Model Deployment using Flask

Name: Jamila Hamdi

Batch code: LISUM01

Submission date: 04/07/2021

Submitted to: Data Glacier

Abstract

This project aims to detect fake news. Using a Passive Aggressive Classifier, a model was created on Spyder using Flask.

What is Flask?

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. The only feature that distinguishes Flask from other frameworks is that it is very easy to use.

What is Fake news?

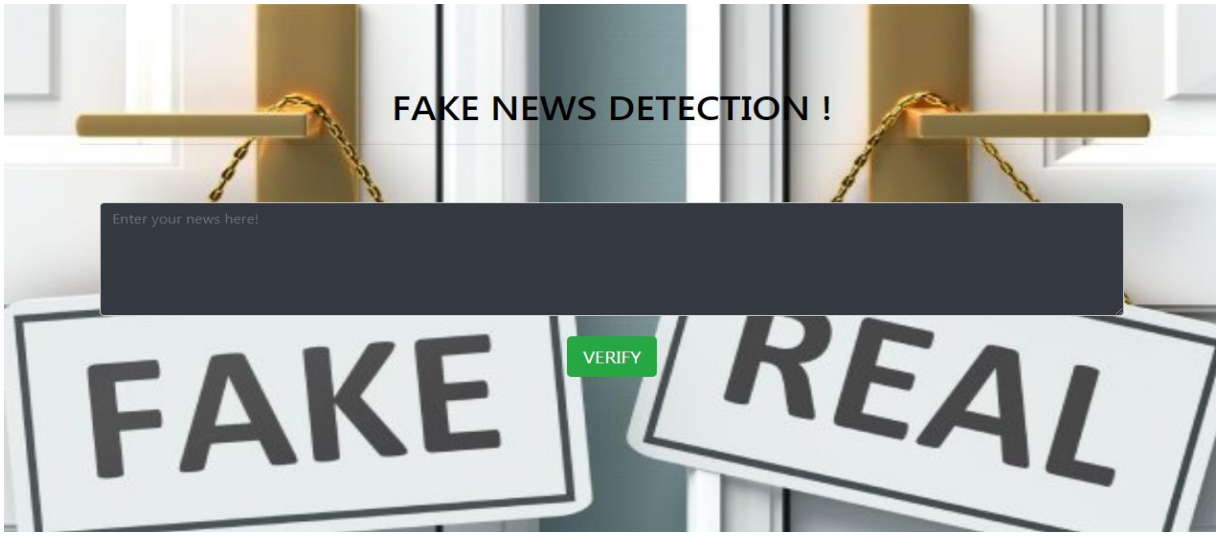
Fake news is a form of news consisting of deliberate disinformation or hoaxes spread via traditional news media or online social media. Digital news has brought back and increased the usage of fake news, or yellow journalism.

Snapshot of each step of deployment

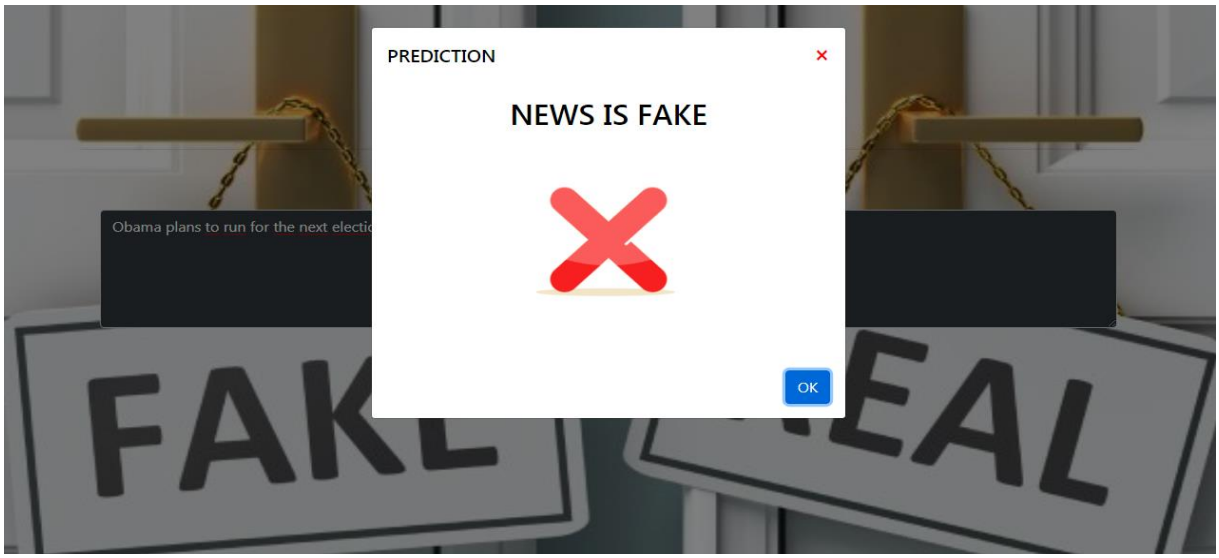
The following snapshots presents the built flask application.

The input in the text box is published news from a known source, which should be verified. The purpose of the VERIFY button is to check whether the news is TRUE or FALSE.

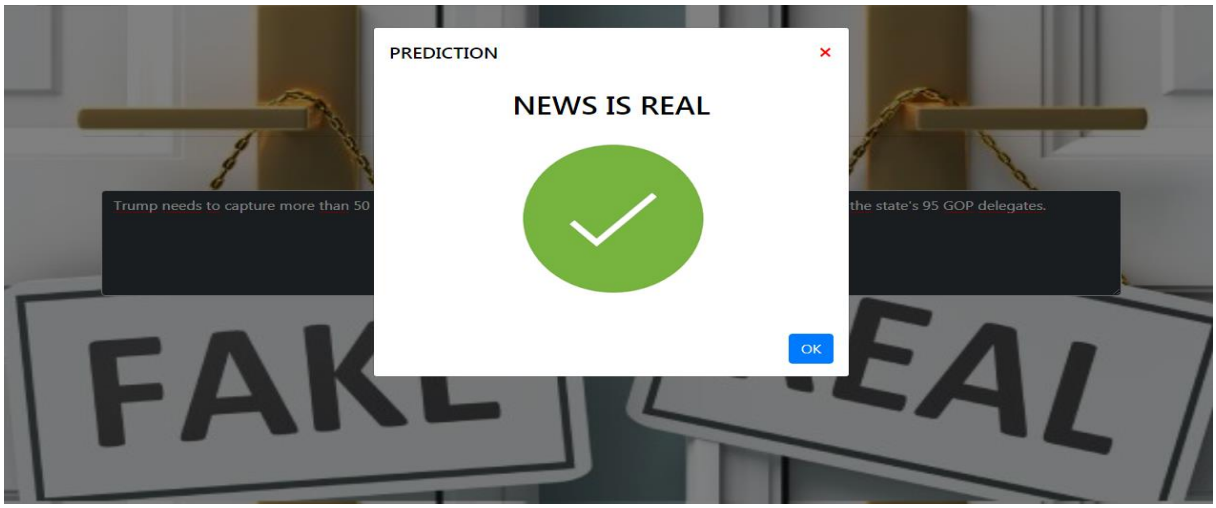
1. **home page**



2. Fake news case



3. Real news case



Now let's present the work behind those snapshots

- ✓ Model Building and save
- ✓ Flask application building

Model building

This model aims to classify a piece of news as REAL or FAKE.

The dataset used for this python project is called **news.csv** and it has a shape of 7796×3. The first column identifies the title of news, the second is the text, and the third column has labels denoting whether the news is **REAL** or **FAKE**.

This is how the dataset looks like:

	title	text	label
0	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
2	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL

Using **sklearn**, we build a **TfidfVectorizer** on our dataset. Then, we initialize a **PassiveAggressive Classifier** and fit the model. In the end, the accuracy score tell us how well our model fares.

```
import numpy as np
import pandas as pd
import pickle
from tensorflow import keras
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers.embeddings import Embedding
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
from keras import utils

df=pd.read_csv("news.csv")
df.head()


labels = df.label
X_train, X_test, y_train, y_test = train_test_split(df['text'],labels,test_size=1)

tfidf_vectorizer = TfidfVectorizer(stop_words='english', max_df=0.7)

# Fit and transform train set, transform test set
tfidf_train = tfidf_vectorizer.fit_transform(X_train)
tfidf_test  = tfidf_vectorizer.transform(X_test)

# Initialize a PassiveAggressiveClassifier
pac = PassiveAggressiveClassifier(max_iter=50)
pac.fit(tfidf_train,y_train)

# saving vectorizer
with open('tfid.pickle','wb') as f:
    pickle.dump(tfidf_vectorizer,f)

# saving model
with open('model_fakenews.pickle','wb') as f:
    pickle.dump(pac,f)
```

After training the model, it's time to make some predictions:

```
#Predict on the test set and calculate accuracy
y_pred=pac.predict(tfidf_test)
score=accuracy_score(y_test,y_pred)
print(f'Accuracy: {round(score*100,2)}%')
```

```
Accuracy: 92.82%
```

After training of the model, we tested our model accuracy which is 92 %.

In order to save the built model, **pickle** was utilized.

Flask application building

```
from flask import Flask, render_template, request, url_for, Markup, jsonify
import pickle

app = Flask(__name__)
pickle_in = open('model_fakenews.pickle','rb')
pac = pickle.load(pickle_in)
tfid = open('tfid.pickle','rb')
tfidf_vectorizer = pickle.load(tfid)

@app.route('/')
def home():
    return render_template("index.html")

@app.route('/newscheck')
def newscheck():
    abc = request.args.get('news')
    input_data = [abc.rstrip()]
    # transforming input
    tfidf_test = tfidf_vectorizer.transform(input_data)
    # predicting the input
    y_pred = pac.predict(tfidf_test)
    return jsonify(result = y_pred[0])


if __name__=='__main__':
    app.run(debug=True)
```

HTML page

```
<!DOCTYPE html>
<html>
<head>
    <title>FAKE NEWS DETECTION</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/bootbox.js/5.4.0/bootbox.min.js"></script>
    <style>
        body{
            background-image: url(../static/fakereal.jpg);
             width: 100%;
             height:100%;
             background-size: cover;
            color: black;
        }
        .modal-body,.modal-footer,.modal-header{
            background:#fff;
            border: none;
            color: black;
        }
        .modal-header .close{
            color: red;
            text-shadow: none;
            size: 1.9rem;
        }
        .close{
            opacity: 1;
        }
        textarea{
            background-color: white;

        }
        p{
        color: red;
        }
```

```
    </style>
</head>
<body>
    <div  align="center" style="margin-top:70px;" class="container">
        <div class="card bg-transparent"style="border:hidden;">
            <div class="card-header bg-transparent">
            <font color="red">
            <p id="source" data-text="FAKE NEWS DETECTION "></p></font>
                <h1 id="dest"></h1>
            </div><br>
            <div class="card-body bg-transparent">
                <form><br>
                    <textarea type="text" class="form-control bg-dark text-white" name="news" id="news" placeholder="Enter your news here!" rows="5"
                    <button type="submit" class="btn btn-lg btn-success" name="submit" id="submit">VERIFY</button>
                </form>
            </div>
        </div>
    </div>
</body>
</html>
<script>
    var source = $('#source').attr('data-text');
    var dest   = $('#dest');
    function typeWriter(text, n) {
        if (n < (text.length)) {
            dest.html(text.substring(0, n+1));
            n++;
            setTimeout(function() {
                typeWriter(text, n)
            }, 150);
        }
    }

    typeWriter(source, 0);

    $SCRIPT_ROOT = "{{ url_for('newscheck') }}";
    $(function() {
        $('#submit').bind('click', function() {
            var news = $('#news').val();
```

```
        $('#submit').bind('click', function() {
            var news = $('#news').val();
            if(news === "" || news === " " || news === "\n" || news === null){
                bootbox.alert({
                    size: "big",
                    title: "EMPTY FIELD",
                    message: "Please enter some news!",
                    backdrop: true
                });
            }
            else{
                $.getJSON($SCRIPT_ROOT, {
                    news: news,
                }, function(data) {
                    if(data.result == "REAL"){
                        var src = "static/success.gif";
                    }
                    else{
                        var src = "static/fail.gif";
                    }
                    bootbox.alert({
                        size: "big",
                        title: "PREDICTION",
                        message: "<div align='center'><h2>NEWS IS "+data.result+"</h2><img style='width:240px;height:232px;' src='"+src+"'/></div>",
                        backdrop: true,
                        callback: function(){
                            setTimeout(function(){
                                //do what you need here
                                location.reload();
                            }, 100);

                        }
                    });
                });
            }
            return false;
        });
    });
</script>
```