Recommendation system

In this segment of the project we aim to produce a system where you import the id of the place which you want to present recommendations based on , this system returns a list of the most similar places .

Luckily our data has a description for every different visiting place on the website , we can compute the similarity between the places using text similarity techniques and the higher similarity the more likely that this place appears in the recommendation segment of the webpage .

At first we imported all the project data into one pandas data frame to be able to work with .

We included just four main features or columns in our process ( id , name , information , location , type ) , then noticed there were some problems in the information column , we worked through it using some regular expression to remove the unneeded parts and used strip functions to remove unneeded spaces .

Then we combined all of the columns except for the id into one column named combined to work with .

Then tried some regular expression to remove non-english words .

Using scikitlearn countvectorizer we removed all English stopwords which won’t be of any good to our process and then converted our data frame into sparse matrix

Also from scikit learn we used cosine similarity to compute similarity between the different text vectors

We also tried to use LDA model to generate topics but the accuracy of the first recommendation system was critically better , we think this is due to the small and limited amount of data and information we have .

To make the API we used flask to make an application able to receive get and post requests , on receiving a get request with place id in it , a function is triggered to investigate the similarity and produce a list with the top 10 similar places and return those places and their types .

Chat bot

In this part we tried to train a chatbot to be able to distinguish between topics and know them and according to that present an answer !

Wen provided the chatbot with a json file included different topics patterns for every topic and possible answers , we tried to extract those patterns from a twitter customer service database but majority of the data was about greeting and so on so we chose to write down those data the hard way .

At first we preprocessed our data , doing lemmatization and removing some unnecessary parts .

After that we extracted the files as pickle file , so we can import them in the prediction stage .

We made a training dataset after vectorizing our documents .

And then we initialized a model using sequential from tensorflow, using a 3 layer neural network contains input layer and one dense layer with rel u activation and an output layer using softmax knowing that we have multiple classes ,we trained the model and reached a very high accuracy 96% with a low loss of .09 , which is fine to our purposes .

The we used flask web application to receive post requests included the sent message , the model tries to classify this message , after that we provide responds according to the json file we made earlier !