International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 04, April-2015

Legal Case Document Classification Application Based on an Improved Hybrid Approach

Ugwu Chidiebere
Department of Computer Science
University of Port Harcourt, Choba
Port Harcourt, Nigeria

Obasi Chinedu Kingsley
Department of Computer Science
University of Port Harcourt, Choba
Port Harcourt, Nigeria

***Abstract*—The automatic classification of legal case documents has become very important, owing to the justice denials, delays and failures observed in the judicial case management systems. Our hybrid text classification model employed extensive pre-processing techniques to prepare the document features, the probabilistic nature of the Naïve Bayes algorithm was integrated to generate vectorized data from the document features for the classifier, and the most important features was selected by feature ranking using the Chi Square method for final classification using the Support Vector Machine. The hybrid text classifier application was implemented using the Object Oriented Analysis and Design Methodology and developed using the Java programming language and MySQL. Results showed that best features were selected and the documents were accurately classified to their right categories using this hybrid application, as proven using standard performance measure metrics.**

Keywords—chi-square statistical analysis;document classification;feature selection;support vector machine; tokenization;vectorization.

I. INTRODUCTION

Many fallouts, delayed and failed judgments in legal institutions have persisted due to the failures in the judicial case management systems employed to handle these cases or the inefficient organization of these cases for easy retrieval and effective use in providing quality and timely judgments in law suits. Experts in the judicial system have always resorted to manually organizing case documents relevant for each law suit to meet their client needs, these methods will fail in cases of fire disasters and mostly observed, this rigorous process has not proven to provide adequate results, and as such justice has been denied in law suits. Case management systems developed as in-house applications has given little or no improvements over these issues.

Seamless and intelligent classification of these documents (law cases) will create easy reference to old cases needed to pass new judgments, making justice in the court very easy. When documents are automatically classified, the opportunity for mining those surfaces, vital knowledge is discovered and this enhances decision making in various processes.

Text categorization or classification according to [15] is defined as assigning new documents to a set of pre-defined categories based on the classification patterns.

Document classification can be defined as the task of automatically categorizing collections of electronic documents into their annotated classes based on their contents [10]. Document classification, supervised or unsupervised undergoes basic steps like document collection, pre-processing, feature selection, classification and most importantly performance evaluation. The supervised learning approach in automatic text classification involves training the classifier with pre-categorized class of text documents, i.e. training set and using the classifier to predict new categories of documents, i.e. test set.

Document collection involves the gathering of the data sets from a domain of interest, which is also applicable to the research area. The pre-processing stage passes the text documents through noise-reduction steps such as tokenization, stop-word removal and stemming to remove unwanted features from the corpus. Feature selection methods are used to choose the most important features through the frequency of occurrence of the words or the most ranked features using suitable methods. Classification techniques are employed to correctly assign the different documents to their respective categories. The performance of the classifier used in the automatic classification process are usually measured using the most adequate measure metrics.

In this paper, our objective is to address the need for automatic classification of legal case documents using our developed hybrid text classification model for an efficient case management system. The use of our hybrid approach demonstrates the implementation of this model which uses extensive pre-processing techniques to handle the raw legal case documents, then combines a feature ranking method using chi-square statistical analysis, Naïve Bayes algorithm for vectorization and Support Vector Machine for final classification of the documents.

II. RELATED WORKS

It is of great interest and importance to review research areas in the text classification process to discover the milestones and results obtained over time, and also to identify the areas of more research.

The strength of k-NN and Rocchio algorithms was combined in [2] to produce a hybrid algorithm based on Variable Precision Rough Set (VPRS) which improved the

performance of the two classifiers, and also increased the efficiency and performance of their text classification model. Although a significant improvement was observed, the presence of a very large training set will reduce the model's performance.

A combination of a machine learning algorithm and a rule-based expert system proposed by (Julio et al, 2011), formed a Machine Learning Expert System (MLES) which achieves a precision in the classification task whose results are improved because the base classifier filters false positives and deals with only false negatives, but there is need for improvement in the preparation of the training corpus and preprocessing of the input text as rules does not work in some algorithms when the category becomes noisy.

A new multi-label classification method based on Naive Bayesian theory was presented in [19]. It is a two-step feature selection strategy based on DF (Document Frequency), chi-square (χ^2) and FCBF (Fast Correlation-Based Filter Solution) algorithm is incorporated into the method to improve classification performance. Experiments on public multi-label corpus showed low computation costs very useful on-line applications, especially for handling high-dimensionality text data, but there is need for improvement in the Naïve-Bayes Multi-label (NBML) algorithm as it does not handle the correlation between labels.

A hybrid feature selection scheme in [16], which is composed of filter and wrapper selection stages, and decision tree and a linear version of the Support Vector Machine was used in classifying the text documents that are received from the feature selection stage, but the incorporation of other feature weighting schemes is a research issue.

In [17], a new algorithm combining elite genetic algorithm (EGA) and support vector machine is presented. The good optimization performance of support vector machines improved the classification performance and the EGA-SVM performed better than the traditional SVM and the GA-SVM when tested with the Iris data .set, though there is need for increased computational efficiency when the values of crossing-over rate, mutation size, and size of population has different values.

A novel heuristic text classification algorithm was designed in [3] based on genetic algorithm and chaotic optimization algorithm called chaos genetic feature selection optimization (CGFSO). The experiment also demonstrated that the CGFSO yields better accuracy even with a large data set since it achieved better performance with the lower number of features, but its performance was tested on Reuters-21567, so the need for testing in tougher data sets arises.

Our new architecture handles the issue of noisy input text by engaging extensive pre-processing techniques and also increase classification performance by employing better classification methods.

III. METHODOLOGY

The need for a suitable methodology in developing any software cannot be over-emphasized as it contributes to the success of every developed application. The Object Oriented Analysis and Design Methodology (OOADM) was used in the development of our hybrid application as it was best found to implement the modularity of the design.

IV. ARCHITECTURAL DESIGN OF THE HYBRID APPROACH

The text categorization model proposed in [4] as shown in Fig. 1, implemented pre-processing techniques, feature selection methods which include the chi-square (χ^2) statistic, BPSO (Binary Particle Swarm Optimization) and PCA (Principal Component Analysis) for reducing the high dimensionality in raw text documents, and classifying the selected features for the documents using C4.5 decision tree and KNN.

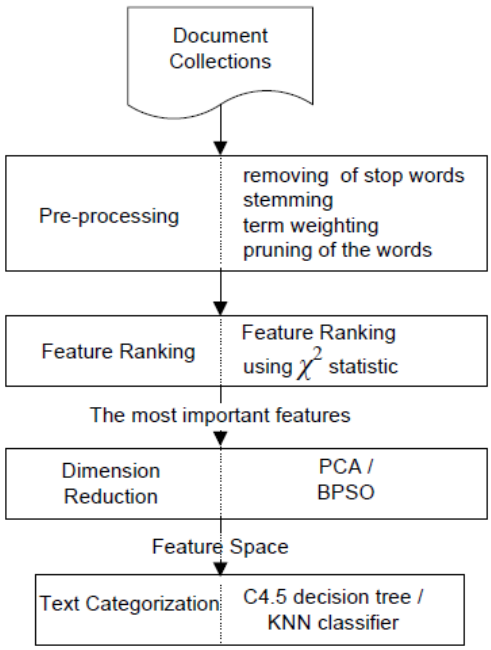

Fig. 1. Architecture of the existing text categorisation model.

Though the use of decision tree and K-NN was engaged in the classification process because of its simplicity, and easy understandability, these classifiers have a very long training time, decision trees also have over-fitting issues, and considering that the K-NN classifier makes use of distance, it becomes computationally expensive as the training size grows. Our developed hybrid approach as seen in Fig. 2 enhances the overall performance of the existing text classification architecture by implementing more extensive pre-processing techniques which includes token filtering by length and case transformation to enhance the pre-processing stage. The use of the Naïve Bayes algorithm as a vectorizer instead of a classifier in the front-end and the Support Vector Machine as the classifier showed a great deal of performance in the classification process.

The supervised hybrid text classification architecture consists of integrated stages which include document collection, pre-processing, dimensionality reduction and

vectorization, training and classification. These modules and their detailed sub-stages are depicted in Fig. 2.

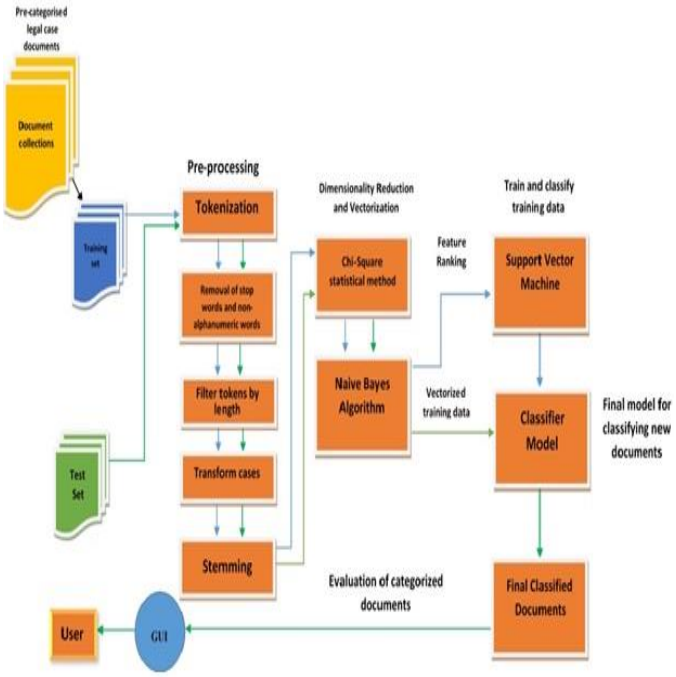

Fig. 2. The developed hybrid text classification architecture.

The documents used in our architecture are legal case documents obtained from [7] and [8]. These cases were gathered from years 2001 through 2011. The documents were in PDFs (Portable Document Formats) and HTML (Hypertext Markup Language) formats, but were converted into text format (.txt) as required by our application as this reduced the size of the application and enhanced faster processing of the documents. 57 documents were used as the training set to train the classifier, while 22 documents were used as the test set for each category to determine the accuracy of the learnt model. The total size of the documents used was 30.6 megabytes.

After the documents were collected, they were prepared to be used in easily processable formats with also the intent of eliminating irrelevant features, referred to as noise. The pre-processing stages include tokenization, stop word and alpha-numeric word removal, token filtering by length, case transformation and stemming.

The tokenization stage of pre-processing involved breaking down of the text documents into tokens (words, phrases or symbols). Considering an excerpt from one of the criminal cases, DANIEL ADEOYE V. STATE, in the Supreme Court in 1999 obtained from [8].

"It is in the interest of the public, the complainant, and the [defendant] himself that the question of guilt or otherwise be determined finally by the verdict of a jury, and not left as something which must remain undecided by reason of a defect in legal machinery."

After tokenization, the case excerpt becomes:

", It, is, in, the, interest, of, the, public, the, complainant, and, the, [, defendant,], himself, that, the, question, of, guilt, or, otherwise, be, determined, finally, by, the, verdict, of, a, jury, and, not, left, as, something, which, must, remain, undecided, by, reason, of, a, defect, in, legal, machinery, . , "

The tokens were then scanned through a file containing the stop words and alpha-numeric words. This stage becomes a necessity because these words occur more frequently and are of less importance in documents. Words like "an", "a", "the", e.t.c are stop words got from the stop list in [9], and the text file contained 571 words used in this work to form the stop word set. The non-alphanumeric words are characters like ! @ # & ( ) – [{ }] : ; ', ? / *etc. which was included in the set of irrelevant words. The resulting excerpt becomes:

interest public complainant defendant himself question guilt otherwise determined finally verdict jury left something remain undecided reason defect legal machinery

The tokens from the preceding stage were filtered and words or features with a particular character length were removed. Experiments with the bigger set of documents revealed much irrelevant words especially with a character length of 3 and below like aaa, aba, bbb. The filtered tokens were passed through the case transformation stage which involves the conversion of the words or tokens in the text documents to a particular case, upper or lower cases to avoid duplication of the same features in different cases. The architectural design implemented the conversion of all the words into lower cases. After the case transformation stage, the tokens were stemmed, this process is also referred to as normalization and it involves the conflation of words that have the same stems, and discarding their remaining suffixes. A particular stem is selected and used as a replacement for other related features with the same prefix. For instance, the stem connect- is selected for use when words like connection and connected are encountered. The Porters Stemming Algorithm in [14] was used in this work to stem the words. The resulting case excerpt remains the same in this case after these stages:

interest public complainant defendant himself question guilt otherwise determined finally verdict jury left something remain undecided reason defect legal machinery

The chi-square (χ^2) statistical analysis was used as a feature ranking method to select features of highest relevance. The chi-square statistics tells us how relevant a word/term is to each class/category, and we removed from the features, the words that are not relevant for that class/category. At the end, terms of high relevance were chosen [1]. The formula for the chi-square statistics is as follows in equation (1) below:

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)} \quad (1)$$

This is the value for a term (t) and a category(c) where:

A is the number of documents of category c containing the term t; B is the number of documents of other category (not c) containing t; C is the number of documents of category c not containing the term t; D is the number of documents of other category not containing t; N is the total number of documents. We can calculate for instance, the chi-square value for the term "interest" for the category "criminal", i.e. χ^2 (interest, criminal).

The Naïve Bayes rule was used to create term vectors instead of it being used as a classifier. Using the probabilistic characteristics of the algorithm, the posterior probability distribution over the categories of text documents for each feature was calculated from the prior probability, the likelihood and evidence of the features and the values form a multi-dimensional vector for the classifier [1, 10]. This was the vectorization process.

The posterior probability of the word in a document for a particular category, denoted as Pr(Cat|Word), which forms the final term vectors used by the SVM can be computed by using equation (2):

$$\Pr(\text{Cat}|\text{Word}) = \frac{\Pr(Word|Cat).\Pr(Cat)}{\Pr(Word)} \quad (2)$$

The training set and test set passes through the same stages as can be seen in our design. After the creation of the term vectors, the vectorized features were classified using the Support Vector Machine (SVM). The vectorized data which are the term weights was used as input to the Support Vector Machine for final training. The SVM used some of the vectorized data for training and kept some proportion of the entire training set as support vectors. This process makes the SVM more efficient and requires less resources than other classifiers like neural networks which require large training data sets in order to generalize accurately. We implemented a reusable library from one of the known SVM developed modules, libSVM to handle this multi-classification problem. After the training, a classifier model is formed, which accepts the test set that has gone through the pre-processing, dimensionality reduction and vectorization stages. The test set contains the set of features from documents to be classified. SVM being one of the best proven techniques for document classification but a bad classifier with good features can easily outperform a good classifier with bad features, hence the feature selection methods adopted in the work. The polynomial and sigmoid kernel functions from the libSVM module was used during the classification process and the best results was got using the sigmoid function. All results can be seen in Fig. 6a and 6b.

To evaluate the performance of our hybrid text classifier, accuracy, precision and recall were the evaluation metrics used on the test set. Accuracy is the ratio between the number of text documents which were correctly categorized and the total number of documents. This is calculated from equation (3):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

where TP (True Positives) is the number of text documents correctly classified in category, TN (True Negatives) is the number of text documents correctly classified as not belonging to the category, FP (False Positives) is the number of text documents incorrectly classified in the category, and FN (False Negatives) is the number of text documents incorrectly classified as not belonging to that category.

Focusing on the sigmoid kernel which gave us the best results, we calculated the accuracy of the different legal categories. Considering the Housing category, after the classification of 22 test documents, the values of TP = 19, FP = 1, TN = 1, FN = 1. The Accuracy can be calculated as follows: $\frac{19+1}{19+1+1+1} * 100\%$ = 90.91%.

Precision is the proportion of the correctly proposed documents to the proposed documents, and it can be calculated from equation (4).

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

Also considering the results, we got from the sigmoid kernel, The precision of the Housing category can be calculated as: $\frac{19}{19+1} * 100\% = 95\%.$

Recall is the proportion of the correctly proposed documents to the test data that have been proposed as in [9]. It can be calculated as seen in equation (5).

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

Considering the results we got from the sigmoid kernel, The recall of the Housing category can be calculated as: $\frac{19}{19+1} * 100\% = 95\%.$ In this work, the accuracy, precision and recall were calculated for each category, considering the classification results of the SVM kernel functions.

The class diagram used in our design is shown in Fig. 3 as it shows the modules of our architecture implemented in our application software. The classes include the performanceEvaluator class for computing the performance evaluation of the SVM kernels used during classification, preprocess class handled the all the pre-processing stages,

vectoriser class, the vectoriser class and the naiveBayes class handled the creation of the term vectors for final classification, the svmClassifier class was used for final classification of the documents, the chiSquare class was used to calculate the chi-square values for proper selection of the most important features in the documents used. The dataBaseAccess class is connected to all the other classes in the application as it is used to store and retrieve all the data and values generated from the respective classes.

The pre-processing, dimensionality reduction, vectorization and classification modules were developed and implemented using the following tools and software packages: NetBeans Integrated Development Environment (version 8.0.1), XAMPP Server (version 3.2.1) containing MySQL database, and libSVM. All experiments were run on a machine with 2.4 GHz CPU, 6 GB of RAM, 1 Terabyte HDD Space, and Windows 8 Operating System.

Fig. 4 shows the Activity diagram of the developed model, showing the sequence of activities of the user from the start process to the end process, detailing the progression of the events in the activities. The activity diagram also shows how a user can use the application to classify text documents to its different categories.

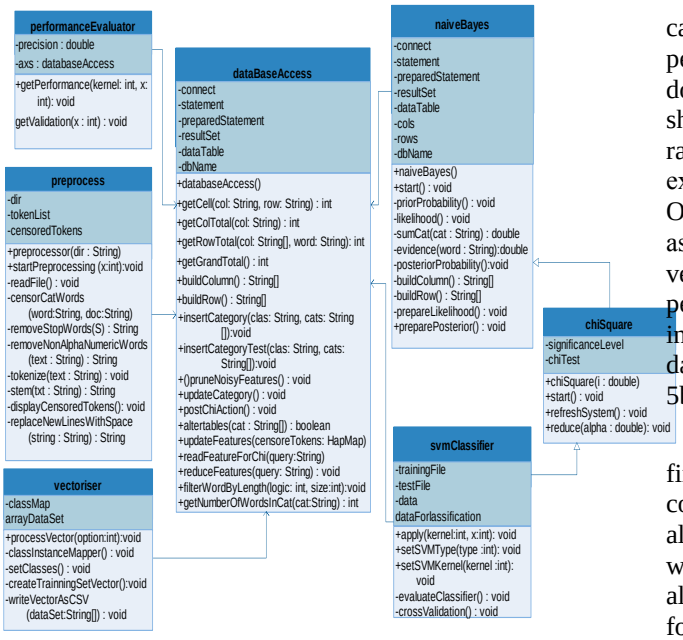

Fig. 3. Class diagram of the hybrid text classifier application.

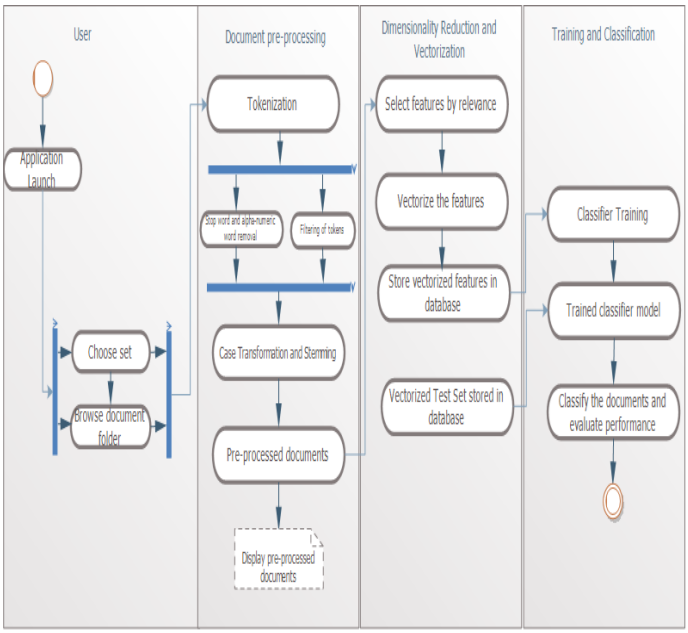

Fig. 4. Activity diagram of the developed model.

V. EXPERIMENTS AND RESULTS

Experiments were conducted for text classification on the categories of legal case documents to examine the performance of the proposed method. The legal case documents obtained from online repositories were renamed to shorter nomenclatures, eliminating the lengthy nature of the raw legal case names before the pre-processing stages. For example a land case with name “DR. RASAKI OSHODI & ORS V. YISA OSENI EYIFUNMI & ANOR” was renamed as “land_case_9”. The dimensionality reduction and vectorization stages by selecting the training/test set was performed by selecting the pre-process button as can be seen in Fig. 5a, and results obtained were stored in the MySQL database as can be seen in the XAMPP server interface in fig. 5b.

The pre-processing process was handled in 5 stages. The first step was the tokenization step involving breaking the contents of the legal cases into tokens. The stop words and alphanumeric characters were eliminated from the corpus when passed through the list of 571 stop words and 18 alphanumeric characters in the second stage. The third and fourth step involved the filtering out of tokens with ignorable length (less than 3 characters), as noticed that they were of no importance to the corpus, and the case transformation also referred to as case folding was implemented to maintain lower case tokens respectively. The Porters algorithm was used for stemming in the last stage. The training set used had a total of 57 documents from civil, land, criminal, finance, housing, politics_and_government categories consisting of 744624 features, but after performing pre-processing we were left with 269935 valid features. The test set was left with 61837 features after the pre-processing stage. Fig. 5b shows the output results of the features got from the individual cases in different categories after pre-processing.

After the pre-processing stage, feature ranking is handled by the chi (χ^2) statistics to select more features in the order of importance. A threshold value of 1.5 was selected to direct the rate of ranking as seen in Fig. 6a. This also contributed in the reduction of the high dimensionality of the feature space. The conversion of the resulting tokens into vectors was achieved using the Naïve Bayes probabilistic calculations, considering the evidence, prior probability and posterior probability of the tokens, this was done by selecting vectorize as also observed in the same figure.

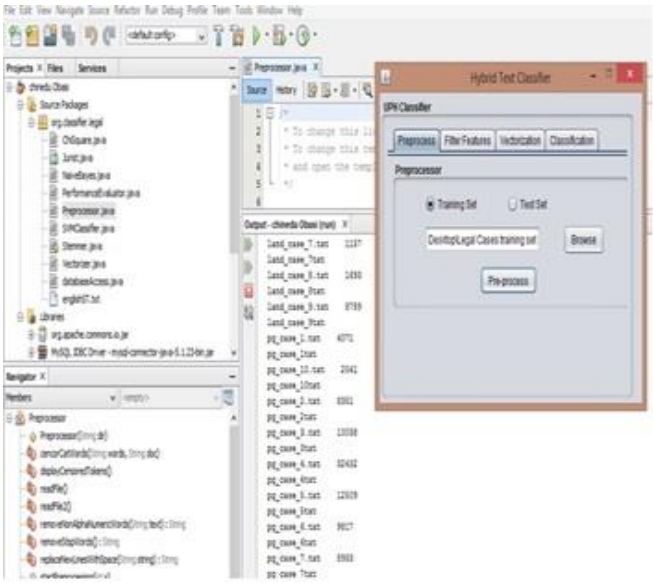

Fig. 5a. Application pre-processing stage.

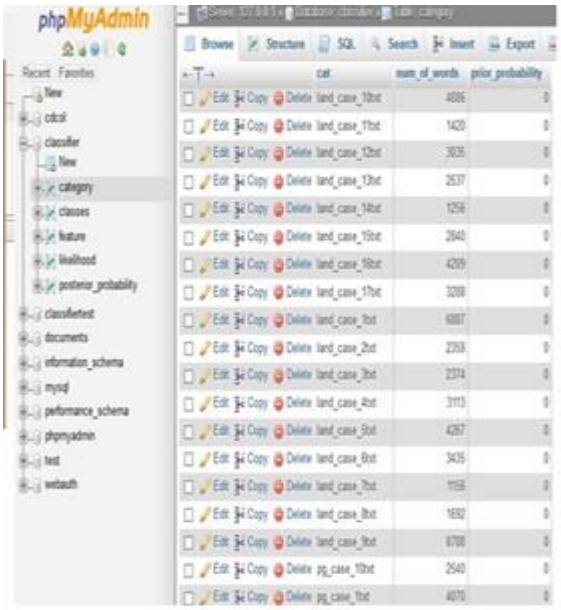

Fig. 5b. Application with pre-processing output results.

Fig. 6b also shows the use of Naïve Bayes method in the vectorization of the features, and the output results shows a clear effect of the filtering process on the individual cases as compared with the cases in Fig. 5b, For instance, land_case_4 had 3113 features before the filtering, and 2932 selected features for the ranking process, the individual prior probabilities as can be seen has been calculated from the Naïve Bayes probabilistic method.

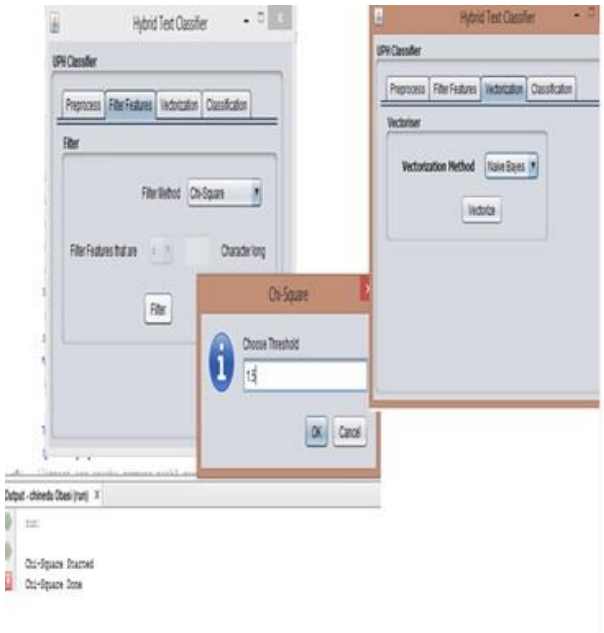

Fig. 6a. Hybrid application showing the chi-square and vectorization process.

After the calculation of the probabilities, the posterior probabilities of the tokens forms a multi-dimensional array consisting of the categories, the features and the probabilities, which are fed into the SVM as input parameters using the libSVM input format. The hybrid application makes use of re-usable libraries from the libSVM tool to implement the Support Vector Machine classification using different set values for different kernel functions.

The SVM picked the best training set vectors as its support vectors for building the model, while the test set vectors were classified according to the built model. The libSVM library implemented in the application was got from [6]. It has different kernels coupled to it. Two different kernel types was tested, but different parameters of the kernel types was tried to select the parameters that will give us the best results. The different kernels of the Support Vector Machine, polynomial and sigmoid was tested with the term vectors of the test set as against the vectors generated from the training set, and performance evaluation metrics like accuracy, precision and recall were used to check the performance of the classifier.

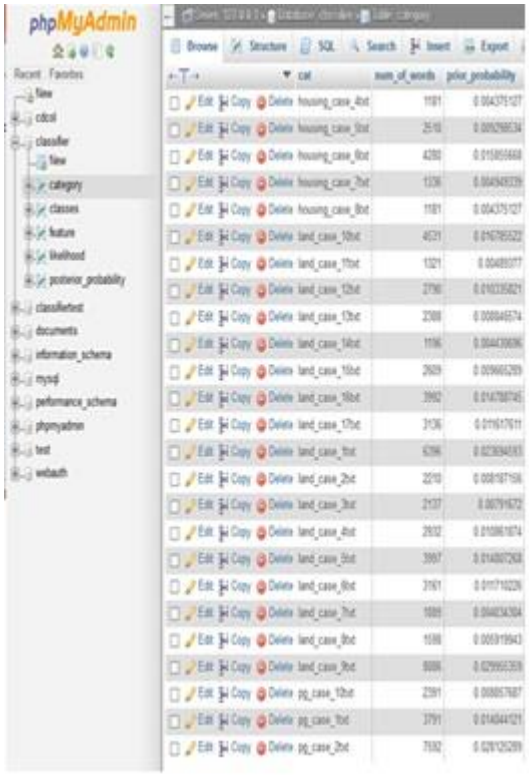

Fig. 6b. Hybrid application showing ranking and vectorization results.

The parameters set for use in the kernels are the C = cost function, the bigger the value of C, the bigger the points will be misclassified, so a smaller of C, say 0.5 for a start helps, the best accuracy value is got, and overfitting avoided, cache_size = cache memory size which is set in megabytes, Є = epsilon = used for setting the tolerance of termination criterion, d = degree = used for setting the degree in the kernel function, especially in the polynomial kernel, gamma = this is by default the reciprocal of the number of features.

The different kernels tested for the Support Vector Machine are the polynomial kernel and the sigmoid kernel. The polynomial kernel parameters that gave the best results used these values: gamma = 0, coef = 0, C = 6, cache size = 95, epsilon = 0.001. Finally the sigmoid kernels were set to these parameters, gamma = 0.065, C = 7.5, epsilon = 0.001, cache size = 80. The test documents was classified using the polynomial kernel as seen in fig. 7a by selecting classification type as SVM, the SVM type selected was C_SVC showing that it is the SVM classification type, and the kernel selected is POLY meaning the polynomial kernel and accuracy, precision and recall shows the results got from using the polynomial kernel. The sigmoid kernel was also used as seen in Fig. 7b as the kernel type selected is SIGMOID, and the results was also obtained. The documents were accurately classified in different categories, this can be observed from the true positive values and percentages got from the performance evaluators of the classifier model as seen in figures 7a and 7b.

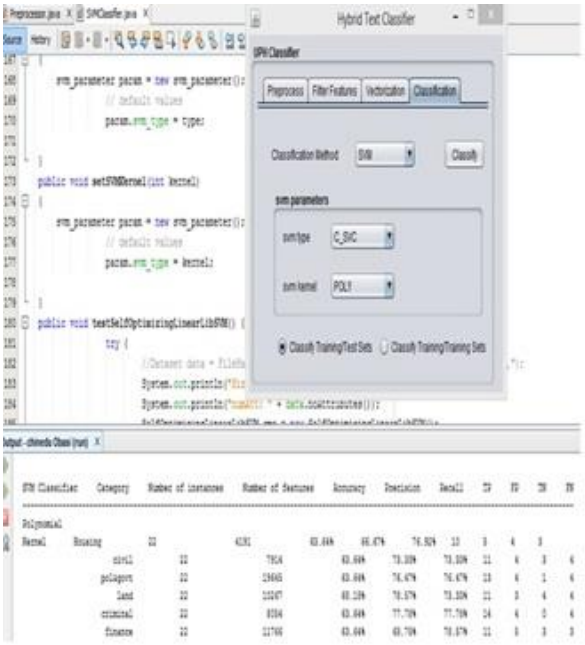

Fig. 7a. Application output results from the polynomial kernel.

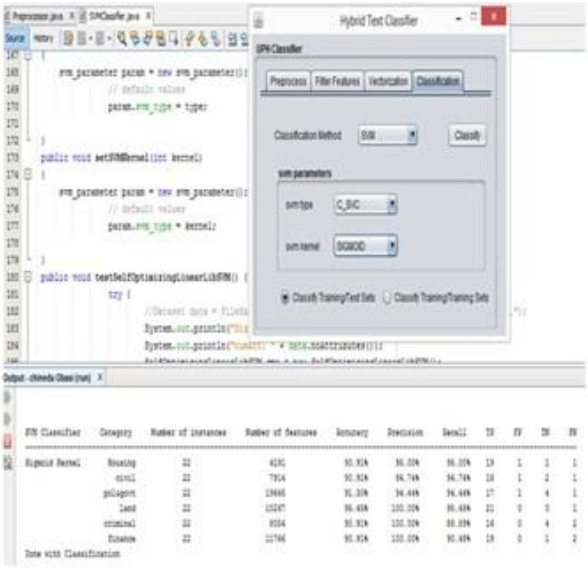

Fig. 7b. Application output results from the sigmoid kernel.

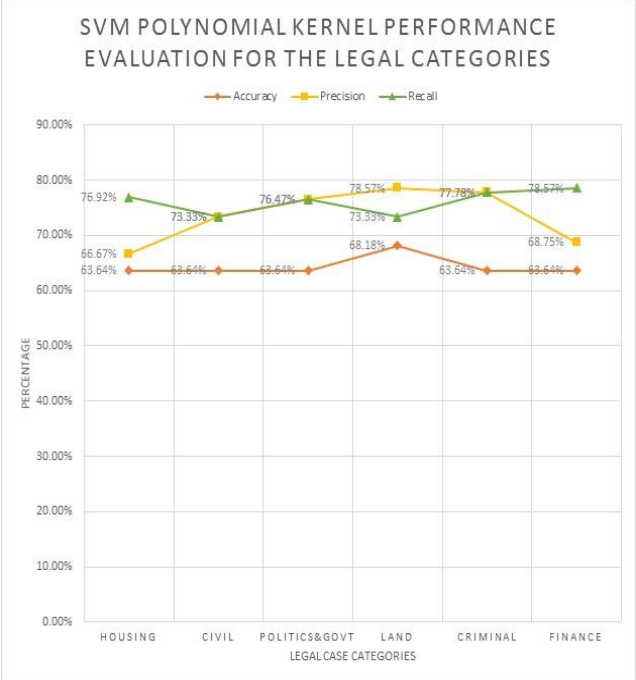

Fig. 8a. Performance of SVM polynomial kernel on the legal documents.

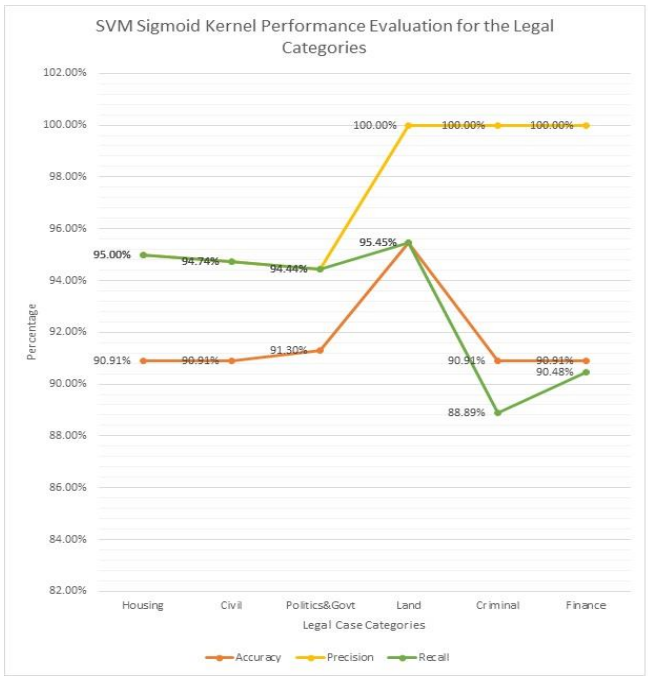

Fig. 8b. Performance of SVM sigmoid kernel on the legal documents.

VI. RESULTS DISCUSSION

Figure 5a shows the application pre-processing the features in the individual case documents, it can be seen that the number of features differ because of the different sizes of the cases and the level of noisy features eliminated. The tokens obtained after pre-processing were ranked using a threshold of 1.5 as seen in figure 6a to select the most important features. It also shows the use of the Naïve Bayes rule for generating term vectors in terms of probabilistic values. Fig. 6b shows the ranked and vectorization values as depicted in this case, the prior probability of the different cases used for the posterior probability calculations.

Fig. 7a shows the results obtained from the different legal case categories when the application is used to classify the documents with the polynomial kernel type, while Fig. 7b shows the results of the application with the sigmoid kernel type. The percentages shows the degree of accuracy, precision and recall in each of the categories.

Fig. 8a shows a graph of the evaluation performance against the legal case documents when the polynomial kernel was used, the Housing category for instance, shows a recall of 76.92% as calculated from the output of the True positive, False Negative and False positive values and it shows the percentage of correctly classified documents when compared to the test documents. 10 documents were accurately classified, 5 were false positives, 4 were true negatives and 3 were false negatives. Fig. 8b shows a graph of the evaluation metrics percentage over the legal categories as the sigmoid kernel of the SVM was used. It can also be seen that the Land category has higher precision in this figure, this was because of the greater availability of more important features used in classification of the category. A precision percentage of 100% seen in the land, criminal and criminal categories shows that there were no false positives and that the proposed documents in these categories were all classified in the right category as against 94.74% in the civil category where we had 1 false positive, this shows that few documents did not accurately classify, but the results showed great accuracy as 18 documents were accurately classified out of the 22 documents used.

VII. CONCLUSION

In this work, a supervised hybrid text classification approach using effective pre-processing techniques such as tokenization, stop words and alpha-numerical words removal, token filtering and stemming was used in the selection of the best features from the set of noisy raw documents; feature ranking using chi (χ^2) statistical method to further select documents of most importance, vectorization using Naïve Bayes to create word vectors suitable for text classification and classification with the Support Vector Machine was implemented to automatically classify legal case documents for a case management system and performance evaluation metrics like accuracy, precision and recall was used to prove the effectiveness of the classifier.

REFERENCES

[1] C.K. Obasi and C. Ugwu, "Feature Selection and Vectorization in Legal Case Documents Using Chi-Square Statistical Analysis and Naïve Bayes Approaches", IOSR Journal of Computer Engineering, vol. 17, Issue 2, pp. 42-50, April 2015.

[2] M. Duoqian, D. Qiguo, Z. Hongyun, and J. Na, "Rough Set Based Hybrid Algorithm for Text Classification", Proceedings from Journal of Expert Systems with Applications, vol. 36, No. 5, pp. 9168 – 9174, 2009.

[3] C. Hao, J. Wen, and L. Canbing, L. Rui, "A Heuristic Feature Selection Approach for Text Categorization by using Chaos Optimization and

Genetic Algorithm", Mathematical Problems in Engineering, Hindawi Publishing Corporation, vol. 2013, No. 524017, 2013, pp. 1-6.

[4] U. Harun, "A Hybrid Approach for Text Categorization by using χ^2 statistics, Principal Component Analysis and Particle Swarm Optimization", Proceedings from Academic Journals Scientific Research and Essays, vol.8, Issue 37, pp. 1818 – 1828, 2013.

[5] Y. Hong, S. Kwong, Y. Chang, and Q. Ren, "Consensus Unsupervised Feature Ranking from Multiple Views", Patt. Rec. Letters, Issue 29, 2008, pp. 595-602.

[6] http://www.csie.ntu.edu.tw/~cjlin/libsvm, Retrieved information on 22nd September 2014.

[7] http://www.lawreportsofcourtsofnigeria.org, Retrieved information on 27th July, 2014.

[8] http://www.lawaspire.com.ng, Retrieved information on 27th July, 2014.

[9] http://members.unine.ch/jacques.savoy/clef/englishST.txt, Retrieved information on 27th July, 2014.

[10] D. Isa, L.H. Lee, V.P. Kallimani, and R. RajKumar, "Text Documents Preprocessing with the Bayes Formula for Classification using the Support Vector Machine, IEEE, Traction of Knowledge and Data Engineering, vol. 20, Issue 9, pp. 1264-1272, 2008.

[11] V. Julio, C. Sonia, L. Sara, and C.G Jose, "Hybrid Approach Combining Machine Learning and a Rule-based Expert System for Text Categorisation", Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference, pp. 323 – 328, 2011.

[12] Y. Li, D.F. Hsu, and S.M. Chung, "Combining Multiple Feature Selection Methods for Text Categorization by Using Rank-Score Characteristics", Proceedings from the 21st IEEE International Conference on Tools with Artificial Intelligence, pp. 508-517, 2009.

[13] L. Liu, J. Kang, J. Yu, and Z. Wang, "A Comparative Study on Unsupervised Feature Selection Methods for Text Clustering", Proceedings of NLP-KE'05, pp. 597-601, 2005.

[14] M.F. Porter, An Algorithm for Suffix Stripping, Program (Auto. Lib. Info. Syst.) 14(3), 1980, pp. 130-137.

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

[16] G. Serkan, "Hybrid Feature Selection for Text Classification", Proceedings from Journal of Turk J Electrical Engineering and Computer Science, vol. 20, No. 2, pp. 1296 – 1311, 2012.

[17] L. Xiaoyong and F. Hui, "A Hybrid Algorithm for Text Classification Problem", PRZEGLĄD ELEKTROTECHNICZNY (Electrical Review), Guangdong Polytechnic Normal University, pp. 8 – 11, 2012.

[18] B. Yu and Z. Xu, "A Comparative Study for Content-based Dynamic Spam Classification using Four Machine Algorithms", Proceedings from Knowledge Based Systems, Elsevier Journal, vol. 21,pp. 355-362, 2008.

[19] H. Zhang, W. Zhihua, Z. Zhifei, W. Li, and M. Duoqian, "A Naïve Bayesian Multi-label Classification Algorithm with Application to Visualize Text Search Results", Proceedings from the International Journal of Advanced Intelligence, vol. 3, No. 2, pp. 173-188, 2011.

[20] Z. Zheng, R. Srihari, and S. Srihari, "A Feature Selection Framework for Text Filtering", Proceedings of the third IEEE International Conference on Data Mining, pp. 705-708, 2003.