An Application of Machine Learning to Native Advertisements

Kevin Grogan, Quinlan Jung

Abstract

Machine learning algorithms are applied to data aggregated from the internet to classify webpages as native advertisements or core content. Four different classifiers are employed: logistic regression, Poisson naive Bayes, multinomial naive Bayes, and support vector machine. The features are selected using a threshold based on the mutual information statistic; additionally, the tf-idf and tf-bns transformations are applied to the training set. Principal component analysis is examined as an a priori feature reduction algorithm. The reduction in the dimensionality of the training set through principal component analysis is found to increase the test error of the logistic regression classifier to an unacceptable level. The tf-idf and tf-bns transformations are found to successfully improve the quality of the classifiers. The logistic regression classifier is shown to produce the lowest test error of all the classifiers examined.

1. Introduction

Native advertisements seek to mimic the core content of websites in order to minimize the disruption to the users viewing experience. Ideally, the advertisements are as fun and informative as the website's core content. However, they are often intrusive and decrease the likelihood of repeat viewings. Therefore, advertisers would greatly benefit from a model that could determine which content is an advertisement and which is not. Tricking such a model post facto would imply that the advertisement blends well with the core content of the website. Additionally, identification of key parameters that betray advertisements would help advertisers develop more organic content.

Hence, this project applies machine learning algorithms to classify articles as native advertisements or core content. For this project, the naive Bayes, support vector machine (SVM), and logistic regression algorithms as classifiers are employed. Additionally, principal component analysis (PCA) to examine its utility in feature reduction.

2. Related Work

Naive Bayes and support vector machines have been the traditional approaches to solving text classification problems. When the dataset is extremely large, logistic regression is sometimes used to approximate an SVM because it is computationally more efficient [4].

Naive Bayes performs moderately well in text classification. Of the different event models, the multivariate Bernoulli performs well with small vocabulary sizes, but the multinomial usually performs better at larger vocabulary sizes, providing on average a 27% reduction in error over the multivariate Bernoulli model at any vocabulary size [3].

SVMs consistently achieve good performance on text classification tasks, outperforming many existing methods [5]. Because most text classification problems are linearly separable, a linear kernel is usually recommended, as it is faster and easier to optimize [6]. Other popular kernels, such as the string kernel, show positive results on modestly sized datasets. For larger documents and datasets, approximation techniques need to be used because non-linear kernels are computationally expensive [7].

3. Data Set and Features

The training set consists of 300,000 HTML pages with class labels provided by kaggle [2]. Since the training set consited of raw html, the following algorithm is used to tokenize the training set:

```
for each HTML document:
        extract the text in the HTML body,
             stripping away tags
        remove all punctuation
        remove all stop words
        merge similar words
```

While embedded javascript and links in the HTML head along with punctuation in the body could provide more insight to determine whether a document is an advertisement, this hidden information is not apparent to the audience and is unlikely to betray a native advertisement. Hence, this project focuses on the text in the body.

Stop words such as "a", "and", and "the", are removed as an a priori method of feature reduction. Additionally, words with the same roots were grouped by detecting common suffixes such as "-ly" and "ing". Implicitly, this assumes that the covariance between these words is sufficiently high that the dimensional reduction will have a negligible effect on the bias of the models.

The feature set, or dictionary, is the vocabulary found in the training set. The collection of tokenized training documents is given to a CountVectorizer[1], and it returns an $n \times m$ frequency matrix, A, where n is the dictionary size, and m is the number of documents. The entries A_{ij} denote the frequency of the word at index i found in document j.

Email addresses: kgrogan@stanford.edu (Kevin Grogan), quinlanj@cs.stanford.edu (Quinlan Jung)

[1]A python class in the scikit-learn library that converts a collection of text documents to a matrix of token counts

3.1. Term Frequency/Inverse Document Frequency

Term frequency/inverse document frequency (tf-idf) is a heuristic metric used in text classification to yield better results with classifiers. The functional form of the tf-idf statistic used for this project is

$$\begin{aligned} \text{tf-idf}_{ij} &= \text{tf}_{ij} \times \text{idf}_i \\ &= \log(1 + A_{i,j}) \times \log\left(\frac{N}{n_i}\right), \end{aligned} \tag{1}$$

where N is the size of the training set, and n_i is overall frequency of word i in the training set. tf-idf makes the reasonable assertion that the importance of a word in a document must be monotonically dependent on the frequency of that word (i.e., the tf statistic). However, this term should be discounted when it is omnipresent in the corpus (i.e., the idf statistic); therefore, words with little embedded meaning and mostly syntactical relevance (such prepositions, conjunctions, and articles) are neglected by the classifiers under this transformation.

3.2. Term Frequency/Bi-Normal Separation

While tf-idf is the most widely used representation for real-valued feature vectors for text classification problems, idf is oblivious to the training class labels and naturally scales some features inappropriately. Alternatively, idf can be replaced with bi-normal separation (bns), which has been previously found to be successful at ranking words for feature selection filtering [9]. The functional form of the tf-bns statistic used is

$$\begin{aligned} \text{tf-bns}_{ij} &= \text{tf}_{ij} \times \text{bns}_i \\ &= \log(1 + A_{i,j}) \times |F^{-1}(\text{tpr}) - F^{-1}(\text{fpr})|, \end{aligned} \tag{2}$$

Where the true positive rate is given by tpr = P(word | positive class), the false positive rate is given by fpr= P(word | positive class), and F^{-1} is the inverse Normal cumulative distribution function.

3.3. Mutual Information

Mutual information provides a metric to assign an importance a given feature to the classification of the document. It is defined as

$$\text{MI}(x_i, y) = \sum_{x_i \in \{0,1\}} \sum_{y \in \{0,1\}} p(x_i, y) \log \frac{p(x_i, y)}{p(x_i)p(y)} . \tag{3}$$

Rewriting Eq. 3 using Jensen's inequality and applying the relation $p(x_i, y) = p(y|x_i)p(x_i)$, the mutual information of a feature, x_i, is shown to be bounded by

$$\text{MI}(x_i, y) \leq \log \text{E}\left[\frac{p(y|x_i)}{p(y)}\right] , \tag{4}$$

where the expectation is taken over the joint probability distribution of x_i and y. This furthers the intuition that the mutual information is a bounded and normalized metric related to the expected contribution of a feature to the successful prediction of a class label over simply the probability of the class label.

However, the bounding of this metric indicates that it may underpredict the contribution of a feature to the classification of a data set. Although, for large feature sets such as the text classification used for this project, mutual information is posited to be a reasonable and computationally efficient criterion to eliminate features and reduce the variance error for a classifier.

4. Methods

4.1. Principal Component Analysis

Principal component analysis seeks to reduce the dimensionality of a feature set. This is done by finding a unit vector u that maximizes the variance in the data when projected onto the vector. Formally, this can be shown to be given by

$$u = \underset{u:||u||=1}{\text{argmax}}\ u^{\text{T}}\Sigma u , \tag{5}$$

where Σ is the covariance matrix. Through the method of Lagrange multipliers, this relation can be demonstrated to yield the eigenvector corresponding to the maximum eigenvalue of the covariance matrix: $u = v_i : \lambda_i = \max\{\lambda_1, ..., \lambda_n\}$.

Hence, an eigenvalue decomposition of the covariance matrix gives a basis of eigenvectors, where the variance of the projection onto each eigenvector is in proportion to the corresponding eigenvalue.

Furthermore, dimensional reduction of the training data may be obtained by a projection onto the eigenvector basis:

$$B_k = V_k^{\text{T}} C, \tag{6}$$

where $B_k \in \mathbb{R}^{k\times m}$ is the reduced representation of the columns of matrix $C \in \mathbb{R}^{n\times m}$ to $k \leq n$ dimensions, and $V_k \in \mathbb{R}^{n\times k}$ is a matrix of the first k principal eigenvectors. For this project, the matrix C is built from the tf-idf transformation of the frequency matrix detailed by Eq. 1, and the rows of C are normalized to have zero mean and unit variance.

Additionally, since the initial set of features yield a covariance matrix with $O(10^9)$ elements, sparsity is enforced using the Cauchy-Schwarz inequality

$$\Sigma_{ij} = \begin{cases} \text{Cov}(c_i, c_j) & \text{for } \frac{[\text{Cov}(c_i,c_j)]^2}{\text{Var}(c_i)\text{Var}(c_j)} > d \\ 0 & \text{o.t.w.} \end{cases} \tag{7}$$

where c_i, c_j are row vectors in C, and d is a threshold parameter set to 0.8. This parameter to yield a matrix with a density just so that the matrix could be manipulated in a practical manner on a laptop.

4.2. Logistic Regression

A logistic regression assumes that the conditional probability of the labeling follows a Bernoulli random variable (i.e, $y|x;\theta \sim \text{Bernoulli}(\phi)$). By defining the hypothesis function as the expectation of the conditional probability, $h_\theta(x) = \text{E}[y|x;\theta]$ and assuming a linear relation between the feature vector and parameter vecotor, θ, the hypothesis function is correspondingly found to be given by the sigmoid function:

$$h_\theta(x) = \frac{1}{1 - \exp(\theta^{\text{T}} x)} , \tag{8}$$

where the parameter vector is found directly through a maximum liklihood estimate (MLE). The MLE of the logistic regression is found using batch gradient ascent where the learning rate is found empirically after several trials; stochastic gradient ascent was attempted, but the parameter vector was found to be unacceptably uncertain.

4.3. Naive Bayes

The Naive Bayes algorithm utilizes Bayes rules to predict a label given a feature set:

$$p(y=1|x) = \frac{p(x|y=1)p(y)}{\sum_{y\in\{0,1\}} p(x|y)p(y)} , \tag{9}$$

where the prior distributions $p(x|y = 1)$ and $p(y)$ must be modeled. The hypothesis function takes the form $h_\theta(x_i) = p(y = 1|x)$ for the naive Bayes algorithm. Additionally, the prior $p(x|y)$ is assumed to be conditionally independent:

$$p(x|y) = \prod_{i=1}^{n} p(x_i|y) . \tag{10}$$

The prior $p(y)$ is modeled using a binomial distribution, while $p(x|y)$ is modeled using a multinomial distribution or a Poisson distribution.

A Poisson distribution is considered under the rational that the number of appearances of a word in a fixed length document is well-approximated by a Poisson process. The prior for a Poisson distribution is given by

$$p(x_i|y=j) = \frac{\lambda_{ij}^{x_i}\exp(-\lambda_{ij})}{x_i!} \tag{11}$$

where λ_{ij} is E$[x_i|y = j]$. Additionally, it can be shown that Eq. 9 can be written in the form of a sigmoid function given by Eq. 8, making implementation straightforward. The parameter vector in this case is given by

$$\theta_k = \begin{cases} \log\frac{\phi_y}{\phi_y - 1} + \sum_{i=1}^{n}\lambda_{i0} - \lambda_{i1} & \text{for } k = 0 \\ \log(\lambda_{k1}/\lambda_{k0}) & \text{o.t.w.} \end{cases} \tag{12}$$

where $\phi_y = p(y)$. The MLE of λ_{ij} is given by the sample mean of each feature corresponding to the labeling $y = j$.

Assuming a Poisson distribution increases the computational efficiency of the classifier but also embeds stronger assumptions into the model. The comparative performance of the classifier will be examined in the Sec. 5.

4.4. Support Vector Machine

The SVM attempts to find the maximum-margin hyperplane that separates the dataset in a higher dimensional feature space. Finding this optimal margin reduces to solving the following convex optimization problem:

$$\min_{\gamma,\omega,b}\frac{1}{2}||\omega||^2 + C\sum_{i=1}^{m}\xi_i \tag{13}$$

Subject to the constraints:

$$\begin{aligned} \text{s.t. } y^{(i)}(\omega^T x^{(i)}) + b) &\geq 1 - \xi_i, i = 1, ..., m \\ \xi_i &\geq 0, i = 1, ..., m \end{aligned} \tag{14}$$

Where ξ_i allows for the 'slack' in the event that the data is not linearly separable. Originally, an unoptimized L2-norm linear SVM was employed, but it proved to be too slow due the high cardinality of the training set. The SVM was modified to use a dual coordinate descent method [8] that defines the primal problem as follows:

$$\begin{aligned} &\min_{\omega_m,\xi_i}\frac{1}{2}\sum_m ||\omega_m||^2 + C\sum_i \xi_i \\ &\text{s.t. } \omega^T x^{(i)} - \omega^T x^{(i)} \geq e_i^m - \xi_i \ \forall m, i \end{aligned} \tag{15}$$

Where $e_i^m = 1 - \sigma_m$ and $\sigma_m = 1$ if $y = m$, $\sigma_m = 0$ if $y_i \neq m$. This method reaches an ϵ-accurate solution in $O(\log(1/\epsilon)$ iterations. Similar to what is done with Naive Bayes, tf-bns transformation is performed on the vocabulary found in the training set. The SVM is set to have parameters $C = 0.23$ and $\xi = 1.0$. The optimal C and ξ are found using k-fold cross validation, where $k = 10$.

5. Discussion

5.1. Feature Reduction using Principal Component Analysis

As described in Sec. 4.1, principal component analysis reduces the dimensionality of the feature set by maximizing the variance of the projection of the data onto a set of basis vectors. Figure 1 shows the effect of the reduced feature set on the test and training errors.

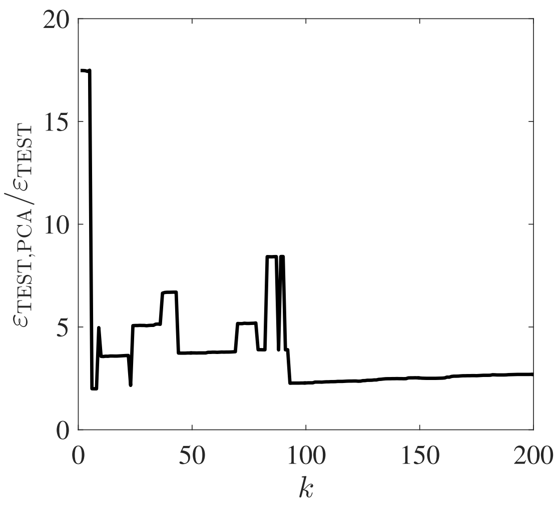

Figure 1: The effect of the reduction in dimensionality through PCA on the test error. The error is normalized by that of the full-rank representation of the data set. Error is found using the logistic regression classifier and k-folds cross validation.

The figure shows that error is significantly increased when the dimensionality of the data is reduced through principal component analysis. Hence, while PCA reduces the feature set, the substantial increase in bias overshadows any reduction in variance. Additionally, PCA is an unsupervised learning algorithm that is agnostic to the class labels. Hence, the belief that a reduction of dimensionality through this algorithm will provide lower test error may be Pollyannaish.

The threshold to enforce the sparsity may have been overly aggressive to yield a successful implementation of PCA; however, the reduction of this threshold would have been impractical for this project. Additionally, the normalization

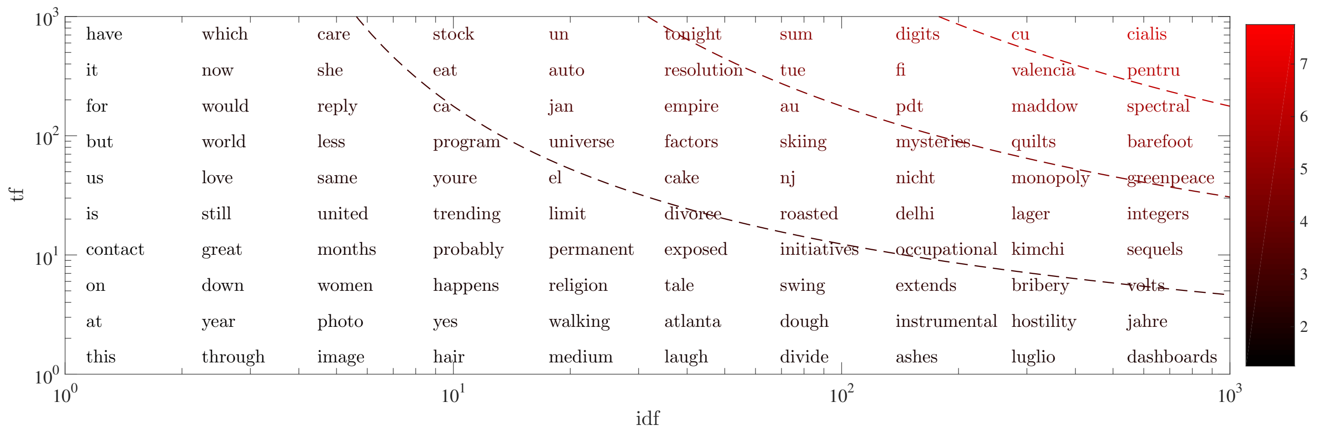

Figure 2: Depiction of the tf-idf transformation. Words are colored and by the tf-idf value given by Eq. 1.

of the feature vectors may have reduced the useful information contained within causing some of the high training error observed; however, not normalizing the data would have certainly reduced the clustering of the data diminishing the efficacy of employing PCA.

As an interesting aside, inspection of the top words corresponding to different eigenvectors seem to align with the suspected orgin of the article; for instance, words with a British spelling such as “organisation” and “stabiliser” corresponded to the third principal eigenvector. Hence, PCA may have more promise as a clustering algorithm.

5.2. *Effect of the tf-idf Transformation*

The effect of the tf-idf transformation discussed in Sec. 3.1 is shown in Fig. 2. As illustrated in the figure, the tf-idf transformation naturally selects words with a high relevance to the document. Words such as “this” and “have” are shown to have a high idf, while the term frequency of the high idf terms are document dependent. Interestingly, “cialis” is shown to have a large tf and idf in this matrix indicating a high importance; considering the connotation, this term likely appeared in an advertisement. Additionally, the tf-idf transformation is found to reduce the test error by 36% for the logistic regression classifier.

5.3. *Application of the tf-bns Transformation to Multinomial Naive Bayes*

Originally, the tf-idf weighting scheme was used on features of multinomial naive Bayes. However, the positive predicted value (ppv)

$$\text{ppv} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{16}$$

is 0, as it is unable to predict even a single native advertisement correctly. This is because idf is oblivious to the class labels in the training set, which can lead to inappropriate scaling. Upon switching to tf-bns, an increase of average ppv to 0.75, where n = 3000, is observed.

5.4. *Feature Reduction through Mutual Information*

As discussed in Sec. 3.3, the mutual information of a feature quantifies the usefulness of a feature in predicting a classification. Figure 3 shows the effect of feature reduction via

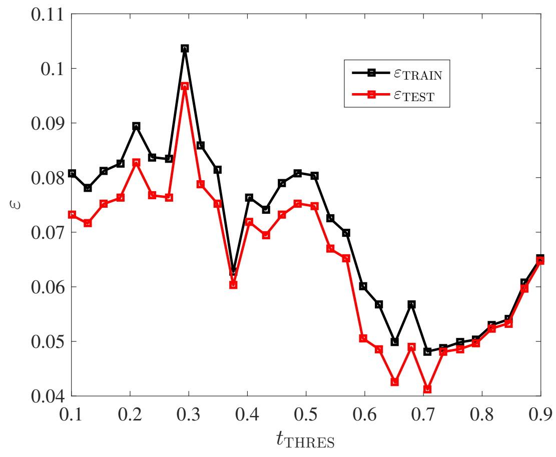

Figure 3: Comparison of the test and training error as a function of the threshold parameter based on the mutual information. Error is determined via k-folds cross validation. Error is shown for the logistic regression classifier. The tf-idf transformation is applied.

the mutual information on the error. The abscissa of the figure corresponds to the threshold parameter t:

$$t(x_i, y) = \frac{\text{MI}(x_i, y) - \min_j \text{MI}(x_j, y)}{\max_j \text{MI}(x_j, y) - \min_j \text{MI}(x_j, y)} \tag{17}$$

where all features such that $t < t_{\text{THRES}}$ are disregarded.

As shown in the figure, the error first reduces for an increasing threshold (i.e., as more features are eliminated), and experiences a minimum near $t_{\text{TRES}} = 0.7$; this is likely due to the decrease in the variance as features are reduced. Subsequently, the error begins to increase as bias error likely becomes dominant. Note that at high thresholds, training error and test error becomes similar indicating a reduction in variance. Hence, the metric of mutual information is shown to reduce the test error in a computationally feasible manner.

5.5. *Comparison of the Classifiers*

A comparison of the test and training errors for the classifiers used in this project is presented in Fig. 4. The logistic regression is shown to yield the lowest test error of the classifiers at the highest cardinality of the training set (i.e.,

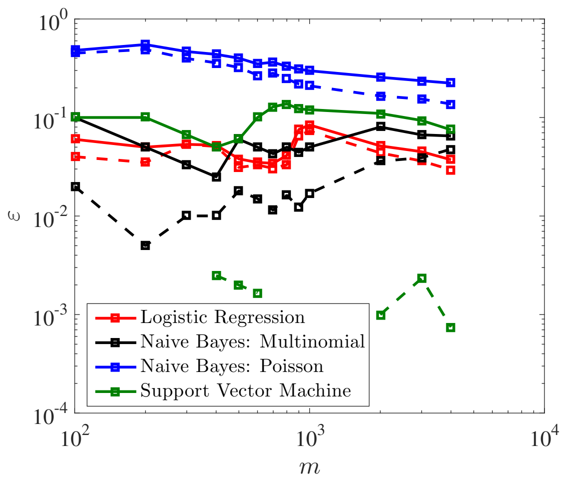

Figure 4: Comparison of the test and training error as a function of cardinality of the training set *m*. Solid lines denote the test error while dashed lines denote the training error. The errors are deduced through a k-folds cross validation.

m). This is posited to be due to the generality and therefore, adaptability of the classifier. The logistic regression is a generalization of the naive Bayes models and it is not unexpected that it performs better since the parameters are selected to directly maximize the liklihood function.

The Poisson naive Bayes model is shown to perform the worst of the classifies. This likely results from the strong assumptions imposed by modeling each feature as a Poisson distributed random variable. Particularly for small cardinalities of the training set, the statistics are insufficiently converged to provide a meaningful classifier, and the error is near 50%. However, the Poisson naive Bayes is found to be the most computationally efficient model performing 63× faster than the logistic regression.

The support vector machine is not shown to generalize particularly well for this problem. While the training error is quite low (zero in some cases), the test error is moderate - between that of the logistic regression and the multinomial naive Bayes. This is likely due to the classifier emphasizing a few support vectors that determine the decision boundary.

All classifiers are shown to require at least 1000 training examples to begin showing a regular slope in with respect to the test error. In this regime, the logistic regression is shown to have the steepest slope and can be seen to be approximately first order.

6. Conclusions

Native advertisements are an effective strategy to market to consumers without damaging the core experience of viewing a website. It is posited that the quality of native advertisements may be improved by using a machine learning model to differentiate advertisements from core content; such would provide a metric for the quality of the advertisement. Hence, several machine learning algorithms are applied to the detection of native advertisements and the following conclusions are drawn:

1. Principal component analysis showed lackluster performance as a means to reduce the dimensionality of the feature set.
2. The tf-idf transformation successfully augments words that are pertinent to a document while discounting those that are superfluous.
3. the tf-bns transformation showed an improvement over tf-idf in the ppv metric when applied to the multinomial naive Bayes model.
4. Feature reduction through the mutual information metric improved the test error of the logistic model.
5. The logistic regression is found to yield the best training error of the classifiers used in this project.

Additionally, the authors note that this report is not a native advertisement as determined by the logistic regression classifier.

7. Acknowledgment

The authors would like to thank Rakesh Ramesh for his early contributions to this project.

References

[1] http://www.stumbleupon.com/

[2] https://www.kaggle.com/c/dato-native/data

[3] Andrew McCallum and Kamal Nigam. 1998. *A comparison of event models for Naive Bayes text classification.* In Proc. of the AAAI-98 Workshop on Learning for Text Categorization, pages 41–48.

[4] J. Zhang, R. Jin, Y. Yang, and A. Hauptmann. *Modified logistic regression: An approximation to SVM and its applications in large-scale text categorization.* In Twentieth International Conference on Machine Learning, pages 472479, 2003.

[5] T. Joachims. *Text categorization with support vector machines: Learning with many relevant features.* In Claire Nedellec and Celine Rouveirol, editors, Proceedings of the European Conference on Machine Learning, pages 137142, Berlin, 1998. Springer.

[6] http://www.svm-tutorial.com/2014/10/svm-linear-kernel-good-text-classification/

[7] H. Lodhi, J. Shawe-Taylor, N. Cristianini, and C. Watkins. *Text classification using string kernels.* In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, Adavances in Neural Information Processing Systems, pages 563569, Cambridge, MA, 2001. MIT Press.

[8] S. Sathiya Keerthi, S. Sundararajan, Kai-Wei Chang, Cho-Jui Hsieh, Chih-Jen Lin, *A sequential dual method for large scale multi-class linear svms*, Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, August 24-27, 2008, Las Vegas, Nevada, USA.

[9] George Forman, *An extensive empirical study of feature selection metrics for text classification*, The Journal of Machine Learning Research, 3, 3/1/2003