FAKE NEWS DETECTION SYSTEM

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge Technologies

R.K.VALLEY

Submitted by:-

J.Akhila-R171117
P.Dhanalakshmi-R171137

Under the Esteemed guidance of Mr.P.SantoshKumar

Department of Computer Science and Engineering

RGUKT RK Valley

Rajiv Gandhi University of Knowledge Techonolgies

RK Valley, Kadapa (Dist), Andhra Pradesh, 516330

CERTIFICATE

This is to certify that the project work titled "**FAKE NEWS DETECTION**" is a bonafied project work submitted by **J.Akhila and P.DhanaLakshmi** in **COMPUTER SCIENCE AND ENGINEERING** in partial fulfillment of requirements for the award of degree of **Bachelor of Technology** for the year **2022-2023** carried out the work under the supervision.

INTERNAL GUIDE	**HEAD OF THE DEPARTMENT**
P SANTOSHKUMAR	N SATYANANDARAM

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task is would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts success.

I am extremely grateful to our respected Director,Prof. K. SANDHYARANI for fostering an excellent academic climate in our institution.

I also express my sincere gratitude to our respected Head of the Department Mr SATYANANDARAM for his encouragement, overall guidance in viewing this project a good asset and effort in bringing out this project.

I would like to convey thanks to our guide at college S SHABANA for his guidance, encouragement, co-operation and kindness during the entire duration of the course and academics.

My sincere thanks to all the members who helped me directly and indirectly in the completion of project work. I express my profound gratitude to all our friends and family members for their encouragement.

INDEX

ABSTRACT

This Project comes up with the applications of NLP (Natural Language Processing) techniques for detecting the ‘fake news’, that is, misleading news stories that comes from the non-reputable sources. Only by building a model based on a count vectorizer (using word tallies) or a (Term Frequency Inverse Document Frequency) tfidf matrix, (word tallies relative to how often they’re used in other articles in your dataset) can only get you so far. But these models do not consider the important qualities like word ordering and context. It is very possible that two articles that are similar in their word count will be completely different in their meaning. The data science community has responded by taking actions against the problem. There is a Kaggle competition called as the “Fake News Challenge” and Facebook is employing AI to filter fake news stories out of users’ feeds. Combatting the fake news is a classic text classification project with a straight forward proposition. Is it possible for you to build a model that can differentiate between “Real “news and “Fake” news? So a proposed work on assembling a dataset of both fake and real news and employ a Naive Bayes classifier in order to create a model to classify an article into fake or real based on its words and phrases.

Introduction

We consume news through several mediums throughout the day in our daily routine, but sometimes it becomes difficult to decide which one is fake and which one is authentic.Every news that we consume is not real. If you listen to fake news it means you are collecting the wrong information from the world which can affect society because a person's views or thoughts can change after consuming fake news which the user perceives to be true.In this fake news detection project,we will focus on text-based news and try to build a model that will help us to identify if a piece of given news is fake or real.

NLP is an area of computer science and artificial intelligence concernedwith the interactions between computers and human (natural) languages, in particular how to program computers to fruitfully process large amounts of natural language data.Natural language processing (NLP) is a subfield of linguistics, computer science,information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyse large amounts of natural language data.

Purpose

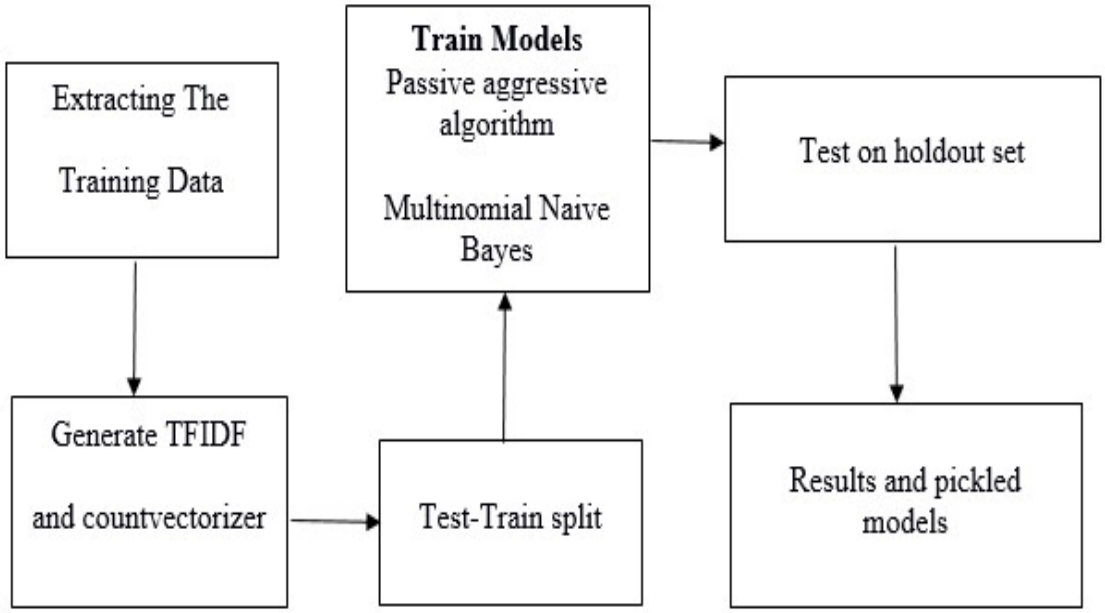
The main purpose of Fake News Detection is is to create a system or model that can use the data of past news reports and predict the chances of a news report being fake or not .The goal is to encourage the development of tools that may help human fact checkers identify deliberate misinformation in news stories through the use of machine learning, natural language processing

Scope

Researchers can further analyze and compare several language generation models with the human writing style and inter-model styles. It will increase their scope and help with better fake news detection. Even though TF-IDF classifiers worked well, there are possibilities of exploring other features to improve the model and make it a generic fit.While the project focused on text-based news articles and language models, AI algorithms can also analyze other features such as images, videos, date and time, sources, website, and domain for valuable information.Teaching the detection model to trace the source of a machine-generated fake news article to the language model from which it originated will be a huge step forward. This development will ensure mitigating and blocking the spread of misinformation.

PROPOSED SYSTEM

In this paper a model is build based on the count vectorizer or a tfidf matrix (i.e) word tallies relatives to how often they are used in other artices in your dataset) can help . Since this problem is a kind of text classification, Implementing a Naive Bayes classifier will be best as this is standard for text-based processing. The actual goal is in developing a model which was the text transformation (count vectorizer vs tfidf vectorizer) and choosing which type of text to use (headlines vs full text). Now the next step is to extract the most optimal features for countvectorizer or tfidf-vectorizer, this is done by using a n-number of the most used words, and/or phrases, lower casing or not, mainly removing the stop words which are common words such as “the”, “when”, and “there” and only using those words that appear at least a given number of times in a given text dataset.



Requirement Specification

HARDWARE:

Ram	:	4GB
Hardisk	:	512GB
Processor	:	2GHz

SOFTWARE:

Language	:	Python3
Tools	:	Jupyter-Notebook
Additional Modules	:	Numpy Matplotlib, Seaborn, Sklearn, Pandas

Data preparation:

```
In [18]: train = pd.read_csv("fake_or_real_news_training_CLEANED.csv")
         test = pd.read_csv("fake_or_real_news_test.csv")
         print(len(train))
         print(len(test))

         3997
         2321
```

```
In [17]: train.head()
```

Out[17]:

	ID	title	text	label	X1	X2
0	8476	You Can Smell Hillary's Fear	Daniel Greenfield | a Shillman Journalism Fell...	FAKE	NaN	NaN
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE	NaN	NaN
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL	NaN	NaN
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9 | 2016 ...	FAKE	NaN	NaN
4	875	The Battle of New York: Why This Primary Matte...	Cruz promised his supporters. ""We're beating...	REAL	NaN	NaN

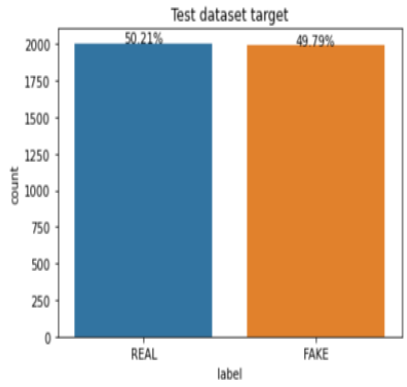
First,we get the dataset into an variable “data”,here dataset include 4 parameters/input to our ML algorithm

Data analysis:

We study if the dataset is unbalanced. From the plot we see this is not the case, as there is a similar amount of Fake and Real news articles. No further actions have to be taken.

```
In [14]: from collections import Counter
         ax = sns.countplot(train.label, order=[x for x, count in sorted(Counter(train.label).items(), key=lambda x: -x[1])])


         for p in ax.patches:
             height = p.get_height()
             ax.text(p.get_x()+p.get_width()/2.,
                     height + 3,
                     '{:1.2f}%'.format(height/len(train)*100),
                     ha="center")
         ax.set_title("Test dataset target")
         show()
```



Data Preprocessing:

In this part we will be cleaning the articles with the help of different NLP techniques, of which we will first explain the concept and its importance.

In order to take into account the title in our accuracy prediction, we created an extra column that combines text and title. We will not do separate predictions on the title since these might classify as e.g. Fake news, whether the actual text with more explanation tells a Real story.

```
In [21]: df['title_and_text'] = df['title'] +' '+ df['text']
         df.tail()
```

Out[21]:

	ID	title	text	label	X1	X2	title_and_text
2316	4490	State Department says it can't find emails fro...	The State Department told the Republican Natio...	None	NaN	NaN	State Department says it can't find emails fro...
2317	8062	The 'P' in PBS Should Stand for 'Plutocratic' ...	The 'P' in PBS Should Stand for 'Plutocratic' ...	None	NaN	NaN	The 'P' in PBS Should Stand for 'Plutocratic' ...
2318	8622	Anti-Trump Protesters Are Tools of the Oligarc...	Anti-Trump Protesters Are Tools of the Oligar...	None	NaN	NaN	Anti-Trump Protesters Are Tools of the Oligarc...
2319	4021	In Ethiopia, Obama seeks progress on peace, se...	ADDIS ABABA, Ethiopia —President Obama convene...	None	NaN	NaN	In Ethiopia, Obama seeks progress on peace, se...
2320	4330	Jeb Bush Is Suddenly Attacking Trump. Here's W...	Jeb Bush Is Suddenly Attacking Trump. Here's W...	None	NaN	NaN	Jeb Bush Is Suddenly Attacking Trump. Here's W...

Here you can read the explanations of the preprocess steps we took:

1.Lowercase the text

This preprocessing step is done so words van later be cross checked with the stopword and pos_tag dictionaries. For future analysis purposes, it could have been benefitial to analyze text with a lot of words in capital letters, by adding a flag variable.

2.Remove the words counting just one letter

3.Remove the words that contain numbers

4.Tokenize the text and remove punctuaion

We performed tokenization with the base python .string function, to split sentences into words (tokens).

5.Remove all stop words

Relevant analysis of the text depends on the most recurring words. Stopwords including words as "the", "as" and "and" appear a lot in a text, but do not give relevant explanation. For this reason they are removed.

6.Remove tokens that are empty

After tokenization, we have to make sure all tokens taken into account contribute to the label prediction.

7.Pos tag the text

We use the pos_tag function included in the ntlk library. This classifies our tokenized words as a noun, verb, adjective or adverb and adds to the understaning of the articles.

8.Lemmatize the text

In order to normalize the text, we apply lemmatization. In this way, words with the same root are processed equally e.g. when took or taken are read in the text, they are lemmatized to take, infinitive of the two verbs.

```
In [30]: ## Save preprocessed df
         df.to_csv("fake_or_real_news_train_PREPROCESSED.csv", index=False)
```

```
In [31]: df = pd.read_csv("fake_or_real_news_train_PREPROCESSED.csv")
         df = df.astype(object).replace(np.nan, 'None')
```

```
In [32]: df.tail()
```

Out[32]:

	ID	title	text	label	X1	X2	title_and_text
6313	4490	State Department says it can't find emails fro...	The State Department told the Republican Natio...	None	None	None	State Department says it can't find emails fro...
6314	8062	The 'P' in PBS Should Stand for 'Plutocratic' ...	The 'P' in PBS Should Stand for 'Plutocratic' ...	None	None	None	The 'P' in PBS Should Stand for 'Plutocratic' ...
6315	8622	Anti-Trump Protesters Are Tools of the Oligarc...	Anti-Trump Protesters Are Tools of the Oligar...	None	None	None	Anti-Trump Protesters Are Tools of the Oligarc...
6316	4021	In Ethiopia, Obama seeks progress on peace, se...	ADDIS ABABA, Ethiopia —President Obama convene...	None	None	None	In Ethiopia, Obama seeks progress on peace, se...
6317	4330	Jeb Bush Is Suddenly Attacking Trump. Here's W...	Jeb Bush Is Suddenly Attacking Trump. Here's W...	None	None	None	Jeb Bush Is Suddenly Attacking Trump. Here's W...

Split Train and Test again after pre-processing is done:

```
In [33]: encoder, train, test, train_cv, train_holdout, train_cv_label, train_holdout_label = split_train_holdout_test(encoder
```

```
Train dataset (Full)
(3997, 8)
Train dataset cols
['ID', 'title', 'text', 'label', 'X1', 'X2', 'title_and_text', 'encoded_label']

Train CV dataset (subset)
(2677, 8)
Train Holdout dataset (subset)
(1320, 8)

Test dataset
(2321, 7)
Test dataset cols
['ID', 'title', 'text', 'label', 'X1', 'X2', 'title_and_text']
```

```
In [34]: encoder
```

```
Out[34]: LabelEncoder()
```

Baseline Modelling:

First, we create a dataframe called models to keep track of different models and their scores.

models = pd.DataFrame(columns=['model_name', 'model_object', 'score'])

Vectorizing dataset:

For any text to be fed to a model, the text has to be transformed into numerical values. This process is called vectorizing and will be redone everytime a new feature is added.

```
In [46]: count_vect = CountVectorizer(analyzer = "word")
         count_vectorizer = count_vect.fit(df.text)
         train_cv_vector = count_vectorizer.transform(train_cv.text)
         train_holdout_vector = count_vectorizer.transform(train_holdout.text)
         test_vector = count_vectorizer.transform(test.text)

In [43]: df.columns

Out[43]: Index(['ID', 'title', 'text', 'label', 'X1', 'X2', 'title_and_text'], dtype='object')

In [47]: count_vect.get_feature_names()[:10]

Out[47]: ['00',
          '000',
          '0000',
          '000000031',
          '00000031',
          '000035',
          '00006',
          '0001',
          '0001pt',
          '0002']
```

MODEL BUILDING

Baseline Model 1:

We create a baseline classification model with a support vector machine, a good model to handle complex classifications.

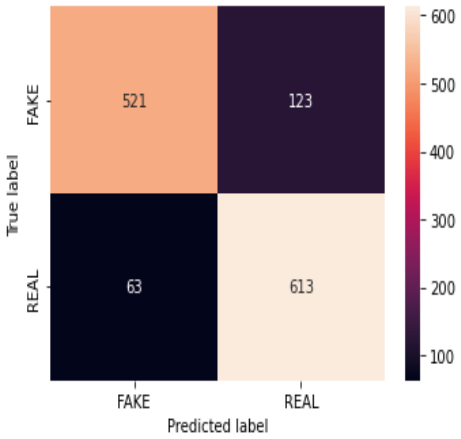
In [48]:
```
SVC_classifier = runModel(encoder,
                train_cv_vector,
                train_cv_label,
                train_holdout_vector,
                train_holdout.label,
                "svc",
                "Baseline Model 1: SVC")
models.loc[len(models)] = SVC
```

```
Baseline Model 1: SVC
GridSearchCV(cv=ShuffleSplit(n_splits=5, random_state=12345, test_size=0.2, train_size=None),
             estimator=SVC(),
             param_grid=[{'C': [1, 10, 50, 100], 'kernel': ['linear']},
                         {'C': [10, 100, 500, 1000], 'gamma': [0.0001],
                          'kernel': ['rbf']}])

CV-scores
Accuracy: 0.871 (+/-0.012) for params: {'C': 100, 'gamma': 0.0001, 'kernel': 'rbf'}
Accuracy: 0.861 (+/-0.026) for params: {'C': 10, 'gamma': 0.0001, 'kernel': 'rbf'}
Accuracy: 0.860 (+/-0.015) for params: {'C': 500, 'gamma': 0.0001, 'kernel': 'rbf'}
Accuracy: 0.854 (+/-0.020) for params: {'C': 1000, 'gamma': 0.0001, 'kernel': 'rbf'}
Accuracy: 0.841 (+/-0.011) for params: {'C': 1, 'kernel': 'linear'}
Accuracy: 0.816 (+/-0.015) for params: {'C': 10, 'kernel': 'linear'}
Accuracy: 0.816 (+/-0.015) for params: {'C': 50, 'kernel': 'linear'}
Accuracy: 0.816 (+/-0.015) for params: {'C': 100, 'kernel': 'linear'}
```

```
Best Estimator Params
SVC(C=100, gamma=0.0001)

Predictions:
['REAL' 'REAL' 'FAKE' ... 'REAL' 'REAL' 'REAL']
```




```
Accuracy:
0.8590909090909091
```

Baseline Model 2: Naïve Bayes:

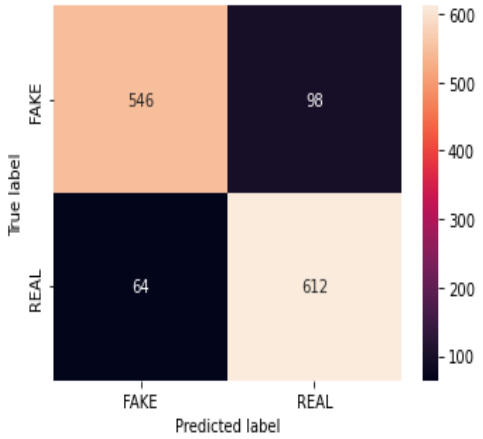
In [49]:
```
NB = runModel(encoder,
              train_cv_vector,
              train_cv_label,
              train_holdout_vector,
              train_holdout.label,
              "nb",
              "Baseline Model 2: Naiive Bayes")
models.loc[len(models)] = NB
```

```
Baseline Model 2: Naiive Bayes
GridSearchCV(cv=ShuffleSplit(n_splits=5, random_state=12345, test_size=0.2, train_size=None),
             estimator=MultinomialNB(), param_grid={})


CV-scores
Accuracy: 0.872 (+/-0.021) for params: {}


Best Estimator Params
MultinomialNB()

Predictions:
['REAL' 'REAL' 'REAL' ... 'REAL' 'REAL' 'REAL']
```




```
Accuracy:
0.8772727272727273
```

Baseline Model 3: MaxEnt Classifier

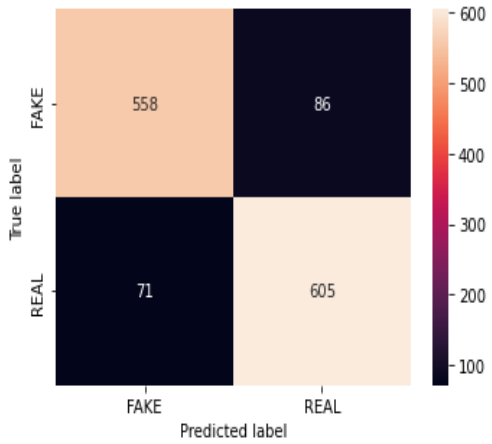
In [50]:
```
maxEnt = runModel(encoder,
                  train_cv_vector,
                  train_cv_label,
                  train_holdout_vector,
                  train_holdout.label,
                  "maxEnt",
                  "Baseline Model 3: MaxEnt Classifier")
models.loc[len(models)] = maxEnt
```

```
Baseline Model 3: MaxEnt Classifier
GridSearchCV(cv=ShuffleSplit(n_splits=5, random_state=12345, test_size=0.2, train_size=None),
             estimator=LogisticRegression(),
             param_grid={'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000],
                         'penalty': ['l1', 'l2']})

CV-scores
Accuracy: nan (+/-nan) for params: {'C': 0.001, 'penalty': 'l1'}
Accuracy: 0.853 (+/-0.017) for params: {'C': 0.001, 'penalty': 'l2'}
Accuracy: nan (+/-nan) for params: {'C': 0.01, 'penalty': 'l1'}
Accuracy: 0.872 (+/-0.010) for params: {'C': 0.01, 'penalty': 'l2'}
Accuracy: nan (+/-nan) for params: {'C': 0.1, 'penalty': 'l1'}
Accuracy: 0.878 (+/-0.011) for params: {'C': 0.1, 'penalty': 'l2'}
Accuracy: nan (+/-nan) for params: {'C': 1, 'penalty': 'l1'}
Accuracy: 0.878 (+/-0.017) for params: {'C': 1, 'penalty': 'l2'}
Accuracy: nan (+/-nan) for params: {'C': 10, 'penalty': 'l1'}
Accuracy: 0.872 (+/-0.017) for params: {'C': 10, 'penalty': 'l2'}
Accuracy: nan (+/-nan) for params: {'C': 100, 'penalty': 'l1'}
Accuracy: 0.881 (+/-0.021) for params: {'C': 100, 'penalty': 'l2'}
Accuracy: nan (+/-nan) for params: {'C': 1000, 'penalty': 'l1'}
Accuracy: 0.884 (+/-0.022) for params: {'C': 1000, 'penalty': 'l2'}

Best Estimator Params
LogisticRegression(C=1000)

Predictions:
['REAL' 'REAL' 'FAKE' ... 'FAKE' 'FAKE' 'REAL']
```




```
Accuracy:
0.8810606060606061
```

Baseline Models Summary:

models

output:

	model_name	model_object	score
0	<class 'sklearn.svm._classes.SVC'>	<class 'sklearn.svm._classes.SVC'>	<class 'sklearn.svm._classes.SVC'>
1	Baseline Model 2: Naiive Bayes	GridSearchCV(cv=ShuffleSplit(n_splits=5, rando...	0.877273
2	Baseline Model 3: MaxEnt Classifier	GridSearchCV(cv=ShuffleSplit(n_splits=5, rando...	0.881061

Feature Engineering:

1. POS Tagging:

Adding a prefix to each word with its type (Noun, Verb, Adjective,...). e.g: I went to school => PRP-I VBD-went TO-to NN-school**.**Also, after lemmatization it will be 'VB-go NN-school', which indicates the semantics and distinguishes the purpose of the sentence.This will help the classifier differentiate between different types of sentences.

2.TF-IDF weighting:

Try to add weight to each word using TF-IDF.We are going to calculate the TFIDF score of each term in a piece of text. The text will be tokenized into sentences and each sentence is then considered a text item.We will also apply those on the cleaned text and the concatinated POS_tagged text.We are going to calculate the TFIDF score of each term in a piece of text. The text will be tokenized into sentences and each sentence is then considered a text item.

Term Frequency:tf is the number of times a termappears in a particular document. So it's specific to a document.

Few Ways to calculate tf is as follows:

tf(t) = (No. of times term 't' occurs in a document) / (No. Of terms in a document)

(or)

tf(t) = (No. of times term 't' occurs in a document) / (Frequency of most common term in a document)

Inverse Document Frequency:

idf is a measure of how common or rare a term is across the entire corpus of documents. So the point to note is that it's common to all the documents.

As per sklearn's online documentation, it uses the below method to calculate idf of a term in a document:

idf(t) = log e [(1+n) / (1 + df(t))] + 1 (defaul=true) and

idf(t) = log e [n / df(t)] + 1

i:e smooth_idf =(when smooth_idf = False)

n = Total number of documents available

t = term for which idf value has to be calculated

df(t) = Number of documents in which the term t appears

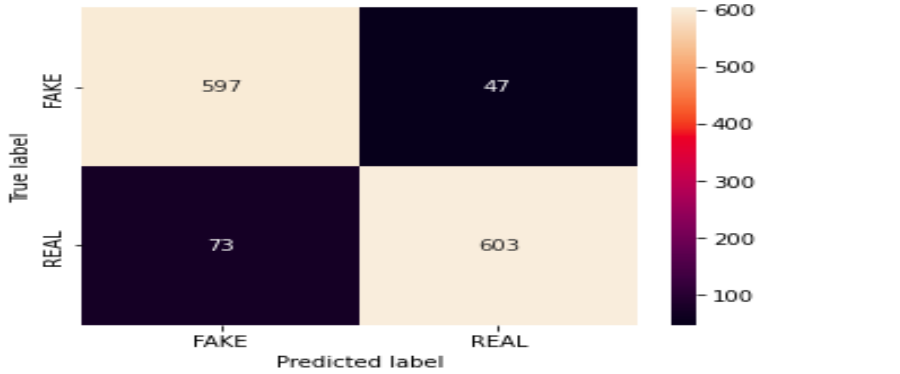
3. Use Bigram Vectorizer instead of regular vectorizer:

For FE3, we use the Trigram vectorizer, which vectorizes triplets of words rather than each word separately. In this short example sentence, the trigrams are "In this short", "this short example" and "short example sentence".

MaEnt on preprocessed + pos-tagged (FE1) + TF-IDF weighted (FE2) + Trigram vectorized text (FE3)

```
Best Estimator Params
LogisticRegression(C=100)

Predictions:
['REAL' 'REAL' 'FAKE' ... 'REAL' 'FAKE' 'REAL']
```

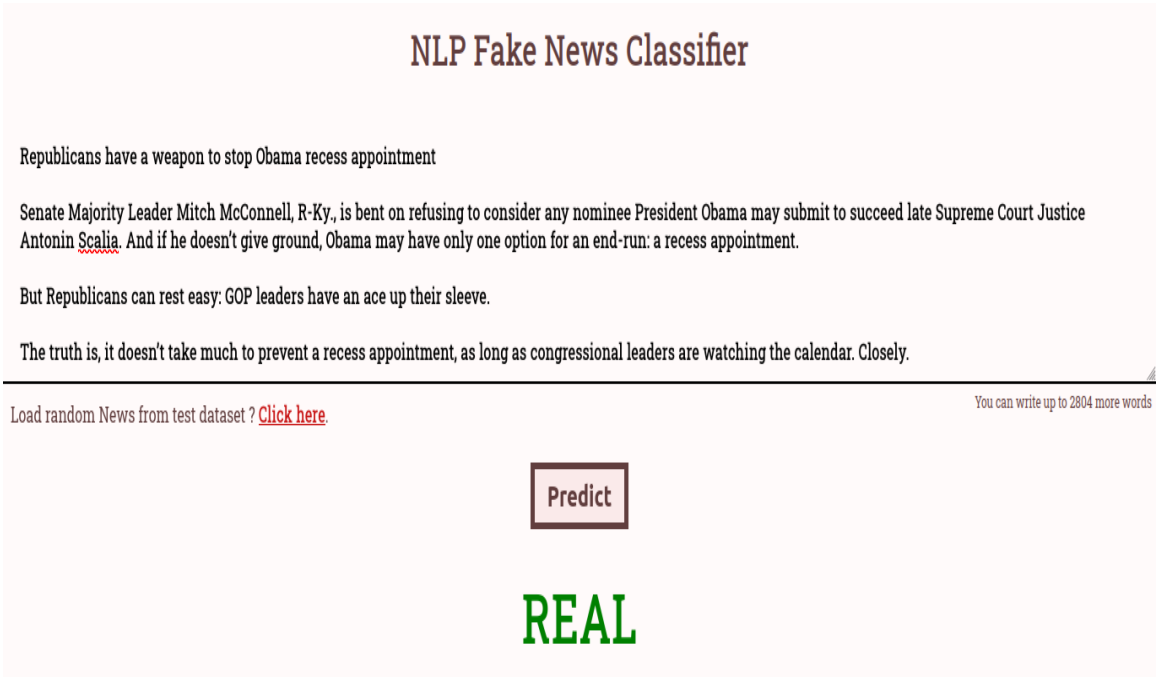



```
Accuracy:
0.9090909090909091
```

```
[100]: models
```

Deploying the ML model in web Application:

After making a predictive system in Jupyter Notebook,we deploy the system in a web application using Flask.Before that we load entire ML code into a pickle filewhich is used to deploy entire project with a single file,we load data to deploy.



Technologies used:

Libraries :

1.Pandas: Pandas is a software library written for the python programming language for data manupulation and analysis.In particular,it offers data structures and operations for manupulating numerical tables and time series.

2.Sklearn: Scikit-learn is a free machine learning library for python.It features various algorithms like support vector machine,random forest and k-neighbours.it also supports python numerical and scientific libraries like numpy and scipy.

3.Matplotlib:matplotlib.py plot is a collection of functions that make matplotlib work like MATLAB.Each pyplot function makes some change to a figure:e.g.,creates a figure,creates a ploting area in a figure,plots some lines in a ploting area,decorates the plot with labels etc.

4.pickle:"Pickling " is the process whereby a python object hierarchy is converted into a byte stream and "Unpickling" is the inverse operation,whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy.

Framework:

Flask: Python offers concise and readable code.While complex algorithms and versatile work flows stand behind machine learning and AI,python's simplicity allows developers to write reliable systems.python code is understandable by humans,which make it easier to build models for machine learning.

HTML: The HyperText Markup Language,or HTML is the standard markup language for documents designed to be displayed in a web browser.

CSS: Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language such as HTML.

JAVASCRIPT: Javascript is a high level,often just-in-time complied,multi-paridigm.It has curly-bracket syntax,dynamic typing,prototype-based object orientation,and first class functions.

CONCLUSION

As examined, fake news detection algorithms provide many benefits, but also have many constraints and limitations. Using this system, fake news can be classified, leading to a more “real” future.We can print a confusion matrix to gain insight into the number of false and true negatives and positives.Analysis and DesignIn this project, I build a text classification to define whether or not a certain article is fake news or real news. Using Natural Language Processing methodologies in Python and Classification Theory, I reached an accuracy of 0.945455 for classifying news as fake.

REFERENCES

1.For understanding of how each algorithm works, simply search the algorithm class (i.e. LogisticRegression()) on scikit-learn website and documentation.

2.Dataset that was used in this project at this link , which takes you to a Kaggle webpage. This was an already-built dataset by someone else.

3.To get a deeper concenptual understanding of the process and all the functions, we refered this article titled "Practical Explanation of NLP for Fake News Detection"