Competition 2
Report

Team: 鮮榨柳丁汁
H24031281 吳萱萱
H24031370 邱宇傑

Competition Problem: Rating Prediction with User Business Review

- **Data Introduction**

Training Data (training_data.csv): 7997 reviews; the column "stars" is the target that we want to predict.

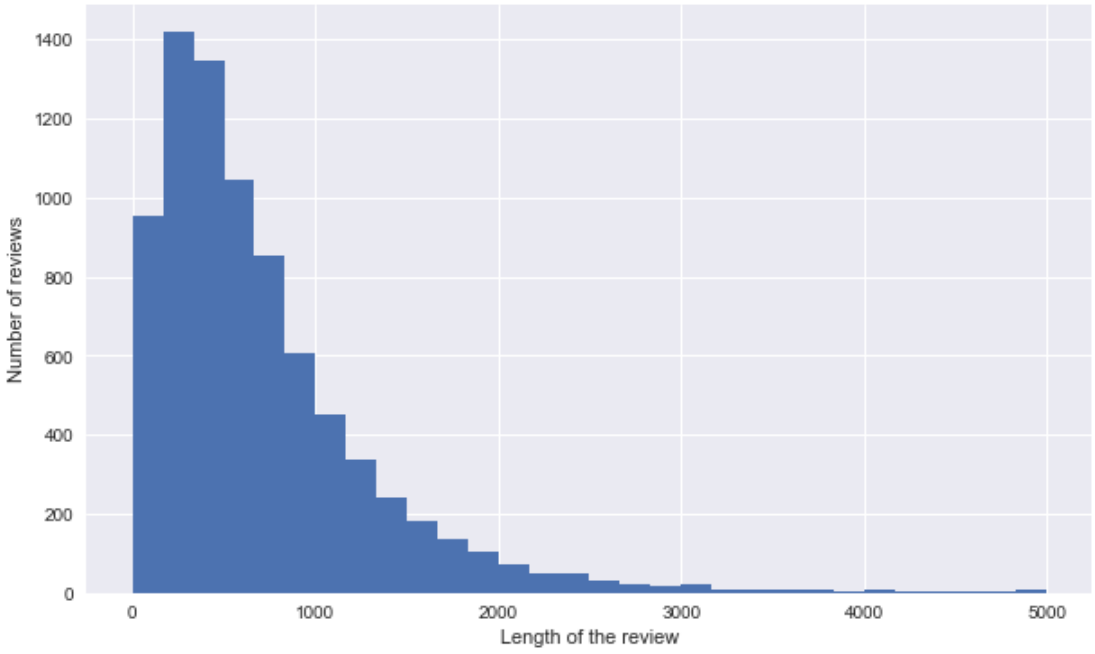
Testing Data (test_data.csv): 2003 reviews without column "stars".

Data Structure:

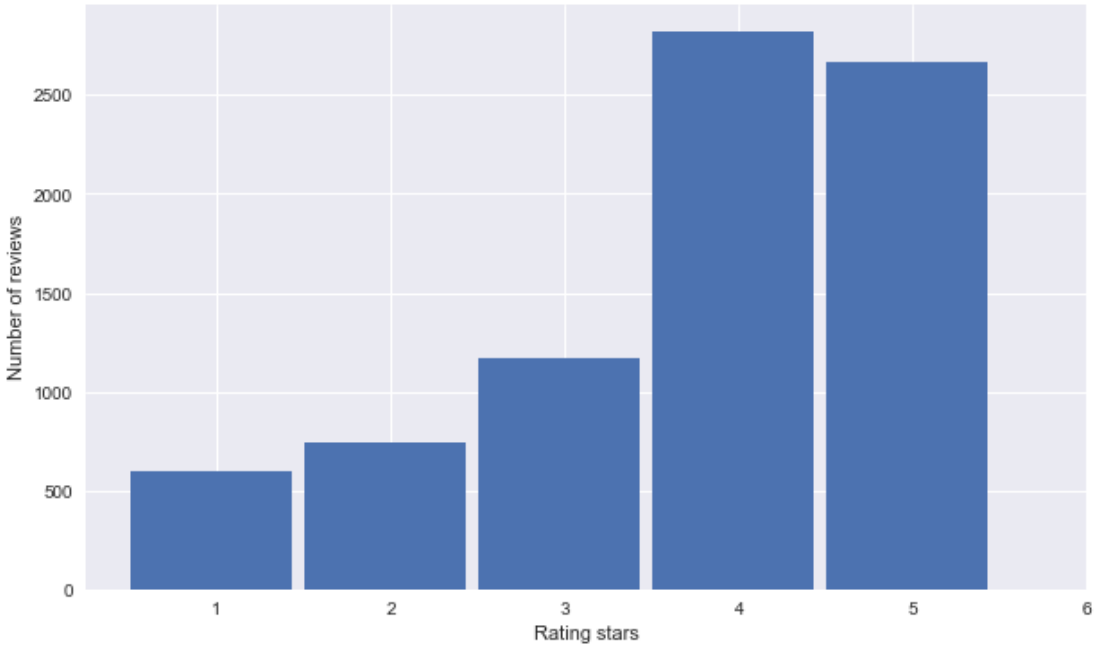
	review_id	business_id	user_id	text	date	stars
0	3223	2055	2533	Sometimes things happen, and when they do this...	2010-12-30	5
1	9938	4165	6371	I know Kerrie through my networking and we ben...	2011-04-26	5
2	7123	869	4929	Love their pizza!!!\nVery fresh. Their cannoli...	2012-09-28	5
3	3601	1603	2789	Being from NJ I am always on the prowl for my ...	2009-06-07	4
4	3948	2347	1245	We have tried this spot a few times and each v...	2011-02-20	4
5	8390	3789	53	This HD is very good. They seem to have knowl...	2012-05-28	4
6	3644	1205	2813	I was initially going to give Riva's 3-stars, ...	2008-02-27	5
7	6689	585	4484	Chino Bandido is a staple for my sister and I....	2012-05-29	5
8	9083	3501	5953	Food--The fire roasted garlic tomato soup is e...	2008-06-06	5
9	4178	2432	1201	Fancy ladies with a few gays sprinkled in. Thi...	2011-09-01	4
10	4263	1208	927	Good food and good service in a convenient loc...	2011-05-23	4

- **Data Preprocessing**

At first, we want to explore all of the reviews, so we calculate the length of each review and plot the chart shown below.



Next, we plot the number of reviews of each rating star, and it can tell that the reviews are mostly with 4 or 5 rating stars. In other words, our data set is unbalanced.



Due to this kind of situation, we'll get less biased prediction if we train the model on a balanced data. Therefore, we use undersampling to balance our data.

```
Counter({4: 2820, 5: 2669, 3: 1168, 2: 741, 1: 599})
Counter({5: 599, 4: 599, 3: 599, 2: 599, 1: 599})
```

Hence, we use the balanced data above to train our model. Before we train our model, we need to do some transformation on our review texts. We use TfidfVectorizer on our texts. This vectorizer breaks text into single words and bi-grams and then calculates the TF-IDF representation. The 'fit' builds up the vocabulary from all the reviews while the 'transform' step turns each indivdual text into a matrix of numbers.

- **Training Model**

Regarding to our data, we know the 'y' of each review in the training set, so we've tried several supervised learning methods to train our model.

- Random Forest Classifier

  We first use GridSearchCV on our training set in order to tune parameters for the random forest classifier, and the result are shown below.

  ```
  {'criterion': 'gini', 'max_depth': 8, 'max_features': 'auto', 'n_estimators': 500}
  0:20:14.719494
  ```

  Therefore, we use the parameters above to train our RFC model, and we predict a 0.4578 accuracy on testing set.

- SVM Classifier

  We use the linear SVC in sklearn.svm, and get the highest accuracy 0.4973.

- Multinomial Naïve Bayes Classifier
  We have searched the method of predicting review texts on the Internet. There are some people using the method Multinomial NB on this kind of data set, so we also give it a try. We get a accuracy with 0.35xx which is not good enough.
- KNN Classifier
  We also use the K-nearest neighbor classifier on this data set. However, the result is not acceptable. (0.2xxx)
- SGD Classifier
  As we searched the method on the internet, we also found that there is a method called Stochastic Gradient Descent, which is used for text classification. We get a pretty good accuracy on this method with 0.4773.

Ranking	Method	Accuracy
1	SVM Classifier	0.4973
2	SGD Classifier	0.4773
3	Random Forest Classifier	0.4578
4	Multinomial Naïve Bayes Classifier	0.35xx
5	KNN Classifier	0.2xxx

Github URL: https://github.com/clairewu0221/cp2.git