BOOKSHELF – A DOCUMENT CATEGORIZATION FOR LIBRARY USING TEXT MINING

Carlo Petalver, Roderick Bandalan and Gregg Victor Gabison

Graduate School of Computer Studies,
University of San Jose – Recoletos, Cebu City, Philippines

ABSTRACT

Categorizing books and other archaic paper sources to a course reference or syllabus is a challenge in library science. The traditional way of categorization is manually done by professionals and the process of seeking and retrieving information can be frustrating. It needs intellectual tasks and conceptual analysis of a human effort to recognize similarities of items in determining the subject to the correct category. Unlike the traditional categorization process, the author implemented the concept of automatic document categorization for libraries using text mining. The project involves the creation of a web app and mobile app. This can be accomplished through the use of a supervised machine learning classification model using the Support Vector Machine algorithm that can predict the given category of data from the book or other archaic paper sources to the course syllabus they belong to.

KEYWORDS

Text Mining, Document Categorization, Classification algorithm, Support Vector Machine, Library.

1. INTRODUCTION

Categorizing and associating books and other archaic paper sources with a course reference or syllabus needs a lot of parameters to be considered: what is it all about? With what course reference or syllabus is associated? Early studies of document categorization in libraries need intellectual tasks and conceptual analysis of an item. In this, there are a lot of parameters to be considered in predicting a document.

Books that are related to the syllabus will serve as the basis of the Librarians in categorization tasks. Librarians may classify books that they may think will fit the course reference or syllabus based on titles, contents, and chapter headings where the human assignment was used. Considering these parameters in categorizing documents needs a serious conceptual analysis of human ability. On the other hand, some parameters are not sufficient to justify the target label of the documents. Some titles and subject headings can be present across categories or an absence from all categories that may lead to confusion and misclassification.

To address such gaps, the study aims to:

- Build a classification model using Support Vector Machine

David C. Wyld et al. (Eds): NLPA, AIS, BDAP, SOENG, IPPR - 2022
pp. 21-32, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121403

- Develop a system that can provide automatic categorization of documents to course reference or syllabus using mobile and web technologies
- Generate a report of classified books

The study focused on the table of contents and index pages only since it carries the significant components of the document (topics, chapters, and sections) relevant for text categorization. This can provide a cost-effective solution to human classifiers in classifying the books to what course syllabus they are assigned for easy reference, efficiency, and improved accuracy.

2. RELATED STUDIES

There are several studies regarding document categorization. Text analysis is an emerging field of study. Fields such as Academia, Marketing, and Governance are already leveraging the process of analyzing and extracting information from textual data.

A notable study of Text Classification is the 'Automatic Categorization of Tagalog Documents Using Support Vector Machines' by April Dae C. Bation, Erlyn Q. Manguilimotan, and Aileen Joan O. Vicente. In this study, the authors created an automated machine learning document classifier suitable for Tagalog documents. The document used was a news article from the Tagalog News Portal. These documents were manually categorized and later subjected to preprocessing techniques such as stem and stopword removal. They used a variety of document representations to find out which of the classifiers performed the best. An SVM classifier using a master dataset represented by a TF-IDF value provided an F-score of 91.99% and overall accuracy of 92%. This surpassed all other combinations of document representations and classifiers.

In the study of Soumick Chatterjee, Pramod George Jose, and Debabrata Datta, SVMs have been used in linear kernels that use the OneVsRest strategy for text classification with multithreaded and CUDA-enhanced SVMs. SVMs are trained on different datasets collected from different sources. Certain words that were less common about 56 years ago may now be widely used due to new trends. Similarly, discoveries can lead to the coining of new words. This technique can also be applied to text blogs that can be crawled and analyzed. This technique should theoretically be able to classify blogs, tweets, or other documents with high accuracy. The pre-processing phase (cleanup, stemming, lemming, etc.) is the longest in any text classification process. Therefore, the author took a multithreaded approach to speed up the process. SVMs have been used in linear kernels that use the OneVs Rest strategy for text classification with multithreaded and CUDA-enhanced SVMs.

Bernhard Scholkopf (the "support vector machine" for IEEE intelligent systems and their applications), the implementation overview, SVM's specific advantages over other learning algorithms to be theoretically analyzed using the concepts of computational learning theory when applied to actual or real problems. Examples of these real applications are provided by Sue Dumais, who explains the text classification problem above and offers the best results ever in the Reuters collection, and Edgar Osuna, who shows powerful results when applied to facial recognition. The fourth author, John Platt, provides practical guides and new techniques for efficiently implementing algorithms.

Lipo Wang, states that SVMs have a solid mathematical structure in statistical learning theory and has a good performance in many real-world applications such as bioinformatics, text mining, facial recognition, and image processing. We have established SVM as one of the most advanced

tools. Machine learning and data mining, and other soft computing technologies (eg B. Neural networks and fuzzy systems)

Roy T. Fielding in his doctoral dissertation, typical Representational State Transfer (REST) is described as an important architectural principle of the World Wide Web, which has received a lot of attention. The majority use REST as an application programming interface (API) for communicating between mobile and the web, as an approach to web service development commonly known as RESTful web services, and as an alternative to other distributed computing specifications.

It is in the light of these theories and related literature that inspires the researcher to develop a 'BookShelf: A Document Categorization for Library using Text Mining' to ease the work of librarians in categorizing or classifying books to the course syllabus they are assigned for easy reference. The above-related studies gave the researcher the idea that the proposed project could be possibly done using the Support Vector Machine (SVM) Algorithm, RESTful Web Services, and different tools and innovations.

3. METHODOLOGY

The systematic software development process of 'BookShelf: A Document Categorization for Library using Text Mining' is described in this section.

Below is the conceptual diagram of BookShelf to explain the process.

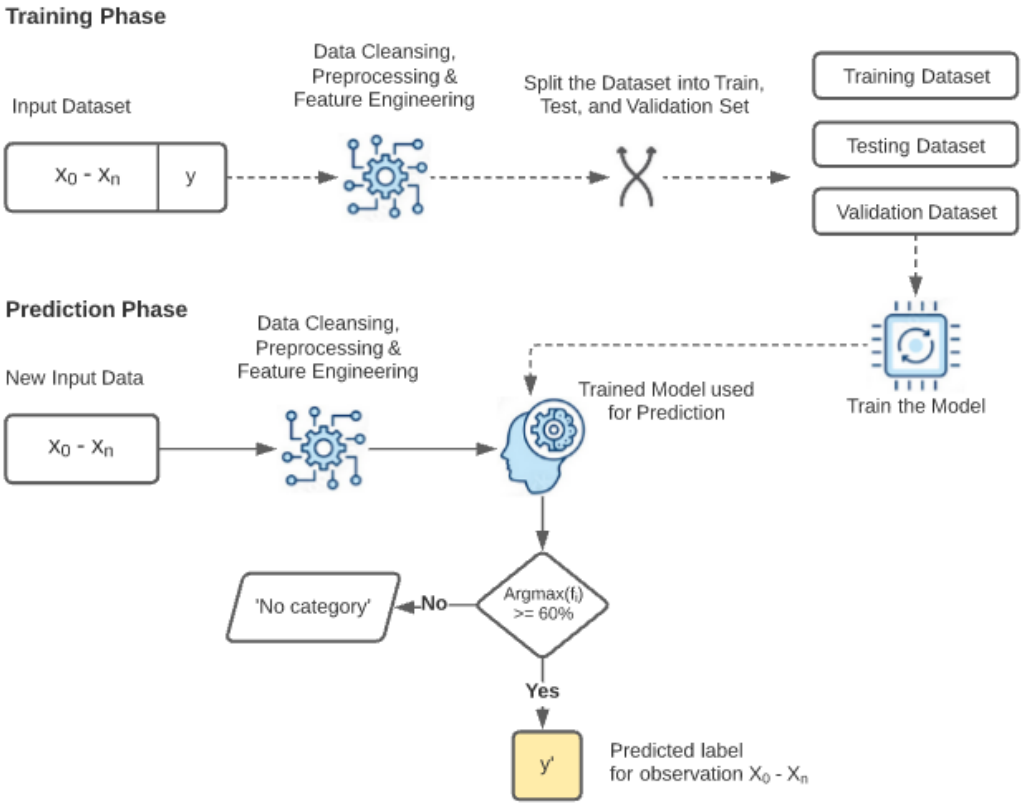

Figure 1. Conceptual diagram for *BookShelf – A document Categorization for Library using Text Mining*

The conceptual diagram shows the detailed process starting from the input, processes, and output of the system. The first step of this process is to create the model that will serve as the classifier in predicting the input data. Input Dataset records, which are text extracted from the table of contents and index pages of specific course references or related books in the curriculum are stored in a database that undergoes data cleansing, pre-processing, and feature engineering techniques. The model is created using the OnevsRest approach of the SVM algorithm.

The *New Input Data* is the texts extracted from the book's table of contents using OCR from the mobile application. The input data will undergo data cleansing, text preprocessing, and feature engineering phase as well. The input data is predicted by the classifier or model created. The predicted document can be assigned to a class or will be predicted as no category. This can be achieved by using a likelihood threshold such as 60%. Maximum values above the threshold are assigned to a class or predicted to be "no category".

3.1. Training phase

This part of the section explains how the model is created using the SVM (Support Vector Machine) algorithm and the components of training, evaluating, and building a model on a supervised approach and using it to classify the input documents. The process being used is through gathering content or text (table of contents and index pages) from the books which is used in training and building the model. The data that was gathered for the creation of the dataset came from the book's table of contents and index pages relevant to the specific course reference or syllabus. We considered five (5) relevant books per course reference or syllabus of different authors for each category in training the model in this project. The texts from the books table of contents and index pages are extracted or pulled out using OCR technology from a mobile app and saved to the database.

During the creation of the model and the prediction of the input data, pre-processing of text and feature engineering techniques were applied as discussed in the next subsections.

3.1.1. Text Processing and Feature Engineering

This includes text cleaning, tokenization, removal of stop words, lemmatization, N-gram, and TF-IDF (term frequency-inverse document frequency). Below are pre-processing and feature engineering techniques that were applied during the creation of the model and the prediction of input data.

3.1.2. Text Cleaning

This phase applies the removal of unwanted characters, some special characters, numbers, and punctuations present in the dataset, and input data using the regular expression. Converting all text to lowercase is also applied in this process.

3.1.3. Tokenization

In the pre-processing phase, it is the next step wherein the text or word sequences are converted into tokens. This process removes the whitespaces and other special characters or punctuations such as {([\t{}():;.])} from the text or document. Each word sequence is converted into tokens. You need to distinguish between the words that make up the string of characters. This is important because you can easily interpret the meaning of the text by analyzing the words that are present in the text. This is also very important in the next step of the process which is the lemmatization.

3.1.4. Lemmatization

The next step is the lemmatization process. This process groups together the inflected forms of a word so that they can be analyzed as a single element (e.g. ‘pointers’ to ‘pointer’, ‘lists’ to ‘list’, etc.). It takes the morphological analysis of the words. Lemmatization can be done after the tokenization process.

3.1.5. Stop Words Removal

The next step is the stop words removal. This process removes auxiliary verbs and prepositions. These words are considered to be unnecessary words that are less informative and are filtered out before processing the dataset and input data. Stop words include *to, at, as, an, a what, where, on, that,* and many others. Words such as (Bibliography, Chapter, Introduction, Contents, Exam, Quiz, Preface, and many others) were also removed since they carry less or no meaning to a document by adding these words to the existing stop word library.

3.1.6. N-gram

The next step is the N-gram process. The basic point of the N-gram is to understand the structure of the language from a statistical point of view, like what word is likely to follow a particular one. It helps capture important differences between the two documents. In this project, we set the parameters of *N-gram* to (1,2) unigram and bigram. It helps determine which n-grams can be chunked into single entities that have a significant impact on mining (such as a "binary tree" chunked as a single word, a "bubble sort" chunked as a single word, etc.). If the n-gram is too short, you may not be able to capture the significant differences. On the other hand, if it is too long, you may not get "general knowledge" and just stick to each case.

3.1.7. TF-IDF (Term Frequency – Inverse Document Frequency)

Before we can start training the dataset, it should be represented in the form of a vector. This is with the help of *TF-IDF* for *Term Frequency-Inverse Document Frequency* as the feature engineering algorithm. In this project, *TfidfVectorizer* has been used to transform text to feature vectors that can be used as input to the estimator.

Words which are present in all the documents for those words will be minimum and hence the total *TF-IDF* values will also come low for those documents. That is how *TF-IDF* tries to balance the unique words in the document that should highlight and come up. Those highlighted words are emphasized and given more weights during the prediction against the input data through which the algorithm/model learns from the pattern by looking at the features. Both training data and input data will undergo this process.

Here is a table with the list of some categories and the highlighted features or keywords from the dataset.

Table 1. *TF-IDF* result of some categories

Data Structures and Algorithms	**Networking**	**Systems Analysis and Design**	**Probability and Statistics**
TF-IDF queue 0.125196 abstract 0.118645 abstract data 0.116436 priority queue 0.112221 tree 0.106971 ... ... formal system 0.000000 formal standard 0.000000 formal argument 0.000000 formal approach 0.000000 zoned decimal 0.000000 25868 rows × 1 column …	TF-IDF basic architecture 0.102553 backbone 0.097059 performance improving 0.086984 improving 0.085592 reducing network 0.083072 ... ... Fragment 0.000000 fractional frame 0.000000 Fractional 0.000000 fra digital 0.000000 zoned decimal 0.000000 25868 rows × 1 column …	TF-IDF language leader 0.152392 concept language 0.148374 leader 0.136394 flow diagram 0.101193 use case 0.091861 ... ... frame margin 0.000000 frame friendly 0.000000 frame free 0.000000 frame frame 0.000000 zoned decimal 0.000000 25868 rows × 1 column …	TF-IDF population 0.104746 normal curve 0.099014 mean 0.094354 test 0.093292 curve 0.091680 ... ... formula random 0.000000 formula probability 0.000000 formula lazy 0.000000 formula grouped 0.000000 zoned decimal 0.000000 25868 rows × 1 column …

The table above shows the list of *TF-IDF* results of a few highlighted keywords for some categories.

3.1.8. SVM

All of the above steps are pre-processing steps that will lead us to use the dataset results for the *SVM* to process. In this project, we used *SVM* as the classifier with a linear kernel which means the data is linearly separable by a straight line. SVM works well even with a small size dataset which is suitable for this project. The *probability estimate* feature of SVM also is used in deciding to assign a class label. This can be achieved by using a probability threshold such as 60%, where the maximum argument value result is equal to or greater than the threshold is mapped to one class or otherwise predicted as 'no category' since some other archaic documents from the library have no relationship to the existing course reference. SVM outperforms the other classifiers in terms of prediction accuracy. This is explained in the next subsection.

3.1.9. Training, Evaluation, and Model Creation

When the dataset is ready, it will be evaluated and the process begins by splitting the dataset using the *train_test_split* of *sklearn* into training and test sets. 67% went to the training set while 33% went to the testing set. We train the model on the training set and evaluate the model on the test set until the best result is obtained and save the model.

Classifier Performance

Table 2. SVM OneVsRestClassifier Performance

Category	Precision	Recall	F1-Score
Fundamentals of Database Systems	0.75	1.00	0.86
Data Communications and Networking	1.00	1.00	1.00
Data Structures and Algorithms	1.00	0.75	0.86
Systems Analysis and Design	1.00	0.60	0.75
Probability and Statistics	0.33	1.00	0.50

Accuracy: 0.82
Macro avg: 0.82 0.87 0.79
Weighted avg: 0.92 0.82 0.84

In the above information, the performance of the model has met 0.823529411765 accuracy. It is expected for this case since we only have a small dataset during the training. Another factor for this is the fine-tuning of SVM parameters such as setting the random_state value during the *train_test_split* process. Linear SVM performs well in text classification tasks.

The classifier performance yields a good result with a high F1-Score after testing it with our dataset with a small number of samples per category. Here is the confusion matrix that depicts the accuracy of each category during prediction.
Confusion Matrix

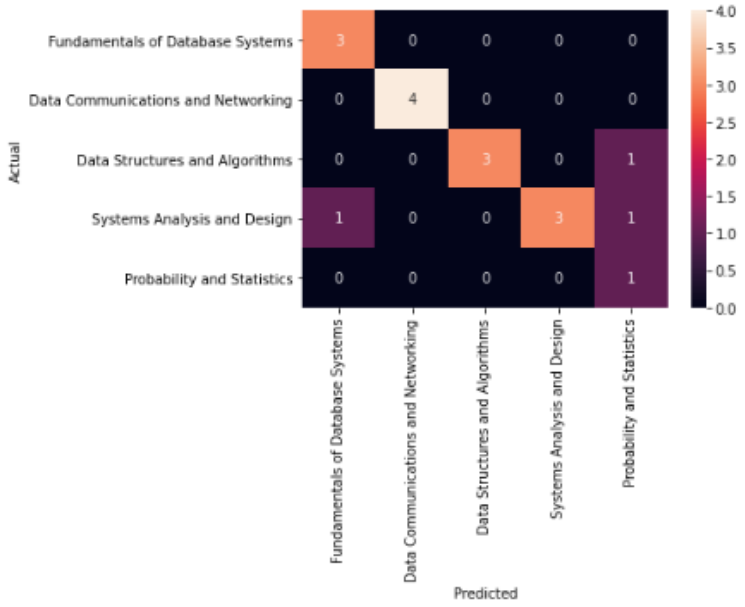

Figure 2. Confusion Matrix result for BookShelf

The above figure depicts some misclassification during the training and testing process and it is only minimal. In some cases, it is expected since both categories might have shared most of the common features.

3.1.10. Cross-Validation

Here is the accuracy result of SVM using the OneVsRestClassifier compared to other classifiers after evaluating them using the cross-validation test with our dataset. Here we achieved a *mean accuracy score* of 0.86 after evaluating the *cross_val_score* with 20% of the dataset in each fold, which is a decent classifier.

Table 3. Model performance using 5-fold cross-validation

Model name	**Accuracy**
MultinomiallNB	0.80
SVM OneVsRestClassifier	0.86
RandomForestClassifier	0.72

As you can see in the above data, the OneVsRestClassifier of SVM has achieved the highest score and outperforms other classifiers during the evaluation using the 5-fold cross-validation method.

Cross-validation summary

Table 4. 5-fold cross-validation summary

Model name	**Fold_idx**	**accuracy**
SVM OneVsRestClassifier	0	1.0
SVM OneVsRestClassifier	1	1.0
SVM OneVsRestClassifier	2	1.0
SVM OneVsRestClassifier	3	0.8
SVM OneVsRestClassifier	4	0.5
RandomForestClassifier	0	1.0
RandomForestClassifier	1	1.0
RandomForestClassifier	2	1.0
RandomForestClassifier	3	0.4
RandomForestClassifier	4	0.2
MultinomialNB	0	1.0
MultinomialNB	1	1.0
MultinomialNB	2	1.0
MultinomialNB	3	0.7
MultinomialNB	4	0.3

The validation technique using the 5-fold cross-validation was used to validate the performance of the SVM OneVsRestClassifier. Training and testing were repeated 5 times on stratified folds for the whole dataset.

3.2. Prediction Phase

The figure below describes the process of predicting input data.

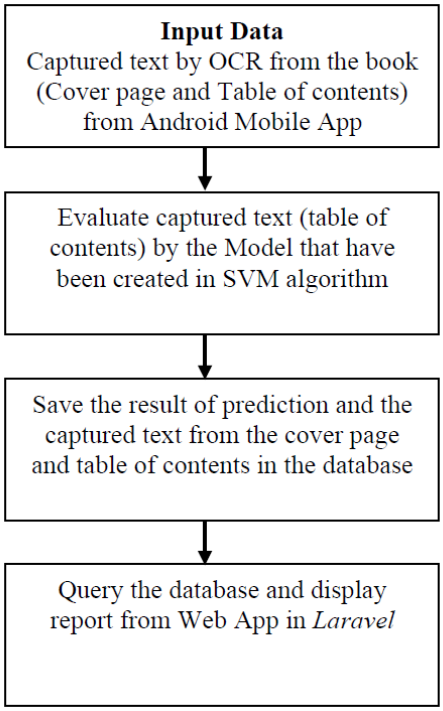

Figure 3. Shows the process of evaluating test files and generating the report

The input data is predicted by the model that has been created using the SVM algorithm. Each document can be assigned to a class or no category. This can be achieved by using a probability threshold, such as 60%, where the highest value equal to or greater than the threshold is mapped to one class or otherwise predicted as 'no category'. The result of the prediction and the text extracted from the cover page, table of contents, or index pages are saved in the database. Data from the database are queried and displayed as the summary or report of classified books in the Web app in *Laravel*.

Here is the prediction result of the input data with probability estimates of each class.

```
2021-08-24 14:27:11,014 INFO api Thread-2 : Prediction result
2021-08-24 14:27:11,014 INFO api Thread-2 : 4
2021-08-24 14:27:11,015 INFO api Thread-2 : Probability estimate
2021-08-24 14:27:11,015 INFO api Thread-2 : 0.89
2021-08-24 14:27:11,015 INFO api Thread-2 : Probability estimate of each
class
2021-08-24 14:27:11,015 INFO api Thread-2 : [0.00685022 0.01537133
0.89467403 0.03967454 0.04342988]
2021-08-24 14:27:11,076 INFO werkzeug Thread-2 : 192.168.43.1 - -
[24/Aug/2021 14:27:11] "#[37mPOST /endpointSVM HTTP/1.1#[0m" 200 -
```

Figure 4. Prediction result of input data

In the above figure, the input data or input document achieved a probability estimate score of 0.89 and is correctly assigned to a category which is Data Structures and Algorithms.

Here is the TF-IDF result of the unseen test input data coming from the *Data Structures and Algorithms* book.

```
2021-08-24 14:27:10,995 INFO api Thread-2 : Tfidf weights
2021-08-24 14:27:10,996 INFO api Thread-2 :
tree            0.125242
programming     0.106337
case            0.104358
assignment      0.102233
library         0.099937
...                  ...
family          0.028677
family tree     0.028677
fcd             0.028677
fcd algorithm   0.028677
xi programming  0.028677
```

Figure 5. TF-IDF result of test input data

Figure 6 is the result of the TF-IDF of the test input data. Since most of the highlighted keywords from the trained dataset are also present in the input data, then the input data is assigned with the label of that category by the classifier. Notice the probability estimate of other classes which is very low for other categories. This is where the probability estimate feature of SVM comes effective in making the decision and classification.

4. CONCLUSION AND RECOMMENDATIONS

4.1. Conclusion

The categorization of books or other archaic documents to a specific target or class is affected by many factors. Such include the collection of relevant datasets or corpus, the feature engineering techniques, and applying regular expressions such as removing unnecessary words.

The selection of the algorithm classifier type suitable for the requirements of the project has a contribution also in the categorization task. The use of Support Vector Machine as the algorithm is very well suited for this project and related projects that do not need a large corpus or dataset. The observation and experimental results show that SVM achieved good performance on text categorization tasks compared to other existing classifiers significantly with the given dataset. The SVM classifier yielded an accuracy of 0.86% during the 5-fold cross-validation technique that outperforms other classifiers.

This mobile and web-based application software are intended to solve the problems of librarians in categorizing books and other archaic documents to the course reference or syllabus. It provides automatic categorization for efficient, effective, and improved accuracy.

4.2. Recommendations

The researcher highly recommends the use of Support Vector Machine (SVM) for the requirement of this project and other document categorization for multi-class classification tasks. A keen and careful approach to removing unnecessary words from the extracted text in the table

of contents and index pages may improve the performance of the model. Words or other languages aside from English may be considered as an account as well.

Lastly, future research should also be able to build a classifier that can classify or categorize the books or other archaic documents with more than one category (multi-label) instead of multi-class classification. Future research should also build a model using incremental online training that can incorporate updates from the new unseen data to the existing model's precision or learning accuracy if the dataset is too large to fit in the memory.

REFERENCES

[1] Victor Diogho Heuer De Carvalho, Ana Paula Cabral Seixas Costa (2022). Exploring Text Mining and Analytics for Applications In Public Security: An In-Depth Dive Into A Systematic Literature Review.Exploring Text Mining and Analytics for Applications in Public Security: An in-depth dive into a systematic literature review | Request PDF (researchgate.net)

[2] Billie S. Anderson (2021). Using Text Mining to glean insights from COVID-19 literature. Using text mining to glean insights from COVID-19 literature - Billie S Anderson, 2021 (sagepub.com)

[3] Rafael Ferreira-Mello, Máverick André, Anderson Pinheiro, Evandro Costa, Cristobal Romer (2019). Text Mining in education. Text mining in education - Ferreira-Mello - 2019 - WIREs Data Mining and Knowledge Discovery - Wiley Online Library

[4] April Dae Bation, Aileen Joan Vicente, Erlyn Manguilimotan (2017). Automatic Categorization of Tagalog Documents Using Support Vector Machines. https://aclanthology.org/Y17-1046.pdf

[5] Sunita Gopal, S. Raghav (2017). Automatic document retrieval using SVM machine learning. https://ieeexplore.ieee.org/document/8358501/authors#authors

[6] Vladimer B. Kobayashi, Stefan T. Mol, Hannah A. Berkers (2017). Text Mining in Organizational Research.https://journals.sagepub.com/doi/full/10.1177/1094428117722619

[7] Graciela H. Gonzalez, Tasnia Tahsin, Britton C. Goodale, Anna C. Greene, Casey S. Greene (2016). Recent Advances and Emerging Applications in Text and Data Mining for Biomedical Discovery.https://doi.org/10.1093/bib/bbv087

[8] Y. Zhang, M. Chen and L. Liu (2015). A review on text mining. https://ieeexplore.ieee.org/document/7339149

[9] Pooja S. Ikka (2015). Data Mining of Social Networks using Clustering Based-SVM. [PDF] DATA MINING OF SOCIAL NETWORKS USING CLUSTERING BASED-SVM | Semantic Scholar

[10] Mrs. Manisha Pravin Mali, Mohammad Atique (2014). Applications of Text Classification using Text Mining. https://www.researchgate.net/publication/287531225_Applications_of_Text_Classification_using_Text_Mining

[11] Mladeni D., Brank J., Grobelnik M. (2011) Document Classification. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_230

[12] [Neelima Guduru (2006). Text Mining with Support Vector Machines and Non-Negative Matrix Factorization Algorithms.Microsoft Word - Thesis-Neelima-Guduru.doc (uri.edu)

[13] M. Ikonomakis, S. Kotsiantis, V. Tampakas (2005). Text Classification Using Machine Learning techniques.(PDF) Text Classification Using Machine Learning Techniques (researchgate.net)

[14] A. Basu, C. Watters, and M. Shepherd (2002). Support Vector Machines for Text Categorization. Support vector machines for text categorization - System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on (dal.ca)

[15] Anuradha K. Bodile , Manali Kshirsagar. Text Mining in Radiology Reports by SVM Classifier. CiteSeerX — Text Mining in Radiology Reports by SVM Classifier (psu.edu)

AUTHOR

Petalver, Carlo D, graduate of Bachelor of Science in Information Computer Science and currently pursuing my Master in Information Technology at University of San Jose-Recoletos. I have an experience with computer languages, including PHP, Laravel, MySQL, Android, Java, and Python. I have had experience also with small projects both in industry and in the academe and at the same time, I have been teaching for several years in one of the private schools in Cebu City in the field of Information Technology.