Greek Article Categorization Using Machine Learning Techniques

Margaritis Georgios,Pinitas Kosmas

Abstract—Automated text categorization using machine learning techniques has been proven to be a very efficient way to analyse and classify a vast collection of documents into different categories. In general, text categorization falls into the field of Natural Language Processing and plays a significant role in information extraction and text summarization. The purpose of this paper is to compare several approaches to Greek Newspaper Article Categorization using different Machine Learning Algorithms and Text Preprocessing Techniques.

Index Terms—greek article categorization,feature selection,classification algorithms

1 Introduction

AUTOMATED text categorization is a technique ubiquitously used today due to the vast amounts of digital information available. Specifically, article categorization incorporates techinques from many different fields, such as Data Mining, Linguistics, Information Retrieval and Natural Language Understanding and can also provide very consistent results nowadays given the very large numbers of text documents available [5].

Generally, document categorization is the process of assigning multiple classes $\{c_1, c_2, ..., c_n\}$ to a certain document $d_i \in \mathcal{D}$ where $\mathcal{D}$ represents a collection of documents. However, in this paper we will examine a simpler form of text categorization where each document belongs to only one class $c_j \in C$, following a one to one relationship [1]. Similarly to every supervised machine learning algorithm, our goal is to is to find a function $f : \mathcal{D} \rightarrow C$ which maps every document $d_i \in \mathcal{D}$ to exactly one class $c_j \in C$.

In our implementation of Greek text categorization, our task was to classify Greek newspaper articles into 6 distinct categories C={Art, Economy, Greece, Politics, Sports, World}. In order to attain this goal, we first collected newspaper articles from enet newspaper article archive. After that, following the resolution of several encoding issues regarding the Greek UTF-8 glyphs and the stripping of redundant characters that convey no information, we adopted the standard procedure that is generally used for text classification problems. This process, as seen on figure 1 involves, Punctuation Deletion, Accent Stripping, Text Tokenization, Neutral Word Removal, Stemming, Vector Representation of documents as a term document matrix, Feature Transformation, Feature Selection.

As in every other machine learning task, the main problem of text categorization is to extract features from data in order to create a numeric representation. Specifically, in text categorization the main task is to extract meaningful numerical components that will be used to describe our documents. However, due to the very large number of distinct words in our documents (which may exceed 60000), it is usually considered as a good practice to implement certain dimensionality reduction techniques in order to increase the efficiency of Machine Learning Models.

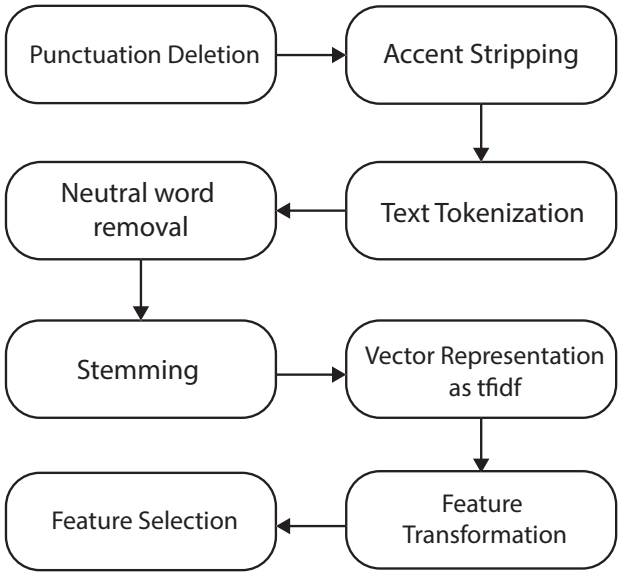

Fig. 1. Preprocessing steps

Therefore, various methods can be employed for dimensionality reduction. The first approach is to select a subset of the original features by using several statistical criteria, such as the variance or the occurrence frequency of each feature among the different classes. Another approach is to create new, more descriptive features from the existing ones by applying a series of feature transformations. In this paper we implemented a combination of the aforementioned methods in order to reduce the dimensionality of our data.

After concluding the stages of preprocessing, our next step was to implement various Classification Algorithms for different sets of parameters. We then extracted evaluation metrics in order to assess the suitability of each algorithm for the given task. Finally, in order to assess the generalization capabilities of our algorithms, we performed kfold cross validation.

2 STATE-OF-THE-ART

A similar approach for text categorization was used by [3]. In that paper, the research team performed an empirical comparison of several classifiers, like KNN and SVM for the problem of text categorization. More specifically, they used tfidf text representations and Latent Semantic Analysis in order to evaluate the accuracies of each model. Their results showed that both SVM and KNN models perform very well in the specific task.

Another more interesting approach towards text classification was described in [4]. In that paper, the research team didn't just use the classic models for text categorization (like SVM, RandomForest,Naive Bayes and ANN), but they also implemented several hybrid approaches. Those approaches included ANNs initialized using Decision Trees, Probabilistic Neural Networks (PNNs) and Bahes Formula for Classification which combines Naive Bayes and SVM classifiers. Additionaly, certain other approaches were also described in the paper, like Neural Text Categorizer Acronyms, Self Organizing Maps and Soft-Supervised Learning.

3 TECHNIQUES USED

3.1 Text Preparation

Before proceeding to feature creation, it was necessary to implement a series of text manipulations in order to prepare for the next steps of the preprocessing phase.

First of all, we stripped our text of punctuation characters that convey no useful information, such as dots, commas, exclamation marks e.t.c. Secondly, due to the fact that Greek language contains certain supplementary accent characters, which may negatively affect the stage of stemming, we had to replace letters having accent characters with bear letters (accent stripping). After having removed redundant characters and converted our text into uppercase, we performed text tokenization by separating our text into words using space delimiters.

However, not all words in a text provide useful information, as there are words like articles, pronouns, prepositions and linking words that just contribute to the cohesion of the text. As a result, the distribution of those words among the texts of different classes are approximately the same and the existence of those words cannot help us identify the class that the article belongs to. That's why we removed those words from the vector representations of our texts.

In text categorization, due to the fact that the main meaning of the words are encapsulated into their roots, it is a common practice to perform stemming or lemmatization in order to extract the root of each word. So, that's why we made use of a Greek Stemmer in order to remove word suffixes. However, we modified the Greek Stemmer so that it would not stem certain words, like acronyms in order not to alter their meaning.

3.2 Articles as TF-IDF vectors

As we wave already mentioned, a document consists of a set of words and thus, in order to be able to implement our classification models, we first have to transform the documents from the word space into a vector space. The first step to realize this transformation was to create a dictionary in order to map words into their respective indices. Due to the fact that in our task the order of words does not play a significant role in determining the category of the document, we employed a model called tfidf that falls into the category of Bag-Of-Words-Models.

According to tfidf model, we can assign weights to different words by multiplying TF $\cdot$ IDF, where tf is the frequency of a term in a particular document and idf is the inverse document frequency of the term [2]. Mathematically, if we denote N_{ij} as the number of occurrences of term t_i in document d_j and W_j the total number of terms in document d_j, then $tf(t_i, d_j) = \frac{N_{ij}}{W_j}$. Similarly, if T_i is the number of texts that contain term t_i, then $idf(t_i) = \log \frac{|\mathcal{D}|}{T_i}$.

Consequently, this model allows us to represent each article d_j as a vector $x_j \in \mathbb{R}^{|V|}$ where $|V|$ is the number of words in our dictionary. Hence, we can represent our collection of articles as a matrix $X \in \mathbb{R}^{|\mathcal{D}| \times |\mathcal{V}|}$ which is referred to as a term-document matrix. This model has been proven to be very robust and difficult to beat, even by more sophisticated models and theories [2].

3.3 Advanced Features

Instead of just using single words as features, it is generally considered a good practice to use contiguous sequences of n words, also known as n-grams, which can help increase prediction accuracy. In this paper, we used n-grams for $n = 2$, which are called bigrams. Generally, in a collection of texts a huge number of bigrams can be found. Hence, in order to select the most descriptive bigrams, we utilized the metric of Pointwise Mutual Information (PMI). More specifically, for 2 terms t_1 and t_2, the PMI of the 2 terms is given by the formula $I(t_1, t_2) = \log_2 \frac{P(t_1,t_2)}{P(t_1)P(t_2)}$.

Another commonly used practice for feature creation is to group certain terms that would otherwise be considered useless, into more general groups. For instance, in our case, we grouped all numerical data into a single feature because we are mostly interested in the existence of a number in the text and not interested in the numerical value of this number. Also, we followed the same strategy for country names by grouping them into a single feature.

3.4 Feature Transformation

In many cases, we can improve the performance of our algorithms by performing a $1-1$ transformation over the elements of a term document matrix, which may lead to a better representation of our dataset. There are several such transformations, but for the scope of our paper we examined the following:

TABLE 1
TFIDF transformations

Transformation	Forumula
Binary	1 if $x_{ij} > 0$ else 0
Logarithmic	$\log(1 + x_{ij})$
Entropy	$\left(1 + \frac{\sum_{i=1}^{n} p(x_{ij}) \log(p(x_{ij}))}{\log n}\right) \cdot log(1 + x_{ij})$

3.5 Feature Selection

Due to the fact that fact that the distinct words in our dataset was very large (exceeding 60000), it was necessary to perform dimensionality reduction to our data. Besides the standard dimensionality reduction techniques of LDA and PCA, we also applied dimensionality reduction according to the statistical characteristics of our features among the different classes.

In particular, we constructed a matrix $Z \in \mathbb{R}^{|C|\times|V|}$, where each element z_{ij} of that matrix refers to the frequency of occurrence of term t_j in class c_i. After that, for each column of matrix Z, we calculated the variance among the different classes and we selected the features with the highest variance value. This way, we kept the features that best differentiate one class from another. It should be noted that for the best possible results, this method was used in conjunction with LDA or PCA.

3.6 Classification Models

In order to classify our articles into the different categories, we used a wide variety of models, such as SVM, RandomForest, GMM, NB, KNN, ANN, CNN. Out of those, we implemented ourselves KNN, NB and GMM. Apart from the above models, we also implemented a classifier that categorizes each sample to the class of which the mean value is closest to the sample, which we will refer to as MEAN Classifier.

It should be noted that for the implementation of GMM, we trained a separate Gaussian Mixture Model for each class and we calculated the logarithmic probability of an article belonging to a certain class by summing the individual log-likelihood probabilities from every Gaussian component.

Also, for algorithms like KNN and MEAN Classifier, we implemented several different distance metrics, such as euclidean distance, mahalanobis distance and cosine similarity measure. These distance metrics are shown on the table below.

TABLE 2
Distance Metrics

Distance	Forumula
Euclidean	$d(x,y) = \|\|x-y\|\|_2$
Mahalanobis	$d(x,y) = \sqrt{(x-y)^T S^{-1}(x-y)}$
Cosine	$d(x,y) = cos^{-1}\left(\frac{xy^T}{\|\|x\|\|_2\|\|y\|\|_2}\right)$

4 Experiments

4.1 Individual Models

In our experiment, we measured the accuracy of our models (GMM, KNN, MEAN, RandomForest, SVM) for different sets of parameters in order to determine the best parameters for each model. The respective results are shown in figures 2-6.

4.2 Overall Comparison

After determining the best set of parameters for every model, we compared the models with one another as shown in figures 7-10. The comparisons were conducted for different reduction methods (LDA-PCA) and different tfidf transformations (binary, entropy and log).

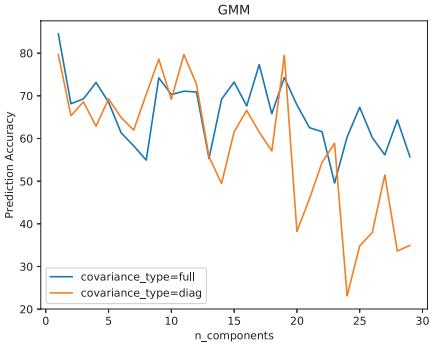

Fig. 2. GMM

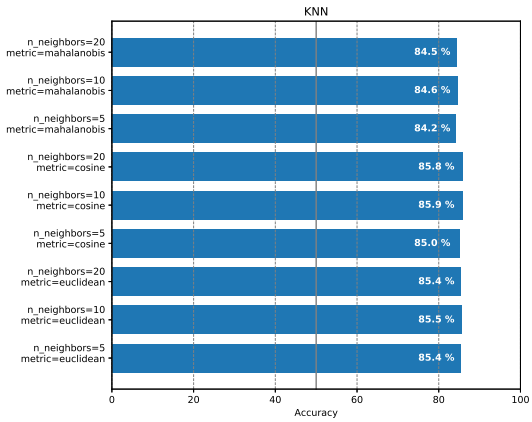

Fig. 3. KNN

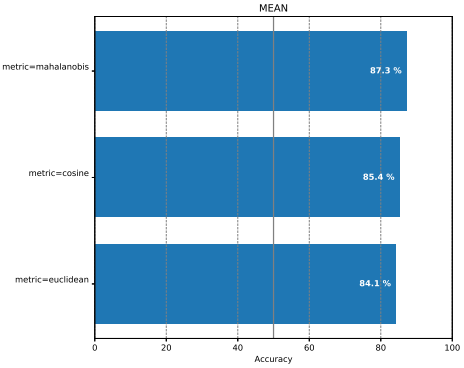

Fig. 4. MEAN

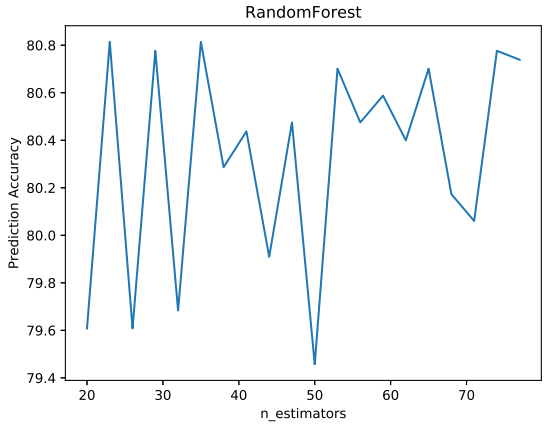

Fig. 5. Random Forest

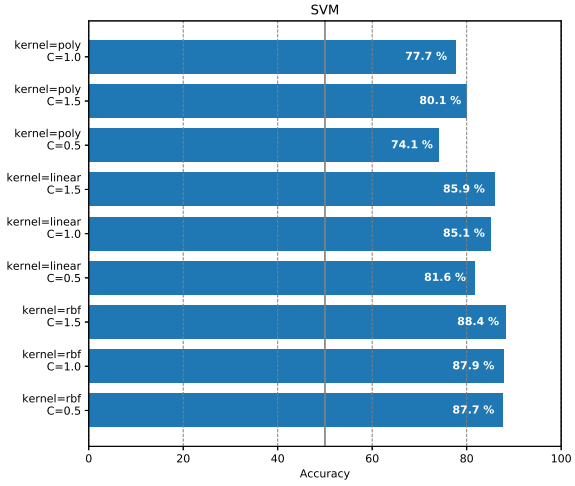

Fig. 6. SVM

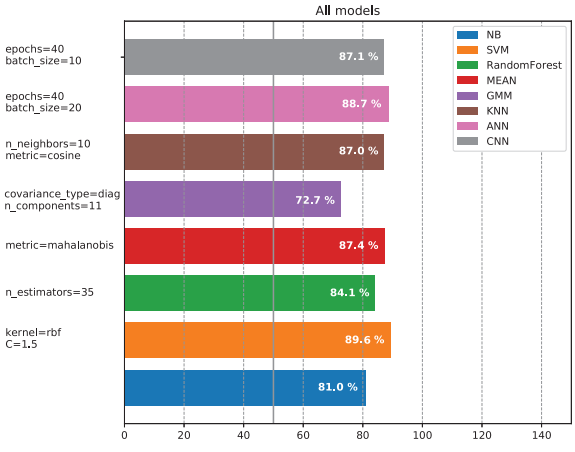

Fig. 7. PCA with entropy transformation (70 components)

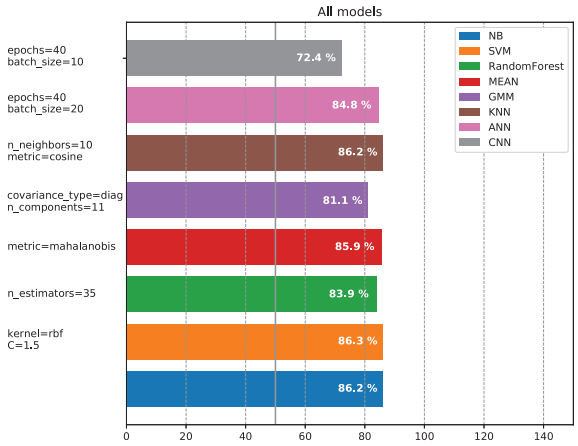

Fig. 8. LDA with entropy transformation

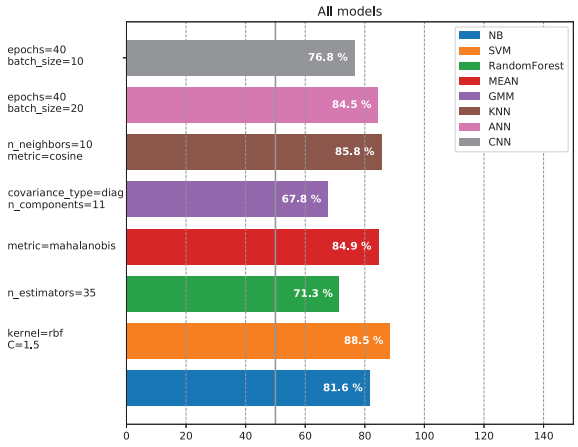

Fig. 9. PCA with binary transformation (70 components)

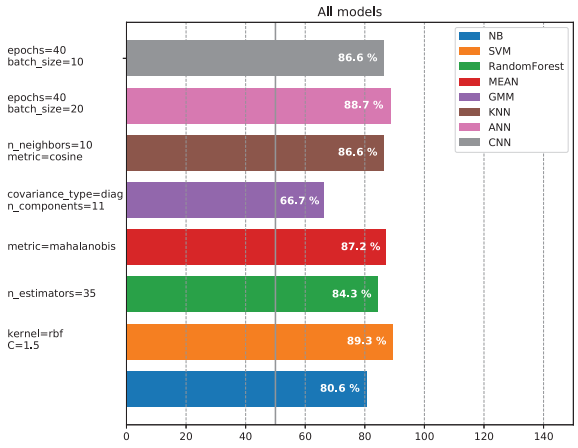

Fig. 10. PCA with log transformation (70 components)

5 Conclusion

5.1 Individual Models

5.1.1 GMM

For the GMM classifier, as seen on figure 2 we can observe that the best accuracy is achieved for 1 gaussian component. However, for a single gaussian component, the GMM model is effectively degenerated into a Gaussian NB classifier. Additionaly, the second best performance is achieved for 11 guassian components and a diagonal covariance matrix. However, we can observe that for different numbers of gaussian components the accuracy of GMM is erratic and not deterministic, rendering GMM a rather unsuitable model for the task. Nevertheless, in order to improve GMM accuracy in cases when we have limited data, it is generally preferable to use a diagonal covariance matrix because it requires a smaller amount of parameters for training.

5.1.2 KNN

For the KNN classifier,as shown on figure 3, it is evident that he best accuracy is observed for 10 neighbors and a cosine similarity distance metric. This is to be expected, as the cosine distance is considered a good distance metric for NLP problems.

5.1.3 MEAN

As shown on figure 4, the best accuracy for the Mean classifier is observed for mahalanobis distance. It should be noted that the accuracy of the MEAN classifier is very good, considering that it is a very simply classifier, taking into account only the distance from the mean of each class. Also, it should be stated that if we combine the MEAN classifier with LDA reduction, the resulting model is referred to as LDA classifier.

5.1.4 Random Forest

For the random forest classifier, as shown on figure 5, we can observe that there is a relatively large variance in accuracy ($\sim 1.5\%$) for different numbers of estimators. Hence, it is evident that this model is not very suitable for the particular task, due to the instability in accuracy and the relatively low (correct) prediction scores.

5.1.5 SVM

In most cases, the accuracy of the SVM classifier is relatively high, compared to other models (figure 6). However, the best accuracy of SVM is observed for penalty parameter $C = 1.5$ and rbf kernel (gaussian). The superiority of the gaussian kernel in this case stems from the fact that our data can be seperated more easily by a linear hyperplane when projected to a gaussian space [3].

5.1.6 ANN and CNN

Although we implemented rather simple architectures of ANNs and CNNs, with 2 hidden layers in the case of ANN and 1 Convolutional layer and 2 dense layers in the case of CNN, we observed that both ANN and CNN perform relatively well. Especially in the case of ANN, this is to be expected as ANNs are considered as solid-performing models for the task [4].

5.2 Overall

5.2.1 PCA vs LDA

By comparing figures 7 and 8, it can be observed that in most models, PCA performs better than LDA. This can be explained by the fact that PCA approaches the task in a more general way by ignoring the individual classes and attempting to find directions that maximize the variance of the dataset. On the other hand, LDA assumes that each class follows a gaussian distribution and thus tries to minimize the within class variance and maximize the between-class variance. However, in models like Gaussian NB and GMM, LDA seems to perform better than PCA which can be explained from the fact those models also assume gaussian distributions of our data, just like LDA.

5.2.2 Comparison of different transformations

In order to study the effect of different transformations to our data, we used entropy transformation (figure 7), binary transformation (figure 9) and log transformation (figure 10) in conjunction with PCA. Overall, it can be observed that binary transformation has the worst average performance out of the 3 transformations. This can be explained by the fact that binary transformation may lead to loss of information by assigning values 0 or 1 to our data, as compared to the other 2 transformations.

As far as the 2 other transformations are concerned, it can be seen that, on average, entropy transformation provides slightly better results than log transformation. This can be explained by the fact that although both transformations contain the term $log(1 + x_{ij})$, the entropy transformation is also multiplied with a term related to the entropy of our data. Hence, entropy transformation encapsulates more information that log transformation.

5.2.3 Model Comparison

By comparing figures 7,8, 9 and 10 it evident that SVM performs best among all models, even surpassing the performance of ANN and CNN. This is to be expected, as SVM is generally regarded as a very robust model for the specific type of problems [3].

Additionally, if consider ANN as a good reference model, we can observe that MEAN classifier has a surprisingly good performance given its very simple implementation. Also, KNN seems to perform very well, given that in most cases it outperforms CNN and in some cases it outperforms ANN. The solid performance of KNN is corroborated by the fact that KNN is also considered very robust for the problems of text classification [3].

References

[1] Sebastiani F., Machine Learning in Automated Text Categorization, ACM Computing Surveys, vol. 34 (1),2002, pp. 1-47.

[2] Stephen Robertson, Understanding inverse document frequency: on theoretical arguments for IDF, Journal of Documentation, Vol. 60 Issue: 5, 2004, pp.503-520

[3] Ana Cardoso-Cachopo, Arlindo L. Oliveira, An Empirical Comparison of Text Categorization Methods, Lecture Notes in Computer Science, Volume 2857, Jan 2003, Pages 183 - 196

[4] Y. Pawar, Pratiksha & Gawande, Shravan. (2012). A Comparative Study on Different Types of Approaches to Text Categorization. International Journal of Machine Learning and Computing. 423-426. 10.7763/IJMLC.2012.V2.158.

[5] Ikonomakis, Emmanouil & Kotsiantis, Sotiris & Tampakas, V. (2005). Text Classification Using Machine Learning Techniques. WSEAS transactions on computers. 4. 966-974.