Automatic Document Classification System for Unstructured Data

Mansur Mukimbekov
School of Computer Science and Engineering
UNIST
mansur@unist.ac.kr

Abstract

With the scientific and technological growth, the volume of unstructured data keeps growing at unprecedent pace. As result, finding a necessary document in this pile of unstructured data takes a considerable amount of time. Even though state-of-the-art machine learning methods, namely Large Language Models are able to efficiently handle this problem, they are still expensive for an average consumer. To solve this problem, this paper will review classical machine learning methods with different feature extractors, and evaluate their performance on the real datasets.

I. Introduction

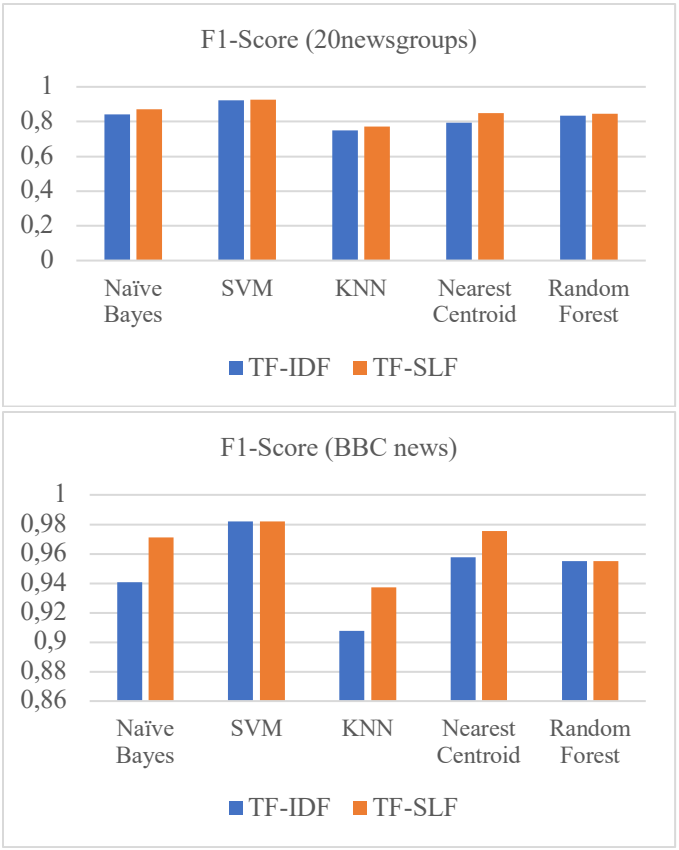
With the scientific and technological progress, the volume of unstructured data is growing at a dramatical pace. According to the research from IDC, 90% of most companies' data is unstructured [1]. In 2023 alone, organizations are predicted to generate over 73 000 exabytes of unstructured data. Searching for the necessary document in this keep growing pile of unmanaged information could take substantial amount of time. Therefore, search is one of the trouble spots in enterprise data management. Companies need automatic systems for searching and analyzing information in order to run their business smoothly.

Text Classification is a promising way of data indexing. By having predefined base of labeled data, it is possible to automatically tag the data using machine learning. However, despite the effectiveness of state-of-the-art solutions that rely on large language models (LLMs) [2], they are still expensive for the average consumers [3].

To solve the unstructured data problem, this paper will review classical machine learning methods that are available for average consumers, and propose a solution for the problem.

II. Literature Review

Over last two decades, there have been numerous research about document classifications systems. Popkov (2014) developed a text classification system using classic machine learning methods for Russian enterprise document classification [4]. However, due to the differences between English and Russian languages [5], the proposed method is not suitable for the English enterprise document classification.



Evaluation results for different feature extraction methods.

Fig. 1

III. Method

Text classification system consist of multiple parts: data preprocessing, feature extraction and model selection.

Data Preprocessing. Before the text could be used for classification, it must be preprocessed to be suitable for machine learning. Intuitively, stemming and stop-word pruning seems as a logical way of preparing the data. However, the research shows that by applying stemming and stop-word pruning the performance would drop [6]. Hence, it was decided to not modify the original text in any way.

Feature Extraction. For evaluation purposes, two feature extraction methods were considered: TF-IDF [4] and TF-SLF [4].

$$TF(\tau_i, d_j) = \frac{fr_{ij}}{\sum_i fr_{ij}} \; where \; 0 \le i \le |T|, 0 \le j \le |D| \; (1)$$

$$IDF(\tau_i, D) = \frac{|D|}{|(d_i \supset \tau_i)|} \; (2)$$

$$TFIDF_t = TF_t * IDF_t \; (3)$$

TF-IDF (3) is obtained by multiplying term-frequency (1) and inverse-document-frequency (2), where *T* and *D* are terms and documents respectively. TF-IDF reflects how important a word is to a document in a whole collection. On the other hand, TF-SLF builds on term being important if it's frequent in specific category, and unimportant if it's important for many categories at the same time.

$$NDF_{tc} = \frac{df_{tc}}{N_c} \; (4)$$

$$R_t = \sum_{c \in C} NDF_{tc} \; (5)$$

Calculating TF-SLF requires to find NDF_{tc} (4), which is a normalized frequency of occurrence of term *t* in category *c*, where N_c and df_{tc} stand for number of documents in category *c* and how many documents there have a term *t*. The NDF_{tc} score is local to the category, so to obtain a global estimate of R_t within the entire corpus, all NDF_{tc} are summed (5).

$$SLF_t = log\frac{|C|}{R_t} \; (6)$$

The logarithmic sum of frequencies is calculated as (6), where C represents categories, which is then multiplied by term-frequency to get TF-SLF (7).

$$TFSLF_t = TF_t * SLF_t \; (7)$$

Model Selection. To achieve a better understanding of classical machine learning methods capabilities on text classification task five methods were considered: Naïve Bayes, Support Vector Machine, KNN, Nearest Centroid, Random Forest.

IV. Application

Using Sci-Kit Learn machine learning library, the following classifications models were taken from there:

MultinomialNB - a classical implementation of Naïve Bayes for multi-classification.

LinearSVC - Support Vector Machine with linear kernel.

KNeighborsClassifier – KNN implementation with the number of neighbors set to 10.

NearestCentroid – Nearest Centroid classifier.

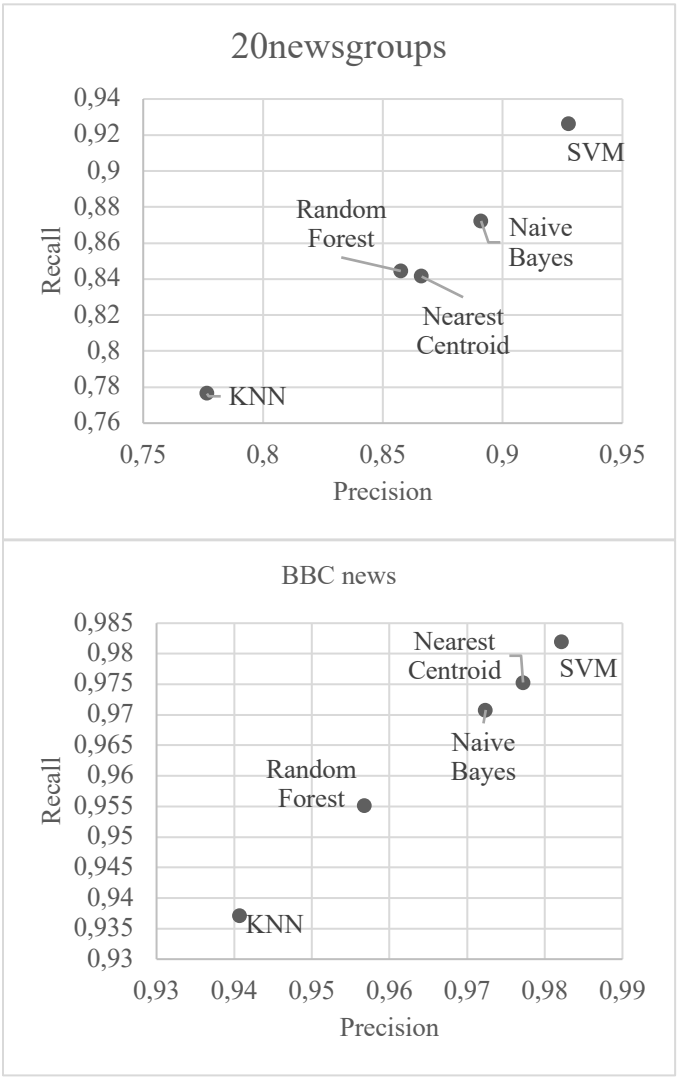
RandomForestClassifier – Random Forest classifier with 100 trees.

For TF-IDF feature extraction **TfidfVectorizer** was taken from Sci-Kit Learn. TF-SLF implementation was done by ourselves on Python.

Due to the near impossibility to find open labeled enterprise datasets, the models were applied on the 20newsgroup and BBC news datasets.

20newsgroup dataset – *Sci-Kit Learn 20 newsgroups dataset* comprises around 18000 newsgroups posts on 20 topics [7].

BBC news dataset - 2225 news documents in five different categories [8].



Precision and Recall for different methods with TF-SLF feature extraction.

Fig. 2

V. Results and Discussion

The models were evaluated on the following metrics: precision, recall, and F1 score.

Feature Extraction Methods Comparison. As we can see from Fig. 1, TF-SLF proved to be better than TF-IDF. This proved to be especially true for small-sized BBC dataset. In hindsight, the hypothesis of term being equally important for the whole document corpus is not appropriate when the size of dataset is only two thousand and the categories are thematically close. So, it seems logical for the metric that included importance of a term in specified category, TF-SLF, to be better than TF-IDF.

Classification Model Comparison. Looking at the Fig. 2, it is evident that SVM has the best performance in terms of recall and precision. Speaking in more details, SVM appears to have achieved more than 90% in both metrics. On the other hand, KNN has the lowest results in both dataset evaluations. Interestingly, on the larger dataset (20 newsgroup), Naïve Bayes has the second-best performance. However, when evaluating on smaller dataset, Nearest Centroid is the second-best model.

Are classical machine learning methods is a good alternative to LLMs? Considering the cost of LLM run, classical machine learning methods seems to be cheaper for both training and inference. Even LLMs are capable of outperforming them, considering the expense, over 90% Recall and Precision (SVM with TF-SLF) seems to better in terms of performance/cost.

VI. Conclusion

Even though LLMs have proven to have superior results, it seems that classical machine learning methods still able to provide adequate results for document classification task. Coupled with the right feature extraction methods for the right model, SVM with TF-SLF feature extraction, it can produce over 90% precision and recall. As a future work, it still leaves better feature extraction methods for research.

References

[1] IDC White Paper, "Untapped Value: What Every Executive Needs to Know About Unstructured Data," Doc. US51128223, August 2023

[2] Adhikari, A., Ram, A., Tang, R., & Lin, J. (2019, April 17). DocBERT: BERT for document classification. *arXiv.org*. https://arxiv.org/abs/1904.08398

[3] Smith, C. S. (2024, January 1). What large models cost you – There is no free AI lunch. *Forbes*. https://www.forbes.com/sites/craigsmith/2023/09/08/what-large-models-cost-you--there-is-no-free-ai-lunch/

[4] Popkov, M. I. (2014). Automatic classification system for enterprise documents. *International Journal of Open Information Technologies*, 2 (7), 11-18.

[5] Kumar, P. (2020). Comparative Grammatical Analysis of English and Russian Language. *ResearchGate*. https://www.researchgate.net/publication/339089294_Comparative_Grammtical_Analysis_of_English_and_Russian_Language

[6] Riloff, E. (1995). Little words can make a big difference for text classification. *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. https://doi.org/10.1145/215206.215349

[7] The 20 newsgroups text dataset — scikit-learn 0.19.2 documentation. (2017). *Scikit-Learn.org*. https://scikit-learn.org/0.19/datasets/twenty_newsgroups.html

[8] BBC Full Text Document Classification. (2024). *Kaggle.com*. https://www.kaggle.com/datasets/shivamkushwaha/bbc-full-text-document-classification