Week 4 – PRACTICAL

Sentiment Classification using IMDB Dataset

Introduction

This practical builds on your previous work in text preprocessing and feature extraction. You will now train and evaluate classical machine learning models, Naïve Bayes and Logistic Regression to classify text based on sentiment. You will also interpret key evaluation metrics and reflect on bias and fairness in text classification.

The dataset used is a subset of the *IMDB movie reviews dataset*, labelled as positive or negative.

Tools Overview

- **pandas** – data loading and manipulation
- **scikit-learn** – text vectorisation (`TfidfVectorizer`) and machine learning models
- **matplotlib / seaborn** – evaluation visualisation
- **numpy** – array handling
- **Dataset**: `*imdb_reviews.csv*` (provided in NILE)
- Columns:
  - `review` → text of the movie review
  - `sentiment` → class label (positive / negative)

Part A – Core Tasks (Complete in Class)

Q1). Load and Inspect the Dataset

```
import pandas as pd
data = pd.read_csv("imdb_reviews.csv")
data.head()
data['sentiment'].value_counts()
```

Reflection:

- Is the dataset balanced between positive and negative reviews?
- Why is balance important for classification?

Q2). Preprocess the Text

Preview the first few rows. Then, proceed to clean text by removing HTML tags, punctuation, numbers, and converting to lowercase

```
import re

def clean_text(text):
    text = re.sub(r'<.?>', '', text)
    text = re.sub(r'[^a-zA-Z]', ' ', text)
    return text.lower()

data['clean_review'] = data['review'].apply(clean_text)
```

Reflection:

- Remember how regex is used for filtering out html tags
- Would removing all numbers always be a good idea?
- What if the numbers carried important meaning?
- How would you adjust the regex to keep numbers?

Q3). Split the Data

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    data['clean_review'], data['sentiment'], test_size=0.2, random_state=42
)
```

Q4). Feature Extraction (TF-IDF)

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(stop_words='english', max_features=7000)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
```

Think:

- Which terms do you expect to have the highest TF-IDF weights in positive or negative reviews?

Q5). Train a Naïve Bayes Classifier

```
from sklearn.naive_bayes import MultinomialNB
nb = MultinomialNB()
nb.fit(X_train_tfidf, y_train)
nb_pred = nb.predict(X_test_tfidf)
```

Q6). Train a Logistic Regression Classifier

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(max_iter=2000)
lr.fit(X_train_tfidf, y_train)
lr_pred = lr.predict(X_test_tfidf)
```

Q7). Evaluate Both Models

```
from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

def evaluate_model(name, y_true, y_pred):
    print(f"\n{name}:\n")
    print(classification_report(y_true, y_pred))
    sns.heatmap(confusion_matrix(y_true, y_pred), annot=True, fmt='d', cmap='Blues')
    plt.title(f"{name} – Confusion Matrix")
    plt.show()

evaluate_model("Naïve Bayes", y_test, nb_pred)
evaluate_model("Logistic Regression", y_test, lr_pred)
```

Reflection:

- Which model performs better in terms of precision, recall, and F1-score?
- Why might Logistic Regression outperform Naïve Bayes?

🟦 Part B – Extended Tasks

Q8). Train a Support Vector Machine

```
from sklearn.svm import LinearSVC
svm = LinearSVC()
svm.fit(X_train_tfidf, y_train)
svm_pred = svm.predict(X_test_tfidf)
evaluate_model("Support Vector Machine", y_test, svm_pred)
```

- Does SVM give similar or better results than Logistic Regression?
- Why might that be?

Q7). Compare All Models

Create a short table comparing accuracy, precision, recall, and F1 for all three models.

Example Structure:

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
Naïve Bayes				
Logistic Regression				
SVM				

Q8). Bias & Fairness Reflection

Spam and sentiment classifiers often misclassify content written in dialect or informal tone.

Discuss briefly:

a) How could this happen in your model?
b) How would you detect or reduce such bias in future experiments?

Q9). Extended Reflection Questions

a) What does it mean that Naïve Bayes is a generative model and Logistic Regression is discriminative?
b) Why is TF-IDF more informative than a simple Bag-of-Words count?
c) What trade-offs exist between model simplicity and accuracy?
d) When is F1-score more reliable than accuracy?
e) How does this experiment connect to the NLP pipeline discussed in Week 3?