Coincidence Strength Factor: A Feature Selection Method for Text Classification of Scale-Based Categorization

Mike Huang

Abstract: This paper introduces a feature selection method, called Coincidence Strength Factor (CSF) threshold filtering, for Support Vector Machine (SVM) text classification of scale-based categorization of ratings (eg: 1 to 5 stars rating scale). To reduce the noise introduced by features that coincide in high frequencies between multiple categories, especially ones further away on the rating scale, this noise is quantified by the CSF and filtered above a threshold (CSF>5). This increased the classification accuracy of the SVM text classifier on Yelp reviews from 47.7% without feature selection to 54.9% with feature CSF>5 filtered. Furthermore, the variance of the classification error was reduced from 1.54 to 1.27. While the accuracy leaves much room for improvement, the results show that CSF threshold filtering is an effective feature selection method for improving SVM text classification of scale-based categorization.

Introduction

Is it possible to input a Yelp review into a classifier model and predict its rating? It is likely possible given that a one star review would have distinctive descriptions compared to a five star one. However, there are still many challenges to classifing reviews accurately. First, many descriptions, aka model features, are not unique to a rating, and this would introduce noise into the model. Second, there are many descriptions that are unique to a review and thus it would either be impossible to train a model with those features or those features will never be used for prediction. Third, the raw text of the review needs to be parsed into meaningful phrases, such as bigrams and trigrams, that characterize each category. This paper explores a method of feature selection to *optimize prediction accuracy.* Furthermore, given that the categorization of reviews is on a scale (1 to 5 stars), the margin of error of the prediction category is relevant, ie: a prediction of four stars for a five star review is more useful than a prediction of one star despite both being a misclassification. This paper also aims to *minimize prediction variance.*

This paper introduces a method to quantify the presence of features in proportion between categories that's weighed by the margin of error on the rating prediction. This quantification is referred to as the *Coincidence Strength Factor*, or CSF. Features above a threshold of $CSF > 5$ are filtered out.

Coincidence Strength Factor

For a given term frequency, f, in each each rating, r, it's found in,

There exists $n = \binom{5}{2} = 10$ combinations of pairwise ratings.

Let f_A and f_B be the respective frequencies of a term in each rating in a pairwise rating, eg: frequency of a term found in 1 star and in 2 stars, respectively

Let Δr be the distance of the pairwise rating, eg: 1 star and 3 stars has a distance of 2.

Then the coincidence strength factor is defined as:

$$CSF(f, r) = \sum_{i=1}^{n} \Delta r_i (1 - \frac{|f_{A,i} - f_{B,i}|}{f_{A,i} + f_{B,i}})$$

This would result in possible CSF values from 0 to 20, with 0 being a feature that's unique to one rating only, and 20 being a feature that's found equally across all five ratings.

Example: The bigram, `go back`, is found in all five ratings in the following frequencies

rating	frequency
1	5858
2	6185
3	6324
4	4621
5	3748

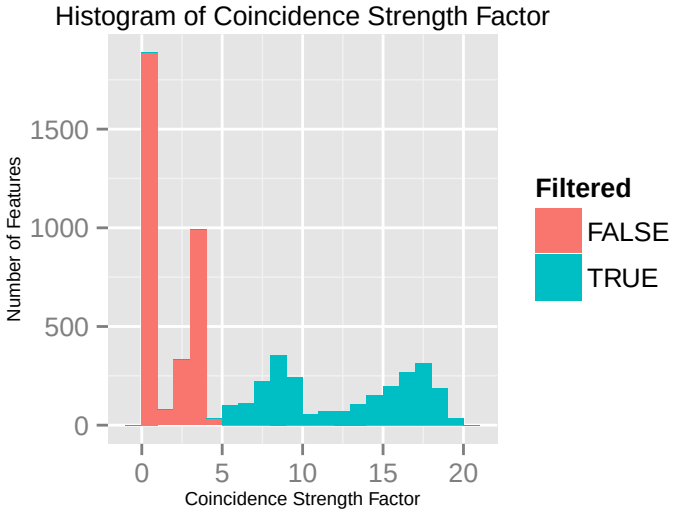
The CSF for each individual pairwise rating is organized into the following matrix of dimensions A*B: $CSF_{A,B} = \Delta r_{A,B}(1 - \frac{|f_A - f_B|}{f_A + f_B})$

```
##       one     two     three   four    five
## one   "-"     "-"     "-"     "-"     "-"
## two   "0.973" "-"     "-"     "-"     "-"
## three "1.92"  "0.989" "-"     "-"     "-"
## four  "2.65"  "1.71"  "0.844" "-"     "-"
## five  "3.12"  "2.26"  "1.49"  "0.896" "-"
```

The final CSF value equals be the sum of all the values in the matrix, 16.852. This is a high CSF value since `go back` is found in nearly even proportions in every category.

CSF>5 Threshold

CSF>5 was chosen as the threshold to be filtered. This would filter out words where the coincidence is high when $\Delta r > 1$. Ideally, filtering words at a lower threshold (ie >1) would further reduce the classification error margin, but practically, this would mean too few features would remain and this would significantly impair the accuracy of the model. A CSF>5 filters out 2437/5515 features or 44.2%. This was found to be a rough sweet spot for accuracy. It's possible that further analysis with ROC curves can optimize this threshold further.



-

Example: The feature, `awesome food`, has a $CSF = 4.60412$. Despite it being present in four ratings, it's highly skewed to a rating of five, thus it's an informative feature.

rating	frequency
1	NA
2	99
3	98
4	238
5	651

Wordcloud of Bigrams

All Features

With all the features present, it's clear that many words are shared between multiple categories in similar frequencies, and thus limiting its relevance in text classification. For example, `go back` and `food good` are prominent in all or nearly all categories, and thus aren't useful features for the model.

One Star

Two Stars

Three Stars

Four Stars

Five Stars

CSF>5 threshold filtered

A feature with `CSF<5` would means it may coincide between neighboring ratings, but no further than that. This would greatly limit the classification error >1. The effects of this is apparent with the bigrams being much more polarized and distinctive to their respective ratings than without the filter.

One Star

Two Stars

Three Stars

Four Stars

Five Stars

Text Classification Algorithms

Three classification algorithms are popular for text classification including Naive Bayes, Support Vector Machine, and Random Forest. According to a study by Joachims 1998, SVM classification outperformed both Naive Bayes and Random Forest in both performance and precision.

CSF threshold filtering is ideal for improving SVM text classification performance

Given that SVM works by finding the vectors that maximally separates the points between the categories, CSF threshold filtering should be ideally suited to increase the performance of the algorithm since it's filtering the features that have strong overlap between categories.

Methods

Text Corpus Preparation

1. The data is obtained from the Yelp Data Science Challenge dataset including data on user, reviews, business, tip, and checkin.
2. The data cleaned so only the restaurant data is retained. Out of these, 400,000 reviews are randomly sampled for training and testing the model.
3. The corpus is built with the text mining package, `tm`. The corpus is transformed to lower case, stopwords containing words that offer no significance (eg. you, her, it's) are filtered out, any punctuation removed.

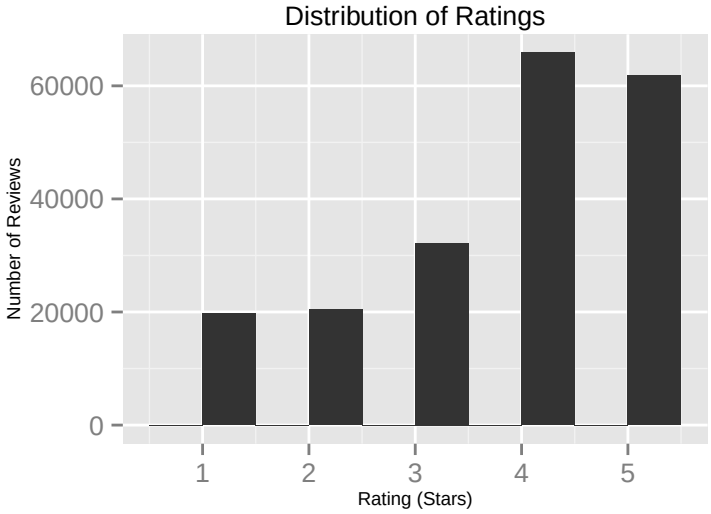
4. The words are organized into bigrams using the package `RWeka`.
5. Corpus is transformed to a Term Document Matrix with dimensions $m*n$, with m dimension representing the documents sampled, and the n dimensions representing the bigram frequency for each respective document.
6. Terms with sparse percentage of above .9992 are removed. This reduces 4,063,604 terms to 5,515. This sparsity results in the most sparse documents to have a minumum sum of 5 count in the features available in the model.

Text Classification

1. The Term Document Matrix was separated into a training and test set with a ratio of 90:10.
2. The `RTextTools` package is used to train the Naive Bayes and SVM classifiers. The Random Forest classifier is sourced from the package `randomForest`.

Oversampling

Given that the distribution of reviews are not even with four stars and five stars being considerably more frequent, reviews from the other ratings are oversampled so that there are an equal amount of samples in each rating. 40,000 samples from each rating is taken, compiled together, and shuffled in random order. This technique was demonstrated to be effective by Hung et al. 2014 to address classification bias towards more frequent rating categores.



Results

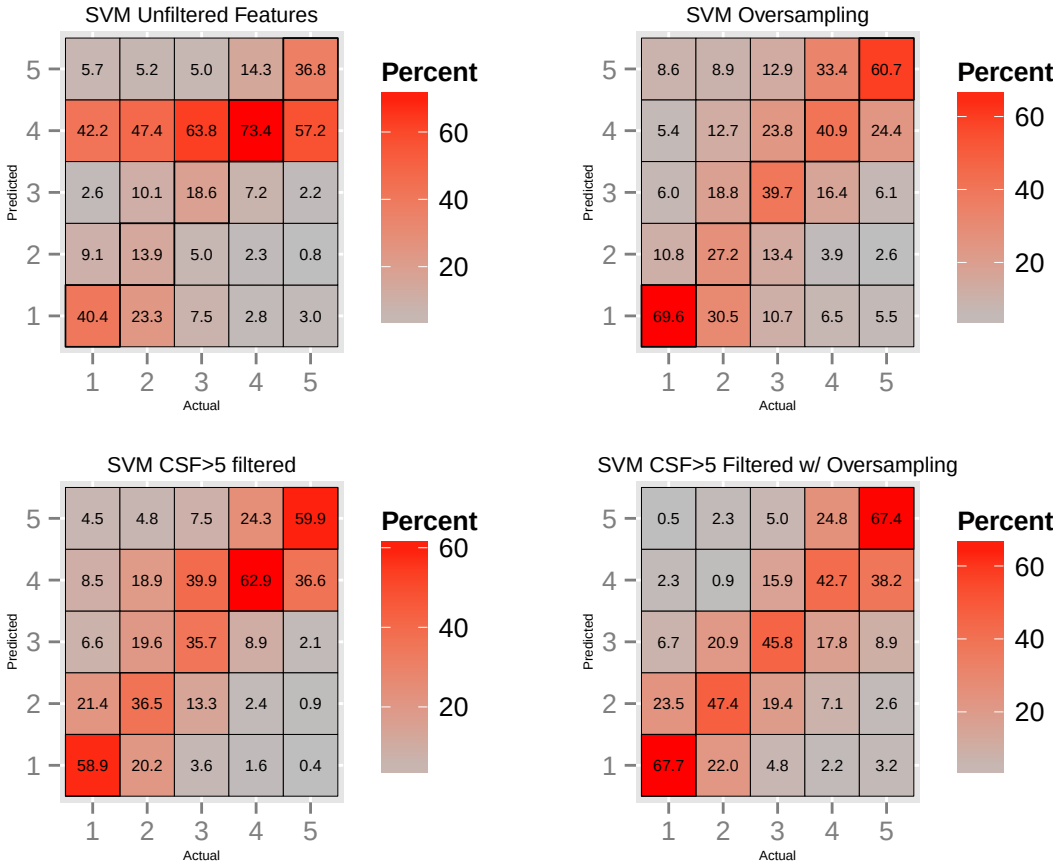
Method	Accuracy	Variance
Naive Bayes all features w/ Oversampling	0.3470368	2.0547833
SVM all features	0.438	1.4370107
SVM all features w/Oversampling	0.4766	1.5395491
SVM CSF>5 filtered	0.538756	1.2517306
SVM CSF>5 filtered w/Oversampling	0.5487078	1.2697095
Random Forest	0.4297	1.6756093

$$Accuracy = \frac{TP}{TP + FP}$$

$$Variance = \frac{1}{N} \sum_{i=1}^{N} |Actual_i - Predicted_i|$$

The SVM classifier performed remarkably better than Naive Bayes and Random Forest. CSF>5 filtering increased the classification accuracy while decreasing the variance of the error.

Confusion Matricies



Support Vector Machine - No Feature Selection

With no feature selection, the model shows a strong classification bias towards a rating of four. This is due to the distribution of ratings being highly skewed to the rating of four in the training sample. To counter this bias, an over sampling of the training set is performed. With oversampling the training set consists of 35000 reviews that are evenly sampled from each of the five ratings (5*7000). This corrected the classification bias towards 4 stars. However, the variance of the classification error is largely unaffected with the error being off by two or more ratings.

Support Vector Machine - CSF>5 Filtered

With the irrelevant features filtered out, the classification accuracy was increased by 23.0% without oversampling and 15.1% with oversampling. While the classification bias towards 4 stars is still present, it is drastically reduced. This is substantial evidence to support the hypothesis that the bias is due to a significantly greater amount of irrelevant features with a $CSF > 5$. With oversampling, the classification error >1 is reduced.

Discussion

While a 54.9% accuracy still leaves much room for improvement, CSF threshold filtering significantly limited the margin of error >1 while improving the accuracy of the SVM classifier.

In a future study, it would be interesting to compare the CSF threshold filtering strategy with other feature selection strategies such as information gain, and chi-squared such as demonstrated in Yang 1997.

References

1. Hung, K., Qui, H., "Yelp Dataset Challenge 2014" 2014 *UCSD Data Science Society* http://kevin11h.github.io/YelpDatasetChallengeDataScienceAndMachineLearningUCSD/

2. Joachims, T., "Text Categorization with Support Vector Machines: Learning with Many Relevant Features" 1998 http://www.cs.cornell.edu/people/tj/publications/joachims_98a.pdf

3. Yang, Y., Pedersen, J.O., "A Comparative Study on Feature Selection on Text Classification" 1997 http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.32.9956&rep=rep1&type=pdf