**Mini Project :Social Network Analysis (SNA)**

Name:Monika Desai

Roll No :31

Class : Msc-cs(SEM-2)

Topic:

# *Amazon Product Reviews Sentiment Analysis with Machine Learning*

**Introduction ->**

Sentiment Analysis (also known as opinion mining or emotion AI)is a sub-field of NLP that measures the inclination of people’s opinions (Positive/Negative/Neutral) within the unstructured text.

**Product reviews are becoming more important with the evolution of traditional brick and mortar retail stores to online shopping.**

**Consumers are posting reviews directly on product pages in real time. With the vast amount of consumer reviews, this creates an opportunity to see how the market reacts to a specific product.**

**Code :**

**import matplotlib.pyplot as plt**

**import pandas as pd**

**import numpy as np**

**import seaborn as sns**

**import math**

**import warnings**

**warnings.filterwarnings('ignore') # Hides warning**

**warnings.filterwarnings("ignore", category=DeprecationWarning)**

**warnings.filterwarnings("ignore",category=UserWarning)**

**sns.set\_style("whitegrid") # Plotting style**

**np.random.seed(7) # seeding random number generator**

**df = pd.read\_csv('amazon.csv')**

**print(df.head())**

**# Describing the Dataset**

**data = df.copy()**

**data.describe()**

**# Information : column name with data type**

**data.info()**

**#We need to clean up the name column by referencing review\_id (unique products)**

**data["review\_id"].unique()**

**review\_id\_unique = len(data["review\_id"].unique())**

**print("Number of Unique Review ID: " + str(review\_id\_unique))**

**#Visualizing the distributions of numerical variables:**

**data.hist(bins=50, figsize=(20,15))**

**plt.show()**

**#we will split it into training set and test sets. Our goal is to train a sentiment analysis classifier.**

**#we will need to do a stratified split on the reviews score (star rating)**

**from sklearn.model\_selection import StratifiedShuffleSplit**

**print("Before {}".format(len(data)))**

**dataAfter = data.dropna(subset=["star\_rating"])**

**# Removes all NAN in star.rating**

**print("After {}".format(len(dataAfter)))**

**dataAfter["star\_rating"] = dataAfter["star\_rating"].astype(int)**

**split = StratifiedShuffleSplit(n\_splits=5, test\_size=0.2)**

**for train\_index, test\_index in split.split(dataAfter,**

**dataAfter["star\_rating"]):**

**strat\_train = dataAfter.reindex(train\_index)**

**strat\_test = dataAfter.reindex(test\_index)**

**#We need to see if train and test sets were stratified proportionately in comparison to raw data:**

**print(len(strat\_train))**

**print(len(strat\_test))**

**print(strat\_test["star\_rating"].value\_counts()/len(strat\_test))**

**#We will use regular expressions to clean out any unfavorable characters in the dataset**

**reviews = strat\_train.copy()**

**reviews.head()**

**print(len(reviews["product\_parent"].unique()), len(reviews["review\_id"].unique()))**

**print(reviews.info())**

**#Entire training dataset average rating**

**print(reviews["star\_rating"].mean())**

**asins\_count\_ix = reviews["product\_title"].value\_counts().index**

**plt.subplots(2,1,figsize=(16,12))**

**plt.subplot(2,1,1)**

**reviews["product\_parent"].value\_counts().plot(kind="bar", title="Product Parent Frequency")**

**plt.subplot(2,1,2)**

**sns.pointplot(x="product\_parent", y="star\_rating", data=reviews)**

**plt.xticks(rotation=90)**

**plt.show()**

**# Using the features in place, we will build a classifier that can determine a review’s sentiment.**

**def sentiments(rating):**

**if (rating == 5) or (rating == 4):**

**return "Positive"**

**elif rating == 3:**

**return "Neutral"**

**elif (rating == 2) or (rating == 1):**

**return "Negative"**

**# Add sentiments to the data**

**strat\_train["Sentiment"] = strat\_train["star\_rating"].apply(sentiments)**

**strat\_test["Sentiment"] = strat\_test["star\_rating"].apply(sentiments)**

**print(strat\_train["Sentiment"][:20])**

**Final Result based on rating =**

