Name: Charita Ramnathsing

Cohort: Oct 2019

Mentor: Dipanjan Sarkar

Report Capstone I - Findings of The Machine Learning Model with ‘LDA\_Mallet’

In respect to the last chapter of this Capstone, I would like to highlight a few findings and explain what the motivation is behind the choosing the machine learning model: LDA\_Mallet. In regard to this, these following paragraphs are merely meant to explain the use of this model in a hypothetical way before actual using it. The paragraphs are sequentially chosen since it’s important to make data ready for eventually use the model. These are the steps taken:

# First: Clean

As for all text data used in NLP projects, it is important to clean the data as good as possible. The less noisy your data is, the better results you can expect. This is just because of the simple fact that you give the model a clear understanding of what is important in your data.

As far as ‘text’ predictions in general, there is a lot written about suggestion to handle problems for text interpretation. There are also further extracting and predicting models using ‘Deep Learning’ as a basis. I have not used these models yet, and therefore might be out of the scope of these findings. Although it must be said that this sensation gives apparently good results for more complex issues related to text automation in, e.g. therapy use, job application optimization, text summarization.

For cleaning the data, I have used different modules from the open source library ‘NLTK’. The NLTK library enhances the process of text for use with Python programming. It has many popular modules[[1]](#footnote-1) for specific language processing work:

A screenshot of a cell phone

Description automatically generated

NLTK is also functioning as an API for use with other lexical databases like WordNet. We can see WordNet as a lexical database for the English language, in which words are linked to semantic relations like synonyms and antonyms. In my notebook I have used WordNet because this is another way to make the used text/data, and later corpus, less noisy. After cleaning the data by using the different modules of ‘nltk’, it is ready to make a good usable dictionary from it in the form of the pandas ‘DataFrame’ for further manipulation.

# Second: Process with Dictionary

There are different open source libraries for topic modeling and Gensim is one of them. I have chosen for this because Gensim has a good and vast framework for unsupervised topic modeling and natural language processing, using modern statistical machine learning. Gensim is designed to handle large text collections using data streaming and incremental online algorithms, which differentiates it from most other machine learning software packages that target only in-memory processing.

Before we can make use of the specific models in Gensim, there is a way to transform the text into a well-defined number based on calculations and vectorization. The following has to be explained for going further:

* Document – the text you would like to have modelled
* Corpus – collection of documents (in the picture ‘Document..’
* Vector – mathematically convenient representation of a document based on frequency

A screenshot of a cell phone

Description automatically generated

In order to make computing exercises on text, it’s important that the model can read the texts as numbers. It would be difficult for the model otherwise to make understandable calculations and predictions. For generating these numbers based on a field of text, we can use a ‘Dictionary’ from the Gensim package. This encapsulates mapping between normalized words and their integer id’s. *The main* ***function*** *is doc2bow, which converts a collection of words to its bag-of-words representation: a list of (word\_id, word\_frequency) 2-tuples.* The module ‘Dictionary’ makes use of a doc2bow-format and that is again good for making use of the ‘LDA\_mallet’ model used in my notebook. See here an extract of the process:

A screenshot of a cell phone

Description automatically generated

After the dictionary and the bag-of-words has been generated we can make use of specific models offered in gensim. A model is an algorithm for transforming vectors from one representation to another. Gensim offers several vector-space model algorithms, one of them is ‘TfIdf’. The TfIdf-model transforms vectors from the bag-of-words representation to a vector space where the frequency counts are weighted according to the relative rarity of each word in the corpus itself. This is great because similarly we can use the made-up TfIdf as ‘bow\_corpus’ in the notebook for ultimately making use of the model of gensim, since it has the ‘TfIdf’-format.

From different models in gensim, there are two which are important to mention since they have the same latent use of finding meaningful structure but use different algorithms to model:

* LSI : Latent Semantic Indexing
* LDA : Latent Dirichlet Allocation

LSI transforms documents from either bag-of-words or (preferably) TfIdf-weighted space into a latent space of a lower dimensionality.

LDA is a probabilistic extension of LSA (also called multinomial PCA), so LDA’s topics can be interpreted as probability distributions over words. These distributions are, just like with LSA, inferred automatically from a training corpus. Documents are in turn interpreted as a (soft) mixture of these topics (again, just like with LSA) but then in a probability structure when trained. The LDA\_Mallet model is a wrapper for LDA. ‘Mallet’[[2]](#footnote-2) is a Java-based package for statistical natural language processing, document classification, clustering, topic modeling, information extraction, and other machine learning applications to text.

# LDA\_MAllet

Compared to other topic modelling algorithms, the MALLET topic model package includes an extremely fast and highly scalable implementation of Gibbs sampling, efficient methods for document-topic hyperparameter optimization, and offer tools for inferring topics for new documents given trained models. Because of this, we can create a nice optimized model which infers the latent structure of the document (text).

What are the parameters of this model?

There are a few parameters needed before it can start working:

* mallet\_path = where can the mallet installation function be found, meaning the package should be installed first
* corpus = collection of text in bag-of-word format
* Id2word = mapping between token ids and words
* num\_topics = number of topics that you think / calculated for optimal results
* iterations = amount of iterations needed for each trained optimal model
* workers= number of threads that will be used for the training

We can see the results of this model:

A screenshot of a social media post

Description automatically generated

We see the contribution that each topic makes towards a document. Eg. Document 3, has as Dominant Topic number ‘3’, which consist of a distribution of the words: law, solidarity, society, trade, justice, mankind, latin\_america, reason, case, freedom. In each topic there is a latent structure and dependency between the words which make the topic unique. The contribution of this topic towards the document is 41.61%. That means that the words used for making this topic, are visible for 41.61% in this specific document.

The model was chosen instead of the ‘regular’ LDA-model because of the accuracy and the speed. I did run the corpus through a ‘regular’ LDA-model, but the obtained a much higher coherence with the LDA\_Mallet.

A screenshot of a cell phone

Description automatically generated

I am very proud of this model because the chosen topics by the model make sense for most of them. It is still hard to make a model with 99.00% accuracy in NLP but hopefully somewhere in the near future it will be possible.

1. http://www.nltk.org/book/ch00.html#tab-modules [↑](#footnote-ref-1)
2. http://mallet.cs.umass.edu/ [↑](#footnote-ref-2)