**AUTOMATED REVIEW RATING SYSTEM**

1. **Objective**

The goal of this project is to automatically predict or assign ratings (like 1–5 stars) to user reviews (text) using Natural Language Processing (NLP) techniques.  
This saves time compared to manual rating and can be used for e-commerce, app reviews, or any platform collecting feedback.

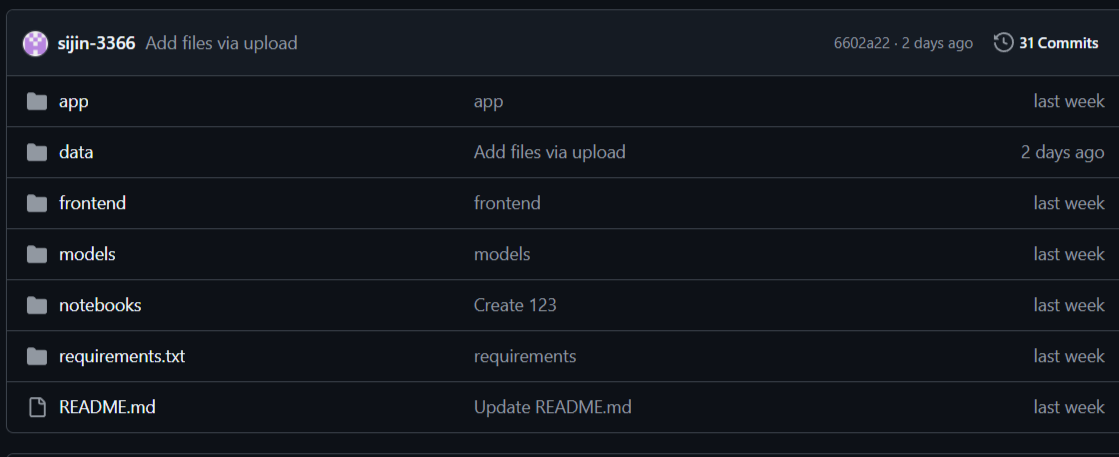
1. **Environment Setup**

* Installed python 3.13.5 with libraries: numpy,pandas ,matplotlib ,seaborn ,scikit-learn,nltk.
* IDE used:VS code

1. **GitHub Setup**

Created a repository : [Automated\_review\_rating\_system](https://github.com/sijin-3366/Automated_review_rating_system)

Structure of directory



1. **Data Collection**

* Datas are collected from Project page of amazon.
* Dataset has 25 lakh rows
* Site link: [https://amazon-reviews-2023.github.io](%20https://amazon-reviews-2023.github.io)
* Dataset link:<https://github.com/sijin-3366/Automated_review_rating_system/tree/2d985f5763234735641c32ba75565e9de834232c/data/amazon_data>

1. **Import packages + Data loading + Concatenate**

**Code:**

**import numpy as np**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**import nltk**

**import warnings**

**warnings.filterwarnings('ignore')**

**df1=pd.read\_csv('product\_data.csv')**

**df2=pd.read\_csv('product\_data1.csv')**

**df=pd.concat([df1,df2],ignore\_index=True)**

**Data Preprocessing**

* **Data preprocessing** is the step of **cleaning and transforming raw data** into a usable format for analysis or model training.
* It improves **data quality**, reduces **noise**, and helps models learn more effectively.

**5.1 Basic Steps**

* Shape: (number\_of\_rows, number\_of\_columns)

df.shape

* Head: Collect first 5 rows

df.head()

* Tail: Collect Last 5 rows

df.tail()

* Columns: Find the column Labels

df.columns

* Drop Unnecessary columns

**5.2 Handling Missing Values**

* Handling missing values is one of the most important parts of data preprocessing.
* Missing values can distort your analysis or model training
* Find Missing values

df.isna().sum()

rating 0

review 1094

* Drop Missing values

df=df.dropna()

df.isna().sum()

rating 0

review 0

**5.3 Unique Rating**

**Code:**

**df['rating'].unique()**

**array([5, 3, 1, 4, 2])**

* 1. **Remove**
  2. **Remove duplicates and their corresponding row**

**Code:**

**duplicate=df\_clean[df\_clean.duplicated(subset=['rating','review'], keep=False)]**

**new\_data=df\_clean.drop(duplicate.index)**

* 1. **Remove Conflicting reviews**

**Code:**

**variable=df.groupby('review')['rating'].nunique().loc[lambda x: x > 1].index**

**df\_clean=df[~df['review'].isin(variable)]**

**df\_clean=df\_clean.reset\_index(drop=True)**

**print(df\_clean)**

* 1. **Visualization**

**Bar chart**

**Code:**

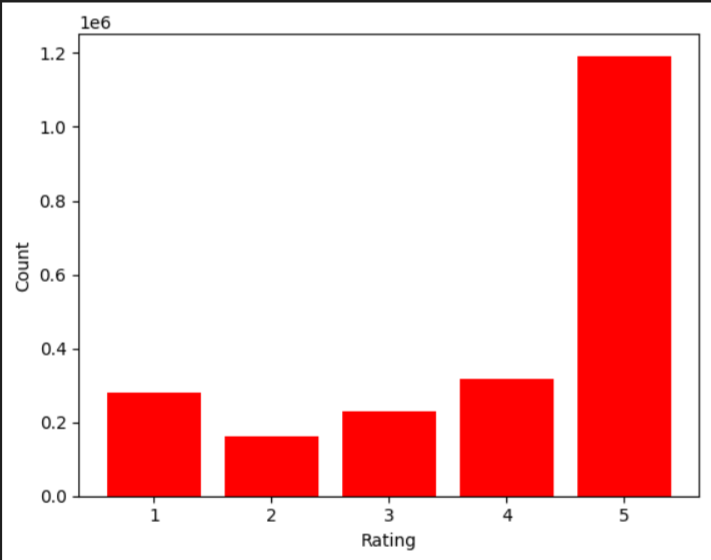
**x=new\_data['rating'].value\_counts().sort\_index()**

**plt.bar(x.index,x.values,color='red')**

**plt.xlabel("Rating")**

**plt.ylabel("Count")**

**plt.show()**

****

**Taking unique rows**

**Code:**

**dat=pd.read\_csv('bal\_clean.csv')**

**drop\_df=new\_data.merge(dat,how='outer',indicator=True).query('\_merge=="left\_only"').drop('\_merge',axis=1)**

**drop\_df['rating'].value\_counts()**

* 1. **Imbalancing**
* **5→20%=75000 ,  4→30%=112500   ,   3→25%=93750   ,   2→15%=56250   ,   1→10%=37500**

**Code:**

imb\_1=drop\_df[drop\_df['rating']==1].sample(37500,random\_state=42)

imb\_2=drop\_df[drop\_df['rating']==2].sample(56250,random\_state=42)

imb\_3=drop\_df[drop\_df['rating']==3].sample(93750,random\_state=42)

imb\_4=drop\_df[drop\_df['rating']==4].sample(112500,random\_state=42)

imb\_5=drop\_df[drop\_df['rating']==5].sample(75000,random\_state=42)

imbalance=pd.concat([imb\_1,imb\_2,imb\_3,imb\_4,imb\_5],ignore\_index=True)

imbalance

**Imbalanced dataset link:** [**https://github.com/sijin-3366/Automated\_review\_rating\_system/tree/5359e90bd33aeab9170ebfe5d4278808c64f263e/data/imbalance\_clean**](https://github.com/sijin-3366/Automated_review_rating_system/tree/5359e90bd33aeab9170ebfe5d4278808c64f263e/data/imbalance_clean)

**Bar chart**

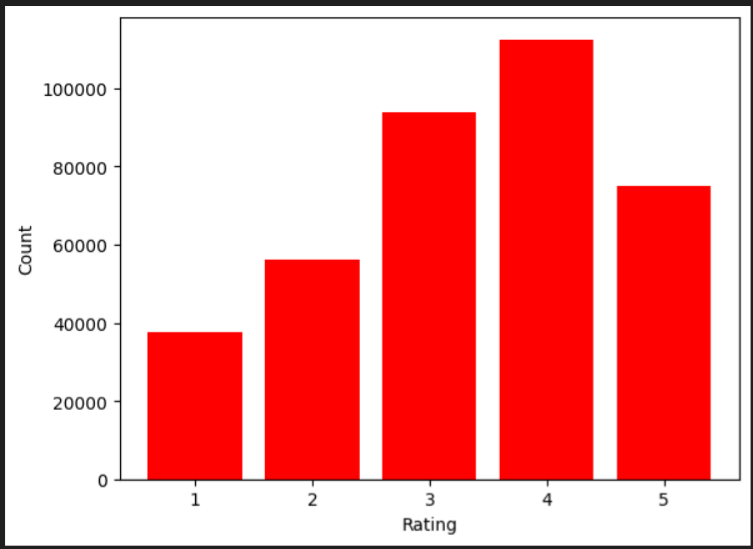
x=imbalance['rating'].value\_counts().sort\_index()

plt.bar(x.index,x.values)

plt.xlabel('Rating')

plt.ylabel('Count')

plt.show

****

**Pie chart**

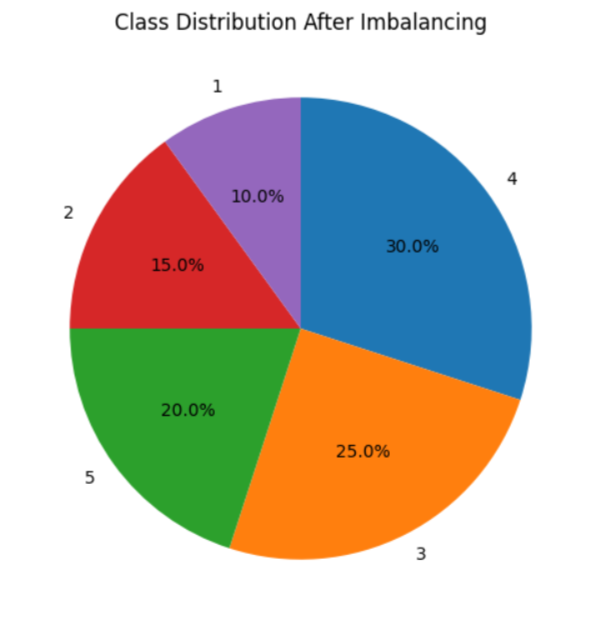
count= balance['rating'].value\_counts()

plt.figure(figsize=(6,6))

plt.pie(count, labels=count.index,autopct='%1.1f%%',startangle=90,counterclock=False)

plt.title('Class Distribution After Balancing')

plt.show()

**e**

1. **NATURAL LANGUAGE PROCESSING**

Natural Language Processing (NLP) is a field of Artificial Intelligence that focuses on enabling computers to understand, interpret and generate human language. In an automatic review rating system, NLP techniques are used to read customer reviews and predict their sentiment or rating automatically

**6.1 Tokenization**

Tokenization is the process of breaking a large text into smaller units called tokens. These tokens can be words, subwords, characters, or even sentences, depending on the level of analysis required.

**Code:**

**from nltk.tokenize import TweetTokenizer**

**tk=TweetTokenizer()**

**review=review.apply(lambda x:tk.tokenize(x)).apply(lambda x:' '.join(x))**

**6.2 Regular Expression**

A regular expression (often abbreviated as regex) is a special sequence of characters that defines a search pattern. It is used to match, find, or manipulate specific strings of text according to defined rules.

**Code:**

**import re**

**review=review.str.replace('[^a-zA-Z0-9]',' ',regex=True)**

**review**

What this code does:

It cleans the text by removing all special characters (punctuation, symbols) and replacing them with spaces, leaving only letters and numbers. This is a common preprocessing step in NLP to normalize text before tokenization or modeling.

**6.3 Filter text**

**Code:**

**from nltk.tokenize import TweetTokenizer**

**review=review.apply(lambda x:' '.join([w for w in tk.tokenize(x) if len(w)>=3]))**

**review**

**What this code does:**

* Tokenizes each text (especially social media-style text) into words and symbols using TweetTokenizer.
* Removes short tokens (less than 3 characters).
* Reconstructs the cleaned tokens into a single string per review.

**6.4 Lemmitization**

Lemmatization is a text-processing technique in Natural Language Processing (NLP).  
It reduces words to their lemma (dictionary form or base form) by considering the word’s meaning and part of speech (POS).The resulting form may not always be a real dictionary word but serves as a common representation for related words.

**Code:**

**from nltk.stem import WordNetLemmatizer**

**nltk.download('wordnet')**

**nltk.download('omw-1.4')**

**lemmatizer = WordNetLemmatizer()**

**review=review.apply(lambda x:[lemmatizer.lemmatize(i.lower(),pos='v') for i in tk.tokenize(x)]).apply(lambda x:' '.join(x))**

**review**

**Why use Lemmatization instead of stemming?**

**Stemming just chops off prefixes/suffixes using simple rules but Lemmatization uses a dictionary + part-of-speech to return an actual word .**

**6.5 Remove Stopwords**

* Stop words are common words (such as *“the”*, *“is”*, *“and”*, *“in”*) that usually carry little meaning by themselves.
* Removing them is a typical preprocessing step to reduce noise and focus on meaningful terms

**Purpose:**

* Reduce dimensionality of text data.
* Speed up processing and training of models.
* Improve accuracy by keeping only informative words.

**Stopwords =[**'i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves',

'you', "you're", "you've", "you'll", "you'd", 'your', 'yours',

'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she',

"she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself',

'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which',

'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am',

'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has',

'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the',

'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of',

'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into',

'through', 'during', 'before', 'after', 'above', 'below', 'to',

'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under',

'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where',

'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most',

'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same',

'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just',

'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o',

're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't",

'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn',

"hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn',

"mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan',

"shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren',

"weren't", 'won', "won't", 'wouldn', "wouldn't"]

* **English stop word list contains 179 stop words.**

**Code:**

**from nltk.corpus import stopwords**

**stop=stopwords.words('english')**

**review=review.apply(lambda x:[i for i in tk.tokenize(x) if i not in stop]).apply(lambda x:' '.join(x))**

**review**

1. **Train-Test Split**

Train–test split is the process of dividing a dataset into two parts:

* a training set used to fit (train) the model, and
* a test set used to evaluate the model’s performance on unseen data.

**Purpose:**

* Prevent overfitting: ensures the model is evaluated on data it hasn’t seen before.
* Estimate generalization: gives a realistic measure of how well the model will perform on new data.
* Provide a fair evaluation: separates data used for learning from data used for testing.

**Code:**

**from sklearn.model\_selection import train\_test\_split**

**x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.30,random\_state=42,stratify=y)**

**Parameters:**

* x,y - Features(x) and Target labels(y)
* test\_size - Fraction or number of samples to use for the test set (e.g., 0.3 = 30%)
* random\_state- Seed for reproducibility; using the same seed gives the same split.
* Stratify=y - Preserves class proportions of y in train/test sets (important for classification).

1. **TF–IDF (Term Frequency – Inverse Document Frequency)**

TF–IDF is a numerical statistic that reflects how important a word is to a document in a collection (corpus).  
It combines two measures:

* Term Frequency (TF): how frequently a word appears in a single document.
* Inverse Document Frequency (IDF): how unique or rare the word is across all documents.

**Formula:**

**TF-IDF(t,d)=TF(t,d)×IDF(t)**

* **TF(t,d) = number of times term *t* appears in document *d* ÷ total number of terms in *d*.**
* **IDF(t) =log(Total Documents/no:of Documents containing t)**
* **t=term(word)**
* **d=document**

**MODEL CREATION**

**Code:**

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

lg=LogisticRegression(max\_iter=25000)

lg.fit(x\_train,y\_train)

y\_pred=lg.predict(x\_test)

accuracy=accuracy\_score(y\_test,y\_pred)

print("Accuracy:",accuracy)

**Model explaination:**[**https://github.com/sijin-3366/Automated\_review\_rating\_system/blob/5359e90bd33aeab9170ebfe5d4278808c64f263e/models/Logistic%20Regression%20Model.docx**](https://github.com/sijin-3366/Automated_review_rating_system/blob/5359e90bd33aeab9170ebfe5d4278808c64f263e/models/Logistic%20Regression%20Model.docx)

**Samples of 1 star**

* **They both broke the same day I bought them for myself as soon as I put them in the (stones) fell out almost simultaneously I'm kinda done buying on amazon I ordered this watch two separate times it never came**
* **It took 2 weeks to get these masks. Was excited to see them but when I opened the package, it was not as expected. The colors were correct but NOT 3 layers! The back of the masks was a thin piece of cloth and had a pocket for a filter, which was not in the description. So disappointed"**
* **They both had broken handle.....The first one I received had broken handle, so I asked to replace it. But, the second one also had the same problem.....OMG"**
* **It lasted less than 2 months before the light button fell off. Don’t plan on buying these again!**
* **I love this company. But I only received one earring which was upsetting**
* **As noted above the watch was shipped out of box. Box is two piece, and was together in a way used for display. Appears watch and open box thrown into package and shipped. Watch band was broken, missing pins, etc. Watch backing plate was off and would not go on, damaged. Watch unusable. So, 39.95 plus shipping to go from the shippers trash can to mine. Pretty unethical stuff.<br />To be fair - its a cheap, cheap watch. But it should show up in one piece, not multiple damaged.**

**Samples of 2 star**

* **Need more value For the Money**
* **The product has an out of proportion top and shoulder area.**
* **The tassels are made from a really cheap yarn/thread. they're uneven. i tried cutting them myself to even out. If i'd know i'd be finishing the product, i would not have paid more than $2.00.**
* **Correct size but the sew was a little loose on both side it look weird**
* **looks cheap and the zipper is bad. i guess you get what you pay for**
* **Wanted more of a ball type cap, not a “truckers” hat, too tall for what the picture shown, was a good quality hat though.**

**Samples of 3 star**

* **Comfortable on the skin Has a pocket for inserting a filter.**
* **A little tight across the hips.**
* **Tunic portion fits as expected - it's loose and looks cute with leggings. Sleeves are ridiculously tight, though, and look comical on anyone like myself who has larger arms. Anyone here figure out a way to stretch the sleeves a bit, to avoid the straight jacket look?"**
* **More a child's size.  Triangles are too close together for it to fit and average sized frame.**
* **3 stars bcus the material of this dress is so cheap but cute otherwise.**
* **Doesn't look anything like the picture once you put it on. For what you pay its a decent dress, just would need a lot of altering to look anything like the picture shows. I'm returning mine bc I'm a size 0 & ordered a small & it fits me like a potato sack. It's BAD."**

**Samples of 4 star**

* **This sweat shirt is an excellent product.  It has kept me warm while shoveling snow here in New England. Made in USA**
* **Bought watch for my daughter  20 years old with special needs. She enjoys the camera and microphone  options in this watch. Very easy to set up just follow the instructions and side note, adult fingers might be too thick for the screen but is manageable for setup. Satisfied with purchase and I would recommend and would purchase again.**
* **Cute top. Looks as pictured..The fabric is not a true sweater material but it is not uncomfortable. The length is great. I sized up to a medium for longer length.**
* **Love it so much I wear it and I get a lot of compliments beautiful jumper**
* **The fabric is thicker and heavier than I expected, so much so that the weight of the skirts feels like it pulls the top down lower, but visually it isn’t noticeable. The extra stretch and weight also made the dress a little longer.**
* **SUPER cute! I get a lot of compliments on these earrings. Only reason I didn't give 5 stars is because they're kind of difficult to put on. Especially if you don't have nimble fingers. But I love how they look!**

**Samples of 5 star**

* **Super comfortable and great quality**
* **I like it and it’s adjustable ‘**
* **Very good and useful product. It protect me every time when I am biking undef the sun.**
* **Super cute and adjusts easily! The long wire bends easily, but a quick adjustment easily fixes it. Overall love it"**
* **Very cute on, a little bigger than I thought. Fabric very nice weight."**
* **Person I bought it for loved it and shipped fast. no complaints about the charm although i recommend you get another chain for it. doesn't look so sturdy.**