Individual Assignment 1

1. Use the tablet review data to calculate the distribution of sentiment polarities (i.e., positive, neutral, negative) for the following Kindle attributes: screen, customer service, weight, and price. Hint: First clean, remove stop words, and stem. Then filter for only Kindle reviews. Then filter the reviews by the attributes in the same way we filtered for "kindle" reviews.

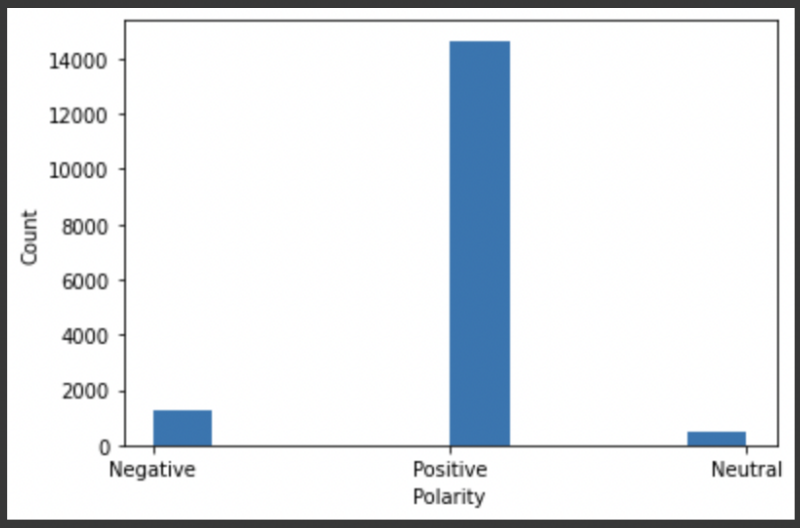
A. What is the ratio of positive reviews to negative reviews for each of these attributes? Hint: Use value_counts from pandas.

Ans: (Python notebook attached after questions)

We begin our analysis by importing the 'products.csv' file as a Pandas DataFrame. In order to conduct a textual analysis of the reviews of the products, we first need to clean and pre-process the data. We then filter out reviews that have the string 'kindle' in order to extract all reviews about the Kindle. Now, we need to perform sentiment analysis for the Kindle attributes: screen, customer service, weight, and price. Just as we extracted 'kindle' reviews and saved them as 'reviews_sample', I created different data frame objects for 'reviews_sample_screen' for reviews that have both 'Kindle' and screen keywords, and so on for all 4 attributes. We then define a function to preprocess the text in the selected reviews, remove all stop words, stem each word before we analyze the data, and save it under a new column 'Final Reviews' for each attribute separately. Then we can finally analyze the polarity score for the text using TextBlob which uses a lexicon-based method for scoring. We calculate the score and polarity for each attribute and then plot it on a bar graph using matplotlib to visualize the polarities.

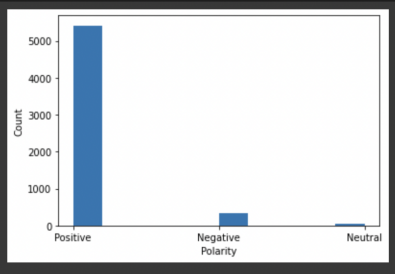
Overall polarity for 'Kindle':

```
# plot the distribution of polarities
plt.hist(reviews_sample['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```



-Polarity and Count for **'Kindle' with 'screen'** as the attribute:

```
plt.hist(reviews_sample_screen['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```



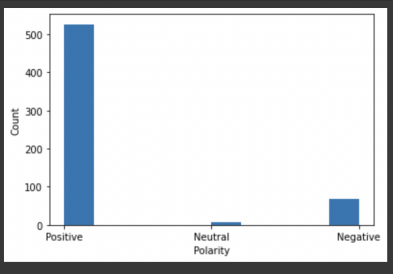
The ratio of Positive: Negative = **5417/343 = 15.8**

```
reviews_sample_screen['Polarity'].value_counts()
```

```
Positive    5417
Negative     343
Neutral       46
Name: Polarity, dtype: int64
```

-Polarity and Count for **'Kindle' with 'customer-service**' as the attribute:

```
[35] plt.hist(reviews_sample_service['Polarity'])
     plt.xlabel('Polarity')
     plt.ylabel('Count')
     plt.show()
```



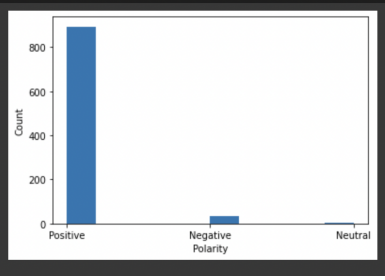
The ratio of Positive: Negative = **526/69 = 7.62**

```
[39] reviews_sample_service['Polarity'].value_counts()
```

```
Positive    526
Negative     69
Neutral       8
Name: Polarity, dtype: int64
```

-Polarity and Count for **'Kindle' with 'weight'** as the attribute:

```
plt.hist(reviews_sample_weight['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```



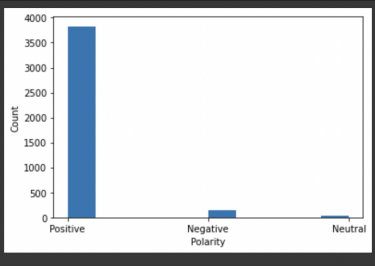
The ratio of Positive: Negative = **893/35 = 25.51**

```
[40] reviews_sample_weight['Polarity'].value_counts()

Positive    893
Negative     35
Neutral       3
Name: Polarity, dtype: int64
```

-Polarity and Count for **'Kindle' with 'price'** as the attribute:

```
[37] plt.hist(reviews_sample_price['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```



The ratio of Positive: Negative = **3828/143 = 26.77**

```
[41] reviews_sample_price['Polarity'].value_counts()

Positive    3828
Negative     143
Neutral       27
Name: Polarity, dtype: int64
```

Also, based on the Regression between each review's Polarity and its star 'Rating' we can see that the polarity **correlates** with the star rating.

2) Run Latent Dirichlet Allocation for the Amazon Kindle reviews after cleaning and removing the stop words (do not stem) for 3, 5, and 7 topics (you can use the same hyperparameters and seed (9651) in the text analysis ipynb reviewed in class).

A) Give names to the topics based on the top ten words of each model:

We begin the analysis by cleaning the reviews which contain the string 'kindle' as a keyword in the Dataframe created from the 'products.csv' file.

We define 3 LDA(Latent Dirichlet Allocation) models based on the number of Topics for the corpus.

-LDA with the number of topics=3 yields the following words(top 10):

```
lda_model_3.print_topics(num_words=10)
```

```
[(0,
  '0.038*"kindle" + 0.037*"fire" + 0.014*"books" + 0.012*"ipad" + 0.011*"great" + 0.011*"love" + 0.011*"read" + 0.010*"reading" + 0.010*"use" + 0.010*"like"'),
 (1,
  '0.039*"kindle" + 0.028*"fire" + 0.017*"amazon" + 0.012*"one" + 0.010*"would" + 0.009*"get" + 0.006*"christmas" + 0.006*"got" + 0.006*"bought" +
0.005*"time"'),
 (2,
  '0.014*"tablet" + 0.010*"device" + 0.010*"android" + 0.009*"app" + 0.009*"apps" + 0.008*"screen" + 0.007*"ipad" + 0.007*"amazon" + 0.007*"like" +
0.006*"use"')]
```

We can also see the weight on each of the top ten words in the three topics.

Based on these, we can see

Topic 0: Like reading on Kindle

Topic 1: Kindle as Christmas Gift

Topic 2: Like apps on tablet

-LDA with the number of topics=5 yields the following words(top 10):

```
lda_model_5.print_topics(num_words=10)
```

```
[(0,
  '0.025*"screen" + 0.012*"kindle" + 0.012*"reading" + 0.012*"battery" + 0.011*"touch" + 0.010*"use" + 0.009*"like" + 0.008*"read" + 0.008*"life" + 0.008*"e"'),
 (1,
  '0.050*"kindle" + 0.046*"fire" + 0.017*"books" + 0.015*"love" + 0.015*"amazon" + 0.011*"read" + 0.010*"one" + 0.010*"would" + 0.010*"use" + 0.009*"great"'),
 (2,
  '0.015*"tablet" + 0.013*"android" + 0.012*"device" + 0.012*"app" + 0.012*"apps" + 0.010*"amazon" + 0.007*"like" + 0.006*"screen" + 0.005*"use" +
0.005*"one"'),
 (3,
  '0.030*"kindle" + 0.019*"fire" + 0.018*"amazon" + 0.011*"one" + 0.010*"get" + 0.010*"would" + 0.007*"problem" + 0.007*"back" + 0.006*"customer" +
0.006*"service"'),
 (4,
  '0.067*"ipad" + 0.023*"fire" + 0.023*"kindle" + 0.015*"great" + 0.014*"tablet" + 0.013*"price" + 0.011*"apple" + 0.009*"device" + 0.009*"screen" +
0.008*"good"')]
```

We can also see the weight on each of the top ten words in the five topics.

Based on these, we can see

Topic 0: Great screen and battery life, like reading on Kindle

Topic 1: Kindle as Christmas Gift

Topic 2: Like apps on tablet, like screen

Topic 3: Bad customer service due to problem

Topic 4: Great price, great screen

-LDA with the number of topics=7 yields the following words(top 10):

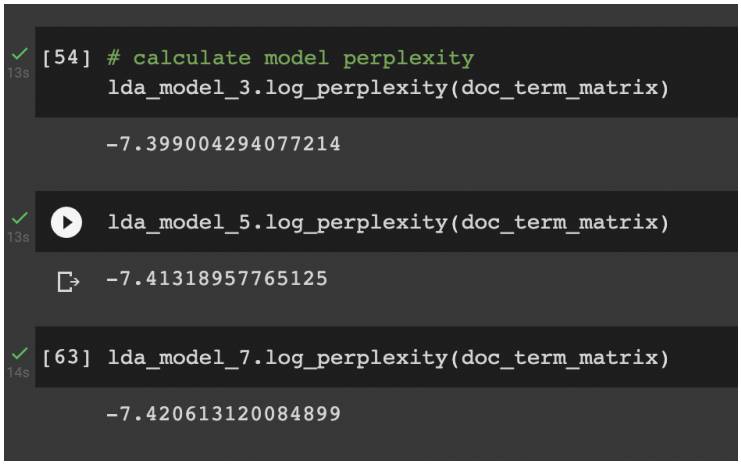
```
lda_model_7.print_topics(num_words=10)

[(0,
  '0.040*"tablet" + 0.021*"android" + 0.012*"apps" + 0.012*"app" + 0.011*"screen" + 0.009*"good" + 0.009*"market" + 0.009*"battery" + 0.008*"great" +
0.007*"get"'),
 (1,
  '0.048*"kindle" + 0.036*"fire" + 0.016*"one" + 0.015*"amazon" + 0.013*"books" + 0.011*"love" + 0.011*"would" + 0.010*"get" + 0.009*"bought" +
0.008*"christmas"'),
 (2,
  '0.009*"use" + 0.009*"usb" + 0.008*"app" + 0.007*"one" + 0.007*"computer" + 0.007*"files" + 0.006*"keyboard" + 0.006*"need" + 0.006*"pc" + 0.006*"laptop"'),
 (3,
  '0.026*"kindle" + 0.020*"amazon" + 0.014*"fire" + 0.011*"problem" + 0.010*"service" + 0.010*"customer" + 0.010*"would" + 0.009*"get" + 0.009*"back" +
0.008*"device"'),
 (4,
  '0.054*"kindle" + 0.045*"fire" + 0.026*"love" + 0.026*"great" + 0.018*"books" + 0.018*"use" + 0.016*"easy" + 0.012*"read" + 0.012*"movies" + 0.011*"games"'),
 (5,
  '0.020*"amazon" + 0.020*"ipad" + 0.019*"device" + 0.015*"fire" + 0.011*"apps" + 0.010*"kindle" + 0.009*"app" + 0.008*"tablet" + 0.007*"like" +
0.007*"android"'),
 (6,
  '0.021*"kindle" + 0.020*"screen" + 0.017*"fire" + 0.013*"reading" + 0.013*"like" + 0.011*"read" + 0.010*"touch" + 0.008*"would" + 0.008*"book" +
0.008*"books"')]
```

We can also see the weight assigned to each of the top ten words in the seven topics.
Based on these, we can see
Topic 0: Great apps, screen, and battery
Topic 1: Kindle as Christmas Gift, Like reading
Topic 2: Hardware related
Topic 3: Bad customer service due to problem
Topic 4: Great use, easy to read books
Topic 5: Like apps on tablet
Topic 6: Great Screen, Like reading on Kindle

B) Do the topics overlap?
Yes, the topics do overlap. Mostly, all Topics have ‘kindle’ as their top word and are about their positive experiences with the screen, battery life, ease of use, apps, and how customers like reading on the tablet or kindle.

C) Calculate the perplexity of each model:
After running the LDA Model for topics 3,5 and 7, we can find how good each model is based on perplexities.



Number of Topics	Perplexity
3	**-7.399**
5	**-7.413**
7	**-7.420**

We can conclude that as the number of topics increases, the perplexity decreases and the predictive performance of the language model keeps improving. Since LDA_Model_7 has the lowest perplexity(compared to models 3 and 5), it is the most effective.

3) Parimus Financial Services (PFS) sells a variety of individual financial products (e.g., life insurance, mutual funds, mortgages) as well as comprehensive financial plans to its customers. It currently has a customer base of 2.1 million accounts generating total revenues of $3.21 billion with a gross margin (profit) of 73 percent. It is estimated that 15 percent of the accounts (comprehensive financial plans buyers) generate 65 percent of the revenues, and the other 85 percent of the accounts (individual product buyers) generate the remaining 35 percent (gross margins are equal across the two groups). PFS has an account retention rate of 93% from year to year.
PFS' chief marketing officer (CMO) is considering the following two strategic options:
-Increase the retention rate from 93% to 96%
-Change the mix of customers from 15% comprehensive plan buyers and 85% individual product buyers to 20% comprehensive plan buyers and 80% individual product buyers.

Calculate the maximum amount of one-time expenditure PFS should be willing to spend right now on each strategic option. Assume a 10% discount rate.
Ans: Based on our calculations(PDF attached after questions),
If we increase the retention rate from 93% to 96%: we need to make a maximum, one-time expenditure of $3.2 Billion.
If we change the mix of customers from 15% comprehensive plan buyers and 85% individual product buyers to 20% comprehensive plan buyers and 80% individual product buyers, we need to make a maximum, one-time expenditure of $2.5 Billion.

4)
Ans: B) ß1, ß2, and ß5 are the coefficients that are identifiable.
5)
Ans: We can interpret that when (k) the Brand(1,2, or 3) changes its Price(attribute) value by 1 percent, it impacts the probabilities of all other brands by the same percentage and the pattern is due to the Independence of Irrelevant Attributes.

Q3. Given:

Customer Base = 2.1 million

Total revenue = $3.24 Billion

margin = 73%.

15% accounts → 65% revenue
85% accounts → remaining 35% of the revenue. } m is equal

retention rate = 93%

$m = \$3.24B \times 73\% = \underline{2.34B}$

Given $i = 10\%$, $r = 95\%$

$$\text{margin multiple} = \frac{r}{1+i-r} = \frac{0.93}{1+0.1-0.93}$$

$$\Rightarrow \frac{0.93}{0.17} \Rightarrow \underline{5.47}$$

$$\therefore CLV = m\left(\frac{r}{1+i-r}\right) - AC$$

$$\Rightarrow (0.73 \times 3.24)(5.47) - AC$$

$$\Rightarrow (2.34B)(5.47) - AC$$

$$\Rightarrow \underline{12.8B - AC}$$

To increase retention rate: 93% - 96%

new $r = 96\%$.

$$\text{new margin multiple} = \frac{0.96}{1+0.1-0.96} = \frac{0.96}{0.14} \Rightarrow \underline{6.85}$$

Let a one-time expenditure = x

$$\therefore \text{new } CLV = (2.34B)(6.85) - AC - x$$

$$\Rightarrow \underline{16B - AC - x}$$

∴ For maximum amount spent

$CLV_{new} - CLV_{old} = 0$

$(16B - AC - x) - (12.8B - AC) = 0$

$x = 3.2B$

∴ Paninus will have spend a maximum amount of one time expenditure worth $3.2B to increase the retention rate from 93% to 96%.

(B) If we change the mix of customers from 15% comprehensive plan buyers
85% individual product buyers
to 20% comprehensive plan buyers
80% individual product buyers:

- 15% : 65% of 3.21B

$\frac{15}{100} \times 2.1 mill$

$\Rightarrow 0.315M$

$\frac{65}{100} \times 3.21B$

$\Rightarrow 2.086B$

$\Rightarrow$ Each comprehensive plan buyer $\Rightarrow \frac{2.086B}{0.315M} = \$6,623.8$

Similarly, remaining
- 85% contribute 35% of revenue

$\Rightarrow$ each individual plan buyer $= \frac{\frac{35}{100} \times 3.21B}{\frac{85}{100} \times 2.1M}$

$\Rightarrow \$629.41$

Now:

20% of 2.1 million customers (comprehensive plan)

↳ $\frac{20}{100} \times 2.1 = 0.42\text{M} \times 6623.8$

$\Rightarrow \underline{\$2.781\text{B}}$

80% of 2.1 million customers (individual plan)

↳ $\frac{80}{100} \times 2.1 \times 629.41 \Rightarrow \underline{\$1.057\text{B}}$

$\therefore$ margin$_C$ $= \$2.781 \times 73\%$ (comprehensive)
$= \$2.03\text{B}$

margin$_I$ $= \$1.057\text{B} \times 73\%$ (individual)
$\Rightarrow 0.771\text{B}$

$\therefore$ Total new margin for ~~total~~ revenue:
$\Rightarrow (2.03 + 0.771) \Rightarrow \underline{\$2.8\text{B}}$

old margin: $\$2.34\text{B}$

$CLV_{new} - CLV_{old} \geq 0$

$\therefore$ For max: $CLV_{new} - CLV_{old} = 0$

$\Rightarrow (2.8 \times 5.471 - \cancel{A} - x) - (2.34 \times 5.471 - \cancel{A}) = 0$

$\Rightarrow$ New $x = (2.8 \times 5.471) - (2.34 \times 5.471)$

$\Rightarrow 5.471(2.8 - 2.34)$

$\approx \underline{2.5\text{B}}$

Q4

(A) Given 3 brands: B_1, B_2 and B_3

with Attributes: Price
Store display $\in (0,1)$

Consumers have characteristics based on demographics
Age, Kids $\in (0,1)$

Let J represent the choice set
Systematic Utility is given by:
Based on Attributes.

$$V_j(x_{ij}) = \beta_1 x_{ij_1} + \beta_2 x_{ij_2} + \dots \beta_k x_{ij_k}$$

Thus, if we consider 3 brands described on 2 attributes over consumers with 2 characteristics we can write individual systematic utility functions as

For Brand 1:

$$V_{iB1} = \beta_{0B1} + \beta_1 \cdot Price_{iB1} + \beta_2 \cdot Storedisplay_{iB1} + \beta_{3B1} Age_i + \beta_{4B1} Kids_i + \beta_5 \cdot Price_{iB1} \cdot Kids_i$$

} for any customer i

Interaction b/w Price & Kids

For Brand 2:

$$V_{iB_2} = \beta_{0B_2} + \beta_1 \cdot Price_{iB2} + \beta_2 + Storedisplay_{iB2} + \beta_{3B2} \cdot Age_i + \beta_{4B2} \cdot Kids_i + \beta_5 \cdot Price_{iB2} \cdot Kids_i$$

for Brand 3

$$V_{iB3} = \beta_{0B3} + \beta_1 \cdot Price_{iB3} + \beta_2 \cdot Store\,display_{iB3} + \beta_3 Age_i + \beta_4 \cdot Kids_i + \beta_5 \cdot Price_{iB3} \cdot Kids_i$$

(B) Assuming Brawll (B1) as base:
we can find the differences b/w utility functions of (B1 & B2) and (B1 & B3)

$V_{iB1} - V_{iB2}$

$\Rightarrow \beta_{0B1} + \beta_1 \text{Price}_{iB1} + \beta_2 \text{Storedisplay}_{iB1} + \beta_{3B1} \text{Age}_i + \beta_{4B1} \text{kids}_i + \beta_5 \text{Price}_{iB1} \text{kids}_i$

$- \left(\beta_{0B2} + \beta_1 \text{Price}_{iB2} + \beta_2 \text{Storedisplay}_{iB2} + \beta_{3B2} \text{Age}_i + \beta_{4B2} \text{kids}_i + \beta_5 \text{Price}_{iB2} \text{kids}_i\right)$

$\Rightarrow (\beta_{0B1} - \beta_{0B2}) + \beta_1 (\text{Price}_{iB1} - \text{Price}_{iB2}) + \beta_2 (\text{Storedisplay}_{iB1} - \text{Storedisplay}_{iB2}) + \text{Age}_i (\beta_{3B1} - \beta_{3B2}) + \text{kids}_i (\beta_{4B1} - \beta_{4B2}) + \beta_5 (\text{Price}_{iB1} \text{kids}_i - \text{Price}_{iB2} \text{kids}_i)$

$\rightarrow$ Thus, we can see that β_1, β_2 and β_5 are identifiable.

Similarly, $V_{iB1} - V_{iB3}$

$\Rightarrow \left[\beta_{0B1} + \beta_1 \text{Price}_{iB1} + \beta_2 \text{Storedisplay}_{iB1} + \beta_{3B1} \text{Age}_i + \beta_{4B1} \text{kids}_i + \beta_5 \text{Price}_{iB1} \text{kids}_i\right]$

$- \left[\beta_{0B3} + \beta_1 \text{Price}_{iB3} + \beta_2 \text{storedisplay}_{B3} + \beta_{3B3} \text{Age}_i + \beta_{4B3} \text{kids}_i + \beta_5 \text{kid}_i \text{Price}_{iB3}\right)$

$\Rightarrow (\beta_{0B1} - \beta_{0B3}) + \beta_1 (\text{Price}_{iB1} - \text{Price}_{iB2}) + \beta_2 (\text{Storedisplay}_{iB1} - \text{Storedisplay}_{iB2}) + \text{Age}_i (\beta_{3B1} - \beta_{3B3}) + \text{kids}_i (\beta_{4B1} - \beta_{4B3}) + \beta_5 (\text{Price}_{iB1} \text{kids}_i - \text{Price}_{iB3} \text{kids}_i)$

Again β_1, β_2 & β_5 are only identifiable

85. Brands [B1, B2, B3]

described by Price Attribute.

price coefficient = -3.1

Given	B1	B2	B3
Brand :			
Price :	1.15	0.95	1.2
Share of Probability :	0.35	0.4	0.25

(a) Own Elasticities of Brands:

$E_{B1} = \beta_m \cdot x_{ijm} (1 - P_{ij})$

$\Rightarrow -3.1 (1.15(1 - 0.35))$

$\Rightarrow -3.1 (1.15 \times 0.65) \Rightarrow \underline{-2.31}$

$E_{B2} = \beta_m \cdot x_{ijm} (1 - P_{ij})$

$\Rightarrow -3.1 (0.95)(1 - 0.40) \Rightarrow -3.1(0.95)(0.60)$

$\Rightarrow \underline{-1.76}$

$E_{B3} = \beta_m x_{ijm} (1 - P_{ij})$

$\Rightarrow -3.1 (1.2)(1 - 0.25) \Rightarrow -3.1 (1.2)(0.75)$

$\Rightarrow \underline{-2.79}$

Cross-Elasticities

⇒ $\frac{\%\text{ change in Probability of } j}{\%\text{ change in attribute } k}$

⇒ $-\beta_m x_{ium} P_{ik}$

(a) $E_{B1 x_{iB2m}}$ (B/w B1 & B2) $= -(-3.1)(0.95)(0.4)$

⇒ 1.178

$E_{B1 x_{iB3m}}$ (B/w B1 & B3) $= -(-3.1)(1.2)(0.25)$

⇒ 0.93

(b) $E_{B2 x_{iB1m}}$ (B/w B2 & B1) $= -(-3.1)(1.15)(0.35)$

⇒ 1.247

$E_{B2 x_{iB3m}}$ (B/w B2 & B3) $= -(-3.1)(1.2)(0.25)$

⇒ 0.93

(c) $E_{B3 x_{iB1m}}$ (B/w B3 & B1) ⇒ $-(-3.1)(1.15)(0.35)$

⇒ 1.247

$E_{B3 x_{iB2m}}$ (B/w B3 & B2) ⇒ $-(-3.1)(0.95)(0.4)$

⇒ 1.178

Brands	B1	B2	B3
B1	-2.31	1.178	0.93
B2	1.247	-1.76	0.93
B3	1.247	1.178	-2.79

We notice a pattern in the table. This is due to Independence of Irrelevant Attributes.

Cross Elasticities are the same for all 'j'

∴ when 'k' → Brand 1, Brand 2 or Brand 3 change its attribute (Price) value by 1%, it impacts the probabilities of all other brands by the SAME percentage.

Text Analysis

```
# import Python packages
import pandas as pd
import numpy as np
from google.colab import files
import matplotlib.pyplot as plt

# regression package
import statsmodels.api as sm

# sentiment analysis packages
import re
import nltk
nltk.download('punkt')
from nltk.tokenize import word_tokenize
from nltk import pos_tag
nltk.download('stopwords')
from nltk.corpus import stopwords
nltk.download('wordnet')
from nltk.corpus import wordnet
from nltk.stem import PorterStemmer
from textblob import TextBlob

# topic modeling packages
import gensim
from gensim import corpora
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
```

Sentiment Analysis

```
# product review data
uploaded = files.upload()
reviews = pd.read_csv('product_reviews.csv')
# take a look at the data
reviews.head(2)
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in

```
Saving product_reviews.csv to product_reviews.csv
```

	Review_ID	Item_ID	Base_item_ID	Review_date	Reviewer_ID	Real_
0	R100E6MT94PK6L	B0051VVOB2	NaN	1/8/2012	A1HGATCAMGXTGF	
1	R100HU42LKLLD0	B0057O9O6K	NaN	4/10/2012	A3GGO95QT2PP47	

```
reviews.shape
```

```
(40741, 58)
```

```
# keep only the review text (Content) and star ratings
reviews_sample = reviews[['Rating','Content']]
```

```
reviews_sample.head(2)
```

	Rating	Content
0	5	I love my fire and highly recommend it to anyo...
1	2	The operating system is is an early android. Y...

Text Preprocessing

```
# remove punctuation and numbers, lower case the text
def clean(text):
    # replace any non-letters with a space
    text = re.sub('[^A-Za-z]+', ' ', text)
    # lower case the text
    text = text.lower()
    return text

# apply the function clean to each review
reviews_sample['Cleaned Reviews'] = reviews_sample['Content'].apply(clean)
reviews_sample.head(2)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:10: SettingWithCopyW
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stab
  # Remove the CWD from sys.path while we load stuff.
```

	Rating	Content	Cleaned Reviews
0	5	I love my fire and highly recommend it to anyo	i love my fire and highly recommend it to anyo

Kindle+Screen

```
# we only want the reviews related to kindle
reviews_sample = reviews_sample[reviews_sample['Cleaned Reviews'].str.contains('kindle
reviews_sample_screen = reviews_sample[reviews_sample['Cleaned Reviews'].str.contains(
```

Kindle+Service

```
# we only want the reviews related to kindle
reviews_sample = reviews_sample[reviews_sample['Cleaned Reviews'].str.contains('kindle
reviews_sample_service = reviews_sample[reviews_sample['Cleaned Reviews'].str.contains
```

Kindle+Weight

```
reviews_sample = reviews_sample[reviews_sample['Cleaned Reviews'].str.contains('kindle
reviews_sample_weight = reviews_sample[reviews_sample['Cleaned Reviews'].str.contains(
```

Kindle+Price

```
reviews_sample = reviews_sample[reviews_sample['Cleaned Reviews'].str.contains('kindle
reviews_sample_price = reviews_sample[reviews_sample['Cleaned Reviews'].str.contains('
```

```
reviews_sample.shape
reviews_sample_screen.shape
reviews_sample_service.shape
reviews_sample_weight.shape
reviews_sample_price.shape
```

```
(3998, 3)
```

```
# tokenize, remove stop words, stem
# we use the Porter stemmer, a process for removing suffixes from words in English
ps = PorterStemmer()

def token_stop_stem(text):
    tokens = word_tokenize(text) # tokenize the text
    newlist = []
    new_review = ""
    for word in tokens:
        if word not in set(stopwords.words('english')): # remove stop words
          word_stem = ps.stem(word) # stem each word
          newlist.append(word_stem)
          new_review = new_review + " " + word_stem
    return new_review

reviews_sample['Final Reviews'] = reviews_sample['Cleaned Reviews'].apply(token_stop_s
reviews_sample.head(2)
```

	Rating	Content	Cleaned Reviews	Final Reviews
1	2	The operating system is is an early android. Y...	the operating system is is an early android yo...	oper system earli android cant use download k...
2	2	I have been reading on	i have been reading on	read kindl sinc kindl releas

```
reviews_sample_screen['Final Reviews'] = reviews_sample_screen['Cleaned Reviews'].appl
reviews_sample_screen.head(2)
```

	Rating	Content	Cleaned Reviews	Final Reviews
9	4	This is a nice little mini computer, but I am ...	this is a nice little mini computer but i am a...	nice littl mini comput littl disappoint bough...
22	4	Kindle Fire, Full Color 7"	kindle fire full color multi	kindl fire full color multi

```
reviews_sample_service['Final Reviews'] = reviews_sample_service['Cleaned Reviews'].ap
reviews_sample_service.head(2)
```

	Rating	Content	Cleaned Revi
242	4	I pre-ordered a Kindle Fire even before the pr...	i pre ordered a kindle fire even before the
328	5	My first two kindles (pre-fire) both died spon...	my first two kindles pre fire both died spor

```
reviews_sample_price['Final Reviews'] = reviews_sample_price['Cleaned Reviews'].apply(
reviews_sample_price.head(2)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyW
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stab
  """Entry point for launching an IPython kernel.
```

	Rating	Content	Cleaned Reviews	Final Reviews
22	4	Kindle Fire, Full Color 7" Multi-touch Display...	kindle fire full color multi touch display wi ...	kindl fire full color multi touch display wi ...
27	5	I bought the Fire for my wife	i bought the fire for my wife	bought fire wife christma

```
reviews_sample_weight['Final Reviews'] = reviews_sample_weight['Cleaned Reviews'].appl
reviews_sample_weight.head(2)
```

	Rating	Content	Cleaned Reviews	Final Reviews
30	5	The Kindle Fire does everything it promised it...	the kindle fire does everything it promised it...	kindl fire everyth promis would opinion much ...
60	5	Everything the average	everything the average	everyth averag person could

Analyzing polarity

```
# we will score the polarity of each review
# polarity ranges from -1 (negative) to 1 (positive)
# under the hood, TextBlob uses a lexicon-based method for scoring
# for details, see https://github.com/sloria/TextBlob/blob/dev/textblob/_text.py
def getPolarityScore(review):
    return TextBlob(review).sentiment.polarity

# function to analyze the reviews
def getPolarity(score):
    if score < 0:
        return 'Negative'
    elif score == 0:
        return 'Neutral'
    else:
        return 'Positive'

reviews_sample['Score'] = reviews_sample['Final Reviews'].apply(getPolarityScore)
reviews_sample['Polarity'] = reviews_sample['Score'].apply(getPolarity)
reviews_sample.head()
```

	Rating	Content	Cleaned Reviews	Final Reviews	Score	Polarity
1	2	The operating system is is an early android. Y...	the operating system is is an early android yo...	oper system earli android cant use download k...	-0.500000	Negative
2	2	I have been reading on Kindle since the Kindle...	i have been reading on kindle since the kindle...	read kindl sinc kindl releas pretti heavili i...	0.067992	Positive
3	5	I bought the Fire because I wanted	i bought the fire because i wanted	bought fire want access app movi	0.211224	Positive

```
reviews_sample_screen['Score'] = reviews_sample_screen['Final Reviews'].apply(getPolar
reviews_sample_screen['Polarity'] = reviews_sample_screen['Score'].apply(getPolarity)
reviews_sample_screen.head()
```

	Rating	Content	Cleaned Reviews	Final Reviews	Score	Polarity
9	4	This is a nice little mini computer, but I am ...	this is a nice little mini computer but i am a...	nice littl mini comput littl disappoint bough...	0.166667	Positive
22	4	Kindle Fire, Full Color 7" Multi-touch Display...	kindle fire full color multi touch display wi ...	kindl fire full color multi touch display wi ...	0.444444	Positive
23	4	The Kindle Fire	the kindle fire does	kindl fire everyth		

```
reviews_sample_service['Score'] = reviews_sample_service['Final Reviews'].apply(getPol
reviews_sample_service['Polarity'] = reviews_sample_service['Score'].apply(getPolarity
reviews_sample_service.head()
```

	Rating	Content	Cleaned Reviews	Final Reviews	Score	Polarity
242	4	I pre-ordered a Kindle Fire even before the pr...	i pre ordered a kindle fire even before the pr...	pre order kindl fire even product st appear m...	0.023512	Positive
328	5	My first two kindles (pre-fire) both died spon...	my first two kindles pre fire both died sponta...	first two kindl pre fire die spontan without ...	0.050000	Positive
329	5	I had a problem with	i had a problem	problem st kindl		

```
reviews_sample_weight['Score'] = reviews_sample_weight['Final Reviews'].apply(getPolar
reviews_sample_weight['Polarity'] = reviews_sample_weight['Score'].apply(getPolarity)
reviews_sample_weight.head()
```

	Rating	Content	Cleaned Reviews	Final Reviews	Score	Polarity
30	5	The Kindle Fire does everything it promised it...	the kindle fire does everything it promised it...	kindl fire everyth promis would opinion much ...	0.191304	Positive

```
reviews_sample_price['Score'] = reviews_sample_price['Final Reviews'].apply(getPolarit
reviews_sample_price['Polarity'] = reviews_sample_price['Score'].apply(getPolarity)
reviews_sample_price.head()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyW
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stab
  """Entry point for launching an IPython kernel.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyW
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stab
```

	Rating	Content	Cleaned Reviews	Final Reviews	Score	Polarity
22	4	Kindle Fire, Full Color 7" Multi-touch Display...	kindle fire full color multi touch display wi ...	kindl fire full color multi touch display wi ...	0.444444	Positive
27	5	I bought the Fire for my wife for Christmas, k...	i bought the fire for my wife for christmas kn...	bought fire wife christma know match ipad use...	0.250000	Positive
29	4	The Kindle Fire is a great price. getting	the kindle fire is a great price getting	kindl fire great price get onlin faster	0.550000	Positive

```
# does the polarity correlate with the star rating?
reviews_sample_reg = sm.add_constant(reviews_sample)

# run regression
ols = sm.OLS(reviews_sample_reg['Rating'],reviews_sample_reg['Score'])
ols_result = ols.fit()
print(ols_result.summary())
```

```
                                 OLS Regression Results
==============================================================================
Dep. Variable:                 Rating   R-squared (uncentered):
Model:                            OLS   Adj. R-squared (uncentered):
Method:                 Least Squares   F-statistic:                       2.
```

```
Date:                Sat, 15 Oct 2022   Prob (F-statistic):
Time:                        14:55:29   Log-Likelihood:                 -
No. Observations:               16381   AIC:                            7.(
Df Residuals:                   16380   BIC:                            7.(
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Score         11.1714      0.066    169.561      0.000      11.042      11.301
==============================================================================
Omnibus:                      694.074   Durbin-Watson:                   1.361
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2258.261
Skew:                          -0.073   Prob(JB):                         0.00
Kurtosis:                       4.813   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not conta
[2] Standard Errors assume that the covariance matrix of the errors is correctly
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWa
  x = pd.concat(x[::order], 1)
```

```
reviews_sample_screen_reg = sm.add_constant(reviews_sample_screen)

# run regression
ols = sm.OLS(reviews_sample_screen_reg['Rating'],reviews_sample_screen_reg['Score'])
ols_result = ols.fit()
print(ols_result.summary())
```

```
                                 OLS Regression Results
==============================================================================
Dep. Variable:                 Rating   R-squared (uncentered):
Model:                            OLS   Adj. R-squared (uncentered):
Method:                 Least Squares   F-statistic:                      1.4
Date:                Sat, 15 Oct 2022   Prob (F-statistic):
Time:                        14:56:01   Log-Likelihood:                 -
No. Observations:                5806   AIC:                            2.5
Df Residuals:                    5805   BIC:                            2.5
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Score         13.4167      0.113    118.672      0.000      13.195      13.638
==============================================================================
Omnibus:                      264.539   Durbin-Watson:                   1.525
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              753.999
Skew:                          -0.194   Prob(JB):                    1.87e-164
Kurtosis:                       4.722   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not conta
[2] Standard Errors assume that the covariance matrix of the errors is correctly
```

```
reviews_sample_service_reg = sm.add_constant(reviews_sample_service)

# run regression
ols = sm.OLS(reviews_sample_service_reg['Rating'],reviews_sample_service_reg['Score'])
ols_result = ols.fit()
print(ols_result.summary())
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                  Rating   R-squared (uncentered):
Model:                             OLS   Adj. R-squared (uncentered):
Method:                  Least Squares   F-statistic:
Date:                 Sat, 15 Oct 2022   Prob (F-statistic):                    1.(
Time:                         14:56:21   Log-Likelihood:                         -
No. Observations:                  603   AIC:
Df Residuals:                      602   BIC:
Df Model:                            1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Score         11.0977      0.380     29.213      0.000      10.352      11.844
==============================================================================
Omnibus:                       21.041   Durbin-Watson:                   1.421
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               38.751
Skew:                          -0.216   Prob(JB):                     3.85e-09
Kurtosis:                       4.164   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not conta
[2] Standard Errors assume that the covariance matrix of the errors is correctly
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWa
  x = pd.concat(x[::order], 1)
```

```
reviews_sample_weight_reg = sm.add_constant(reviews_sample_weight)

# run regression
ols = sm.OLS(reviews_sample_weight_reg['Rating'],reviews_sample_weight_reg['Score'])
ols_result = ols.fit()
print(ols_result.summary())
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                  Rating   R-squared (uncentered):
Model:                             OLS   Adj. R-squared (uncentered):
Method:                  Least Squares   F-statistic:
Date:                 Sat, 15 Oct 2022   Prob (F-statistic):                    3.(
Time:                         14:56:25   Log-Likelihood:                         -
No. Observations:                  931   AIC:
Df Residuals:                      930   BIC:
Df Model:                            1
```

```
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Score         14.3438      0.279     51.432      0.000      13.796      14.891
==============================================================================
Omnibus:                       75.660   Durbin-Watson:                   1.608
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              201.372
Skew:                          -0.424   Prob(JB):                     1.87e-44
Kurtosis:                       5.114   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not conta
[2] Standard Errors assume that the covariance matrix of the errors is correctly
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWa
  x = pd.concat(x[::order], 1)
```

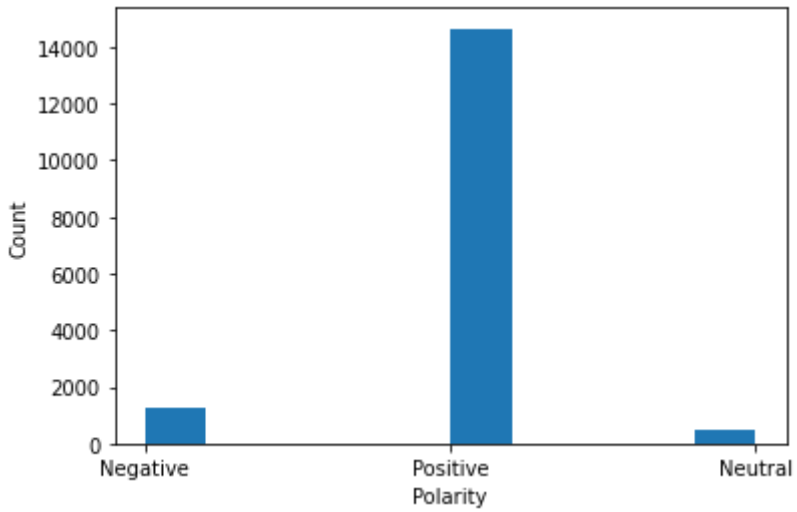
```
reviews_sample_price_reg = sm.add_constant(reviews_sample_price)

# run regression
ols = sm.OLS(reviews_sample_price_reg['Rating'],reviews_sample_price_reg['Score'])
ols_result = ols.fit()
print(ols_result.summary())
```

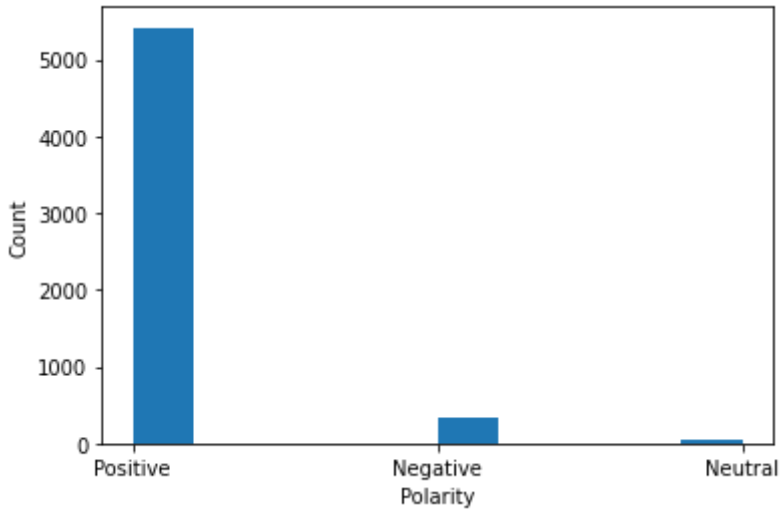
```
                                 OLS Regression Results
==============================================================================
Dep. Variable:                 Rating   R-squared (uncentered):
Model:                            OLS   Adj. R-squared (uncentered):
Method:                 Least Squares   F-statistic:                      1.
Date:                Sat, 15 Oct 2022   Prob (F-statistic):
Time:                        14:56:27   Log-Likelihood:                   -
No. Observations:                3998   AIC:                              1.
Df Residuals:                    3997   BIC:                              1.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Score         12.7722      0.126    101.291      0.000      12.525      13.019
==============================================================================
Omnibus:                      302.017   Durbin-Watson:                   1.551
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1055.621
Skew:                          -0.332   Prob(JB):                    5.95e-230
Kurtosis:                       5.428   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not conta
[2] Standard Errors assume that the covariance matrix of the errors is correctly
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWa
  x = pd.concat(x[::order], 1)
```

```
# plot the distribution of polarities
plt.hist(reviews_sample['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```



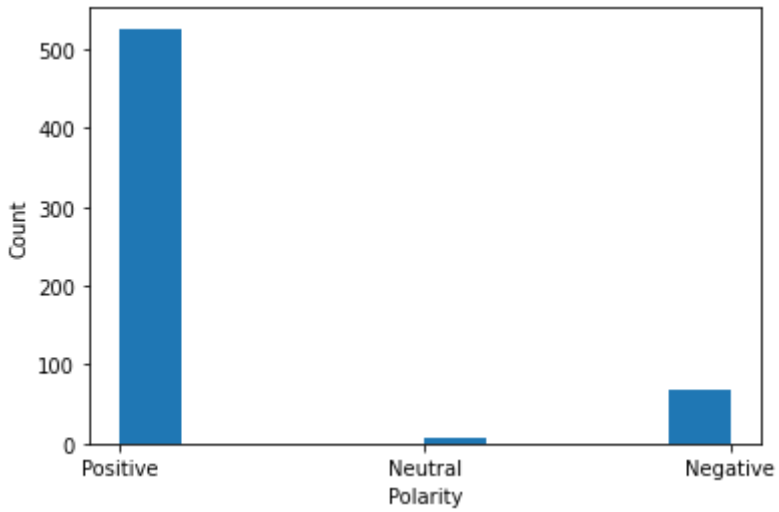

```
plt.hist(reviews_sample_screen['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```




```
reviews_sample_screen['Polarity'].value_counts()
```

```
Positive    5417
Negative     343
Neutral       46
Name: Polarity, dtype: int64
```

```
plt.hist(reviews_sample_service['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```




```
reviews_sample_service['Polarity'].value_counts()
```

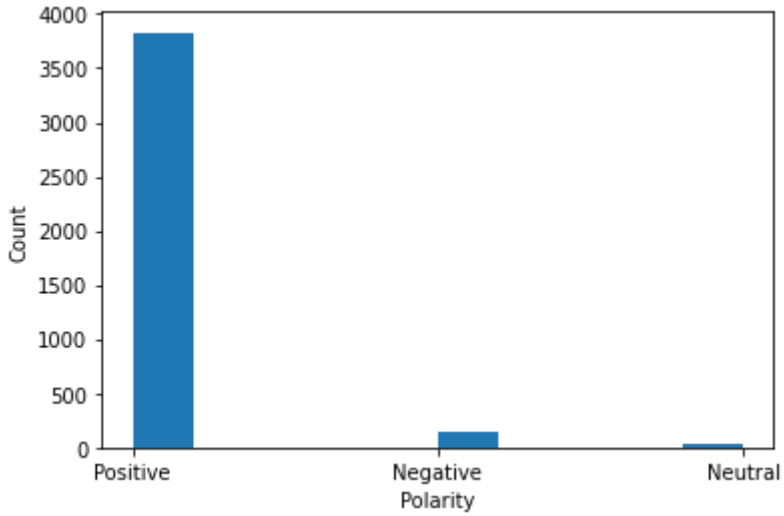
```
Positive    526
Negative     69
Neutral       8
Name: Polarity, dtype: int64
```

```
plt.hist(reviews_sample_weight['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```

```
reviews_sample_weight['Polarity'].value_counts()
```

```
Positive    893
Negative     35
Neutral       3
Name: Polarity, dtype: int64
```

```
plt.hist(reviews_sample_price['Polarity'])
plt.xlabel('Polarity')
plt.ylabel('Count')
plt.show()
```




```
reviews_sample_price['Polarity'].value_counts()
```

```
Positive    3828
Negative     143
Neutral       27
Name: Polarity, dtype: int64
```

Topic Modeling

```
# tokenize, remove stop words, return tokens

def token_stop(text):
    tokens = word_tokenize(text) # tokenize the text
    newlist = []
    for word in tokens:
        if word not in set(stopwords.words('english')): # remove stop words
```

```
            newlist.append(word)
    return newlist

reviews_sample['LDA Reviews'] = reviews_sample['Cleaned Reviews'].apply(token_stop)
reviews_sample.head(2)
```

	Rating	Content	Cleaned Reviews	Final Reviews	Score	Polarity	LDA Reviews
1	2	The operating system is is an early android. Y...	the operating system is is an early android yo...	oper system earli android cant use download k...	-0.500000	Negative	[operating, system, early, android, cant,

```
dict_ = corpora.Dictionary(reviews_sample['LDA Reviews'])
print(dict_)
```

```
    Dictionary(28518 unique tokens: ['android', 'barns', 'cant', 'com', 'download'].
```

```
# convert list of documents (corpus) into Document Term Matrix using the dictionary
doc_term_matrix = [dict_.doc2bow(i) for i in reviews_sample['LDA Reviews']]


# doc_term_matrix


num_topics = 3
chunksize = 1000
passes = 20
iterations = 400
eval_every = None  # don't evaluate model perplexity, takes too much time

lda = gensim.models.LdaModel

lda_model_3 = lda(doc_term_matrix,
    num_topics=num_topics,
    id2word = dict_,
    iterations=iterations,
    passes=passes,
    eval_every=eval_every,
    random_state=9651
)


# set training parameters
num_topics = 5
chunksize = 1000
passes = 20
iterations = 400
eval_every = None  # don't evaluate model perplexity, takes too much time
```

```
lda = gensim.models.LdaModel

lda_model_5 = lda(doc_term_matrix,
    num_topics=num_topics,
    id2word = dict_,
    iterations=iterations,
    passes=passes,
    eval_every=eval_every,
    random_state=9651
)



num_topics = 7
chunksize = 1000
passes = 20
iterations = 400
eval_every = None  # don't evaluate model perplexity, takes too much time

lda = gensim.models.LdaModel

lda_model_7 = lda(doc_term_matrix,
    num_topics=num_topics,
    id2word = dict_,
    iterations=iterations,
    passes=passes,
    eval_every=eval_every,
    random_state=9651
)


lda_model_3.print_topics(num_words=10)

     [(0,
       '0.038*"kindle" + 0.037*"fire" + 0.014*"books" + 0.012*"ipad" + 0.011*"great"
     + 0.011*"love" + 0.011*"read" + 0.010*"reading" + 0.010*"use" + 0.010*"like"'),
      (1,
       '0.039*"kindle" + 0.028*"fire" + 0.017*"amazon" + 0.012*"one" + 0.010*"would"
     + 0.009*"get" + 0.006*"christmas" + 0.006*"got" + 0.006*"bought" +
     0.005*"time"'),
      (2,
       '0.014*"tablet" + 0.010*"device" + 0.010*"android" + 0.009*"app" +
     0.009*"apps" + 0.008*"screen" + 0.007*"ipad" + 0.007*"amazon" + 0.007*"like" +
     0.006*"use"')]


lda_model_5.print_topics(num_words=10)

     [(0,
       '0.025*"screen" + 0.012*"kindle" + 0.012*"reading" + 0.012*"battery" +
     0.011*"touch" + 0.010*"use" + 0.009*"like" + 0.008*"read" + 0.008*"life" +
     0.008*"e"'),
      (1,
       '0.050*"kindle" + 0.046*"fire" + 0.017*"books" + 0.015*"love" + 0.015*"amazon"
```

```
      + 0.011*"read" + 0.010*"one" + 0.010*"would" + 0.010*"use" + 0.009*"great"'),
       (2,
        '0.015*"tablet" + 0.013*"android" + 0.012*"device" + 0.012*"app" +
      0.012*"apps" + 0.010*"amazon" + 0.007*"like" + 0.006*"screen" + 0.005*"use" +
      0.005*"one"'),
       (3,
        '0.030*"kindle" + 0.019*"fire" + 0.018*"amazon" + 0.011*"one" + 0.010*"get" +
      0.010*"would" + 0.007*"problem" + 0.007*"back" + 0.006*"customer" +
      0.006*"service"'),
       (4,
        '0.067*"ipad" + 0.023*"fire" + 0.023*"kindle" + 0.015*"great" + 0.014*"tablet"
      + 0.013*"price" + 0.011*"apple" + 0.009*"device" + 0.009*"screen" +
      0.008*"good"')]
```

```
lda_model_7.print_topics(num_words=10)
```

```
      [(0,
        '0.040*"tablet" + 0.021*"android" + 0.012*"apps" + 0.012*"app" +
      0.011*"screen" + 0.009*"good" + 0.009*"market" + 0.009*"battery" + 0.008*"great"
      + 0.007*"get"'),
       (1,
        '0.048*"kindle" + 0.036*"fire" + 0.016*"one" + 0.015*"amazon" + 0.013*"books"
      + 0.011*"love" + 0.011*"would" + 0.010*"get" + 0.009*"bought" +
      0.008*"christmas"'),
       (2,
        '0.009*"use" + 0.009*"usb" + 0.008*"app" + 0.007*"one" + 0.007*"computer" +
      0.007*"files" + 0.006*"keyboard" + 0.006*"need" + 0.006*"pc" + 0.006*"laptop"'),
       (3,
        '0.026*"kindle" + 0.020*"amazon" + 0.014*"fire" + 0.011*"problem" +
      0.010*"service" + 0.010*"customer" + 0.010*"would" + 0.009*"get" + 0.009*"back"
      + 0.008*"device"'),
       (4,
        '0.054*"kindle" + 0.045*"fire" + 0.026*"love" + 0.026*"great" + 0.018*"books"
      + 0.018*"use" + 0.016*"easy" + 0.012*"read" + 0.012*"movies" + 0.011*"games"'),
       (5,
        '0.020*"amazon" + 0.020*"ipad" + 0.019*"device" + 0.015*"fire" + 0.011*"apps"
      + 0.010*"kindle" + 0.009*"app" + 0.008*"tablet" + 0.007*"like" +
      0.007*"android"'),
       (6,
        '0.021*"kindle" + 0.020*"screen" + 0.017*"fire" + 0.013*"reading" +
      0.013*"like" + 0.011*"read" + 0.010*"touch" + 0.008*"would" + 0.008*"book" +
      0.008*"books"')]
```

```
count = 0
for i in range(5):
    print("doc : ",count,lda_model_3[doc_term_matrix][i])
    count += 1
```

```
     doc :  0 [(0, 0.020009162), (1, 0.38637593), (2, 0.5936149)]
     doc :  1 [(0, 0.6361592), (1, 0.04456405), (2, 0.3192767)]
     doc :  2 [(0, 0.7704586), (1, 0.17863332), (2, 0.050908145)]
     doc :  3 [(0, 0.73829776), (2, 0.25468963)]
     doc :  4 [(0, 0.9054254), (1, 0.016063394), (2, 0.07851113)]
```

```
count = 0
for i in range(5):
    print("doc : ",count,lda_model_5[doc_term_matrix][i])
    count += 1
```

```
doc :  0 [(0, 0.010685884), (1, 0.010917938), (2, 0.5897765), (3, 0.3778801), (4
doc :  1 [(0, 0.4109323), (1, 0.117191955), (2, 0.20412049), (3, 0.12667385), (4
doc :  2 [(0, 0.052070197), (1, 0.8619343), (2, 0.055994958), (4, 0.028720334)]
doc :  3 [(1, 0.6013006), (2, 0.22636475), (4, 0.1663838)]
doc :  4 [(0, 0.35545313), (1, 0.4150662), (2, 0.1651335), (4, 0.05668152)]
```

```
count = 0
for i in range(5):
    print("doc : ",count,lda_model_7[doc_term_matrix][i])
    count += 1
```

```
doc :  0 [(0, 0.42677534), (1, 0.2906102), (2, 0.14822535), (3, 0.111780144)]
doc :  1 [(3, 0.048466038), (5, 0.4408344), (6, 0.50742257)]
doc :  2 [(0, 0.022721397), (1, 0.42841393), (3, 0.04202307), (4, 0.36525804), (
doc :  3 [(0, 0.2642299), (1, 0.18879099), (4, 0.45317417), (6, 0.08757336)]
doc :  4 [(4, 0.3821926), (6, 0.59120274)]
```

```
reviews_sample["Content"].iloc[1]
```

```
'I have been reading on Kindle since the Kindle 2 was released. I am pretty heav
ily invested in Kindle at this point so the Fire was a logical upgrade...or so I
thought. It is a very nice device that is made unuseable, in my opinion, by seve
ral flaws. This thing is heavy. Much heavier than it looks. The major problem he
re is it is difficult to hold. The size and thin bezel just aggravate the situat
ion. The iPad is heavier, but it is bigger so I can lay down, rest my elbows on
the bed and hold it comfortably over my chest. I cant get into a comfortable pos
ition with the Fire. It is not easy to hold with one hand without accidental tou
```

```
reviews_sample_screen["Content"].iloc[1]
```

```
'Kindle Fire, Full Color 7" Multi-touch Display, Wi-Fi I AM WELL PLEASED WITH MY
KINDLE FIRE PURCHASE. I GOT IT MAINLY FOR READING AND WAS WARY BEFORE I GOT IT,
BECAUSE I WASN\'T SURE I WOULD ENJOY READING IN THIS FORMAT. I HAVE BEEN PLEASAN
TLY SURPRISED. ONCE ONE GETS USED TO THE TOUCH SCREEN; SUCH FEATURES AS BOOKMARK
ING, HIGHLIGHTING, AND ADDING NOTES BECOME INTUITIVE AND ARE VERY USEFUL FEATURE
S. MAGNIFYING IMAGES REQUIRES ONLY A LIGHT DOUBLE TAP. THE SELECTION OF READING
MATERIAL IS IMMENSE AND MUCH MORE REASONABLY PRICED THAN PRINT VERSIONS. THE REA
SONS I DIDN\'T GIVE IT 5 STARS: 1) I READ MOSTLY NON-FICTION, AND THE INDEXES AR
```

```
reviews_sample_service["Content"].iloc[1]
```

```
'My first two kindles (pre-fire) both died spontaneously without any damage incu
rred on them. The kindle fire on several occassions has double charged me. The w
ebsite says that there is a 7 day return policy, but refuse to return any purcha
ses that I made. The audio quality of audio downloads is extremely poor quality,
as are the speakers. The kindle fire dies after about 2 hours of use. Customer s
ervice is mostly red tape as they transfer you to a "different department" that
```

```
reviews_sample_weight["Content"].iloc[1]
```

```
'Everything the average person could want in a $200 tablet device. I bought this
because: 1. It is Amazon, which is a very customer-oriented company; 2. The 7in
display and 14ounce weight make this very portable and usable all day long; 3. T
he software interface is very easy to use and well thought out; 4. this is my fi
rst tablet and did not feel comfortable with the iPAD price, size, weight, and s
ophistication; 5. I wanted a Kindle for ebooks and office PDFs, and it is very e
asy to use for these specific uses, if not for anything else. I prefer the touc
h/tap interface to constantly having to press buttons to scroll; 6. This device
```

```
reviews_sample_price["Content"].iloc[1]
```

```
'I bought the Fire for my wife for Christmas, knowing it was not a match for the
Ipad 2. I use the Ipad for just about everything (newspaper, books, email, etc.)
but she thought she would primarily use the Fire for reading (She already had th
e second generation Kindle). Boy, was she surprised! She uses it for just about
```

```
# calculate model perplexity
lda_model_3.log_perplexity(doc_term_matrix)
```

```
-7.399004294077214
```

```
lda_model_5.log_perplexity(doc_term_matrix)
```

```
-7.41318957765125
```

```
lda_model_7.log_perplexity(doc_term_matrix)
```

```
-7.420613120084899
```

```
lda_model_10 = lda(doc_term_matrix,
    num_topics=10,
    id2word = dict_,
    iterations=iterations,
    passes=passes,
    eval_every=eval_every,
    random_state=9651
)
```

```
lda_model_10.log_perplexity(doc_term_matrix)
```

```
-7.426654494119423
```

```
lda_model_20 = lda(doc_term_matrix,
    num_topics=20,
    id2word = dict_,
    iterations=iterations,
    passes=passes,
    eval_every=eval_every,
```

```
    random_state=9651
)

lda_model_20.log_perplexity(doc_term_matrix)
```

```
-7.461322326107518
```