**EURASIAN JOURNAL OF MATHEMATICAL AND COMPUTER APPLICATIONS**
ISSN 2306–6172
Volume 5, Issue 3 (2017 ) 4 – 14

# BUILDING A MODEL TO PREDICT CLASSIFIER ACCURACY

**Aubakirov S.S., Trigo P., Ahmed-zaki D. Zh.**

**Abstract** In this paper, we propose an optimization workflow to predict classifiers accuracy based on the exploration of the space composed of different data features and the configurations of the classification algorithms. The overall process is described considering the text classification problem. We take three main features that affect text classification and therefore the accuracy of classifiers. The first feature considers the words that comprise the input text; here we use the N-gram concept with different N values. The second feature considers the adoption of textual pre-processing steps such as the stop-word filtering and stemming techniques. The third feature considers the classification algorithms hyperparameters. In this paper, we take the well-known classifiers K-Nearest Neighbors (KNN) and Naive Bayes (NB) where K (from KNN) and a-priori probabilities (from NB) are hyperparameters that influence accuracy. As a result, we explore the feature space (correlation among textual and classifier aspects) and we present an approximation model that is able to predict classifiers accuracy.

**Key words:** text classification, learning algorithms, genetic algorithm, distributed computing

**AMS Mathematics Subject Classification:** 68T50, 91F20

## 1 Introduction

Because of the rapid progress on computer-based communications and information dissemination, large amounts of data are daily generated and available in many domains. Data classification is the process of organizing data into categories for its most effective and efficient usage. Nowadays there is a great choice of classification algorithms. Each classification algorithm is tuned via hyperparameters that affect the learning process and the final accuracy of the prediction model. In the context of text classification, the input (text) is pre-processed by a set of operators and therefore each operator also influences the prediction accuracy.

Hence, in the context of the text classification problem, we can formulate a feature space composed of: a) classification algorithms, b) hyperparameters of the classification algorithms, and c) textual pre-processing operators. According to [1] the task of selecting the right features is nontrivial and checking all possible combinations is an NP-complete task. In this research, we propose to deal with this task using an optimization workflow that explores the feature-space with the goal of maximizing the classifier's accuracy. We outline the optimization workflow and throughout this paper, we describe each step of the workflow and show that we can approximate the maximization goal using a regression model.

Paper is structured in three sections. The next section describes the problem of text classification and the method to combine classifiers. Then we address the task of evaluating binary classification of texts and analyzing the relationship between classification

quality and classifier parameters. Next, we describe a genetic algorithm that combines all attributes and evaluates each possible solution. We propose a method of formal description of combination and its evaluation. We follow an evolutionary approach (via Genetic algorithm) to generate the data used to train a prediction model. The model goal is to predict the classifier's accuracy without going through the classifier's algorithmic computing.

Finally, we show that model's predictions are less time consuming than the process of training and evaluating each classifier by itself.

## 2 Related work

A large study of classification algorithms shows that not only accuracy of the algorithm depends on selected features and input data, but training time, scalability and interpretability of algorithm [2]. Another research [3] points out the challenges associated with automated text, such as a) choosing an appropriate data structure to represent the documents, b) choosing an appropriate objective function to optimize in order to avoid overfitting, c) obtain good generalization, and c) dealing with algorithmic issues arising because of the high formal dimensionality of the data. This last challenge can be addressed [3] via a prior selection of a subset of the features available for describing the data before. Such selection occurs before applying a learning algorithm and setting its operational parameters. A large number of studies on feature selection have focused on text domains both for binary and multi-class problems. This fails to investigate the best possible accuracy obtainable for any single class [4].

Those studies deal with feature selection and provide an in-depth analysis of the problem of simultaneously selecting a learning algorithm and setting its hyperparameters In consequence of [5] work, researchers provide a tool that effectively identifies machine learning algorithms and hyperparameter settings appropriate to their applications. Proposed approaches still require high computational resources to evaluate each model.

Most feature selection studies are conducted in a non-automatic way or in semi-automatic way. This fails to explore all possible features, attributes and algorithms. We present a methodology to build effective automatic feature and algorithms selection tool.

## 3 The feature-space and optimization-goal

The first feature that affects classifier accuracy is an input data pre-processing. Pre-processing, i.e., applying methods for cleaning up and structuring the input text for further analysis, is a core component in practical text mining studies. The goal of data pre-processing (pp) is to clean and prepare the text for classification and the whole process is as pipeline with several stages, namely: pp1) text cleaning, pp2) white space removal, pp3) expanding abbreviation, pp4) stemming, pp5) stop words removal, pp6) negation handling, pp7) short words removal and pp8) feature selection. The stages from pp1 to pp7 are usually referred to as transformations (T), while pp8 (last step,

that applies some functions to select the required patterns) is referred to as filtering (F) [6].

In this work, we do not use all transformations as a part of our feature space. We consider that all input text is already pipelined through pp1, pp2, pp3 and pp5 stages. We have a set T consisting of stages pp4 and pp7. On stage pp4 we decide whether input text will be processed with stemmer (S) or not (NS). On stage pp7 we decide whether input text will be processed with word length filter (LF) or not (NLF). Length filter is dropping out the word that is shorter than four symbols. There are combinations of pp7 and pp4 features, for example, if we want to apply both stemming and short word filtering. As a result, we have a set T of 4 different features:

$$T = \{S + LF, S + NLF, SN + LF, SN + NLF\}$$

The second feature is pp8 - an input data text feature selection by extracting Ngrams. For our research, we choose 1-gram, 2-gram, 3-gram, 4-gram, 5-gram and altogether from 1-5 gram extraction. As a result, we have a set N of 6 different features:

$$N = \{1Gram, 2Gram, 3Gram, 4Gram, 5Gram, 1 - 5Gram\}$$

The third feature is classifier (C). Each classifier has hyperparameters that affect the classification accuracy. In this work, we consider the well-known classifiers K-Nearest Neighbors (KNN) and Naive Bayes (NB) where K (from KNN) and $\Gamma$ -priori probabilities (from NB) are hyperparameters that influence accuracy. Our experiments have shown that KNN classifiers accuracy dramatically low for $K > 200$, that is why we decide to bound K from 1 to 200. We have a set of hyperparameters features C = (0-200), where values $(1 - 200)$ mapped to K in KNN classifier and value 0 mapped to NB classifier. We have a set C of 201 different features:

$$C = [0, 1, 2...199, 200]$$

The next feature is text fields (F). As there are two fields available, title and body, we have a set, F, of three different features:

$$F = \{body, title, body + title\}$$

Totally, we have a combination of 24 pre-processing features (4 transformations and 6 filtering), 201 classifiers and 3 fields - it is 14472 different combinations. By combining all features, we assemble classifier. Finally, our feature space (FS) can be expressed using formula as follows:

$$FS = \{c_i, n_j t_k f_l\},$$

$i \in [0 - 200]$, $c_i \in C - c_i$ corresponds to classifier feature,
$j \in [0 - 6]$, $n_j \in N - n_j$ corresponds to Ngram filtering feature,
$k \in [0 - 3]$, $t_k \in T - t_k$ corresponds to transformation feature,
$l \in [0 - 3]$, $f_l \in F - f_l$ corresponds to text field feature

We propose tuple notation to describe each assembled classifier, it is shown in figure 1.

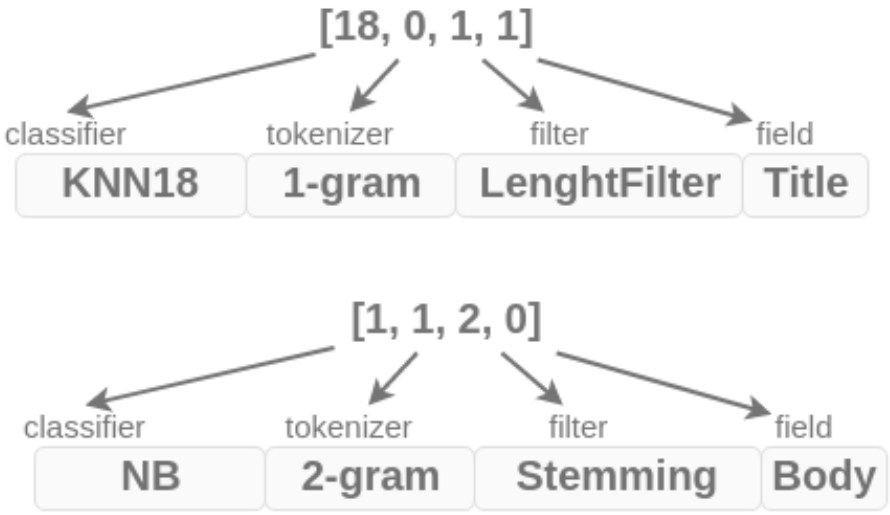


Figure 1: Tuple notation describes each classifier with all features.

In the next chapter, we describe the classifier evaluation metrics and analyze the relations between N-grams, indexes and accuracy of classification.

## 4 The classifier evaluation method

The evaluation resorts to the k-fold cross validation method. According to research [8] and [7] we choose Kappa and MCC coefficient measures to evaluate binary classification learning. Both, Kappa and MCC are well balanced, takes into account true and false positives and negative. A coefficient of 1 represents a perfect prediction, 0 no better than random prediction and $-1$ indicates total disagreement between prediction and observation.

We measure each classifier using k-fold cross validation method. The best results show classifiers that are using unigrams, only body field, Stemmer and with short words filtering.

Results are shown in figure 2. The figure shows the best results for both KNN and NB in terms of different value N for extracted Ngrams. NB classifier best results: $MCC = 0.8$, $Kappa = 0.79$, F-measure often shows values close to 1, this is because F-measure calculation ignores FN errors. That is why we do not use it as an evaluation measure because our main goal is to minimize both FP and FN.

For KNN algorithm, with $K = 3$, best result is $MCC = 0.68$, $Kappa = 0.67$. Furthermore, we noticed that increasing number of neighborhoods leads to decreasing accuracy of the classifier.

Our experiment shows that single classifier algorithm does not produce accuracy higher than 0.8 of Kappa coefficient and higher than 0.79 of MCC coefficient. To improve accuracy we implement voting classifier algorithm. In the next chapter, we describe this approach.

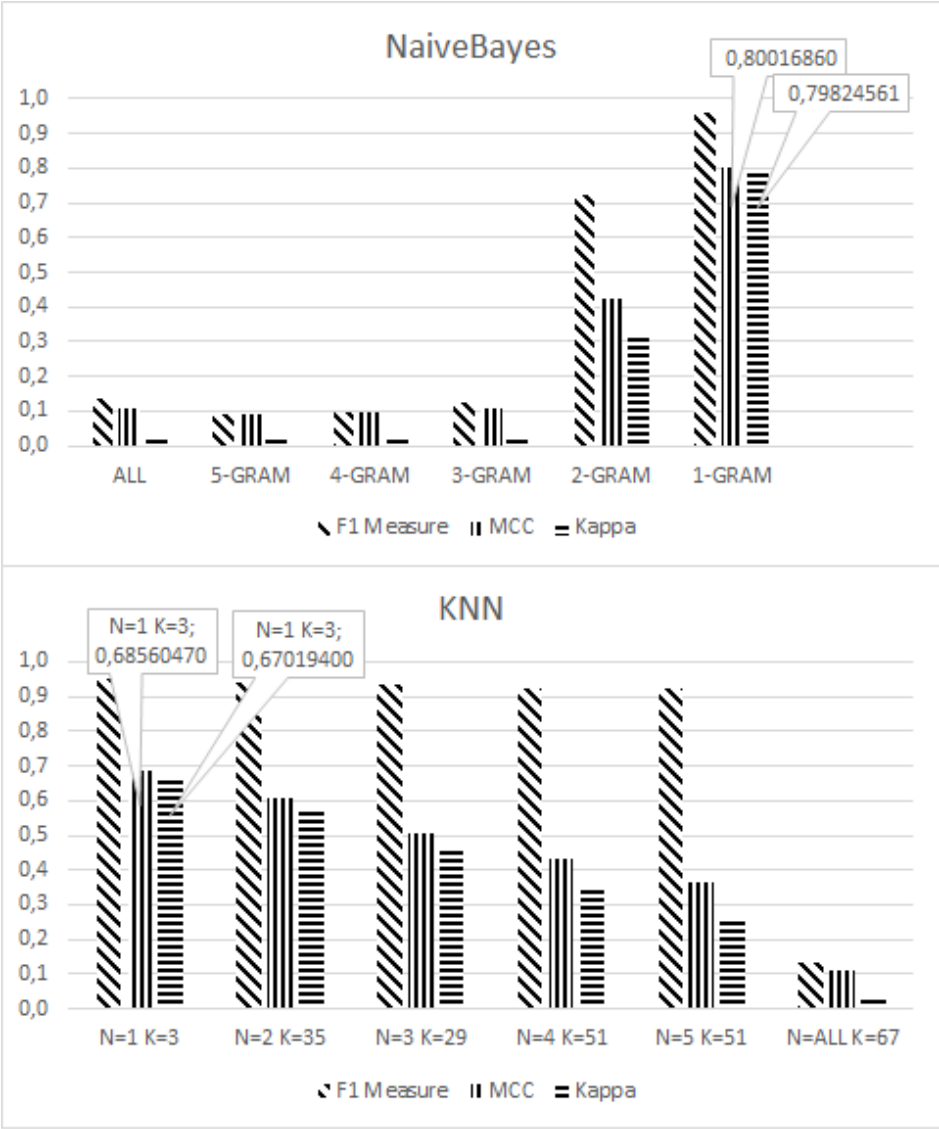


Figure 2: Correlation between the value N for NGrams extracted and classifier accuracy.

## 5 Classifier combination method

We built a voting classifier based on work [9] to combine classifiers. To make a decision we need a simple majority of voters. Suppose we have five voters, then combination of all possible classifiers would be the combination of given 14472 set of classifiers of 5 elements. Which is $5 * 10^{18}$ possible combinations. We use a classical genetic algorithm to deal with this complexity. The chromosome will consist of 5 genes, where the gene is one of the combinations of the classifier, analyzer and field. By assigning each characteristic to a number, you can define a set of genes. Thus, the gene is an array of 4 digits, where each of the digits is responsible for the characteristics of the classifier. An example of chromosome shown in Figure 3.

Optimization schema is shown in figure 4. As a fitness function, we will use the value of the Kappa coefficient, based on confusion matrix after the cross-validation.

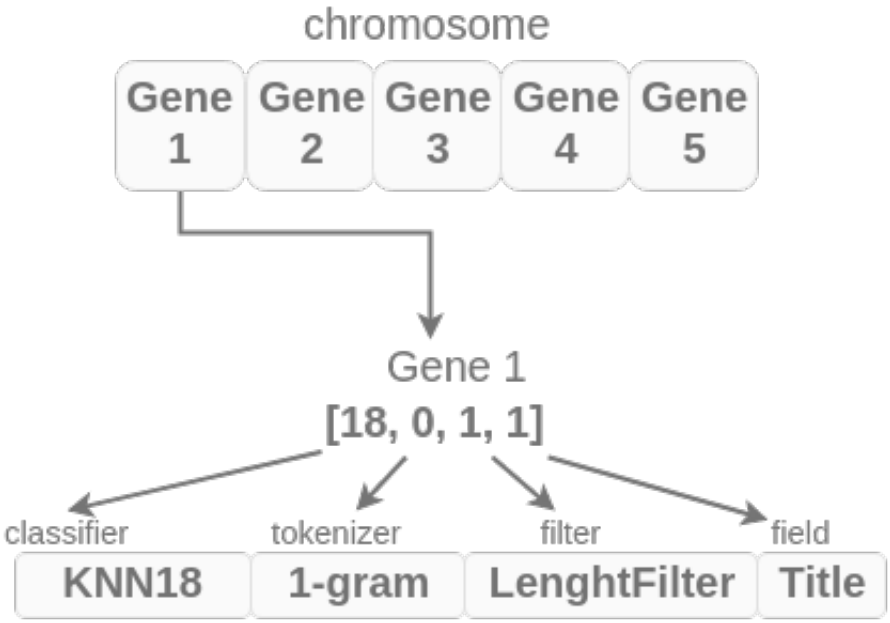


Figure 3: An example of chromosome for voting classifier.

The more it is, the better. The stopping limit for the genetic algorithm is set to 0.95. Selection will occur using the roulette selection method. This method increases the probability of selecting an individual with a high fitness function. The higher the Kappa coefficient, the more likely it is to get into the next generation.

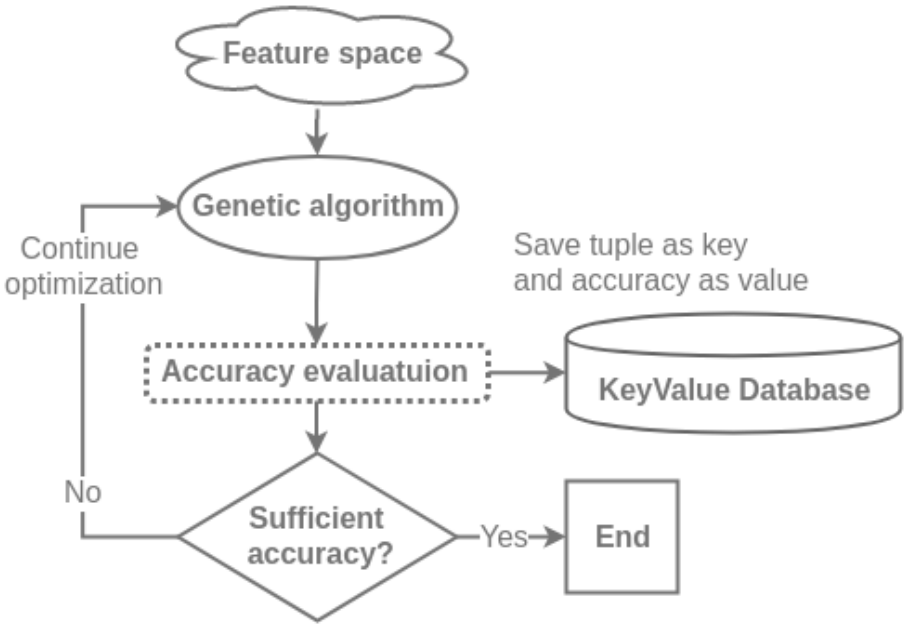


Figure 4: Optimization schema workflow.

We use simple One-Point Crossover as a crossover function. The point at which the crossing will occur selected randomly. The mutation of each chromosome triggered with a 35% chance. This means that after each cycle of the algorithm, only 35% of the population can mutate, and only 2 of the 5 genes will mutate.

This simple genetic algorithm produces well-stratified data to train prediction model. Figure 4 shows that we save each tuple as a key and accuracy as the value. We execute algorithm until it generates enough results to approximate accuracy function.

Proposed voting classifier description is a vector in an N-dimensional space, where N is a number of features of the classifier. Nearly any regression model can be used to approximate relationship between features and accuracy. As proof of concept, we built multilayer perceptron (MLP) with one hidden layer in order to predict voting classifier accuracy without computation, training and testing classifier. The prediction-learning schema is in figure 5.

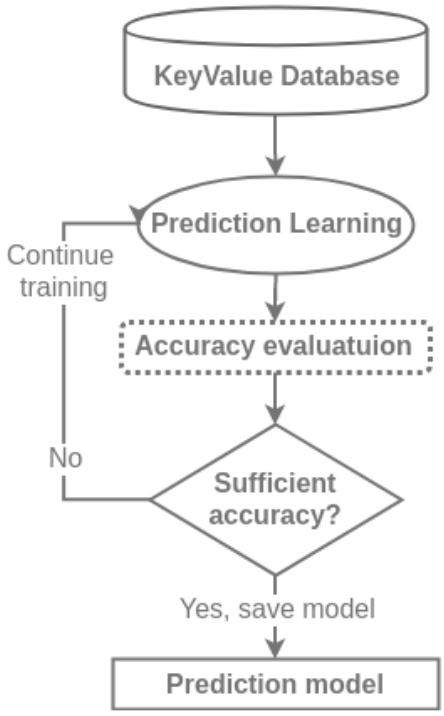


Figure 5: Process of training regression model.

Multilayer perceptron was trained using data that genetic algorithm produced. In order to archive maximum performance, we implement a genetic algorithm in a distributed computing manner using Java Message Passing Interface (MPJ-Express).

## 6 The Results and Discussion

We measure each classifier using k-fold cross validation method. Our experiments show that 87% of the learned models exhibit a Kappa and an MCC less than 0.5, which, according to [7] can be considered as "poor" models. The best results show classifiers

that are using unigram tokenizer, only body field, Snowball stemmer and with short tokens filtering. Classifiers based on indexes with Ngrams with $N > 1$ shows low accuracy: $MCC < 0.5$, $Kappa < 0.5$.

The proposed multilayer perceptron predicts an accuracy of the voting classifier with the maximum Kappa coefficient of 0.85. The accuracy of MLP is not perfect, but according to [7] almost perfect agreement. In addition, the goal of MLP was to predict accuracy and avoid calculation of weak voting classifier. We interpret these results as encouraging evidence for the usefulness of MLP for deciding usefulness of voting classifier.

The speedup ratio and parallel efficiency plots are shown in figure 6. Speedup and efficiency provide an estimate for how well a code sped up if it was parallelized. The yellow plot shows ideal speedup and efficiency, the ratio of one hundred percent parallelized code.

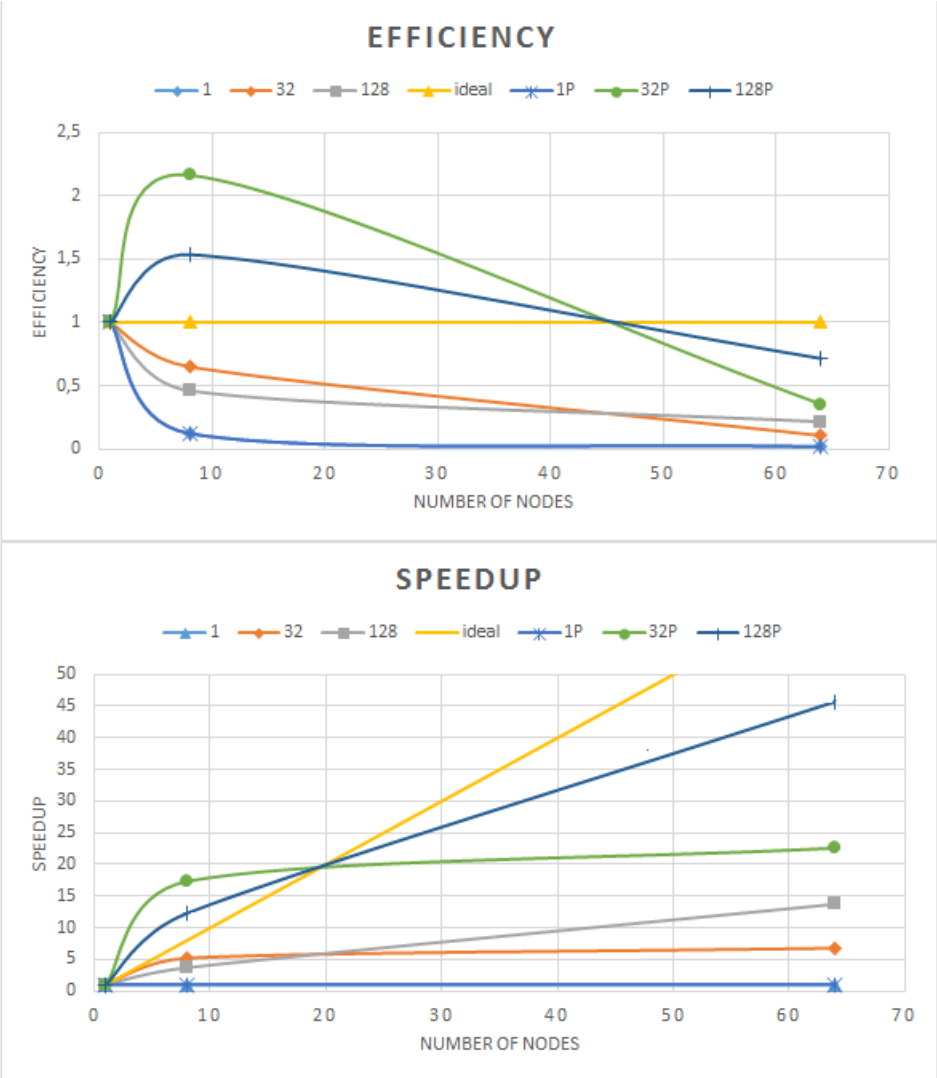


Figure 6: Speedup ratio and parallel efficiency. Plots with prefix "P" shows computation using prediction model.

The green (32P) and blue (128P) plot stand for 32 and 128 classifiers evaluation respectively. The prefix "P" means that optimization process uses prediction model in order to drop out classifiers with insufficient accuracy. This helps to identify weak classifier before implicit training, testing and evaluating. That is why plots sometimes higher than an ideal line.

## 7 Conclusion

It is difficult nowadays to decide which classification algorithm to use and how to preprocess text input data. We design a workflow of algorithms that can significantly reduce the amount of time to find out correct attributes of the exact problem. Figure 7 shows proposed workflow of algorithms.

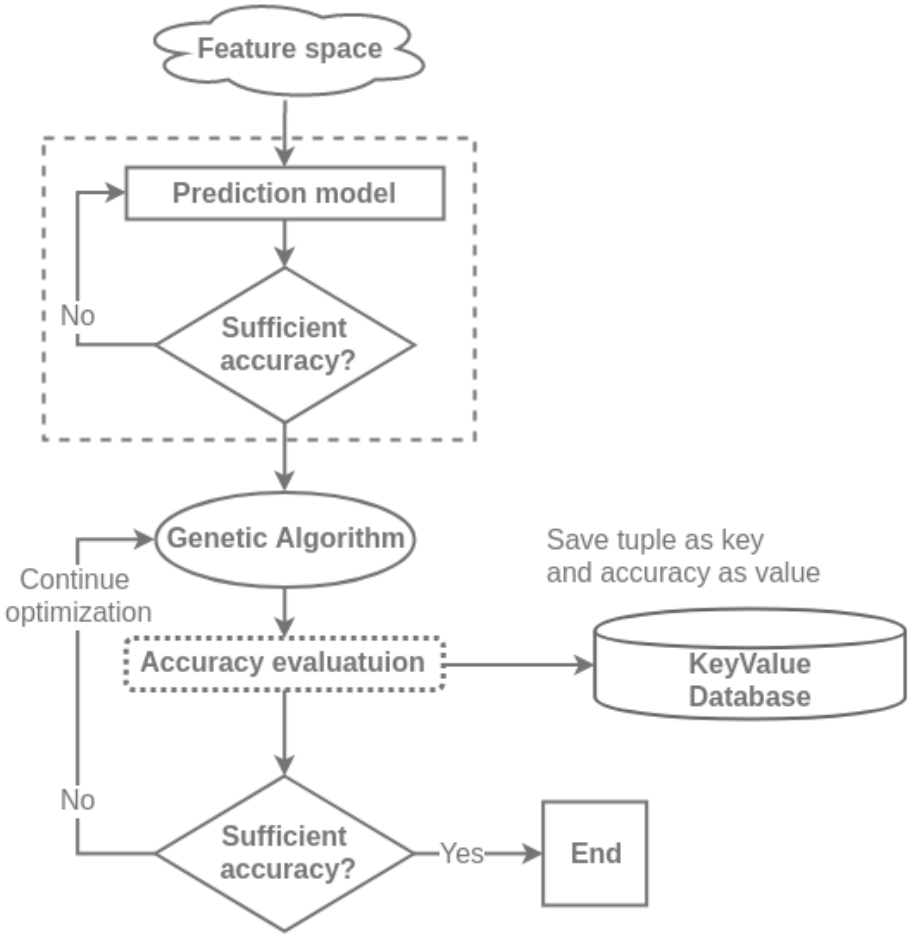


Figure 7: Final optimization workflow. The dotted line highlights the prediction model.

Workflow is very effective and has accuracy up to 0.8 value of Kappa coefficient. Furthermore, the accuracy of the whole workflow can be improved by selecting better approximation model, for example, better MLP architecture.

We intended to improve the whole workflow in a much larger study that will allow

us to thoroughly test MLP evaluation. In this study, we also plan to investigate the impact of the accuracy of extracted text summary to final evaluation.

## Acknowledgement

This work was partially supported by the grant of the Committee of Science of the Ministry of Education and Science of the Republic of Kazakhstan (project 3350/GF4 MON RK).

# References

[1] Garey M.R., Johnson D.S., *The Theory of NP-Completeness*, Computers and Intractability; A Guide to the Theory of NP-Completeness, 1990, W. H. Freeman & Co. New York, NY.

[2] Lim TS., Loh WY., Shih YS., *A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms*, Machine Learning, Vol. 40, Issue 1 (2000), 203-228.

[3] Dasgupta A., Drineas P., Harb B., Josifovski V., Mahoney M.W., *Feature Selection Methods for Text Classification*, KDD '07 Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining 2007, 230-239. San Jose, California, USA.

[4] Forman G., *An Extensive Empirical Study of Feature Selection Metrics for Text Classification*, Journal of Machine Learning Research, Issue 3 (2003), 1289-1305.

[5] Thornton C., Hutter F., Hoos H. H., Leyton-Brown K., *Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms*, KDD '13 Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013, 847-855.

[6] Haddi E., Liu X., Shi Y., *The Role of Text Pre-processing in Sentiment Analysis*, Procedia Computer Science. Vol. 17 (2013), 26-32.

[7] Landis J.R., Koch G.G., *The Measurement of Observer Agreement for Categorical Data*, International Biometric Society, Vol. 33 (1977). 1, 159-174.

[8] Powers, D.M.W. , *Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation*, Journal of Machine Learning Technologies, Vol. 2 (1) 2011, 37-63

[9] Bauer E., Kohavi R., *An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants*, Machine Learning, Volume 36, Issue 1 (1999), pp 105вЂ“139.

S.S. Aubakirov,
al-Farabi Kazakh National university,
Almaty, 050000, Kazakhstan,
Email: aubakirov.sanzhar@gmail.com,
P. Trigo,
Instituto Superior de Engenharia de Lisboa, Biosystems and Integrative Sciences Institute Agent and Systems Modeling,
Lisbon, 1000-260, Portugal,
Email: ptrigo@deetc.isel.ipl.pt,

D. Zh.. Ahmed-Zaki,
al-Farabi Kazakh National university,
Almaty, 050000, Kazakhstan,
Email: darhan.ahmed-zaki@kaznu.kz,

Received 29.06.2017, Accepted 14.08.2017