Single-Label Text Categorization using Machine Learning

Wajahat Riaz  
*Department of Electrical Engineering*University of Engineering and Technology Lahore, Pakistan  
<mailto:wajahatriazmirza@gmail.com>

***Abstract***— **In this paper we present a comprehensive comparison of the performance of several text categorization methods in WebKB data sets. In particular, we evaluate Naive Bayes, SVM, Entropy-based Decision Tree, and Random Forest and compare their performance on the given data set. The paper also shows that the traditional TF-IDF term weighting approach remains very effective, even when compared to more recent approaches.**

Keywords—Text Categorization, Multi-class Classification, Logistic Regression, SVM, Random Forest, Stochastic Gradient Descent, Decision Trees, TF-IDF Scikit-learn

# Introduction

The main goal of text categorization (TC) is to derive models for the categorization of natural language text [1]. The objective is to derive models that, are given a set of training and test documents with known categories. We then train and evaluate by determining the category it belongs to, a problem called single-label text categorization. In these models, usually based on statistical analysis, a text document is represented as an n-dimensional vector of index terms or keywords. Each index term corresponds to a word in the initial text and has a weight associated with it, which should reﬂect how important this index term is, for that document and/or for the collection of documents. Usually, **TF-IDF** [1] term weighting is used and documents are normalized so that their length is one [2], [3]. In this paper, we are interested in ﬁnding out how relatively simple models behave, when compared with two well-known traditional TC models and with a state-of-the-art classiﬁer based on Support Vector Machines. We also analyze the effect that term weighting has on the accuracy of the obtained results.

# data set: web-kb

The documents in the WebKB are webpages collected by the World-Wide Knowledge Base (Web-Kb) project of the CMU text learning group [4].

These pages were collected from computer science departments of various universities in 1997 and manually classified into seven different classes: student, faculty, staff, department, course, project, and other. The class other is a collection of pages that were not deemed the ``main page'' representing an instance of the previous six classes. For example, a particular faculty member may be represented by a home page, a publications list, a vitae, and several research interests pages. Only the faculty member's home page was placed in the faculty class. The publications list, vitae, and research interests pages were all placed in the other category. For each class, the collection contains pages from four universities: Cornell, Texas, Washington, Wisconsin, and other miscellaneous pages collected from other universities. Some classes were discarded by Ana Cardoso such as Department and Staff because there were only a few pages from each university. She also discarded the class Other because the pages were very different in this class. Because there is no standard train/test split for this dataset, and to be consistent with the previous ones, she randomly chose two-thirds of the documents for training and the remaining third for testing. For this particular split, the distribution of documents per class is the following:

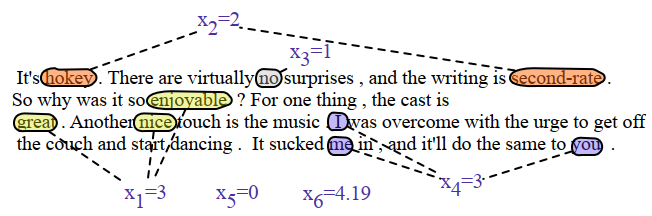
|  |  |  |  |
| --- | --- | --- | --- |
| **Classes** | **Number of Training Documents** | **Number of Testing Documents** | **Total Documents** |
| **Project** | 336 | 168 | 504 |
| **Course** | 620 | 310 | 930 |
| **Faculty** | 750 | 374 | 1124 |
| **Student** | 1097 | 544 | 1641 |
| **Total** | 2803 | 1396 | 4199 |

Table. 1. Classification Report for Linear SVC

# Methodology

Unstructured text data, like the contents of a book or a tweet, is both one of the most interesting sources of features and one of the most complex to handle. Different strategies for transforming text into information-rich features will be discussed. This is not to say that the recipes covered here are comprehensive. There exist entire academic disciplines focused on handling this and similar types of data, and the contents of all their techniques would fill a small library. Despite this, there are some commonly used techniques for this data set (WebKB), and a knowledge of these will add valuable tools to our pre-processing toolbox.

Firstly, we will unstructured the text documents by reading them and arranging the data in such a way that you have a column of classes and then another column containing text corresponding to the classes. An example is shown below:



This is the text corresponding to a particular class instance. We will process through different steps to make features for machine learning, discussed below:

1. We will strip the whitespace from each text.
2. Removing punctuations
3. Tokenizing text
4. Removing stop-words
5. Stemming words
6. Tagging parts of text instances and encoding them
7. Encoding text as a bag of words
8. Weighting word importance using **TF-IDF**
9. Splitting the data set for training and evaluation

# Experiment

After splitting the data set for training and evaluation. We pass it to our machine learning model. The five most common classifiers are used and they are implemented using Scikit-learn.

## Decision Trees (Entropy)

Tree-based learning algorithms are a broad and popular family of related non-parametric, supervised methods for both classification and regression. The basis of tree-based learners is the decision tree wherein a series of decision rules are chained. The result looks vaguely like an upside-down tree, with the first decision rule at the top and subsequent decision rules spreading out below. In a decision tree, every decision rule occurs at a decision node, with the rule creating branches leading to new nodes. A branch without a decision rule at the end is called a leaf. Decision tree learners attempt to find a decision rule that produces the greatest decrease in impurity at a node. While there are several measurements of impurity, entropy is used for impurity for this particular paper. As discussed previously, the reason for the popularity of tree-based models is their interpretability. Decision trees can be drawn out in their complete form to create a highly intuitive model. From this basic tree system comes a wide variety of extensions from random forests to stacking.

## Random Forest

A common problem with decision trees is that they tend to fit the training data too closely (i.e., overfitting). This has motivated the widespread use of an ensemble learning method called random forest. In a random forest, many decision trees are trained, but each tree only receives a bootstrapped sample of observations (i.e., a random sample of observations with a replacement that matches the original number of observations), and each node only considers a subset of features when determining the best split. This forest of randomized decision trees (hence the name) votes to determine the predicted class.

## Support Vector Machine (LinearSVC)

Support vector machines classify data by finding the hyperplane that maximizes the margin between the classes in the training data. In a two-dimensional example with two classes, we can think of a hyperplane as the widest straight “band” (i.e., line with margins) that separates the two classes. LinearSVC implements a simple SVM classifier. SVCs work well in high dimensions i.e. multi-class classification;

## Naïve-Bayes (Gaussian)

In naive Bayes, we compare an observation’s posterior values for each possible class. Specifically, because the marginal probability is constant across these comparisons, we compare the numerators of the posterior for each class. For each observation, the class with the greatest posterior numerator becomes the predicted class, ŷ. There are two important things to note about naive Bayes classifiers. First, for each

feature in the data, we have to assume the statistical distribution of the likelihood, P(xj | y). The common distributions are the normal (Gaussian), multinomial, and

Bernoulli distributions. The distribution chosen is often determined by the nature of features (continuous, binary, etc.). Second, naive Bayes gets its name because we assume that each feature, and its resulting likelihood, is independent. This “naive” assumption is frequently wrong, yet in practice does little to prevent building high-quality classifiers.

# Results and discussion

We train and evaluate our data five times and recorded classification reports and confusion matrices.

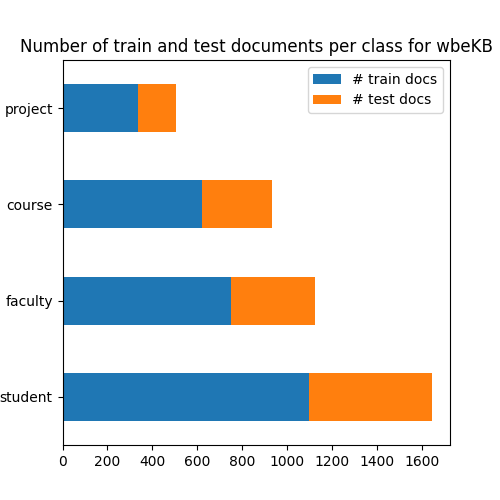
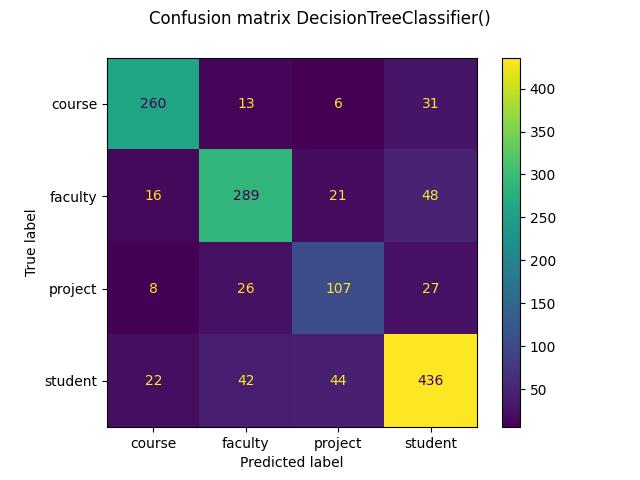
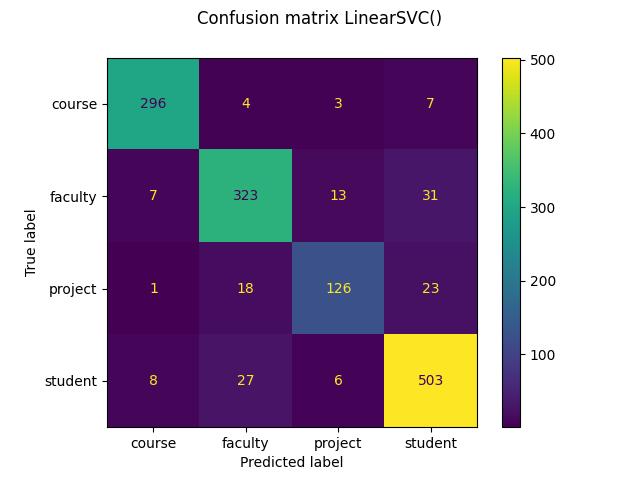


Fig. 1. Number of train and test documents per class



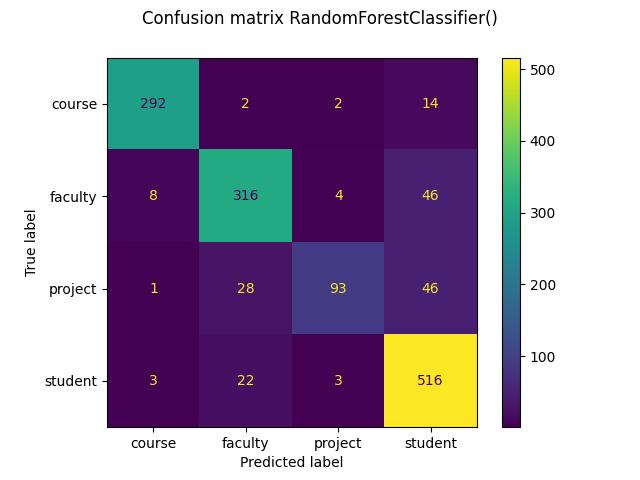
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-score** | **Support** |
| **1** | 0.85 | 0.84 | 0.84 | 310 |
| **2** | 0.78 | 0.77 | 0.78 | 374 |
| **3** | 0.60 | 0.64 | 0.62 | 168 |
| **4** | 0.80 | 0.80 | 0.80 | 544 |
| **Accuracy** | 0.78 | 0.78 | 0.78 | 0.78 |
| **Macro avg** | 0.76 | 0.76 | 0.76 | 1,396 |
| **Weighted avg** | 0.78 | 0.78 | 0.78 | 1,396 |

Fig. 2 Classification Report for Decision Tree Classifier



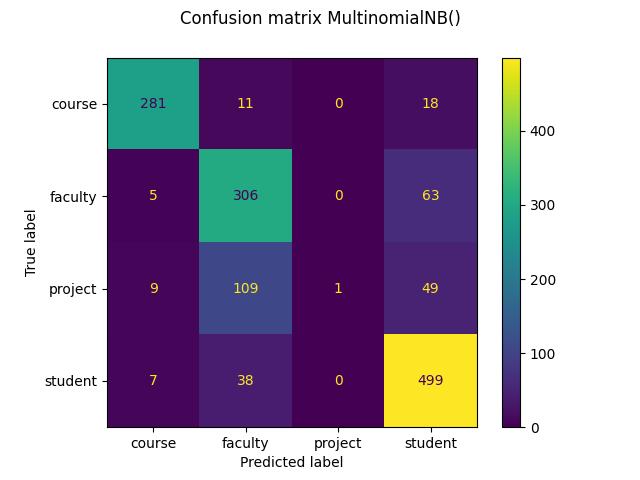
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-score** | **Support** |
| **1** | 0.95 | 0.95 | 0.95 | 310 |
| **2** | 0.87 | 0.86 | 0.87 | 374 |
| **3** | 0.85 | 0.75 | 0.80 | 168 |
| **4** | 0.89 | 0.92 | 0.91 | 544 |
| **Accuracy** | 0.89 | 0.89 | 0.89 | 0.89 |
| **Macro avg** | 0.89 | 0.87 | 0.88 | 1,396 |
| **Weighted avg** | 0.89 | 0.89 | 0.89 | 1,396 |

Fig. 3. Classification Report for Linear SVC



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-score** | **Support** |
| **1** | 0.96 | 0.94 | 0.95 | 310 |
| **2** | 0.86 | 0.84 | 0.85 | 374 |
| **3** | 0.91 | 0.55 | 0.69 | 168 |
| **4** | 0.83 | 0.95 | 0.89 | 544 |
| **Accuracy** | 0.87 | 0.87 | 0.87 | 0.87 |
| **Macro avg** | 0.89 | 0.82 | 0.84 | 1,396 |
| **Weighted avg** | 0.88 | 0.87 | 0.87 | 1,396 |

Fig. 4. Classification Report from Random Forest Classifier



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-score** | **Support** |
| **1** | 0.93 | 0.91 | 0.92 | 310 |
| **2** | 0.66 | 0.82 | 0.73 | 374 |
| **3** | 1 | 0.01 | 0.01 | 168 |
| **4** | 0.79 | 0.92 | 0.85 | 544 |
| **Accuracy** | 0.78 | 0.78 | 0.78 | 0.78 |
| **Macro avg** | 0.85 | 0.66 | 0.63 | 1,396 |
| **Weighted avg** | 0.81 | 0.78 | 0.73 | 1,396 |

Fig. 5. Classification Report for Naïve Bayes (Gaussian) Classifier

The models are compared in the table below:

|  |  |
| --- | --- |
| **Classifier** | **Accuracy Score** |
| **Naïve Bayes (Gaussian)** | 0.778653 |
| **Linear SVM** | 0.893983 |
| **Decision Tree (Entropy)** | 0.782235 |
| **Random Forest** | 0.871777 |

Fig. 6. Comparison of Accuracy Scores of Classifiers

# Conclusion

In summary, we achieved classification reports for our data set by implementing different models. Support Vector Machine-based models happen to give the best performance for the given data set followed by Random Forest.

# Acknowledgments

I used multiple machine learning models and compare how well they perform on single-label text classification tasks using some well-known public datasets discussed by Ana Cardoso that are actively used for research. The main goal is to reproduce part of her Ph.D. work using state-of-the-art libraries in python consisting of sklearn, matplotlib, pandas, etc. I consider her work to be successful if I can reproduce the initial "related work" from her thesis. I expect results to be approximately the same as her published results.

##### References

1. [Sklearn.feature\_extraction.text.TfidfTransformer — scikit-learn 1.2.0 documentation](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#sklearn.feature_extraction.text.TfidfTransformer)
2. F. Sebastiani. Machine learning in automated text categorization. ACM Computing Surveys, 34(1):1–47, 2002.
3. Y. Yang and X. Liu. A re-examination of text categorization methods. In Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval pages 42–49. ACM Press, 1999
4. [The 4 Universities Data Set Homepage](http://www.google.com/url?q=http%3A%2F%2Fwww.cs.cmu.edu%2Fafs%2Fcs.cmu.edu%2Fproject%2Ftheo-20%2Fwww%2Fdata%2F&sa=D&sntz=1&usg=AOvVaw3sn3EgrlvyUavPg9zj8z8K).

.