**Recommendation System**

After gathering, cleaning, and analyzing data, we built a personalized recommendation system for Luigi. As our dataset consists of many ingredient features according to the respective recipe, we decided to implement a dimensionality reduction technique on our ingredients columns to improve our recommendation model, which helped us escape the curse of dimensionality. Moreover, we also used cosine similarity to find similarities between our recipes and to finalize our recommendations.

To implement such techniques, we have leveraged the use of the scikit-learn module of Python.

**Scikit-learn**

A robust and intuitive Python machine-learning toolkit, scikit-learn, also goes by the name sklearn, and provides a full suite of tools for creating and implementing machine-learning models. Regardless of their machine learning experience, users may easily experiment with different algorithms and strategies because of scikit-learn's consistent and user-friendly API. Numerous supervised and unsupervised learning algorithms are implemented, including regression, classification, clustering, and dimensionality reduction. By building upon the NumPy and SciPy libraries, this library uses its array data structures and mathematical functions for practical calculation while guaranteeing interoperability with other scientific computing tools inside the Python ecosystem.

One of the key strengths of scikit-learn lies in its focus on simplicity, performance, and flexibility. It prioritizes ease of use, making machine learning accessible to users of all skill levels while optimizing algorithms for performance with implementations in low-level languages such as C and Cython. scikit-learn also provides a comprehensive suite of tools for data preprocessing, model evaluation, and performance metrics, allowing users to seamlessly preprocess data, evaluate model performance, and fine-tune hyperparameters. Furthermore, the library supports custom implementations and extensions, enabling advanced users to create custom transformers, estimators, and pipelines tailored to their specific requirements.

All things considered, scikit-learn provides a strong foundation for creating, honing, and implementing machine learning models in Python, making it a valuable and essential tool for machine learning practitioners.

**PCA (Principle Component Analysis)**

PCA, which stands for principal component analysis, is a statistical technique for dimensionality reduction in data analysis and machine learning tasks. It is a method for converting high-dimensional data into a new coordinate system (a set of linearly uncorrelated variables known as principal components) in such a way that the most significant variance lies along the first coordinate (the first principal component), the second-greatest variance lies along the second coordinate (the second principal component) and so on.

The main goal of the PCA here is to reserve as much information as possible while decreasing the number of features. Several benefits of the PCA are dimentionality reduction, insight extraction, noise reduction and visualization. It can also be used in the customer segmentation, risk assessment and product development.

**The Role of PCA in our Recommendation System**

Applying PCA to our features helped us manage the features more efficiently and tackled the multidimensional issue of the dataset. In our system , we have reduced our ingredients matrix to retain only 10 principle components. We have used the n\_components variable while feeding our ingredients features into the PCA.

The reduced matrix that we gained after undergoing the dimensionality reduction is more robust and easy to manipulate.

**Cosine Similarity**

The basis of our recommendation system relies heavily upon the ingredients of the recipes. The system that we implemented finds the similarity between the recipes, in this case the Italian recipes, by comparing the recipes' ingredients. To compare the reduced ingredients matrix that we developed with the help of PCA, we have decided to use cosine similarity.

The main idea behind the use of cosine similarity is to measure the cosine angle between recipe vectors in multidimensional space. It can be calculated by using the following:

A black background with a black square

Description automatically generated with medium confidence\

Fig [4]. The mathematical intuition behind cosine similarity

By leveraging cosine similarity, our recipe recommendation system can provide personalized and relevant recipe suggestions based on the ingredients specified by the user, facilitating the exploration and discovery of new and similar recipes.

**Getting the Recommendations**

After utilizing the necessary machine learning algorithms to reach our goal of creating a personalized recipe recommendation system for Luigi, we have implemented a python function that combines the techniques of PCA and cosine similarity and gives the top 10 most similar recipe according to the recipe entered. The result of the function can be seen below.

A screenshot of a computer program

Description automatically generated

Fig [5] Results of the recommendation function

**Analyzing the performance**

To analyze the accuracy of our recommendations system we tried to construct a function which evaluates the accuracy of a recipe recommendation system by computing the cosine similarity between actual ingredients and recommended recipes, and then averaging the similarity scores across multiple queries.

While calculating the accuracy we stumbled upon many issues that held us to examine the performance of the system. Moreover, while looking at the input recipe and recommendations that system made it can be said that the system is performing well.

**Flask Web APP**

To display our recommendations system in action, we have decided to develop a web app in the Flask architecture of Python. Flask is lighweight, adoptable, and contains a structure that can be used as an API and a Web application. We have decided to consume its capabilities and built a system that allows users to select the recipe from the dropdown menu and after the submission web application returns the top 10 recommendations for the chosen recipe. Below figures displays the functionality of our system:

A screen shot of a computer program

Description automatically generated

Fig [6] Flask code that hosts our web application locally

A screenshot of a computer

Description automatically generated

Fig [7] Recommendation system home page with dropdown menu

A screenshot of a computer

Description automatically generated

Fig [8] Recommendations based upon the selected recipe