

Analyzing Amazon's Alexa Reviews Using Natural Language Processing



amazon alexa

MIS 753 – Independent Study

Murilo Gustineli

Dr. Gregory Moody

Spring 2020

Contents

| | |
|---|----|
| 1. Introduction | 3 |
| 2. Structured vs Unstructured Data..... | 3 |
| 2.1 Structured Data..... | 4 |
| 2.2 Unstructured Data | 5 |
| 3. Methodology | 6 |
| 4. Instructions | 6 |
| 4.1 Installing..... | 6 |
| 4.2 Software..... | 6 |
| 4.3 Libraries | 6 |
| 4.4 Running the program | 7 |
| 4.5 Explore the program..... | 8 |
| 5. Understanding the dataset..... | 8 |
| 5.1 Columns..... | 9 |
| 5.2 Rows | 9 |
| 6. Results..... | 10 |
| 6.1 Data Visualizations | 13 |
| 6.2 3D Scatter Plot..... | 24 |
| 6.3 spaCy..... | 25 |
| 6.4 Natural Language Processing..... | 26 |
| 7. Conclusion..... | 31 |
| 7.1 Moving Forward | 31 |
| 8. Self-reflection..... | 32 |
| 9. Resources | 32 |

1. Introduction

Natural Language Processing (NLP) is a mixture of linguistics, computer science, information engineering, and artificial intelligence concerned with the interaction between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data [\[1\]](#). There is essentially an infinite number of ways to arrange words in a sentence. It is practically impossible to give computers a dictionary of all possible sentences to help them understand what humans are talking about. Companies know that by better analyzing their data, they can improve their operations, thereby saving money and keeping their employees satisfied [\[2\]](#). However, not all data is created equally. It is known that only 20% of the data needed to do NLP analysis is in a structured format, such as spreadsheets and databases, which is data that computers can easily use. The other 80% of data is classified as unstructured data, and it comes in many different forms, such as text, video, audio, etc. Unstructured data not only constitutes the biggest majority of data, but it is also extremely valuable. Though, due to its size and structure, it has greatly been inaccessible to data analytics [\[3\]](#).

Thanks to this field of AI called Natural Language Processing, computers can now analyze and understand textual data. NLP algorithms are not capable of reading text like humans can. However, computers can look for patterns, and find these patterns by turning huge amounts of text into matrices. When analyzing text, the algorithm might first remove words that don't really offer as much value. Those words are called "stop words" [\[4\]](#). Then, the algorithm might split the sentences into groups of words and count how many times each group of words appears in each document and how many documents have that group of words out of all the documents being analyzed. Without knowing anything about the text, the algorithm can tell how often a given word or phrase appears in a given document, and how many documents contain that phrase out of all of the documents. Thus, tokens that appear more often in the majority of documents may not have much significance. On the other hand, tokens that appear frequently in only a few documents have a much higher importance for the overall results [\[5\]](#).

2. Structured vs Unstructured Data

As previously mentioned, not all data is created equal. For instance, the data generated from social media apps are completely different from the data generated by point-of-sales or supply chain systems. Some data is structured, but most is unstructured. The way this data is collected, processed, and analyzed all depends on its format [\[3\]](#).

The main difference between structure and unstructured data is that structured data is highly organized and formatted in a way that is easily searchable in relational databases. Unstructured data has no pre-defined format or organization, making it much more difficult to collect, process, and analyze [\[3\]](#).

2.1 Structured Data

Most often categorized as **quantitative** data, structured data is the kind of data would be typically found in relational databases and spreadsheets. Examples of structured data are names, dates, addresses, credit card numbers, stock information, geolocation, etc. Structured data is also highly organized and easily understood by machine language. Those working with relational databases can input, search, and manipulate structured data relatively quickly since everything is nicely organized in terms of tables and rows. This is the most attractive feature of structured data [3]. Every table is assigned with a particular topic or entity and have a defined relationship between those entities. SQL (structured query language) is the language used for managing structured data. It is worth mentioning that structured data revolutionized paper-based systems that companies relied on for business intelligence decades ago. While structured data is still useful, more companies are looking to deconstruct unstructured data for future opportunities.

The picture below is an example structured data in a relational database.

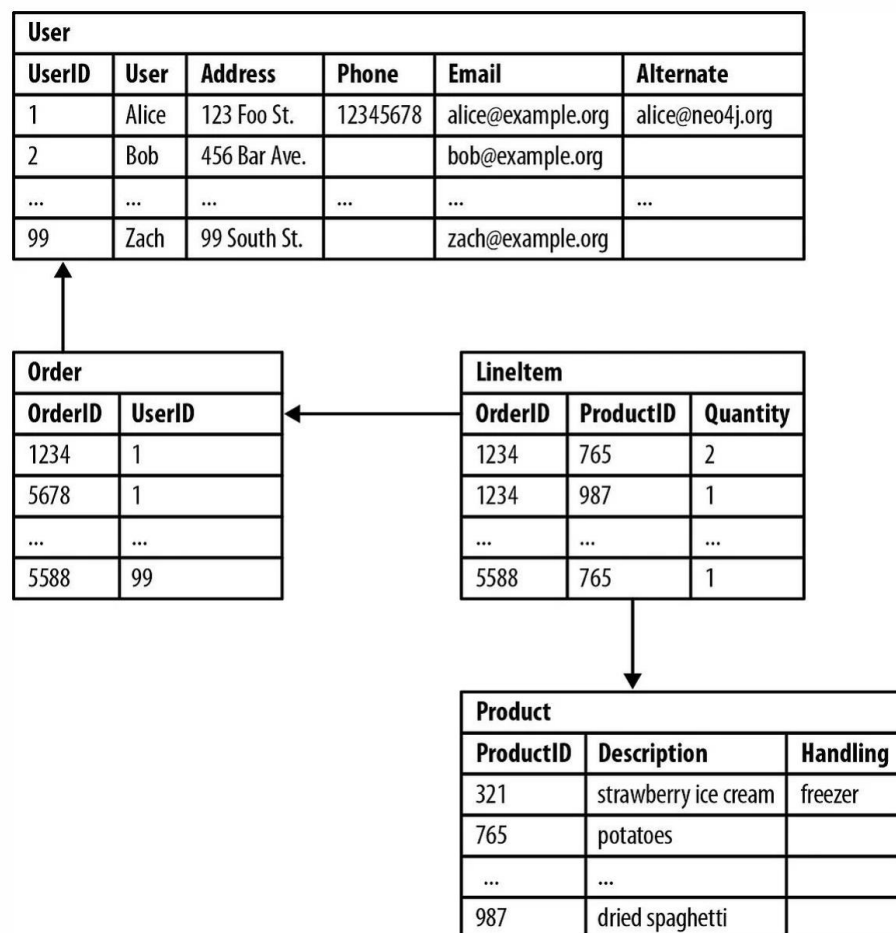


Figure 1: <https://learn.q2.com/structured-vs-unstructured-data>

2.2 Unstructured Data

Most often categorized as **qualitative** data. Unstructured data cannot be processed and analyzed using conventional tools and methods. Examples of unstructured data are text, video, audio, mobile activity, social media activity, satellite imagery, surveillance imagery, etc. Unlike structured data, unstructured data is difficult to deconstruct because it has no pre-defined model, thus it cannot be organized in relational databases. Non-relational, or NoSQL (Not Only SQL) databases are the best fit for managing unstructured data [6]. As mentioned previously, more than 80% of all generated data is considered unstructured, and this number will continue to rise with the prominence of the internet of things [3]. Because of its flexibility, unstructured data requires advanced analytics and a high level of technical expertise to really make a difference. Companies that are able to harness unstructured data are at a competitive advantage.

Structured vs Unstructured data

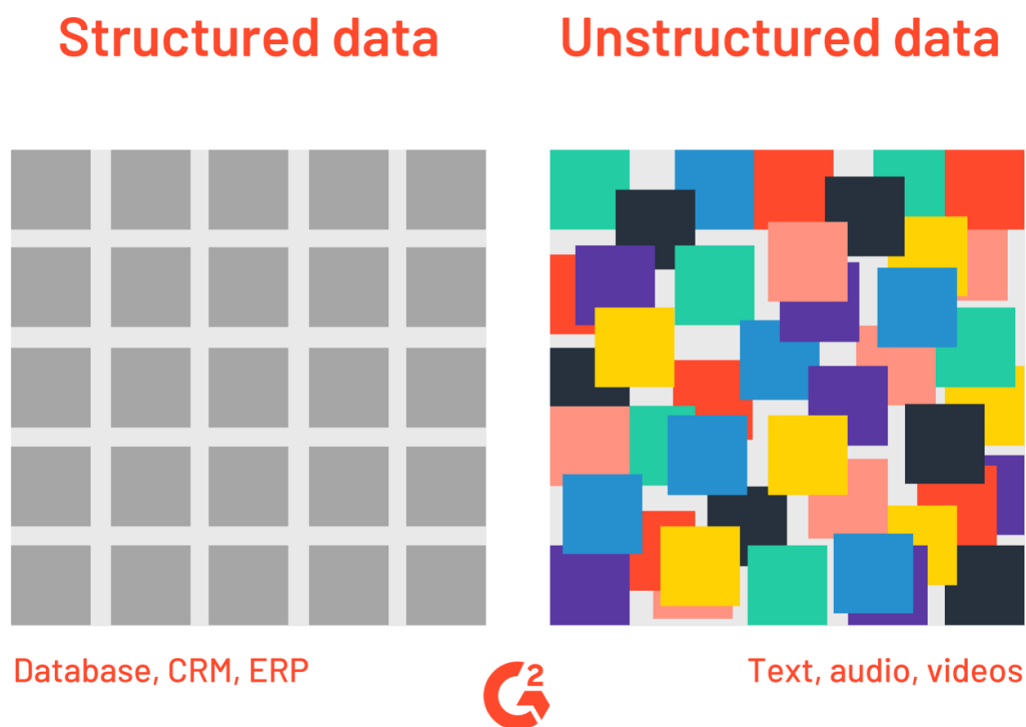


Figure 2: <https://learn.q2.com/structured-vs-unstructured-data>

3. Methodology

For this project, I will be using a dataset containing Amazon's Alexa reviews. This dataset consists of a nearly 3000 Amazon customer reviews (input text), star ratings, date of review, variant and feedback of various amazon Alexa products like Alexa Echo, Echo dots, Alexa Firesticks etc. for learning how to train machine for sentiment analysis. I will also use this dataset to analyze Amazon's Alexa product; discover insights into consumer reviews and assist with machine learning models. Moreover, I will train the machine learning models in order to obtain sentiment analysis and analyze customer reviews regarding the relationship between positive and negative reviews. The reviews were extracted directly from Amazon's website.

In order to analyze and deliver results from the dataset, I will be using the programming language Python. Python is open source, interpreted, high level language and provides great approach for object-oriented programming. It is one of the best languages used by data scientist for various data science projects/application. The reason why I am choosing Python over other programming languages is because Python provides great functionality to deal with mathematics, statistics and scientific functions. It also provides great libraries to work with data science applications [7].

4. Instructions

Follow the next instructions if you would like to download and run the program at your own computer.

4.1 Installing

- The code and the dataset for this project can be found at my GitHub page github.com/murilogustinel1
- The dataset can also be found at kaggle.com/sid321axn/amazon-alexa-reviews
- If you wish to run the program on your computer, make sure to go to my GitHub page and follow the next steps.

4.2 Software

For this program, you will need Jupyter Notebook installed on your computer. Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. I recommend installing the Anaconda distribution system; it comes with Jupyter Notebook pre-installed.

- You can install Anaconda following this link <https://www.anaconda.com/distribution/>

4.3 Libraries

The Anaconda distribution system comes with many libraries already pre-installed. However, because I will be doing analysis of linguistic patterns, I will be using other

libraries that are not included in the Anaconda distribution system. Thus, these libraries need to be installed individually.

Here's a list of all the libraries that will be used in the program and how to install them. Note that many libraries come already pre-installed with Anaconda.

- NumPy – numpy.org/
 - Comes pre-installed with Anaconda
- Pandas – pandas.pydata.org/
 - Comes pre-installed with Anaconda
- Matplotlib – matplotlib.org/
 - Comes pre-installed with Anaconda
- Seaborn – seaborn.pydata.org/
 - Comes pre-installed with Anaconda
- Plotly – plotly.com/install/
 - Follow the link and installation guide
- Scikit-learn – scikit-learn.org/stable/install.html
 - Follow the link and installation guide
- Wordcloud – pypi.org/project/wordcloud/
 - Follow the link and installation guide
- NLTK – nltk.org/install.html
 - Follow the link and installation guide
- spaCy – spacy.io/usage
 - Follow the link and installation guide

Note: make sure you have all of these libraries installed on your computer prior to running the code. If you try running the program without one of the libraries installed, it might give an error.

4.4 Running the program

Once you installed the Anaconda distributed system and also all of the libraries listed above, open Jupyter Notebook using Anaconda.

- Make sure the database file (amazon_alex.tsv) and the Jupyter Notebook file (amazon_alex_NLP.ipynb) are in the same directory. Otherwise, the program will crash due to path issues.

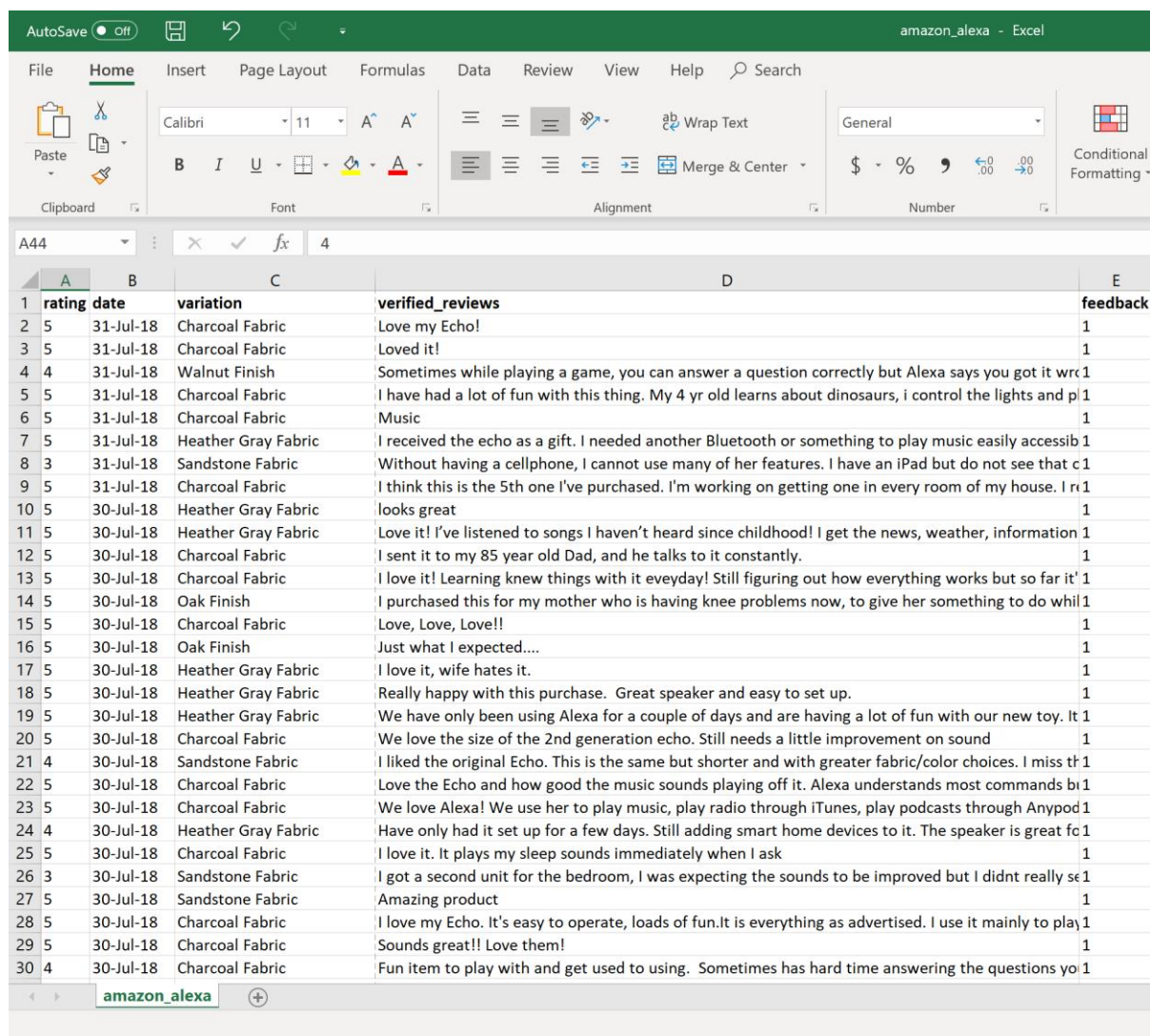
4.5 Explore the program

After completing all of the steps above, you should be able to run the program on your own computer. If you have any concerns regarding installation or any questions in general, feel free to send me a comment under “Issues” on the project page.

5. Understanding the dataset

The dataset can be found as a TSV file. A tab-separated values (TSV) file is a simple text format for storing data in a tabular structure, e.g., database table or spreadsheet data, and a way of exchanging information between databases. Each record in the table is one line of the text file. Each field value of a record is separated from the next by a tab character [8].

In order to better understand the dataset, I converted the TSV file into an Excel spreadsheet to easier understand the relationship between rows and columns.



| amazon_alexa - Excel | | | | |
|--|-----------|---------------------|--|----------|
| File Home Insert Page Layout Formulas Data Review View Help Search | | | | |
| Clipboard Font Alignment Number Conditional Formatting | | | | |
| A44 | | | | |
| rating | date | variation | verified_reviews | feedback |
| 5 | 31-Jul-18 | Charcoal Fabric | Love my Echo! | 1 |
| 5 | 31-Jul-18 | Charcoal Fabric | Loved it! | 1 |
| 4 | 31-Jul-18 | Walnut Finish | Sometimes while playing a game, you can answer a question correctly but Alexa says you got it wr | 1 |
| 5 | 31-Jul-18 | Charcoal Fabric | I have had a lot of fun with this thing. My 4 yr old learns about dinosaurs, i control the lights and p | 1 |
| 5 | 31-Jul-18 | Charcoal Fabric | Music | 1 |
| 5 | 31-Jul-18 | Heather Gray Fabric | I received the echo as a gift. I needed another Bluetooth or something to play music easily accessib | 1 |
| 3 | 31-Jul-18 | Sandstone Fabric | Without having a cellphone, I cannot use many of her features. I have an iPad but do not see that c | 1 |
| 5 | 31-Jul-18 | Charcoal Fabric | I think this is the 5th one I've purchased. I'm working on getting one in every room of my house. I r | 1 |
| 5 | 30-Jul-18 | Heather Gray Fabric | looks great | 1 |
| 5 | 30-Jul-18 | Heather Gray Fabric | Love it! I've listened to songs I haven't heard since childhood! I get the news, weather, informati | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | I sent it to my 85 year old Dad, and he talks to it constantly. | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | I love it! Learning knew things with it eveyday! Still figuring out how everything works but so far it' | 1 |
| 5 | 30-Jul-18 | Oak Finish | I purchased this for my mother who is having knee problems now, to give her something to do whil | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | Love, Love, Love!! | 1 |
| 5 | 30-Jul-18 | Oak Finish | Just what I expected.... | 1 |
| 5 | 30-Jul-18 | Heather Gray Fabric | I love it, wife hates it. | 1 |
| 5 | 30-Jul-18 | Heather Gray Fabric | Really happy with this purchase. Great speaker and easy to set up. | 1 |
| 5 | 30-Jul-18 | Heather Gray Fabric | We have only been using Alexa for a couple of days and are having a lot of fun with our new toy. It | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | We love the size of the 2nd generation echo. Still needs a little improvement on sound | 1 |
| 4 | 30-Jul-18 | Sandstone Fabric | I liked the original Echo. This is the same but shorter and with greater fabric/color choices. I miss th | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | Love the Echo and how good the music sounds playing off it. Alexa understands most commands b | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | We love Alexa! We use her to play music, play radio through iTunes, play podcasts through Anypod | 1 |
| 4 | 30-Jul-18 | Heather Gray Fabric | Have only had it set up for a few days. Still adding smart home devices to it. The speaker is great f | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | I love it. It plays my sleep sounds immediately when I ask | 1 |
| 3 | 30-Jul-18 | Sandstone Fabric | I got a second unit for the bedroom, I was expecting the sounds to be improved but I didnt really se | 1 |
| 5 | 30-Jul-18 | Sandstone Fabric | Amazing product | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | I love my Echo. It's easy to operate, loads of fun.It is everything as advertised. I use it mainly to play | 1 |
| 5 | 30-Jul-18 | Charcoal Fabric | Sounds great!! Love them! | 1 |
| 4 | 30-Jul-18 | Charcoal Fabric | Fun item to play with and get used to using. Sometimes has hard time answering the questions yo | 1 |

The Excel spreadsheets represent a good idea of columns and rows that will be analyzed in the NLP model, and it's easily understood by the user. It's worth pointing out that I will be using the TSV file as the main source of data for the project. The Excel spreadsheet created is just an illustration for the text file.

We can notice that the file contains five columns (rating, date, variation, verified reviews, and feedback) and over three thousand rows (3,150 rows to be exact).

5.1 Columns

- **Rating:** how many stars the customer gave its review. The smallest being 1 and the highest being 5 stars.
- **Date:** the day in which the review was posted.
- **Variation:** The different model variations of the Amazon's Alexa product. We will dive deeper and analyze the differences between each model in the program.
- **Verified Reviews:** Those are the typed reviews made by the Alexa consumers. They are the body of text that will be analyzed in the NLP model.
- **Feedback:** feedback can be either 0 or 1. Negative feedback equals 0, and positive feedback equals 1. If the customers gave a rating of 2 stars or below, the feedback equals 0. If the customer gave a rating of 3 stars or above, the feedback equals 1.

5.2 Rows

- Each row represents one review made by one Alexa's customer. Each review has a rating, date, product type (variation), and the comments (verified reviews) made by that customer. There is a total of 3150 reviews.

6. Results

1. Importing basic libraries to load the dataset and run the analysis.

```
In [1]: # for basic operations
import numpy as np
import pandas as pd

# for basic visualizations
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('fivethirtyeight')

# for advanced visualizations
import plotly.offline as py
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
init_notebook_mode(connected = True)
import plotly.figure_factory as ff

# for providing the path
import os
print(os.listdir('C:/Users/muril/Documents/UNLV - MIS/MIS 753 - Independent Study/Amazon Alexa/'))

['.ipynb_checkpoints', 'amazon_alexas.tsv', 'amazon_alexas_reviews.ipynb']
```

I will be using libraries such as NumPy and Pandas for calculation basic numerical and statistical operations. The libraries Matplotlib and Seaborn to show visualizations. And the library Plotly to show advanced visualizations.

2. Reading the data

```
In [2]: data = pd.read_csv('amazon_alexas.tsv', delimiter = '\t', quoting = 3)

# getting the shape of the data
data.shape
```

Out[2]: (3150, 5)

As previously mentioned, we can see that the database has 3150 rows and 5 columns.

3. This code shows the first 5 rows of the database

```
In [3]: data.head()
```

Out[3]:

| | rating | date | variation | verified_reviews | feedback |
|---|--------|-----------|-----------------|---|----------|
| 0 | 5 | 31-Jul-18 | Charcoal Fabric | Love my Echo! | 1 |
| 1 | 5 | 31-Jul-18 | Charcoal Fabric | Loved it! | 1 |
| 2 | 4 | 31-Jul-18 | Walnut Finish | "Sometimes while playing a game, you can answe... | 1 |
| 3 | 5 | 31-Jul-18 | Charcoal Fabric | "I have had a lot of fun with this thing. My 4... | 1 |
| 4 | 5 | 31-Jul-18 | Charcoal Fabric | Music | 1 |

4. Describing the dataset

```
In [4]: # describing the data set
data.describe()
```

Out[4]:

| | rating | feedback |
|-------|-------------|-------------|
| count | 3150.000000 | 3150.000000 |
| mean | 4.463175 | 0.918413 |
| std | 1.068506 | 0.273778 |
| min | 1.000000 | 0.000000 |
| 25% | 4.000000 | 1.000000 |
| 50% | 5.000000 | 1.000000 |
| 75% | 5.000000 | 1.000000 |
| max | 5.000000 | 1.000000 |

This code generates descriptive statistics of the dataset. Descriptive statistics include those that summarize the central tendency, dispersion and shape of a dataset's distribution, excluding NaN values. Analyzes both numeric and object series, as well as Data Frame column sets of mixed data types. The output will vary depending on what is provided [\[9\]](#).

5. Checking if there are any null values in the dataset

```
In [5]: # checking if there are any null values in the dataset
data.isnull().any().any()
```

Out[5]: False

Sometimes data files have null values and are displayed as NaN in the Data Frame. In this case, the Data Frame has no null values. If there were null values, we could use the Pandas library to replace null values in the Data Frame.

6. Describing the data according to the length of the reviews

```
In [6]: # adding a length column for analyzing the length of the reviews
```

```
data['length'] = data['verified_reviews'].apply(len)
data.groupby('length').describe().sample(10)
```

Out[6]:

| | rating | | | | | | | | feedback | | | | | | | |
|--------|--------|----------|----------|-----|-----|-----|-----|-----|----------|------|-----|-----|-----|-----|-----|-----|
| | count | mean | std | min | 25% | 50% | 75% | max | count | mean | std | min | 25% | 50% | 75% | max |
| length | | | | | | | | | | | | | | | | |
| 467 | 1.0 | 5.000000 | NaN | 5.0 | 5.0 | 5.0 | 5.0 | 5.0 | 1.0 | 1.0 | NaN | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 461 | 1.0 | 3.000000 | NaN | 3.0 | 3.0 | 3.0 | 3.0 | 3.0 | 1.0 | 1.0 | NaN | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 43 | 22.0 | 4.772727 | 0.528413 | 3.0 | 5.0 | 5.0 | 5.0 | 5.0 | 22.0 | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 684 | 1.0 | 5.000000 | NaN | 5.0 | 5.0 | 5.0 | 5.0 | 5.0 | 1.0 | 1.0 | NaN | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 632 | 1.0 | 4.000000 | NaN | 4.0 | 4.0 | 4.0 | 4.0 | 4.0 | 1.0 | 1.0 | NaN | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 28 | 9.0 | 4.888889 | 0.333333 | 4.0 | 5.0 | 5.0 | 5.0 | 5.0 | 9.0 | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 278 | 1.0 | 5.000000 | NaN | 5.0 | 5.0 | 5.0 | 5.0 | 5.0 | 1.0 | 1.0 | NaN | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 270 | 4.0 | 5.000000 | 0.000000 | 5.0 | 5.0 | 5.0 | 5.0 | 5.0 | 4.0 | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 360 | 1.0 | 5.000000 | NaN | 5.0 | 5.0 | 5.0 | 5.0 | 5.0 | 1.0 | 1.0 | NaN | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 237 | 2.0 | 5.000000 | 0.000000 | 5.0 | 5.0 | 5.0 | 5.0 | 5.0 | 2.0 | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |

Here we are analyzing the reviews based on their length. We got the length of every single one of the reviews from the column “verified_reviews” in our dataset. We are displaying a 10-row sample as the result.

7. Describing the data according to the ratings

```
In [7]: data.groupby('rating').describe()
```

Out[7]:

| | feedback | | | | | | | | length | | | | | | | |
|--------|----------|------|-----|-----|-----|-----|-----|-----|--------|------------|------------|-----|-------|-------|--------|--------|
| | count | mean | std | min | 25% | 50% | 75% | max | count | mean | std | min | 25% | 50% | 75% | max |
| rating | | | | | | | | | | | | | | | | |
| 1 | 161.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 161.0 | 195.658385 | 212.928219 | 1.0 | 36.00 | 120.0 | 284.00 | 1126.0 |
| 2 | 96.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 96.0 | 250.020833 | 270.179472 | 1.0 | 78.75 | 165.0 | 311.25 | 1688.0 |
| 3 | 152.0 | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 152.0 | 208.098684 | 272.582517 | 1.0 | 54.00 | 131.0 | 286.00 | 1956.0 |
| 4 | 455.0 | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 455.0 | 179.338462 | 216.415268 | 1.0 | 34.50 | 100.0 | 242.00 | 1362.0 |
| 5 | 2286.0 | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 2286.0 | 109.006562 | 152.505019 | 1.0 | 27.00 | 65.0 | 136.00 | 2853.0 |

We can notice the difference number of reviews between good and bad ratings. Out of the total 3150 reviews, we can see that 2286 customers gave a 5-star rating. Followed by 455 customers giving 4 stars, 152 customers giving 3 stars, 96 customers giving 2 stars, and 161 customers giving only 1 star.

By looking at this table, we can already conclude that the vast majority of customers gave at least 4 stars to Alexa. Thus, they must be very pleased with the product.

8. Describing the data according to the feedback

```
In [8]: data.groupby('feedback').describe()
```

```
Out[8]:
```

| | feedback | rating | | | | | | | | length | | | | | | | |
|---|----------|--------|----------|----------|-----|-----|-----|-----|-----|--------|------------|------------|-----|------|-------|-------|--------|
| | | count | mean | std | min | 25% | 50% | 75% | max | count | mean | std | min | 25% | 50% | 75% | max |
| 0 | 0 | 257.0 | 1.373541 | 0.484688 | 1.0 | 1.0 | 1.0 | 2.0 | 2.0 | 257.0 | 215.964981 | 236.895519 | 1.0 | 53.0 | 137.0 | 291.0 | 1688.0 |
| 1 | 1 | 2893.0 | 4.737643 | 0.546544 | 3.0 | 5.0 | 5.0 | 5.0 | 5.0 | 2893.0 | 125.274456 | 175.036515 | 1.0 | 29.0 | 70.0 | 155.0 | 2853.0 |

We can see that 2893 customers gave a rating of 3 stars or above, and only 257 customers gave a rating of 2 stars or below. Again, the customer feedback is showing very positive reviews for Alexa.

6.1 Data Visualizations

9. Distribution of Ratings for Alexa

```
In [9]: ratings = data['rating'].value_counts()

label_rating = ratings.index
size_rating = ratings.values

colors = ['pink', 'lightblue', 'aqua', 'gold', 'crimson']

rating_piechart = go.Pie(labels = label_rating,
                          values = size_rating,
                          marker = dict(colors = colors),
                          name = 'Alexa', hole = 0.3)

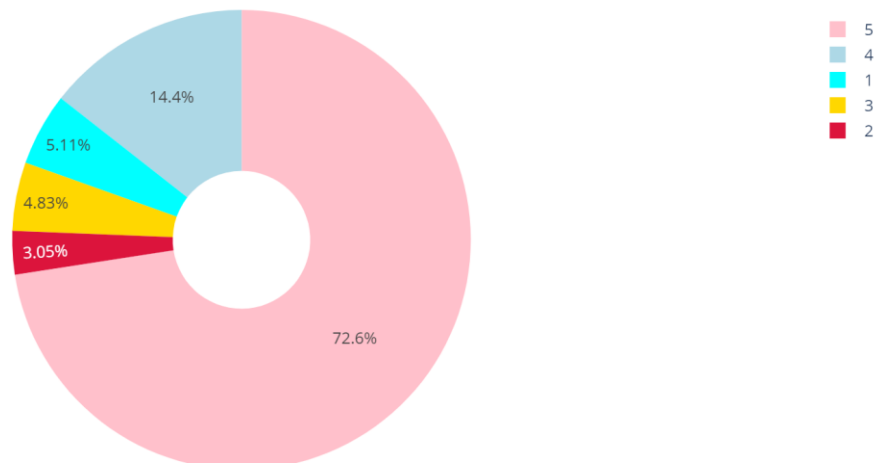
df = [rating_piechart]

layout = go.Layout(
    title = 'Distribution of Ratings for Alexa')

fig = go.Figure(data = df,
                layout = layout)

py.iplot(fig)
```

Distribution of Ratings for Alexa

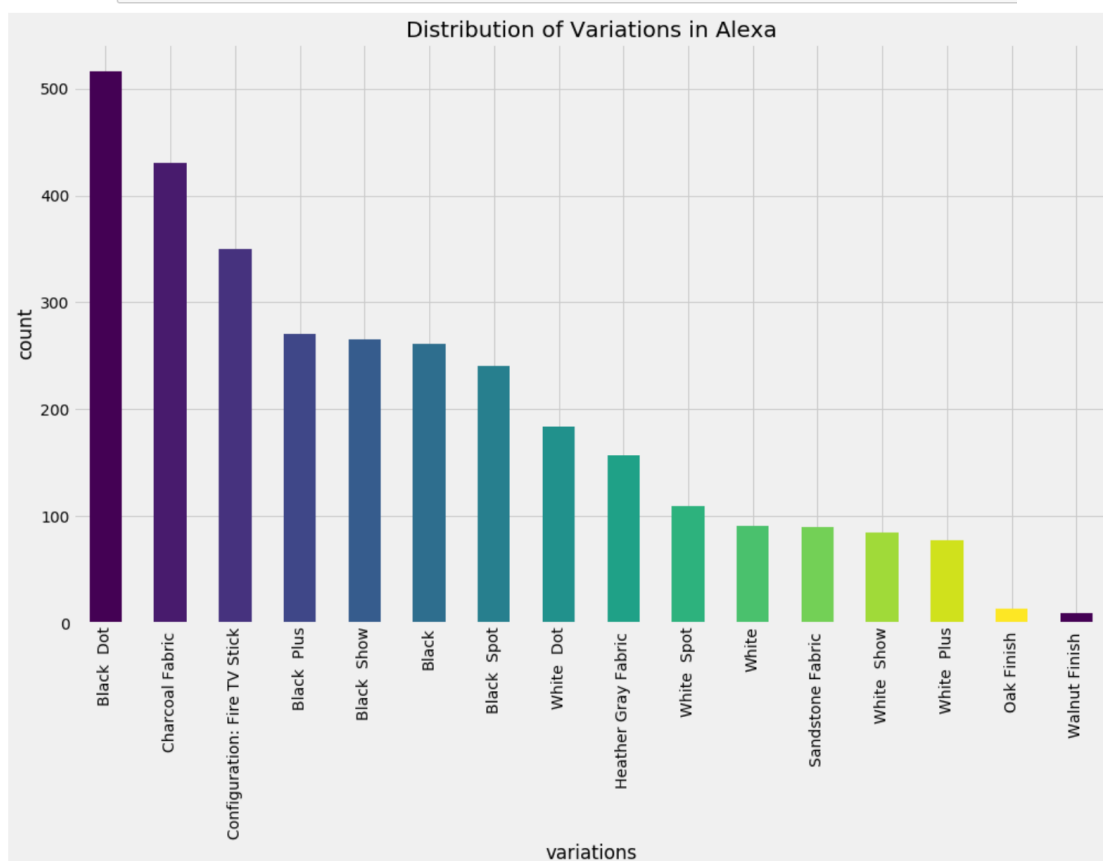


By looking at the pie chart above, we can conclude that most of the ratings are positive for Alexa.

- 72.6% customers have given Alexa a rating of 5 stars.
- 14.4% customers have given Alexa a rating of 4 stars.
- That means that 87% of the total customers have given Alexa at least a good rating.
- 4.38% of customers have given Alexa a rating of 3 stars.
- 3.05% of customers appear to not like Alexa as much as the other customers and chose to give only a 2-star rating to Alexa, whereas 5.11% people did not like Alexa and decided to give only 1-star rating.
- This feedback shows a total of 8.16% of the customers were not satisfied with Alexa.
- Overall, the ratings feedback is very positive, showing almost 90% of the customers being very satisfied with the product.

10. Distribution of variations in Alexa

```
In [10]: color = plt.cm.viridis(np.linspace(0, 1, 15))
data['variation'].value_counts().plot.bar(color = color, figsize = (15, 9))
plt.title('Distribution of Variations in Alexa', fontsize = 20)
plt.xlabel('variations')
plt.ylabel('count')
plt.show()
```



One of the columns in our dataset is called “variations”. The bar chart shows all the different variations of Amazon’s Alexa and their popularity as well. We see that there are 16 different variations of Alexa models. It goes from Black Dot all the way to Walnut Finish. It is clear that Black Dot is the most popular variation of Alexa with more than 500 units out of the 3150 samples in the dataset.

- Black Dot, Charcoal Fabric and Configuration: Fire TV Stick are the most popular models of Amazon’s Alexa.
- Oak Finish and Walnut Finish are the least popular variations

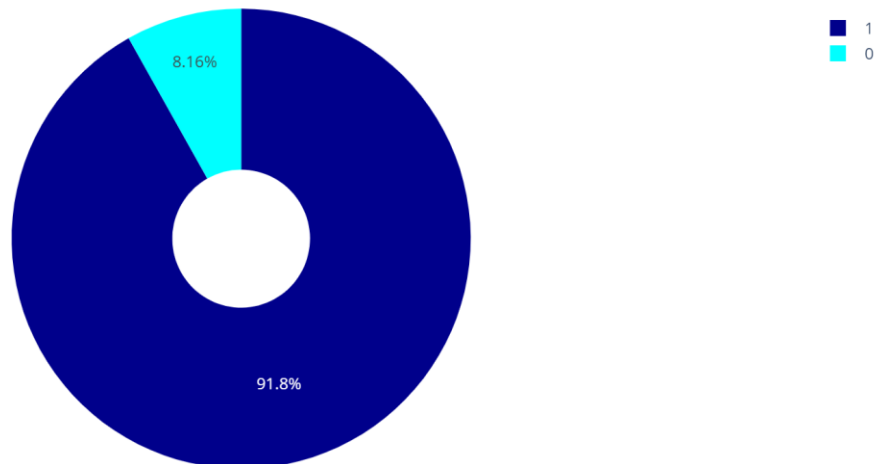
11. Distribution of feedback for Alexa

```
In [11]: feedbacks = data['feedback'].value_counts()
label_feedback = feedbacks.index
size_feedback = feedbacks.values
colors = ['darkblue', 'aqua']

feedback_piechart = go.Pie(labels = label_feedback,
                             values = size_feedback,
                             marker = dict(colors = colors),
                             name = 'Alexa', hole = 0.3)

df2 = [feedback_piechart]
layout = go.Layout(
    title = 'Distribution of Feedbacks for Alexa')
fig = go.Figure(data = df2,
                 layout = layout)
py.iplot(fig)
```

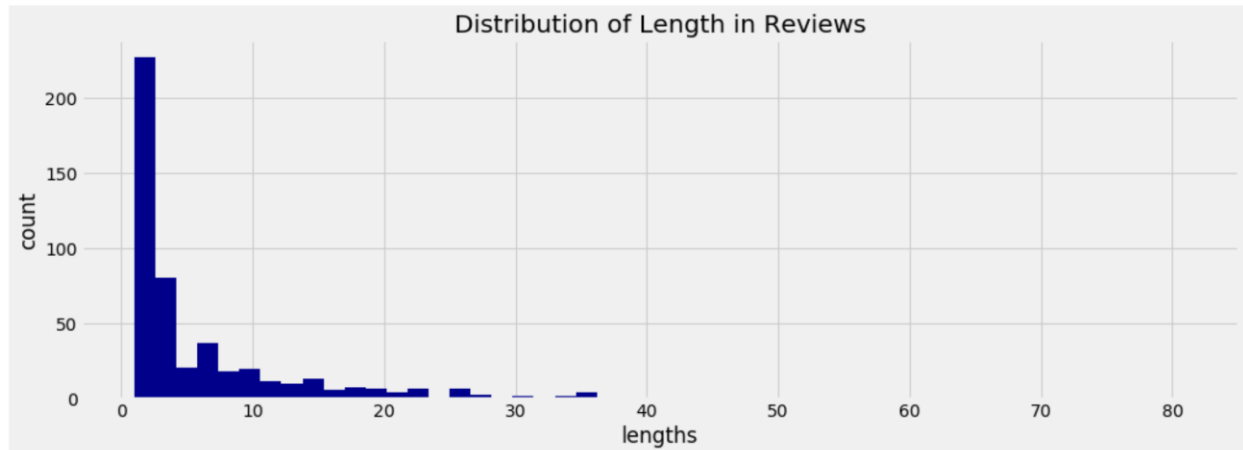
Distribution of Feedbacks for Alexa



This pie chart represents the distribution feedback for Amazon’s Alexa. 91.8% of customers have given a positive feedback for Alexa (3 stars or above), and only 8% of customers have given a negative feedback to Alexa (2 stars or below). This confirms that Alexa is has a very positive feedback from the majority of its customers, and only a small percentage did not like the product.

12. Distribution of length reviews

```
In [12]: data['length'].value_counts().plot.hist(color = 'darkblue', figsize = (15, 5), bins = 50)
plt.title('Distribution of Length in Reviews')
plt.xlabel('lengths')
plt.ylabel('count')
plt.show()
```



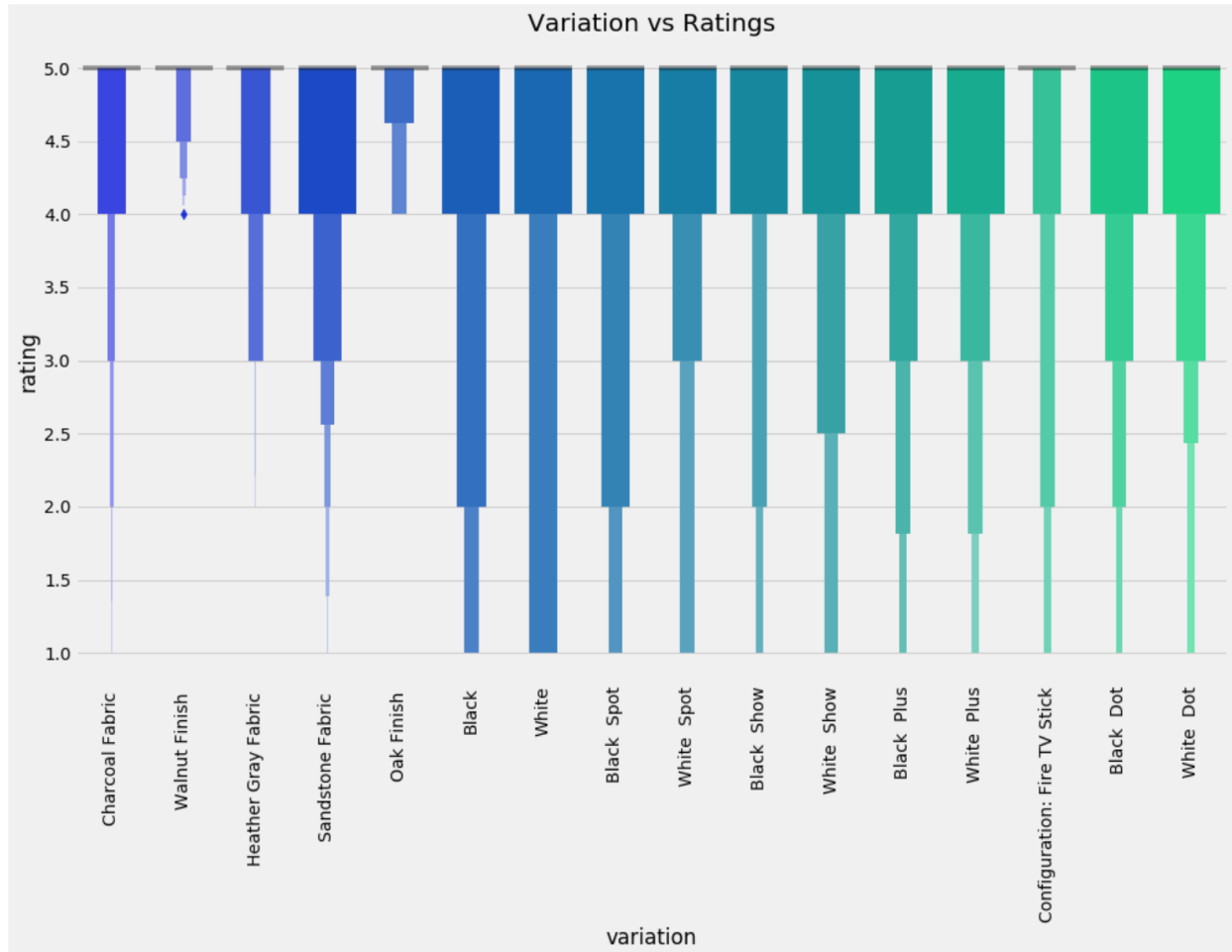
The above Distribution Plot shows the distribution of the length of the reviews written by the customers. This shows what is the average of the length of the reviews written by the customers of Amazon's Alexa. Most of the Reviews are very short, written in less than 3 words total. We can see that most customers write reviews that are between 5 to 20 words long. Very few customers write reviews that are longer than 30 words.

13. Variation vs Ratings

```
In [13]: plt.rcParams['figure.figsize'] = (15, 9)
plt.style.use('fivethirtyeight')

sns.boxenplot(data['variation'], data['rating'], palette = 'winter')
plt.title("Variation vs Ratings")
plt.xticks(rotation = 90)
plt.show()
```

I will be using Matplotlib and Seaborn to make a Bivariate plot in order to analyze the differences between “variations” and “ratings” of Amazon’s Alexa. I am using a Bivariate plot because I am dealing with two variables (variations and ratings).



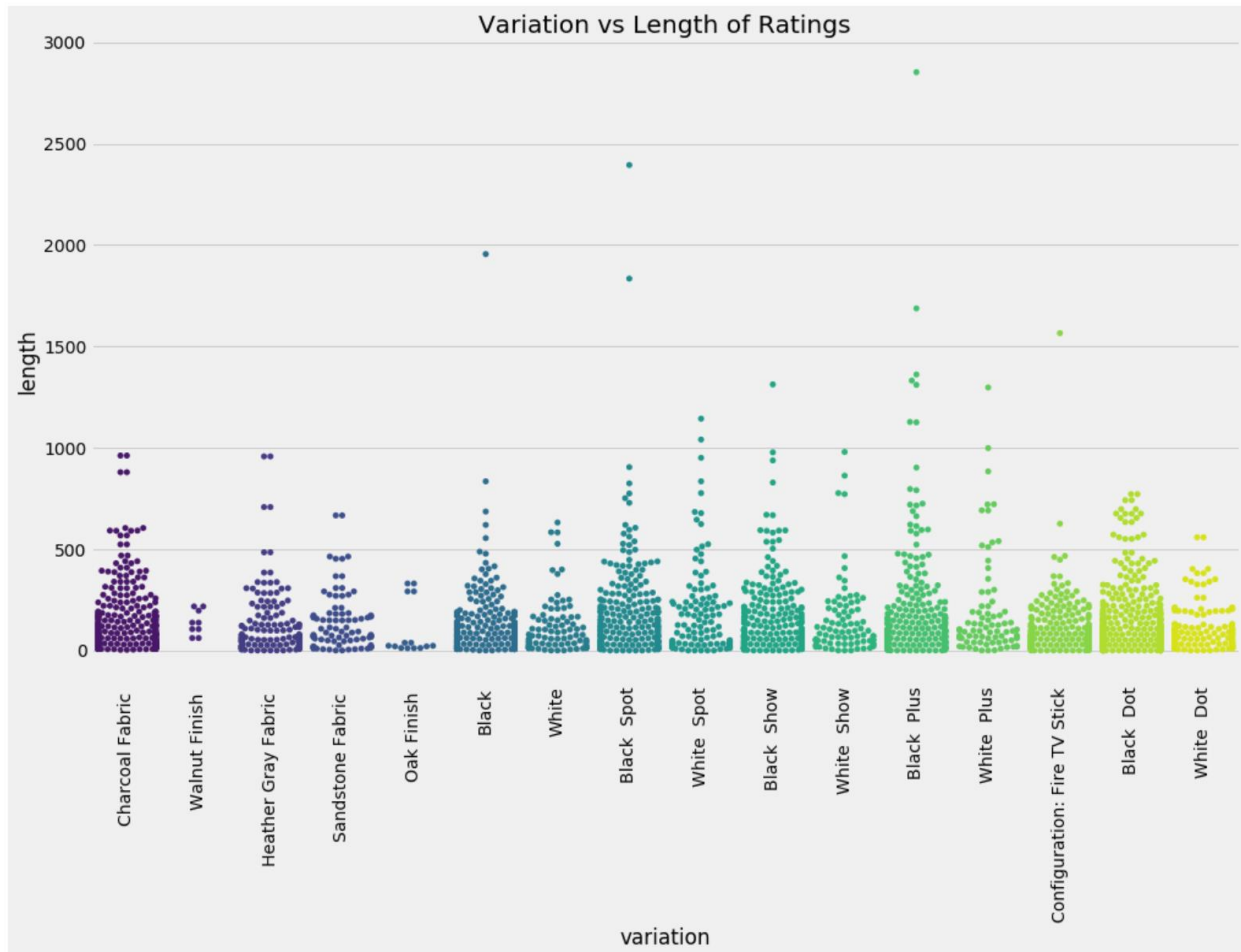
We can draw very interesting conclusions from this plot.

- Even though Walnut Finish and Oak Finish were the least popular variations of Alexa according to #10, they have very high ratings ranging from 4.5 to 5 stars. It shows that these Alexa variations are not very common among its customers, however, they have very positive ratings.
- The darker variations of Alexa, such as the Black variation model, have some negative ratings. We can infer this conclusion because of its popularity among the different Alexa model variations. Thus, due to the majority of customers owning this variation, it will have some negative reviews, even though most of them are positive.
- The White model besides being one of the least popular variation models according to #10, it is also one of the models that have the most negative ratings. We can see that there is a substantial number of ratings below 3 stars for the White model, especially below 2 stars.

14. Variation vs Length of Ratings

```
In [14]: plt.rcParams['figure.figsize'] = (15, 9)
plt.style.use('fivethirtyeight')

sns.swarmplot(data['variation'], data['length'], palette = 'viridis')
plt.title("Variation vs Length of Ratings")
plt.xticks(rotation = 90)
plt.show()
```



The Bivariate plot above shows the relationship between “variation” and “length” of Amazon’s Alexa reviews. In this case, we are looking for which model variation of Alexa’s customers have written the longest and shortest reviews.

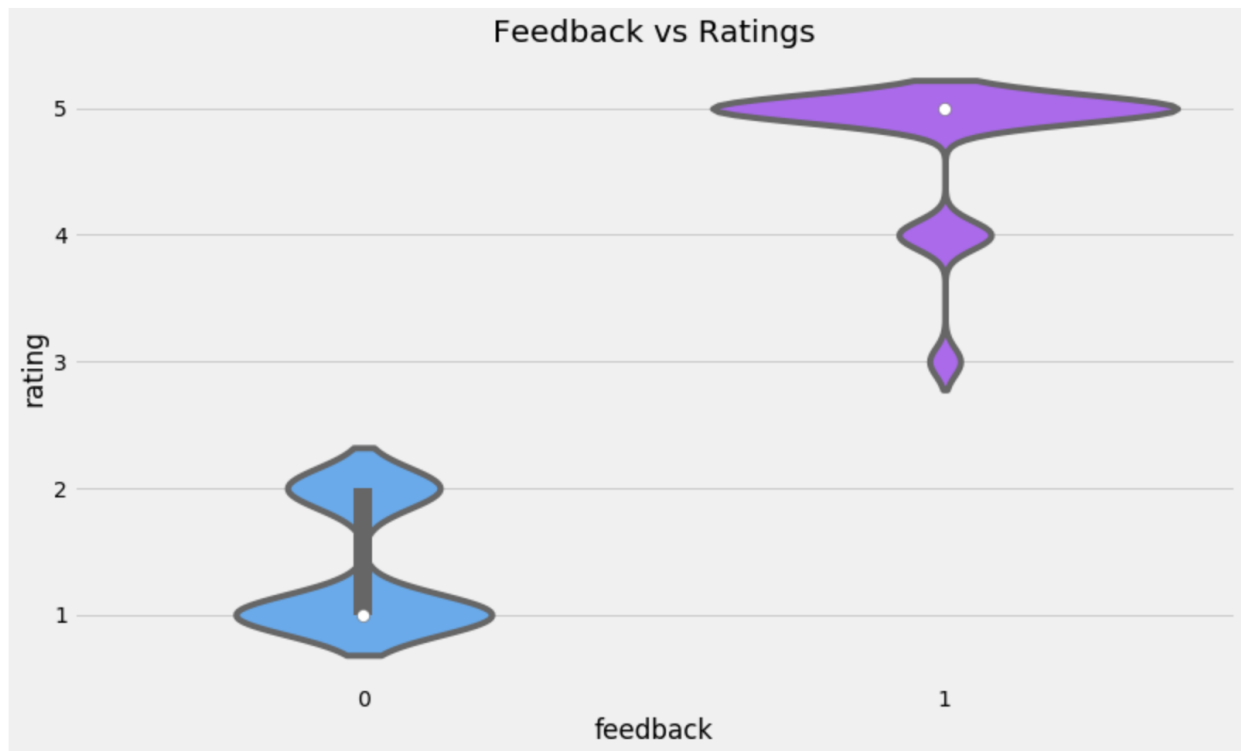
- By looking at the graph, we can spot that the longest reviews were written by customers who own the Black Plus model, with the longest reviews with almost 3 thousand characters (including white space).
- Dissatisfied customers usually tend to leave long reviews explaining their reasons for not liking the product. Based on that, I was hoping to see long reviews for the White model since the ratings were low in this particular model accordingly with the previous graph. However, that was not the case. The reviews for the White model were relatively short when compared with other models.

15. Feedback vs Ratings

```
In [15]: import warnings
warnings.filterwarnings('ignore')

plt.rcParams['figure.figsize'] = (12, 7)
plt.style.use('fivethirtyeight')

sns.violinplot(data['feedback'], data['rating'], palette = 'cool')
plt.title("Feedback vs Ratings")
plt.show()
```



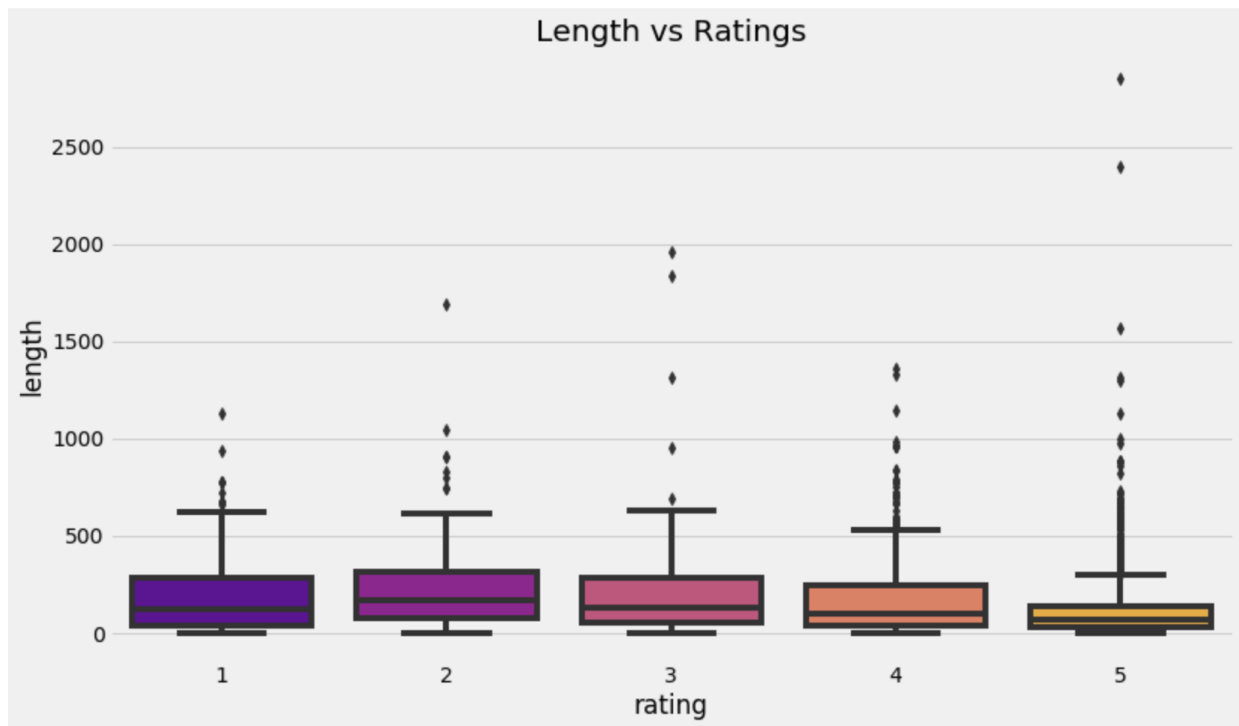
The Bivariate plot above is a violin plot between Feedback and Rating. We can conclude from the graph that the Alexa reviews that have 0 feedback have lower ratings ranging from 1 to 2 stars, but mostly 1 star only. Whereas the Alexa reviews having a feedback equal 1 have higher ratings between 3 to 5 stars. According to the graph, the majority of the feedback reviews has a 5-star rating.

16. Length vs Rating

```
In [16]: import warnings
warnings.filterwarnings('ignore')

plt.rcParams['figure.figsize'] = (12, 7)
plt.style.use('fivethirtyeight')

sns.boxplot(data['rating'], data['length'], palette = 'plasma')
plt.title("Length vs Ratings")
plt.show()
```



The Bivariate plot shows the relationship between length and rating. Here we are looking how long the customer reviews are based on their rating. It is worth noting that all the reviews have pretty similar lengths regardless of their rating. However, there's a clear difference between the length of low rating reviews and high rating reviews. According to the graph and as previously mentioned, low rating reviews tend to be longer than high rating reviews. Most customers that gave Alexa 5-stars wrote shorter reviews than customers that gave 1 or 2-stars. That might be due to the fact that unsatisfied customers feel the need to explain the reasons for not liking the product. While satisfied customers feel happy, hence do not feel the same urgency to write long reviews.

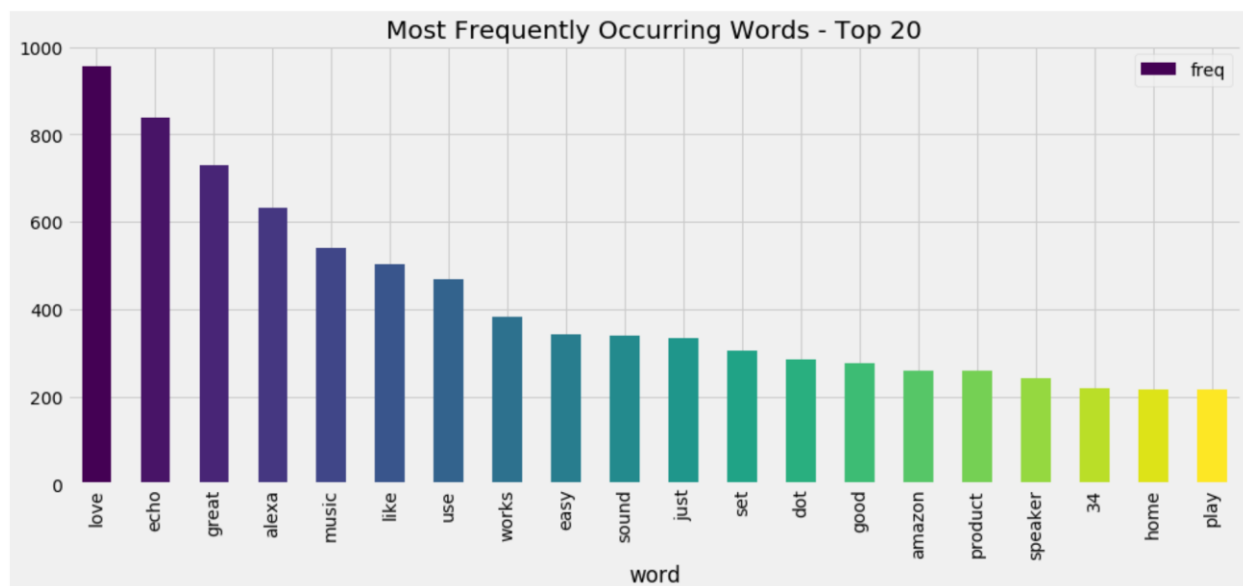
17. Most Frequently Occurring Words – Top 20

```
In [17]: from sklearn.feature_extraction.text import CountVectorizer

cv = CountVectorizer(stop_words = 'english')
words = cv.fit_transform(data.verified_reviews)
sum_words = words.sum(axis=0)

words_freq = [(word, sum_words[0, idx]) for word, idx in cv.vocabulary_.items()]
words_freq = sorted(words_freq, key = lambda x: x[1], reverse = True)
frequency = pd.DataFrame(words_freq, columns=['word', 'freq'])

plt.style.use('fivethirtyeight')
color = plt.cm.viridis(np.linspace(0, 1, 20))
frequency.head(20).plot(x='word', y='freq', kind='bar', figsize=(15, 6), color = color)
plt.title("Most Frequently Occuring Words - Top 20")
plt.show()
```



The bar plot represents the most frequent words among all of the reviews analyzed from the customers. By looking at the graph, we can have a good idea on how the customers think and feel regarding Amazon's Alexa.

The words “love” and “great” are two of the most frequent words among all of the reviews which suggests that most customers had very positive feelings towards Alexa. This is foreseen since 91.8% (#11) of the reviews had a positive rating. Other frequent words that suggest Alexa is doing well are “amazing”, “like”, “easy”, “works”, and “good”.

18. Vocabulary from Reviews

```
In [18]: from wordcloud import WordCloud

wordcloud = WordCloud(background_color = 'lightcyan', width = 2000, height = 2000).generate_from_frequencies(dict(words_freq))

plt.style.use('fivethirtyeight')
plt.figure(figsize=(10, 10))
plt.axis('off')
plt.imshow(wordcloud)
plt.title("Vocabulary from Reviews", fontsize = 20)
plt.show()
```

The code above imports and executes a library called Word Cloud. Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud [\[10\]](#).



This Word Cloud visualization shows all the most frequently used and most relevant words analyzed from the customer reviews. The bigger the word, the higher is the frequency for that word been written by a customer.

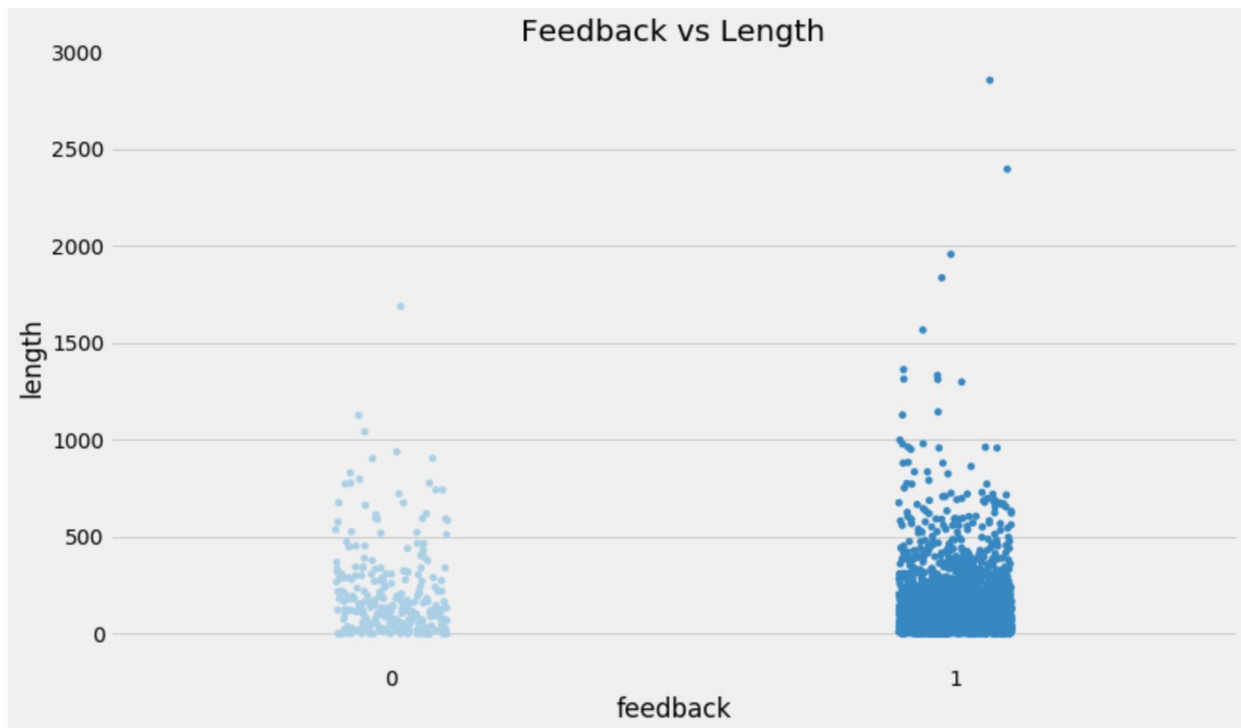
As seen in the previous result, “love”, “great”, “like”, are very frequent words written by Alexa customers. This reinforces the customer’s positive feedback towards Alexa.

19.Feedback vs Length

```
In [19]: import warnings
warnings.filterwarnings('ignore')

plt.rcParams['figure.figsize'] = (12, 7)
plt.style.use('fivethirtyeight')

sns.stripplot(data['feedback'], data['length'], palette = 'Blues')
plt.title("Feedback vs Length")
plt.show()
```



The Bivariate graph above shows the relationship between feedback and length. We notice that there is a lot more positive feedback than negative feedback. This makes sense since the majority of customers had a positive rating towards Alexa. We also notice that the length of positive feedbacks is greater than the negatives one. This might be due to the total number of positive feedback reviews greatly surpassing the number of negative feedback reviews. Customers that are satisfied with Alexa wrote longer reviews than customers that did not like Alexa as much.

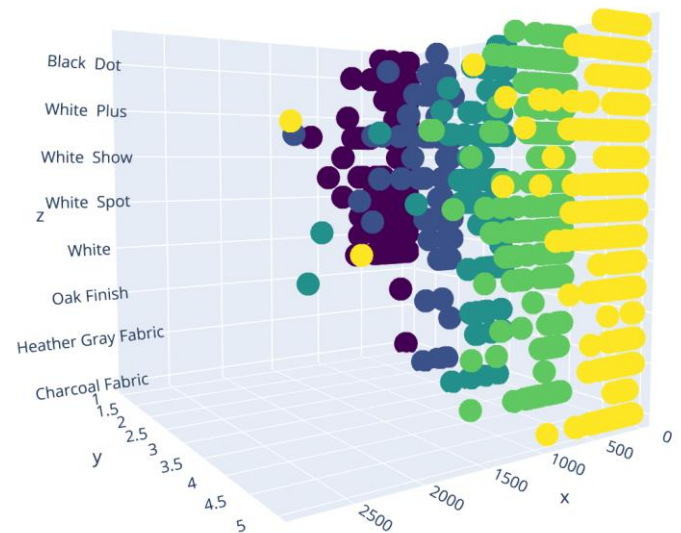
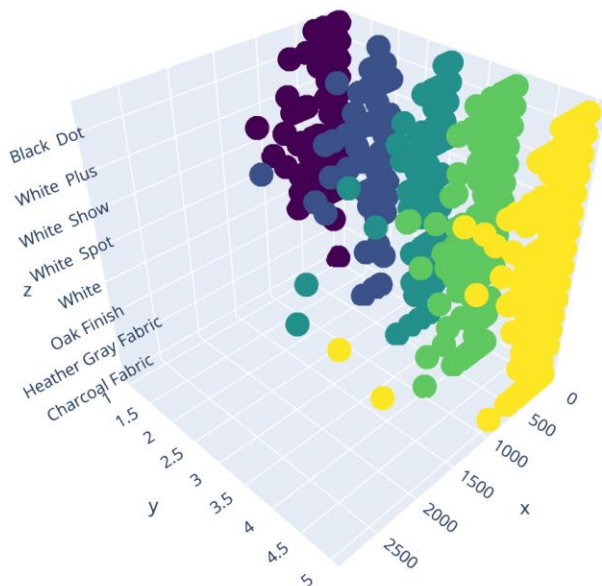
6.2 3D Scatter Plot

20. Ratings vs Length vs Variation

```
In [20]: trace = go.Scatter3d(
    x = data['length'],
    y = data['rating'],
    z = data['variation'],
    name = 'Amazon Alexa',
    mode='markers',
    marker=dict(
        size=10,
        color = data['rating'],
        colorscale = 'viridis',))

df = [trace]
layout = go.Layout(
    title = 'Length vs Variation vs Ratings',
    margin=dict(
        l=0,
        r=0,
        b=0,
        t=0))

fig = go.Figure(data = df, layout = layout)
iplot(fig)
```



The two pictures above represent the same 3-dimension plot exploring the relationships between ratings, length, and variation of Amazon's Alexa reviews. Each dot in this graph represents a review made by a customer and how that particular review relates to its rating, length and model variation.

6.3 spaCy

spaCy is an open-source software library for advanced natural language processing. Unlike NLTK, which is widely used for teaching and research, spaCy focuses on providing software for production usage. It is designed with the applied data scientist in mind, meaning it does not weigh the user down with decisions over what esoteric algorithms to use for common tasks and it's fast. Incredibly fast (it's implemented in Cython). If you are familiar with the Python data science stack, spaCy is your NumPy for NLP – it's reasonably low-level, but very intuitive and performant [\[11\]](#).

21. spaCy

```
In [21]: import spacy
nlp = spacy.load('en_core_web_sm')

def explain_text_entities(text):
    doc = nlp(text)
    for ent in doc.ents:
        print(f'Entity: {ent}, Label: {ent.label_}, {spacy.explain(ent.label_)})')

for i in range(15, 50):
    one_sentence = data['verified_reviews'][i]
    doc = nlp(one_sentence)
    spacy.displacy.render(doc, style='ent', jupyter=True)
```

We have only been using **Alexa ORG** for **a couple of days DATE** and are having a lot of fun with our new toy. It like having a new household member! We are trying to learn all the different features and benefits that come with it.

We love the size of the **2nd ORDINAL** generation echo. Still needs a little improvement on sound

"I liked the original **Echo LOC**. This is the same but shorter and with greater fabric/color choices. I miss the volume ring on top, now it's just the plus/minus buttons. Not a big deal but the ring w as comforting. :) Other than that, well I do like the use of a standard USB charger /port instead of the previous round pin. Other than that, I guess it sounds the same, seems to work the same, still answers to **Alexa/Echo/Computer ORG**. So what's not to like? :)"

Love the **Echo LOC** and how good the music sounds playing off it. Alexa understands most commands but it is difficult at times for her to find specific playlists or songs on Spotify. She is good with **Amazon Music ORG** but is lacking in other major programs.

"We love **Alexa ORG** ! We use her to play music, play radio through **iTunes GPE**, play podcasts through **Anypod PERSON**, and set reminders. We listen to our flash briefing of news and weather **every morning TIME**. We rely on our custom lists. We like being able to voice control the volume. We're sure we'll continue to find new uses. Sometimes it's a bit frustrating when **Alexa ORG** doesn't understand what we're saying."

"Have only had it set up for **a few days DATE**. Still adding smart home devices to it. The speaker is great for playing music. I like the size, we have it stationed on the kitchen counter and it's not intrusive to look at."

You may notice that spaCy identifies key words in a sentence and is able to assign meaning to these words. For example, it knows that “Alexa”, “Alexa/Echo/Computer”, “Amazon Music” are organization names instead of the name of a person. It also knows that “every morning” and “a few days” are time and date references. This is useful to identify the key concepts in each sentence and have an idea of what the customer is talking about without actually having to read the entire review.

6.4 Natural Language Processing

22. Importing the libraries for NLP

```
In [22]: # importing the Libraries for Natural Language Processing
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\muril\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
```

23. Creating a corpus

```
In [23]: # cleaning the text
corpus = []
for i in range(0, 3150):
    review = re.sub('[^a-zA-Z]', ' ', data['verified_reviews'][i])
    review = review.lower()
    review = review.split()
    ps = PorterStemmer()
    review = [ps.stem(word) for word in review if not word in set(stopwords.words('english'))]
    review = ' '.join(review)
    corpus.append(review)
```

In linguistics, a **corpus** is a large and structured set of texts (nowadays usually electronically stored and processed). In corpus linguistics, they are used to do statistical analysis and hypothesis testing, checking occurrences or validating linguistic rules within a specific language territory [\[12\]](#). We are using the column “verified_reviews” within the dataset to create a corpus that will later be analyzed. The field “verified_reviews” represents the written reviews made by the Alexa consumers. They are the body of text that will be analyzed in the NLP model. Also, we are transforming all words to lower case, taking these words and putting into a list, removing the stop words (words that don’t really offer as much value), and transforming that list into a text corpus once again.

24. Creating a Bag of Words

```
In [24]: # creating bag of words
from sklearn.feature_extraction.text import TfidfVectorizer
cv = CountVectorizer(max_features = 2500)
x = cv.fit_transform(corpus).toarray()
y = data.iloc[:, 4].values
```

The **bag-of-words** is a representation of text that describes the occurrence of words within a document. The model is simple to understand and implement. It is a way of extracting features from the text for use in machine learning algorithms [\[13\]](#). The count vectorizer converts a collection of text documents to a matrix of token counts. The max_features function creates a matrix of words when analyzing the corpus. In this

case, we passed 2500 to max_features, that means creating a feature matrix out of the 2500 most frequent words across the text document.

In summary, we are only analyzing the 2500 most frequent words among all of the Alexa reviews.

25. Splitting the dataset into Training and Test data

```
In [25]: # splitting the dataset into Training and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
```

Here we are importing the **train_test_split** function from the Scikit-learn library. The train_test_split function is used for splitting data arrays into two subsets: for training data and for testing data. With this function, there is no need to divide the dataset manually. By default, Scikit-learn train_test_split will make random partitions for the two subsets. However, it can also be specified a random state for the operation.

The reason for using this function is because when using the same dataset for both training and testing, it can leave room for miscalculations. Thus, this increases the chances of inaccurate predictions. The train_test_split function easily breaks a dataset while pursuing an ideal model [\[14\]](#).

26. Random Forest

```
In [26]: # Fitting Random Forest classifier with 100 trees to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)
```

```
Out[26]: RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                                criterion='entropy', max_depth=None, max_features='auto',
                                max_leaf_nodes=None, max_samples=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, n_estimators=100,
                                n_jobs=None, oob_score=False, random_state=0, verbose=0,
                                warm_start=False)
```

The random forest is a classification algorithm consisting of many decision trees. ***It uses bagging and feature randomness when building each individual tree to create an uncorrelated forest of trees*** whose prediction by committee is more accurate than that of any individual tree. Basically, each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. ***A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.*** The low correlation between models is the key. The reason for this wonderful effect is that the trees protect each other from their individual errors (as long as they don't constantly all err in the same direction). While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction [\[15\]](#).

27. Predicting results

```
In [27]: # Predicting the Test set results
y_pred = classifier.predict(X_test)
```

Assigning the predicted results into the “y_pred” variable.

28. Confusion Matrix

```
In [28]: # Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

In the field of machine learning and classification, a confusion matrix is a table that is often used to describe the performance of classification model (classifier) on a set of test data (X_test, y_test) for which the true values are known. A confusion matrix is basically a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which the classification model is confused when it makes predictions. It gives insight not only into the errors being made by a classifier but more importantly the types of errors that are being made [\[16\]](#).

Confusion Matrix can be divided into the following definition terms (*see figure 3 below*):

- Positive (P): Observation is positive (for example: is an apple).
- Negative (N): Observation is not positive (for example: is not an apple).
- True Positive (TP): Observation is positive and is predicted to be positive.
- False Negative (FN): Observation is positive but is predicted negative.
- True Negative (TN): Observation is negative and is predicted to be negative.
- False Positive (FP): Observation is negative but is predicted positive.

29. Classification Report

```
In [29]: print(cm)
print(classification_report(y_test,y_pred))
print(accuracy_score(y_test, y_pred))

[[ 18  36]
 [   0 576]]
              precision    recall  f1-score   support

         0              1.00      0.33      0.50         54
         1              0.94      1.00      0.97        576

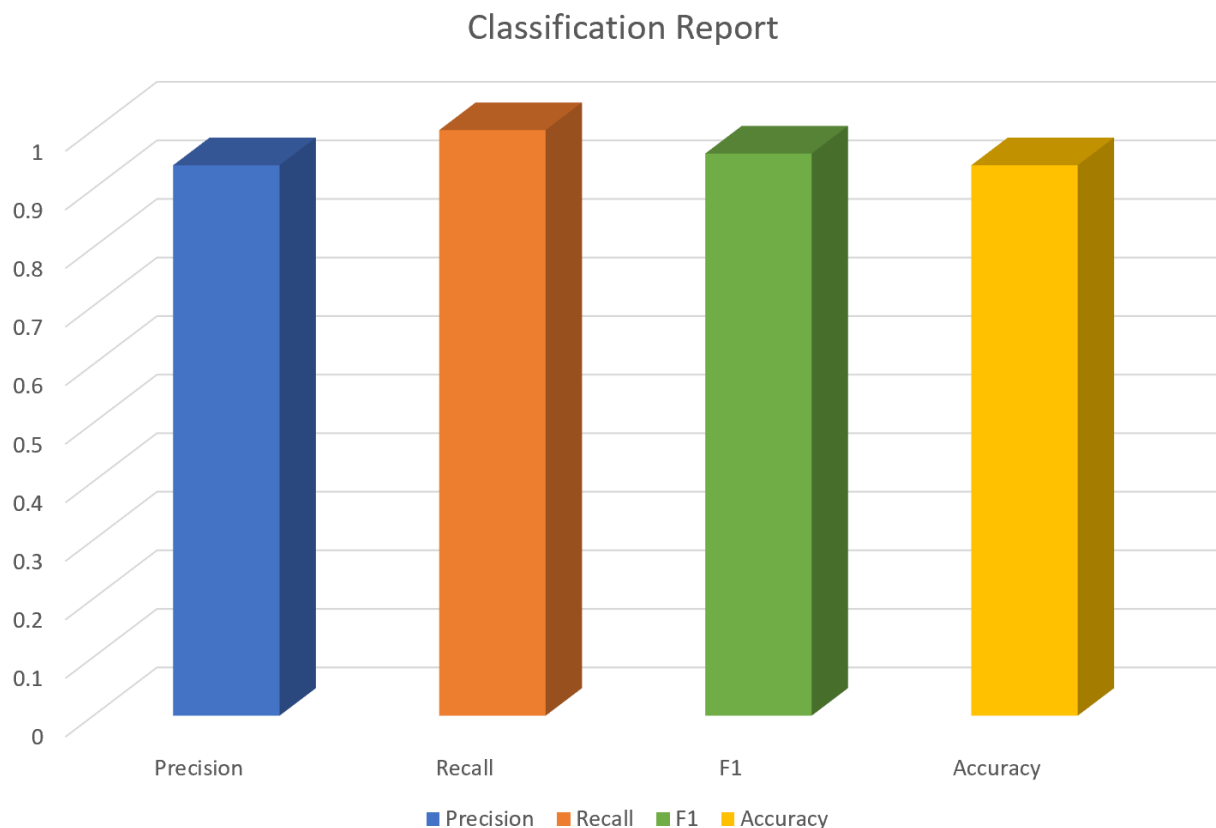
 accuracy              0.94         630
 macro avg              0.97      0.67      0.73         630
 weighted avg           0.95      0.94      0.93         630

0.9428571428571428
```

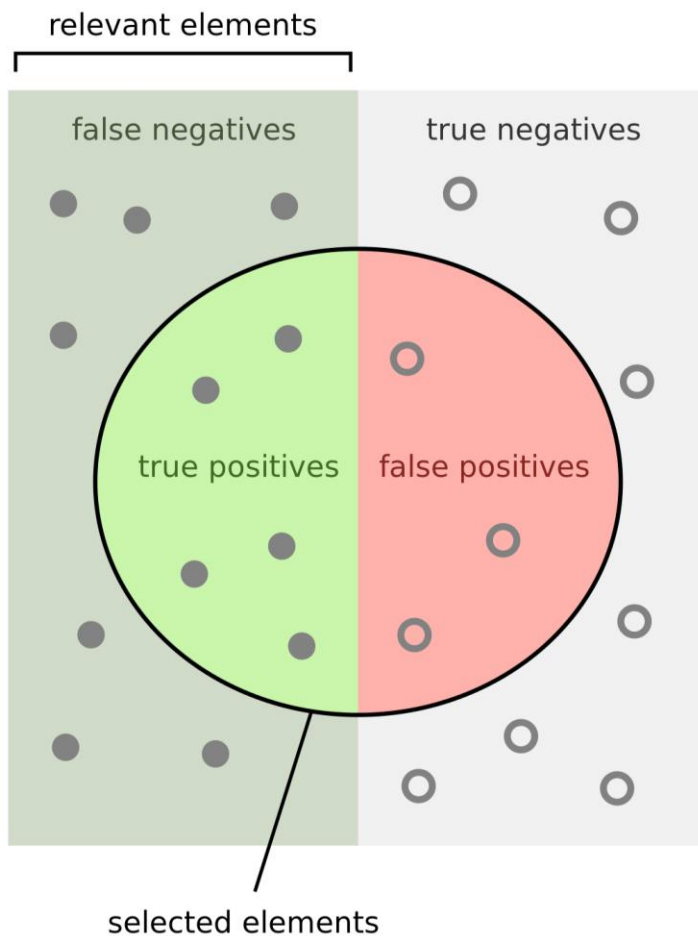
As mentioned previously on #28, the above table is known as confusion matrix. Below is the results with precision of 94%, recall of 100% (perfect), F1 Score of 97%, and overall accuracy of 94.28%.

- **Precision:** the fraction of relevant instances retrieved from total instances retrieved. This is represented by the number of correct predictions among the reviews predicted to be positive.
- **Recall:** the fraction of relevant instances retrieved over the total relevant instances retrieved. This is represented by the percent of all positive reviews that the model found.
- **F1-Score:** is the harmonic mean of precision and recall. F1-Score takes both precision and recall into account.
- **Accuracy:** The percentage of instances where the model predicted the correct value. This is represented by the percent of correct cases in predicting if a review would be positive or negative [\[17\]](#).

In conclusion, the Random Forest classifier algorithm with 100 trees (#26) worked efficiently to train the model in predicting positive and negative reviews made by customers, with an overall accuracy of 94.28%. The precision percentage, in other words, the number of correct predictions among the reviews predicted to be positive is 94%. Moreover, the recall percentage of all positive reviews that the model found is 100% (perfect score). And the F1-Score, a combination of precision and recall is 97%. The graph below is a visual representation of the results from the classification report.



Evaluating ML Models: Precision, Recall, F1 and Accuracy



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

Figure 3: <https://medium.com/analytics-vidhya/evaluating-ml-models-precision-recall-f1-and-accuracy-f734e9fcc0d3>

7. Conclusion

In conclusion, the results show the vast majority of reviews written by Amazon's Alexa customers were highly positive. Overall, 87% of the customers gave Alexa at least 4-star ratings. When it comes to positive and negative feedback scores, 91.8% of customers have given a positive feedback, and only 8% of customers have given a negative feedback to Alexa. This shows that Alexa customers are very pleased with their purchase. Only a small percentage had some kind of complaint towards Alexa or did not like the product.

The different model variations of Alexa that have darker color tones (e.g. Black Dot model) were ranked more popular against models with lighter color tones (e.g. White model). For the White model, besides being one of the least popular models, it is also among the models with the most negative ratings.

Regarding the length of reviews written by customers, most reviews are very short, written in less than 3 words total. Most customers write reviews between 5 to 20 words long. Very few customers write reviews that are longer than 30 words. Although the models Walnut Finish and Oak Finish were the least popular model variations of Alexa, they have very high ratings ranging from 4.5 to 5 stars. Also, the average of the longest reviews was written by customers who own the Black Plus model.

In regard to the relationship between length of reviews and model variation, I was hoping to see longer reviews for the models who had very low rating scores. Thus, it was expected to see longer reviews for the White model since the ratings were low in this particular Alexa variation. However, that was not the case. The reviews for the White model were relatively short when compared with other models. The relationship between the length of reviews and their ratings, most customers that gave Alexa 5-star ratings wrote shorter reviews than customers that gave 1 or 2-stars. This result is expected since dissatisfied customers write longer reviews explaining their reasons for not liking the product.

The NLP model was very effective in predicting the difference between positive and negative reviews. With 94.28% overall accuracy, I conclude that the random forest classifier algorithm is very effective and works really well for linguistic analysis.

7.1 Moving Forward

For a future study, it would be interesting to analyze the existence of similar patterns in the **negative** reviews that were identified by the NLP model. It would allow the discovery of similar patterns of dissatisfied customers by selecting the reviews that have a negative feedback. Within those negative reviews, it would be possible to identify common words that are often used, thus identifying problems that are frequently found by customers. Hence, this information can be used to improve newer versions of Alexa that are yet to be created and distributed to future customers. Hoping that in the future, the number of positive reviews will considerably increase, making more customers satisfied with their purchase and adding value to Amazon products.

8. Self-reflection

When deciding what to study for this course, I considered a few things as the following. I have always been fascinated with data and how it can be used to help businesses make smarter and better data-driven decisions. As I approach the last semester of my master's degree, I wanted to gain a deeper knowledge of machine learning and data science methodologies as I believe it would be valuable for my professional career. I decided to study Natural Language Processing because it is something that can be easily translated to the real world, has profound insights and adds meaningful results to business problems. I believe the skills that I learned throughout this course will be extremely helpful in my future career as I am very interested in the technology fields of data science and business intelligence engineering.

Moreover, I would like to thank Dr. Moody for being a great mentor and offering me this wonderful opportunity to continue to learn new skills and take new challenges. This course was fun yet challenging, and I deeply appreciate his help and guidance. After finishing this course, I feel more prepared to tackle real life problems, add valuable data-driven business decisions, identify better business opportunities, and spot inefficient business processes.



9. Resources:

1. Natural Language Processing (NLP)
https://en.wikipedia.org/wiki/Natural_language_processing
2. The unexpected benefits of data analytics
<https://www.cio.com/article/3249905/the-unexpected-benefits-of-data-analytics.html>
3. Structured vs Unstructured Data
<https://learn.g2.com/structured-vs-unstructured-data>
4. Stop Words
<https://www.geeksforgeeks.org/removing-stop-words-nltk-python/>
5. The Basics of Natural Language Processing
<https://www.youtube.com/watch?v=d4gGtcobq8M>
6. NoSQL vs Relational Databases
<https://www.mongodb.com/scale/nosql-vs-relational-databases>

7. Python for Data Science
<https://www.geeksforgeeks.org/python-for-data-science/>
8. TSV file definition
https://en.wikipedia.org/wiki/Tab-separated_values
9. pandas.DataFrame.describe
<https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.describe.html>
10. Word Cloud
<https://www.geeksforgeeks.org/generating-word-cloud-python/>
11. A short introduction to NLP in Python with spaCy
<https://towardsdatascience.com/a-short-introduction-to-nlp-in-python-with-spacy-d0aa819af3ad>
12. Text corpus
https://en.wikipedia.org/wiki/Text_corpus
13. Bag-of-Words
<https://medium.com/greyatom/an-introduction-to-bag-of-words-in-nlp-ac967d43b428>
14. Splitting Datasets With the Sklearn train_test_split Function
<https://www.bitdegree.org/learn/train-test-split>
15. Understanding Random Forest
<https://towardsdatascience.com/understanding-random-forest-58381e0602d2>
16. Confusion Matrix in Machine Learning
<https://www.geeksforgeeks.org/confusion-matrix-machine-learning/>
17. Evaluating ML Models: Precision, Recall, F1 and Accuracy
<https://medium.com/analytics-vidhya/evaluating-ml-models-precision-recall-f1-and-accuracy-f734e9fcc0d3>