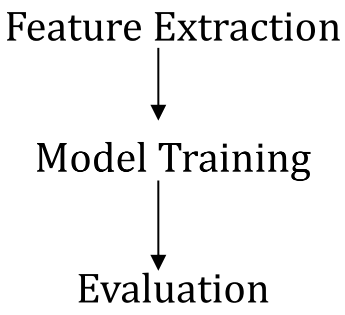
Fake News Detection Using NLP

Phase – 4

In this Phase we did this three steps



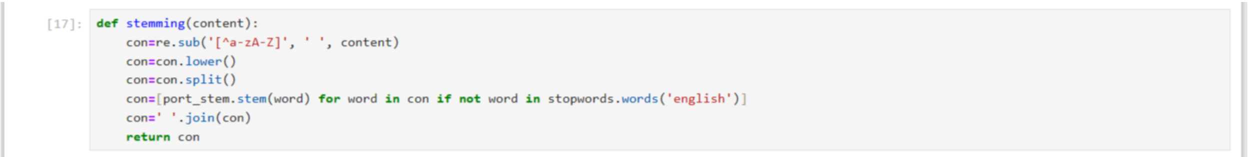
Feature Extraction

Model Training

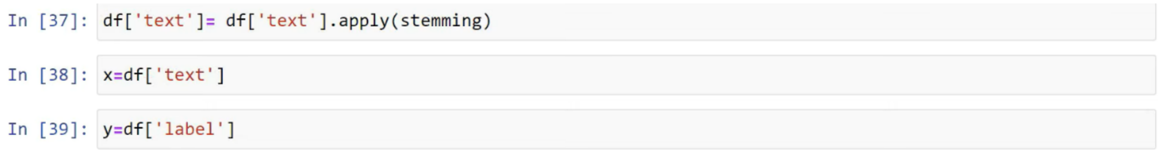
Evaluation

Feature Extraction:

It involves tasks such as removing non-alphabetic characters, converting text to lowercase, splitting it into words, removing **stopwords**, and applying **stemming**. These preprocessing steps are essential for preparing text data for classification or other NLP tasks..



```
[17]: def stemming(content):
          con=re.sub('[^a-zA-Z]', ' ', content)
          con=con.lower()
          con=con.split()
          con=[port_stem.stem(word) for word in con if not word in stopwords.words('english')]
          con=' '.join(con)
          return con
```



```
In [37]: df['text']= df['text'].apply(stemming)

In [38]: x=df['text']

In [39]: y=df['label']
```

Model Training:

Training the Model

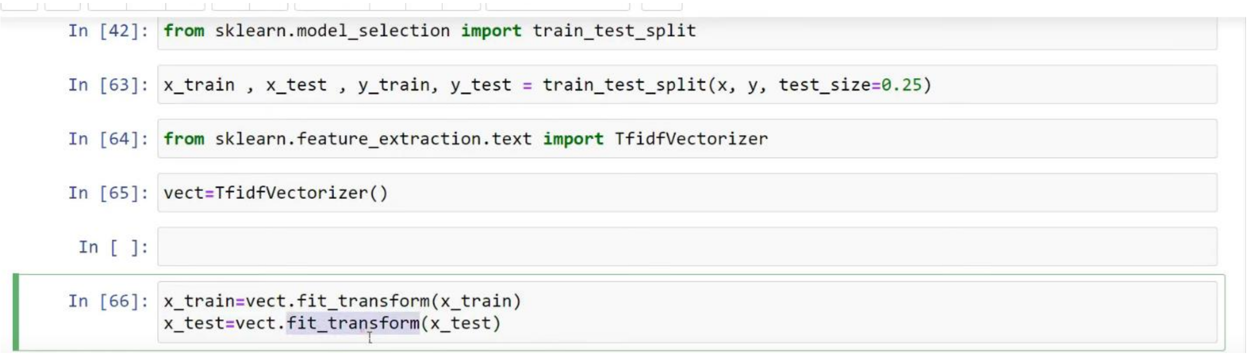
1. **train_test_split(x, y, test_size=0.25)**: This function is used to split your dataset x and corresponding labels y into training and testing sets. The test_size parameter specifies the proportion of the dataset that should be allocated to the testing set (in this case, 25% of the data). It returns x_train, x_test, y_train, and y_test.

2. **from sklearn.feature_extraction.text import TfidfVectorizer**: This line imports the **TfidfVectorizer** class from scikit-learn (a popular machine learning library). **TfidfVectorizer** is used for text data preprocessing and feature extraction, particularly for converting text into numerical features using the TF-IDF (Term Frequency-Inverse Document Frequency) method.

3. **vect = TfidfVectorizer():** This line creates an instance of the TfidfVectorizer class. vect will be used to transform your text data into TF-IDF features.

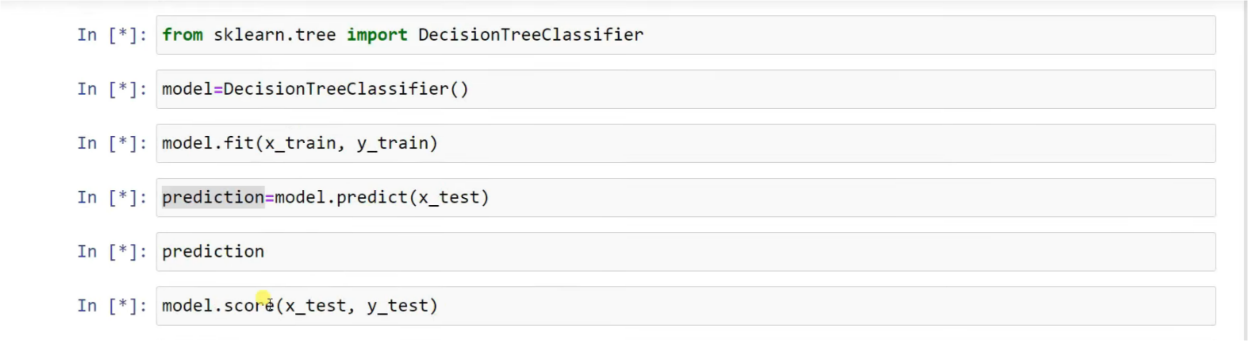
4. **x_train = vect.fit_transform(x_train):** It applies the TF-IDF transformation to the training data x_train, converting the text data into numerical features. This will replace the original x_train with the transformed data.

5. **x_test = vect.fit_transform(x_test):** Similarly, it applies the TF-IDF transformation to the testing data x_test, converting it into numerical features. However, this line should be corrected to use transform instead of fit_transform. So, it should be x_test = vect.transform(x_test) to ensure that the testing data is transformed using the same TF-IDF settings as the training data.



```
In [42]: from sklearn.model_selection import train_test_split

In [63]: x_train , x_test , y_train, y_test = train_test_split(x, y, test_size=0.25)

In [64]: from sklearn.feature_extraction.text import TfidfVectorizer

In [65]: vect=TfidfVectorizer()

In [ ]:

In [66]: x_train=vect.fit_transform(x_train)
         x_test=vect.fit_transform(x_test)
```

Evaluation:



```
In [*]: from sklearn.tree import DecisionTreeClassifier

In [*]: model=DecisionTreeClassifier()

In [*]: model.fit(x_train, y_train)

In [*]: prediction=model.predict(x_test)

In [*]: prediction

In [*]: model.score(x_test, y_test)
```

1. **from sklearn.tree import DecisionTreeClassifier**: This line imports the Decision Tree Classifier class from scikit-learn. A decision tree classifier is a type of machine learning model that makes decisions by learning simple decision rules inferred from the training data.

2. **model = DecisionTreeClassifier():** This creates an instance of the Decision Tree Classifier model.

3. **model.fit(x_train, y_train):** It trains the Decision Tree Classifier using the training data x_train and corresponding labels y_train. The model learns to make predictions based on this training data.

4. **prediction = model.predict(x_test):** This line makes predictions on the testing data x_test using the trained decision tree model and stores the predictions in the prediction variable.

5. **model.score(x_test, y_test):** This calculates the accuracy of the trained model on the testing data. It compares the model's predictions (based on x_test) to the actual labels y_test and returns the accuracy score, which represents the proportion of correctly classified instances in the testing data.

Team:

MOHAMED ABDUL RAHMAN S	(au951221104023)
RAHUL B	(au951221104034)
NAGA MARI MUTHU A	(au951221104028)
SAM SELIN D	(au951221104039)
YABESH JESILEN V	(au951221104059)