Brewing Recommendations: A Data-Driven Approach to Coffee Recommendations Using Linear Algebra

Angelica Tellez and Prof. Johann Thiel

New York City College of Technology, Emerging Scholars Program

Abstract

This project explores the application of linear algebra to develop a personalized coffee recommendation system based on individual preferences. We first analyzed five randomly chosen user preferences across five different features that define their ideal coffee (e.g., flavor, aroma, aftertaste, acidity, and body). Then, we created a mathematical model that uses these preferences to recommend five additional coffee beans tailored to each user. Using Python, we developed a recommendation function that compares suggested options with each user's unique coffee profile. Through this research, we demonstrate how linear algebra concepts (such as the dot product, vector normalization) can inform everyday choices, down to the specific coffee we enjoy. In the future, we aim to gather real user data to refine the model, ideally generating accurate recommendations with even fewer inputs.

Introduction

Recommendation systems simplify decision-making by tailoring suggestions to individual preferences. This project applies such systems to coffee beans, where various features makeup the perfect cup. Everyone has different preferences when it comes to coffee, and with countless options available, finding the perfect match can be challenging. By leveraging linear algebra techniques like dot product and vector normalization, we developed a Python-based model to recommend coffee beans based on user preferences. While this recommendation function is applied to coffee here, it is versatile and can be adapted to any dataset, enabling personalized suggestions across various datasets. This approach demonstrates how linear algebra can enhance everyday experiences.

Set Up

To build our recommendation system, we began by selecting a dataset.

- In our case, we used a coffee bean dataset containing quantitative features for each bean. The dataset, originally compiled by Data Scientist James LeDoux, was sourced from the Coffee Quality Institute's review pages in January 2018.

After obtaining the dataset, in Python, we cleaned the data to ensure consistency and **selected the most important features (such as flavor, aroma, aftertaste, acidity, and body)** to describe the coffee beans.

- This is our **item matrix**, as shown in Table 1.

Table 1: Item Matrix

Item	Flavor	Aroma	Aftertaste	Acidity	Body
1	8.83	8.67	8.67	8.75	8.50
2	8.67	8.75	8.50	8.58	8.42
3	8.50	8.42	8.42	8.42	8.33
…	…	…	…	…	…
1340	6.67	6.75	6.50	6.83	6.92

To simulate user data, we generated random values to mimic user preferences, as gathering real user feedback was beyond the scope of this project.

Each user, denoted $U_1, U_2, \dots, U_5$, represents an individual with a specific coffee preference profile.

- These user profiles were generated using normalized random values based on the mean and standard deviation of the item matrix features.
- This created a **user matrix**, where each row corresponds to a simulated user's preferences, as shown in Table 2.

Table 2: User Matrix

User	Flavor	Aroma	Aftertaste	Acidity	Body
U_1	7.879669	8.294074	7.733325	7.798247	7.082887
U_2	7.712614	7.380127	7.120496	7.756013	7.812298
U_3	7.463884	7.649667	8.055015	7.858184	7.924623
U_4	8.054389	7.622334	7.552440	8.036422	7.42287
U_5	7.614702	7.264234	7.343988	7.926420	7.223269

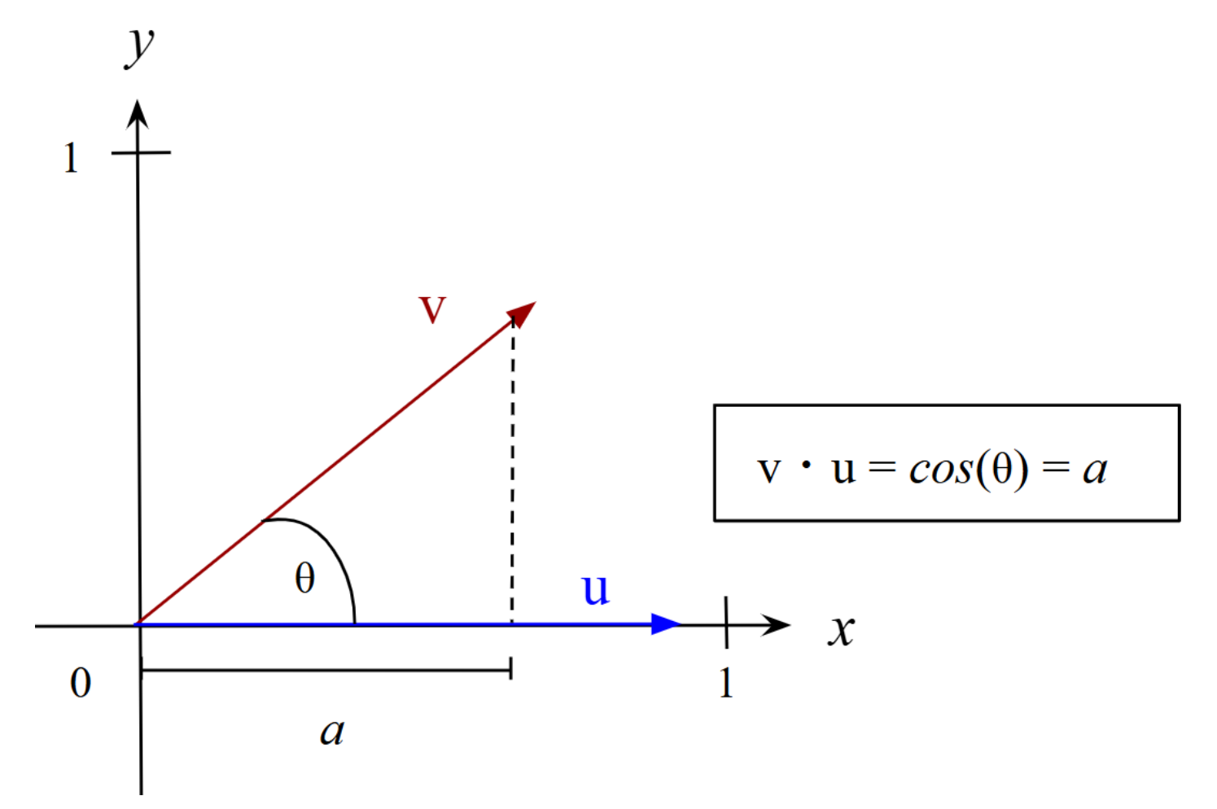
Recommendation System

Now that we have the two matrices, the **user matrix** and the **item matrix**, we need to determine their relationship by comparing each user's vector to every coffee bean in the item matrix. This allows us to identify which coffee beans are most closely aligned with each user's preferences.

- In linear algebra, **the dot product** (or scalar product) is a key tool used to measure the alignment between two vectors, shown in our context, analytically, in Figure 2.1 and 2.2.
- **To ensure consistency and interpretability in the resulting scores, before calculating the dot product, we normalized each vector.**
- **Normalization** rescales the values to a range of 0 to 1, preserving the direction of the vector while removing any influence of magnitude, as shown in Figure 2.3.
- The scalar, a, we are calculating corresponds to the cosine of the angle between the two unit-vectors, where the larger the value is, the closer the vectors are to pointing in the same direction, as shown visually in Figure 1.

.

Figure 1: Dot Product Visualization



In Python, we began by creating a **recommendation function** that takes the item and user matrices as inputs.

Within the function, nested for loops are used to normalize each vector and compute the dot product between the i-th row of the user matrix and the j-th row of the item matrix.

By performing the dot product on the normalized vectors, $\hat{u}_i \cdot \hat{v}_j$, we obtained a scalar value, a_{ij}, for each user-bean pair, as shown in Figure 2.3.

- The closer this scalar value is to 1, the more aligned the user's preferences are with the coffee bean's features.

Then, the resulting scalar is placed in the (i, j) position of the result matrix, as shown in Figure 2.1 and Figure 3.

Figure 2: Normalized Dot Product Matrix

(1)
$$u_i \cdot v_j = u_{i1}v_{1j} + u_{i1}v_{1j} + u_{i1}v_{1j} + u_{i1}v_{1j} + u_{i1}v_{1j} = a_{ij}$$

(2)
$$\begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{array}\begin{bmatrix} 7.879669 & 8.294074 & 7.733325 & 7.798247 & 7.082887 \\ 7.712614 & 7.380127 & 7.120496 & 7.756013 & 7.812298 \\ 7.463884 & 7.649667 & 8.055015 & 7.858184 & 7.924623 \\ 8.054389 & 7.622334 & 7.552440 & 8.036422 & 7.422870 \\ 7.614702 & 7.264234 & 7.343988 & 7.926420 & 7.223269 \end{bmatrix} \bullet \begin{array}{ccccc} v_1 & v_2 & v_3 & \cdots & v_{1340} \\ \hline 8.83 & 8.67 & 8.50 & \cdots & 6.67 \\ 8.67 & 8.75 & 8.42 & \cdots & 6.75 \\ 8.67 & 8.50 & 8.42 & \cdots & 6.50 \\ 8.75 & 8.58 & 8.42 & \cdots & 6.83 \\ 8.50 & 8.42 & 8.33 & \cdots & 6.92 \end{array} = R$$

(3)
$$\hat{u}_i = \frac{u_i}{\|u_i\|}, \quad \hat{u}_1 \cdot \hat{v}_1 = \hat{u}_{11}\hat{v}_{11} + \hat{u}_{12}\hat{v}_{21} + \hat{u}_{13}\hat{v}_{31} + \hat{u}_{14}\hat{v}_{41} + \hat{u}_{15}\hat{v}_{51} = a_{11}$$
$$\hat{u}_1 \cdot \hat{v}_2 = \hat{u}_{11}\hat{v}_{12} + \hat{u}_{12}\hat{v}_{22} + \hat{u}_{13}\hat{v}_{32} + \hat{u}_{14}\hat{v}_{42} + \hat{u}_{15}\hat{v}_{52} = a_{12}$$

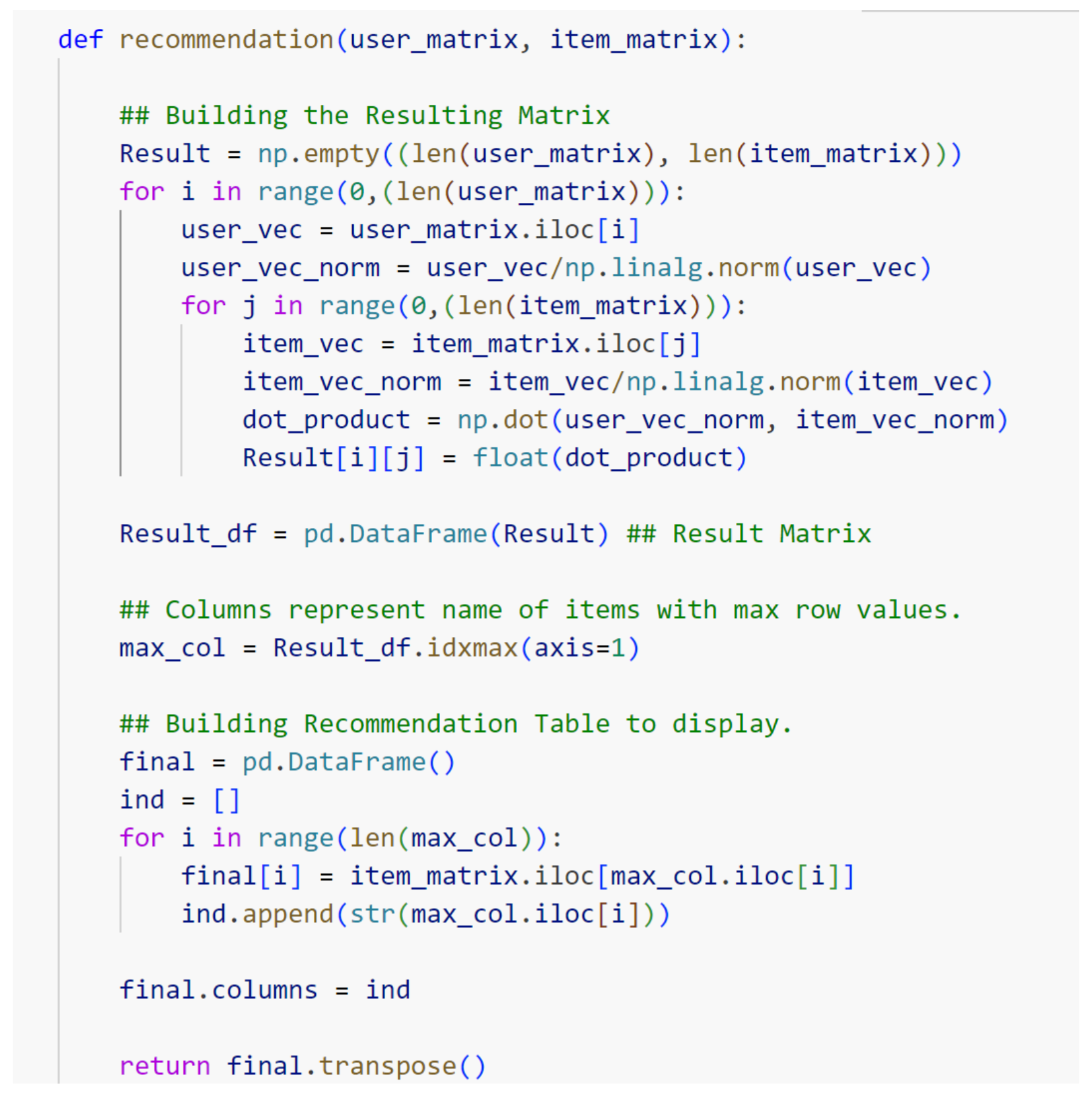
(4)
$$R = \begin{array}{c|ccccc} & \hat{v}_1 & \hat{v}_2 & \hat{v}_3 & \cdots & \hat{v}_{1340} \\ \hline \hat{u}_1 & a_{11} & a_{12} & a_{13} & \cdots & \cdots \\ \hat{u}_2 & a_{21} & a_{22} & a_{23} & \cdots & \cdots \\ \hat{u}_3 & a_{31} & a_{32} & a_{33} & \cdots & \cdots \\ \hat{u}_4 & a_{41} & a_{42} & a_{43} & \cdots & \cdots \\ \hat{u}_5 & a_{51} & a_{52} & a_{53} & \cdots & \cdots \end{array}$$

Finally, once all dot products are calculated, we identify the maximum value in each row (idxmax function in Python) of the result matrix , R, which represents the most closely aligned coffee bean for each user.

- This process continues until every user vector has been compared with every item vector.

With these results, **we construct a data frame, *final*, that displays the recommended coffee beans and their corresponding features for each user.**

Figure 3: Recommendation Function Python Code



```
def recommendation(user_matrix, item_matrix):

    ## Building the Resulting Matrix
    Result = np.empty((len(user_matrix), len(item_matrix)))
    for i in range(0,(len(user_matrix))):
        user_vec = user_matrix.iloc[i]
        user_vec_norm = user_vec/np.linalg.norm(user_vec)
        for j in range(0,(len(item_matrix))):
            item_vec = item_matrix.iloc[j]
            item_vec_norm = item_vec/np.linalg.norm(item_vec)
            dot_product = np.dot(user_vec_norm, item_vec_norm)
            Result[i][j] = float(dot_product)

    Result_df = pd.DataFrame(Result) ## Result Matrix

    ## Columns represent name of items with max row values.
    max_col = Result_df.idxmax(axis=1)

    ## Building Recommendation Table to display.
    final = pd.DataFrame()
    ind = []
    for i in range(len(max_col)):
        final[i] = item_matrix.iloc[max_col.iloc[i]]
        ind.append(str(max_col.iloc[i]))

    final.columns = ind

    return final.transpose()
```

Results

This output allows us to present a clear and interpretable list of personalized coffee **recommendations**, providing an enhanced and data-driven approach to selecting the ideal coffee, as shown in Table 3.

Table 3: Recommendations

Item	Flavor	Aroma	Aftertaste	Acidity	Body
1104	7.33	7.58	7.17	7.25	6.75
1201	7.08	6.75	6.67	7.25	7.17
38	7.75	7.92	8.08	8.08	8.08
40	8.33	7.83	7.83	8.25	7.58
1298	6.50	6.33	6.33	6.83	6.33

Conclusion

This project demonstrates how linear algebra concepts can be applied to build a personalized recommendation system, using coffee beans. Techniques like vector normalization and the dot product were used to quantify the alignment between user preferences and coffee features. By comparing a user matrix against an item matrix, we identified the most closely aligned coffee beans for each user. The recommendation function, implemented in Python, can be adapted to other datasets, making it versatile for a range of applications. Future work will focus on incorporating real user data and expanding the model's accuracy. This approach shows how linear algebra can transform everyday decisions.

Sources

1. Casalegno, Francesco. "Recommender Systems — A Complete Guide to Machine Learning Models." 2022. *medium*, https://towardsdatascience.com/recommender-systems-a-complete-guide-to-machine-learning-models-96d3f94ea748.

2. Coffee Quality Institute. *Review Pages*, January 2018, https://database.coffeeinstitute.org/.

3. Donald, Sam. "Coffee CSV File." *CORGIS Dataset Project*, 2022, https://corgis-edu.github.io/corgis/csv/coffee/.

4. Kuttler, Ken. *A First Course in Linear Algebra*. Edited by Lyryx Learning, CreateSpace Independent Publishing Platform, 2017.