Topic Modelling
Project Overview

Business Context

With the advent of big data and Machine Learning along with Natural Language Processing, it has become the need of an hour to extract a certain topic or a collection of topics that what document is about. Think when you have to analyze or go through thousands of documents and categorize under 10 – 15 buckets. How tedious and boring will it be ?

Thanks to Topic Modeling where instead of manually going through numerous documents, with the help of Natural Language Processing and Text Mining, each document can be categorized under a certain topic.

Thus, we expect that logically related words will co-exist in the same document more frequently than words from different topics. For example, in a document about space, it is more possible to find words such as: planet, satellite, universe, galaxy, and asteroid. Whereas, in a document about the wildlife, it is more likely to find words such as: ecosystem, species, animal, and plant, landscape. A topic contains a cluster of words that frequently occurs together. A topic modeling can connect words with similar meanings and distinguish between uses of words with multiple meanings.

A sentence or a document is made up of numerous topics and each topic is made up of numerous words.

Data Overview

The dataset has odd 25000 documents where words are of various nature such as Noun,Adjective,Verb,Preposition and many more. Even the length of documents varies vastly from having a minimum number of words in the range around 40 to maximum number of words in the range around 500. Complete data is split 90% in the training and the rest 10% to get an idea how to predict a topic on unseen documents.

Objective

To extract or identify a dominant topic from each document and perform topic modeling.

Tools and Libraries

We will be using Python as a tool to perform all kinds of operations.
Main Libraries used are

- Pandas for data manipulation, aggregation
- Matplotlib and bokeh for visualization of how documents are structured.
- NumPy for computationally efficient operations.
- Scikit Learn and Gensim packages for topic modeling
- nltk for text cleaning and preprocessing
- TSNE and pyLDAvis for visualization of topics

Approach

Topic EDA

- Top Words within topics using Word Cloud
- Topics distribution using t-SNE
- Topics distribution and words importance within topics using interactive tool pyLDAvis

Documents Pre-processing

- Lowering all the words in documents and removing everything except alphabets.
- Tokenizing each sentence and lemmatizing each word and storing in a list only if it is not a stop word and length of a word is greater than 3 alphabets.
- Joining the list to make a document and also keeping the lemmatized tokens for NMF Topic Modelling.
- Transforming the above pre-processed documents using TF IDF and Count Vectorizer depending on the chosen algorithm

Topic Modelling algorithms

- Latent Semantic Analysis or Latent Semantic Indexing (LSA)
- Latent Dirichlet Allocation (LDA)
- Non-Negative Matrix Factorization (NMF)
- Popular topic modelling metric score known as Coherence Score
- Predicting a set of topics and the dominant topic for each documents
- Running a python script end to end using Command Prompt

Code Overview

1. Complete dataset is splitted into 90% for training and 10% for predicting unseen documents.
2. Preprocessing is done to avoid noise
    - Lowering all the words and replacing words in their normal form and keeping only alphabets.
    - Making a new document after tokenizing each sentence and lemmatizing every word.
4. For LSA and LDA Topic Modeling
    - TF IDF Vectorizer and Countvectorizer is fitted and transformed on a clean set of documents and topics are extracted using sklean LSA and LDA packages respectively and proceeded with 10 topics for both the algorithms.
5. For NMF Topic Modeling
    - TF IDF Vectorizer is fitted and transformed on clean tokens and 13 topics are extracted and the number was found using Coherence Score.
6. Topics distribution is analyzed using t-SNE algorithm and iterative tool using pyLDAvis.
7. For unseen documents, topics were predicted using the above three algorithms.

Modularized code

The ipython notebook is modularized into different functions so that the user can use those functions instantly whenever needed. The modularized code folder is structured in the following way.

```
input
  |__documents.csv
src
  |__engine.py
  |__ML_pipeline
           |__utils.py
           |__dataset.py
           |__pre_processing.py
           |__vectorizing_dataset.py
           |__lsa_model.py
           |__predict_lsa.py
           |__topic_modeling.py
           |__predict_topic.py
           |__tuning_lda.py

lib
  |__Topic_Modeling.ipynb

output
  |__lda_trained.pkl
  |__lsa_model_trained.pkl
  |__nmf_trained.pkl
  |__lda_trained.pkl
  |__train_documents.csv
  |__test_documents.csv
  |__test_lda.csv
  |__test_nmf.csv
```

Once you unzip the modular_code.zip file you can find the following folders within it.

1. input
2. src
3. output
4. lib

1. The input folder contains all the data that we have for analysis. In our case, it will contain a csv file called documents.csv
2. The src folder is the heart of the project. This folder contains all the modularized code for all the above steps in a modularized manner. It further contains the following.
    a. ML_pipeline
    b. engine.py

The ML_pipeline is a folder that contains all the functions put into different python files which are appropriately named. These python functions are then called inside the engine.py file

3. The output folder contains all the models that we trained for this data saved as .pkl files and the output data frames stored as .csv files. These models can be easily loaded and used for future use and the user need not have to train all the models from the beginning.

4. The lib folder is a reference folder. It contains the original ipython notebook that we saw in the videos.

Project Takeaways

- Understanding the problem statement
- How and what kind of text cleaning needs to be done
- What tokenization and lemmatization is
- Performing EDA on documents word and POS counts, most occurring words
- Types of vectorizer such as TF IDF and Countvectorizer
- Understanding the basic math and the working behind various Topic Modeling algorithms
- Implementation of Topic Modeling algorithms such as LSA(Latent Semantic Analysis), LDA(Latent Dirichlet Allocation), NMF(Non-Negative Matrix Factorization)
- Hyper parameter tuning using GridSearchCV
- Analyzing top words for topics and top topics for documents
- Distribution of topics over the entire corpus
- Visualizing distribution of topics using TSNE
- Visualizing top words in a topic using WordCloud
- Visualizing the distribution of topics and the occurrence and weightage of words using interactive tool which is pyLDAvis
- Comparing and checking the distribution of the topics using metrics such as Perplexity and Coherence Score
- Training and predicting the documents using LDA and NMF in a modular code using python script.