Crave – A Customized Drive-Thru Experience

Pooja Kataria
Department of Computer Engineering San Jose State University
San Jose, USA
pooja.kataria@sjsu.edu

Saumil Shah
Department of Computer Engineering San Jose State University
San Jose, USA
saumil.j.shah@sjsu.edu

Shivang Mistry
Department of Computer Engineering San Jose State University
San Jose, USA
shivang.mistry@sjsu.edu

Tejas Desai
Department of Computer Engineering San Jose State University
San Jose, USA
tejas.desai@sjsu.edu

Team 10

***Abstract*— This paper describes the implementation of Crave, which is our implementation of customized drive-thru for fast food joints using customers data. Crave uses machine learning algorithms like Apriori to make a customized menu for the user to predict users' preferred items and suggest other items for a combination of meals. Our application also uses Pattern Matching algorithm to detect car number plates to identify the user. Moreover, Crave implements weather forecasting using the zip code of the location, which will alter the menu depending on how cold or warm the weather is. Overall, these components will help create a customized menu for the user, to help them choose a meal for themselves.**

***Keywords*— Customized menu, Machine Learning, Apriori, Pattern Matching, Weather Forecasting.**

I. Introduction

A lot of fast food joints are competing with each other in order to be better than others. With the amount of people visiting a fast food joint is very high and to maintain the service levels to reduce waiting time is always a challenge. To improve the service speed, several fast food joints have started a drive-thru service which helps to reduce time. But again, every customer spends a lot of time on deciding their order which again brings up the issue of time and efficiency. So, to tackle this issue of time and efficiency, our application, Crave will help a customer choose their order by giving suggestions depending on their previous choices, weather and other factors and help them create a meal for themselves to reduce the time taken by each customer, by identifying each customer by their car number plate to give them an experience of a customized menu, by collecting user data every time they visit and place an order.

II. PATTERN MATCHING

A. Theory

Pattern Matching is the act of checking a given sequence of tokens for the presence of the constituents of some pattern. In contrast to pattern recognition, the match usually has to be exact: "either it will or will not be a match." The patterns generally have the form of either sequences or tree structures. Uses of pattern matching include outputting the locations (if any) of a pattern within a token sequence, to output some component of the matched pattern, and to substitute the matching pattern with some other token sequence. Sequence patterns are often described using regular expressions and matched using techniques such as backtracking. Tree patterns are used to process data based on its structure. For simplicity and efficiency reasons, these tree patterns lack some features that are available in regular expressions. Often it is possible to give alternative patterns that are tried one by one, which yields a powerful conditional programming construct. Pattern matching sometimes includes support for guards. Parsing algorithms often rely on pattern matching to transform strings into syntax trees.

B. Our Use-Case

In our case, we will be collecting user data by scanning their car number plate every time they visit. By implementing Pattern Matching, we will scan the number plate and identify the customer and then create a customized menu for that particular customer. If it is a returning customer, the system will the user details and according to the previous stored data, a menu will be displayed. Pattern matching will help to identify a car and check if it is a new customer or a returning customer which will help to generate a customized menu.

Fig. 1: A sample image of a car number plate.

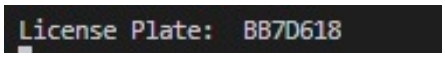

Fig. 2: License Plate scan result

Fig. 3: Database Information of that customer

III. Weather Forecasting

A. Theory

Weather forecasting is the application of science and technology to predict the conditions of the atmosphere for a given location and time. Weather forecasts are made by collecting quantitative data about the current state of the atmosphere at a given place and using meteorology to project how the atmosphere will change. Weather forecasting now relies on computer-based models that take many atmospheric factors into account. Human input is still required to pick the best possible forecast model to base the forecast upon, which involves pattern recognition skills, teleconnections, knowledge of model performance, and knowledge of model biases. The inaccuracy of forecasting is due to the chaotic nature of the atmosphere, the massive computational power required to solve the equations that describe the atmosphere, the error involved in measuring the initial conditions, and an incomplete understanding of atmospheric processes.

B. Our Use-Case

In our case, we will be analyzing the weather to generate a customized menu. Depending on the weather, we will classify it as warm or cold and then suggest food or beverage items to the customer. Depending on the zip code of the fast food drive-thru, we will analyze the weather. For example, if the temperature is greater than 70°F, then it will be classified as hot and will suggest beverages like hot chocolate, espresso, etc. and if the temperature is below 70°F, then beverages like cold-brew, Coca-Cola, Frappuccino will be suggested. This way, weather analysis plays an important role in our implementation.

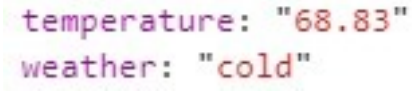

Fig. 4: Weather Analysis depending on the temperature

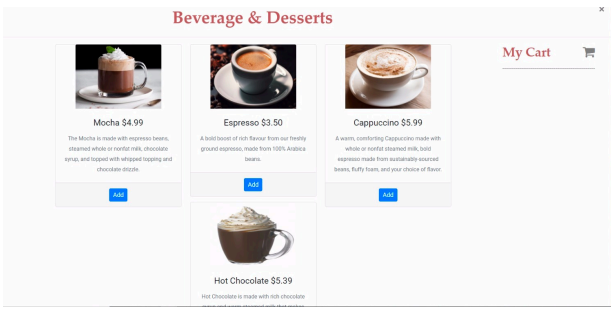

Fig. 5: Customized menu depending on the weather

IV. Apriori Algorithm

A. Theory

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

B. Our Use Case

In our case, we will collect user data to offer them their favorite meals and items depending on several factors and their previous choices. We have then implemented Apriori algorithm to add additional meal options along with their favorite items to create a customized meal option. This will help to suggest additional items that will go with their choice of meal. For e.g. If the customer prefers a Beef Burger, our algorithm will offer Fries and a choice of drink to go with the burger to improve the sales and offer appropriate items for an improved customer experience.

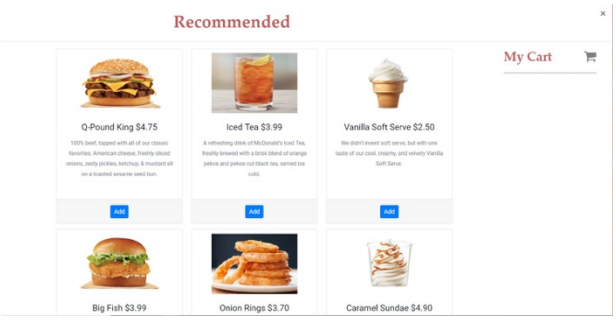

Fig. 6: Suggestions generated using Apriori algorithm.

V. Architecture Flow

The architecture of Crave is as follows:

- The camera in the drive-thru will capture the image of the number plate.
- The system will recognize the customer.
- The system will capture the weather and analyze the temperature and weather conditions and decide the beverage to offer depending on the weather.
- The system will analyze the customer details (previous orders) of a returning customer.
- In case of a new customer, most popular meals will be collected.
- Next, a customized menu will be generated depending on these factors.
- Implementing the Apriori algorithm, the system will generate meal items that will go best with the items selected by the customer.
- The order will be placed.
- The new consumer details will be stored in the database (Mongo DB) for future use.

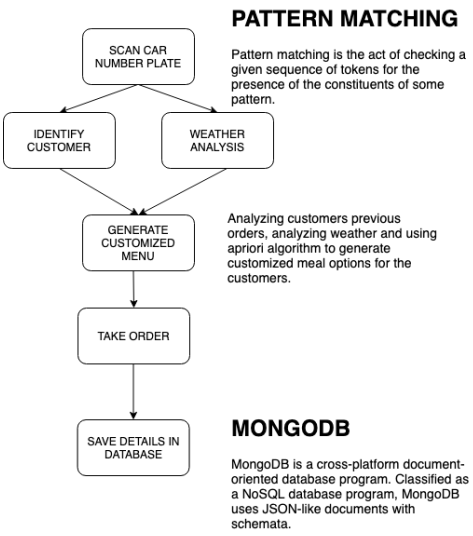

Fig. 7: System architecture Flow Diagram of Crave.

VI. Implementation

Fig. 8 shows the backend part, which recognizes the car number plate, analyzes the weather to classify it as hot or cold. We have implemented this part by Pattern matching using OCR and Weather Forecasting respectively.

```
Current temperature:  65.05
Current weather:  cold
License Plate:  BB7D618
```

Fig. 8: Backend Part

Fig. 9 shows the UI of our application. It is the dashboard of our system, which offers 4 options to choose from to the customer.

The customer can either go for his favorites, suggestions, suggested beverages or explore the entire menu.

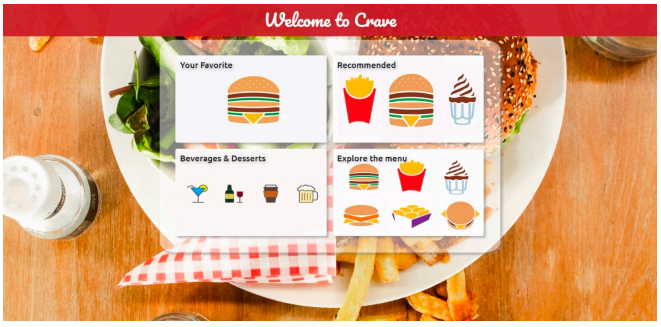

Fig. 9: UI of our system (Dashboard)

Fig. 10 shows the customers favorites, depending on his previous orders, for a returning customer and for a new customer, it will show the most popular meals. This will help to choose the meal faster and not remember what to order everytime.

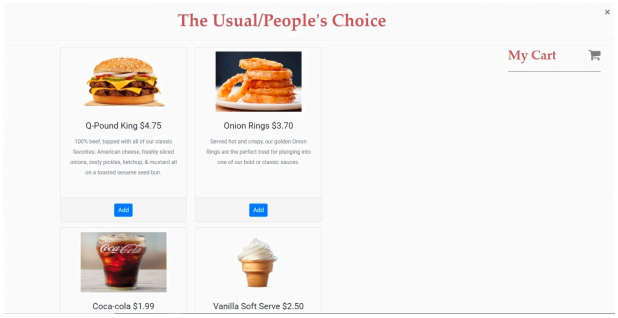

Fig. 10: UI of the system (Usual/People's Choice)

VII. Conclusions

In this project, we developed a customized drive-thru experience with the help of user data and implementation of several algorithms. We implemented Pattern Matching using OCR (optical character recognition) to identify car number plates. To analyze weather, we pulled weather information of a zip code and depending on the temperature, we classified it as cold or hot. For better user experience, we also implemented Apriori algorithm to suggest them more meal options appropriate with their choice of food. All these factors were considered to generate a customized menu for the customer for an improved experience.

For the future work, we would like to extend this application for the benefit of many fast food joints. Also, we aim to assign more users to a particular car and an option to select the user visiting the fast food joint. For a better and improved experience in the future, we would also like to include an option for customer feedback.

VIII. Acknowledgement

We would like to thank Professor Rakesh Ranjan for his constant motivation to design innovative solution for the existing real-time challenges helped us to closely understand the existing problem and succeed in implementation of the project.

IX. Project Repository

[1] https://github.com/SJSU272Spring2019/Project-Group-10

X. References

[1] https://www.openalpr.com/on-premises.html
[2] https://openweathermap.org
[3] https://www.fool.com/investing/2019/03/27/mcdonalds-just-spent-300-million-on-drive-thru-per.aspx
[4] http://citeseerx.ist.psu.edu