Section Based Document Representation for Text Classification

submitted in partial fulfillment for the degree of master of science

Taxiarchis Papakostas-Ioannidis
11407387

master information studies
data science
faculty of science
university of amsterdam

2018-06-28

	Internal Supervisor	External Supervisor
Title, Name	Prof. Dr. Evangelos Kanoulas	Dr. George Tsatsaronis
Affiliation	UvA	Elsevier
Email	e.kanoulas@uva.nl	g.tsatsaronis@elsevier.com

Universiteit van Amsterdam

Amsterdam
Data Science

ELSEVIER

Section based document representation for text classification

Taxiarchis Papakostas-Ioannidis
University of Amsterdam
Amsterdam, The Netherlands
taxiarchis.papakostas@gmail.com

Prof. Dr. Evangelos Kanoulas
University of Amsterdam
Amsterdam, The Netherlands
e.kanoulas@uva.nl

Dr. George Tsatsaronis
Elsevier
Amsterdam, The Netherlands
g.tsatsaronis@elsevier.com

ABSTRACT

Text classification algorithms usually treat documents as a unified text representation. In literature only a few attempts have been made to segment and represent a document in parts (passages), for training a machine learning text classification algorithm. The scope of this work is to study whether a representation of a document in terms of sections, outperforms in text classification the traditional full text document representation. For this purpose, different text modellings have been applied, on a section/subsection level, and have been tested, compared and evaluated on different machine learning classifiers. This research has been carried out in collaboration with Elsevier, as a thesis internship, and was performed in a period of three months.

KEYWORDS

text classification, text representation, test segmentation, majority vote, tf-idf, LDA

1 INTRODUCTION

Text classification or text categorization is one of the most important parts of natural language processing. Some text mining applications can be found in information retrieval, sentiment analysis, ranking and classifying documents etc. With the data volume exponentially increased during the last two decades, classification of text data plays a strategic role for companies that leverage them. Web-pages, web-apps, twitter, and blogs are some examples of web sources containing vast amounts of text data on the web. The academic community constantly investigates new and more robust algorithms and methods, for increasing the accuracy of the text document classification task [1, 2]. However, text classification algorithms may not always perform the same way with different and more diverse data-sets [3]. For this purpose, several ensemble models based, on different text representations and algorithms are being built, tested and evaluated for each case.

Most of the information retrieval and natural language processing algorithms in literature treat documents as whole text representation. That is, a text classifier is trained on a pre-classified text document (full document), by representing it as a vector in a n dimensional vector space. Only a few attempts have been made to segment (split) the document and represent it by means of its passages [4–6]. Since then, more expressive text modelling representations have been designed and several classification algorithms have been improved. Examples are the Latent Dirichlet Allocation (LDA) [7] and word2vec [8]. Usually, scientific papers, books and articles contain an internal structure given by their authors, based on a section and sub-section level. However, the impact that the internal structure of a document by means of sections/subsections might have, in the task of text classification with the use of more recent algorithms and text representations, has not been investigated so far to the best of our knowledge..

1.1 Thesis Motivation

Elsevier is an information company which provides worldwide scientific, technical and medical information. The number of articles that publishes every year in about 2,500 journals are more than 400,000. Over13 million documents and 30,000 e-books, in electronically text format, exist in its archives and databases[1]. In this sense, it is the ideal environment for conducting a research in the area of natural language processing and informational retrieval.

The motivation behind this research thesis is to investigate if a section-based representation in scientific papers is more effective than the whole text representation in an application of a machine learning text classification. In the following sections, a novel approach is presented for classifying documents and scientific papers, based on their internal section/subsections structure, by taking in consideration the work of Jinsuk Kim and Myoung Ho Kim [6] and Thien Hai Nguyen and Kiyoaki Shirai [9]. This procedure is applied to a data-set of scientific papers, and the findings are compared with the traditional full document representation procedure, described in section 5. In the end, a concluding analysis is carried out among the two procedures.

In this work, two different text representation methods have been applied, namely tf-idf and LDA, and five different classification algorithms. These are, multinomial naive bayes, multinomial logistic regression, linear support vector machine (SVM), decision tree and random forest.

2 RELATED WORK

2.1 Text Classification

Text classification is an important task in natural language processing (NLP) and information retrieval (IR). Given a set of x documents and a set of n pre-defined classes, an algorithm is being trained to recognize and assign labels (classes) to a set of y un-labeled documents as illustrated in figure 1. Since machines are not able to extract information from raw text, as humans do, documents are modeled in a numerical form. Several text modeling representations convert words and sentences to probabilities or numerical vectors in some n dimensional feature space. Thus, a great variety of machine learning algorithms, deep learning algorithms and ensemble methods can be found in the literature [10].

[1] https://en.wikipedia.org/wiki/Elsevier

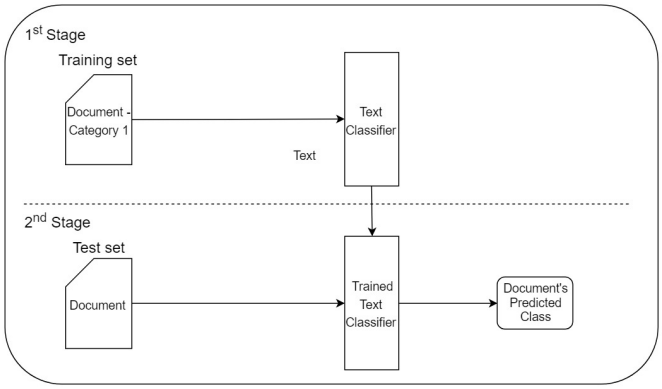

Figure 1: Traditional Text Classification

These classification algorithms can be divided into three major categories: the supervised, the unsupervised and the semi-supervised algorithms. In the first category, algorithms are trained upon a labeled data-set with pre-defined number of classes. Some examples of them are the naive bayes, the logistic regression, the support vector machine etc. On the other side, unsupervised algorithms use unlabeled data points, text in our case, and try to infer the number of classes following a set of un-supervised algorithmic steps. Clustering algorithms is an example category of un-supervised algorithms. Finally, semi-supervised algorithms combine the above mentioned techniques by using only a limited amount of labeled examples or labeled parts [11].

Thus, a classifier that assigns labels among only two classes is called a binary classifier and the one that predicts among them more than two classes is called a multi-class classifier. The task of predicting more than two classes is considered more difficult and will be the main goal of this thesis.

During the last decades, several studies on text classification have been presented in the literature. State of the art results have been produced by deep neural networks, like convolution neural networks (CNN) [12] and long short-term memory networks (LSTM) [13], but also from ensemble models, a combination of machine learning algorithms, like stacking support vector machines (SVMs) [14]. Nevertheless, more simple algorithms like naive bayes, multinomial logistic regression and linear support vector machines, generally have a good performance and notable accuracy in most of the cases. Peng et al.[18] and Li et al. [4] are examples of works, where high accuracy results were obtained by using a support vector machine classifier.

Since in this work classifiers are treated as black boxes in order to evaluate the proposed novel approach, more complicated algorithms, like deep neural networks, are beyond the scope of this research and are not used. Furthermore, based on initial choice to work with Apache Spark[2], a distributed big data environment, only certain number of algorithms are available. These are implemented to work in a cluster of computers by using parallelization optimization techniques and processes. These algorithms were mentioned in section 1.1 and are multinomial naive bayes, multinomial logistic regression, linear support vector machine (SVM), decision tree and random forest.

[2]https://spark.apache.org/

2.2 Text Segmentation

Documents usually have a conceptual structure provided by their authors. This structure is, in most cases, composed by sentences, paragraphs, sections and subsections in continuous text segments. Humans can distinguish these different parts of a text and categorize more easily what they read. For example, the structured content of a medical scientific paper helps a doctor to distinguish the paper's main topic and decide if it is useful for a treatment or a surgery. Therefore, there is a clear indication that this conceptual representation might lead to a better performance of algorithms related with natural language processing and information retrieval.

James P. Callan [15] mentions in his paper that past research results show better performance in the task of information retrieval and cites three passage segmentation techniques. Furthermore, Jinsuk Kim and Myoung Ho Kim in their proposed method [15] use one of the techniques proposed by Callan for subdividing and classifying documents. These text segmentation techniques, as presented by the above mentioned authors in their literature review, are: the paragraph passages, the bounded-paragraph passages and the window passages. Paragraph passages are based on the semantic document structure indicated by the author. This basically means that the document is segmented according to the author's concept for the structure of the manuscript. The bounded-paragraph technique divides paragraph's text into smaller parts (passages) with a fixed length interval between 50 and 200 words [15]. Paragraphs which contain a number of words under the minimum, i.e. 50, are merged with the following ones. In this way, the above mentioned two techniques assume if a writer correctly created the semantic structure of the document (paragraphs, sections, subsection).

The window passages technique follows a different procedure. It overlooks the structure of the document and treats it as a unified text without an internal structure (paragraphs, section etc.). The text is treated, regardless of its structure, as a set of passages, which are called "windows". It also introduces an overlapping mechanism where the second half of each window passage is being shared with the following window. This technique eliminates both issues, namely the writer's ambiguous conceptual text segmentation and the problematic single non-overlapping windows division. Similarly, the length of windows can be varied.

In a more recent paper, Thien Hai Nguyen and Kiyoaki Shirai [9], investigate the impact major sections, like abstract, introduction and conclusions might have in the text classification of a technical paper. However, the authors do not take in their consideration all the sections and subsections of the paper.

So, the current research tries to investigate whether a text segmentation in a section and subsection level improves text classification of scientific papers. By splitting the document based on how the authors arranged their scientific work in a conceptual level of sections, a novel approach is presented in the following.

2.3 Text Representation

Due to the inability of computers in interpreting plain text, words and sentences are converted to probabilities or vectors in the space.

In the literature, several techniques which face the problem of representing text for processing have been presented. This work focuses more in two robust models: a) term frequency - inverse term frequency representation (tf-idf), which is going to be used as the simplest text feature vector representation model, and b) the Latent Dirichlet Allocation (LDA) [7] as an advance text feature vector representation. The focus models (a) and (b) have been used in various research works [16–18] of modelling documents for text classification.

2.3.1 Tf-Idf Vector Space Model. A very simple and efficient way of representing text data, when modeling text with machine learning algorithms, is the bag-of-words model (BoW). In the BoW approach, by counting the frequency of words/terms (tf), a document d can be represented as a fixed length vector containing those frequencies ($tf_{t,d}$). The length of the vector represents the unique words (corpus) inside a collection of N documents. However, this modeling of documents has a flaw. More frequently words have more weight. For that purpose, the invert term frequency (idf) is used in addition, which scales the frequency number by dividing the total number of documents in the collection (N) with the number of documents containing the term in the collection (document frequency - df_t), and then the logarithm of the quotient is calculated[3]. By multiplying the term frequency with the inverse term frequency, a new weight is assigned to each word, which normalizes the big differences caused due to the tf by creating a weighted vector for each document, expressed with equation (1).

$$tfidf(t;d) = tf_{t,d} \cdot \log\left(\frac{N}{df_t}\right) \tag{1}$$

2.3.2 Latent Dirichlet Allocation. Latent dirichlet allocation (LDA) is an unsupervised probabilistic model and is clearly defined by Blei et al. [7]. Based on the assumption that a document is generated and is composed by a κ number of topics, and by having a collection of X documents, LDA tries to identify, probabilistically, those topics that better describe this collection. There is not any golden rule of how many topics should be chosen, so, in each case this is defined by the user after experimenting with several values. Then, by using an expectation maximization (EM) algorithm, as an optimization function with a pre-defined number of iterations, LDA algorithm is trained in the whole collection of documents trying to maximize the maximum likelihood. Finally, the trained LDA model transforms each document in a κ length (dimension) vector, in which the distribution of the topics probabilities is contained. Most of the time, the number of topics is rather smaller than the number of corpus terms, and this leads to a significant feature dimension reduction [17]. In this point, it should be mentioned that the algorithm does not assign a label to each topic but only learns to recognize it, and assigns each time a probability to the document based on the percentage that each topic is appeared in the text. After that, assuming that documents from the same class have an identical topic distribution, a text classification algorithm can be trained by giving the entire document to the classifier as a κ-length vector, containing the probabilities of all the topics. This basically means that LDA modeled and represented the text in features in a lower dimension space.

Peng Li, Jun-Qing He and Cheng-Long M [18] in their work, used LDA for modeling short parts of text, having been trained in a universal data-set. This helped them to generate more features for training a text classifier. Also, they experimented with different number of κ topics, ranging from 20 to 200, picking the one that gave them the higher accuracy result for their classifier. In another paper, Lei Li and Yimeng Zhang [17] experimented with a range of different numbers of κ topics, form 0 to 50. Based on their experiments, the 50 topics was found to be the number which maximizes their support vector machine (SVM) classifier accuracy. However, since in this research five different classifiers are used, the ideal number of κ topics, which provides the best accuracy for most of the classifiers, must be properly set.

So, for this research, LDA is used as an advanced text modeling numerical representation in order to evaluate the performance of our proposed model by experimenting with a lower dimension feature vector representation. Furthermore, we investigate if the algorithm is more robust when it is trained in shorter parts of text, based on the section/subsection representation, than to the whole document representation .

3 EXPERIMENTAL SETUP

3.1 Experimental Environment

As an experimental environment for this thesis, Elsevier's Databricks[4] platform has been selected and used for writing the machine learning pipelines. Databricks platform was created by the Apache Spark inventors. The company, by using Apache Spark, offers a variety of tools for helping clients to deploy cloud based big data solutions [5]. Apache Spark is an open-source big data processing environment. In this project thesis, Pyspark, Python's API for Spark, has been used for building code. Furthermore, Spark contains higher level components, like Mlib, for scalable Machine learning on the cloud. For this purpose, Databricks was the ideal platform for writing a distributed code in python which can be scaled further in a much larger data-set and cluster of computers. In addition, Elsevier's xml articles database is stored in an Amazon S3 bucket[6] which was easier accessible via Databricks.

3.2 The Dataset

In Elsevier's databases there are more than 72 million unlabeled published articles from Scopus in an XML format. For the purpose of this research, a subset of these articles is needed to be matched with their labels for constructing a final gold data-set. ArXiv [7] is a repository of electronic preprints (known as e-prints) approved for publication after moderation, that contains open access published scientific articles from several domains, like Physics, Mathematics etc. Elsevier has, until now, published about 30% of the arXiv's articles. Each of them is classified with different hierarchical labels. Based on the assumption that machine learning text classification algorithms do not demonstrate a good performance when they

[3] https://en.wikipedia.org/wiki/Tf-idf

[4] https://databricks.com/

[5] https://en.wikipedia.org/wiki/Databricks

[6] https://aws.amazon.com/s3/

[7] https://arxiv.org/

have to distinguish lower level differences between documents, four classes from the third level the Physics domain have been chosen to test our hypothesis (figure 2).

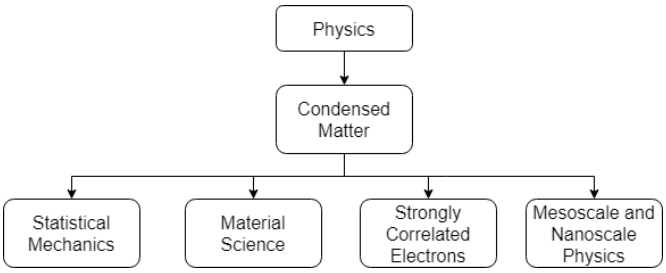

Figure 2: Task counter

For this task, the Digital Object Identifiers (DOIs) of the arXiv's articles meta-data are needed to be matched with the Elsevier's Publisher Item Identifier (PII) for retrieving the .xml articles from the database. For achieving this, the meta-data (in .xml format) from arXiv's Physic domain articles (around 850.000) have been downloaded using its API and parsed locally by creating a list of matching DOIs per document classes (labels). Subsequently, the DOIs have been matched with the PIIs on the Elsevier's cloud platform (Databricks) and 6916 articles (in .xml format) have been merged with their categories based on their common DOIs. The final imbalanced document collection consists of four categories as it is presented in table 1.

Category	Number of documents
0: Materials - Science	2375
1: Statistical Mechanics	2061
2: Mesoscale and Nanoscale	1460
3: Strongly Correlated Electrons	1020

Table 1: Number of documents per category of the chosen data-set

In the final stage, an Elsevier's xml Java parser has been selected among others and reconfigured for extracting the sections and subsections, in a text format, for all the matching articles. Furthermore, for the same data-set, the articles have been saved in a full document format in order to compare the two representations, namely full-text representation and section-based representation, based on the main research hypothesis question. Ultimately, tables have been created on Databricks by uploading the final transformed data-sets to an Amazon S3 bucket.

3.3 Hyper-parameter Tuning

Hyper-parameter tuning is a very important process for improving the performance of machine learning algorithms on a particular data-set or data-sets. So, as a first step, all the machine learning classifiers were needed to be tuned in accordance with their most important hyper-parameter. Therefore, a two fold cross-validation method was applied for tuning the more robust hyper-parameters which generalizes better the models. Accuracy was selected as the measure for evaluating the performance of those different hyper-parameters. Two folds have been chosen for this task since the computational cost grew substantially when the number of folds is increased. This offered the chance for trying a great variety of hyper-parameters combinations and for defining those which give the best results.

After the optimization of the classifiers, it was important to specify the number of κ topic for the LDA modeling representation. Following the work of [17, 18], we focused between five different κ numbers of topics because of the computational power limitations. This was the case because five different machine learning classifiers had to be trained simultaneously and measured based on their accuracy for each candidate number. The numbers of κ topics chosen to experiment with, were 20, 30, 50, 70, 100. The selection was based on the performance of the section-based methodology for the algorithms. This means that the LDA algorithm trained each time on document's sections and not in the full text of the document. The results of the algorithms performance are presented in figure 3.

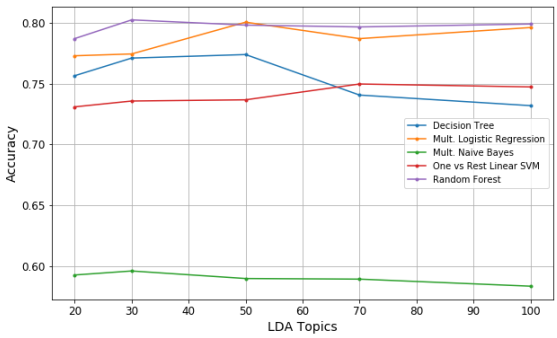

Figure 3: Classifier's accuracy versus LDA topics

It is clearly from figure 3, that 2 out of 5 classifiers, namely logistic regression and decision tree, had their best performance with a number of 50 topics. Also linear SVM and random forest performance are very close to their best with this number of topics. The only one that performed slightly worst was the multinomial naive bayes which seems to be the worst classifier. Thus, the concluding number of topics for the LDA algorithm was the 50 number of κ topics.

3.4 Evaluation Measures

For evaluating the performance of the selected methods, four different evaluation measures have been used. These were accuracy, precision, recall and f1-score. In addition to that, for the above last three, their micro, macro and weighted values were reported.[8] Micro-scores measure the global performance of an algorithm by counting the total true positives (TP), false negatives (FN) and false positives (FP). On the other hand, the macro scores take in their

[8] http://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_recall_fscore_support.html

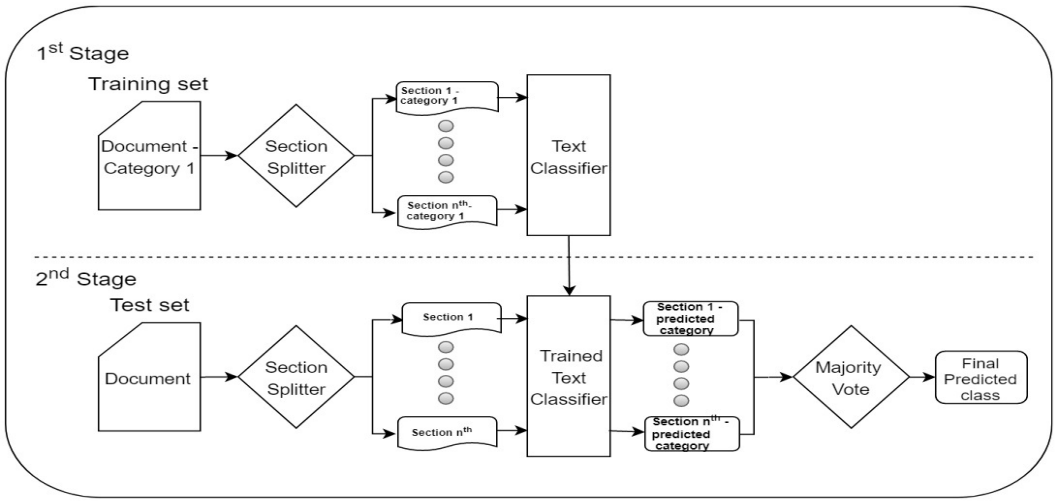

Figure 4: Section - based Text Classification (Proposed)

consideration each class by computing their TP, FN and FP and report a final score based on them. However, they are not a suitable measure for an imbalanced data-set. For that purpose weighted-scores counting the TP, FN and FP of each class are calculated in order to find the average of them. This fixes the issue introduced by macro-score. In order to convert the scores to percentages, all the measures were rounded and multiplied with 100.

3.4.1 Accuracy. Accuracy is the ratio of the total number of the correct classifier's predictions divided by the total number of elements in a test-set (2). However, this only gives an abstract picture of a classifier's performance since it does not take in its consideration the accuracy scores per class. Thus, this might leads to a bias classifier which correctly predicts a subset of classes each time and misclassify each time the rest due to an imbalanced training data-set.

$$accuracy = \frac{TP}{ALL} \tag{2}$$

3.4.2 Recall . Recall is the ratio of the true positives predictions divided by the number of true positives plus the false negatives (3). When a low recall is observed, there is an indication that the classifier categorizes many documents as FN.

$$recall = \frac{TP}{TP + FN} \tag{3}$$

3.4.3 Precision . Precision is the ratio of the true positives predictions divided by the number of true positives plus the false positives (4). When a low precision is observed, there is an indication that the classifier categorizes many documents as FP.

$$precision = \frac{TP}{TP + FP} \tag{4}$$

3.4.4 F-score. The f-score is the equally weighted harmonic mean of the above two scores, and is the ration of the precision multiplied by recall twice, divided by their summation (5). This means that the f-score is the balanced score between precision and recall and it constitutes a very robust measure for evaluating properly an algorithm.

$$precision = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{5}$$

4 METHODOLOGY

4.1 Proposed Methodology

Our proposed methodology introduces a different way of training a classifier and making predictions, as it is presented in figure 4. As a first step, using only the labeled training set, a manuscript's internal structure is recognized by an xml parser which splits the document in sections/subsections. Each part of them, is assigned with the same class of the original document and is fed to a classifier. After the classifier is trained, the same procedure is followed for the test set. By splitting the documents in sections/subsections, the classifier does not predict the document's class directly, but through an intermediate step. Thus, in each piece of text, a class is being given (predicted) from the classifier and by performing a majority vote in the end of the pipeline, the final class of the article is defined. The majority vote step can be a simple counting, where the document's final class is the most frequent class, or a probabilistic counting, where the probabilities for each class are being summed up for choosing the category with the highest summed probability for the final document prediction. This 'divide and conquer' technique, is compared with the traditional classification training and test procedure, as presented in section 2.

4.2 Machine Learning Pipeline

A machine learning pipeline, is a discrete number of steps which results to a complete solution to a specific problem. Thus, the pipeline is the framework for a) converting raw data to data usable by a machine learning algorithm, b) training a machine learning algorithm, and finally c) using the output of the machine learning

algorithm to perform actions in the real-world[9]. In this thesis, two major pipelines were designed, built, tested and evaluated. One for classifying articles as a full document and one as an ensemble classification based on the section and subsections of articles, following the framework shown in figure 5.

Initially, in both experimented pipelines, tokenization of the raw text was applied and all the stop words and punctuation symbols have been removed. Moreover, all words were converted to lowercase and were represented on a vector space by using tf-idf text modeling or the LDA algorithm. In that point, LDA algorithm needed first to be trained into the whole document collection in order to model text as a distribution of κ topics. Afterwards, in the first pipeline the traditional text classification method has been used which treats documents as a one text representation, and in the second one proposed methodology was applied for splitting text in sections/subsections. Finally, the predicted results were evaluated based on the selected evaluation measures. For both pipelines, the same five multiclass classification algorithms were selected. These are: Multinomial Naive bayes, one-vs-all Liner SVM, Multinomial Logistic Regression, Decison Trees and Random Forest.

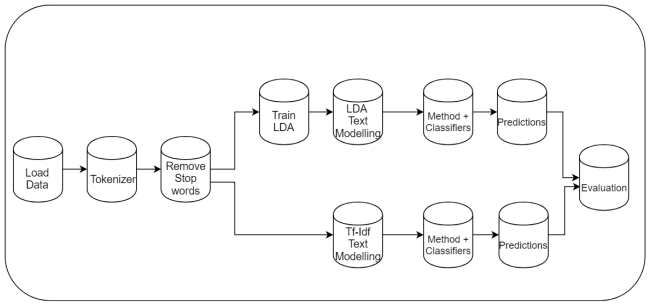

Figure 5: Abstract Pipeline Framework

5 EXPERIMENTAL RESULTS AND EVALUATION

5.1 Oracle Section-based Performance

Based on the assumption that there is a method, algorithm or ensemble, which could correctly recognize and assign final document's class, with only one section's prediction, then the potential of this scenario needs to be investigated and the upper bound performance to be set. This basically means, it should be at least one section that has been correctly labeled among all the others, which would be spotted and chosen by one efficient and enough robust method. The purpose of calculating these upper bound limits was to validate and compare the final evaluation scores as well as to illustrate the potentials of the section-based prediction in comparison with the document-based one. Thus, all documents which had a least one correct predicted section label, were set as true positives and the weighted f1-score, for each algorithm (NB = multinomial naive bayes, LR = multinomial logistic regression, SVM = linear support vector machine , DT = decision tree and RF = random forest) and for each class separately, were calculated as they are presented in table 7 and table 3. Therefore, there is some strong evidence, in both text modellings, namely tf-idf, lda, of the great potentials in the section-based proposed method. In the following sections, the experimental results as well as a deeper analysis in a class and document level, are presented, by using the proposed methodology as described in section 4.1.

	Section-based Oracle				
Classes	NB	LR	SVM	DT	RF
all	94.47	95.98	95.14	93.13	72.92
0	98.32	98.03	98.40	99.50	100.0
1	98.93	99.59	99.43	98.84	99.43
2	94.65	96.81	95.50	94.77	68.70
3	94.46	96.08	94.46	86.41	28.65

Table 2: Tf-Idf - Section-based Oracle

	Section-based Oracle				
Classes	NB	LR	SVM	DT	RF
all	51.70	93.16	85.62	94.19	92.69
0	99.86	98.62	99.06	98.47	98.99
1	98.84	99.09	99.43	99.51	99.26
2	10.86	92.78	89.24	93.66	91.63
3	0.00	91.27	62.61	93.36	89.89

Table 3: LDA - Section-based Oracle

5.1.1 Tf-idf Results. The experimental results for each one of the algorithms applied in the tf-idf modelling are illustrated in table 4. The results from all algorithms are presented for: a) the full document-based methodology and for two variations of the section-based methodology, that is for b) majority vote and c) probabilistic majority vote.

For a global evaluation, the accuracy, the macro averages (precision, recall, and f1-score), micro-averages (precision, recall and f1-score) and weighted averages (precision, recall and f1-score) were calculated and presented as percentages. As it can be seen, there is either a slight or a significant improvement in all of the measures for most of the algorithms using the proposed methodology, except in the case of random forest classifier. More specifically, the section-based method achieved the highest scores and managed, with the application of the naive bayes algorithm, to overcome by 1%, and in some measures more than 1%, the highest score obtained with the full-document representation. Furthermore, the scores were improved significantly by 3% to 4% with the application of algorithms like logistic regression, linear support vector machines and decision tree. In addition to that, logistic regression and linear support vector machine scores were increased and approached the highest accuracy score, that is of naive bayes in the full-document

[9]https://www.quora.com/What-is-a-pipeline-and-baseline-in-machine-learning-algorithms

	Full Document					Majority Vote					Probabilistic Majority Vote				
Measures	NB	LR	SVM	DT	RF	NB	LR	SVM	DT	RF	NB	LR	SVM	DT	RF
Accuracy	**82.55**	**78.35**	**79.40**	69.60	75.57	**83.13**	**82.02**	**82.46**	69.70	62.69	**83.52**	**82.46**	-	**72.89**	63.90
Macro-Prec.	81.23	76.92	78.54	66.89	82.21	82.01	80.93	83.23	69.08	80.77	82.67	81.40	-	71.57	80.75
Macro-Recall	80.66	76.09	76.32	67.03	67.05	81.07	79.98	80.09	63.51	51.00	81.68	80.45	-	66.77	51.70
Macro-FScore	80.89	76.29	77.04	66.96	67.43	80.89	76.29	77.04	66.96	67.43	82.12	80.85	-	67.47	45.33
Micro-Prec.	82.55	78.35	79.40	69.60	75.57	82.94	82.02	83.33	69.84	63.22	83.52	82.46	-	72.89	64.08
Micro-Recall	82.55	78.35	79.40	69.60	75.57	82.94	82.02	83.33	69.84	63.22	83.52	82.46	-	72.89	64.08
Micro-FScore	82.55	78.35	79.40	69.60	75.57	82.94	82.02	83.33	69.84	63.22	83.52	82.46	-	72.89	64.08
Weigh.-Prec.	**82.78**	78.48	**79.21**	69.75	**79.27**	**82.78**	78.48	**79.21**	69.75	**79.27**	**83.66**	**82.30**	-	72.44	**75.77**
Weigh.-Recall	82.55	78.35	79.40	69.60	75.57	82.94	82.02	83.33	69.84	63.22	83.52	82.46	-	72.89	64.08
Weigh.-FScore	**82.61**	**78.26**	**79.04**	69.67	72.49	**82.96**	**81.87**	**82.99**	68.41	53.34	**83.53**	**82.32**	-	**71.48**	54.49

Table 4: Tf-idf - Evaluation Measures

	Full Document					Majority Vote					Probabilistic Majority Vote				
Measures	NB	LR	SVM	DT	RF	NB	LR	SVM	DT	RF	NB	LR	SVM	DT	RF
Accuracy	75.29	**81.00**	**78.97**	69.73	**80.85**	58.68	**78.83**	72.50	**75.50**	**79.07**	59.01	**79.02**	-	**77.09**	**79.99**
Macro-Prec.	77.97	79.75	78.40	67.62	80.20	55.97	77.85	74.54	75.42	79.38	56.09	78.64	-	76.37	79.85
Macro-Recall	66.64	78.24	73.43	66.42	77.18	46.13	73.66	63.95	70.89	74.45	46.46	74.52	-	72.42	75.02
Macro-FScore	66.30	78.87	74.44	66.85	78.26	36.66	74.81	64.02	71.95	75.76	36.90	75.71	-	73.49	76.32
Micro-Prec.	75.29	81.00	78.97	69.73	80.85	58.63	78.44	72.74	75.79	79.36	59.01	79.02	-	77.09	79.99
Micro-Recall	75.29	81.00	78.97	69.73	80.85	58.63	78.44	72.74	75.79	79.36	59.01	79.02	-	77.09	79.99
Micro-FScore	75.29	81.00	78.97	69.73	80.85	58.63	78.44	72.74	75.79	79.36	59.01	79.02	-	77.09	79.99
Weigh.-Prec.	76.78	**80.81**	**78.71**	69.42	**80.79**	60.68	**78.28**	74.14	**75.65**	**79.41**	60.83	**78.91**	-	**76.83**	**80.00**
Weigh.-Recall	75.29	81.00	78.97	69.73	80.85	58.63	78.44	72.74	75.79	79.36	59.01	79.02	-	77.09	79.99
Weigh.-FScore	72.00	**80.80**	**77.84**	69.41	**80.49**	46.14	**77.69**	69.45	**74.82**	**78.56**	46.43	**78.35**	-	**76.32**	**79.19**

Table 5: LDA - Evaluation Measures

representation. However, the weighted average precision score and the macro average f-score are slightly lower, which indicates that the section-based representation tends to predict more fall positives (FP) for these two algorithms (LR and SVM).

In the case of the random forest classifier, the results showed that RF classifier does not perform as well as the other classifiers do. The observed low scores may be due to the high dimensionality and sparseness of the tf-idf text representation when applied to a short text (sections/subsections). Moreover, the high scores of the macro-averaged and weighted average precision in contrast with the low macro-averaged and weighted average recall scores indicate that the sparseness of the tf-idf vectors influences only the false negative predictions (FN), a fact that leads to a significant drop of accuracy and f1-score. The same applies for the decision tree classifier, but in a smaller scale and regardless of the improvement observed with the section-based approach.

Moreover, the high scores of the macro-averaged and weighted average precision in contrast with the low macro-averaged and weighted average recall scores indicate that the sparseness of the tf-idf vectors influences only the false negative predictions (FN), a fact that leads to a significant drop of accuracy and f1-score.

5.1.2 LDA Results. Table 5 illustrates the corresponding results for the LDA modelling. Although with the tf-idf text modelling most of the algorithms increased their performance, with the LDA modelling the results showed a different trend. With the exception of the decision tree classifier, the other algorithms performed poorer compared to the proposed section-based methodology. The resulted scores dropped by 1% to 17% in almost all measures, with multinomial naive bayes to having the lowest. One possible explanation for the naive bayes low scores is the small data-set's class imbalance (skewness) in combination with the naive's bayes assumption of treating independently text features, but also the LDA's feature dimensionality reduction. However, this biased behavior was not further investigated, since this was out of the scope of this research. On the other hand, there was a 7% increase in the decision tree text classifier's scores. This is an indication that decision tree might perform better in comparison with the other algorithms when small size text is expressed with more condensed feature vectors (LDA).

5.2 Methods Comparison

From the above presented results, it may be concluded that there is a difference between the full-document and the section-based

methodology. The second, which is proposed in the present work, performs better with distributed (tf-idf) features than with the probabilistic ones (LDA). Thus, for shorter text length, tf-idf modelling can be considered more robust for the majority of the tested classifiers, as it increases substantially the performance of the section-based approach. Unfortunately, because Spark has not yet implemented the linear SVM classifier for multi-class classification, the algorithm was only tested by applying the one versus all technique which returns only the final class prediction and not the prediction's probabilities. For that reason, there are no results for the probabilistic majority vote.

Nevertheless, it should be noted that the decision tree classifier shows a considerably improved performance, in both text modellings, when the probabilistic majority vote is applied. This performance improvement ranges from 1% to 3%. This is an indication that in small size texts, in our case sections/subsections, the probability 'level' of the algorithm is higher, which technically means that the predictions are more accurate. However, the macro-averaged recall's low scores, indicate that the decision tree classifier might categorize the same amount of documents as false negatives (FN), like in the case of full-document representation.

Another key point is that between the simple majority approach and the probabilistic one, there are no significant differences in accuracy, only a slight improvement of 0.5%-1%. It is also worth to mention that in multinomial logistic regression with tf-idf features, weighted average precision increased by almost 4%. This indicates that the probabilistic method compared with the simple majority, categorizes fewer documents as false positives (FP).

6 EVALUATION ANALYSIS FOR EACH CLASS RESULTS

6.1 Analysis of Top Performance for Each Class

To gain a deeper understanding of the section-based proposed methodology performance, a deeper analysis has been conducted for each combination of the examined algorithms, modellings and methods, by measuring the f1-score on data-set's classes separately. Table 6 illustrates those combinations for the top two observed performances for each one of the classes. A detailed table of all the f1-score measures can be found in the appendix.

The column with the title "Method" presents the combinations of text modellings and methodologies in which the biggest f1-score was observed for each class. The other two columns with titles "Algorithm" and "F1-score" present the applied algorithm and the weighted average f1-score respectively. By looking the first column in the upper and down part of table 6, it is clear that the section-based methodology tends to perform better for most of the classes. Furthermore, among all text modellings and methodologies, probabilistic majority vote seems to be more robust than the simple majority vote.

From the table's data, it is concluded that the multinomial naive bayes algorithm performs better for classes 0, 2, 3. Surprisingly, naive bayes algorithm is not included in neither of the top performances of the class 1. The highest weighted average f1-score among the classes is 91.71%, a fact that indicates that Linear SVM and Logistic regression capture accurately the curve of the class data.

	Top Performance		
Classes	Method	Algorithm	F1-Score
Class 0	Tf-Idf+Prob. Majority	Naive Bayes	83.81
Class 1	Tf-Idf+Full Document	Linear SVM	91.71
Class 2	Tf-Idf+Simple Majority	Naive Bayes	77.78
Class 3	Tf-Idf+Probab. Majority	Naive Bayes	75.64
	Second Top Performance		
Classes	Method	Algorithm	F1-Score
Class 0	LDA+Full Document	Random Forest	83.65
Class 1	Tf-Idf+Probab. Majority	Log. Regression	91.50
Class 2	Tf-Idf+Probab. Majority	Log. Regression	76.70
Class 3	Tf-Idf+Simple Majority	Naive Bayes	74.48

Table 6: F1-scores for each class

Another key point is that class 2 and class 3 have the lower scores, 77.78% and 75.64% accordingly, among all the classes. Since the weighted average f1-score is the measuring index, the data-set's small class imbalance cannot be considered the reason for these low score observations. One possible explanation for this behavior could be that the vocabulary of these classes cannot be easily be distinguished from the first two classes.

6.2 Analysis of Differences Between Classes

Besides the analysis of top performance for each class, an attempt has also been made to investigate and mark if any differences appear between the four classes, by means of averaged document length and averaged number of sections. For that purpose, any patterns between the observed true positives (TP) and false positives (FP) predictions of each class have been investigated, for all combinations of the methods, algorithms and text modellings that have been used during this research. In this point, it should be mentioned that in the created data-set some outliers, in this case wrong parsed documents, which were not filtered out properly. This might be the cause of some differences observed during this analysis between document's averaged length numbers or the averaged sections numbers. Nevertheless, there was no strong evidence indicating that the article's length or the number of sections/subsections influences the results between the two compared methodologies, namely full-document and section-based, in total and for each class separately. One possible suggestion for further work could be to measure the cross-entropy, then to calculate the cosine similarity of each class, and finally to compare the cosine similarities between all classes. This might identify any significant differences between the vocabulary distributions of the classes. All the relevant tables of this analysis can be found in the appendix.

6.3 Confusion Matrix Analysis of Best Algorithms

Since the created data-set contains four different classes from the same scientific domain (physics-condensed matter) an analysis in the confusion matrices could result in some interesting insights.

	NB Confusion Matrix				LR Confusion Matrix			
Predict. Actual	0	1	2	3	0	1	2	3
0	602	23	49	24	572	31	60	35
1	23	551	19	20	22	574	11	6
2	79	10	349	15	69	17	349	18
3	34	6	42	223	36	21	37	211

Table 7: Confusion matrices of the top performed algorithms

Thus, the confusion matrices of the top two performed algorithms, that is to say the multinomial naive bayes and the multinomial logistic regression, have been selected and analyzed to identify possible patterns with the proposed methodology. In table 7, these confusion matrices are illustrated according to the classifier's results on the probabilistic majority vote, in combination with the tf-idf numerical representation.

By looking both confusion matrices column-wise the same pattern is observed, that is the amount of false positives (FP) are more in class 0 and 2, in comparison with the other two, surpassing the value of 110 in each. Class 1 and 2 on the other hand seem to have less than 60 false positives, which is a clear indication that they do not influence the predictions of the other classes to the same extent. A second observed pattern, by looking both confusion matrices row-wise, is that most of the false negatives (FN) from each class are false classified as class 0 and 2 in most of the cases. The only exception is noticed in class 0, while class 3 seems to be the second most false positive predicted class.

Finally, an overall analysis in the matrices suggests that class 2 and 3 are the most probable of misclassified, which means that the probability a section or a subsection to be classified as class 0 or class 1 is higher if an article comes from these two domains.

7 CONCLUSIONS

Text classification algorithms usually treat documents as a unified text representation. This work presents a new methodology which leverages the internal structure of a scientific paper by means of sections and subsections and investigates if this methodology shows a better performance in the task of text classification. In more details, the proposed strategy in combination with different algorithms and text modellings was applied to a sample of 6916 scientific papers, downloaded from Elsevier's databases, and the experimental results were compared with the results from the application of the traditional text classification process of full-document classification.

The results showed that a better performance was achieved by the proposed methodology as compared to the traditional one. The reported results in accuracy and weighted-average f1-score, 83.52% and 83.53% accordingly, outperformed by 1% the results from the full-document approach. Furthermore, it was found that most of the algorithms improved by 3% and 4%. The evidence from this study suggests that the application of the proposed methodology has great potential for an increase in accuracy since scores of almost 95.98% can be achieved.

Moreover, the in depth analysis which was conducted in the four classes of the data-set, for exploring possible major differences in performance between the methods, algorithms and modelling investigated on this thesis, showed that the section-based methodology performed better in the majority of the classes. This is based on the reported top two weighted-average f1-score performances which achieve 83.81% in class 0, 91.50% in class 1, 77.78% in class 2 and 75.64% in class 3. Most of this work's source code can be found in the following github repository: https://github.com/aris-papakos/Thesis.git.

8 FUTURE WORK

Selection of a right text modelling and classifier in conjunction with a different and more robust majority vote are the two important aspects on which future experimental investigations should focus.

Therefore, by taking into account the proposed methodology in this thesis, further work needs to be done for demonstrating whether different text modellings and more advanced text classification algorithms could result to a better performance. Different text numeric representations such as word2vec or doc2vec [19] are suggested to be used instead of tf-idf and latent dirichlet allocation (LDA). Training LDA in the full-document text and then applying it to each section could be considered also as an alternative approach, since it was not tested in the present work, due to time limitations.

Moreover, state of the art text classification algorithms are recommended to be developed and tested in the section-based text classification methodology. More specifically, deep neural networks such as long short-term memory (LSTM) and convolution neural networks (CNN) are two suggestions to be examined. In addition, the performance of non-linear support vector machine (SVM) classifiers may be a framework for future research. The proposed methodology in this work gives a great potential for improvements which might be explored with algorithms like deep neural networks in combination with word embeddings.

Finally, there are clear indications that there might be a more robust method than the two proposed variations of the majority vote, which better synthesizes document's sections predictions in order to define a final class. It is recommended to undertake further research in data engineering and to a more weighted majority vote that could possibly lead to substantial improvements. In addition, it should be considered that some text sections probably add only noise to the final prediction of the majority vote. A well-designed majority vote could possible filter out all the unnecessary sections and weight accordingly the rest. In the end, an open issue is also whether an ensemble method of different algorithms and text modellings can be considered as a suggestion for leveraging the difference in their characteristics.

ACKNOWLEDGMENTS

First of all, I would like to express my very great appreciation to Prof. Dr. Evangelos Kanoulas and Dr. George Tsatsaronis, my research supervisors, for their valuable expert advice, useful critiques, and constructive recommendations throughout this thesis process.

I would also like to offer my grateful thanks to Elsevier for offering all the appropriate resources and giving me the chance to work in an inspiring and collaborative work environment.

I would especially like to thank my family for their continuous encouragement, support and help.

REFERENCES

[1] Lai S, Xu L, Liu K, Zhao J. Recurrent Convolutional Neural Networks for Text Classification. InAAAI 2015 Jan 25 (Vol. 333, pp. 2267-2273)
[2] Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T., 2016. Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759.
[3] Wang, S. and Yao, X., 2009, March. Diversity analysis on imbalanced data sets by using ensemble models. In Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on (pp. 324-331). IEEE.
[4] Li, K., Xie, J., Sun, X., Ma, Y. and Bai, H., 2011. Multi-class text categorization based on LDA and SVM. Procedia Engineering, 15, pp.1963-1967.
[5] Hearst, M.A., 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. Computational linguistics, 23(1), pp.33-64.
[6] Kim, J. and Kim, M.H., 2004. An evaluation of passage-based text categorization. Journal of Intelligent Information Systems, 23(1), pp.47-65.
[7] Blei, D.M., Ng, A.Y. and Jordan, M.I., 2003. Latent dirichlet allocation. Journal of machine Learning research, 3(Jan), pp.993-1022.
[8] Mikolov, T., Chen, K., Corrado, G. and Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
[9] Nguyen, T.H. and Shirai, K., 2013, June. Text classification of technical papers based on text segmentation. In International Conference on Application of Natural Language to Information Systems (pp. 278-284). Springer, Berlin, Heidelberg.
[10] Aggarwal, C.C. and Zhai, C., 2012. A survey of text classification algorithms. In Mining text data (pp. 163-222). Springer, Boston, MA.
[11] Nigam, K., McCallum, A. and Mitchell, T., 2006. Semi-supervised text classification using EM. Semi-Supervised Learning, pp.33-56.
[12] Kim, Y., 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.
[13] Lai, S., Xu, L., Liu, K. and Zhao, J., 2015, January. Recurrent Convolutional Neural Networks for Text Classification. In AAAI (Vol. 333, pp. 2267-2273).
[14] Sebastiani, F., 2002. Machine learning in automated text categorization. ACM computing surveys (CSUR), 34(1), pp.1-47.
[15] Callan, J.P., 1994, August. Passage-level evidence in document retrieval. In Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 302-310). Springer-Verlag New York, Inc..
[16] Zhang, W., Yoshida, T. and Tang, X., 2011. A comparative study of TF* IDF, LSI and multi-words for text classification. Expert Systems with Applications, 38(3), pp.2758-2765.
[17] Li, L. and Zhang, Y., An empirical study of text classification using Latent Dirichlet Allocation.'
[18] Peng, L.I., Jun-Qing, H.E. and Cheng-Long, M.A., 2016. Short Text Classification Based on Latent Topic Modeling and Word Embedding. DEStech Transactions on Computer Science and Engineering, (aice-ncs).
[19] Le, Q. and Mikolov, T., 2014. Distributed representations of sentences and documents. In International Conference on Machine Learning 2014(pp. 1188-1196).

A APPENDIX

	Full Document					Majority Vote					Probabilistic Majority Vote				
Classes	NB	LR	SVM	DT	RF	NB	LR	SVM	DT	RF	NB	LR	SVM	DT	RF
0	83.53	79.12	80.81	72.23	79.52	83.45	81.57	82.81	73.74	70.29	83.81	81.89	-	76.56	70.97
1	91.71	88.09	87.72	79.84	86.05	91.42	91.28	90.88	79.37	82.14	91.50	91.40	-	83.02	81.08
2	75.27	72.04	73.85	64.33	67.81	75.52	76.11	77.78	63.88	21.79	76.32	76.70	-	67.13	25.90
3	72.88	66.43	65.93	51.43	36.13	74.48	72.73	70.94	35.99	1.30	75.64	73.39	-	43.17	3.86

Table 8: Tf- idf - F1 scores per class (1) Full Document, (2) Majority Vote, (3) Probabilistic Majority Vote

	Full Document					Majority Vote					Probabilistic Majority Vote			
Features	NB	LR	SVM	DTC	RFC	NB	LR	SVM	DT	RF	NB	LR	DT	RF
Class 0														
TP Avg Length 0	3582	3576	3505	3705	3588	3543	3767	3599	3634	3555	3571	3767	3645	3549
TP Avg Sections 0	9	9	9	10	9	9	10	10	10	9	9	10	10	9
FP Avg Length 0	3348	3390	3508	3631	3145	3576	3471	3371	3157	3079	3593	3615	3206	3018
FP Avg Sections 0	9	8	8	9	8	9	8	9	9	8	9	9	9	8
Class 1														
TP Avg Length 1	4590	4659	4637	4052	4585	4606	4613	4637	4877	4641	4621	4602	4761	4665
TP Avg Sections 1	10	11	10	10	10	10	10	10	11	10	10	10	10	10
FP Avg Length 1	5798	6427	7065	5820	5299	5770	4894	5877	5544	7435	5918	5035	5669	7299
FP Avg Sections 1	13	14	14	12	12	13	11	12	11	14	13	12	12	14
Class 2														
TP Avg Length 2	4188	4325	4280	3908	4413	4297	4214	4336	4454	6463	4287	4201	4237	5791
TP Avg Sections 2	9	9	9	9	9	9	9	9	9	10	9	9	9	9
FP Avg Length 2	4144	4249	4336	4791	5129	4122	3515	3946	4370	3587	4075	3516	3857	3646
FP Avg Sections 2	9	9	9	10	11	9	8	9	9	7	9	8	9	7
Class 3														
TP Avg Length 3	5232	4586	4050	5008	7429	5201	5025	4965	6301	7081	5075	4953	5671	7591
TP Avg Sections 3	11	10	10	11	14	11	11	11	12	13	11	11	12	11
FP Avg Length 3	4612	3279	3309	5056	7851	4113	4026	3399	4317	NaN	4105	3975	5040	NaN
FP Avg Sections 3	10	8	8	10	16	9	9	8	10	NaN	9	9	10	NaN

Table 9: Tf-df - Analysis TP FP

	Full Document					Majority Vote					Probabilistic Majority Vote				
Classes	NB	LR	SVM	DT	RF	NB	LR	SVM	DT	RF	NB	LR	SVM	DT	RF
0	78.96	81.04	80.80	71.73	83.65	67.60	80.61	78.83	79.20	81.72	67.74	81.20	-	80.38	81.87
1	86.37	90.34	88.71	80.76	89.32	79.13	88.75	84.91	84.50	88.71	79.80	88.84	-	85.52	89.59
2	67.01	73.67	73.12	63.15	74.77	0.44	71.97	66.26	69.23	72.90	0.44	72.41	-	69.93	74.00
3	34.04	66.43	60.66	54.39	68.39	0.00	57.94	28.18	57.79	62.23	0.00	60.94	-	56.78	61.93

Table 10: LDA - F1 scores per class (1) Full Document, (2) Majority Vote, (3) Probabilistic Majority Vote

	Full Document					Majority Vote					Probabilistic Majority Vote			
Features	NB	LR	SVM	DTC	RFC	NB	LR	SVM	DT	RF	NB	LR	DT	RF
Class 0														
TP Avg Length 0	3502	3559	3475	3715	3687	3516	3552	3562	3606	3627	3511	3551	3667	3617
TP Avg Sections 0	9	9	9	10	10	9	9	9	9	9	9	9	9	9
FP Avg Length 0	2935	3373	3255	3493	3586	3423	3086	3120	3211	3359	3424	3118	3204	3323
FP Avg Sections 0	7	8	8	8	9	8	8	8	8	8	8	8	8	9
Class 1														
TP Avg Length 1	4599	4577	4610	4635	4518	4740	4563	4596	4664	4662	4718	4561	4663	4651
TP Avg Sections 1	10	10	10	10	10	10	10	10	10	10	10	10	10	10
FP Avg Length 1	6492	7269	6725	6655	6392	7256	5675	4948	6146	7003	7363	5702	5944	6393
FP Avg Sections 1	13	15	14	13	13	14	12	12	13	14	14	13	13	14
Class 2														
TP Avg Length 2	4457	4256	4385	3849	4235	6489	4382	4680	4319	4173	6489	4373	4329	4163
TP Avg Sections 2	9	9	9	8	9	9	9	9	9	9	9	9	9	9
FP Avg Length 2	4426	3644	4150	3718	3746	NaN	3987	4355	2882	3805	NaN	4160	2995	3927
FP Avg Sections 2	11	9	10	9	9	NaN	9	11	7	9	NaN	9	7	9
Class 3														
TP Avg Length 3	5966	4524	4605	4455	4772	NaN	5498	7684	5094	4750	NaN	5257	5347	5272
TP Avg Sections 3	10	10	9	10	10	NaN	11	11	11	NaN	11	11	11	11
FP Avg Length 3	5102	5199	5150	4002	5427	NaN	5676	5621	3793	4118	NaN	5736	3606	4036
FP Avg Sections 3	10	11	10	9	12	NaN	11	13	9	9	NaN	11	9	9

Table 11: LDA - Analysis TP FP