PRODUCT RATING PREDICTION BASED ON REVIEWS FROM AMAZON MUSIC

Louis Lu, Wei-Ru Lin

Department of Electrical and Computer Engineering, University of California San Diego

ABSTRACT

Predict products' rating based on reviews of Amazon Electronics products. Applied N-gram, TF-IDF, to train, validate and test and find the relating between review text and review rating.

***Index Terms*—** N-gram, TF-IDF, KNN, Logistic Regression, SVM, sentiment analysis

1. INTRODUCTION

Online shopping is getting more and more popular now. At first, we can only buy books and small products online. But now, almost everything is selling online. Electronics products, groceries, and foods etc. Even cars such large items can be sold online and delivery to your home. After Covid-19 outbreaks, in order to avoid contacting with people, the trend of buying products online is even more unstoppable.

One advantage of online shopping is that customers can give rating and write reviews for the products they buy. They can give high rating and leave positive comments to the products they are satisfied with. On the other hand, if customers dislike the products, they can give them low rating and point out the drawback of products in their reviews.

Most potential customers will also consider the rating and read the reviews of products before buying products. Therefore, analyzing these rating and reviews is important for the manufactures and retailer to strengthen their products and make more profits. In this assignment, we tried to analyze buyers' reviews on electronics products and predict the rating of products so that manufactures can get more detailed inside of their customers and customers' reviews.

2. LITERATURE REVIEW

People understand the sentence in a fraction. However, machines cannot process text data in raw form. They understand the text which is broken down into a numerical format. Bag-of-words and TF-IDF are techniques that convert text sentences into numeric format. Bag-of-words model is utilized in document classification where the frequency of each word which is utilized as a feature for training a classifier. This method can extract features from text documents and these features can be using for training machine learning algorithms and natural language processing. TF-IDF which is a feature term is more important if it has a higher frequency in a text, known as Term Frequency (TF); and feature term is less important if it appears in different text documents in a training set, known as Inverse Document Frequency (IDF).It can be successfully used for words filtering in various subject fields, such as text summarization and classification.

People are interested to find positive and negative opinions shared by other users for features of particular product or service. Sentiment Analysis is a good fit in this application. It is the process of detecting positive or negative sentiment in text, feelings and emotions, urgency or not, and intentions, knows what people comment about product, service topic, issue and event, and aims at extracting opinions and sentiments. Since customers express their feelings and their thoughts everywhere, especially online shopping. They write their feelings or their usage experience online to products. Therefore, Sentiment analysis becomes an essential and effective tool to monitor their customers.

3. DATASET

After researching and discussion, we decided to choose Amazon Review - Electronics category to analyze. This dataset contains 20,994,353 reviews. For the training performance and accuracy, we selected 70,000 reviews from 20,994,353 reviews and spitted them into three potion as the training set, validation set and test set. There are 50000, 10000, 10000 samples in the datasets respectively. All the reviews we extracted are written after date Jan 1, 2010 and the reviewer had been verified as a true buyer.

3.1. Data Schemes

Each records has several features, including overall rating, weather a verified buyer or not, product's ASIN, submission time of review, reviewer's ID, reviewer's name, and review text. The features we are interested in and going to predict are review text and overall rating score. More detail about the features are shown in Table.

3.2. Data Analysis

In this section, we would try to dig out more valuable fact from the dataset. For example, the rating score distribution, rating score distribution over years, popularity of products,

3.2.1. Rating Score Distribution

We extracted the rating score distribution of the dataset to see how people rate the products. The bar chart is shown as Figure 1. We found that most people tend to give their products five points, so most of them are satisfied with the products they bought.

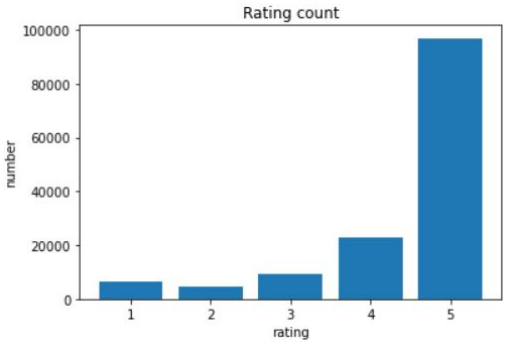

Fig. 1. Rating distribution

3.2.2. Rating score distribution over years

To analyze the recent reviews, we only kept review after Jan 1, 2011. As the Figure 2 shown, most of the reviews we randomly selected from the original data are from 2013 to 2016.

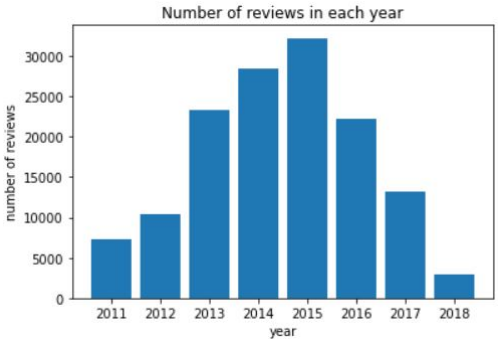

Fig. 2. Number of reviews in each year

3.2.3. Popularity of products

We defined the popularity of products as the number of reviews for certain product over the number of total reviews.

3.2.4. Words with high rating and low rating

We try to figure out what kinds of words would people use when they gave high rating or low rating. To analyze, we classified 4 and 5 as high rating and 1 to 3 as low rating. Then we count the most common words in high rating and low rating. Ignore the neutral words such as 'I', 'the', 'and'. We found that reviews with high rating tend to use positive words like 'great', 'good'.

In the review of low rating, it is interesting that not many people use negative words as we expected. In fact, a potion of reviewers used 'good' in their reviews. These reviewers mentioned some good parts of products and then pointed out the drawbacks. Moreover, we found that people tend to use the words 'will', 'when' and 'if' that don't appear very often in the reviews with high rating. It is because in these low rating reviews, people wrote about their unhappy experiences during some circumstance, so they used words 'if' or 'when' to describe or assume some situations. For example, "If you wanted to make business calls, we think you might be disappointed." "Their mute buttons were not reliable if not attached to phones"

4. PREDICTIVE TASK

In this assignment, we try to predict products' rating based on the reviews' rating using linear regression. The model will be trained using training set, tuned using validation set and tested using test set. Performance of the model will be evaluated using MSE.

4.1. Feature selection

To extract the features for training, we used N-gram and TF-IDF methods.

4.1.1. N-gram

An n-gram is a contiguous sequence of n items from a given sample of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. The n-grams typically are collected from a text or speech corpus.

In this assignment, we used both unigram model and bigram model to extract words. All the punctuations and new line characters in the text are removed before extraing the words. The unigram model extracts individual word from the text. The scheme and the implementation of unigram are easy, but the disadvantages are that it loses the information of the order and the combination of words.

Bigram model stores the partial order of words by preserving the 2-word sequence. It try to solve the problem of unigram that missing the order of words, but it also increases the number of entries. With n-gram model, features such as

word counts or word frequency can be constructed to help us identify the importance of these n-word sequences.

4.1.2. TF-IDF

TF-IDF is short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. It is a combination of test frequency and inverse document frequency, where the two terminologies are defined below.

The definition of TF:

$$td(t,d) = \frac{countoft}{wrodsind} \tag{1}$$

The definition of IDF:

$$idf(t,D) = \log(\frac{N}{|d \in D : t \in d|}) \tag{2}$$

The definition of TFIDF:

$$tfidf(i,d,D) = tf(t,d) \times idf(t,D) \tag{3}$$

The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general.

4.2. Evaluating function

Evaluating the algorithms is an essential part of the project. For evaluating the performance of the models, We chose two methods Classification Accuracy and Confusion Matrix.

4.2.1. Classification Accuracy

Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.

$$Accuracy = \frac{NumberofCorrectPredictions}{TotalNumberofPredictionsMade} \tag{4}$$

4.2.2. Confusion matrix

In the field of machine learning and specifically the problem of statistical classification, a confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class (or vice versa). The name stems from the fact that it makes it easy to see if the system is confusing two classes (i.e. commonly mislabeling one as another).

It is a special kind of contingency table, with two dimensions ("actual" and "predicted"), and identical sets of "classes" in both dimensions (each combination of dimension and class is a variable in the contingency table).

5. DESCRIPTION OF MODELS

In this assignment, we implemented three classification methods, K-Nearest Neighbors, Logistic Regression, and Support Vector Machine.

5.1. Baseline Model

A naive classifier we used is GaussianNB in sklearn. GaussianNB is a classifier that makes predictions using simple rules, which is useful as a simple baseline to compare with other (real) classifiers. The main idea for this classifier is that we generates predictions by respecting the training set's class distribution. It simply takes the unigram segment and make predictions based on the training set.

A normal distribution:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp(\frac{(x_i - \mu_y)^2}{2\sigma_y^2}) \tag{5}$$

5.2. K Nearest Neighbors (KNN)

The first model we implemented is KNN. The KNN algorithm is one of the simplest of all the supervised machine learning algorithms. It simply calculates the distance of a new data point to all other training data points. The distance can be of any type e.g Euclidean or Manhattan etc. It then selects the K-nearest data points, where K can be any integer. Finally the data point is assigned to the class to which the majority of the K data points belong.

For our task, using KNN is straight forward since similar reviews have similar features and they should lead to similar rating on one product. The detailed algorithms is as 2.The benefits of KNN on our task is that as a non-linear classifier, it is good for predictions in complex feature space distribution. However, the original dataset has more than 2,000,000 reviews, we randomly selected 100,000 reviews for training. Otherwise, it took too much time to train the data. Texts are transformed into bag of words and n-gram for analysis in this model.

The disadvantage of KNN is that it doesn't work well with high dimensional data. Because it is more difficult for the algorithm to calculate the distance in each dimension when the number of dimensions is large. Also, the KNN algorithm has a high prediction cost for large datasets, because in large datasets the cost of calculating distance between new point and each existing point becomes higher. Hence, we implemented other models to compare with KNN.

5.3. Logistic Regression

The second model we implemented is Logistic Regression. Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Logistic regression transforms its output using the logistic sigmoid function to return a probability value.

Logistic Regression is a binary classification model, but the class LogisticRegression in scikit-learn supports multi-class case. In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme. It means a class is compared with the rest of the classes. For instance, problem in this assignment is split into 5 binary classification problems:

Problem 1: rating 1 vs [rating 2, 3, 4, 5]
Problem 2: rating 2 vs [rating 1, 3, 4, 5]
Problem 3: rating 3 vs [rating 1, 2, 4, 5]
Problem 4: rating 4 vs [rating 1, 2, 3, 5]
Problem 5: rating 5 vs [rating 1, 2, 3, 4]

After training, validating, and tuning the model, we can get the best result when setting C to 10 and the max iteration number to 1000. The results are shown in the following section and in Table 3.

5.4. Support Vector Machine (SVM)

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they're able to categorize new text.

Even though classic SVM only support binary classification, scikit-learn library do also support multi-class classification problems by SVC class. The multi-class support is handled according to a one-vs-one strategy. One-vs-One strategy splits a multi-class classification into one binary classification problem per each pair of classes. For example, multi-class problem in this assignment is split into 10 binary classification problems:

Problem 1: rating 1 vs rating 2
Problem 2: rating 1 vs rating 3
Problem 3: rating 1 vs rating 4
.
.
.
Problem 9: rating 3 vs rating 5
Problem 10: rating 4 vs rating 5

After training, validating, and trying different combinations of parameters for each schemes, we can get the best predictions when we set C to 1 and the number of iterations to 2000. The result are shown in Table4.

6. RESULT

In this section, we will show the result of our model. Our pipeline for prediction is preprocessing data and extraing features from the data first, then training model using train set, validating using validation set and testing using test set.

6.1. Baseline

We use Gaussian Naive Bayes model as our baseline. The performance of this model is shown in Table 1. Although the accuracy is pretty low, it is the great performance on baseline.

Scheme	Word bag		TFIDF	
Model	unigram	bigram	unigram	bigram
Train	0.436	0.375	0.303	0.362
Test	0.454	0.375	0.332	0.363

Table 1. The Result of GaussianNB

6.2. KNN

We can see the performance of KNN classifier on rating product is as Table 2. In the result of this model, we can see that the accuracy of bigram model is larger than unigram scheme regardless of Word bad or TFIDF. And the result wouldn't works very good in validation set and test set. The reason probably lies on the fact that it does't work well with high dimensional data.

Scheme	Word bag		TFIDF	
Model	unigram	bigram	unigram	bigram
Train	0.734	0.739	0.721	0.733
Validation	0.619	0.663	0.641	0.667
Test	0.623	0.668	0.623	0.668

Table 2. The Result of KNN

6.3. Logistic regression

The accuracy of each scheme using Logistic Regression is shown in Table3. In the result of this model, we found that the accuracy of bigram scheme is lower than unigram scheme no matter the input features are word bag or TF-IDF score. It is interesting because it is opposite to the result of KNN model that the accuracy of bigram sheme is higher than unigram scheme.

The problem we suffered during training is that it kept showing warning telling that the model is not convergent. We tried to normalized the data first before training, but it did

not solve the warning. The second method we tried was increasing the number of iterations. This did solve convergent problems for some scheme such bigram and bigram using TFIDF score, but others still showed the warnings. We guess our dataset might not be very suitable for Logistic Regression model.

Scheme	Word bag		TFIDF	
Model	unigram	bigram	unigram	bigram
Train	0.7423	0.72616	0.7448	0.7178
Validation	0.7193	0.7021	0.7167	0.7074
Test	0.7264	0.7042	0.7253	0.7055

Table 3. The Result of Logistic Regression

6.4. SVM

The accuracy of each scheme using Logistic Regression is shown in Table4. We also found that the accuracy of bigram schemes is lower than the accuracy of unigram schemes that is the same with Linear Regression model but opposite to the KNN model. Before training, validating and testing, we thought the result of SVM would similar with KNN model because both of them calculate the distances between two samples. However, the result did not match our assumption.

Scheme	Word bag		TFIDF	
Model	unigram	bigram	unigram	bigram
Train	0.7344	0.71868	0.7346	0.71848
Validation	0.7177	0.7069	0.7187	0.7074
Test	0.7222	0.705	0.7241	0.7044

Table 4. The Result of SVM

7. SUMMARY

In dataset, we can know that most customers tend to give five points to their product. In model, we use three classifiers, K-Nearest Neighbors, Logistic Regression and Support Vector Machine to implement the performance. The results are shown the performance in Train, Validation, and Test of SVM and Logistic Regression are 70 percentage above. Logistic Regression model isn't suitable to our dataset since the model is not convergent, in spite of normalizing the data before training. The performance of KNN only get 60 percentage above. KNN model doesn't work well with high dimensional data. As a result, we think SVM model is a better classifier in our dataset.

8. REFERENCES

[1] Harris, Zellig (1954). "Distributional Structure".
[2] Minyong Shi, Wenqian Shang, and Zhiguo Hong "Improved Feature Weight Algorithm and Its Application to Text Classification"
[3] Akiko Aizawa "An information-theoretic perspective of tf–idf measures"
[4] Muhammad Zubair Asghar, Aurangzeb Khan, Shakeel Ahmad, Fazal Masud Kundi "A Review of Feature Extraction in Sentiment Analysis"