THE 4 UNIVERSITIES DATASET

Abzal Yessengazy s2005252
Shu Jiang s1983583
Huiyang Li s1932989
Bingbing Yang s1902618

Abstract

The 4 Universities Data Set contains 8282 web pages that were manually classified into 7 classes each including web pages from 4 universities. Our objective was to conduct the data mining cycle that includes data pre-processing, exploratory data analysis, modelling and testing the machine learning classifier. The pre-processing includes html parsing, text extraction, followed by converting the text into bag-of-words and TF-IDF vectors. The training and testing data sets were compiled by using the data for 3 universities and one held-out university respectively. Naive Bayes, Logistic Regression, Random Forest and kNeighbors were used for classification task, while stratified k-fold cross validation and split validation set were used to avoid overfitting and to select hyperparameters of models. F1 accuracy score was used as a performance measure.

1 Introduction

We are dealing with supervised learning task, namely multi-class and single-label classification[1] of web-pages into categories. The 4 Universities Data Set contains html pages collected from computer science departments of various universities in January 1997 by the World Wide Knowledge Base (WebKb) project of the CMU text learning group. These 8,282 pages were manually pre-classified into seven classes: student, faculty, staff, department, course, project and other. For each class the data set contains pages from main four universities: Cornell, Texas, Washington, Wisconsin as well as 4,120 mixed pages from other universities.

Our aim is to conduct the data mining cycle, including data pre-processing, exploratory data analysis, feature selection and finally, depending on these steps, building appropriate machine learning classifiers to classify the texts into 7 classes.

1.1 Previous Work

Since late 1990'th numerous researches have been conducted to identify the most efficient ways of text classification as part of general Natural Language Processing (NLP) tasks. One of the early researches that inspired further works was *"Text Classification from Labeled and Unlabeled Documents using EM"*[1] that presented two extensions to the Naive Bayes algorithm using Expectation-Maximization procedure. The author achieved improvement by 30% in classification accuracy under two main conditions: (1) a weighting factor to modulate the contribution of the unlabeled data, and (2) the use of multiple mixture components per class. Besides, the author's thoughts on WebKB data preprocessing and leave-one-out cross validation played a certain guiding role for our experiments.

More recent publication in NLP that has relevance to our work was *"Text Classification Using Machine Learning Methods-A Survey"*[2]. In this research, authors deal with important text classification issues such as dimensionality reduction of large feature space, sparsity and propose ways of eliminating noisy and redundant features from the data set.

[1] the single-label problem is when instances categorized into precisely one class among more than two classes

The *"Linguistic Regularities in Continuous Space Word Representations"*[3] on the other hand, considers the vector-space word representations that are implicitly learned by the input-layer weights, which can efficiently capture syntactic and semantic regularities in language. For instance, authors claim that the male/female relationship can be automatically learnt using this technique and with the induced vector representations, "King - Man + Woman" can result in a vector very close to "Queen."

These and other papers presented in relevant sections of current report, inspired us to investigate the proposed methods more broadly and attempt adapting them to our specific case.

1.2 Relevance

For the last few decades, the importance of the automatic text classification algorithms has been growing due to increased information content such as blogs, online newsletters and social media. Despite the fact, that researchers in the text mining field have conducted numerous work in order to achieve enhancements in processes, there is still a room for further improvements. Particularly, one of the main challenges in building an efficient model remains the pre-processing and post handling of high-dimensional and sparse data. Precisely implemented pre-processing and reasonable feature selection along with proper dimensionality reduction are able to enhance the quality of learning, resulting in improved classifier performance and the reduced computational cost.

2 Data Pre-processing

The given html files are organized into a directory structure, one directory for each class. Each of these seven directories contain 5 subdirectories, one for each of the 4 universities and one for the miscellaneous pages.

In the data pre-processing stage, we transformed the raw data into a pandas data-frame utilizing step by step the html parsing, bag-of-words (BoW) and Term Frequency-Inverse Document Frequency (tf-idf) transformation methods. As a note, two above mentioned methods have the same goal but different representations. For the sake of the project, we deployed both BoW and tf-idf, in the following sections we describe them in detail.

2.1 HTML Parsing

HTML parsing is required to analyze web pages, extract meaningful information and convert it into the format to which machine learning algorithms can be applied.

For our case, we applied well-known tokenization[2] technique to extract textual content from all parts of raw HTML files, namely by using python's BeautifulSoup library. As a result, we have been able to extract text from all parts of html pages including headlines, body texts, paragraphs, and by further removing irrelevant characters, symbols and numbers we compiled the .txt files only containing meaningful English words.

Upon tokenizing the data, we have removed terms using a English stopwords list [3] and then performed stemming on the remaining terms, eventually getting a list of 54398 unique words. Surprisingly, later on we discovered that classifiers perform worse when we include the words that only appear once in entire corpus. This phenomenon was also confirmed in[1]. Hence, we took another approach and removed those terms, and kept only the terms occurring more than once. This time we ended up having a list of 32728 unique words.

2.2 Bag-Of-Words

BoW is widely used feature extraction procedure for sentences and documents in text classification tasks. We normally view a text as a sequence of words after tokenization. This gives a BoW, which simply counts how many times each word occurs in a page (or document). The simple principle of BoW is illustrated below:

[2]Tokenization is the act of breaking up a sequence of strings into pieces such as words, keywords, phrases, symbols and other elements called tokens.

[3]Stopwords list include English words such as: a, about, above, after, again, against,...,yours yourself, yourselves.

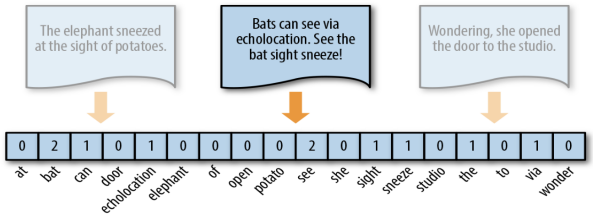


Figure 1: Principle of BoW Format

In order to obtain the BoW representation we used the Count Vectorizer method provided by SKLearn, which converts a collection of text documents into a matrix of token counts.

2.3 Term Frequency — Inverse Document Frequency

The higher occurrence of a word in a document leads to higher term frequency (TF) and the fewer occurrence of word in a document yields greater importance (IDF) of a word to the particular document. TF-IDF is the multiplication of term frequency (TF) and inverse document frequency (IDF).[4]

TF-IDF calculates weighted term frequency scores that represent relevance of words to particular classes. The higher the TFIDF score, the rarer the term across other pages.

In order to obtain the TF-IDF matrix we used the Tfidf Transformer method provided by SKLearn.

3 Exploratory data analysis

EDA is a process of understanding the data set by summarizing its main characteristics usually by visual illustrations. This step is very important especially for further proceeding with modeling phase.

We start EDA with understanding the general data structure and a sense of class labels distributions. The below Figure 2 illustrates the huge imbalance in class distribution in training data.

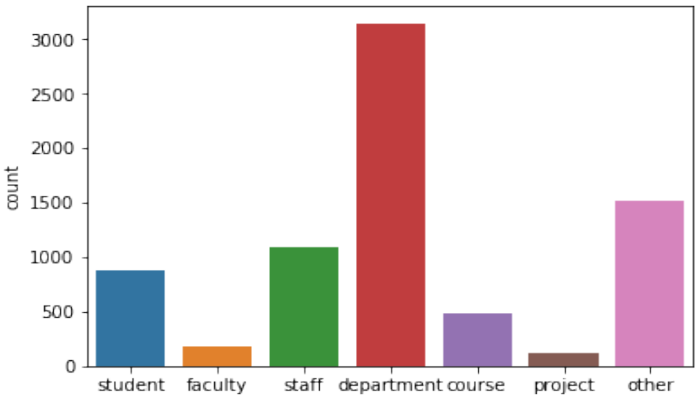
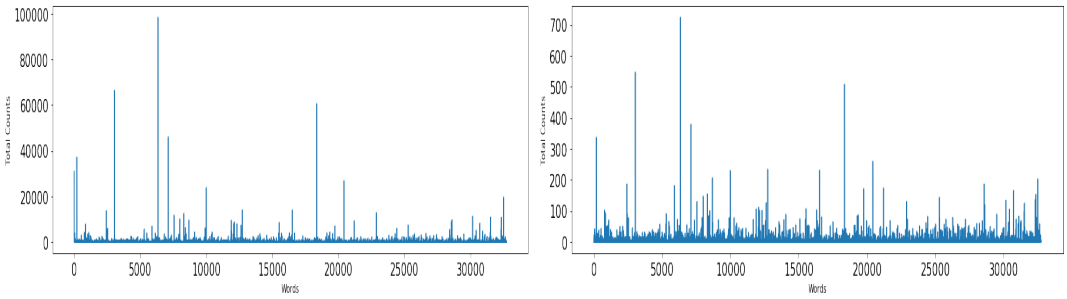


Figure 2: Class Label Distribution

The Figure 3 illustrates the TF's in BoW model and corresponding scores in TF-IDF representations. As you might notice, there is a huge difference in numbers, e.g. some words have TF of 1000, whereas other around 10, meaning that classifiers may suffer from biased frequent and weighty words and less focus on others.

Therefore, we used sklearn pre-processing method StandardScaler which normalizes the data by subtracting mean of training samples and dividing by its standard deviation.



(a) TF's for Each Word (b) TF-IDF scores for Each Word

Figure 3: Words TF's BoW and TF-IDF Scores

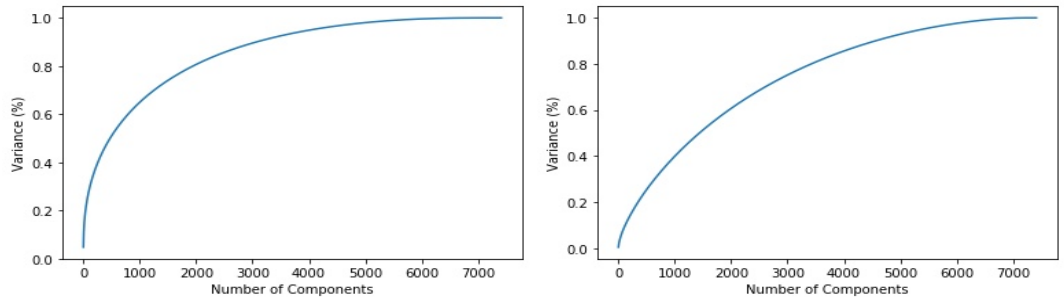
4 Dimensionality Reduction

Dimensionality reduction of the raw input data is an essential pre-processing step in the classification task. There are two main reasons to keep the dimensionality of features as small as possible: computational cost and classification accuracy.

4.1 Principal Component Analysis (PCA)

PCA is one of the most widely used methods for dimensionality reduction. Finding an orthogonal set of projection vectors for extracting the features is the ultimate goal of PCA which is done by maximizing the variance of data. It is an efficient technique, however considering PCA's unsupervised nature, it has several drawbacks while used in feature selection in classification task [5]. In the following sections we report the results of classifiers that confirm that this a case.

In order to find balance between proper representation of the data by threshold explained variance of 90% and efficient dimensionality reduction, we used the variance(%) vs Number of Components Plot as shown in Figure 4. We observe that 3000 and 5000 features for BoW and TF-IDF formats respectively explain overall the threshold of 90% of the total variance.



(a) BoW Dataset Explained Variance (b) TF-IDF Dataset Explained Variance

Figure 4: Explained Variance for BoW and TF-IDF Model

Therefore, we reduce our number of features down to 3000 for BoW model and 5000 for TF-IDF.

4.2 Mutual Information (MI)

Another efficient way of dimensionality reduction is using the MI - feature selection method. MI measures dependency between two variables. Specifically, it measures dependency between class labels and features, resulting in values in range from 0 to 1. It is equal to zero if and only if feature and class are independent, and higher values mean higher dependency.

Using the MI feature selection, we achieved reduction in vocabulary size of 3273 most informative words. This feature selection method is in practice lead to more robust performance of classifiers.

5 Modeling

5.1 Training and Testing

As the instructions suggest, we have used 3 universities and 1 miscellaneous html page samples to compile a training set and validation data for model tuning. Whereas, the 4-th university was used as a held-out unbiased testing set to evaluate true generalization performance of the classifiers.

5.1.1 Imbalanced Data and Overfitting

As mentioned in EDA section, main challenge with our data set is imbalanced class distribution. This may result in various misleading outcomes, mainly overfitting of the classifier. In general, algorithms aim to maximize accuracy rate, thus classifier may start learning the noise of the most frequent class in the training set and gain high accuracy, however, poorly generalizes on unseen data. Many researches propose special techniques to avoid overfitting of classifiers on imbalanced training data, such as oversampling the minority class, undersampling the majority sample or generating the synthetic sample. However, we decided to utilize robust performance measures such as K-fold Cross Validation and F1 score to overcome the class imbalance.

5.1.2 K-Fold Cross Validation and F1 Score

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.

It is a popular method because of simplicity, and in general results in a less biased and less optimistic estimate of the model performance than other methods, such as a simple train/test split.

The general procedure of CV is as follows:

- Shuffle the dataset randomly.
- Split the dataset into k groups

For each unique group:

- Take the group as a hold out or test data set
- Take the remaining groups as a training data set
- Fit a model on the training set and evaluate it on the test set
- Retain the evaluation score (F1) and discard the model
- Summarize the skill of the model using the sample of model evaluation scores

We used Stratified variation of K-Fold CV with k=5 folds. The folds are made by preserving the percentage of samples for each class which is the most efficient in our case considering not even class distribution, limited sample size (html pages) and larger feature size (words as features).

The F1 score can be interpreted as a weighted average of the precision and recall [4], where an F1 score reaches its best score at 1 and worst at 0. Using the *"micro"* parameter ensures we calculate metrics globally by counting the total true positives, false negatives and false positives, meaning it is robust for cases with unevenly distributed classes like ours.

5.2 Baseline Model

Considering our specific case and challenges described in 4.2, namely having one dominating class, motivated us to choose a simple "*Dummy Classifier*" as a baseline model with a parameter strategy

[4] Precision is defined as the number of true positives over the number of true positives plus the number of false positives. Recall is defined as the number of true positives over the number of true positives plus the number of false negatives.

"*most-frequent*" and used "*f1 score*" as a performance measure. The results are discussed in the result section of the report.

5.3 Naive Bayes

In spite of evolving text classification algorithms, majority of researchers still highlight the advantage of using Naive Bayes classifier in terms of simplicity, accuracy and computational efficiency. The paper [6] claims the algorithms as the most efficient in text classification tasks and provides with practical evidence comparing the results with other classifiers such Neural Nets and SVM. In our case, we namely make use of Multinomial Naive Bayes algorithm. MNB is ideally designed for text classification tasks with discrete features such as word counts, i.e. a good fit for using after BoW, however, several researches revealed that using MNB with fractional counts such as tf-idf may work efficiently as well[7]. Hence, we have experimented the usage of MNB with both BoW and TF-IDF and captured the results accordingly.

5.4 Random Forest

The second classifier that we used for task is RF, specifically we used an improved version as suggested by paper[8]. Particularly, the paper suggests a novel approach in feature weighting method for subspace sampling and tree selection method, so we can effectively reduce subspace size and improve classification performance without increasing error bound. The authors used the RF in combination with tf-idf, so our aim was to follow the same path.

Random Forest is a popular machine learning algorithm used for several types of classification tasks[9]. A Random Forest is an ensemble of tree-structured classifiers[10]. Every tree of the forest gives a unit vote, assigning each input to the most probable class label. It is a fast method, robust to noise and it is a successful ensemble which can identify non-linear patterns in the data. It can easily handle both numerical and categorical data. One of the major advantages of Random Forest is that it does not suffer from over fitting, even if more trees are appended to the forest.

Other advantages of RF to mention, it can process data of very high dimensions with many features and fast training speed.

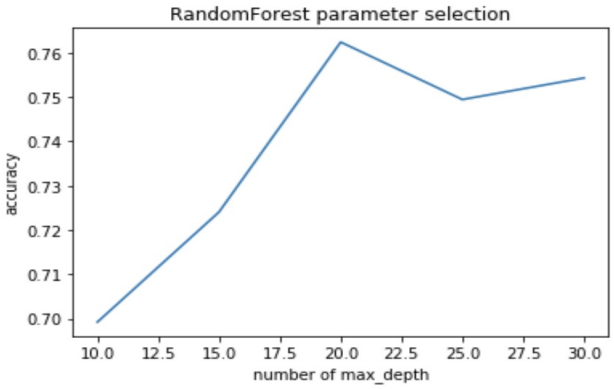


Figure 5: RF Parameter selection

5.5 Logistic Regression

Although LR is not a broadly utilized classification algorithm in text mining tasks, there are number of papers actively arguing with this fact. One of these articles is [11], where authors prove that the appropriate hyperparameter tuning might help to gain a good classification performance.

We used the LR with both multinomial and One-vs-Rest parameters, lbfgs solver and L1 regularization. The main advantage of LR is its less prone to over-fitting once regularization is used. On top of that, it is easy to implement and efficient in training. The main disadvantage, it assumes the linearity between class labels and features.

5.6 k-Nearest Neighbor

KNN is completely different algorithm from what we have seen so far, but considered as efficient. The difference from others, it does not require training time before making predictions, new data can be added seamlessly which will not impact the accuracy of the algorithm. The KNN algorithm assumes that similar classes located close to each other, thus calculates distance to k nearest neighbors in order to make predictions. One of the main drawbacks might be the ordering of the training data that can hugely affect the outcome, as well as the proper selection of number of k neighbors. We used the same cross-validation technique discussed earlier for parameter tuning. The results are discussed in the results section.

6 Results & Evaluation

The Table below illustrates the performance results of different approaches discussed in the previous sections:

Table 1: Classifiers' Performance Results

	MI+BoW	**MI+TFIDF**	**PCA+BoW**	**PCA+TFIDF**
Classifier	**F1 Score**	**F1 Score**	**F1 Score**	**F1 Score**
Dummy	0.42	0.42	0.42	0.49
Logistic Regression	0.78	0.80	0.78	0.76
Multinomial NB	0.65	0.65	0.63	0.62
Random Forest	0.75	0.74	0.56	0.54
KNeighbors	0.67	0.62	0.66	0.67

Results observation:

- Logistic Regression classifier has the highest accuracy rate because of the linearity dependence between features and labels. Particularly, using Mutual Information with TF-IDF matrix and running LR gave the highest F1 score of 80%
- In majority of cases Mutual Information method outperformed the usual PCA, meaning for imbalanced class distribution cases more accurate feature selection can be done using MI
- In all cases classifiers performed better than Baseline Dummy Model
- Random Forest classifier had huge performance difference when used with PCA and MI, hypothesis: since RF as DT in general rely on maximizing the information gain in node selection, and the MI fulfils that demand whereas PCA does not

From performances of all classifiers, we choose the Logistic Regression as the predictive classifier for test data (the Cornell university), and after transforming the test data into TF-IDF format, we use MI as the feature reduction method. We get the value of F1 score of 0.77, which shows that the classifier we choose performs well for separating these 7 labels, and the web data is well processed by MI and TF-IDF methods.

7 Conclusion

Key findings and highlights of the project are as follows:

- Not all well-known data processing methods similarly efficient to specific data sets. For instance, in our case, deploying the stemming and stopwords list during HTML parsing actually harmed the classifier performance.
- Both BoW or TF-IDF vector representations can be used, however the classifiers performance on these matricies vary depending on the data structure and characteristics.
- Proper use of feature selection and dimensionality reduction methods can both significantly reduce computational cost and accuracy.

- Using K-Fold Cross Validation in optimization and hyperparameter tuning of the model can greatly improve classifier's performance. And there is no common dimensionality reduction method and word vector representation that are suitable for all classifiers. We can only try different combinations to determine the best performing classifier.
- Random Forest classifier has huge performance difference when used with PCA and MI, hypothesis: since RF as DT in general rely on maximizing the information gain in node selection, and the MI fulfils that demand whereas PCA does not.

Finally, good topics for further research would be to explore more robust measures to dealing with imbalanced data distributions as well as the usage of large unlabeled data in order to gain higher performance of classifiers.

8 Contribution

All team members contributed equally to the project. Data pre-processing stage was done by LI Huiyang and he also supported throughout the entire coding part. JIANG Shu was responsible for experiments with BoW branch and PCA feature selection. YANG Bingbing focused on TF-IDF and MI feature selection. Abzal Yessengazy conducted EDA and model selection. All members contributed in reporting the findings.

References

[1] Kamal Nigam, Andrew Kachites Mccallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39:103–134, May 2000.

[2] Basant Agarwal and Namita Mittal. Text classification using machine learning methods-a survey. 236:701–709, February 2014.

[3] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. pages 746–751, January 2013.

[4] Shahzad Qaiser and Ramsha Ali. Text mining: Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181:25–29, July 2018.

[5] Shadvar Ali. Dimension reduction by mutual information feature extraction. *International Journal of Computer Science Information Technology*, 181:13–24, July 2012.

[6] S.L. Ting, W.H. Ip, and Albert Tsang. Is naïve bayes a good classifier for document classification? *International Journal of Software Engineering and its Applications*, 5:37–46, January 2011.

[7] Ashraf Kibriya, E. Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial naive bayes for text categorization revisited. *Advances in Artificial Intelligence*, pages 488–499, January 2004.

[8] Baoxun Xu, Xiufeng Guo, Yunming Ye, and Jiefeng Cheng. An improved random forest classifier for text categorization. *Journal of Computers*, 7:2913–2920, December 2012.

[9] David Cutler, Thomas Edwards, Karen Beard, Adele Cutler, Kyle Hess, Jacob Gibson, and Joshua Lawler. Random forests for classification in ecology. *Ecology*, 88:2783–2792, December 2007.

[10] Archana Chaudhary, Savita Kolhe, and Raj Kamal. An improved random forest classifier for multi-class classification. *Information Processing in Agriculture*, 3:215–222, September 2016.

[11] Alexander Genkin, David Lewis, and David Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49:291–304, August 2007.