Applied Computer Science, vol. 15, no. 2, pp. 73–88
doi:10.23743/acs-2019-15

Submitted: 2019-04-30
Revised: 2019-06-14
Accepted: 2019-06-25

Data Mining, association rules,
apriori algorithm, combos, WebService

Rosa Maria VAZQUEZ[*], Edmundo BONILLA[*],
Eduardo SANCHEZ[*], Oscar ATRIANO[**], Cinthya BERRUECOS[**]

APPLICATION OF DATA MINING TECHNIQUES TO FIND RELATIONSHIPS BETWEEN THE DISHES OFFERED BY A RESTAURANT FOR THE ELABORATION OF COMBOS BASED ON THE PREFERENCES OF THE DINERS

Abstract

Currently, blended food has been a common menu item in fast food restaurants. The sales of the fast-food industry grow thanks to several sales strategies, including the "combos", so, specialty, regional, family and buffet restaurants are even joining combos' promotions. This research paper presents the implementation of a system that will serve as support to elaborate combos according to the preferences of the diners using data mining techniques to find relationships between the different dishes that are offered in a restaurant. The software resulting from this research is being used by the mobile application Food Express, with which it communicates through webservices.

[*] Tecnológico Nacional de México, Campus Apizaco, Computer Systems Department, Fco I Madero, Barrio de San José, 90300 Apizaco, Tlaxcala, México, rvazlean@hotmail.com, edbonn@hotmail.com, esanlu@hotmail.com
[**] Smartsoft America BA, Calle Adolfo López Mateos, Texcacoac, 90806 Chiautempan, México, oatriano@smartsoftamerica.com.mx, cberruecos@smartsoftamerica.com.mx

1. INTRODUCTION

The restaurant industry is a sector of great importance for economic activity, and undoubtedly, it is a factor of economic growth in many regions. The main objective in the field of food and beverages is to generate high income and have an effective management of costs. The menu is, among other factors, a key element that influences the success or failure of a restaurant.

Nowadays, blended food has been a common menu item in fast food restaurants. Food combos are a selling strategy used by restaurants to obtain higher profits and sell more products, however, most of the combos offered are formed according to the criterion of the restaurant, taking into account psychology, marketing and design to attract the attention of customers. On the other hand, on many occasions, the same customer is the one who creates his own combo based on the menu to create his own combination of meals. These last combos are the ones that take very strong interest in this research project as, if the restaurant offers combos based on the preference of the diners, you will have the security of knowing which combos are more likely to be sold and that are of the satisfaction of the diner.

The software resulting from this research work is in operation, since it is part of the Food Express application, whose objective is to order food at home, reserve or order to take away. The application is developed for the iOS and Android platforms as well as a web system (www.foodexpress.com.mx). From this application the preferences of the diners are obtained through the use of web-services to which the client has access.

The main objective of this research work is to implement a system that serves as support to assemble combos according to the preferences of the diners, which will be achieved with the historical data that will be obtained from the knowledge base that stores the food orders of the restaurants, likewise, it is intended to support the analysis of the menu and determine which are the most profitable dishes for the restaurant and the preference of the diner, what will allow to assemble the combos that could be of the diners' preference and offer them as an alternative of election. On the other hand, the system will offer to provide the diner with a list of suggested dishes that have been sold together with a dish that has been chosen by the diner.

2. LITERATURE REVIEW

This section describes some research work about association rules to classify, discover and highlight acts that occur in common within a data set and observe the relationship between them.

Some research works use the Apriori algorithm to find association rules in databases, such is the case of the work carried out by Park S. & Park Y. (2018) in which he uses the Apriori algorithm to analyze the relationship between math scores and problem solving patterns of students through the data of students' math tests by searching for association rules. On the other hand, the work of Zulfikar, Wahana, Uriawan & Lukman (2016) was carried out with the objective of helping the XMART company of Indonesia, to elaborate sales strategies. The author uses the Apriori algorithm, with values of support and confidence between the products in order to analyze the high frequency pattern and determine the association rules based on the combination of products and thus be able to show the promotions that the company can make to increase the product sales.

Other works propose making combinations of the Apriori algorithm with other algorithms to improve and optimize the results obtained. Huang, Lin & Li (2018) evaluate the use of a method called Apriori-BM applied to the conventional Apriori algorithm, which finds the association rules in the calculation process early, so it does not require repeated scanning of the database as the conventional Apriori algorithm does. In the research work carried out by Harikumar & Dilipkumar (2016) it is proposed a variant of the Apriori algorithm that uses the concept of QR decomposition to reduce the dimensions of the databases, thus reducing the complexity of the traditional Apriori algorithm. Pei (2013) proposes the application of the Apriori algorithm, improving it with the Eclat algorithm, performing operations more quickly and efficiently. In the same way, the research work of Zheng (2007) proposes the implementation of the Apriori algorithm by complementing it with the CBA-CB algorithm (Classification Based on Associations-Classifier Builder), which is applied on the rules generated by the Apriori algorithm, and thus be able to filter the most significant.

Other researchers propose other techniques for the extraction of association rules, such is the case of the work carried out by Tom & Annaraud (2017) which is mainly focused on the restaurant sector and specifically on the menu analysis. The purpose of this study is to apply fuzzy multi-criteria decision making (FMCDM) techniques to restaurant menu engineering by utilizing menu items' popularity and contribution margin. This study uses the model proposed by Kasavana and Smith where each menu item is classified into one of four segments created by a two-by-two matrix of high and low popularity, and above and below average contribution margin. Finally, Kabir, Xu, Kang & Zhao (2015) presents an evolutionary approach to find sets of frequent maximum elements of large databases by using the principles of Genetic Algorithm (GA).

3. METHODOLOGY

The development of this research work is mainly based on the implementation of the Apriori algorithm for the extraction of association rules to develop a system that allows the elaboration of combos and provide suggestions of dishes according to the preferences of the diners, serving as support to the Food Express application.

3.1. Association rules

Association rules are used to discover facts that occur in common within a given set of data. Based on the concept of strong rules Agrawal and Srikant (1994) introduced association rules to discover regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. This information can be used as a basis to make decisions about marketing activities. Following the definition of Agrawal and Srikant (1994), an association rule is defined as:

Let I = $\{i_1, i_2, ..., i_n\}$ be a set of n binary attributes called *items*. Let D = $\{t_1,t_2,...,t_n\}$ be a set of transactions called the *database*. Each transaction in **D** has a unique transaction **ID** and contains a subset of the items in **I**. A rule is defined as implication of the form $\mathbf{X \rightarrow Y}$ where $\mathbf{X}$, $\mathbf{Y \subseteq I}$ and $\mathbf{X \cap Y = \emptyset}$**.** The set of items (for short *itemsets*) X and Y are called antecedent (left-hand-side or LHS) and consequent (right-hand-side or RHS) of the rule respectively.

To limit the number of rules obtained, metrics are used to measure the importance or interest of a rule, these being the following:

Support. The support **supp(X)** of an itemset **X** is defined as the proportion of transactions in the data set which contains the itemset.

$$Supp(X) = \frac{NoTransactions\_Itemset\_X}{Total\_No\_Transactions} \tag{1}$$

Given a rule "If A => B", the support of this rule is defined as the number of times or the (relative) frequency with which A and B appear together in a transaction database. Support can be defined for individual items, however, it can also be defined for the rule. A first criterion that can be imposed to limit the number of rules is that they comply with minimal support.

Confidence. The confidence of a rule is defined:

$$Conf(X \rightarrow Y) = \frac{Supp(X \cup Y)}{Supp(X)} \tag{2}$$

Given a rule "If A => B", the confidence of this rule is given by the quotient of the support of the rule and the support of the antecedent only. If the support measures frequency, confidence measures the strength of the rule. In language of probability, confidence is a strength of the rule.

Lift. The lift of a rule is defined as:

$$Lift(X \to Y) = \frac{Supp(X \cup Y)}{Supp(X) * Supp(Y)} \tag{3}$$

Given a rule "If A => B", the lift of this rule is given by the quotient of the support of the rule and the product of the individual supports. If the Lift = 1 or very close to 1, it indicates that the relationship is very close to having been the product of chance. If the Lift > 1, it indicates a really strong relationship (controlling for the frequency with which both occur), that is, it indicates that the relationship is much more frequent than what chance (complements) indicates. If the Lift < 1, it indicates a relatively weak relation (controlling for the frequency with which both occur), that is, that the relation appears less frequently than the chance one indicates.

3.2. Apriori algorithm

Apriori is an algorithm proposed by R. Agrawal and R. Srikant (1994) to mine sets of frequent elements for Boolean association rules. The name of the algorithm is based on the fact that the algorithm uses prior knowledge of the frequent properties of the set of elements. Apriori employs an iterative approach known as a level search, where k-sets of elements are used to explore (k + 1) – itemset. First, the set of frequent 1-itemset is found by scanning the database to accumulate the count of each element and collecting the elements that satisfy the minimum support. The resulting set is denoted by **L1**. Next, **L1** is used to find **L2**, which is used to find **L3**, and so on, until more frequent **k-itemsets** are found. The finding of each **Lk** requires a complete analysis of the database.

To improve the efficiency of the generation of sets of frequent elements at level, an important property called Apriori property is used to reduce the search space. The Apriori property is based on the fact that if a set of elements does not satisfy the minimum support threshold (min_supp), then it is not frequent, that is, P(I) < min_supp. If an element A is added to the set of elements, then the resulting element (that is, I∩A) cannot happen more often than I. Therefore, I∩A is also not frequent, that is, P(I∩A) < min_sup.

This property belongs to a special category of properties called antimonotonicity in the sense that, if a set cannot pass a test, all its supersets will also fail the same test. This is called antimonotonicity because the property is monotonous in the context of the failure of a test (Han, Kamber & Pei, 2012). Figure 1 shows the Apriori algorithm.

C_k: Candidate item set of size k
L_k: Frequent item set of size k
L_1 = {frequent items};
For (k = 1; L_k! = ϕ; k++) do begin
C_{k+1} = candidates generated from L_k;
For each transaction t in database do
Increment the count of all candidates in C_{k+1}
Those are contained in t
L_{k+1} = candidates in C_{k+1} with min_support
End
Return U_k L_k;

Fig. 1. Apriori Algorithm

An application example of the algorithm is shown below. Let be the item set:

TID	Items
1	A, B
2	A, B, C, D
3	A, B, D, F
4	A, C, D, E
5	B, C, D, F
6	A, B, D

Where, the value as minimum support will be 0.5, that is:

$$\text{min\_supp} = 0.5$$

L1 is calculated (where L1 are the elements that satisfy the minimum support and those that do not satisfy it are eliminated):

L1:

Item	Frecuency	Support
A	5	5/6=0.83
B	5	5/6=0.83
C	3	3/6=0.5
D	5	5/6=0.83
E	1	1/6=0.16
F	2	2/6=0.33
L1		

Resulting set
A
B
C
D
L2

With the **L2** list, the cartesian product is calculated, thus obtaining the possible combinations that can there be of the products (**C2**), that is, **C2 = L2XL2**, and subsequently, the minimum support for **L2** is calculated, resulting in **L3**.

C2 = L2XL2

L2
A
B
C
D

X

L2
A
B
C
D

=

C2
{A,B}
{A,C}
{A,D}
{B,C}
{B,D}
{C,D}

→

C2	Count	Support
{A,B}	4	4/6=0.66
{A,C}	2	2/6=0.33
{A,D}	4	4/6=0.66
{B,C}	2	2/6=0.33
{B,D}	4	4/6=0.66
{C,D}	3	3/6=0.5
C2 Support		

→

Resulting set
{A,B}
{A,D}
{B,D}
{C,D}
L3

With **L3**, the Cartesian product is calculated, thus obtaining the possible combinations that can there be of the products (**C3**), that is, **C3** = **L3XL3**, and subsequently, the minimum support for **C3** is calculated, resulting in **L4**.

C3=L3XL3

L3
{A,B}
{A,D}
{B,D}
{C,D}

X

L3
{A,B}
{A,D}
{B,D}
{C,D}

=

C2
{A,B,D}
{A,B,C,D}
{A,D,C}
{B,C,D}

→

C2	Count	Support
{A,B,D}	3	4/6=0.66
{A,B,C,D}	1	1/6=0.16
{A,D,C}	2	2/6=0.33
{B,C,D}	2	2/6=0.33
C3 Support		

→

Resulting set
{A,B,D}
L4

Because **L4** only has one itemset, it is no longer possible to make more combinations, so the resulting rules are derived from **L2**, **L3** and **L4**, being as follows:

	Rules obtained	
L2	{A}	Dishes related with the selected dish
	{B}	
	{C}	
	{D}	
L3	{A,B}	Combos
	{A,D}	
	{B,D}	
	{C,D}	
L4	{A,B,D}	

4. IMPLEMENTATION OF APRIORI ALGORITHM

The programming of the classes and methods necessary for the implementation of the Apriori algorithm was done in the Java programming language through the IDE editor Netbeans. Likewise, it was necessary to create a database for the functionality of the system with the MySQL relational database manager with the help of the phpMyAdmin tool, since it allows us to manage the database in a more dynamic and simple way.

The database consists of three tables, the first corresponds to the table that stores the keys of the restaurants that are registered, the second stores the keys of the dishes offered by each restaurant, and the third table stores the orders registered by the restaurants. Figure 2 shows the tables of the database and their relationships.

Fig. 2. Database of the system and its relations

The algorithm returns two possible values: one, **the list of suggested combos of the selected restaurant**, or two, **the list of dishes that are sold together with a dish that the diner has selected**.

4.1. Steps to get the combos and frequent dishes

To calculate the list of combos and frequent dishes, the programmed Apriori algorithm executes the following main steps:

1. You get from the table **dishes** from the Restaurants database the list of dishes offered by the restaurant and it is added to the *Candidates Set*.
2. The transactions made by the restaurant are obtained from the table **orders** of the Restaurant database.
3. The database of the transactions is scanned and the support of each subset of elements k of the *Candidate Set* is calculated.
4. The value of the support obtained from each subset of candidate elements *k* is compared with the *minimum support* value.
5. If the support value of each subset of candidate elements *k* is greater than or equal to the *minimum support* value, the subset of candidate elements k is added to the *Auxiliary Candidate Set*, that is, elements that do not comply with the *minimum support* are eliminated.

6. The elements of the *Auxiliary Candidate Set* are added to the *Association Rules Set* together with their corresponding support value.
7. The Cartesian product of the *Auxiliary Candidate Set* is calculated.
8. The values of the *Candidate Set* are replaced by the values obtained by calculating the Cartesian product.
9. If the number of elements of the *Candidate Set* is greater than 1, then return to step 3, otherwise, the value of the *Candidate Set* is added to the *Association Rules Set* with its support value.
10. The *Association Rules Set* is ordered according to the support value of each subset of candidate elements *k*.

In case of requiring the list of combos suggested by the Apriori algorithm, subsets that contain more than one element will be taken from the *Association Rules Set*.

In case of requiring the *list of dishes purchased together with a dish selected by the diner*, the database of transactions will consist of the **extraction of the transactions that contain the dish that the diner selected**, that is, there will not be taken into account the totality of the orders or transactions registered by the restaurant. The procedure will begin in step 3. Finally, from the *Association Rules Set* those subsets that contain only one element will be taken.

4.2. Functioning tests of the Apriori algorithm

To verify the proper functioning of the Apriori programmed algorithm, tests were carried out on data obtained from surveys to diners about the dishes that are their preference from the menu of a restaurant in the region. The number of dishes offered by the restaurant is 108, while the transactions or orders obtained from the survey were 191.

The minimum support value was reduced until the algorithm showed association rules, since the greater the support, the fewer the number of rules obtained, leaving as a minimum support value of 0.1. The association rules obtained are **12** and are shown in figure 3 and the resulting combos are shown in figure 4.

```
INFORMACIÓN:   Rule 1:  {support= 0.35, rule=PL-A }
INFORMACIÓN:   Rule 2:  {support= 0.25, rule=PL-XXA }
INFORMACIÓN:   Rule 3:  {support= 0.21, rule=PL-K }
INFORMACIÓN:   Rule 4:  {support= 0.19, rule=PL-A PL-XXA }
INFORMACIÓN:   Rule 5:  {support= 0.17, rule=PL-A PL-K }
INFORMACIÓN:   Rule 6:  {support= 0.15, rule=PL-BT }
INFORMACIÓN:   Rule 7:  {support= 0.14, rule=PL-K PL-XXA }
INFORMACIÓN:   Rule 8:  {support= 0.14, rule=PL-A PL-K PL-XXA }
INFORMACIÓN:   Rule 9:  {support= 0.12, rule=PL-D }
INFORMACIÓN:   Rule 10:  {support= 0.12, rule=PL-G }
INFORMACIÓN:   Rule 11:  {support= 0.11, rule=PL-T }
INFORMACIÓN:   Rule 12:  {support= 0.10, rule=PL-M }
```

Fig. 3. Association rules obtained with a minimum support threshold value of 0.1

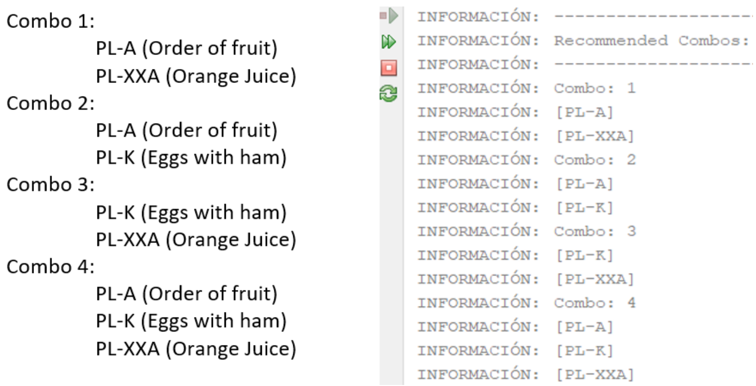

Fig. 4. Combos obtained with a minimum support threshold value of 0.1

On the other hand, if you want to show the diner a list of dishes that have been purchased together with a dish you have selected, the results could be the following:

Assuming that the dish selected by the diner is the one with the key PL-B (fruit with yogurt and granola), the association rules obtained with the minimum threshold value are 15 and are shown in figure 5.

The dishes purchased frequently with the dish PL-B (fruit with yogurt and granola) are PL-G (Buñuelo), PL-XXA (Orange Juice) and PL-BT (Chocolate Milk) as shown in figure 6.

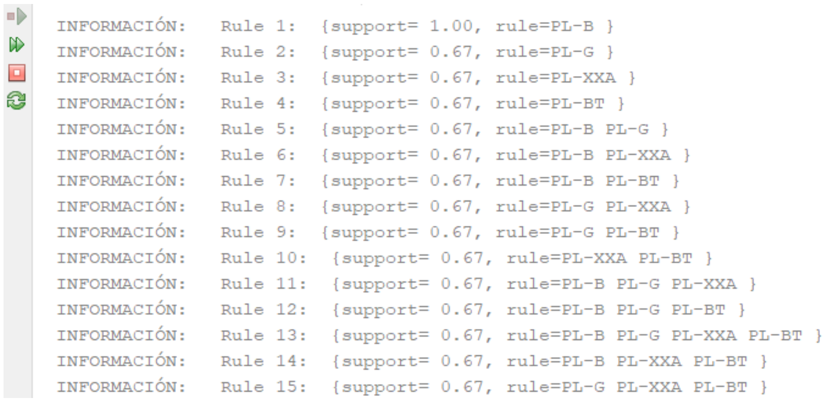

Fig. 5. Association rules obtained with the minimum support threshold value of 0.1 for the PL-B dish

```
INFORMACION: ---------------------
INFORMACIÓN: Dishes related with the dish selected by the user
INFORMACIÓN: ---------------------
INFORMACIÓN: Item: 1
INFORMACIÓN: [PL-G]
INFORMACIÓN: Item: 2
INFORMACIÓN: [PL-XXA]
INFORMACIÓN: Item: 3
INFORMACIÓN: [PL-BT]
```

Fig. 6. Frequent dishes acquired with the PL-B dish with minimum support of 0.1

5. INTEGRATION OF THE COMBOS GENERATION MODULE TO THE FOOD EXPRESS APPLICATION

Because the system will be used by the Food Express application, WebService technology will be used, since they are distributed applications that are based on a series of protocols and standards for exchanging information, allowing applications to communicate with each other, regardless of language or platform in which they are developed. To implement the web service, the following resources were used: Java Development Kit JDK, NetBeans IDE 8.2 and the Application Server GlassFish Server.

To allow communication between the digital platform and the Apriori algorithm module, the communication protocol for the transfer of REST information and the response format for JSON data exchange was used.

The Food Express application will make requests to the WebService through HTTP requests and the Web Service will return as an answer either an array in JSON format that contains the list of the combos that result from the execution of the algorithm or, in another case, an array in JSON format that stores the list of dishes that have been purchased together with the dish that the diner has selected. The above is illustrated in Figure 7, which shows the communication between the client and the web service (Module).

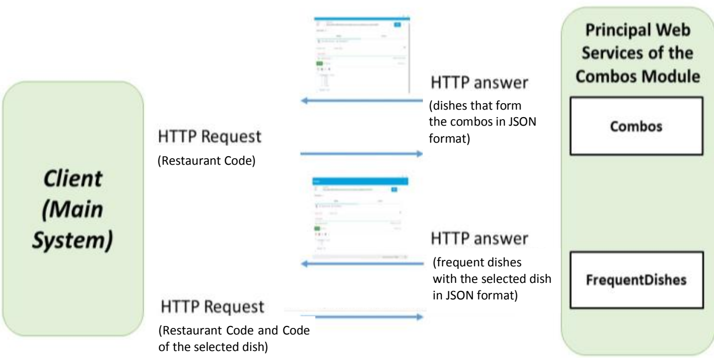

Fig. 7. Communication between the client and the web service

To carry out the communication between the webservice and the combos' module, the necessary methods and operations for communication, administration and configuration of the services offered by the Apriori module were created. Figure 8 shows a diagram with the services created and offered by the WebService, as well as the methods each service has communication with.

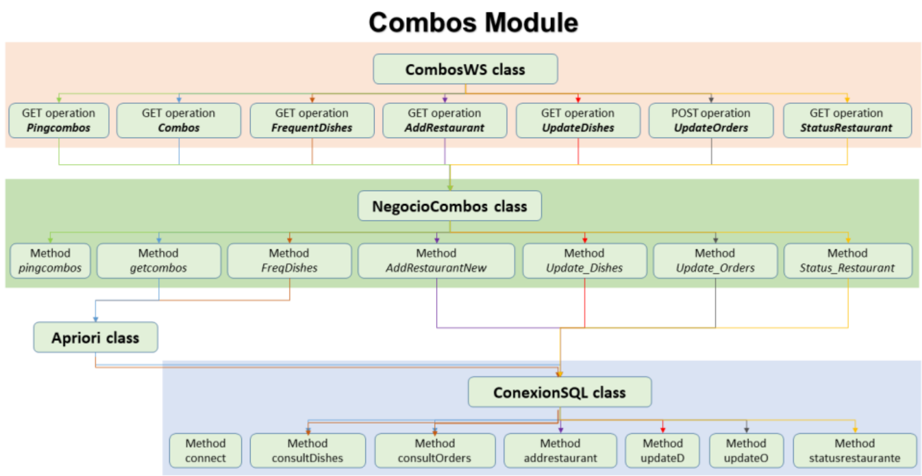

Fig. 8. Diagram of Web Service Combos classes

Table 1 specifies the necessary values for its correct operation, as well as the services implemented.

Tab. 1. WebServices for the administration, configuration and communication of the module

WS' name: ***Pingcombos***	
Operation type: GET	**Input values (URL):** WebServiceCombos/service/moduloService/pingcombos
Descripción: Check the connection to the server by returning the message "In connection with the server".	
WS' name: ***AddRestaurant***	
Operation type: GET	**Input values (URL):** WebServiceCombos/service/moduleService/AddRestaurant/{IdRestaurant}
Descripción: Insert a restaurant in the database of the module. It receives as parameter the restaurant ID and returns as a response the message "Restaurant added".	
WS' name: ***UpdateDishes***	
Operation type: GET	**Input values (URL):** WebServiceCombos/service/moduleService/updateDishes/{IdRestaurant}/{KeyDish}
Descripción: Insert a dish into the module database. It receives as parameter the key of the restaurant and the key of the dish. Returns the message "Dish added".	

Tab. 2. WebServices – cont.

<table>
<tr><td colspan="2">WS' name: UpdateOrders</td></tr>
<tr><td>Operation type:
POST</td><td>Input values (URL):
WebServiceCombos/service/moduleService/updateOrders</td></tr>
<tr><td colspan="2">Descripción:
Add a command to the module database. It receives as parameter an object in JSON format with the Id of the restaurant, the Id of the command and the Id of the dish. Returns the message “Successful element added”.</td></tr>
<tr><td colspan="2">WS' name: StatusRestaurant</td></tr>
<tr><td>Operation type:
GET</td><td>Input values (URL):
WebServiceCombos/service/moduleService/StatusRestaurant/{IdRestaurant}/{Status}</td></tr>
<tr><td colspan="2">Descripción:
Change the status of a restaurant in the database of the module, receiving as parameter 1 (activated) or 0 (Disabled). Returns the message “Element on / Element deactivated successfully”.</td></tr>
<tr><td colspan="2">WS' name: FrequentDishes</td></tr>
<tr><td>Operation type:
GET</td><td>Input values (URL):
WebServiceCombos/service/moduleService/frequentDishes/{IdRestaurant}/{selected Dishes}/{dishesLimit}</td></tr>
<tr><td colspan="2">Descripción:
Returns the dishes that are frequently purchased with a selected dish, limiting them to the number of elements requested. It receives as parameters the Id of the restaurant, the Key of the selected dish and the limit number of dishes related to be displayed.</td></tr>
<tr><td colspan="2">WS' name: Combos</td></tr>
<tr><td>Operation type:
GET</td><td>Input values (URL):
WebServiceCombos/service/moduleService/combos/{IdRestaurant}/{CombosLimit}</td></tr>
<tr><td colspan="2">Descripción:
Returns the combos suggested by the algorithm, limiting them to the number of combos requested. It receives as parameters the Id of the restaurant and the limit number of combos to be displayed.</td></tr>
</table>

5.1. Acquisition of the clients' preferences from the mobile application

The implementation of the module of combos leads to the interaction with the own processes of the mobile Food Express application to which it is integrated. This interaction is represented in cases of usage of each service belonging to the module. Figure 9 shows the case of the use of Combos service and figure 10 shows the case of usage of frequent meals service.

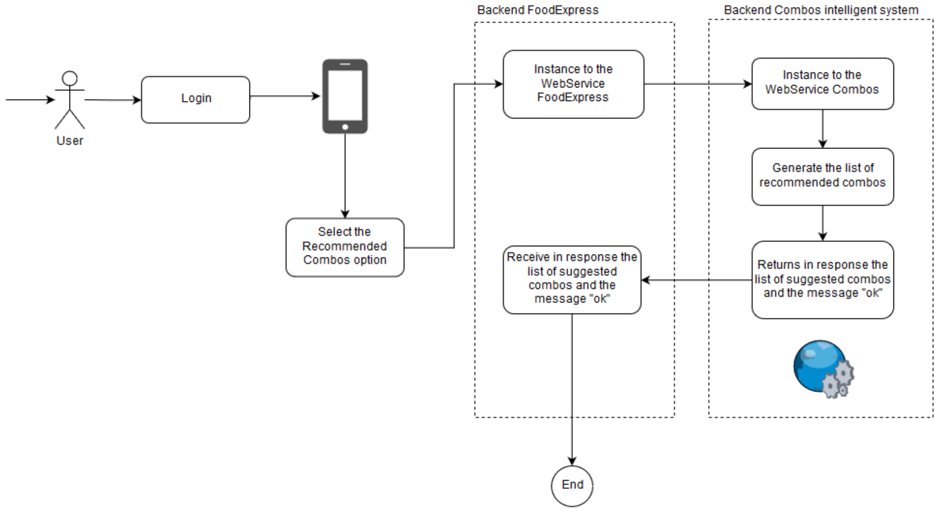

Fig. 9. Use Case of the Combos Service

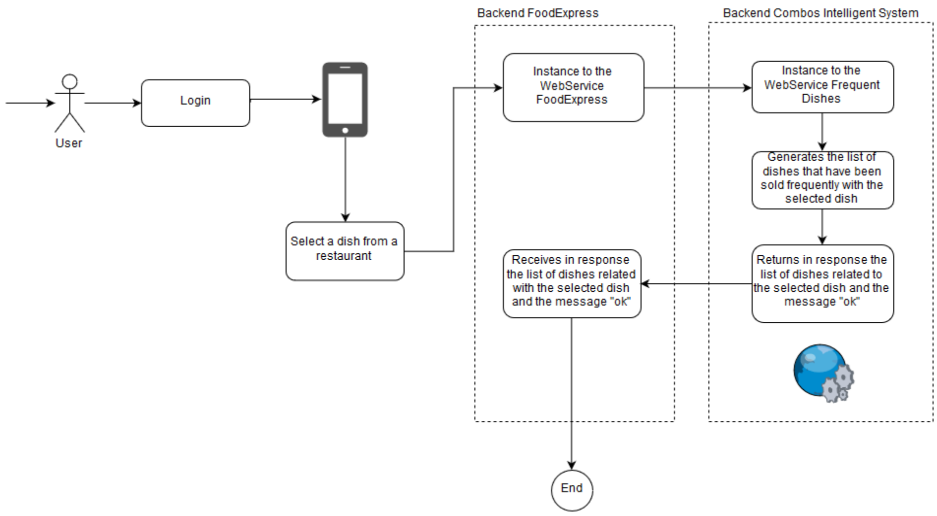

Fig. 10. Use Case of the Frequent Dishes Service

5.2. Monitoring of outputs

To realize the tests with the webservice and observe the outputs, the Advanced REST Client application provided by Google was downloaded to create and test the personalized HTTP requests. Advanced REST Client is considered as the only REST client that makes the connection directly in the socket that gives total control over the connection and the request/response headers. With the application, it was possible to corroborate that the service works correctly with the specified values, so it was integrated as a web service for the Food Express application.

6. CONCLUSIONS

In this document, information is presented on the development of the Module for the Generation of Combos in restaurants for the Food Express application. Comparing several investigated methodologies, it was concluded that the Apriori Algorithm is perfectly suited to solve the problems posed.

When programming the Apriori algorithm, tests were performed to optimize the classes and methods to maintain a certain level of code quality in the program. To carry out the tests, the SonarQube application was used to audit the code of the application during programming. SonarQube is an open source tool for the analysis and improvement of the code quality of a program.

On the other hand, considering the importance of performing stress tests on the developed software, it became necessary to undertake the task of performing the tests with the Apache JMeter ™ application, which is an open source software designed to load the functional behavior of Test and measure performance.

The processes for the implementation of the combos module were evaluated in a timely manner, carrying out the necessary tests to monitor that the module carried out the operative processes efficiently and effectively, and thus, detect any problems that may arise for its proper functioning.

The functioning tests of the Apriori algorithm show that the algorithm yields the frequent items within a transaction base, which guarantees that the combos suggested by the algorithm are obtained based on the preferences of the diners, acting in the same way to obtain the elements that are frequently acquired with a selected element. Likewise, it was observed that the Apriori algorithm programmed in the Java language has problems when executing high dimension data.

The module will run as a web service, which will be accessed by the Food Express application through HTTP requests made by the client. All the processes are executed in the server and this will be responsible for returning the results to the device that requests it, so that the Apriori algorithm is carried out in the form of a black box, thus achieving its independence.

This project is a contribution that allows to generate a change in the way in which diners interact with restaurants, at the same time, it is a resource available for the restaurant industry, which will be the one that dictates through the use and operation of this system the viability of them.

REFERENCES

Agrawal, R. & Srikant, R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the 20th International Conference on Very Large Data Bases* (pp. 487-499). Santiago de Chile.

Han, J., Kamber, M., & Pei, J. (2012). Data Mining Concepts and Techniques. Third Edition. Waltham, USA: Elsevier.

Harikumar, S., & Dilipkumar, D. U. (2016). Apriori algorithm for association rule mining in high dimensional datas. In *2016 International Conference on Data Science and Engineering (ICDSE)* (pp. 1–6). Cochin. doi:10.1109/ICDSE.2016.7823952.

Huang, Y., Lin, Q., & Li, Y. (2018). Apriori-BM Algorithm for Mining Association Rules based on Bit Set Matrix. In *2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)* (pp. 2580–2584). Xi'an. doi:10.1109/IMCEC.2018.8469367

Kabir, M. M. J., Xu, S., Kang B. H., & Zhao, Z. (2015). Comparative Analysis of Genetic Based Approach and Apriori Algorithm for Mining Maximal Frequent Item Sets. In *2015 IEEE Congress on Evolutionary Computation (CEC)* (pp. 39–45). Sendai. doi:10.1109/ CEC.2015.7256872.

Park, S., & Park, Y. (2018). Analysis of Association Between Students' Mathematics Test Results Using Association Rule Mining. In *2018 International Conference on Platform Technology and Service (PlatCon)* (pp. 1–6). Jeju. doi:10.1109/PlatCon.2018.8472756

Pei, S. (2013). Application of data mining technology in the tourism product's marketing CRM. In *2013 2nd International Symposium on Instrumentation and Measurement, Sensor Network and Automation (IMSNA)* (pp.400–403). Toronto, ON. doi:10.1109/IMSNA.2013.6743300.

Tom, M., & Annaraud, K. (2017). A fuzzy multi-criteria decision making model for menu engineering. In *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (pp. 1–6). Naples. doi:10.1109/FUZZ-IEEE.2017.8015612.

Zheng, X. Z. (2007). Building Personalized Recommendation System in E-Commerce using Association Rule-Based Mining and Classification. In *2007 International Conference on Machine Learning and Cybernetics* (pp. 4113–4118). Hong Kong. doi:10.1109/ ICMLC.2007.4370866.

Zulfikar, W. B., Wahana, A., Uriawan, W., & Lukman, N. (2016). Implementation of association rules with apriori algorithm for increasing the quality of promotion. In *2016 4th International Conference on Cyber and IT Service Management* (pp. 1–5). Bandung. doi:10.1109/CITSM.2016.7577586.