Sentiment Analysis of Amazon Reviews

Table of Contents

Introduction – Background

Amazon.com, Inc. is an American multinational technology company based in Seattle, Washington, which focuses on e-commerce, cloud computing, digital streaming, and artificial intelligence. Amazon Customer Reviews (a.k.a. Product Reviews) is one of Amazon's iconic products. In a period of over two decades since the first review in 1995, millions of Amazon customers have contributed over a hundred million reviews to express opinions and describe their experiences regarding products on the Amazon.com website. This dataset provides an interesting opportunity by analyzing review data and provide actionable analytics on overall sentiment such as positive, negative and give abstract summary by providing key describing terms. For this project, we will be focusing on specific category Electronics. We analyzed the dataset describing reviews of all Electronics products to understand various trends and able to find overall sentiments by using library PyCaret.

Dataset:

Over 130+ million customer review are available to researchers as part of this release. The review data is available in TSV files in the ***amazon-reviews-pds*** S3 bucket in AWS US East Region. Each line in the data files corresponds to an individual review (tab delimited, with no quote and escape characters).

Below table provides more information about each column in the dataset.

Column Name	Description
marketplace	2 letter country code of the marketplace where the review was written.
customer_id	Random identifier that can be used to aggregate reviews written by a single author.
review_id	The unique ID of the review.
product_id	The unique Product ID the review pertains to. In the multilingual dataset the reviews
product_parent	Random identifier that can be used to aggregate reviews for the same product.for the same product in different countries can be grouped by the same product_id.
product_title	Title of the product.
product_category	Broad product category that can be used to group reviews. (also used to group the dataset into coherent parts).
star_rating	The 1-5 star rating of the review.
helpful_votes	Number of helpful votes.
total_votes	Number of total votes the review received.
vine	Review was written as part of the Vine program.
verified_purchase	The review is on a verified purchase.
review_headline	The title of the review.
review_body	The review text.
review_date	The date the review was written.
review_date	The date the review was written.

We will be scoping the specific category Electronics for this project. The dataset for Electronics category can be downloaded locally via link https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Electronics_v1_00.tsv.gz.

Data Preparation and Cleaning

We will download zipped version the dataset directly from Jupyter notebook. Jupyter notebook is an open source web application that allows you to create and share documents that contain live code, visualization, and narrative text. We will do all the steps associated with the dataset in the interactive jupyter notebook. We will use read_csv method of Pandas library to directly read from the downloaded zip links as show in fig 1.

```
df_electronics = pd.read_csv('./data/amazon_reviews_us_Electronics_v1_00.tsv.gz',
                             sep="\t",
                             header=0,
                             error_bad_lines=False)
```

Figure 1 Read dataset from zipped version

First, we will look at the datatypes of each field in the dataset.

```
df_electronics.dtypes
```

```
marketplace          object
customer_id           int64
review_id            object
product_id           object
product_parent        int64
product_title        object
product_category     object
star_rating           int64
helpful_votes         int64
total_votes           int64
vine                 object
verified_purchase    object
review_headline      object
review_body          object
review_date          object
dtype: object
```

Figure 2 Datatypes of each column in the dataset

As you can notice, for review_date column which has datetime records is not in correct datatypes, so lets change that via pandas function ***to_datetime().***

```
df_electronics['review_date'] = pd.to_datetime(df_electronics.review_date)
```

Figure 3 Change datatype of review_date column

Let's check null records in each columns of the dataset.

```
null_columns = df_electronics.columns[df_electronics.isnull().any()]
df_electronics[null_columns].isnull().sum()
```

```
product_title       4
review_headline    31
review_body        88
review_date        24
dtype: int64
```

Figure 4 Check null values in each column of dataset

For the scope of this dataset , we have selected only few columns – review_id, customer_id, star_rating, review_headline, review_body.

```
df_reviews = df_electronics[['review_id','customer_id','star_rating','review_headline','review_body']]
df_reviews.isnull().sum()
```

```
review_id           0
customer_id         0
star_rating         0
review_headline    31
review_body        88
dtype: int64
```

Figure 5 select columns and check null

We will drop the records with null to prepare for further analysis.

```
df_reviews = df_reviews.dropna()
df_reviews.isnull().sum()
```

```
review_id          0
customer_id        0
star_rating        0
review_headline    0
review_body        0
dtype: int64
```

Figure 6 Drop null records from dataset

For Natural Language Programming (NLP) analysis, we will be using PyCaret library to load the sample of the data.

```
# sampling the data to select only 1000 documents
data = df_reviews.sample(1000, random_state=786).reset_index(drop=True)
data=data.drop(['star_rating'], axis=1)
data.head()
```

	review_id	customer_id	review_headline	review_body	sentiment
0	R3FCHTWDX6M0NY	43758182	Five Stars	nice!	Negative
1	R1GNTVXZQMKWHW	30156333	as advertised	had to build custom boxes and this worked perf...	Negative
2	R1UXFD06PQXGF8	47478367	I need help!	The instructions are so difficult I cant get i...	Positive
3	RSPOKAESK9MEY	36856462	It's a battery.	It's a battery.	Negative
4	R25JSD0FT6NTJC	23678831	Replaiment	I have not uses it yet but I'm sure it will wo...	Positive

Figure 7 Load sample data via PyCaret

We will later use its setup method to prepare and clean dataset. The set up will automatically clean the dataset by performing following checks. We will also select target column as review_body which contains review text and on which we want to perform sentiment analysis.

- Removing numeric characters
- Removing special characters
- Word tokenization
- Stopword removal
- Bigram extraction
- Trigram Extraction
- Lemmatizing
- Custom Stopwords

```
from pycaret.nlp import *
exp_nlp101 = setup(data = data, target = 'review_body', session_id = 123)
```

Description	Value
session_id	123
Documents	1000
Vocab Size	3668
Custom Stopwords	False

Figure 8 Setup the environment via PyCaret

Once the setup is successfully executed it prints the information grid with the following information:

- session_id: A pseudo-random number distributed as a seed in all functions for later reproducibility. If no session_id is passed, a random number is automatically generated that is distributed to all functions. In this experiment session_id is set as 123 for later reproducibility.
- #Documents: Number of documents (or samples in dataset if dataframe is passed).
- Vocab Size: Size of vocabulary in the corpus after applying all text pre-processing such as removal of Stopwords, bigram/trigram extraction, lemmatization etc.

Data Wrangling

We will create a new column called sentiment in the dataset and derive values as Positive and negative based on the column star_rating. Anything with star rating 4 and above will be tagged as Positive , else it will be tagged as Negative.

```
df_reviews['sentiment'] = df_reviews['star_rating'].apply(lambda x: 'Positive' if x <= 4 else 'Negative')
```

Figure 9 Calculate sentiment based on star_ratings

Data Visualization

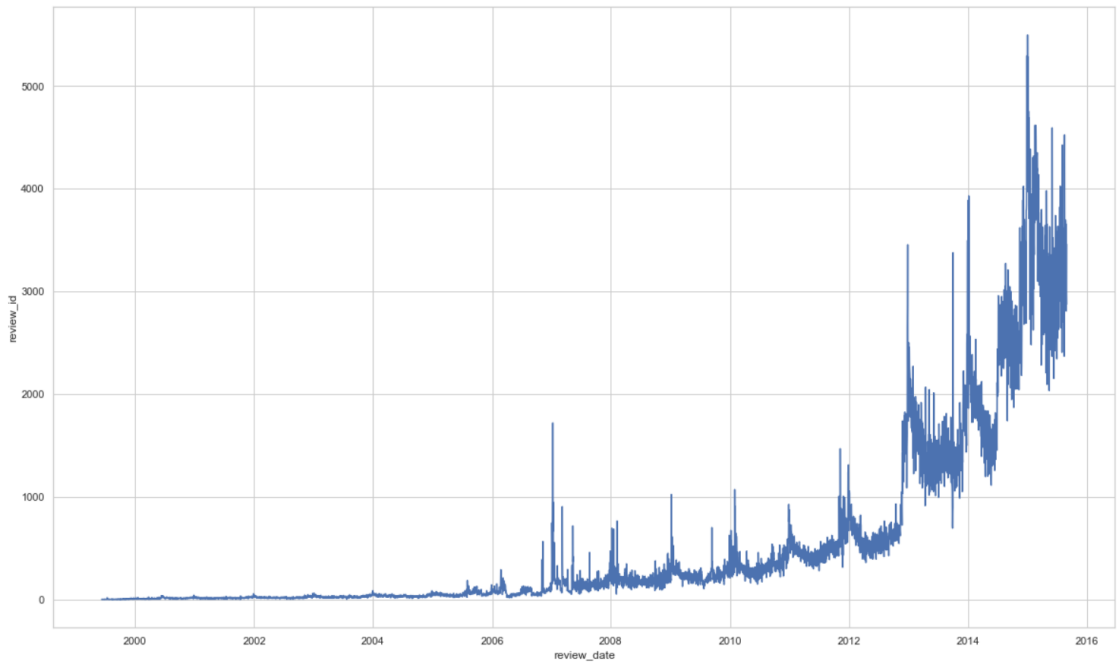

Figure 10 Monthly Trend of Reviews

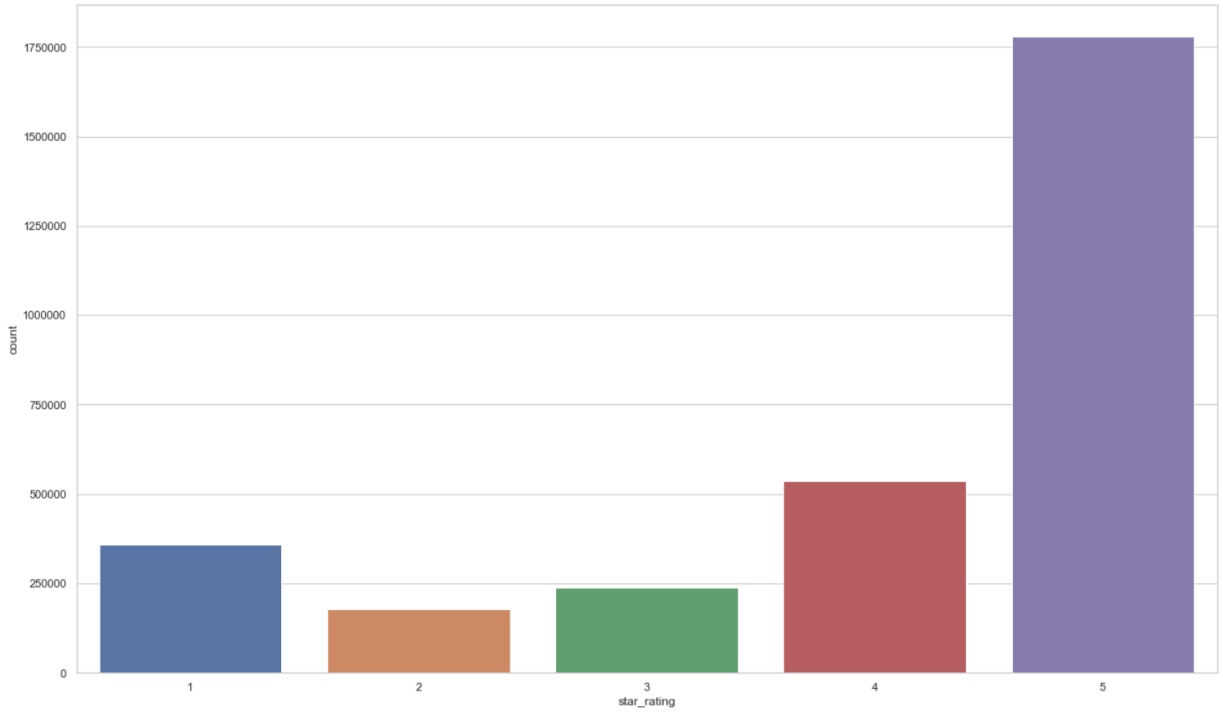

Figure 11 Review Count for Each Star Rating

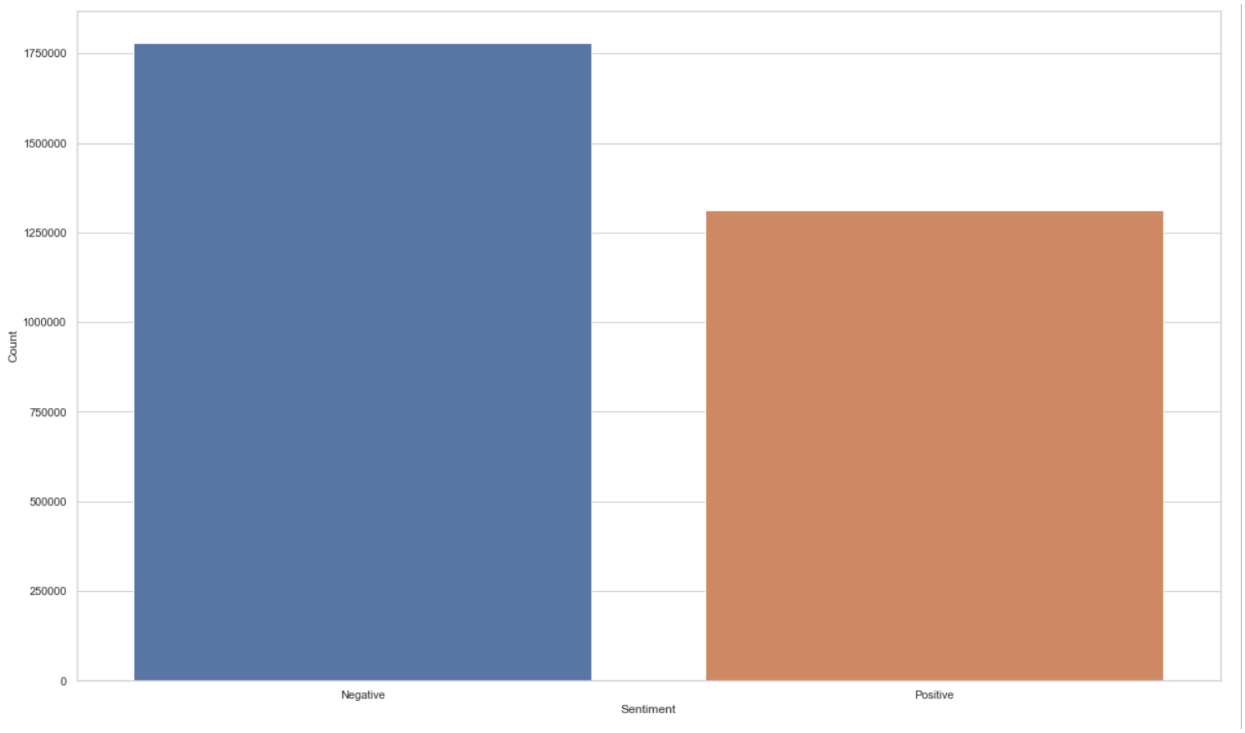

Figure 12 Sentiment distribution of Dataset

Data Modelling

A topic model is created using ***create_model()*** function which takes one mandatory parameter i.e. name of model as a string. This function returns a trained model object. There are 5 topic models available in PyCaret. see the docstring of ***create_model()*** for complete list of models.

Below we will create LDA model with 6 topics and we will also set multi_core parameter to True. When multi_core is set to True Latent Dirichlet Allocation (LDA) uses all CPU cores to parallelize and speed up model training.

```
lda = create_model('lda', num_topics = 6, multi_core = True)
print(lda)
```

```
LdaModel(num_terms=3668, num_topics=6, decay=0.5, chunksize=100)
```

Figure 13 Create LDA model via PyCaret

Now that we have created a topic model, we would like to assign the topic proportions to our dataset (6818 documents / samples) to analyze the results. We will achieve this by using ***assign_model()*** function.

```
lda_results = assign_model(lda)
lda_results.head()
```

	review_id	customer_id	review_headline	review_body	sentiment	Topic_0	Topic_1	Topic_2	Topic_3	Topic_4	Topic_5	Dominant_Topic	F
0	R3FCHTWDX6M0NY	43758182	Five Stars	nice	Negative	0.580920	0.083476	0.083914	0.083337	0.084800	0.083553	Topic 0	
1	R1GNTVXZQMKWHW	30156333	as advertised	build custom box work perfectly cover good qua...	Negative	0.012958	0.012941	0.012898	0.012969	0.012926	0.935308	Topic 5	
2	R1UXFD06PQXGF8	47478367	I need help!	instruction difficult work make sound sit shel...	Positive	0.018748	0.018565	0.406675	0.018726	0.018741	0.518545	Topic 5	
3	RSPOKAESK9MEY	36856462	It's a battery.	battery	Negative	0.084364	0.084229	0.083461	0.083374	0.083530	0.581042	Topic 5	
4	R25JSD0FT6NTJC	23678831	Replaiment	use sure work come clean	Positive	0.028073	0.027974	0.028004	0.027899	0.860028	0.028022	Topic 4	

Figure 14 Assign LDA model via PyCaret

Notice how 6 additional columns are now added to the dataframe. ***review_body*** is the text after all pre-processing. Topic 0 ... Topic 5 are the topic proportions and represents the distribution of topics for each document. Dominant Topic is the topic number with highest proportion and Perc_Dominant_Topic is the percentage of dominant topic over 1 (only shown when models are stochastic i.e. sum of all proportions equal to 1) .

plot_model() function can be used to analyze the overall corpus or only specific topics extracted through topic model. Hence the function ***plot_model()*** can also work without passing any trained model object.

```
1 plot_model()
```

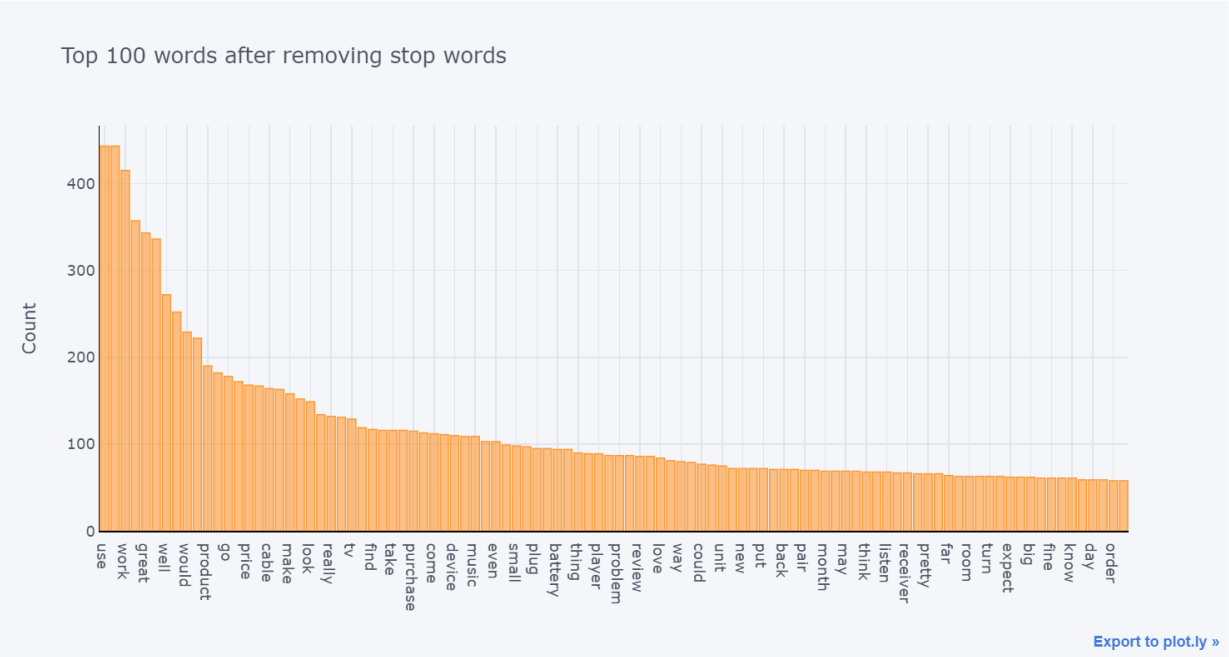

Figure 15 Frequency distribution of entire Corpus

```
1 plot_model(plot = 'bigram')
```

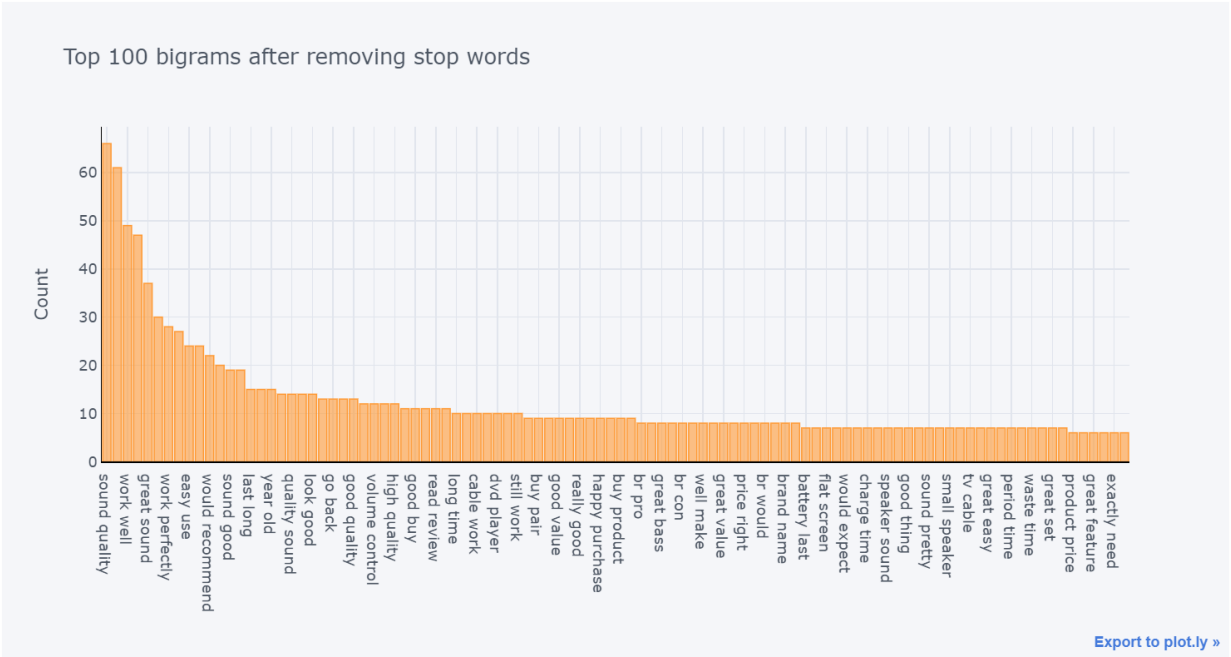

Figure 16 Top 100 Bigrams on Entire Corpus

```
plot_model(lda, plot = 'frequency', topic_num = 'Topic 1')
```

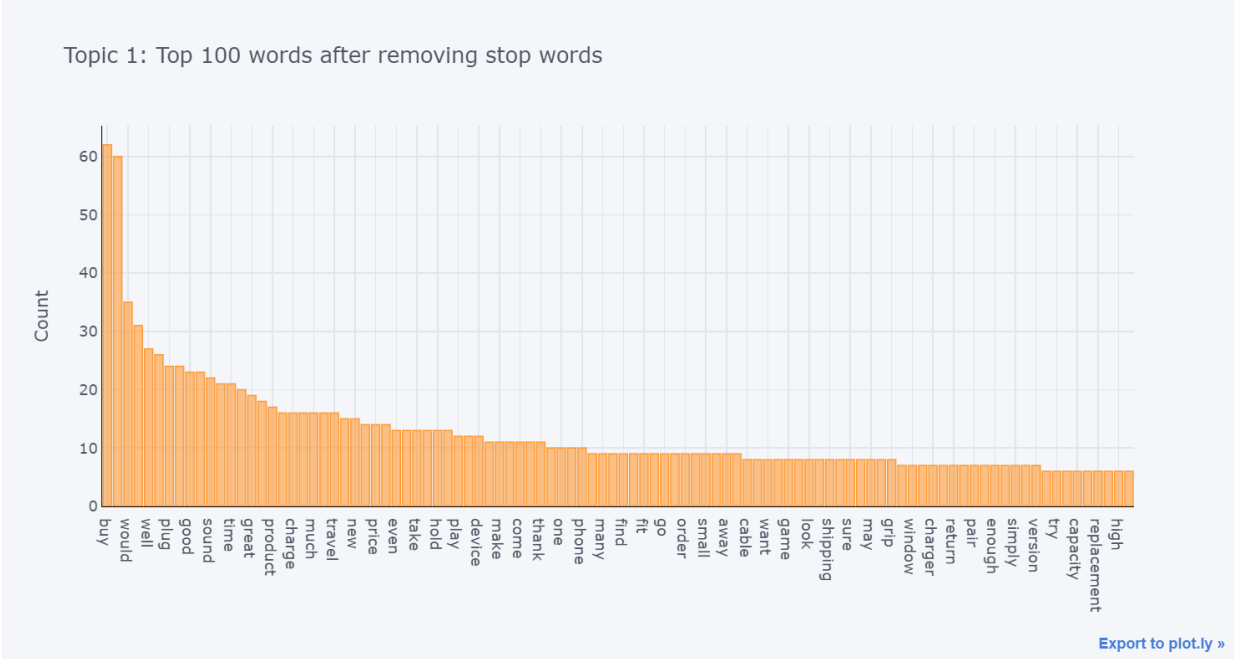

Figure 17 Frequency distribution of Topic 1

```
plot_model(lda, plot = 'topic_distribution')
```

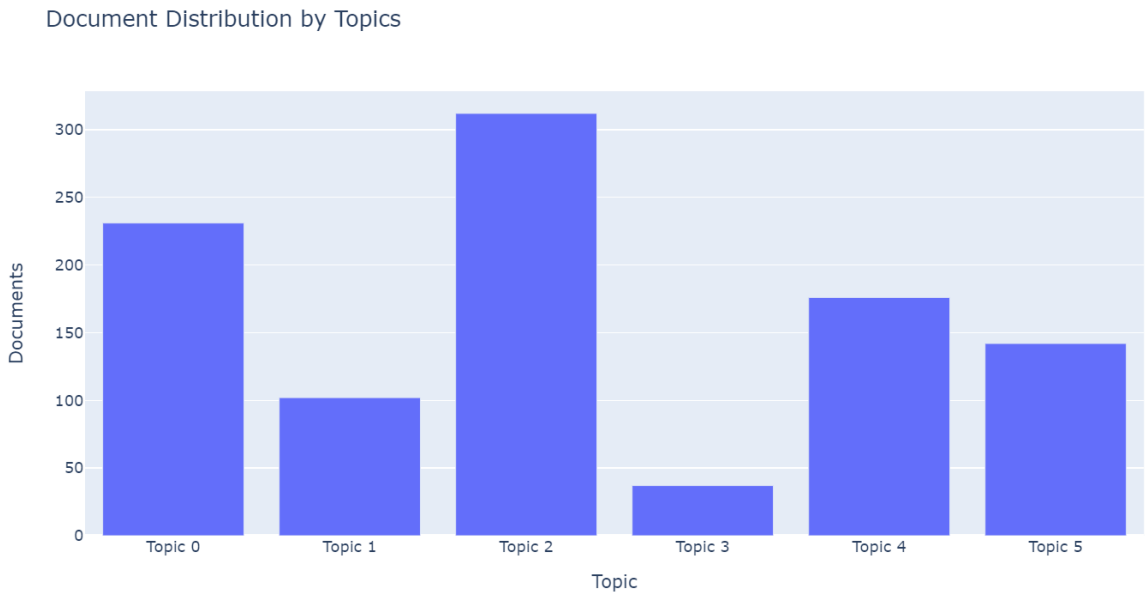

Figure 18 Document Distribution by Topics

Each document is a distribution of topics and not a single topic. Although, if the task is of categorizing document into specific topics, it would not be wrong to use the topic proportion with highest value to categorize the document into a topic. In above plot, each document is categorized into one topic using the largest proportion of topic weights. We can see most of the documents are in Topic 2

T-distributed Stochastic Neighbor Embedding (t-SNE) is a nonlinear dimensionality reduction technique well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions.

```
1 plot_model(lda, plot = 'tsne')
```

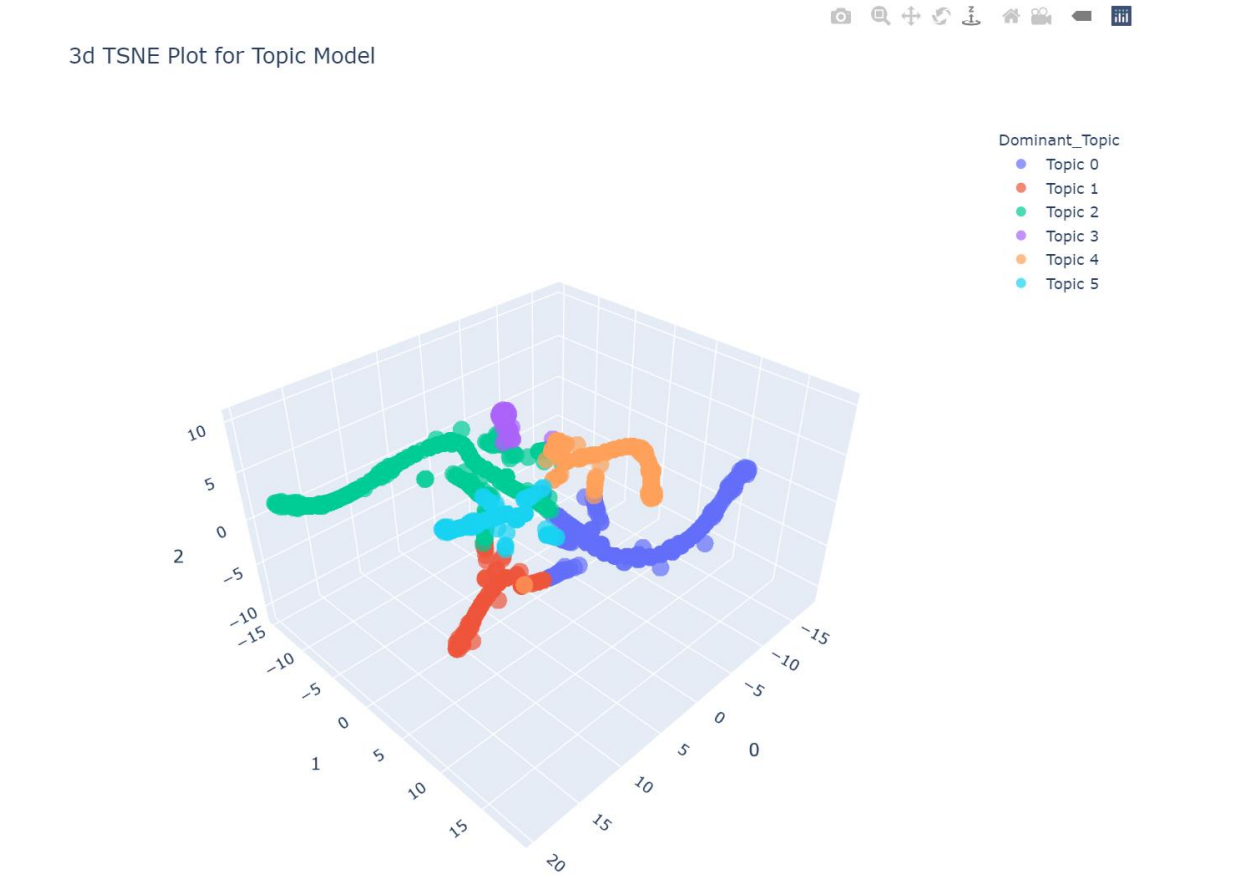

Figure 19 3d t-SNE Plot for Topic Model

UMAP (Uniform Manifold Approximation and Projection) is a novel manifold learning technique for dimensionality reduction. It is similar to tSNE and PCA in its purpose as all of them are techniques to reduce dimensionality for 2d/3d projections. UMAP is constructed from a theoretical framework based in Riemannian geometry and algebraic topology.

```
1 plot_model(lda, plot = 'umap')
```

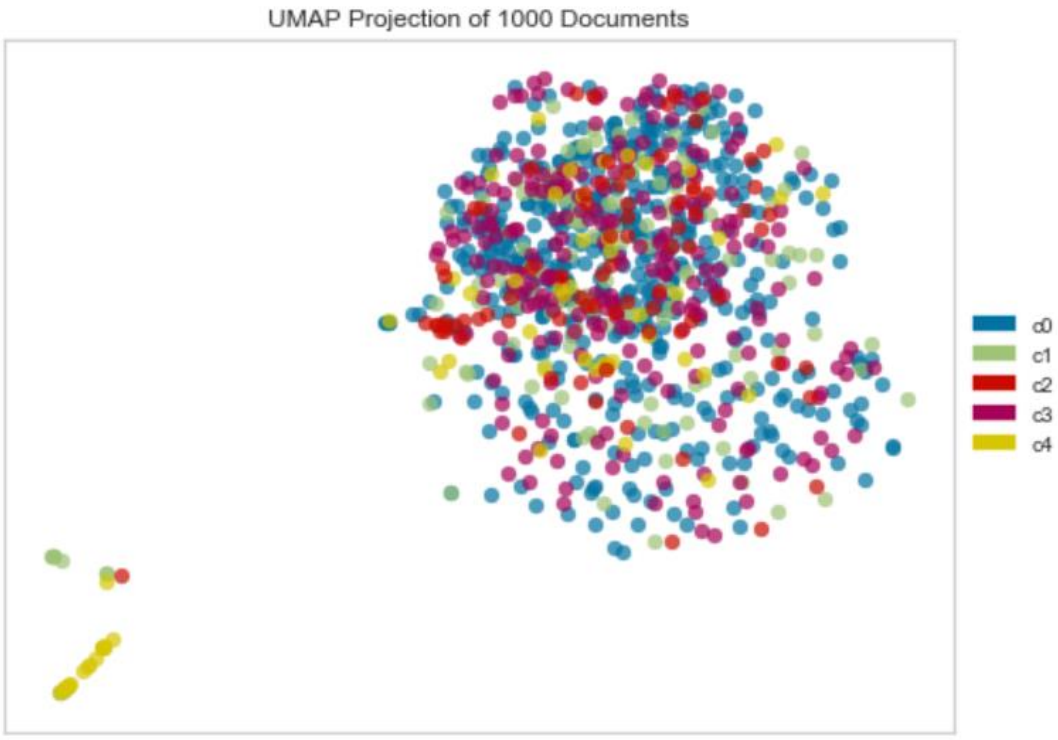

Figure 20 Uniform Manifold Approximation and Projection Plot

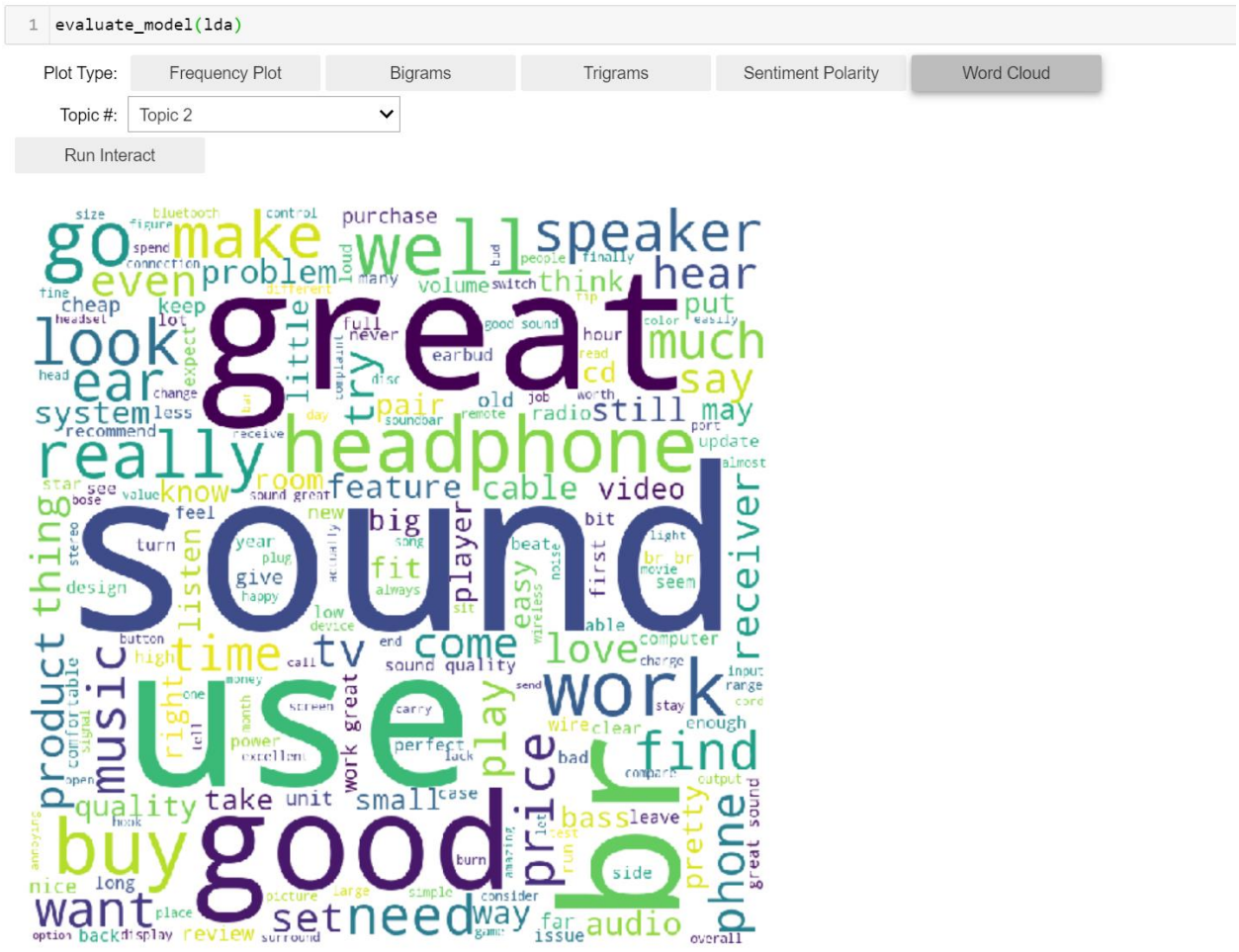

Figure 21 Word Cloud for Topic 2 via PyCaret

You can save the model via ***save_model()*** function and load it later via ***load_model*** function.

In the second part, we can also perform binary classification by using PyCaret.

First load the sample data and also remove star_rating column since the value of the column clearly predicts the sentiment output. We will as split the data into data and data_unseen so that we can train the model with data and run analysis on unseen data.

```
# sampling the data to select only 1000 documents
df_classification = df_reviews.sample(1000, random_state=786).drop(['star_rating'], axis=1).reset_index(drop=True)
df = df_classification.reset_index(drop=True)

# sampling the data to select only 1000 documents
data = df.sample(frac=0.95, random_state=786).reset_index(drop=True)
data_unseen = df_classification.drop(data.index).reset_index(drop=True)

print('Data for Modeling: ' + str(data.shape))
print('Unseen Data For Predictions: ' + str(data_unseen.shape))
```

```
Data for Modeling: (950, 5)
Unseen Data For Predictions: (50, 5)
```

Figure 22 Data Split for Binary Classification

We will run set the environment and use target as sentiment column. We will then run compare_models() to see which model performs best and provides highest accuracy score.

Based on the output of compare_model , we can then see Gradient boosting classifier performs better with highest accuracy score.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
0	Gradient Boosting Classifier	0.6128	0.6256	0.1568	0.6653	0.2442	0.1161	0.1763	1.5325
1	Random Forest Classifier	0.5979	0.5720	0.0499	0.7000	0.0915	0.0568	0.1394	0.1491
2	Ridge Classifier	0.5844	0.0000	0.1500	0.4117	0.1954	0.0576	0.0799	0.1891
3	Ada Boost Classifier	0.5843	0.6270	0.3454	0.5227	0.4100	0.1107	0.1193	0.9097
4	Logistic Regression	0.5768	0.4768	0.0000	0.0000	0.0000	0.0000	0.0000	0.0716
5	Naive Bayes	0.5768	0.5235	0.0000	0.0000	0.0000	0.0000	0.0000	0.0631
6	Decision Tree Classifier	0.5768	0.5571	0.4275	0.4997	0.4588	0.1162	0.1173	0.0910
7	K Neighbors Classifier	0.5709	0.5615	0.4341	0.4942	0.4602	0.1072	0.1084	0.2516
8	SVM - Linear Kernel	0.4846	0.0000	0.6000	0.2539	0.3568	0.0000	0.0000	0.2178
9	Quadratic Discriminant Analysis	0.4232	0.5000	1.0000	0.4232	0.5947	0.0000	0.0000	0.2090

We can then create model using Gradient Boosting Classifier and tune it via tune_model function. We can then plot ROC curve, Precision-Recall curve, feature importance model and lastly confusion matrix.

```
plot_model(tuned_gbc, plot = 'confusion_matrix')
```

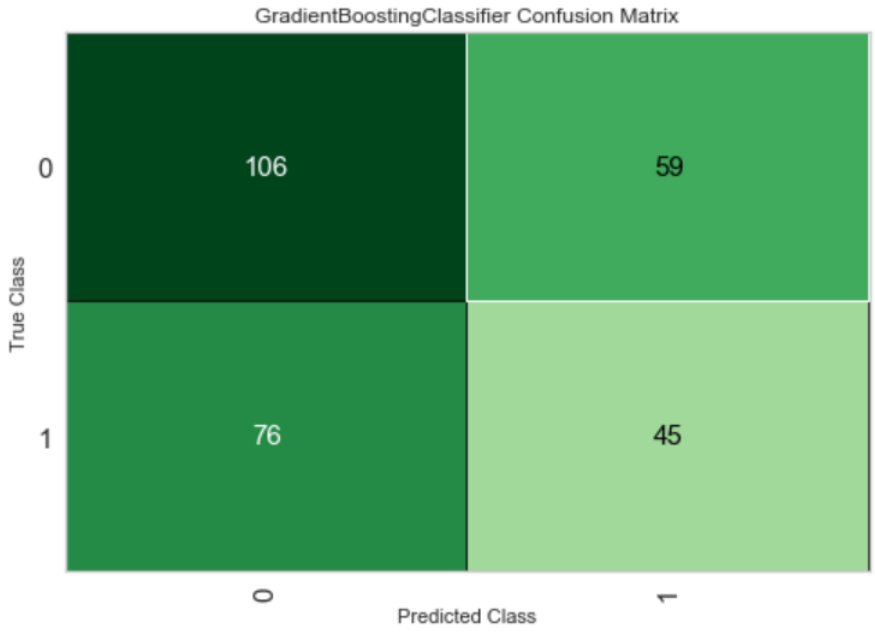

Figure 23 Confusion Matrix for the Model

As you can see from the output of Confusion matrix, model accuracy was around 69%.

We can then create model object via finalize model and then run predict_model on the unseen data.

```
final_gbc = finalize_model(tuned_gbc)
predict_model(final_gbc);
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Gradient Boosting Classifier	0.6993	0.792	0.5455	0.6804	0.6055	0.3673	0.3732

```
unseen_predictions = predict_model(final_gbc, data=data_unseen)
unseen_predictions.head()
```

	review_id	customer_id	review_headline	review_body	sentiment	Label	Score
0	R5QP9BEQHCHXU	14237851	cables	Cables work well and I needed some longer HDMI...	Negative	Positive	0.5697
1	R1ZG28PSYYTZ76	32189944	Good sound, Easy to use	I haven't had it but for a few weeks. So far i...	Negative	Positive	0.7925
2	RNDE78F9PG152	28826538	Fast shipping unlike others -- Beware	My new laptop did not support VGA cables, so I...	Negative	Negative	0.2867
3	R16PQ52CH1LPJ2	4361260	great buy!	cheap and great quality. I swear these are bet...	Negative	Negative	0.1713
4	RFJGBT2CL0BE9	52637759	Five Stars	Item delivered on time, was as described	Negative	Negative	0.1197

Figure 24 Predict model PyCaret

Conclusion and Final Thoughts

We started sentiment analysis of Amazon review dataset for Electronics category. After downloading the dataset, we performed some cleaning data cleaning steps to prepare the data for analysis. We also derived sentiment based on star rating column in the dataset. After visualizing the sentiment

distribution, we used PyCaret library to setup the environment with sample of the dataset. We then ran LDA model to run NLP sentiment analysis and visualized the data in multiple ways. In later part of the section, we again used PyCaret library to run binary classification model by dropping star_rating column. We then compared various models and chose the model Gradient Boost Classifier (GBC) model with the highest accuracy score. We then created, tuned the GBC model, and visualized it. Lastly, we saved the model object and showed how we can run prediction on unseen data. From the confusion matrix, we same model provided 69% accuracy score. We also tried TF-IDF method using sci-kit learn Logistic regression method to run binary classification algorithm and plotted confusion matrix at the end.

Project Code References

- Project GitHub Repository : Project GitHub Repository : https://github.com/kirti-chaudhari/SpringBoard_DataScience_Career

- Jupyter Notebook : https://github.com/kirti-chaudhari/SpringBoard_DataScience_Career/blob/master/CapstoneProject2/Capstone%202%20-%20NLP.ipynb