# README

Project 5 for CS 412 – Introduction to Machine Learning (University of Illinois at Chicago)
Name : Tanveer Shaikh – tshaik4@uic.edu
Github repo link: https://github.com/tanveertshaikh/LDA-topic-modeling

(A) My data is the collection of all the news articles published by Vox before March 2017. I am only extracting 'title' + 'blurb' from the entire data. My task is to figure out the 'categories' of the news articles marked as "The Latest".

(B) I am using Topic Modeling using Latent Dirichlet Allocation (LDA) as a solution because this is an unsupervised ML problem. I chose LDA because it is highly praised by researchers since it was published in 2003. It was one of the best probabilistic graphical models for my application/task.

(C) I chose to evaluate success based on Perplexity Score and Coherence Score. The lower the perplexity score, the better the model can predict the correct topic. A higher coherence score is always preferred.

(D) I decided to choose Jupyter Notebook as it is flexible, and I could execute blocks of code on the fly. I wanted to do this project on Python3 as the Gensim and spaCy library of python3 are very popular LDA and Natural Language Processing (NLP).

(E) I got pretty good results by training the model with 100 unique topics. This value was chosen by cross-validation over coherence scores. The evaluation metrics given by my optimized model are as follows:

- Perplexity Score: -22.93
- Coherence Score: 0.575

(F) It is evident from the image below of the topic clusters that they are overlapping. This is an incorrect prediction of topic in most of the cases and the model is bound to make errors. However, the area which is covered by just a single cluster will give accurate topic to unseen documents.

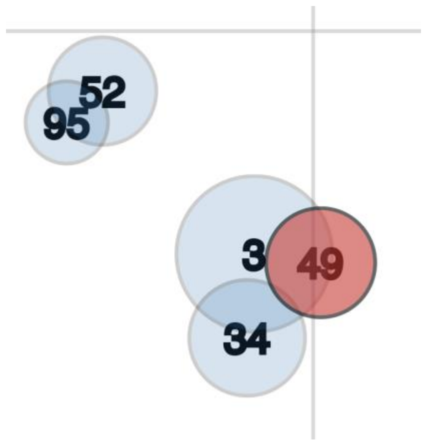
In the future, I would like to also include bi-gram tokens to my document vectors as they will surely increase the coherence score. Moreover, I would like to use a smarter implementation of the LDA algorithm like the Mallet's LDA implementation. It will give better results and it would also train faster. http://mallet.cs.umass.edu/dist/mallet-2.0.8.zip

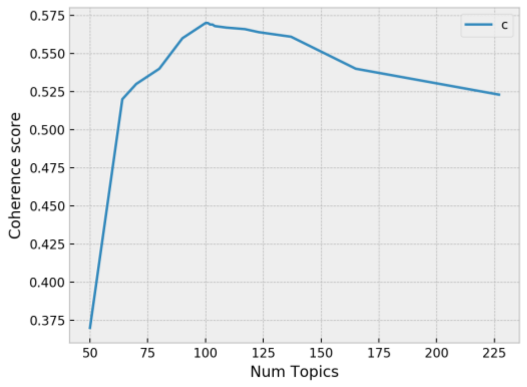
**Data Pre-Processing –**

**Preliminary:** Loaded Data → Dropped unwanted columns → Removed missing values → Removed all non-ascii and non-Unicode characters → Removed duplicates

**Actual Pre-Processing:** Text cleaning → Tokenization → Lemmatization → Stopwords Removal

**Experimental Setup:** Train : Dev : Test = 72% : 9% : 19%





Various LDA models with different number of topics (k) were tested and their coherence scores were plotted; where k is a hyperparameter. We are supposed to choose that k which gives the highest coherence score from the above learning curve.

Choosing a 'k' that marks the end of a rapid growth of topic coherence usually offers meaningful and interpretable topics. Picking an even higher value can sometimes provide more granular sub-topics. Picking a higher stable value will lead to overfitting (more variance); whereas, picking a small value will underfit our model (more bias).

If you see the same keywords being repeated in multiple topics, it's probably a sign that the 'k' is too large.

**By observing the graph above, the optimum value of number of topics is chosen to be 100 topics.**

If the coherence score seems to keep increasing, it may make better sense to pick the model that gave the highest coherence value before flattening out. This is exactly the case here.

## CREDITS

This project wouldn't have been possible without these python3 libraries:

| | |
|---|---|
| nltk (Natural Language Toolkit) | matplotlib |
| spaCy (NLP Engine) | scikit-learn |
| pickle | seaborn |
| numpy | pandas |
| Gensim | pyLDAvis |