Dependencies

- Numpy: for making numpy arrays
- Panda: for creating data frames and storing data in the data frame
- Re: Regular Expression use for searching the words in a text or paragraph
- Nltk.corpus, stopwords: nltk stands of natural language toolkit. corpus means the body of that particular text which is important, stopwords which doesn't add much value to a paragraph vertex (eg. The, a)
- Nltk.stem.porter: we perform a function called as stemming so this stemming takes a word and removes the prefix and suffix of that word and returns root word of it
- Sklearn.feature_extraction.text, TfidVectorizer: for converting text into feature vector. Feature Vectors are nothing but numbers
- Sklearn.model_selection, train_test_split: for spiting our data set into training data and test data.
- Sklearn.linear_model, LogisticRegression
- Sklearn_matrics, accuracy_score

Preprocessing and Cleaning

Data Pre-processing

Loading the dataset to a pandas Data Frame. Pandas Data Frame loads dataset into a more structured table. We create a new variable name as news_dataset, where we store our dataset file which is in .csv (Comma Separated Value) filetype.

```
news_dataset = pd.read_csv('./content/train.csv')
```

Let check the numbers of rows and column of the dataset

```
news_dataset.shape
```

Output

```
(20800, 5)
```

So, we have 20,800 rows and 5 columns which also shows we have 20,800 news articles and 5 features

We are checking first 5 rows of this data frame

```
# print the first 5 rows of the dataframe
news_dataset.head()
```

Output:

	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
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...	1
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1

In above dataset we have:

- Id: unique id for a news article
- Title: the title of a news article
- Author: author of the news article
- Text: the text of the article could be incomplete
- Label: a label that marks whether the news article is real or fake. (0: Real News, 1: Fake News)

Cleaning

Now we check the missing values in dataset and count total number in each column.

```
# counting the number of missing values in the dataset
news_dataset.isnull().sum()
```

Output:

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

From above output we can see there are no missing values in column “id” and “label”. But in title, author and text columns we have 558, 1957, and 39 missing values respectively out of 20,800 news.

While we are preparing the dataset, we might not get the title of a news or author of a particular news or the text of the particular news. So, that is why we have missing value in our dataset.

We can drop these missing values or we can replace it with null string. If more values are missing, we use some methods like imputation (processing to replace those missing values with appropriate values)

For now, we are replacing the missing values with null string by doing,

```
# replacing the null values with empty string
news_dataset = news_dataset.fillna('')
```

Implementing Classification

In our project we are going to include title and author for prediction. Because of that we are going to combine ‘title’ and ‘author’ column together. We are not going to use ‘text’ column because it can be huge paragraph and it takes a lot of time for processing.

Here we create a new column called “content” and store “title” and “author” together.

```
# merging the author name and news title
news_dataset['content'] = news_dataset['author']+' '+news_dataset['title']
```

Output

```
print(news_dataset['content'])
```

```
0        Darrell Lucus House Dem Aide: We Didn’t Even S...
1        Daniel J. Flynn FLYNN: Hillary Clinton, Big Wo...
2        Consortiumnews.com Why the Truth Might Get You...
3        Jessica Purkiss 15 Civilians Killed In Single ...
4        Howard Portnoy Iranian woman jailed for fictio...
                              ...
20795    Jerome Hudson Rapper T.I.: Trump a ’Poster Chi...
20796    Benjamin Hoffman N.F.L. Playoffs: Schedule, Ma...
20797    Michael J. de la Merced and Rachel Abrams Macy...
20798    Alex Ansary NATO, Russia To Hold Parallel Exer...
20799              David Swanson What Keeps the F-35 Alive
Name: content, Length: 20800, dtype: object
```

So now we are going to use this “content” column and “label” column to make predictions.

Now we are separating the “content” column and “label” column. In this case, the date is “content” column and label are “label” column.

```
# separating the data & label
X = news_dataset.drop(columns='label', axis=1)
Y = news_dataset['label']
```

Here, in variable “X” we have dataset without “label” column and in variable “Y” we have only the “label” column.

Output

```
print(X)
print(Y)
```

```
          id  ...                                            content
0          0  ...  Darrell Lucus House Dem Aide: We Didn’t Even S...
1          1  ...  Daniel J. Flynn FLYNN: Hillary Clinton, Big Wo...
2          2  ...  Consortiumnews.com Why the Truth Might Get You...
3          3  ...  Jessica Purkiss 15 Civilians Killed In Single ...
4          4  ...  Howard Portnoy Iranian woman jailed for fictio...
...      ...  ...                                                ...
20795  20795  ...  Jerome Hudson Rapper T.I.: Trump a ’Poster Chi...
20796  20796  ...  Benjamin Hoffman N.F.L. Playoffs: Schedule, Ma...
20797  20797  ...  Michael J. de la Merced and Rachel Abrams Macy...
20798  20798  ...  Alex Ansary NATO, Russia To Hold Parallel Exer...
20799  20799  ...            David Swanson What Keeps the F-35 Alive

[20800 rows x 5 columns]
0        1
1        0
2        1
3        1
4        1
        ..
20795    0
20796    0
20797    0
20798    1
20799    1
Name: label, Length: 20800, dtype: int64
```

Stemming

Stemming is the process of reducing a word to its Root word. Example: actor, actress, acting --> act. This is very important step because we need to reduce words as much as possible to have better performance on our model.

```
port_stem = PorterStemmer()
```

We create a function called as “stemming” which will search and store the alphabets and replace all numbers and special character with empty string (‘’) from our “context” column.

Afterward, we convert all the words into lowercase and converted into List using split method. Then we take each word and convert them into there root word (stemming) and if found any stopwords we will remove it. Once we have done all above steps we will join all the words together.

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

We apply above created ‘stemming’ function to our content column

```
news_dataset['content'] = news_dataset['content'].apply(stemming)
```

Output

```
print(news_dataset['content'])
```

```
0        darrel lucu hous 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 exercis ...
20799                            david swanson keep f aliv
Name: content, Length: 20800, dtype: object
```

We are separating the “content” and “label” columns in X and Y respectively, which we are going to feed our machine learning model.

```
#separating the data and label
X = news_dataset['content'].values
Y = news_dataset['label'].values
```

Output

```
print(X)
```

```
['darrel lucu hous dem aid even see comey letter jason chaffetz tweet'
 'daniel j flynn flynn hillari clinton big woman campu breitbart'
 'consortiumnew com truth might get fire' ...
 'michael j de la merc rachel abram maci said receiv takeov approach hudson bay new york time'
 'alex ansari nato russia hold parallel exercis balkan'
 'david swanson keep f aliv']
```

```
print(Y)
```

```
[1 0 1 ... 0 1 1]
```

We can see from above output that our "content" values are still in textual form. But computer cannot understand text so that we need to convert all these text into meaningful numbers that computer understands.

Clustering

So, for this we will be using vectorizer function. Which convert text into feature vector. Feature Vectors are nothing but numbers.

```
# converting the textual data to numerical data
vectorizer = TfidfVectorizer()
vectorizer.fit(X)

X = vectorizer.transform(X)
```

Here, we are using special vectorizer function called TfidfVectorizer, where "Tf" stands for Term Frequency and "idf" stands for inverse document frequency. "Tf" basically counts the number of times a particular word is repeating in a document, text, paragraph. Repetition tells the model that it is a very important word and it assign a particular numerical value to that word. "idf" finds those values or word which are repeating many times and it detects that those words are not significant and it reduce its importance value. So, we are fitting the 'content' column to TfidfVectorizer function.

Output

```
print(X)
```

```
  (0, 15686)    0.28485063562728646
  (0, 13473)    0.2565896679337957
  (0, 8909)     0.3635963806326075
  (0, 8630)     0.29212514087043684
  (0, 7692)     0.24785219520671603
  (0, 7005)     0.21874169089359144
  (0, 4973)     0.233316966909351
  (0, 3792)     0.2705332480845492
  (0, 3600)     0.3598939188262559
  (0, 2959)     0.2468450128533713
  (0, 2483)     0.3676519686797209
  (0, 267)      0.27010124977708766
  (1, 16799)    0.30071745655510157
  (1, 6816)     0.1904660198296849
  (1, 5503)     0.7143299355715573
  (1, 3568)     0.26373768806048464
  (1, 2813)     0.19094574062359204
  (1, 2223)     0.3827320386859759
  (1, 1894)     0.15521974226349364
  (1, 1497)     0.2939891562094648
  (2, 15611)    0.41544962664721613
  (2, 9620)     0.49351492943649944
  (2, 5968)     0.3474613386728292
  (2, 5389)     0.3866530551182615
  (2, 3103)     0.46097489583229645
  :     :
  (20797, 13122)        0.2482526352197606
  (20797, 12344)        0.27263457663336677
  (20797, 12138)        0.24778257724396507
  (20797, 10306)        0.08038079000566466
  (20797, 9588) 0.174553480255222
  (20797, 9518) 0.2954204003420313
```

We can from above outputs that our data is converted in the numerical form from textual form.

Splitting the dataset to training & test data

Now we are splitting our data into two data which are “test” and “strain”. “train” represents training data and “test” represents testing data. We want 80 percentage of data to be training data and 20 percentage of the data to be testing data (test_size = 0.2). These 20 percentages of our data will be store in X_test and rest of 80 percentage will be store in X_train. The label for X_train data will be stored in Y_train and label for X_test data will be stored in Y_test. Variable Y basically contains 0 and 1 which represents real and fake news respectively. By doing Stratify = Y, we segregated in equal proportion, there will be similar proportion as it was in the original dataset. Random_state=2 is for reproduce a particular code in same manner for all. For example, while you are preparing this code if you mention 2 here that data will be splitted in a same way as it is splitting for me. It can be any integer value.

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, stratify=Y, random_state=2)
```

Association Algorithms

Training the Model: Logistic Regression

```
model = LogisticRegression()
```

We now train our model on the basics of obtaining data (X_train and Y_train) and it will plot sigmoid function curve using Logistic Regression.

```
model.fit(X_train, Y_train)
```

Output

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

Data visualization / Evaluation

As it is trained, we need to find its accuracy and other scores. Here, model will be asked to predict values and model's prediction will be compared to the original label values.

Let's check the accuracy score on the training data by making prediction on X_train data and all the prediction will be stored in X_train_prediction variable. For accuracy score, we use predicted data verses original labels.

```
# accuracy score on the training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

Output

```
print('Accuracy score of the training data : ', training_data_accuracy)
```

```
Accuracy score of the training data :  0.9865985576923076
```

So, our accuracy score is 0.9865985576923076 meaning it is 98 percent accurate. This means that our training data is really good.

Now, let's check the accuracy of test data which is the real deal.

```
# accuracy score on the test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

Output

```
print('Accuracy score of the test data : ', test_data_accuracy)
```

```
Accuracy score of the test data :  0.9790865384615385
```

So, our accuracy score is 0.9790865384615385 meaning it is almost 98 percent accurate. This means our model has not overtrained with training data and it is performing very well.

We have successfully train our model and we have also evaluated our model with high accuracy score.

Making a Predictive System

We build a new Predictive System where if you give a new news data to our model it should predict whether the news is real or fake.

Here, X_new is variable were store new news data. Then we will predict whether news is real or fake using our trained model and show us a result.

```
X_new = X_test[3]

prediction = model.predict(X_new)
print(prediction)

if (prediction[0]==0):
  print('The news is Real')
else:
  print('The news is Fake')
```

Output

```
[0]
The news is Real
```

Our model predicts that the new news we input is Real.

While checking the respective column in original label for accuracy I found

```
print(Y_test[3])
```

```
0
```

Which is also true. So, this means that my model is working fine.

Conclusion

As mentioned earlier, Media trust worldwide has dropped by 8% between 2020 and 2021. In 2020, only 29% of US adults said they mostly trust news media. 52% of Americans say they regularly encounter fake news online. 67% of US adults say they've come across false information on social media. To reduce that different approach are can be made and of the approach with Machine Learning with Logistic Regression model is the one we choose. This project would make a very persuasive argument for the implementation of a full-scale data mining and analysis.

References

1. https://letter.ly/fake-news-statistics/
2. Data Mining: Concepts and Techniques, 3rd ed. Jiawei Han, Micheline Kamber, and Jian Pei. Morgan Kaufmann Series in Data Management Systems Morgan Kaufmann Publishers, July 2011.
3. Introduction to Data Mining, 2nd ed. Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar. Pearson Publisher, 2019.
4. Mining of Massive Datasets by Jure Leskovec, Anand Rajaraman, Jeff rey D. Ullman, 2014.
5. https://www.youtube.com/watch?v=nacLBdyG6jE