|  |  |
| --- | --- |
| EX. NO: 1 | **WEEK 1 - A**  **CLASSIFY EMAIL USING NAVIE BAYES ALGORITHM** |
| 02-12-24 |  |

**AIM:**

To write a program to classify the email as spam or not spam using Navie Bayes Algorithm.

**CODE:**

import pandas as pd

from sklearn.feature\_extraction.text import CountVectorizer

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

from sklearn.naive\_bayes import MultinomialNB

from sklearn.metrics import accuracy\_score, classification\_report

# Sample data

data = {

"message": [

"Hi John, just checking in to see if we’re still on for lunch tomorrow. Let me know what time works for you!",

"Limited time offer: Get 50% off your next purchase. Act now before it's too late!",

"Dear friend, we noticed your account has been inactive. Please verify your details by logging in here.",

"Hey, are you free to chat later today? Let me know when you're available!",

"Get your free trial today! Don't miss out on the exclusive deal for premium services!",

"Reminder: Your dentist appointment is at 3:00 PM tomorrow. Let us know if you need to reschedule.",

"You’ve been selected for a special offer. Click here to claim your reward!",

"Good morning! Please find the attached document I mentioned last week.",

"Win an iPhone 15 today! Just pay shipping and handling to receive your prize.",

"Hey Sarah, here’s the updated report I was talking about. Let me know if you have any questions.",

"Free cruise tickets for you and a guest! Just fill out a quick survey and you're all set.",

"Looking forward to our meeting this Friday. Please let me know if you need any further information.",

"Exclusive offer! Get a 30% discount on all luxury watches. Shop now before they’re gone!",

"It was great seeing you last weekend. Hope you’re doing well!"

],

"label": [

"personal", "promotion", "spam", "personal", "promotion", "reminder", "spam", "work",

"spam", "work", "spam", "work", "promotion", "personal"

]

}

# Create DataFrame

df = pd.DataFrame(data)

# Split the data

X\_train, X\_test, y\_train, y\_test = train\_test\_split(df['message'], df['label'], test\_size=0.2, random\_state=42)

# Transform text data to feature vectors

vectorizer = CountVectorizer()

X\_train\_vectorized = vectorizer.fit\_transform(X\_train)

X\_test\_vectorized = vectorizer.transform(X\_test)

# Train the model

model = MultinomialNB()

model.fit(X\_train\_vectorized, y\_train)

# Make predictions

y\_pred = model.predict(X\_test\_vectorized)

# Calculate accuracy

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

print(f'Accuracy: {accuracy}')

# Classification report

report = classification\_report(y\_test, y\_pred)

print(f'Classification Report: \n{report}')

# Predict function for new emails

def predict\_spam(email):

email\_vectorized = vectorizer.transform([email])

prediction = model.predict(email\_vectorized)

return prediction[0]

# Test the function with a new email

new\_email = "Hi John, just checking in to see if we’re still on for lunch tomorrow. Let me know what time works for you!"

result = predict\_spam(new\_email)

print(f"New Email: '{new\_email}'\nPrediction: {result}")

OUTPUT:

Accuracy: 0.3333333333333333

Classification Report:

precision recall f1-score support

personal 0.33 1.00 0.50 1

work 0.00 0.00 0.00 2

accuracy 0.33 3

macro avg 0.17 0.50 0.25 3

weighted avg 0.11 0.33 0.17 3

New Email: 'Hi John, just checking in to see if we’re still on for lunch tomorrow. Let me know what time works for you!'

**Prediction: personal**

**RESULT:**

Thus, the program is run successfully executed.