The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)
ISSN: 2602-3199

The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 2024

Volume 28, Pages 438-447

ICBASET 2024: International Conference on Basic Sciences, Engineering and Technology

Food Recommendation System for a Healthy Liver Using Machine Learning

Shyti Bederiana
University of Elbasan "Aleksander Xhuvani"

Stergu Argesta
University of Elbasan "Aleksander Xhuvani"

Valera Dhurata
University of Elbasan "Aleksander Xhuvani"

Papajani Blerina
University of Elbasan "Aleksander Xhuvani"

Abstract: Nowadays, the work routine, the problems we face, make us not pay proper attention to healthy eating. As a result, people get sick. We have created through mathematical knowledge (statistics, probability, linear algebra, geometry), combining the concepts of Machine Learning (Content Based Filtering, TD IDF Vectorizer Algorithm, Conditional Independence, Count Vectorization) with Python language, a Recommendation System for all people suffering from liver. Liver is one of the most important organs of our body, because it makes 500 essential functions in the organism. But we have to be aware about the types of food that we use. Also, we must be very careful with our lifestyle and diet on foods. The aim of this article is to recommend the most appropriate foods for a healthy liver. This article will help the people who suffer from liver disease to check the best foods maintaining the liver in a good physical condition.

Keywords: Recommendation system, Liver, Foods, Health, Machine learning.

Introduction

Nowadays, the majority of individuals have become addicted to social networks and as a result lead a sedentary life. Some other individuals get lost in the routine and workload. Some people get tired doing physical work. The way some individuals eat, how they build their lifestyle, how they choose to live, directly affects the health of the organism. The most solid and sensitive organ to the diet and the types of food we consume is the liver. This article presents a simple recommendation system, using the subfields of mathematics (Linear Algebra, Statistics, Probability, Geometry) and the knowledges of Machine Learning as a subfield od Artificial Intelligence with Python language. The main purpose of this article is to suggest people who suffer from liver diseases, to choose best foods to maintain the liver in the good conditions. This Recommendation System can be useful, also, for people of all ages, who has not time to search in the pages of internet about healthy foods. This article also helps people to prevent liver disease. The article is organized in some sections.

The Relationship between Mathematics, Machine Learning and Recommendations Systems

There is a hard relationship between Mathematics, Machine Learning and Recommendation Systems (Li et al., 2021). We have created a code based on the statistical data collected in a database from the website for different

- Selection and peer-review under responsibility of the Organizing Committee of the Conference

types of food products based on the nutritional values and essential chemical elements that the body needs to function properly. There are three types of Recommendation Systems techniques:

- Content-based filtering recommendation
- Collaborative filtering system
- Hybrid technique

The first technique is used in this article to make the food recommendation and to emphasize the relationship between the science of Mathematics, RSs and ML.

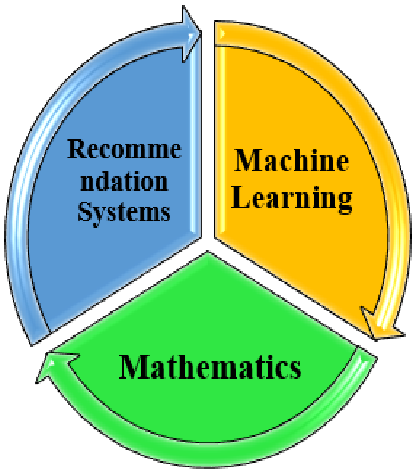

Figure 1. The triple mathematics, machine learning and recommendation Systems

Data Implementing in the Code

First of all, we have taken data from the page “U.S. DEPARTMENT AGRICULTURE” (FoodData Central (usda.gov)). We have saved them in CSV format in excel with the title “Best_food_for_kidney”. Then, we have implemented these data in the code using Python language, in the Spyder 3.11 of Anaconda Navigator package. Based on the amount of nutrient that the foods contain we have programmed the code to recommend the best foods for a healthy liver. To make the code, we have used the libraries of Python, such as: Numpy, Pandas, Matplotlib, Scikit-learn. The Matplotlib library is made for creating 2D arrays, making different plots (Hunter et al., 2017).

Table 1. The foods with their nutrition

	Foods	Ingridients	Protein	Fiber	Calories	Fat	Charbohic	Sugar	Vitamin
2	Avocado	[{"Magnesium": 58, "Potasium": 485, "Phosphorus": 48.67, "So	2	6.7	160	14.66	8.53	0.66	[{"Vitamin": "C", "Vitamin": "A"}]
3	Bluberries	[{"Magnesium": 6.5, "Potasium": 77, "Phosphorus": 14.38, "Soc	0.74	2.4	57	0.33	14.49	9.96	[{"Vitamin": "C", "Vitamin": "A"}]
4	Broccoli	[{"Magnesium": 21, "Potasium": 316, "Phosphorus": 73, "Sodiu	2.82	2.6	34	0.37	6.64	1.7	[{"Vitamin": "C", "Vitamin": "A"}]
5	Butter	[{"Magnesium": 3, "Potasium": 24, "Phosphorus": 508, "Sodiun	0.85	0	717	81.11	0.06	0.06	[{"Vitamin": "A", "Vitamin": D}]
6	Cabbage	[{"Magnesium": 12, "Potasium": 246 , "Phosphorus": 26, "Sodiu	1.44	2.3	24	0.12	5.58	3.58	[{"Vitamin": "C", "Vitamin": "A"}]
7	Beef	[{"Magnesium": , "Potasium": 315, "Phosphorus": , "Sodiun	26.33	0	288	19.54	0	0	[{"Vitamin":}]
8	Couliflow	[{"Magnesium": 15, "Potasium": 303, "Phosphorus": 44, "Sodiu	1.98		25	0.1	5.3	2.4	[{"Vitamin": "C", "Vitamin": "A"}]
9	Apples	[{"Magnesium": 5, "Potasium": 107, "Phosphorus": 11, "Sodiun	0.26	2.4	52	0.17	13.81	10.39	[{"Vitamin": "C", "Vitamin": "A"}]
10	Bananas	[{"Magnesium": 32, "Potasium": 358, "Phosphorus": 22 , "Sodiu	1.09	2.6	89	0.33	22.84	12.33	[{"Vitamin": "C", "Vitamin": "A"}]
11	Onions	[{"Magnesium": 15.27 , "Potasium": 144, "Phosphorus": 29, "So	0.92	1.4	42	0.08	10.11	4.28	[{"Vitamin": "C", "Vitamin": "A"}]
12	Oranges	[{"Magnesium": 11, "Potasium": 181, "Phosphorus": 22, "Sodiu	0.94	2.4	47	0.12	11.75	9.35	[{"Vitamin": "C", "Vitamin": "A"}]
13	Pineaple	[{"Magnesium": 12, "Potasium": 115, "Phosphorus": 8, "Sodiun	0.54	1.4	48	0.12	12.63	9.26	[{"Vitamin": "C", "Vitamin": "A"}]
14	Pomegran	[{"Magnesium": 12, "Potasium": 259, "Phosphorus": 95.2 , "Sod	0.95	0.6	68	0.3	17.17	16.57	[{"Vitamin": "C", "Vitamin": "A"}]
15	Strawberr	[{"Magnesium": 22, "Potasium": 153, "Phosphorus": 24 , "Sodiu	0.67	2	32	0.3	7.68	4.66	[{"Vitamin": "C", "Vitamin": "A"}]
16	White Bre	[{"Magnesium": 26, "Potasium": 252, "Phosphorus": 103, "Sodi	9.7	6.09	246	4.2	46.1	5.57	[{"Vitamin": "C", "Vitamin": "A"}]
17	Corn	[{"Magnesium": 42.60 , "Potasium": 270, "Phosphorus": 13, "So	3.22	2.7	86	1.18	19.02	3.22	[{"Vitamin": "C", "Vitamin": "A"}]
18	Lettuce	[{"Magnesium": 12, "Potasium": 141, "Phosphorus": 28, "Sodiu	0.9	1.2	14	0.14		1.76	[{"Vitamin": "C", "Vitamin": "A"}]
19	Olives	[{"Magnesium": 6.33, "Potasium": 1, "Phosphorus": 28, "Sodiu	0	0	884	100	0	0	[{"Vitamin": }]
20	Artichoke	[{"Magnesium": 0, "Potasium": 23.4, "Phosphorus": 107.01, "S	3	2	30	0	6	2	[{"Vitamin": "C", "Vitamin": "B6", "Vitamin": "A"}]
21	Spinach	[{"Magnesium": 79, "Potasium": 558, "Phosphorus": 49, "Sodiu	2.86	2.2	23	0.39	3.63	0.42	[{"Vitamin": "C", "Vitamin": "A"}]
22	Mushroon	[{"Magnesium": 9 , "Potasium": 318, "Phosphorus": 87, "Sodiur	3.09	1	22	0.34	3.28	1.65	[{"Vitamin": "C", "Vitamin": "D", "Vitamin": "A"}]
23	Carrots	[{"Magnesium": 12, "Potasium": 320 , "Phosphorus": 35 , "Soc	0.93	2.8	41	0.24	9.58	4.54	[{"Vitamin": "C", "Vitamin": "A"}]
24	Blackberri	[{"Magnesium": 20, "Potasium":162, "Phosphorus": 22, "Sodiur	1.39	5.3	43	0.49	9.61	4.88	[{"Vitamin": "C", "Vitamin": "A"}]
25	Fish	[{"Magnesium": 76, "Potasium": 351, "Phosphorus": 689, "Sodi	17.76	0	84	0.92	0	0	[{"Vitamin": "C", "Vitamin": "A"}]
26	Milk	[{"Magnesium": 12, "Potasium": 153, "Phosphorus": 85, "Sodiu	3.39	0	52	2.06	4.86	5.32	[{"Vitamin": "A"}]
27	Cherriesh	[{"Magnesium": 11 , "Potasium": 173, "Phosphorus": 16, "Sodiu	1	1.6	50	0.3	12.18	8.49	[{"Vitamin": "C", "Vitamin": "A"}]
28	White Ric	[{"Magnesium": 29.62, "Potasium": 35 , "Phosphorus": 33, "Sod	2.66	0.4	129	0.28	27.9	0.05	[{"Vitamin":}]
29	Potatoes	[{"Magnesium": 23, "Potasium": 317, "Phosphorus": 38, "Sodiu	1.66	1.7	104	2.4	19.36	0.82	[{"Vitamin": "C", "Vitamin": "A"}]
30	Tomatoes	[{"Magnesium": 194, "Potasium": 237, "Phosphorus": 356, "Sod	0.88	1.2	18	0.2	3.92	2.63	[{"Vitamin": "C", "Vitamin": "A"}]
31	Cucumber	[{"Magnesium": 15.50, "Potasium": 136 , "Phosphorus": 24, "So	0.59	0.7	12	0.16	2.16	1.38	[{"Vitamin": "C", "Vitamin": "A"}]
32	Coconuts	[{"Magnesium": 32, "Potasium": 358, "Phosphorus": 113.02, "S	3.33	9	354	33.49	15.23	6.23	[{"Vitamin": "C", "Vitamin": "A"}]
33	Cheese	[{"Magnesium": 19, "Potasium": 187, "Phosphorus": 91 , "Sodiu	22.21	0	350	26.91	4.71	3.54	[{"Vitamin": "C", "Vitamin": "A"}]
34	Green Tea	[{"Magnesium": 0 , "Potasium": 21 , "Phosphorus": 290, "Sodiu	0	0	2	0	0.47	0	[{"Vitamin"}]
35	Green Pea	[{"Magnesium": 39, "Potasium": 244, "Phosphorus": 187.2, "So	5.42	5.1	81	0.4	14.46	5.67	[{"Vitamin": "C", "Vitamin": "A"}]

Female_aged_30_to_70

The database contains the names of foods, the vitamins of each of them, the amount of fiber, saturated fat, the ingredients. We have calculated the amount of saturated fat and the amount for each ingredient for 100 grams of food. The main ingredients are: magnesium, potassium, phosphorus. There is a screenshot with thirty-five foods with their own nutrition.

The Phosphorus

The phosphorus helps maintain normal pH levels in extracellular fluid. It plays crucial roles in the body, including being a component of bones, teeth, *DNA*, *RNA*, and cell membranes. Phosphorus is also involved in energy production and enzyme activation. Three classes of *NA/Pi* contraspoters are encoding by the cDNAs(Takeda et al., 2004). We have made a table for these main ingredients classified in two huge group based on gender: male and female. Each of these groups are classified in three subgroups based on the age: male aged 19-30, male aged 30-70, male over 70, female aged 19-30, female aged 30-70 and female over 70. For each group we have define the type of foods recommended for a healthy liver. To make the table we have collected data in the page "U.S. FOOD & DRUG" (Using the Nutrition Facts Label: For Older Adults | FDA). The table below will help us to calculate the amount of some nutrition based on the daily value (DV)

Table 2. The daily values of some nutrition for a healthy body

Nutrition	**Male aged** 19-30	**Male aged** 30 to 70	**Male over** 70	**Female aged** 19-30	**Female aged** 30 to 70	**Female over** 70
Potassium	3400 mg	4700 mg	4700 mg	2600 mg	2600 mg	2600 mg
Magnesium	400 mg	420 mg	400 mg	310 mg	350 mg	320 mg
Phosphorus	1250 mg	1596 mg	580-1000 mg	580-1000 mg	1189 mg	1189mg
Sodium	1500 mg	1500-2300 mg	1200 mg	1300 mg	1500-2300 mg	1200 mg

Male and female aged 30-70 years. The amount of nutritions in the artichoke;

Magnesium

$$15\%\ of\ 350\ mg = \frac{15}{100}\ of\ 350\ mg = (350 : 100) * 15 = 3.5 * 15 = 52.5\ mg$$

Potassium

$$9\%\ of\ 2600\ mg = \frac{9}{100}\ of\ 2600\ mg = (2600\ : 100) * 9 = 2.6 * 9 = 23.4\ mg$$

Phosphorus

$$9\%\ of\ 1189\ mg = \frac{9}{100}\ of\ 1189\ = (1189 : 100) * 9 = 107.01\ mg$$

Methods and Techniques

Content Based Filtering is a powerful method of Machine Learning to make Recommendation Systems (Son & Kim, 2017). This method is a type of Information Filtering System. The code uses all the data implemented from the external dataset in CSV format in excel and filters them (Zhang et al., 2023). The input data such as all the values of ingredients, vitamins, protein, fat, carbohydrates, fibers are trained by Content Based Filtering and give as outputs the recommender types of foods. The concept of Content Based Filtering means that the same data of a collection or Database are usen and usen once again, depends on the preferences of different users (Swathi et al., 2007).

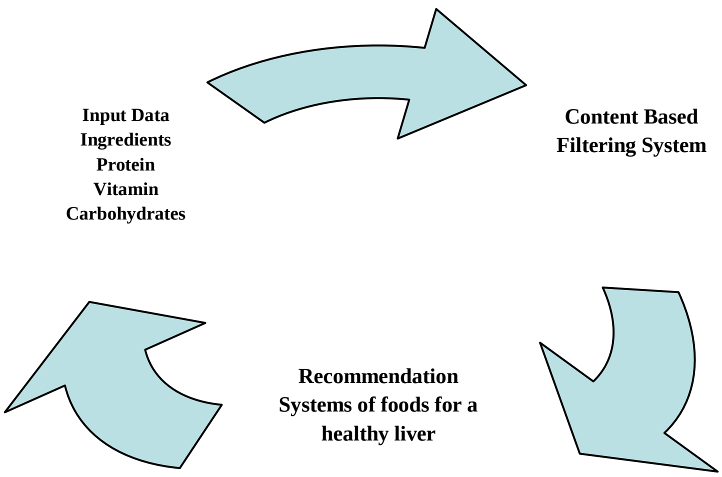

Figure 2. The process of recommendation systems of food

Applying the Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import pyplot as plt
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
hp = pd.read_csv('Female_aged_30_to_70.csv')
print(hp)
```

We choose two columns with data form the database and implement their data in the code.

```
hp ['Foods'] = hp ['Foods'].str.split('/')
hp ['Foods'] = hp['Foods'].fillna("").astype('str')
foods = pd.read_csv('Female_aged_30_to_70.csv', encoding='latin-1', usecols=['Foods', 'Ingridients'])
print(foods)

plt.figure(figsize=(20,10))
plt.bar(hp['Foods'].head(101), hp['Ingridients'].head(80))
term_freq = TfidfVectorizer(analyzer='word', ngram_range=(1, 2), min_df=1, encoding='utf-8',
                    lowercase=True, stop_words='english')
tfidf_matrix = term_freq.fit_transform(foods['Ingridients'])
tfidf_matrix.shape
```

Results and Discussion

First of all, we implement all the data in the code. There is a matrix with 34 rows and 14 columns.

```
Foods  ...                              Vitamin
0       Avocado  ...              [{"Vitamin": "C", "Vitamin": "A"}]
1     Bluberries  ...             [{"Vitamin": "C", "Vitamin": "A"}]
2      Broccoli  ...              [{"Vitamin": "C", "Vitamin": "A"}]
3       Butter  ...               [{"Vitamin": "A", "Vitamin": D}]
4      Cabbage  ...               [{"Vitamin": "C", "Vitamin": "A"}]
5         Beef  ...                          [{"Vitamin":}]
6    Couliflower  ...             [{"Vitamin": "C", "Vitamin": "A"}]
7       Apples  ...               [{"Vitamin": "C", "Vitamin": "A"}]
8      Bananas  ...               [{"Vitamin": "C", "Vitamin": "A"}]
9       Onions  ...               [{"Vitamin": "C", "Vitamin": "A"}]
10      Oranges  ...              [{"Vitamin": "C", "Vitamin": "A"}]
11     Pineaple  ...              [{"Vitamin": "C", "Vitamin": "A"}]
12   Pomegranates  ...            [{"Vitamin": "C", "Vitamin": "A"}]
13   Strawberries  ...            [{"Vitamin": "C", "Vitamin": "A"}]
14   White Bread  ...             [{"Vitamin": "C", "Vitamin": "A"}]
15        Corn  ...               [{"Vitamin": "C", "Vitamin": "A"}]
16      Lettuce  ...              [{"Vitamin": "C", "Vitamin": "A"}]
17       Olives  ...                         [{"Vitamin": }]
18    Artichoke  ...  [{"Vitamin": "C", "Vitamin": "B6", "Vitamin": ...
19      Spinach  ...              [{"Vitamin": "C", "Vitamin": "A"}]
20    Mushrooms  ...  [{"Vitamin": "C", "Vitamin": "D", "Vitamin": "...
21      Carrots  ...              [{"Vitamin": "C", "Vitamin": "A"}]
22   Blackberries  ...            [{"Vitamin": "C", "Vitamin": "A"}]
23        Fish  ...               [{"Vitamin": "C", "Vitamin": "A"}]
24        Milk  ...                       [{"Vitamin": "A"}]
25    Cherriesh  ...              [{"Vitamin": "C", "Vitamin": "A"}]
26   White Rice  ...                         [{"Vitamin":}]
27     Potatoes  ...              [{"Vitamin": "C", "Vitamin": "A"}]
28     Tomatoes  ...              [{"Vitamin": "C", "Vitamin": "A"}]
```

```
29     Cucumber  ...          [{"Vitamin": "C", "Vitamin": "A"}]
30     Coconuts  ...          [{"Vitamin": "C", "Vitamin": "A"}]
31       Cheese  ...          [{"Vitamin": "C", "Vitamin": "A"}]
32    Green Tea  ...                            [{"Vitamin"}]
33   Green Peas  ...          [{"Vitamin": "C", "Vitamin": "A"}]
```

[34 rows x 14 columns]

The second result of the code:

```
Foods                          Ingridients
0       Avocado  [{"Magnesium": 58, "Potasium": 485, "Phosphoru...
1    Bluberries  [{"Magnesium": 6.5, "Potasium": 77, "Phosphoru...
2      Broccoli  [{"Magnesium": 21, "Potasium": 316, "Phosphoru...
3        Butter  [{"Magnesium": 3, "Potasium": 24, "Phosphorus"...
4       Cabbage  [{"Magnesium": 12, "Potasium": 246 , "Phosphor...
5          Beef  [{"Magnesium":     , "Potasium": 315, "Phospho...
6   Couliflower  [{"Magnesium": 15, "Potasium": 303, "Phosphoru...
7        Apples  [{"Magnesium": 5, "Potasium": 107, "Phosphorus...
8       Bananas  [{"Magnesium": 32, "Potasium": 358, "Phosphoru...
9        Onions  [{"Magnesium": 15.27 , "Potasium": 144, "Phosp...
10      Oranges  [{"Magnesium": 11, "Potasium": 181, "Phosphoru...
11     Pineaple  [{"Magnesium": 12, "Potasium": 115, "Phosphoru...
12  Pomegranates  [{"Magnesium": 12, "Potasium": 259, "Phosphoru...
13  Strawberries  [{"Magnesium": 22, "Potasium": 153, "Phosphoru...
14  White Bread  [{"Magnesium": 26, "Potasium": 252, "Phosphoru...
15         Corn  [{"Magnesium": 42.60 , "Potasium": 270, "Phosp...
16      Lettuce  [{"Magnesium": 12, "Potasium": 141, "Phosphoru...
17       Olives  [{"Magnesium": 6.33, "Potasium": 1,  "Phosphor...
18    Artichoke  [{"Magnesium": 0,  "Potasium": 23.4, "Phosphor...
19      Spinach  [{"Magnesium": 79, "Potasium": 558, "Phosphoru...
20    Mushrooms  [{"Magnesium": 9 , "Potasium": 318, "Phosphoru...
21      Carrots  [{"Magnesium":  12, "Potasium": 320   , "Phosp...
22  Blackberries  [{"Magnesium": 20, "Potasium":162, "Phosphorus...
23         Fish  [{"Magnesium": 76, "Potasium": 351, "Phosphoru...
24         Milk  [{"Magnesium":  12, "Potasium": 153, "Phosphor...
25     Cherriesh  [{"Magnesium": 11 , "Potasium": 173, "Phosphor...
26   White Rice  [{"Magnesium": 29.62, "Potasium": 35 , "Phosph...
27     Potatoes  [{"Magnesium": 23, "Potasium": 317, "Phosphoru...
28     Tomatoes  [{"Magnesium": 194, "Potasium": 237, "Phosphor...
29     Cucumber  [{"Magnesium": 15.50, "Potasium": 136 , "Phosp...
30     Coconuts  [{"Magnesium": 32, "Potasium": 358, "Phosphoru...
31       Cheese  [{"Magnesium": 19, "Potasium": 187, "Phosphoru...
32    Green Tea  [{"Magnesium":  0 , "Potasium": 21 , "Phosphor...
33   Green Peas  [{"Magnesium": 39, "Potasium": 244, "Phosphoru...
```

There is time to plot the graph using the code as below.

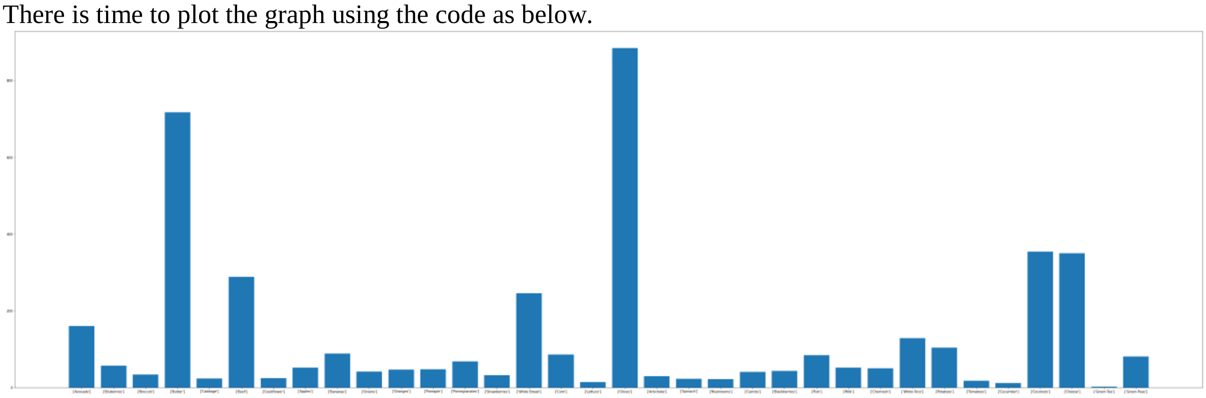

Figure 2. The distribution of calories for different foods

```
plt.figure(figsize=(65,10))
plt.bar(hp['Foods'].head(101), hp['Calories'].head(70))
term_freq = TfidfVectorizer(analyzer='word', ngram_range=(1, 2), min_df=1, encoding='utf-8',
                lowercase=True, stop_words='english')
tfidf_matrix = term_freq.fit_transform(foods['Calories'])
tfidf_matrix.shape
```

The chart above shows the dependence of two variables. The variable x is for "Foods" and the variables y is for "Calories".

TD IDF (Term Frequency – Inverse Document Frequency)

How the mathematics' knowledges help Machine Learning to create Recommendation System?

TD IDF is the product of two statistics term frequency with inverse term frequency. IDF defines the measure of quantify of a term across a collection of terms in a database. This technique uses some knowledge of Linear Algebra and Calculus to calculate how important is a term within a document (Karabiber, 2022). It expresses the logarithm of the ratio between the number of terms appears in the document with the total number of terms. Mathematically the formula is written as below:

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} ft', d}$$

Which is the role of each variable in the formula above?

- $t \rightarrow term$
- $d \rightarrow document$
- $D \rightarrow set\ of\ documents$

There are other formulas of TD IDF

$$TF(d,t) = \begin{Bmatrix} 0 & f\,req(d,t) = 0 \\ 1 + \log(f\,req\,(d,t)) & f\,req(d,t) \neq 0 \end{Bmatrix}$$

$$IDF\ (t) = N/\,df(t) = N/N(t)$$
$$IDF\ (t) = log(N/\,df(t))$$
$$IDF\ (t) = log\,(1 + |d|)/(dt)$$
$$IDF\ as\ ID\ (t) = log\,[\,n\,/\,(df(t) + 1)\,])$$

When we implement the mathematical formula in the code, it takes another form, using Python language. There are some steps (Wang, 2018):

- Preproccessing the data
- Defining a function to calculate "Term Frequency"
- Defining a function to calculate "Inverse Document Frequency"
- Combining the TD IDF
- Applying the TD IDF model in our text

There is the code for the TD IDF technique

```
term_freq = TfidfVectorizer(analyzer='word', ngram_range=(1, 2), min_df=1, encoding='utf-8',
                lowercase=True, stop_words='english')
tfidf_matrix = term_freq.fit_transform(foods['Ingridients'])
tfidf_matrix.shape
print(tfidf_matrix)
cosSim = cosine_similarity(tfidf_matrix, tfidf_matrix)
cosSim[:4, :4]
print(cosSim)
```

```
titleHotels = foods['Foods']
print(titleHotels)
signIndication = pd.Series(foods.index, index=foods['Foods'])
print(signIndication)
```

Creating a function called fd_recommendations and take some attributes

```
def fd_recommendations(Foods):
    hx = signIndication[Foods]
    scores_similarity = list(enumerate(cosSim[hx]))
    scores_similarity = sorted(scores_similarity, key=lambda x: x[1], reverse=True)
    scores_similarity = scores_similarity[1:21]
    foods_signIndication = [i[0] for i in scores_similarity]
    return titleHotels.iloc[foods_signIndication]
  print(fd_recommendations('Avocado').head(10))
```

The output of the code using the technique TD-IDF

The first number means the “Document Index”, while the second number means “Word Index”.

```
(0, 224)      0.2499995926057404
(0, 177)      0.2499995926057404
(0, 255)      0.2499995926057404
(0, 363)      0.11147348132417445
(0, 195)      0.2499995926057404
(0, 161)      0.2499995926057404
(0, 302)      0.2499995926057404
(0, 163)      0.2499995926057404
(0, 341)      0.2499995926057404
(0, 183)      0.2499995926057404
(0, 278)      0.2499995926057404
(0, 21)       0.1526409237963958
(0, 222)      0.07470797316296354
(0, 176)      0.2499995926057404
(0, 238)      0.06660618464588744
(0, 345)      0.06472982641585853
(0, 194)      0.2499995926057404
(0, 160)      0.2499995926057404
(0, 282)      0.06472982641585853
(0, 162)      0.2499995926057404
(0, 312)      0.06472982641585853
(0, 182)      0.2499995926057404
(0, 262)      0.06472982641585853
(1, 100)      0.276241039860636
(1, 246)      0.276241039860636
:     :
(32, 345)     0.08681253862380181
(32, 282)     0.08681253862380181
(32, 312)     0.08681253862380181
(32, 262)     0.08681253862380181
(33, 232)     0.2662968175733716
(33, 158)     0.2662968175733716
(33, 252)     0.2662968175733716
(33, 54)      0.2662968175733716
(33, 290)     0.2662968175733716
(33, 80)      0.2662968175733716
(33, 327)     0.2662968175733716
(33, 145)     0.2662968175733716
(33, 276)     0.2662968175733716
(33, 83)      0.2662968175733716
```

```
(33, 79)      0.2662968175733716
(33, 144)     0.2662968175733716
(33, 52)      0.23834020131879124
(33, 157)     0.21850466641818692
(33, 363)     0.11874032677831936
(33, 222)     0.07957811168127973
(33, 238)     0.07094817562313468
(33, 345)     0.06894949946500069
(33, 282)     0.06894949946500069
(33, 312)     0.06894949946500069
(33, 262)     0.06894949946500069
[[1.         0.04840005 0.0262073  ... 0.0261234  0.05257826 0.04175946]
 [0.04840005 1.         0.03235488 ... 0.0322513  0.10743963 0.0515552 ]
 [0.0262073  0.03235488 1.         ... 0.02556716 0.10086411 0.0717837 ]
 ...
 [0.0261234  0.0322513  0.02556716 ... 1.         0.08261124 0.07985327]
 [0.05257826 0.10743963 0.10086411 ... 0.08261124 1.         0.05600579]
 [0.04175946 0.0515552  0.0717837  ... 0.07985327 0.05600579 1.        ]]
0      Avocado
```

Cosine Similarity

It is a metric that we use to determine the similarity of data objects, based on their size. The language of Python together with all libraries can help us to find the similarity between two or more sentences. The data objects are treated by the Cosine Similarity as vectors. This technique is beneficial because if two data objects are very far similar to each other, calculating by the Euclidian distance, they can have a very small angle between them. If the angle is small, the similarity is high. The range of Cosine Similarity is [0, 1] [12]. There is the formula of this geometric concept:

$$similarity(x,y) = \frac{\sum_{i\in I}^{n} xy * (rx,i - r\bar{x}) * (rx,i - r\bar{y})}{\sqrt{(rx,i - r\bar{x})^2 * \sum_{i\in a_{xy}}^{n} (ry,i - r\bar{y})^2}}$$

$$similarity(A,B) = cos(\Theta) = \frac{A * B}{|A| * |B|} = \frac{\sum_{i=1}^{n} a_i b_j}{\sqrt{\sum_{i=1}^{n} a_i * \sum_{i}^{n} b_j}}$$

Each of the segment's point, we subtract from the mean average value of the couple of coordinates. In the formula above, Θ is the angle between two vectors, *A* * *B* is the dot product, which is equal with the sum of unit vectors; |*A*| is the magnitude of the vector A. |*B*| is the magnitude of the vector B. Combining the similarity calculated from the scoring matrix and the similarity obtained from the commodity type, the calculation of the similarity will be more accurate (Schubert, 2021).

Foods

Avocado	0
Bluberries	1
Broccoli	2
Butter	3
Cabbage	4
Beef	5
Couliflower	6
Apples	7
Bananas	8
Onions	9
Oranges	10
Pineaple	11
Pomegranates	12
Strawberries	13
White Bread	14

```
Corn            15
Lettuce         16
Olives          17
Artichoke       18
Spinach         19
Mushrooms       20
Carrots         21
Blackberries    22
Fish            23
Milk            24
Cherriesh       25
White Rice      26
Potatoes        27
Tomatoes        28
Cucumber        29
Coconuts        30
Cheese          31
Green Tea       32
Green Peas      33
dtype: int64
```

Conclusion

No one should be demoralized if they are experiencing a difficult health condition. The best way would be to prevent that disease, showing care in the way we eat and in the lifestyle we lead. The foods we consume affect our body. One of the causes of the large spread of diseases is malnutrition, bad lifestyle, and the breakdown of the food regime. What should we do to prevent liver diseases? Through the code we offer a recommendation of the best quality foods that positively affect the liver organ and help it function normally.

The final output with best food for the healthy liver is below

```
7          Apples
11       Pineaple
12   Pomegranates
17         Olives
16        Lettuce
32      Green Tea
21        Carrots
4         Cabbage
24           Milk
20      Mushrooms
Name: Foods, dtype: object
```

Recommendations

This article modestly recommends the foods for a healthy liver. In the future we will rich the folder (where the database and the code are located), with other CSV files, why not for other organs, such as: kidney, heart, lungs, stomach, etc. This recommendation can help dietitians to list foods that can be used for a healthy diet. We may in the future create a platform like a "Blogger" to recommend foods for various diseases. This recommendation can be included in curriculums of our University, as an application of Machine Learning as a sub-field of Artificial Intelligence in the Medicine.

Scientific Ethics Declaration

The authors declare that the scientific ethical and legal responsibility of this article published in EPSTEM journal belongs to the authors.

Acknowledgements or Notes

* This article was presented as an oral presentation at the International Conference on Basic Sciences, Engineering and Technology (www.icbaset.net) held in Alanya/Turkey on May 02-05, 2024.

References

Hunter. J., Dale, D., Firing, E., & Droetboom, M. (2017). *Matplolib* https://matplotlib.org/2.1.2/Matplotlib.pdf

Kuter, K. (2023). *Math 345 -probability.* LibreText

Li, L., Zhang, Z., & Zhang, S. (2021). Hybrid algorithm based on content and collaborative filtering in recommendation system optimization and simulation. *Scientific Programming*, *2021*(4), 1-11.

Mu, R., Zeng, X., & Han, L. (2018) A survey of recommender systems based on deep learning. *IEEE Access,* (99), 1-1.

Schubert, E. (2021). A triangle inequality for cosine similarity -similarity search and applications. *International Conference SISAP 2021* (pp. 32-44)

Son, J., & Kim, S. B. (2017). Content-based filtering for recommendation systems using multiattribute networks. *Expert Systems with Applications*, *89*, 404-412.

Takeda, E., Taketani, Y., & Sawada, N. (2004,). The regulation and the function of phosphate in human body. *Bio Factors, 21* (1-4), 345-355.

Wang, Y., Zhang, D.,… & Liu, Q. (2018,). Improvement of TD-IDF algorithm based on knowledge graph. *16 th International Conference on Software Engineering Research, Management and Applications (SERA).* IEEE.

Zhang, Z., Patra, B. G., Yaseen, A., Zhu, J., Sabharwal, R., Roberts, K., & Wu, H. (2023). Scholarly recommendation systems: A literature survey. *Knowledge and Information Systems*, *65*(11), 4433-4478.

Author Information

Bederiana Shyti
Lecturer, Department of Mathematics, Faculty of Natural Sciences, University of Elbasan "A.Xhuvani" Str.Ismail Zyma, 3001, Albania

Argesta Stergu
Department of Mathematics, Faculty of Natural Sciences, University of Elbasan "A.Xhuvani" Str.Ismail Zyma, 3001, Albania
Contact e-mail: argesta.stergu@uniel.edu.al

Dhurata Valera
Lecturer, Department of Mathematics, Faculty of Natural Sciences, University of Elbasan "A.Xhuvani" Str.Ismail Zyma, 3001, Albania

Papajani Blerina
Lecturer, Department of Mathematics, Faculty of Natural Sciences, University of Elbasan "A.Xhuvani" Str.Ismail Zyma, 3001, Albania

To cite this article:

Shyti, B., Stergu, A., Valera, D., & Blerina, P. (2024). Food recommendation system for a healthy liver using machine learning. The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 28, 438-447.