IST707 Applied Machine Learning
School of Information Studies
Syracuse University

Hendi Kushta

HW8: Lie Detection and Sentiment Classification with Text Mining

Scenario: Some people claim that machine learning algorithms can figure out whether a person is lying or not. Do you believe that? To test this claim, we have assembled a collection of customer reviews — some true and some false — on which you are going to test how good Naïve Bayes and SVMs can be for **fake review detection**. This data set also has sentiment label for each review. You will also compare NB and SVMs performance in **sentiment classification**.

For both tasks, try different tuning parameters and report the results for the **best model of each type** in a table like the following.

Model	Parameter Settings	Accuracy Lie Detection	Precision Lie Detection	Recall Lie Detection	Accuracy Sentiment	Precision Sentiment	Recall Sentiment
SVM	Degree: 1,2,3 scale: 0.001, 0.01, 0.1, 1.0 C: 0.1, 2, length = 10 method: cv, number 5	0.7842	1	0.3333333	0.6152	0.6428571	1
NB	laplace 0, 0.5, 1 method: cv, number = 5, userkernel false true, adjust : false true	0.538	NaN	0	0.615	0.615	1

1. Explain the data and the pre-processing steps you took to prepare for each classification task.

   The data we have is a set of reviews about different restaurants. Each review has several pieces of information associated with it, such as whether the review is real or fake, the sentiment of the review (positive or negative), and the actual text of the review.

   To prepare the data for classification tasks, several preprocessing steps were taken. These steps involve manipulating the data to make it easier to work with and extract meaningful information.

   The first step was to add a new column called "Index" to the dataset, which simply numbers each row so that we can keep track of it throughout the preprocessing steps.

The second step was to tokenize the text of each review, which means breaking the text up into individual words. This is done to make it easier to analyze the content of the reviews. The resulting data is stored in a new column called "word".

The third step was to remove stop words from the "word" column. Stop words are common words like "a", "an", and "the" that do not add much meaning to the text, so removing them can help to focus on the important words.

The fourth step was to stem and lemmatize the "word" column. This involves reducing each word to its base form, such as converting "eating" to "eat" or "eats". This step can help to group similar words together and reduce the overall number of unique words. The results are stored in new columns called "word_stemmed" and "word_lemmatized".

The fifth step was to create a wide format data frame that shows the count of each lemmatized word for each review. This step involves counting the number of occurrences of each word and organizing the data so that each row represents a review, and each column represents a unique word.

The sixth step was to create a similar wide format data frame, but this time showing the count of each word for each review and including the sentiment of the review as a separate column.

Finally, the "Index" column was removed from both of these data frames, since it is no longer needed.

These preprocessing steps were performed separately for two classification tasks: one to classify reviews as positive or negative based on their sentiment, and another to classify reviews as real or fake based on their authenticity. This involves adding the sentiment or authenticity column to the data frames created in the preprocessing steps so that they can be used for classification.

2. Explain your initial parameter tuning strategy — which parameter to tune, to what option, and the theoretical foundation for your choice. Does your strategy help you get better results?

SVM

The output you provided shows the results of a Support Vector Machine (SVM) model with a polynomial kernel that was trained and tuned on two different datasets. The first dataset is called "your_data_wide_sentiment," and the SVM model was trained to predict the sentiment of each sample as either "positive" or "negative." The second dataset is called "wide_data_lie," and the SVM model was trained to predict whether each sample contained a lie or not.

The model was trained and tested using cross-validation with 5 folds. The SVM model was tuned on different combinations of hyperparameters, including the degree of the polynomial kernel, the scale of the kernel, and the regularization parameter C. For each

combination of hyperparameters, the model's accuracy and Kappa statistic were calculated.

The results show that the best accuracy achieved on the "your_data_wide_sentiment" dataset was 61.52%, and the best Kappa statistic was 0.11. The best accuracy achieved on the "wide_data_lie" dataset was 78.42%, and the best Kappa statistic was 0.57. These results suggest that the SVM model performed better at predicting lies than predicting sentiment.

Recall for "your_data_wide_sentiment" dataset is 1 meaning that the model has correctly identified all positive cases in the dataset. While the recall for your_data_wide_lie is 0.3333333 meaning the model identified only 33% of all positive cases.

Overall, tuning the hyperparameters of an SVM model can be a crucial step in improving its performance. It is also important to note that the performance of a model may vary depending on the dataset and the problem it is trying to solve.

Naive Bayes

Firstly I split the data into training and testing sets. The createDataPartition function from the caret package is used to randomly split the data into 70% training and 30% testing sets.

Next, the code trains two Naive Bayes models on the training sets - one for sentiment analysis and another for detecting lies. Naive Bayes is a popular machine learning algorithm for text classification tasks, where the goal is to predict the category or sentiment of a piece of text (in this case, whether it is positive or negative, or whether it contains a lie or not).

After training the models, the code uses cross-validation to tune their hyperparameters. Hyperparameters are settings of the model that are not learned from the data, but are set by the user. Tuning the hyperparameters involves trying out different settings and selecting the ones that give the best performance on the validation data.

Finally, the code evaluates the performance of the models on the testing sets. It makes predictions on the testing data using the tuned models, and then calculates various metrics to assess their performance. The metrics reported in this code include accuracy, precision, and recall for each of the two tasks - sentiment analysis and lie detection.

The results of the performance evaluation are printed to the console using the cat function. The sentiment analysis model achieved an accuracy of 0.615, precision of 0.615, and recall of 1.0 (which means it correctly identified all the positive instances in the testing set, but missed some negative instances). The lie detection model achieved an accuracy of 0.538, but its precision and recall could not be calculated because there were no true positive instances in the testing set.

3. Compare performance differences in sentiment classification and lie detection and tell us which task is harder. Try to explain why one may be harder than the other.

Based on the provided results, it seems that sentiment classification is easier than lie detection. This can be observed from the fact that the SVM model achieves a higher accuracy and precision for sentiment classification (0.6152 and 1, respectively) than for lie detection (0.7842 and 0.3333, respectively). Additionally, the recall for sentiment classification is also higher than for lie detection (0.6429 and 0, respectively).

On the other hand, the Naive Bayes model performs poorly on both tasks, with low accuracy and precision for lie detection (0.538 and NaN, respectively) and a relatively low accuracy and precision for sentiment classification (0.615 and 0.615, respectively).

One reason why lie detection may be harder than sentiment classification is that lying is a more complex and nuanced behavior than expressing a sentiment or emotion. Detecting lies involves not only understanding the meaning of the words being used but also interpreting the speaker's tone, body language, and other subtle cues. In contrast, sentiment classification can be based mainly on the words used and does not necessarily require an understanding of the speaker's intent or context.