Recipe Recommender Assignment EDA

~ Yugansh Sood, Supritha Chakraborty and Sudeep Das

Task 1: Read the data

1. Read RAW_recipes.csv from S3 bucket.
2. Ensure each field has the correct data type.

```
# Task 01 Cell 1 out of 1

raw_recipes_df = spark.read.csv("s3://yuganshsood/RAW_recipes_cleaned.csv",inferSchema=True,header=True)
                  # argument 1, Add an argument to communicate to the compiler that there is a header in the raw data.
                  # argument 2, Add an argument to ask the complier to estimate the data types for all columns.
raw_recipes_df.show(5)
# Please forward the exact name of data frames and columns as suggested in the code.
# It will ensure that the assert commands function correctly.
```

```
+--------------------+------+-------+-------------+----------+--------------------+--------------------+-------+-----------
--------+--------------------+--------------------+-------------+
|                name|    id|minutes|contributor_id| submitted|                tags|           nutrition|n_steps|
steps|         description|         ingredients|n_ingredients|
+--------------------+------+-------+-------------+----------+--------------------+--------------------+-------+-----------
--------+--------------------+--------------------+-------------+
|arriba   baked wi...|137739|     55|        47892|2005-09-16|['60-minutes-or-l...|[51.5, 0.0, 13.0,...|     11|['make a ch
oice a...|autumn is my favo...|['winter squash',...|            7|
|a bit different  ...| 31490|     30|        26278|2002-06-17|['30-minutes-or-l...|[173.4, 18.0, 0.0...|      9|['preheat o
ven to...|this recipe calls...|['prepared pizza ...|            6|
|all in the kitche...|112140|    130|       196586|2005-02-25|['time-to-make', ...|[269.8, 22.0, 32....|      6|['brown gro
und be...|this modified ver...|['ground beef', '...|           13|
|  alouette  potatoes| 59389|     45|        68585|2003-04-14|['60-minutes-or-l...|[368.1, 17.0, 10....|     11|['place pot
atoes ...|this is a super e...|['spreadable chee...|           11|
|amish  tomato ket...| 44061|    190|        41706|2002-10-25|['weeknight', 'ti...|[352.9, 1.0, 337....|      5|['mix all i
ngredi...|my dh's amish mot...|['tomato juice', ...|            8|
+--------------------+------+-------+-------------+----------+--------------------+--------------------+-------+-----------
--------+--------------------+--------------------+-------------+
only showing top 5 rows
```

Task 2: Extract individual features from the nutrition column.

- the nutrition column is read as a string when it should be an array of float values. Each row in the nutrition column contains seven values. Each value represents nutrition information. Our task is to separate the array into seven individual columns.

Sample input: Nutrition column without the brackets.

id	nutrition
137739	51.5, 0.0, 13.0, 0.0, 2.0, 0.0, 4.0
31490	173.4, 18.0, 0.0, 17.0, 22.0, 35.0, 1.0
112140	269.8, 22.0, 32.0, 48.0, 39.0, 27.0, 5.0
59389	368.1, 17.0, 10.0, 2.0, 14.0, 8.0, 20.0
44061	352.9, 1.0, 337.0, 23.0, 3.0, 0.0, 28.0

Sample output: Nutrition column split into multiple

id	nutrition	calories	total fat (PDV)	sugar (PDV)	sodium (PDV)	protein (PDV)	saturated fat (PDV)	carbohydrates (PDV)
137739	51.5, 0.0, 13.0, 0.0, 2.0, 0.0, 4.0	51.5	0.0	13.0	0.0	2.0	0.0	4.0
31490	173.4, 18.0, 0.0, 17.0, 22.0, 35.0, 1.0	173.4	18.0	0.0	17.0	22.0	35.0	1.0
112140	269.8, 22.0, 32.0, 48.0, 39.0, 27.0, 5.0	269.8	22.0	32.0	48.0	39.0	27.0	5.0
59389	368.1, 17.0, 10.0, 2.0, 14.0, 8.0, 20.0	368.1	17.0	10.0	2.0	14.0	8.0	20.0
44061	352.9, 1.0, 337.0, 23.0, 3.0, 0.0, 28.0	352.9	1.0	337.0	23.0	3.0	0.0	28.0

- So we do string operations to remove square brakets by pyspark function to replace string character

- raw_recipes_df = (raw_recipes_df
- .withColumn('nutrition',(F.regexp_replace("nutrition","[\[\]]",""))))

- Hint: Visit this page to learn more about splitting columns

Task 3: Standardize the nutrition values.

- The nutritional values in absolute terms will have a lot of variation. For example, a recipe serving six people (recipe A) will have more sugar than a recipe meant to serve one person (recipe B). But that does not necessarily imply that the sugar per person in recipe A is more than in recipe B.
- Standardize the nutrition values. Convert the nutritional values to per 100 calories.

Sample input: Nutrition column without the brackets.

id	nutrition
137739	51.5, 0.0, 13.0, 0.0, 2.0, 0.0, 4.0
31490	173.4, 18.0, 0.0, 17.0, 22.0, 35.0, 1.0
112140	269.8, 22.0, 32.0, 48.0, 39.0, 27.0, 5.0
59389	368.1, 17.0, 10.0, 2.0, 14.0, 8.0, 20.0
44061	352.9, 1.0, 337.0, 23.0, 3.0, 0.0, 28.0

Sample output: Nutrition column split into multiple

id	nutrition	calories	total fat (PDV)	sugar (PDV)	sodium (PDV)	protein (PDV)	saturated fat (PDV)	carbohydrates (PDV)
137739	51.5, 0.0, 13.0, 0.0, 2.0, 0.0, 4.0	51.5	0.0	13.0	0.0	2.0	0.0	4.0
31490	173.4, 18.0, 0.0, 17.0, 22.0, 35.0, 1.0	173.4	18.0	0.0	17.0	22.0	35.0	1.0
112140	269.8, 22.0, 32.0, 48.0, 39.0, 27.0, 5.0	269.8	22.0	32.0	48.0	39.0	27.0	5.0
59389	368.1, 17.0, 10.0, 2.0, 14.0, 8.0, 20.0	368.1	17.0	10.0	2.0	14.0	8.0	20.0
44061	352.9, 1.0, 337.0, 23.0, 3.0, 0.0, 28.0	352.9	1.0	337.0	23.0	3.0	0.0	28.0

Sample Output:

All nutrition columns standardized to per 100 calories

id	total fat (PDV)	sugar (PDV)	sodium (PDV)	protein (PDV)	saturated fat (PDV)	carbohydrates (PDV)
137739	0.0	13.0	0.0	2.0	0.0	4.0
31490	18.0	0.0	17.0	22.0	35.0	1.0
112140	22.0	32.0	48.0	39.0	27.0	5.0
59389	17.0	10.0	2.0	14.0	8.0	20.0
44061	1.0	337.0	23.0	3.0	0.0	28.0

only showing top 5 rows

id	total_fat_per_100_cal	sugar_per_100_cal	sodium_per_100_cal	protein_per_100_cal	saturated_fat_per_100_cal	carbohydrates_per_100_cal
137739	0.0	25.24271844660194	0.0	3.883495145631068	0.0	7.766990291262136
31490	10.380623202758338	0.0	9.80392191371621	12.687428358926859	20.18454511647455	0.5767012890421299
112140	8.154188656554616	11.860638045897625	17.79095706884644	14.455152618437731	10.007413351226122	1.8532246946715039
59389	4.618310165205302	2.71665303835606	0.543330607671212	3.8033142536984843	2.173322430684848	5.43330607671212
44061	0.2833663976467306	95.49447600694822	6.517427145874804	0.8500991929401919	0.0	7.934259134108458

Task 4: Convert the tags column from a string to an array of strings.

- raw_recipes_df = (raw_recipes_df

.withColumn('tags', F.regexp_replace("tags","[\\[\\]\\']","")

)

.withColumn('tags', F.split("tags",", ")

)

)

```
raw_recipes_df.printSchema()
```

```
root
 |-- name: string (nullable = true)
 |-- id: integer (nullable = true)
 |-- minutes: integer (nullable = true)
 |-- contributor_id: integer (nullable = true)
 |-- submitted: string (nullable = true)
 |-- tags: array (nullable = true)
 |    |-- element: string (containsNull = true)
 |-- nutrition: string (nullable = true)
 |-- n_steps: integer (nullable = true)
 |-- steps: string (nullable = true)
 |-- description: string (nullable = true)
 |-- ingredients: string (nullable = true)
 |-- n_ingredients: integer (nullable = true)
 |-- calories: float (nullable = true)
 |-- total_fat_PDV: float (nullable = true)
 |-- sugar_PDV: float (nullable = true)
 |-- sodium_PDV: float (nullable = true)
 |-- protein_PDV: float (nullable = true)
 |-- saturated_fat_PDV: float (nullable = true)
 |-- carbohydrates_PDV: float (nullable = true)
 |-- total_fat_per_100_cal: double (nullable = false)
 |-- sugar_per_100_cal: double (nullable = false)
 |-- sodium_per_100_cal: double (nullable = false)
 |-- protein_per_100_cal: double (nullable = false)
 |-- saturated_fat_per_100_cal: double (nullable = false)
 |-- carbohydrates_per_100_cal: double (nullable = false)
```

Task 5: Read the second data file

- Read the RAW_interaction.csv and join this interaction level file with the recipe level data frame. The resulting data frame should have all the interactions.

- interaction_level_df = raw_ratings_df.join(raw_recipes_df,raw_ratings_df.recipe_id==raw_recipes_df.id,"inner"

)

Task 06: Create time-based features

```
interaction_level_df.schema["days_since_submission_on_review_date"].dataType == IntegerType()
```

```
True
```

```
(interaction_level_df.filter((interaction_level_df.user_id == 428885) & (interaction_level_df.recipe_id == 335241))
                             .select('days_since_submission_on_review_date').collect()[0][0]) == 77
```

```
True
```

```
(interaction_level_df.filter((interaction_level_df.user_id == 2025676) & (interaction_level_df.recipe_id == 94265))
                             .select('months_since_submission_on_review_date').collect()[0][0]) == 153.22580645
```

```
True
```

```
(interaction_level_df.filter((interaction_level_df.user_id == 338588) & (interaction_level_df.recipe_id == 21859))
                             .select('years_since_submission_on_review_date').collect()[0][0]) == 4.564516129166667
```

```
True
```

Summary

- As an ML engineer at food.com, our task is to design a recommender system that suggests recipes to users based on their choices and the recipe they are currently viewing. A successful recommender system can increase user engagement and lead to more business opportunities. The performance of the recommendation engine will directly impact the revenue generated by the website. However, building a recommender from scratch is time-consuming. In this assignment, you will explore data and create features to build the recommender.
- **Steps to approach the problem:**
- Create and launch an EMR Cluster on Amazon AWS
- Create and launch a Jupyter Notebook on top of this cluster
- Perform all the necessary tasks provided in task list
- **Task-List**
- Task 1: Read the data: Read RAW_recipes.csv from S3 bucket. Ensure each field has the correct data type.
- Task 2: Extract individual features from the nutrition column: Separate the array into seven individual columns to create new columns named calories, total_fat_PDV, sugar_PDV, sodium_PDV, protein_PDV, saturated_fat_PDV, and carbohydrates_PDV.
- Task 3: Standardize the nutrition values: Convert the nutritional values to per 100 calories.
- Task 4: Convert the tags column from a string to an array of strings: Convert the tags column from a string to an array of strings.
- Task 5: Read the second data file: Read the RAW_interaction.csv and join this interaction level file with the recipe level data frame. The resulting data frame should have all the interactions.
- Task 6: Create time-based features: Create features that capture the time passed between one review and the date on which the recipe was submitted. Use the review_date and the submitted columns after you join the two data files.