BDV3 PDF Report

General

During our research, we came across artificial intelligences that try to generate good recipes automatically. The results are not really good, but very funny. This gave us the idea to take a closer look at this topic. Our main focus is not to write such an artificial intelligence ourselves, but we would like to address it with the machine learning approach in this project.

Team Members

- Patrik Huber
- Claudia Leibetseder

Environment

- Docker Image: jupyter/pyspark-notebook
- Jupyter Notebook
- Python 3
- Apache Spark, MLlib, Matplotlib, Pandas (just for visualization)

Dataset

Our 'Recipe Ingredients and Reviews' dataset contains approximately 12,000 recipes with approximately 1,000,000 reviews. The dataset is publicly available at the following address:

https://www.kaggle.com/kanaryayi/recipe-ingredients-and-reviews?select=recipes.csv

The data set consists of two tables:

1. Recipes
2. Reviews

The Recipes table contains the following columns relevant for us:

1. Name
2. Total Time
3. Ingredients
4. RecipeID

The Reviews table contains the following columns relevant for us:

1. Rate
2. RecipeID

A rating can be made in whole numbers from 1 - 5.

Data Preparation

The method convert_time given here converts the following the given time specifications in the column TotalTime, PrepareTime, CookTime into a uniform minute format. The following formats are possible:

1. 10m
2. 1h
3. 1d
4. 1h 10m
5. 1d 10m
6. 1d 10h
7. 1d 10h 10m
8. X

The method convert_time therefore breaks down the time specification into its individual components and converts them. The only exception is the X, which is replaced by 0 because no further information is available.

In the Recipes table there is an Ingredients column. This contains the ingredients as a serialized array and is therefore not good for analysis. Therefore, a single row is created for each ingredient and the result is saved as a DataFrame.

The Reviews table is grouped to get to the cumulative ratings. This means that for each recipe the total rating is calculated and the number of ratings is summed up.

Hard Facts

- recipes.count(): 12351
- reviews.count(): 1563566
- ingredients.count(): 102517

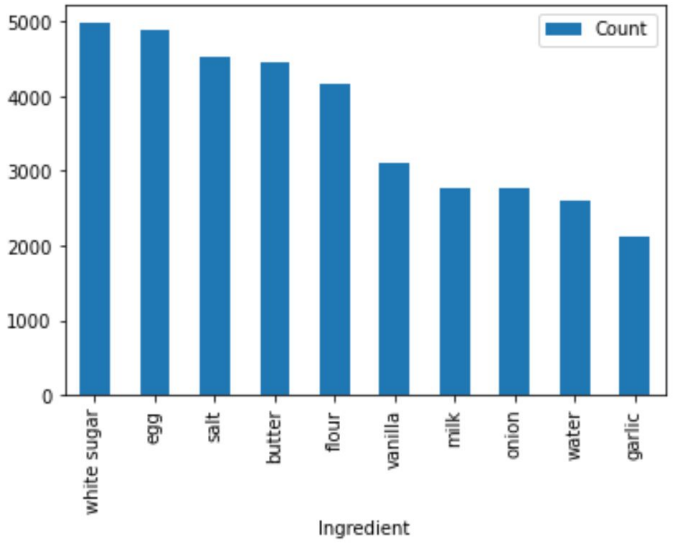
Research Questions and findings during analysis

1. Which 10 ingredients are used most often?

Assumption: At the beginning, we had no concrete assumptions about the expected result of this question. When the result was that white sugar was the most common ingredient, this sounded like the only plausible solution, since sugar is the food that needs to be reduced in daily consumption due to its overconsumption.

Approach/Result: Due to our already prepared table Ingredients, it is an easy task to group them by the column Ingredient count them and sort them by total number.

	Ingredient	Count
0	white sugar	4986
1	egg	4880
2	salt	4516
3	butter	4456
4	flour	4158
5	vanilla	3111
6	milk	2767
7	onion	2763
8	water	2606
9	garlic	2116



Summary: The result reminded us very much of a German children's song:

Backe, backe Kuchen,
Der Bäcker hat gerufen.
Wer will guten Kuchen backen,
der muss haben sieben Sachen,
Eier und Schmalz,
Zucker und Salz,
Milch und Mehl,
Safran macht den Kuchen gehl!
Schieb, schieb in'n Ofen 'nein.

Except for saffron, we have all the ingredients in our list, this may indicate that many cake recipes are used. However, it can be seen from onions and garlic, of course, sour recipes are also represented.

2. Which three ingredients are used most often in the top 10 rated recipes?

Assumption: According to the result of the first question, our assumption was that the top 10 rated recipes would also have white sugar as the most common ingredient. The following evaluation shows whether we are correct in this assumption.

Approach/Result: First, the top ten recipes with names ingredients, etc. are filtered out. They are joined with the Ingredients table so that the frequency of the ingredients is only counted for the top ten recipes and sorted afterwards.

	Ingredient	Count
0	white sugar	7
1	flour	5
2	butter	4

Summary: It can be seen that the ten most common ingredients hardly differ from the ingredients in the ten top-rated recipes.

3. What is the average total time needed for a recipe?

Assumption: The assumption here was that the time should settle at about 1h 30m on average. We assumed that most recipes will take about 1h to not be too long and there will be a few exceptional cases with about 5h.

Approach/Result: Here the already prepared Recipes with the converted time data are used again. In this case, the entries that have 0 for the TotalTime are filtered out in order not to falsify the result. The result gets grouped and then the average value is taken. With the help of Python extensions, the result is converted back into the original time format and output textually.

The median is also calculated in order to exclude any falsification of the result due to outliers. The result shows that, as we will show later, there are outliers with a huge TotalTime. The median shows that in reality 50% of the recipes need less than or equal 55 minutes. In comparison, the average is 2h 30m. This clearly shows the outliers.

```
average_time(format HH:MM): 02h 30m
median_time(format HH:MM): 00h 55m
```

Summary: It is important to list both the average time and the median here. Because here it becomes clear that there is a strong deviation and this points to some outliers that last much longer. This assumption was based on the very high average total time, which is far from the 50% that last less than 55 minutes. We got the explanation for this irregularity in the next task.

4. How good are the 5 recipes with the longest preparation time rated?

Assumption: Our questioning was aimed at the fact that our assumption was that if something takes a very long time, then a meal could be particularly good or one could also be annoyed that these recipes take so long and give a rather negative rating.

Approach/Result: For this evaluation we use two already existing tables. First, the Recipes table is sorted by the TotalTimeConverted. Then the five longest recipes are joined with the ratings, sorted by the average rating and output in table form.

	RecipeID	RecipeName	AvgRate	CountRate	TotalTime	TotalTimeConverted
0	7900	Thirty Day Friendship Cake Recipe	4.525424	59.0	30 d 2 h	43320
1	19810	Homemade Vanilla Recipe	4.300000	30.0	20 d 20 h	30000
2	24056	Rock Candy Recipe	2.833333	30.0	10 d 10 h	15000
3	8470	Noel Fruitcake Recipe	NaN	NaN	30 d 4 h	43440
4	25544	Pickled Garlic Recipe	NaN	NaN	21 d 1 h	30300

Summary: It can be seen that there are recipes that have no rating with two of our five longest recipes this is unfortunately the case. Our initial assumption turned out not to be thought far enough. Because a long TotalTime does not automatically mean that a recipe becomes many times more elaborate. Because often a long TotalTime includes steps like preserving cherries in sugar syrup for more than two weeks, this does not mean much more effort. Therefore, in conclusion, as in the other recipes, ratings can be found in different ways.

5. Can we predict how good the rating will be based on the ingredients and preparation time?

Assumption: We were very confident to see a correlation between ingredients used and the rating as well as the TotalTime and the rating. For the ingredients, we assumed that there were some popular foods and that this would result in a better rating.

Approach/Result Linear Regression: In the first step the required tables are merged and the textual columns are converted to numeric columns using StringIndexer. The data is also cleaned. This means that the entries that do not have a total for the time will be sorted out. The next step was to use the linear regression implementation of PySpark and the evaluator for the model.

Approach/Result Random Forest: The already transformed data can be used from the linear regression as well in this model training. Due to the not so good results of our first experiment, we still wanted to try if maybe there is no linear relationship in our data. Therefore, we wanted to try a different approach to see if there was a different relationship in our data. For this we decided to use RandomForest. This is a combination of several decision trees.

Summary: Our assumptions were completely disproved, which became apparent after the linear regression. The results were very poor although the prediction was not far from the actual average ratings. We noticed for the first time at this point that the standard deviation in the

ratings is not very high. To be precise, it is 0.301017, indicating irregular data. That means, there are more very good recipes than bad recipes. We tried to deal with this problem using Random Forest. However, this also failed, since no linear correlation could be found, but also no more complex correlation.

After these failures we had the following idea. It is possible to transform the data set by a RollUp in such a way that the 100 most frequent ingredients are used as a feature vector. This vector is filled as follows. If an ingredient is not used, 0 is entered for it. If it is used, the quantity is standardized to grams and brought down to one person. From this feature vector it is possible to try to make the prediction. As you can see here, this transformation is very complex and would have gone beyond the scope of this project, so we decided not to implement it. Nevertheless, it was important to us to mention it here.