Data Science Application

Final Project

Prepared by:

Mahmoud Yahia Ahmed

Mahmoud Maged Elwan

Umar Mohamed Ibrahim

Abdulrahman Ahmed

I) Problem Formulation.

Nutrition is one of the crucial topics nowadays, especially Food dissipation is one of the most challenging problems around the world, and we believe that the huge effect is caused by small changes, so we started with a very simple problem which "lunch dilemma", here we try to suggest different ways cook with the same or similar ingredient which will be extracted from user text automatically, and that idea can extend to be a really creative food recommender for a personal suggestion or commercial one with a restaurant and online food apps.

the system contains different parts chained in a logical flow,

1- The input text: the user's text

2- preprocessing: remove non-interesting parts and standardize data

3- POS & Dependency tree: extract the grammatical tags and logical words dependency.

4- Premise entities extraction: extract the potential entities.

5- Few-shot NER: extract entities using few-shot learning.

6- Cuisine type prediction (classification): predict causing type using extracted ingredients.

7- Suggestion based clustering: suggest different dishes which relate to or are similar to those extracted ingredients.

II) Data Sets.

We used 2 data sets:

- **Indian-food. (Clustering)**

Data content

Indian cuisine consists of a variety of regional and traditional cuisines native to the Indian subcontinent.

Data description

[name, ingredients, diet, prep_time, cook_time, flavor profile, course, state, region]

- **Recipe-ingredients-dataset. (Classification)**

Data content

Different cuisine types with theme ingredients, it collected from different countries around the world.

Data description

[id, cuisine type, ingredients]

III) Data Preparation.

1- Remove punctuations

Removing punctuation that adds up noise that brings ambiguity while training the model.

2- Remove Digits

Meaningful and meaningless are considered subjective without any sort of reference where adding digits to word could change the meaning.

3- Convert to lowercase

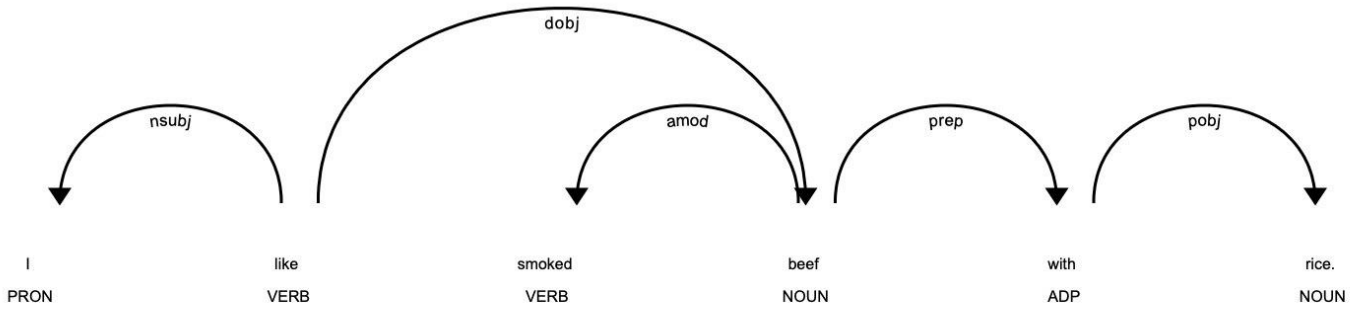
Converting all your data to lowercase helps in the process of preprocessing and in later stages in the NLP application, when you are doing parsing.

4- Remove Stop Words and Stemming

A group of words which are highly frequently used without any additional information, such as articles, determiners and prepositions called stop-words. By removing this very commonly used words from text, we can focus on the important words instead. Also applying stemming on specific language and differ with respect to performance and accuracy.

IV) POS and Dependency tree.

In Dependency parsing, various tags represent the relationship between two words in a sentence. These tags are the dependency tags. Extracting grammatical tags and contextual dependency.



V) Premise Entities extraction.

the POS tags and Dependency tree results had used here, some Dependency rules have been defined using grammatical awareness of language and logical dependency, so the Dependency Matcher of Spacy used here to extract the defied rules, to get all potential entities, and it will be filtered in the next step to just keep the related entities.

here are some rules we tried to extract:

```
# ADJ -> NOUN
[{"RIGHT_ID": "adj", "RIGHT_ATTRS": {"POS": "ADJ"}},
 {"LEFT_ID": "adj", "REL_OP": "<", "RIGHT_ID": "subject", "RIGHT_ATTRS": { "POS": "NOUN"}}],

# NOUN
[{"RIGHT_ID": "noun", "RIGHT_ATTRS": {"POS": "NOUN"}}],

# NOUN . NOUN
[{"RIGHT_ID": "noun", "RIGHT_ATTRS": {"POS": "NOUN"}},
 {"LEFT_ID": "noun", "REL_OP": ".", "RIGHT_ID": "subject", "RIGHT_ATTRS": { "POS": "NOUN"}}],

# PROPN
[{"RIGHT_ID": "pnoun","RIGHT_ATTRS": {"POS": "PROPN"}}],

# PROPN << NOUN
[{"RIGHT_ID": "pnoun","RIGHT_ATTRS": {"POS": "PROPN"}},
 {"LEFT_ID": "pnoun", "REL_OP": "<<", "RIGHT_ID": "subject", "RIGHT_ATTRS": { "POS": "NOUN"}}],

# PROPN << PROPN
[{"RIGHT_ID": "pnoun", "RIGHT_ATTRS": {"POS": "PROPN"}},
 {"LEFT_ID": "pnoun", "REL_OP": "<<", "RIGHT_ID": "subject", "RIGHT_ATTRS": { "POS": "PROPN"}}],

# VERB < NOUN
[{"RIGHT_ID": "verb","RIGHT_ATTRS": {"POS": "VERB"}},
 {"LEFT_ID": "verb", "REL_OP": "<", "RIGHT_ID": "subject", "RIGHT_ATTRS": { "POS": "NOUN"}}],
```

VI) FS-NER

Name Entities Extraction NER based on few-shot learning FSL has used here, where you don't need to train your model for specific entities, which is very helpful for the cases that suffer from data scarcity.

so instead of input the text to the model and it extracts the entities that have already been learned, it takes the text with entities definition, and it will try to extract them based on its latent knowledge representation.

the only challenge here is you need to identify your entities with meaningful words and each entity can have a few examples that are related to the entity name, and the more informative words you use, the more accuracy you will get.

So, using mDeBERT model produced by Microsoft, this idea becomes feasible, so it takes text as input with some potential classes and it produces the association probability for each class (as SoftMax), it is using its intensive knowledge representation that is built using huge datasets in 16 different languages following few-shot approach.

So, we try to discriminate which given class is more associated with the given text and using the soft-classification we can get which entity is more relevant to input text.

Input text

```
text = "I like smoked beef with rice."
```

Input entities

```
org_entities = {"fruite"     :    ["apple", 'avocado', 'banana', 'orange'],
                "vegetable"  :    ["carrot", "cabbage", 'broccoli'],
                "meat"       :    ["beef", "lamb", 'chicken'],
                "fish"       :    ["salamon", "shrimp", "tuna"],
                "spices"     :    ["angelica", "allspice", "cumin"],
                "ingredient" :    ['oil', 'salt', 'sugar'],
                "other"      :    []}
```

NER Output

i like **smoked beef** meat (80%) with **rice** ingredient (40%) .

VII) Error analysis FS-NER

here we test that with some hard examples e.g using hard words, add new entities without samples, try adding some samples, test the confidence, etc.

```
"taking Iced watermelon with lemon is very delicious after eating shrimp soup, while driving a car."
```

a) here the model success to identify most entities but did not recognize "car" as entity and that's because no relevant entities exist within the given ones

```
{"fruite"     :    ["apple", 'avocado', 'banana', 'orange'],
 "vegetable"  :    ["carrot", "cabbage", 'broccoli'],
 "meat"       :    ["beef", "lamb", 'chicken'],
 "fish"       :    ["salamon", "shrimp", "tuna"],
 "spices"     :    ["angelica", "allspice", "cumin"],
 "ingredient" :    ['oil', 'salt', 'sugar'],
 "other"      :    []}
```

taking **iced watermelon** fruite (42%) with **lemon** fruite (33%) is very delicious after eating **shrimp soup** fish (43%) , while driving a car.

b) here we added "vehicle" as new entity without any examples (zero-shot), and the mode could to identify the "car" with 85% confidence.

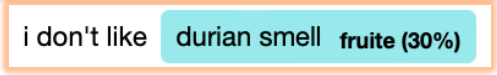
```
{"fruite"     :    ["apple", 'avocado', 'banana', 'orange'],
 "vegetable"  :    ["carrot", "cabbage", 'broccoli'],
 "meat"       :    ["beef", "lamb", 'chicken'],
 "fish"       :    ["salamon", "shrimp", "tuna"],
 "spices"     :    ["angelica", "allspice", "cumin"],
 "ingredient" :    ['oil', 'salt', 'sugar'],
 "vehicle"    :    [],
 "other"      :    []}
```

c) here some examples were added to "vehicle" entity, but notice that "car" word didn't added here, and the accuracy increased to 86%.

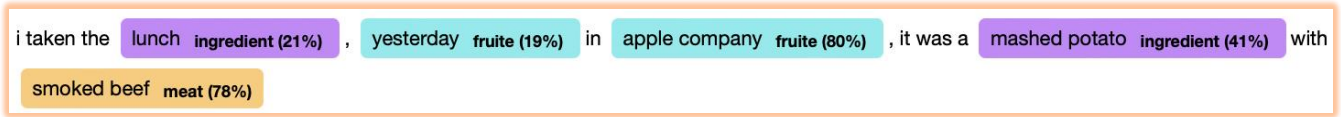
```
{"fruite"     :    ["apple", 'avocado', 'banana', 'orange'],
 "vegetable"  :    ["carrot", "cabbage", 'broccoli'],
 "meat"       :    ["beef", "lamb", 'chicken'],
 "fish"       :    ["salamon", "shrimp", "tuna"],
 "spices"     :    ["angelica", "allspice", "cumin"],
 "ingredient" :    ['oil', 'salt', 'sugar'],
 "vehicle"    :    ['bus', 'truck'],
 "other"      :    []}
```

d) here is an example for strange fruit called "durian", and the model could to capture it with confidence 30%, and to increase it, we need to add more informative examples to "fruit" entity.

```
text = "I don't like durian smell"
```



e) here the model identify two wrong tokens, "launch" and "yesterday" but both of them has confidence less than 20%, but with "Apple company" it got 80%, so it identify the word Apple as a fruit which is contextually wrong, and that is one of the drawbacks so far.

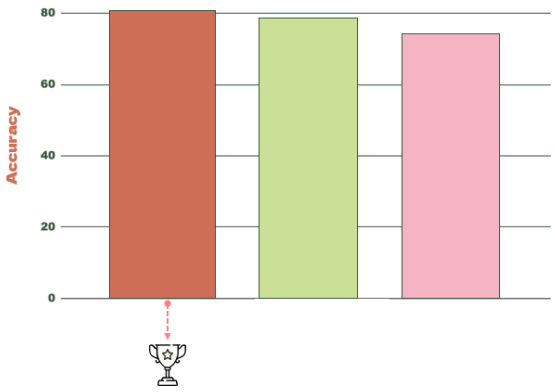


VIII) Cuisine type prediction (Classification)

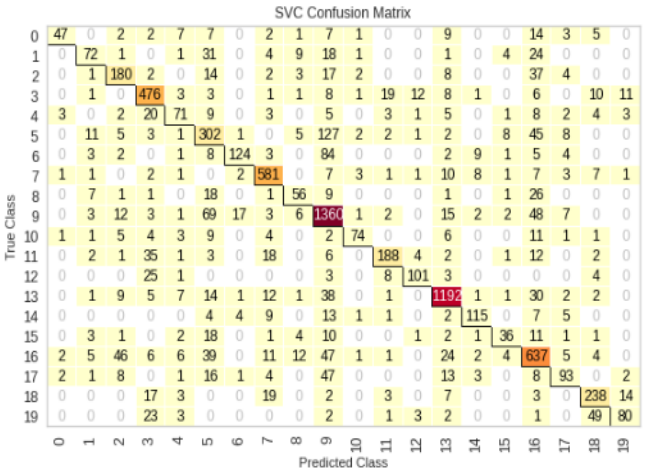
Applying feature engineering using: TF-IDF

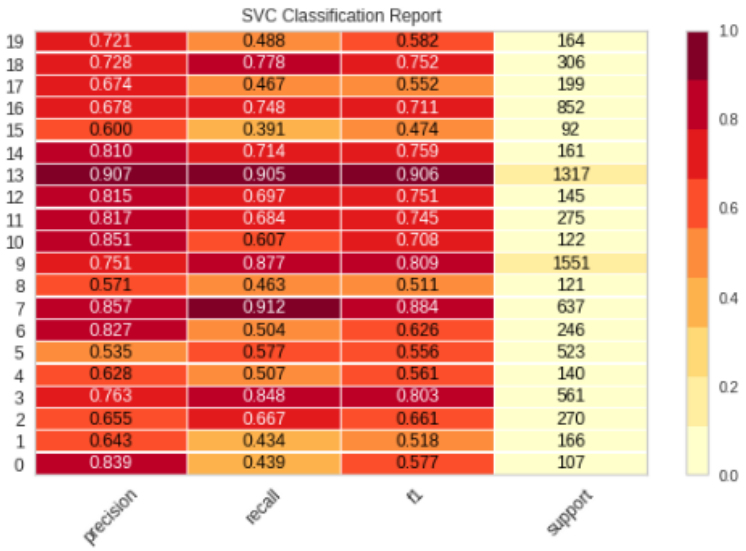
Using 3 models and getting the Champion model based on the following:

- the classifier is very conservative - does not risk too much in saying that a sample
- The model maximizes the number of True Positives
- high F-score can be the result of an imbalance between Precision and Recall



Champion Model

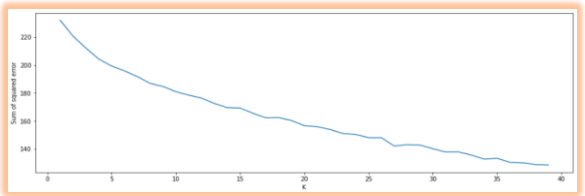


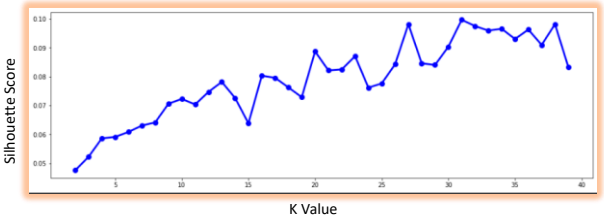


IX) Suggest Based Clustering

Kmeans algorithm were used here to classify the dishes based on the ingredients and the predicted cuisine type that got from the classification step, and TF-IDF used here as features embedding, and the results was as expected there no special K number to chose, but 28 cluster was good according to the figure of WSSE and silhouette score, and it has the highest Silhouette score, this step is aimed at suggest different dishes but using same or similar ingredient, then 5 random dished extracted from same cluster, and will be suggested to the user based on the given ingredients.

Actually our dataset used here is limited to Indian food, but the Indian cuisine already has vast dishes and a lot of ingredients, but add new cuisines from different countries will significantly improve the suggestion quality.





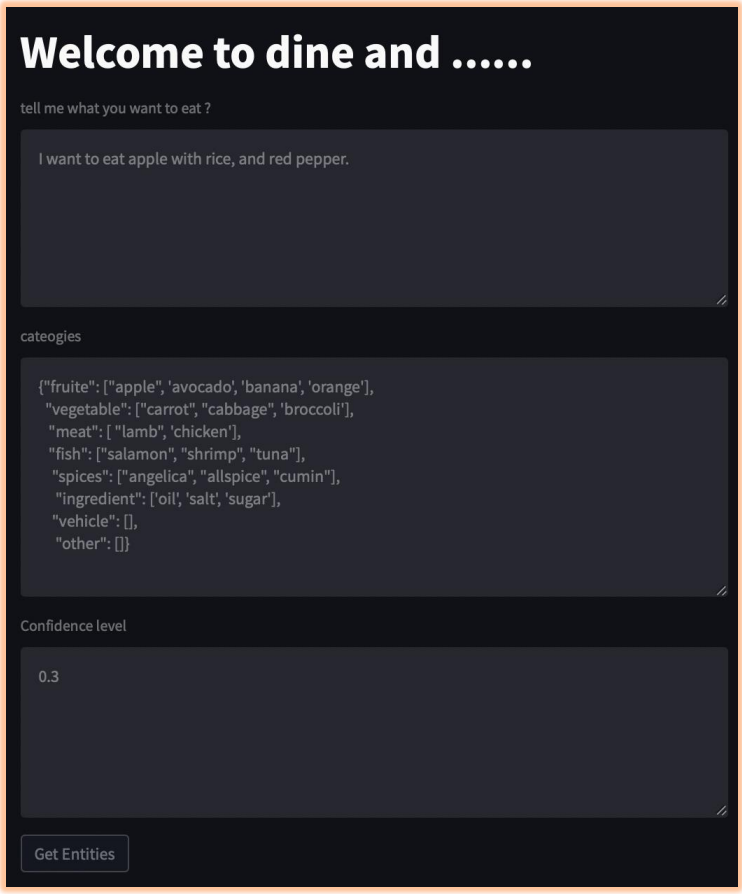
X) Contribution

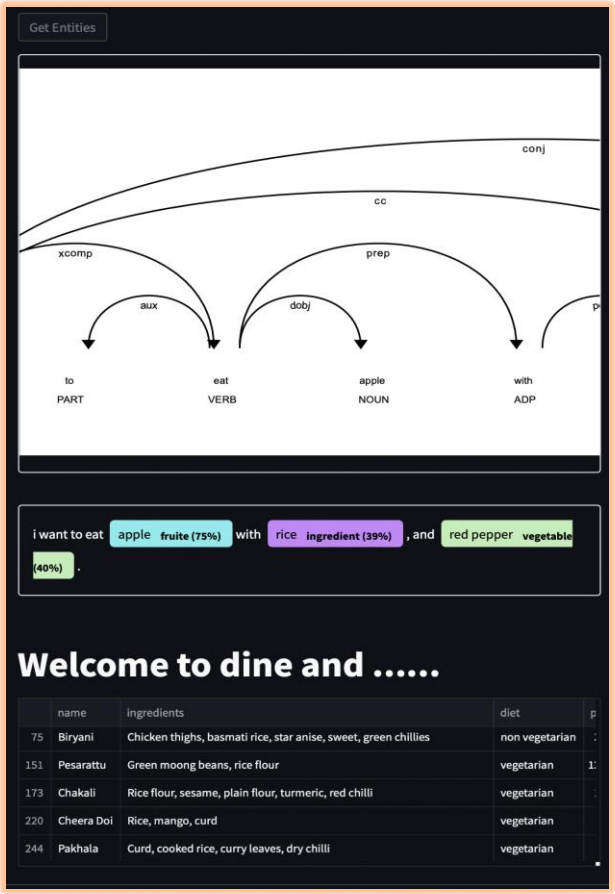
We tried here to provide an innovative framework based on few-shot approach, the used core model here was trained on 100 languages and is therefore also suitable for multilingual zero-shot classification, also POS with Dependency Tree (DT) have used to extract the patterns, so by changing those parts POS & DT to be suitable with other language, the flow will still the same. our demo here works with Arabic language too, but the output of POS is not good because its specified for English language only.

The APP is not specified for foods only, it can work in any other areas without changing anything, which a great start for any project that doesn't has any data.

XI) Live Demo

Proof of concept Demo has developed using Streamlit, for ease of use and test, so you can insert your text and identify the entities with examples, and set the minimum confidence level, and you will show the extracted dependency tree, also the extracted entities with them confidence score, also you will have table of 5 recommended dishes that you can cook based on the extracted entities, and other information like the name & cuisine & Instructions for cook & Ingredient and other.





XII) Limitations

Limitations the POS and DT that has used here are dedicated for English language only, but the APP can be customized for other language by changing those parts, also the core model here mDeBERTa-v3 has really good knowledge representation, but there are other different huge models can be used for better representation. also increasing number of examples of entities or number of entities mean long time in inference, it's a trade-off but it can improve, and the entities should have informative word so the context and meaning can be captured or you got bad results.

XIII) Conclusion

This app opens the horizon to tens of possible ideas and it can involve in a lot of applications, and we aim to improve it by adding new capabilities to work with different languages in automated way, also replace the Dependency Rule Matcher with more advanced techniques, and add some new capabilities to this app like identifying the ingredients that user doesn't like or has allergic for, from the context in automated way, also we can add more advanced features in that app like build chatbot or Q&A, also improve the recommendation part with more sophisticated one for better or more relevant recommendations.

References

- https://www.kaggle.com/datasets/kaggle/recipe-ingredients-dataset?datasetId=683&select=test.json

- https://www.kaggle.com/datasets/pes12017000148/food-ingredients-and-recipe-dataset-with-images?select=Food+Ingredients+and+Recipe+Dataset+with+Image+Name+Mapping.csv

- https://www.kaggle.com/datasets/nehaprabhavalkar/indian-food-101

- https://arxiv.org/pdf/2111.09543.pdf

- https://huggingface.co/MoritzLaurer/mDeBERTa-v3-base-mnli-xnli

- Cuisine Classification with accuracy 78.88% | Kaggle