MINI PROJECT REPORT

Data Analytics 2: Text Mining and Natural Language Processing

Name: Utkarsh Sawant

Matriculation No: 11038703

**Task 1: Importing the models and Data**

In this step I started with first going through the open source options to get a audio file. Then I got a speech of a father in church about the Christianity and Jesus Christ. Then with respect to my data I went on to look for a perfect model to turn my audio file to text. I got too many options such as google cloud api, meta api and open ai whisper also. Then after researching about all the models, came to conclusion that whisper is the perfect fit for me. Whisper is open source and provides higher accuracy when handling the large data with background noise. In whisper there are many models such as base, tiny, large, medium and turbo. I choose turbo due to its better speed with data. I referred about the models from the github repo of whisper. Then I installed whisper and ffmpeg (ffmpeg part of whisper model). Later on I converted my audio\_file.mp3 to Wav file because wav file is better in handling the lossless audio quality.then I converted wav file to text file(result\_full.txt)

**Challenges Faced:** While loading the whisper model faced error due to its version controls and took help of GenAi tool and youtube. Also due to the huge audio file of about half hour the model used to run for more than 1 hour to extract the text out of it.

**Task 2: Data Preprocessing**

These is one of the crucial step for handling the data and maintaining the quality of the data. I started preprocessing by removing the unwanted calculations and statistics such as probability of speech ,etc. These were generated due to the Whisper model then I used Trimming method to remove the text after my end line. Then I imported specific libraries for NLP data preprocessing such as punkt and stopwords. Moving ahead I started cleaning the data by first lowering the data so that same words with case sensitivity can he handled easily. Then I used regular expressions to remove punctuations, Special characters and Digits as well. Decision of removing digits was taken after manually going through the data. Then in first instance I removed the stop words and tokenized the words to use them. Later on as I started performing sentiment analysis I got to know negative words like not never should not be removed as they might affect my sentiment analysis part. So to solve these problem I created set of negative words and removed it from stop words then I used it to handle stopwords in my data. These process made sure that negative parts are still remained in the data.

**Challenges Faced:** in the step I faced challenges with the importing of libraries of punkt and stopwords as they were giving me loop up error, later on I solved it by rectifying proper installation process. I initiated further steps without removing negative words which were affecting my sentiment score so I again performed preprocessing and kept negative words to get proper sentiments out of it.

**Task 3: Sentiment Analysis**

In this task, I implemented sentiment analysis using TextBlob, focusing on chunk-based analysis for efficient processing of large text data from a file (result\_tokenize\_v1.txt). TextBlob was chosen for its simplicity and built-in sentiment analysis capabilities, which provide quick polarity and subjectivity scores without the need for extensive model training. Compared to other libraries like VADER or spaCy, TextBlob offers a straightforward setup and is well-suited for general sentiment tasks, making it ideal for this project’s initial exploration of sentiment patterns. I also tried spacy and vader but faced that being the data is very positive in tone(got these from source) still they were identifying it neutral and negative. By splitting the text into 200-word chunks, I enhanced TextBlob's ability to handle long documents, allowing for a nuanced analysis of sentiment at the chunk level. The overall polarity was then calculated by averaging the chunk polarities, with a 0.05 neutral threshold for classifying sentiment as positive, negative, or neutral. Then I went ahead to perform the aspect based sentiment analysis I started importing all the necessary libraries for it then I got the aspects form my data and cross verified it through some gen ai options and almost 80 percent were matching. Then I calculated the aspect based sentiment scores of those aspects(words). **Aspect-Based Sentiment Analysis** (ABSA) is valuable for gaining a deeper, more granular understanding of sentiment by focusing on specific aspects or features within a text, rather than analyzing sentiment at a general level. Then I tried experimenting with it and got the emotions score of the data by using transformers. I used these model to get the idea of emotions of my data “j-hartmann/emotion-english-distilroberta-base”. In the end I added some visualisation to complete the sentiment analysis task.

**Challenges Faced:**

Some challenges included deciding on an optimal chunk size and adjusting the neutral threshold for best results, as these parameters affect the sensitivity of the analysis. TextBlob’s basic sentiment analysis can be limited in detecting nuanced or context-specific sentiments, especially in complex texts with sarcasm or slang, requiring careful decision-making around chunk size and threshold settings. Additionally, TextBlob’s pre-trained model may not perform as accurately on specific jargon or idiomatic expressions, necessitating further experimentation with thresholds or considering other NLP tools for more refined analyses.

**Task 4: Topic Modeling**

In this task, I implemented Latent Semantic Analysis (LSA) to extract topics from a text document using Python. LSA was chosen for its effectiveness in uncovering latent themes in unstructured text, which is particularly useful when content themes overlap.. Then I split the data into 300 word chunks. I used TfidfVectorizer instead of CountVectorizer to convert the text into a numerical representation, capturing term importance through TF-IDF scores, which improved topic differentiation for the LSA model. With the text converted into a TF-IDF matrix, I applied Truncated Singular Value Decomposition (TruncatedSVD) as the core of the LSA process, specifying five topics. I tried manipulating the number of topics and later finalised 5 topics, providing distinct themes within the Text. Then I performed some visualizations,such as heatmaps showing topic distribution across documents and the word cloud for each topic showcasing all the words into the topics. I explored alternatives like using different chunk sizes and methods, but the final method yielded the most interpretable topics, providing an effective thematic analysis.

**Challenges Faced:**

Key challenges included determining an optimal chunk size to balance and specificity in topic extraction Another challenge was converting the tokenized data to align with the TfidfVectorizer format for effective LSA processing, ensuring that the numerical representation captured the document’s thematic essence without sacrificing model performance. Managing topic overlaps was another complexity, as some themes appeared across multiple topics.

**Task 5: Summarization**

Here I started by using transformer model then I imported the pipeline for the summarization. Then while choosing the model I tried and tested various model 1 from facebook , few from google and open ai also. Later my result showed the Google t5 base gives the best result. I evaluated the result by bleu score and rogue method and found that these model is giving better result when compared to facebook bart , peagisus google and etc. In task I also divided the data into chunks because the model is not able to handle the all the text file at once. I kept the parameters of max to 150 and min to 50 these were the best suitable parameters based on my analysis.  
**Challenges Faced:**

While evaluating the best fit model for summarization it was very long and time taking process as the size of data is also huge. In models initially I suffered the limit but later I converted it to chunks to handle that limit. In the evaluation part getting the proper method was also a bit of concern.  
  
**Conclusion:**

In summary, this project effectively demonstrated an end-to-end NLP pipeline, from transcribing audio to extracting insights from text. Starting with Whisper’s Turbo model for accurate transcription, I handled extensive preprocessing to retain sentiment-relevant details. Sentiment analysis with TextBlob and aspect-based techniques provided nuanced sentiment insights, while LSA-based topic modeling identified thematic patterns, despite some topic overlap. For summarization, Google’s T5 model proved optimal, delivering concise outputs with coherent summaries. This project strengthened my proficiency in NLP tools and methods, equipping me with practical skills for analyzing and interpreting large, unstructured text data.