Discovering Evergreen Content on the Web

Ushnish De, Xiaotong Suo McClure, Konstanin Stulov

***Abstract*—In this work we explore a dataset of websites which have been classified as evergreen (interesting in the long run) or ephemeral based on the content of the page. The objective is to build a classifier which focusses on key attributes of the content on a page to determine whether users would have rated it one way or the other. We used various algorithms which were implemented on different representations of the text, and while we attained up to 83% accuracy in some cases, we could not exceed this percentage by any single method. We believe that the accuracy can be improved by more careful preprocessing of the raw data and applying an optimal combination of classification algorithms as an ensemble.**

I. Introduction

StumbleUpon is a user-curated website that aggregates content which users have up-voted as interesting or engaging. While some popular content is considered **ephemeral** or interesting for only a temporary amount of time, like seasonal recipes or news articles, other content can be considered **evergreen** or interesting in the long run, such as advice forums or how-to websites. While users themselves can flag content as evergreen, StumbleUpon is interested in knowing ahead of time which websites are likely to be evergreen and show these prominently. The main aim of this project is to apply techniques of Machine Learning to build a classifier that is best capable of predicting how users would label the websites as evergreen or ephemeral. In the process, we want to determine which attributes of a website have the greatest significance in whether it is considered evergreen by users.

II. Background

The data set consists of the raw HTML content scraped from approximately 10,000 websites, as well as a .tsv text file with information aggregated from these websites, possibly using an HTML parser. There are essentially two types of fields in the aggregated data set that we can use to determine what evergreen websites have in common - text fields and numeric fields. At the start, we tried experimenting with both the numeric and the text fields by applying classification algorithms (e.g. Naive Bayes, Decision Trees, etc.). However, due to a high number of missing values and the presence of clearly wrong entries in the numeric fields, we decided to mainly use the boilerplate text only. The boilerplate text field consists of the title and body text of the website in a JSON string format. In other words, it looks like "{"title":"Easy Coconut Rice Recipe","body":"body text"}".

III. Pre-Processing Steps

1) Tokenization: cut sequence of characters into word tokens.
2) Normalization: transform different formats of different words into the same word.
3) Stemming: match words from the same root.
4) Stop Words: remove common words such as 'a', and 'the'.

IV. Feature Representation

The first task in text classification is to transform the document to a representation on which we can perform classification algorithms. The format used for text representation can have an impact on the effectiveness of these algorithms. Hence, care must be taken in choosing an appropriate feature representation.

The vector space model is a widely used document representation model.[2] In this model, every document is represented as a vector and each entry of the vector, denoted $w_{t,d}$, is a real number (a weight) that corresponds to each term t in each document d. Different values can be used for the weights $w_{t,d}$. In this paper, we explore three different representations of the text documents - binary representation, TF model and TF-IDF model. All of these representations of documents ignore the ordering of the words. For example, the phrase "Mary likes Cake" has the same representation as "Cake likes Mary". Research in information retrieval has shown that the ordering of words in a document has minor impact for many tasks[1], which leads us to adopt the bag of words model. We first introduce the following notation:

A. Binary Representation

This is the simplest model. We use binary values for the real valued function.

$$w_{t,d} = \begin{cases} 1 & \text{if word } t \text{ appears in the document } d \\ 0 & \text{otherwise.} \end{cases}$$

B. Tf model

The Term frequency (Tf) model takes into account the number of times that a word appears in the document. This model assumes that words that appear more often in the document should be given more weight. $tf_{t,d}$ denotes the number of times term t appears in the document d. However, if one term appears ten times more than another term, it does not mean that the term is ten times more informative than the other term. Therefore, we use $\log_{10}$ to dampen this effect and define:

$$w_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d} & \text{if } tf_{t,d} > 0 \\ 0 & \text{otherwise.} \end{cases}$$

C. $Tf - idf$ model

This model takes both term frequency and document frequency into account. Besides the assumption of the TF model, this model also assumes that if a term appears in too many documents, it is less informative than a rare term. We use df_t to denote the document frequency of the term t (i.e. the number of documents that contain the term t), which gives an inverse of the informativeness of the term t. Thus, given a total of N documents, we define

$$idf = \log_{10} \frac{N}{df_t},$$

where $\log_{10}$ was used to dampen the effect of idf. Our TF-IDF model is then defined as:

$$w_{t,d} = \log(1 + tf_{t,d}) * \log_{10} \frac{N}{df_t}.$$

V. Feature Selection

Feature spaces that come up in text classification problems tend to have very high dimensions. Hence, feature selection is used to reduce the size of these feature spaces. Research has shown that aggressive reduction of the feature spaces leads to little loss in accuracy and, in many cases, to significant improvement in performance.[3] Rogati and Yang found that χ^2 statistic is more effective for optimization classification results in text classification. We applied χ^2 statistic to feature selection and analyzed the results with and without the χ^2 statistic.

VI. Evaluation

In text classification, Precision, Recall, F_1 measure and Accuracy are often used for validation of results. We denote tp as true positive, fp as false positive, tn as true negative, and fn as false negative, where positive and negative refer to the output of a classification algorithm.

$$\text{Accuracy} = \frac{tp + fn}{tp + fp + tn + fn},$$

$$p = \text{Precision} = \frac{tp}{tp + fp},$$

$$r = \text{Recall} = \frac{tp}{tp + fn},$$

$$F_1 = \frac{2pr}{p + r}.$$

VII. Classification Algorithms

We applied Naive Bayes, SVM, Logistic regression, Decision Tree and K-NN classification algorithms on this dataset.

VIII. Discussion and Results

A. Naive Bayes Text Classifier

Our first pass at classifying the websites was using the words in the boilerplate text directly to classify the websites. We were hoping that the occurrences of certain words would determine whether or not a website was considered evergreen or not. We ran a text classifier on the data with and without preprocessing (deleting stop words, spelling errors and stemming). After the model was trained on this set we computed the accuracy percentage of the model on the examples which were withheld from the model, i.e. the test set. Figure 1 shows how the training and the test accuracy of the Naive Bayes model is changing with the number of training examples. The two curves are converging as the number of examples is increased. The peak performance was achieved when the number of examples was 1200 and the test accuracy was 75.5%. Beyond 1200 examples there was no more learning since the classification accuracy stopped improving.

This implementation also allowed us to analyze which specific keywords were the most indicative of a website being from a particular category. Figure 2 shows the words sorted by a "score" of how indicative they were of a category. From this preliminary analysis itself we can see that the users have tended to rate websites related to food as evergreen and websites related to sports, technology or news as ephemeral.

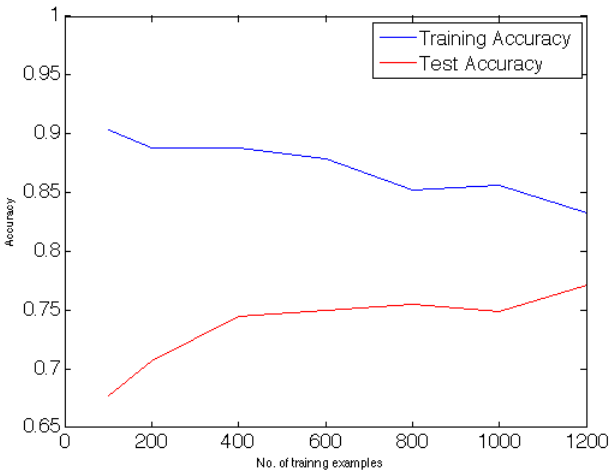

Fig. 1. Training and Test Accuracy % VS No. of training examples

Evergreen	Ephemeral
recipe	upload
tablespoon	download
creamy	application
olive	swimsuit
chill	nfl
peanut	director
soften	generation
oven	analysis
shred	url
preheat	court
mixture	device

Fig. 2. Table of words sorted by indicativeness of category

B. SVM

We implemented SVM with both linear and gaussian kernel. We used 10-fold cross-validation to choose the parameter C. We determined that $C = 0.5$ with L_1 penalty and gives the best result for both models. Since SVM with gaussian kernel gives worse results than linear SVM in every case for this dataset, the results of gaussian SVM are not included in this report. We found that SVM with non-linear kernel causes overfitting to happen on our dataset.

C. Decision Tree

Decision Trees are used to select informative features based on an information gain criterion, and to predict the categories of each document according to the values of the features in each document. In this case the features are words and the values are 1 or 0 depending on whether the word exists in the document. DT also allowed us to incorporate numerical features (e.g. number of links in the web-page, fraction of misspelled words, etc.) into the model.

A major source of difficulty with DT model was model over-fitting and high variance in predictions. The training error was consistently 95-100% and the test error was 70-80%. Some of the DT variants that were implemented included numerical feature reconstruction (using median values in place of missing values), feature vectors with full and reduced vocabulary and Tf-idf values.

The best results ($\sim$ 80% classification accuracy) were produced by using Tf-idf valued feature vectors with the Extra-trees ensemble method (extremely randomized DT), which builds each tree in the ensemble by sampling with replacement (i.e., a bootstrap) from the training set.

D. Decision Tree with Naive Bayes

In order to reduce over-fitting of the Decision Tree Classifier, we attempted another approach that used the list of informative words which were output by the Naive Bayes Text Classifier trained on the same data set. This list of informative words was used as the vocabulary of words for the decision tree instead of the full vocabulary. We believed that this could make the decision tree algorithm focus on the occurrence of words which are most strongly indicative of whether a website was labelled evergreen or not. However, this approach did not give any statistically significant improvement over the previous approach. We believe that this could be because the over-fitting of decision trees masks any potential gain that we get from using the most informative features produced by the Naive Bayes classification.

E. K-Nearest Neighours classification

We implemented K-NN classification using different choices of k and used 10-fold cross-validation to determine how the accuracy of the model varies with k. As Figure 3 shows, the accuracy improves until k reaches 10, and plateaus from then on. This figure also demonstrates how the accuracy of the K-NN model fluctuates with the parity of k.

F. N-grams

We ran all of the above algorithms using n-grams of different lengths, and these were essentially the terms in our Tf-idf model. It turned out that in every algorithm, 1-grams provided the best accuracy. Figure 4 shows how each algorithm performed for different sizes of n-grams. Although this is counter-intuitive, this may occur because there is not enough text within the boilerplate texts of

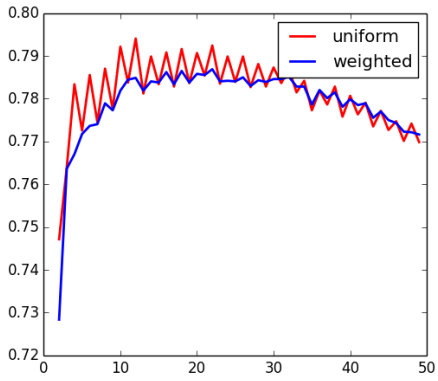

Fig. 3. Accaccuracy of K-NN vs the number of neighbors, using a uniform and weighted distance metric

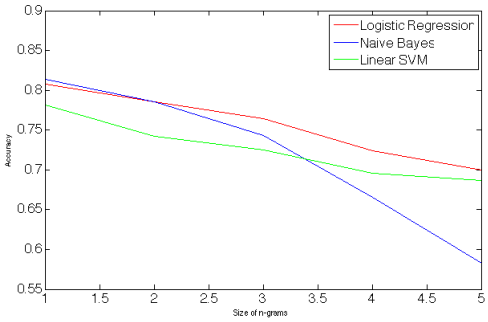

Fig. 4. Performance of classification algorithms for different sizes of n-grams

most websites and the Tf-idf vectors get rapidly sparse for even small values of n.

G. Feature Selection

We also used χ^2 statistic to reduce the feature space. As Figure 5 shows, the accuracy increases until the feature size is 10^4, and stabilizes afterwards.

H. Validations and Comparisons

Table I shows that Naive Bayes is the best classification algorithm when we use Tf-idf model on this dataset. We also compared this result with binary representation model and Tf model, and Tf-idf model indeed gives us the best accuracy. Table II shows the results of using the top 10^4 features after feature selection using χ^2 statistic. In comparison with two tables, we can see that the accuracy does not change significantly between the full model and the reduced model.

Algorithm	Precision	Recall	F_1 score	Accuracy
Naive Bayes	0.80	0.83	0.82	0.83
Logistic Regression	0.86	0.74	0.79	0.81
Linear SVM	0.82	0.80	0.80	0.78
Random Forest (n=10)	0.84	0.70	0.77	0.78

TABLE I
ALGORITHMS APPLIED ON FULL TF-IDF MODEL

Algorithm	Precision	Recall	F_1 score	Accuracy
Naive Bayes	0.86	0.79	0.82	0.83
Logistic Regression	0.87	0.75	0.80	0.81
Linear SVM	0.84	0.78	0.80	0.81
Random Forest (n=10)	0.84	0.72	0.77	0.78

TABLE II
ALGORITHMS APPLIED ON TOP 10^4 FEATURES

IX. More Research

We have implemented a number of different classification algorithms, but our results have not improved past 83% regardless of how we changed the parameters for each algorithm. Therefore we propose some obvious extensions to this project which could be implemented at a later time. Firstly, we have been relying on the data which has been provided to us by the HTML parser, but we believe that we might have attained better classification accuracy if we had used the text from the raw HTML files for each website. Secondly, since a lot of numeric features had missing or invalid values, we might have implemented other software to compute those fields for each website and incorporated them into our analysis. Thirdly, instead of relying on classification algorithms being implemented in isolation, it is possible that computing different linear combinations of the outputs of all the algorithms might have given us better accuracy.

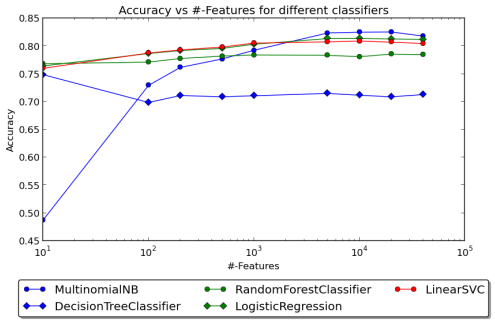

Fig. 5. Feature Selection of the top k features vs Accuracy

X. Conclusion

We have determined that the user classification for this dataset has been heavily biased towards food and recipe related websites, and biased against websites related to news articles, technology and sports. It is possible that this particular group of users believes that recipe websites are more likely to be of interest in the long run. An unexpected trend we noticed was that, in none of our algorithms did cleaning up the text (by removing stop words, spelling errors, stemming, removing duplicate words) improve the prediction accuracy. In fact in some cases the prediction accuracy actually became slightly worse. This could be because latent factors in the original text, such as the stop words and different forms of a word may have influenced a user's opinion of whether that website can be considered evergreen or not.

As an after-thought we generated a word cloud which displays that words related to food, sports and news show up prominently in the data.

Fig. 6. Word Cloud of the top 500 informative words after TF-IDF

Acknowledgment

We would like to thank Victor Minden for his help on python.

References

[1] Thorsten Joachims. 1997. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In Proceedings of the Fourteenth International Conference on Machine Learning (ICML '97), Douglas H. Fisher (Ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 143-151.

[2] Ma Zhanguo; Feng Jing; Chen Liang; Hu Xiangyi; Shi Yanqin, "An Improved Approach to Terms Weighting in Text Classification," Computer and Management (CAMAN), 2011 International Conference on , vol., no., pp.1,4, 19-21 May 2011

[3] Monica Rogati and Yiming Yang. 2002. High-performing feature selection for text classification. In Proceedings of the eleventh international conference on Information and knowledge management (CIKM '02). ACM, New York, NY, USA

[4] Yiming Yang and Jan O. Pedersen. 1997. A Comparative Study on Feature Selection in Text Categorization. In Proceedings of the Fourteenth International Conference on Machine Learning (ICML '97), Douglas H. Fisher (Ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 412-420.

[5] George Forman. 2003. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. 3 (March 2003), 1289-1305.