• **Product Reviews Prediction with Multinomial Naive Bayes**

This project focuses on Product Review Sentiment Prediction using Multinomial Naive Bayes. The aim is to classify product reviews as positive or negative by analyzing the text content of reviews.

• **Import Libraries**

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
```

• **Import Dataset**

```
df = pd.read_csv("product_reviews.csv")
print(df.head())
```

```
 ID.   SIZE.  TITLE.   RATING. REVIEW
101.   M.     Shirt.      4.       Positive
102.   L.     T-shirt.    1.       Negative
103.   S.     Jeans.      2.       Negative
104.   XL.    Top.        5.       Positive
```

•**Missing Values**

```
print(df.isnull().sum())  # check
df.dropna(inplace=True)   # drop
missing values if any
```

•**Define Target and Features**

1–2 = Negative

4–5 = Positive

3 = Neutral (can be dropped for binary classification)

• Train **Test Split**

```
X_train, X_test, y_train, y_test =
train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

• **Feature Extraction and Model Training**

```
vectorizer =
TfidfVectorizer(stop_words='english')
X_train_vec =
vectorizer.fit_transform(X_train)
X_test_vec =
vectorizer.transform(X_test)

model = MultinomialNB()
model.fit(X_train_vec, y_train)
```

• **Model Prediction**

```
y_pred = model.predict(X_test_vec)
print("Predictions:", y_pred)

# Evaluation
print("Accuracy:",
accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n",
confusion_matrix(y_test, y_pred))
print("Classification Report:\n",
classification_report(y_test, y_pred))
```

• **Recognize Ratings**

```
def recategorize(rating):
    if rating <= 2:
        return "Negative"
    elif rating >= 4:
        return "Positive"
    else:
        return "Neutral"

df['sentiment'] =
df['rating'].apply(recategorize)

# Drop neutral for simplicity
df = df[df['sentiment'] != 'Neutral']

X = df['review']   # features (text)
y = df['sentiment']  # target
```

• **Test Custom Reviews**

```
sample = ["The product quality is excellent and very useful"]
sample_vec = vectorizer.transform(sample)
print("Custom Review Prediction:", model.predict(sample_vec)[0])
```

•**Conclusion**

In this project, we successfully built a Product Review Sentiment Prediction model using Multinomial Naive Bayes. The model classifies reviews as Positive or Negative with good accuracy. This demonstrates that Naive Bayes is simple yet effective for text classification tasks.