CS 634: DATA MINING

Midterm Project Spring 2017

SUBMITTED BY:

Krupali Patel

kjp59@njit.edu

Table of Contents

Project Description:

The following project implements Apriori Algorithm to generate association rules from a set of transactions given.

Apriori Algorithm:

Apriori is an algorithm used to generate frequent item sets and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and then extending them to larger and larger item sets as long as those items appear sufficiently often in the database. The frequent item sets determined by Apriori can be used to determine association rules which highlight general trends in the database. [1]

Platform and Language:

The platform used to develop the project is Eclipse Java Neon. The project is implemented using Java programming language.

Implementation:

The project takes into consideration 5 input transactional files. The user specifies the support and confidence percentage values that the associations should have. There are two class implementations in the project the main, and helper java class. The main class takes the file parameter which is hard coded, and also takes in the user specified support and confidence percentage values and passes the following to the helper.java class. The helper class then performs the Apriori algorithm implementation and prints the association rules for the set of transaction files.

[1] https://en.wikipedia.org/wiki/Apriori_algorithm

Source Code:

Main.java

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Iterator;
import java.util.List;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Scanner;
import java.util.Set;

public class Main {

	public static void main(String args[]) throws FileNotFoundException {
		System.out.println("Apriori Algorithm");
		//input the value of minimum support and confidence
		System.out.println("Enter the minumum support and confidence percentage");

		Scanner sc = new Scanner(System.in);
		int support = sc.nextInt();
		int confidence = sc.nextInt();


		String fileName = "C:/Users/krupali.patel/workspace/Apriori/src/input5.txt";


		Helper helper = new Helper();
		//Pass value of filename, support, & confidence to the helper function
		helper.toMap(fileName,support,confidence);



	}

}
```

Helper.java

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.HashMap;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

public class Helper {
	//Create an HashMap to store key value pairs corresponding to transaction key and
associated values.
	HashMap<String, List<String>> hmap = null;
	//value stores the list of all the values in the hashmap
	List<String> value = null;
	//powerSets contains the list of the combination of pairs of values
	List<String> powerSets;
	//uniqueList stores all unique values of elements
	List<String> uniqueList = null;
	//supportList stores the list of values that are greater than the minimum support
	List<String> supportList = null;
	int support, confidence;

	public Helper() {
		uniqueList = new ArrayList<String>();
		hmap = new HashMap<String, List<String>>();
		value = new ArrayList<String>();
		supportList = new ArrayList<String>();
	}

	//toMap function adds the values from the transaction map into a HashMap.
	// the function also calls the unique values function and calls other functions.
	public void toMap(String fileName, int supp, int conf) {
		support = supp;
		confidence = conf;

		String line = null;
		try {
			// FileReader reads text files in the default encoding.
			FileReader fileReader = new FileReader(fileName);

			// Always wrap FileReader in BufferedReader.
			BufferedReader bufferedReader = new BufferedReader(fileReader);
```

```
                while ((line = bufferedReader.readLine()) != null) {

                    String[] parts1 = line.split("\\t");
                    String[] parts2 = parts1[1].split(";");

                    List<String> hmapValue = Arrays.asList(parts2);
                    value.addAll(hmapValue);
                    if (parts1.length >= 2) {
                        String key = parts1[0];
                        // List will have ";" of parts1[1]

                        hmap.put(key, hmapValue);

                    } else {
                        System.out.println("ignoring line: " + line);
                    }
                }
                bufferedReader.close();

                uniqueValuesFunction(value);

                int r = 10;
                int n_s = uniqueList.size();

                for (int j = 1; j <= r; j++) {
                    printCombination(uniqueList, n_s, j, false);
                }

                int n_c = supportList.size();
                printCombination(supportList, n_c, 2, true);
            } catch (FileNotFoundException ex) {
                System.out.println("Unable to open file '" + fileName + "'");
            } catch (IOException ex) {
                System.out.println("Error reading file '" + fileName + "'");

            }

        }

        //takes as input a String list of all the values present in the transaction file and calculates the
unique elements.
        public void uniqueValuesFunction(List<String> value) {
            Set<String> uniqueValues = new HashSet<String>(value);

            for (String s : uniqueValues) {
                uniqueList.add(s);
            }
```

```
	}
	//this function calculates the combination of the values
	public void combinationUtil(List<String> arr, String data[], int start, int end, int index, int r,
			boolean call_conf) {
		if (index == r) {
			String data_append = "";
			for (int j = 0; j < r; j++) {
				if (!call_conf) {
					data_append = data_append + data[j] + " ";
				} else {
					data_append = data_append + data[j] + ";";
				}
			}

			if (!call_conf) {
				//if call confidence is false, call calculateSupport function
				if (calculateSupport(data_append)) {
					supportList.add(data_append);
					//System.out.println(data_append);

				}

			}

			else {
				//else if call_conf is true, call calcuateConfidence function
				if (calculateConfidence(data_append)) {

					data_append = data_append.replace(";", "->");
					String data_cut =
data_append.substring(0,data_append.length()-2);

					System.out.println(data_cut);
				}
			}

			return;
		}

		for (int i = start; i <= end && end - i + 1 >= r - index; i++) {
			data[index] = arr.get(i);
			combinationUtil(arr, data, i + 1, end, index + 1, r, call_conf);
		}
	}

	public void printCombination(List<String> arr, int n, int r, boolean call_conf) {
		// A temporary array to store all combination one by one
		String data[] = new String[r];

		// Print all combination using temporary array 'data[]'
		combinationUtil(arr, data, 0, n - 1, 0, r, call_conf);
	}

//calculates the support of the value pairs, and returns true if the combination has support greater or
equal to minimum support
	public boolean calculateSupport(String combi_result) {
		boolean insert = false;
		int occuranceItemSet = 0;

		List<String> combi_result_list = Arrays.asList(combi_result.split(" "));

		int tcount = hmap.size();
		int suppCount = 0;

		for (String key : hmap.keySet()) {
			boolean itemPresent = true;
			List<String> keyValue = hmap.get(key);
			for (String check : combi_result_list) {

				if (itemPresent == true) {
					itemPresent = keyValue.contains(check.trim());
				}
			}
			if (itemPresent == true) {
				occuranceItemSet++;
			}

		}
		if (occuranceItemSet > 0) {
			suppCount = (occuranceItemSet * 100) / tcount;
			if (suppCount >= support) {
				insert = true;
			}

		}
		return insert;
	}

	//calculates the confidence of the value pairs that have support >= minimum support, and
returns true if the combination has support greater or equal to minimum confidence
	public boolean calculateConfidence(String sup_list) {
		boolean insert = false;
		//occuranceLeft keeps track of number of occurances of value on the left side of the
itemset
```

```
			int occuranceLeft = 0;
			//occuranceLeft keeps track of number of occurances of value on the left, and right
side of the itemset apprearing together
			int occuranceAll = 0;
			//calc_conf is the value of computed confidence
			int calc_conf = 0;


			String[] left_right_array = sup_list.split(";");
			String left = left_right_array[0];
			String right = left_right_array[1];

			List<String> left_list = Arrays.asList(left.split(" "));
			List<String> right_list = Arrays.asList(right.split(" "));
			// Check weather left items exist

			for (String key : hmap.keySet()) {
				boolean leftpresent = true;
				boolean rightpresent = true;

				List<String> keyValue = hmap.get(key);
				for (String check : left_list) {

					if (leftpresent == true) {
						leftpresent = keyValue.contains(check.trim());
					}
				}

				if (leftpresent == true) {
					occuranceLeft++;
					for (String checkright : right_list) {
						if (rightpresent == true) {
							rightpresent = keyValue.contains(checkright.trim());
						}

					}
					if (rightpresent == true) {
						occuranceAll++;
					}
				}

			}
			if (occuranceLeft > 0) {
				//return true if itemset has confidence > minimum confidence
				calc_conf = (occuranceAll) * 100 / (occuranceLeft);
			}
			if (calc_conf > confidence) {
```

```
                        return true;
                } else {
                        return false;
                }
        }

}
```

Input Files and Output:

This segment describes the input files and the corresponding output for the same.

input1.txt:

```
T01 wheat;rice;tamarind;flour;oil
T02 horlicks;bournvita;noodles;chocolate;dhal
T03 wheat;rice;tamarind;bournvita;noodles
T04 wheat;rice
T05 tamarind;bournvita;noodles
T06 rice;tamarind;horlicks;chocolate;dhal;flour;oil
T07 bournvita;noodles;chocolate;dhal
T08 horlicks;bournvita;wheat;rice
T09 rice;tamarind;flour
T10 noodles;chocolate;dhal
T11 horlicks;bournvita;noodles;chocolate
T12 flour;oil;horlicks
T13 rice;tamarind;flour;oil;horlicks
T14 tamarind;flour;oil
T15 wheat;tamarind;oil;bournvita;chocolate
T16 rice;flour;horlicks;noodles;dhal
T17 tamarind;oil;bournvita;chocolate
T18 rice;flour;horlicks
T19 noodles;dhal
T20 flour;horlicks
```

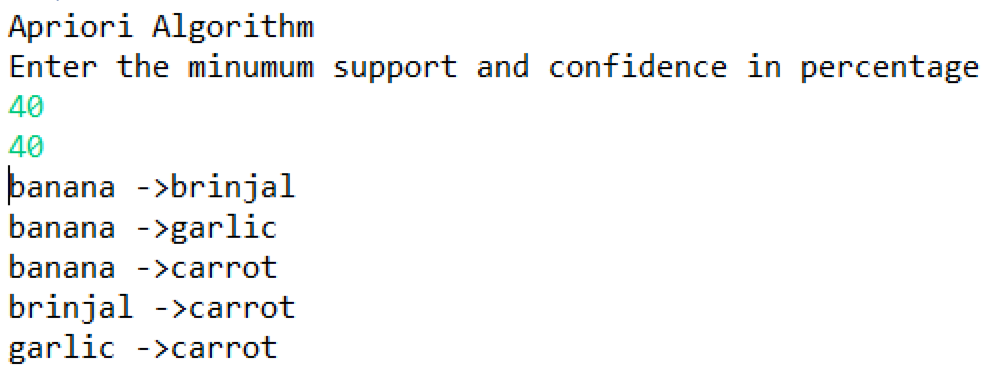
Output1:

```
Enter the minumum support and confidence in percentage
30
40
dhal ->noodles
dhal ->chocolate
dhal ->horlicks
flour ->tamarind
flour ->oil
flour ->rice
flour ->horlicks
flour ->flour rice
flour ->flour horlicks
flour ->tamarind oil
noodles ->bournvita
noodles ->chocolate
tamarind ->bournvita
tamarind ->oil
tamarind ->rice
tamarind ->flour rice
tamarind ->tamarind oil
bournvita ->chocolate
chocolate ->oil
chocolate ->horlicks
chocolate ->tamarind oil
oil ->rice
oil ->horlicks
oil ->flour rice
oil ->flour horlicks
oil ->tamarind oil
rice ->horlicks
rice ->flour rice
rice ->flour horlicks
horlicks ->flour rice
horlicks ->flour horlicks
flour rice ->flour horlicks
flour rice ->tamarind oil
```

input2.txt:

```
T01 broccoli;carrot;garlic;banana;grapes
T02 brinjal;cucumber;avocado;onion;tomato
T03 broccoli;carrot;garlic;cucumber;avocado
T04 broccoli;garlic
T05 garlic;cucumber;avocado
T06 carrot;garlic;tomato;banana;grapes
T07 cucumber;avocado;onion;tomato
T08 brinjal;cucumber;broccoli;carrot
T09 carrot;garlic;banana
T10 avocado;onion;tomato
T11 brinjal;onion
T12 banana;grapes;brinjal
T13 carrot;garlic;banana;grapes;brinjal
T14 garlic;banana;grapes
T15 broccoli;garlic;grapes;cucumber;onion
T16 carrot;banana;brinjal;avocado;tomato
T17 garlic;grapes;cucumber;onion
T18 carrot;banana;brinjal
T19 avocado;tomato
T20 banana;brinjal
```

Output2:



```
Apriori Algorithm
Enter the minumum support and confidence in percentage
40
40
banana ->brinjal
banana ->garlic
banana ->carrot
brinjal ->carrot
garlic ->carrot
```

input3.txt:

```
T01 bandaid;comb;hairdye;basket;hairband
T02 bottle;crayons;axedeo;ink;paper
T03 bandaid;comb;hairdye
T04 bandaid;hairdye;axedeo
T05 hairdye;crayons;axedeo
T06 comb;hairdye;bottle;ink;paper
T07 crayons;axedeo;ink;paper
T08 bottle;crayons;bandaid;comb
T09 comb;hairdye;basket
T10 axedeo;ink;paper
T11 bottle;crayons;axedeo;ink
T12 basket;hairband;bottle
T13 comb;hairdye;basket;hairband;bottle
T14 hairdye;basket;hairband
T15 bandaid;hairdye
T16 comb;basket;bottle;axedeo;paper
T17 hairdye;hairband;crayons;ink
T18 comb;basket;bottle
T19 axedeo;paper
T20 basket;bottle
```

Output3:

```
Apriori Algorithm
Enter the minumum support and confidence in percentage
30
40
basket ->comb
basket ->hairdye
basket ->bottle
crayons ->axedeo
crayons ->bottle
crayons ->ink
axedeo ->paper
axedeo ->ink
comb ->hairdye
comb ->bottle
paper ->ink
```

input4.txt

```
T01  SurfacePro4;XboxOne;MicrosoftDisplayDock;MicrosoftBand;Lumia950;RaspberryPi
T02  XboxOne;MicrosoftBand;Fallout4;GoProHERO4;Titanfall
T03  MicrosoftDisplayDockCase;MicrosoftDisplayDock;XboxOne;Titanfall;RaspberryPi

T04  MicrosoftDisplayDock;MicrosoftDisplayDockCase;SurfacePro4;XboxOne;GoProHERO4
T05  SurfacePro4;MicrosoftDisplayDock;MicrosoftDisplayDockCase;XboxOne;MicrosoftBand
T06  XboxOne;MicrosoftBand;Lumia950;RaspberryPi
T07 MicrosoftDisplayDockCase;MicrosoftDisplayDock;XboxOne;SurfacePro4;GoProHERO4;Fallout4
T08 MicrosoftDisplayDock;XboxOne;SurfacePro4;MicrosoftDisplayDockCase;BeatsPowerbeats;RaspberryPi
T09  MicrosoftBand;MicrosoftDisplayDock;XboxOne;SurfacePro4;MicrosoftDisplayDockCase
T10  MicrosoftDisplayDock;MicrosoftDisplayDockCase;Titanfall;Fallout4
T11  Fallout4;MicrosoftDisplayDock;XboxOne;SurfacePro4;RaspberryPi
T12  MicrosoftDisplayDock;XboxOne;SurfacePro4;BeatsPowerbeats
T13  BeatsPowerbeats;Titanfall;RaspberryPi;
T14  MicrosoftDisplayDock;XboxOne;SurfacePro4;RaspberryPi;MicrosoftBand
T15  MicrosoftDisplayDock;MicrosoftDisplayDockCase;XboxOne;MicrosoftBand
T16  SurfacePro4;RaspberryPi;GoProHERO4;BeatsPowerbeats
T17 MicrosoftDisplayDock;SurfacePro4;XboxOne;MicrosoftDisplayDockCase;BeatsPowerbeats;Fallout4
T18  MicrosoftDisplayDock;XboxOne;Titanfall;BeatsPowerbeats
T19 BeatsPowerbeats;MicrosoftDisplayDock;MicrosoftDisplayDockCase;Titanfall;RaspberryPi;MicrosoftBand
T020            MicrosoftDisplayDock;XboxOne;SurfacePro4;GoProHERO4;MicrosoftBand
```

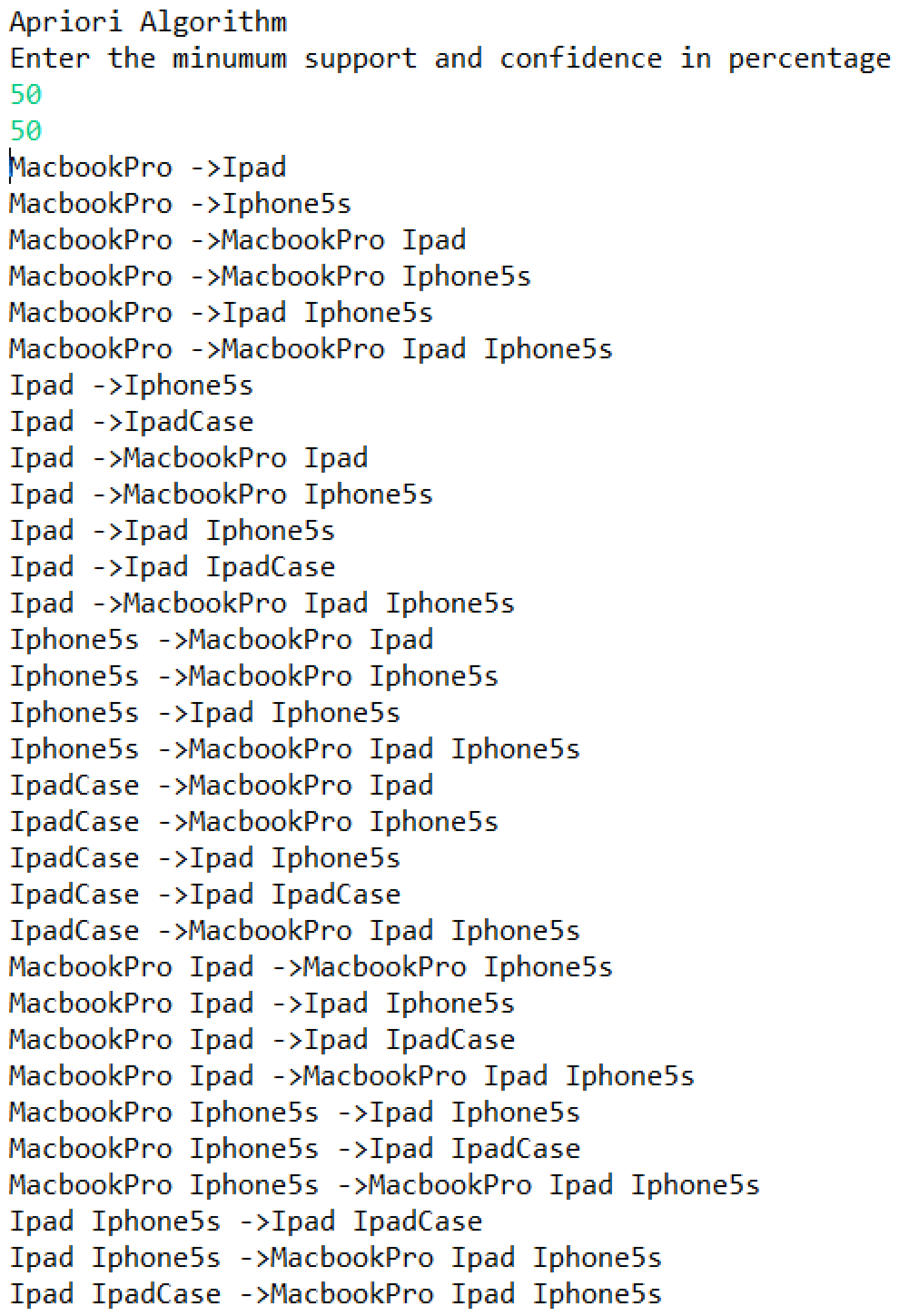
Output4:

```
Apriori Algorithm
Enter the minumum support and confidence in percentage
50
50
XboxOne ->SurfacePro4
XboxOne ->MicrosoftDisplayDock
XboxOne ->XboxOne SurfacePro4
XboxOne ->XboxOne MicrosoftDisplayDock
XboxOne ->SurfacePro4 MicrosoftDisplayDock
XboxOne ->XboxOne SurfacePro4 MicrosoftDisplayDock
MicrosoftDisplayDockCase ->SurfacePro4
MicrosoftDisplayDockCase ->MicrosoftDisplayDock
MicrosoftDisplayDockCase ->XboxOne SurfacePro4
MicrosoftDisplayDockCase ->XboxOne MicrosoftDisplayDock
MicrosoftDisplayDockCase ->MicrosoftDisplayDockCase MicrosoftDisplayDock
MicrosoftDisplayDockCase ->SurfacePro4 MicrosoftDisplayDock
MicrosoftDisplayDockCase ->XboxOne SurfacePro4 MicrosoftDisplayDock
SurfacePro4 ->MicrosoftDisplayDock
SurfacePro4 ->XboxOne SurfacePro4
SurfacePro4 ->XboxOne MicrosoftDisplayDock
SurfacePro4 ->SurfacePro4 MicrosoftDisplayDock
SurfacePro4 ->XboxOne SurfacePro4 MicrosoftDisplayDock
MicrosoftDisplayDock ->XboxOne SurfacePro4
MicrosoftDisplayDock ->XboxOne MicrosoftDisplayDock
MicrosoftDisplayDock ->MicrosoftDisplayDockCase MicrosoftDisplayDock
MicrosoftDisplayDock ->SurfacePro4 MicrosoftDisplayDock
MicrosoftDisplayDock ->XboxOne SurfacePro4 MicrosoftDisplayDock
XboxOne SurfacePro4 ->XboxOne MicrosoftDisplayDock
XboxOne SurfacePro4 ->MicrosoftDisplayDockCase MicrosoftDisplayDock
XboxOne SurfacePro4 ->SurfacePro4 MicrosoftDisplayDock
XboxOne SurfacePro4 ->XboxOne SurfacePro4 MicrosoftDisplayDock
XboxOne MicrosoftDisplayDock ->MicrosoftDisplayDockCase MicrosoftDisplayDock
XboxOne MicrosoftDisplayDock ->SurfacePro4 MicrosoftDisplayDock
XboxOne MicrosoftDisplayDock ->XboxOne SurfacePro4 MicrosoftDisplayDock
MicrosoftDisplayDockCase MicrosoftDisplayDock ->SurfacePro4 MicrosoftDisplayDock
MicrosoftDisplayDockCase MicrosoftDisplayDock ->XboxOne SurfacePro4 MicrosoftDisplayDock
SurfacePro4 MicrosoftDisplayDock ->XboxOne SurfacePro4 MicrosoftDisplayDock
```

input5.txt:

```
T01     MacbookPro;Iphone5s;Ipad;WirelessRouter;LEDTV;MicroSDCard
T02     Iphone5s;WirelessRouter;USBDrive;KindleFireHD;Blender
T03     IpadCase;Ipad;Iphone5s;Blender;MicroSDCard
T04     Ipad;IpadCase;MacbookPro;Iphone5s;KindleFireHD
T05     MacbookPro;Ipad;IpadCase;Iphone5s;WirelessRouter
T06     Iphone5s;WirelessRouter;LEDTV;MicroSDCard
T07     IpadCase;Ipad;Iphone5s;MacbookPro;KindleFireHD;USBDrive
T08     Ipad;Iphone5s;MacbookPro;IpadCase;AppleTV;MicroSDCard
T09     WirelessRouter;Ipad;Iphone5s;MacbookPro;IpadCase
T10     Ipad;IpadCase;Blender;USBDrive
T11     USBDrive;Ipad;Iphone5s;MacbookPro;MicroSDCard
T12     Ipad;Iphone5s;MacbookPro;AppleTV
T13     AppleTV;Blender;MicroSDCard;
T14     Ipad;Iphone5s;MacbookPro;MicroSDCard;WirelessRouter
T15     Ipad;IpadCase;Iphone5s;WirelessRouter
T16     MacbookPro;MicroSDCard;KindleFireHD;AppleTV
T17     Ipad;MacbookPro;Iphone5s;IpadCase;AppleTV;USBDrive
T18     Ipad;Iphone5s;Blender;AppleTV
T19     AppleTV;Ipad;IpadCase;Blender;MicroSDCard;WirelessRouter
T20     Ipad;Iphone5s;MacbookPro;KindleFireHD;WirelessRouter
```

Output5:



```
Apriori Algorithm
Enter the minumum support and confidence in percentage
50
50
MacbookPro ->Ipad
MacbookPro ->Iphone5s
MacbookPro ->MacbookPro Ipad
MacbookPro ->MacbookPro Iphone5s
MacbookPro ->Ipad Iphone5s
MacbookPro ->MacbookPro Ipad Iphone5s
Ipad ->Iphone5s
Ipad ->IpadCase
Ipad ->MacbookPro Ipad
Ipad ->MacbookPro Iphone5s
Ipad ->Ipad Iphone5s
Ipad ->Ipad IpadCase
Ipad ->MacbookPro Ipad Iphone5s
Iphone5s ->MacbookPro Ipad
Iphone5s ->MacbookPro Iphone5s
Iphone5s ->Ipad Iphone5s
Iphone5s ->MacbookPro Ipad Iphone5s
IpadCase ->MacbookPro Ipad
IpadCase ->MacbookPro Iphone5s
IpadCase ->Ipad Iphone5s
IpadCase ->Ipad IpadCase
IpadCase ->MacbookPro Ipad Iphone5s
MacbookPro Ipad ->MacbookPro Iphone5s
MacbookPro Ipad ->Ipad Iphone5s
MacbookPro Ipad ->Ipad IpadCase
MacbookPro Ipad ->MacbookPro Ipad Iphone5s
MacbookPro Iphone5s ->Ipad Iphone5s
MacbookPro Iphone5s ->Ipad IpadCase
MacbookPro Iphone5s ->MacbookPro Ipad Iphone5s
Ipad Iphone5s ->Ipad IpadCase
Ipad Iphone5s ->MacbookPro Ipad Iphone5s
Ipad IpadCase ->MacbookPro Ipad Iphone5s
```

Conclusion:

The following project implements Apriori algorithm and generates association rules for market trends.