Geographic data-mining and extraction of association rules using the Apriori algorithm (Case study: Capital of Iranian provinces)

Farimah Houshmand-Nanehkaran

Islamic Azad University of Kashan

Seyed Mohammadreza Lajevardi (✉ smart.kashanlajevardi@gmail.com)

Islamic Azad University of Kashan <https://orcid.org/0000-0002-4744-2784>

Mahmoud Mahlouji-Bidgholi

Islamic Azad University of Kashan

Manuscript

Keywords: Apriori algorithm, association rules, similar cities, transaction database, urban data mining

Posted Date: February 1st, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-160264/v1>

License: This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Geographic data-mining and extraction of association rules using the Apriori algorithm (Case study: Capital of Iranian provinces)

Farimah houshmand-Nanehkaran¹ . Seyed-Mohammadreza Lajevardi¹ . Mahmoud Mahlouji-Bidgholi¹

Abstract

Extracting of association rules between urban features provides latent and considerable information for urban planners about the relationships between urban characteristics and their similarities. For this purpose, in this paper, the most famous and well-known Apriori algorithm is used. We present the Fariori algorithm to delay the characteristics that can be deleted during execution, as well as to achieve main and frequent features in the early stages with efficient changes to the Apriori algorithm. Although the spatial and temporal complexity of both algorithms is exponential based on the number of features, in practice, by implementing the Fariori algorithm in MATLAB, we achieved more rules than the existing software (R, Weka, Market Basket Analysis and, Yarpiz). In the proposed algorithm, it is possible to determine the degree of similarity by adjusting the support and confidence ratio parameters to identify a coherent set of similar cities. The used database includes cities of 31 in the provincial capitals of Iran. Discovering the association rules leads to similar cities finding and can be an efficient aid in the decision-making process.

Keywords Apriori algorithm . association rules . similar cities . transaction database . urban data mining

✉ Farimah houshmand-Nanehkaran
hoshmandcomputer@gmail.com
ORCID: 0000-0003-1687-1719

Seyed-Mohammadreza Lajevardi (Corresponding author)
R.Lajevardi@iaukashan.ac.ir
ORCID: 0000-0002-4744-2784

Mahmoud Mahlouji-Bidgholi
M.Mahlouji@iaukashan.ac.ir
ORCID: 0000-0001-8895-8501

¹ Department of Computer Engineering, Kashan Branch, Islamic Azad University, Kashan, Iran

1 Introduction

Recognizing the similarity of cities is a useful method for urban planning that leads to the repetition of positive decisions for similar cities. Association rules make it possible to identify cities by discovering the relationships between urban features and provide a better perspective of them. One of the most popular methods to extract association rules is Apriori. In this paper, we have provided useful modifications to the Apriori method by suggestions to increase the speed and accuracy of it.

We explain the association rules with an example. Suppose city *A* has features 1 and 2, city *B* has features 2 and 3, city *C* has features 1 and 3, and city *D* has one feature 1. If the rule is $1 \Rightarrow 3$, that is, every city has feature 1, it will probably have feature 3, as well. Among cities, only *C* has both attributes, but since cities *A*, *C*, and *D* have feature 1, they are likely to have feature 3. It allows urban planners to decide not only based on an existing attribute (feature 1) but also on the possibility of existence feature 3 in cities *A*, *C*, and *D*. Although this also depends on the amount of probability that is called confidence (conf). In this example, since 1 out of 3 cities follows that rule, so the confidence of the rule is 33.4%. It allows urban decision-makers to consider more comprehensive issues and increase the likelihood of success in urban projects by considering the future needs of cities.

Using the proposed Fariori algorithm for data collected from Iranian cities, we have extracted the association rules and also explained how to interpret them in geographical applications by finding hidden patterns. In this paper, based on the 11 main features presented in Tables 1 and 2, 31 cities in the center of the provinces of Iran (Fig. 1) are examined to find similarities. Information is obtained from the 2016 annual reports of the following two websites:

1. Meteorological Organization of Iran (IRIMO) (“I.R. of IRAN Meteorological Organization,” 2020)
2. Statistical Center of Iran (SCI) (*Statistical Center of Iran > Iran Statistical Yearbook > Statistical-Yearbook-2016-2017*, 2020).

The geographical coordinates of the cities are shown in Appendix A, Table 13.

[Figure 1 about here.]

[Table 1 about here.]

[Table 2 about here.]

2 Related Works

In data mining, association rule learning is a popular and well-researched method to discover interesting relationships between variables and identify hidden patterns among items in the big database. The concept of association rules first was presented by Agrawal et al [3]. They introduced association rules to discover patterns among products in the large-scale transaction database. Today, the association rules are very successful in many applications, such as market basket [4], computer networks [5], recommendation systems [6], and healthcare [7]. In the field of geographic information sys-

tem (GIS), we can also mention the use of exploring association rules to improve transportation and better access. Appice et al [8] presented some of the association rules that could use new knowledge at the urban level and help direct resources to improve facilities, especially in areas with poor access to transportation.

Sengstock et al [9] have used association rules to improve the extraction of concepts from voluntary geographic information (VGI). Kashian et al [10] have used them to analyze and discover meaningful patterns among points registered in a location database such as Open Street Map (OSM).

Some of the well-known algorithms for creating association rules are Apriori, Eclat, and FP-Growth [11]. Apriori is a basic algorithm and breadth-first search (BFS) strategy that has been proposed with different features and functions. One of the changes to the Apriori algorithm is the passing reduction on databases [12]. In another method, sampling is used instead of examining the entire database [13]. Also, the use of parallel algorithms to discover the association rules is proposed and investigated in [14].

Versichele et al [15] have proposed spatial association rules for analyzing tourist attractions. In this paper, it is stated that focusing on different tourist attractions can be strengthened by using association rules. Distributing tourists to diverse areas is better than focusing on a limited one. Goswami and Sah [16] provided a template for exploring frequent patterns of spatial objects. This pattern would be appropriate for a place where the location of things is close to each other. Yum and Kim have provided a combination filtering technique based on customer behaviour patterns and spatial patterns on commercial sites using association rules [17].

Osadchiy et al [18] used association rules to create a dietary intake recommendation system that is resistant to the cold-start problem. Their system builds a model for identifying the similarity of preferences from transactions. Cheng and Cheng [19] extracted the similarity of plant features using the Association Rule-Based Similarity (ARBS) method. Borozanov et al [20] developed an architectural model to search for similar measures and components in enterprise architecture and used the association rules to find interesting elements.

Zhu [21] used the Apriori algorithm to discover the rules of the physical education student database and introduced the lift-measure interest measurement. In this method, by removing some rules and extracting more interesting of them, the complexity of time and space is slightly reduced, but the degree of complexity is like the classic Apriori.

The EAFIM (Efficient Apriori-based Frequent Item-set Mining) method, introduced by Raj et al [22], Simultaneously generates candidates and counts their support values. This method calculates the updated input dataset by eliminating useless items and transactions. Therefore, it reduces the size of input dataset in each iteration.

The FTARM (Fast Top-K Association Rule Miner) method [23] to find the set of top-k association rules using the RGPP (Rule Generation Property Pruning) technique reduces the search space by analyzing the internal relationships between items. This method seeks to find the K important rules, while our method tries to find all the possible rules by showing its computational power because the existing algorithms sometimes cannot find practically all the rules.

3 Materials and methods

For an item-set ($I = \{i_1, i_2 \dots i_n\}$) as a set of n binary features and for a set of transactions ($D = \{T_1, T_2 \dots T_m\}$) as a database that each transaction of T in D is identified by a unique transaction code (TID) and is a set of items I , a rule shows how frequently an item-set occurs in a transaction. A rule is defined as $X \rightarrow Y$ that $X, Y \subseteq I$ and $X \cap Y = \emptyset$. X and Y are called antecedent and consequent item-sets, respectively. Usually, a database involves hundreds or thousands of transactions.

To find the existing rules between the cities' features and analyzing geographical data, the most crucial steps are: 1) Creating an accurate database. 2) Using the appropriate software for data analysis. In this study, we attempt to create a precise and small sample of urban database and provide a suitable method to recognize the patterns by the proposed algorithm.

In Table 3, the 11 main features of cities are divided into 57 sub-features according to the amount and type of features. In each row of Table 4, all the sub-features of a city are identified. In Table 5, the presence or absence of each sub-feature is investigated for each city. 1 indicates a city has sub-features, and 0 indicates that the city does not have sub-features. In this table, five features in columns are not present in any cities (Rounded line). So practically, 52 features are checked by the algorithm. Köppen's climate classification is well-known, and detailed descriptions are shown in Appendix A, Table 14.

[Table 3 about here.]

[Table 4 about here.]

[Table 5 about here.]

Cities are considered as transactions and urban features as items. The unique names of cities are called transaction identifiers. Each column of Tables 1 and 2 can be numeric, binary, character, binomial, and so on. These tables are a pre-processing classification, and the most important principle in the creation of an urban database is observing the principles of classification and determining the precise type of variables.

If a rule is obtained after processing the data in form $4 \rightarrow 54$, it means that a city with an altitude of 1501 to 2000 meters probably has an average annual precipitation of 301 to 600 mm. In the database, the cities with the feature of 4 are Arak, Hamadan, Isfahan, Kerman, Yasuj, and Zanjan. Of these six cities, only Arak, Hamadan, and Yasuj have an average annual precipitation of 301 to 600 mm. So the confidence of this rule is 50%. It means that with a 50% probability, the cities with feature 4 will also have feature 54. Therefore, each rule will have a specific geographical interpretation.

3.1 Metrics for accepting association rules

For choosing attractive rules from all possible of them, various restrictions are applied to the metrics, importance of metrics, and attractiveness. The most popular limitations are the minimum threshold for support and confidence. Support of item X indicated as $\text{supp}(X)$ is the ratio of the number of transactions that include item X to the total number of transactions. In Table 5, the set of features $\{3, 6\}$ has the support of 0.387 because they occurred in 38.7% of transactions (12 out of 31 transactions).

The confidence of a rule is defined as $conf(X \rightarrow Y) = supp(XUY) / supp(X)$. For instance, in the case of $\{3, 6\} \rightarrow \{11\}$, the confidence is $0.387 / 0.387 = 1$. It means, this rule satisfies 100% of the transactions that include features 3 and 6 (Eq. 1).

$$Rule : (X \rightarrow Y) \left\{ \begin{array}{l} Support = \frac{fre(X, Y)}{N} \\ Confidence = \frac{fre(X, Y)}{fre(X)} \\ Lift = \frac{Support}{Supp(X) \times Supp(Y)} \end{array} \right. , \quad (1)$$

where ‘*fre*’ denotes the frequency of items, and ‘*N*’ is the number of transactions.

In a set of transactions, a rule $X \rightarrow Y$ has a support value of s if X appears in $s\%$ of transactions, and has a confidence value of c if $c\%$ of transactions that include X also includes Y . The rules are called strong that have the support and confidence more than the minimum threshold of support (min-supp) and the minimum threshold of confidence (min-conf), respectively. The original purpose of association rule mining algorithms is to discover patterns that represent strong rules.

In most algorithms, including Apriori, finding association rules is a two-step process. In the first step, a set of important items (items that have more support value than min-supp) is identified, and in the second step, this set of items is used to generate strong association rules. It is important to note that the general efficiency of association rule mining algorithms depends on the first stage of the process [24], and the extraction of association rules is easy after finding important items. In addition to support and confidence metrics, other criteria have been proposed to measure the attractiveness of rules. Important metrics are as follows: 1) Lift [25] 2) Conviction [26] 3) Jaccard [27] 4) Leverage of Novelty [28] 5) Dependency Matrix [29].

3.2 Apriori Algorithm and new Suggestions

Apriori is a machine learning algorithm (unsupervised learning) that seeks to extract association rules from the transactions database. The minimum support is given to the algorithm as an input parameter to determine the minimum number of items that should appear in an association rule.

The Apriori algorithm uses prior knowledge of the important features of item-sets [11]. A set of k -items is called C_k , and a set of k important items is called L_k . This algorithm uses an incremental method in which C_k is used to obtain C_{k+1} . Initially, important single-member sets are chosen, which those are L_1 set, and it includes all items that appear in at least a min-supp percentage of transactions. The L_1 set is used to obtain L_2 , and similarly, L_{k-1} is used to obtain L_k . An important feature that helps improve the performance of the algorithm in extracting L_k s is that all non-empty subsets of an important item-set are important, too. The validity of this proposition is easily understandable; accordingly, a set of items not possessing the minimum support is considered as unimportant items. Adding extra items to such an item-set does not change its rank to important.

The production steps of L_k s are summarized as follows: First, to obtain L_k , a set of k -items is produced by combining the members of L_{k-1} as a candidate that call C_k . This combination is performed only if the members of L_{k-1} have $k-2$ shared items exactly. The next stage regards the pruning stage. C_k is a superset of L_k , and all the members of L_k are in it, but all members of C_k are not necessarily important. A one-pass investigation of the database can identify important members, but a better solution is also available. For this purpose, the Apriori feature is used. That is if any of the subsets of $k-1$ members belonging to C_k are not in L_{k-1} , then surely C_k cannot be an important set. In this case, the database only can be reviewed for some C_k members that meet the conditions.

Suppose a set according to Table 6 contains five transactions. First, each item (A, B, C, etc.) is a member of the candidate set (C_1), and the algorithm finds all items in the database without considering their repetition (Table 7). If the min-supp threshold is 60% (min-supp = 3), items are selected that are in at least 3 transactions. For each item, the number of iterations in the transactions is specified, so item E is removed because the number of iterations is 2 that is less than min-supp. Therefore, a set of one important item (L_1) is easily obtained from C_1 , which includes A, B, C, D, and F. To obtaining L_2 , the algorithm first generates C_2 , which consists of a set of duplicate items. Since items {C, F} and {D, F} are only present in the two transactions, they are removed, and the rest items make up the L_2 set.

[Table 6 about here.]

[Table 7 about here.]

When generating C_3 and C_4 , triple items should not include {C, F}, and {D, F}. To generating C_3 using the Apriori algorithm, which states that all subsets of an important set must be important, it can be detected that, for example, two item-sets {B, C, F} and {B, D, F} cannot be important. For {B, C, F} set, it is observed that the subset {C, F} is not a member of L_2 , and subsequently, in the set {B, D, F}, the subset {D, F} is not a member of L_2 . As a consequence, without further analysis, it can be found out that these two sets cannot be important, and their supports' calculation is not required. As mentioned before, all 2-items subsets of an important set {B, C, D} are important, and after calculating the support, these subsets are placed in L_3 .

3.2.1 The First Suggestion

Table 5 is ready to be used in the algorithm, and depending on the chosen software, appropriate storage files have to be created. The sum of 1s in each column indicates the amount of support for that feature. For instance, $Supp(1) = 6/31$ means feature 1 exists in 6 out of 31 cities. In the next step, the extraction of rules is followed, and finally, strong rules will be extracted by having various minimum confidence.

Since Apriori is a first-breadth search algorithm, to find support for an item, it is necessary to scan the entire database from left to right and top to bottom. To reducing the calculations, it is suggested that another table be created with the number of columns equal to the number of columns in Table 6, and each item be placed separately in itself

column. For example, consider column 1 for item *A*, column 2 for item *B*, etc. (Table 8). The total number of items in each column is considered as the item support for that column. In the example above, Table 8, like Table 6, will have six columns and five rows.

[Table 8 about here.]

The number of operations required to calculate all the supports is obtained according to Eq. 2.

$$\begin{aligned} \text{Number of operations in the Fariori} = \\ (\text{Number of transactions}) \times (\text{Number of items}) \times 2 \end{aligned} \quad (2)$$

Half of the operation is for placing items in the table, and the other half is counting items in all columns. The calculated value for the above example is 60 ($5 \times 6 \times 2$).

In the Apriori method, all operations to obtain support are calculated using Eq. 3.

$$\begin{aligned} \text{Number of operations in the Apriori} = \\ (\text{Total number of items in all transactions}) \times (\text{Number of items}) \end{aligned} \quad (3)$$

In the five transactions of the above example, there are a total of 22 items in 6 different types (*A*, *B*, *C*, *D*, *E*, and *F*). For each item, the operations are repeated 22 times, which requires a total of 132 (22×6) actions. Now, in the same five transactions, suppose the number of items is 100, and the number of items in each transaction is also 100. In the proposed method, the number of operations is 1000 ($5 \times 100 \times 2$), and through the Apriori method, the required operations to calculate the supports are equal to 50000 (500×100), which is 50 times more than the required operations of the proposed method.

3.2.2 The Second Suggestion

In the Apriori algorithm, first, a set of two items is created (step C2), then depending on the amount of support, some items are pruned, and in each step, one item is added to the item-set (i.e., 3-item-set, 4-item-set, etc.). At this step, the proposed method refuses to create a 2-item-set in step C2 and operates intelligently. In our example, the order of the items is arranged based on support from large to small and left to right (Table 9). The last column (*E*) is deleted because it is less than the min-supp value. Then, in Table 10, the items in the columns are entered one by one from left to right, and duplicate combinations of items are not included in the table.

[Table 9 about here.]

[Table 10 about here.]

In the first step, two items *B* and *A* are considered, and the support of the compound item-set {*B*, *A*} is obtained through Table 8 ($Supp(B, A)=4$). In the second step, item *C* is added to the items, and all combinations of these three items are derived without prior

iteration, and support values are calculated for each item-set. In this step, we will have two sets of dual items ($\{A, C\}$, $\{B, C\}$) and a set of triple items ($\{B, A, C\}$). Previously, the set of two items $\{B, A\}$ was examined in the first step and is not repeated now. The amounts of support for each item-set indicates that these are frequent. If a set has less support than min-supp, it will be removed like the Apriori algorithm, and its combinations will not exist in the next steps.

By adding one item in each step, all item combinations are considered that are not similar to the Apriori algorithm. When the itemset is an n -item, all combinations of 2, 3, ..., n items are obtained in the same step ($n-1$). The repetition of single item-sets has previously calculated in Table 9.

3.2.3 The Third Suggestion

We suggest that at each stage, instead of adding one item to produce the compounds, the items should be used that have equal support and have the most support than the others. For example, instead of adding item A to item B , items A, C, D are added because they have $Supp=4$, and in the first step, all combinations of these items are obtained. In other words, several items can be combined instead of one at each stage, provided that the added items have equal or more support. This method reduces the number of example steps from 4 to 2 (Table 11).

[Table 11 about here.]

3.2.4 Association Rules' Learning

As mentioned, the Apriori algorithm does not generate association rules and only creates frequent patterns for generating association rules, which means that after finding a set of important sets of items, the next step is to find strong association rules from this set of items. In this method, the association rules are obtained as follows:

- For each important item-set, all non-empty subsets are generated.
- For each subset, if it was important, the rule is added to the set of rules.

Since important sets have already been generated, there is no need to recalculate the support value for them and their subsets.

The time and space complexity are $O(2^i)$ for the Apriori and Fariori algorithms, where i is the total number of items in the transactional dataset. The worst state occurs only if the threshold support is zero, or the support for each subset (set of unique items) is greater than the threshold (Fig. 2). In the Apriori algorithm, at the first step, it needs to store i_{C_1} candidates, in 2nd i_{C_2} , in 3rd it will be i_{C_3} , till we reach to the last step where it needs to store i_{C_i} items. So if we sum up the above terms, it will be $i_{C_1} + i_{C_2} + \dots + i_{C_i} = 2^i - 1$, which is $O(2^i)$. The important difference between the Apriori algorithm and the Fariori algorithm is that the Fariori algorithm creates important items in the early steps and delays the combination of items with a small amount of support. Our goal is to creating association rules in the city database, and the implementation of the Fariori algorithm has enabled us to provide more rules.

[Figure 2 about here.]

4 Discussion

The extracted association rules are valid until a change is made to the database, and if a new transaction set is added, the support and confidence of these rules may change. Therefore, the dataset should be re-examined in terms of strength and robustness. In general, item-sets of the original database and their derived rules, may not be considered important sets in the new database, and vice versa, item-sets that were not previously important may now become important sets. The easiest way to solve the problem is to recalculate important item-sets in the new database, but it is costly. In the Fariori method, the important item-sets of any size are found in each pass of the database. The candidates are created according to the set of important items of the previous step.

The proposed method is implemented in the MATLAB program, and with a support value of 0.3 for the urban database, only 73 rules are created, according to Fig. 3. Rule 73 is $\{25, 49\} \rightarrow \{11\}$ and states that cities with features 25 and 49 with a probability of 0.3258 will also have features 11. As shown in Fig. 4, 10 out of 31 cities have features 49 and 25. Therefore, these cities could also have feature 11 with a probability of 0.3258. The algorithm is implemented in such a way that it can differentiate the cities in terms of rules and make city analysis easy and convenient.

This method increases the ability of the software to obtain more rules by delaying removable items. When the three features 25, 11, and 49 are identified as a frequent item-set in some cities, the following rules are obtained:

$$\begin{array}{llll} \{25\} \rightarrow \{11\}, & \{49\} \rightarrow \{11\}, & \{11\} \rightarrow \{25\}, & \{11\} \rightarrow \{49\}, \\ \{25\} \rightarrow \{49\}, & \{49\} \rightarrow \{25\}, & \{25, 49\} \rightarrow \{11\}, & \{25, 11\} \rightarrow \{49\}, \\ \{49, 11\} \rightarrow \{25\}, & \{25\} \rightarrow \{49, 11\}, & \{49\} \rightarrow \{25, 11\}, & \{11\} \rightarrow \{25, 49\} \end{array}$$

[Figure 3 about here.]

[Figure 4 about here.]

We run our urban database in the “arules” package (Package Authors: Michael Hahsler and Bettina Gruen) in *R* software. The results are as follows:

```
> data <- as(data3, "itemMatrix")
> rules <- apriori(data, parameter = list(minlen=2,
supp = 0.5, conf = 0.6, target = "rules"))
> summary(rules)

set of 6 rules

rule length distribution (lhs + rhs):sizes
2          6

Min.      1st Qu.      Median      Mean      3rd Qu.      Max.
 2          2          2          2          2          2
```

summary of quality measures:				
support	confidence	coverage	lift	count
Min.:0.5161	Min.:0.6207	Min.:0.5161	Min.:0.9621	Min.:16.00
1st Qu.:0.5565	1st Qu.:0.8198	1st Qu.:0.5968	1st Qu.:0.9739	1st Qu.:17.25
Median:0.5806	Median:0.9222	Median:0.6935	Median:1.0393	Median:18.00
Mean:0.6183	Mean:0.8764	Mean:0.7258	Mean:1.0234	Mean:19.17
3rd Qu.:0.7016	3rd Qu.:0.9861	3rd Qu.:0.8871	3rd Qu.:1.0690	3rd Qu.:21.75

The package description states:

“Note: Apriori only creates rules with one item in the RHS (Consequent). The default value in APparameter for minlen is 1. That means that rules with only one item (i.e., an empty antecedent/LHS) like $\{\} \rightarrow \{\text{beer}\}$ ”

In references [30], [31], and [32] there is an implementation of the Apriori algorithm, which we call Market Basket Analysis (MBA), and we run this algorithm for the urban database to get the results for comparison. The corresponding code is shown in Listing 1. Another algorithm used for comparison is the reference [33], which we call Yarpiz.

By keeping the confidence value constant at 0.6 and changing the support value from 0.1 to 0.8, the changes in the number of rules and execution time obtained from R, Weka, MBA, Yarpiz, and Fariori are shown in Table 12.

[Table 12 about here.]

4.1 Software problem

- Weka. At min-supp=0.4 and conf=0.6, one of the rules obtained from Weka is $\{53\} \rightarrow \{6,11\}$. According to the urban database, the value of support for this rule is $12/31 = 0.387$, which is less than min-supp, and it is a problem. Weka calculates the support 0.4, which is 12 out of 31 items and is equal to 12.4 mathematically. This problem arises when it rounds the number of items, so it has a shift in calculations.
- Yarpiz. The Yarpiz algorithm cannot solve the following example.

$$T = \left\{ \begin{array}{l} [1 \ 2 \ 3 \ 4 \ 5 \ 6] \\ [2 \ 3 \ 4 \ 5 \ 6 \ 7] \\ [1 \ 4 \ 5 \ 8] \\ [1 \ 4 \ 6 \ 9 \ 10] \\ [2 \ 4 \ 5 \ 10 \ 11] \end{array} \right\}$$

But the Fariori algorithm generates an association rule for this example in the following form:

$$\begin{array}{llll} 4 \rightarrow 5 & \text{Conf} = 0.8 & \text{Lift} = 1 & \text{Support} = 0.8 \\ 5 \rightarrow 4 & \text{Conf} = 1 & \text{Lift} = 1 & \text{Support} = 0.8 \end{array}$$

Listing 1. Part of the MBA Code in MATLAB

```

minSup = 0.5; minConf = 0.6;

transactions={
{'1', '7', '11', '19', '27', '29', '33', '41', '47', '50', '53'};...
{'4', '6', '11', '22', '24', '28', '36', '41', '46', '50', '54'};...
{'3', '6', '11', '22', '23', '28', '36', '38', '45', '51', '53'};...
{'1', '6', '11', '19', '27', '28', '33', '42', '46', '49', '53'};...
{'3', '6', '11', '18', '25', '28', '35', '39', '47', '49', '53'};...
{'3', '6', '11', '16', '24', '29', '36', '39', '46', '49', '53'};...
{'1', '6', '11', '17', '27', '28', '33', '39', '47', '51', '53'};...
{'1', '6', '11', '21', '25', '30', '34', '38', '44', '50', '54'};...
{'4', '6', '11', '16', '24', '29', '37', '38', '47', '50', '54'};...
{'3', '6', '11', '21', '25', '30', '34', '41', '47', '50', '54'};...
{'4', '7', '11', '16', '25', '28', '34', '39', '47', '49', '53'};...
{'3', '7', '11', '16', '25', '29', '35', '39', '46', '49', '53'};...
{'4', '6', '11', '18', '25', '28', '34', '38', '47', '50', '53'};...
{'3', '6', '11', '21', '25', '30', '35', '39', '46', '49', '54'};...
{'3', '6', '11', '21', '25', '32', '35', '40', '47', '49', '54'};...
{'2', '8', '12', '16', '24', '30', '35', '38', '46', '49', '53'};...
{'3', '6', '11', '21', '24', '29', '36', '38', '46', '50', '53'};...
{'2', '7', '11', '18', '25', '28', '35', '40', '47', '51', '53'};...
{'1', '6', '11', '20', '25', '32', '33', '38', '43', '50', '57'};...
{'3', '6', '11', '22', '24', '29', '36', '39', '47', '48', '54'};...
{'1', '6', '11', '20', '25', '31', '33', '38', '43', '50', '55'};...
{'3', '6', '11', '19', '25', '29', '34', '38', '47', '49', '53'};...
{'5', '6', '11', '16', '24', '31', '37', '40', '47', '51', '54'};...
{'3', '7', '11', '16', '25', '30', '33', '40', '47', '49', '53'};...
{'3', '7', '11', '16', '25', '29', '36', '39', '46', '49', '53'};...
{'3', '10', '15', '16', '25', '29', '33', '38', '46', '52', '53'};...
{'3', '6', '11', '21', '24', '28', '36', '38', '46', '50', '53'};...
{'4', '6', '11', '21', '25', '31', '34', '39', '47', '49', '54'};...
{'3', '6', '11', '18', '25', '28', '33', '41', '47', '50', '53'};...
{'3', '6', '11', '19', '25', '28', '34', '42', '47', '49', '53'};...
{'4', '6', '11', '16', '24', '28', '36', '38', '46', '50', '53'}};

vars={
'1 ', '2 ', '3 ', '4 ', '5 ', '6 ', '7 ', '8 ', '10', '11', '12', ...
'15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '25', ...
'27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', ...
'38', '39', '40', '41', '42', '43', '44', '45', '46', '47', '48', ...
'49', '50', '51', '52', '53', '54', '55', '57'};

cities={
'Ahvaz', 'Arak', 'Ardabil', 'Bandar Abbas', 'Birjand', ...
'Bojnord', 'Bushehr', 'Gorgan', 'Hamadan', 'Ilam', ...
'Isfahan', 'Karaj', 'Kerman', 'Kermanshah', 'Khorramabad', ...
'Mashhad', 'Qazvin', 'Qom', 'Rasht', 'Sanandaj', ...
'Sari', 'Semnan', 'Shahrekord', 'Shiraz', 'Tabriz', ...
'Tehran', 'Urmia', 'Yasuj', 'Yazd', 'Zahedan', ...
'Zanjan'};

items = unique([transactions{ : }]);
[F, S] = findFreqItemsets(transactions, minSup);
rules = generateRules(F, S, minConf);

```

5 Conclusion

In the case of cities, we believe that despite geographical differences, some features are similar in cities, and similar cities exist not only in one country but also globally. Finding these similar features will help urban experts in urban planning studies. These findings support decisions related to some urban activities such as providing appropriate strategies, traffic management, optimization of urban energy systems, and even the correct position of city offices, organizations, companies, etc. Today, the association rules learning is used in a variety of geographic applications, and unlike sequence mining, the order of the rules is not considered in both items and transactions.

City-data mining and finding urban dependencies using the Fariori algorithm identifies more rules of city similarities in the urban database and gives urban planners more choice. When the output conditions of the algorithm are: frequent = 6, support = 0.194, feature = '4', the identified cities are Arak, Hamadan, Isfahan, Kerman, Yasuj, and Zanjan. It means that 19.4% of the cities in our database, including these six cities, are at similar altitudes and are about 1501 to 2000 meters high (Feature 4). Under these conditions, one of the rules obtained is as follows:

$$\{4\} \rightarrow \{54\} \quad \text{conf}=0.5 \quad \text{lift}=1.73 \quad \text{support}=0.1$$

It means that these six cities, with 50% confidence, have an average annual precipitation of 301-600 mm (feature='54'). Thus, through the interpretation of each association rule, we discover the hidden similarities of cities.

When the computer does not have the storage space for Table 5, the Fariori method fails, and this is when the number of features or the number of columns is too large. But in the application of geography and the similarity of cities, where the features are usually not more than 1000, this method is more efficient than the Apriori method.

The preparation of urban databases, along with their timely publication, is our request to geographical institutions. It is better to have a global database of cities to make decisions. For example, knowing what aspects of city *X* in Iran are similar to city *Y* in Netherlands and what decisions are currently being made in city *Y* helps managers make similar decisions for city *X*. In this way, urban studies become more coherent and accurate.

Table 5 can be the basis for implementing genetic algorithms to minimize errors in the similarity of cities.

We will present this algorithm as a package in R software so that urban experts and decision-makers can use it easily.

Appendix A: geographical information

The geographical coordinates of the cities are shown in Table 13, and the Koppen climate classification (required by the article cities) is shown in Table 14.

[Table 13 about here.]

[Table 14 about here.]

References

1. (2020) I.R.of IRAN Meteorological Organization. <http://www.irimo.ir/eng/index.php>. Accessed 18 Oct 2020
2. (2020) Statistical Center of Iran > Iran Statistical Yearbook > Statistical-Yearbook-2016-2017
3. Agrawal R, Srikant R (1994) Fast Algorithms for Mining Association Rules in Large Databases. In: Proc. of the 20th International Conference on Very Large Data Bases (VLDB'94). pp 487–499
4. Efrat AR, Gernowo R, Farikhin (2020) Consumer purchase patterns based on market basket analysis using apriori algorithms. J Phys Conf Ser 1524:. <https://doi.org/10.1088/1742-6596/1524/1/012109>
5. Redhu S, Hegde RM (2020) Optimal relay node selection in time-varying IoT networks using apriori contact pattern information. Ad Hoc Networks 98:. <https://doi.org/10.1016/j.adhoc.2019.102065>
6. Santosh Kumar MB, Balakrishnan K (2019) Development of a model recommender system for agriculture using apriori algorithm. Adv Intell Syst Comput 768:153–163. https://doi.org/10.1007/978-981-13-0617-4_15
7. Sornalakshmi M, Balamurali S, Venkatesulu M, et al (2020) Hybrid method for mining rules based on enhanced Apriori algorithm with sequential minimal optimization in healthcare industry. Neural Comput Appl. <https://doi.org/10.1007/s00521-020-04862-2>
8. Appice A, Ceci M, Lanza A, et al (2003) Discovery of spatial association rules in geo-referenced census data: A relational mining approach. Intell Data Anal 7:541–566. <https://doi.org/10.3233/IDA-2003-7604>
9. Sengstock C, Gertz M (2010) Exploring volunteered geographic information using scale-dependent frequent pattern mining. In: Proceedings of GIScience
10. Kashian A, Richter K, Rajabifard A, et al (2016) Mining the co-existence of POIs in OpenStreetMap for faulty entry detection. In: CEUR Workshop Proceedings. p online
11. Ranjan R, Sharma A (2019) Evaluation of Frequent Itemset Mining Platforms using Apriori and FP- Growth Algorithm. Dep Comput Sci Eng DIT Univ 2:1–6
12. ur Rehman MH, Liew CS, Abbas A, et al (2016) Big data reduction methods: a survey. Data Sci Eng 1:265–284
13. Weiß CH (2015) Sampling in data mining. Wiley StatsRef Stat Ref Online. <https://doi.org/10.1002/9781118445112.stat04222.pub2>
14. Vasoya A, Koli N (2016) Mining of association rules on large database using distributed and parallel computing. Procedia Comput Sci 79:221–230. <https://doi.org/10.1016/j.procs.2016.03.029>
15. Versichele M, De Groote L, Bouuaert MC, et al (2014) Pattern mining in tourist attraction visits through association rule learning on Bluetooth tracking data: A case study of Ghent, Belgium. Tour Manag 44:67–81. <https://doi.org/10.1016/j.tourman.2014.02.009>
16. Goswami KR, Shah A (2014) Survey on Improving Frequent Pattern mining in Spatial Database. Int J Comput Sci Eng Technol 5:173–177
17. Kim YS, Yum BJ (2011) Recommender system based on click stream data using association rule mining. Expert Syst Appl 38:13320–13327. <https://doi.org/10.1016/j.eswa.2011.04.154>
18. Osadchiy T, Poliakov I, Olivier P, et al (2019) Recommender system based on pairwise

- association rules. Expert Syst Appl 115:535–542. <https://doi.org/10.1016/j.eswa.2018.07.077>
19. Cheng SC, Cheng YP (2019) An Adaptive Approach to Quantify Plant Features by Using Association Rule-Based Similarity. IEEE Access 7:32197–32205. <https://doi.org/10.1109/ACCESS.2019.2901968>
 20. Borozanov V, Hacks S, Silva N (2019) Using Machine Learning Techniques for Evaluating the Similarity of Enterprise Architecture Models: Technical Paper. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). pp 563–578
 21. Zhu S (2019) Research on data mining of education technical ability training for physical education students based on Apriori algorithm. Cluster Comput 22:14811–14818. <https://doi.org/10.1007/s10586-018-2420-8>
 22. Raj S, Ramesh D, Sreenu M, Sethi KK (2020) EAFIM: efficient apriori-based frequent itemset mining algorithm on Spark for big transactional data. Knowl Inf Syst 62:3565–3583. <https://doi.org/10.1007/s10115-020-01464-1>
 23. Liu X, Niu X, Fournier-Viger P (2020) Fast Top-K association rule mining using rule generation property pruning. Appl Intell. <https://doi.org/10.1007/s10489-020-01994-9>
 24. Song C (2016) Research of association rule algorithm based on data mining. In: 2016 IEEE International Conference on Big Data Analysis (ICBDA). pp 1–4
 25. Benouaret I, Amer-Yahia S, Roy SB, et al (2020) Enabling Decision Support Through Ranking and Summarization of Association Rules for TOTAL Customers. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). pp 160–193
 26. Bazaldua DL, Baker R, Pedro MO (2014) Comparing expert and metric-based assessments of association rule interestingness. In: Educational Data Mining 2014
 27. Lior R (2014) Data mining with decision trees: theory and applications. World scientific
 28. Sudarsanam, N., Kumar, N., Sharma, A., & Ravindran B (2020). R of change analysis for interestingness measures. Knowl Inf Syst 239–258
 29. Yuan Y, Huang T (2005) A matrix algorithm for mining association rules. In: International Conference on Intelligent Computing. pp 370–379
 30. Bhatia P (2019) Introduction to Data Mining. Pearson Education India
 31. (2020) Introduction to Market Basket Analysis» Loren on the Art of MATLAB. <https://blogs.mathworks.com/loren/2015/01/29/introduction-to-market-basket-analysis/#dd75d035-2129-4417-b848-b2725411df56>. Accessed 18 Oct 2020
 32. (2020) GitHub - toshiakit/apriori: Market Basket Analysis with MATLAB. <https://github.com/toshiakit/apriori>. Accessed 18 Oct 2020
 33. (2020) Apriori Association Rule Mining in MATLAB - Yarpiz. <https://yarpiz.com/89/ypml115-apriori>. Accessed 18 Oct 2020

Tables

Table 1 Capital cities of Iran's provinces and their features (features continue in Table 2)

	Cities	Province	Features		
			1	2	3
			Altitude (m)	City area (Km ²)	Population (people)
1	Ahvaz	Khuzestan	22.5	127.54	1184788
2	Arak	Markazi	1702.8	53.45	520944
3	Ardabil	Ardabil	1335.2	49.06	529374
4	Bandar Abbas	Hormozgan	9.8	43.69	526648
5	Birjand	South Khorasan	1491	26.44	203636
6	Bojnord	North Khorasan	1112	29.02	228931
7	Bushehr	Bushehr	6	31.18	223504
8	Gorgan	Golestan	0	44.65	350676
9	Hamadan	Hamadan	1740.8	72.52	554406
10	Ilam	Ilam	1337	27.73	194030
11	Isfahan	Isfahan	1550	158.24	1961260
12	Karaj	Alborz	1292.9	131.03	1615218
13	Kerman	Kerman	1754	62.39	537718
14	Kermanshah	Kermanshah	1318.5	95.97	946651
15	Khorramabad	Lorestan	1147.8	36.95	373416
16	Mashhad	Razavi Khorasan	999	208.60	2987323
17	Qazvin	Qazvin	1279.1	64.13	402748
18	Qom	Qom	879.1	112.38	1201158
19	Rasht	Gilan	24.8	54.84	679995
20	Sanandaj	Kurdistan	1373.4	37.61	412767
21	Sari	Mazandaran	23	34.75	309820
22	Semnan	Semnan	1127	28.00	185129
23	Shahrekord	Chaharmahal and Bakhtiari	2050	22.31	190441
24	Shiraz	Fars	1488	178.89	1565572
25	Tabriz	East Azerbaijan	1345	129.83	1558693
26	Tehran	Tehran	1191	450.93	8693706
27	Urmia	West Azerbaijan	1328	84.62	736224
28	Yasuj	Kohgiluyeh and Boyer-Ahmd	1816.3	21.95	134532
29	Yazd	Yazd	1230.2	81.88	529673
30	Zahedan	Sistan and Baluchestan	1370	63.93	587730
31	Zanjan	Zanjan	1659.4	63.93	430871

Table 2 Capital cities of Iran's provinces and their features (continued features from Table 1)

Cities		Features							
		4	5	6	7	8	9	10	11
		Köppen climate type	Average annual temperature (°C)	Maximum rainfall in one day (mm)	Number of annual freezing days	Number of annual dusty days	Number of annual sunny hours	Maximum annual wind speed (m/s)	Average annual precipitation (mm)
1	Ahvaz	BWh	25.5	23.0	1	91	3202.2	21	113.1
2	Arak	Dsa	14.0	46.5	91	84	3026.8	22	373.5
3	Ardabil	Dsa	09.4	16.2	111	13	2596.5	25	230.2
4	Bandar Abbas	BWh	27.0	18.5	0	122	3106.3	16	148.5
5	Birjand	BWk	16.5	18.6	64	43	3435.0	15	97.3
6	Bojnord	BSk	13.4	22.0	96	39	2825.4	19	275.8
7	Bushehr	BSh	25.1	18.1	0	43	3164.9	26	80.1
8	Gorgan	Csa	17.8	44.1	38	2	2369.0	24	491.1
9	Hamadan	BSk	11.5	26.8	125	24	3201.8	22	344
10	Ilam	Csa	16.9	53.1	56	90	3153.5	20	449.1
11	Isfahan	BSk	16.4	13.5	56	45	3477.8	19	71.1
12	Karaj	BSk	15.2	26.8	67	33	3109.6	16	211.1
13	Kerman	BWk	16.0	16.7	59	15	3331.4	21	98.3
14	Kermanshah	Csa	14.5	53.4	80	40	2993.4	15	457.1
15	Khorramabad	Csa	17.2	89.3	61	64	3142.3	18	562.5
16	Mashhad	BSk	14.4	40.9	62	10	3001.5	16	286.8
17	Qazvin	Csa	14.1	21.4	91	14	3011.2	22	297.1
18	Qom	BWk	18.4	14.2	64	64	3214.1	27	138.9
19	Rasht	Cfa	16.1	85.4	18	0	1672.6	20	1478.7
20	Sanandaj	Dsa	13.7	26.6	101	40	3173.0	13	317
21	Sari	Cfa	18.0	63.0	17	0	2016.0	22	824.4
22	Semnan	BWh	18.2	29.8	50	7	3226.0	17	127.7
23	Shahrekord	BSk	11.6	63.2	136	56	3282.8	26	317.4
24	Shiraz	BSk	17.8	49.5	27	54	3354.6	18	283.9
25	Tabriz	BSk	12.6	23.3	106	36	2818.5	17	263
26	Tehran	BSk	17.5	28.4	28	18	3110.1	32	193.6
27	Urmia	Csa	11.6	19.4	117	17	2891.2	20	286
28	Yasuj	Csa	15.2	72.4	49	44	3349.8	15	532.5
29	Yazd	BWk	19.4	06.2	24	78	3486.9	24	23.7
30	Zahedan	BWh	18.6	19.4	37	111	3371.5	19	55.1
31	Zanjan	BSk	11.2	18.6	101	20	2903.6	20	230

Table 3 Classification of urban features

1) Altitude (<i>m</i>):				
1=[0-500],	2=[501-1000],	3=[1001-1500],	4=[1501-2000],	5=[2001-2500]
2) city area (<i>km²</i>)				
6=[1-100],	7=[101-200],	8=[201-300],	9=[301-400],	10=[401-500]
3) Population (<i>people</i>):				
11=[1-2000000],	12=[2000001-4000000],	13=[4000001-6000000],		
	14=[6000001-8000000],	15=[8000001-10000000]		
4) Köppen climate type:				
	16=BSk,	17=BSk,	18=BWk,	
	19=BWh,	20=Cfa,	21=Csa,	22=Dsa
5) Average annual temperature (°C):				
23=[6-10],	24=[11-15],	25=[16-20],	26=[21-25],	27=[26-30]
6) Maximum rainfall in one day (<i>mm</i>):				
28=[1-20],	29=[21-40],	30=[41-60],	31=[61-80],	32=[81-100]
7) Number of frost days:				
33=[1-30],	34=[31-60],	35=[61-90]	36=[91-120],	37=[121-150]
8) Number of dusty days:				
38=[1-25],	39=[26-50],	40=[51-75],	41=[76-100],	42=[101-125]
9) Number of sunny days:				
43=[1651-2020],	44=[2021-2390],	45=[2391-2760],	46=[2761-3130],	47=[3131-3500]
10) Maximum wind speed (<i>m/s</i>):				
48=[10-14],	49=[15-19],	50=[20-24],	51=[25-29],	52=[30-34]
11) Average annual precipitation (<i>mm</i>):				
53=[1-300],	54=[301-600],	55=[601-900],	56=[901-1200],	57=[1201-1500]

Table 4 Inserting feature numbers in the database

	City	features										
		1	2	3	4	5	6	7	8	9	10	11
		Subfeatures										
		1-5	6-10	11-15	16-22	23-27	28-32	33-37	38-42	43-47	48-52	53-57
1	Ahvaz	1	7	11	19	27	29	33	41	47	50	53
2	Arak	4	6	11	22	24	28	36	41	46	50	54
3	Ardabil	3	6	11	22	23	28	36	38	45	51	53
4	Bandar Abbas	1	6	11	19	27	28	33	42	46	49	53
5	Birjand	3	6	11	18	25	28	35	39	47	49	53
6	Bojnord	3	6	11	16	24	29	36	39	46	49	53
7	Bushehr	1	6	11	17	27	28	33	39	47	51	53
8	Gorgan	1	6	11	21	25	30	34	38	44	50	54
9	Hamadan	4	6	11	16	24	29	37	38	47	50	54
10	Ilam	3	6	11	21	25	30	34	41	47	50	54
11	Isfahan	4	7	11	16	25	28	34	39	47	49	53
12	Karaj	3	7	11	16	25	29	35	39	46	49	53
13	Kerman	4	6	11	18	25	28	34	38	47	50	53
14	Kermanshah	3	6	11	21	25	30	35	39	46	49	54
15	Khorramabad	3	6	11	21	25	32	35	40	47	49	54
16	Mashhad	2	8	12	16	24	30	35	38	46	49	53
17	Qazvin	3	6	11	21	24	29	36	38	46	50	53
18	Qom	2	7	11	18	25	28	35	40	47	51	53
19	Rasht	1	6	11	20	25	32	33	38	43	50	57
20	Sanandaj	3	6	11	22	24	29	36	39	47	48	54
21	Sari	1	6	11	20	25	31	33	38	43	50	55
22	Semnan	3	6	11	19	25	29	34	38	47	49	53
23	Shahrekord	5	6	11	16	24	31	37	40	47	51	54
24	Shiraz	3	7	11	16	25	30	33	40	47	49	53
25	Tabriz	3	7	11	16	25	29	36	39	46	49	53
26	Tehran	3	10	15	16	25	29	33	38	46	52	53
27	Urmia	3	6	11	21	24	28	36	38	46	50	53
28	Yasuj	4	6	11	21	25	31	34	39	47	49	54
29	Yazd	3	6	11	18	25	28	33	41	47	50	53
30	Zahedan	3	6	11	19	25	28	34	42	47	49	53
31	Zanjan	4	6	11	16	24	28	36	38	46	50	53

Table 5 Binary database of urban features (*Continued on the next page*)

Cities		Features																		
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	Ahvaz	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1
2	Arak	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
3	Ardabil	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
4	Bandar Abbas	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1
5	Birjand	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0
6	Bojnord	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0
7	Bushehr	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0
8	Gorgan	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
9	Hamadan	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0
10	Ilam	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
11	Isfahan	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0
12	Karaj	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0
13	Kerman	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0
14	Kermanshah	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
15	Khorramabad	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
16	Mashhad	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
17	Qazvin	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
18	Qom	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0
19	Rasht	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
20	Sanandaj	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
21	Sari	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
22	Semnan	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1
23	Shahrekord	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	1	0	0	0
24	Shiraz	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0
25	Tabriz	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0
26	Tehran	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0
27	Urmia	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
28	Yasuj	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
29	Yazd	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0
30	Zahedan	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1
31	Zanjan	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0

Cities		Features																		
		20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
1	Ahvaz	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0
2	Arak	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0
3	Ardabil	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
4	Bandar Abbas	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0
5	Birjand	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0
6	Bojnord	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0
7	Bushehr	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0

Table 5 (Continued)

Cities		Features																		
		20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
8	Gorgan	0	1	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	1
9	Hamadan	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1
10	Ilam	0	1	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	1	0
11	Isfahan	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0
12	Karaj	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0
13	Kerman	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	1
14	Kermanshah	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0
15	Khorramabad	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0
16	Mashhad	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	1
17	Qazvin	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1
18	Qom	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0
19	Rasht	1	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	1
20	Sanandaj	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0
21	Sari	1	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1
22	Semnan	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	1
23	Shahrekord	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0
24	Shiraz	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0
25	Tabriz	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	0
26	Tehran	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	1
27	Urmia	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	1
28	Yasuj	0	1	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0
29	Yazd	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0
30	Zahedan	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0
31	Zanjan	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1

Cities		Features																		
		39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57
1	Ahvaz	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0
2	Arak	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
3	Ardabil	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0
4	Bandar Abbas	0	0	0	1	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0
5	Birjand	1	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0
6	Bojnord	1	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0
7	Bushehr	1	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0
8	Gorgan	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0
9	Hamadan	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0
10	Ilam	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0
11	Isfahan	1	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0
12	Karaj	1	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0
13	Kerman	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0
14	Kermanshah	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0

Table 5 (Continued)

Cities		Features																			
		39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	
15	Khorramabad	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	1	0		0	
16	Mashhad	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0		0	
17	Qazvin	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0		0	
18	Qom	0	1	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0		0	
19	Rasht	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0		0	
20	Sanandaj	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0		0	
21	Sari	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1		0	
22	Semnan	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0		0	
23	Shahrekord	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0		0	
24	Shiraz	0	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0		0	
25	Tabriz	1	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0		0	
26	Tehran	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0		0	
27	Urmia	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0		0	
28	Yasuj	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1	0		0	
29	Yazd	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0		0	
30	Zahedan	0	0	0	1	0	0	0	0	1	0	1	0	0	0	1	0	0		0	
31	Zanjan	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0		0	

Table 6 Transactional database

TID	List of items				
100	A	B	C	D	
200	A	B	C	D	F
300	B	C	D	E	
400	C	B	A	D	F
500	A	B	E	F	

Table 7 Apriori Algorithm

<table><tr><th colspan="2">C_1</th></tr><tr><th>Items</th><th>Support</th></tr><tr><td>{A}</td><td>4</td></tr><tr><td>{B}</td><td>5</td></tr><tr><td>{C}</td><td>4</td></tr><tr><td>{D}</td><td>4</td></tr><tr><td>{E}</td><td>2</td></tr><tr><td>{F}</td><td>3</td></tr></table>	C_1		Items	Support	{A}	4	{B}	5	{C}	4	{D}	4	{E}	2	{F}	3	<table><tr><th colspan="2">L_1</th></tr><tr><th>Items</th><th>Support</th></tr><tr><td>{A}</td><td>4</td></tr><tr><td>{B}</td><td>5</td></tr><tr><td>{C}</td><td>4</td></tr><tr><td>{D}</td><td>4</td></tr><tr><td>{F}</td><td>3</td></tr></table>	L_1		Items	Support	{A}	4	{B}	5	{C}	4	{D}	4	{F}	3	<table><tr><th colspan="2">C_2</th></tr><tr><th>Items</th><th>Support</th></tr><tr><td>{A, B}</td><td>4</td></tr><tr><td>{A, C}</td><td>3</td></tr><tr><td>{A, D}</td><td>3</td></tr><tr><td>{A, F}</td><td>3</td></tr><tr><td>{B, C}</td><td>4</td></tr><tr><td>{B, D}</td><td>4</td></tr><tr><td>{B, F}</td><td>3</td></tr><tr><td>{C, D}</td><td>4</td></tr><tr><td>{C, F}</td><td>2</td></tr><tr><td>{D, F}</td><td>2</td></tr></table>	C_2		Items	Support	{A, B}	4	{A, C}	3	{A, D}	3	{A, F}	3	{B, C}	4	{B, D}	4	{B, F}	3	{C, D}	4	{C, F}	2	{D, F}	2	<table><tr><th colspan="2">L_2</th></tr><tr><th>Items</th><th>Support</th></tr><tr><td>{A, B}</td><td>4</td></tr><tr><td>{A, C}</td><td>3</td></tr><tr><td>{A, D}</td><td>3</td></tr><tr><td>{A, F}</td><td>3</td></tr><tr><td>{B, C}</td><td>4</td></tr><tr><td>{B, D}</td><td>4</td></tr><tr><td>{B, F}</td><td>3</td></tr><tr><td>{C, D}</td><td>4</td></tr></table>	L_2		Items	Support	{A, B}	4	{A, C}	3	{A, D}	3	{A, F}	3	{B, C}	4	{B, D}	4	{B, F}	3	{C, D}	4
C_1																																																																													
Items	Support																																																																												
{A}	4																																																																												
{B}	5																																																																												
{C}	4																																																																												
{D}	4																																																																												
{E}	2																																																																												
{F}	3																																																																												
L_1																																																																													
Items	Support																																																																												
{A}	4																																																																												
{B}	5																																																																												
{C}	4																																																																												
{D}	4																																																																												
{F}	3																																																																												
C_2																																																																													
Items	Support																																																																												
{A, B}	4																																																																												
{A, C}	3																																																																												
{A, D}	3																																																																												
{A, F}	3																																																																												
{B, C}	4																																																																												
{B, D}	4																																																																												
{B, F}	3																																																																												
{C, D}	4																																																																												
{C, F}	2																																																																												
{D, F}	2																																																																												
L_2																																																																													
Items	Support																																																																												
{A, B}	4																																																																												
{A, C}	3																																																																												
{A, D}	3																																																																												
{A, F}	3																																																																												
{B, C}	4																																																																												
{B, D}	4																																																																												
{B, F}	3																																																																												
{C, D}	4																																																																												
<table><tr><th colspan="2">C_3</th></tr><tr><th>Items</th><th>Support</th></tr><tr><td>{A, B, C}</td><td>3</td></tr><tr><td>{A, B, D}</td><td>3</td></tr><tr><td>{A, B, F}</td><td>3</td></tr><tr><td>{A, C, D}</td><td>3</td></tr><tr><td>{B, C, D}</td><td>4</td></tr></table>	C_3		Items	Support	{A, B, C}	3	{A, B, D}	3	{A, B, F}	3	{A, C, D}	3	{B, C, D}	4	<table><tr><th colspan="2">L_3</th></tr><tr><th>Items</th><th>Support</th></tr><tr><td>{A, B, C}</td><td>3</td></tr><tr><td>{A, B, D}</td><td>3</td></tr><tr><td>{A, B, F}</td><td>3</td></tr><tr><td>{A, C, D}</td><td>3</td></tr><tr><td>{B, C, D}</td><td>4</td></tr></table>	L_3		Items	Support	{A, B, C}	3	{A, B, D}	3	{A, B, F}	3	{A, C, D}	3	{B, C, D}	4	<table><tr><th colspan="2">C_4</th></tr><tr><th>Items</th><th>Support</th></tr><tr><td>{A, B, C, D}</td><td>3</td></tr></table>	C_4		Items	Support	{A, B, C, D}	3	<table><tr><th colspan="2">L_4</th></tr><tr><th>Items</th><th>Support</th></tr><tr><td>{A, B, C, D}</td><td>3</td></tr></table>	L_4		Items	Support	{A, B, C, D}	3																																		
C_3																																																																													
Items	Support																																																																												
{A, B, C}	3																																																																												
{A, B, D}	3																																																																												
{A, B, F}	3																																																																												
{A, C, D}	3																																																																												
{B, C, D}	4																																																																												
L_3																																																																													
Items	Support																																																																												
{A, B, C}	3																																																																												
{A, B, D}	3																																																																												
{A, B, F}	3																																																																												
{A, C, D}	3																																																																												
{B, C, D}	4																																																																												
C_4																																																																													
Items	Support																																																																												
{A, B, C, D}	3																																																																												
L_4																																																																													
Items	Support																																																																												
{A, B, C, D}	3																																																																												

Table 8 Sorted database based on items

TID	Items					
100	A	B	C	D		
200	A	B	C	D		F
300		B	C	D	E	
400	A	B	C	D		F
500	A	B			E	F
Support	4	5	4	4	2	3

Table 9 Sorted database based on support

TID	Items					
100	B	A	C	D		
200	B	A	C	D	F	
300	B		C	D		E
400	B	A	C	D	F	
500	B	A			F	E
Support	5	4	4	4	3	2

Table 10 Fariori algorithm (second suggestion)

	Step 1 Items: B, A	Step 2 Items: B, A, C	Step 3 Items: B, A, C, D	Step 4 Items: B, A, C, D, F																												
Double Item-sets	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A}</td><td>4</td></tr></table>	Items	Support	{B, A}	4	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, C}</td><td>4</td></tr><tr><td>{A, C}</td><td>3</td></tr></table>	Items	Support	{B, C}	4	{A, C}	3	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, D}</td><td>4</td></tr><tr><td>{A, D}</td><td>3</td></tr><tr><td>{C, D}</td><td>4</td></tr></table>	Items	Support	{B, D}	4	{A, D}	3	{C, D}	4	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, F}</td><td>3</td></tr><tr><td>{A, F}</td><td>3</td></tr><tr><td>{C, F}</td><td>3</td></tr><tr><td>{D, F}</td><td>3</td></tr></table>	Items	Support	{B, F}	3	{A, F}	3	{C, F}	3	{D, F}	3
Items	Support																															
{B, A}	4																															
Items	Support																															
{B, C}	4																															
{A, C}	3																															
Items	Support																															
{B, D}	4																															
{A, D}	3																															
{C, D}	4																															
Items	Support																															
{B, F}	3																															
{A, F}	3																															
{C, F}	3																															
{D, F}	3																															
Triple Item-sets		<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A, C}</td><td>3</td></tr></table>	Items	Support	{B, A, C}	3	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A, D}</td><td>3</td></tr><tr><td>{B, C, D}</td><td>4</td></tr><tr><td>{A, C, D}</td><td>3</td></tr></table>	Items	Support	{B, A, D}	3	{B, C, D}	4	{A, C, D}	3	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A, F}</td><td>3</td></tr></table>	Items	Support	{B, A, F}	3												
Items	Support																															
{B, A, C}	3																															
Items	Support																															
{B, A, D}	3																															
{B, C, D}	4																															
{A, C, D}	3																															
Items	Support																															
{B, A, F}	3																															
Four Item-sets			<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A, C, D}</td><td>3</td></tr></table>	Items	Support	{B, A, C, D}	3																									
Items	Support																															
{B, A, C, D}	3																															

Table 11 Fariori algorithm (third suggestion)

	Step 1 Items: B, A, C, D	Step 2 Items: B, A, C, D, F																								
Double item-sets	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A}</td><td>4</td></tr><tr><td>{B, C}</td><td>4</td></tr><tr><td>{B, D}</td><td>4</td></tr><tr><td>{A, C}</td><td>3</td></tr><tr><td>{A, D}</td><td>3</td></tr><tr><td>{C, D}</td><td>4</td></tr></table>	Items	Support	{B, A}	4	{B, C}	4	{B, D}	4	{A, C}	3	{A, D}	3	{C, D}	4	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, F}</td><td>3</td></tr><tr><td>{A, F}</td><td>3</td></tr><tr><td>{C, F}</td><td>2</td></tr><tr><td>{D, F}</td><td>2</td></tr></table>	Items	Support	{B, F}	3	{A, F}	3	{C, F}	2	{D, F}	2
Items	Support																									
{B, A}	4																									
{B, C}	4																									
{B, D}	4																									
{A, C}	3																									
{A, D}	3																									
{C, D}	4																									
Items	Support																									
{B, F}	3																									
{A, F}	3																									
{C, F}	2																									
{D, F}	2																									
Triple item-sets	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A, C}</td><td>3</td></tr><tr><td>{B, A, D}</td><td>3</td></tr><tr><td>{B, C, D}</td><td>4</td></tr><tr><td>{A, C, D}</td><td>3</td></tr></table>	Items	Support	{B, A, C}	3	{B, A, D}	3	{B, C, D}	4	{A, C, D}	3	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A, F}</td><td>3</td></tr></table>	Items	Support	{B, A, F}	3										
Items	Support																									
{B, A, C}	3																									
{B, A, D}	3																									
{B, C, D}	4																									
{A, C, D}	3																									
Items	Support																									
{B, A, F}	3																									
Four item-sets	<table><tr><th>Items</th><th>Support</th></tr><tr><td>{B, A, C, D}</td><td>3</td></tr></table>	Items	Support	{B, A, C, D}	3																					
Items	Support																									
{B, A, C, D}	3																									

Table 12 Comparison of the Fariori algorithm with software that implements the Apriori algorithm (R, Weka, MBA, and Yarpiz)

Min-Conf=0.6		Min-Supp							
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
R	Number of rules	1607	204	59	7	6	2	2	0
	Execution time (s)	0.0210	0.0170	0.0251	0.0160	0.0220	0.0155	0.0154	0.0200
Weka	Number of rules	4582	275	102	28	6	2	2	0
MBA*	Number of rules	2639	264	69	6	5	0	0	0
	Execution time (s)	5.4689	1.0811	0.7023	0.5435	0.5940	0.4854	0.4597	1.1442
Yarpiz	Number of rules	-	-	-	5	4	2	2	0
	Execution time (s)	-	-	-	0.0560	0.0505	0.0324	0.0334	0.0916
Fariori	Number of rules	16074	532	73	7	6	2	2	0
	Execution time (s)	1.4207	0.0498	0.0186	0.0082	0.0132	0.0078	0.0070	0.0116

* Market Basket Analysis method

Table 13 Geographical coordinates of Iranian cities

Cities of Iran	Latitude ($^{\circ}N$)	Longitude ($^{\circ}E$)
Ahvaz	31° 20' 41"	48° 44' 39"
Arak	34° 04' 19"	49° 47' 00"
Ardabil	38° 13' 05"	48° 19' 43"
Bandar Abbas	27° 12' 53"	56° 22' 22"
Birjand	32° 53' 26"	59° 17' 01"
Bojnord	37° 29' 14"	57° 18' 12"
Bushehr	28° 54' 14"	50° 49' 16"
Gorgan	36° 54' 18"	54° 24' 49"
Hamadan	34° 52' 10"	48° 32' 05"
Ilam	33° 35' 16"	46° 23' 52"
Isfahan	32° 31' 03"	51° 42' 22"
Karaj	35° 48' 25"	50° 57' 14"
Kerman	30° 15' 22"	56° 57' 45"
Kermanshah	34° 21' 08"	47° 09' 12"
Khorramabad	33° 26' 20"	48° 17' 01"
Mashhad	36° 14' 11"	59° 37' 52"
Qazvin	36° 15' 42"	50° 03' 41"
Qom	34° 46' 29"	53° 51' 19"
Rasht	37° 19' 21"	49° 37' 27"
Sanandaj	35° 15' 16"	47° 00' 53"
Sari	36° 32' 12"	52° 59' 12"
Semnan	35° 35' 18"	53° 25' 17"
Shahrekord	28° 54' 14"	50° 50' 22"
Shiraz	29° 33' 41"	52° 36' 09"
Tabriz	38° 07' 20"	46° 14' 33"
Tehran	35° 41' 36"	51° 18' 34"
Urmia	37° 39' 31"	45° 03' 19"
Yasuj	30° 41' 55"	51° 33' 18"
Yazd	36° 39' 37"	48° 31' 23"
Zahedan	29° 28' 20"	60° 54' 01"
Zanjan	36° 39' 37"	48° 31' 18"

Table 14 Description of Köppen climate symbols

Symbol	Description	Criteria*
BWh	Arid	$B : MAP < 10 \times P_{threshold}$
	Desert	$W : MAP < 5 \times P_{threshold}$
	Hot	$h : MAT \geq 18$
BWk	Arid	$B : MAP < 10 \times P_{threshold}$
	Desert	$W : MAP < 5 \times P_{threshold}$
	Cold	$k : MAT < 18$
BSh	Arid	$B : MAP < 10 \times P_{threshold}$
	Steppe	$S : MAP \geq 5 \times P_{threshold}$
	Hot	$h : MAT \geq 18$
BSk	Arid	$B : MAP < 10 \times P_{threshold}$
	Steppe	$S : MAP \geq 5 \times P_{threshold}$
	Cold	$k : MAT < 18$
Csa	Temperate	$C : T_{hot} > 10 \text{ \& } 0 < T_{cold} < 18$
	Dry Summer	$s : P_{sdry} < 40 \text{ \& } P_{sdry} < P_{wwet}/3$
	Hot Summer	$a : T_{hot} \geq 22$
Cwa	Temperate	$C : T_{hot} > 10 \text{ \& } 0 < T_{cold} < 18$
	Dry Winter	$w : P_{wdry} < P_{swet}/10$
	Hot summer	$a : T_{hot} \geq 22$
Cfa	Temperate	$C : T_{hot} > 10 \text{ \& } 0 < T_{cold} < 18$
	Without dry season	$f : \text{Not (Cs) or (Cw)}$
	Hot Summer	$a : T_{hot} \geq 22$
Dsa	Cold	$D : T_{hot} > 10 \text{ \& } T_{cold} \leq 0$
	Dry Summer	$s : P_{sdry} < 40 \text{ \& } P_{sdry} < P_{wwet}/3$
	Hot Summer	$a : T_{hot} \geq 22$

* MAP = mean annual precipitation

MAT = mean annual temperature

 T_{hot} = temperature of the hottest month T_{cold} = temperature of the coldest month P_{sdry} = precipitation of the driest month in summer P_{wdry} = precipitation of the driest month in winter P_{swet} = precipitation of the wettest month in summer P_{wwet} = precipitation of the wettest month in winter $P_{threshold}$ = varies according to the following rules(if 70% of MAP occurs in winter then $P_{threshold} = 2 \times MAT$,if 70% of MAP occurs in summer then $P_{threshold} = 2 \times MAT + 28$ otherwise $P_{threshold} = 2 \times MAT + 14$).

Figures

Fig. 1 31 provinces of Iran (Tavakol, Amin, Luus-Powell, & Halajian, 2015)

Fig. 2 Space complexity for Apriori and Fariori algorithms

```

Command Window
41- {6 47 }=>{11 } Conf=1 Lift=1.069 Support=0.3871
42- {28 }=>{11 6 } Conf=0.83333 Lift=1.1232 Support=0.32258
43- {11 28 }=>{6 } Conf=0.83333 Lift=1.1232 Support=0.32258
44- {6 28 }=>{11 } Conf=1 Lift=1.069 Support=0.32258
45- {38 }=>{11 6 } Conf=0.83333 Lift=1.1232 Support=0.32258
46- {11 38 }=>{6 } Conf=1 Lift=1.3478 Support=0.32258
47- {6 38 }=>{11 } Conf=1 Lift=1.069 Support=0.32258
48- {50 }=>{11 6 } Conf=0.91667 Lift=1.2355 Support=0.35484
49- {11 50 }=>{6 } Conf=0.91667 Lift=1.2355 Support=0.35484
50- {6 50 }=>{11 } Conf=1 Lift=1.069 Support=0.35484
51- {53 25 }=>{11 } Conf=0.90909 Lift=0.97179 Support=0.32258
52- {3 }=>{11 53 } Conf=0.6875 Lift=1.184 Support=0.35484
53- {11 53 }=>{3 } Conf=0.61111 Lift=1.184 Support=0.35484
54- {11 3 }=>{53 } Conf=0.73333 Lift=1.1367 Support=0.35484
55- {53 3 }=>{11 } Conf=0.91667 Lift=0.97989 Support=0.35484
56- {47 }=>{11 53 } Conf=0.625 Lift=1.0764 Support=0.32258
57- {11 47 }=>{53 } Conf=0.625 Lift=0.96875 Support=0.32258
58- {53 47 }=>{11 } Conf=1 Lift=1.069 Support=0.32258
59- {28 }=>{11 53 } Conf=0.91667 Lift=1.5787 Support=0.35484
60- {11 53 }=>{28 } Conf=0.61111 Lift=1.5787 Support=0.35484
61- {11 28 }=>{53 } Conf=0.91667 Lift=1.4208 Support=0.35484
62- {53 28 }=>{11 } Conf=1 Lift=1.069 Support=0.35484
63- {3 }=>{11 25 } Conf=0.625 Lift=1.1397 Support=0.32258
64- {11 3 }=>{25 } Conf=0.66667 Lift=1.1481 Support=0.32258
65- {25 3 }=>{11 } Conf=0.90909 Lift=0.97179 Support=0.32258
66- {25 }=>{11 47 } Conf=0.61111 Lift=1.184 Support=0.35484
67- {47 }=>{11 25 } Conf=0.6875 Lift=1.2537 Support=0.35484
68- {11 25 }=>{47 } Conf=0.64706 Lift=1.2537 Support=0.35484
69- {11 47 }=>{25 } Conf=0.6875 Lift=1.184 Support=0.35484
70- {25 47 }=>{11 } Conf=1 Lift=1.069 Support=0.35484
71- {49 }=>{11 25 } Conf=0.76923 Lift=1.4027 Support=0.32258
72- {11 49 }=>{25 } Conf=0.83333 Lift=1.4352 Support=0.32258
73- {25 49 }=>{11 } Conf=1 Lift=1.069 Support=0.32258

-----
minsup= 0.3
minconf= 0.6
Frequent Itemsets=50
fx >>

```

Fig. 3 Results of the Fariori algorithm for city database (implemented in MATLAB)

The image shows a screenshot of a 'Command Window' from a software application. It displays two association rules, numbered 49 and 50. Each rule lists a set of city features in curly braces, followed by its frequent itemset count, support value, and the features it includes. Rule 49 has a frequent itemset of 11, a support of 0.35484, and features 11, 25, and 47. Rule 50 has a frequent itemset of 10, a support of 0.32258, and features 11, 25, and 49. The cities listed are Birjand, Ilam, Isfahan, Kerman, Khorramabad, Qom, Semnan, Shiraz, Yasuj, Yazd, and Zahedan. The window has a standard scrollbar on the right and a status bar at the bottom.

```
Command Window
49- Frequent=11 Support=0.35484 Features=11 25 47
{5-Birjand}
{10-Ilam}
{11-Isfahan}
{13-Kerman}
{15-Khorramabad}
{18-Qom}
{22-Semnan}
{24-Shiraz}
{28-Yasuj}
{29-Yazd}
{30-Zahedan}
50- Frequent=10 Support=0.32258 Features=11 25 49
{5-Birjand}
{11-Isfahan}
{12-Karaj}
{14-Kermanshah}
{15-Khorramabad}
{22-Semnan}
{24-Shiraz}
{25-Tabriz}
{28-Yasuj}
{30-Zahedan}
```

Fig. 4 Separation of cities for association rules in the Fariori algorithm

Figures

Figure 1

31 provinces of Iran (Tavakol, Amin, Luus-Powell, & Halajian, 2015)

Figure 2

Space complexity for Apriori and Fariori algorithms

```
Command Window

41- {6 47 }=>{11 } Conf=1 Lift=1.069 Support=0.3871
42- {28 }=>{11 6 } Conf=0.83333 Lift=1.1232 Support=0.32258
43- {11 28 }=>{6 } Conf=0.83333 Lift=1.1232 Support=0.32258
44- {6 28 }=>{11 } Conf=1 Lift=1.069 Support=0.32258
45- {38 }=>{11 6 } Conf=0.83333 Lift=1.1232 Support=0.32258
46- {11 38 }=>{6 } Conf=1 Lift=1.3478 Support=0.32258
47- {6 38 }=>{11 } Conf=1 Lift=1.069 Support=0.32258
48- {50 }=>{11 6 } Conf=0.91667 Lift=1.2355 Support=0.35484
49- {11 50 }=>{6 } Conf=0.91667 Lift=1.2355 Support=0.35484
50- {6 50 }=>{11 } Conf=1 Lift=1.069 Support=0.35484
51- {53 25 }=>{11 } Conf=0.90909 Lift=0.97179 Support=0.32258
52- {3 }=>{11 53 } Conf=0.6875 Lift=1.184 Support=0.35484
53- {11 53 }=>{3 } Conf=0.61111 Lift=1.184 Support=0.35484
54- {11 3 }=>{53 } Conf=0.73333 Lift=1.1367 Support=0.35484
55- {53 3 }=>{11 } Conf=0.91667 Lift=0.97989 Support=0.35484
56- {47 }=>{11 53 } Conf=0.625 Lift=1.0764 Support=0.32258
57- {11 47 }=>{53 } Conf=0.625 Lift=0.96875 Support=0.32258
58- {53 47 }=>{11 } Conf=1 Lift=1.069 Support=0.32258
59- {28 }=>{11 53 } Conf=0.91667 Lift=1.5787 Support=0.35484
60- {11 53 }=>{28 } Conf=0.61111 Lift=1.5787 Support=0.35484
61- {11 28 }=>{53 } Conf=0.91667 Lift=1.4208 Support=0.35484
62- {53 28 }=>{11 } Conf=1 Lift=1.069 Support=0.35484
63- {3 }=>{11 25 } Conf=0.625 Lift=1.1397 Support=0.32258
64- {11 3 }=>{25 } Conf=0.66667 Lift=1.1481 Support=0.32258
65- {25 3 }=>{11 } Conf=0.90909 Lift=0.97179 Support=0.32258
66- {25 }=>{11 47 } Conf=0.61111 Lift=1.184 Support=0.35484
67- {47 }=>{11 25 } Conf=0.6875 Lift=1.2537 Support=0.35484
68- {11 25 }=>{47 } Conf=0.64706 Lift=1.2537 Support=0.35484
69- {11 47 }=>{25 } Conf=0.6875 Lift=1.184 Support=0.35484
70- {25 47 }=>{11 } Conf=1 Lift=1.069 Support=0.35484
71- {49 }=>{11 25 } Conf=0.76923 Lift=1.4027 Support=0.32258
72- {11 49 }=>{25 } Conf=0.83333 Lift=1.4352 Support=0.32258
73- {25 49 }=>{11 } Conf=1 Lift=1.069 Support=0.32258

-----
minsup= 0.3
minconf= 0.6
Frequent Itemsets=50
fx >>
```

Figure 3

Results of the Fariori algorithm for city database (implemented in MATLAB)

```
Command Window
49- Frequent=11 Support=0.35484 Features=11 25 47
    {5-Birjand}
    {10-Ilam}
    {11-Isfahan}
    {13-Kerman}
    {15-Khorramabad}
    {18-Qom}
    {22-Semnan}
    {24-Shiraz}
    {28-Yasuj}
    {29-Yazd}
    {30-Zahedan}
50- Frequent=10 Support=0.32258 Features=11 25 49
    {5-Birjand}
    {11-Isfahan}
    {12-Karaj}
    {14-Kermanshah}
    {15-Khorramabad}
    {22-Semnan}
    {24-Shiraz}
    {25-Tabriz}
    {28-Yasuj}
    {30-Zahedan}
fx ----- Association Rules -----
< >
```

Figure 4

Separation of cities for association rules in the Fariori algorithm