Classifying Ephemeral vs Evergreen Content on the Web

Gurjot Singh Walia (140020121)

Abhinav Rondi (140050054)

B Srinivas Naik (140050064)

Goutham Arukonda (140050065)

Overview

We can divide the web content information in two broad classes:

- **Newsy, Ephemeral:** This class contains documents that are time sensitive and ephemeral. Typical examples are news articles, classified listings, blog posts about current or local events. Typically these documents receive a big, short lived traffic spike
- **Evergreen:** These are documents that endure the test of time. Typical examples are Well-written, informative articles that tell the users how something works, High quality humor or opinion pieces, Literature, arts and entertainment articles. These can be truly timeless

Problem Statement

The challenge we face in this project is classifying the contents in a website into one of these categories mentioned. This information is valuable piece of cake for web advertising industries as they can use this information as basis to decide the place and time to show their advertisements on the website, to catch up right people at right time. This information can also be used for content sites interested in capacity planning for hosting different pages based on expected longevity.

Goals

- Learning and exploring many new machine learning classification algorithms
- Getting exposed to various interesting features and components of web pages used in predicting user’s interests which form major part of modern day advertising market

The following table includes field descriptions for train.tsv and test.tsv:

FieldName	Type	Description
url	string	Url of the webpage to be classified
urlid	integer	StumbleUpon's unique identifier for each url
boilerplate	json	Boilerplate text
alchemy_category	string	Alchemy category (per the publicly available Alchemy API found at www.alchemyapi.com)
alchemy_category_score	double	Alchemy category score (per the publicly available Alchemy API found at www.alchemyapi.com)
avglinksize	double	Average number of words in each link
commonLinkRatio_1	double	# of links sharing at least 1 word with 1 other links / # of links
commonLinkRatio_2	double	# of links sharing at least 1 word with 2 other links / # of links
commonLinkRatio_3	double	# of links sharing at least 1 word with 3 other links / # of links
commonLinkRatio_4	double	# of links sharing at least 1 word with 4 other links / # of links
compression_ratio	double	Compression achieved on this page via gzip (measure of redundancy)
embed_ratio	double	Count of number of <embed> usage
frameBased	integer (0 or 1)	A page is frame-based (1) if it has no body markup but have a frameset markup
frameTagRatio	double	Ratio of iframe markups over total number of markups
hasDomainLink	integer (0 or 1)	True (1) if it contains an <a> with an url with domain
html_ratio	double	Ratio of tags vs text in the page
image_ratio	double	Ratio of <img> tags vs text in the page
is_news	integer (0 or 1)	True (1) if StumbleUpon's news classifier determines that this webpage is news
lengthyLinkDomain	integer (0 or 1)	True (1) if at least 3 <a> 's text contains more than 30 alphanumeric characters
linkwordscore	double	Percentage of words on the page that are in hyperlink's text
news_front_page	integer (0 or 1)	True (1) if StumbleUpon's news classifier determines that this webpage is front-page news
non_markup_alphanum_characters	integer	Page's text's number of alphanumeric characters
numberOfLinks	integer	Number of <a> markups
numwords_in_url	double	Number of words in url
parametrizedLinkRatio	double	A link is parametrized if it's url contains parameters or has an attached onClick event
spelling_errors_ratio	double	Ratio of words not found in wiki (considered to be a spelling mistake)
label	integer (0 or 1)	User-determined label. Either evergreen (1) or non-evergreen (0); available for train.tsv only

Preprocessing

- The body text, title and header node contents were common English-language words, and were stemmed and lemmatized using Porter stemmer and Wordnet Lemmatizer respectively
- The major parts of dataset that needed preprocessing were URL and boilerplate content of the page in a sample

Preprocessing(contd.)

- **Processing URL of the page:** This includes stripping digits, stop words and stripping words like 'http', 'https', 'www', 'com', 'net', 'org', 'm', 'html', 'htm'. The intermediate form obtained is then further processed by applying stemming algorithms to extract the domain from the URL
- **Preprocessing page content:** This includes processing body and title in the boilerplate separately if either of them are actually present for example considered. Then this data is broken down to finer granularity of sentences and words to which stemming and lemmatizing algorithms are applied

Lemmatization and Stemming

- Lemmatization is closely related to stemming. The difference is that a stemmer operates on a single word without knowledge of the context, and therefore cannot discriminate between words which have different meanings depending on part of speech. However, stemmers are typically easier to implement and run faster, and the may have lower accuracy.
- We computed the term frequencies of all words appearing in the entire training and test set, and discarded the most frequently appearing words by using some threshold on the term frequency. The rationale for this approach is to discard the filler words in the English language (such as “a” or “the”) which have high frequency but little information.

TF-IDF Model

The term frequency-inverse document frequency of each word is used as a feature

The tf-idf is the product of the term frequency, indicating the number of times a word appears in a given document, and the inverse document frequency, which measures how commonly the word appears across all documents

$$idf(t, D) = \log((1+|D|)/(1+|\{d<-D:t<-D\}|)) + 1$$

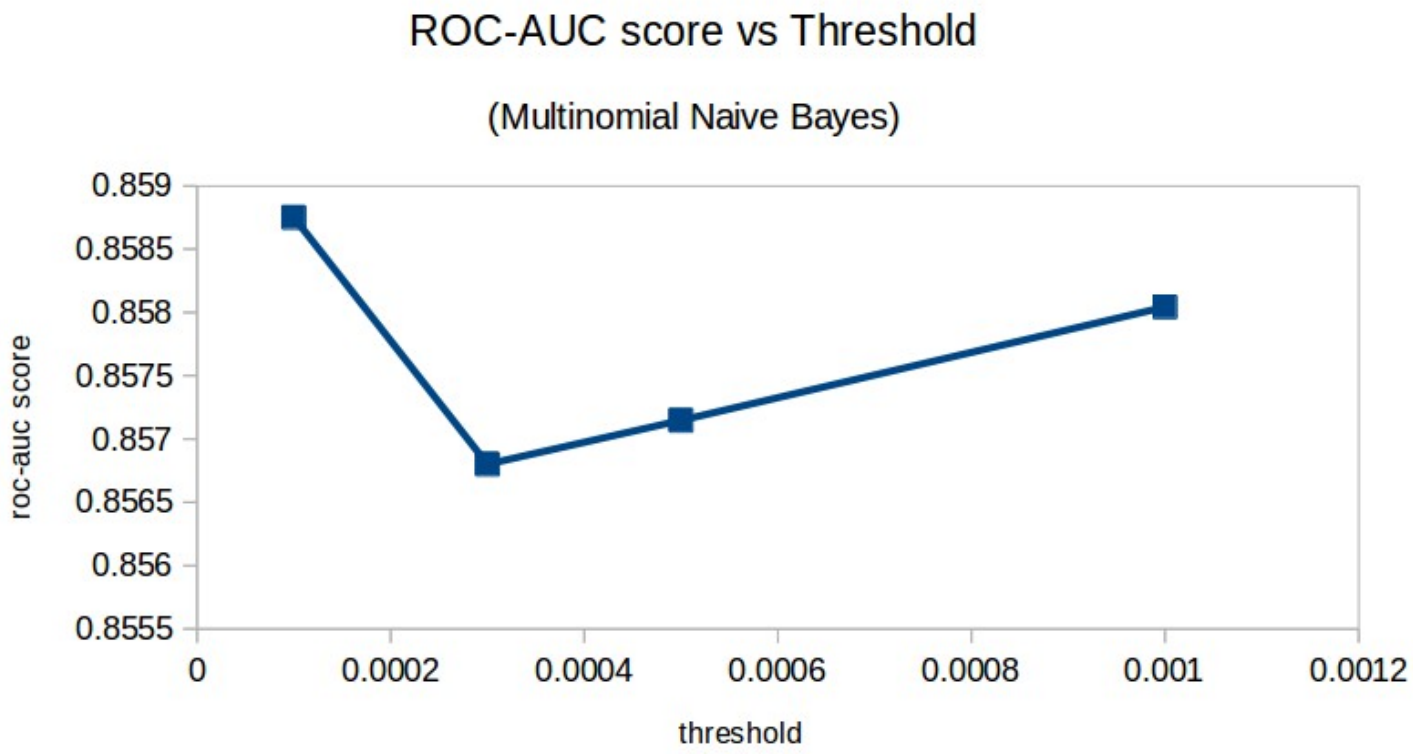
Metrics

The metric used for evaluation is the area under the receiver operating characteristic curve (ROC AUC). The ROC is a characterization of the true positive rate against the false positive rate of a classifier. The true positive rate is also known as sensitivity, recall or probability of detection. The false positive rate is also known as fall-out or (1-specificity). Each prediction result or instance of a confusion matrix represents one point in ROC space. The area under the ROC curve is equal to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. So closer the value of roc-auc to 1, the better is our model.

Classification and Results obtained

Naive Bayesian:

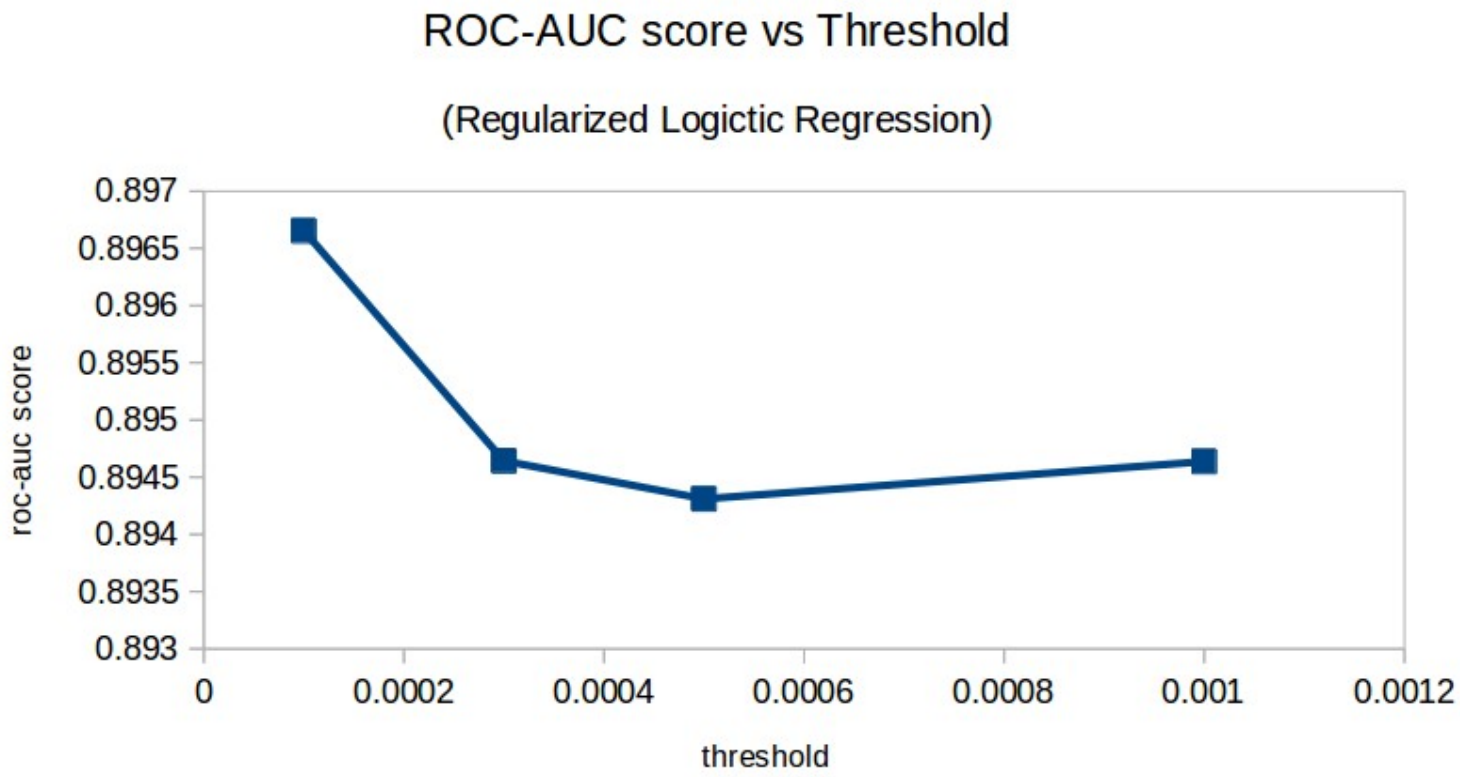
We performed feature selection using TF-IDF and then applied that to the Multinomial Naive Bayes classifier. Figure shows the variation of roc_auc score for different thresholds on fraction of frequency of occurrence of words in dataset.



Classification and Results obtained

Regularized Logistic Regression:

The roc-auc score using regularized logistic regression using 10-fold cross validation is shown here.



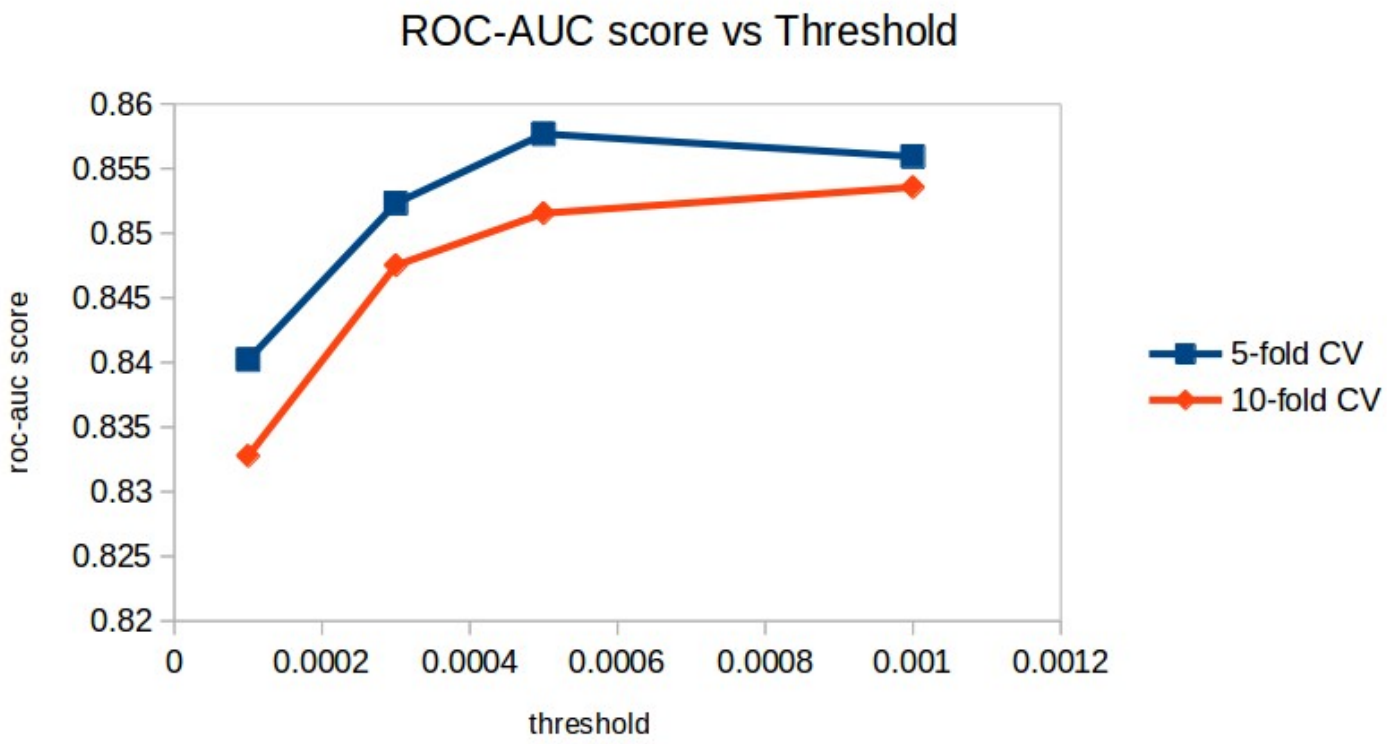
Classification and Results obtained

Support Vector Machines:

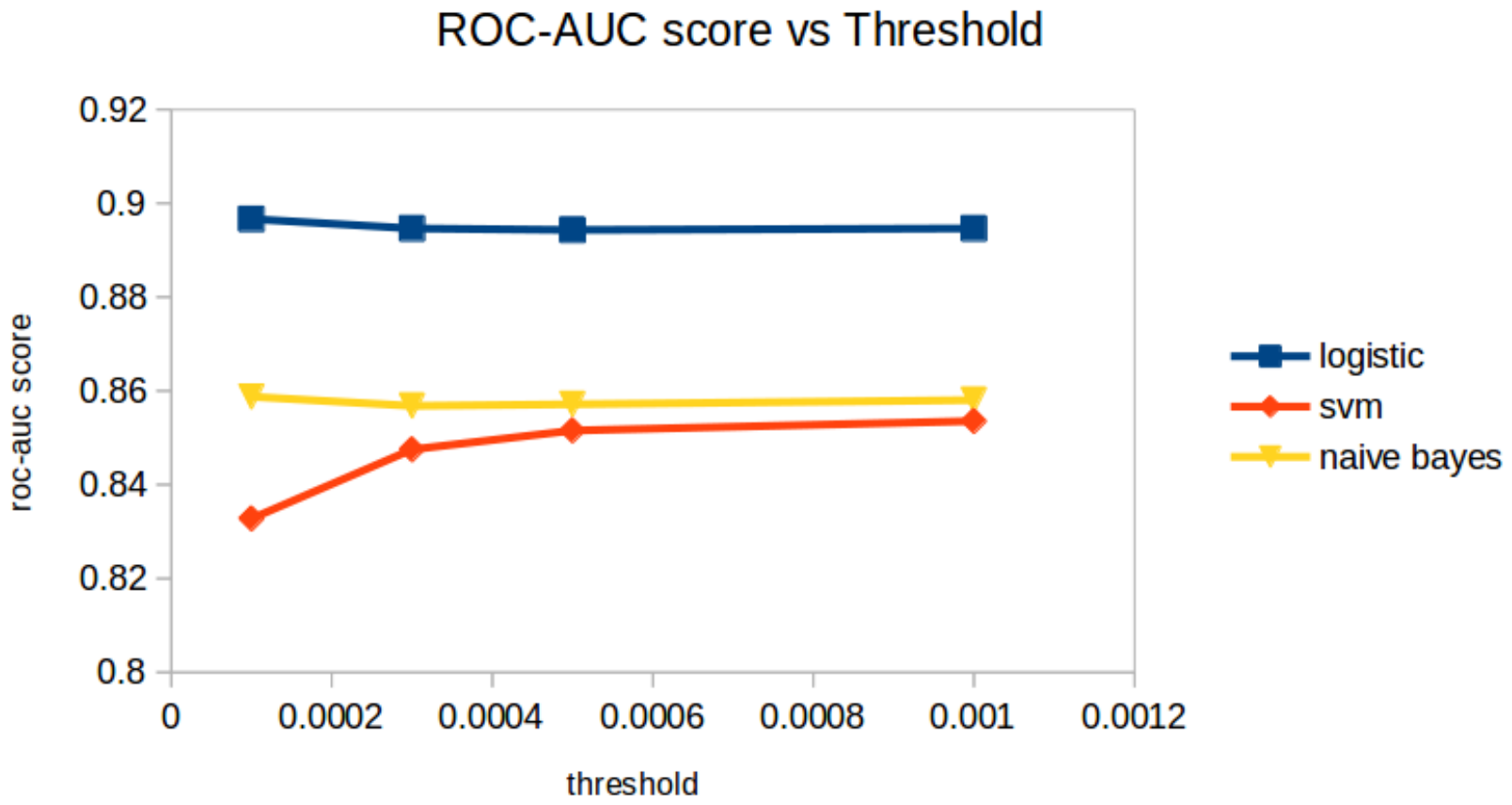
We implemented SVM with both linear and gaussian kernel. We choose C (Penalty parameter of the error term) as 1.0.

Since SVM with gaussian kernel gives bad result as compared to linear SVM in every case for this dataset, the results of gaussian SVM are only included in the figure.

The figure describes variation of roc-auc score as threshold on fraction of frequency of number of words trimmed changes, on 5-fold and 10-fold cross validation using SVM.



Classification and Results obtained (using 10 fold CV)



Conclusions

The web page classification problem is vital to many web-mining applications and we presented a method to effectively solve the Evergreen vs Ephemeral Challenge. 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.

Future Scope

There is clearly some headroom for improving the performance on the Kaggle / StumbleUpon dataset. We have implemented a number of different classification algorithms, but our results have not improved past 87% 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.

- We have been relying on the data which has been provided to us by the HTML parser, but we believe that we might attain better classification accuracy if we had used the text from the raw HTML files for each website.
- 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.

Individual Contribution

- **Pre-midstage:**

Everyone in the team has equally contributed in studying and researching various algorithms to preprocess data and implementing stemming, lemmatization and url cleaning

- **Post-midstage:**

- B Srinivas Naik (140050064): revised mid-stage's source code and completed source code for feature extraction, and classification using regularized logistic regression and SVM, also contributed in documentation and testing
- Goutham Arukonda (140050065): source code for Multinomial Naive Bayesian classification and contributed in documentation and testing
- Abhinav Rondi (140050054): source code for svm, contributed in documentation and testing
- Gurjot Singh Walia (140020121): source code for regularized logistic regression, contributed in documentation and testing

Resources:

1. http://cs229.stanford.edu/proj2013/Chen-ClassifyingEphemeralVsEvergreenContentOnTheWeb.pdf
2. http://cs229.stanford.edu/proj2014/Elaine%20Zhou,%20Lingtong%20Sun,%20Evergreen%20or%20Ephemeral%20-%20Predicting%20Webpage%20Longevity%20Through%20Relevancy%20Features.pdf
3. http://www.cs.toronto.edu/~frank/csc2501/Readings/R2_Porter/Porter-1980.pdf
4. http://www.nltk.org/api/nltk.stem.html
5. http://www.nltk.org/_modules/nltk/tokenize.html
6. http://www.eecs.wsu.edu/~holder/courses/cse6363/spr04/slides/ROC.pdf
7. http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
8. http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html
9. http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html
10. http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html