Customer Sentiment Analysis

Post Graduate Program in Data Science Engineering
Location: **Hyderabad** Batch: **PGPDSE-Nov 23**

Submitted by

Udaya Ediga
K Neeraj Kumar
Suveg Nimje
Gaddam Venkata Rohith
Kadali J S V Satya Deva

Mentored by

Trupti Mathapati

Table of Contents

INTRODUCTION:

When we want to buy something, "What other people think" has always been a significant factor in our decision-making process for most of us. Customers share their thoughts and opinions on a variety of products and services. Customers, however, are unable to read all evaluations due to the large number of them. A lot of research is being done in Sentiment Analysis to answer this problem. Sentiment Analysis is an approach to classifying the sentiments of user reviews, documents, and other sources in terms of positive (good), negative (poor), or neutral (surprise). Most sentiment analysis methods nowadays, provide an overall polarity of the text. However, for fine-grained analysis, it is desirable to comprehend the mood of each aspect of various entities. As a result, we propose a system that uses supervised learning to classify Women's Clothing E-Commerce Reviews into three categories: positive, negative, and neutral.

Dataset Information:

Welcome. This is a Women's Clothing E-Commerce dataset revolving around the reviews written by customers. Its nine supportive features offer a great environment to parse out the text through its multiple dimensions. Because this is real commercial data, it has been anonymized, and references to the company in the review text and body have been replaced with "retailer".

This dataset includes 23486 rows and 10 feature variables. Each row corresponds to a customer review, and includes the variables:

Clothing ID: Integer Categorical variable that refers to the specific piece being reviewed.
Age: Positive Integer variable of the reviewers age.
Title: String variable for the title of the review.
Review Text: String variable for the review body.
Rating: Positive Ordinal Integer variable for the product score granted by the customer from 1 Worst, to 5 Best.
Recommended IND: Binary variable stating where the customer recommends the product where 1 is recommended, 0 is not recommended.
Positive Feedback Count: Positive Integer documenting the number of other customers who found this review positive.
Division Name: Categorical name of the product high level division.
Department Name: Categorical name of the product department name.
Class Name: Categorical name of the product class name.

Sentiment: Categorical name that refers to the Sentiment of the customer.(Target Variable, Feature Engineering)

Problem Statement:

The analysis specifically targets reviews or comments related to women's clothing items sold through an E-Commerce platform. The primary objective is to analyze the sentiment of reviews or comments related to women's e-commerce clothing. This involves classifying text data into categories such as positive, negative, or neutral sentiments. The sentiment analysis results can be used to gain insights into customer opinions, identify areas for improvement in products or services, and tailor marketing strategies accordingly.

Variable Categorization with Description:

Categorical:

Variable	Datatype	Description
Title	Object	String variable for the title of the review.
Review Text	Object	String variable for the review body.
Division Name	Object	Categorical name of the product high level division.
Department Name	Object	Categorical name of the product department name.
Class Name	Object	Categorical name of the product class name
Sentiment	Object	Categorical name that refers to the Sentiment of the customer.

Numerical:

Variable	Datatype	Description
Clothing ID	Integer	Integer Categorical variable that refers to the specific piece being reviewed.
Age	Integer	Positive Integer variable of the reviewers age.
Recommended IND	Integer	Binary variable stating where the customer recommends the product where 1 is recommended, 0 is not recommended.
Positive Feedback Count	Integer	Positive Integer documenting the number of other customers who found this review positive.
Rating	Integer	Positive Ordinal Integer variable for the product score granted by the customer from 1 Worst, to 5 Best.

Target Variable:

We added a new categorical feature named “Sentiment”. It is added using the Rating feature. The column takes values as Positive, Neutral, and Negative. Positive is given to the records with a

rating greater than 3, Neutral is given to the records with a rating equal to 3 and Negative is given to the records with a rating less than 3.

Data Preprocessing:

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always the case that we come across clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put it in a formatted way. So for this, we use data pre-processing tasks.

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data pre-processing is required for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

Missing Value Treatment:

The next step of data pre-processing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

Missing Value Percentages:

Attribute	Null Value Percentage
Age	**0.000000**
Clothing ID	**0.000000**
Review Text	**3.597888**
Rating	**0.000000**
Recommended IND	**0.000000**
Positive Feedback Count	**0.000000**
Division Name	**0.059610**
Department Name	**0.059610**
Class Name	**0.059610**

Title	**16.222430**

We are not dropping any null values in the data. We are using only the review text column as an independent variable and the sentiment column as the target variable.

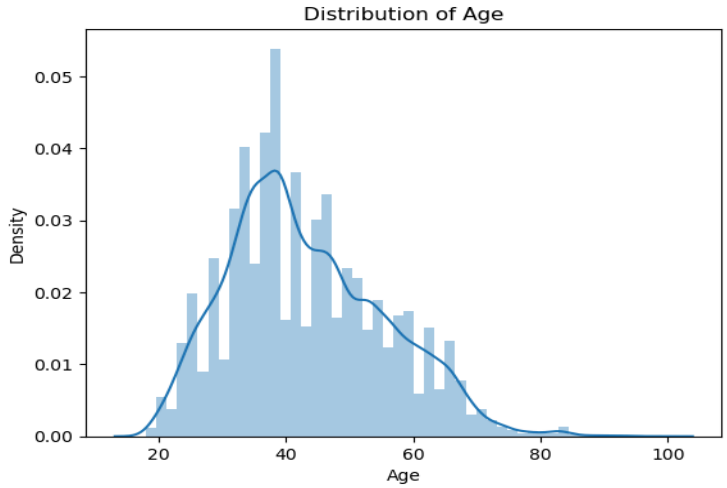
Check for Outliers:

Data has outliers present in each of the numerical columns. For making the base model, We do not perform any outlier treatment and retain all the rows present in the data.

EXPLORATORY DATA ANALYSIS:

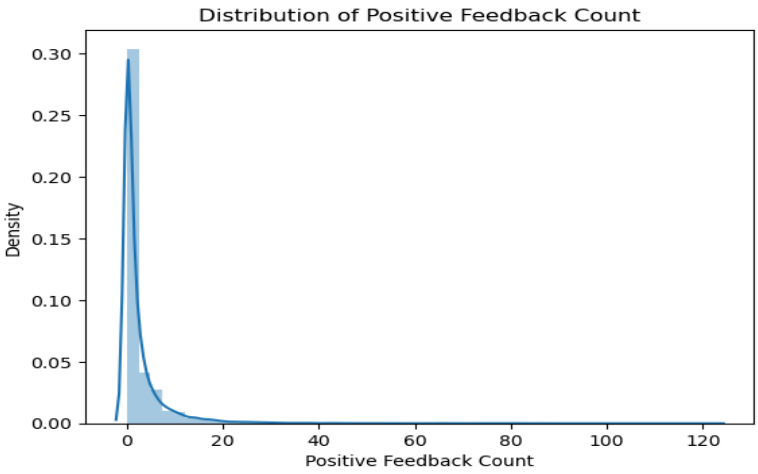
Univariate Analysis:

1. **Age:**



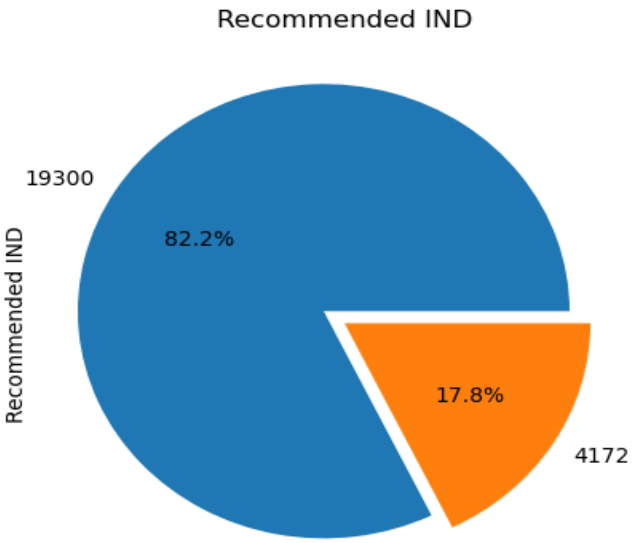
- The age distribution in the dataset spans from 20 to 70 years old, indicating a broad range of ages represented in the data.

2. **Positive Feedback count:**



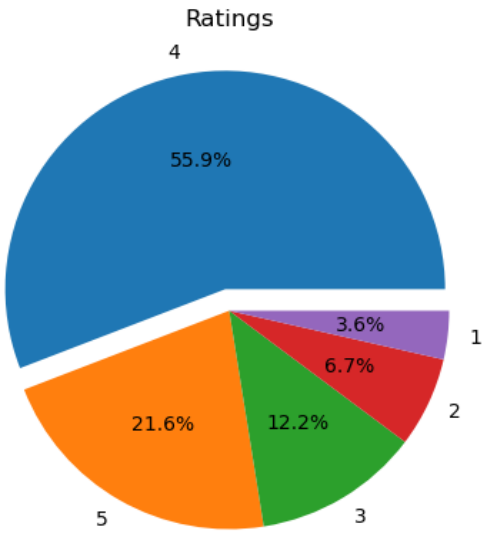
- In this plot we can see that we have peek in between 2-5 counts

3. Recommended IND:



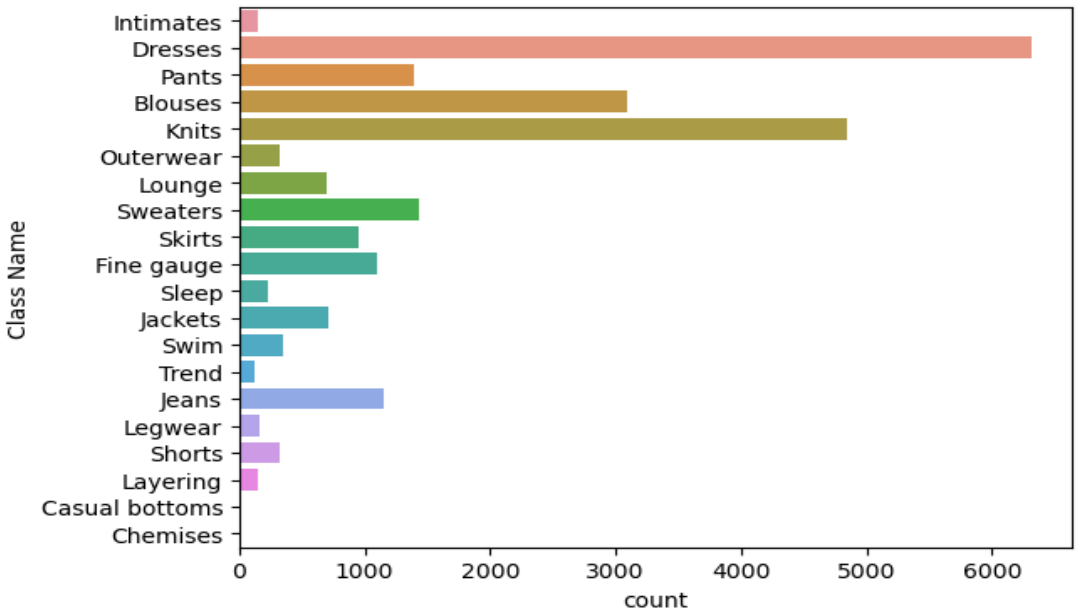
- 82.2% of the customers recommended .
- 17.9% of the customers did not recommend.

4. Ratings:



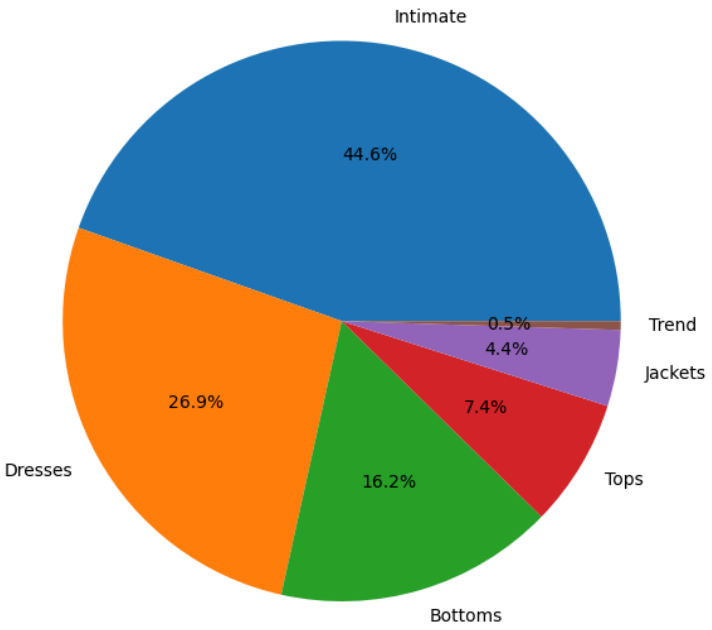
- We can see that almost 56% of the products are having 4 stars rating and 21.6% are having 5 star rating.

5. Class Name:



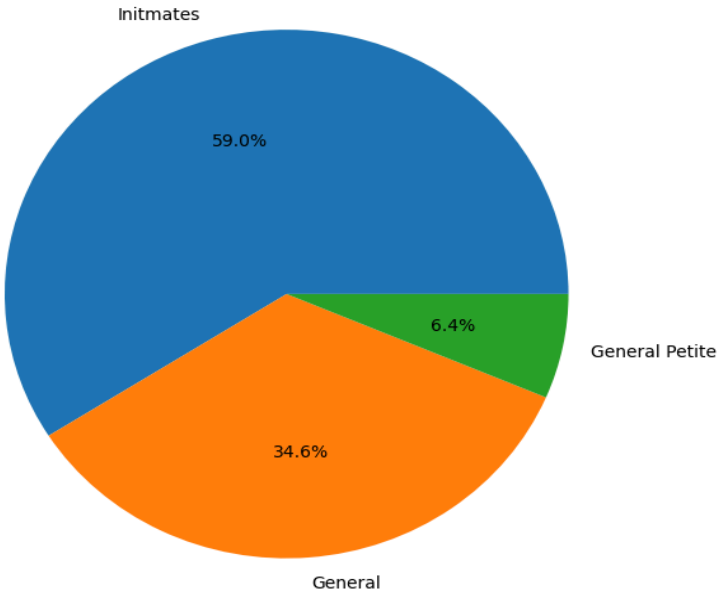
- We can see that Dresses, Knits, Blouses classes are dominating the most sales of 6319,4843,3097 respectively.

6. Department Name:



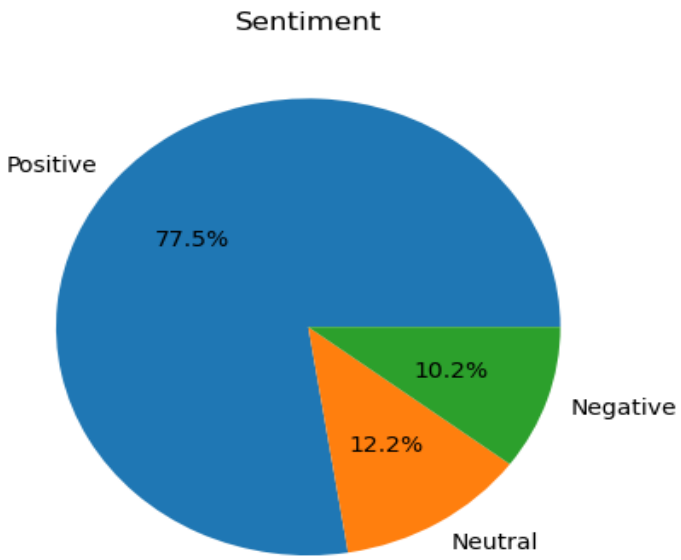
- In Departments we can see that Intimate and Dresses are dominating the most sales of 44.6% and 26.9% respectively.

7. Division Name:



- Intimate Division have the most sales(59%) and General Division have the second most sales(34.6%)

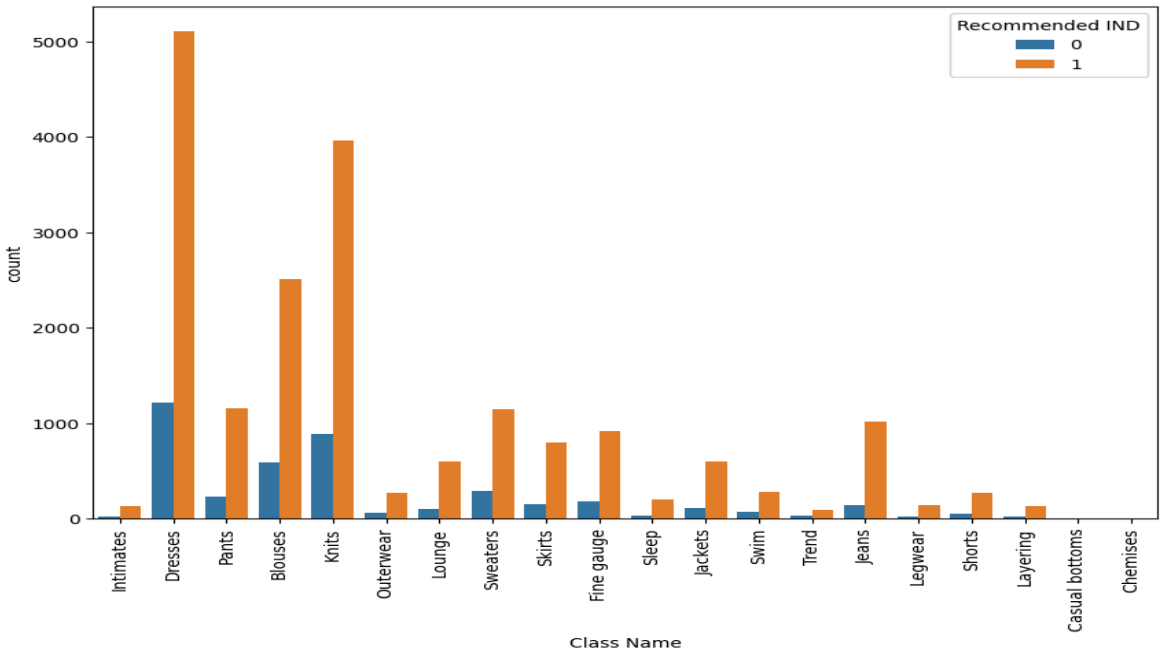
8. Sentiment:



- From this Feature Engineered we can see that 77.5% of the products have received positive reviews that tells us about the quality of the products.

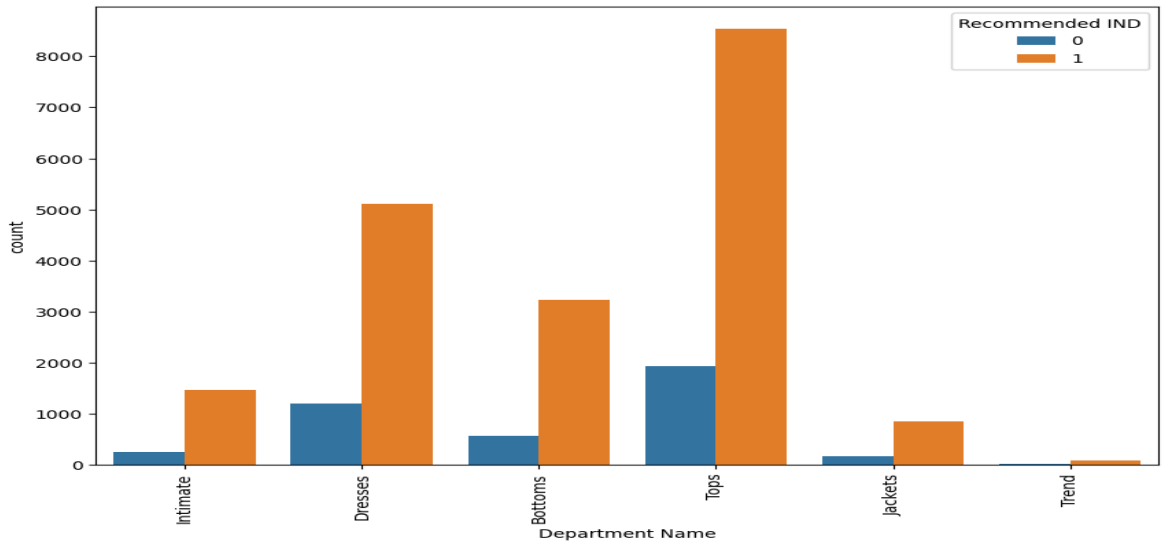
Bi-Variate Analysis:

1. Class Name Vs Recommended IND:



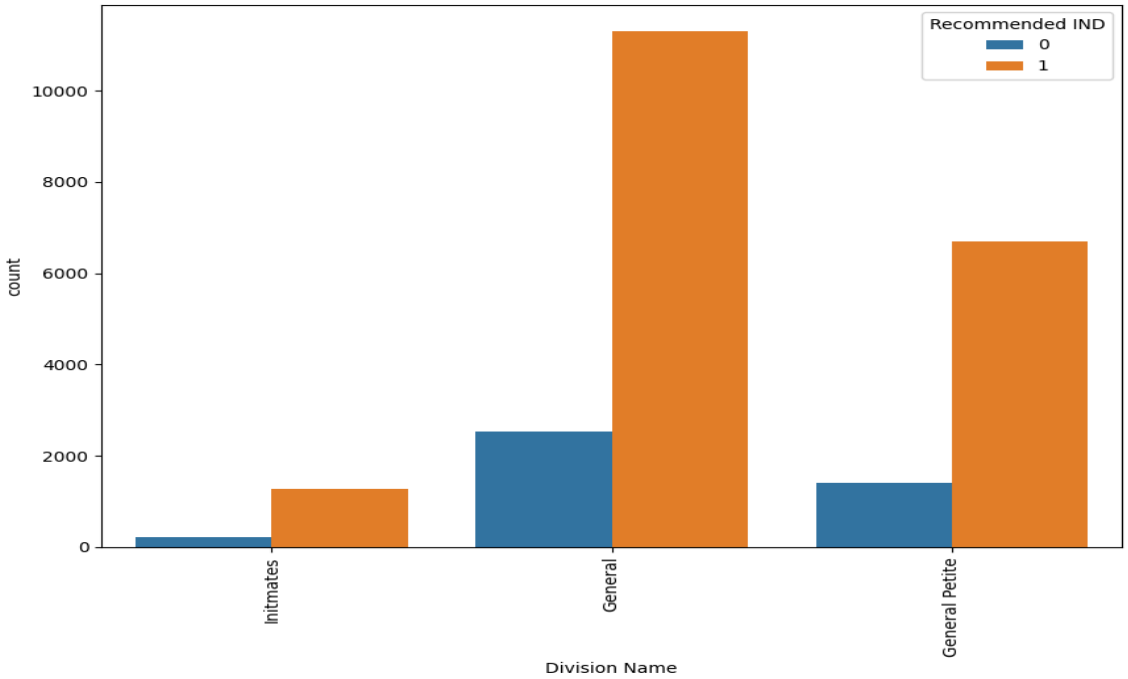
- From this plot we can see that Dresses are most recommended and Intimates are least recommended in Classes

2. Department Name Vs Recommended IND:



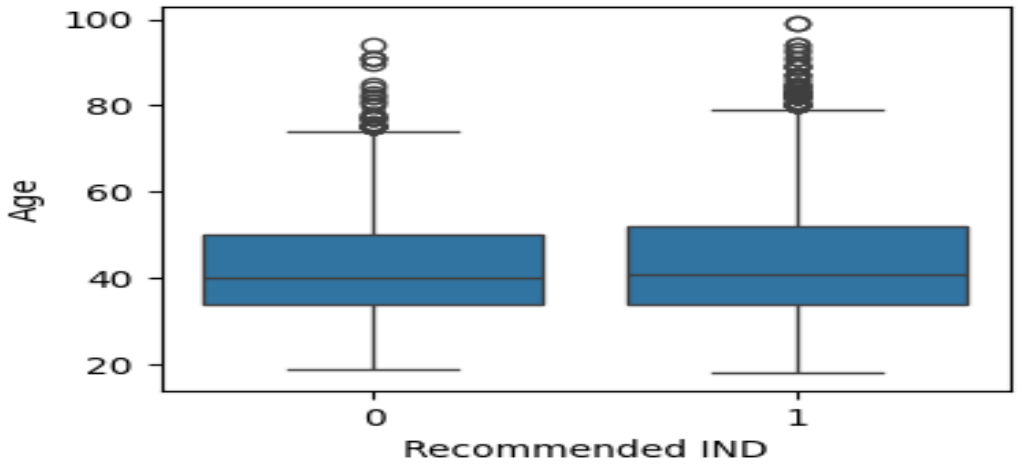
- From this plot we can see that Tops is most recommended and Trends are least recommended in Department.

3. Division Name Vs Recommended IND:



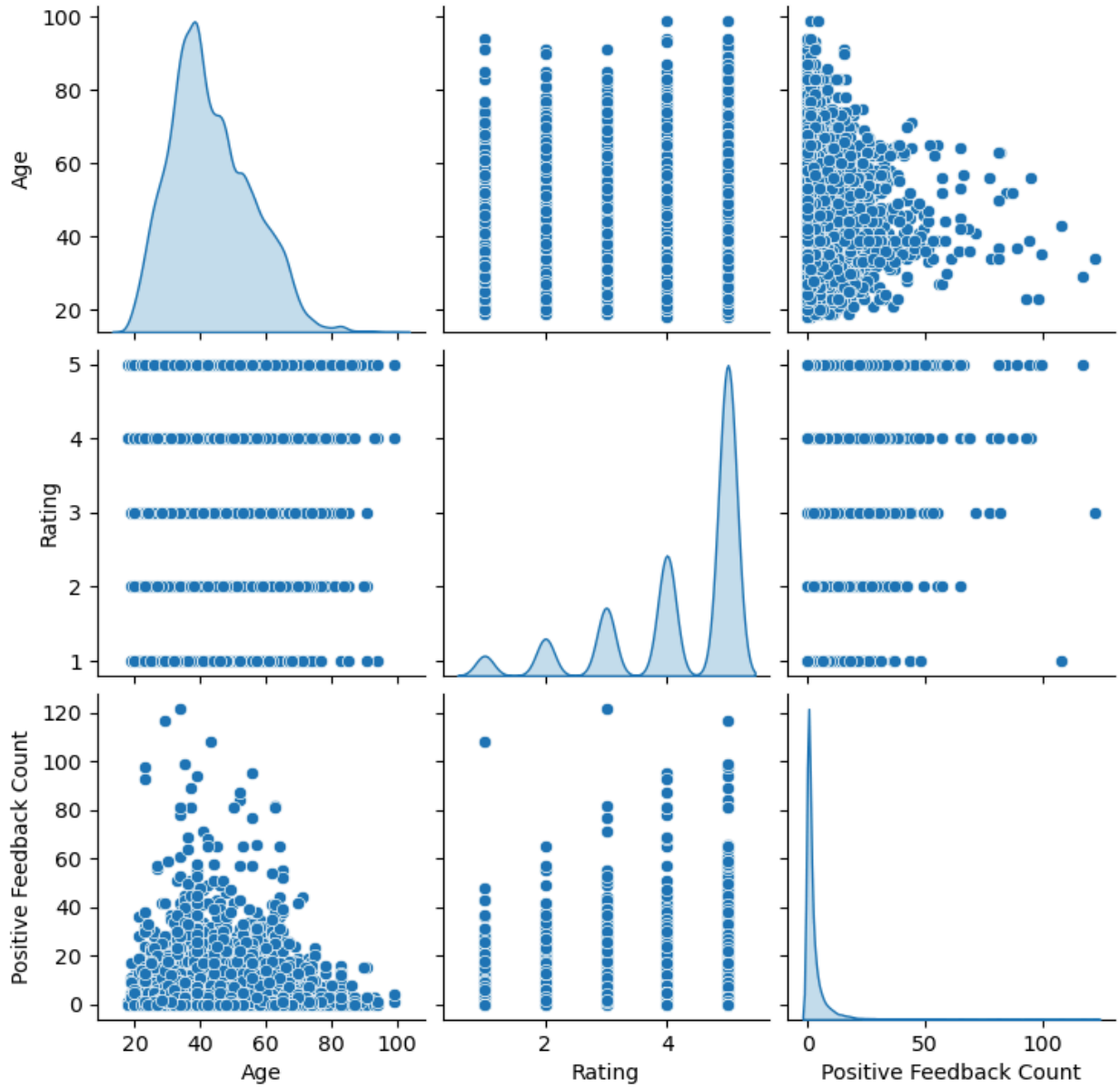
- From this plot we can see that General is most recommended and Intimates is least recommended in division.

4. **Age Vs Recommended IND:**

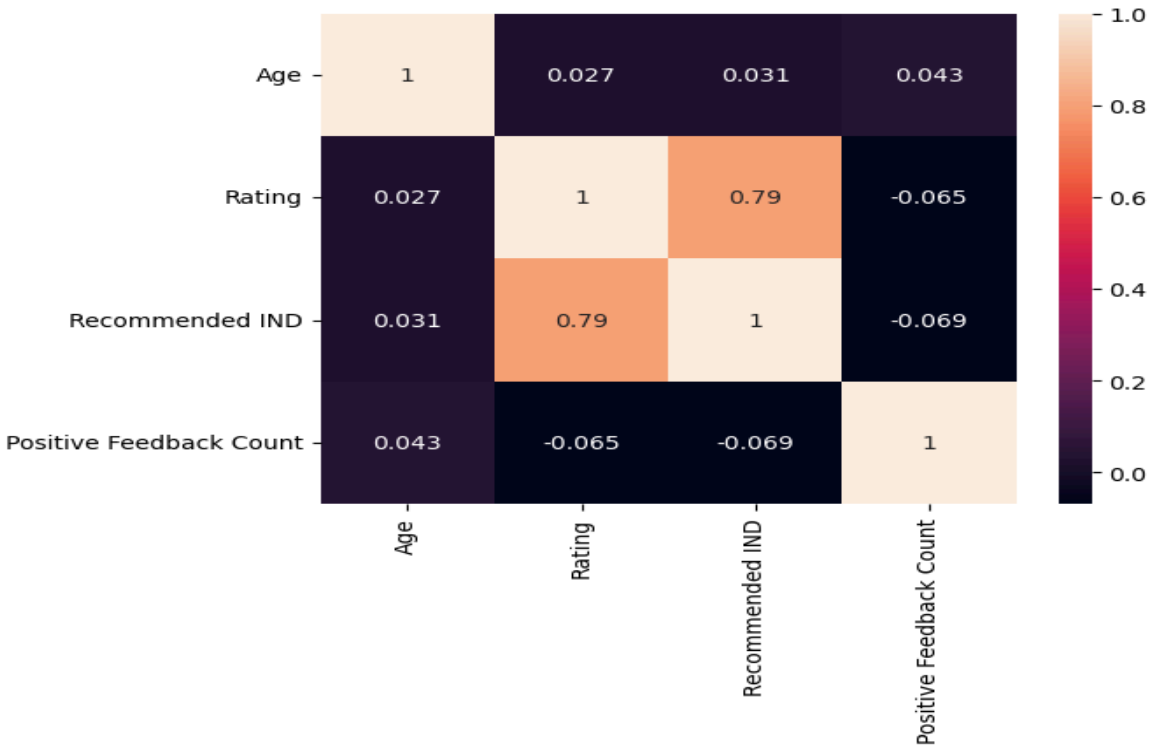


- In this box plot we can see that the Recommended Ind is almost equally distributed in between the age groups of 35-50

Multi-Variate Analysis:



Correlation Matrix for Numerical Variables:



- Recommended IND and Rating are having strong positive correlation between them

Base Model Building:

Count Vectorization:

```
vectorizer = CountVectorizer()

train_data,test_data = train_test_split( df,test_size=0.1)

X_train = vectorizer.fit_transform(train_data["Review Text"])

y_train = train_data['Sentiment']

X_test = vectorizer.transform(test_data["Review Text"])

y_test = test_data['Sentiment']
```

Logistic Regression model:

```
import datetime as d
from sklearn.linear_model import LogisticRegression
logistic_model = LogisticRegression()

# Train the model
logistic_model.fit(xtrain, ytrain)

# Predict on the testing data
y_pred = logistic_model.predict(xtest)

# Calculate accuracy
accuracy = accuracy_score(ytest, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.8292890591741167
```

Classification Report:

```
Logistic Regression
Classification Report on test data
              precision    recall  f1-score   support

    Negative       0.60      0.50      0.55       264
     Neutral       0.43      0.36      0.39       263
    Positive       0.90      0.94      0.92      1822

    accuracy                           0.83      2349
   macro avg       0.64      0.60      0.62      2349
weighted avg       0.82      0.83      0.82      2349
```

TF-IDF Vectorization and Logistic Regression Model:

```
vectorizer = TfidfVectorizer()

x = df["Review Text"]
y = df["Sentiment"]


xtrain, xtest, ytrain, ytest = train_test_split(x,y,test_size=0.1, random_state=10)

xtrain = vectorizer.fit_transform(xtrain)

xtest = vectorizer.transform(xtest)
```

```
logistic_model = LogisticRegression()

start= d.datetime.now()

# Train the model
logistic_model.fit(xtrain, ytrain)

# Predict on the testing data
y_pred = logistic_model.predict(xtest)

# Calculate accuracy
accuracy = accuracy_score(ytest, y_pred)

print('Elapsed time: ',str(d.datetime.now()-start))
print("Accuracy:", accuracy)
```

```
Elapsed time:  0:00:01.358070
Accuracy: 0.8365261813537676
```

Classification Report:

Logistic Regression
Classification Report on test data

	precision	recall	f1-score	support
Negative	0.66	0.48	0.55	264
Neutral	0.46	0.28	0.35	263
Positive	0.88	0.97	0.92	1822
accuracy			0.84	2349
macro avg	0.67	0.57	0.61	2349
weighted avg	0.81	0.84	0.82	2349

Final Model Building:

Feature Extraction:

Preprocessing on the "Review text" variable:

Performed some Natural Language Preprocessing methods on the Review text. The following preprocessing methods have been applied on the data in each record of the Review text:

- Cleaning
- Tokenization
- Stop Words removal
- Lemmatization

The final "Lemma" variable is the new independent variable using which we train the models.

Natural Language Preprocessing Methods Applied to Review Text

1. Cleaning:

- Cleaning is the initial step in the text preprocessing pipeline. This process involves removing or correcting unwanted characters and formatting issues that may be present in the raw text data. Common cleaning operations include:
  - ➔ Removing special characters (e.g., punctuation, numbers).
  - ➔ Correcting spelling errors.
  - ➔ Removing HTML tags and URLs.
  - ➔ Lowercasing all text to maintain uniformity.
- The goal of cleaning is to ensure that the text data is in a standardized format, free from noise and inconsistencies. This makes the subsequent processing steps more effective and the final model more accurate.

2. Tokenization

- Tokenization is the process of breaking down text into individual units called tokens, which can be words, phrases, or symbols. This step converts the cleaned text into a structured format that can be easily analyzed and processed by machine learning models.
- Tokenization is essential for transforming text data into a form that algorithms can process. It helps in segmenting the text into meaningful components, which are necessary for feature extraction and analysis.

3. Stop Words Removal:

- Stop words are common words that typically do not contribute significant meaning to the text and are often removed during preprocessing. Examples of stop words include "is", "and", "the", etc.
- Removing stop words helps in reducing the dimensionality of the text data and focuses on the words that are more meaningful and relevant for the analysis. This can lead to improved performance of machine learning models by eliminating noise.

4. Lemmatization

- Lemmatization is the process of reducing words to their base or root form, known as a lemma. Unlike stemming, which simply trims the end of words, lemmatization considers the context and converts words to their meaningful base forms (e.g., "running" to "run", "better" to "good").
- Lemmatization helps in consolidating different forms of a word into a single term, thereby reducing the complexity of the text data and improving the efficiency of the model. It ensures that variations of a word are treated as a single feature.

Final Preprocessed Variable: "Lemma"

- The final "Lemma" variable represents the text data after it has undergone all the preprocessing steps: cleaning, tokenization, stop words removal, and lemmatization. This variable serves as the new independent variable used for training the models.
- Using the "Lemma" variable ensures that the text data is in its most informative and reduced form, facilitating more accurate and efficient machine learning models. It allows the model to focus on the essential features of the text, leading to better generalization and performance.

Conclusion:

In conclusion, the preprocessing steps applied to the review text are essential for transforming raw text data into a structured and meaningful format suitable for machine learning. Each step, from cleaning to lemmatization, contributes to enhancing the quality of the data and the performance of the models. By focusing on the "Lemma" variable, we ensure that our models are trained on the most relevant and informative features of the text data.

TF-IDF Vectorizer Post Preprocessing

```
vectorizer = TfidfVectorizer()

train_data,test_data = train_test_split(df,test_size=0.1)

X_train = vectorizer.fit_transform(train_data["Lemma"])

Y_train = train_data['Sentiment']

X_test = vectorizer.transform(test_data["Lemma"])

Y_test = test_data['Sentiment']
```

Building Various Machine Learning Models:

Classification Reports:

Logistic Regression:

Elapsed time: 0:00:01.050464
Testing Accuracy: 0.8424861643252448
Training Accuracy: 0.8634621753323556

Logistic Regression

	precision	recall	f1-score	support
Negative	0.67	0.46	0.54	236
Neutral	0.45	0.23	0.30	262
Positive	0.88	0.98	0.93	1851
accuracy			0.84	2349
macro avg	0.67	0.56	0.59	2349
weighted avg	0.81	0.84	0.82	2349

KNeighbours:
Elapsed time: 0:00:40.729139
Testing Accuracy: 0.7969348659003831
Training Accuracy: 0.839523111132138

KNeighbours

	precision	recall	f1-score	support
Negative	0.49	0.32	0.39	236
Neutral	0.28	0.13	0.17	262
Positive	0.85	0.95	0.90	1851
accuracy			0.80	2349
macro avg	0.54	0.47	0.49	2349
weighted avg	0.75	0.80	0.77	2349

Support Vector Machine(SVM):
Elapsed time: 0:03:29.807062
Testing Accuracy: 0.8441890166028098
Training Accuracy: 0.9539196669347589

Support Vector Machine (SVM)

	precision	recall	f1-score	support
Negative	0.71	0.44	0.54	236
Neutral	0.50	0.20	0.28	262
Positive	0.87	0.99	0.93	1851
accuracy			0.84	2349
macro avg	0.70	0.54	0.58	2349
weighted avg	0.81	0.84	0.82	2349

Naive Bayes:
Elapsed time: 0:00:00.049795
Testing Accuracy: 0.7905491698595147
Training Accuracy: 0.7782088281213039

Naive Bayes

	precision	recall	f1-score	support
Negative	1.00	0.02	0.04	236
Neutral	0.50	0.00	0.01	262
Positive	0.79	1.00	0.88	1851
accuracy			0.79	2349
macro avg	0.76	0.34	0.31	2349
weighted avg	0.78	0.79	0.70	2349

Decision Tree:
Elapsed time: 0:00:10.587380
Testing Accuracy: 0.7445721583652618
Training Accuracy: 0.9974925486114397

Decision Tree

	precision	recall	f1-score	support
Negative	0.32	0.28	0.30	236
Neutral	0.22	0.20	0.21	262
Positive	0.86	0.88	0.87	1851
accuracy			0.75	2349
macro avg	0.47	0.46	0.46	2349
weighted avg	0.73	0.75	0.74	2349

MODEL EVALUATION:

- Though we got the same testing accuracy for the SVM model and Logistic Regression model. the training accuracy is significantly very high. So the SVM model is an underfit model. The time taken by SVM model compared to Logistic Regression model is very high.Hence, we concluded Logistic Regression as the best fit model.

- The training accuracy is 86% and the testing accuracy is 84%. There is no significant difference between the accuracies. The model is not overfit or underfit. The time taken for training is very less.

BEST FINAL MODEL:

AFTER PREPROCESSING:

```
# Logistic Regression Model
logistic_model = LogisticRegression()

start= d.datetime.now()

# Train the model
logistic_model.fit(X_train, Y_train)

# Predict on the test and train data
Y_test_pred = logistic_model.predict(X_test)
Y_train_pred = logistic_model.predict(X_train)

# TIME TAKEN
print('Elapsed time: ',str(d.datetime.now()-start))

# Calculate accuracy
accuracy = accuracy_score(Y_test, Y_test_pred)
print("Testing Accuracy:", accuracy)
print("Training Accuracy:",accuracy_score(Y_train, Y_train_pred))
```

```
Elapsed time:  0:00:01.050464
Testing Accuracy: 0.8424861643252448
Training Accuracy: 0.8634621753323556
```

Logistic Regression

	precision	recall	f1-score	support
Negative	0.67	0.46	0.54	236
Neutral	0.45	0.23	0.30	262
Positive	0.88	0.98	0.93	1851
accuracy			0.84	2349
macro avg	0.67	0.56	0.59	2349
weighted avg	0.81	0.84	0.82	2349

LIMITATIONS:

In the field of sentiment analysis, achieving a balanced dataset is crucial for building robust and unbiased models. However, in our current dataset, we observe a significant imbalance favoring positive sentiment reviews. This imbalance poses several challenges that can affect the performance and generalizability of our sentiment analysis models.

CHALLENGES:

In the context of natural language processing (NLP) and text analytics, preprocessing is a crucial step that significantly impacts the performance of subsequent models and analyses. One common preprocessing task involves removing stop words from text data. In this report, we detail the preprocessing of the Review text variable, which comprises 23,486 records. This task, although computationally intensive, is essential for enhancing the quality and relevance of the data used in downstream applications.

SCOPE:

In future work, we can do the following models. We believe that these models will give us more accurate results.

VADER (Valence Aware Dictionary and sEntiment Reasoner):

- It is a lexicon and rule-based sentiment analysis tool specifically attuned to sentiments expressed in social media. It is particularly effective for text from microblogging platforms, such as Twitter, and other informal text formats.
- VADER is especially useful for quick and accurate sentiment analysis in real-time applications where the text is informal and unstructured. Its simplicity and effectiveness make it a popular choice for many natural language processing tasks.

RoBERTa (A Robustly Optimized BERT Pretraining Approach):

- It is an advanced language model developed by Facebook AI, designed to improve upon the BERT (Bidirectional Encoder Representations from Transformers) model. It uses a transformer architecture and focuses on pretraining methods to enhance language understanding.
- RoBERTa is a powerful language model that offers improved performance over BERT by optimizing the pretraining process and using larger datasets. The Hugging Face transformers library makes it easy to use RoBERTa for a variety of NLP tasks, whether you need to perform inference or fine-tune the model on your specific dataset. By leveraging these tools, you can efficiently implement state-of-the-art NLP solutions.