COMP5318: Text Categorization using Ensemble Learning (maybe, heheee)

Xt Hughes, Vanush Vaswani

May 9, 2014

Introduction

In our connected world, the amount of data being produced is ever increasing. Approaches within the field of machine learning and data mining seeks to extract meaning and predict data for economic or scientific benefit. In this report, one approach to text classification is described. The given dataset, consists of "apps", or mobile applications, their descriptions and their labels. In this report, a supervised learning algorithm is devised to be able to predict the category of application from prior examples. Basic models within the field are evaluated using a k-fold cross validation technique, with an ensemble of classifiers with majority voting having the best overall result.

Problem Formulation

Natural language text must be transformed for computers to be able to construct and evaluate models using their data. A common approach is the Vector Space Model, where each unique word in the corpus is represented in a matrix formulation. Thus, i observations (e.g. documents) and a corpus consisting of j unique words results in a matrix of dimension i x j. If a term appears in a certain document (or row), a 'one' (1) value is set for this cell. This approach is also known as a Bag-of-Words (or "BOW") representation and results in a sparse matrix with a large number of features.

Feature Weighting: tf-idf

Blah blah blah blah

Text Categorization

The goal of text categorization is the classification of documents into a number of predefined categoies [1]. Considering the high dimensionality of the vector space, it is known within the field that linear classifiers give the best performance

[1], with numerous low-level software libraries [2] available to take advantage of the sparsity of the data. In addition, the large vector space formed by the BOW representation results in classes having a high probability of being linearly seperable [3]. Using machine learning, we construct a statistical model (a classifier) from which we can predict unlabeled observations. The model can be summarized as a function

$$y = f(w \cdot x) = f\left(\sum_j w_j x_j\right)$$

where w is a vector of weights that are learnt during training with examples x. In practice, the features form this set.

Dimensionality Reduction

The problem of text categorization by classification implicity suffers from the curse of dimensionality, where basic algorithms suffer a loss of performance as feature vector size increases [TODOREF - why?]. Dimensionality reduction seeks to transform the data to a dimension of lower rank while retaining the predictive power of the features. This process is divided into two categories

1. Feature extraction: Typically unsupervised, the data is projected into a space of lower dimensionality while retaining a user-defined amount of variance in the dataset [4] (for example, SVD or PCA). For a text classification problem, the generally accepted number of components (hence features) are 200 - 500.

2. Feature selection: Features are selected according to a univariate statistical metric. Some metrics include Information Gain (IG) and the χ^2-test.

Methodology

Considering the above, the method chosen for categorizing text data is as follows

1. Run a random guess classifier to determine baseline performance

2. Preprocess the data by dimensionality reduction

3. Split the data into a training set and test set

4. Train appropriate classifiers that are known to work well for text classification

5. Run 10-fold cross validation and report results

As is typical, the method is highly empirical as every dataset exhibits different characteristics. Each step is described in detail below. The machine learning framework Scikit-Learn [5] was used to perform these tasks.

Baseline performance

Pre-processing

Sparse representation - scipy sparse - etc
dimensinality reduction, try PCA, top 20% of features etc.

Train/Test Split

90/10.. does this need to be a section? LaWl

Classifier Evaluation and Selection

Ridge, NB, SVM, etc

Ensemble of Classifiers

Why this is better.. is it actually better ? Reduces variance in prediction? improves accuracy? Might be inherent nonlinearities in the data? blobob

Nonlinear Classification

Why they suck and over fit

Results

Test A

Test B

Test C

Conclusion

References

[1] S. M. Weiss, N. Indurkhya, T. Zhang, and F. Damerau, *Text mining: predictive methods for analyzing unstructured information.* Springer, 2010.

[2] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[3] T. Joachims, *Text categorization with support vector machines: Learning with many relevant features.* Springer, 1998.

[4] C. M. Bishop *et al.*, *Pattern recognition and machine learning.* springer New York, 2006, vol. 1.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, “Scikit-learn: Machine learning in python,” *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.