SLIDE 2:

SHIFA

Social media has changed how people get news, making it quick and easy. But it also spreads "fake news" — news that's intentionally wrong. This fake news can trick people into believing things that aren't true, affecting how they think and even causing panic during crises like the COVID-19 pandemic.

Detecting fake news on social media is a big challenge because it doesn't always look obviously fake in the articles themselves, thanks to our editing technologies.

It's hard to tell real news from fake news online these days. This project aims to build a tool that can automatically decide if a news article is true or false based on what it says. This tool will help people avoid falling for false information online.

By giving people a way to check if news is true, this project hopes to make it easier for everyone to trust the information they see online and make better decisions.

SLIDE 3:

SHIFA

In this project, we have utilized essential libraries to streamline data handling, text preprocessing, model development, evaluation, and visualization. numpy and pandas were essential for loading and cleaning the dataset (train.csv).

For text preprocessing, nltk provided tools for tokenization, stemming with PorterStemmer, and removing stopwords.

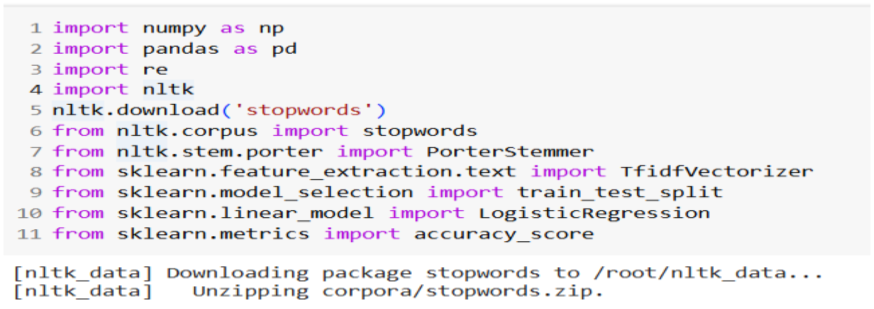
For model training and evaluation, we used sklearn for Confusion Matrix , accuracy score.

For visualization, matplotlib and seaborn enabled creating visualizations like count plots and confusion matrices to analyze data distribution and model performance.

Let’s look at the code now.

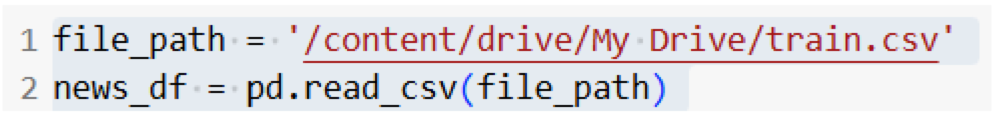
(Move on the Collab) – don’t say.

SOFIA



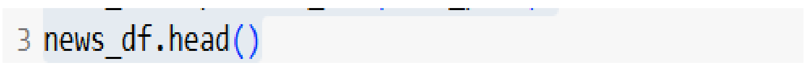
```
import numpy as np
import pandas as pd
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

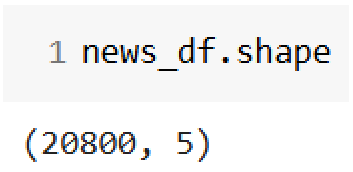


```
file_path = '/content/drive/My Drive/train.csv'
news_df = pd.read_csv(file_path)
```

Firstly, we imported necessary libraries such as NumPy, Pandas, NLTK, and scikit-learn. Our dataset is loaded from Google Drive from the file train.csv. Let's dive into the initial exploration.



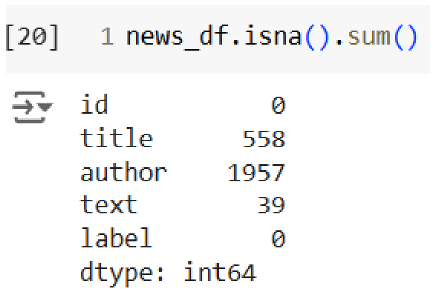
```
news_df.head()
```



```
news_df.shape
```

```
(20800, 5)
```

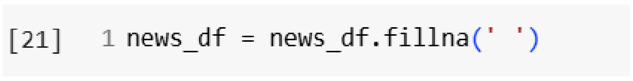
We began by loading the dataset and examining its structure. We printed the first few rows to get an overview of the fields and content. The dataset consists of 20800 rows and 5 columns, giving us a view of its size.



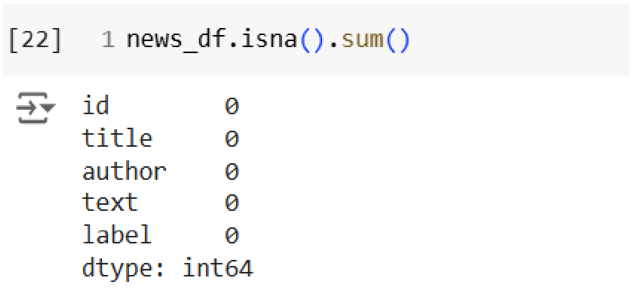
```
[20]  news_df.isna().sum()
```

```
id           0
title      558
author    1957
text        39
label        0
dtype: int64
```

To ensure our dataset is clean and ready for analysis, we check for missing values. And we have found a few rows that have null values. So, we will be removing that using ‘fillna’.



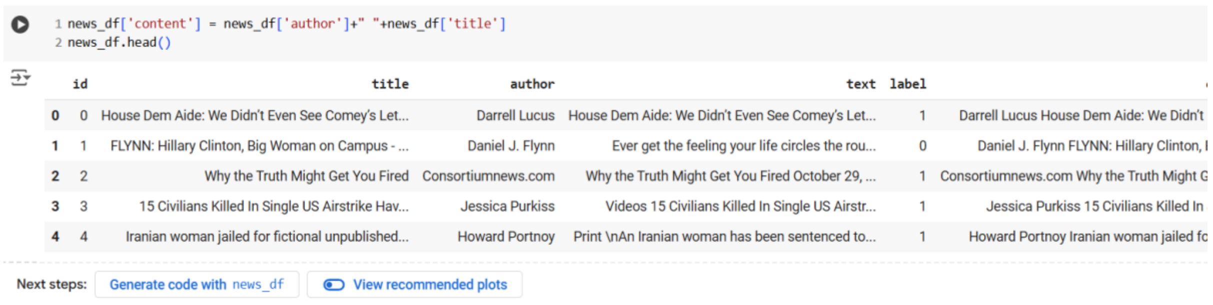
```
[21]  news_df = news_df.fillna(' ')
```



```
[22]  news_df.isna().sum()
```

```
id        0
title     0
author    0
text      0
label     0
dtype: int64
```

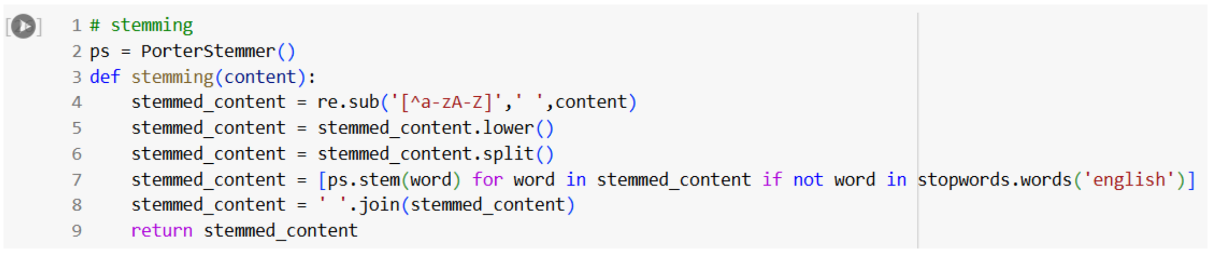
We fill empty cells with blank spaces to maintain data integrity. And now we have no rows with null values.



```python
news_df['content'] = news_df['author']+" "+news_df['title']
news_df.head()
```

	id	title	author	text	label	
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's Let...	1	Darrell Lucus House Dem Aide: We Didn't
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0	Daniel J. Flynn FLYNN: Hillary Clinton, I
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1	Consortiumnews.com Why the Truth Might G
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...	1	Jessica Purkiss 15 Civilians Killed In
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1	Howard Portnoy Iranian woman jailed fc

Next, we merge the author and title fields into a single content column, which is a crucial step in preparing our text data for analysis.



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

Using NLTK's Porter stemming algorithm, we standardize the text by converting it to lowercase, removing non-alphabetic characters, and stemming words to their root form.

Additionally, we removed common English stopwords to focus on the most meaningful words for classification purposes.

ps = PorterStemmer() : Stemming reduces words to their root form by removing suffixes. For example, words like "running", "runs", and "ran" would all be reduced to "run"

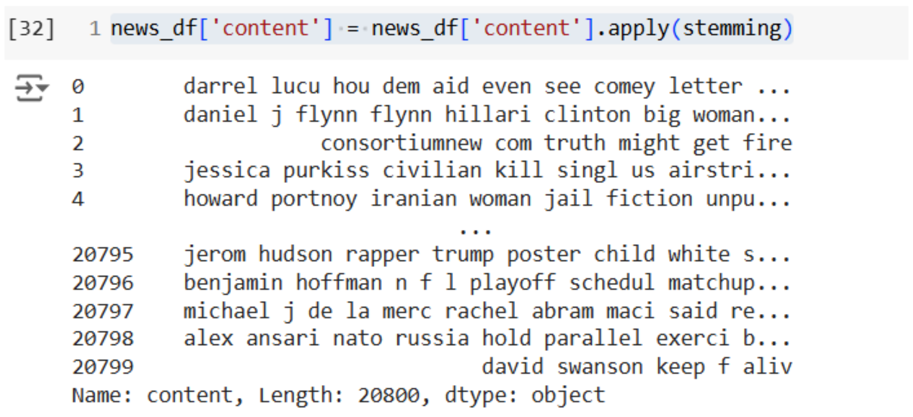
stemmed_content = re.sub('[^a-zA-Z]',' ',content) : This helps in cleaning the text by removing punctuation, numbers, and other non-alphabetical characters that are not useful for analysis

stemmed_content = stemmed_content.lower() : Converts the string to lowercase.

stemmed_content = stemmed_content.split() : Splits into a list of words. By doing this, we prepare it for further processing where each word can be examined separately.

stemmed_content = [ps.stem(word) for word in stemmed_content if not word in stopwords.words('english')] : Stopwords are excluded to focus on meaningful words that convey more semantic information.

stemmed_content = ' '.join(stemmed_content) : After stemming and removing stopwords, this step concatenates the list of processed words back into a single string



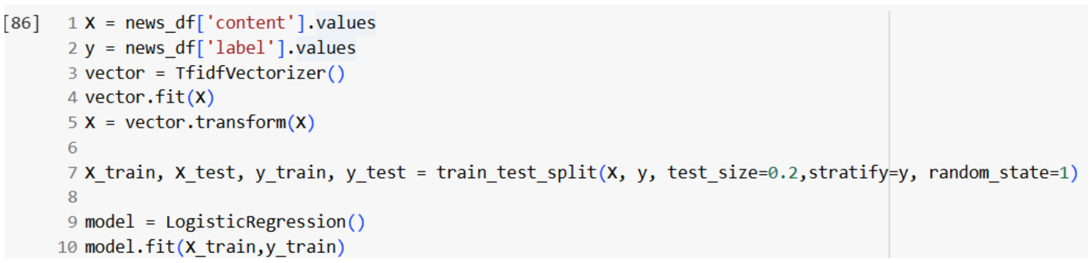
```
[32] news_df['content'] = news_df['content'].apply(stemming)
```

```
0        darrel lucu hou dem aid even see comey letter ...
1        daniel j flynn flynn hillari clinton big woman...
2                   consortiumnew com truth might get fire
3        jessica purkiss civilian kill singl us airstri...
4        howard portnoy iranian woman jail fiction unpu...
                                ...
20795    jerom hudson rapper trump poster child white s...
20796    benjamin hoffman n f l playoff schedul matchup...
20797    michael j de la merc rachel abram maci said re...
20798    alex ansari nato russia hold parallel exerci b...
20799                           david swanson keep f aliv
Name: content, Length: 20800, dtype: object
```

Let see how our data looks after preprocessing. Here we can see the effects of stemming and stopwords removal.

This concludes our initial preprocessing steps. Stay tuned as we move forward to vectorizing our text data and training our machine learning model

UMAAMA



```
[86]
X = news_df['content'].values
y = news_df['label'].values
vector = TfidfVectorizer()
vector.fit(X)
X = vector.transform(X)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,stratify=y, random_state=1)

model = LogisticRegression()
model.fit(X_train,y_train)
```

- X and y are created to hold the features (text content) and labels (news labels) from news_df.
- TfidfVectorizer (vector) is initialized to convert text data into TF-IDF vectors, which quantify the importance of words in documents relative to the entire dataset.
- vector.fit(X) learns the vocabulary and IDF from X.
- X = vector.transform(X) : transforms the text data into TF-IDF encoded sparse matrix X.
- train_test_split : splits the data into training and testing sets for model validation. It reserves 20% of the data for testing , ensures class balance I.e, ensures that the train and test sets have the same proportion of classes as the original dataset y (stratify=y), and sets a random seed.
- LogisticRegression (model) : is initialized for binary classification based on TF-IDF transformed data.
- model.fit(X_train, y_train) trains the logistic regression model on the training data.

```
[87]  # Distribution of labels
      plt.figure(figsize=(6,4))
      sns.countplot(x='label', data=news_df)
      plt.title('Distribution of Real and Fake News')
      plt.xlabel('Label')
      plt.ylabel('Count')
      plt.show()
```

Now for visualisation of the observations, let's use `countplot` which is available in seaborn.

`sns.countplot(x='label', data=news_df)` uses `seaborn`'s `countplot` function to create a bar plot of counts for each category in the `'label'` column of the `news_df` DataFrame.

So, we have almost equal number of Real and Fake news in the dataset.

Let's find out the accuracy of our model.

```
[88]  train_y_pred = model.predict(X_train)
      print("Train accurracy :",accuracy_score(train_y_pred,y_train)*100)
```

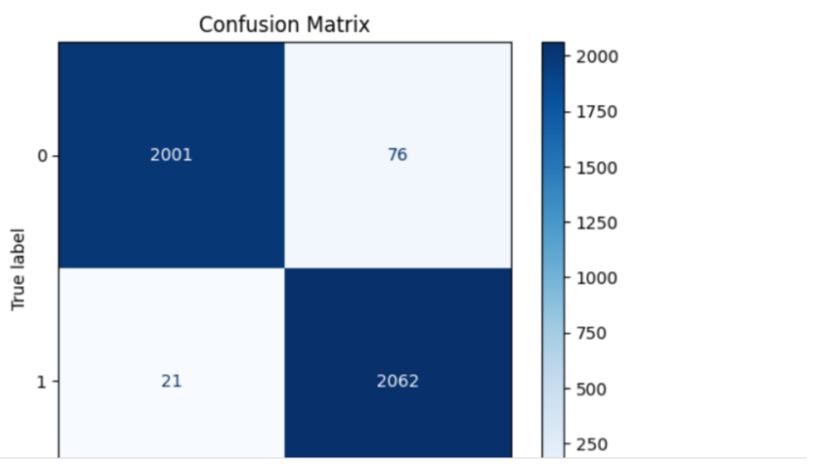
```
Train accurracy : 98.68389423076923
```

```
[89]  test_y_pred = model.predict(X_test)
      print("Test accurracy :",accuracy_score(test_y_pred,y_test)*100)
```

```
Test accurracy : 97.66826923076923
```

We use accuracy_score which is there in sklearn library to find the score. We have 96 % accuracy for Training data and 97 % accuracy for Testing.

```
[90]  cm = confusion_matrix(y_test, test_y_pred)
      disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=model.classes_)
      disp.plot(cmap=plt.cm.Blues)
      plt.title('Confusion Matrix')
      plt.show()
```



We will now build a confusion matrix to evaluate the classification performance on our testing set.

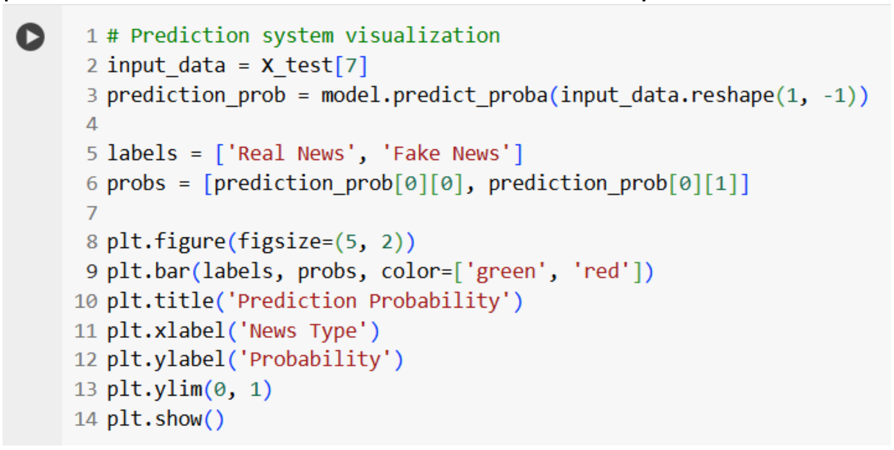
The confusion matrix consists of four basic characteristics that are used to define the measurement metrics of a classifier. They are:

TP (True Positive): True positives are the cases where the model correctly predicts the news article as fake when it is actually fake

TN (True Negative): True negatives are the cases where the model correctly predicts the news article as real when it is actually real.

FP (False Positive): False positives are the cases where the model incorrectly predicts the news article as fake when it is actually real.

FN (False Negative): False negatives are the cases where the model incorrectly predicts the news article as real when it is actually fake



```python
# Prediction system visualization
input_data = X_test[7]
prediction_prob = model.predict_proba(input_data.reshape(1, -1))

labels = ['Real News', 'Fake News']
probs = [prediction_prob[0][0], prediction_prob[0][1]]

plt.figure(figsize=(5, 2))
plt.bar(labels, probs, color=['green', 'red'])
plt.title('Prediction Probability')
plt.xlabel('News Type')
plt.ylabel('Probability')
plt.ylim(0, 1)
plt.show()
```

- Visualizes the prediction probability of a specific news article being classified as 'Real News' or 'Fake News'.
- This visualization helps interpret the model's confidence in predicting whether the selected news article is 'Real News' or 'Fake News'.
- The height of each bar represents the predicted probability of the news article belonging to each category.
- Green color indicates higher probability for 'Real News', while red color indicates higher probability for 'Fake News'.

In conclusion, our project focused on using machine learning techniques to classify news articles as either real or fake. We leveraged a logistic regression model trained on a dataset of labeled news articles, achieving a test accuracy of 97%.

SLIDE 4:

SHIFA

I would like to extend my heartfelt thanks to CFI-EVP for providing me with the opportunity to present this project and gain valuable experience during my internship. Special appreciation goes to Akhil Sir for his engaging teaching style, which made learning both effective and enjoyable. Sincere thanks to Farhath for her support in clearing our doubts and ensuring a comfortable learning experience throughout.

SOFIA

Looking ahead, I am eager to dedicate more time and effort to further explore topics in machine learning and data analysis and I am motivated to continue learning and conducting research to contribute meaningfully in the future.

UMAAMA

Finally a big thanks to everyone for listening attentively to our presentation. Thank you for your time and attention.