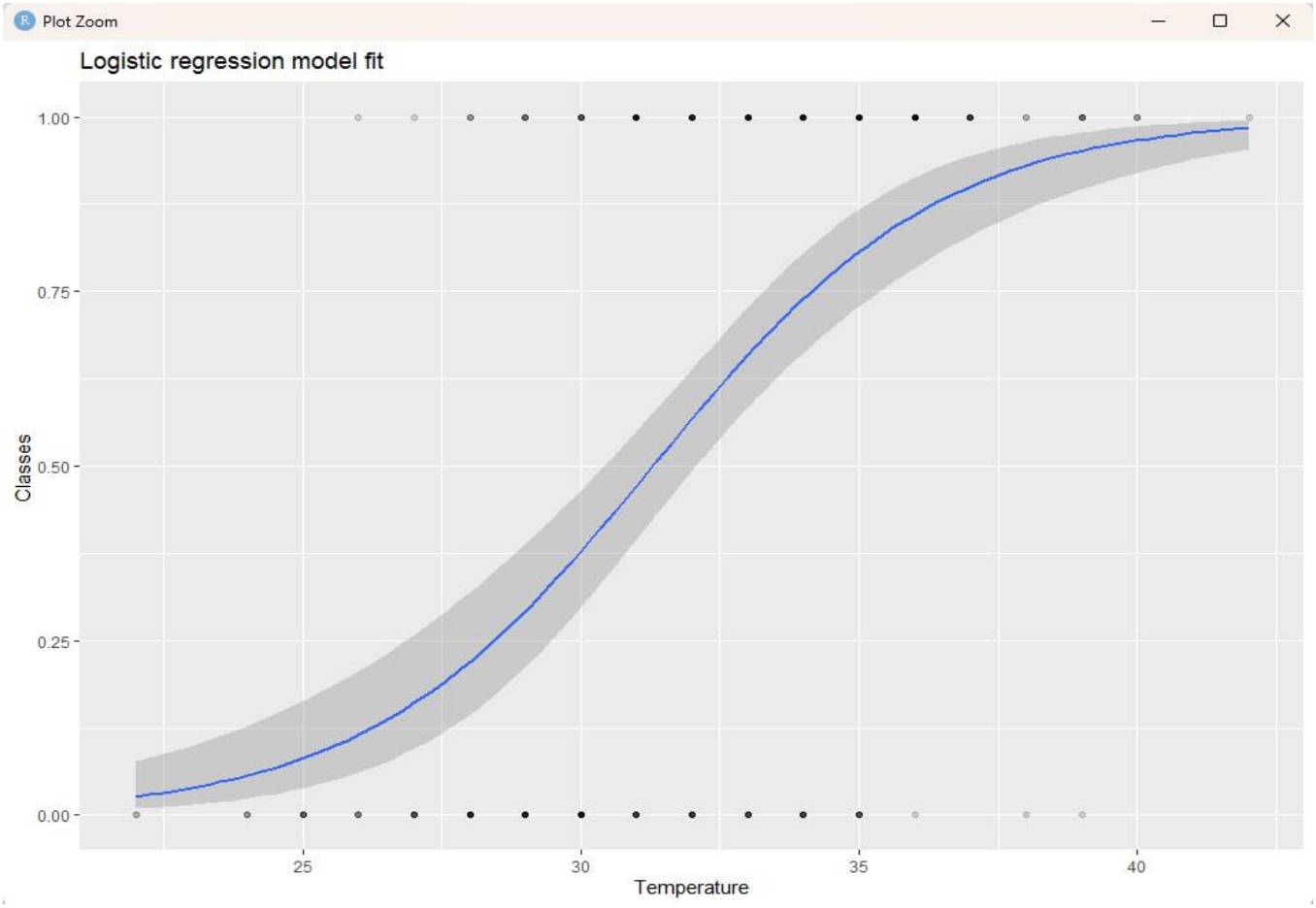

Figure 48

2.5. Implementation in R

- Install packages to R.
- The R packages installed are as follows:
  1. **party** – While it's not directly used for logistic regression, decision trees and random forests are often part of the modelling process alongside logistic regression.
  2. **epitools** – it is used for data preparation and exploration.
  3. **ggplot2** – it is essential for data visualization, including model diagnostics and results.
  4. **GGally** – it is useful for exploratory data analysis and visualization.
  5. **tidyverse** – it is a collection of packages which includes ggplot2, dplyr, tidyr and more; which are important for data manipulation, visualization, and modelling. It provides a consistent and efficient workflow.
  6. **corrplot** – it is used for visualizing correlation matrices which is done by understanding correlations between variables is crucial during model development.
  7. **RColorBrewer** – it is used to provide color palettes for creating visually appealing plots which is important for customizing visualizations.

- Calling out the required libraries for the classification of the dataset.

```
install.packages("party")
install.packages("epitools")
install.packages("ggplot2")
install.packages("GGally")
install.packages("tidyverse")
install.packages("corrplot")
install.packages("RColorBrewer")
library(party)
library(epitools)
library(ggplot2)
library(GGally)
library(tidyverse)
library(corrplot)
library(RColorBrewer)
```

Figure 49

- Import and view the dataset. For our convenience, checking the head (which shows the 1st 6 rows of the dataset), the str (string – which shows the structure of the dataset) and the dim (which shows the dimension, the number of rows and the columns of the dataset).

```
# 1 - Read the data

mydata<- read.csv("AA.csv")
mydata<-mydata[-1:-4]
mydata$Classes<-as.factor(mydata$Classes)
head(mydata)
str(mydata)
dim(mydata)
```

Figure 50

```
mydata<- read.csv("AA.csv")
```

Figure 51 (a)

```
mydata$Classes<-as.factor(mydata$Classes)
head(mydata)
Temperature RH Ws Rain FFMC DMC   DC ISI BUI FWI Classes
          29 57 18  0.0 65.7 3.4  7.6 1.3 3.4 0.5       0
          29 61 13  1.3 64.4 4.1  7.6 1.0 3.9 0.4       0
          26 82 22 13.1 47.1 2.5  7.1 0.3 2.7 0.1       0
          25 89 13  2.5 28.6 1.3  6.9 0.0 1.7 0.0       0
          27 77 16  0.0 64.8 3.0 14.2 1.2 3.9 0.5       0
          31 67 14  0.0 82.6 5.8 22.2 3.1 7.0 2.5       1
```

Figure 51 (b)

```
> str(mydata)
'data.frame':   244 obs. of  11 variables:
 $ Temperature: int  29 29 26 25 27 31 33 30 25 28 ...
 $ RH         : int  57 61 82 89 77 67 54 73 88 79 ...
 $ Ws         : int  18 13 22 13 16 14 13 15 13 12 ...
 $ Rain       : num  0 1.3 13.1 2.5 0 0 0 0 0.2 0 ...
 $ FFMC       : num  65.7 64.4 47.1 28.6 64.8 82.6 88.2 86.6 52.9 73.2 ...
 $ DMC        : num  3.4 4.1 2.5 1.3 3 5.8 9.9 12.1 7.9 9.5 ...
 $ DC         : num  7.6 7.6 7.1 6.9 14.2 22.2 30.5 38.3 38.8 46.3 ...
 $ ISI        : num  1.3 1 0.3 0 1.2 3.1 6.4 5.6 0.4 1.3 ...
 $ BUI        : num  3.4 3.9 2.7 1.7 3.9 7 10.9 13.5 10.5 12.6 ...
 $ FWI        : num  0.5 0.4 0.1 0 0.5 2.5 7.2 7.1 0.3 0.9 ...
 $ Classes    : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 2 1 1 ...
```

Figure 51 (c)

```
> dim(mydata)
[1] 244  11
>
```

Figure 51 (d)

- Getting the correlations with each variable.

```
25  # 2 - Correlations with each variables
26
27  data_cor<-cor(mydata[,-11])
28  data_cor
29  corrplot(data_cor, type="upper", order="hclust", col=brewer.pal(n=11,
30                                                                  name="RdYlBu"))
```

Figure 52

```
> data_cor<-cor(mydata[,-11])
> data_cor
            Temperature         RH          Ws       Rain       FFMC         DMC          DC         ISI         BUI         FWI
Temperature   1.0000000 -0.6544434 -0.27813207 -0.3267856  0.6774908  0.48310471  0.37051080  0.60755132  0.45550423  0.55839283
RH           -0.6544434  1.0000000  0.23608382  0.2229681 -0.6456580 -0.40513286 -0.22034450 -0.69063716 -0.34858739 -0.56999660
Ws           -0.2781321  0.2360838  1.00000000  0.1701695 -0.1632550 -0.00124599  0.07625270  0.01524818  0.02975592  0.02879895
Rain         -0.3267856  0.2229681  0.17016949  1.0000000 -0.5440445 -0.28854761 -0.29680760 -0.34710507 -0.29917128 -0.32268240
FFMC          0.6774908 -0.6456580 -0.16325496 -0.5440445  1.0000000  0.60239081  0.50391880  0.73973002  0.58965159  0.68603332
DMC           0.4831047 -0.4051329 -0.00124599 -0.2885476  0.6023908  1.00000000  0.87536210  0.67449874  0.98207327  0.87477822
DC            0.3705108 -0.2203445  0.07625270 -0.2968076  0.5039188  0.87536210  1.00000000  0.49892595  0.94190638  0.74018609
ISI           0.6075513 -0.6906372  0.01524818 -0.3471051  0.7397300  0.67449874  0.49892595  1.00000000  0.63589098  0.90746053
BUI           0.4555042 -0.3485874  0.02975592 -0.2991713  0.5896516  0.98207327  0.94190638  0.63589098  1.00000000  0.85777133
FWI           0.5583928 -0.5699966  0.02879895 -0.3226824  0.6860333  0.87477822  0.74018609  0.90746053  0.85777133  1.00000000
>
```

Figure 53 (a)

```
> corrplot(data_cor, type="upper", order="hclust", col=brewer.pal(n=11,
+                                                                 name="RdYlBu"))
```

Figure 53 (b)

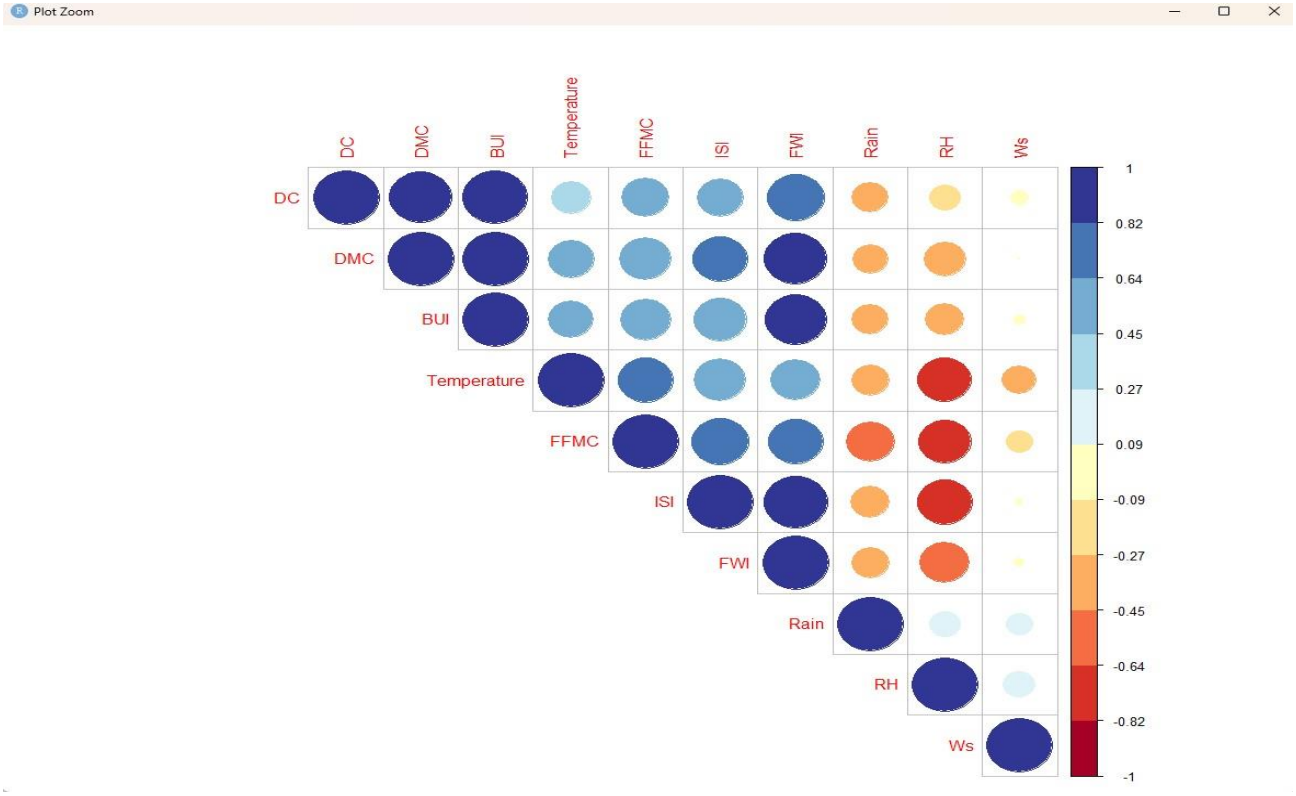

Figure 42

- 1st plot - Creating pairwise plots between the variables.

```
31  # 3 - 1st Plot
32
33  ggpairs(data=mydata, title="Forest Fire data")
34
```

Figure 54

```
+                                                                                                    name= RDYIBU ))
> ggpairs(data=mydata, title="Forest Fire data")
 plot: [11, 1] [=============================================================================================================>---------] 92% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 2] [=============================================================================================================>---------] 93% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 3] [=============================================================================================================>---------] 93% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 4] [=============================================================================================================>-------] 94% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 5] [=============================================================================================================>-----] 95% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 6] [=============================================================================================================>-----] 96% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 7] [=============================================================================================================>----] 97% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 8] [=============================================================================================================>---] 98% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 9] [=============================================================================================================>--] 98% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 10] [=============================================================================================================>-] 99% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
```

Figure 55

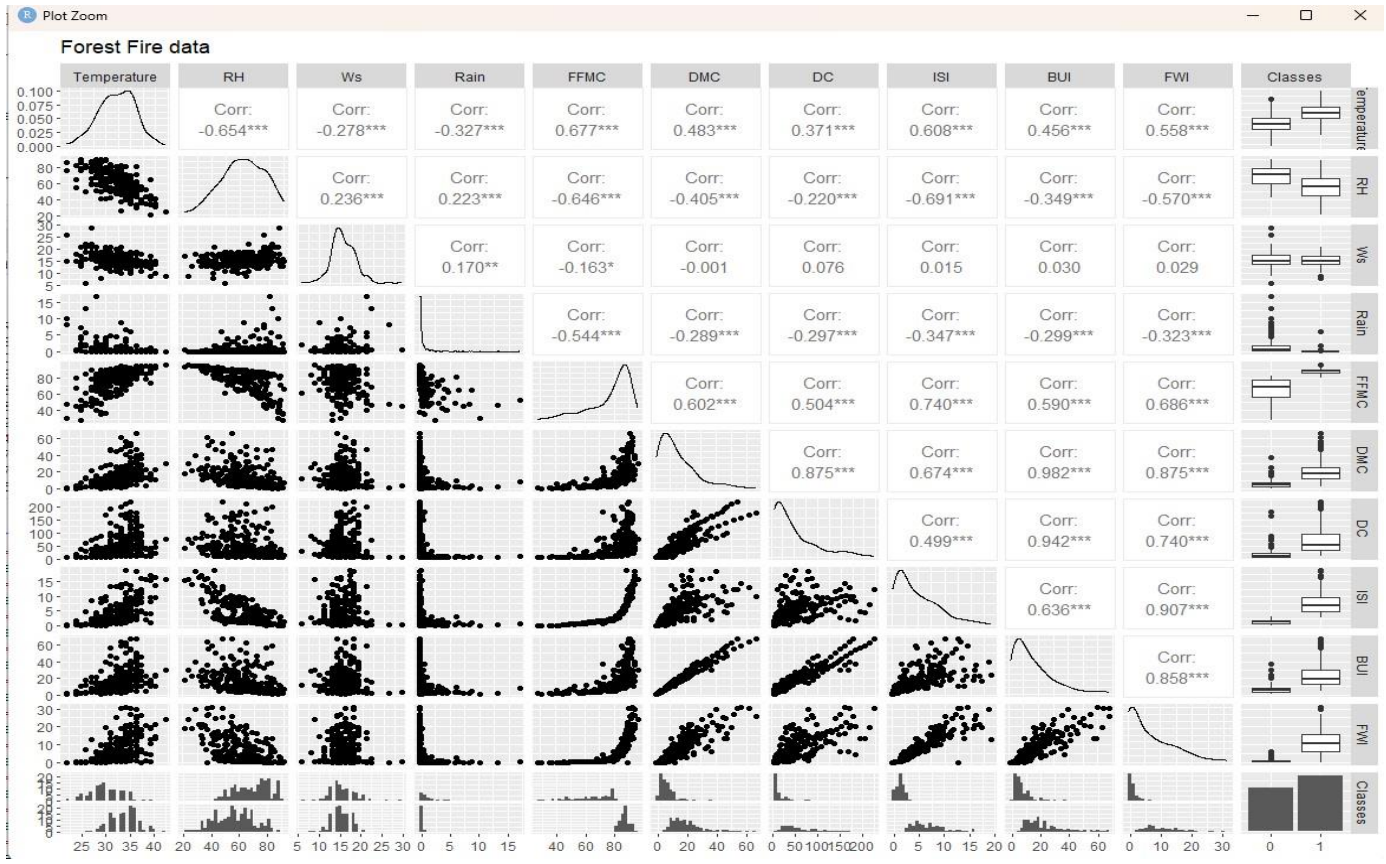

Figure 43

- 2nd plot – creating a map to visualize the relationship between variables.

```
35  # 4 - 2nd Plot
36
37  ggpairs(data=mydata,mapping=aes(color=Classes),title="Forest Fire data")
38
```

Figure 56

```
> ggscatmat(data=mydata, color="Classes",alpha=0.8)
Warning message:
In ggscatmat(data = mydata, color = "Classes", alpha = 0.8) :
  Factor variables are omitted in plot
>
```

Figure 57 (a)

```
> ggpairs(data=mydata,mapping=aes(color=Classes),title="Forest Fire data")
 plot: [11, 1] [=============================================================================================================>---------] 92% est: 1s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 2] [=============================================================================================================>---------] 93% est: 1s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 3] [=============================================================================================================>--------] 93% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 4] [=============================================================================================================>-------] 94% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 5] [=============================================================================================================>------] 95% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 6] [=============================================================================================================>------] 96% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 7] [=============================================================================================================>----] 97% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 8] [=============================================================================================================>---] 98% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 9] [=============================================================================================================>--] 98% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [11, 10] [============================================================================================================>-] 99% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
```

Figure 57 (b)

Figure 44

- 3rd plot – creating visuals from numerical variables.

```
39 # 5 - 3rd Plot
40 # A matrix with scatterplots in the lower diagonal, densities on the diagonal and correlations written in the upper diagonal
41
42 ggscatmat(data=mydata, color="Classes",alpha=0.8)
```

Figure 58

```
> ggscatmat(data=mydata, color="Classes",alpha=0.8)
Warning message:
In ggscatmat(data = mydata, color = "Classes", alpha = 0.8) :
  Factor variables are omitted in plot
>
```

Figure 59

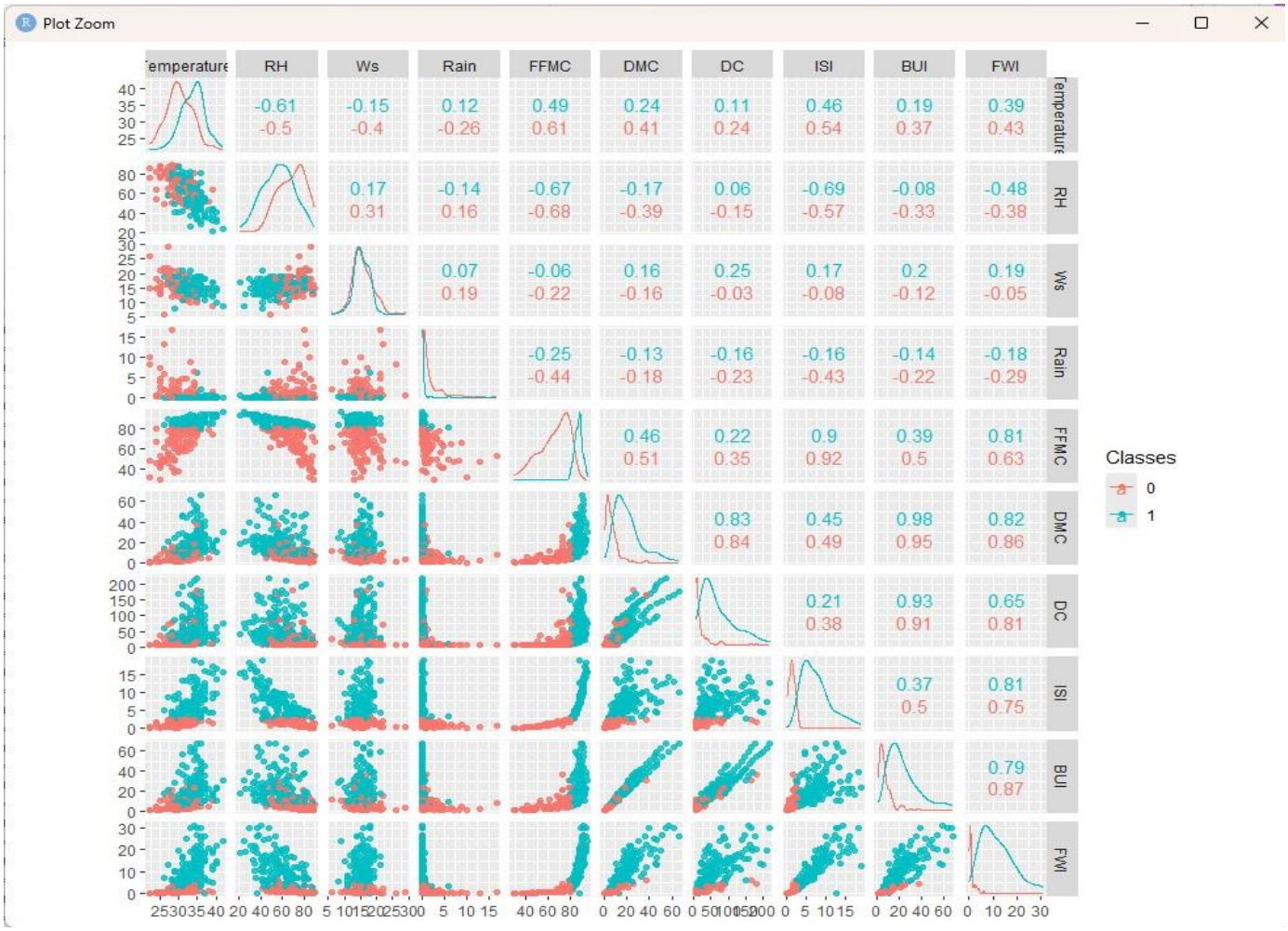

Figure 45

- Model 01 - Building a linear regression model with “Temperature” and “Rain” variables.

```
# 6 - Model 1 - Building a linear regression model with Temperature and Rain variables

lm1 <- lm(Temperature ~ Rain  , data = mydata)
summary(lm1)

ggplot(mydata, aes(x = Temperature, y =   Rain)) +
  geom_point() +
  geom_smooth(method = "lm", color = "red")

```

Figure 60

```
> lm1 <- lm(Temperature ~ Rain  , data = mydata)
> summary(lm1)

Call:
lm(formula = Temperature ~ Rain, data = mydata)

Residuals:
    Min      1Q  Median      3Q     Max
-8.5645 -2.6239  0.3761  2.4949  9.3761

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  32.6239     0.2358 138.365  < 2e-16 ***
Rain         -0.5939     0.1104  -5.379 1.77e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.441 on 242 degrees of freedom
Multiple R-squared:  0.1068,    Adjusted R-squared:  0.1031
F-statistic: 28.93 on 1 and 242 DF,  p-value: 1.766e-07
```

Figure 61 (a)

```
> ggplot(mydata, aes(x = Temperature, y =   Rain)) +
+   geom_point() +
+   geom_smooth(method = "lm", color = "red")
`geom_smooth()` using formula = 'y ~ x'
>
```

Figure 61 (b)

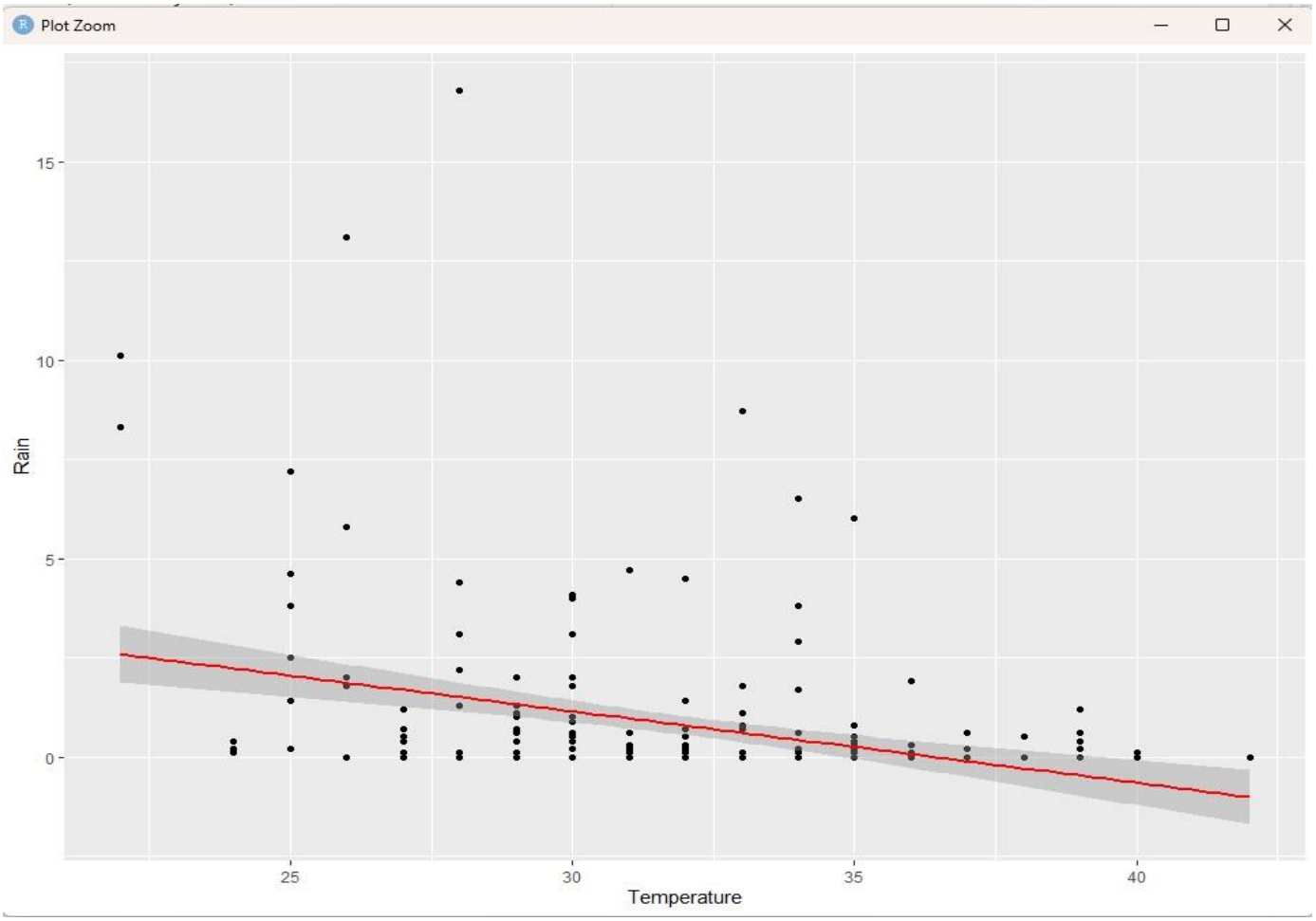

Figure 46

- Model 02 - Building a linear regression model with “Temperature” and “Ws” (wind speed).

```
53  # 7 - Model 2 - Building a linear regression model with Temperature vs ws (wind speed)
54
55  lm2 <- lm( Temperature  ~  Ws , data =  mydata)
56  summary(lm2)
57
```

Figure 62

```
> lm2 <- lm( Temperature  ~  Ws , data =  mydata)
> summary(lm2)

Call:
lm(formula = Temperature ~ Ws, data = mydata)

Residuals:
     Min       1Q   Median       3Q      Max
-10.3534  -2.3534  -0.0333   2.7255   7.6466

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 37.74822    1.25799  30.007  < 2e-16 ***
Ws          -0.35965    0.07984  -4.504 1.04e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.498 on 242 degrees of freedom
Multiple R-squared:  0.07736,   Adjusted R-squared:  0.07354
F-statistic: 20.29 on 1 and 242 DF,  p-value: 1.036e-05
```

Figure 63

- Model 02 – the code and the result for building a linear regression model with “Temperature” and “Ws” (wind speed).

```
58  # 8 - Plotting the linear regression model
59
60  ggplot(mydata, aes(x = Temperature   , y = Ws)) +
61    geom_point() +
62    geom_smooth(method = "lm", color = "blue")
```

Figure 64

```
> ggplot(mydata, aes(x = Temperature   , y = Ws)) +
+   geom_point() +
+   geom_smooth(method = "lm", color = "blue")
`geom_smooth()` using formula = 'y ~ x'
>
```

Figure 65

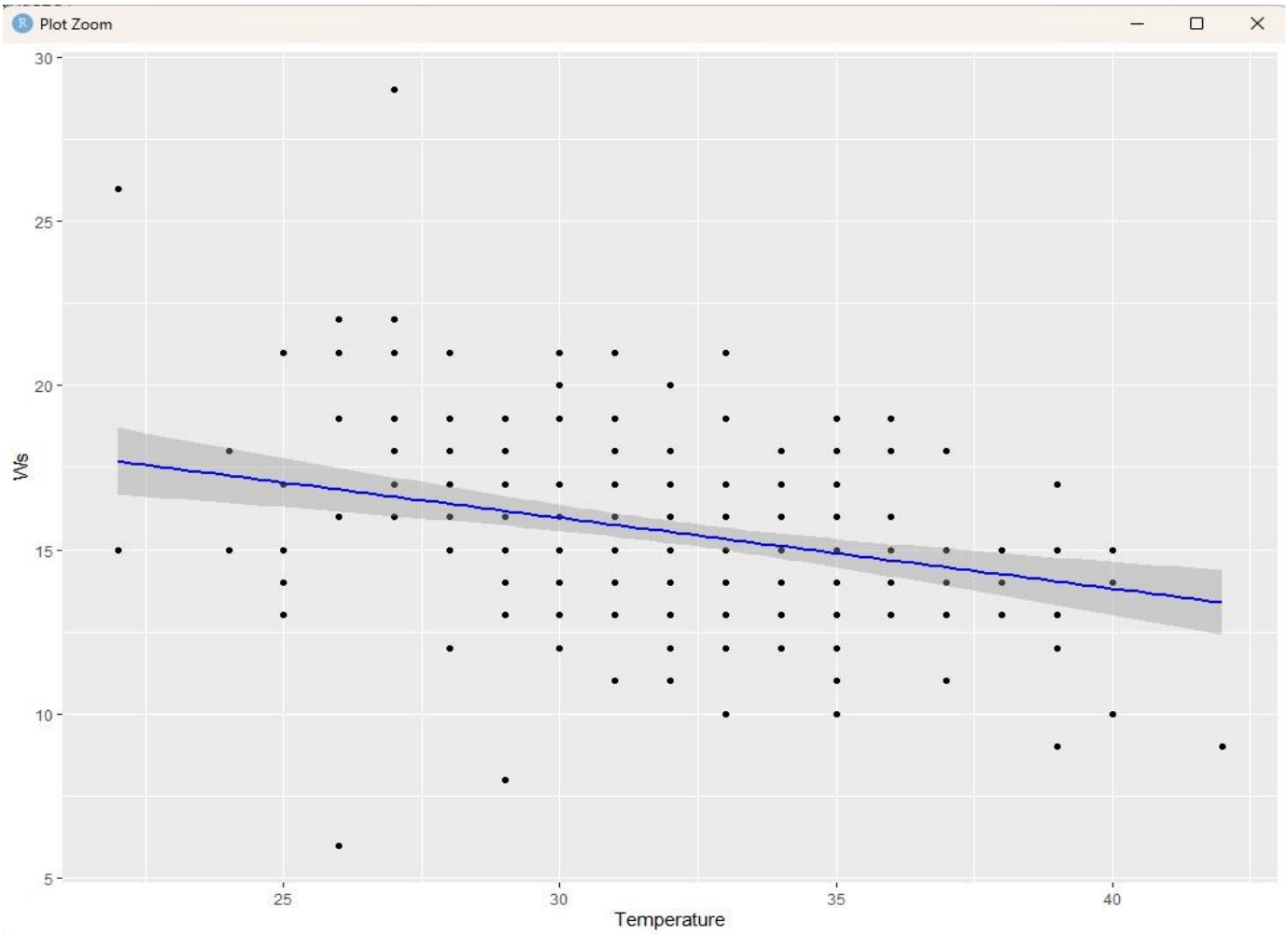

Figure 47

- Dividing the dataset sample to train and test the model. Train = 80%, Test = 20%.

```
64  # 9 - Porttion data - train(80%) and test(20%)
65
66  pd <- sample(2, nrow(mydata),replace=TRUE, prob=c(0.8,0.20))
67  pd
68  train <- mydata[pd==1,]
69  head(train)
70  test <- mydata[pd==2,]
71  head(test)
72
```

Figure 66

```
> pd <- sample(2, nrow(mydata),replace=TRUE, prob=c(0.8,0.20))
> pd
  [1] 1 1 2 1 2 1 1 2 1 1 1 1 1 2 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1
 [69] 1 1 2 1 1 2 2 2 2 1 1 2 2 1 1 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 2 2 1 1 1 2 2 1 1 1 2 2 1 1 2 2 2 1 1 1 1 1 1 2 1 2 1 1 1 1 2
[137] 1 1 1 1 1 1 2 1 2 1 1 2 1 1 1 1 1 1 1 2 1 1 2 1 1 1 1 1 2 2 2 1 1 1 1 1 2 1 1 1 1 1 1 2 1 1 1 2 1 2 1 1 2 1 1 1 1 1 1 1
[205] 1 2 1 2 2 1 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 2 1 1 1 2 1 1 2 1 1 1 1 1 2 1 2 1 1 1 1
> train <- mydata[pd==1,]
> head(train)
  Temperature RH Ws Rain FFMC DMC   DC ISI  BUI FWI Classes
1          29 57 18  0.0 65.7 3.4  7.6 1.3  3.4 0.5       0
2          29 61 13  1.3 64.4 4.1  7.6 1.0  3.9 0.4       0
4          25 89 13  2.5 28.6 1.3  6.9 0.0  1.7 0.0       0
6          31 67 14  0.0 82.6 5.8 22.2 3.1  7.0 2.5       1
7          33 54 13  0.0 88.2 9.9 30.5 6.4 10.9 7.2       1
9          25 88 13  0.2 52.9 7.9 38.8 0.4 10.5 0.3       0
> test <- mydata[pd==2,]
> head(test)
   Temperature RH Ws Rain FFMC  DMC   DC ISI  BUI FWI Classes
3           26 82 22 13.1 47.1  2.5  7.1 0.3  2.7 0.1       0
5           27 77 16  0.0 64.8  3.0 14.2 1.2  3.9 0.5       0
8           30 73 15  0.0 86.6 12.1 38.3 5.6 13.5 7.1       1
15          28 80 17  3.1 49.4  3.0  7.4 0.4  3.0 0.1       0
18          31 78 14  0.3 56.9  1.9  8.0 0.7  2.4 0.2       0
33          32 76 20  0.7 63.1  2.6  9.2 1.3  3.0 0.5       0
>
```

Figure 67

- Model 03 – creating a logistic regression model.

```
73  # 10 - Model 3 - Logistic regression model
74
75  model_glm <- glm(Classes ~ Temperature, data = train, family = "binomial")
76
77  summary(model_glm)
78
```

Figure 68

```
> model_glm <- glm(Classes ~ Temperature, data = train, family = "binomial")
> summary(model_glm)

Call:
glm(formula = Classes ~ Temperature, family = "binomial", data = train)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -12.00737    1.97986  -6.065 1.32e-09 ***
Temperature   0.37595    0.06119   6.144 8.03e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 258.59  on 186  degrees of freedom
Residual deviance: 203.70  on 185  degrees of freedom
AIC: 207.7

Number of Fisher Scoring iterations: 4
```

Figure 69

- Plotting the logistic regression model.

```
79  # 11 - Plotting the logistic regression model
80
81  mydata %>%
82    mutate(Out = ifelse( Classes== "1", 1, 0)) %>%
83    ggplot(aes(Temperature, Out )) +
84    geom_point(alpha = .15) +
85    geom_smooth(method = "glm",method.args = list(family = "binomial")) +
86    ggtitle("Logistic regression model fit") +
87    xlab("Temperature") +
88    ylab("Classes ")
89
```

Figure 70

```
> mydata %>%
+    mutate(Out = ifelse( Classes== "1", 1, 0)) %>%
+    ggplot(aes(Temperature, Out )) +
+    geom_point(alpha = .15) +
+    geom_smooth(method = "glm",method.args = list(family = "binomial")) +
+    ggtitle("Logistic regression model fit") +
+    xlab("Temperature") +
+    ylab("Classes ")
`geom_smooth()` using formula = 'y ~ x'
>
```

Figure 71

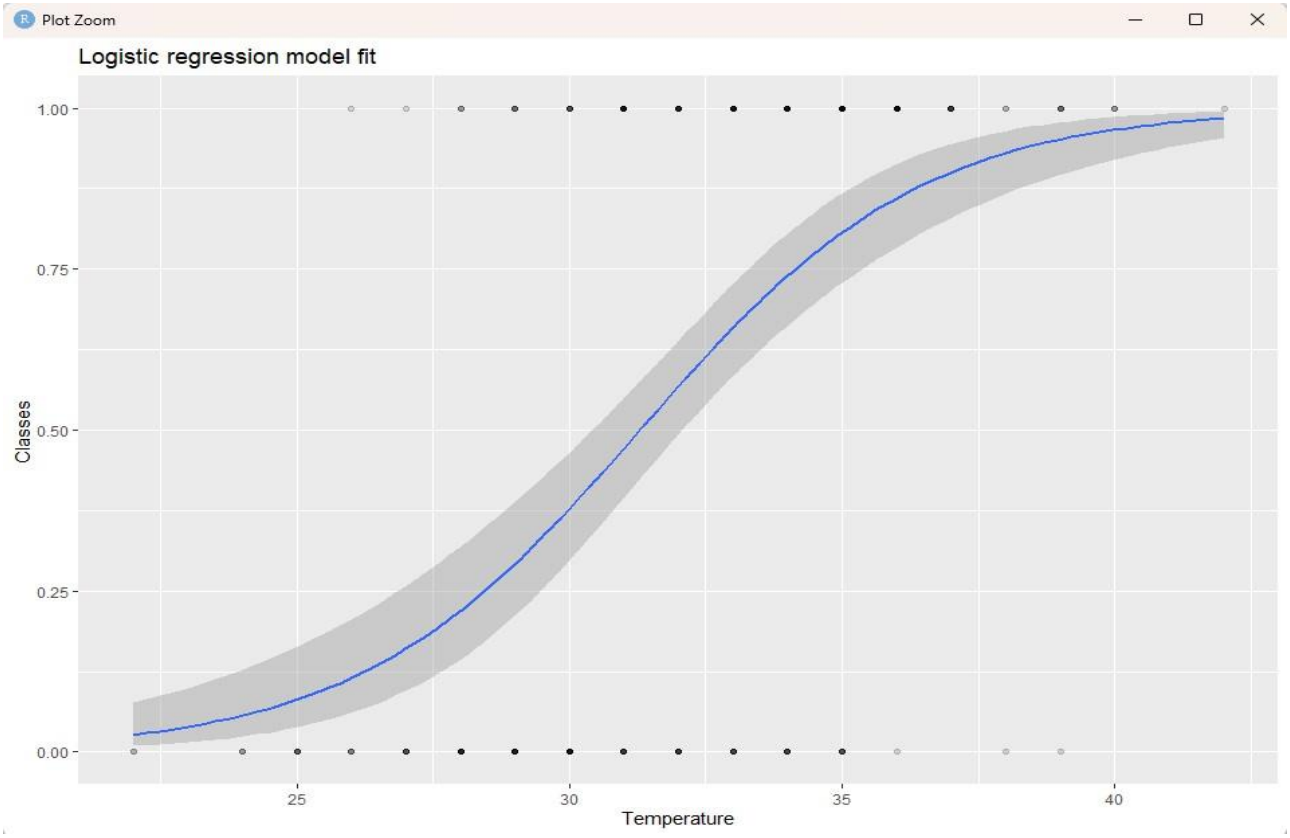

Figure 48

- Model 04 - Building a logistics regression model to check whether we can predict the dependent variable (“Classes”) given all the independent variables.

```
90  # 12 - Model 4 - Building a logistic regression model to check whether we can predict
91  #                the dependent variable ("Classes") given all the independent variables
92
93  my_model<- glm(Classes ~ ., data = train, family = "binomial")
94  summary(my_model)
```

Figure 72

```
> my_model<- glm(Classes ~ ., data = train, family = "binomial")
Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
> summary(my_model)

Call:
glm(formula = Classes ~ ., family = "binomial", data = train)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.713e+03  1.812e+06  -0.001    0.999
Temperature -2.180e+01  3.753e+03  -0.006    0.995
RH           4.429e-01  8.880e+02   0.000    1.000
Ws          -5.920e+00  5.538e+03  -0.001    0.999
Rain         5.602e+01  3.053e+04   0.002    0.999
FFMC         2.690e+01  2.116e+04   0.001    0.999
DMC         -2.782e+01  2.915e+04  -0.001    0.999
DC          -1.200e+00  5.094e+03   0.000    1.000
ISI          7.975e+01  5.551e+04   0.001    0.999
BUI          2.904e+01  3.486e+04   0.001    0.999
FWI          7.433e+00  4.261e+04   0.000    1.000

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2.5859e+02  on 186  degrees of freedom
Residual deviance: 2.1304e-07  on 176  degrees of freedom
AIC: 22

Number of Fisher Scoring iterations: 25

>
```

Figure 73

- Creating logistic regression model predictions.

```
96  # 13 - Prediction
97
98  P1<-predict(my_model,train,type='response')
99  head(P1)
00  head(train)
```

Figure 74

```
> P1<-predict(my_model,train,type='response')
> head(P1)
           1            2            4            6            7            9
2.220446e-16 2.220446e-16 2.220446e-16 1.000000e+00 1.000000e+00 2.220446e-16
> head(train)
  Temperature RH Ws Rain FFMC DMC   DC ISI  BUI FWI Classes
1          29 57 18  0.0 65.7 3.4  7.6 1.3  3.4 0.5       0
2          29 61 13  1.3 64.4 4.1  7.6 1.0  3.9 0.4       0
4          25 89 13  2.5 28.6 1.3  6.9 0.0  1.7 0.0       0
6          31 67 14  0.0 82.6 5.8 22.2 3.1  7.0 2.5       1
7          33 54 13  0.0 88.2 9.9 30.5 6.4 10.9 7.2       1
9          25 88 13  0.2 52.9 7.9 38.8 0.4 10.5 0.3       0
>
```

Figure 75

- Misclassification of error of trained data.

```
102  # 14 - Misclassification error - train data
103
104  pred1<-ifelse(P1>0.5,1,0)
105  pred1
106  tab1<-table(Predicted=pred1,Actual=train$Classes)
107  tab1
108  accuracy_train<-sum(diag(tab1))/sum(tab1)
109  accuracy_train
```

Figure 76

```
> pred1<-ifelse(P1>0.5,1,0)
> pred1
  1   2   4   6   7   9  10  11  12  13  14  16  17  19  20  21  22  23  24  25  26  27  28  29  30  31  32  34  35  37  38  39  40  41  42
  0   0   0   1   1   0   0   1   1   0   0   0   0   0   0   0   1   0   1   1   1   1   1   1   0   1   0   0   0   1   0   1   0   0   0   0
 43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  66  67  68  69  70  72  73  78  79  82  83  84  85
  0   0   0   0   1   1   1   1   1   0   0   0   1   1   1   1   1   1   1   0   0   0   1   1   1   0   1   1   0   1   1   1   1   1   1
 86  88  89  90  91  92  93  94  96  97  98  99 102 103 104 105 108 109 113 114 115 116 117 118 119 121 123 124 125 126 127 130 132 133 134
  1   1   1   1   0   0   0   0   0   1   0   0   0   0   0   0   0   1   1   0   0   0   0   0   1   0   0   0   0   0   0   0   1   1   1
135 137 138 139 140 141 142 144 146 147 149 150 151 152 153 154 155 157 158 160 161 163 164 165 166 170 171 172 173 174 176 177 178 179 181
  0   0   0   0   0   0   0   0   1   1   1   1   1   0   0   1   1   1   1   1   1   0   1   1   1   1   1   1   1   1   0   0   0   0   1
182 183 184 185 186 187 189 190 191 193 195 196 198 199 200 201 202 203 204 205 207 210 211 212 213 216 217 218 219 220 223 224 225 227 228
  1   1   0   1   1   1   0   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   0   1   0   0   0   1   0   0   1   1
229 231 232 234 235 236 237 239 241 242 243 244
  1   1   1   0   1   1   1   0   0   0   0   0
> tab1<-table(Predicted=pred1,Actual=train$Classes)
> tab1
         Actual
Predicted  0  1
        0 88  0
        1  0 99
> accuracy_train<-sum(diag(tab1))/sum(tab1)
> accuracy_train
[1] 1
>
```

Figure 77

- Misclassification error of test data.

```
11  # 15 - Misclassification error - test data
12
13  P2<-predict(my_model,test,type='response')
14  pred2<-ifelse(P2>0.5,1,0)
15  pred2
16  tab2<-table(Predicted=pred2,Actual=test$Classes)
17  tab2
18  accuracy_test<-sum(diag(tab2))/sum(tab2)
19  accuracy_test
```

Figure 78

```
> P2<-predict(my_model,test,type='response')
> pred2<-ifelse(P2>0.5,1,0)
> pred2
  3   5   8  15  18  33  36  65  71  74  75  76  77  80  81  87  95 100 101 106 107 110 111 112 120 122 128 129 131 136 143 145 148 156 159
  0   0   1   0   0   0   1   1   1   1   1   1   0   1   1   1   0   0   0   0   0   1   1   1   0   0   1   1   0   0   1   0   1   1   1
162 167 168 169 175 180 188 192 194 197 206 208 209 214 215 221 222 226 230 233 238 240
  0   1   1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   1   1   1   0   1
> tab2<-table(Predicted=pred2,Actual=test$Classes)
> tab2
         Actual
Predicted  0  1
        0 18  1
        1  0 38
> accuracy_test<-sum(diag(tab2))/sum(tab2)
> accuracy_test
[1] 0.9824561
>
```

Figure 79

- Goodness of fit test.

```
# 16 - Goodness_of_fit_test

with(my_model,pchisq(null.deviance-deviance,df.null-df.residual,lower.tail=F))

```

Figure 80

- Result of goodness of fit test.

```
> with(my_model,pchisq(null.deviance-deviance,df.null-df.residual,lower.tail=F))
[1] 8.465084e-50
>
```

Figure 81

2.6. Results analysis and discussion

The logistic regression of the dataset predicts the chance of an occurrence of a fire outbreak; whether a forest fire would occur or not from the dependent variable “Classes” where 0 = “Not Fire” and 1 = “Fire” which is predicted through the independent variables of the dataset.

The dataset is divided to train and test the model where 80% of the dataset is trained to predict the occurrence while the remaining 20% is used to test the dataset.

Accordingly, it builds two linear regression models (lm1 and lm2) using the lm() function to predict "Temperature" based on "Rain" and "Ws" (wind speed) respectively. Then, it displays summaries of these models and visualizes them using ggplot() while three types of plots are created using the ggpairs(), ggscatmat(), and ggplot() functions to visualize relationships between variables in the dataset.

During the model evaluation, it calculates and displays misclassification error rates for both the training and testing datasets and we got accuracy level which is 96.15.

Finally, using the goodness-of-fit test using the chi-square test (pchisq()) we assess how well the logistic regression model fits the data where we got the result as 8.465.

2.7.Conclusion

The dataset includes 244 instances which consists of 245 rows and 15 columns where 14 columns contain feature variables which are considered as the independent variables while 1 column is a target variable which is considered as a dependant variable.

The dependent variable “Classes” is considered to be the target variable while the independent variables of the dataset are the feature variables. Here, according to the logistic regression technique, the dataset is divided to train and test the model where 80% of the dataset is trained to predict the occurrence while the remaining 20% is used to test the dataset.

Our findings on regards to the dataset, emphasizes the significance of numerous major parameters in affecting fire occurrence, such as temperature, humidity, wind speed, and moisture content of forest fuel. These variables influence fire behavior and spread, highlighting the intricate connections between climatic conditions and vegetation traits in fire-prone environments.

This model allows us to evaluate the contribution of each predictor variable to the chance of a fire outbreak, providing important insights into the causes of forest fires which can be used by forest management authorities, emergency responders, and policymakers as an effective decision making on regards to resource allocation, improving forest fire management and resilience by providing decision-makers with useful tools for anticipating and mitigating the hazards posed by wildfires in Algeria's forests ultimately reducing the impact of forest fires.

3. References

1. Abid, F. and Izeboudjen, N. (2020). Predicting Forest Fire in Algeria Using Data Mining Techniques: Case Study of the Decision Tree Algorithm. Advances in Intelligent Systems and Computing, pp.363–370. doi:https://doi.org/10.1007/978-3-030-36674-2_37.

2. çiçekli, U. and Kabasakal, İ. (2021). Market Basket Analysis of Basket Data with Demographics: A Case Study in E-Retailing. In: Alphanumeric Journal 9(1):1-12. [online] Available at: https://www.researchgate.net/publication/352806803_Market_Basket_Analysis_of_Basket_Data_with_Demographics_A_Case_Study_in_E-Retailing.

3. data.world. (n.d.). data.world. [online] Available at: https://data.world/miles-away/market-basket-analysis/workspace/file?filename=groceries_final.csv [Accessed 5 May 2024].

4. Stanley, M. (2022). Wildfires. [online] education.nationalgeographic.org. Available at: https://education.nationalgeographic.org/resource/wildfires/.

4. Appendices

***Figure 01*:** Market basket analysis dataset.
***Figure 02*:** preprocessing of the dataset – downloading the dataset as a csv file.
***Figure 03*:** preprocessing of the dataset – sorting the columns of the dataset according to the ascending order.
***Figure 04 (a)*:** preprocessing of the dataset – removing the duplicates from the columns of the dataset.
***Figure 04 (b)*:** preprocessing of the dataset – removing the duplicates from the columns of the dataset.
***Figure 05*:** transpose the order of the columns and the rows of the dataset.
***Figure 06*:** replacing the characters attributes of the dataset with "Yes" or "No".
***Figure 07 (a)*:** plotting a barplot to explore the dataset.
***Figure 07 (b)*:** plotting a barplot to explore the dataset.
***Figure 08*:** plot to plot rules.
***Figure 09*:** plot to plot rules in groups.
***Figure 10*:** plot to display the support, confidence and lift with a scatterplot matrix.
***Figure 11 (a)*:** plotting a scatterplot to plot the association rules.
***Figure 11 (b)*:** plotting a scatterplot to plot the association rules.
***Figure 12*:** importing, viewing and reading the dataset file.
***Figure 13*:** inspecting the dataset.
***Figure 14 (a)*:** results from inspecting the dataset.
***Figure 14 (b)*:** results from inspecting the dataset.
***Figure 14 (c)*:** results from inspecting the dataset.
***Figure 15*:** checking the dimensions of the dataset.
***Figure 16*:** the results from checking the dimensions of the dataset.
***Figure 17*:** plotting and exploring the dataset.
***Figure 18 (a)*:** the results from plotting and exploring the dataset.
***Figure 18 (b)*:** the results from plotting and exploring the dataset.
Figure 18 (c)*:** the results from plotting and exploring the dataset ***(Figure 07 (a), (b))
***Figure 19*:** installing "arules" package.
***Figure 20*:** creating the association rule.
***Figure 21*:** the results from creating the association rule.
***Figure 22*:** getting the summary of the rules and inspecting them.
***Figure 23*:** the results from getting the summary of the rules and inspecting them.
***Figure 24*:** reducing the number of rules to a smaller quantity.
***Figure 25*:** the results from reducing the number of rules to a smaller quantity.
***Figure 26*:** getting the summary of the rules, inspecting them and getting the summary of the dataset.
***Figure 27 (a)*:** the results of getting the summary of the rules.
***Figure 27 (b)*:** the results of inspecting the rules.
***Figure 27 (c)*:** the results of inspecting the rules.