A Generalized 2-Layer Weighted Ensemble Classifier with Superior Text Sentiment Classification

Abstract

A two layer ensemble classifier for the purpose of generalized sentiment classification was produced using seven different data-sets from four source types–SMS, tweets, product reviews, and movie reviews. Classifiers trained on each data-set were used to create a weight matrix who's combined output were used to train a final classifier in order to increase the aggregate accuracy of polarity-sentiment classification.

Single layer TF-IDF Naive Bayes and SVM Linear Kernel classifiers tend to successfully classify sentiment of data from the same source type on which they are trained, but typically do little better than chance with sentiment analysis of other data-sets. As the English language and human emotion are common to all of these data-sets, we argue that this constitutes a level of overfitting which we improved upon using an ensemble method.

1. Application

The aim of the project was to devise an online sentiment classifier that could be plugged into any environment in which text based conversation was occurring in real time and be able to classify the sentiment of the conversation over time. The original application envisioned would be Facebook messenger, however the aim was to be able to generalize to other applications such as SMS, Twitter, email correspondence, a standalone app on a phone, a customer service hotline, review sites, or online forums. For the purposes of this report, testing data from SMS conversation, tweets, product reviews, and movie reviews are classified.

Using Naive Bayes and Support Vector Machine classifiers for text classification of data-sets into positive, neutral, and negative sentiment classes has been shown to be relatively successful using a bag of words approach when limited to the context in which the classifier was trained on. This approach was constructed and evaluated for testing purposes and is elaborated on in the evaluation section. However, when tested on datasets other than those the classifier was trained on, performance decreased significantly - even with very similar datasets. For example, the SVM trained on SMS dataset A achieved a 69.7% testing accuracy score with a 66.8% F-score. However the same SVM classifier achieved a 31.1% testing accuracy score with a 18.3% F-score when tested on SMS dataset B from the same source. This overfitting and lack of generalizability clearly do not lend themselves well to our intended application.

Now that the classifier has been successfully built and evaluated, the authors intend to continue building on the project and extend the work beyond the scope of the class during their spare time. The two initial applications of the final model are 1) classifying the sentiment of messages on Facebook's Messaging platform using Facebook's messaging API to identify the most negative and alarming of your friends messages and later 2) help companies classify the sentiment of their product on social media, website reviews and support conversations in order to generate a gauge of product performance.

1.1. Motivation

According to psychologist Albert Mehrabian in his book Silent Messages, 93 percent of communication in a face-to-face interaction is actually non-verbal (i.e. body language, tone of voice, facial expressions, etc.)(Mehrabian, 1981). Whether this exact value is accurate or not, it stands that a great deal of information is lost in text based communication whether it be sms texting, tweeting, facebook messaging, email, or other forms of communication. What's more, many people struggle with accurately evaluating friend and partners emotional state - the so called emotional intelligence.

We seek to recover this crucial set of lost information which is a byproduct of communication in the 21st century. Both for social and professional purposes, aiding an individual's ability to gauge the sentiment of those they are interacting with in a variety of contexts has strong positive economic and social health implications.

University of Pennsylvania, CIS 419/519 Course Project.

2. Survey of Related Work

Sentiment classification of text is hardly a novel concept. An early foray into the subject was undertaken by Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan in 2002 using Naive Bayes, maximum entropy, and support vector machines. Pang et. al. were able to achieve 70-80 percent success on the classification of product reviews (Bo Pang & Vaithyanathan, 2002). The intention of this project is to improve on those concepts using newer text classification ideas such as TF-IDF to augment the process for higher accuracy, and apply the process to the more novel area of evaluating mental and emotional health and states of individuals.

A paper published by Tang et. al. speaks to the efficacy of using some of these more recent techniques in improving the accuracy of sentiment classification (Tang et al., 2009). Many of these techniques again are limited in testing scope to very specific types of data and do not generalize well.

Another survey of work in the area of sentiment classification for the purposes of augmenting individual's ability to interpret text based information by Kim et. al reviews recent work to provide deeper analysis than simply negative, neutral, positive text classification (Kim & Zhai, 2011). However, this work is even tighter in scope with classifiers written specifically for analyzing product reviews only and generating summaries, of information within that specific domain. While this domain specific classification is beneficial to those with the resources to develop such deep and specific models for their particular purposes, it does nothing to address the ability to classify sentiment in human language in a broader scope.

The novelty of this projects approach is combining data from such a variety of sources into an intelligent two-layer ensemble classifier that leverages relevance measures for how relevant each individual, granular classifier is to each input datum for weighting of coefficients into the second layer. Thus, given some intelligence about the context in which the ensemble classifier is operating, it will be able to generalize to a broad spectrum of applications of data sources.

3. Approach

3.1. Data

For each iuput data set, both a tf-idf Naive Bayes and a SVM linear kernel classifier were created. Each of these models' predictions were combined with relevance and accuracy scores to determine the weights for the coeffieicients passed into the second layer. The final weighted, two-layer model is generalized for many different types of input data and has superior accuracy to a one-model approach.

The data are collected from a variety of sources to enable a robust ensemble classifier. The sources come from the following institutions, studies, and resources: (Gao & Sebastiani), (John Blitzer, 2007), (Hu & Liu, 2004), (et. al, 2015), (Bing Liu & Cheng, 2005), and (Bo Pang & Vaithyanathan, 2002). The data span across multiple sources including SMS, Twitter, movie reviews, product reviews. The training data are classified into positive, neutral, or negative. Note that some of the data sources strictly classified the training data into either positive or negative.

3.2. Data Processing and Feature Generation

3.3. Algorithm Steps

Part A: The algorithm starts with offline processing of the training data sets. For each training data set, 20 percent of the set is saved for testing purposes. From the remaining 80 percent, 6/7ths is used for the first layer, and 1/7th is used for the second layer training. Two models are created and trained for each data set: 1) a Naive Bayes classifier and 2) a SVM linear kernel.
Part B: Next, a relevance estimate is generated for all combinations of models when used to predict target values for data from a different type (i.e. SMS, Twitter, Product Review, or Movie Review). The relevance is calculated as follows: for model A (trained on data type A), the accuracy (call it A-on-A accuracy) of the model is calculated when tested on itself. Then, model A is used to predict values for training data from data type B. This A-on-B accuracy is compared to the A-on-A accuracy and the percentage difference is calculated. The higher the percentage difference, the lower the relevance score for using models from data type A on data from data type B (or B on A).
Part C: At this point, we start generating the features for the second layer. The first feature is the data type of the input datum. Second, each of the n*2 (where n is the number of data sets) models are used to predict the probabilities for each input datum of being classified in the negative, neutral, and positive classes. Third, each of the classifiers is assigned a weight that is based on 1) its accuracy when trained over its own training data and 2) its relevance to the input datum type as calculated in Part B.
Part D: These input features are then trained on a SVM classifier with an rbf kernel to produce the final predicted class.

4. Results

4.1. Summary Data

Performance Tested Over a Mixed Collection of All Data Types				
Classifier	Average of Accuracy	Average of Precision	Average of Recall	Average of F-score
Ensemble Classifier	**0.659353913**	**0.673589125**	**0.659353913**	**0.645066061**
Movie-Review Naive Bayes	0.485097001	0.423082888	0.485097001	0.440402281
Movie-Review SVM	0.486097511	0.423872071	0.486097511	0.442288145
Product-Review-A Naive Bayes	0.494054465	0.429983151	0.494054465	0.443720837
Product-Review-A SVM	0.491369807	0.443990459	0.491369807	0.448403621
Product-Review-B Naive Bayes	0.478876147	0.417827931	0.478876147	0.424040821
Product-Review-B SVM	0.502604355	0.43459631	0.502604355	0.456248458
SMS-A Naive Bayes	0.495762987	0.42342102	0.495762987	0.444214278
SMS-A SVM	0.495217884	0.420759605	0.495217884	0.444902735
SMS-B Naive Bayes	0.344937988	0.58862545	0.344937988	0.317676573
SMS-B SVM	0.413597434	0.670582519	0.413597434	0.426082345
Twitter-A Naive Bayes	0.254437484	0.130884576	0.254437484	0.1680969
Twitter-A SVM	0.254437484	0.130884576	0.254437484	0.1680969
Twitter-B Naive Bayes	0.276761012	0.398238841	0.276761012	0.227611442
Twitter-B SVM	0.297229494	0.534863273	0.297229494	0.277465871
Grand Total	0.428655664	0.436346786	0.428655664	0.384954485

4.2. Performance

The ensemble classifier does significantly better than every other single-source classifier. Table 4.1 displays the accuracy, precision, recall, and F-score statistics for each classifier tested with a proportional, mixed collection of testing data from each of the data sources used to train the individual classifiers. For the general classification of a sentence, the ensemble classifier has significantly higher accuracy than all other classifiers. The improved performance is due to a few aspects of the machine learning approach.

Investigation of the chart reveals that movie and product review classifiers tend to generalize better for classification of all data types than SMS and Twitter classifiers based on the average statistics in the chart. The accuracy for the generalized testing of the movie and product review granular classifiers is higher than the accuracy of the SMS and Twitter classifiers. This suggests that in future iterations of the generalized model, placing more weight on the movie and product review classifiers could be a worthwhile decision to consider.

4.3. Ensemble

First, setting up an ensemble of classifiers with a variety of data sets promotes diversity and avoids over-fitting of the data. The data sets come from four types: 1) SMS messages 2) Tweets 3) Product Reviews and 4) Movie Reviews. There is a wide variety between the data types with factors including the words that constitute positive, neutral, and negative sentiments and the average length of each datum. An extension of the project (to be discussed more in section 5) is to perform machine learning analysis to classify an input datum as either a SMS, tweet, product review, or movie review.

4.4. Weighted Relevance Features

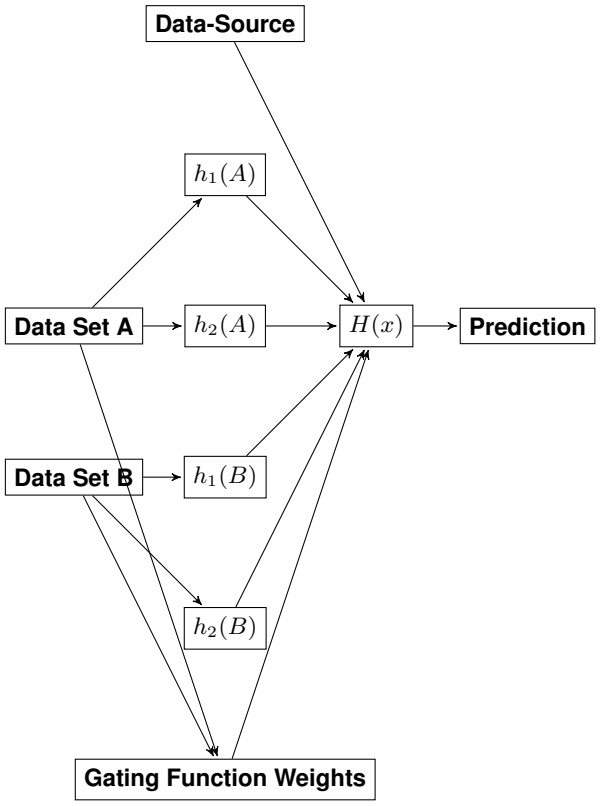
Relevance Scores for Mixes of Classifiers on Different Data Types				
	Movie Review	Product Review	SMS	Tweet
Movie Review	1.000	0.744	0.445	0.218
Product Review		1.000	0.472	0.238
SMS			1.000	0.521
Tweet				1.000

Each classifiers predictions in the first layer are associated with 1) a weight that incorporates the relevance of the classifier to the current testing datum and 2) the accuracy of the classifier when tested on its training data. The relevance scores for the different mixes of data sources are summarized in the table above. For instance, a classifier trained with SMS data and used to predict on movie review data has a relevance score of 0.445. This relevance score for type A vs. type B is calculated by 1) finding the average accuracy of all classifiers from data type A on all data sets of data type A 2) finding the average accuracy of all classifiers from data type A on all data sets of data type B and 3) subtracting the percentage decrease in average accuracy from 1. This relevance calculation enables different classifiers to contribute different amounts

of influence to the final prediction for each datum based on how relevant that classifier is predicted to be for the datum.

4.5. 2nd Layer

The predictions of the first layer are combined with gated weights to produce 51 features for the 2nd layer classifier. The 51 features include (4 data sets that predict negative, neutral, or positive * 2 models each * 3 predicted probabilities) + (3 data sets that predict negative or positive * 2 models each * 2 predicted probabilities) + (7 data sets * 2 models for each data set * 1 weight (composed of relevance and accuracy) for each model) + the input datum type = 51 total input features to the 2nd level. The 2nd layer is trained on a validation set. The existence of the 2nd layer allows the model to incorporate a variety of different features including the predictions of a collection of varied classifiers and the relevance of each of these classifiers. This enables the model to avoid over-fitting biases caused by only have training on one type of data and creates a more robust, general classifier.



5. Future Work

There are two general areas that lend themselves two future work: 1) classifier generalization improvement and 2) application implementation. For classifier generalization, the next logical step in improvement would be to add a layer of classification that attempts to classify the training dataset that the input is most similar to. The reason for this is that in tuning the classifier, we noticed a 5% increase in both accuracy and F-score of the ensemble classifier when the classifier was given an additional feature of the type of data on which it was classifying (e.g. Twitter Data, SMS Data etc.). While a potential user is fully capable of inputting the context in which the classifier is currently classifying, having another layer in which the ensemble classifier uses machine learning to detect the learned dataset to which the input is most similar would make for a more efficient and seamless classifier. For application implementation, the authors had the original intention of plugging directly into the Facebook messenger API to help give a gauge of peers mental health. This would require a few pieces of extension that were discussed, but not implemented. While we did create code to read the inputs of the user in real time and classify them, all of the inputs are classified as unique data points and thus a graph of a users sentiment over time would seem stochastic even though it is likely that a normal individuals emotional state would not change randomly from second to second. Thus a low pass filter on the emotional state of an individual as determined by their sentiment over time would help improve the accuracy of the classification in a real-time context. Also, because we are assuming an individuals emotional state is steadier for smaller time steps between inputs, the time step in-between inputs should be considered. One plausible candidate for doing such a calculation would be to classify an individuals emotional state based on the sentiment of their current input plus the change in their sentiment over time multiplied by some constant.

6. Conclusions

The ensemble method significantly outperforms the individual classifiers because the individual classifiers suffer from over-fitting for one specific data type. The ensemble method takes into account many different classifiers predictions with a weight applied to the relevance and accuracy of each individual classifier. This limits the over-fitting and produces a much more accurate generic classifier.

Acknowledgments

"None."

References

Bing Liu, Minqing Hu and Cheng, Junsheng. Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of the 14th International World Wide Web conference (WWW-2005)*, pp. 168–177, Chiba, Japan, 2005. ACM. ISBN 1-58113-888-1. doi: http://doi.acm.org/10.1145/1014052.1014073.

Bo Pang, Lillian Lee and Vaithyanathan, Shivakumar. Thumbs up? sentiment classification using machine learning techniques, proceedings of emnlp 2002, 2002.

et. al, Kotzias. From group to individual labels using deep features, 2015.

Gao, Wei and Sebastiani, Fabrizio.

Hu, Minqing and Liu, Bing. Mining and summarizing customer reviews. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 168–177, New York, NY, USA, 2004. ACM. ISBN 1-58113-888-1. doi: http://doi.acm.org/10.1145/1014052.1014073.

John Blitzer, Mark Dredze, Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification, 2007.

Kim, H. D., K. Ganesan P. Sondhi and Zhai, C. Comprehensive review of opinion summarization, 2011.

Mehrabian, A. *Silent messages: Implicit communication of emotions and attitudes*. Belmont, CA: Wadsworth (currently distributed by Albert Mehrabian, email: am@kaaj.com), 1981.

Tang, Huifeng, Tan, Songbo, and Cheng, Xueqi. A survey on sentiment detection of reviews. *Expert Syst. Appl.*, 36(7):10760–10773, September 2009. ISSN 0957-4174. doi: 10.1016/j.eswa.2009.02.063. URL http://dx.doi.org/10.1016/j.eswa.2009.02.063.

Algorithm 1 Text Sentiment And Content 2 Layer Ensemble Classification

Input: $numOfDataSets$, $numOfDataTypes$, $inputData$
for $i = 1$ **to** $numOfDataSets$ **do**
  Create both a Naive Bayes classifier and a SVM with linear kernel classmble classifier for the ith data set
end for
for $i = 1$ **to** $numOfDataTypes$ **do**
  for $j = 1$ **to** $numOfDataTypes$ **do**
    Set $curAverageNativeAccuracy$ to the average accuracy for each model from data type i tested on all data sources from data type i.
    Set $curAverageForeignAccuracy$ to the average accuracy for each model from data type i tested on all data sources from data type j.
    Find the percentage decrease between $curAverageNativeAccuracy$ and $curAverageForeignAccuracy$ and set (1 - this percentage decrease) as the relevance for data type i vs. data type j (as well as j vs. i).
  end for
end for
Set the type of the input data as the first feature for the second layer.
for $i = 1$ **to** 2*$numOfDataSets$ **do**
  For the ith classifier, find the probability of negative classification, neutral classification, and positive classification. Set each of these as features for the second layer.
  Set $curWeight$ to (curAccuracy of the ith classifier trained on all data sources from the type of the ith classifier) * (relevance score for the input data type vs. the type of the ith classifier) and set this as a feature for the second layer.
end for
Create a SVM with kbf kernel classifier with all of the second layer features.
Predict the classes of the input data using the second layer classifier.
Return predictions