Name	Hatim Sawai
UID No.	2021300108
Experiment No.	4

Experiment 4

Aim	Classification using suitable classification model (NB).

1. Installation of NLTK and downloading the required corpus

In [ ]:
```
import re
import numpy as np
import pandas as pd
from collections import defaultdict
from sklearn.model_selection import train_test_split
```

In [ ]:
```
import warnings
warnings.filterwarnings("ignore")
```

2. Preprocessing

In [ ]:
```
def preprocess(text):
    text = text.lower()
    text = re.sub(r'[^\w\s]', '', text) # remove punctuation
    text = text.replace("\n", " ") # remove \n
    text = re.sub(r'\W', ' ', text)  # Remove non-word characters
    text = re.sub(r'\s+', ' ', text).strip()  # Remove extra whitespaces
    return text
```

In [ ]:
```
category_mapping = {
    0: "Politics",
    1: "Sport",
    2: "Technology",
    3: "Entertainment",
    4: "Business",
}
```

3. Making the model

In [ ]:
```
class NaiveBayesClassifier:
    def __init__(self):
        self.class_word_counts = defaultdict(lambda: defaultdict(int))
        self.class_counts = defaultdict(int)
        self.vocab = set()

    def train(self, X, y):
        for i in range(len(X)):
            text = X.iloc[i]
            label = y.iloc[i]
            self.class_counts[label] += 1
            words = text.split()
            for word in words:
                self.class_word_counts[label][word] += 1
                self.vocab.add(word)

    def predict(self, X):
        predictions = []
        probabilities = []
        for i in range(len(X)):
            text = X.iloc[i]
            max_score = float("-inf")
            best_class = None
            words = text.split()
            class_probs = {}
            for label in self.class_counts.keys():
                score = np.log(
                    self.class_counts[label] / sum(self.class_counts.values())
                )
                for word in words:
                    count = self.class_word_counts[label][word] + 1
                    total_count = len(self.vocab) + sum(
                        self.class_word_counts[label].values()
                    )
                    score += np.log(count / total_count)
                class_probs[label] = score
                if score > max_score:
                    max_score = score
                    best_class = label
            predictions.append(best_class)
            probabilities.append(class_probs)
        return predictions, probabilities
```

4. Model Evaluation

In [ ]:
```
# Load and preprocess the dataset
data = pd.read_csv("./Dataset/df_file.csv")
data["Text"] = data["Text"].apply(preprocess)
```

In [ ]:
```
# Split the dataset into training and testing sets
train_data, test_data = train_test_split(data, test_size=0.2, random_state=42)
```

In [ ]:
```
# Train the Naive Bayes classifier
classifier = NaiveBayesClassifier()
classifier.train(train_data["Text"], train_data["label"])
```

In [ ]:
```
# Testing the classifier
predictions, probabilities = classifier.predict(test_data["Text"])
```

In [ ]:
```
# Evaluate the classifier based on: Accuracy, Precision, Recall, F1 Score
true_labels = test_data["label"]
accuracy = sum(predictions == true_labels) / len(true_labels)
print(f"Accuracy: {accuracy:.4f}")

confusion_matrix = np.zeros((5, 5))
for i in range(len(true_labels)):
    confusion_matrix[true_labels.iloc[i], predictions[i]] += 1
precision = np.zeros(5)
recall = np.zeros(5)
f1 = np.zeros(5)
for i in range(5):
    precision[i] = confusion_matrix[i, i] / sum(confusion_matrix[:, i])
    recall[i] = confusion_matrix[i, i] / sum(confusion_matrix[i, :])
    f1[i] = 2 * precision[i] * recall[i] / (precision[i] + recall[i])

precision = np.mean(precision)
recall = np.mean(recall)
f1 = np.mean(f1)
```

```
Accuracy: 0.9708
```

In [ ]:
```
def predict_single(sentence):
    sentence = sentence.lower().strip()
    prediction, probability = classifier.predict(pd.Series([sentence]))
    print("Probabilities:")
    for label, prob in probability[0].items():
        print(f"{label}: {np.exp(prob)}")
    print("Predicted Class:", category_mapping[prediction[0]])
```

5. Results

In [ ]:
```
# Print the results
print(f"Accuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1 Score: {f1}")
print("Confusion Matrix:")
print(conf_matrix)
```

```
Accuracy: 0.9707865168539326
Precision: 0.9693088273733436
Recall: 0.9706189736139932
F1 Score: 0.9698990525261447
Confusion Matrix:
[[89  0  1  0  2]
 [ 0 98  0  0  0]
 [ 1  0 74  2  0]
 [ 1  0  1 74  0]
 [ 2  0  2  1 97]]
```

6. Predict a single sentence

In [ ]:
```
# Cell 10: Test prediction on a single input sentence
input_sentence = "The company has announced a new product launch."
predict_single(input_sentence)
```

```
Probabilities:
3: 1.1273390387023854e-26
4: 2.2940067653534148e-24
1: 1.099906355643834e-27
0: 3.074579541547951e-27
2: 1.7965457825652816e-25
Predicted Class: Business
```

In [ ]:
```
input_sentence = "The government and the Army are in conflict."
predict_single(input_sentence)
```

```
Probabilities:
3: 4.53257744586028e-24
4: 1.279231085662771e-22
1: 1.444676989561495e-24
0: 6.2252662487369875e-22
2: 3.7607383704056025e-23
Predicted Class: Politics
```

In [ ]:
```
input_sentence = "The football match was amazing."
predict_single(input_sentence)
```

```
Probabilities:
3: 6.109187318151378e-19
4: 1.1082028232645113e-18
1: 3.5351599308067194e-16
0: 7.626843714808113e-19
2: 5.960515007472776e-19
Predicted Class: Sport
```

6. Curiosity Questions

Q1. What is the relation between accuracy and precision?

Ans: Accuracy measures how close the result is to the actual value you were trying to achieve. Precision measures how close your results are to one another. While accuracy can be used in one instance, precision will be measured over time.
Mathematically:
Accuracy is the ratio of correctly predicted observation to the total observations. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

Formulae as per confusion matrix:
Accuracy = (TP+TN)/(TP+FP+FN+TN)
Precision = TP/(TP+FP
Where,
TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative

Q2. Give example where precision is significant compared to accuracy?

Ans: In the case of spam detection, precision is more important than accuracy. If the precision is low, then the user will get a lot of spam messages in the inbox. But if the accuracy is low, then the user will get some spam messages in the inbox, but not as much as in the case of low precision.

Q3. Give example where accuracy is significant compared to precision?

Ans: In the case of cancer detection, accuracy is more important than precision. If the accuracy is low, then the patient will not be diagnosed with cancer, but if the precision is low, then the patient will be diagnosed with cancer, but it may not be true.

6. Conclusion

In this experiment we learned about the classification using Naive Bayes model. We also learned about the difference between accuracy and precision and their significance in

different scenarios.