A Performance Comparisons of Machine Learning Classification Techniques for Job Titles Using Job Descriptions

Shreyans Mittal[1], Shubham Gupta[1], Sagar[1], Apoorv Shamma[1], Ishaan Sahni[1], Narina Thakur[2]
[1]Computer Science Engineering Department, Bharati Vidyapeeth's College of Engineering, New Delhi
Email: mittalshreyans99@gmail.com, sg741236985@gmail.com, sagar01441@gmail.com, apoorv.shamma@gmail.com, ishaansahni10@gmail.com, narinat@gmail.com

***Abstract*— *An inappropriate candidate shortlisted and a potential one missed simply means an inappropriate resume linked to the incorrect keyword. Document classification is being excessively researched upon these days, due to growing interest in text classification which has become a major contributor to the online texts and documents. The repetitive tasks of a person categorizing the details can be handled by the machinery using an expert system that correctly captures and identifies the text and then classifies it into different categories defined. After the preprocessing of the data, the classification is done as a comparative analysis of Bernoulli's Naïve Bayes, Multinomial Naïve Bayes, Random Forest, Linear SVM and LSVM with elastic penalty classification on the Top 30 Job listing dataset with different parameters and thus we are able to analyze the dependencies between different terms in classes with varying densities and accounts. The accuracy was evaluated and LSVM provides the best accuracy in classifying job entitlements based on the queries submitted and was able to achieve 96.25% accuracy for 55000 samples.*

Keywords - TF-IDF, LSVM, BNB, MNB, RF.

1. Introduction

Document classification is gaining popularity nowadays because it forms a primary and major step taken towards document processing [4]. In this age of world-wide-web, the internet is just getting embedded in our day-to-day lives and what is the most important factor in this automated generation is to know how we can punch in some useful and informative data from the one that is existing on the internet. To counter these problems of extracting, transforming and classification of the data into useful information, automated document classification is one of the basic methods that can be performed to cater to the issues. Earlier the task of classification of text was done by human knowledge based on their decision-making abilities. It is a time consuming and tedious process of employing humans to do a mechanical repetitive task which can be automated using machines with a better accuracy. Due to an unstructured document present as the data to be fed, there are many features that can be extracted and selected using feature engineering and thus, this poses to be a very significant challenge during document classification. Thus, there are a lot of tasks that automated document classification performs. To start with, documents can be automatically distributed into different spatial locations and can be archived instantly without any compromise of human time. To cite an example of this, after a proper business meeting, the article or the letter can be sent to the apt department for further processing of it [6]. Document classification can be domain specific and domain dependent knowledge that can be imparted to our system for classification of specialized documents only and thus, it performs specific tasks efficiently. Document classification can be used to extract a lot of heterogeneous documents from the database to extract different documents and then classify them based on same visual quality [8], tables, graphics, etc. This can enhance the document image retrieval expert system and make it more robust. Document classification has been widely studied in machine learning, data processing and data retrieval communities and has also been applied in various industrial settings. An automatic approach to job title classification reduces to the matter of text document classification with machine learning. A machine learning-based document classification approach requires labeling documents with predefined classes to form a group of coaching data. Several classification algorithms like Support Vector Machines (SVM) [1] and k-Nearest-Neighbor (kNN) [2] have been applied to numerous industrial problems with good empirical results. Machine learning applications in industry frequently have the necessity to process very large datasets. Several large-scale distributed machine learning frameworks have adapted machine learning algorithms to distributed and scale up architectures [3]. These frameworks are applicable to the increasingly common application scenarios that involve very large datasets and have counts for computationally demanding training of complex learning models likewise as near real-time inference constraints. In this paper, a system has been proposed of classification of documents based on text classification. The aim to generate the text vector using TF-IDF vector technique in which the term frequency that is construction of a dictionary of words that are occurring in the document and then constructing a vector out of it. This vector is then also constructed for inverse frequency words, which means the numeric value of the documents in which the term can occur. Based on these two vectors which are constructed out of the extracted features, preprocessing them and transforming them to a desired form learning algorithms are applied to them, which are Bernoulli's Naïve Bayes (BNB), Multinomial Naïve Bayes (MNB), Random Forest (RF) and Linear SVM (LSVM) for classification into 30 categories of the job titles given to the worker.

The paper has been organized this in the following way. Section 2 discusses the literature survey that is carried out, whereas, section 3 brings about the experimental setup which describes the data set and the evaluation methodology. Section 4 tabulates the results, followed by section 5 which concludes the paper.

2. Document Classification

Numerous studies have been done on various techniques of document classification. There is a great variety of document classifiers that are being researched upon, basically depending on the features that are being constructed for the purpose of classification. As we have already discussed that document classification has major applications, the features used for the purpose of classification can be image and pixel-based features or they can be text-based features present within the document for classifying it into different categories. Within text classification, a lot of work has been done to classify the words with maximum accuracy that help in enhancing the document classification system. One model suggests that the words can be [4] vectored in binary classification (as performed in one-hot encoding method) to see which all words occur (denoted by a 1) and which all don't occur (denoted by 0) and form a vector of the dictionary of occurrences and non-occurrences. The following discusses the feature extraction, preprocessing and classification techniques that have been employed in history by the researchers so far.

2.1. Feature Extraction

In general texts and documents contain an unstructured flow of data. This conversion of extracting meaningful and useful data from the unstructured parameters by modeling them mathematically in a structured space refers to feature extraction. In conversion of the dataset, there might be some parameters that occur as a part of our modeling and do not contribute to the accuracy of our model, these can be eliminated later using the preprocessing techniques. TF [9], IDF, TF-IDF, Word2Vec [10], Global vectors for word representations (GLOVE) [11] for embedding the words are such techniques that can prove to be very good in extraction of features from the document. Broadly, the technique used is TF-IDF which is explained in the following paper.

A. Term frequency

The sentence that is used to train a model for text mining or text classification is known as a corpus. In information retrieval-based systems, term frequency-inverse document frequency is abbreviated as TF-IDF which is a numerical statistical means of fitting the importance of a word in a document in a corpus. Often being used as a weighting factor that weighs off the words present in the document, it abides to this role in many information retrieval systems, text mining applications as well as in user modeling. There is a direct proportionality of the weight of the word with respect to its occurrence in the corpus that suggests us that some words are more frequently occurring than the other terms and they contribute to comparatively higher weight than the others. Suppose we have a set of English text documents and wish to rank which document is most relevant to the query, "the brown cow". A simple way to start out is by eliminating documents that do not contain all three words "the", "brown", and "cow", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document; the number of times a term occurs in a document is called its term frequency. However, in the case where the length of documents varies greatly, adjustments are often made. The weight of a term that occurs in a document is simply proportional to the term frequency.

Table 1 - Variants of TF weights

Weighting scheme	Term frequency
Binary	0,1
Raw count	$f_{t,d}$
Term frequency	$f_{t,d} / \sum_{t_r \in d} t_r, d$
Log normalization	Log (1+$f_{t,d}$)

Table 1 shows different variants of Term frequency

B. Inverse document frequency

The term "the" is so common, term frequency will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "brown" and "cow". The term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms, unlike the less-common words "brown" and "cow". Hence an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. These words are stop words that needs to be handled carefully in order to impart more accuracy to our model. The specificity of a term can be quantified as an inverse function of the number of documents in which it occurs is inverse frequency of the word.

Table 2 - Variants of IDF weights

Weighting scheme	IDF weights
Unary	1
Inverse document frequency	Log ($\frac{N}{n1}$)= -log$\frac{n1}{N}$
Inverse document frequency smooth	Log ($\frac{N}{1+n1}$)
Probabilistic Inverse document frequency	Log $\frac{N-n1}{n1}$

Table 2 shows different variants of Inverse Document Frequency

C. Term frequency–Inverse document frequency

The tf–idf is calculated as:

$$tf - idf\ (t, d, D) = tf(t, d).idf\ (t, D)$$

A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms.

Since the ratio inside the idf's log function is always greater than or equal to 1, the value of idf (and tf–idf) is greater than or equal to 0. As a term appears in more documents, the ratio inside the logarithm approaches 1, bringing the idf and tf–idf closer to 0.

2.2. Dimensionality reduction

As data that is obtained in the highly structured format by performing feature engineering on the unstructured data, there are a lot of terms and uniquely identified words that can occur in the text and as the length of the document increases, the dimensionality of the dataset also increases manifold. This directly contributes in increasing the compilation time, time and space complexities of the model and the efficiency of the model hampers. Thus, dimensionality reduction techniques [12] like PCA, Linear Discriminant Analysis, non-negative matrix factorization (NMF), random projections, autoencoders and t-distribution stochastic methods have been used in past. Nowadays, some modern methods like stop word removal [21] and stemming [23] are mostly used and they broadly fall under the category of pre-processing.

2.3. Classification

This is the most crucial and researched upon step as it is very important to choose the model that will be classifying the document based on the text. Without a deep understanding of how the model is functioning and how the algorithm is working mathematically and statistically, we cannot put our extracted features to draw a good and remarkable accuracy. One of the simplest classifications based on Logistic Regression was done for classifying the document and has been considered as the steppingstone in many text mining applications [13]. Naïve Bayes [1] has also been a very popular technique even in the earliest techniques of classification of text particularly because of its assumptions of independence and needs a very low memory for execution. Supervised learning methods required a labeled structured dataset for classification. Thus, a lot of other supervised learning algorithms have been employed to exploit this field of text mining in past for document classification. This includes K-nearest neighbors (KNN) [14], Support Vector Machine. These techniques have been used in bioinformatics, image sensing, human activity sensing, paging, indexing of the documents in the digital library and in so many real-life applications that makes them a field of research themselves.

Naïve Bayes model of classification works on many assumptions. One such assumption that the model makes is that in multinomial classification [15] (in modeling the text into more than two classes of classification) the length of the document has no relationship with the classes. This is in line with the Naïve Bayes assumptions that the probability of occurrences of each word in the document is independent of its position or place of occurrences in the document, which makes the model simpler and lot less complicated in forming a network of words for constructing a vocabulary for categorization of the document. One of the main utilizations of SVMs as falls for characterizing enormous heterogeneous web content was talked about in [4]. The methodology utilized SVMs as both coarse- and fine-grained classifiers with the condition that an edge score must be surpassed at the coarse level before a grouping could be made at the fine-grained level. The Deep Classification strategy by Xue et al [20] handles progressive characterization utilizing a two-stage methodology. In the main stage a classification search is led to acquire a lot of competitor records for the inquiry archive. These up-and-comer reports are utilized to prune the chain of importance and train a classifier to mark the question record. A major downside of this methodology is the over the top computational prerequisite to train a classifier for each question report These candidate documents are used to prune the hierarchy and train a classifier to label the query document. A big drawback of this approach is the exorbitant computational requirement to train a classifier for every query document. For determining the quality of a classifier other than accuracy scores, confusion matrices [25] can also be used. The accuracy of the classifier is verified and checked with the help of chi-square selection [24], which is mentioned in Section 4.

A. Support Vector Machines

SVM is a supervised machine learning algorithm that is used for both regression as well as for classification. In SVM, the data points are plotted in an n-dimensional hyperplane that tries to differentiate the classes that we want our data to get categorized to. SVM has been used [16] to categorize only the positive data and classify them into a one class classification method and this method gave tremendously best results. In one such method [17], SVM has been used in document ranking and provided whether the document was relevant or irrelevant.

Another type of sub - categorized model is Linear SVM with elastic penalty which defaults to L2 the standard regularized for LINEAR SVM Models. The regularizer is a penalty added to the loss function that shrinks model parameters towards zero vector using either the squared Euclidean norm L2 or the absolute norm L1 or a combination of both. Using *'elasticnet'* as penalty value brings sparsity to model which cannot be achieved by 'L2' [18].

B. K-Nearest Neighbor

The basic idea here is to [2] determine the nearest neighbor that implies to the nearest plot point in the document space. A vector is constructed for each test set and centroid vector for each class is constructed that calculates the similarity between each class and the vector constructed and then classifies it based on the class it matches the most or for the class for which similarity is maximum.

C. Naïve Bayes

Naïve Bayes is the simplest of all the algorithms yet imparts a very good accuracy in document classification model. Bayesian classifiers have always used the classic mathematical and statistical means and assumptions for classification. In case of text mining and its applications, the presence or

absence of the word determines the outcome prediction. The keywords are searched for and stored in a map or a vector that tries to index each word that is occurring in the document and the words that can most likely occur in the document. Then it calculates the probability of the word for the test document and then classifies the test documents based on the probabilities calculated.
The algorithm is however, considered Naïve because all the terms occurring in the document are independent of each-other.

D. Random Forest

The idea of Random Forests [26] was presented by Breiman in 2001. It is a group of un-pruned classification or regression trees and it is made using the random selection of the training data. In the induction process, random features are selected. Prediction is made by the prediction of the ensemble. This classifier usually gives better results as compared to single tree [27].

3. Experimental Setup

In this section Top 30 Job entitlement classification dataset used for the experiment has been illustrated with the performance evaluation methodology.

A. Data set

The dataset used for the purpose of research in this paper is Top 30 Job entitlement classification dataset named dataset.zip which contains CSV file named Top30.csv. It has 72292 non-null rows and query of job and their description as the attributes. It has 4 columns out of which 2 were redundant. There are 30 distinct classes which consist of the job that are to be entitled to different people based on their submitted query description. The size of the dataset is 18 MBs. The query and the description attributes of the dataset are of type Object. Most of the jobs were related to Administrative Assistant and Customer Service Representative with 4395 and 4200 queries, respectively. We got this dataset from Kaggle [22].

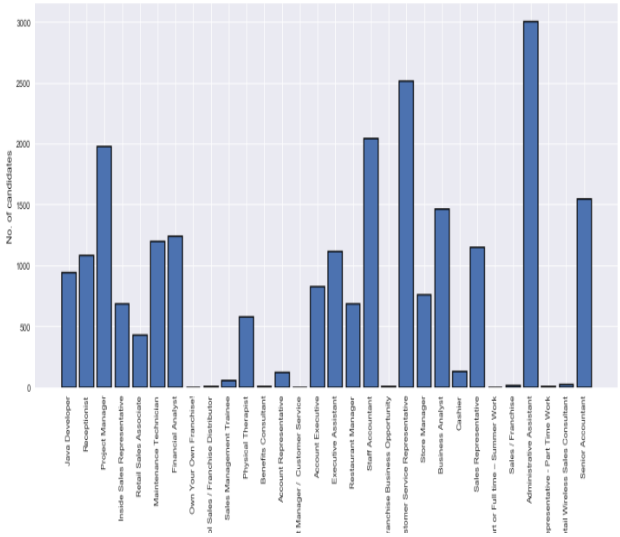

Figure 1 - Distribution of various jobs across the dataset

The data is splitted in the ratio of 80:20 into training and testing data. Table 3 illustrates the sample of the stratified partitioning of the data to the 80:20 ratio.

Table 3 - Example of stratified holdout data splitting

Label	Test	Train	Total	% of Total
Cashier	26	102	128	80
Java Developer	189	755	944	80
Maintenance Technician	239	957	1196	80
Project Manager	394	1578	1971	80
Receptionist	217	867	1084	80
Retail Sales Associate	85	341	426	80
Staff Accountant	409	1634	2043	80

B. Evaluation Methodology

A document classifier model undergoes several processes which can be depicted from Figure 2. It depicts the methodology beginning with data preprocessing to the model evaluation. Thus, less important features can be removed and so the computational time can be improved which is broadly mentioned in Section 2. As for the classification phase, different classifiers (such as SVM, RF and NB) are employed to generate the model. Finally, the model is trained by a set of training data and evaluated by a set of testing data which is divided in the ratio 80:20 as discussed in Section 4. In order to test the classification ability of the model, several evaluation measures were calculated in the form of prediction time, training time and accuracy scores for different classifiers. The proposed system as depicted in Figure 2, explains how the whole process of feature extraction, selection and classification is being carried out.

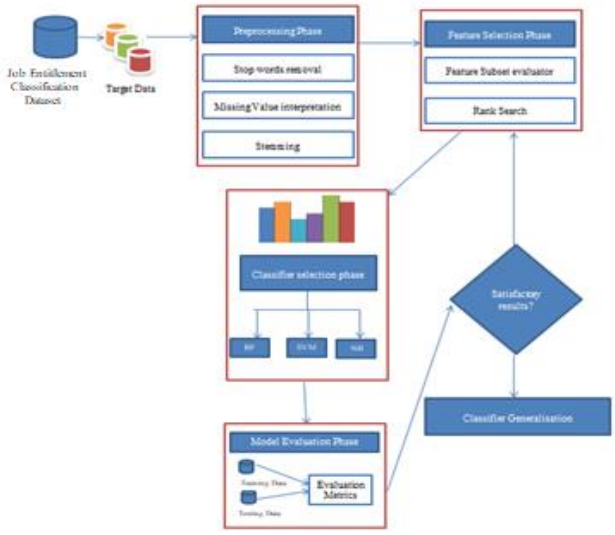

Figure 2 - Overview of experimental architecture

Used the words within a document as “features” to help predict the classification of a document. To classify these documents, start by taking all the words in the documents in our training set and creating a table or vector from these words. Then for each of the training documents, create a vector by assigning a 1 if the word exists in the training document and a 0 if it does not, tagging the document with the appropriate class. When a new untagged document arrives

for classification, create a word vector for it by marking the words which exist in our classification vector. If comparison of this vector for the document of unknown class, to the vectors representing trained model document classes, comparison can be made as to which class does it closely resembles.

Figure 3 overall summarizes the process to explain the flow of the procedure being carried out in this whole analysis.

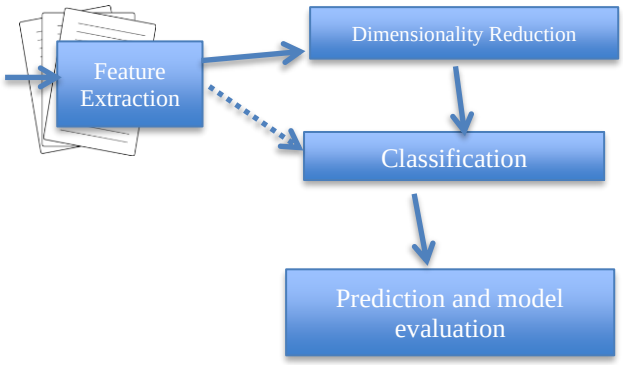

Figure 3 - Flow Chart

4. Results and Discussions

In this paper, experiment was conducted by extracting features using TF-IDF technique and then classification of the document was done using various classification techniques. Number of unique vocabulary words in the description column of the data frame after removal of stop words are 272143 which was followed by stemming. In this research, Porter stemmer was used. By using TF-IDF, the features were reduced to 42113, which is approximately $1/7^{th}$ of the original sample. This technique resulted in lower training time as well as prediction time of the model without affecting the accuracy significantly. The dataset is sliced into training and testing dataset in the ratio 80:20 and the duplications are constantly removed.

Table 4.1 - Results with 5000 samples

Technique	Accuracy scores
BNB	0.42
MNB	0.46
RF	0.80
LSVM	0.90

Table 4.1 discusses that upon taking 5000 samples for training and testing the data, the LSVM classifier works the best upon classifying different job titles based on the query descriptions.

Table 4.2 - Results with 10000 samples

Technique	Accuracy scores
BNB	0.53
MNB	0.47
RF	0.87
LSVM	0.92
LSVM with elastic penalty	0.93

Table 4.2 comprises of the results tabulated after taking 10000 samples and shows that while Naïve Bayes based classifiers show very poor tendencies of classification of documents, LSVM promises good results with the best being achieved in elastic penalties.

Table 4.3 - Results with 20000 samples

Technique	Accuracy scores
BNB	0.60
MNB	0.48
RF	0.887
LSVM	0.95
LSVM with elastic penalty	0.945

Table 4.3 tabulates results of 20000 samples and shows a somewhat different result. LSVM with elastic penalty drop their accuracy by 0.05% compared to LSVM which has achieved 95% accuracy in correctly identifying the job titles.

Table 4.4 - Results with 55000 samples

Technique	Accuracy scores
BNB	0.7237
MNB	0.5527
RF	0.9059
LSVM	0.9625
LSVM with elastic penalty	0.96038

As shown in table 4.4, there are 55000 samples and here LSVM achieves a better accuracy as compared to other classifiers and accurately identifies the job entitlement by 96.25%. Thus, this is the maximum accuracy that can be achieved on this dataset.

Figure 4 - Comparison of training time of different classifiers

Figure 4 depicts the comparison of training time of different classifiers with the minimum training time of MNB followed by BNB. LSVM with the maximum accuracy takes the maximum training time.

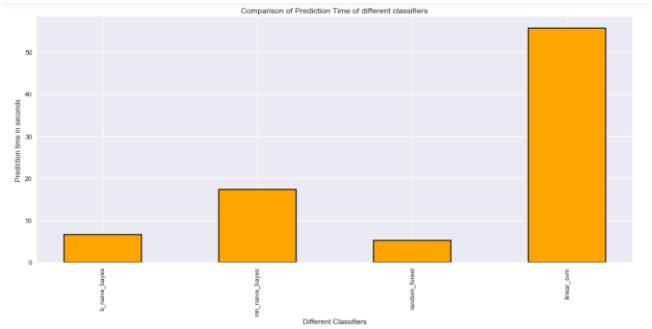

Figure 5: Comparison of prediction time of different classifiers

Figure 5 depicts the comparison of prediction time of different classifiers. Prediction time is the time required in predicting the class of a document which is found to be minimum in case of RF followed by BNB. LSVM has the maximum accuracy but takes the maximum prediction time.

The result is verified and checked for better accuracy with "chi-square selection" where 5000 top features are obtained from 42113 features by TF-IDF. The dataset is sliced into training and testing dataset in the ratio 80:20. The accuracy obtained is 96.31% which is approximately equal to 96.25% as obtained earlier in case of LSVM. Thus, on decreasing the sample size the accuracy increased with chi-square selection.

5. Conclusion

This paper mainly discusses classification algorithms based on a dataset that classifies job titles based on the query description. The experimental results show that the method not only in large datasets, set satisfactory classification results but has also shown a good classification performance on a small sample. However, traditional Bayesian classification algorithm based on word frequency in the two different datasets has a different classification effect. The best results on different data size is best demonstrated by the linear SVM. While the Naïve Bayes based classifiers poorly identified the documents, there was however a lesser calculation and execution time that was utilized by them as compared to other learning techniques. The linear SVM with elastic penalty achieved accuracies more than 90% even upon increasing the number of samples which with linear SVM, the best result on this dataset could be achieved 96.25% of the times upon correctly classifying the job entitlements. Also, applying advanced Machine learning like kernel based [16] Naïve based classifier, kernel based SVM classifier and Deep Learning techniques can improve the efficiency appreciably. Document classification linked with sentiment analysis can be used in progressive social media [19] marketing and for opinion mining as well. This classification system can be extended to all the job search engines and websites wherein multiple applicants submit their details for getting a job. This system will be helpful in successfully classifying them into categories that they want to apply for and get networked with the most appropriate employers. This way CV based document classification can curb unemployment and help people get their desired jobs. Thus, there is a lot of scope of improving the efficiency of the existing systems.

References

[1] Liu, P. Q., & Feng, J. J. (2010). An improved Naïve Bayes Text Classification Algorithm. *Microcomputer Information*, *26*(27), 187-188.

[2] Meena, M. J., & Chandran, K. R. (2009, December). Naive Bayes text classification with positive features selected by statistical method. In *2009 First International Conference on Advanced Computing* (pp. 28-33). IEEE.

[3] Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

[4] Chen, N., & Blostein, D. (2007). A survey of document image classification: problem statement, classifier architecture and performance evaluation. *International Journal of Document Analysis and Recognition (IJDAR)*, *10*(1), 1-16.

[5] Aghila, G. (2010). A Survey of Na\" ive Bayes Machine Learning approach in Text Document Classification. *arXiv preprint arXiv:1003.1795*.

[6] Brückner, T., Suda, P., Block, H. U., & Maderlechner, G. (1996, April). In-house mail distribution by automatic address and content interpretation. In *Proceedings of the 5th Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, USA* (pp. 67-75).

[7] Cesarini, F., Lastri, M., Marinai, S., & Soda, G. (2001, September). Encoding of modified XY trees for document classification. In *Proceedings of Sixth International Conference on Document Analysis and Recognition* (pp. 1131-1136). IEEE.

[8] Shin, C., Doermann, D., & Rosenfeld, A. (2001). Classification of document pages using structure-based features. *International Journal on Document Analysis and Recognition*, *3*(4), 232-247.

[9] Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, *24*(5), 513-523.

[10] Goldberg, Y., & Levy, O. (2014). word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

[11] Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

[12] Jensen, R., & Shen, Q. (2004). Semantics-preserving dimensionality reduction: rough and fuzzy-rough-based approaches. *IEEE Transactions on knowledge and data engineering*, *16*(12), 1457-1471.

[13] Brzezinski, J. R., & Knafl, G. J. (1999, September). Logistic regression modeling for context-based classification. In *Proceedings. Tenth International Workshop on Database and Expert Systems Applications. DEXA 99* (pp. 755-759). IEEE.

[14] Gayathri, K., & Marimuthu, A. (2013, January). Text document pre-processing with the KNN for classification using the SVM. In *2013 7th International Conference on Intelligent Systems and Control (ISCO)* (pp. 453-457). IEEE.

[15] A McCallum, K.Nigam. A comparison of event models for naïve Bayes text classification. AAAA-98 workshop on Learning for text Categorization, 2004

[16] Scholkopf, B., Mika, S., Burges, C. J., Knirsch, P., Muller, K. R., Ratsch, G., & Smola, A. J. (1999). Input space versus feature space in kernel-based methods. *IEEE transactions on neural networks*, *10*(5), 1000-1017.

[17] Cao, Y., Xu, J., Liu, T. Y., Li, H., Huang, Y., & Hon, H. W. (2006, August). Adapting ranking SVM to document retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 186-193).

[18] Zhou, Quan & Chen, Wenlin & Song, Shiji & Gardner, Jacob & Weinberger, Kilian. (2014). A Reduction of the Elastic Net to Support Vector Machines with an Application to GPU Computing.

[19] Yu, B., & Kwok, L. (2011). Classifying business marketing messages on Facebook. *Beijing, China: Association for Computing Machinery Special Interest Group on Information Retrieval*.

[20] G-R Xue, X. Dikan, Q. Yang, and T. Yu, "Deep classification in largescale text hierarchies," In Proceedings of ACM SIGIR '08, pp. 619-626, 2008.

[21] K., Jaideepsinh & Saini, Jatinderkumar. (2016). Stop-Word Removal Algorithm and its Implementation for Sanskrit Language. International Journal of Computer Applications. 150. 15-17. 10.5120/ijca2016911462.

[22] Bman93 - https://www.kaggle.com/bman93/dataset
Job Ttile & Job Description Dataset (April,2020)

[23] Jivani, Anjali. (2011). A Comparative Study of Stemming Algorithms. Int. J. Comp. Tech. Appl.. 2. 1930-1938

[24] Jin, Xin & Xu, Anbang & Bie, Rongfang & Guo, Ping. (2006). Machine Learning Techniques and Chi-Square Feature Selection for Cancer Classification Using SAGE Gene Expression Profiles. Lect Notes Comput Sci. 3916. 106-115. 10.1007/11691730_11.

[25] Visa, Sofia & Ramsay, Brian & Ralescu, Anca & Knaap, Esther. (2011). Confusion Matrix-based Feature Selection.. CEUR Workshop Proceedings. 710. 120-127.

[26] Breiman, L., Random Forests, Machine Learning 45(1), 5-32, 2001

[27] Ali, Jehad & Khan, Rehanullah & Ahmad, Nasir & Maqsood, Imran. (2012). Random Forests and Decision Trees. International Journal of Computer Science Issues(IJCSI). 9.

.

Acknowledgements

This research paper would not have been possible without the help of a few people.

Firstly, thanks to our mentor, Dr. Narina Thakur, who imparted valuable advice and guidance throughout the process. Her unmatched support and expertise played a crucial role in completing our research on time.

We are also proud of the way each one of us contributed significantly in this research. Without the efforts of any one of the team members, this research would not have been successful. At the end of the day, we are proud that we make a great team with utmost level of dedication and honesty, by always being self-motivated to achieve this goal.

Lastly, we would like to thank Elsevier SSRN team for taking out their valuable time to go through our research paper and henceforth publish this in their reputed series.