Project Report : Sentiment Analysis of Amazon Data

1) Introduction

Everyday we come across various products in our lives, on the digital medium we swipe across hundreds of product choices under one category. It will be tedious for the customer to make selection. Here comes 'reviews' where customers who have already got that product leave a rating after using them and brief their experience by giving reviews. As we know ratings can be easily sorted and judged whether a product is good or bad. But when it comes to sentence reviews we need to read through every line to make sure the review conveys a positive or negative sense. In the era of artificial intelligence, things like that have got easy with the Natural Langauge Processing(NLP) technology.

2) What is sentiment analysis?

Sentiment Analysis is the most common text classification tool that analyses an incoming message and tells whether the underlying sentiment is positive, negative our neutral.Understanding people's emotions is essential for businesses since customers are able to express their thoughts and feelings more openly than ever before.It is quite hard for a human to go through each single line and identify the emotion being the user experience.Now with technology, we can automatically analyzing customer feedback, from survey responses to social media conversations, brands are able to listen attentively to their customers, and tailor products and services to meet their needs.

3) Objective of Project

3.1) Reviews Preprocessing and Cleaning

3.2) Story Generation and Visualization from reviews

3.3) Extracting Features from Cleaned reviews

3.4) Model Building: Sentiment Analysis

4) Data

The shape of the data is (row, column):(10261, 9)

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 10261 entries, 0 to 10260

Data columns (total 9 columns):

#	Column	Non-Null Count	Dtype
0	reviewerID	10261 non-null	object
1	asin	10261 non-null	object
2	reviewerName	10234 non-null	object
3	helpful	10261 non-null	object
4	reviewText	10254 non-null	object
5	overall	10261 non-null	float64
6	summary	10261 non-null	object
7	unixReviewTime	10261 non-null	int64
8	reviewTime	10261 non-null	object

dtypes: float64(1), int64(1), object(7)

memory usage: 721.6+ KB

None

4.1) Data Description

This file has reviewer ID , User ID, Reviewer Name, Reviewer text, helpful, Summary(obtained from Reviewer text),Overall Rating on a scale 5, Review time

Description of columns in the file:

reviewerID - ID of the reviewer, e.g. A2SUAM1J3GNN3B

asin - ID of the product, e.g. 0000013714

reviewerName - name of the reviewer

helpful - helpfulness rating of the review, e.g. 2/3

reviewText - text of the review

overall - rating of the product

summary - summary of the review

unixReviewTime - time of the review (unix time)

reviewTime - time of the review (raw)

5) Preprocessing and cleaning

5.1) Handling NaN values:

We got null values in reviewer names and review text. Reviewer names doesn't add any value(we got id's instead) to our objective of the project. So let's focus on review text. I don't

think dropping wouldn't be a problem as there are only 7 null values, but instead I'm thinking to impute that as missing and explore why they didn't leave any review . Could it be due to ratings?

5.2) Concatenating review text and summary

This is an important preprocessing phase, we are deciding the outcome column (sentiment of review) based on the overall score. If the score is greater than 3, we take that as positive and if the value is less than 3 it is negative If it is equal to 3, we take that as neutral sentiment

5.3) Handling time column

Here we have an unusual review time column which has date and year, once we split both we will split the date further into month and date.

5.4) Finding the helpfulness of the review

From the main dataframe we can see the helpful feature with values in list [a,b] format. It says that a out of b people found that review helpful. But with that format, it could not add value to the machine learning model and it will be difficult to decrypt the meaning for the machine. So I have planned to create helpful_rate feature which returns a/b value from [a,b]. The following codeblock contains the complete processing step.

5.5) Review text-Punctuation Cleaning

We have removed all punctuation in our review column

5.6) Review text-Stop words

General nltk stop words contains words like not,hasn't,would'nt which actually conveys a negative sentiment. If we remove that it will end up contradicting the target variable(sentiment). So I have curated the stop words which doesn't have any negative sentiment or any negative alternatives.

6) Extracting Features from Cleaned reviews

Before we build the model for our sentiment analysis, it is required to convert the review texts into vector formation as computer cannot understand words and their sentiment. In this project, we are going to use TF-TDF method to convert the texts

6.1) Stemming the reviews

Stemming is a method of deriving root word from the inflected word. Here we extract the reviews and convert the words in reviews to its root word. for example,
Going->go
Finally->fina
If you notice, the root words doesn't need to carry a semantic meaning. There is another technique knows as Lemmatization where it converts the words into root words which has a semantic meaning. Simce it takes time. I'm using stemming

6.2) TFIDF(Term Frequency — Inverse Document Frequency)

TF-IDF stands for “Term Frequency — Inverse Document Frequency”. This is a technique to quantify a word in documents, we generally compute a weight to each word which signifies the importance of the word in the document and corpus. This method is a widely used technique in Information Retrieval and Text Mining. Here we are splitting as bigram (two words) and consider their combined weight. Also we are taking only the top 5000 words from the reviews.

6.3) Handling Imbalance target feature-SMOTE

In our target feature, we noticed that we got a lot of positive sentiments compared to negative and neutral. So it is crucial to balanced the classes in such situatio. Here I use SMOTE(Synthetic Minority Oversampling Technique) to balance out the imbalanced dataset problem.It aims to balance class distribution by randomly increasing minority class examples by replicating them.
SMOTE synthesises new minority instances between existing minority instances. It generates the virtual training records by linear interpolation for the minority class. These synthetic training records are generated by randomly selecting one or more of the k-nearest neighbors for each example in the minority class. After the oversampling process, the data is reconstructed and several classification models can be applied for the processed data.

7) Model Building: Sentiment Analysis

As we have successfully processed the text data, not it is just a normal machine learning problem. Where from the sparse matrix we predict the classes in target feature.

7.1) Model selection

First select the best peforming model by using cross validaton. Let's consider all the classification algorithm and perform the model selection process

7.2) **Logistic Regression with Hyperparameter tuning**

We use regularization parameter and penality for parameter tuning. let's see which one to plug.
We have got 94% accuracy. That ain't bad. But for classification problems we need to get confusion matrix and check f1 score rather than accuracy

7.3) Classification metrics

Check out the diagonal elements(2326+2195+1854), they are correctly predicted records and rest are incorrectly classified by the algorithm

8) Conclusion

We have done a pretty neat job on classifying all the classes starting from splitting the sentiments based on overall score,text cleaning, customize the stopwords list based on requirement and finally handling imbalance with smote. Here are few insights from the notebook.

- Consider welcoming ngram in sentiment analysis as one word can't give is proper results and stop words got to be manually checked as they have negative words. It is advised to avoid using stop words in sentiment analysis
- Most of our neutral reviews were actual critic of product from the buyers, so amazon can consider these as feedback and give them to the seller to help them improve their products
- Most of the reviews in this dataset were about string instruments such as guitar.

- Balancing the dataset got me a very fruitful accuracy score. Without balancing, I got good precision but very bad recall and inturn affected my f1 score. So balancing the target feature is important
- In sentiment analysis, we should concentrate on our f1 score where we got an average of 94% so we did a pretty good job.