Soc. Netw. Anal. Min. (2017) 7:37
https://doi.org/10.1007/s13278-017-0456-z

ORIGINAL ARTICLE

Improving detection of untrustworthy online reviews using ensemble learners combined with feature selection

Brian Heredia[1] · **Taghi M. Khoshgoftaar**[1] · **Joseph D. Prusa**[1] · **Michael Crawford**[1]

Received: 27 March 2017 / Revised: 18 July 2017 / Accepted: 20 July 2017 / Published online: 4 August 2017

Abstract As fake reviews become more prominent on the web, a method to differentiate between untruthful and truthful reviews becomes increasingly necessary. However, detection of false reviews may be difficult, as determining the validity of a review based solely on text can be nearly impossible for a human. In this study, we aim to determine the effectiveness of machine learning techniques, specifically ensemble techniques and the combination of feature selection and ensemble techniques, for the detection of spam reviews. In addition to traditional ensemble techniques, such as Boosting and Bagging, we employ techniques that combine ensemble methods with a form of feature selection: Select-Boost, Select-Bagging and Random Forest. For Select-Boost and Select-Bagging, we combine the Boosting and Bagging approaches with three different feature rankers. Random Forest was performed using 100, 250, and 500 trees. Our results show a combination of Select-Boost, multinomial naïve Bayes and, either Chi-squared or signal-to-noise, significantly outperforms all methods except Random Forest using 500 trees. There is no significant difference between the feature subset sizes tested when using Select-Boost with multinomial naïve Bayes, regardless of the feature selection technique employed. To the best of our knowledge, this is the first study to examine the effect of a combination of ensemble techniques and feature selection in the domain of spam review detection.

Keywords Spam review · Ensemble · Random Forest · Select-Boost · Select-Bagging · Feature selection

✉ Brian Heredia
bheredia@fau.edu

[1] Florida Atlantic University, Boca Raton, FL, USA

1 Introduction

People rely on consumer reviews and recommendations for insight when purchasing products, trying new restaurants, finding a primary care physician, and other goods and services. In the past, consumer reviews and recommendations were passed via word of mouth or publications, but with the ubiquitous nature of the Internet, more people are obtaining reviews and recommendations via the web. Google, Yelp, and Amazon are a few of the websites loaded with user reviews on a multitude of topics. Reviews can be created by anyone who has access to the web. This open access leads to one of the main issues with online reviews—spam.

Spam reviews can be classified into three categories: untruthful reviews, reviews on brands, and non-reviews (Dixit and Agrawal 2013). Untruthful reviews are fake reviews created with malicious intent. Reviews on brands are reviews not specific to a product, but the entity creating the product. Non-reviews contain all reviews that do not actually review a product. This research focuses on the detection of untruthful reviews, as these have been shown to be the most difficult to detect (Jindal and Lui 2008).

Spam reviews are created by "spammers" or suspicious persons to either harm or bolster the reputation of an establishment or product. As these reviews are all public, usually posted by an unknown user, and not passed via word of mouth, it becomes increasingly difficult to determine whether the review is coming from a reputable source. Up to one-third of all online reviews are considered untruthful; this increase in false reviews has led to a drop in credibility of online reviews (I.C. Government of Canada 2014). To combat this issue, many methods using machine learning tools, specifically supervised learning, have been proposed.

Supervised machine learning algorithms are trained on labeled data sets, in our case spam review data sets. These algorithms learn to classify reviews as truthful or untruthful based on features describing the reviews. These labeled data sets contain instances (the reviews), features describing the review, and their corresponding class label. The features describing a review typically include review text features, the meta data found in the review (rating, date, time, etc.), and reviewer-oriented features, which describe the user that created the review. Since we are interested in predicting spam reviews based only on the text, we only explore the effects of text-based features and choose to leave other features for future work.

Unfortunately, due to the text-based nature of reviews, the feature space associated with a data set in the domain of spam detection is large. Having a large feature space is commonly known as high dimensionality and creates challenges, such as higher computational costs and redundant or useless features (Haykin 1998). Classification algorithms are not adept at handling a high dimensional feature space and will see degraded performance due to overfitting (Crawford et al. 2016). A family of machine learning techniques, known as feature selection, can be implemented to combat the problem of high dimensionality. Feature selection has only recently been employed in spam review detection studies (Crawford et al. 2016; Mukherjee et al. 2013).

In addition to feature selection, a family of techniques called ensemble learning techniques have been shown to increase classifier performance and reduce overfitting (Dietterich 2000). In previous work, we have found ensemble techniques, such as Bagging and Boosting, increase classification performance in other text classification domains (Prusa et al. 2015). However, ensemble techniques alone do not combat the issue of high dimensionality (Heredia et al. 2016). To combat high dimensionality, while still taking advantage of the ensemble framework, feature selection can be embedded within ensemble techniques. We employed three of these techniques: Select-Boost, Select-Bagging, and Random Forest. These three techniques combine feature selection and ensemble learning to form a classifier which is resilient to the adverse effects brought upon by high dimensionality. We build upon the results found in Crawford et al. (2016) by examining ensemble learners, which are known to improve performance in a variety of domains, and determine how the addition of feature selection to ensemble algorithms impacts classification performance. To the best of our knowledge, this is the first study to employ the combination of ensemble and feature selection techniques in the domain of spam review detection.

In this study, we present results for feature selection and aim to determine the impact of ensemble techniques and the combination of feature selection and ensemble techniques on spam review detection. We discuss results for a total of ten feature selection techniques across five classifiers to demonstrate the effects of feature selection. We explore the effectiveness of Boosting and Bagging when identifying review spam. Our Select-Boost and Select-Bagging algorithms are implemented using five base classifiers: multinomial naïve Bayes (MNB), naïve Bayes (NB), support vector machines (SVM), logistic regression (LR), and C4.5 Decision Tree (C4.5). We embed signal-to-noise (S2N), Chi-Squared (CS), and Mutual Information (MI) within the Select-Boost and Select-Bagging approaches. Random Forest is conducted with three tree sizes: 100 (RF100), 250 (RF250), and 500 (RF500). All models are evaluated using four runs of fivefold cross-validation with model performance measured using the area under the receiver operator characteristic curve (AUC) metric. Our study is unique in that we provide a comprehensive overview of feature selection, ensemble methods, and the combinations of both in spam review detection.

The highest performing classifier, using no ensemble techniques, for spam review detection is multinomial naïve Bayes (Crawford et al. 2016; Heredia et al. 2016). When using MNB, no significant changes in performance occur with the addition of an ensemble technique (Bagging or Boosting). Moreover, the inclusion of feature selection when training MNB results in a lower AUC score for smaller subset sizes. However, applying a combination of ensemble and feature selection (Select-Boost) shows significant improvement in performance when compared to using solely MNB. Our results indicate the combination of Select-Boost, MNB, and Chi-Squared (or signal-to-noise) to be the best performing model, significantly outperforming all other methods, with the exception of RF500. Select-Boost does produce a higher AUC score than RF500; however, the difference between the two is not significant. The Select-Boost framework combines re-weighting and feature selection in each iteration, which is fundamental for improving model performance. Our results are tested for significance using ANalysis Of VAriance (ANOVA) and Tukey's honestly significant difference (HSD) tests (Berenson et al. 1983).

The remainder of the paper is organized as follows. Section 2 contains works related to spam detection, ensemble learning, and feature selection. Section 3 provides insight into the feature selection techniques used. Section 4 describes ensemble techniques, such as Boosting and Bagging. In Sect. 5, we examine techniques that combine feature selection and ensemble learning: Select-Boost, Select-Bagging, and RF. Section 6 summarizes our methodology including data set information, learners, cross-validation, and the performance metric. Section 7 contains the first case study, which presents baseline results

for each base learner and the effectiveness of feature selection. In Sect. 8, we present our second case study, discussing the results for ensemble techniques and the combination of ensemble and feature selection. Section 9 provides a discussion of the results. Finally, in Sect. 10, we present our final conclusions and possible avenues for future work.

2 Related work

The area of spam review detection includes various studies dedicated to detecting untruthful reviews using supervised learning. There exist multiple types of feature sets which may be used for detection of untruthful reviews (Crawford et al. 2015). However, one of the main problems with spam review detection is labeled data set availability. A study by Jindal and Lui (2008) used review-oriented features to predict whether a review was classified as spam or not spam. A data set was generated by collecting 5.8 million reviews of products offered by Amazon. A subset of 222,000 instances were sampled from the original data set and a method known as w-shingling (Broder 1997) was utilized to detect near-duplicate reviews. The near-duplicate reviews were classified as untruthful reviews. From these reviews, 36 additional features were derived and used in a logistic regression model to predict spam reviews. These additional features consisted of Part Of Speech (POS) and reviewer-centric features. Using the full feature set, their model resulted in an AUC of 0.78 when detecting duplicate reviews. The data set was tested against two other review categories: brand only and non-reviews. For brand only and non-reviews, the resulting AUC values were over 0.98, indicating the detection of untruthful reviews to be a more difficult task.

A study by Ott et al. (2011) proposed a data set of considerably smaller size with an equal number of positive (spam) and negative (non-spam) instances. The authors put forth a data set created using Amazon Mechanical Turk (AMT) (Buhrmester and Goslining 2011). AMT is a service provided by Amazon that allows access to a scalable work force. By leveraging AMT, the authors were able to generate fake positive reviews for hotels. The data set consists of 800 positive reviews with half the positive reviews being false, while the remaining half was obtained from TripAdvisor. The best model created had an accuracy measure of 89.8% using a support vector machine (SVM) classifier with a combination of Linguistic Inquiry and Word Count (LIWC) and bigram features. Although this data set has been used in many studies, it was shown to have a major flaw in that it is not representative of real-world review scenarios. The word distributions found in the AMT reviews are significantly different from distributions found in truthful reviews, allowing for easier detection of false reviews (Mukherjee et al. 2013). A more realistic data set was proposed by Li et al. (2014) which extends the AMT data set created by Ott et'al. (2011) through the addition of expert reviews and by combining reviews from three different domains: hotels, restaurants, and doctors. To create a more comprehensive and accurate real-world data set, Li et al. (2014) obtained reviews from employees of the locations being reviewed in addition to reviews from AMT. The authors elected to use the hotel reviews to train the model and the doctor reviews to test the model. Using a classification algorithm modeled off the Sparse Additive Generic Model (SAGE), they were able to obtain an accuracy of 64.7 and 63.4% when using LIWC and POS features, respectively. We elected to use this data set in our study, since it most closely represents real-world review scenarios.

In Shojaee et al. (2013), stylometric attributes were used for detection of untruthful reviews. Stylometric features consist of lexical and syntactic features. These include features which give an indication of the grammar and vocabulary used by the writer. Lexical features focus on word or character use, while syntactic features represent the style of the writer, such as their use of the words "the", "of", "a", etc. Three sets of experiments were performed using the data set from Ott et'al. (2011). The first set used solely lexical features, the second used syntactic features, and the final set used a combination of both. Two classifiers were trained with each feature set: SVM and naïve Bayes. The classifiers were trained using tenfold cross-validation. In all three feature sets, SVM outperformed naïve Bayes in terms of F-measure. The best F-measure, 0.84, was obtained using the feature set containing the combination of lexical and syntactic features. The increase in feature space, due to the addition of stylometric features, requires higher computational costs and no feature selection was done to optimize the feature space.

A study by Mukherjee et al. (2013) used reviewer-centric features to detect spam reviews. The data set used was a combination of the data set found in Ott et'al. (2011) and reviews collected from Yelp. In total, there were three feature sets, one consisting of LIWC features, one consisting of POS features, and the final set contained bigram features. The study applied feature selection using Information Gain (IG) to select the top 1 and 2% of features from each set. The models created using the reduced feature space did not generalize well to real-world data. Feature selection was found to offer no improvement on classification performance. However, only a single feature selection technique was tested; thus, no conclusive results can be drawn from the effect of feature selection on the classification of spam reviews. A study by Crawford et al. (2016) determined the effects of ten feature rankers on

classification of spam review. Ten filter-based feature selection techniques were applied across five classifiers and a range of subset sizes. Out of the ten feature selection techniques, Chi-Squared was chosen for comparison against a word frequency feature selection method. Results show multinomial naïve Bayes has the highest performance and Chi-Squared performs better than word frequency at low subset sizes across all classifiers. However, at large subset sizes, word frequency performs just as well as Chi-Squared, if not better.

Ensemble techniques have largely gone unused in spam review detection studies in exchange for a more traditional supervised learning approaches. Our study is the first to apply a wide range of ensemble techniques in the area of spam review detection. Ensemble techniques may be useful for detecting untruthful reviews as they have been shown to increase performance in noisy data (Dietterich 2000). There is evidence of the effects of ensemble techniques on classifier performance in related domains such as sentiment detection (Prusa et al. 2015). A previous study on tweet sentiment classification showed the improved effects of employing ensemble classifiers with feature selection (Prusa et al. 2015). The study compares the performance of Select-Boost and Select-Bagging against various feature selection techniques using five base learners: K-Nearest Neighbor (KNN), C4.5 decision tree, multilayer perceptron (MLP) and logistic regression (LR). Two tweet sentiment data sets were used: SemEval and Sentiment140. A Threshold-Based Feature Selection (TBFS) technique, and the area under the Receiver Operator Characteristic (ROC) curve metric was employed within Select-Boost and Select-Bagging. Results showed Select-Boost performing significantly better than Select-Bagging and plain classification on both tweet sentiment data sets.

Our study is unique since it explores the effects of Boosting, Bagging, Select-Boost, Select-Bagging, and Random Forest in the spam review domain. Moreover, no other study has examined the effect of Boosting and Bagging in this domain. We also compare the classification performance of these techniques against feature selection and the base learners. To the best of our knowledge, this is the first study to present a thorough comparison of these techniques in the realm of spam review detection.

3 Feature selection

Feature selection can be performed using wrapper-based, filter-based, embedded or hybrid techniques. Wrapper-based techniques use a classifier to rank the performance of a subset of features. Wrapper-based techniques do not scale well with a high feature space, due to their computationally expensive approach. Filter-based techniques can be categorized into two types: filter-based subset evaluation and filter-based ranking. Filter-based subset evaluation techniques attempt to find the subset of features which maximizes classification performance. Usually, this process uses a greedy approach, where a feature set is created by adding the initial best feature that discriminates between the classes (or starting with all features and removing the feature which discriminates the least between the classes). Features are then added to the subset individually and tested for performance, the feature with the highest increase in performance is added to the subset. This process is repeated until the best subset of features is found. Similar to wrapper-based techniques, filter-based subset evaluation does not scale well with very high dimensional data as it is computationally expensive.

Unlike the previous two approaches, feature rankers provide the type of fast and scalable feature selection suitable for very large feature spaces. They use various measures to rank features based on performance. Features are given a score relative to the class label based on the performance metric used. In this study, filter-based feature rankers were employed, as they scale well with high dimensional data sets and have been previously used in detection of untruthful reviews (Crawford et al. 2016). We applied ten different feature rankers to the data set from three different feature selection families: commonly used (Dittman et al. 2010), Threshold-Based Feature Selection (TBFS) (Dittman et al. 2010), and first-order statistics (FOS) (Khoshgoftaar et al. 2012). These families use distinctly different methods to rank features. The following three subsections describe the feature ranker techniques used and the families they belong to.

3.1 Commonly used techniques

This family of commonly used techniques consists of techniques that are widely used and easily accessible. These techniques are found in most open-source machine learning tool-kits, such as the Weka toolkit (Hall et al. 2009). From this family of techniques, we employ the Chi-Squared (CS) ranker (Witten and Frank 2005).

Chi-Squared utilizes the Chi-Squared statistic to measure the relationship between a feature and the class. This feature ranker measures the independence of two events, in this case the events are the occurrence of the feature and the occurrence of a specific class. If we consider the occurrence of a feature as A and the occurrence of a class as B, then these occurrences are independent if $P(AB) = P(A) * P(B)$. The formula below shows the common form of the Chi-Squared ranker:

$$\chi^2 = \sum_{k=1}^{n} \frac{(O_k - E_k)^2}{E_k}$$

where O is the observed value and E is the expected value. Thus, a higher Chi-Squared value indicates some form of dependence between the feature and the class, whether positive or negative.

3.2 Threshold-based feature selection techniques

TBFS techniques are bi-variate procedures, where each feature is evaluated against the class, independent of all other features (Dittman et al. 2010). All attributes are first normalized in a range [0,1], then a classifier is built for all thresholds $t \in [0, 1]$. Two classification rules are used for building the simple classifiers. Rule 1 classifies instances with a normalized value greater than t as positive, and examples with a normalized value less than t as negative. Rule 2 is the converse of rule 1 and classifies instances with a normalized value greater than t as negative, while examples with a normalized value less than t are positive. The metric ω is used to create the attribute ranking. These ω metrics are primarily used to measure classification performance (Dittman et al. 2010). We use Mutual Information (MI), the area under the Precision–Recall Curve (PRC), the area under the receiver operator characteristics curve (ROC), Gini-Index, Kolmogorov–Smirnov (KS), significance analysis of microarrays (SAM) and probability ratio as our ω metrics used to rank the features.

Mutual Information determines the importance of a feature by measuring the contribution of the presence, or absence, of a feature toward a correct classification (Peng et al. 2005). The formula below describes mutual information:

$$I(W, C) = \sum_{e_W \in [0,1]} \sum_{e_C \in [0,1]} p(e_W, e_C) \log \frac{p(e_W, e_C)}{p(e_W)p(e_C)}$$

where W is the word and C is the class. This formula can be simplified to $I(X, Y) = H(Y) - H(Y|X)$ or $I(X, Y) = H(X) - H(X|Y)$, where $H(X)$ and $H(Y)$ are marginal entropy values and $H(X|Y)$ and $H(Y|X)$ are conditional entropy values. $H(X|Y)$ and $H(Y|X)$ are values that indicate the amount of uncertainty left after $H(X)$ or $H(Y)$ are learned.

The PRC metric measures the area under the curve of Recall on the x-axis and Precision on the y-axis. Recall is defined as the True Positive Rate (TPR) and Precision is defined as the fraction of instances classified as positive that are actually positive.

The ROC metric measures the area under the curve of False Positive Rate (FPR) on the x-axis and TPR on the y-axis. This directly indicates model performance, where the higher the area under the ROC curve the better the model performance across all thresholds.

Probability Ratio was used for text classification in Forman (2003) and was defined as the probability of the feature given the positive class divided by the sample estimate probability of the feature given the negative class. The formula (TPR/FPR) describes the probability ratio between the positive and negative classes.

The Gini-Index was originally utilized for measuring income over a population. The Gini-Index has a range from 0 to 1, which indicates the following: GI = 0 implies the income is split evenly across the population, and GI = 1 means that a single person receives all the income. The feature ranker version of the Gini-Index analyzes the distribution of a feature across the classes.

The KS statistic is a nonparametric test that measures the difference in the cumulative distribution of two data sets.

SAM was originally created for detecting significance in microarrays within the bioinformatics domain. SAM compares the observed and expected values of the parameter to identify features whose associated values differ by an amount of statistical significance among the sets (Tusher et al. 2001).

3.3 First-order statistics techniques

The FOS family of techniques are uni-variate feature rankers that use first-order statistic measures, such as mean, mode and standard deviation, to rank features in order of importance. In our experiments, two feature rankers from this family are used: signal-to-noise (S2N) (Chen and Wasikowski 2008) and Wilcoxon Rank Sum (WRS) (Breitling and Herzyk 2005).

The signal-to-noise ratio represents the ratio of signal information to noisy background information. This process can be used to determine how well a feature discriminates between two classes. The formula for S2N is:

$$S2N = \frac{(\mu_P - \mu_N)}{(\sigma_P + \sigma_N)}$$

where μ_P is the mean of the positive class, μ_N is the mean of the negative class, σ_P is the standard deviation of the positive class, and σ_N is the standard deviation of the negative class.

The Wilcoxon Rank Sum is a variation of the standard t-statistic. In a standard t-statistic, the distribution is assumed to be normal, while in the WRS no assumption on the distribution is made. There are two steps to be performed before defining the WRS: (1) all instances are ranked based on the value of the feature, and (2) all the rankings in the positive class are summed as W_p. The formula for WRS is:

$$\mathrm{WRS} = \frac{\left(W_p - \frac{n_p(n_p+1)}{2}\right) - \frac{n_p n_n}{2}}{\sqrt{\frac{n_p n_n (n_p + n_n + 1)}{12}}}$$

4 Ensemble techniques

Two different ensemble algorithms are used in our study: Boosting and Bagging. These techniques are similar in that they combine multiple instances of a base learner to generate a more robust and generalized classifier; however, the process by which this is achieved is different. Boosting takes an iterative approach; training and evaluating a model using a classifier, then training and evaluating subsequent classifiers based on the results of the previous classification (Freund and Schapire 1996). In our study, we use the AdaBoost family of techniques, specifically AdaBoost.M1. AdaBoost works by training a base learner then re-weighting the misclassified instances and repeating this process every iteration. Therefore, the subsequent iterations have more focus on the classification of previously misclassified instances, due to the weights. All initial weights are set to one. The algorithm runs for a predetermined number of iterations, then aggregates the results of all models to form a final decision. In our study, ten boosting iterations are performed and model results are aggregated using majority voting. Due to the re-weighting step, Boosting is not compatible with certain base learners; however, a separate approach, where the instances are re-sampled from the original data set according to the weights, allows these base learners to be compatible with Boosting. This data sampling approach is applied to re-select instances in accordance with the assigned weights.

Bagging, also known as bootstrap aggregating, uses data sampling with replacement (re-sampling from the original data set where the same instances can be re-selected) to create a predefined number of bootstrap data sets. In our study, ten bootstrap data sets are created. These new bootstrap data sets are the same size as the original and contain instances from the original data set. The bootstrap data sets are then used to train a single base classifier each. Once all the classifiers have been trained an aggregation technique, in our case majority voting, is used to determine the final classification. Bagging has been shown to increase performance of weak classifiers but adversely affect the performance of stable learners (Breiman 1996).

5 Ensemble techniques with feature selection

In addition to Boosting and Bagging, we want to examine the effects of ensemble techniques that take advantage of feature selection. Three of these techniques are used in our study and explained in this section: Select-Boost, Select-Bagging, and Random Forest.

Select-Boost is a modified version of the AdaBoost.M1 algorithm developed by our research team. The Select-Boost (Prusa et al. 2015) framework embeds feature selection within the Boosting ensemble framework. As with Boosting, Select-Boost is also incompatible with certain base learners due to the re-weighting step. To mitigate this, we re-sample from the original data set according to the weights assigned and create a new training data set where the probability of selecting an instance is equal to its weight. Initially, all samples in the training data are assigned equal weights. The data is then sampled from the original data set. Feature selection is applied and classifiers are trained on the reduced feature space. The weights are then updated based on misclassified instances and the Select-Boost algorithm begins anew. As with Boosting, we use ten iterations of the Select-Boost framework. Figure 1 depicts the Select-Boosting framework. The dotted line represents the path the framework takes once all iterations have finished.

Select-Bagging (Prusa et al. 2015), like Select-Boosting, adds a feature selection step to the Bagging algorithm. The Select-Bagging framework first creates a predefined number of bootstrap data sets from the original through sampling with replacement. As with Bagging, ten bootstrap data sets are created for Select-Bagging. Feature Selection is applied to every bootstrap data set and then each data set is used to train a classifier. The results of those trained classifiers are aggregated and a final classification is made through majority voting. Figure 2 shows the Select-Bagging framework.

6 General methodology

This section describes the general methodology for our experiments with ensemble and feature selection techniques. We describe the data set, the base learners, cross-validation, and the performance metric.

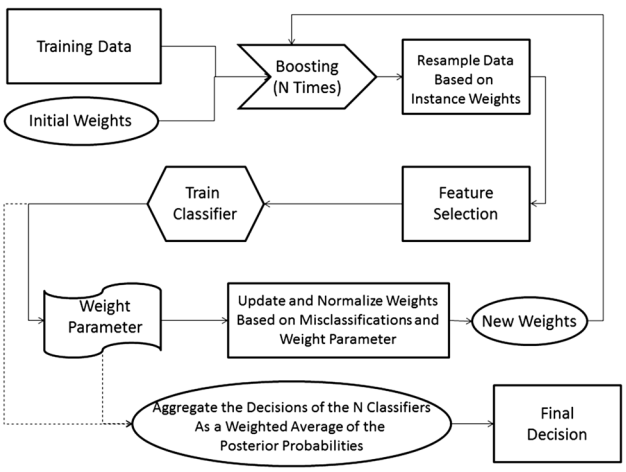


Fig. 1 Select-Boosting framework

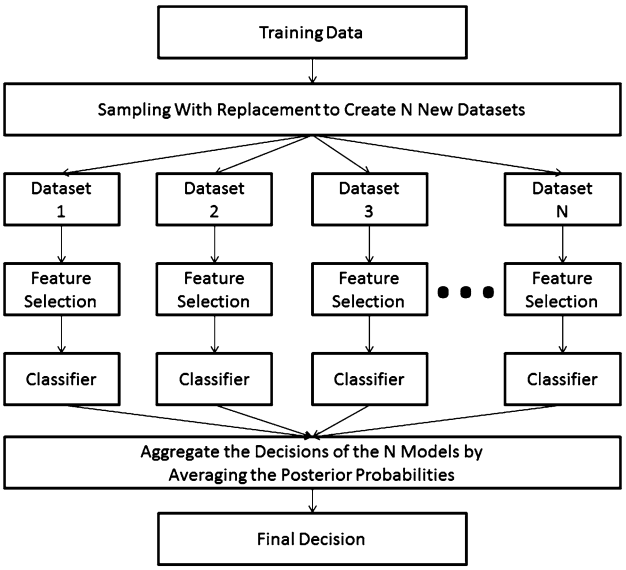


Fig. 2 Select-Bagging framework

6.1 Data set

Our data set originated from the study by Li et al. (2014). The data set contains spam reviews from three distinct domains: doctors, hotels, and restaurants. The original data set was obtained through as similar process as used in Ott et'al. (2011), using AMT. AMT (Buhrmester and Goslining 2011) is a service provided by Amazon which gives customers access to an on-demand, scalable workforce. This workforce was employed to create fake reviews using real reviews as a baseline. As AMT uses regular workers, whom may have no knowledge of domain keywords, the word distributions found in the reviews are significantly different from word distributions found in real reviews (Mukherjee et al. 2013). These synthetic reviews are not always representative of real-world spam data; to counteract this, domain experts were hired to create false reviews for their specific domains. For example, an employee in a doctor's office was asked to write false reviews for their practice. Li et al. (2014) posited employees would have a better grasp on daily processes which would lead to a better synthetic spam reviews. Two examples of reviews from the data set can be found below. From a human standpoint, determining which is spam between the two from reading the text alone is near impossible. The following review is an untruthful spam review:

> The rates at The Talbott Hotel were cheaper than I had expected, and that was my reason for booking a room. I had been prepared for service and a room similar to what I had experienced in the past, and I was quite pleased when I did stay. The room was neat and clean, and the halls were quiet at night. The traffic noise was muffled to the point where it was no problem sleeping either. I did ask one question at the service desk and they answered it nicely, which is good because normally hotel workers can be a bit snippy, especially at night. Overall I had no problems with The Talbott Hotel and I would stay at this here again if I were in the area a second time.

The following review is a truthful review, created by a guest who stayed in the hotel:

> My husband & I stayed at the Fitzpatrick in early June 2004 for my birthday-great hotel! Location provides easy walking to Navy Pier, Marshall Field, Michigan Avenue & John Hancock. Room seemed spacious even though its only about 300 sq ft. Lots of room in bathroom; comfortable bed; very quiet, upscale hotel. We would definitely stay here again!

The data set characteristics can be found in Table 1. The domain of review spam detection suffers from a lack of data sets representative of real-world scenarios. To the best of our knowledge, this data set is the closest publicly available text-based spam review data set representative of real-world data. The data set contains the text found in the review, the rating given, the domain the review belongs to, and the class label. For our purposes, the rating and domain information are not required as we are interested in using only text; thus, they are removed from the data set. To create the feature set, the StringToWordVector filter in Weka is used. While StringToWordVector can output a variety of word vector representations, we elect to use a bag-of-words representation over TF-IDF as preliminary studies have found it to be more effective. The StringToWordVector filter in Weka returns the number of words specified in the WordsToKeep parameter based on word frequency. However, if there is a tie in the word frequencies, it returns all the words with that specific frequency, causing more words to be returned than the specified number. To remedy this, we created the ExactStringToWordVector filter that returns the exact number of words specified in WordsToKeep parameter. ExactStringToWordVector samples randomly from the

Table 1 Data set characteristics

Domain	Truthful	Spam	Total
Doctors	200	356	556
Hotels	800	1080	1880
Restaurants	200	200	400
Total	1200	1636	2836

words that have equal frequency. If a value larger than the number of unique words in a document is specified for the WordsToKeep parameter, all unique words are extracted as features.

6.2 Base learners

In this study, we use five base learners: naïve Bayes (NB), multinomial naïve Bayes (MNB), support vector machine (SVM), logistic regression (LR), and C4.5 decision tree (C4.5) in combination with the Boosting and Bagging ensemble techniques. We provide only a brief discussion of these learners here, since they are all well-understood classifiers. However, an interested reader may consult the provided references for more information. All models in this paper are built using the Weka data mining open-source software package (Hall et al. 2009) with default parameter values, unless otherwise specified. Note that any changes to default parameter values were applied when experimentation showed an overall improvement of the classification performance based on preliminary analysis.

Naïve Bayes (Rish 2001) falls under the category of Bayesian learners. Naïve Bayes uses Bayes' theorem to approximate the posterior probability of an instance belonging to a class based on its feature values (Witten and Frank 2005). Naïve Bayes makes the "naïve" assumption that all features are independent. Although, in general, this is not the case with the majority of features, naïve Bayes still offers good performance and we consider it a good baseline learner for comparisons. The formula for naïve Bayes is found below:

$$\hat{y} = p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

where $\hat{y}$ is the predicted class, $p(C_k)$ is the probability of the instance belonging to class k, and $\prod_{i=1}^{n} p(x_i|C_k)$ is the product of all conditional probabilities for the features associated with the instance.

Multinomial naïve Bayes (McCallum and Nigam 1998) is a variant of the naïve Bayes learner. The main difference between NB and MNB is the way in which the probabilities are calculated. Naïve Bayes uses conditional probabilities of each feature to help determine classification status. In multinomial naïve Bayes for text classification, the instance (a document in this example) is assigned to the class which has the highest conditional probability of $P(C|X)$. To calculate this probability, a count is done of the words which overlap between the document and the class, and then the count is divided by the total number of words. If $P(C_1|X) > P(C_2|X)$ then the document is classified as C1; otherwise, it is classified as C2. The MNB algorithm does not take any parameters, thus no changes can be made to the default function within Weka.

Support vector machine (Hsu et al. 2003) constructs a hyper-plane that divides the instances into two groups. The data may be transformed via a kernel function into linearly separable spaces. The transformations allow for nonlinear boundaries to be formed around the data. The best such hyper-plane would be the one that maximizes the distance between the hyper-plane and members of each class. For our models, the complexity constant "c" was set to 5.0 and the "buildLogisticModels" parameter set to "true."

Logistic regression (Hosmer and Lemeshow 2004) seeks to find a linear relationship between the features and the class label. This process is different from linear regression as it is used for the task of classification. Logistic regression aims to create a probability function that uses features as inputs and returns the probability of that instance belonging to a class as an output. Logistic regression was chosen due to its simplicity and effectiveness. The formula for logistic regression can be found below:

$$\log \frac{P}{1-P} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

where $\frac{P}{1-P}$ is the odds ratio, X_i is the value for that feature, and β_i is the coefficient associated with feature X_i.

The final learner, C4.5 decision tree (Quinlan 2014), creates a tree based on the features that discriminate the most between classes. The decision tree process uses IG to determine a decrease in entropy when examining a certain feature. The tree splits at the feature that minimizes entropy and maximizes information for a class, meaning the more discriminant features will be found toward the top of the tree. The final classification is found in the leaves of the tree. When constructing the trees, we implement "Laplace smoothing" and "no pruning" as this increased performance in previous studies (Witten and Frank 2005).

6.3 Cross-validation and performance metric

All models are trained using four runs of fivefold cross-validation. For each run of cross-validation, the data is split into five separate folds. At any time, four of the fivefolds are used to train the data, while the last fold is used to evaluate the model. This process is repeated five times, varying the fold that is used for testing so every fold is used for testing once. We chose to use cross-validation over random sampling as all the data is used both to test and train the model in cross-validation, while only part of the data is used with a random sampling approach. Four runs of fivefold cross-validation are performed to reduce the bias due to an unlucky split in the data when creating the folds. The results of the four runs of fivefold cross-validation are averaged for final results.

We elect to use the Area Under the receive operator characteristic Curve (AUC) as a performance metric

(Witten and Frank 2005). This metric is chosen because it plots the performance of the model across all decision thresholds. The AUC is a graph of the False Positive Rate versus the True Positive Rate. The area under the curve shows performance of the model across all decision thresholds, thus the larger the area under the curve, the better the performance of the model. This is not to be confused with the TBFS technique ROC, which uses AUC to rank features.

7 Case Study I: plain classification and feature selection

7.1 Non-ensemble

We consider results for classification using the base learners and word frequency in the interest of having a baseline for comparison with ensemble methods and feature selection techniques. For this purpose, we examine how our five learners perform for a variety of features, as we vary the number of words extracted by changing the WordToKeep parameter which selects words based on frequency. In Fig. 3, we see the resulting AUC values of the models generated using the base learners and varying levels of features based on frequency. We use a modified version of the StringToWordVector in the WEKA toolkit (Hall et al. 2009). This modified version returns the number of words specified in the frequency selection option WordsToKeep. For example, if you specify 100 WordsToKeep, you will receive 100 distinct words based on the frequency of the word in the data set. This allows for comparison between word frequency performance and performance of features chosen by a feature ranker. Word frequency selection is used with 46 feature subset sizes ranging from 100 to 20,000 to determine performance of word frequency as a feature ranker (20,000 is chosen as the upper limit and encompasses the full feature set as the data set contains less than 20,000 unique words).

In Fig. 3, it can be seen that the best performing learner is multinomial naïve Bayes and its performance improves as WordsToKeep (the number of features used to train the classifier) increases; however, other learners do not follow this trend. C4.5 and NB level off at approximately 2000 features. LR has the highest performance when using approximately 6500 features. SVM performance levels off at approximately 10,000 features. To aid in the choice of the best learner and number of features, we conduct a two-factor ANalysis Of VAriance (ANOVA) to confirm both the choice of learner and the number of features that are significantly impacting classifier performance. The results of the ANOVA test are presented in Table 2A and show both factors and their interactions are significant.

We conduct further tests using Tukey's honestly significant difference (HSD) test to group factors by conducting pairwise similarity tests and determining whether the choice between two levels of a factor is significant. First, we wish to determine which learner, without ensemble technique, is best. If the classifiers share the same letter, then the difference in performance is not statistically significant. Table 2B shows that multinomial naïve Bayes is the best learner across all levels. From Fig. 3, we see that performance for multinomial naïve Bayes increases with number of features; however, from the figure alone we

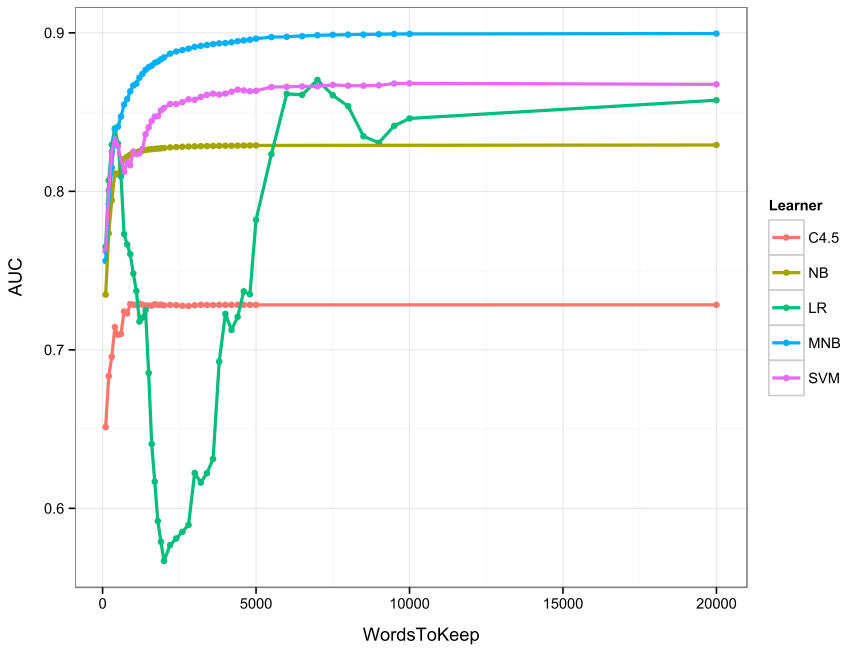


Fig. 3 Classification with word frequency

Table 2 ANOVA and Tukey's HSD test for plain classification

	Df	Sum Sq	Mean Sq	*F* value	Pr(>*F*)
(A) ANOVA for plain classification					
Subset Size	45	4.34	0.10	127.73	0.0000
Learner	4	15.01	3.75	4976.04	0.0000
Subset Size:Learner	160	7.25	0.05	60.09	0.0000
Residuals	3990	3.01	0.00		

Rank	Learner	Group	AUC	SD
(B) Tukey's HSD test for the base learners using word frequency				
1	MNB	a	0.878	0.034
2	SVM	b	0.847	0.027
3	NB	c	0.821	0.032
4	LR	d	0.734	0.107
5	C4.5	e	0.722	0.025

cannot determine when this increase is no longer significant. To determine the top performing subset size, an additional HSD test, found in Table 3, is conducted with feature subset sizes ranging from 100 to 20,000. We found there is no significant difference when using more than 900 features; however, the highest AUC (0.8995) is observed using the full feature set. Thus, multinomial naïve Bayes with the full feature set will be used as the plain classifier in comparison with feature selection in the following section.

7.2 Feature selection

We would like to confirm the effects of feature selection on spam review detection to determine whether to use word frequency or feature selection as a baseline with which to compare ensemble techniques in Case Study II. To test performance of each feature selection technique, we use all base learners and subset sizes ranging from 100 to 1000, as this was shown to be a good range of feature subset size (Crawford et al. 2016). As it is difficult to distinguish significant differences between the rankers, a Tukey's HSD test, at a 95% confidence level, is used to conduct pairwise similarity tests to determine whether the differences between two given rankers is statistically significant. Table 4 presents the results of this test, summarized by placing rankers into groups. Both Chi-squared and signal-to-noise belong to the top group, while ROC, KS, MI and PRC are all in the second group. The remaining rankers perform considerably worse than these 6 rankers.

In Fig. 4, classification results for Chi-Squared are compared against those using word frequency. Chi-Squared is chosen over signal-to-noise as it is more commonly used and the results for both signal-to-noise and Chi-squared are similar across all classifiers as shown in Table 4. We do not provide the results for every feature selection technique against every learner, as the emphasis of this study is on ensemble methods with feature selection and not feature selection alone. We see that as subset size increases, the difference between Chi-Squared and word frequency decreases. Subfigures a and c show performance for subset sizes up to 5000, while subfigures b, d, and e show performance up to 10,000. This subset size difference is due to performance; for C4.5 and NB there is no difference in performance between word frequency and Chi-Squared past 5000 features, while for LR, MNB, and SVM there is a difference up to 10,000 features. Figure 4 shows word frequency performing as well as a feature ranker when working with larger subset sizes. As more features (words) are used, the overlap in feature sets between feature ranker and word frequency increases, leading to similar performance. It is important to note that, with either Chi-Squared or word frequency, multinomial naïve Bayes has the highest AUC scores for most subset sizes.

Additionally, multinomial naïve Bayes' performance increases with the number of features when using Chi-Squared, though these changes are not significant. The difference between Chi-Squared and word frequency is minimal when using larger subset sizes. In the case study on algorithms combining ensemble and feature selection techniques presented in the following section, signal-to-noise, Chi-squared, and Mutual Information are chosen as the feature selection techniques to be embedded within ensemble learners. We elect to use S2N and CS due to their performances, as both techniques fall under group 'a'. MI is selected because of it's performance in other social media/text domains, such as sentiment analysis (Prusa

Table 3 Tukey's HSD test for WordsToKeep with MNB

Rank	WordsToKeep	Group	AUC	SD
1	20,000	a	0.900	0.013
2	10,000	a	0.899	0.013
3	9500	a	0.899	0.014
4	9000	a	0.899	0.014
5	8500	a	0.899	0.013
6	8000	a	0.899	0.013
7	7500	a	0.899	0.013
8	7000	a	0.898	0.013
9	6500	a	0.898	0.013
10	6000	a	0.898	0.014
11	5500	a	0.897	0.014
12	5000	a	0.896	0.014
13	4800	a	0.896	0.014
14	4600	a	0.895	0.014
15	4400	ab	0.895	0.014
16	4200	ab	0.894	0.014
17	4000	ab	0.894	0.015
18	3800	ab	0.893	0.015
19	3600	ab	0.893	0.015
20	3400	ab	0.892	0.015
21	3200	ab	0.892	0.016
22	3000	abc	0.891	0.016
23	2800	abc	0.890	0.016
24	2600	abc	0.889	0.016
25	2400	abc	0.888	0.016
26	2200	abc	0.887	0.016
27	2000	abc	0.884	0.017
28	1900	abcd	0.883	0.016
29	1800	abcd	0.882	0.017
30	1700	abcd	0.881	0.016
31	1600	abcd	0.879	0.017
32	1500	abcd	0.878	0.017
33	1400	abcde	0.877	0.019
34	1300	abcdef	0.874	0.019
35	1200	abcdef	0.872	0.018
36	1100	abcdef	0.868	0.018
37	1000	abcdef	0.867	0.018
38	900	abcdef	0.863	0.018
39	800	bcdef	0.858	0.018
40	700	cdef	0.855	0.021
41	600	defg	0.847	0.021
42	500	efg	0.841	0.022
43	400	fg	0.839	0.022
44	300	gh	0.815	0.023
45	200	hi	0.792	0.025
46	100	i	0.756	0.025

Table 4 Tukey's HSD test of feature selection techniques across all classifiers with feature subset sizes from 100 to 1000

Feature selection	Group	AUC	SD
S2N	a	0.822	0.061
CS	a	0.821	0.062
ROC	b	0.818	0.054
KS	b	0.818	0.054
MI	b	0.817	0.054
PRC	b	0.816	0.054
SAM	c	0.661	0.060
WRS	d	0.614	0.050
GI	e	0.605	0.072
PR	e	0.604	0.070

et al. 2015). We consider this to be a similar domain as both are considered social media, both are online and publicly viewable, both are user-generated, and both contain short texts. This allows us to explore the effects of Select-Boost and Select-Bagging on top scoring rankers and a group 'b' ranker, resulting in a more robust experiment.

8 Case Study II: ensemble techniques

This section presents results related to the ensemble techniques and the combination of ensemble techniques and feature selection. We first present results for Boosting and Bagging; then, we present results for RF, Select-Boost, and Select-Bagging.

8.1 Boosting and Bagging

Results for our experiments utilizing Boosting and Bagging can be found in Table 5. From Table 5, we observe the top performing combination is MNB with Boosting. Bagging shows higher AUCs for three of the five learners (C4.5, LR, SVM), while Boosting produces higher AUCs for the remaining two (MNB, NB).

To determine whether Boosting or Bagging statistically improve over the base learners, we perform a two-factor ANOVA and a Tukey's HSD test. Table 6 presents both the ANOVA and Tukey's HSD results for Boosting, Bagging and the base learners. The two-factor ANOVA test is presented in Table 6A. The factors for this ANOVA are ensemble techniques and base learners. Ensemble techniques include Boosting and Bagging, while the base learners are NB, MNB, SVM, C4.5, and LR. The ANOVA results show there is a significant difference between base

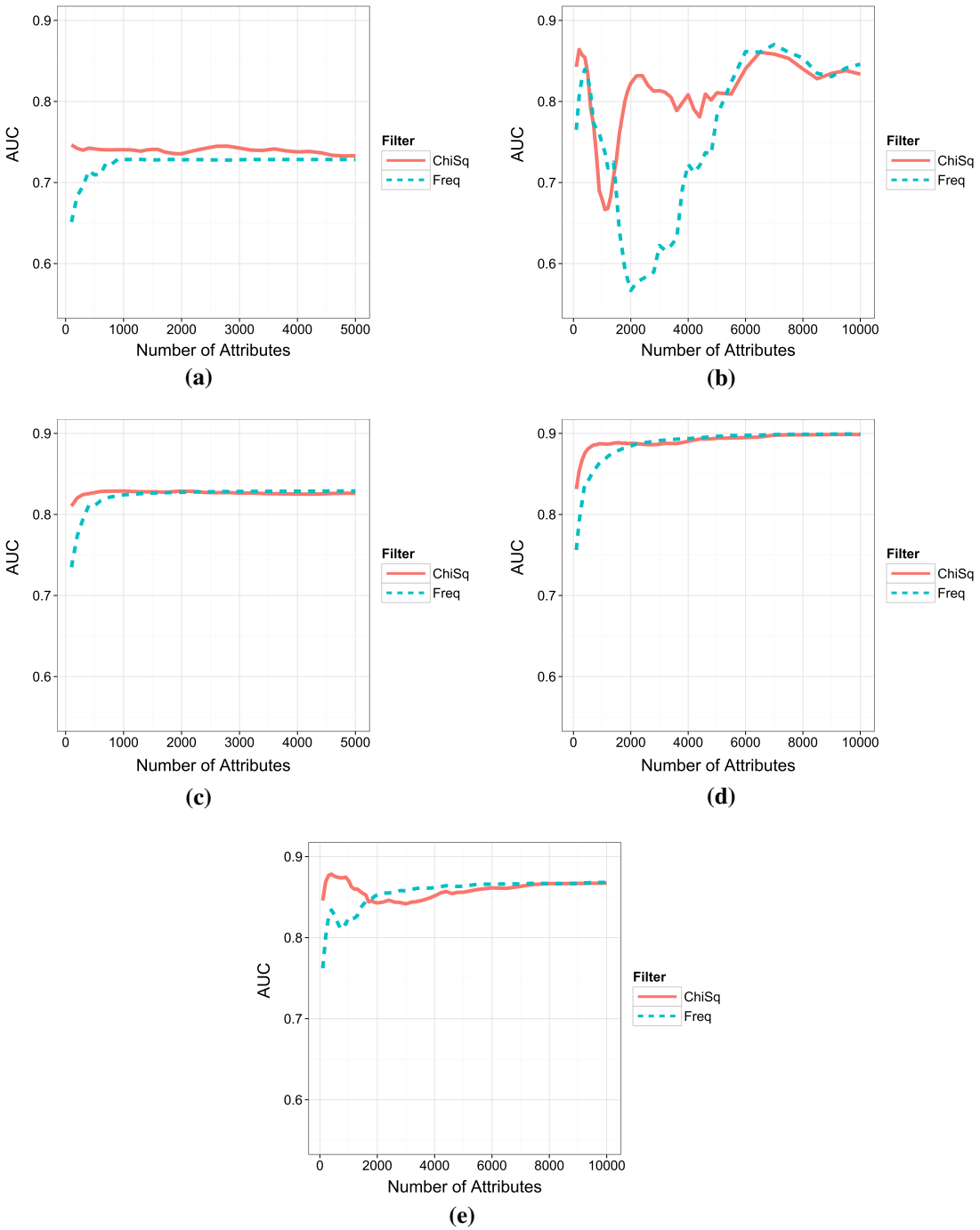


Fig. 4 Average AUC score using feature selection. **a** C4.5, subset sizes from 100 to 5000, **b** logistic regression, subset sizes from 100 to 10,000, **c** naïve Bayes, subset sizes from 100 to 5000, **d** multinomial naïve Bayes, subset sizes from 100 to 10,000 and **e** support vector machines, subset sizes from 100 to 10,000

learners, ensemble methods, and the interaction between them.

From Table 6B, we see the average AUC and Tukey's groups across all experiments for Boosting and Bagging. There are eight distinct groupings: 'a' through 'f'. These results indicate that ensemble techniques significantly increase performance of SVM, LR, NB, and C4.5. Bagging significantly increases performance over the base SVM and LR learners, while Boosting significantly increases performance of LR, NB, and C4.5. We see that the best base learner is MNB. Moreover, the differences between MNB with and without ensemble techniques are not significant; however, not using an ensemble technique is faster and less computationally costly. Boosting with

Table 5 AUC results for Boosting and Bagging across five base learners and all feature subset sizes

Rank	Learner	Boosting	Bagging
1	MNB	0.902	0.900
2	SVM	0.879	0.889
3	LR	0.876	0.888
4	NB	0.874	0.833
5	C4.5	0.827	0.823

Table 6 ANOVA and Tukey's HSD for Boosting and Bagging

	Df	Sum Sq	Mean Sq	*F* value	Pr(>*F*)
(A) ANOVA Results for Boosting and Bagging					
Learner	4	0.41	0.10	345.73	0.0000
Ensemble	2	0.07	0.04	123.41	0.0000
Learner:Ensemble	8	0.09	0.01	37.31	0.0000
Residuals	275	0.08	0.00		

Rank	Model	Group	AUC	SD
(B) Tukey's HSD test AUC values for Boosting and Bagging				
1	MNB:Boosting	a	0.902	0.009
2	MNB:Bagging	a	0.900	0.014
3	MNB:None	a	0.900	0.013
4	SVM:Bagging	ab	0.889	0.011
5	LR:Bagging	ab	0.888	0.011
6	SVM:Boosting	bc	0.879	0.012
7	LR:Boosting	bcd	0.876	0.015
8	NB:Boosting	bcd	0.874	0.011
9	SVM:None	cd	0.867	0.014
10	LR:None	d	0.857	0.027
11	NB:Bagging	e	0.833	0.028
12	NB:None	e	0.829	0.026
13	C4.5:Boosting	e	0.827	0.013
14	C4.5:Bagging	e	0.823	0.019
15	C4.5:None	f	0.728	0.018

MNB does produce the highest AUC and the lowest standard deviation.

8.2 Random Forest

Random Forest provides an different ensemble approach when compared to Bagging and Boosting and performs its own internal random feature selection; thus, the classifier is trained using the full feature set. This ensemble technique does not directly use feature rankers, but does perform random feature subspace selection at every node within a tree. Presented in Table 7, a Tukey's HSD test includes AUC scores for each of the three RF tree sizes and for the C4.5 learner. The result for the C4.5 learner is included as a baseline for comparison, as it is a single decision tree. All RF tree sizes significantly outperform the C4.5 baseline. Moreover, the results show that RF250 and RF500 perform significantly better than RF100, but there are no significant differences between RF250 and RF500. If we compare results in Table 7, we see RF500 generates the highest AUC score.

Table 7 Tukey's HSD test for Random Forest and C4.5

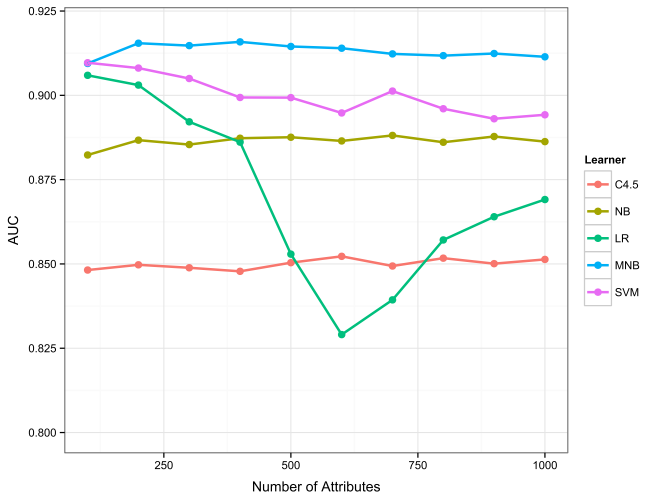
Rank	Model	Group	AUC	SD
1	RF500	a	0.907	0.016
2	RF250	a	0.899	0.017
3	RF100	b	0.876	0.015
4	C4.5	c	0.729	0.020

8.3 Select-Boost and Select-Bagging

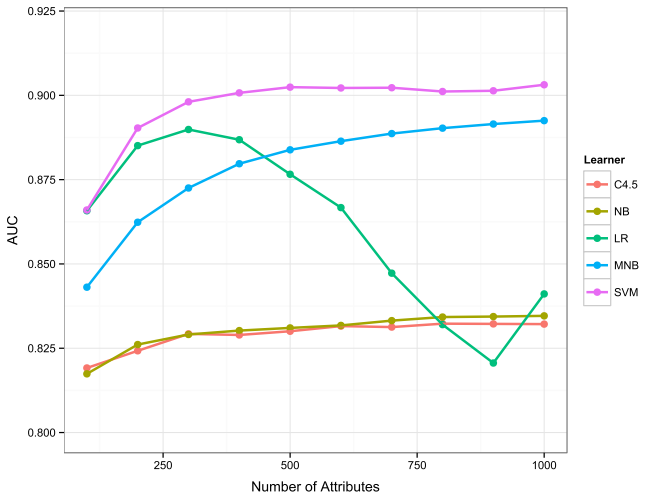
Figure 5 presents results for ensemble classifiers trained using Select-Boost and Select-Bagging grouped by feature ranker (Signal-to-Noise, Chi-Squared, and Mutual Information) and ensemble method (Select-Bagging and Select-Boosting). In each subfigure, AUC for each learner is plotted against the number of attributes selected.

From the subfigures, there are several important trends that warrant further discussion. First, Select-Boost yields higher performance than Select-Bagging for all rankers. We have seen this trend in previous work, where we found Select-Boost outperforms Select-Bagging in the text classification domain (Prusa et al. 2015). Second, as seen in Fig. 5, multinomial naïve Bayes produces the highest AUC for any base learner using Select-Boost.

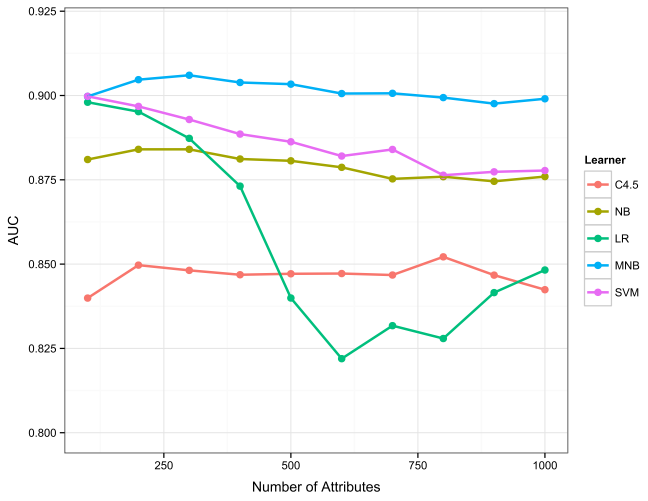
We are interested in which factors (base learner, feature selection technique, ensemble approach, and subset size) are significant; thus, we conduct a four-factor ANOVA. Results are shown in Table 8 and indicate that all four factors and many of their interactions are significant. To investigate each factor: base learner, feature selection (FS in the table), ensemble method (ensemble in the table), and subset size, we conduct Tukey's HSD tests on each factor. Our HSD test for base learners, presented in Table 9A, shows that multinomial naïve Bayes and SVM are the top performing learners and the difference between them and other learners is significant; however, the difference between MNB and SVM is not significant. Table 9B shows that Chi-Squared or Signal-to-Noise should be chosen over Mutual Information. As the difference between using Signal-to-Noise and Chi-Squared is not significant and Chi-Square is more readily available, we will be using Chi-Squared for future experiments. Table 9C shows Select-Boost performs significantly better than Select-


AUC
0.925
0.900
0.875
0.850
0.825
0.800
250
500
750
1000
Number of Attributes
Learner
C4.5
NB
LR
MNB
SVM

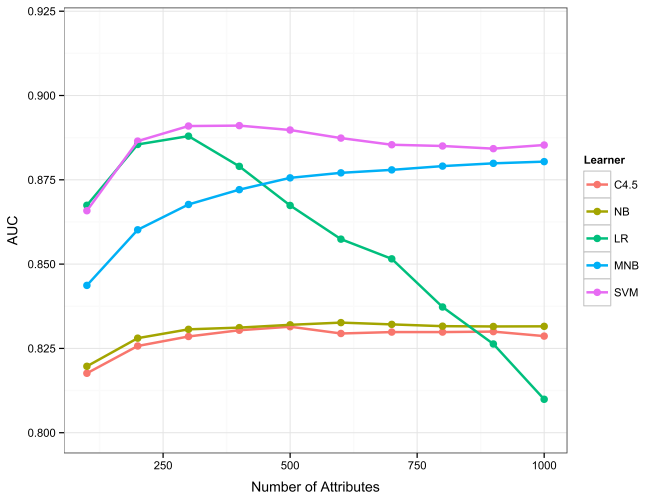
(a)


AUC
0.925
0.900
0.875
0.850
0.825
0.800
250
500
750
1000
Number of Attributes
Learner
C4.5
NB
LR
MNB
SVM

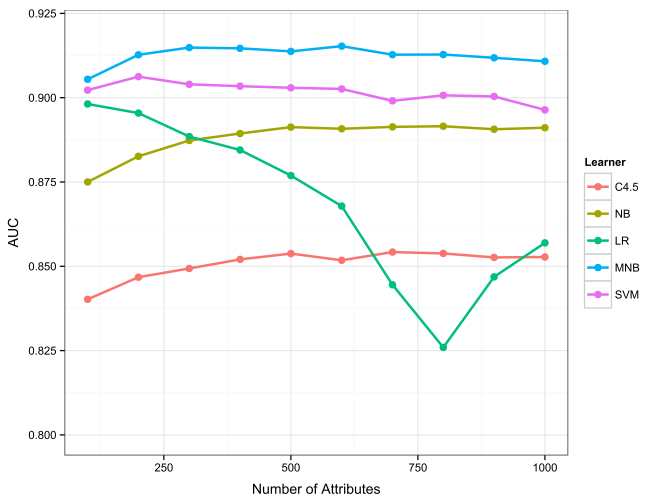
(b)


AUC
0.925
0.900
0.875
0.850
0.825
0.800
250
500
750
1000
Number of Attributes
Learner
C4.5
NB
LR
MNB
SVM

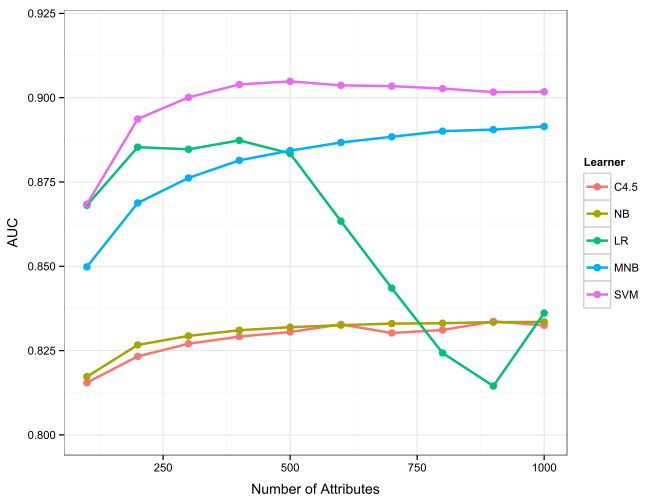
(c)


AUC
0.925
0.900
0.875
0.850
0.825
0.800
250
500
750
1000
Number of Attributes
Learner
C4.5
NB
LR
MNB
SVM

(d)


AUC
0.925
0.900
0.875
0.850
0.825
0.800
250
500
750
1000
Number of Attributes
Learner
C4.5
NB
LR
MNB
SVM

(e)


AUC
0.925
0.900
0.875
0.850
0.825
0.800
250
500
750
1000
Number of Attributes
Learner
C4.5
NB
LR
MNB
SVM

(f)

◀**Fig. 5** Results for Select-Boost and Select-Bagging. **a** Chi-Squared with Select-Boosting, **b** Chi-Squared with Select-Bagging, **c** Mutual Information with Select-Boosting, **d** Mutual Information with Select-Bagging, **e** Signal-to-Noise with Select-Boosting, **f** Signal-to-Noise with Select-Bagging

Bagging. These tests indicate the best performing classifiers are trained with Select-Boost using Chi-Squared, or Signal-to-Noise, as a ranker and MNB, or SVM, as a base learner.

In Fig. 5, we see that the AUC values for Select-Boost are higher than Select-Bagging and Table 9C shows this difference is significant. Furthermore, we observe higher AUC values when using MNB as the base learner within Select-Boost, thus we elect to examine this combination further. There is little change in AUC for multinomial naïve Bayes, across feature subset sizes with Select-Boost. As this is our best performing learner, further investigation is warranted to determine the optimal subset size. A Tukey's HSD test will determine if feature subset size significantly impacts classifier performance with MNB as the learner and Chi-Squared as the ranker. Results, presented in Table 10, show that there are no significant differences as all subset sizes are in the same group; however, 400 features resulted in the highest observed AUC.

8.4 Comparisons

The results presented in the previous sections indicate that multinomial naïve Bayes is the best performing learner, when using no ensemble technique, and one of the top base learners when using the Select-Boosting ensemble framework. Thus, it is of interest to compare how this learning algorithm performs, both with and without the Select-Boost framework, and also how it compares to the best performing models generated with Random Forest. Additionally, we have yet to compare ensemble algorithms with no feature selection and ensemble methods with feature selection. Thus, we compare RF250 and RF500, MNB as a classifier, and Select-Boost with Chi-Squared and 400 features, Bagging, and Boosting using MNB.

Table 11A presents a one-factor ANOVA result (choice of model) for the previously stated classifiers. In the case of this

Table 9 Tukey's HSD for base learners, feature rankers, and ensemble techniques

Rank	Learner	Group	AUC	SD
(A) Tukey's HSD test for base learners				
1	SVM	a	0.895	0.015
2	MNB	a	0.893	0.024
3	LR	b	0.862	0.029
4	NB	c	0.857	0.034
5	C4.5	d	0.839	0.020
Rank	**Feature selection**	**Group**	**AUC**	**SD**
(B) Tukey's HSD test for feature rankers				
1	CS	a	0.872	0.034
2	S2N	a	0.872	0.034
3	MI	b	0.864	0.031
Rank	**Ensemble**	**Group**	**AUC**	**SD**
(C) Tukey's HSD test for ensemble approaches				
1	Select-Boost	a	0.881	0.028
2	Select-Bagging	b	0.858	0.034

Table 8 ANOVA for ensemble classifiers

	Df	Sum Sq	Mean Sq	*F* value	Pr(>*F*)
Base Learner	4	2.80	0.70	2851.65	0.0000
Subset Size	9	0.09	0.01	42.25	0.0000
FS	2	0.08	0.04	155.24	0.0000
Ensemble	1	0.80	0.80	3253.94	0.0000
Base Learner:Subset Size	36	0.58	0.02	65.14	0.0000
Base Learner:FS	8	0.02	0.00	11.96	0.0000
Subset Size:FS	18	0.02	0.00	4.93	0.0000
Base Learner:Ensemble	4	0.53	0.13	544.35	0.0000
Subset Size:Ensemble	9	0.10	0.01	44.91	0.0000
FS:Ensemble	2	0.01	0.00	17.89	0.0000
Base Learner:Subset Size:FS	72	0.03	0.00	1.87	0.0000
Base Learner:Subset Size:Ensemble	36	0.11	0.00	12.92	0.0000
Base Learner:FS:Ensemble	8	0.00	0.00	1.84	0.0656
Subset Size:FS:Ensemble	18	0.01	0.00	1.83	0.0176
Base Learner:Subset Size:FS:Ensemble	72	0.02	0.00	1.15	0.1886
Residuals	5700	1.40	0.00		

Table 10 Tukey's HSD test for feature subset size with multinomial naïve Bayes and Select-Boost with Chi-Squared

Rank	Feature set size	Group	AUC	SD
1	400	a	0.916	0.011
2	200	a	0.915	0.010
3	300	a	0.915	0.010
4	500	a	0.914	0.010
5	600	a	0.914	0.011
6	900	a	0.912	0.008
7	700	a	0.912	0.009
8	800	a	0.912	0.009
9	1000	a	0.911	0.012
10	100	a	0.909	0.010

ANOVA, model encompasses the combination of base learner, feature selection technique and ensemble technique. From Table 11A, we note that the choice of model is significant, thus, we use a Tukey's HSD test to determine the difference between the six rankers. Table 11B presents a Tukey's HSD test showcasing the AUC values for each of these techniques grouped by pairwise similarity. Figure 6 depicts the results found in Table 11B where each model is shown along with their confidence interval. If there is no overlap between two models, then their averages are significantly different. We observe that there are two groups 'a' and 'b'. The bottom group, 'b', contains RF250, MNB as a classifier, and Boosting and Bagging using MNB. Group 'a' contains the Select-Boost algorithm with the previously listed components, implying Select-Boost performs significantly better than RF250, Bagging, Boosting, and MNB with no ensemble technique. RF500 is in group 'ab' indicating no statistically significant difference from the other classifiers.

9 Discussion

Our results show a combination of Select-Boost, multinomial naïve Bayes and, either Chi-Squared or Signal-to-Noise, significantly outperforms all methods except RF500. However, it is interesting to note that while Select-Boost improves performance significantly, the components which create the Select-Boost framework do not increase performance when applied separately.

Feature selection techniques alone do not improve classification over the full data set, in fact, feature selection shows degraded performance with lower feature set sizes. This could be due to properties specific to this data set. The data set contains reviews from three domains, features (words) that determine spam in one domain may not be the same for a different domain. For example, the word 'small' may be positive in respect to cell phones, but negative in respect to hotel rooms. Thus, the reduced feature set may not be able to discriminate between truthful and untruthful reviews across all three domains, leading to lower classification performance.

We also observe that, while boosting increases the AUC of our MNB model, the difference is not significant. It is likely that Boosting and Bagging show no improvement over the base MNB model because they do not address the high dimensionality of the data. When we compare Select-Boost to ensemble techniques using feature selection (RF and Select-Bagging), we see closer performance. Select-Boost still outperforms Select-Bagging, RF100, and RF250. However, the difference between RF500 and Select-Boost not significant. Overall, we recommend Select-Boost over RF500, since Select-Boost has less variance than RF500, most likely due the independent trees, and is faster than RF500.

Table 11 ANOVA and Tukey's results for models

	Df	Sum Sq	Mean Sq	*F* value	Pr(>*F*)
(A) ANOVA for RF500 and RF250, and Select-Boost, Bagging, Boosting and plain classification using MNB					
Model	5	0.00	0.00	4.94	0.0004
Residuals	114	0.02	0.00		

Rank	Model	Group	AUC	SD
(B) Tukey's HSD test for RF500 and RF250, and Select-Boost, Bagging, Boosting and plain classification using MNB				
1	Select-Boost	a	0.916	0.011
2	RF500	ab	0.907	0.016
3	Boosting	b	0.902	0.009
4	Bagging	b	0.900	0.014
5	MNB	b	0.900	0.013
6	RF250	b	0.899	0.017

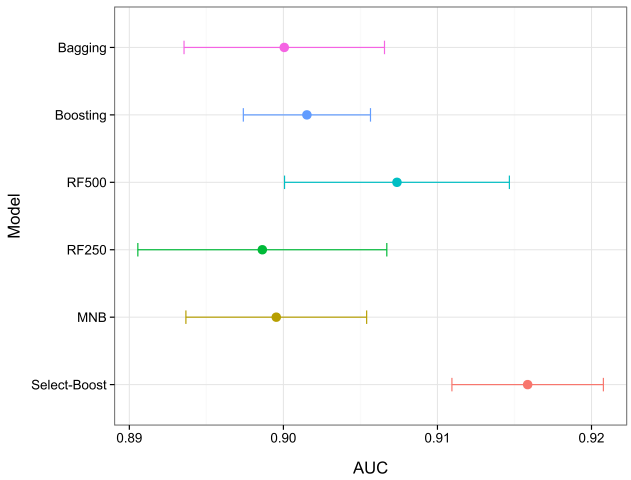


Fig. 6 Visualization of Tukey's HSD for models

Select-Boost significantly outperforms Select-Bagging, implying an aspect of the Select-Boost's iterative framework significantly affects performance. This performance difference may be due to a combination of the re-weighing step found in Boosting and feature selection done in Select-Boost. The Select-Boost framework re-weights instances every iteration, with greater weight being placed on previously misclassified instances, by re-sampling (with replacement) from the original data set with probabilities of selecting instances determined by their weight. As feature selection is performed during every iteration, after the re-sampling step (where misclassified instances are more likely to be present), the weights assigned to the instances directly affect the features chosen by our feature rankers. Every iteration allows for more focus on misclassified instances and their features, which leads to a more optimized feature set as the Select-Boosting framework progresses. Thus, Select-Boost is more effective than Select-Bagging, which performs feature selection on randomly sampled data bootstraps, and offers a significant improvement in classification performance, whereas Boosting and feature selection, individually, do not.

10 Conclusion

In this study, we tested the performance of ensemble classifiers, classification using feature selection, and the combination of both on spam review detection. We evaluated the performance of Select-Boost and Select-Bagging against Random Forest, feature selection, Boosting and Bagging. Select-Boost and Select-Bagging were employed with five learners, three feature selection techniques, and subset sizes ranging from 100 to 1000 in increments of 100. Random Forest was done using three tree sizes: 100, 250 and 500. Ten feature selection techniques were used with subset sizes ranging from 100 to 20,000. The effects of ensemble classifiers on spam review detection had previously not been explored. As spam reviews make up more than a third of all online reviews, the exploration of advanced machine learning techniques, to further current methods for the detection of spam reviews, becomes increasingly necessary.

Our results show that feature selection has degradative effects on classification performance when compared to using the full feature set. The performance of the model steadily increases as feature set size increases, although this increase is no longer significant above 900 features when using word frequency. Ensemble techniques, without feature selection, had no significant effect on classification performance for spam detection. However, using Select-Boost, a combination of feature selection and boosting, significantly increased classification performance. Select-Boost significantly outperformed the base MNB classifier, Boosting, Bagging, Select-Bagging, RF100, and RF250. Select-Boost provides a higher AUC value than all other learners; however, the difference between Select-Boost and RF500 was not significant.

We recommend the use of Select-Boost over Select-Bagging. The re-weighting step in Select-Boost is crucial in generating an optimized, reduced feature set, allowing for significantly better performance, since each iteration selects a new subset of features that aid in the correct classification of previously misclassified instances. Select-Boost should be chosen over RF500 as it results in a higher AUC, although not significantly different, and is faster. We found using the combination of Select-Boost, multinomial naïve Bayes, Chi-squared and a subset size of 400 has the highest AUC when detecting spam reviews, although Chi-Squared and Signal-to-Noise may be interchanged. There is no significant difference in performance between feature subset sizes when using feature selection within Select-Boost, thus, while the highest AUC is observed when selecting 400 features, computational resources can be preserved by using a smaller number of features without significantly degrading classification performance.

Future work may involve testing this process on other data sets to see if results generalize. Additionally, work should be conducted to establish what types of features are the most beneficial, since we only employed text-based features.

Acknowledgements The authors would like to thank the anonymous reviewers and the Editor for the constructive evaluation of this paper and also the various members of the Data Mining and Machine Learning Laboratory, Florida Atlantic University, for assistance with the reviews. Also, we acknowledge partial support by the NSF (CNS-1427536). Opinions, findings, conclusions, or recommendations in this paper are the authors and do not reflect the views of the NSF.

References

Berenson ML, Goldstein M (1983) Intermediate statistical methods and applications: a computer package approach, 2nd edn. Prentice Hall, Upper Saddle River

Breiman L (1996) Bagging predictors. Mach Learn 24:123–140. doi:10.1007/BF00058655

Breitling R, Herzyk P (2005) Rank-based methods as a non-parametric alternative of the t-statistic for the analysis of biological microarray data. J Bioinform Comput Biol 3(5):1171–1189. http://view.ncbi.nlm.nih.gov/pubmed/16278953

Broder A (1997) On the resemblance and containment of documents. In: Proceedings of compression and complexity of sequences 1997, pp 21–29

Buhrmester K, Goslining (2011) Amazon's mechanical turk a new source of inexpensive, yet high-quality data? Perspect Psychol Sci 6(1):3–5. http://pps.sagepub.com/content/6/1/3

Chen X, Wasikowski M (2008) Fast: a ROC-based feature selection metric for small samples and imbalanced data classification problems. In: Proceedings of the 14th ACM SIGKDD international conference knowledge discovery and data mining (KDD '08). ACM, New York, Aug 2008, pp 124–132

Crawford M, Khoshgoftaar TM, Prusa JD (2016) Reducing feature set explosion to facilitate real-world review spam detection. In: Proceedings of the 29th international FLAIRS conference, pp 304–309

Crawford M, Khoshgoftaar TM, Prusa JD, Richter AN, Al Najada H (2015) Survey of review spam detection using machine learning techniques. J Big Data 2(1):1–24. http://link.springer.com/article/10.1186/s40537-015-0029-9

Dietterich TG (2000) Ensemble methods in machine learning. In: International workshop on Multiple classifier systems, pp 1–15

Dittman DJ, Khoshgoftaar TM, Wald R, Van Hulse J (2010) Comparative analysis of dna microarray data through the use of feature selection techniques. In: Proceedings of the ninth IEEE international conference on machine learning and applications (ICMLA), vol 1857. Springer, Berlin, Heidelberg, pp 147–152

Dixit S, Agrawal A (2013) Survey on review spam detection. Int J Comput Commun Technol 4(2):68–72. http://interscience.in/IJCCT_Vol4Iss2/68-72.pdf

Forman G (2003) An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 3:1289–1305

Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: Proceedings of the 13th international conference on machine learning, pp 148–156

Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2009) The weka data mining software: an update. SIGKDD Explor Newsl 11(1):10–18

Haykin S (1998) Neural networks: a comprehensive foundation, 2nd edn. Prentice Hall, Upper Saddle River

Heredia B, Khoshgoftaar TM, Prusa JD, Crawford M (2016) An investigation of ensemble techniques for detection of spam reviews. In: 2016 15th international conference on machine learning and applications (ICMLA), pp 127–133

Hosmer DW Jr, Lemeshow S (2004) Applied logistic regression. Wiley, Hoboken

Hsu CW, Chang CC, Lin CJ (2003) A practical guide to support vector classification, Technical report, Department of Computer Science, National Taiwan University

I.C. Government of Canada (2014) Don't buy into fake online endorsements—not all reviews are from legitimate consumers. http://www.competitionbureau.gc.ca/eic/site/cb-bc.nsf/eng/03782.html

Jindal N, Lui B (2008) Opinion spam and analysis. In: Proceedings of the 2008 international conference on web search and data mining. https://www.cs.uic.edu/~liub/FBS/opinion-spam-WSDM-08.pdf

Khoshgoftaar TM, Dittman DJ, Wald R, Fazelpour A (2012) First order statistics based feature selection: a diverse and powerful family of feature selection techniques. In: Proceedings of the eleventh international conference on machine learning and applications (ICMLA). ICMLA, pp 151–157

Li J, Myle O, Cardie C, Hovy E (2014) Towards a general rule for identifying deceptive opinion spam. In: Proceedings of the 52nd annual meeting of the Association for Computational Linguistics, pp 1556–1576. http://anthology.aclweb.org/P/P14/P14-1147.pdf

McCallum A, Nigam K (1998) A comparison of event models for Naive Bayes text classification. In: AAAI-98 workshop on learning for text categorization

Mukherjee A, Venkataraman V, Liu B, Glance N (2013) What yelp fake review filter might be doing? In: Seventh international AAAI conference on weblogs and social media. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM13/paper/view/6006

Ott M, Choi Y, Cardie C, Hancock J (2011) Finding deceptive opinion spam by any stretch of the imagination. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies, vol 1, pp 309–319

Peng H, Long L, Ding C (2005) Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans Pattern Anal Mach Intell 27(8):1226–1238

Prusa JD, Khoshgoftaar TM, Dittman DJ (2015) Impact of feature selection techniques for tweet sentiment classification. In: Proceedings of the 28th international FLAIRS conference, pp 299–304

Prusa JD, Khoshgoftaar TM, Napolitano A (2015) Using feature selection in combination with ensemble learning techniques to improve tweet sentiment classification performance. In: Proceedings of the 27th international conference on tools with artificial intelligence, pp 186–193

Quinlan RJ (2014) C4.5: programs for machine learning. Elsevier, Amsterdam

Rish I (2001) An empirical study of the Naive Bayes classifier. In: IJCAI 2001 workshop on empirical methods in artificial intelligence

Shojaee S, Murad M, Sharef N, Nadali S (2013) Detecting deceptive reviews using lexical and syntactic features. In: 2013 13th international conference on intelligent systems design and applications (ISDA)

Tusher VG, Tibshirani R, Chu G (2001) Significance analysis of microarrays applied to the ionizing radiation response. Proc Natl Acad Sci 98(9):5116–5121. http://www.pnas.org/content/98/9/5116.abstract

Witten IH, Frank E (2005) Data mining: practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, Burlington