Analysis on Beijing Policy Data

Jinhua Su *
*Renmin University in China

$\{2017201620\}@ruc.edu.cn$

I. Introduction

This paper is aiming to show my qualification on Data Crawling and Text Analysis. The specific contents of this study are as followed:

- Use **Scrapy** to crawl HTML data.
- Use **MongoDB** to store data.
- Use **Jieba** to cope with Chinese Word Segmentation.
- Use **Sklearn** to extract major feature from text and apply some simple Mechine Learning Models.
- Use **Gensim** to train the LDA model for topic extraction.
- Use **Matplotlib** to visualize a few results.

The code I use are open source in Github.[1]

II. Crawling Data

A. *Scrapy*

I use Scrapy[2] to crawl the news from Beijing Government Website. Scrapy is an application framework for crawling web sites and extracting structured data which can be used for a wide range of useful applications, like data mining, information processing or historical archival.

Since I have read its document before, I could deploy my spider app in a few lines of codes. Unlike JD.com or Taobao.com, Beijing Government Website has a fairly simple front-end structure, where I can get all data in the json format. Finally I got 500 pieces of news.

B. *MongoDB*

As for web app, NoSQL is more and more popular for its flexibility. Nearly 50 percent of website use MongoDB as major database, I also choose it to store my crawling data in my Ubuntu system. MongoDB is a general purpose, document-based, distributed database built for modern application developers and for the cloud era. No database makes you more productive.

For each news data, MongoDB will give an unique ID. The advantage of MongoDB is that I needn't to design several table for DB, that is to say, I can save totally different organized data from different government website in the same collection. I can organize it later and output it to a common format as I like. For submission, I use pandas to convert mongoDB data into csv format.

III. Data Processing

A. *Jieba*

The tokenization of Chinese is much more complicated than English. To tokenize English words, we just need to split words in sentence by blank or punctuation. Chinese doesn't have blank between words. An additional step of tokenization is, therefore, needed.

Jieba(Chinese for ”to shutter”) Chinese text tokenization is a Chinese word tokenization module. The algorithm of Jieba is probability language modeling. It generates a trie tree based on a dictionary transcendentally and also calculate the frequency of words in the dictionary. When dealing with the sentence that is needed to be tokenized, it generates a DAG(Directed Acyclic Graph) to record every possible tokenization.A DAG is a dictionary, where the keys are the starting position of a word in the sentence and the values are lists of possible ending position.

For every possible words in the DAG, Jieba calculates their probability based on the transcendental dictionary. Then it find the path with the largest probability from the right side of the sentence to the left side. This largest probability path gives us the most possible tokenization.

In the case where the sentence includes words that are not in the dictionary, Jieba uses HMM(Hidden Markov Model) and Viterbi algorithm to tokenize. Every character has four conditions based on its possible condition in a word: B(Begin), M(Middle), E(End) and S(Single). The process of tokenizing words not in the dictionary is based on their conditions mainly. With three probability tables from the training of a large amount of texts, Jiaba then applies Viterbi algorithm to calculate the most possible condition of a word and uses the conditions chain to tokenize.

IV. Data Mining

A. *Frequency Analysis*

After word segmentation, the easiest way to describe the mathematical feature of text data is to count the words. To visualize the result, I choose word cloud.

A word cloud is a novelty visual representation of text data, typically used to depict keyword metadata (tags) on websites, or to visualize free form text. Tags are usually single words, and the importance of each tag is shown with font size or color. This format is useful for quickly perceiving the most prominent terms to determine its relative prominence. When

[1] https://github.com/JinhuaSu/Sample$_{task}$
[2] https://scrapy.org/

used as website navigation aids, the terms are hyperlinked to items associated with the tag.

Fig. 1.

B. Topic Model

In machine learning and natural language processing, a topic model is a type of statistical model for discovering the abstract ”topics” that occur in a collection of documents. Topic modeling is a frequently used text-mining tool for discovery of hidden semantic structures in a text body. Intuitively, given that a document is about a particular topic, one would expect particular words to appear in the document more or less frequently: ”dog” and ”bone” will appear more often in documents about dogs, ”cat” and ”meow” will appear in documents about cats, and ”the” and ”is” will appear equally in both. A document typically concerns multiple topics in different proportions; thus, in a document that is 10 percent about cats and 90 percent about dogs, there would probably be about 9 times more dog words than cat words. The ”topics” produced by topic modeling techniques are clusters of similar words. A topic model captures this intuition in a mathematical framework, which allows examining a set of documents and discovering, based on the statistics of the words in each, what the topics might be and what each document’s balance of topics is.

I use LDAmodel of Gensim to train my topic model. the latent Dirichlet allocation (LDA) is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. For example, if observations are words collected into documents, it posits that each document is a mixture of a small number of topics and that each word’s presence is attributable to one of the document’s topics. LDA is an example of a topic model and belongs to the machine learning toolbox and in wider sense to the artificial intelligence toolbox.

C. Text Cluster

1) TFIDF: In information retrieval, tf–idf or TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.[1] It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. tf–idf is one of the most popular term-weighting schemes today. A survey conducted in 2015 showed that 83 percent of text-based recommender systems in digital libraries use tf–idf.

TABLE I
THE REGRESSION RESULT OF THE STANDARD SENTIMENTAL FACTOR WITH SSE

Topic	Compositions ***
1	(0.36*+0.12*)
2	0.45*+0.29*
3	(0.33*+0.31*)

$$tf(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{max(f_{t',d:t'ind)}(1)}$$

$$idf(t,D) = \log \frac{N}{f_{t',d:t'ind)}(2)}$$

2) Cluster: Text clustering is the application of cluster analysis to textual documents. It has applications in automatic document organization, topic extraction and fast information retrieval or filtering.

I use TF-IDF matrix as variables, but it has 1e7 columns. Therefore I use PCA to decrease dimensions and use K-means to cluster. However for its high dimension, I could not visualize it.

TABLE II
THE REGRESSION RESULT OF THE STANDARD SENTIMENTAL FACTOR WITH SSE

Cluster Type	count ***
1	172
2	131
3	107
4	90

D. Relation with stock market

Policy will have an effect on stock market. Since I find most policy file about the Coronavirus disease. To stimulate economic recovery, some policy about cutting the interest rates have been promoted. I use Tushare to get the market data to find relation between policy and stock market.

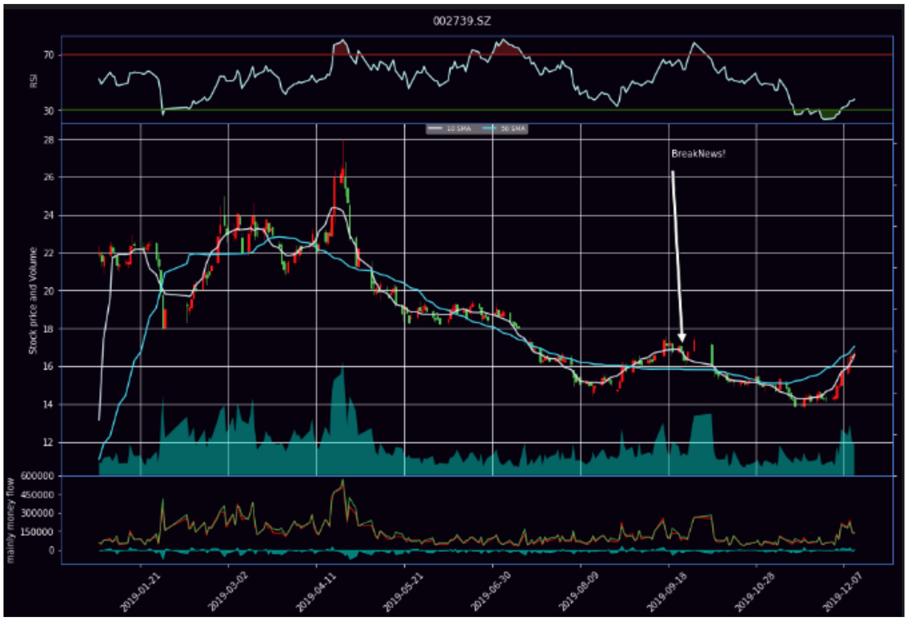
002739.SZ
RSI
70
30
10 SMA
50 SMA
BreakNews!
Stock price and Volume
28
26
24
22
20
18
16
14
12
mainly money flow
600000
450000
300000
150000
0
2019-01-21
2019-03-02
2019-04-11
2019-05-21
2019-06-30
2019-08-09
2019-09-18
2019-10-28
2019-12-07

Fig. 2.