In [21]:
```
!pip install pandas nltk sklearn
!pip install scikit-learn
```

```
Requirement already satisfied: pandas in c:\users\info\anaconda3\lib\site-packages (1.5.3)
Requirement already satisfied: nltk in c:\users\info\anaconda3\lib\site-packages (3.7)
Requirement already satisfied: sklearn in c:\users\info\anaconda3\lib\site-packages (0.0.post1)
Requirement already satisfied: numpy>=1.21.0 in c:\users\info\anaconda3\lib\site-packages (from pandas) (1.23.5)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\info\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\info\anaconda3\lib\site-packages (from pandas) (2022.7)
Requirement already satisfied: regex>=2021.8.3 in c:\users\info\anaconda3\lib\site-packages (from nltk) (2022.7.9)
Requirement already satisfied: click in c:\users\info\anaconda3\lib\site-packages (from nltk) (8.0.4)
Requirement already satisfied: tqdm in c:\users\info\anaconda3\lib\site-packages (from nltk) (4.64.1)
Requirement already satisfied: joblib in c:\users\info\anaconda3\lib\site-packages (from nltk) (1.1.1)
Requirement already satisfied: six>=1.5 in c:\users\info\anaconda3\lib\site-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Requirement already satisfied: colorama in c:\users\info\anaconda3\lib\site-packages (from click->nltk) (0.4.6)
Requirement already satisfied: scikit-learn in c:\users\info\anaconda3\lib\site-packages (1.2.1)
Requirement already satisfied: scipy>=1.3.2 in c:\users\info\anaconda3\lib\site-packages (from scikit-learn) (1.10.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\info\anaconda3\lib\site-packages (from scikit-learn) (2.2.0)
Requirement already satisfied: numpy>=1.17.3 in c:\users\info\anaconda3\lib\site-packages (from scikit-learn) (1.23.5)
Requirement already satisfied: joblib>=1.1.1 in c:\users\info\anaconda3\lib\site-packages (from scikit-learn) (1.1.1)
```

In [ ]:
```
import pandas as pd
file_path = "Downloads/BBC News.csv"
data = pd.read_csv(Downloads/BBC News.csv)
```

In [ ]:
```
import pandas as pd
import nltk
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score

file_path = ('Downloads/BBCnews.csv', 'r')
data = pd.read_csv(file_path)
stop_words = set(nltk.corpus.stopwords.words('english'))
lemmatizer = nltk.stem.WordNetLemmatizer()
def preprocess_text(text):
    text = text.lower()
    tokens = nltk.word_tokenize(text)
    tokens = [token for token in tokens if token not in stop_words and token.isalpha()]
    tokens = [lemmatizer.lemmatize(token) for token in tokens]
    text = " ".join(tokens)
    return text
import pandas as pd

data = pd.DataFrame({'id': [1, 2, 3], 'category': ['tech', 'business', 'sport'], 'text': ['...', '...', '...']})
data['Processed_Article'] = data['Article'].apply(preprocess_text)
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(data['Processed_Article'])
y = data['Category']
train_size = int(0.8 * len(data))
X_train = X[:train_size]
X_test = X[train_size:]
y_train = y[:train_size]
y_test = y[train_size:]
clf = MultinomialNB()
clf.fit(X_train, y_train)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

In [ ]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix

# Load data
#data = pd.read_csv('heart_disease_uci.csv')
file_path = 'Downloads/BBC News.csv'
data = pd.read_csv(file_path)

# Preprocessing
X = data.drop('target', axis=1)
y = data['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Regression and Classification
lr = LogisticRegression()
dt = DecisionTreeClassifier()
rf = RandomForestClassifier()

lr.fit(X_train, y_train)
dt.fit(X_train, y_train)
rf.fit(X_train, y_train)

# Prediction and evaluation
lr_pred = lr.predict(X_test)
dt_pred = dt.predict(X_test)
rf_pred = rf.predict(X_test)

print('Logistic Regression Accuracy:', accuracy_score(y_test, lr_pred))
print('Decision Tree Accuracy:', accuracy_score(y_test, dt_pred))
print('Random Forest Accuracy:', accuracy_score(y_test, rf_pred))

print('Confusion Matrix for Logistic Regression:')
print(confusion_matrix(y_test, lr_pred))
```

In [ ]:

In [ ]:

In [ ]: