IST 664 NATURAL LANGUAGE PROCESSING

Drug Reviews Analysis and Sentiment Classification

Instructor: Prof. Lu Xiao
Group 4: Xiwei Shen, Yanning Fan

Table of Contents

Table of Contents

1. Abstract

The Drug Review Dataset is regarding people's reviews on different kinds of drugs with the condition respectively, including the rate of the satisfaction during usage of the drug. This dataset was collected by crawling online pharmaceutical review sites.
The goal of our project is to study the sentiment polarity of drug reviews, which reflects users' using experience of the drug, including the effectiveness and side effects. We plan to apply several models on the drug review dataset. For this purpose, the dataset has been split into a train dataset (75%) and a test dataset (25%). In the first place, an exploratory data analysis will be conducted to give an overview to the whole dataset. After preprocessing the data, combining with the EDA, we would like to do a sentiment analysis about the drug reviews and using classification models to predict the sentiment label based on customer reviews. Moreover, we also try to predict the rating of the drug based on the reviews by using regression models. We will conduct various models with machine learning algorithms in the analysis, including **Multinomial Naive Bayes**, **Linear Support Vector Machine (LSVM), Random Forest**, **Light GBM** and **Random Forest Regressor**. Finally, we will assess the performance of models and give a conclusion for the project.

2. Introduction

2.1 Overview

Nowadays, for those small health problems, more and more people would like to purchase medicine online, which is convenient and less time-consuming. Meanwhile, choosing a best suitable drug is not as easy as choosing a battery for a remote. Most people lack professional knowledge about drugs, and they can only find some useful information from other customers' reviews, regarding drugs' effectiveness and side effects. Although these reviews are not 100% reliable, they are precious sources for other patients to learn, which came from people who have the same conditions. Therefore, it is useful and meaningful for us to do the drug reviews analysis to help people from a data analysis perspective instead of a medical point.

2.2 Research Questions

The research questions we are trying to solve in this project include:

1. What are the most common conditions?
2. What are the best drugs for each condition?
3. What vectorization method for the reviews is the most efficient and allowing for the most prediction accuracy?
4. What machine learning models work best for predicting the sentiment or rating based on a review?
5. Is this dataset better suited for classification or regression?

3. Dataset Description

3.1 Data Information

The dataset was collected by crawling online pharmaceutical review sites. (Data source: https://www.kaggle.com/jessicali9530/kuc-hackathon-winter-2018) After concatenating train dataset and test dataset, the dataset including 215063 rows for various drugs reviews regarding different drugs and conditions and 7 columns: “uniqueID”, “drugName”, “condition”, “review”, “rating”, “date” and “usefulCount”.

Variable	Type	Description
uniqueID	integer	Unique identifier of a user
drugName	object	Name of drugs
condition	object	Name of condition
review	object	Patients’ reviews
rating	integer	Patients’ rating score (1-10)
date	object	Date of giving reviews
usefulCount	integer	Number of users who think the review is useful

Here we take the first 5 rows to take a look at the content in the dataset.

	uniqueID	drugName	condition	review	rating	date	usefulCount
0	206461	Valsartan	Left Ventricular Dysfunction	"It has no side effect, I take it in combinati...	9	20-May-12	27
1	95260	Guanfacine	ADHD	"My son is halfway through his fourth week of ...	8	27-Apr-10	192
2	92703	Lybrel	Birth Control	"I used to take another oral contraceptive, wh...	5	14-Dec-09	17
3	138000	Ortho Evra	Birth Control	"This is my first time using any form of birth...	8	3-Nov-15	10
4	35696	Buprenorphine / naloxone	Opiate Dependence	"Suboxone has completely turned my life around...	9	27-Nov-16	37

We noticed that columns “uniqueID”, “drugName” and “condition” may have duplicates in each column, to check the unique “uniqueID”, “drugName” and “condition”, we used function “set()” to output the number of “uniqueID”, “drugName” and “condition” without duplicates, and obtained the number is 215063, 3671 and 917 respectively.

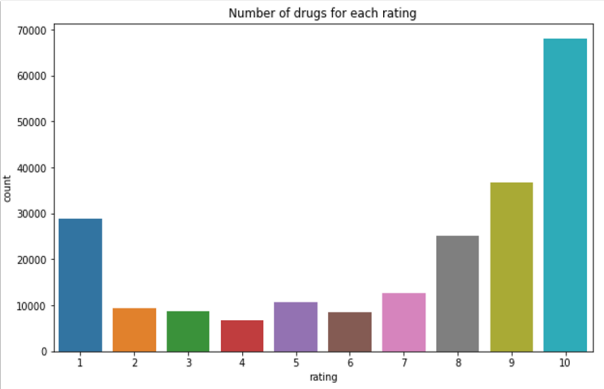
When we checked the dataset, we found that there are some abnormal values in the column “conditions”, which are marked as “users found this comment helpful.”, and they are 1171 in the “conditions” column. Considering they may affect the data analysis and they are not helpful for understanding the data, we decided to remove them. Afterwards, 213892 reviews remained.

3.2 Exploratory Data Analysis

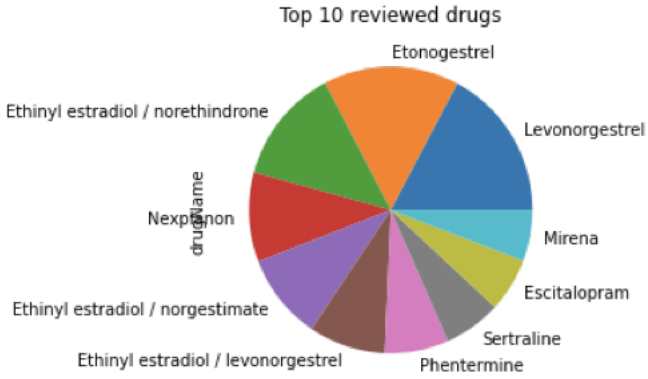
Then we did some exploratory data analysis in order to get some insight about the data. First, we got an average rating score of nearly 7 out of 10. The average number of people who think the review is useful is 28. The review with the maximum number of people showing their attitudes has 1291 people think it is useful.

	rating	usefulCount
count	215063	215063
mean	6.990008	28.001004
min	1	0
max	10	1291

As we can see from the plot about the number of drugs for each rating, most people would like to give 10 points out of 10 to the drug. There are also many people giving 7,8,9 points, showing that they believe that the drug is not perfect but good enough to relieve their illness or resolve their problems. While the third most people like to give 1 point to the drug, showing that they were not satisfied with the effectiveness or side effects of the drug.

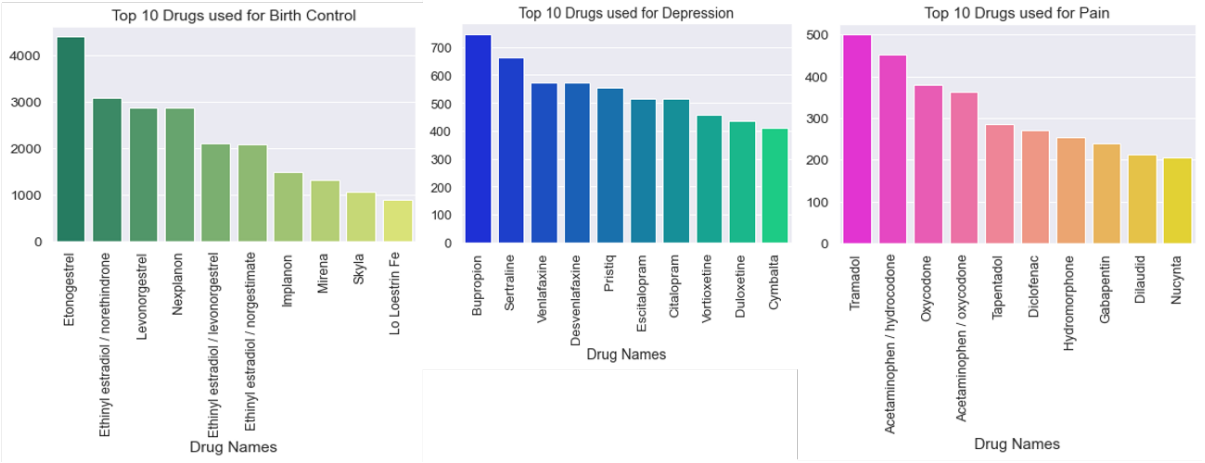


The following left pie chart shows the top10 most popular drugs people are using. We learned that the most popular one is levonorgestrel, which represents the blue part in the chart, and the second most used is the Etonogestrel, which is orange in the chart. And they are all used to birth control. The right pie chart shows top10 most common conditions people are suffering from. As we can see from this chart, birth control, depression and pain are 3 most common conditions people will give reviews for using the corresponding drugs.

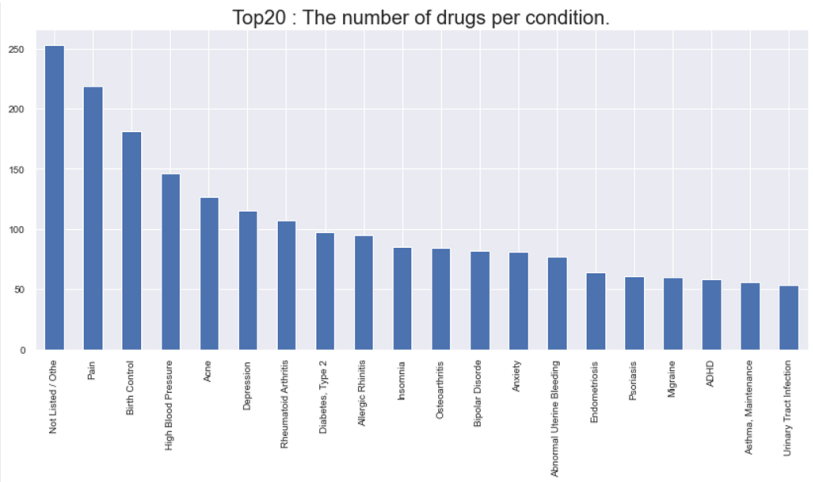


Since there are so many drugs for each condition, it is necessary for us to analyze the reviews to predict what the top-rated drugs are for each condition. Therefore, we draw three plots about the top 10 Drugs used for “Birth Control”, “Depression” and “Pain”, which are the top

3 most common conditions. The Etonogestrel is the most common drug for birth control; Bupropion is the most common drug for depression and Tramadol is most common for pain.



Considering that there are many drugs for the same condition, we conducted a plot to see what kind of conditions have the most kinds of drugs, which may reflect which kind of condition the users are most concerned about. Here is the top 20 the number of drugs per condition. As we can see from the plot, except for the “Not Listed” or “Other”, “Pain”, "Birth Control” and “High Blood Pressure” have the most drugs for them. We inferred that young people accounted for a lot for customers who would like to give reviews. Meanwhile, “Pain” and “High Blood Pressure” are the most common illness among people.



4. Implementation

4.1 Data Pre-processing

For the data preprocessing steps, we first checked the missing values in our dataset. We found that the condition column is the only one that contains missing values, it contains 1194 missing values. Since data that contain missing values are less than 1% of the total amount of data, we just dropped those entries. As we mentioned that during our data exploration process, we found that there are some rows of data containing misleading information in their condition field. These rows of data have the same content in their condition column, which is "users found this comment helpful", but the condition field should include the drug's related conditions so these information are very misleading. Since there are only about a thousand of them, we just drop those data to avoid other influences on our analysis.

Since we want to apply classification algorithms on our dataset for the purpose of sentiment analysis of reviews, we create a new column by categorizing the rating variable and use that to indicate the sentiment polarity of a review. If the rating of a drug is less than or equal to 5, we will classify the review of that drug to be a negative review. If the rating of a drug is greater than 5, we will take the drug's review to be a positive review. Since the review column in our dataset contains raw text. In order to apply machine learning models, we need to convert the text of the review into a data format that can be used by classification models. In this case, we are using the process of vectorization. By vectorizing the “review” column, we can allow different lengths of text to be converted into a numerical format which can be processed by the classifier. For vectorization, there are basically two options, one is Countvectorizer, which convert a collection of text documents to a matrix of token counts, another one is Tfidfvectorizer, which converts a collection of text documents to a matrix of TF-IDF features. We have done some experiments on these two, and it turns out that the Tfidfvectorizer provides a better result. The TF-IDF method will represent the entire text of reviews as a large matrix where each row of the matrix represents one of the reviews and each column represents a token occurrence.

```
from sklearn.feature_extraction.text import TfidfVectorizer
#vectorizer = TfidfVectorizer(use_idf=True, stop_words='english')
#Convert a collection of raw documents to a matrix of TF-IDF features.

vectorizer = TfidfVectorizer(ngram_range=(1,2), stop_words='english', min_df=3)
```

Here is the vectorizer we finally used, and we want the vectorizer to accept both unigram and bigram tokens. Also, it will remove the stop words and ignore tokens that appear in less than 3 documents.

4.2 Train-test Split

The raw dataset has already been split into two datasets “drugsComTrain_raw” and “drugsComTest_raw”, so we used them as train dataset and test dataset directly.

4.3 Modeling

4.3.1 Multinomial Naive Bayes

Multinomial Naive Bayes is a specialized version of Naive Bayes that is designed more for text documents. The features in the multinomial naive Bayes are generated from the simple multinomial distribution, describing the probability of observing the number of counting or score among a series of categories. Meanwhile, in our project, we are aiming to conduct a classification model regarding the reviews and rating score, where the Multinomial Naive Bayes can be applied to.

4.3.2 Linear Support Vector Machine (LSVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be applied to both classification and regression. The key of conducting a linear SVM classifier is to find a line between two groups. The data points will be labeled as one class if they are falling into one side. The criterion of finding that line is to maximize the distance between the line and any data points. The linear SVM is good at and frequently used in the text classification, so we will also use this model in our project.

4.3.3 Random Forest

Random Forest is made up by a large number of individual decision trees, each tree will make their own prediction regarding class separately. Then, the final class prediction is determined by the vote of those decision trees. The key of Random Forest is the low correlation between the models, which prevents trees from the individual errors. Random Forest is very suitable for classification tasks and it is the most popular ensemble method. Therefore, we want to deploy this model on our dataset as well.

4.3.4 Light GBM

Light GBM is a gradient boosting framework that makes use of tree-based learning algorithms. Light GBM is considered to be a very powerful algorithm when it comes to mass computation. It is considered to be a fast processing algorithm.While other algorithms trees grow horizontally, Light GBM algorithm grows vertically, meaning it grows leaf-wise and other algorithms grow level-wise. Light GBM chooses the leaf with large loss to grow. It can lower down more loss than a level wise algorithm when growing the same leaf. LightGBM is called “Light” because of its computation power and giving results faster. LightGBM can handle both small and large volumes of datasets.

5. Results

5.1 Multinomial Naive Bayes

The first model we deployed on our dataset is Multinomial Naive Bayes. This model is easier to achieve, and it has been commonly used in text classification problems. We are using accuracy, precision, and recall as evaluation metrics to check our model performance. For our models, we are using TF-IDF features from the vectorization step as the input and using the defined sentiment label as the target variable.

```
y_train_rating = drug_df.sentiment
clf = MultinomialNB().fit(X_train, y_train_rating)

y_test_rating = drug_test.sentiment
pred = clf.predict(X_test)

print("Accuracy: %s" % str(clf.score(X_test, y_test_rating)))
print("Confusion Matrix")
print(confusion_matrix(pred, y_test_rating))
```

```
Accuracy: 0.7706794337117316
Confusion Matrix
[[ 4028   171]
 [12091 37181]]
```

Based on the output of the Multinomial Naive Bayes model, we can see that the model accuracy is not that good. If we look at the confusion matrix, we can see that there is a very large False Negative value, which is 12091. It means that 12091 reviews were labeled as positive polarity, but the model predicts them to be negative polarity. This might be the reason for such low model accuracy.

	precision	recall	f1-score	support
0	0.96	0.25	0.39	16119
1	0.75	1.00	0.86	37352
accuracy			0.77	53471
macro avg	0.86	0.62	0.63	53471
weighted avg	0.82	0.77	0.72	53471

We used the classification report to check for precision and recall of the model. Based on the recall, we can see that the MNB model did well in identifying positive reviews. The model can correctly identify all of the positive examples. Within those positive reviews identified by the MNB model, 75% of them are actual positive reviews. But for reviews that have been labeled as negative reviews, the MNB model can only identify 25% of them. The result is consistent with what we got from the confusion matrix.

5.2 Linear SVM

The next model we built on our dataset is Linear Support Vector Machine, which is a model that is also commonly used in text classification problems. Starting from Linear SVM, we applied hyperparameter tuning by using the grid search on our models, trying to improve the performance of models. For the Linear SVM, the parameter we need to tune is the cost C, which is the penalty associated with misclassification. This parameter makes SVM become a

soft margin classification, which allows a small number of misclassifications. The model with higher costs will have lower bias but the risk of overfitting. The model with lower cost will have higher bias but lower variance. The initial cost parameter was set to 1. We tried to vary the cost from 0 to 5 and we found that the model has the highest accuracy when the cost parameter equals 2.

```
svm_clf = LinearSVC(C=2).fit(X_train, y_train_rating)
pred3 = svm_clf.predict(X_test)

print("Accuracy: %s" % str(svm_clf.score(X_test, y_test_rating)))
print("Confusion Matrix")
print(confusion_matrix(pred3, y_test_rating))
```

```
Accuracy: 0.9292513699014419
Confusion Matrix
[[13941  1605]
 [ 2178 35747]]
```

The accuracy of Linear SVM model is much higher than the accuracy of Multinomial Naive Bayes. If we check the confusion matrix, the number of False Positive and False Negative data are also in an acceptable range.

	precision	recall	f1-score	support
0	0.90	0.86	0.88	16119
1	0.94	0.96	0.95	37352
accuracy			0.93	53471
macro avg	0.92	0.91	0.92	53471
weighted avg	0.93	0.93	0.93	53471

Based on the classification report, we can see that the Linear SVM did well on identifying both of the reviews that have been labeled as positive and negative. The model can correctly identify 96% of total positive reviews and 86% of total negative reviews. Within those positive reviews identified by the Linear SVM model, 94% of them are actual positive reviews, and for those negative reviews, 90% of them are actual negative reviews. Compared with the MNB model, Linear SVM performed much better on this classification task.

5.3 Random Forest

Next, we built tree-based ensemble models on our dataset, which are Random Forest and Light GBM. These two models usually have better performance, since ensemble methods combine multiple models together. For Random Forest, we have tried to tune three parameters to improve the model performance. The first one is "n_estimators", which is the number of trees in the forest. The second one is "min_samples_split", which is the minimum number of samples required to split an internal node. The third one is "max_depth", which is the maximum depth of a single tree. During the grid search process, we found that the model accuracy will drop significantly if we set a maximum depth of the tree. Therefore, we just keep this parameter as default. Here is our final Random Forest model:

```
rf_clf = RandomForestClassifier(n_estimators = 100, min_samples_split = 2).fit(X_train, y_train_rating)
pred2 = rf_clf.predict(X_test)

print("RF Accuracy: %s" % str(rf_clf.score(X_test, y_test_rating)))
print("Confusion Matrix")
print(confusion_matrix(pred2, y_test_rating))
```

```
RF Accuracy: 0.9068654036767594
Confusion Matrix
[[11429   290]
 [ 4690 37062]]
```

The accuracy of the Random Forest is much higher than the MNB model, but it's not good as Linear SVM. Based on the confusion matrix, we can see that the predicted result of Random Forest is more unbalanced. The model tends to predict more positive reviews to be negative ones.

	precision	recall	f1-score	support
0	0.98	0.71	0.82	16119
1	0.89	0.99	0.94	37352
accuracy			0.91	53471
macro avg	0.93	0.85	0.88	53471
weighted avg	0.91	0.91	0.90	53471

If we look at the classification report, this difference is more obvious. The Random Forest did well on identifying reviews that have been labeled as positive reviews. The model can correctly identify 99% of total positive reviews, but for total negative reviews, this number becomes 71%.

5.4 Light GBM

Gradient Boosting Machine is another popular ensemble method, which is also a set of decision trees. Unlike Random Forest build each tree independently, GBM builds one tree at a time. Since we have a large dataset, in order to reduce the execution time, we used Light GBM in our project. We are following a similar logic with Random Forest by applying grid search on three parameters. The first two parameters are consistent with the Random Forest. The third one is called "num_leaves", which is the max number of leaves in one tree. After the grid search process, we finally set up our model with 2000 trees, maximum depth equals 10, and the maximum number of leaves equal to 80.

```
lgb_clf = LGBMClassifier(n_estimators=2000, max_depth = 10, num_leaves = 80).fit(X_train, y_train_rating)
pred4 = lgb_clf.predict(X_test)

print("Accuracy: %s" % str(lgb_clf.score(X_test, y_test_rating)))
print("Confusion Matrix")
print(confusion_matrix(pred4, y_test_rating))
```

```
Accuracy: 0.9021712704082587
Confusion Matrix
[[12522  1634]
 [ 3597 35718]]
```

The Light GBM has a very similar accuracy with the Random Forest, both of them are slightly over 90%. The difference is that Random Forest doesn't have many False Positive errors, the majority of its errors is False Negative. While for Light GBM, the misclassified examples are more balanced than the Random Forest.

	precision	recall	f1-score	support
0	0.88	0.78	0.83	16119
1	0.91	0.96	0.93	37352
accuracy			0.90	53471
macro avg	0.90	0.87	0.88	53471
weighted avg	0.90	0.90	0.90	53471

Based on the classification report, we can see that the Light GBM also did well in identifying positive reviews. It can correctly identify 96% of positive reviews and 78% of negative reviews. Among all those identified positive reviews, 91% of them are actual positive reviews, while for all identified negative reviews, 88% of them are actual negative reviews.

5.5 Random Forest Regressor

The rating variable in the original dataset contains the value from 1 to 10, and it seems to be a good target variable for regression analysis. In order to solve our last research question, we tried to deploy a regression model, and explore if this dataset is better suited for regression analysis. The model we chose is Random Forest regressor, which is still a tree-based ensemble method but used for regression tasks. The logic to build a regression model is similar with the classification one.

```
reviews = np.vstack((drug_df.review.values.reshape(-1, 1),
                     drug_test.review.values.reshape(-1, 1)))

vectorizer2 = TfidfVectorizer(ngram_range=(1,2), stop_words='english', min_df=3, max_features=1000)
# Vectorize reviews
X = vectorizer2.fit_transform(reviews.ravel()).toarray()

# Get ratings
ratings = np.concatenate((drug_df.rating.values, drug_test.rating.values)).reshape(-1, 1)
y = ratings

X_train, X_test = X[:drug_df.values.shape[0], :], X[drug_df.values.shape[0]:, :]
y_train, y_test = y[:drug_df.values.shape[0]], y[drug_df.values.shape[0]:]
```

We first extract all the review text from the train and test sets and do the TF-IDF vectorization, but this time we only take 1000 most frequent features. For the target variable, we used the rating variable directly and there is no need to predefine a sentiment polarity based on the rating value.

```
rf_reg = RandomForestRegressor()
rf_reg.fit(X_train[:10000], y_train[:10000])

reg_pred = rf_reg.predict(X_test[:10000])
```

Next, we built the Random Forest Regression model on our dataset by using TF-IDF features as the input and rating as the target variable. During the experiment, I found that if we fit the

regression using all of our training set, it will take extremely long to finish. Therefore, I just used the first 10000 entries of the training set to fit the regression model and predict the rating for the first 10000 entries of the test set.

```
mse = mean_squared_error(y_test[:10000], reg_pred)
print('Mean Squared Error (MSE):', mse)
rmse = np.sqrt(mse)
print('Root Mean Squared Error (RMSE):', rmse)
```

```
Mean Squared Error (MSE): 7.734617686732298
Root Mean Squared Error (RMSE): 2.7811180641483557
```

Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) are two metrics that have been commonly used to evaluate a regression model. The MSE of our regression model is around 7.73 and the RMSE of our regression model is only about 2.78. Since we only did a few preprocessing steps and there is no parameter tuning, we would say that the performance of this regression model is really impressive.

6. Conclusion

The research questions that we listed were all properly solved. Two of them were solved during our EDA process, and the rest were solved by our analysis work.

Summary Method	Measurement	Multinomial Naive Bayes	Linear SVM	Random Forest	Light GBM
Accuracy		0.7707	0.9293	0.9069	0.9022
Negative Sentiment Reviews	Precision	0.96	0.90	0.98	0.88
	Recall	0.25	0.86	0.71	0.78
	F1-score	0.39	0.88	0.82	0.83
Positive Sentiment Reviews	Precision	0.75	0.94	0.89	0.91
	Recall	1.00	0.96	0.99	0.96
	F1-score	0.86	0.95	0.94	0.93

Here is a summary table of all of our models. The majority vote of our dataset is about 69%, so the models except Multinomial Naive Bayes do provide improvement for the prediction. Overall, the Linear SVM is the model that works best for predicting the sentiment based on a review, we can see that the model has the highest prediction accuracy, which is nearly 93%. This result indicates that probably our dataset is more linearly separable. The precision and recall of the Linear SVM model also reflect that the model performed well on this classification task. Moreover, these two metrics are relatively balanced on positive reviews and negative reviews, which means the Linear SVM model did well on identifying reviews of both categories. The ensemble methods, which are Random Forest and Light GBM, have a

very similar performance and high enough accuracy. But compared with the Linear SVM, these two models only did well on identifying positive reviews, but not for identifying negative reviews. It seems that these two models are not the optimal option for this classification task. Besides, based on our experiment of the regression model, we also got an impressive result by using reviews to predict the rating variable directly. Therefore, we would say that this dataset is also suitable to be framed as a regression problem. The dataset performed well on both classification and regression approaches.

7. Limitations

The major limitation of our project is that we are not sure about the credibility of the reviews in this dataset. Therefore, determining a sentiment based on reviews may not be too reliable. Unlike other products or services, it is risky if a customer just picks a drug based on its rating and reviews. The idea we had in mind about the solution to this problem is to use another attribute "useful count", which is the number of users who think the review is useful, to reweight the rating or review attributes. But this approach still does not address the review credibility concern, since people can be hired to give reviews and to rate. Therefore, the analysis we have done on this project is more about focusing on this specific dataset itself. It might not be a good option to apply our result to a real-world problem directly. This is a drawback of our project. Another drawback of our project is that since this project is finished by two students, the analysis we conducted is limited. We solved all of our research questions and achieved the general purpose of this project, but we don't have enough time to conduct more additional analysis. For future works, we can do a feature exploration, to find out what are the most important features for different review categories. Since we are dealing with a text classification problem, we can also conduct an error analysis, to find out if there are some common patterns in those misclassified examples.

8. References

[1] Manning, Christopher D., et al. Introduction to Information Retrieval. Cambridge University Press, 2018.
[2] Sullivan, WIlliam. Machine Learning for Beginners Guide Algorithms: Decision Tree & Random Forest Introduction. Healthy Pragmatic Solutions, 2017.