**Team**

Chong Zhou (Andy)

Diana Batista

Marcus Moyses

Tabassum Kakar

1 November, 2014

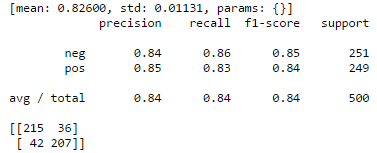
Case Study 3: Textual Analysis of Movie Reviews

The purpose of this case study is to develop a predictive model based on textual movie reviews. In order to do this, we downloaded data from the v2.0 polarity dataset, which includes 1000 positive and 1000 negative reviews. The target in this dataset is the positive review, which is represented by 1 and negative review, represented by 0. Unlike the movie review dataset in case study 2, these reviews are textual only.

Part 1: Sentiment Analysis on Movie Reviews

In order to assess the accuracy of any classifier, the data must first be randomly split into training (75%) and testing sets (25%). The purpose of this step is to test the accuracy of the classifier once it has been developed. If the model/classifier is tested on the training data only, it may cause overfitting and there would be no way of establishing whether the model is effective. For this reason, testing data is used to test the accuracy of the classifier on the test set.

The linear SVC classifier yields the following results:



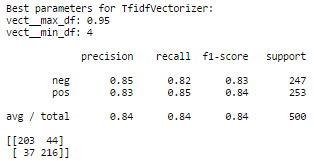
The 215 in the confusion matrix above represents the number of negative reviews that were accurately classified as negative, the 36 is the number of negative reviews that were incorrectly classified as positive (false positive). The 42 represents the number of positive reviews that were incorrectly classified as negative (false negative) and the 207 is the number of positive reviews correctly classified as positive. Overall, this model has an 84% accuracy.

Problem 2: Explore the Scikit-learn TfidVectorizer class

The first part of this case study follows the example in Scikit-learn. In part 2 we explore the different parameters that affect the accuracy of the model, namely min\_df, max\_df, and ngram-range.

The min\_df and max\_df is relevant to the TF-IDF parameter. The TF-IDF, also known as term frequency-inverse document frequency, weighs the importance of words based on its frequency within the data, but penalizes it if it appears too often. If a word appears too many times, it probably does not provide any value-added information, such as the word “the.” Almost all reviews will include the word “the,” regardless of whether the review is positive of negative, therefore it should have a very low weight in assessing the sentiment of the reviews. The min\_df determines how many times a word needs to appear in the document in order to be included in the analysis while the max\_df states the maximum frequency of a word in a document in order to be included.

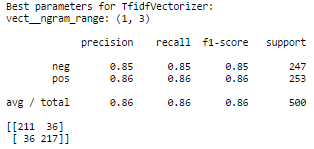
By defining the min\_df as a range between 1 and 5 and max\_df between .85 and .95, all possible combinations of the two dfs will be assessed in order to determine which combination of parameters will yield the highest accuracy.



For this dataset, a min\_df of 4 and max\_df of .95 yields the best results, however, this does not change the accuracy of the original model.

The ngram-range parameter takes context into consideration by setting the range of the combined words used as a feature. Words alone cannot be trusted to accurately reflect the sentiment of a review. As an example, the word “amazingly” can be proceeded by “bad” or “good,” however on its own, it tends to have a positive connotation.

Our model yields the following results:



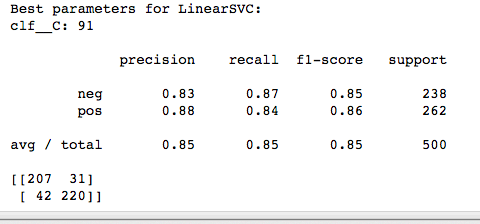
A 3-gram increases the model’s accuracy by 2 percentage points.

Problem 3: Machine Learning Algorithm

From Problem-2 the parameters that gave the maximum accuracy were min\_df=4, max\_df=0.95, ngram\_range=(1, 3). Hence same parameters are used to transform the training and testing data into Tf-idf-weighted document-term matrix.

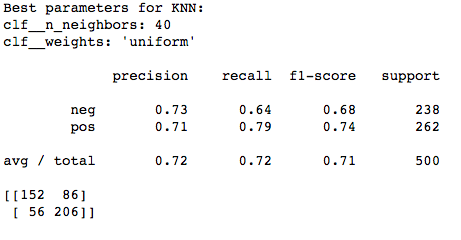
**LinearSVC**

For LinearSVC we used the grid search to find the best penalty parameter C of the error term which is also known as the soft margin constant, i.e. it chooses a hyperplane that splits the examples as cleanly as possible, by maximizing the distance to the nearest cleanly split examples. Out of a range of 1-2000 with a step size of 10, 11 is selected as the best parameter which gives an accuracy of 85% on the test data.

****

**KNeighborsClassifier**

For KNeighborsClassifier we used a grid search to find the best n\_neighbor parameter which gives higher accuracy and out of 1-50, 40 is selected as the best which gave an accuracy of 74% on the testing data. The LinearSVC performs well with an accuracy of 85% on test data. The parameter that gives the best result is C = 91. In comparison the KNeighborsClassifier gives an accuracy of 72%. We tested the n\_neighbors parameters for 1-50 neighbors and weights for distance and uniform value, and the best values selected for each parameter is 40 and uniform respectively. The reason why KNeighborsClassifier did not perform well is because by using 40 neighbors it has over fitted the training data and when used the same model for testing, it failed to generate the same accuracy.



**Example of False Positive:**

“The subtitles of our print were white , often on a white background making them often hard to read , but I don't think there was a whole lot of meaning there that was lost.”

The above review is one of the false positives. The review is totally negative, it does not have a negative word but has a sarcastic tone. It is misclassified because the model is trained to detect negative words as an indication of negative remarks but it is not trained to detect the negative remarks hidden in the form of sarcasm.

**Example of false negative:**

“No matter how disgusting or revolting " mr . splein " may be , you still can't help but laugh.”

The above review is misclassified as negative though it is positive. The reason is the word “disgusting “as it is a negative remark and there is no other joint word that make it positive such as “Not disgusting”, that is why it is classified as negative.

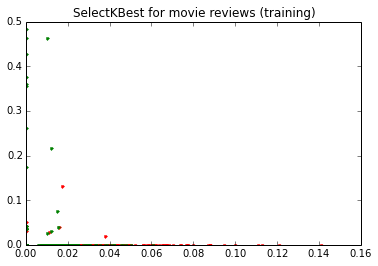
Problem 4:

The question is to transform the multidimensional data into two numbers to be represented by a two-dimensional plot. The key idea is to reduce the high dimensional data into two dimensions, to help us distinguish between negative and positive reviews.

Our first approach is to use SelectKbest method to select two important features in the data and the result is as follows:

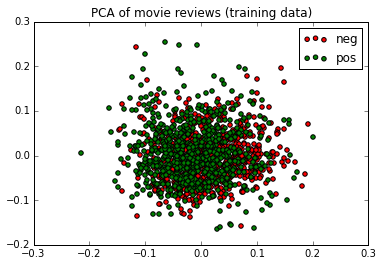


From above result the select two most important features are “bad” and “mulan”. “Bad” can be an important feature in making a review negative or positive. On the other hand “mulan” is just a movie name and it has nothing to do with the negativity or positivity of the movie review. The below figure visualizes the above result.



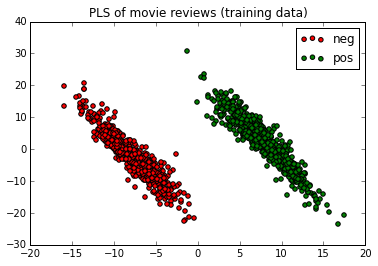
The above figure is not clear, as there are many points along with the x-axis and y-axis, because the data set is a high dimensional sparse matrix and when two of these dimensions are selected it leaves many instances with a zero value causing the points to remain on the axis.

As our first approach did not work well, our second approach is to use Principle Component Analysis (PCA) for dimensionality reduction. We used PCA on the training data and go the below result.

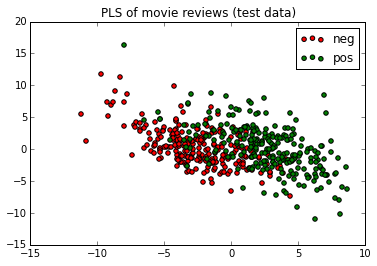


From the above figure we can see that the positive and negative reviews are not separated clearly from each other even in the training data. Though the result is better than SelectKbest Method but still there is no clear boundary between the positive and negative reviews.

Our next approach is to use Partial Least Squares (PLS) dimension reduction on the training data the results are as follows:



As we can see for the training data the negative and positive reviews are separated clearly and is much better than the previous two methods. We then used the PLS method on the testing data and got the below result.



Even for the test data, the negative reviews are on left side and positive reviews are on the right side with a little mix-up in the middle.

Why PLS works better?

PLS is a supervised dimensionality reduction method while PCA is an unsupervised dimensionality reduction method. In principle PLS finds projection direction for which the covariance between x and y is maximized i.e. to find the projection for the variables that are strongly related to the class variable. While PCA finds the projection direction with highest variance and highest variance, it doesn’t guarantee the separation of the class.

In another words, we want to distinguish the target attribute, and PCA does not considers the target attribute while PLS does, and this can explain the above performance of both methods.