

Sentimental Analysis on Amazon Food Reviews

Spurthi Shetty
Northeastern University
shetty.sp@husky.neu.edu

Ronakkumar Shingala
Northeastern University
shingala.r@husky.neu.edu

Abstract— Sentimental Analysis is one of the platform to find the reviews of every user and distinguish them appropriately. Using Amazon food reviews dataset, we are able to collaborate customer reviews, helpfulness of the review provided, and the prediction value of every unique review. We mainly classify the reviews as Positive, Negative or Neutral reviews. Few algorithms are used to review and understand the overall scenario of the sentiment in each review and extract that particular important aspect of it. By analysing the analysis on sentiments of every review, better view can be given to the customer who try to seek different customer review on particular food.

I. INTRODUCTION

Sentimental Analysis is used for extracting individual or group of words that affect the polarity of any review. Nowadays every food has a review that can be distinguished according to the positive, negative or neutral value, through which a customer can analyse which food product has to be ordered. In this paper, we try to implement different machine learning algorithms for classifying the review value. There are a few reviews which include sarcastic sentences and can't really find the polarity of that review, so sometimes that is termed as neutral review since it includes emoticons. This sentimental analysis is purely based on the reviews that don't include emoticons and all.

With the help of these recommendation systems it actually does help customers to get better perspective on what they should buy. Sometimes when there is all data present but still it is very difficult to predict value of certain product by seeing the raw data. Review can be too long to read or process, so using wordcloud for it is best way to eliminate stopwords. As a result one can do sentimental analysis by this process.

Amazon fine food reviews has led to people's reviews and preferences based on what they like and dislike. This kind of recommendation can help a user if he/she is referred to a certain user can easily solve or make their decision easy. Building a recommendation system according to people's preferences can make the decision much easier and helpful for the user who need to make a certain sophisticated food decisions for Amazon Users according to restaurant's rating.

By using the certain recommendation system for the dataset, it gives us the proper evaluation results of the accuracy metrics applied on the data frame. A new user might lose lot of time in reading or going through reviews provided. We can change this process by gathering all reviews together and only displaying valuable aspects of certain reviews which can help the users and a provides better platform for evaluation of reviews

II. DATASET

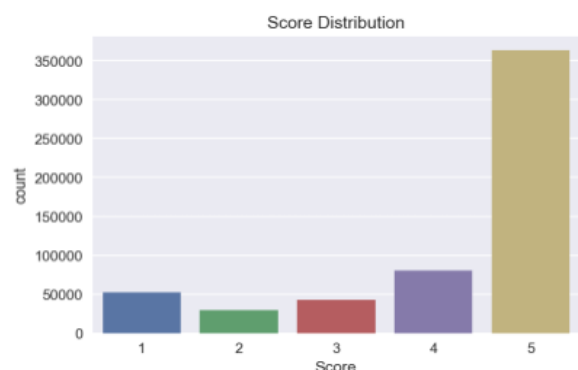
The data frame comprises of Helpfulness numerator and denominator, Prediction Value, Summary, Text, User Id, Product Id of a particular product of a customer and more. The dataset comprises of around 500,000 unique rows. Dataset of amazon fine food reviews is filtered by removing null values. Also the data frame is split into train and test data set and then trained accordingly by applying algorithms. These algorithms are used to measure the accuracy of the model and which model predicts best accuracy rate.


III. EVALUATION METRIC

Evaluation metric that can be done on sentimental analysis allows users to evaluate the reviews or recommendations given to certain products based on Root mean Squared error and Mean absolute error metrics through Cross Validations. Root mean squared error (RMSE) and Mean absolute error (MAE) both are perfect way to find the accuracy for the review system and also they are used to measure the performance and working of recommendation system. Reviews or Ratings can be given from 1 that is meant to be the lowest score for any food product and 5 that is the highest value for any food product. With help of these ratings/prediction values are arrived from test and train data set for further evaluation of the system.

IV. DATA ANALYSIS

Firstly in order to understand clustering model and classifier of the recommendation system we should understand what amount of data is present which needs to be reviewed according to the rating of every customer or reviewer. Sentiment behind every review is different and needs to be properly understood before evaluating like predictions from 1 to 5 are distributed according to individual person's perspective





A bar chart titled "Sentiment Positive vs Negative Distribution". The y-axis is labeled "count" and ranges from 0 to 400,000 with major grid lines every 100,000. The x-axis is labeled "Sentiment" and has two categories: "POSITIVE" and "NEGATIVE". The "POSITIVE" bar is blue and reaches a count of approximately 450,000. The "NEGATIVE" bar is green and reaches a count of approximately 125,000.

Sentiment	count
POSITIVE	450000
NEGATIVE	125000

There are also some unstructured text which do not really specify either they are positive or negative review, sometimes these kind of reviews are specified as neutral reviews since the polarity or estimation cannot be found.

Word cloud is used to visualize certain words of data that represent strong word for any reviews. Also there is a part of stop words, which removes unnecessary words like the is these kinds of words don't hold any importance in the sentimental analysis

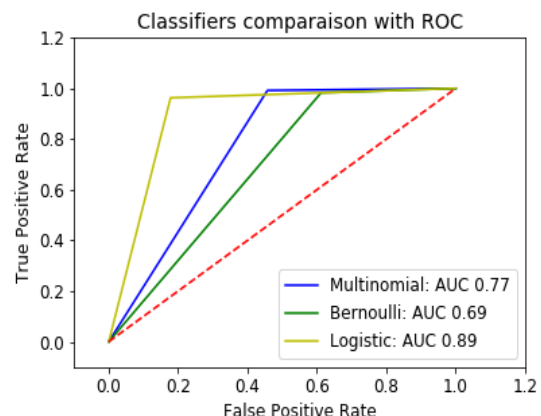
A. K- Means Clustering

K means Clustering Clusters certqin kind of reviews in one cluster, basically it is a kind of classification that is used This enables to make sure that specific related are data are particularly distinguished and comparison can be done.

k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

B. LOGISTIC REGRESSION

Like all regression analyses, the logistic regression is a predictive analysis that is used in amazon food review data set. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. When Comparing it with Multinomial and Bernoulli Bayes method with logistic regression it is found that logistic has good accuracy level.



The ROC curve of Logistic Regression shows the accuracy of 89% Which is the highest when compared to Naïve Bayes algorithm

C. RANDOM FOREST CLASSIFIER

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.[1][2] Random decision forests correct for decision trees' habit of overfitting to their training set.

It is a kind of Decision Tree Classifier which classifies the reviews according to the data present and it looks onto the dependency graph. Like if any review is relatively dependent on any other food review According to this classification Random Forest train the data and generates accuracy of the dataset.

D. Naïve Bayes Classifier

Naïve Bayes remains a popular (baseline) method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines.

Multinomial naïve Bayes

In this data set we have used Multinomial naïve bayes and Bernoulli Naïve Bayes classifier

With a multinomial event model, samples (feature vectors) represent the frequencies with which certain events have been generated by

a multinomial
$$(p_1, \dots, p_n)$$

where
$$p_i$$
 is the probability that event i occurs (or K such multinomials in the multiclass case). A feature vector
$$\mathbf{x} = (x_1, \dots$$

$x_n)$ is then a histogram, with
$$x_i$$
 counting the number of times event i was observed in a particular instance. This is the event model typically used for document classification, with events representing the occurrence of a word in a single document (see bag of words assumption).

Bernoulli naïve Bayes

In the multivariate Bernoulli event model, features are independent booleans (binary variables) describing inputs. Like the multinomial model, this model is popular for document classification

The accuracy rate of Multinomial is higher than Bernoulli bayes classifier.

Which states that Multinomial Naïve bayes is a better prediction or accuracy method for reviews .

VII. Looking at best words by the co-efficient

The below image shows the coefficient of every word that is used in the review.

This method is used to eliminate the slick words that are used in the dataset

With help of the co-efficient of the words highly positive and negative reviews can be easily distinguished and also the review clarification can be done.

	feature	coef
902026	worst	-27.192929
546975	not	-22.979784
766283	terrible	-20.125508
916987	yuck	-19.735845
56939	awful	-18.790482
391675	horrible	-18.179134
530140	nasty	-17.544776
207314	didn	-16.149930
875207	weak	-15.785580
624454	poor	-15.113796
720269	stale	-15.103276
693936	sick	-15.004484
59304	bad	-14.987607
212047	disgusting	-14.431706
547521	not as good as	-14.193296

ESTIMATING THE REVIEWS INDIVIDUALLY

Sample estimated as POSITIVE: negative prob 0.010804, positive prob 0.989196

Sample estimated as NEGATIVE: negative prob 0.999793, positive prob 0.000207

Sample estimated as NEGATIVE: negative prob 0.943927, positive prob 0.056073

This method that is used in to findout the probability of how much of positive or negative words are present which affects the polarity of the summary or the review

Decision Classifier

```
results_svd = pd.DataFrame()

foldnum = 0
tfprediction = {}
cprediction = {}
for name,model in models.items():
    model.fit(tr_features_truncated, Train_labels)
    tfprediction[name] = model.predict(te_features_truncated)
    tfaccuracy = metrics.accuracy_score(tfprediction[name],Test_labels)

    #model.fit(ctr_features_truncated,Train_labels)
    #cprediction[name] = model.predict(cte_features_truncated)
    #caccuracy = metrics.accuracy_score(cprediction[name],Test_labels)

    results_svd.loc[foldnum,'Model']=name
    results_svd.loc[foldnum,'TF-IDF Accuracy']=tfaccuracy
    #results_svd.loc[foldnum,'Count Accuracy']=caccuracy
    foldnum = foldnum+1
print (results_svd)
```

	Model	TF-IDF Accuracy
0	BernoulliNB	0.792402
1	Logistic	0.844697
2	Decision Tree	0.862501
3	Perceptron	0.822134

All the algorithms that are used in this classifier Logistic tends to have better accuracy rate than Bernoulli
But Decision Tree Classifier has the best accuracy of all by implementing SVD.

VIII. DEEP LEARNING METHOD IMPLEMENTATION.

• RECURRENT NEURAL NETWORK

Tuning hyperparameters

For our recurrent neural network we will have to tune the following hyperparameters:

Optimizer hyperparameters:

1.) Learning rate 2.) Minibatch size 3.) Number of epochs

Model hyperparameters:

1.) Number of hidden layers (LSTM layers)
2.) Number of units in the LSTM cells
3.) Dropout rate

Long Short Term Memory

Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). A RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a **cell**, an **input gate**, an **output gate** and a **forget gate**. RNN network keeps on evolving with Recurring phases. Each of the three *gates* can be thought of as a "conventional" artificial neuron, as in a multi-layer neural network: that is, they compute an activation

The next step is to pass the embedded words to the LSTM cells. In the graph, you can observe that there are connections between the first, second and third LSTM cells. In contrast to

other models that assume that all the inputs are independent of each other, these connections here allow us to pass information contained in the sequence of words. One of the strengths of the LSTM cell is the ability to add or remove information to the cell state.

In this LSTM we use Relu and softmax as activation functions Relu is a better match for generating proper epochs and quickly compared to other activation functions

Improvement

Deep Learning techniques tend to outperform every other algorithm when we get access to lots of data. So if we could train our model on a larger dataset, the model should definitely improve. With RNN, the model creates it's own representation of the sentence. The reviews contain vocabulary specific to the food product. If we had more data, our model should be able to identify the specific characteristics that make a good product. In order to improve this model, we should also investigate if we could include the product name/product type. If we had this information maybe our RNN would be able to more easily identify the important characteristics for each product.

```
# evaluate the model
scores = model.evaluate(x_train, y_train)
print("\n%s: %.2f%%" % (model.metrics_names, scores*100))
```

999/999 [=====] - 0s 211us/step

['loss']: 1.53%

Simple RNN

To train the RNN on the review dataset we have to create an array with 200 columns, and enough rows to fit one review per row. Then store each review consisting of integers in the array. If the number of words in the review is less than 200 words, we can pad the list with extra zeros. If the number of words in the review are more than 200 words then we can limit the review to the first 200 words.

```
# Adding the input layer and the LSTM layer
model = Sequential()
model.add(SimpleRNN(128,input_shape = (None,1)))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid',kernel_initializer='uniform'))
model.compile(optimizer='adam', loss='binary_crossentropy')
```

```
y_train = training_table[1:1000]
```

```
model.fit(x_train, y_train, batch_size=8, epochs=100)
```

```
Epoch 1/100
999/999 [=====] - 1s 890us/step - loss: 0.3744
Epoch 2/100
999/999 [=====] - 1s 541us/step - loss: 0.0416
Epoch 3/100
999/999 [=====] - 1s 543us/step - loss: 0.0208
Epoch 4/100
999/999 [=====] - 1s 545us/step - loss: 0.0170
```

IX. FUTURE WORK

In the future, we would augment the current analysis to include review text and user rating evaluations (whether other users thought a particular user's review was funny, useful, or helpful) as features in the prediction model. We would also explore further hybrid approaches and evaluate their performances.

X. REFERENCES

- [1] Burke, R., Hybrid Recommender Systems: Survey and Experiments, <<http://josquin.cs.depaul.edu/rburke/pubs/burke-umuai02.pdf>>.
- [2] Gunawardana A., Shani G, Evaluating Recommendation Systems, <<http://research.microsoft.com/pubs/115396/evaluationmetrics.tr.pdf>>.
- [3] Wikipedia, Singular Value Decomposition, <http://en.wikipedia.org/wiki/Singular_value_decomposition>.
- [4] Recommender Systems, Dimensionality Reduction and the Singular Value Decomposition, <<http://www.cs.carleton.edu/comps/0607/recommend/recommender/svd.html>>.
- [5] Koren, Y., Factorization Meets the Neighborhood: a Multifaceted Col-laborative Filtering Model, <<http://public.research.att.com/volinsky/netflix/kdd08koren.pdf>>.
- [6] Zhou, T., et al., Bipartite network projection and personal recommendation, Physical Review E, 2007. 76(046115)
- [7] Shang, M., Fu, Y., Chen, D., Personal Recommendation Using Weighted BiPartite Graph Projection, Apperceiving Computing and Intelligence Analysis, 2008. ICACIA 2008. International Conference on , vol., no., pp.198,202, 13-15 Dec. 2008
- [8] Breese, J.S., Heckerman, D., Kadie, C., Empirical Analysis of Predictive Algorithms for Collaborative Filtering, <<http://research.microsoft.com/pubs/69656/tr-98-12.pdf>>

LINK FOR CODE DOCUMENTATION:

<https://github.com/shingalaronak/Advance-Data-Science-Final-Project>