ChatGPT

standup-nlp-portfolio

Project Overview

In this project I demonstrate an end-to-end natural language processing (NLP) pipeline using stand-up comedy transcripts. It showcases the full workflow from data collection and cleaning to analysis, visualization, and generation. I use Python (3.8+) and common NLP libraries to process transcripts of comedy routines. The project highlights skills in web scraping, text normalization, exploratory data analysis, sentiment analysis, Markov chain text generation, and topic modeling.

📚 Notebooks

- **Data Cleaning & Transformation:** I scrape stand-up comedy transcripts from web sources and clean the text data. This notebook handles text normalization steps like lowercasing, removing punctuation, and filtering unwanted content.
- **Exploratory Data Analysis:** I analyze word frequencies and create visualizations such as word clouds. This includes counting words, highlighting common terms, and examining profanity usage across the transcripts.
- **Sentiment Analysis:** I use TextBlob to compute polarity and subjectivity scores for the transcripts. The notebook plots sentiment trends and distributions to reveal how the comedic tone varies over time or between shows.
- **Text Generation:** I implement a Markov chain model on Ronny Chieng's transcript to generate new text. This notebook constructs a network of probable next-word transitions and demonstrates simple NLP-based text generation.
- **Topic Modeling:** I apply Latent Dirichlet Allocation (LDA) using Gensim to discover themes in the comedy corpus. The notebook preprocesses the data into a document-term matrix and interprets the main topics found.

📊 Key Visualizations

A word cloud is used to highlight the most frequent words in the corpus. This visualization quickly shows dominant terms from the stand-up transcripts and the themes they suggest. The example word cloud below illustrates common words extracted from all cleaned transcripts:

Word Cloud Example
Word cloud of the most frequent terms in the stand-up comedy transcripts.

To analyze sentiment, I plot how the polarity (positive/negative sentiment) of the jokes changes over time or across comedians. The sentiment trend plot shows average sentiment scores computed by TextBlob for each transcript segment. The figure below displays the overall sentiment polarity trend across the dataset:

Sentiment Trend
Sentiment polarity trend across the stand-up transcripts.

For text generation, the Markov chain model's word transitions are visualized as a network graph. I identify the top probable next-word links from a given word, which reveals common word transitions in the text. The network graph below shows the strongest word-to-word connections inferred by the Markov model:

Markov Chain Network
Network graph of top next-word transitions in the Markov chain model.

⚙ Setup & Installation

- Clone the repository and navigate into it:

```
git clone https://github.com/dhillonarman/standup-nlp-portfolio.git
cd standup-nlp-portfolio
```

- Install **Python 3.8+** if it's not already installed.
- (Optional) Create and activate a virtual environment:

```
python3 -m venv venv
source venv/bin/activate   # On Windows use `venv\Scripts\activate`
```

or using conda:

```
conda create -n standup-nlp python=3.8
conda activate standup-nlp
```

- Install the required Python packages:

```
pip install -r requirements.txt
```

► Running the Notebooks

1. Activate your Python virtual environment if you created one.
2. Launch Jupyter Notebook from the repository directory:

```
jupyter notebook
```

3. In the browser interface, open the desired `.ipynb` notebook (e.g., `Data_Cleaning_Transformation.ipynb`).
4. Run the notebook cells sequentially to reproduce the analysis and see the results.

requirements.txt

Below are the main packages listed in `requirements.txt` for reference. These dependencies support the data processing and analysis tasks in the project.

```
beautifulsoup4
gensim
jupyter
markovify
matplotlib
networkx
nltk
pandas
requests
seaborn
scipy
textblob
wordcloud
```
