589 Extra Credit

Chengzhe Li

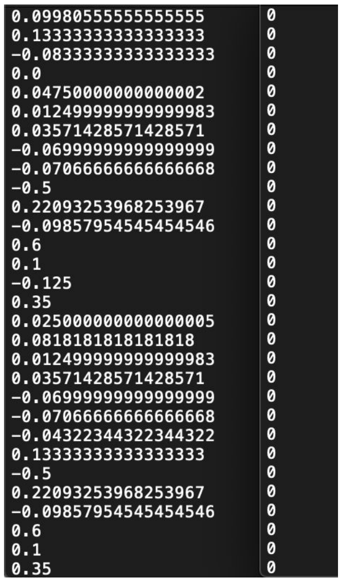
Overall

I’m so confused when I first get this assignment and have totally no idea about how to process the data, make a dictionary like dataset, and input that dataset to sklearn or some other libraries. Then after slow googling techniques, syntax and thinking. I come with three approaches. First one is to get the text from review and rating, use a library named textbolb to get the score of the sentence sentiment, and model a SVM to classify the samples. This is the easiest one and also the one I’m currently reporting. The second one is to first process the data into dictionary like word count by some specific library like nltk or use the processed_balanced review directly, and implement a naive bayes classifier by sklearn, just like what we’ve taught in class about classifying spam emails by naive bayes classifier. And the third way is to design a conventional neural network and classify the reviews. In my opinion the accuracy may be 3rd > 2nd > 1st, and the difficulty to implement is the same. I spent several days to implement an easy version of first approach, so I may just be able to report the first one.

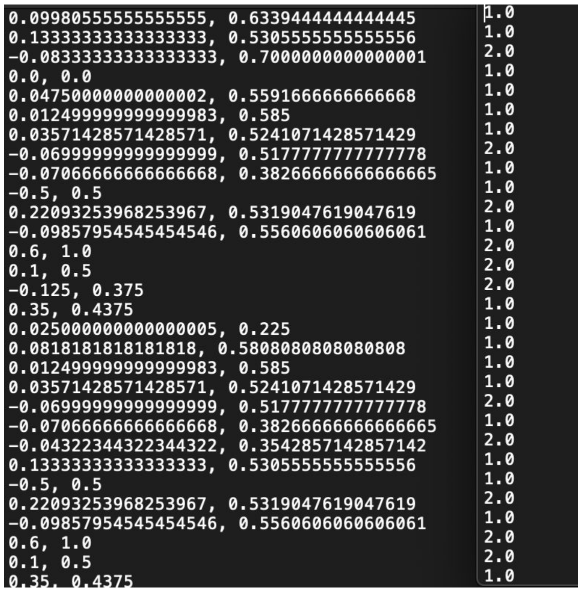
Part1

Process: I first wanted to try some library to handle to file, and I tried xmltodict to transform the file from xml to dict and then play with the dict, but it turned out that the review files are using pseudo XML and the library cannot work on that, then I decided to read the file line by line and use strip() command to find tags, save the text in a list and process the list again to make it becomes sentiment value, and output it to files.

Data: (UPDATED with 0/1 label)



```
0.09980555555555555      0
0.13333333333333333      0
-0.08333333333333333     0
0.0                      0
0.04750000000000002      0
0.012499999999999983     0
0.03571428571428571      0
-0.06999999999999999     0
-0.07066666666666668     0
-0.5                     0
0.22093253968253967      0
-0.09857954545454546     0
0.6                      0
0.1                      0
-0.125                   0
0.35                     0
0.025000000000000005     0
0.0818181818181818       0
0.012499999999999983     0
0.03571428571428571      0
-0.06999999999999999     0
-0.07066666666666668     0
-0.04322344322344322     0
0.13333333333333333      0
-0.5                     0
0.22093253968253967      0
-0.09857954545454546     0
0.6                      0
0.1                      0
0.35                     0
```

or 

```
0.09980555555555555, 0.6339444444444445        1.0
0.13333333333333333, 0.5305555555555556        1.0
-0.08333333333333333, 0.7000000000000001       2.0
0.0, 0.0                                       1.0
0.04750000000000002, 0.5591666666666668        1.0
0.012499999999999983, 0.585                    1.0
0.03571428571428571, 0.5241071428571429        1.0
-0.06999999999999999, 0.5177777777777778       2.0
-0.07066666666666668, 0.38266666666666665      1.0
-0.5, 0.5                                      1.0
0.22093253968253967, 0.5319047619047619        2.0
-0.09857954545454546, 0.5560606060606061       1.0
0.6, 1.0                                       2.0
0.1, 0.5                                       2.0
-0.125, 0.375                                  2.0
0.35, 0.4375                                   1.0
0.025000000000000005, 0.225                    1.0
0.0818181818181818, 0.5808080808080808         1.0
0.012499999999999983, 0.585                    1.0
0.03571428571428571, 0.5241071428571429        1.0
-0.06999999999999999, 0.5177777777777778       2.0
-0.07066666666666668, 0.38266666666666665      1.0
-0.04322344322344322, 0.3542857142857142       2.0
0.13333333333333333, 0.5305555555555556        1.0
-0.5, 0.5                                      1.0
0.22093253968253967, 0.5319047619047619        2.0
-0.09857954545454546, 0.5560606060606061       1.0
0.6, 1.0                                       2.0
0.1, 0.5                                       2.0
0.35, 0.4375                                   1.0
```

Text sentiment and rating it relate with

The processed data is "review sentiment value" and label with rating.
I used textblob to evaluate the sentiment of the whole review. I tried to use bolb.sentences to tokenize the review and calculate the mean value of the sentiment, but the result looks not very well, so I choose to just use the sentiment value of the whole review. Also, the sentiment of textbolb evaluate the sentence in two respects, polarity and subjectivity. In my opinion only polarity is useful in this project because reviews are a subject things indeed so we don't need a value to evaluate the subjectivity. But it can also be tried to use both respects to evaluate the review. UPDATED: both features are used.

Data labeling: my label is 1.0, 2.0, 4.0, 5.0, dropped all 3.0 reviews. The reason I keep the rating is I'm using SVM for this project, and in my opinion keeping the rating will have a better performance than just label them as positive(1) and negative(0). I will try it out to justify my thoughts if I have time. UPDATE: finally decided to use 0 and 1 to help the model prediction.

Data balancing: Only 1500 negative reviews, but 8000 positive reviews, one of the possible ways to implement data balancing is use all negative review and randomly pick 1500 positive reviews from all positive reviews, but that will also cause a lot of information loss(because this may mean people tend to give a better review rather than negative), so I decided to keep all reviews and see what it gives.

Alternative approach: it will also be possible to use nltk or some other library to tokenize the review into words, and that may provide a better accuracy and make the model more general. I also noticed that some value of the bolb sentiment is 0, which because some of them are incomplete sentence or written in other languages like spanish. These can be considered as noisey and dropped.

Part2

Process: first import the data from txt file and reshape it to 2d array, then implement a 5-fold cross validation, create a SVR model, and fit it with training data. Then I met a problem is because I chose to preserve the rating of the reviews, so cannot easily use function like accuracy score to get the accuracy. Then I tried to write a function to calculate the accuracy of the prediction but it turns out to be not very good. Then I find out the result the regression model is almost all 5, this may be because of the data and also the labeling method. I decide to change the label into 0 and 1 to represent negative and positive. After changed data label, the model can be easily fit and predict. With all default parameters, the accuracy of 5 fold is 0.8537933330413072, which looks nice, but depending on the dataset, it's just slightly better than guess all positive.

Features: because the way I choose so the feature is just a number or two represent polarity and subjectivity generated by bolbtext

Model used: SVC,5 Fold cross validation, and used randomizedSearch and manually tuning. The final parameter choose is `C = 4.67702170675376, kernel = 'rbf', degree = 3, gamma = 5` because we have a very imbalanced dataset and not very much information for the sample(long paragraph converted to two numbers which means a lot of information loss) so the C value and gamma value is shown to be high so it will use more information from the data. It's kind of overfitting but that will make the model have a better performance on these category. The choice on the kernel is not surprise "rbf" because of the feature of the dataset.

Tuning: During the process of tuning the model, I found that the accuracy 0.8540091988260545 is the highest possible value I can get, and sadly I figured out that's the value predicting all positive's accuracy. The reasons may be the model is too simple, and the data is not processes good. I tried to tune the model to make it predict more "negative", but seems it only reduce the accuracy. Then I tried two use two features (both polarity and subjectivity), the accuracy can increase a little bit to 0.854225. When I'm doing the tuning I suddenly realize the dataset may be SORTED by default. And go to check the rate.txt manually. As I guessed, almost all 0 are listed at the beginning, that also impact the model's performance. After using KFold with shuffle, the performance again improved to 0.85905. Then using randomized search CV on choosing parameter, it gives accuracy 0.868352, and a minor change manually make it to 0.869, but it may just because of the data is shuffled.

Things to improve: the data can be balanced again and drop the values with 0 which can be considered as noisy. Most things is about cleaning the data, or switch to other models or use boosting/stacking for this.

Part3

Used all data from apparel to train a svm model, and use the same way to generate test X and Y with all.review from Electronic category. And test the model with all samples from Electronic category, the accuracy is 0.79395. This score is slightly better than what I guess, but when looking at the sample of Electronic category, 5000 negative review and 18000 positive reviews, so it's still just a little bit better than betting on all 1(which will have an accuracy 0.781). Then I tried several other categories but it turns out all of them have about 80% positive rate, well I guess shopping is always happy :) But this makes me really curious about the result of training the model by balanced data. I will go back and do balancing stuff on the dataset.

The reason that my model is not performing well may because several reasons, firstly the word used to describe one category is not representing the same meaning for another category, and the model is not generic enough. Also, I choose to not balance the data, this may also be a reason. For example most reviews in apparel are positive, means people are always happy about this category stuff so they may be generative(permissive/ tolerant) on this category, but be very picky on some other category. So the model is still not generic enough. I'm not sure if it should be called overfitting, depends on the intention of the model. If we only want to predict on apparel category then it's reasonable to use only that data, but also, is match with the definition of overfitting, only perform well on training, existing data, bad on prediction. But they are different category data, maybe should be predicted by different model if we want a better performance. However, if we really want to use one model for all category, the model should be more general and use training data from different category so it can perform better.

Part4 (after doing data balancing)

I thought the data balancing will be easy just use what we used on HW5, do newXP = train_X[train_Y==1] and get all negative and positive samples separately, then use random.choice to resample from the positive array and combine them together, but it turns out to be really hard. The dataset structure is very different to the homework. After spending a lot of time but still can't get it, I choose to implement by a naive way. Most of the negative review is on the top of the array, so I just cut the dataset by number index, and combine them together. Another approach can be done by traverse the dataset and drop positive review by a certain chance. After trimming the dataset, the accuracy dropped as I guessed. The accuracy drop to

0.7142857142857143 which is acceptable, but the accuracy for another category will drop to 0.2193924.

This let me think about, is balancing the data a correct operation? For example, if most of the review is positive, do we still have drop most of the data? Is the 80% positive rate also a kind of feature of the data? I believe there’s some other way to clean the data.

Reflection

This assignment is really interesting and inspiring, but it’s also very hard to be honest. Some techniques are never introduced, for example the skill about processing data and transform data to the form that library can use(I haven’t use python and machine learning library before). I know this is an extra credit assignment and we should be able to learn stuff by ourselves, I also know that this is a class focusing on the theory side. But it may also be interesting to cover some methods to really do the project in this class(just a small suggestion! I know there's a lot of things need and worth to be covered in this class). And this assignment really gives us a lot of space to implement a project. Maybe we can have more kaggle and assignment like this in the future. At the end, thanks for instructors and TAs for these semester! Thank you very much.