COOKR HACKATHON

Kitchen Description Builder

1.Problem Statement

Description: Build algorithms to automate text generation for the kitchen type based on the menus that kitchen offers. The summary should be 2-3 lines in the mobile screen width.
For instance:
• This home chef specializes purely Punjabi dishes specifically country / village-style dishes. Generally, these dishes are made with organic ingredients.
• This kitchen brings the best out of Keralite food, very authentic and delicious specials.
• Prema is passionate about making delicacies of Italian food and enjoys cooking unique culinary specials of Italy.

2.Approach

Description: This approach is entirely focusing on the keywords extraction and text generation with the RNN(recurrent neural networks) in NLP which uses texts as inputs for processing the data provided.

- Clean and format dataset for consistency.
- Extract keywords using TF-IDF and NER.
- Create concise prompts with keywords.
- Generate descriptive text using GPT API.

3.Implementation

Data Loading and Preprocessing:

- The code begins by importing necessary libraries such as pandas for data manipulation.
- It loads the dataset from a CSV file named "ssssampled_Swiggy_1_20.csv" into a pandas DataFrame.
- The dataset contains columns named "cuisine", "menu", and "item".
- Missing values are handled by filling them with empty strings.
- The text from the "cuisine", "menu", and "item" columns is concatenated into a single text column named "text".

TF-IDF Vectorization:

- The TfidfVectorizer from scikit-learn is used to convert the text data into a TF-IDF matrix.
- TF-IDF (Term Frequency-Inverse Document Frequency) is a numerical statistic that reflects the importance of a term in a document relative to a collection of documents.
- The fit_transform method of the TfidfVectorizer is applied to the concatenated text to compute the TF-IDF matrix.

Keyword Extraction:
- The feature_names attribute of the TfidfVectorizer is used to get the feature names (i.e., the terms) in the TF-IDF matrix.
- The sum of TF-IDF scores for each term across all documents is computed using the sum(axis=0) method.
- The top 10 keywords with the highest sum of TF-IDF scores are extracted and stored in the top_keywords list.

Prompt Creation:
- The top_keywords list is converted into a single string separated by commas.
- This string is used to create a prompt for text generation using the GPT API.
- The prompt includes the extracted keywords and instructs the GPT model to generate a descriptive summary based on them.

Text Generation with GPT API:
- The OpenAI library (openai) is imported, and the API key is set to authenticate with the GPT API.
- A client object is created using the OpenAI class, specifying the API key.
- The client's completions.create method is called to generate text based on the provided prompt.
- The generated summary is loaded to a dataset of our choice.

4.Challenges Faced

- Accuracy of keyword extraction
- Formulating concise prompts
- API integration and limitations

Kitchen Review Summary

1.Problem Statement:

Description: Build algorithms to automate the generation of summaries for customer reviews. We receive numerous reviews from different customers for various orders, and we need to generate summaries from the hundreds of reviews collected.
For instance:
Review 1: Praises about the quality
Review 2: Praises about the Timeliness
Review 3: Complaints about Quantity
So, the summary review can be - People usually praise this kitchen’s quality,
but they have an issue with the quantity of the food provided. They also appreciate the proper timely delivery from this kitchen.

2.Approach

Description: This approach is centered around automating the generation of review summaries by integrating sentiment analysis techniques with keyword extraction and text generation using both convolutional neural networks (CNNs) and recurrent neural networks (RNNs) in NLP.

- Clean and format the dataset for consistency.
- Categorize reviews as positive, negative, or neutral.
- Extract significant terms using TF-IDF and NER.
- Construct concise prompts with sentiment-labeled keywords.
- Utilize RNNs to generate descriptive summaries based on prompts.

3.Implementation

Data Preprocessing:
- Use Python's pandas library to read and preprocess the dataset.
- Clean the data by removing unnecessary characters, handling missing values, and converting text to lowercase for uniformity.

Sentiment Analysis:
- Implement sentiment analysis using libraries such as NLTK, TextBlob, or VADER.
- Apply sentiment analysis algorithms to classify each review as positive, negative, or neutral based on sentiment scores.

Keyword Extraction:
- Utilize TF-IDF from the scikit-learn library to extract keywords from the reviews.
- Implement NER using libraries like spaCy or NLTK to identify relevant entities in the reviews.

Prompt Creation:
- Combine sentiment labels and extracted keywords to create prompts for text generation.
- Structure prompts to guide text generation based on sentiment and key themes identified in the reviews.

Text Generation with GPT:
- Integrate with OpenAI's GPT API for text generation.
- Send the constructed prompts to the GPT API and retrieve the generated text responses.

4.Challenges in integrating GPT:

-Ensuring prompts capture sentiment and key themes concisely.
-Maintaining coherence and relevance in generated summaries.
-Managing rate limits, errors, and optimizing API usage.