Phase II: INNOVATION

This documentary mainly focuses on how the fake news is being detected through various methods in machine learning algorithm.

Why fake news detector and how ML plays an important role in it

I. **Need of a detector:**
Nowadays due to the increase of various technological impacts a lot of people use it in a wrong way to misuse someone, to corrupt their privilege and to abuse them by spreading rumors through social media platforms. In order to avoid this, we have developed the program in such a way that this finds out the unwanted or fake content about the person.

II. **What does ML do:**

Machine Learning Techniques have shown promising results in detecting fake news with the help of analysing vast amounts of data, in which it identifies patterns and it provides outcomes that are based on those patterns.

Steps invloved in ML:

1.Importing libraries and datasets:

This step invloves installation of the required libraries.

2.Data preprocessing:

In this step,we fetch the dataset and start to remove unwanted columns and null values.The first step is we read the dataset and next the unwanted values are removed which makes the processing simple and also neat.

3.preprocessing and analysis of new column:

Once when the step 2 has ended the required information is viewed and the datas are splitted into dependent and independent variables.

4.converting text to vectors:

This convertion step plays an important role for developing various preprocessing measures and for implementing various techniques It's necessary to proceed with this step so that both programmer and the user could understand the entire concept.

5.Modelling:

in this program we have concentrated mainly in the following five methods:

a) **Logistic regression:**This is a supervised machine learning algorithm which mainly focuses on predicting the probability for the targetted value.
   As this provides a result of 98.76%accuracy,we have given importance for this in our program.

b) **Decision Tree Classifier:**Decision tree algorithm mainly focuses on grabbing the majority of possibility for any particular dataset.

We have used this as this classifier understands the pattern of the news pattern and provides the correct output whether the news is fake or a correct one.

c)Gradient Boost: This provides a stronger result by combining and undersatnding the framwork of the dataset. As this provides an accurate value we have included this as a major part in our code.

d)Random forest: This is mainly used for predicting and classifing the data.As this is an ensemble learning technique it has the capability of finding the desired result by combing various possiblities from the decision tree classifier.

Source code:

```
#importing the libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
import re
import string

#reading the dataset
dataframe_fake = pd. read_csv("Fake.csv")
dataframe_true = pd. read_csv("True.csv")
dataframe_fake. head()
dataframe_true. head()
```

```
dataframe_true["class"] = 0
dataframe_true["class"] = 1

 #shape of both the dataset
dataframe_fake. shape, dataframe_true. shape

# We will remove the last 10 rows for manual testing
dataframe_fakedataframe_fake_manual_testing = dataframe_fake. tail(10)
for i in range(23480,23470,-1):
dataframe_fake. drop([i], axis = 0, inplace = True)
dataframe_truedataframe_true_manual_testing = dataframe_true. tail(10)
for i in range(21416,21406,-1):
dataframe_true.drop([i], axis = 0, inplace = True)
# Let's have a look at the change in the shape of both the dataset
dataframe_fake.shape, dataframe_true.shape
#Inserting the class column in both of the manual testing datasets
 dataframe_fake_manual_testing["class"] = 0
 dataframe_true_manual_testing["class"] = 1
 dataframe_fake_manual_testing.head(10)
 dataframe_true_manual_testing.head(10)
 dataframe_merge = pd.concat([dataframe_fake, dataframe_true], axis =0 )
 dataframe_merge. head(10)
 # We will remove the columns that are required for us
 dataframe = dataframe_merge.drop(["title", "subject","date"], axis = 1)
 # Let's check if there are any null values in the dataset
 dataframe.isnull().sum()
# Here is the random shuffling of the rows in dataset
 dataframedataframe = dataframe.sample(frac = 1)
 dataframe.head()
 def wordopt(t):
  tt = t.lower()
  t = re.sub('\[.*?\]', '', t)
  t = re.sub("\\W"," ",t)
  t = re.sub('https?://\S+|www\.\S+', '', t)
  t = re.sub('<.*?>+', '', t)
  t = re.sub('[%s]' % re.escape(string.punctuation), '', t)
  t = re.sub('\n', '', t)
  t = re.sub('\w*\d\w*', '', t)
```

```
 return t
 dataframe["text"] = dataframe["text"].apply(wordopt)
 #Now we will define the dependent variable and independent variables
 x = dataframe["text"]
 y = dataframe["class"]
 # Splitting the Dataset into a Training and Testing Set
 x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25)
 from sklearn.feature_extraction.text import TfidfVectorizer
  vectorization = TfidfVectorizer()
  xv_train = vectorization.fit_transform(x_train)
  xv_test = vectorization.transform(x_test)
  from sklearn.linear_model import LogisticRegression
   LR = LogisticRegression()
  LR.fit(xv_train,y_train)
  pred_lr=LR.predict(xv_test)
  LR.score(xv_test, y_test)
  print(classification_report(y_test, pred_lr))
  from sklearn.tree import DecisionTreeClassifier
   DT = DecisionTreeClassifier()
  DT.fit(xv_train, y_train)
  pred_dt = DT.predict(xv_test)
 DT.score(xv_test, y_test)
 print(classification_report(y_test, pred_dt))
 from sklearn.ensemble import GradientBoostingClassifier
  GBC = GradientBoostingClassifier(random_state=0)
GBC.fit(xv_train, y_train)
 print(classification_report(y_test, pred_gbc))
 from sklearn.ensemble import RandomForestClassifier
 RFC = RandomForestClassifier(random_state=0)
RFC.fit(xv_train, y_train)
pred_rfc = RFC.predict(xv_test)
RFC.score(xv_test, y_test)
print(classification_report(y_test, pred_rfc))
def output_lable(n):
    if n == 0:
        return "Fake News"
    elif n == 1:
```

```
        return "Not A Fake News"

def manual_testing(news):
    testing_news = {"text":[news]}
    new_def_test = pd.DataFrame(testing_news)
    new_def_test["text"] = new_def_test["text"].apply(wordopt)
    new_x_test = new_def_test["text"]
    new_xv_test = vectorization.transform(new_x_test)
    pred_LR = LR.predict(new_xv_test)
    pred_DT = DT.predict(new_xv_test)
    pred_GBC = GBC.predict(new_xv_test)
    pred_RFC = RFC.predict(new_xv_test)


    return print("\n\nLR Prediction: {} \nDT Prediction: {} \nGBC Prediction: {} \nRF
C Prediction: {}".format(output_lable(pred_LR[0]),output_lable(pred_DT[0]),output
_lable(pred_GBC[0]), output_lable(pred_RFC[0])))
news = str(input())
manual_testing(news)
```

Observed result:

The model we have made is producing accurate results, considering the accuracy of all the models, which was almost 99%, so we can say machine learning can be used as a tool for detecting fake news.

Conclusion:

By detecting fake news articles before they are widely disseminated, machine learning algorithms can prevent the harm caused by fake news. However, it is important to use diverse datasets and other techniques, such as fact-checking, to verify the authenticity of news articles.