AUTOMATED REVIEW SYSTEM

1.Project Overview

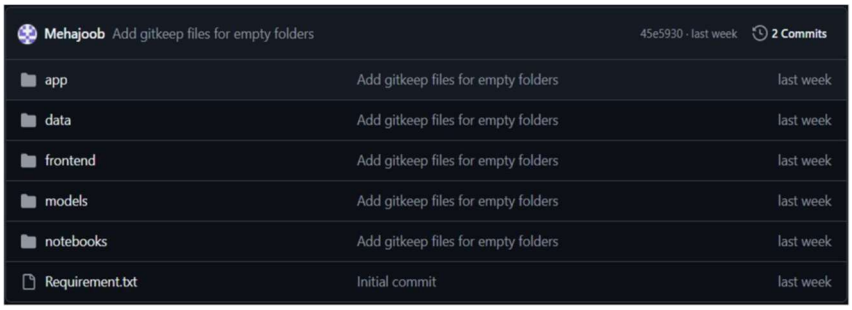
The Automated Review Classification System is a Natural Language Processing (NLP) project developed using the Amazon Fine Food Reviews dataset. The primary objective is to interpret customer opinions and automatically categorize each review into a rating scale from 1 to 5. The workflow includes comprehensive text preprocessing, TF-IDF feature extraction, and sentiment oriented NLP techniques to capture underlying opinion patterns. By transforming raw textual feedback into structured insights, the model helps assess customer satisfaction trends and supports decision-making for business improvement and product enhancement.

2. Environment Setup

The project was executed in Python 3.10 using Jupyter Notebook. Core libraries such as Pandas, NumPy, Matplotlib, Seaborn, and Scikit-learn supported data handling, visualization, and model development. NLTK was used for text preprocessing, and Imbalanced-Learn helped address rating imbalance. All dependencies were managed using pip within an Anaconda virtual environment.

3.GitHub Project Setup

3.1 Created GitHub Repository : Automated-Review-Rating-System Structure of directory



4.Data Collection

The dataset was sourced from Kaggle, containing Amazon customer feedback and associated star ratings. After importing the dataset, unnecessary attributes such as product IDs, profile details, and timestamps were removed, retaining only the review text and rating score for model training.

The distribution of ratings in the collected data is as follows:

Rating Count

5 ⭐ 363,122

4 ⭐ 80,655

3 ⭐ 42,640

2 ⭐ 29,769

1 ⭐ 52,268

Dataset link:
https://github.com/Mehajoob/Automated_review_rating_system/tree/main/data/imbalanced

5. Data Preprocessing

The preprocessing stage covered both dataset cleaning and text refinement. Duplicate entries, missing values, and non-review text were removed to maintain reliability. All reviews were transformed to lowercase, and noise elements such as HTML tags, URLs, emojis, punctuation, and special symbols were stripped out. Stopwords were eliminated, and lemmatization was used to convert words to their base form. Additionally, extremely short and unusually long reviews were filtered out to ensure that only meaningful and readable text was retained for model training.

url_pattern = r"https?://\S+|www\.\S+" html_pattern = r"<.*?>"

emoji_pattern = r"[\U00010000-\U0010ffff]|[\u263a-\U0001f645]"

special_pattern = r"[^a-zA-Z0-9\s]" def clean_basic(text):

text = text.lower() #(converting to lowercase) text =

re.sub(url_pattern, " ", text) #(removing url) text =

re.sub(html_pattern, " ", text) #(removing html) text =

re.sub(emoji_pattern, " ", text) #(removing emoji) text =

re.sub(special_pattern, " ", text) #(removing special patterns) doc =

nlp(text)

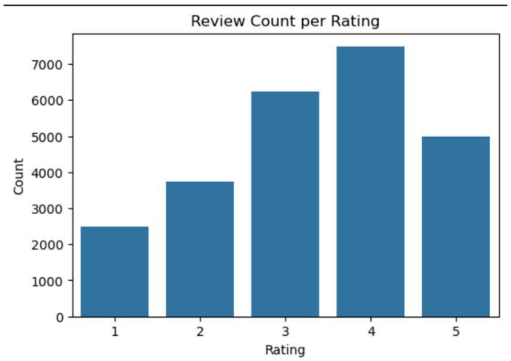
tokens = [token.lemma_ for token in doc if token.text not in STOP_WORDS and len(token.text) >

1] #(lemmetization) cleaned = " ".join(tokens) return cleaned

df["cleaned_text"] = df["Text"].apply(clean_basic) #(applied to our text)

6.imbalanced Dataset

An **imbalanced dataset** occurs when the distribution of target classes is not uniform one class significantly outnumbers the others.In such cases, machine learning models tend to become biased toward the majority class, ignoring the minority class. This imbalance can reduce model performance, especially in detecting rare or critical cases.Techniques like **resampling, class weighting, or anomaly detection** are often used to address this issue.



7. Natural Language Processing (NLP)

NLP techniques were implemented to convert raw review text into a structured format suitable for analysis. The process involved tokenization and stopword filtering to eliminate low-value and repetitive words that do not contribute significantly to sentiment understanding. After cleaning, the text was transformed into numerical features using TF-IDF vectorization, allowing the model to assess how important specific terms are across different reviews.

7.1 Stopword Removal

Stopword removal was carried out using spaCy's English stopword list, which filters common words such as the, is, at, and, etc. These words typically do not carry meaningful sentiment and can introduce noise into the dataset. By removing them, the model focuses on terms that better reflect opinion strength and emotional tone, leading to clearer sentiment signals in classification

{'top', 'may', 'three', 'bottom', 'do', 'fifty', 'few', 'in', 'more', 'see', 'on', 'still', 'well', 'take', 'to', 'us', 'keep', 'these', 'have', 'but', 'get', 'everyone', 'nobody', 'less', 'further', 'anyone', 'now', 'over', 'third', 'be', 'put', 'hundred', 'anyway', 'part', 'own', 'not', 'made', 'call', 'front', 'together', 'mine', 'amount', 'back', 'up', 'two', 'sixty', 'being', 'side', 'between', 'become', 'show', 'move', 'please', 'empty', 'will', 'give', 'say', 'one', 'which', 'someone', 'make', 'ten', 'down', 'can', 'an', 'me', 'even', 'are', 'full', 'off', 'out', 'last', 'them', 'four', 'name', 'go'}

7.2 Lemmatization

Lemmatization is the process of converting words to their root or dictionary form (lemma) while preserving their correct meaning and grammatical role. Examples:

Original Word	Lemma
running	run
better	good
cars	car

This process ensures that different inflections of a word are treated as a single token, thereby improving vocabulary consistency and reducing dimensionality in the TF-IDF feature space.

7.2.1 Why Lemmatization Instead of Stemming?

Feature	Stemming	Lemmatization
Output form	Often truncated, non-dictionary terms	Proper dictionary words
Meaning preservation	Low	High
Linguistic understanding	Simply strips suffixes	Considers grammar + vocabulary rules
Example	**studies → studi**	**studies → study**

Stemming mechanically cuts word endings, often producing invalid or incomplete forms and losing semantic clarity. In contrast, lemmatization uses linguistic rules and part-of-speech tagging to generate accurate base forms, ensuring that context is maintained.

7.3 Conclusion

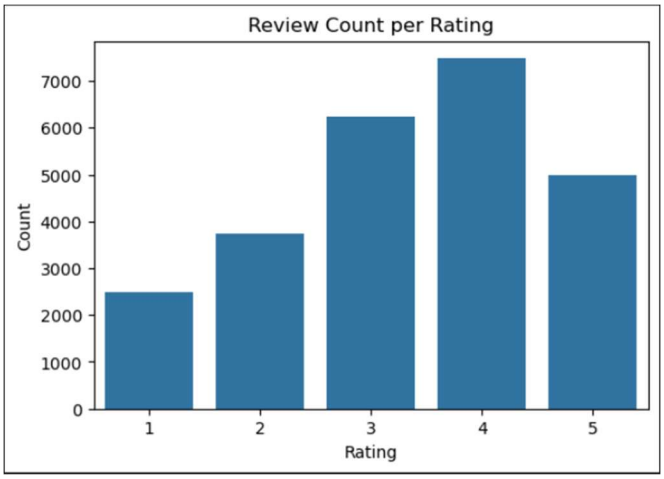
Lemmatization was chosen over stemming to retain true word meaning, support clearer sentiment interpretation, and enhance the reliability of TF-IDF feature representation by reducing noise and vocabulary redundancy.

8.Data Visualization

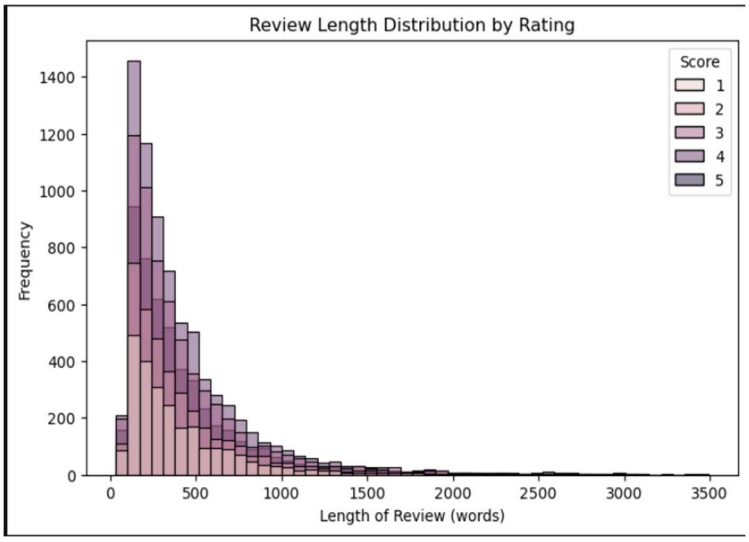
Data visualization helps interpret the review distribution and content patterns. Bar plots show the number of reviews per rating, clearly highlighting imbalance and supporting the need for balancing techniques. Histograms and box plots illustrate word count variations and outliers across ratings, while sample reviews provide quick qualitative insight into sentiment differences.

8.1 Imbalanced Data

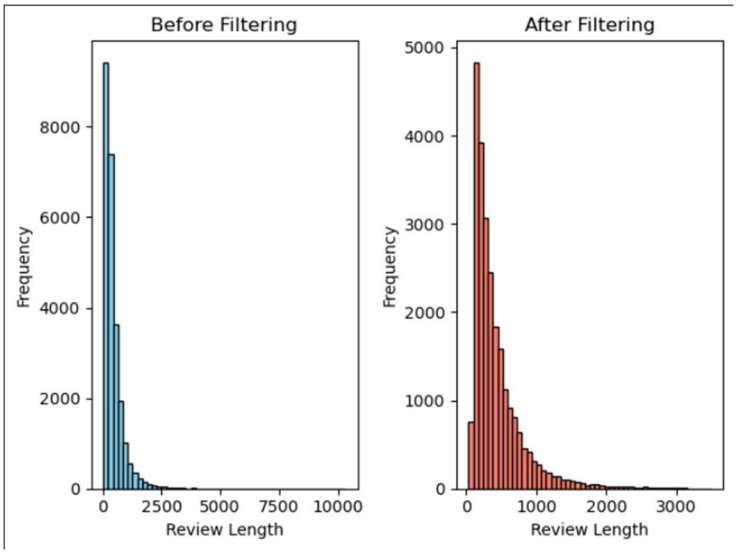
8.1.1 Review Count Per Rating



8.1.2 Review Length Distribution by Rating



8.1.3 Before and after filtering



8.1.4 Sample Reviews

==========

Rating: 1 | Showing 10 sample reviews

==========

Review 22163:

try find specific type cup husband enjoy couldn find store check amazon large quantity cheap buy store win

Review 20823:

royal oak virginia salt peanut great highly recommend reorder time time buy pack total 40 oz can

Review 14082:

nutritional benefit extra virgin olive oil legendary robust flavor pompeian thickness hook oil year

Review 21550:

record vocal home pc shure c606 mic big improvement mic stereo sure background noise prevailant mic method star instead gain gain knob turn way add gain software sound loud overall great product regret buy recommend

Review 10861:

find wild thyme year ago continue use product lemon salad refresher favorite salad dress light cal tasty excellent green pasta salad especially good fresh heirloom tomato fresh avocado slice yum customer service excellent helpful site share great recipe fan

Review 24832:

taste like coffee nice mild roast taste want bold maybe near burn taste like starbuck easy stomach course way grind fine grind frankly fine ground coffee espresso go easy way acidic easy digestive system rougher grind

Review 19195:

like cocoa butter make excellent chocolate definately order good quality ship quickly happy overall

Review 8298:

herbal tea cup bedtime help rest well love discount order month supply

Review 8824:

good cup todate watery like try join subscribe save group

Review 19262:

standard poodle pomeranian wonderful food switch different food price couple time end go right natural balance

==========

Rating: 2 | Showing 10 sample reviews

==========

Review 22655:

consumer report rat cracker high inorganic arsenic rice cracker market like cracker long eat arsenic issue brown rice resolve

Review 7273:

product great add egg water butter oil waffle scratch

Review 2122:

star wasn unpleasant coffee didn taste like remind hot water folger coffee bag dunk little bit don like strong coffee way underpowered coffee enjoy bold coffee coffee people organic bold dark magic buy nantucket

Review 13291:

product give great energy boost crashing calorie well tasting drink diet problem taste

Review 15609:

husband bring bottle home restaurant cali pork chop think good marinade meat make great dipping sauce

Review 14557:

product delicious upfront thou semi messy place peanut butter center instead taste great great start eat day good ruin enjoyment actually good stop eat guess value excellent snack friend love family shipping fast amazon prime membership day package box total contain individually wrap granola thin granola bar flat peanut butter seriously ruin granola eat day notice avoid mess open scissor pealing make mess exclusively

Review 575:

dog hurt neck pain pill day godsend think treat whine hide rest mind 26 apiece price buy large quantity need everyday use relieve product dog like pill capsule highly recommend

Review 2159:

arm hammer bake soda aluminum free don write packaging wasn sure call say aluminum state people product internally lead believe isn aluminum bake soda research show past anymore long time resent bob red mill write aluminum free package charge premium product arsenic free hope label arsenic free assume brand aren don like kind marketing

Review 24529:

buy january get review hear people rave japanese snack call pocky figure try like description state biscuit stick cover chocolate wouldn taste mind boggle easy eat virtually mess long chocolate doesn melt anyway case wonder order seller name asia amazon com gp help seller glance html ie utf8 isamazonfulfille orderid 103 7071944 5595443 marketplaceseller seller a3jw437i54sei1 believe take week actually maybe import come packed damage whatsoever definitely consider stock future find local store edit oct 11 2012 inquire type pocky seller finalize purchase decide restock couldn fight craving anymore unfortunately batch time good order different seller cheap amazon prime receive picture show page pack japanese pocky pack pocky ish thing thailand opinion giant pack compare small pack definitely step packaging fine order october weather bit cool aka melted chocolate arrive door stick coat type thick chocolate bit bitter taste similar dark chocolate milk chocolate original japanese pocky melt immediately mouth melt slowly sit mouth personally like taste dark chocolate pocky flavor terrible self control buy thing come small portion order eat avoid make fridge mess clutter ton open item honestly win overeat come batch pocky stomach dislike win change original rating hold true japanese pocky rate thailand pocky taste isn terrible definitely liking possible upload picture receive

Review 8115:

enjoy new coffee maker try different type coffee wasn favorite figure louisiana strong wasn go try look favorite

==========

Rating: 3 | Showing 10 sample reviews

==========

Review 902:

love pack flavor love able pick one want choice available flavor favorite caramel hazelnut vanilla wouldn mind get order possible

Review 4512:

soup lack key ingredient can clam chowder great partially hydrogenate vegetable oil soy cottonseed dehydrate potato modify starch kind hydrolyze corn gluten msg potassium chloride sodium phosphate disodium inosinate disodium guanylate succinic acid good classic new england clam chowder classic ingredient instead ingredient list clam ll tell get butter oh note bar harbor make good taste clam

chowder list ingredient come snow campbell condense new england clam chowder bar harbor stuff worth

Review 8850:

find scotland fall love bit disappointed find type isn gluten free order time delicious bit soft cheese

Review 9823:

look alternative meat source backpack trip husband take soon discover great taste vegan meat jerky taste great bit salty tastebud

Review 23214:

son love chick chick gobble gobble super salmon change recipe corn taste bad nutrition value low doesn like totally pissed

Review 4225:

love jell bake cheesecake disappoint shelf life short get christmas expiration date april 2011 know order order 12 box lot use month order

Review 12267:

highly recommend bar actually get end like tend low blood sugar attack carry food case hungry surprised bar taste good stay hunger away great deal time wish know purchase locally

Review 2069:

love keurig brewer purchase highly price tea coffee cup know need cost efficient solution purchase cup unit allow use regular can coffee go tea try open tea bag tea weak flat try tea work perfect word caution remove tea leaf filter moist removal easy brew entire teapot quick morning cup waiting tea bag brew taste great weak taste perfect tea consumption increase point use amazon auto ship receive box time run keurig love tea product perfect solution win disappoint

Review 19840:

frustrated hook product local walmart stop carry find local store go amazon good exactly advertise arrive good time

Review 1322:

buy online site amazon one cheap love fact ve grow addicted unfortunatley ton sodium contain msg cause gain lb water weight stall weight loss progress effectively try maintain wouldn problem win buy reason

==========

Rating: 4 | Showing 10 sample reviews

==========

Review 7331:

labs aren know picky eater gross understatement know wow zuke ll list toxic stuff

Review 10230:

chip delicious deliver quickly perfect condition order wish option choose flavor want

Review 5413:

hear popchip think end taste like kettle cook chip cape cod style chip wrong light airy great taste note maker popchip need well quality control randomly bag chip lack well description ll hard majority batch buy salt pepper variety flavor love nice kick heat want stay away flavor stick favorite original bottle water nearby definitely recommend lover chip

Review 5530:

lot reviewer describe basic sort low end decent tea think go strong bitter brewing temperature high don let water boil husband drink tea year ve try expensive tea teavana come favorite brew right tangy smoky complexity unbeatable sick drink tea awhile tired drink day good hot iced nice find amazon bulk

Review 3026:

juice drinker daughter buy try wasn expect happy announce like essentially carbonated fruit juice nice different flavor kid hit house ll definitely buy

Review 2454:

fussy dog owner read label annoy like ingredient think star product lot owner dog certainly like flavor aren fussy crew feel good biscuit reason hold star dog greyhound sheltie senior hard treat comfortable treat hard break piece soak water couple minute help dog genetically tartar prone tooth extraction year dachshund work crunch size remember couldn break hardness manage medium dog good tooth highly recommend constantly look soft treat ideal need little soak

Review 7805:

coffee delicious cool latin flavor excellent star coffee hope amazon rid coffee hard find local supermarket hpe get low price packet coffee bag 49 packet good deal amazon hurray

Review 16779:

jack russell terrier quirki dog world try give half treat wouldn eat wouldn let let sight crunch grass spend 15 minute eat look crumb dog foodie biscuit good think half biscuit big small dog crunch

Review 11154:

sucker caramel love flavor aroma important taste honestly like flavor coffee morning exception

Review 3480:

delicious healthy plus amazon cheap buy mrs may actual website

==========

Rating: 5 | Showing 10 sample reviews

==========

Review 9332:

favorite breakfast coffee smooth bitterness cup strong cup starbuck

Review 7815:

love pasta cook little firm cook bit long pesto good husband tell difference know good fuss

Review 8428:

great low carb high protein snack 140 calorie 34 gram oz package like small bag chip vend machine tasty filling good provide real chrunch wife popular low carb package diet plan use lot shake soup etc plan work great 40 far miss crunch bite solid like chip taco shell pretzel stick satisfy craving let diet well order second batch soon good

Review 5343:

love use syrup cut cost calorie latte home use almond milk 35 calorie sugar syrup nice look forward morning drink cup coffee day make nice reason bed nice variety

Review 21301:

love cooky aren grainy moiste tasty easy enjoy extra chip wouldn need good

Review 10396:

star approximately 50 puff little cheese star one lot maybe eat michael season puff curl year help notice production shift flavor light flavor write company mixed flavor produce batch light batch flavor mix expect notice wife love pick light flavor puff leave flavor create kind bonding experience eat annoy contain partially organic ingredient taste overall brand beat order

Review 20782:

product arrive predict coffee pod coffee good flavor aroma isn rich regular coffee

Review 6224:

red mill gluten free oat bran fulfil expectation excellent product shall definitely order

Review 6799:

get box malher hilachas style sauce mother set aside rainy day today rain mix package water add rice pepper onion chicken potato let simmer entire family love picky grandson want find

Review 21347:

coffee write home honestly couldn distinguish generic canister brand local grocery store

9. Train–Test Split

The train–test split is an essential step for assessing how well the model generalizes to new, unseen data. In this stage, the dataset is separated into two parts:

1. Training Set: Used for model learning and parameter optimization, generally covering 70% of the data.

2. Testing Set: Used strictly for performance evaluation on unseen reviews, making up the remaining 30%.

Configuration Details

• test_size: Indicates the portion of data allocated for testing (0.30 was used in this project).

• random_state: Sets a fixed seed value to ensure the split remains consistent across multiple runs.

• shuffle: Randomizes the dataset prior to splitting to prevent any ordering bias; enabled by default.

• stratify: Ensures equal representation of all rating categories in both sets, which is especially critical when working with previously imbalanced ratings.

This approach guarantees that the model is trained effectively while still being tested fairly on data it has never encountered.

```
# splitting to target and feature
x = df_filtered['Text'] #(f)
y = df_filtered['Score'] #(t)
x.head()
# Splitting into Train and Test
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)
```

10. Vectorization

Once the reviews were cleaned and lemmatized, they were transformed into numerical form using

TF-IDF vectorization. This method assigns weights to words based on how frequently they appear

within a review while considering how rare they are across the entire dataset. As a result, the feature matrix focuses on words that carry meaningful sentiment information rather than simply frequent terms.

10.1 TF-IDF (Term Frequency–Inverse Document Frequency)

TF-IDF is a numerical statistic used to reflect how important a word is to a document in a collection (corpus).

It reduces the weight of commonly occurring words and increases the weight of rare but informative terms.

Equation

$$TF\text{-}IDF(t,d)=TF(t,d)\times IDF(t)$$

10.2 Why TF-IDF Instead of Bag of Words

While both TF-IDF and Bag of Words (BoW) convert text into numerical vectors, TF-IDF was chosen due to its ability to differentiate between common and informative words.

Key Reasons for Choosing TF-IDF

• It down-weights common words that may not hold specific sentiment value.

• It highlights unique, impactful terms that distinguish one rating level from another.

• TF-IDF produces a more informative and discriminative feature set, which improves model accuracy.

• Compared to BoW, TF-IDF reduces noise, especially in long review texts where filler words repeat often.

10.2 Summary

• Bag of Words: counts words but treats all with equal importance.

• **TF-IDF:** assigns higher value to words that matter sentiment-wise, improving classification capability.

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(max_features=10000, ngram_range=(1,2))

x_train_vect = vectorizer.fit_transform(x_train)

x_test_vect = vectorizer.transform(x_test)

print("Training data shape:", x_train_vect.shape)

print("Test data shape:", x_test_vect.shape)
```

Model Building (Imbalanced Dataset)

To analyze real-world performance, models were also trained on the **original imbalanced dataset**, where some ratings occur more frequently than others. Class imbalance reflects practical scenarios but can bias models toward majority classes.

TF-IDF vectorization was used to transform text reviews into numerical features.

Logistic Regression (Imbalanced)

- Linear multi-class classifier
- class_weight='balanced' used to reduce class bias

Performance:

- Train Accuracy: ~0.67
- Test Accuracy: ~0.51

Observation:
The model performs reasonably well despite imbalance. Minority classes show lower recall, but overall generalization is better compared to tree-based models.

Code:

```
from sklearn.linear_model import LogisticRegression
logit = LogisticRegression(
    class_weight='balanced',
    C=0.5,
    solver='lbfgs',
    max_iter=2000
)
logit.fit(x_train_vect,y_train_imbalanced)

y_pred_train = logit.predict(x_train_vect)
y_pred_test = logit.predict(x_test_vect)

from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
print("train:",accuracy_score(y_train_imbalanced,y_pred_train))
print("test:",accuracy_score(y_test_imbalanced,y_pred_test))

print("train:",classification_report(y_train_imbalanced,y_pred_train))
print("test:",classification_report(y_test_imbalanced,y_pred_test))
```

Random Forest Classifier (Imbalanced)

- Ensemble model with class balancing

Performance:

- Train Accuracy: ~1.00
- Test Accuracy: ~0.46

Observation:
Severe overfitting is observed. The model memorizes training data but fails to generalize on unseen reviews due to high-dimensional sparse TF-IDF features.

Code:

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(class_weight='balanced', n_estimators=300,
    max_depth=None,
```

```
    min_samples_split=2,
    min_samples_leaf=1,
    max_features='sqrt',
    bootstrap=True,
    random_state=42,
    n_jobs=-1)
rf.fit(x_train_vect,y_train_imbalanced)

y_pred_train_1 = rf.predict(x_train_vect)
y_pred_test_1 = rf.predict(x_test_vect)

from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
print("train:",accuracy_score(y_train_imbalanced,y_pred_train_1))
print("test:",accuracy_score(y_test_imbalanced,y_pred_test_1))

print("train:",classification_report(y_train_imbalanced,y_pred_train_1))
print("test:",classification_report(y_test_imbalanced,y_pred_test_1))
```

Support Vector Machine (Linear SVM – Imbalanced)

- Margin-based linear classifier with class weighting

Performance:

- Test Accuracy: ~0.48

Observation:
Performs better on majority classes but struggles with minority ratings. Sensitive to overlapping review sentiments.

Code:

```
from sklearn.svm import LinearSVC
svm_model = LinearSVC(class_weight='balanced')
svm_model.fit(x_train_vect, y_train_imbalanced)

y_pred_svm = svm_model.predict(x_test_vect)
```

print("Accuracy:", accuracy_score(y_test_imbalanced, y_pred_svm))

print("\nClassification Report:\n", classification_report(y_test_imbalanced, y_pred_svm))

Naive Bayes (Imbalanced)

- Probabilistic text classifier assuming feature independence

Performance:

- Train Accuracy: ~0.59
- Test Accuracy: ~0.48

Observation:
Simple and fast, but performance is limited due to strong independence assumptions. Works better for extreme ratings than neutral ones.

Code:

from sklearn.naive_bayes import MultinomialNB

nb = MultinomialNB()

nb.fit(x_train_vect, y_train_imbalanced)

 # Predictions

y_pred_train_nb = nb.predict(x_train_vect)

y_pred_test_nb = nb.predict(x_test_vect)

 # Accuracy

print("Train Accuracy:", accuracy_score(y_train_imbalanced, y_pred_train_nb))

print("Test Accuracy:", accuracy_score(y_test_imbalanced, y_pred_test_nb))

 # Classification Report

print("\nClassification Report (Train):\n")

print(classification_report(y_train_imbalanced, y_pred_train_nb))

print("\nClassification Report (Test):\n")

print(classification_report(y_test_imbalanced, y_pred_test_nb))

Model Comparison (Imbalanced)

Model	Train Acc	Test Acc	Overfitting
Logistic Regression	~0.67	~0.51	Low
Random Forest	~1.00	~0.46	Very High
Linear SVM	—	~0.48	Moderate
Naive Bayes	~0.59	~0.48	Low

Conclusion (Imbalanced Dataset)

Logistic Regression achieved the most stable performance on imbalanced data. Random Forest suffered from heavy overfitting, while Linear SVM and Naive Bayes showed moderate but consistent results. Overall, **linear models handled class imbalance better than tree-based approaches**.