Technical Report: Project 3

Fan Yang (fanyang3) and Xiaozhu Ma (xiaozhu3)

Apr 24, 2021

## Pre-Processing: Construct the feature matrix with the bag of words

First we use the the **sklearn.feature\_extraction.text.CountVectorizer** class to transform the list of reviews into a matrix of the bag of words, where each **row** **i** represents a review, and each **column** **j** represents a word we select (thus 1000 columns in total). The value in **(i, j)** represents the frequency of the **word j** appearsin **review i**.

## How We Construct the Vocabulary

Using the preprocessing approach mentioned above, we create a feature matrix with words up to 4-grams, excluding the stop words defined in **nltk’s stopwords** module.

We tried two methods to select the vocabulary:

1. We run T-tests for each word vector, comparing between the ones corresponding to positive reviews and the ones corresponding negative reviews, and select the top 2000 words with the largest absolute t-statistic values.
2. We run a lasso model and select the words with non-zero coefficients. This gives us 2126 words.

Then we choose the words that appear in the results of both of the approaches. This gives us 866 words. This is the set of words we will use in the 5-fold prediction.

## Model

### Use Ridge Model

We choose Ridge model, instead of Lasso, as we already use Lasso to select the relevant 866 variables, and Ridge will not further shrink the coefficients of already-selected-variables to zero.

### Cross Validation

To decide the best value of the tuning parameter lambda (the regularization strength), for each fold, we use the **sklearn.model\_selection.GridSearchCV** class to search lambda in a range of 10 numbers spaced evenly on a log scale from 0.01 to 1.

### Alternative Models Tried

We also tried Gradient boosting trees with 1000 iterations, but its performance, surprisingly, is not better than Ridge. As the training time for Gradient boosting is almost 100x longer than Ridge, we prefer using Ridge in the prediction.

## Interpretability

We used the the first review of fold 0’s test data as an example:

*"Naturally in a film who's main themes are of mortality, nostalgia, and loss of innocence it is perhaps not surprising that it is rated more highly by older viewers than younger ones. However there is a craftsmanship and completeness to the film which anyone can enjoy. The pace is steady and constant, the characters full and engaging, the relationships and interactions natural showing that you do not need floods of tears to show emotion, screams to show fear, shouting to show dispute or violence to show anger. Naturally Joyce's short story lends the film a ready made structure as perfect as a polished diamond, but the small changes Huston makes such as the inclusion of the poem fit in neatly. It is truly a masterpiece of tact, subtlety and overwhelming beauty."*

We define the **contribution** of a word to a review’s sentiment as the product of its coefficients in our Ridge model and the frequency of that word in the review. Below are all the words with non-zero contributions to the review above:

|  |  |
| --- | --- |
| word | contribution |
| **perfect** | 0.66 |
| **tears** | 0.46 |
| **masterpiece** | 0.39 |
| **innocence** | 0.38 |
| *themes* | 0.34 |
| *makes* | 0.29 |
| beauty | 0.24 |
| highly | 0.21 |
| enjoy | 0.20 |
| small | 0.17 |
| truly | 0.13 |
| natural | 0.07 |
| film | 0.00 |
| story | (0.01) |
| made | (0.07) |

Generally, we find that the words with top contributions (**highlighted**) make sense. But there are also some too-generic words, such as **themes** and **makes,** which suggest that our selection of words could be better.

## Performance

Please see the highlighted column below for our final model’s performance and running time. We use a 2016 Macbook Pro (2GhZ of Intel i5 and 8 GiB memory).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Top 1000 by t-test value** | | | **Top 2000 by t-test value** | | | **866 words selected by both Lasso and T-test** | | |
| data | training auc | test auc | running time | training auc | test auc | running time | training auc | **test auc** | ***running time*** |
| 0 | 0.959 | 0.944 | 55.464 | 0.966 | 0.947 | 78.786 | 0.963 | **0.945** | ***57.669*** |
| 1 | 0.955 | 0.947 | 51.533 | 0.963 | 0.950 | 73.630 | 0.959 | **0.949** | ***53.007*** |
| 2 | 0.956 | 0.946 | 50.832 | 0.963 | 0.949 | 72.297 | 0.958 | **0.949** | ***51.826*** |
| 3 | 0.956 | 0.947 | 49.276 | 0.962 | 0.949 | 82.019 | 0.959 | **0.948** | ***47.224*** |
| 4 | 0.956 | 0.947 | 50.103 | 0.964 | 0.949 | 80.011 | 0.960 | **0.948** | ***50.985*** |