Natural Language Processing for Lithuanian language

Mindaugas Venckus

2017 m. gegužės 26 d.

The main intention of this research is to study and learn natural language processing (NLP) principals for Lithuanian language. It is interesting to analyze classical NLP methods and see how they work on it, so in this work I implemented text classification, topics extraction, search query and clustering ideas.

Santrauka

1 Data

Data analysis can't be established without having textual data, due to that my work started from getting raw data from most popular news website www.delfi.lt. I decided to crawl articles from 5 categories (Criminals, Music, Movies, Sports, Science) such approach helps me to understand the performance of NLP methods better. Below basic statistics of data that was analyzed is presented 1.

1 lentelė: Documents

	documents count
criminals	227
movies	167
music	120
science	204
sports	136

2 Preprocessing

At first we need to convert unstructured data (raw text) to structured (matrix based approach) - such transformation is friendly for machine learning algorithms. Preprocessing goal is to convert unstructured data to structured matrix

where rows are document id, columns - token, values - term frequency inverse document frequency (TFIDF) statistic.

$$\text{tfidf\_matrix} = \begin{array}{c} \\ doc_1 \\ doc_2 \\ doc_3 \\ . \\ doc_n \end{array} \begin{array}{c} \begin{array}{ccccc} token_1 & token_2 & token_3 & . & token_n \end{array} \\ \begin{pmatrix} 0 & 0 & & . & 0 \\ 0 & & 0.14 & . & 0 \\ 0 & 0 & 0 & . & 0 \\ . & . & . & . & . \\ 0 & 0.011 & 0 & . & 0 \end{pmatrix} \end{array}$$

2.1 Tokenization

Suppose we have the sample of raw text bellow:
Pranešimas gautas apie 14.14 val. Kaune, Švenčionių g. prie Nemuno upės ant kranto rastas žmogaus kūnas su galimai durtine žaizda krūtinėje. Aplinkybės tikslinamos. Tokenization is the process of breaking a text into smaller words, phrases and symbols. In this case we are trying to break text into the words.

1. Break text into the words using regex expression W+ which breaks text by any non-word character.
   (a) Remove digits
   (b) Remove words smaller than 4 letters
   (c) Remove stop words (unnecessary/common words)
   (d) Remove word endings

Text above is transformed to list representation t = [aplinkyb, durtin, gaut, kaun, krant, krūtin, kūn, nemun, praneš, rast, tikslinam, up ,švenčion, žaizd, žmog]

2.2 TFIDF

TFIDF 1 is statistic which measure token importance in document with relation to the document of corpus.

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D) \tag{1}$$

where term frequency $tf(t, d)$ is a measure of how many times token t occurs in the document d and inverse document frequency $idf(t, D)$ is a measure of how much information the word provides, that is, whether the term is common or rare across all documents.

$$idf(t, D) = \log \frac{N}{|d \in D : t \in d|} \tag{2}$$

with

1. N - total number of documents in corpus $N = |D|$
2. $|d \in D : t \in d|$ - number of documents where term t appears.

TFIDF intuitive explanation - importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus.

2.3 TFIDF matrix construction

Main steps before TFIDF matrix construction are:

1. For each document 90% - train set, 10% - test set.
   (a) Collect tokens for each document separately.
   (b) Collect unique tokens.
   (c) Collect document categories (will be used for classification performance measuring)
   (d) Collect document titles (will be used for clustering)

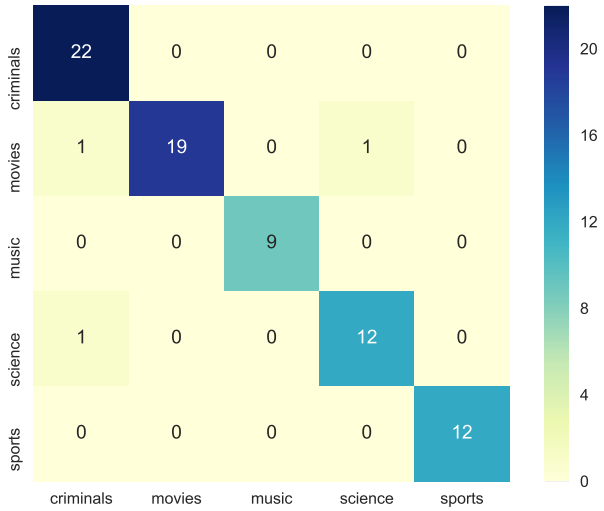
TFIDF matrix rows are all documents and columns unique tokens filled with TFIDF statistic. Additionally we reduce matrix dimension by removing tokens (entire column) which appears among documents rarely, more precisely once a time.

3 Classification

Main idea of classification is to convert unseen text to vectorized token representations where tokens are taken from TFIDF matrix and do multiplication between vector and mentioned matrix. After that use k-nearest neighbors approach to make prediction for unseen document by taking mode of most similar k documents categories.

3.1 Classification results

Classification performance is measured using confusion matrix where rows are true category and columns predicted category. The Figure 1 shows the results of categories prediction. For example for 21 movies got 1 criminals, 1 science and 19 movies predictions. In overall such approach reach above 90% recall and 90% precision.



1 pav.: Classification performance

4 Topics extraction

Latent semantic analysis (LSA) is a technique in natural language processing, in particular distributional semantics, of analyzing relationships between a set of documents and the terms they contain by producing a set of concepts related to the documents and terms. Let's mark tfidf matrix as X, after applying singular value decomposition (SVD) we get

$$A = U\Sigma V^T \tag{3}$$

Let A be a $m \times n$ matrix with column vectors $a_1, a_2, ..., a_n$. In the SVD of, U will be $m \times m$, Σ will be $m \times n$ and V will be $n \times n$. Dimensionality reduction is done by keeping k biggest values from Σ i.e $\sigma_1, .., \sigma_k \neq 0$; $\sigma_{r+1}, ..., \sigma_n = 0$. Therefore columns of U beyond k^{th} column and rows of V beyond k^{th} row do not contribute to A and are usually omitted, leaving U an $m \times k$ matrix, Σ and $k \times k$ diagonal matrix and V an $r \times n$ matrix. After that terms are represented by the row vectors of the $m \times k$ matrix 4

$$Q = U_k\Sigma_k \tag{4}$$

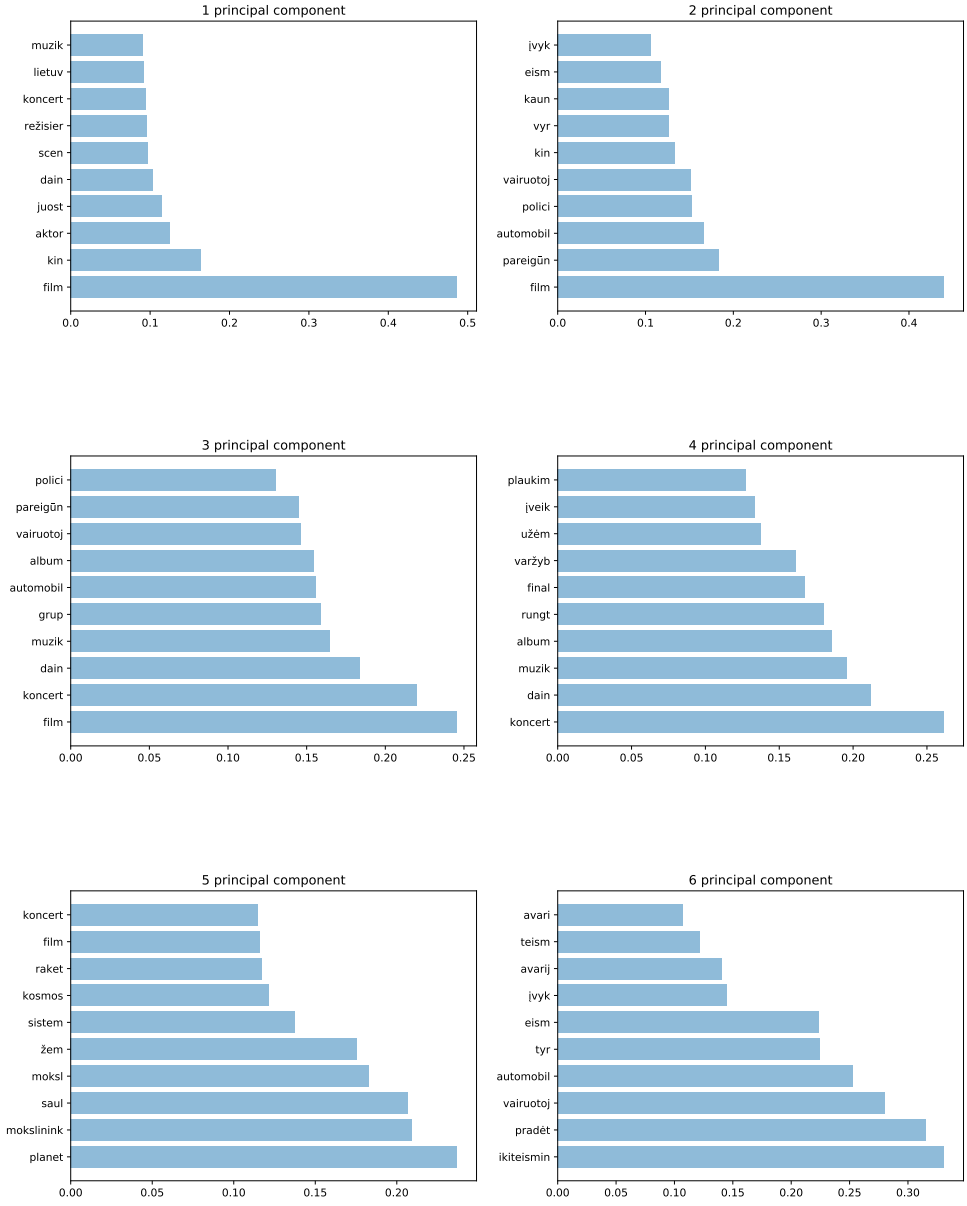
whereas the documents by the column vectors the $k \times n$ matrix 5

$$W = \Sigma_k V_k^T \tag{5}$$

Topics can represented as rows from matrix U^T $(k \times m)$, where rows are principal components, columns are tokens.

1. Extract first x rows (components).
2. Sort each row weights in descending order.
3. Extract first y highest weights and map them to the tokens we know.

Results figure 2 shows 6 components with 10 tokens for each component. From these results we can detect most important words and intuitively guess topic for each principal component. For example 4 principal component store information about sports and music whereas 5 principal component store information about criminals.



2 pav.: Main term components