DS 5230 - Unsupervised Machine Learning

Classifying Food Items into Appropriate Categories

Pranshu Diwan (001347331) and Shagun Jindal (001596493)

Overview:

We plan to classify a dataset of unstructured food items into appropriate categories.

Why does this matter, and what is the impact?

By classifying the food items into appropriate categories, we will have a generalized idea of what the food item is. Once we have that, we can link this dataset to any restaurant orders database which has customer order data. Thus, we will be able to analyze which customers prefer which type (categories) of food items. Based on this data, we can create personalized offers and recommendations for customers of that restaurant. For example, if a customer loves eating desserts, we could send them an exclusive discount for their next order or send the new dessert dish promotional email rather than sending generalized offers. This helps a lot for businesses to retain their customers and grow their loyal customer base.

About the dataset

The dataset contains more than 13,000 rows of food items from Indian restaurants.

item_name	id
dahi vada	5367
toffee pancake	15083
chicken fried rice	3915
fish koliwada	15863
orange juice	20433
mushroom tikka grilled roll	9738
crispy chilli potato	4651
watermelon	1839
cafe latte	17005
combo 2 cgcburger+ f.fries+shake	9242

item_name	id
aloo tikka	10406
marshmallow dark white pancake	14554
chocolate volcano pancake	15917
choclate hazelnut shake	26601
plain french fries	27576
oats bread-500grams	17983
#nc fries	6468
classic lemon iced tea	2264
bombon coffee cold	12843
green apple mojito	2725

Dataset:

We have a dataset available that contains a list of 13,338 food items and their id's. These food items are actually the items on the menu of restaurants for various restaurants in Mumbai, India. Some examples include 'Chocolate Belgian Waffles', 'Butter Chicken', '30ml - Smirnoff', 'crispy chilli potato' etc.

The dataset also contains the corresponding item_id for every item. It may so happen that the item_id is unique but the food item may be the same. For example, the food item 'Egg Fried Rice' might have item_id 34 and 46. This is because some food items are common across all restaurants, and item_ids are assigned based on the food items for a particular restaurant.

Based on the dataset, we will consider categorizing different types like veg/non-veg, alcoholic/non-alcoholic beverages, Indian cuisine, Chinese cuisine etc to classify different food items.

Data Cleaning

The dataset contained several garbage values. We performed some basic text processing like:

- Unwanted Punctuations
- Special Symbols
- Converted to lowercase
- Food names starting with certain patterns

Some restaurants did not provide with the food names so garbage values were filled in for every food id

	item_name	id
0	delivery charge@30	4463
1	gi--3557	4545
2	35526644	11204
3	zomato-87737	3382
4	subway-334655	6692
5	fgsaf76asd	9908

Data Cleaning:

The dataset is raw and has many errors. We implemented four major changes in our dataset. They are:

1. Converting all data to lowercase
2. Getting rid of punctuations
3. Removing numbers from the dataset
4. Removing specific elements with no significance use (like delivery charges@30, gi–3557, zomato-87737)

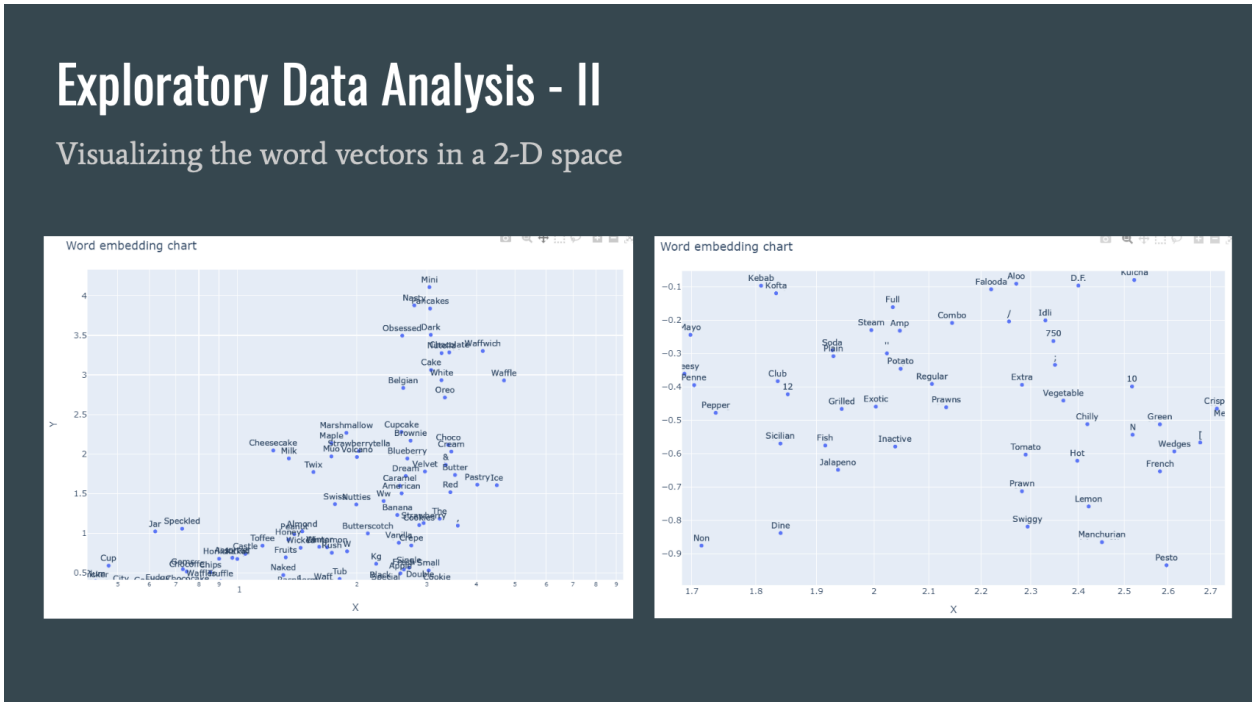
Problems Faced:

We did face some errors while cleaning data like after removing the punctuation, it deletes the id as well for some reason. Also, the number of rows are reduced which should be in sync with the id column before slicing it. While removing specific elements with no significance, since we did not know the index of the words in the database, we used the argwhere() function. The problem was that we defined an array in every new line and the names of defined arrays are confusing.

Exploratory Data Analysis using Word Cloud:

After looking at the data we saw that some food items may be common. To better understand this, we visualized the most prevalent food items through a word cloud.

From the word cloud we see that items like ‘oreo’, ‘chicken’, ‘paneer’, ‘salad’, ‘veg’, ‘fried rice’ etc are common words occurring in our data. Based on this we will later decide how categories will be selected and which food item goes in which category. But we have a basic idea like ‘chicken’ is a non-vegetarian dish. But we see that it can be available in different cuisines like Indian, Chinese etc. Similarly we know that we have ‘salad’ but salad can be both vegetarian and non-vegetarian. These findings will be very helpful in deciding our category of food items.



Exploratory Data Analysis:

We vectorized food items using Word2Vec and plotted them using PCA. We found distinct clusters depicting desserts like ‘cupcake’, ‘brownie’, ‘cheesecake’ etc which can be seen in the first image. This gives us a clear idea that one of our categories can be desserts because of the wide range in our dataset. But at the same time in this visualization we see items like ‘Chocolate Milk’ which lies in the category dessert but also in the category non-alcoholic drinks. So, there is a duplication of a food item in different categories. Hence, we need to select out categories very carefully.

In the second image we see a wider range of food items specially spicy foods like ‘Jalapeno’, ‘manchurian’, ‘fish’ etc. But at the same time we see items like ‘mayo’, ‘plain soda’ which can lie in different categories like vegetarian food items or non-alcoholic drinks.

Clustering using K-means: Deciding on categories:

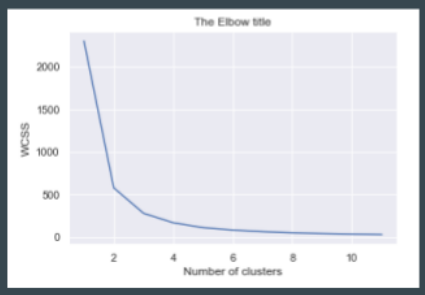
We used the K-means clustering algorithm to cluster our food items into appropriate categories. We wanted to select a suitable number of categories such that they are not too specific, but at the same time try to distinguish the food items as much as possible.

We first thought of setting the categories based on cuisines. But since the data is from many Indian restaurants so there is a bias towards Indian cuisine, hence it was a failure and we didn't consider it. Next, we thought of setting the categories like fast food, lunch/breakfast/dinner items, desserts, and beverages. Though these categories provided a good depth, they didn't quite capture every type of food item in the database and had some ambiguities in the categories. For example, some people can have a breakfast sandwich for lunch or dinner. So it was ambiguous to classify such food items.

Lastly, we finalized on setting the categories as vegetarian, non-vegetarian, desserts, alcohols, and non-alcoholic beverages. This gave us great insight into classifying food items without them being too specific.

K-Means Clustering

Optimizing the number of clusters using the elbow method



Elbow method suggests number of clusters = 3

Evaluating accuracy

- We manually created a test dataset of 100 food items with their correct categories.
- We then evaluated the K-Means algorithm on these 100 items.
- Accuracy: 56/100

Clustering using K-means: Implementing K-means:

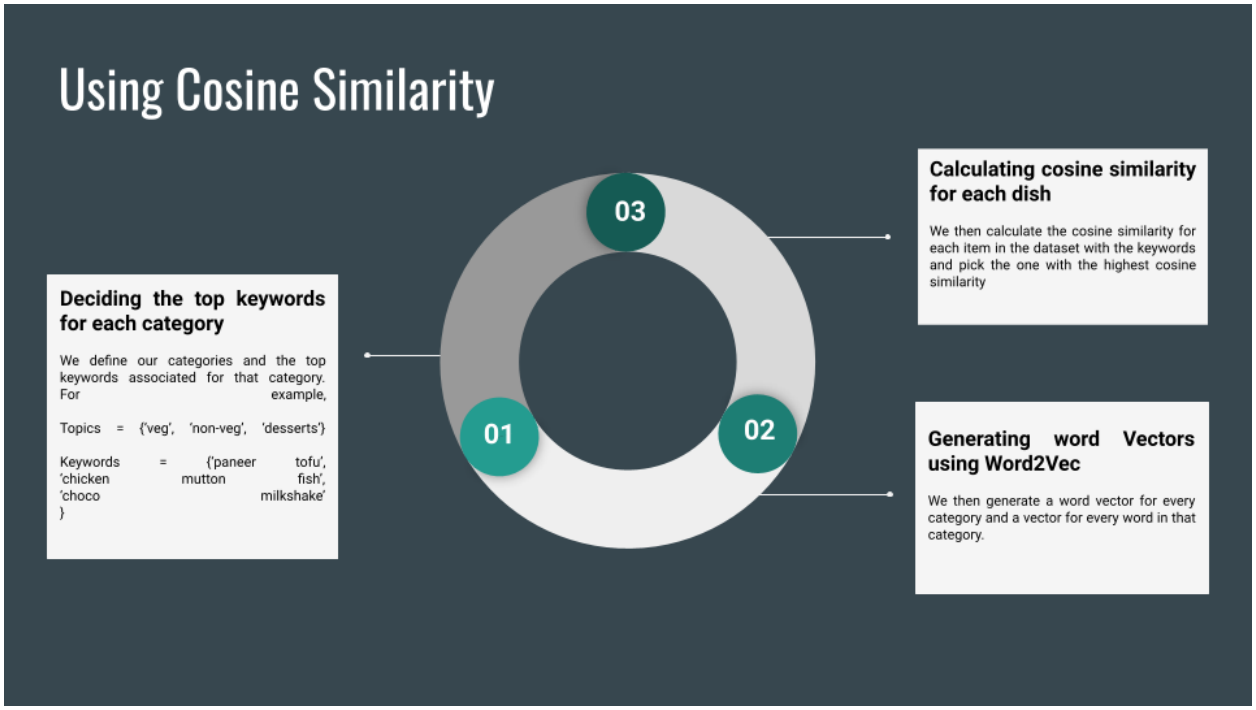
We implemented the K-means algorithm using the Scikit-Learn library. To visualize the number of clusters that the algorithm thinks is optimal, we ran the elbow method. This gave us the optimal number of clusters to be 3. But we could not move forward with such low clusters so we set the number of clusters as 5.

To evaluate this, we randomly sampled 100 food items and manually labeled them into the categories they belonged to: veg, non-veg, desserts, alcohols, and non-alcohols. Next, we ran the K-means algorithm over this “test” dataset. Out of 100, the K-means algorithm was able to correctly classify 56 food items.

Excellent Failure:

This is a low accuracy score and we thus realized that the K-means algorithm won’t always work as expected. Thus, in practical life, we should be able to find a workaround for every solution.

Thus, we decided to move forward with another approach that uses cosine similarities.



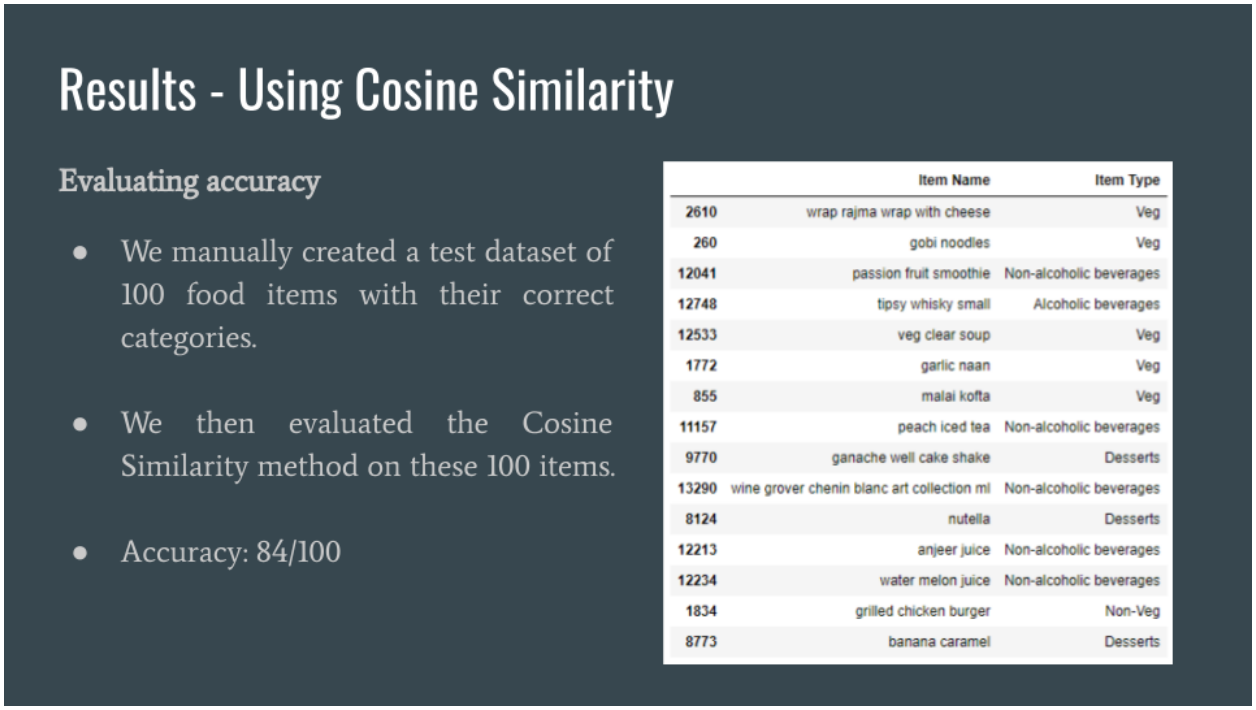
Using a cosine-similarity + Word2Vec approach:

In this approach, we followed three steps. Let us elaborate them here:

1. First, we decided on the top keywords for each category. For example, for desserts, we decided that the top keywords representing desserts are {'choco', 'vanilla', 'pancakes'}. We call this the "keyword dictionary".
2. Once we had a bunch of such "representative words", we then vectorized these keywords. This was true for every category. We also vectorized every food item in the database. The vectorization was done using Word2Vec.
3. Finally, we performed cosine similarity on the food items on to the keywords dictionary. This gave us the closest match of a food item belonging to a category, based on the cosine similarity score.

How this approach works:

Now that we have all vectorized food items and the vectorized keywords, we start by taking the first food item. Let's say it is "chocolate pancake". We calculate the cosine similarity of this food item with all vectors in the keywords list. So the vector of chocolate pancakes will be compared with the vector of every category (veg, non-veg, desserts, ...) and the highest cosine similarity will be picked, and since the cosine similarity of chocolate pancake is greatest with desserts, chocolate pancakes will be assigned a desserts tag.



	Item Name	Item Type
2610	wrap rajma wrap with cheese	Veg
260	gobi noodles	Veg
12041	passion fruit smoothie	Non-alcoholic beverages
12748	tipsy whisky small	Alcoholic beverages
12533	veg clear soup	Veg
1772	garlic naan	Veg
855	malai kofta	Veg
11157	peach iced tea	Non-alcoholic beverages
9770	ganache well cake shake	Desserts
13290	wine grover chenin blanc art collection ml	Non-alcoholic beverages
8124	nutella	Desserts
12213	anjeer juice	Non-alcoholic beverages
12234	water melon juice	Non-alcoholic beverages
1834	grilled chicken burger	Non-Veg
8773	banana caramel	Desserts

Understanding the results:

We used the SpaCy library available in Python to perform this approach. We also used the cosine similarity module available in Scikit-Learn to generate and match the cosine similarities to the most similar food category.

To evaluate this, we randomly sampled 100 food items and manually labeled them into the categories they belonged to: veg, non-veg, desserts, alcohols, and non-alcohols. Next, we ran the K-means algorithm over this “test” dataset. Out of 100, our cosine similarity approach was able to correctly classify 84 food items.

This was a significant improvement over the K-means approach. This approach could have been tweaked a bit by editing the food items defined in the keywords dictionary. We could take a look at the most frequent dishes and accordingly assign them in the keywords dictionary to optimize this approach further.

As we can see in some of the results generated although this approach does not yield the best results, it definitely gives much better results than the K-means algorithm and can be used as a baseline approach while developing a more sophisticated model.

Conclusion:

From our observations we can conclude that:

1. Simple k-means clustering algorithm alone has a low accuracy score and is an excellent failure
2. Changing the number of categories of food items ie clusters for out dataset will vary the results
3. Generating more complex word vectors might have better results

Looking Forward:

Later in the project we can use advanced clustering algorithms like K-medoids, hierarchical clustering etc to seek better results. We can also use the results we obtain from clustering to create a dataset for supervised machine learning and apply classification algorithms. Apart from that we can perform hyperparameter tuning or maybe grid search to look for better clustering results.