Journal of
Data Acquisition and Processing

ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.7766118

COMPARATIVE ANALYSIS OF MACHINE LEARNING MODELS FOR FAKE CONTENT DETECTION

Makhan kumbhkar
Research Scholar, School of Computer Science & Information Technology, DAVV, Indore

Shraddha Masih
Associate Professor, SCSIT, DAVV, Indore
makhan@christianeminent.com, shraddhadandel@yahoo.com

Abstract - The cultural impact of fake content is enormous today. Detecting fake content is a crucial step. This research aims to identify fake content using several machine-learning classifiers. Five popular classifiers are used in the experiments: Naïve Bayes, Logistic Regression, Support Vector Machine, Decision Tree and Random Forest. Data cleaning (concatenating the data frame, Shuffling the data, dropping the title and date, converting to lowercase) and pre-processing (Removing punctuation, removing stop words) are the most important methods before using any machine learning classifier. In our research work, different types of models and accuracies were observed. The Decision Tree classifier has gained the highest accuracy, and the Naïve Bayes algorithm has taken very less time to execution.

Keywords - Naive Bayes, Logistic Regression, Decision Tree and SVM classifier.

1. INTRODUCTION

The world is changing day by day. In this era, we have several advantages and disadvantages of this digital world. Internet is the best resource for browsing and downloading information from any web. But we do not know how much legitimate or fake content is one of them. Users can easily spread fake content, which harms a person's or society's reputation. Fake content can be related to a political party or an organization. There are several web platforms where a person can share fake content. There are many social platforms, including Facebook, twitter etc. Machine learning helps in the making, such as automatically learn system and perform different actions. Algorithms for machine learning come in supervised, unsupervised, and reinforcement learning varieties. Now a days machine learning is widely used technology for prediction or detection of hidden pattern. There are a lot of web platforms for users to access and with very little amount easily. But these contents challenge cyber criminals to detect timely and accurately. So fake content detection is not an easy task.it is a big challenge. (wang,2017). Fake content is circulating at high speed on social media. If the fake content is not detected in the primary stage, then the social media user circulates it to others and the other user act on it. It destroys the names and fame of individuals, organizations, or political parties through fake content. To identify phony content, they deployed machine learning algorithms. Fake news, according to researchers (zhou et al., 2019), has been growing over time. Because of this, false content detection is necessary. Now a day's, different machine learning algorithms are trained to fulfil this purpose.

The remaining sections of the paper are organized in the following way: section two incorporates related work, section three contains the methodology, section four contains the results and discussion, and section five incorporates the conclusion. References for the papers covered in this literature review are provided at the end.

2. RELATED WORK

In their study [1], Mykhailo Granik et al. provide a straightforward method for identifying false news using a naive Bayes classifier. This approach collected data from Facebook post as well as three large web platforms (Politico, CNN, ABC News) for implementation of system. They achieved an algorithm accuracy is 74%. Marco L. Della Vedova et. al. [2] proposed a model with combining news content and social features. System increased the accuracy up to 78.8%. They used Facebook Messenger Chabot in real time environment which increased the accuracy 78.8% to 81%. Cody Buntain et. al. [3] developed an automatic fake news detection with CREDBANK and PHEME dataset. They use this technique to analyses Twitter content that was obtained from BuzzFeed's fake news dataset.

3. METHODOLOGY

Methodology is most important part of the any research. we have following to steps classify our fake or legitimate content with good accuracy.

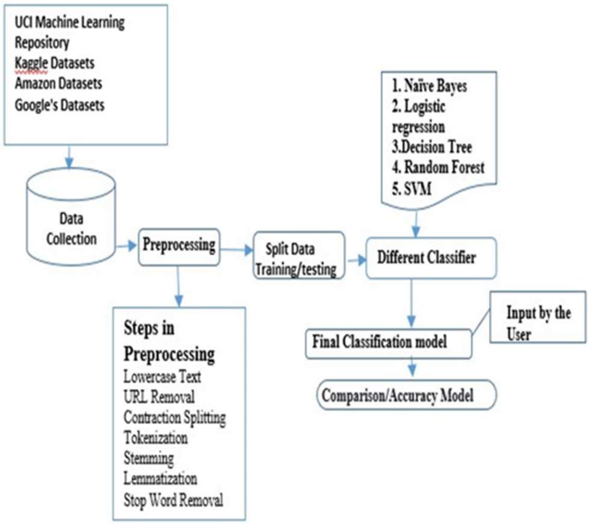
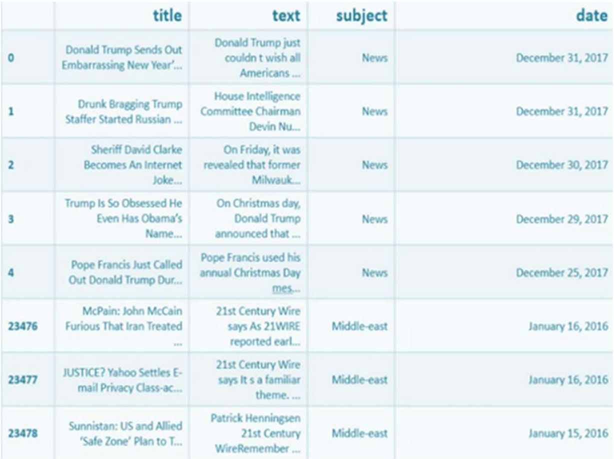


Fig 1: Methodology

3.1 Data Collection

Using an open-source dataset is the simplest and quickest way to gather data for our ML model. We used Kaggle, UCI machine, Amazon and Government datasets for our research. Our data set have different features title, subject, text and date which shows when the content uploaded on social media platform.

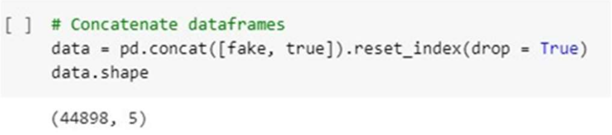


	title	text	subject	date
0	Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017
3	Trump Is So Obsessed He Even Has Obama's Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017
23476	McPain: John McCain Furious That Iran Treated ...	21st Century Wire says As 21WIRE reported earl...	Middle-east	January 16, 2016
23477	JUSTICE? Yahoo Settles E-mail Privacy Class-ac...	21st Century Wire says It s a familiar theme. ...	Middle-east	January 16, 2016
23478	Sunnistan: US and Allied 'Safe Zone' Plan to T...	Patrick Henningsen 21st Century WireRemember ...	Middle-east	January 15, 2016

Fig 2: Dataset Description

3.2 Pre-processing

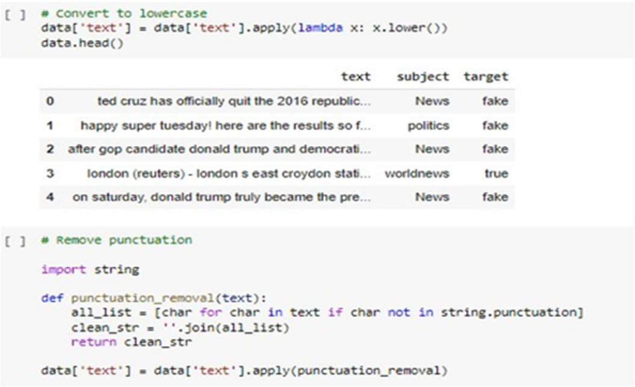
In our research work Data cleaning and preparation comes in the data pre-processing steps can be a challenge especially with the total number of ways at which this process can take place. In this paper we use Missing values, Noisy Data, Removing outliers, Data Transformation and Data Reduction pre-processing methods on the collected data set. We also use Natural Language Toolkit for Removing punctuation, Removing stop words for data cleaning process. We also merge our true and fake dataset into new dataset which shown below



```
[ ]  # Concatenate dataframes
     data = pd.concat([fake, true]).reset_index(drop = True)
     data.shape

     (44898, 5)
```

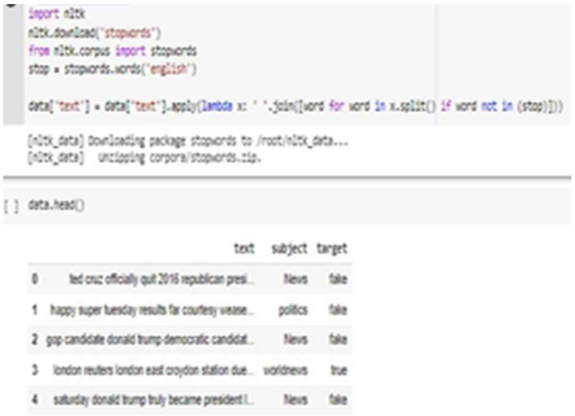
Fig 3: Concatenate data frame

Which have total 44898 records and 5 features. We also shuffle the data, remove the unrelated date feature from the dataset. We also converted data into lowercase, removing punctuation and stop words from dataset. Mentioned images shown practical implementation of some main points related to the pre-processing.



```
[ ]  # Convert to lowercase
     data['text'] = data['text'].apply(lambda x: x.lower())
     data.head()

                                                  text     subject  target
     0        ted cruz has officially quit the 2016 republic...        News    fake
     1     happy super tuesday! here are the results so f...    politics    fake
     2  after gop candidate donald trump and democrati...        News    fake
     3     london (reuters) - london s east croydon stati...   worldnews    true
     4    on saturday, donald trump truly became the pre...        News    fake

[ ]  # Remove punctuation

     import string

     def punctuation_removal(text):
         all_list = [char for char in text if char not in string.punctuation]
         clean_str = ''.join(all_list)
         return clean_str

     data['text'] = data['text'].apply(punctuation_removal)
```

Fig 4: Remove punctuation



```
import nltk
nltk.download("stopwords")
from nltk.corpus import stopwords
stop = stopwords.words('english')

data['text'] = data['text'].apply(lambda x: ' '.join([word for word in x.split() if word not in (stop)]))

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

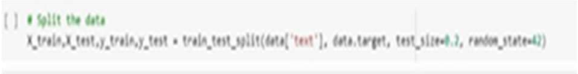
```
[ ] data.head()
```

```
                                             text    subject  target
0         ted cruz officially quit 2016 republican presi...       News    fake
1  happy super tuesday results far courtesy wease...   politics    fake
2  gop candidate donald trump democratic candidat...       News    fake
3  london reuters london east croydon station due...  worldnews    true
4  saturday donald trump truly became president l...       News    fake
```

Fig 5: Removing the title

3.3 Data set Splitting

Train-test split is used to divide the dataset into two parts. We can evaluate the machine learning model's performance with the help of splitting the dataset. For good performance, we give maximum data to the training part and less data to the test to the machine. By default, the training set is divided into 70% of the actual data and the test set into 30% of the actual data. In our study, training accounts for 80% and testing for 20%.



```
[ ] # Split the data
    X_train,X_test,y_train,y_test = train_test_split(data['text'], data.target, test_size=0.2, random_state=42)
```

Fig 6: Splitting data

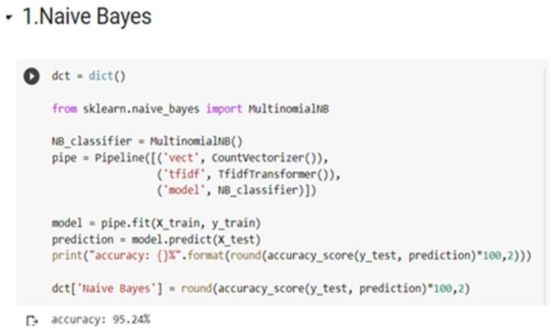
Machine learning models generally divide the dataset into training and testing. We trained our model over an original dataset and then evaluated it for prediction using different classifiers. Training and testing datasets are key concepts of machine learning, where we will use training datasets for fitting the model and test datasets for evaluating the model.

3.4 Algorithm

Machine learning is a technology that automatically sorts or categorizes data into one or more categories using different classifiers. We used to describe classes using target, label and or features of the dataset. The Most important characteristic of fake content is to classify the fake or real content with the help of a machine learning classifier which examines the content and filters it according to dataset attributes. The classifier estimates a mapping function (f) from the given input variable (X) to discrete target variables is known as classification. We used different classifier in our research.

3.4.1 Naïve Bayes Classifier

We used different types of classifiers to train machine. It will automatically detect the fake or legitimate content form dataset. We use the classifier separates data into different classes according to the Bayes' Theorem.it make relationship between all input features in a class is independent. Hence, the model is called naïve. Basically, it used for binary or categorical input values. We classified our target feature with the help of Naïve Bayes classifier. We got 95.24% accuracy with model.



▾ 1.Naive Bayes

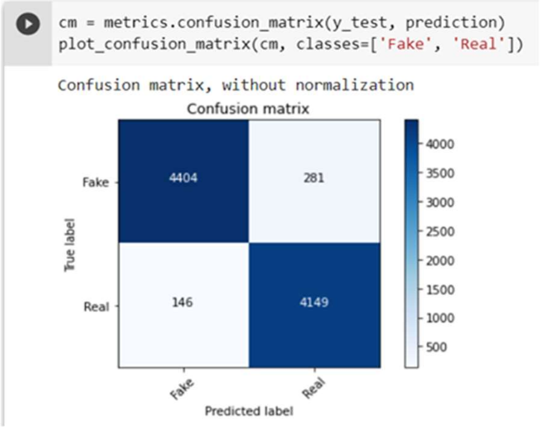
```
dct = dict()

from sklearn.naive_bayes import MultinomialNB

NB_classifier = MultinomialNB()
pipe = Pipeline([('vect', CountVectorizer()),
                 ('tfidf', TfidfTransformer()),
                 ('model', NB_classifier)])

model = pipe.fit(X_train, y_train)
prediction = model.predict(X_test)
print("accuracy: {}%".format(round(accuracy_score(y_test, prediction)*100,2)))

dct['Naive Bayes'] = round(accuracy_score(y_test, prediction)*100,2)
```

```
accuracy: 95.24%
```

Fig 7: Naïve Bayes Classifier

3.4.1.1 Confusion matrix

A machine learning classification performance metric is the confusion matrix. By displaying a data table of the outcomes and predictions, a confusion matrix helps the user visualise the varied results of a classification operation. The whole predicted and actual values of a classifier are plotted in a table. Consider a confusion matrix made for a classifier that classifies content based on whether they fake or legitimate.



```
cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```

Fig 8: Confusion matrix

From the above diagram, we can see that:

True Positives (TP) = 4404

True Negatives (TN) = 4149

False Positives (FP) = 281

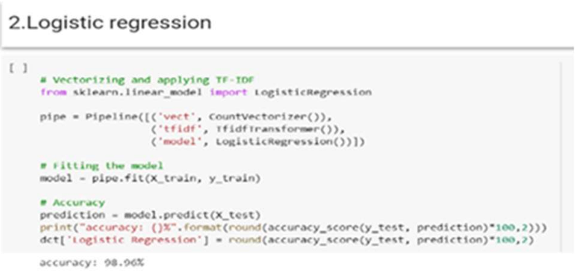
False Negatives (FN) = 146

Accuracy = (TP+TN)/(TP+TN+FP+FN)

Accuracy = 4404+4149/4404+4149+281+146   =8553/8980 =95.24%

3.4.2. Logistic Regression Classifier

Supervised learning includes logistic regression. It's employed to figure out or forecast the likelihood that a binary (yes/no) event will take place. In our research work dependent variable is title. The independent variable is label. We classified the fake and legitimate content with 98.96% accuracy.



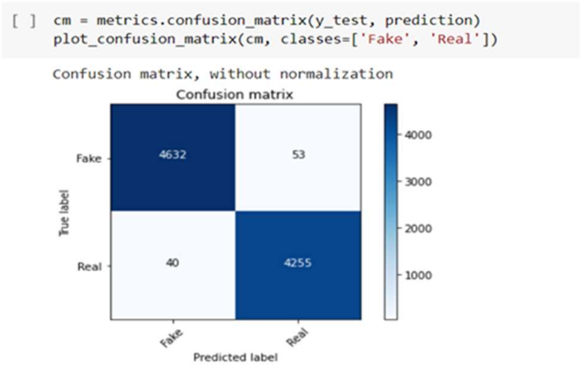
2.Logistic regression

```
[ ]
     # Vectorizing and applying TF-IDF
     from sklearn.linear_model import LogisticRegression

     pipe = Pipeline([('vect', CountVectorizer()),
                      ('tfidf', TfidfTransformer()),
                      ('model', LogisticRegression())])

     # Fitting the model
     model = pipe.fit(X_train, y_train)

     # Accuracy
     prediction = model.predict(X_test)
     print("accuracy: {}%".format(round(accuracy_score(y_test, prediction)*100,2)))
     dct['Logistic Regression'] = round(accuracy_score(y_test, prediction)*100,2)

     accuracy: 98.96%
```

Fig 9: Logistic Regression Classifier

3.4.2.1 Confusion matrix

It provides the measurement of any machine learning model.

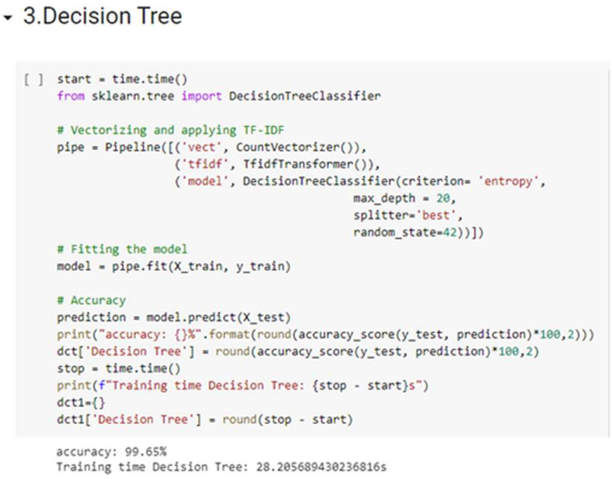


```
[ ]  cm = metrics.confusion_matrix(y_test, prediction)
     plot_confusion_matrix(cm, classes=['Fake', 'Real'])

     Confusion matrix, without normalization
```

Fig 10: Confusion matrix

3.4.2 Decision Tree Classifier

Decision Tree is also coming in the supervised machine-learning technique for different regression and classification problems. Generally, it is preferred for solving classification problems. It is based on a tree structure classifier where internal nodes and branches is part of the structure. Internal nodes represent the features of a dataset, branches represent the decision rules, and each leaf node represents the outcome. Our work also uses to predict fake content and gives a good accuracy of 99.67%.

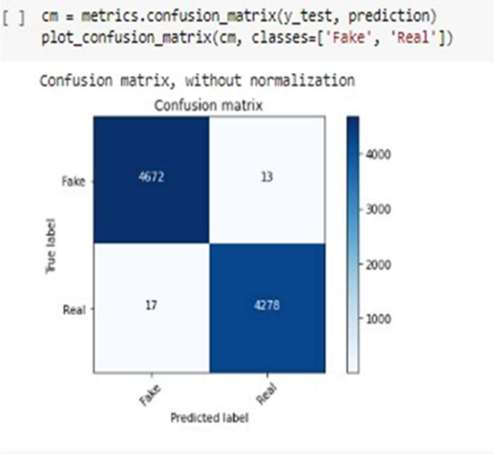


▾ 3.Decision Tree

```
[ ]  start = time.time()
     from sklearn.tree import DecisionTreeClassifier

     # Vectorizing and applying TF-IDF
     pipe = Pipeline([('vect', CountVectorizer()),
                      ('tfidf', TfidfTransformer()),
                      ('model', DecisionTreeClassifier(criterion= 'entropy',
                                                 max_depth = 20,
                                                 splitter='best',
                                                 random_state=42))])
     # Fitting the model
     model = pipe.fit(X_train, y_train)

     # Accuracy
     prediction = model.predict(X_test)
     print("accuracy: {}%".format(round(accuracy_score(y_test, prediction)*100,2)))
     dct['Decision Tree'] = round(accuracy_score(y_test, prediction)*100,2)
     stop = time.time()
     print(f"Training time Decision Tree: {stop - start}s")
     dct1={}
     dct1['Decision Tree'] = round(stop - start)

     accuracy: 99.65%
     Training time Decision Tree: 28.205689430236816s
```

Fig 11: Decision Tree Classifier

3.4.3.1 Confusion matrix



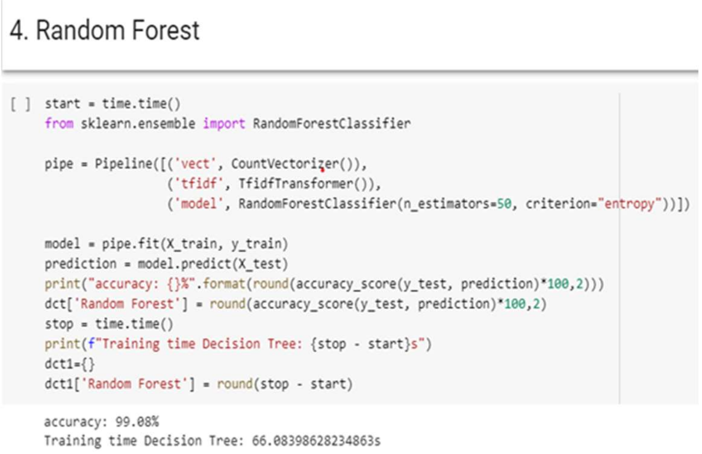
```
[ ] cm = metrics.confusion_matrix(y_test, prediction)
    plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```

Fig 12: Confusion matrix

3.4.4 Random Forest Classifier

It is an important classifier for improving the accuracy of the algorithm. Random forest is a mixture of decision trees. We can develop a tree using a random subset of a training dataset. In the random forest, the test condition represents a node, and the result represents edges. By using Random Forest, we gain a maximum accuracy of 99.14%.



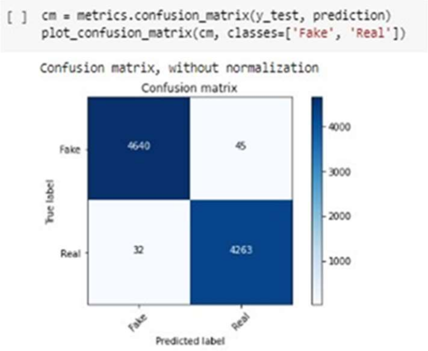
4. Random Forest

```
[ ] start = time.time()
    from sklearn.ensemble import RandomForestClassifier

    pipe = Pipeline([('vect', CountVectorizer()),
                     ('tfidf', TfidfTransformer()),
                     ('model', RandomForestClassifier(n_estimators=50, criterion="entropy"))])

    model = pipe.fit(X_train, y_train)
    prediction = model.predict(X_test)
    print("accuracy: {}%".format(round(accuracy_score(y_test, prediction)*100,2)))
    dct['Random Forest'] = round(accuracy_score(y_test, prediction)*100,2)
    stop = time.time()
    print(f"Training time Decision Tree: {stop - start}s")
    dct1={}
    dct1['Random Forest'] = round(stop - start)
```

```
accuracy: 99.08%
Training time Decision Tree: 66.08398628234863s
```

Fig 13: Random Forest Classifier

3.4.4.1 Confusion matrix



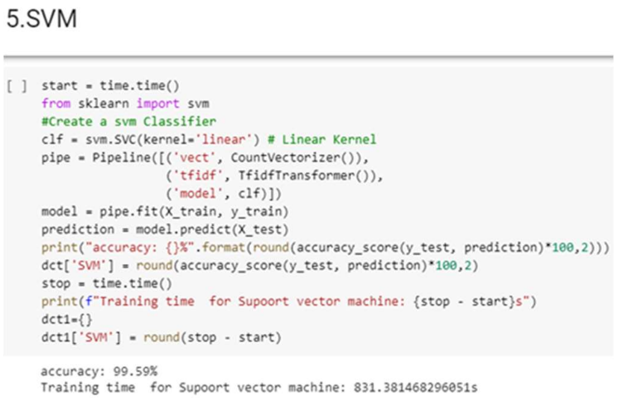
```
[ ] cm = metrics.confusion_matrix(y_test, prediction)
    plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```

Fig 14 : Confusion matrix

3.4.5 Support vector machine

Support vector machine is a discriminative classifier. With the help of hyperplane, we can separate two of more than two classes. The hyperplane divides a line in two parts with the corresponding classes on either side of the hyperplane. SVM give us 99.54% accuracy on given dataset.

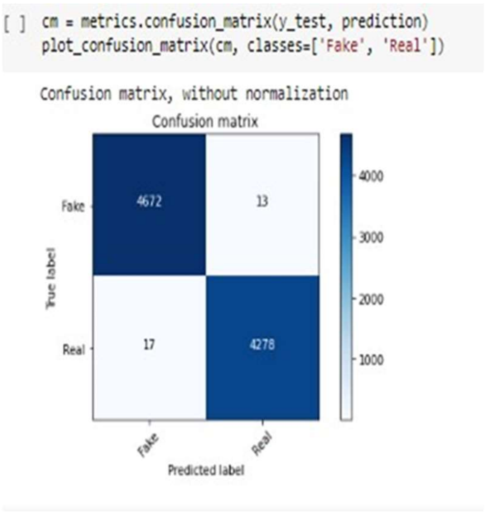


5.SVM

```
[ ] start = time.time()
    from sklearn import svm
    #Create a svm Classifier
    clf = svm.SVC(kernel='linear') # Linear Kernel
    pipe = Pipeline([('vect', CountVectorizer()),
                     ('tfidf', TfidfTransformer()),
                     ('model', clf)])
    model = pipe.fit(X_train, y_train)
    prediction = model.predict(X_test)
    print("accuracy: {}%".format(round(accuracy_score(y_test, prediction)*100,2)))
    dct['SVM'] = round(accuracy_score(y_test, prediction)*100,2)
    stop = time.time()
    print(f"Training time  for Supoort vector machine: {stop - start}s")
    dct1={}
    dct1['SVM'] = round(stop - start)

    accuracy: 99.59%
    Training time  for Supoort vector machine: 831.381468296051s
```

Fig 15: SVM Classifier

3.4.5.1 Confusion Matrix



```
[ ] cm = metrics.confusion_matrix(y_test, prediction)
    plot_confusion_matrix(cm, classes=['Fake', 'Real'])

    Confusion matrix, without normalization
```

Fig 16: Confusion matrix

4. Results and analysis

After, the gathering of the data, different machine-learning models were trained. For solving the problem of fake content, the different machine learning models trained Naïve Bayes, Random Forest, Support vector machine, Logistic regression, and Decision Tree Classifiers were trained to solve the problem of fake or legitimate content for an existing dataset. The dataset consisted of the following features: (i) Title: Content Headline. It can be either a user, a news agency or an organisation. (ii) Text: The body of the content is in detail. (iii) Subject: Types of the content (iv) Target: The classification label either "Real" or "Fake". (v) Date: Content published date. The below table shown different comparative study of the algorithms:

Table 1 : Results and analysis

Sno	Algorithm	Accuracy	Time
1	Naïve Bayes	95.24%	12.22s
2	Logistic regression	98.96%	20.72s
3	Random Forest	99.14%.	51.66s
4	Decision Tree	99.67%.	33.021s
5	Support vector machine	99.54%	1002.464s

4.1 Comparison of classifiers

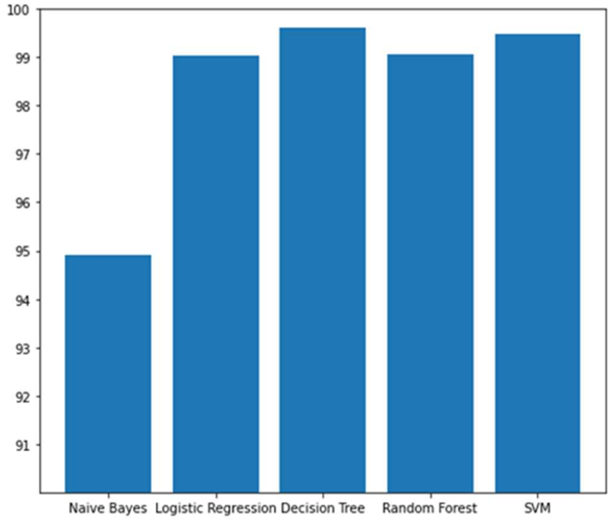
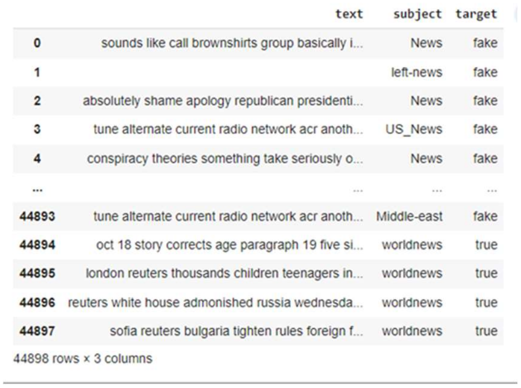


Fig :17 Comparison of classifiers

The above model shows decision tree classifier which gives the best accuracy as compared to other classifier.

4.2. Dataset view



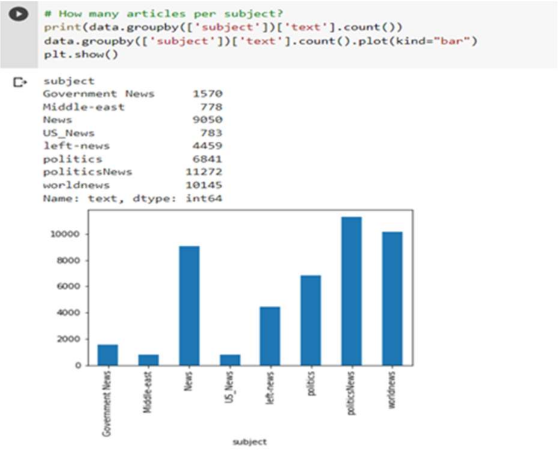
	text	subject	target
0	sounds like call brownshirts group basically i...	News	fake
1		left-news	fake
2	absolutely shame apology republican presidenti...	News	fake
3	tune alternate current radio network acr anoth...	US_News	fake
4	conspiracy theories something take seriously o...	News	fake
...	...	...	...
44893	tune alternate current radio network acr anoth...	Middle-east	fake
44894	oct 18 story corrects age paragraph 19 five si...	worldnews	true
44895	london reuters thousands children teenagers in...	worldnews	true
44896	reuters white house admonished russia wednesda...	worldnews	true
44897	sofia reuters bulgaria tighten rules foreign f...	worldnews	true

44898 rows × 3 columns

Fig 18: Dataset Description

4.3 Articles per subject

In the below images article are shown per subject where different types of subject Government, East middle, News, News form USA, politics, News from politics as well as news from world are described.



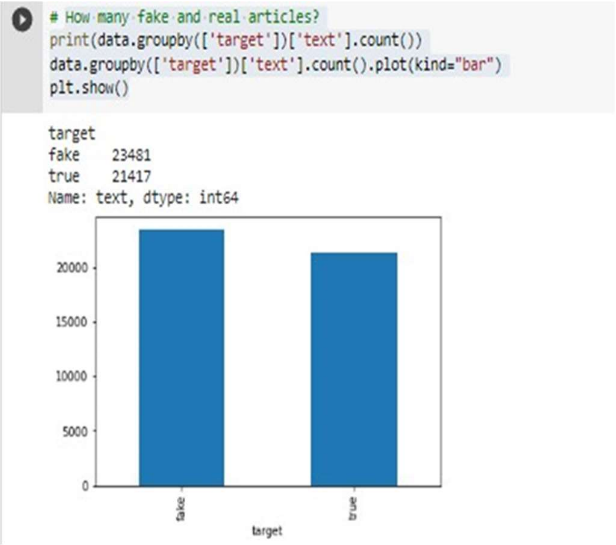
```
# How many articles per subject?
print(data.groupby(['subject'])['text'].count())
data.groupby(['subject'])['text'].count().plot(kind="bar")
plt.show()
```

```
subject
Government News     1570
Middle-east          778
News                9050
US_News              783
left-news           4459
politics            6841
politicsNews       11272
worldnews          10145
Name: text, dtype: int64
```

Fig (19) Article per subjects

4.4 Fake and Real Articles

Total number of fake and real articles are described in below image. We calculated with the help of group by function of pandas library .Our function comprises of target and test feature of our dataset.



```
# How many fake and real articles?
print(data.groupby(['target'])['text'].count())
data.groupby(['target'])['text'].count().plot(kind="bar")
plt.show()
```

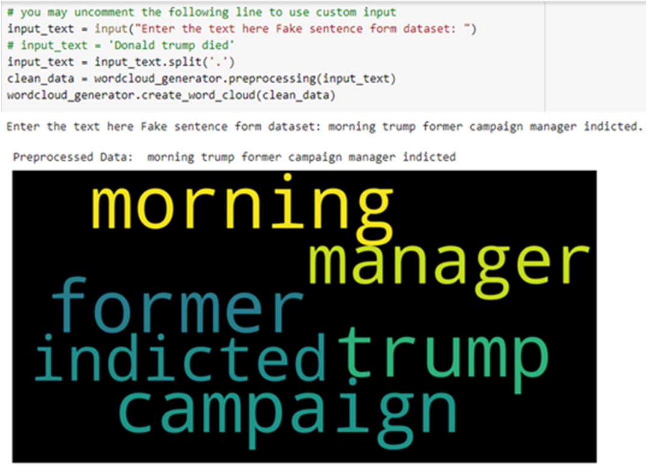
```
target
fake    23481
true    21417
Name: text, dtype: int64
```

Fig 20 : Total fake and real content

In the shown figure total fake articles are 23,481 and true articles are 21,417.

4.5 Fake content displaying using Word cloud

It is a python library. Word cloud is used for data visualization in which the frequency of word is counted. The size of word depends on its frequency. With the help of this library fake textual data are highlighted. It is used to analyse data from social network websites. One fake sentence entered by the user which displayed using word cloud.



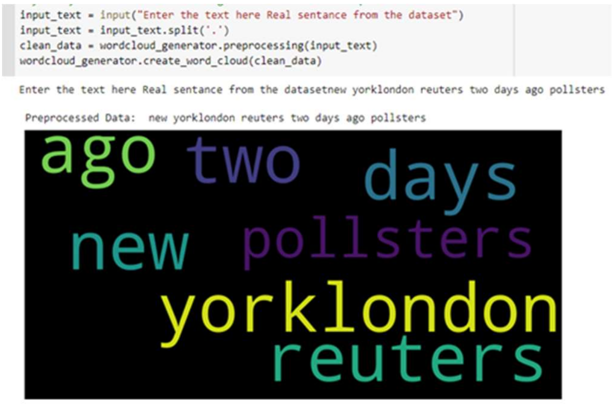
```
# you may uncomment the following line to use custom input
input_text = input("Enter the text here Fake sentence form dataset: ")
# input_text = 'Donald trump died'
input_text = input_text.split('.')
clean_data = wordcloud_generator.preprocessing(input_text)
wordcloud_generator.create_word_cloud(clean_data)
```

```
Enter the text here Fake sentence form dataset: morning trump former campaign manager indicted.

 Preprocessed Data:  morning trump former campaign manager indicted
```

Fig 21: Fake content displaying using Word cloud

4.6 Real content displaying using word cloud

We can also show real content using word cloud library.
User entered the real content wwhich shown in below image



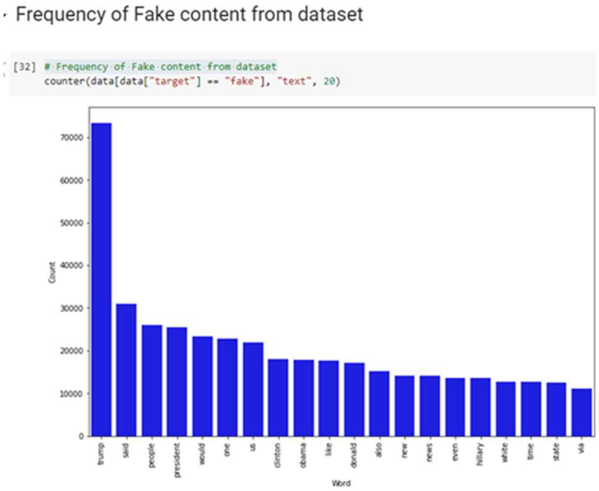
```
input_text = input("Enter the text here Real sentance from the dataset")
input_text = input_text.split('.')
clean_data = wordcloud_generator.preprocessing(input_text)
wordcloud_generator.create_word_cloud(clean_data)
```

```
Enter the text here Real sentance from the datasetnew yorklondon reuters two days ago pollsters

 Preprocessed Data:  new yorklondon reuters two days ago pollsters
```

Fig 22: Real content displaying using word cloud

4.7 Frequency of fake content

NLTK is used to teach a computer to understand Human languages and interpret the same as humans. It is used to extract information and hidden pattern from data. We can count fake content from dataset using NLTK library. Below images shows the frequency of fake content using NLTK



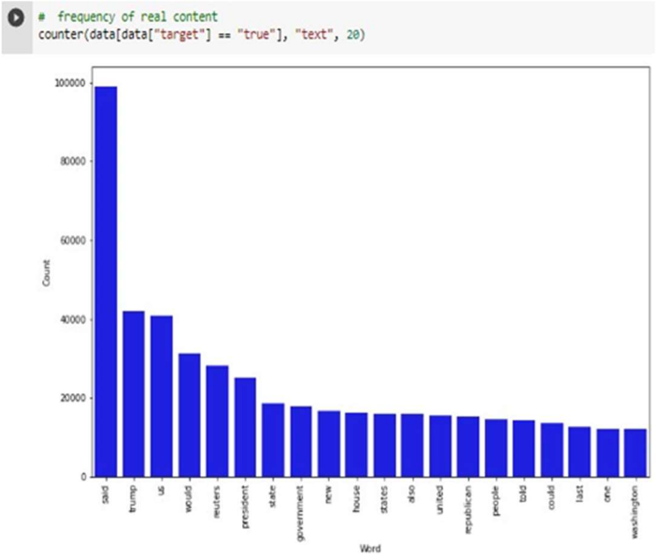
Frequency of Fake content from dataset

```
[32] # Frequency of Fake content from dataset
     counter(data[data["target"] == "fake"], "text", 20)
```

Fig 23: Frequency of fake content

4.8 Frequency of real content

We can also count frequency of real word using NLTK library which shown in below image.



```
# frequency of real content
counter(data[data["target"] == "true"], "text", 20)
```

Fig 24: Frequency of real content

5. Conclusion

This work demonstrates an accuracy of different classifier for detecting fake content from existing dataset. Fake content identification is most important task is online social media platform. We can detect fake content using machine learning as well as artificial intelligence. We used many libraries for comparison of models Sklearn, seaborn, pandas, matplotlib, NumPy. Sklearn library is used for classification as well as regression model. It provides selection of efficient tools for machine learning. Seaborn & matplotlib is used for visualization of data. Pandas library is used to analysing, cleaning, exploring for manipulating the dataset. NumPy is used for working with array. The result shown the different accuracy of classifier that Naïve Bayes 95.24%, Logistic regression 98.96%, Decision Tree 99.67%, Random Forest

99.14% and SVM 99.54%. The decision tree classifier gives the best accuracy for predicting fake content using machine learning and achieved 99.67% highest accuracy as compared to other models and very less time taken by the Naïve Bayes classifier.

REFERENCES

1. makhan kumbhkar, prachee vengurlekar ,"fake news detection: state-of- art approaches, challenges and research directions", journal of emerging technologies and innovative research,2018 jetir october 2018, volume 5, issue 10.
2. tao jiang et al, "a novel stacking approach for accurate detection of fake news", "ieee open access", vol-9,22626-22639.
3. c. buntain and j. golbeck, "automatically identifying fake news in popular twitter threads," 2017 ieee international conference on smart cloud (smartcloud), new york, ny, 2017.
4. mykhailo granik and volodymyr mesyura. fake news detection using naive bayes classifier. ieee first ukraine conference on electrical and computer engineering (ukrcon), pages 900–903.ieee, 2017.
5. m. granik and v. mesyura, "fake news detection using naive bayes classifier," ieee first ukraine conference on electrical and computer engineering (ukrcon), kiev, 2017.
6. h. gupta, m. s. jamal, s. madisetty and m. s. desarkar, "a framework for real-time spam detection in twitter," 10th international conference on communication systems & networks (comsnets), bengaluru, 2018.
7. jamal abdul nasir et al, "fake news detection: a hybrid cnn-rnn based deep learning approach", "international journal of information management data insights".
8. meng-zhe huang et al ,"application research of fake news and rumors detection in complex network environment",hindawi mathematical problems in engineering volume 2022, article id 8787958, 6 pages.
9. z. wang and y. guo, "rumor events detection enhanced by encoding sentimental information into time series division and word representations," neurocomputing, vol. 397, pp. 224–243, 2020.
10. shubha mishra et al, "analyzing machine learning enabled fake news detection techniques for diversified datasets", hindawi wireless communications and mobile computing volume 2022, article id 1575365, 18 pages.
11. k. s. adewole, t. han, w. wu, h. song, and a. k. sangaiah, "twitter spam account detection based on clustering and classification methods," the journal of supercomputing, vol. 76, no. 7, pp. 4802–4837, 2020.
12. h. alzahrani, s. acharya, p. duverger, and n. p. nguyen, "contextual polarity and influence mining in online social networks," computational social networks, vol. 8, no. 1, p. 21, 2021.
13. j. a. nasir, o. s. khan, and i. varlamis,"fake news detection: a hybrid cnn-rnn based deep learning approach," international journal of information management data insights, vol. 1, no. 1, article 100007, 2021.
14. d. h. lee, y. r. kim, h. j. kim, s. m. park, and y. j. yang, "fake news detection using deep learning," journal of information processing systems, vol. 15, no. 5, pp. 1119–1130, 2019.

15. g. sansonetti, f. gasparetti, g. d’aniello, and a. micarelli, “unreliable users detection in social media: deep learning techniques for automatic detection,” ieee access, vol. 8, pp. 213154–213167, 2020.
16. pp. 213154–213167, 2020. [39] w. antoun, f. baly, r. achour, a. hussein, and h. hajj, “state of the art models for fake news detection tasks,” in 2020 ieee international conference on informatics, iot, and enabling technologies (iciot), pp. 519–524, doha, qatar, 2020.
17.-x. lin, b.-y. wu, t.-h. chou, y.-j. lin, and h.-y. kao, bidirectional perspective with topic information for stance detection,” in 2020 international conference on pervasive artificial intelligence (icpai), pp. 1–8, taipei, taiwan, 2020.
18. m. paixão, r. lima, and b. espinasse, “fake news classification and topic modeling in brazilian portuguese,” in 2020 ieee/ wic/acm international joint conference on web intelligence and intelligent agent technology (wi-iat), pp. 427–432, melbourne, australia, 2020.
19.shalini pandey et al,“fake news detection from online media using machine learning classifiers”, journal of physics: conference series, conf. ser. 2161 012027.