Proceedings of the 8th International Food Operations & Processing Simulation Workshop (FoodOPS), 002
19th International Multidisciplinary Modeling & Simulation Multiconference

2724-0355 © 2022 The Authors.
doi: 10.46354/i3m.2022.foodops.002

Identification of Similarities and Clusters of Bread Baking Recipes Based on Data of Ingredients

Stefan Anlauf[1,2,*], Melanie Lasslberger[1], Rudolf Grassmann[3], Johannes Himmelbauer[4] and Stephan Winkler[1,2]

[1]University of Applied Sciences Upper Austria, Bioinformatics, Softwarepark 11, 4232 Hagenberg, Austria
[2]Johannes Keppler Universität, Computer Science, Altenberger Straße 69, 4040 Linz, Austria
[3]backaldrin International The Kornspitz Company GmbH, Kornspitzstraße 1, 4481 Asten, Austria
[4]Software Competence Center Hagenberg, Softwarepark 32a, 4232 Hagenberg, Austria

*Corresponding author. Email address: stefan.anlauf@fh-hagenberg.at

Abstract

We define the similarity of bakery recipes and identify groups of similar recipes using different clustering algorithms. Our analyses are based on the relative amounts of ingredients included in the recipes. We use different clustering algorithms to find the optimal clusters for all recipes, namely k-means, k-medoid, and hierarchical clustering. In addition to standard similarity measures we define a similarity measure using the logarithm of the original data to reduce the impact of raw materials that are used in large quantities. Clustering recipes based on their ingredients can improve the search for similar recipes and therefore help with the time-consuming process of developing new recipes. Using the k-medoid method, we can separate 1271 recipes into six different clusters. We visualize our results via dendrograms that represent the hierarchical separation of the recipes into individual groups and sub-groups.

Keywords: machine learning; clustering; ingredient; baking recipes

1. Introduction and Overview

When developing new baking recipes, bakers never start at zero. In most cases, they use existing baking recipes and slightly change them. The most time-consuming part is to find those recipes that are similar to the new ones. To assist the bakers, our aim is to use different clustering algorithms and similarity measures to identify similar recipes. We define several ingredient-based similarity measures to find the approach that works best.

The literature already provides some approaches to using machine learning with recipes. Su Han et al. describe the process of using different classification methods to identify the optimal cuisine classification based on the ingredients of recipes. Furthermore, they want to identify essential ingredients for a cuisine [Su et al. (2014)]. On the contrary, Hanna Kicherer et al. describe the idea of classifying recipes into predefined classes based on the textual data from social media recipes [Kicherer et al. (2018)].

In contrast to these approaches, our primary goal is to find the best way to cluster similar baking recipes based on their ingredients, without any prior knowledge about possible classes of the recipes. Additionally, another goal is not only to find optimal clusters of recipes, but also to define the best similarity measure to compare two recipes. This shall lead to an improved developing process of new recipes, as the new recipes are often developed from similar recipes. In addition to that, another goal is to identify the essential ingredients for baking recipes and those that can be easily replaced.

In Section 2.1 we summarize the data we have collected and used in this study as well as necessary pre-processing steps, including the different similarity measures we used. In Section 2.2 we describe the different clustering approaches we used. In Section 3 we summarize and discuss the results of our study, and finally Section 4 concludes this paper.

2. Methods

This section describes the structure and the processing of the data used in our study, and the similarity measures used for analysing the data are defined. Additionally, the methods used for clustering the data are described.

2.1. Data Base and Data Pre-Processing

The database used to define the similarity of recipes is based on recipes and the relative amount of ingredients. Thus, each recipe consists of one or more ingredients. We use the relative amount of the ingredients to calculate the similarity. Table 1 shows the number of recipes and ingredients used within this analysis. As Table 1 also indicates, we are not using all the recipes. We removed those recipes containing missing values, more precisely, those of which we do not know all the ingredients. In the end, we use 1271 recipes and 981 ingredients for clustering, resulting in a 1271 (data samples) x 981 (features) matrix. Examples of recipes are white bread, bread roll, pretzel stick, examples of ingredients are flour, salt, barley.

Table 1. Statistical information about the data

Type	Quantity
Ingredients	981
recipes before data cleaning	1339
Recipes after data cleaning	1271

In the following, we will describe the different similarity measures we used for the clustering. The measures explain how similar two recipes are based on their ingredients. At the project's current state, we mainly focus on the Euclidean distance, defined as the direct distance between two data points. A data point represents a data sample with values of all features. For multidimensional space, the following formula (Formula 1) describes the distance between two data samples p and q.

$$d(p,q) = \sqrt{\sum_{i=1}^{N}(q_i - p_i)^2} \tag{1}$$

Since some recipes consist of 80% of the same raw material, almost only this raw material is recognized as significant by Euclid's distance. To include this property, we use the original data and the logarithm of the original data. We apply the logarithm to each feature before the distance calculation takes place. To avoid negative numbers after performing the logarithm, we add 1 to each feature value. So, if a feature was 0 it is 0 again after calculating the logarithm. This has the effect that the small values remain the same, but the large values lose much of their relevance. In the end, we get two different datasets using two different similarity measures.

2.2. Clustering Algorithms

In this section we describe the methods and algorithms to cluster the recipes. We test different clustering algorithms to find out which one is best suited to find similar recipes. However, since we did not already have a class affiliation for any of the recipes, we had to resort to unsupervised learning methods. Unsupervised learning is a class of algorithms, which learn to classify unlabelled data. One of the most common unsupervised learning algorithms is the K-means clustering algorithm [Barlow (1989)]. We applied three different clustering algorithms to achieve the optimal combination of clustering algorithm and similarity measure. Also, not every clustering algorithm works with all similarity measures. The following paragraph explains the used algorithms in more detail:

K-means clustering is an unsupervised learning algorithm that groups data points into a fixed number of clusters (k). At the beginning, define k random cluster centroids and then perform the following two steps:

1. Assign each data point to its nearest cluster centroid.
2. Calculate new cluster centroids by the mean of all data points of each cluster

These two steps are iterated until the optimal cluster centroids are found. In most cases, those centroids do not represent actual data points. Therefore, you cannot use a precalculated similarity matrix since the similarity to the current centroid must be calculated iteratively [Sinaga and Yang (2020)].

K-medoids clustering is a very similar algorithm to the K-means clustering. The most significant difference is that the k defined medoids represent actual data points. After initializing those k medoids randomly, perform the following two steps.

1. Assign each data point to its nearest cluster centroid.
2. Calculate new cluster medoids by iterating all points in the cluster and choosing the one with the highest similarity to all others.

These two steps are iterated until the optimal cluster medoids are found. The advantage of the K-medoids is that you can use a precalculated similarity matrix since each medoid is a point from the data, so only the similarity of 2 points is needed [Park and Jun (2009)].

Hierarchical Clustering is also an unsupervised learning algorithm. For the initialization of this algorithm, all points are considered as a separate cluster, which is followed by an iterative repetition of the following step: Identify the two most similar clusters and merge them. This is continued until all clusters are merged or k predefined clusters are formed. In the end, the algorithm provides a hierarchical order, in which the further down you look, the more similar the clusters are [Johnson (1967)].

We use the implementations of these methods available in Python (specifically Scikit-learn, which contains implementations of the different clustering algorithms [Pedregosa et al. (2012)]).

2.3. Presentation of the Results

In this chapter we briefly explain the different methods we used to represented the results. We use both visual and numerical representations of the results.

2.3.1. *Visual Representation*

One of the easiest ways to visually represent clustering is with the help of a dendrogram. A dendrogram can be used to visualize the hierarchical clustering results in a tree-like structure, where every leaf is considered as one data point. Figure 1 shows a simple example of a possible dendrogram. Here, all examples are located on the x-axis and the similarity on the y-axis. The lower the connection between two samples, the more similar they are. In the end, separating into different clusters is defined by adding a horizontal line. Depending on the height of the horizontal line, the data gets separated into different amount of classes. In Figure 1 separating the data with line A results in 2 clusters, separating with line B in 4 clusters [Johnson (1967)]. A dendrogram cannot be used for visualizing the results of k-medoid or k-means clustering.

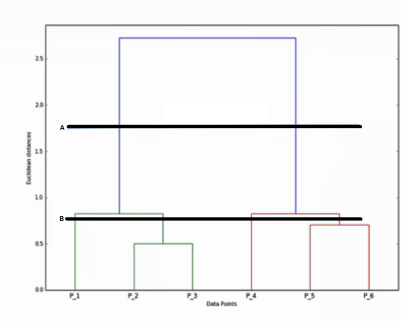

Figure 1. An example of a dendrogram after hierarchical clustering, including two possible cut lines A and B separating the data into different groups.

Besides the hierarchical clustering the other two methods need different approaches to be visually represented, as otherwise, it would not be possible to display data points with more than 900 features correctly. Specifically, methods are required to reduce the number of features to show the data points in two-dimensional space. To achieve this, the following two methods were used. We use the two methods not only individually, but also in combination.

Principal component analysis (PCA) is a method of multivariate statistics in which a large number of statistical variables are approximated by a smaller number of linear combinations that are as meaningful as possible. One minimises the correlation of multidimensional features by transferring them into a vector space with a new basis (in our case basis two). The aim is to lose as little information as possible, while at the same time reducing the amount of features [Abdi and Williams (2010)].

t-distributed stochastic neighbor embedding (t-SNE) is a method, based on stochastic neighbor embedding , to reduce high-dimensional data into a low-dimensional space of two or three dimensions. The t-SNE models each high-dimensional object into a low dimensional base by grouping similar objects by nearby points [Rogovschi et al. (2017)].

2.3.2. *Numerical Features*

To evaluate the performance of the different clustering algorithms, we use the Davies-Bouldin index. To calculation of the Davies-Bouldin index starts by calculating two measures: the intra-cluster dispersion (S) and the distance between centroids of clusters (M). Described in the following two formulas. In Formula 2 T is the number of samples per cluster, X is the center of a cluster and T is the amount of samples within a cluster. In Formula 3 A describes a cluster centroid.

$$S_i = \left\{ \frac{1}{T_i} \sum_{j=1}^{T_i} |X_j - A_i|^q \right\}^{\frac{1}{q}} \tag{2}$$

$$M_{ij} = ||A_i - A_j||_p \tag{3}$$

After calculating the two measures have to be combined. This is done by adding the intra-cluster dispersion (S) of two clusters, divide by the distance between the cluster centroids (M)., which is described with the following Formula 4.

$$R_{ij} = \frac{S_i + S_j}{M_{ij}} \tag{4}$$

After knowing how similar each cluster is to all others, the most similar cluster is selected for each cluster (Formula 5).

$$R_k = maximum(R_{ij}) \quad (5)$$

In the end, the average the similarity of each cluster with the cluster most similar to it is calculated (featured in Formula 6). Here k represents the number of clusters.

$$DBI(k) = \frac{1}{N}\sum_{k=1}^{N} R_k \quad (6)$$

The Davies-Bouldin index is the average value of the closest distances between the different clusters. The smaller it is, the better-defined clusters are represented because the lower the value is, the lower is the maximum similarity for each cluster to the other clusters [Davies and Bouldin (1979)].

3. Results and Discussion

This section presents the results of the conducted experiments, which is separated into two parts. The first part shows the results using the original data. The second part shows the results using the logarithm of the original data before performing the clustering.

3.1. Clustering Results Using the Original Data

The first section presents the results using the original data. This is the part where the more minor ingredients have nearly no weighting.

Figure 2 shows the best results are achieved with two or three clusters using the k-medoid algorithm , but separating only into a few clusters would not help find similar recipes. As Table 2 shows, the best hierarchical clustering results into 98 different clusters, which also does not help to divide the recipes into similar groups. Also, the dendrogram in Figure 7 shows that with the original data, no real hierarchical clustering can be done.

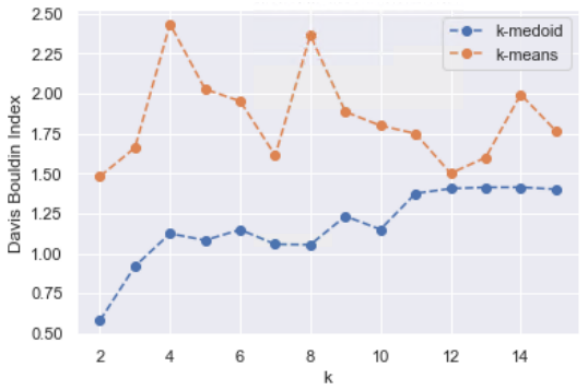

Figure 2. Distribution of the Davies-Bouldin index with different values of k, using the two clustering algorithm with the original data.

Table 2. Best numeric results of the different tested clustering algorithms, using the original data.

clustering algorithm	Davies-Bouldin index	# clusters	clustering threshold
hierarchical	2.14	98	135
k-medoid	1.05	8	-
k-means	1.5	12	- -

3.2. Clustering Results Using the Logarithm of the Original Data

The second section presents the results using the logarithm of the original data. This is the part where the more minor ingredients get more weighting.

As Table 3 shows, the best results can be achieved using the k-medoids with six different clusters or the hierarchical clustering with eleven different clusters. The k-means algorithm always performs worse than the other two.

Table 3. Best numeric results of the different tested clustering algorithms, using the logarithm of the original data

clustering algorithm	Davies-Bouldin index	# clusters	clustering threshold
hierarchical	0.69	5	35
hierarchical	1.64	11	29
k-medoid	1.16	6	-
k-means	2.06	8	-

Figure 3 shows that not all three used methods perform equally well. The best results are achieved with two or three clusters for all methods, but separating only into two clusters would not help find similar recipes. For example, besides the small cluster amounts, the best Davis-Bouldin index was achieved by the hierarchical clustering separating the data only into 5 clusters.

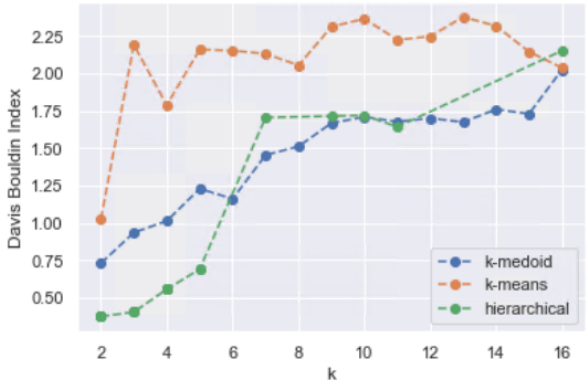

Figure 3. Distribution of the Davies-Bouldin index with different values of k, using the three described clustering algorithm with the logarithm of the original data.

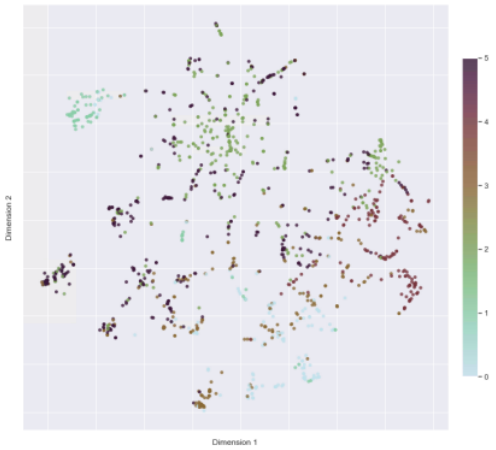

Figure 4. 2D representation, using PCA and t-SNE of the best clustering result, calculated by using k-medoid and the logarithm of the original data.

To visualize the best results (k=6) of the k-medoid algorithm, we had to reduce the dimensions using the previously described methods Principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE). Figure 4 shows the two-dimensional representation of the result, which shows that most of the six different colors are mixed within the Figure. This means that the dimension reduction loses important feature information, leading to an uninformative graphic. However, in order to display some of the clustering results graphically, we have created some dendrograms for the results of the hierarchical clustering (Figures 5, 6 and 7).

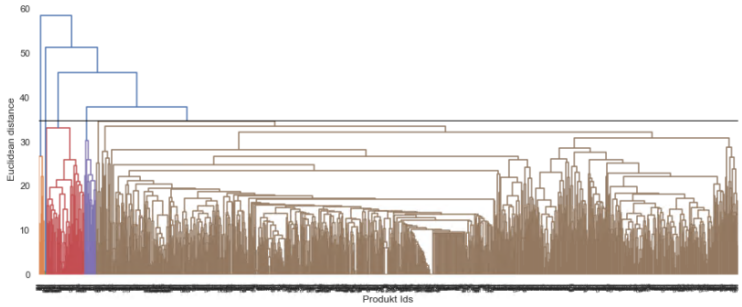

Figure 5. Dendrogram of the best hierarchical clustering result, using the logarithm of the original data (threshold line at 35).

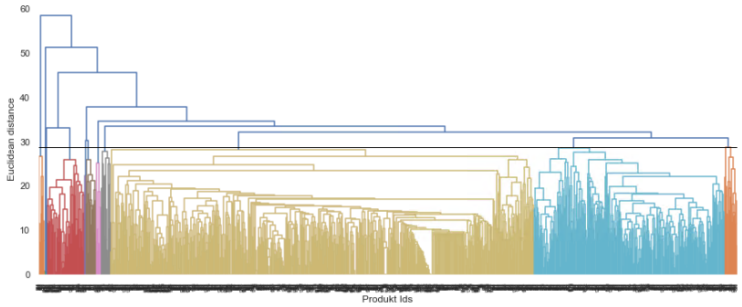

Figure 6. Dendrogram of the best hierarchical clustering result, using the logarithm of the original data (threshold line at 29)

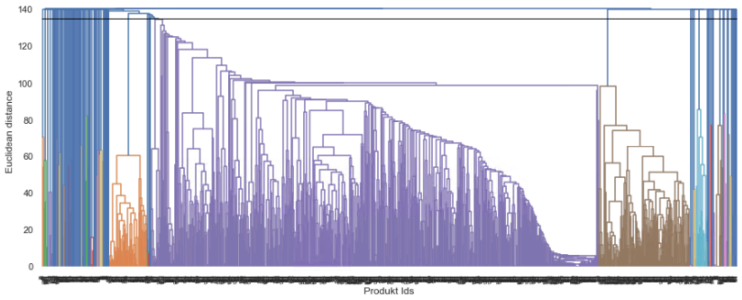

Figure 7. Dendrogram of the best hierarchical clustering result, using the original data (threshold line at 135).

Comparing the dendrograms shown in Figures 5, 6 (using the logarithm of the data) and 7 (using the original data), we immediately see that the middle one (Figure 6) looks the most promising, separating the data better into individual groups. In addition, the dendrogram shows many branches very far down, which represents a low distance and thus a high similarity. So Figure 6 separates the classes far better than the dendrogram in Figure 5, which puts over 90% of the data points in a single cluster. Figure 7, on the other hand, has a lot of recipes that have no similarity to all the other recipes, which is shown with all the blue lines that go from the bottom to the top. The y-axis of the dendrograms show that the overall Euclidean distance is far higher using the original data, indicating a worse similarity between the recipes.

4. Conclusion and Outlook

In this paper, we have shown that it is possible to identify similar baking recipes based on their relative amount of ingredients. We also discussed that transforming the data (using the logarithm of the original data) improves the identification of similar recipes by decreasing the influence of abundant ingredients like flour. Nevertheless, the identification is not completely satisfying at the moment; the available data basis is not yet big enough to utilize the full potential of the here applied methods. As shown in the results, features preprocessing (using the logarithm of the original data) improved the clustering results. Therefore, we will include a weighting of how relevant different ingredients are. We also include different definitions of the similarity of two recipes, e.g., using the information about the nutrients for the calculation of similar recipes. Additionally, we will extend this methodology even further by including information about different recipes' sizes, shapes, volumes and tastes.

Acknowledgements

The research reported in this paper has been funded by BMK, BMDW, and the State of Upper Austria in the frame of the COMET Programme managed by FFG. The project is a cooperation between University of Applied Sciences Upper Austria, Software Competence Center Hagenberg and backaldrin International The Kornspitz Company GmbH. The data was provided by the company backaldrin International The Kornspitz Company GmbH. Special thanks to Sebastian Dorl for proofreading this paper.

References

Abdi, H. and Williams, L. J. (2010). Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

Barlow, H. (1989). Unsupervised Learning. *Neural Computation*, 1(3):295–311.

Davies, D. and Bouldin, D. (1979). A cluster separation measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-1:224 – 227.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254.

Kicherer, H., Dittrich, M., Grebe, L., Scheible, C., and Klinger, R. (2018). What you use, not what you do: Automatic classification and similarity detection of recipes. *Data and Knowledge Engineering*, 117:252–263.

Park, H.-S. and Jun, C.-H. (2009). A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 36(2, Part 2):3336–3341.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., VanderPlas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2012). Scikit-learn: Machine learning in python. *CoRR*, abs/1201.0490.

Rogovschi, N., Kitazono, J., Grozavu, N., Omori, T., and Ozawa, S. (2017). t-distributed stochastic neighbor embedding spectral clustering. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1628–1632. IEEE.

Sinaga, K. P. and Yang, M.-S. (2020). Unsupervised k-means clustering algorithm. *IEEE Access*, 8:80716–80727.

Su, H., Lin, T.-W., Li, C.-T., Shan, M.-K., and Chang, J. (2014). Automatic recipe cuisine classification by ingredients. UbiComp '14 Adjunct, page 565–570, New York, NY, USA. Association for Computing Machinery.