CIS 4130 CMWA

Professor Richard D Holowczak

Tulasa Chitrakar

tulasa.chitrakar@baruchmail.cuny.edu

Proposal

The dataset that I will be working on is the Amazon Customer Review dataset. Customers of Amazon have given feedback for products on amazon.com. People give a rating to different products and also leave a review. I intend to predict the ratings for any product based on the words used in the content of the reviews.

Some of the attributes in this dataset are as follows:

- marketplace: Country code of the marketplace where the review was written
- customer_id: unique id used for review writers
- product_title: Title of the product
- product_category: category of the product that can be used to group them
- star_rating: a 1 to 5 rating for the product
- helpful_votes: number of helpful votes
- review_headline: the title of the review
- review_body: the entire text of the review
- review_date: the date the review was written

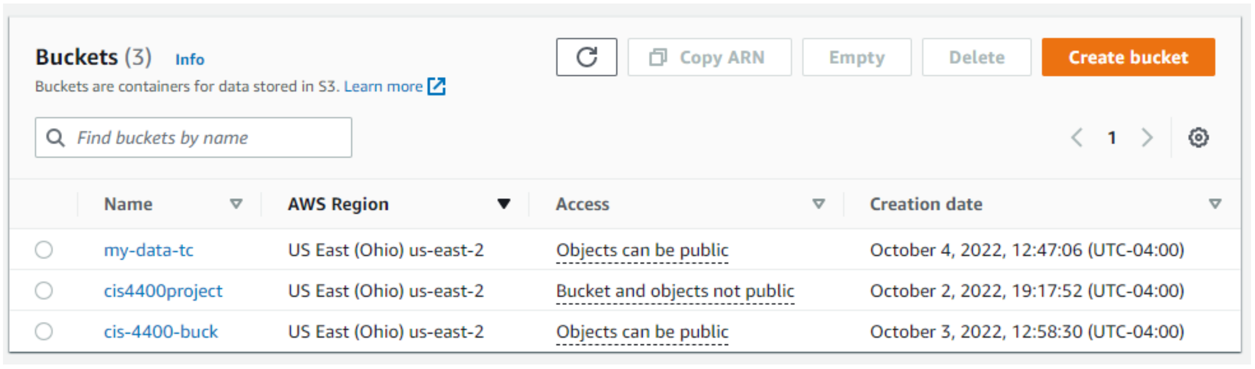
With this information, I will be able to categorize the products and observe the words used in reviews for certain types of products. As a result, I will be able to predict the star rating of the products using those words. The dataset for the Amazon Customer Reviews can be found at the following link: Amazon Customer Reviews Dataset

Data acquisition

Firstly, I was able to find a dataset in Kaggle but since the reviews were done on Amazon, I was able to find the dataset on AWS S3. To store the dataset to my bucket on S3, I first created a bucket using the Command Line Interface (CLI) through the instance on EC2 created for this project. Once I opened the CLI, the code I used to create the bucket is: aws s3api create-bucket --bucket cis4400project –region us-east-2 --create-bucket-configuration LoactionConstraint=us-east-2

The next step was to add the dataset to our bucket. First, I made a folder on the bucket called 'amazon-customer-review' to store the dataset. Amazon had already categorized them and had separate URLs for them. For every URL, I used the "curl -SL https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Apparel_v1_00.tsv.gz | aws s3 cp - s3://cis4400project/amazon-customer-review/apparel.tsv" code. The folder on the bucket and the datasets in that folder look as follows:

S3 buckets:



Buckets are containers for data stored in S3. Learn more

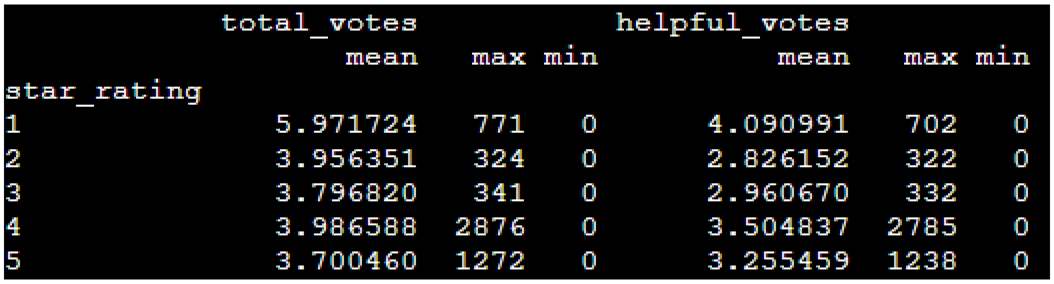
	Name	AWS Region	Access	Creation date
○	my-data-tc	US East (Ohio) us-east-2	Objects can be public	October 4, 2022, 12:47:06 (UTC-04:00)
○	cis4400project	US East (Ohio) us-east-2	Bucket and objects not public	October 2, 2022, 19:17:52 (UTC-04:00)
○	cis-4400-buck	US East (Ohio) us-east-2	Objects can be public	October 3, 2022, 12:58:30 (UTC-04:00)

Exploratory Data Analysis

For this part, I used the EC2 CLI to get the descriptive statistics of the dataset. There are various ways I was able to get the statistics like count, mean, minimum value, and maximum value of the numeric columns of the dataset. However, I used the .describe() function in the panda library. The code I used for this is as follows:

```
>>> import boto3
>>> import pandas as pd
>> files =
['https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Gift_Card_v1_00.tsv.gz',
'https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Personal_Care_Appliances_v1_00.tsv.gz']
>>> for f in files:
...     df = pd.read_csv(f, sep = '\t', error_bad_lines = False)
...     df_allFiles = df.append(df, ignore_index = True)
>>> stats = df_allFiles.groupby('star_ratings').agg({'total_votes':['mean', 'max', 'min'],
'helpful_votes:['mean', 'max', 'min']})
>>> print(stats)
```

I only used two categories as it would require more processing time to do all the categories at once. Using the above code, I created and saved the two categories together as df_allFiles. Using the groupby function, I was able group them using ‘star_ratings’ to get the meam, maximum value, and minimum value of df_allFiles. The statistical results for the above mentioned categories are as follows:



```
            total_votes           helpful_votes
                   mean   max min          mean   max min
star_rating
1              5.971724   771   0      4.090991   702   0
2              3.956351   324   0      2.826152   322   0
3              3.796820   341   0      2.960670   332   0
4              3.986588  2876   0      3.504837  2785   0
5              3.700460  1272   0      3.255459  1238   0
```

Coding and Modeling

To build a model to predict the star ratings using the reviews on amazon, I had the following outline:

→ importing all the necessary libraries

→ creating a dataframe of the dataset using read.csv()

→ getting a random sample of the data

→ dropping the records with null values

→ creating a count of review words

→ dropping the columns I didn't need to make the prediction

→ splitting the data into training and testing sets in 70-30 ratio

→ creating a variable to do the logistic regression

→ creating an indexer, encoder, and assembler

→ creating a pipeline using the above variables

→ creating a grid to hold hyperparameters

→ building a parameter grid

→ creating a binary classification evaluator to see how well the model works

→ creating a cross-validation using the hyperparameter grid

→ training the models

→ testing the predictions

→ calculating the area under the ROC curve

→ creating a confusion matrix

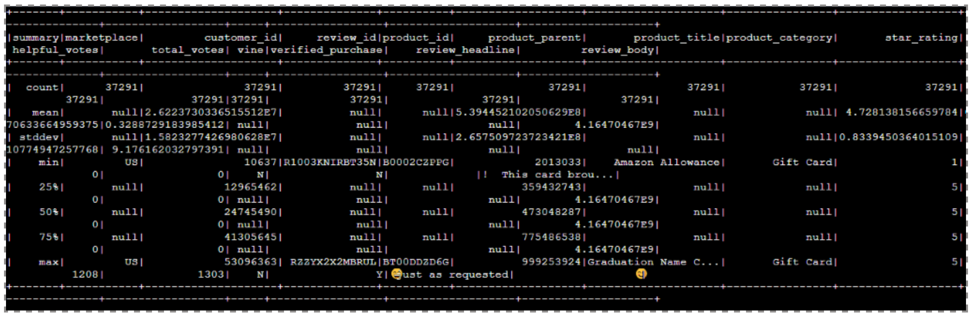
→ Getting a stage of the pipeline to get the ROC curve

→ extracting the coefficients on each of the variables

→ extracting the original column names and storing them in a dictionary

→ printing the coefficients for the variables using a loop

First, I imported all the libraries that was necessary to do this modeling successfully. Some of the libraries are pandas, numpy, and functions like StringIndexer, OneHotEncoder, VectorAssembler, Pipeline, LogisticRegression, BinaryClassificationEvaluator, CrossValidation, ParamGridBuilder from pyspark's machine learning library. Next, I used aws s3 cp s3://amazon-reviews-pds/tsv/amazon_reviews_us_Gift_Card_v1_00.tsv.gz . to copy my files locally and used hdfs dfs -put amazon_reviews_us_Gift_Card_v1_00.tsv.gz hdfs:/// to put them into my cluster. Once I was able to do that, I created a dataframe using spark.read.csv(). After doing so, I got a sample of 25% of my dataset, dropped records with null value using df.na.drop() function and got a summary of the dataframe as follows:

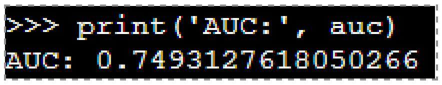


```
+-------+-----------+--------------------+--------------+----------+--------------------+--------------------+----------------+--------------------+
-------------+-------------------+-----+-----------------+-------------------+-------------------+
|summary|marketplace|         customer_id|     review_id|product_id|      product_parent|       product_title|product_category|         star_rating|
 helpful_votes|        total_votes| vine|verified_purchase|    review_headline|        review_body|
+-------+-----------+--------------------+--------------+----------+--------------------+--------------------+----------------+--------------------+
-------------+-------------------+-----+-----------------+-------------------+-------------------+
|  count|      37291|               37291|         37291|     37291|               37291|               37291|           37291|               37291|
        37291|              37291|37291|            37291|              37291|              37291|
|   mean|       null|2.622373033651551 2E7|          null|      null|5.394452102050629E8|                null|            null| 4.728138156659784|
70633664959375|0.3288729183985412| null|             null|               null|        4.16470467E9|
| stddev|       null|1.5823277426980628E7|          null|      null|2.657509723723421E8|                null|            null|0.8339450364015109|
10774947257768| 9.176162032797391| null|             null|               null|               null|
|    min|         US|               10637|R1003KNIRBT35N|B0002CZPPG|             2013033|    Amazon Allowance|       Gift Card|                   1|
            0|                  0|    N|                N|                 |!  This card brou...|
|    25%|       null|            12965462|          null|      null|           359432743|                null|            null|                   5|
            0|                  0| null|             null|               null|        4.16470467E9|
|    50%|       null|            24745490|          null|      null|           473048287|                null|            null|                   5|
            0|                  0| null|             null|               null|        4.16470467E9|
|    75%|       null|            41305645|          null|      null|           775486538|                null|            null|                   5|
            0|                  0| null|             null|               null|        4.16470467E9|
|    max|         US|            53096363| RZZYX2X2MBRUL|BT00DDZD6G|           999253924|Graduation Name C...|       Gift Card|                   5|
         1208|               1303|    N|                Y|😁ust as requested|                😃|
+-------+-----------+--------------------+--------------+----------+--------------------+--------------------+----------------+--------------------+
-------------+-------------------+-----+-----------------+-------------------+-------------------+
```

Furthermore, I created a count for review words by adding a column into the dataframe. I also added a column to get the binary output level of the star rating over 3 as 1.0, and the rest as 0.0. Then, I dropped the columns I wouldn't need to get my prediction such as “marketplace”, and “customer_id”. To split the dataset into 70-30 ratio, I used the randomSplit function and 3456 as the seed.

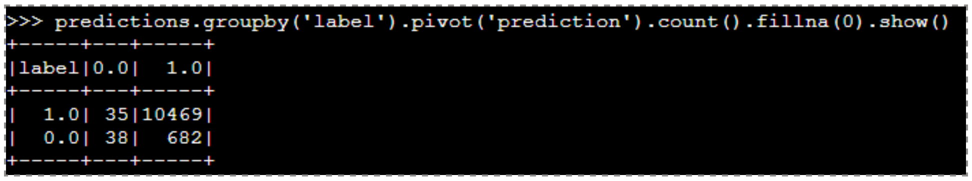
One of the most important steps of building the model is to creating indexer, encoder, and assembler using the Indexer, OneHotEncoder, and VectorAssembler respectively. Once, this step was done, I created a LogisticRegression Estimator ans using those values as the stages, I was able to create a pipeline. Next, I created a grid using ParamGridBuilder and added regParam and elasticNetParam for logistic regression. After, building the grids to hold the hyperparameters, I created an evaluator using BinaryClassificationEvaluator to see how was the model doing. After this, using the hyperparamter grids, I created a CrossValidator estimator.

The next step of this process was to train and test the data. Using the .fit() function to train the data and .transform() to test the data, I was able to get a value for the area under the ROC curve as follows:



```
>>> print('AUC:', auc)
AUC: 0.7493127618050266
```

By creating a confusion matrix, I was able to see the true negative [0][1], false negative [1][1], false positive [0][2], and true positive [1][2] values.

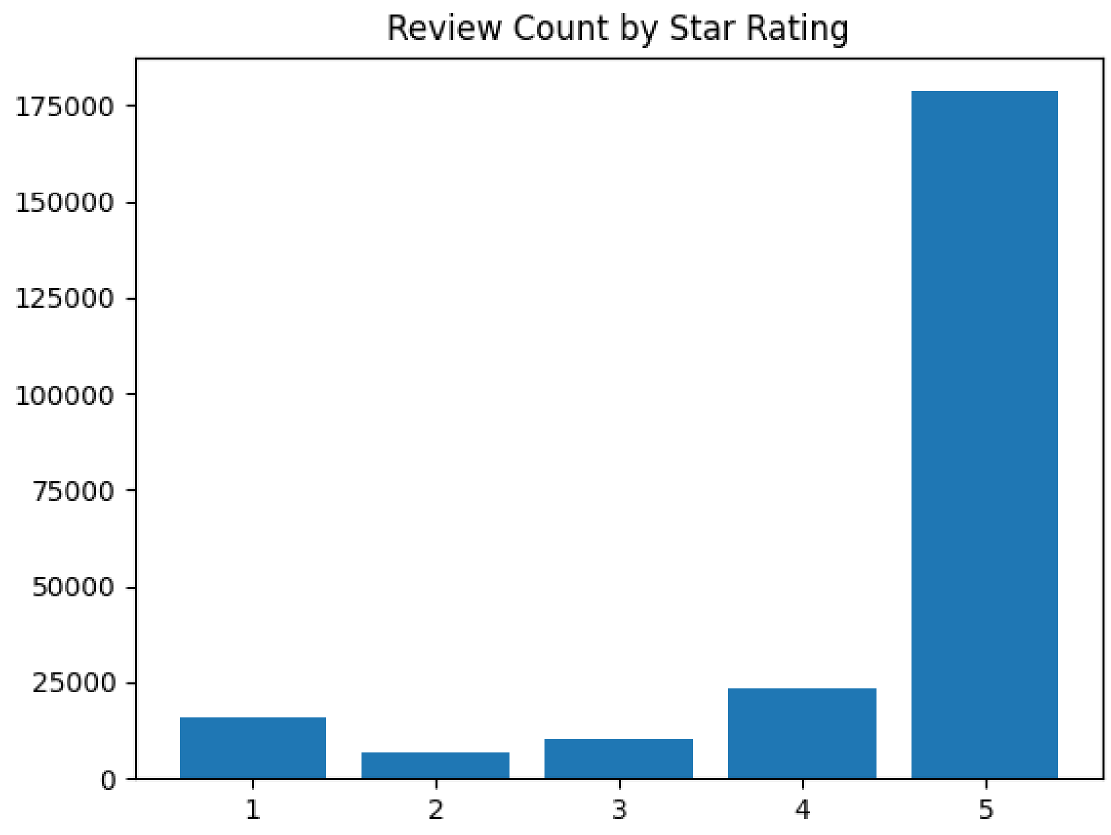


```
>>> predictions.groupby('label').pivot('prediction').count().fillna(0).show()
+-----+---+-----+
|label|0.0|  1.0|
+-----+---+-----+
|  1.0| 35|10469|
|  0.0| 38|  682|
+-----+---+-----+
```

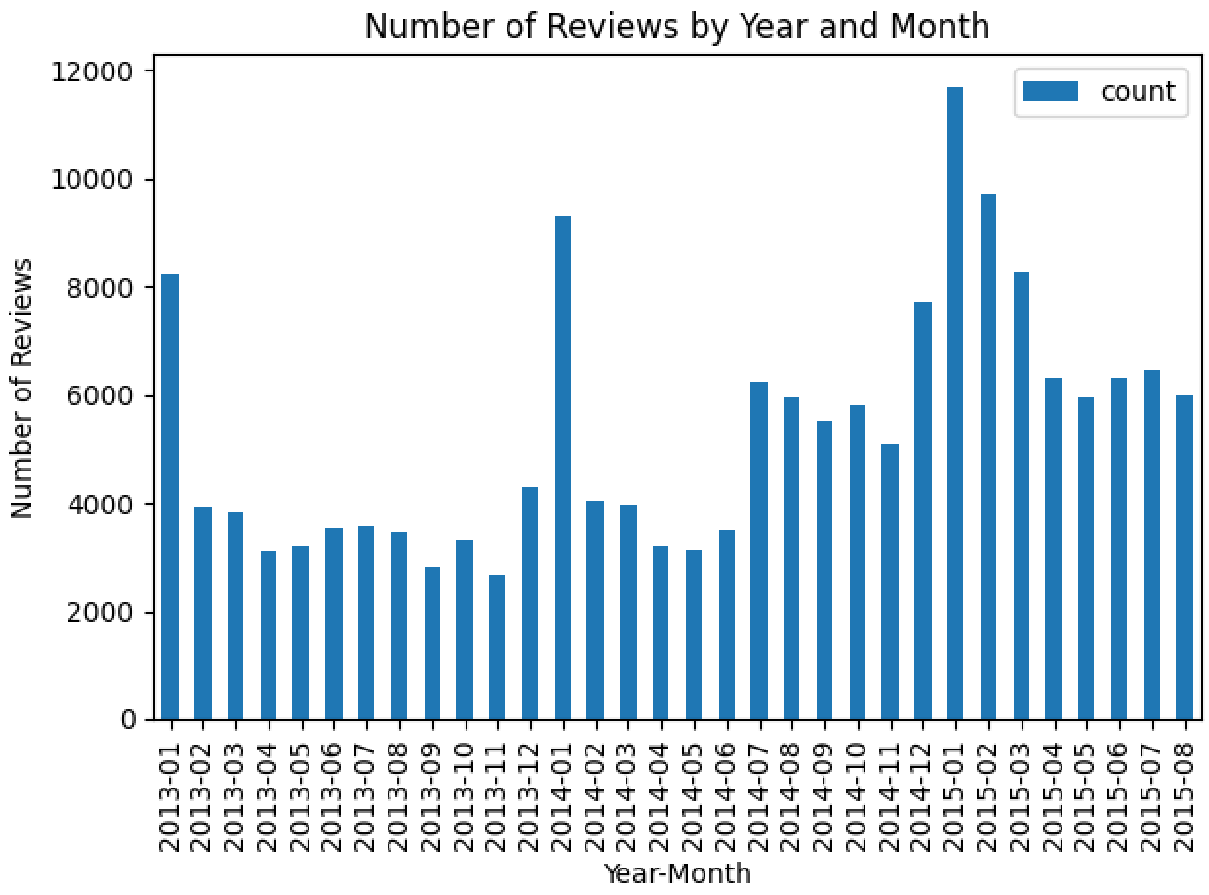
By using these values, I was able to calculate the values for precision, recall, accuracy and f1 score of my data. Next, I used cv.bestModel from the pipeline create graphs for the true positive and negative values along the ROC curve.

Visualizing Results

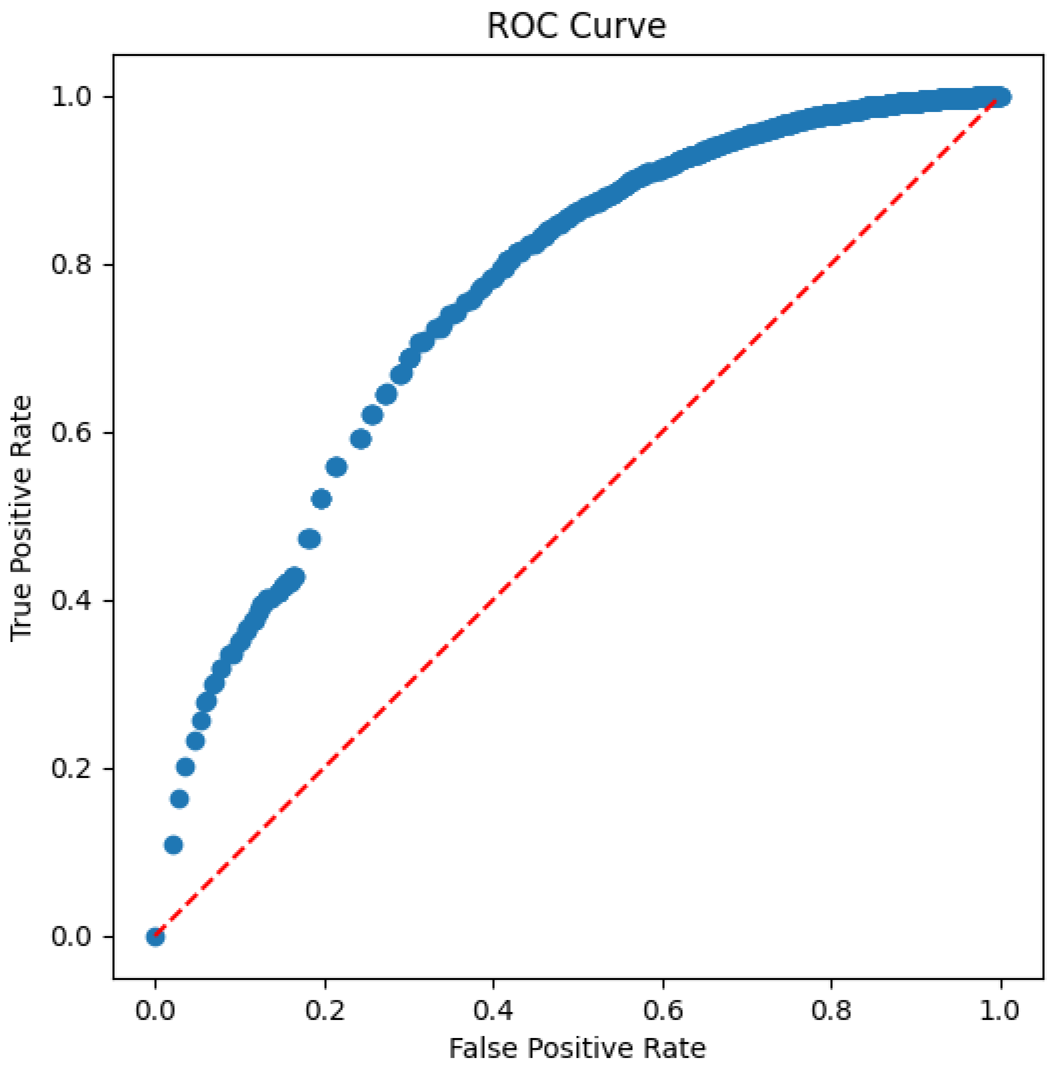
The following graphs represents the findings of the prediction:



From this graph, we can see that the review count is higher for 5-star rating.



In the month of January, the reviews on the products are higher from 2013-2015.



Here, we can see that the true positive and true negative along the ROC curve.

Summary and Conclusion

To conclude this project, there are many ways to do a prediction of various columns of a dataset. On my mission to predict the star ratings of gift cards and personal care appliances, I was able to see that most of the predictions were a false negative, which means the predicted values were not able estimate true rating of a product using the reviews. However, it was able to get the true outcome of the prediction of true values as well. And since, I only used two categories among various other categories, I might be able to get better results using a bigger sample size than 25%. Lastly, looking at the visualization, we can see that the month of January was the highest month to get the reviews and ratings as it was right after holiday season, and gift cards and personal care appliances would’ve been the perfect holiday gifts. The following code can also be found here: https://github.com/tulasa54/Amazon_Review_CIS_4130

Citation

- https://bbhosted.cuny.edu/webapps/blackboard/execute/content/file?cmd=view&content_id=_72299946_1&course_id=_2142454_1