Restaurant Recommendation System

1) **Objective - to develop an AI / ML based model which is trained on the transaction history of 100,000 users for 12 months - including their actual history and cravings for each month, for the objective of predicting what the dishes the user will eat in future.**

2) **Context - there are 15 dishes in total. Each month has the history of all 15 dishes as 0s or 1s and the craving for that month, which is 70% exact match of the transaction history for that month. Each restaurant has either Japanese, Indian or Chinese cuisine. Each cuisine has 5 dishes. It is assumed that if a restaurant has say Japanese cuisine, it has all the 5 Japanese dishes. Furthermore, the months and the dishes are divided into seasons.**

```
season_months = {
    "Winter": ['Jan', 'Feb', 'Mar', 'Nov', 'Dec'],
    "Summer": ['Apr', 'May', 'Jun', 'Jul'],
    "Monsoon": ['Aug', 'Sep', 'Oct'],
}
```

```
dish_to_name = {
    'dish1': "Aamras Poori", 'dish2': "Boondi Raita", 'dish3': "Pakora", 'dish4': "Sarson da Saag & Makki di Roti", 'dish5': "Pav Bhaji",
    'dish6': "Sushi Rolls", 'dish7': "Matcha Ice Cream", 'dish8': "Udon Noodles", 'dish9': "Tonkotsu Ramen", 'dish10': "Gyoza",
    'dish11': "Chilli Paneer", 'dish12': "Gobi Manchurian", 'dish13': "Chilli Mushroom", 'dish14': "American Chopsuey", 'dish15': "Schezwan Fried Rice"
}
```

3) **Files Generated -**

- `flattened_by_seaons_with_userinfo.parquet - contains the transaction history of 100,000 users and their userids`

- Lgbm_dish_recommender_by_seasons.pkl - the ML model used after test and train
- Users.csv - contains the locations of the users in lat / long and cities
- Restaurants_with_cuisine.csv - contains the restaurants, their lat/long and their cuisine.
- model300season.py - this is the test and train file used to train the LightGBM Classifiers
- deploy300test.py - this is the deploy part of the code

4) **<u>Transaction history</u> - the ML model is trained on the generated transaction history of the users. Each month has a 15 character long of either 1s or 0s.**

Jan - str	Jan_craving - str	Feb - str	Feb_craving - str	Mar - str	Mar_craving - str	Apr - str	Apr_craving - str	May - str	May_craving - str	Jun - str	Jun_craving -
001000000000000	001000000000000	101000000000000	100000000000000	101000000000000	101000000000000	110010000000000	100010000000000	110010000000000	110010000000000	100000000000000	100000000000
000000010000000	000000010000000	000000010000000	000000010000000	000000011000000	000000000000000	000001000000000	000001000000000	000001100000000	000001100000000	000000001100000	000000001100
000000000000010	000000000000010	000000000010001	000000000010001	000000000000011	000000000000011	000000000010000	000000000010000	000000000010101	000000000010101	000000000010000	000000000010
000001011000000	000001011000000	000000011100000	000000000100000	000001111000000	000001010000000	000000100000000	000000100000000	000000100100000	000000100100000	000000111100000	000000100100
000000000000000	000000000000000	000000011000000	000000011000000	000000101000000	000000101000000	000001100100000	000000100000000	000001100000000	000001100000000	000001000000000	000001000000
000000000010111	000000000010001	000000000001010	000000000000010	000000000000101	000000000000001	000000000001000	000000000000000	000000000010001	000000000010001	000000000010000	000000000000
000000000000011	000000000000011	000000000000100	000000000000100	000000000000111	000000000000111	000000000010011	000000000000000	000000000010000	000000000010000	000000000011101	000000000001

5) **<u>Test and train -</u> recommender system is based on the LightGBM model and its MultiOutput Classifier. Instead of having a single model, the MultiOutput Classifiers act like 1 model per dish so now we have 15 models for the 15 dishes. The classifiers are trained on cravings, history, seasonal biases to make either a yes or no prediction as to the dish will be eaten when asked. The biggest benefit of this method is that each model will determine the custom probability cutoff for each dish on whether to predict yes or no for a dish.**

6) Test and train results with F1 scores and confusion matrix -

F1 Scores per dish:
Dish 1 best F1=0.668 at threshold=0.80
Dish 2 best F1=0.665 at threshold=0.80
Dish 3 best F1=0.717 at threshold=0.80
Dish 4 best F1=0.597 at threshold=0.85
Dish 5 best F1=0.604 at threshold=0.85
Dish 6 best F1=0.666 at threshold=0.80
Dish 7 best F1=0.665 at threshold=0.80
Dish 8 best F1=0.714 at threshold=0.75
Dish 9 best F1=0.718 at threshold=0.80
Dish 10 best F1=0.601 at threshold=0.80
Dish 11 best F1=0.662 at threshold=0.85
Dish 12 best F1=0.599 at threshold=0.85
Dish 13 best F1=0.598 at threshold=0.85
Dish 14 best F1=0.714 at threshold=0.75
Dish 15 best F1=0.715 at threshold=0.75

Dish 1 confusion matrix:
[[200258 6658]
[13141 19943]]

	precision	recall	f1-score	support
0	0.94	0.97	0.95	206916
1	0.75	0.60	0.67	33084
accuracy			0.92	240000
macro avg	0.84	0.79	0.81	240000
weighted avg	0.91	0.92	0.91	240000

Dish 2 confusion matrix:
[[200266 6812]
[13140 19782]]

	precision	recall	f1-score	support
0	0.94	0.97	0.95	207078
1	0.74	0.60	0.66	32922
accuracy			0.92	240000
macro avg	0.84	0.78	0.81	240000
weighted avg	0.91	0.92	0.91	240000

Dish 3 confusion matrix:
[[195526 8099]
[11515 24860]]

	precision	recall	f1-score	support
0	0.94	0.96	0.95	203625
1	0.75	0.68	0.72	36375
accuracy			0.92	240000
macro avg	0.85	0.82	0.83	240000
weighted avg	0.92	0.92	0.92	240000

Dish 4 confusion matrix:
[[205425 5113]
[14741 14721]]

	precision	recall	f1-score	support
0	0.93	0.98	0.95	210538
1	0.74	0.50	0.60	29462
accuracy			0.92	240000
macro avg	0.84	0.74	0.78	240000
weighted avg	0.91	0.92	0.91	240000

Dish 5 confusion matrix:
[[205409 4870]
[14757 14964]]

	precision	recall	f1-score	support
0	0.93	0.98	0.95	210279
1	0.75	0.50	0.60	29721
accuracy			0.92	240000
macro avg	0.84	0.74	0.78	240000
weighted avg	0.91	0.92	0.91	240000

Dish 6 confusion matrix:
[[199727 6813]
[13372 20088]]

	precision	recall	f1-score	support

0	0.94	0.97	0.95	206540
1	0.75	0.60	0.67	33460
accuracy			0.92	240000
macro avg	0.84	0.78	0.81	240000
weighted avg	0.91	0.92	0.91	240000

Dish 7 confusion matrix:
[[199620 6797]
[13454 20129]]

	precision	recall	f1-score	support
0	0.94	0.97	0.95	206417
1	0.75	0.60	0.67	33583
accuracy			0.92	240000
macro avg	0.84	0.78	0.81	240000
weighted avg	0.91	0.92	0.91	240000

Dish 8 confusion matrix:
[[194635 8392]
[11775 25198]]

	precision	recall	f1-score	support
0	0.94	0.96	0.95	203027
1	0.75	0.68	0.71	36973
accuracy			0.92	240000
macro avg	0.85	0.82	0.83	240000
weighted avg	0.91	0.92	0.91	240000

Dish 9 confusion matrix:
[[194705 8244]
[11708 25343]]

	precision	recall	f1-score	support
0	0.94	0.96	0.95	202949
1	0.75	0.68	0.72	37051
accuracy			0.92	240000

```
   macro avg       0.85      0.82      0.83    240000
weighted avg       0.91      0.92      0.92    240000
```

Dish 10 confusion matrix:

```
[[204694   5138]
 [ 14987  15181]]
              precision    recall  f1-score   support

           0       0.93      0.98      0.95    209832
           1       0.75      0.50      0.60     30168

    accuracy                           0.92    240000
   macro avg       0.84      0.74      0.78    240000
weighted avg       0.91      0.92      0.91    240000
```

Dish 11 confusion matrix:

```
[[199988   6899]
 [ 13307  19806]]
              precision    recall  f1-score   support

           0       0.94      0.97      0.95    206887
           1       0.74      0.60      0.66     33113

    accuracy                           0.92    240000
   macro avg       0.84      0.78      0.81    240000
weighted avg       0.91      0.92      0.91    240000
```

Dish 12 confusion matrix:

```
[[204938   4957]
 [ 15112  14993]]
              precision    recall  f1-score   support

           0       0.93      0.98      0.95    209895
           1       0.75      0.50      0.60     30105

    accuracy                           0.92    240000
   macro avg       0.84      0.74      0.78    240000
weighted avg       0.91      0.92      0.91    240000
```

Dish 13 confusion matrix:

```
[[205057   5048]
 [ 14999  14896]]
              precision    recall  f1-score   support

           0       0.93      0.98      0.95    210105
           1       0.75      0.50      0.60     29895

    accuracy                           0.92    240000
   macro avg       0.84      0.74      0.78    240000
weighted avg       0.91      0.92      0.91    240000
```

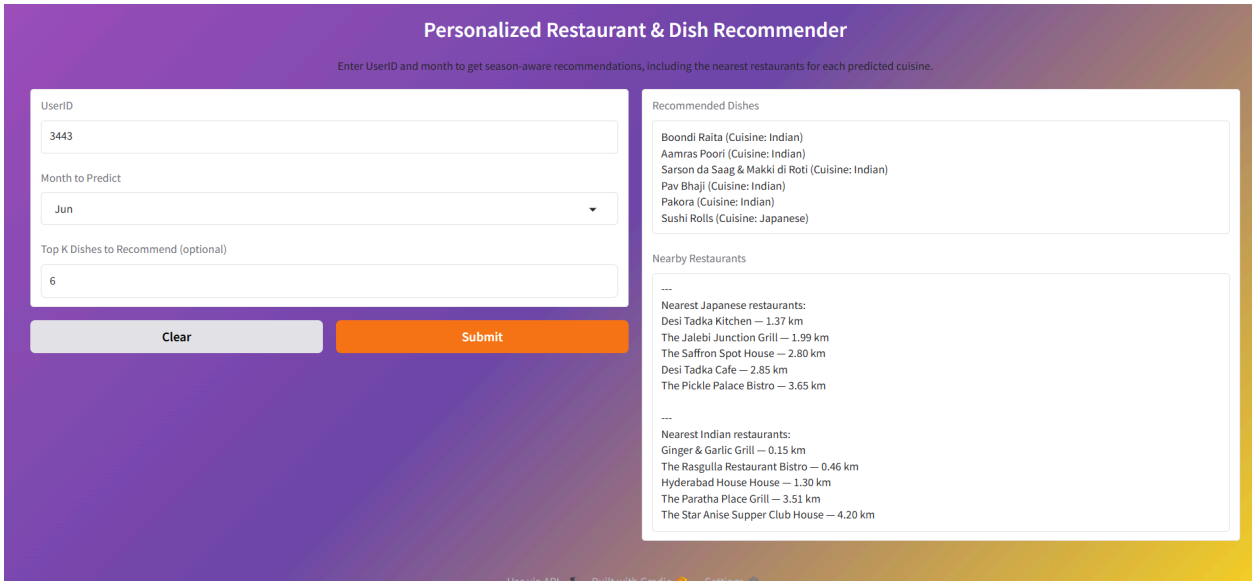
```
Dish 14 confusion matrix:
[[194964   8475]
 [ 11550  25011]]
              precision    recall  f1-score   support

           0       0.94      0.96      0.95    203439
           1       0.75      0.68      0.71     36561

    accuracy                           0.92    240000
   macro avg       0.85      0.82      0.83    240000
weighted avg       0.91      0.92      0.92    240000
```

```
Dish 15 confusion matrix:
[[194941   8415]
 [ 11585  25059]]
              precision    recall  f1-score   support

           0       0.94      0.96      0.95    203356
           1       0.75      0.68      0.71     36644

    accuracy                           0.92    240000
   macro avg       0.85      0.82      0.83    240000
weighted avg       0.91      0.92      0.92    240000
```

7) **Deploy block - the deploy block uses gradio to deploy and create a simple UI interface for the user. The user enters the userid and the location and the model recommends the top dishes for that month. The model first looks up the month that is entered and then looks up the other months in the same**

seasons of that month to increase prediction accuracy as the flattened data has a bias towards the seasonality of the dishes and the months. The deploy block upon deciding the dishes to recommend, looks up the cuisines of the restaurants, it will recommend 5 restaurants per cuisine. Now, the model will look up the restaurant_with_cuisine.csv to find the closest restaurant to the user to recommend the top dishes.

Example :



8) **<u>Final Feedback</u> - the model predicts each dish with a high degree of accuracy as each dish has its own classifier to determine the probability cutoff. This method is suitable for multiple labelled outputs with limited columns like 15.**
The Restaurant Recommender System works on LightGBM model and uses MultiOutputClassifiers to make predictions for each dish. Another benefit of the LightGBM Method is the ability to use class=weighted option to ensure that the minority cases are not looked over, instead the model prioritises the rare cases in training.