

**A Project Report**  
**on**  
**Computational Gastronomy : NLP Experiments**

**by**  
**Shubham Kale - MT23094**

**Independent project**  
**Summer Semester 2024**



**Complex System Laboratory**

**Under Guidance**

**of**

**Prof. Ganesh Bagler**

**Infosys Center for Artificial Intelligence**

**Department of Computational Biology, IIIT-Delhi, New Delhi**

# Task 1

## Web Scraping

1. Visited several websites on the web to find the best possible website from where I can scrap the data for the NLP Experiments.

### Tools found :

1. <https://github.com/hhursev/recipe-scrappers>

- The git repository contain the a simple recipe scraping tool which can be used using bellow command
- **pip install recipe-scrappers**
- The package contains around 100+ websites from where the web scraping can be done easily and a large number of recipes can be scraped.

Below are the list of things which can be scraped :

1. scraper.host()
2. scraper.title()
3. scraper.total\_time()
4. scraper.image()
5. scraper.ingredients()
6. scraper.ingredient\_groups()
7. scraper.instructions()
8. scraper.instructions\_list()
9. scraper.yields()
10. scraper.to\_json()
11. scraper.links()
12. scraper.nutrients() # not always available
13. scraper.canonical\_url() # not always available
14. scraper.equipment() # not always available

- 15. scraper.cooking\_method() # not always available
- 16. scraper.keywords() # not always available
- 17. scraper.dietary\_restrictions() # not always available

## 2. Recipe-book :

<https://github.com/dpapathanasiou/recipebook>

- This is the simple application for scraping and parsing food recipe. It is available in hRecipe format and which also produce result in .json file format
- Data -  
Recipes from the recipe book using crawlers are now available in a parallel repository

Link - <https://github.com/dpapathanasiou/recipes>

The above recipes have an Alphanumeric Index .

- Like recipes starting with number (0 - 9)
- Recipes starting with letter (A - Z )

The recipe is in .json format as shown below :

```
{
  "directions": [
    "In a large pot over medium heat, cook chicken pieces in oil until browned on both sides. Stir in onion and cook 2 minutes more. Pour in water and chicken bouillon and bring to a boil. Reduce heat and simmer 45 minutes.",
    "Stir in celery, carrots, garlic, salt and pepper. Simmer until carrots are just tender. Remove chicken pieces and pull the meat from the bone. Stir the noodles into the pot and cook until tender, 10 minutes. Return chicken meat to the pot just before serving."
  ],
  "ingredients": [
    "2 tablespoons vegetable oil",
    "2 skinless chicken leg quarters",
    "1/2 cup chopped onion",
    "2 quarts water",
    "3 cubes chicken bouillon, crumbled",
    "1 stalk celery, chopped",
    "3 carrots, chopped",
    "1 clove roasted garlic, minced",
    "salt and pepper to taste",
    "1 (12 ounce) package thin egg noodles"
  ],
  "language": "en-US",
  "source": "allrecipes.com",
  "tags": [],
  "title": "A-1 Chicken Soup",
  "url": "http://allrecipes.com/recipe/25651/a-1-chicken-soup/"
}
```

}

So from the above platform the recipes will be scraped and used for further NLP experiments.

Also i will be manually coding to scrap the recipe from some of the website to gain insight about how web scraping works

Scraped around 1000+ recipes from the [here](#)

The recipes where stored in .json format [here](#)

Let's look at the format how the recipes were stored

```
[
  {
    "title": "Crispy Fried Falafel",
    "ingredients": [
      " ",
      "1 cup dried broad beans",
      "1 cup dried garbanzo beans",
      " water (enough to cover beans)",
      " ",
      "4 tablespoons fresh parsley , roughly chopped",
      "4 tablespoons of fresh mint , roughly chopped",
      "4 tablespoons fresh coriander or 4  tablespoons cilantro , roughly chopped",
      "1 medium onion , peeled and chopped",
      "2 garlic cloves , peeled and crushed",
      "1 teaspoon dried coriander or 1  teaspoon dried cilantro",
      "1 teaspoon cumin",
      "1/2 teaspoon turmeric",
      "1/2 teaspoon cayenne",
      "1/2 teaspoon salt",
      "1 teaspoon bicarbonate of soda",
      " vegetable oil, for deep-frying"
    ],
    "directions": [
      "In a bowl add broad beans and chick peas then pour water into the bowl to cover the beans",
      "Cover the bowl with cling wrap and all them to soak overnight",
      "Drain the broad beans and chick peas then remove and discard shells",
      "Place beans in a large food processor, add parsley, mint, coriander, onion, garlic, dried coriander, cumin, turmeric, cayenne and salt",
      "Process until it's all chopped up and mixed",
      "Transfer the mixture to a large bowl and add bicarbonate of soda",
      "Knead the combine thoroughly, cover with cling wrap and leave to rest in fridge for about one hour",
      "After one hour make patties",
      "To do this mould the mixture into walnut size ball and then flatten it slightly, place it on a tray",
      "Repeat step until mixture is complete",
      "Heat the vegetables oil in a heavy-based frying pan",
      "Once heated gently lower the patties into the hot oil in batches, (patties will be delicate and may not hold its shape)",
      "Deep fry for about five to six minutes, once cooked place them on clean paper towels to drain oil",
      "Repeat step until all patties have been cooked",
      "Serve the falafel either hot or cold and enjoy"
    ],
    "cooking_time": "1hr 50mins"
  },
]
```

The .json file contain 1019 recipes with the Title , Ingredients , directions , Cooking\_time

# Task 2

## Recipes Embeddings

Task - To extract the embedding for recipe using pre-trained language model  
i.e

1. Vanilla BERT
2. Sentence BERT

Using data preprocessing removed recipe with same title ,also added all text content of recipe in one column before extracting embeddings.

First ,

For Vanilla BERT i used the bert-base-uncased which can be load from below code

```
# Load pre-trained BERT model and tokenizer
```

```
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
```

```
model = BertModel.from_pretrained('bert-base-uncased')
```

This generates the 768 embedding from the text given to it.

v_emb							
	recipe_id	embedding_0	embedding_1	embedding_2	embedding_3	embedding_4	embedding_5
0	1	-0.262028	0.139195	0.236266	0.155400	0.328113	-0.138488
1	2	-0.213843	0.182655	0.221657	0.150422	0.355628	-0.123803
2	3	-0.210638	0.154153	0.303195	0.147520	0.339003	0.040504
3	4	-0.330708	0.125723	0.338822	0.092337	0.232397	0.111925
4	5	-0.303564	0.159836	0.096404	0.145001	0.299905	-0.009336
...	...	...	...	...	...	...	...
890	891	-0.137300	-0.018568	0.275461	0.081565	0.084093	-0.118755
891	892	-0.322973	0.109014	0.254887	0.263739	0.351826	-0.075757
892	893	-0.223426	0.203399	0.209337	0.109677	0.424135	-0.135610
893	894	0.045518	0.014256	0.274799	0.100052	0.152619	-0.016060
894	895	-0.261918	0.222937	0.124933	0.185909	0.242130	-0.099001

895 rows × 769 columns

Dimensions of embedding excluding id column- 895\*768

Then,

I used <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2> sentence bert model which maps the text to the 384 dimensional dense vector space which can be further used for the task like clustering and semantic search

s_emb							
	recipe_id	embedding_0	embedding_1	embedding_2	embedding_3	embedding_4	embedding_5
0	1	-0.043613	-0.018790	0.001034	-0.017085	-0.077300	-0.030520
1	2	-0.088772	-0.014221	-0.017543	-0.019059	-0.007498	-0.023840
2	3	0.003340	-0.010799	-0.025417	0.043355	-0.047669	0.010352
3	4	-0.019419	-0.030488	0.007062	-0.019521	-0.060563	-0.011525
4	5	-0.088165	-0.045828	-0.062601	0.004691	-0.028209	-0.004604
...	...	...	...	...	...	...	...
890	891	-0.082921	0.010770	0.001853	0.017936	0.029310	0.031683
891	892	-0.056471	-0.052263	-0.033441	0.036171	-0.056696	0.037490
892	893	-0.013081	0.028458	-0.027735	0.042218	-0.006050	-0.057268
893	894	-0.055919	0.029853	-0.025598	-0.007382	-0.081740	-0.040251
894	895	-0.112035	-0.000831	-0.006288	0.007898	-0.082886	0.009837

895 rows × 385 columns

Dimensions of embedding excluding id column- 895\*384



# Task 3

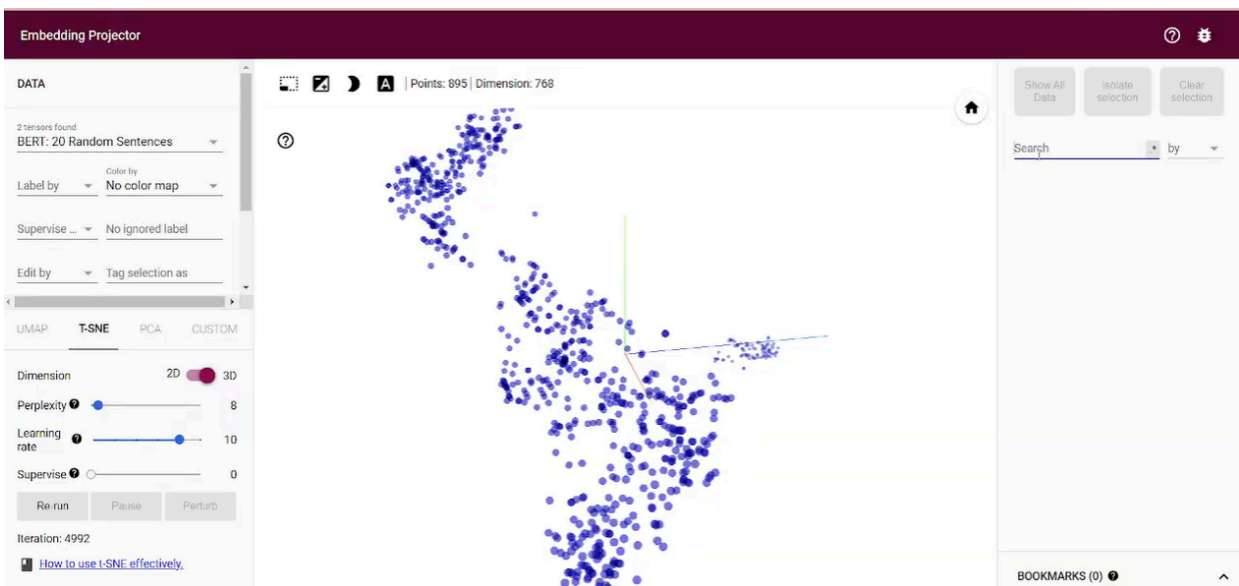
## Visualization of Embeddings

I used the online tool to visualize the embedding the embedding generated by both the model -

Website for visualization : <https://projector.tensorflow.org/>

### 1. Visualization for Vanilla BERT Embeddings ( t-SNE)



No of iteration used - 4992







[to watch video click on above image](#)

I have find the 5 nearest point using euclidean distance for recipe id - 890  
I got 5 recipe nearest to recipe id 890 that are 521 ,360, 448, 554, 442

Search  
890  by 

neighbors   5

distance COSINE EUCLIDEAN

Nearest points in the original space:

521	0.543
360	0.562
448	0.578
554	0.581
442	0.602

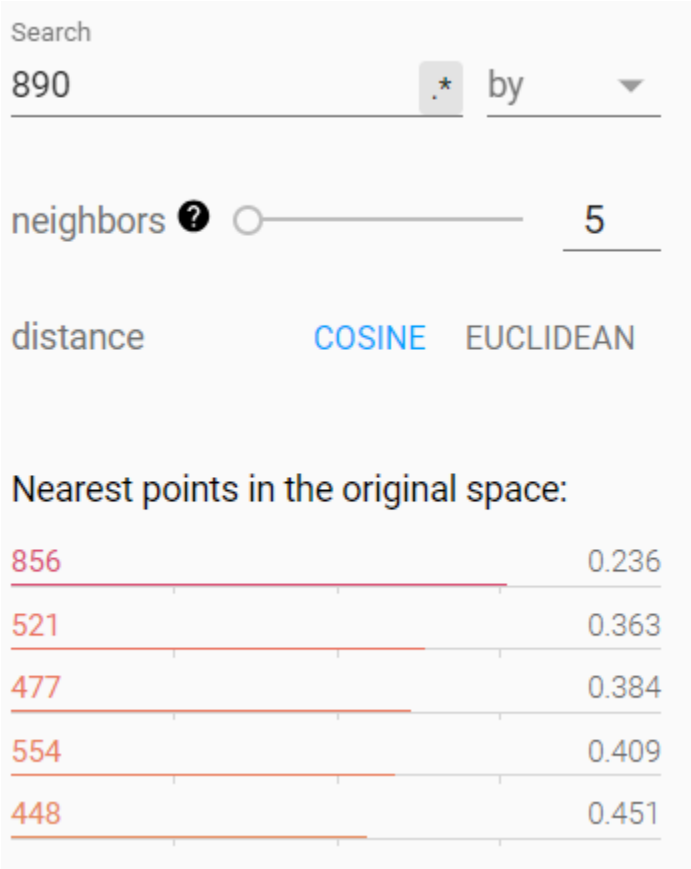
## 2. Visualization for Sentence BERT Embeddings ( t-SNE) -

No of iteration = 4995



[to watch video click on above image](#)

I have find the 5 nearest point using euclidean distance for recipe id - 890  
I got 5 recipe nearest to recipe id 890 that are 856, 521, 477, 554, 448



## Conclusion after Visualization :

### Using Vanilla BERT Embeddings -

I got 5 Nearest recipe to recipe id 890 using cosine distance are listed below as i can conclude that recipe are very much similar also there cooking time lie between 20 - 35 min for majority of them

	A	B	C	D	E	
1	title	ingredient	directions	cooking_ti	recipe_id	
2	Wonderful Parmesan Zucchini Strips	['1/3 cup	['Preheat c	35mins	360	
3	Pork Tenderloin	['1 garlic cl	['Mix garlic	5hrs 15min	442	
4	Tilapia Al Ajillo (Garlic Tilapia)	['1 1/2 lbs	['Season ti	20mins	448	
5	Baked Pollock	['4 -6 fish f	['Preheat c	25mins	521	
6	Parmesan Crusted & Baked Salmon	['1 lb salm	['Preheat c	35mins	554	
7	Easy! Oven-Baked Cod	['1 1/2 cup	['Preheat c	25mins	890	
8						

## Using Sentence BERT Embeddings -

I got 5 Nearest recipe to recipe id 890 using cosine distance are listed below as i can conclude that recipe are very much similar also there cooking time lie between 20 - 35 min for majority of them

	A	B	C	D	E
1	title	ingredient	directions	cooking_ti	recipe_id
2	Tilapia Al Ajillo (Garlic Tilapia)	['1 1/2 lbs	['Season ti	20mins	448
3	Lemon Butter Tilapia	['12 -16 oz	['Chop par	15mins	477
4	Baked Pollock	['4 -6 fish f	['Preheat c	25mins	521
5	Parmesan Crusted & Baked Salmon	['1 lb salm	['Preheat c	35mins	554
6	Lemon Baked Cod	['1 lb cod f	['If fish fille	35mins	856
7	Easy! Oven-Baked Cod	['1 1/2 cup	['Preheat c	25mins	890

## Conclusion

Both models return recipes with high similarity, but Sentence BERT has one recipe that is a perfect match in terms of ingredients, cooking method, and type of fish: **Lemon Baked Cod** (Recipe ID 856).

### Best Match

- **Sentence BERT: Lemon Baked Cod** (Recipe ID 856)

**Both model give 3 similar recipe id - 448 , 521 , 544**

# Task 4

## Web scraping of specific recipes i.e pizza and biryani from the food.com

### Extracted Pizza and biryani -

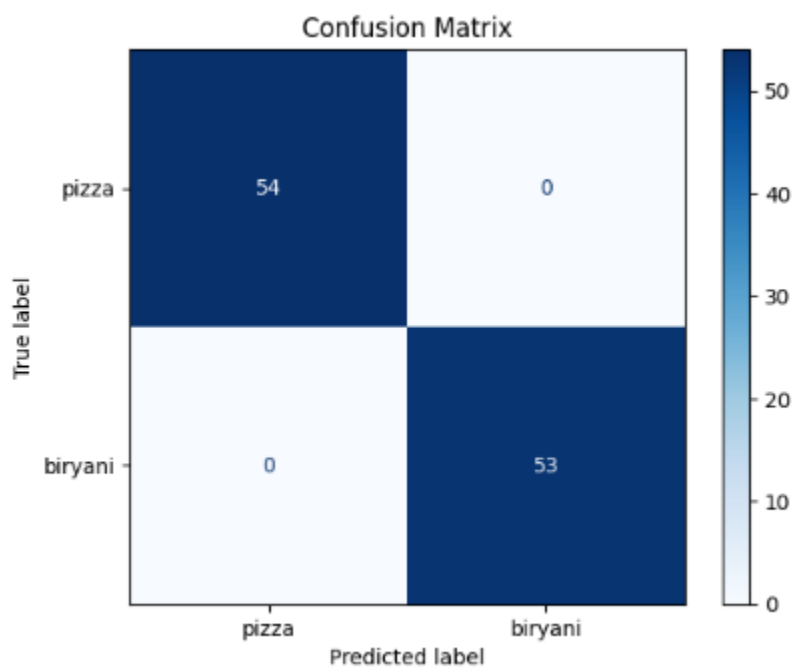
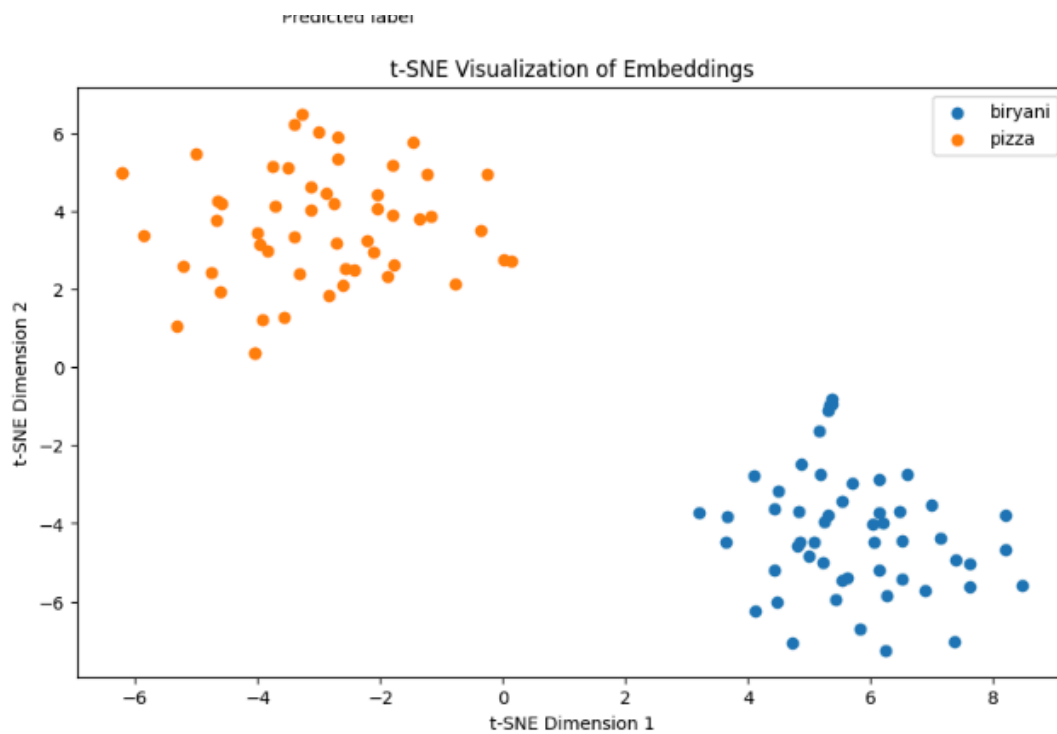
	title	ingredients	directions	cooking_time	label
0	fresh fig caramelized onion and goat cheese go...	[1 lb prepared pizza dough room temperature i ...	[in a pan saute onions in oil till a caramel c...	50mins	pizza
1	gourmet pesto pizza	[20 ounces whole wheat pizza dough storebought...	[if the pizza dough has been refrigerated take...	35mins	pizza
2	leftover bbq pulled pork or shredded chicken p...	[1 loaf rhodes bread dough thawed risen or piz...	[form dough into a ball then roll into approxi...	55mins	pizza
3	ground beef pizza	[12 lb lean ground beef, 1 small onion chopped...	[in a skillet cook ground beef onion garlic or...	25mins	pizza
4	general tsos chicken pizza	[1 lb store bought pizza dough, 6 ounces froze...	[remove pizza dough from the refrigerator and ...	1hr 55mins	pizza
...	...	...	...	...	...
102	vegetable biryani	[2 cups wholemilk yogurt, 6 medium tomatoes di...	[first take your yogurt and mix in your diced ...	1hr 40mins	biryani
103	turkey biryani	[5 tablespoons peanut oil, 3 large onions fine...	[heat 2 tbsp of the oil in a large flameproof ...	1hr 15mins	biryani
104	subramaniam's indian fish biryani manav loves	[50 g garlic pounded, 50 g ginger pounded, 3 g...	[marinate the fish in a bowl with the spicesma...	1hr 33mins	biryani
105	sindhi biryani	[1 1/2 kg mutton, 5 cups basmati rice soaked i...	[slice the onion and fry it in oil until it is...	1hr 30mins	biryani
106	vegetable biryani	[3/4 cup dry green lentils, 2 tablespoons extra...	[in saucepan of boiling water cook lentils for...	35mins	biryani

107 rows × 5 columns

Next Step model Used :

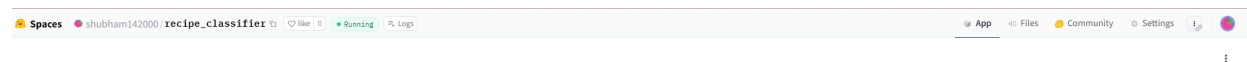
**model\_name = "sentence-transformers/all-MiniLM-L6-v2"**

**Based on the embedding i got following Clusters**



Model to test the approach :

[https://huggingface.co/spaces/shubham142000/recipe\\_classifier](https://huggingface.co/spaces/shubham142000/recipe_classifier)



## Conclusion

The approach for classifying text as "Biryani," "Pizza," or "Neither" is highly effective, achieving good accuracy in the current tests. Here's an overview of the approach:

### 1. Model and Tokenizer:

- A pre-trained model and tokenizer from the **sentence-transformers** library, specifically the **all-MiniLM-L6-v2** model, were used to generate text embeddings.

### 2. Embedding Generation:

- Input text is tokenized and passed through the model to obtain embeddings. The mean of the last hidden state of the model's output is used as the text embedding.

### 3. Classification:



- **Cosine similarity is calculated between the input text embedding and pre-existing embeddings from the dataset. The label of the most similar embedding is assigned to the input text, provided the similarity exceeds a specified threshold; otherwise, the label "neither" is assigned.**

#### **4. Visualization:**

- **A 2D t-SNE visualization of the embeddings was generated to demonstrate the clustering of "Biryani" and "Pizza" recipes. The input text embedding was projected into this t-SNE space to show its placement relative to the existing clusters.**

#### **5. Results:**

- **The classifier successfully distinguishes between "Biryani" and "Pizza" with clear, distinct clusters in the t-SNE visualization. The input text embedding consistently falls within the correct cluster, validating the model's accuracy.**

**This approach leverages state-of-the-art NLP techniques to create robust and accurate text classification models, effectively handling the specific task of classifying recipe-related text.**

.

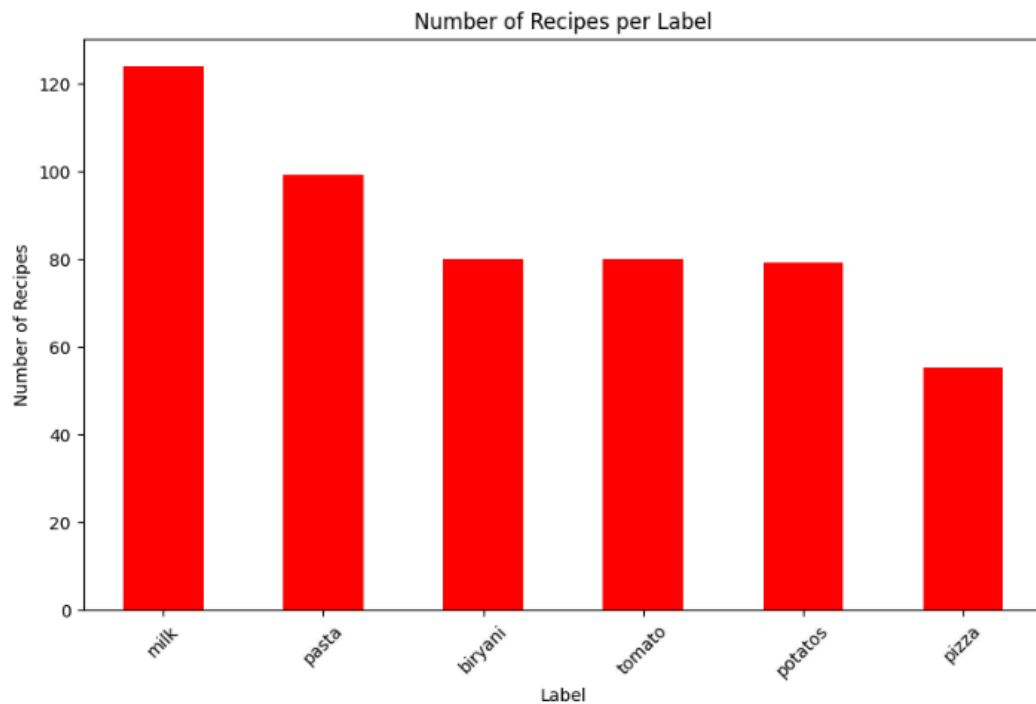
# Task 5

## Multiclass Receipe classification based on the embeddings

1. Biryani
2. Pizza
3. Tomato
4. Potato
5. Milk
6. Other

First off all i scrapped the recipe in json format from the [www.food.com](http://www.food.com) website

Below is the data-distribution of the recipe scrapped



Used the hugging faces pretrained model

**model\_name = "sentence-transformers/all-MiniLM-L6-v2"**

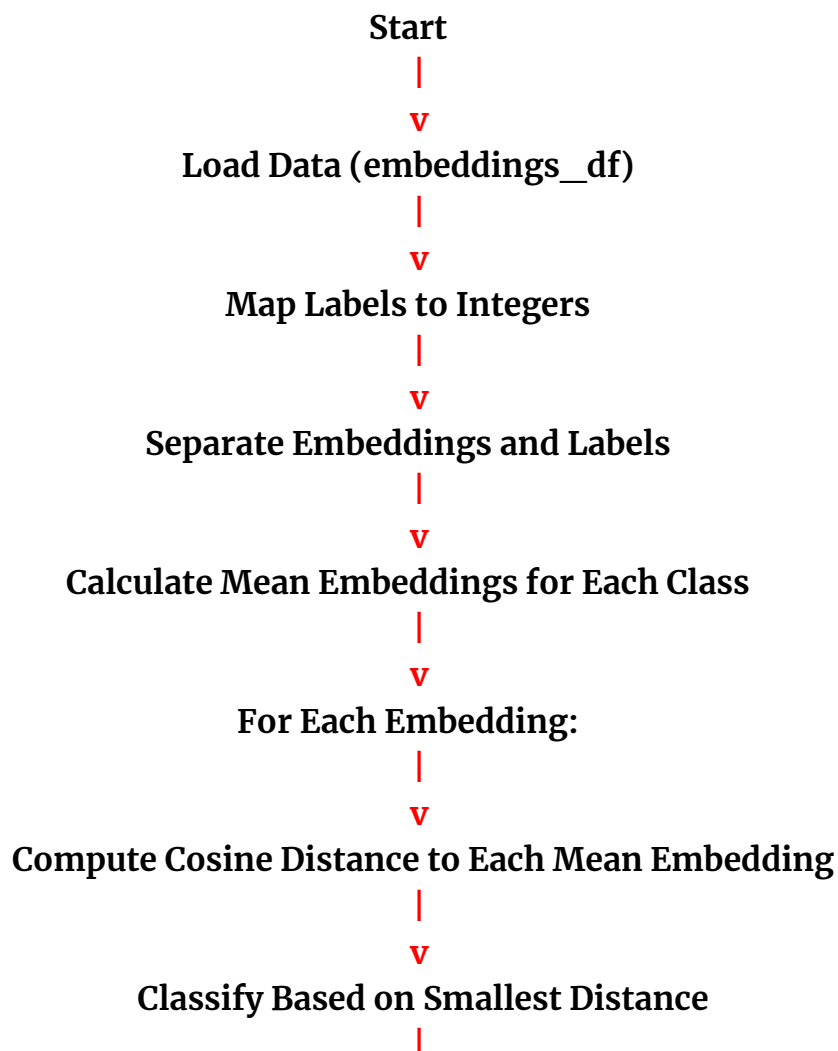
Approach to classify recipe :

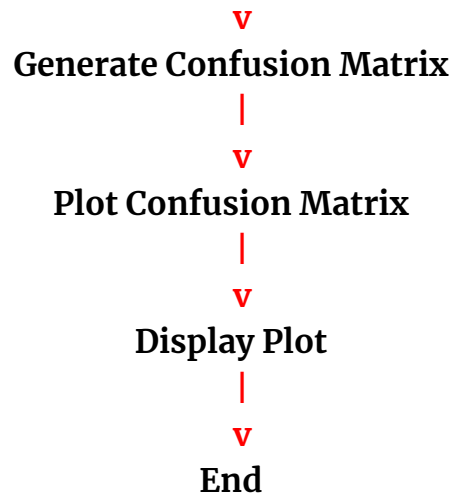
1. Cosine Distances between Embeddings Generated
2. Training MLP classifier to predict the recipe class based on the given Embedding

## **Results and Visualization :**

### **1. Cosine Distances between Embeddings Generated**

Flow Chart of Approach :





Confusion matrix :

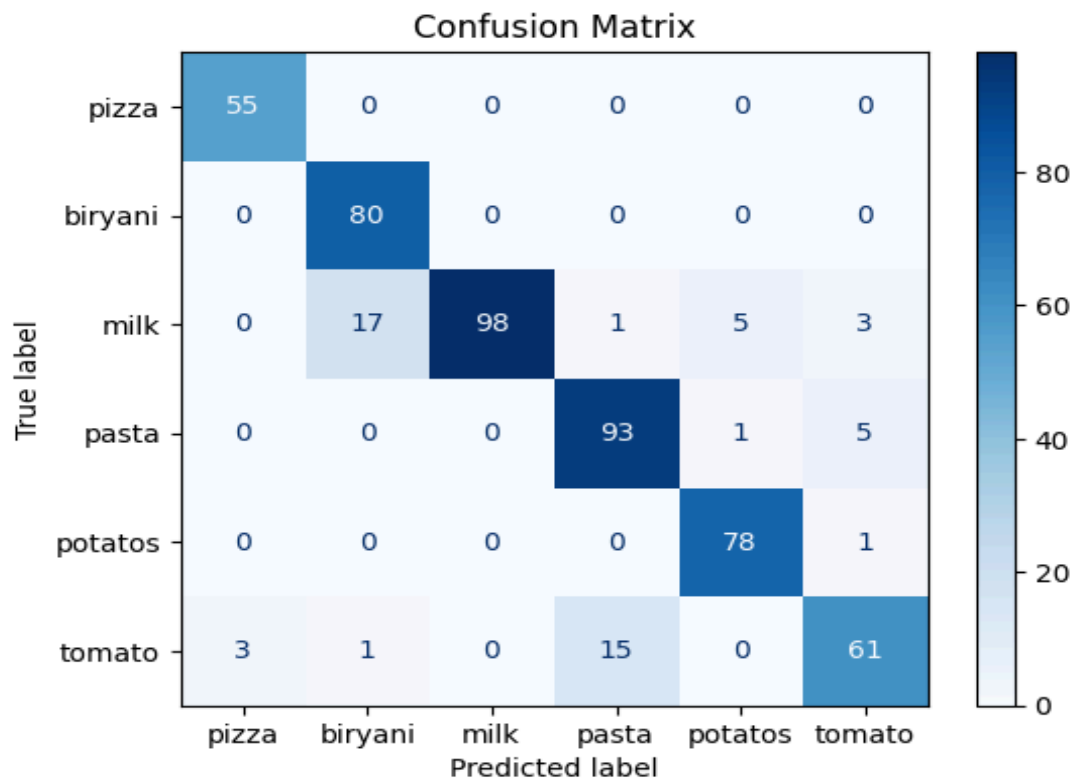


Fig : Confusion Matrix of classification using Cosine distance

The confusion matrix shows that the Approach of cosine distance is doing a good job of classifying the different recipes. The majority of the predictions are correct, and there are only a few misclassifications.

Overall, this confusion matrix suggests that the Approach of cosine distance is performing well and is able to accurately classify different recipes. However, there

are still some areas where the Approach could be improved. For example, the Approach misclassified 3 tomato images as pizza.

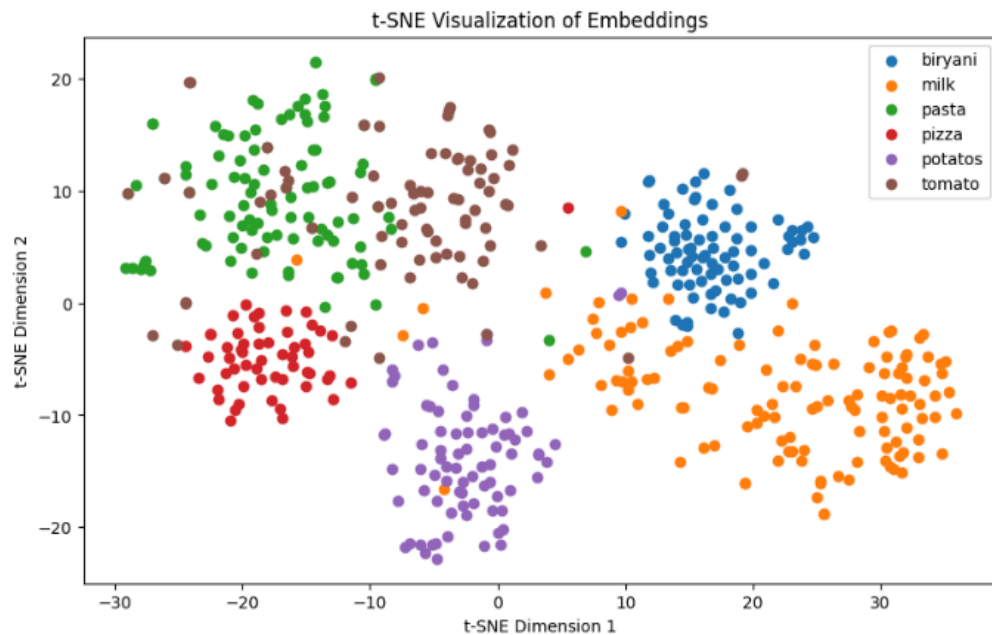


Fig : t-SNE plot forming distinct clusters

From above plot we can say that the embedding when represented in 2d plane form a distinct cluster some of the recipes are close due to some similarity .  
The Plot was perform using t-SNE with parameter `n_components = 2`

## 2. Training MLP classifier to predict the recipe class based on the given Embedding

Here i used MLP classifier with below parameter as shown below the model was giving very good results and was able to distinguish between the different recipes based on there feature.

```
MLPClassifier
MLPClassifier(hidden_layer_sizes=(64, 64), max_iter=500, random_state=42)
```

```

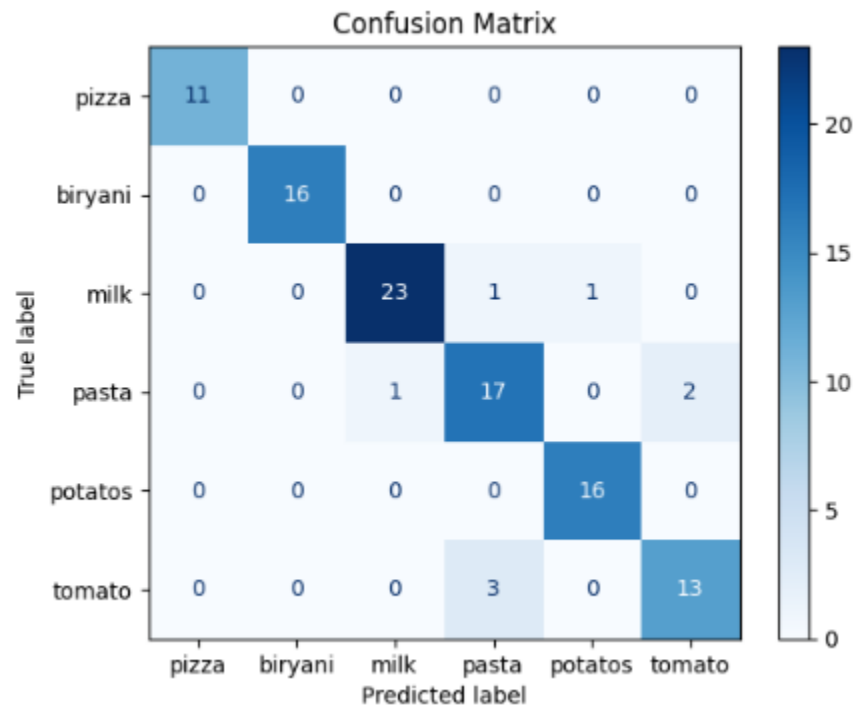
Classification Report:
              precision    recall  f1-score   support

    pizza           1.00        1.00        1.00         11
   biryani           1.00        1.00        1.00         16
      milk           0.96        0.92        0.94         25
      pasta           0.81        0.85        0.83         20
    potatos           0.94        1.00        0.97         16
      tomato           0.87        0.81        0.84         16

 accuracy           0.92
  macro avg           0.93        0.93        0.93         104
 weighted avg           0.92        0.92        0.92         104

```

Classification using MLP gives **92%** accuracy

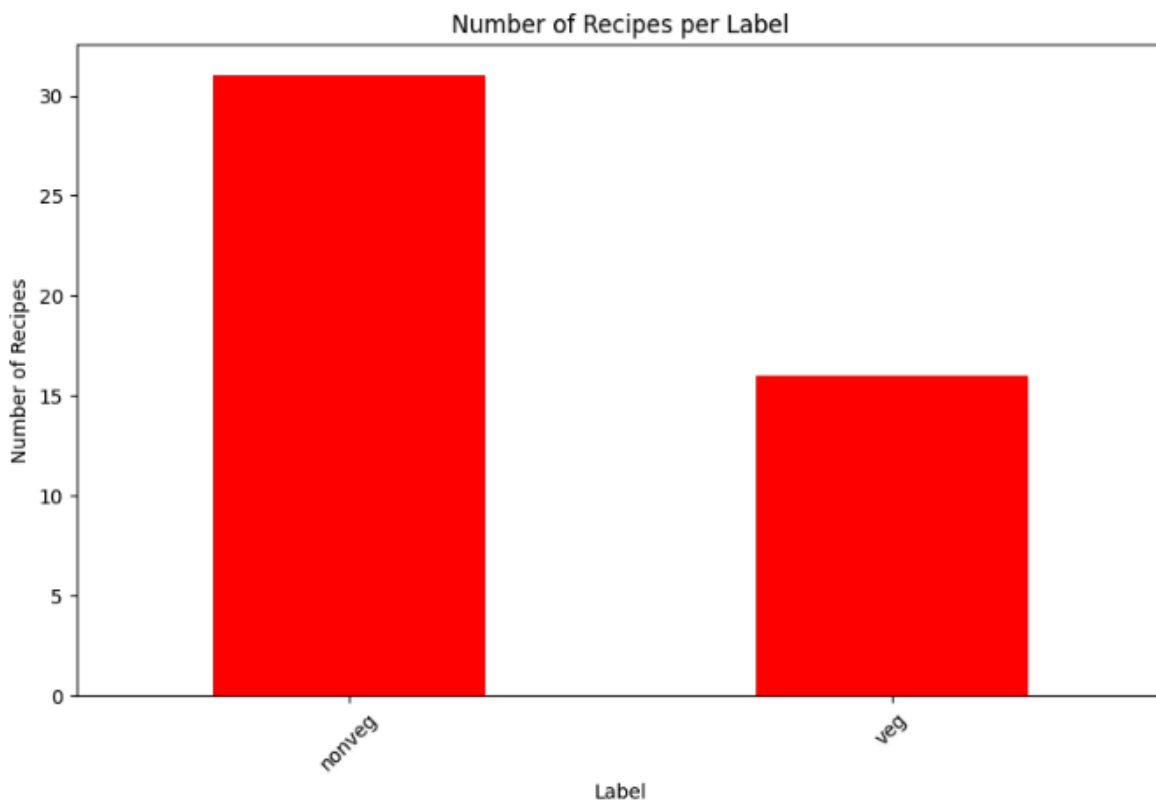


Then confusion matrix shows good results as here we have very few misclassification as compared to the previous approach .

# Task 6

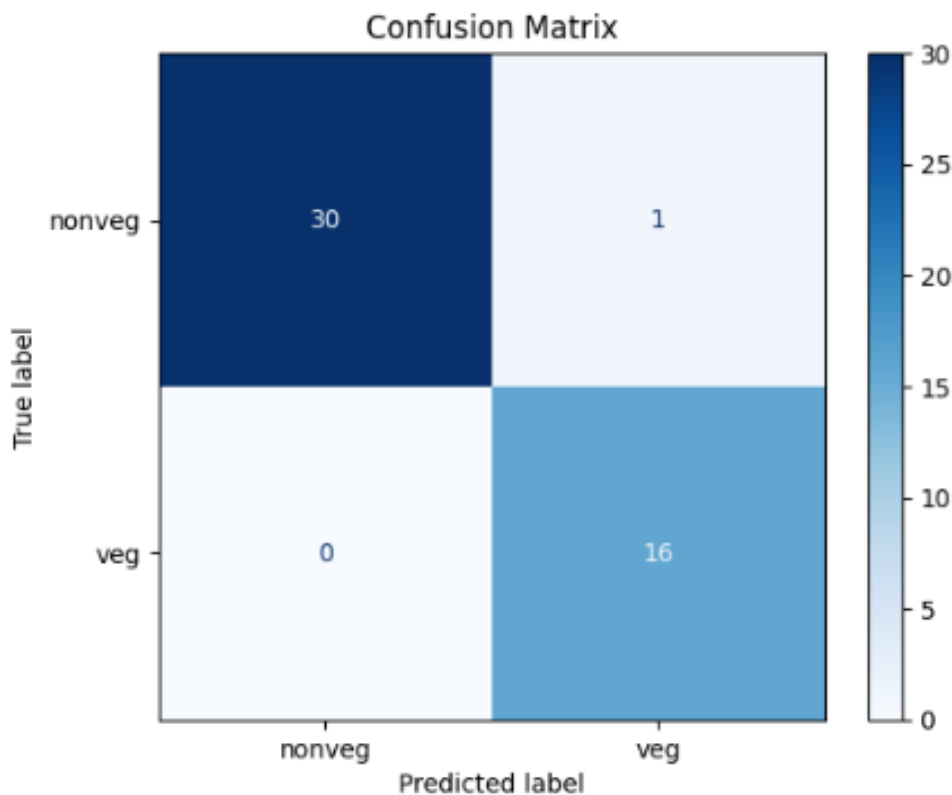
## Intra recipe classification between Veg-Biryani and Non-Veg Biryani

### 1. Cosine Distance based Approach



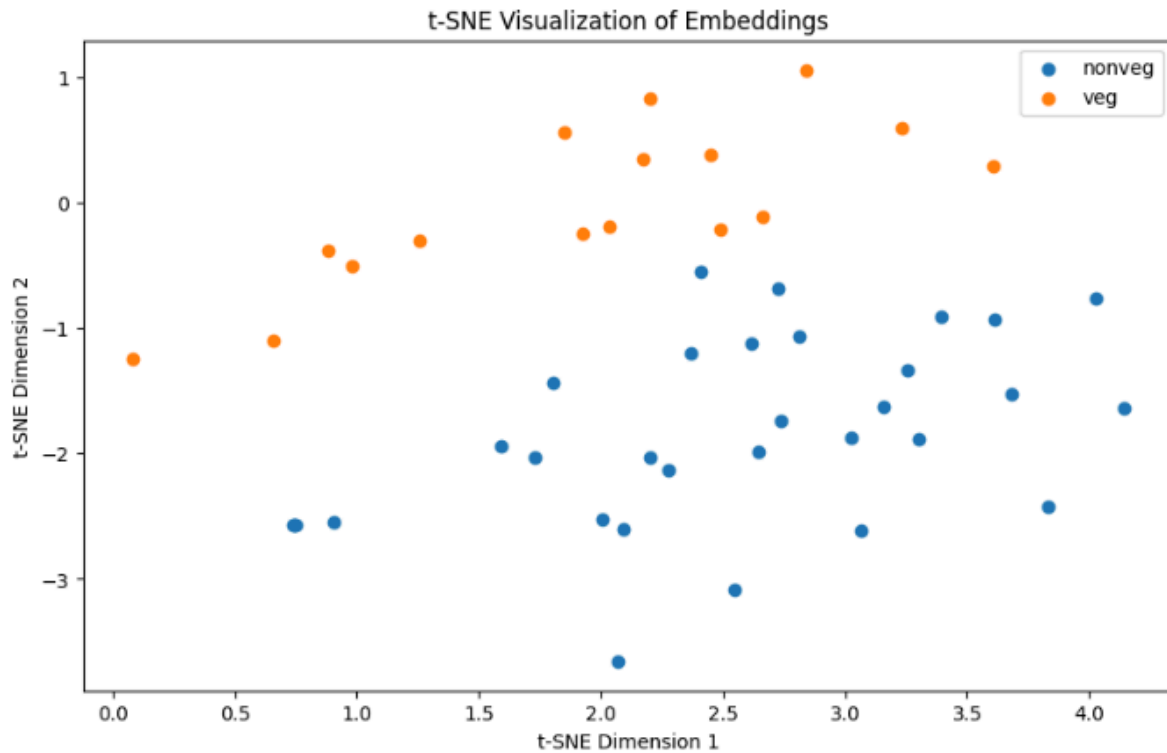
Scraped the recipes for veg and non-veg biryani from the [www.food.com](http://www.food.com) got 31 non veg biryani and around 16 veg biryani .Next i extracted the embedding using the sentence bert.

First, I prepared the data by ensuring it contained labels, recipe IDs, and embedding columns, mapping the labels "veg" and "nonveg" to integers. I then calculated the mean embedding for each class. For each embedding, I determined its classification by calculating the cosine distance to the mean embeddings of both classes, assigning it to the closer class. Using the true and predicted labels, I generated and plotted a confusion matrix. I identified and listed recipe IDs for false positives (non-veg predicted as veg) and false negatives (veg predicted as non-veg). To visualize the embeddings, I used PCA for dimensionality reduction, reducing the embeddings to two dimensions for a faster and clear visualization.



From above confusion matrix we have 1 recipe with ID 31 misclassified other all where correctly classified to its respective class.





The above figure shows the cluster for both class we can find the decision boundary hence sentence bert embedding where able to find the correct parameter which lead to form a distinct cluster.

We can say the sentence bert embedding are able to generate and classify between intra class recipe also like veg and non veg biryani

## 2. MLP based Approach

Below are the model parameter for MLP classifier been used in this approach

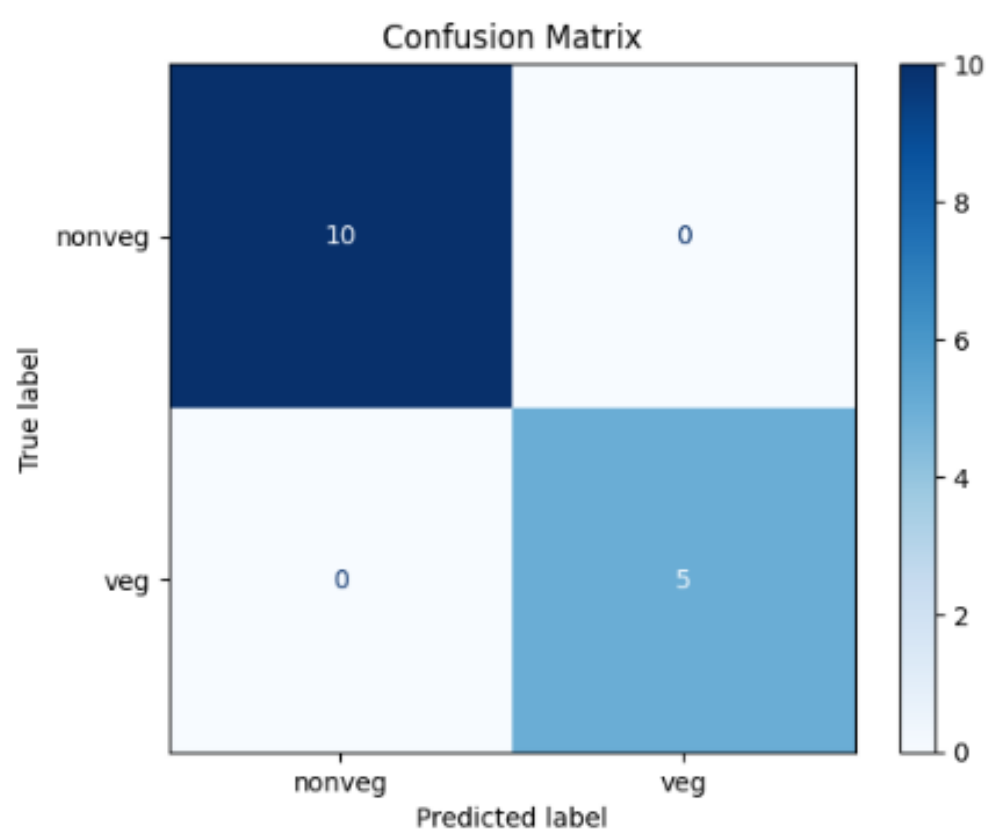
```

MLPClassifier
MLPClassifier(hidden_layer_sizes=(64, 64), max_iter=500, random_state=42)

```

Classification Report:				
	precision	recall	f1-score	support
nonveg	1.00	1.00	1.00	10
veg	1.00	1.00	1.00	5
accuracy			1.00	15
macro avg	1.00	1.00	1.00	15
weighted avg	1.00	1.00	1.00	15

Classification report shows 100 percent accuracy using MLP model for Intra recipe Classification



Conclusion -

In this approach, I used Sentence-BERT embeddings and cosine distance to classify recipes into veg and non-veg categories. The process began by mapping labels to integers and calculating mean embeddings for each class. Each recipe was then classified based on its cosine distance to these mean embeddings.

The confusion matrix revealed the classification's accuracy, and identifying false positives and negatives helped pinpoint areas for improvement. PCA was used for dimensionality reduction, allowing for a clear 2D visualization of the embeddings. This visualization showed distinct clusters for veg and non-veg recipes, demonstrating the effectiveness of the embeddings in capturing essential features.

Overall, this method proved capable of not only distinguishing between broad categories but also capturing fine-grained variations, such as differentiating between veg and non-veg biryani. The combination of Sentence-BERT embeddings and cosine distance provided a robust and insightful classification approach.

# Model Deployment

I deployed three models for the previous 2 tasks . The model was able to give the classification label/ prediction based on the input recipe text given by the user and the prediction generated by approaches used where giving really good results.

The Deployed model can be find at my hugging face profile :

<https://huggingface.co/shubham142000>

Deployed model 1 :

[https://huggingface.co/spaces/shubham142000/recipe\\_classifier](https://huggingface.co/spaces/shubham142000/recipe_classifier)

Deployed model 2 :

[https://huggingface.co/spaces/shubham142000/multi\\_class\\_recipe\\_classifier](https://huggingface.co/spaces/shubham142000/multi_class_recipe_classifier)

I will upload everything associated with the project in the below public github repository including code , datasets etc :

Github repository : [https://github.com/Shubham23094/CG\\_NLP\\_Experiments-](https://github.com/Shubham23094/CG_NLP_Experiments-)