Recipe Recommendation System

-Aamna Ahsan

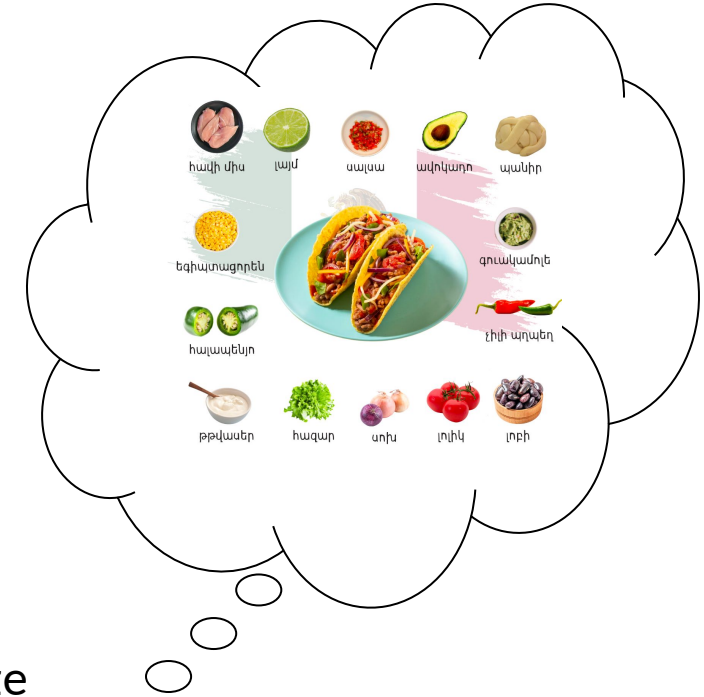
Purpose

Having difficulty deciding what to cook?

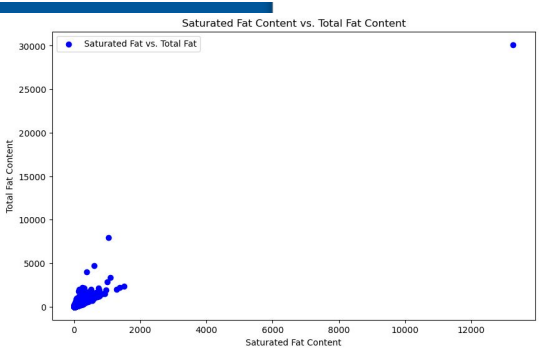
OR

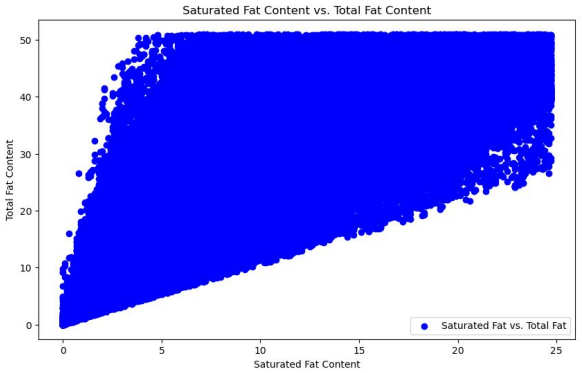
Are you looking to expand your cuisine based on special diet, ingredients or culture?

A recipe recommendation system can personalize your needs based on popularity, exploring new recipes based on your favorite ingredients, cuisines, and dietary preferences.



Example1: Fat Content vs Saturated Fat:





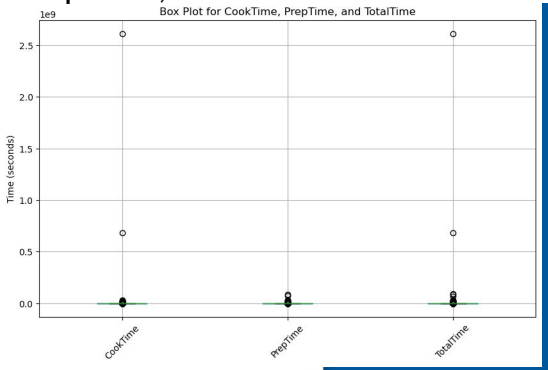
1. Data:

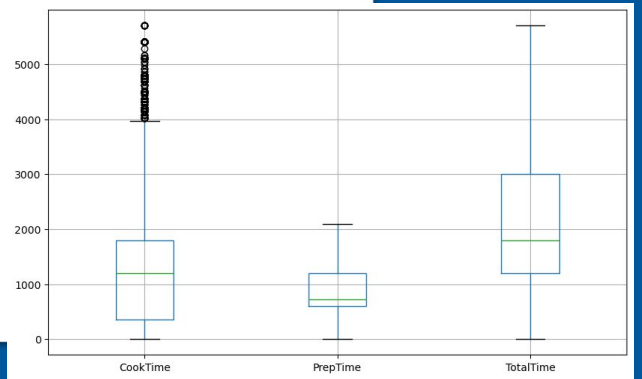
Recipe.csv file was extracted from kaggle.

➔ **Clean-up**

-remove NaN values, duplicates and outliers in the data.

Example 2: Box plot of CookTime, PrepTime, and TotalTime.

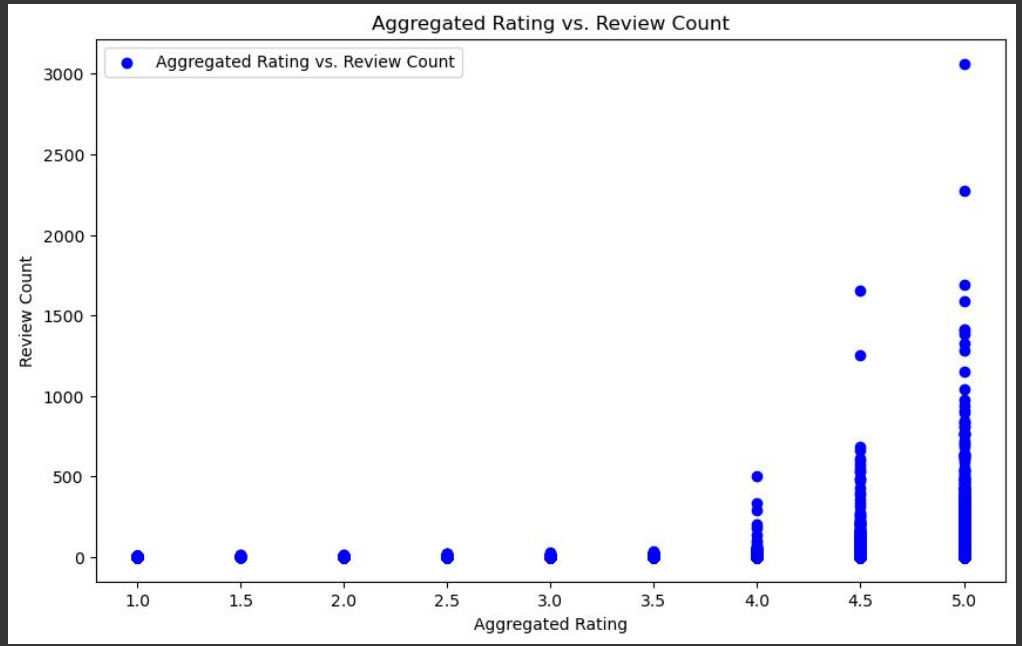




This chart shows how the recipes were rated in the data:

Looks like there were a lot of recipes that people loved.

Most ratings were 5.0/5.0



Specialized Features:

1. Dietary preference: Gluten-Free, Dairy-Free, Low-carb.

2. Categories: Meat, seafood, Vegetarian :

```
print(df[['Name','MealType']].head(-10))
```

```
                                                Name    MealType
0                       Low-Fat Berry Blue Frozen Dessert  Vegetarian
1                                       Cabbage Soup  Vegetarian
2             Buttermilk Pie With Gingersnap Crumb Crust  Vegetarian
3                               A Jad - Cucumber Pickle  Vegetarian
4                                  Butter Pecan Cookies  Vegetarian
...                                                 ...         ...
325259    Chicken Pot Pie with Mashed Potato Crust        Meat
325260              Masala Maggi Noodles in a Mug  Vegetarian
325261                    Chocolate Rum Snowballs  Vegetarian
325262            Cookie Cutter Shortbread Hearts  Vegetarian
325263               11-Minute Microwave Caramels  Vegetarian

[325264 rows x 2 columns]
```

Specialized Features Continued:

```
]:
#Testing the feature:
indian_recipes = df[df['CuisineType'] == 'Indian']
print(indian_recipes)
```

	RecipeId	Name	AuthorId	CookTime
58	159	Chicken Curry	148316	2520.0
94	236	Chicken Curry II	1597	1.0
321	675	Curried Chicken II	1543	1.0
605	2490	Cucumber Raita	1533	1.0
684	2673	West Indian Bread Pudding	1549	1.0
...	...	...	...	...
324980	540873	Charishma's Gur Ki Roti With Khoya	6357	2100.0
325020	540939	Chettinad Chicken	78626	1800.0
325047	540977	INDIAN BASMATI With DRY FRUITS	2001004241	1800.0
325143	541143	Instant Rava Dhokla Recipe	2002835253	120.0
325260	541356	Masala Maggi Noodles in a Mug	2002835253	240.0

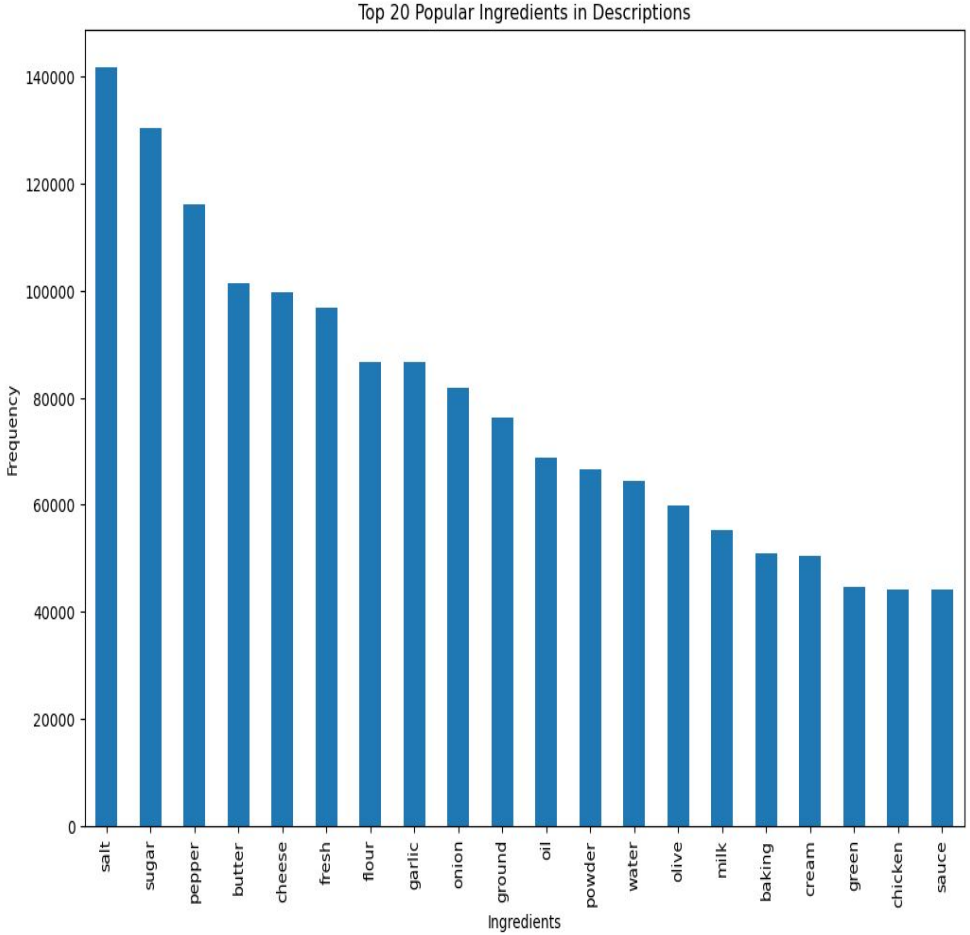
3. Categorizing based on level of difficulty: Easy= 30min, Medium= 30min-1hr, Hard=2hr +

```
# Display the first few rows of the DataFrame to see the new column
print(df[['Name', 'TotalTime', 'Difficulty']].head())
```

	Name	TotalTime	Difficulty
0	Low-Fat Berry Blue Frozen Dessert	89100	Difficult
1	Cabbage Soup	3000	Medium
2	Buttermilk Pie With Gingersnap Crumb Crust	4800	Medium
3	A Jad - Cucumber Pickle	1500	Easy
4	Butter Pecan Cookies	3840	Medium

4. Categorizing the cuisines: Italian, Mexican, Chinese, Indian, Japanese, Thai, Asian, Persian, American, Middle East.

(Results are shown on the left)



Top 20 Ingredients

This Graph shows another feature which can be used for recommending recipes based on popular ingredients. For now, this feature was added to the data.

Model 1:

Recommendation based on K-nearest Neighbor (Instance-based Learning)

```
# Example usage
recipe_id = 6
recommendations = get_recommendations(recipe_id)
print(recommendations)
```

```
# Display the recommended recipes
recommended_recipes = df.iloc[recommendations]
print(recommended_recipes)
```

	RecipeId	Name	AuthorId	CookTime	PrepTime	TotalTime	Description	Images	RecipeCategory	Keywords	...	oil	powder	water	olive	milk	baking	cream	green	chicken	sauce
190306	318937	Mom's Famous Oatmeal Cookies	437991	900.0	600	1500	71911	"https://img.sndimg.com/food/image/upload/w_55...	78	40044	...	0	0	0	0	1	0	0	0	0	0
88742	154332	Creamy Cheese Grits	282965	900.0	600	1500	122019	character(0)	31	65016	...	0	0	1	0	0	0	0	0	0	0
164575	277990	Creamy Rice and Broccoli	721861	1200.0	600	1800	122641	character(0)	284	76544	...	0	1	1	0	0	0	0	0	0	0
224781	374800	Spinach Pasta	517443	420.0	900	1320	90955	"https://img.sndimg.com/food/image/upload/w_55...	176	57542	...	0	0	0	0	0	0	0	0	0	0
31116	56134	Cheese Grits	19044	300.0	600	900	110188	character(0)	31	76802	...	0	1	1	0	0	0	0	0	0	0

[5 rows x 59 columns]

This output is based on recipes most similar to recipe_id =6.

Model 2:

Recommendation based Ingredients (content-based filtering):

This model is targeted to taste:

based on similarity between the ingredients used in the recipes. In this case, this example shows recipes that are similar to recipe_id= 6.

```python
# Function to get ingredient-based recommendations
def get_ingredient_recommendations(recipe_id, n_recommendations=5):
    # Get the similarity scores for the given recipe
    similarity_scores = ingredient_similarity_matrix[recipe_id]
    # Get the indices of the most similar recipes
    similar_indices = similarity_scores.argsort()[-(n_recommendations + 1):][::-1]
    # Exclude the input recipe itself from the recommendations
    similar_indices = similar_indices[similar_indices != recipe_id]
    return similar_indices[:n_recommendations]

# Example usage
recipe_id = 6
ingredient_recommendations = get_ingredient_recommendations(recipe_id)
print(ingredient_recommendations)

# Display the recommended recipes
recommended_recipes = df.iloc[ingredient_recommendations]
print(recommended_recipes)
```

```
[9999 3329 3336 3335 3334]
      RecipeId                          Name  AuthorId  CookTime  PrepTime  \
9999     20209  Chicken - Artichoke Sandwiches      5060    1080.0       600
3329      8985       Fried Macaroni and Cheese      2178    1800.0       900
3336      8997                     Turkey Loaf      9441    3000.0       300
3335      8995                Chocolate Mousse      9045     300.0       900
3334      8994                      Rose Lassi      6357       1.0       600

      TotalTime  Description  \
9999       1680       111894
3329       2700       132967
3336       3300       277774
3335       1200          943
3334        600       174282
```

Model 3:

Recommendation based on Reviews (collaborative filtering):

```
# Example usage
recommendations = recommend_recipes()
print(recommendations[['RecipeId', 'Name', 'Popularity']])
```

	RecipeId	Name	Popularity
785	2886	Best Banana Bread	2.000000
19529	35813	Oatmeal Raisin Cookies	1.620326
30070	54257	Yes, Virginia There is a Great Meatloaf	1.608887
13303	25690	Pancakes	1.505939
41235	73440	Beer Bread	1.457545

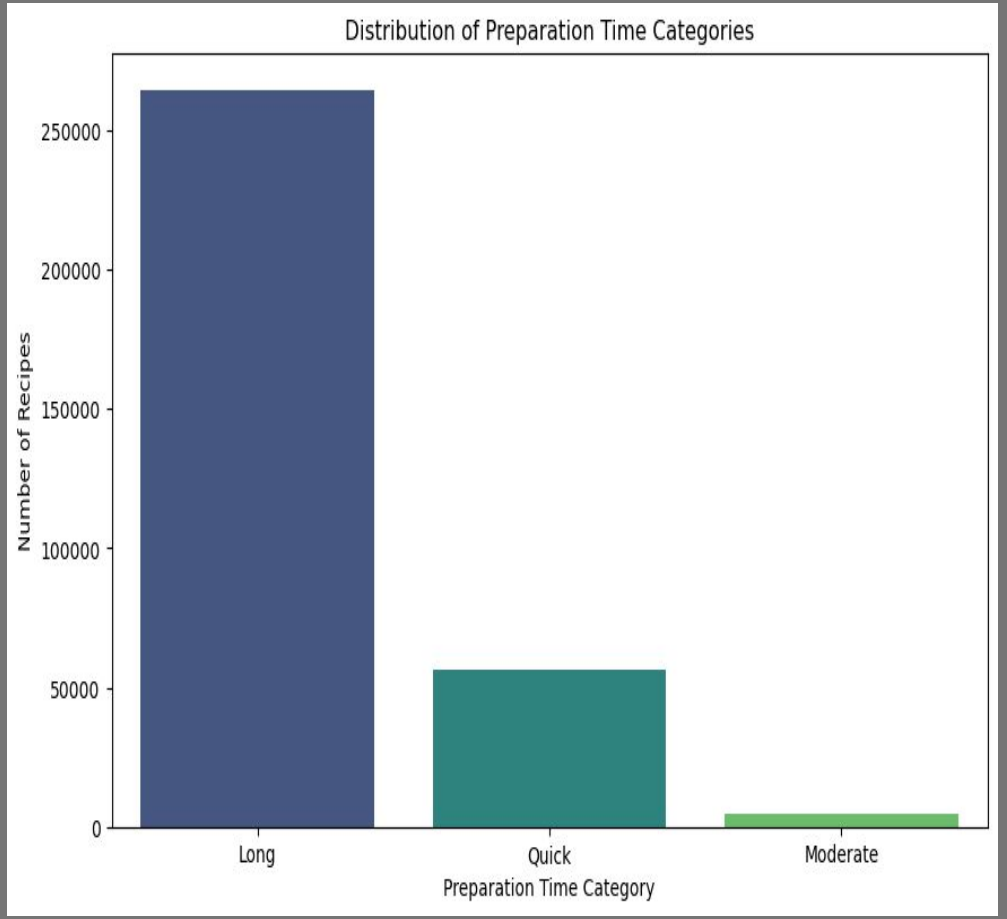
These recipes have similarly reviews to: recipe_id = 6

Future Improvements:

There were a lot of time consuming recipes in this data.

As the goal of this system is to provide ease especially for those who have a busy schedule, There should be more available short recipes.

- ➔ **Scrape and Add new data**
- ➔ **Perhaps create a feedback with user recommendation.**



Further Improvements:

Adding a “like” or thumbs up feature, to create a user based recommender

The recommendation system can be categorized as “easy”, “moderate”, and difficult. Adding more data is essential, and having shorter recipes as mentioned in previous slide is also very important.

- Adding new cuisines.
- Seasonal Recommendations
- More cuisine categories.