Fake News Detection - Embeddings + Neural Networks: Content

1. Initial Data Cleaning and Exploration

- Checking for and removing duplicate news
- Deciding which features to use for analysis by checking for relationship between features and labels.

2. Data Preprocessing

- Removing punctions and unneeded characters from news text.
- Removing stop words
- Tokenization
- Stemmatization

3. Feature Extraction and Model Training

- Using TF-IDF and basic classification algorithms(Naive Bayes and Logistic Regression)
- Using Word embeddings from scracth + neural networks
- Using pre-trained word embeddings(GloVe) + neural networks

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
import tensorflow as tf
import matplotlib.pyplot as plt
from tensorflow.keras.layers import Embedding,LSTM,Dense,Dropout, Bidirectional
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.text import one_hot
from wordcloud import WordCloud
import nltk
import re
from nltk.corpus import stopwords
from sklearn.metrics import classification_report, accuracy_score
```

```
# Connect to Google Drive
# Upload the dataset to your Google drive so it can be loaded here

from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
# Reading the Fake and Real News data set

fake_news = pd.read_csv('/content/gdrive/My Drive/Colab Notebooks/fake-news/Fake.cs
fake_news['credibility'] = 0
fake_news
```

	title	text	subject	date	credibility
0	Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017	0
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017	0
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017	0
3	Trump Is So Obsessed He Even Has Obama's Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017	0
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017	0
...	...	...	...	...	...
	McPain: John McCain	21st Century Wire says	Middle	January	

```
real_news = pd.read_csv('/content/gdrive/My Drive/Colab Notebooks/fake-news/True.cs
real_news['credibility'] = 1
real_news
```

	title	text	subject	date	credibility
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017	1
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017	1
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017	1
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017	1
4	Trump wants Postal Service to charge 'much mor	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017	1

Data Exploration

```
# checking for the content of some rows
pd.set_option('max_colwidth', None)
all_news = fake_news.append(real_news, ignore_index=True)
all_news.sample(3)
```

title	text	subject	date	credibility
U.S. praises Saudi Arabia	WASHINGTON (Reuters) - The United States on Monday praised Saudi Arabia for exposing Iran s role in Yemen and Tehran s provision of missile systems to Houthi militia fighting there, following the interception of a missile fired toward the Saudi capital Riyadh on Saturday. We continue to maintain strong defense		November	

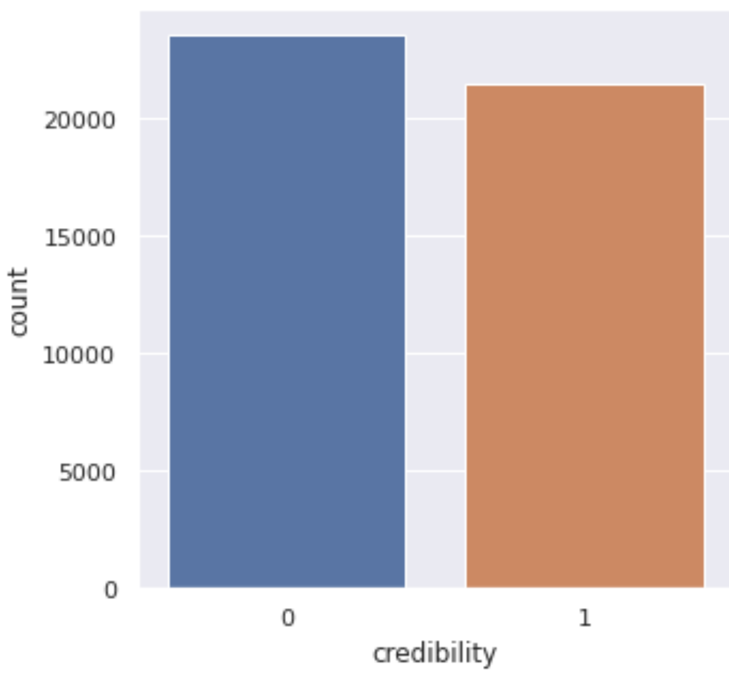
39014	for exposing Iran's role in Yemen	We continue to maintain strong defense ties with the Kingdom of Saudi Arabia and work together on common security priorities to include combat operations against violent extremist organizations, and neutralizing Iran s destabilizing influence in the Middle East region, said Pentagon spokesman Marine Major Adrian Rankine-Galloway.	worldnews	November 6, 2017	1
36586	U.N. rights boss urges Mexico not to enshrine army's role	GENEVA (Reuters) - The United Nations human rights boss called on Mexico s Senate on Tuesday not to adopt a proposed law on internal security, saying it would enshrine the role of the military in law enforcement at a time when a stronger police force was needed. Zeid Ra ad al-Hussein, U.N. High Commissioner for Human Rights, said that more than a decade after the armed forces were deployed in the war on drugs , violence had not abated and extrajudicial killings, torture and disappearances continue to be committed by various state and non-state actors . In a statement recognizing Mexico s huge security challenge and violence sown by powerful organized crime groups, Zeid said: Adopting a new legal framework to regulate the operations of the armed forces in internal security is not the answer. The current draft law risks weakening incentives for the civilian authorities to fully assume their law enforcement roles.	worldnews	December 5, 2017	1
		BAGHDAD (Reuters) - Kurdish Peshmerga fighters rejected a warning from an Iraqi paramilitary force to withdraw from a strategic junction south of Kirkuk, which controls access to some of the region s main oilfields, a			

```
all_news.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44898 entries, 0 to 44897
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   title        44898 non-null  object
 1   text         44898 non-null  object
 2   subject      44898 non-null  object
 3   date         44898 non-null  object
 4   credibility  44898 non-null  int64 
dtypes: int64(1), object(4)
memory usage: 1.7+ MB
```

```
import seaborn as sns

# checking for class imbalance
sns.set(rc={'figure.figsize':(5,5)})
sns.countplot(x='credibility', data=all_news)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa864c08850>
```



From the above, it is clear that the dataset is balanced for both fake and real news

```
# checking for duplicate text

from hashlib import sha256
from tqdm import tqdm
list_ = [ ]
for text in tqdm(all_news['text']):
    hash_ = sha256(text.encode('utf-8')).hexdigest()
    list_.append(hash_)
all_news['hash'] = list_
pd.reset_option('max_colwidth')
all_news
```

```
100%|██████████| 44898/44898 [00:00<00:00, 77833.58it/s]
```

	title	text	subject	date	credibility	
0	Donald Trump Sends Out Embarrassing New Year’...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017	0	91dad1c56705a1f9ba
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017	0	63967cde9e4b4f9a50
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017	0	53a760ba38dc28b80
3	Trump Is So Obsessed He Even Has Obama’s Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017	0	7f2367f844e5293359
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017	0	74e3191c20629d8bc

```
t = all_news.groupby(['hash']).size().reset_index(name='count')
duplicate = t[t['count']>1]
print('there are ',duplicate.shape[0], 'duplicate texts')
```

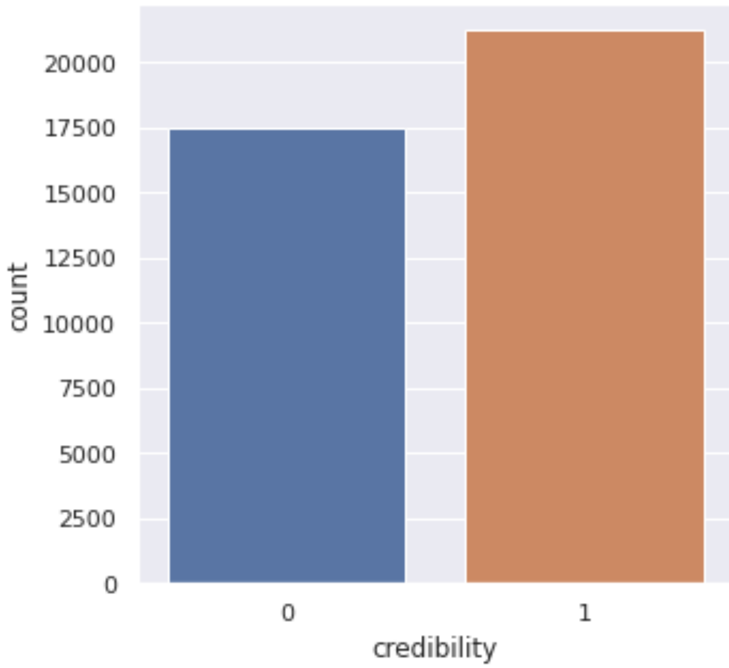
```
there are  5140 duplicate texts
```

```
# removing rows with duplicate text
all_news.drop_duplicates(subset='hash', inplace=True)
all_news.reset_index(inplace=True, drop=True)
all_news.drop('hash', axis=1, inplace=True)
all_news
```

	title	text	subject	date	credibility
0	Donald Trump Sends Out Embarrassing New Year’...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017	0
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017	0
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017	0
3	Trump Is So Obsessed He Even Has Obama’s Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017	0
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017	0
...	...	...	...	...	...
	'Fully committed' NATO	BRUSSELS (Reuters) -		August	

```
# checking for class imbalance after dropping duplicates
sns.set(rc={'figure.figsize':(5,5)})
sns.countplot(x='credibility', data=all_news)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa86373f390>
```



After dropping duplicates, the count of fake news has reduced, meaning most of the duplicate text were from fake news. However, the dataset set is still balanced

Checking for Relationship between features(subject, date, title) and labels(credibility)

Checking for relationship between news subject and news credibility

```
import seaborn as sns

# checking for relationship between credibility and subject
sns.set(rc={'figure.figsize':(11,5)})
sns.countplot(x='subject', data=all_news, hue='credibility')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-3ad02521b652> in <module>()
      3 # checking for relationship between credibility and subject
      4 sns.set(rc={'figure.figsize':(11,5)})
----> 5 sns.countplot(x='subject', data=all_news, hue='credibility')

NameError: name 'all_news' is not defined
```

SEARCH STACK OVERFLOW

- From the plot above, it is clear that real news are only centered around politicNews and worldnews subject areas, while fake news are centered around the other subject areas.
- This indicates that the subject area can help determine if news is fake or real.

Checking for relationship between news date and news credibility

```
#converting date string to datetime format

#removing url in date column
url_pattern = "http"
filter1 = all_news['date'].str.contains(url_pattern)
all_news = all_news[~filter1]
all_news
```

	title	text	subject	date	credibility
0	Donald Trump Sends Out Embarrassing New Year’...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017	0
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017	0
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017	0
3	Trump Is So Obsessed He Even Has Obama’s Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017	0
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017	0
...	...	...	...	...	...
	'Fully committed' NATO	BRUSSELS (Reuters) -		August	

```
# removing other texts in date column
date_pattern = "Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec"
filter2 = all_news['date'].str.contains(date_pattern)
all_news = all_news[filter2]
all_news.reset_index(drop=True, inplace=True)
```

```
# converting date string to datetime format
all_news_copy = all_news.copy()
all_news_copy['date'] = pd.to_datetime(all_news_copy['date'])
all_news_copy.sort_values(by=['date'], inplace=True)
all_news_copy.reset_index(drop=True, inplace=True)
pd.reset_option('max_rows')
all_news_copy
```

	title	text	subject	date	credibility
0	APPLE’S CEO SAYS RELIGIOUS FREEDOM LAWS ARE ‘D...	The gay mafia has a new corporate Don. This i...	politics	2015-03-31	0
1	OH NO! GUESS WHO FUNDED THE SHRINE TO TED KENNEDY	Nothing like political cronyism to make your s...	politics	2015-03-31	0
2	BENGHAZI PANEL CALLS HILLARY TO TESTIFY UNDER ...	Does anyone really think Hillary Clinton will ...	politics	2015-03-31	0
3	HILLARY RODHAM NIXON: A CANDIDATE WITH MORE BA...	The irony here isn t lost on us. Hillary is be...	politics	2015-03-31	0
4	FLASHBACK: KING OBAMA COMMUTES SENTENCES OF 22...	Just making room for Hillary President Obama t...	politics	2015-03-31	0
...	...	...	...	...	...
	IT BEGINS…RINO MEGA-	A longtime Republican		2018	

```
# creating a dataframe of fake news counts by date
fake = all_news_copy[all_news_copy['credibility']==0]
fake['count'] = 0
fake = fake.groupby(['date'])['count'].count()
fake = pd.DataFrame(fake)
fake
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCop
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
  This is separate from the ipykernel package so we can avoid doing imports ur
```

date	count
2015-03-31	6
2015-04-01	2
2015-04-02	1
2015-04-04	4
2015-04-05	7
...	...
2018-02-15	9
2018-02-16	8
2018-02-17	7
2018-02-18	7
2018-02-19	3

1010 rows × 1 columns

```
# creating a dataframe of real news counts by date
real = all_news_copy[all_news_copy['credibility']==1]
real['count'] = 0
real = real.groupby(['date'])['count'].count()
real = pd.DataFrame(real)
real
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCop
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
  This is separate from the ipykernel package so we can avoid doing imports ur
```

	count
date	
2016-01-13	29
2016-01-14	14
2016-01-15	23
2016-01-16	5
2016-01-17	3
...	...
2017-12-27	53
2017-12-28	5
2017-12-29	6
2017-12-30	1
2017-12-31	2

716 rows × 1 columns

```
# creating lineplots of fake and real news over time
sns.set(rc={'figure.figsize':(11,5)})
sns.lineplot(x=real.index, y=real['count'])
sns.lineplot(x=fake.index, y=fake['count'])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-536ae16c97e0> in <module>()
      1 # creating lineplots of fake and real news over time
----> 2 sns.set(rc={'figure.figsize':(11,5)})
      3 sns.lineplot(x=real.index, y=real['count'])
      4 sns.lineplot(x=fake.index, y=fake['count'])

NameError: name 'sns' is not defined
```

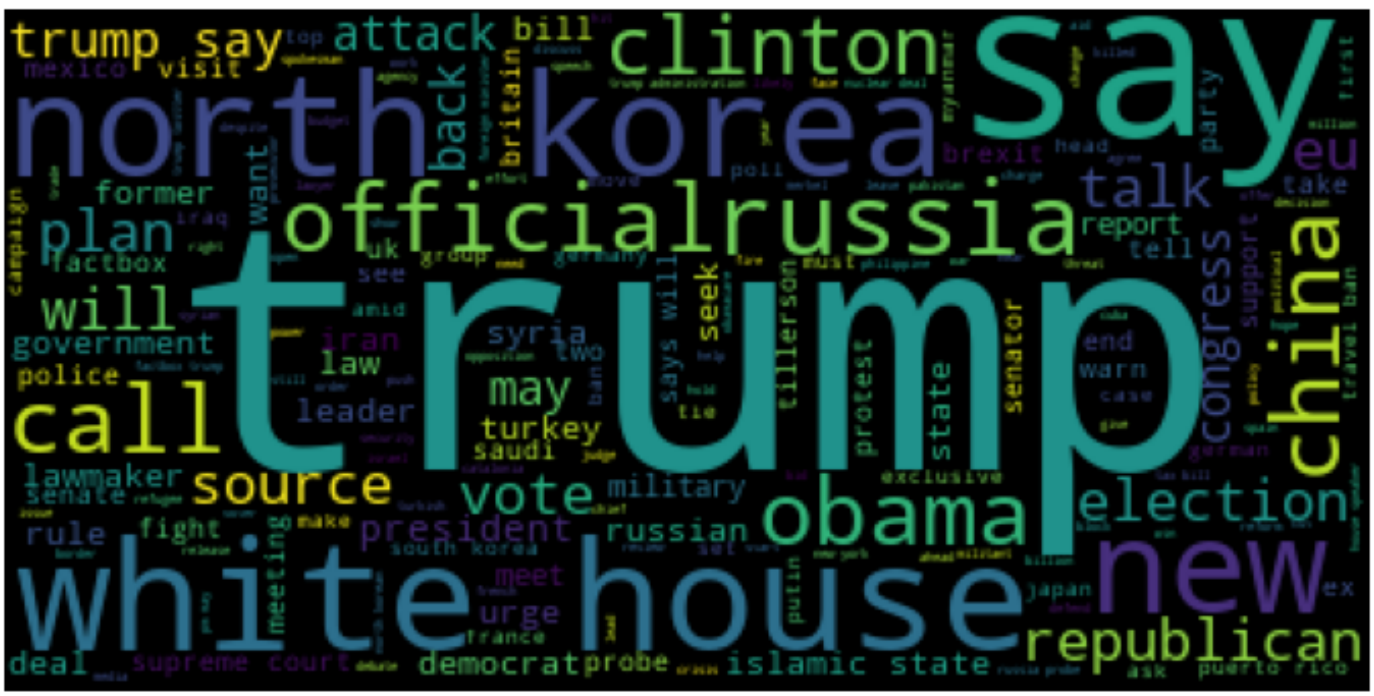
SEARCH STACK OVERFLOW

From the plot below, it seems there is some correlation between date a news article was created and its credibility. There was a sharp rise in fake news in later years, while real news dropped marginally.

Checking for relationship between news title, news text and news credibility

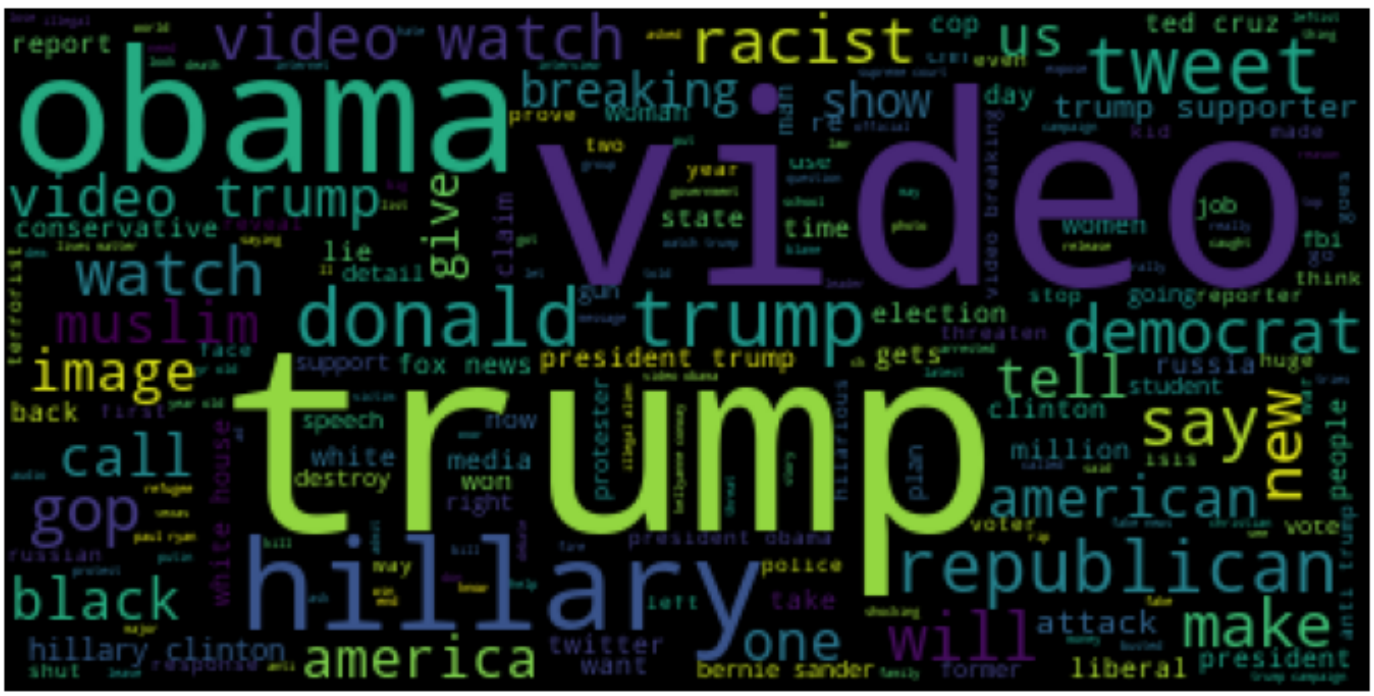
```
# word cloud for real news title

import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS
stopwords = set(STOPWORDS)

real_words = ""
for line in all_news[all_news['credibility']==1]['title']:
    line = str(line) # change each line item to string
    tokens = line.split() # split line text into word tokens

    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower() # convert each token into lower case
    real_words += " ".join(tokens)+" "

wordcloud_ = WordCloud(stopwords=stopwords).generate(real_words)
plt.figure(figsize = (12, 16), facecolor = None)
plt.axis('off')
plt.imshow(wordcloud_)
```

```
<matplotlib.image.AxesImage at 0x7ff7f67c6890>
```



```
# word cloud for fake news title
fake_words = ""
for line in all_news[all_news['credibility']==0]['title']:
    line = str(line) # change each line item to string
    tokens = line.split() # split line text into word tokens

    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower() # convert each token into lower case
    fake_words += " ".join(tokens)+" "

wordcloud_ = WordCloud(stopwords=stopwords).generate(fake_words)
plt.figure(figsize = (12, 16), facecolor = None)
plt.axis('off')
plt.imshow(wordcloud_)
```

```
<matplotlib.image.AxesImage at 0x7ff7f4bd2d90>
```

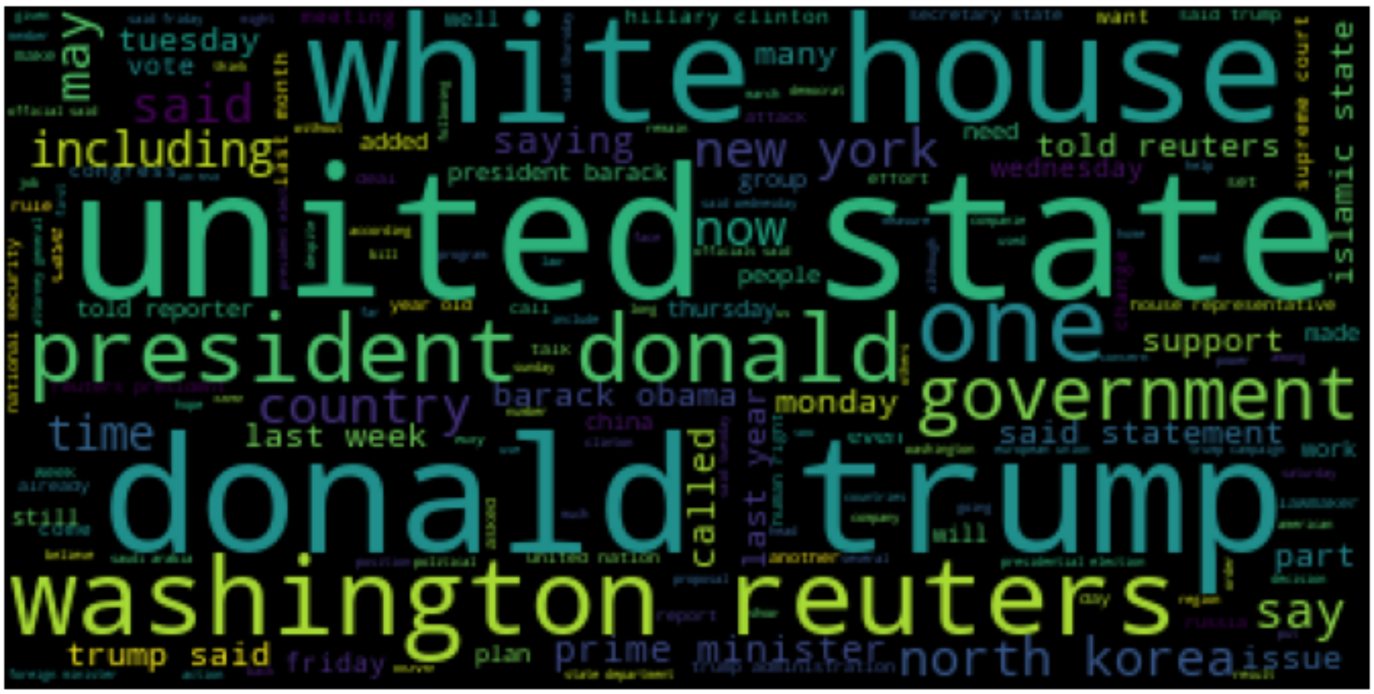


Similar common words in both fake news and real news titles include: Trump, Obama, etc. But there are words like White House, US, North Korea, Russia, that are very common in real news titles but are not so common in fake news titles. On the other hand, there are words like Video, tweet, hillary, watch, gop, that are common in fake news titles, but are not so common in real news titles. This shows that there is some distinguishing feature between most real and fake news titles, and including titles in our analysis can add some information to our model

Checking for relationship between news text and credibility

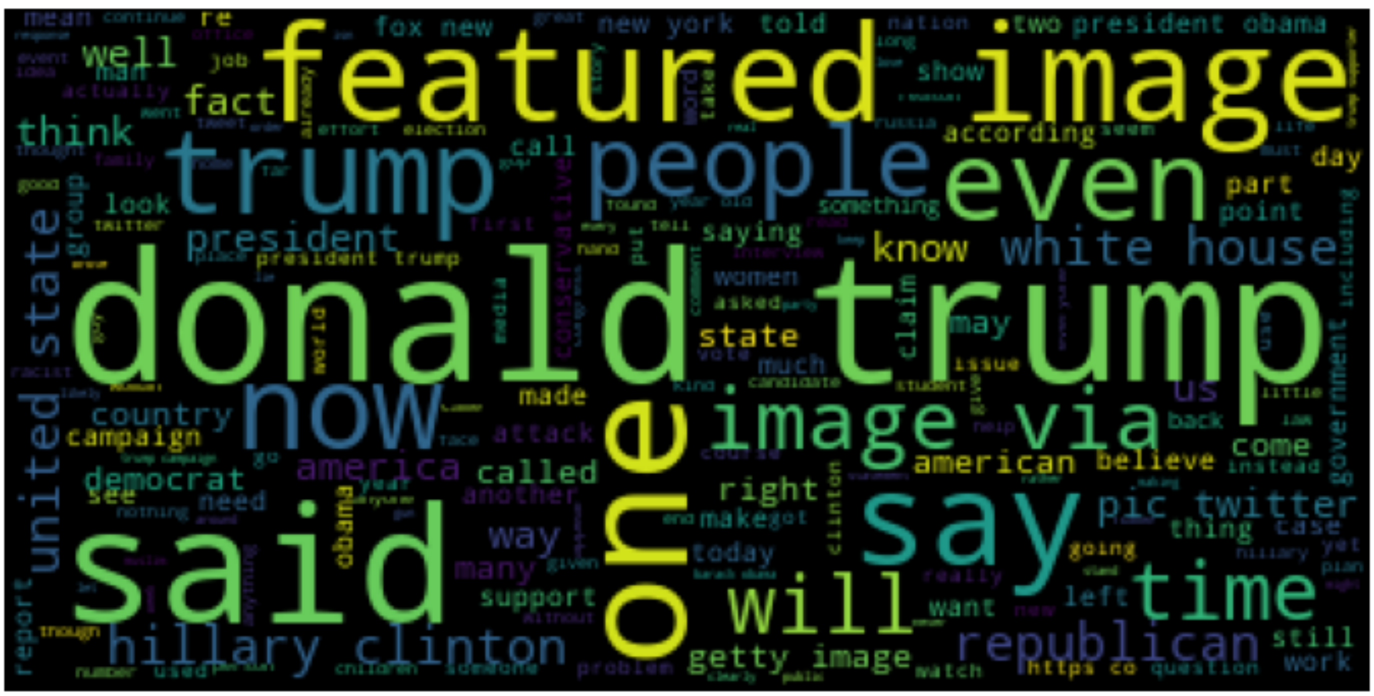
```
# word cloud for real news text
real_words = ""
for line in all_news[all_news['credibility']==1]['text']:
    line = str(line) # change each line item to string
    tokens = line.split() # split line text into word tokens

    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower() # convert each token into lower case
    real_words += " ".join(tokens)+" "

wordcloud_ = WordCloud(stopwords=stopwords).generate(real_words)
plt.figure(figsize = (12, 16), facecolor = None)
plt.axis('off')
plt.imshow(wordcloud_)
```

```
<matplotlib.image.AxesImage at 0x7ff7f4b0b150>
```



```
# word cloud for fake news text

fake_words = ""
for line in all_news[all_news['credibility']==0]['text']:
    line = str(line) # change each line item to string
    tokens = line.split() # split line text into word tokens

    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower() # convert each token into lower case
    fake_words += " ".join(tokens)+" "

wordcloud_ = WordCloud(stopwords=stopwords).generate(fake_words)
plt.figure(figsize = (12, 16), facecolor = None)
plt.axis('off')
plt.imshow(wordcloud_)
```

```
<matplotlib.image.AxesImage at 0x7ff7f6859890>
```



Although there are similar common words in both real news text and real news titles, there are still some distinguishing common words like people, featured image, percent, wednesday, thursday, tuesday, US, one, etc. This shows that the text of a news article is also a determinate factor in its credibility.

Data Preprocessing

```
all_news['news_text'] = all_news['title'] + ' ' + all_news['text']+ ' ' + all_news|
all_news.drop(['title', 'text', 'subject', 'date'], axis=1, inplace=True)
all_news
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCop
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
  """Entry point for launching an IPython kernel.
/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py:4174: SettingWithC
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
  errors=errors,
```

	credibility	news_text
0	0	Donald Trump Sends Out Embarrassing New Year'...
1	0	Drunk Bragging Trump Staffer Started Russian ...
2	0	Sheriff David Clarke Becomes An Internet Joke...
3	0	Trump Is So Obsessed He Even Has Obama's Name...
4	0	Pope Francis Just Called Out Donald Trump Dur...
...	...	...
38635	1	'Fully committed' NATO backs new U.S. approach...
38636	1	LexisNexis withdrew two products from Chinese ...
38637	1	Minsk cultural hub becomes haven from authorit...
38638	1	Vatican upbeat on possibility of Pope Francis ...
38639	1	Indonesia to buy $1.14 billion worth of Russia...

38640 rows × 2 columns

```
pd.set_option('max_colwidth', None)
all_news = all_news[['news_text', 'credibility']]
all_news.sample()
```

	news_text	credibility
	In the battle for Hollywood endorsements - and cash - Clinton rules LOS ANGELES (Reuters) - Democratic presidential candidate Bernie Sanders' positions on fracking, free tuition and breaking up big banks wouldn't sound out of place in an Oscar winning-actor's acceptance speech. But in famously liberal Hollywood, long used as an ATM by Democratic campaigns, Sanders' message is not resonating as loudly as in other progressive bastions. The more moderate Hillary Clinton has far outpaced the Vermont senator in fundraising and has a deep line-up of A-list stars and top executives among her backers. Celebrities don't sway votes, but they can persuade people to listen to a candidate's message, said historian Steven Ross, author of "Hollywood Left and Right: How Movie Stars Shaped American Politics." "It puts a candidate on their radar," he said. Hollywood actors, studio executives and other employees of the film, TV and music industries have donated at least $8.4 million to Clinton's campaign and the independent Super PAC that supports her bid, Priorities USA Action, according to a Reuters analysis of campaign finance data through March 31. A pair of Clinton fundraisers held by actor George Clooney this month, at which tickets went for as much as $353,000 per couple, is not included in that total, but were reported by Deadline Hollywood to have raised an additional $15 million. By contrast, Sanders' campaign had raised about $1 million from entertainment industry donors through March 31, according to the campaign finance data. The Vermont senator, who called the price of the Clooney event "obscene," is not associated with a Super PAC and says he does not court wealthy donors. (Graphic on Hollywood flows to Sanders and Clinton: tmsnrt.rs/1U98K4g) All Republican presidential candidates combined collected $460,000, roughly 5 percent of entertainment industry donations, the data showed. Clinton's support in Hollywood can be traced back to strong ties her husband built during his first presidential campaign in 1992, said Donna Bojarsky, a Democratic public policy consultant who worked as national entertainment coordinator for Bill Clinton's campaign. Bill Clinton connected deeply with Hollywood, she said, in part because "he showed a real respect for and appreciation of pop culture. He followed it, and he enjoyed it." Another reason for Hillary Clinton's success in Hollywood is the backing of executives. While business people in Hollywood may be liberals, they are less likely to embrace a candidate who attacks corporations, said Ross, the historian. Sanders describes himself as a democratic socialist who wants higher taxes on wealthy people and corporations to help pay for college, healthcare, and other programs. "Corporate Hollywood is about business and the bottom line," Ross said. The biggest Hollywood contributors to date are	

The newstext column contains characters like brackets, @symbols, links, and a lot of other characters or texts that might not add much information to our model, so have to clean and preprocess the data to remove such characters before we fit the text to our model.

```
import nltk
from nltk.corpus import stopwords

nltk.download('words')
nltk.download('stopwords')
stop = stopwords.words('english')
```

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

```
import re
from bs4 import BeautifulSoup

def clean_text(text):
    text = strip_html(text)
    text = remove_between_square_brackets(text)
    text = remove_stopwords(text)
    text = remove_twitter_handles(text)
    text = remove_parenthesis(text)
    return text

def strip_html(text):
    soup = BeautifulSoup(text, "html.parser")
    return soup.get_text()

#removing the square brackets
def remove_between_square_brackets(text):
    return re.sub('\[[^]]*\]', '', text)

#remove twitter handles
def remove_twitter_handles(text):
    return re.sub(r'\(@([A-Za-z0-9_]+)\)', '', text)

# Removing URL's
def remove_between_square_brackets(text):
    return re.sub(r'http\S+', '', text)
```

```
#removing parenthesis
def remove_parenthesis(text):
    return re.sub(r'\([^()]*\)', '', text)

#Removing the stopwords from text
def remove_stopwords(text):
    final_text = []
    for i in text.split():
        if i.strip().lower() not in stop:
            final_text.append(i.strip())
    return " ".join(final_text)


#Apply function on review column
all_news['news_text']=all_news['news_text'].apply(clean_text)
all_news.sample()
```

	news_text	credibility
	BALTIMORE'S OVERZEALOUS PROSECUTOR BUSTED "FAVORITING" RACIST TWEETS [Video] B b..but victim .Earlier month, two controversial tweets favorited personal Twitter account belonging Baltimore City State Attorney Marilyn Mosby. first tweet referred officers charged death Freddie Gray 6 THUG cops praised Mosby claimed INFURIATES certain kind white person. Mosby office claiming two favorited tweets work hacker. Mosby official Twitter account personal account hacked, Baltimore City State Attorney Office reportedly told Kelly File Wednesday. know long going on,	

```
nltk.download('punkt')
from nltk import word_tokenize
all_news_1 = all_news.copy()
all_news_1['news_text'] = all_news_1['news_text'].apply(lambda x: word_tokenize(str
all_news_1.sample()
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

	news_text	credibility
34182	[Iraq, top, shi'ite, cleric, Sistani, asks, government, protect, Kurds, BAGHDAD, -, Iraq, top, Shi, ite, cleric, Grand, Ayatollah, Ali, al-Sistani, Friday, called, government, protect, Kurdish, population, northern, Iraq, ., Sistani, call, ,, issued, Friday, prayer, holy, Shi, ite, city, Kerbala, one,	1

```
from nltk.stem import SnowballStemmer

snowball = SnowballStemmer(language='english')
all_news_1['news_text'] = all_news_1['news_text'].apply(lambda x: [snowball.stem(y)
all_news_1.sample()
```

	news_text	credibility
21082	[turkey, summon, u.s., envoy, washington, street, brawl, ankara, -, turkey, summon, u., ambassador, monday, protest, treatment, turkish, secur, offici, unit, state, visit, presid, tayyip, erdogan, last, week, ,, foreign, ministri, said, ., brawl, erupt, protest, turkish, secur, personnel, outsid, turkish, ambassador, ’, s, resid, erdogan, ’, s, visit, washington, meet, u.s., presid, donald, trump, .,	1

```
all_news_1['news_text'] = all_news_1['news_text'].apply(lambda x: ' '.join(x))
all_news_1.sample()
```

	news_text	credibility
	women male-domin career field watch uniqu u.s. presidenti campaign los angel - dr. linda liau work precis master , peer patient ’ s head magnifi loup remov brain tumor . liau call emerg room surgeon 20 year ago help treat car crash victim , anoth member medic team assum nurs . even today , 49-year-old neurosurgeon sometim get surpris reaction new patient expect man . assumpt common career field domin men . neurosurgeri , weld , ventur capit , construct , film direct electr trade - six job u.s. women made inroad still vast outnumb . one posit , u.s. presid , never fill woman . presumpt democrat nomine hillari clinton seek becom first break barrier , sever women career field made most men told reuter saw candidaci signific . “ i think ultim goal would gender-blind complet , fact we ’ re even talk femal presid novelti is , way , sad , ” liau said . construct site , joundi white , 31 , often remind gender . earli career , remind pet name “ sweetheart ” “ honey. ” now , rare shake sens outnumb . “ i eat lunch alon , ” white said . “ i don ’ t peopl relat work . “ don ’ t get wrong , identifi guy , them , ultim , i ’ m girl. ” wear hard hat , white pass heavi steel beam , walk along commut train track help build working-class neighborhood southern los angel . welder darlen thompson , 45 , also stranger construct site , hostil say women often encount field . day , teach	

Feature Extraction and Model Training

Using TF-IDF

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    all_news_1['news_text'],all_news_1['credibility'],
    test_size=0.3,
    stratify=all_news_1['credibility']
)
```

```
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer()
train_vectors = vectorizer.fit_transform(X_train)
test_vectors = vectorizer.fit_transform(X_test)

print("Train vector shape:",train_vectors.shape)
print("Test vector shape:", test_vectors.shape)
```

```
Train vector shape: (27048, 78754)
Test vector shape: (11592, 51026)
```

- There's a mismatch in test vector shape and train vector shape as a result, we need to reshape test vectors

```
import scipy
from scipy.sparse import csr_matrix

train_vectors = csr_matrix(train_vectors)
test_vectors = csr_matrix(test_vectors, shape = (test_vectors.shape[0], train_vecto
# creates a sparse matrix with the given shape
test_vectors
```

```
<11592x78754 sparse matrix of type '<class 'numpy.float64'>'
        with 1898025 stored elements in Compressed Sparse Row format>
```

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score


clf = MultinomialNB()
clf.fit(train_vectors, y_train)

y_pred = clf.predict(test_vectors)
print("Accuracy:", accuracy_score(y_pred, y_test))
```

```
Accuracy: 0.4747239475500345
```

```
from sklearn.metrics import classification_report

report = classification_report(y_test,y_pred)

print(report) ## check classification report
```

```
              precision    recall  f1-score   support

           0       0.46      0.92      0.61      5235
           1       0.62      0.11      0.18      6357

    accuracy                           0.47     11592
   macro avg       0.54      0.51      0.40     11592
weighted avg       0.55      0.47      0.38     11592
```

```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(solver='liblinear', penalty='l1', C=100)
clf.fit(train_vectors, y_train)

y_pred = clf.predict(test_vectors)
print("Accuracy:", accuracy_score(y_pred, y_test))
```

```
Accuracy: 0.4561766735679779
```

```
from sklearn.metrics import classification_report

report = classification_report(y_test,y_pred)

print(report) ## check classification report
```

```
              precision    recall  f1-score   support

           0       0.45      0.94      0.61      5235
           1       0.54      0.06      0.11      6357

    accuracy                           0.46     11592
   macro avg       0.49      0.50      0.36     11592
weighted avg       0.50      0.46      0.33     11592
```

Note: In previous versions of this notebook, I vectorized the data before I did the train_test split and I got an accuracy of about 95%, but I received feedback that doing so before splitting the data causes information from the training set to mix with that of the test set. After doing the train test split before vectorizing, I the accuracy has reduced drastically for each of the regression models used. This confirms that the initial 95% accuracy was not really representative of the actual model performance.

Using Word Embeddings

Creating Word Embedding from Scratch

```
all_news_2 = all_news.copy()
all_news_2.sample()
```

	news_text	credibility
	Beloved NBA Coach Openly Calls Trump Dangerous Liar San Antonio Spurs coach Gregg Popovich fan Donald Trump. sharply criticized Trump run-up election, Trump officially office, seems interest giving Orange One much inch. recent interview, Popovich openly called Trump liar, specifically criticized activities went speak CIA especially Trump speaking size crowds front wall representing fallen operatives. also said massive women march great. Popovich said: message important, could whole lot groups marching. somebody said TV, message? Well, message obvious. president comes lowest [approval] rating anybody ever came office. majority people there, since Hillary [Clinton] popular vote, buy act. Popovich went criticize Trump bigoted	

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(all_news_2['news_text']
                                                    ,all_news_2['credibility'], tes
```

```
import tensorflow
from tensorflow.keras.preprocessing.text import Tokenizer

tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_train)
# updates internal vocabulary with words in train set
# each word is represented with an integer based on the frequency of the word in th
# words with a higher frequency gets lower integer values


sequences = tokenizer.texts_to_sequences(X_train)
# creates a sequence of integers that represents each word in each row of train dat


word_index = tokenizer.word_index
# creates a dictionary of unique words and their integer values

vocab_size = len(word_index)
print('Training vocabulary size: ', vocab_size)

test_tokens = Tokenizer()
test_tokens.fit_on_texts(X_test)
test_sequences = test_tokens.texts_to_sequences(X_test)
test_word_index = test_tokens.word_index
test_vocab_size = len(test_word_index)
print('Testing vocabulary size: ', test_vocab_size )
```

```
Training vocabulary size:  113761
Testing vocabulary size:  74738
```

```
from tensorflow.keras.preprocessing.sequence import pad_sequences
X_train = pad_sequences(sequences, padding = 'post')
X_test = pad_sequences(test_sequences, padding = 'post')
```

```
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, Embedding, GlobalAveragePooling1D

embedding_dim=200

model = Sequential([
  Embedding(vocab_size + 1, embedding_dim, name="embedding"),
  Dropout(0.2),
  GlobalAveragePooling1D(),
  Dropout(0.2),
  Dense(16, activation='relu'),
  Dense(1)
])

model.summary()
```

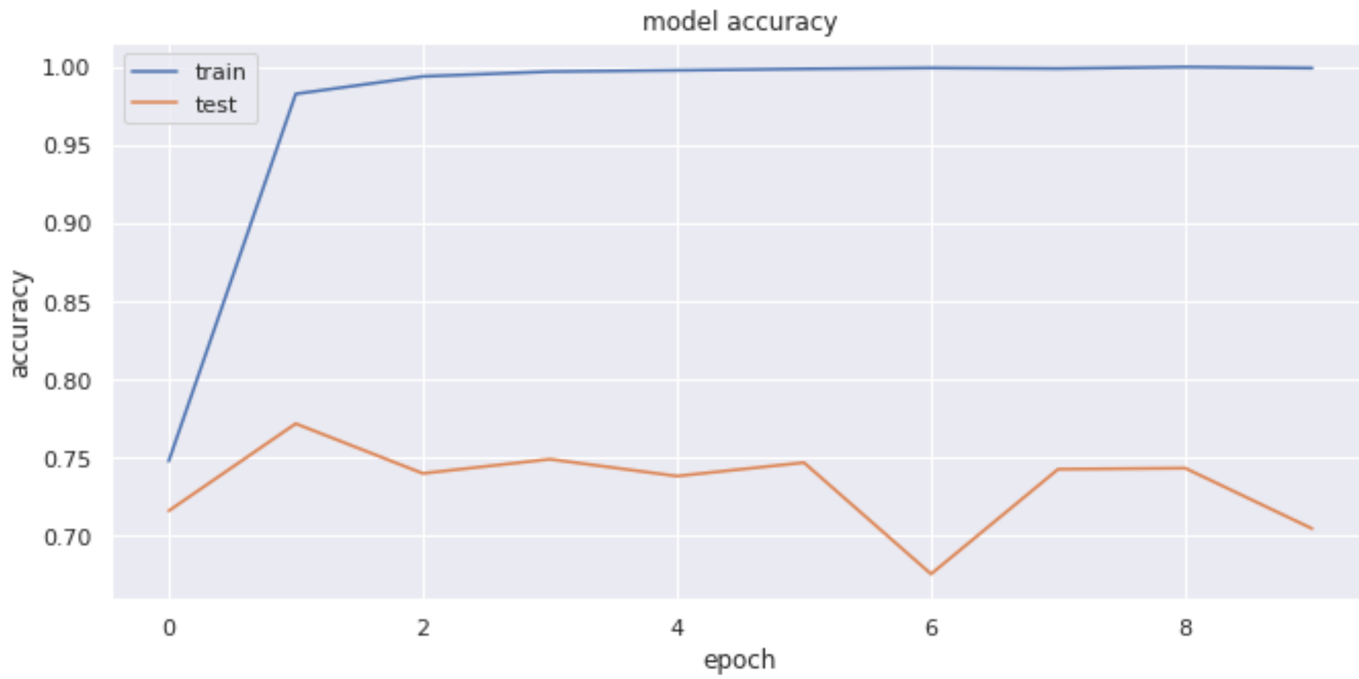
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 200)         22752400
_________________________________________________________________
dropout (Dropout)            (None, None, 200)         0
_________________________________________________________________
global_average_pooling1d (Gl (None, 200)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 200)               0
_________________________________________________________________
dense (Dense)                (None, 16)                3216
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 17
=================================================================
Total params: 22,755,633
Trainable params: 22,755,633
Non-trainable params: 0
_________________________________________________________________
```

The Embedding layer that maps from integer indices (which stand for specific words) to dense vectors (their embeddings). The dimensionality (or width) of the embedding is a parameter you can experiment with to see what works well for your problem.

```
model.compile(optimizer='adam',
              loss=tensorflow.keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=['accuracy'])


history = model.fit(
    X_train,
    y_train,
    validation_data=(X_test, y_test),
    epochs=10
    )
```

```
Epoch 1/10
846/846 [==============================] - 596s 701ms/step - loss: 0.5948 - a
Epoch 2/10
846/846 [==============================] - 600s 709ms/step - loss: 0.0978 - a
Epoch 3/10
846/846 [==============================] - 585s 692ms/step - loss: 0.0369 - a
Epoch 4/10
846/846 [==============================] - 582s 688ms/step - loss: 0.0183 - a
Epoch 5/10
846/846 [==============================] - 567s 670ms/step - loss: 0.0140 - a
Epoch 6/10
846/846 [==============================] - 559s 660ms/step - loss: 0.0075 - a
Epoch 7/10
846/846 [==============================] - 563s 666ms/step - loss: 0.0053 - a
Epoch 8/10
846/846 [==============================] - 565s 667ms/step - loss: 0.0032 - a
Epoch 9/10
846/846 [==============================] - 560s 662ms/step - loss: 0.0032 - a
Epoch 10/10
846/846 [==============================] - 557s 658ms/step - loss: 0.0025 - a
```

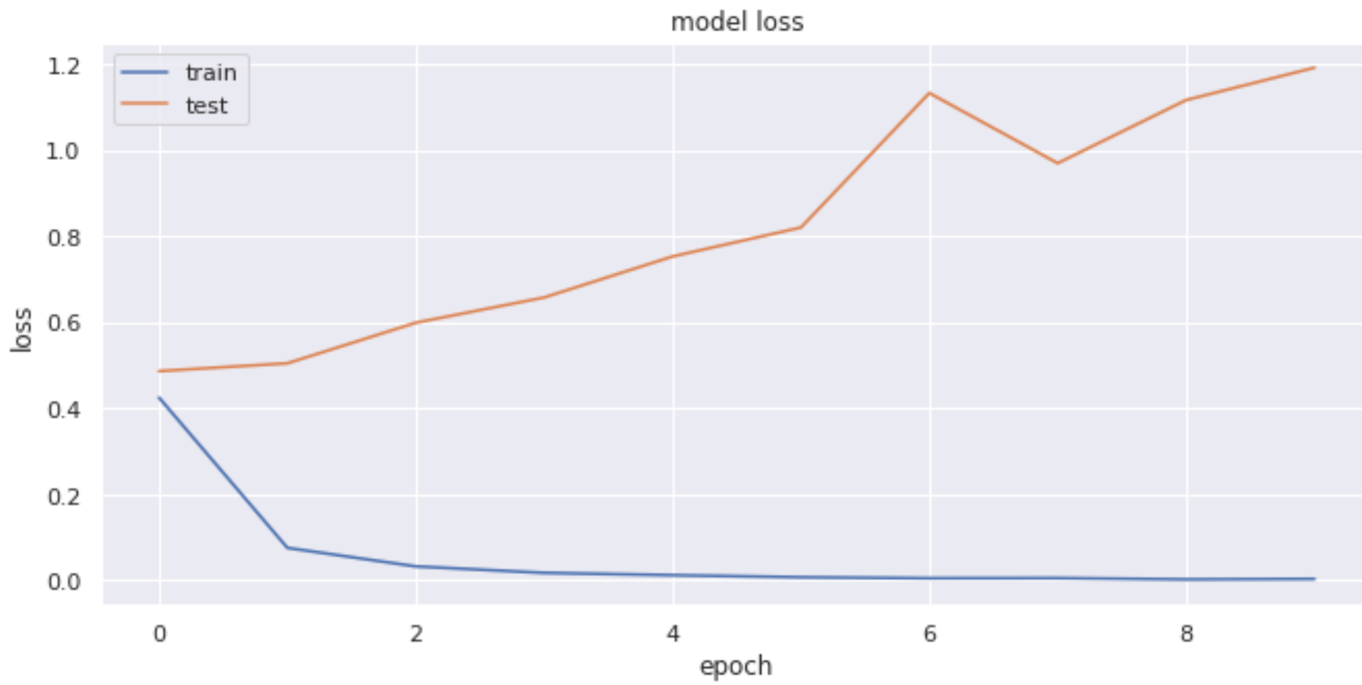
```
import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```




```
Train_results = model.evaluate(X_train, y_train, verbose=0)
Test_results = model.evaluate(X_test, y_test, verbose=0)
print(f'Train accuracy: {Train_results[1]*100:0.2f}')
print(f'Test accuracy: {Test_results[1]*100:0.2f}')
```

```
Train accuracy: 99.97
Test accuracy: 70.43
```

```
import matplotlib.pyplot as plt

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```



Using Pre-trained Word Embeddings: GloVe

```
glove_dir = '/content/gdrive/My Drive/Colab Notebooks/fake-news/glove.6B.100d.txt'
embedding_dimension = 100

embeddings_index = {}
f = open(glove_dir)
print('Loading GloVe from:', glove_dir,'...', end='')
for line in f:
    values = line.split()
    word = values[0]
    embeddings_index[word] = np.asarray(values[1:], dtype='float32')
f.close()
print("Done.\n Proceeding with Embedding Matrix...", end="")

# for train data
embedding_matrix = np.random.random((len(word_index) + 1, embedding_dimension))
for word, i in word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector


# for test data
test_embedding_matrix = np.random.random((len(test_word_index) + 1, embedding_dimer
for word, i in test_word_index.items():
    test_embedding_vector = embeddings_index.get(word)
    if test_embedding_vector is not None:
        test_embedding_matrix[i] = test_embedding_vector
print(" Completed!")
```

```
Loading GloVe from: /content/gdrive/My Drive/Colab Notebooks/fake-news/glove.(
 Proceeding with Embedding Matrix... Completed!
```

```
model = Sequential([
  Embedding(vocab_size + 1, embedding_dimension, weights = [embedding_matrix],
            name="embedding"),
  Dropout(0.5),
  GlobalAveragePooling1D(),
  Dropout(0.5),
  Dense(32, activation='relu'),
  Dropout(0.5),
  Dense(16, activation='relu'),
  Dense(1)
])

model.summary()
```

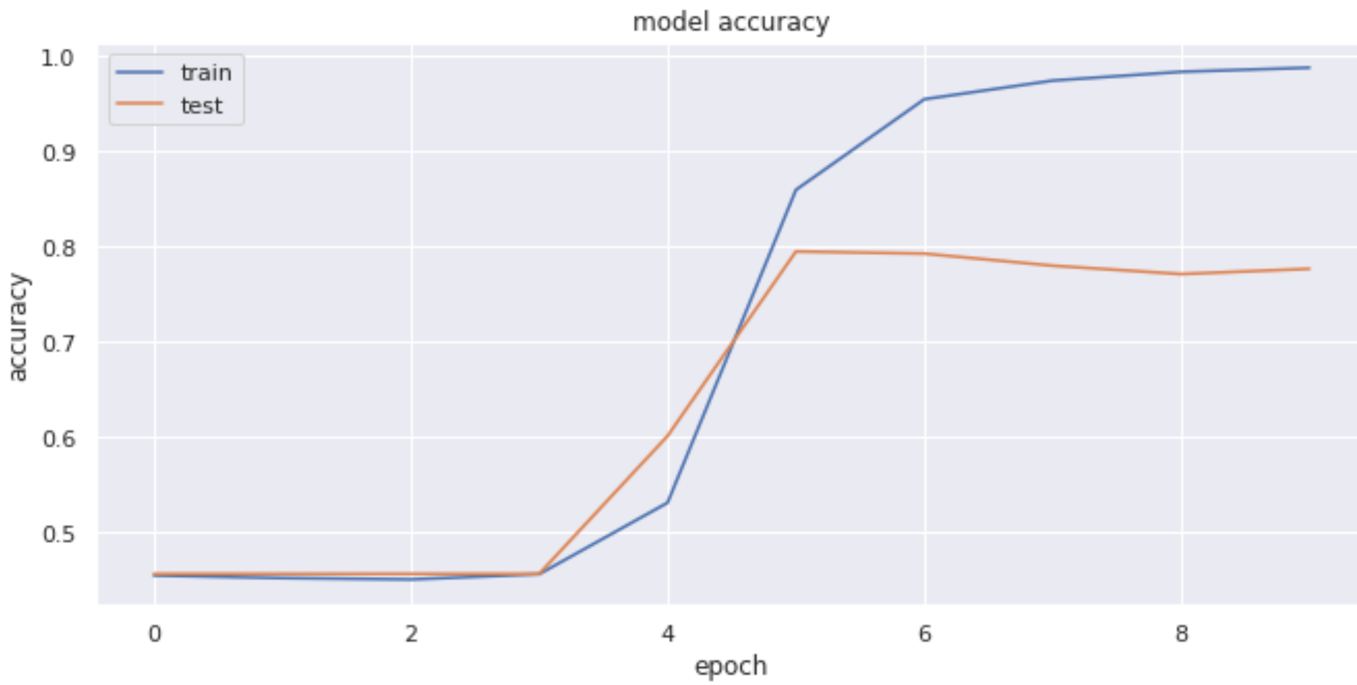
```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 100)         11376200
_________________________________________________________________
dropout_2 (Dropout)          (None, None, 100)         0
_________________________________________________________________
global_average_pooling1d_1 ( (None, 100)               0
_________________________________________________________________
dropout_3 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_2 (Dense)              (None, 32)                3232
_________________________________________________________________
dropout_4 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_3 (Dense)              (None, 16)                528
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 17
=================================================================
Total params: 11,379,977
Trainable params: 11,379,977
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(optimizer='adam',
              loss=tensorflow.keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

```
history_glove = model.fit(X_train,
    y_train,
    validation_data=(X_test, y_test),
    epochs=10)
```

```
Epoch 1/10
846/846 [==============================] - 290s 342ms/step - loss: 0.6971 - ac
Epoch 2/10
846/846 [==============================] - 286s 338ms/step - loss: 0.6886 - ac
Epoch 3/10
846/846 [==============================] - 284s 335ms/step - loss: 0.6884 - ac
Epoch 4/10
846/846 [==============================] - 283s 334ms/step - loss: 0.6863 - ac
Epoch 5/10
846/846 [==============================] - 283s 334ms/step - loss: 0.6778 - ac
Epoch 6/10
846/846 [==============================] - 279s 330ms/step - loss: 0.4156 - ac
Epoch 7/10
846/846 [==============================] - 276s 327ms/step - loss: 0.1404 - ac
Epoch 8/10
846/846 [==============================] - 274s 324ms/step - loss: 0.0835 - ac
Epoch 9/10
846/846 [==============================] - 274s 323ms/step - loss: 0.0618 - ac
Epoch 10/10
846/846 [==============================] - 277s 328ms/step - loss: 0.0423 - ac
```

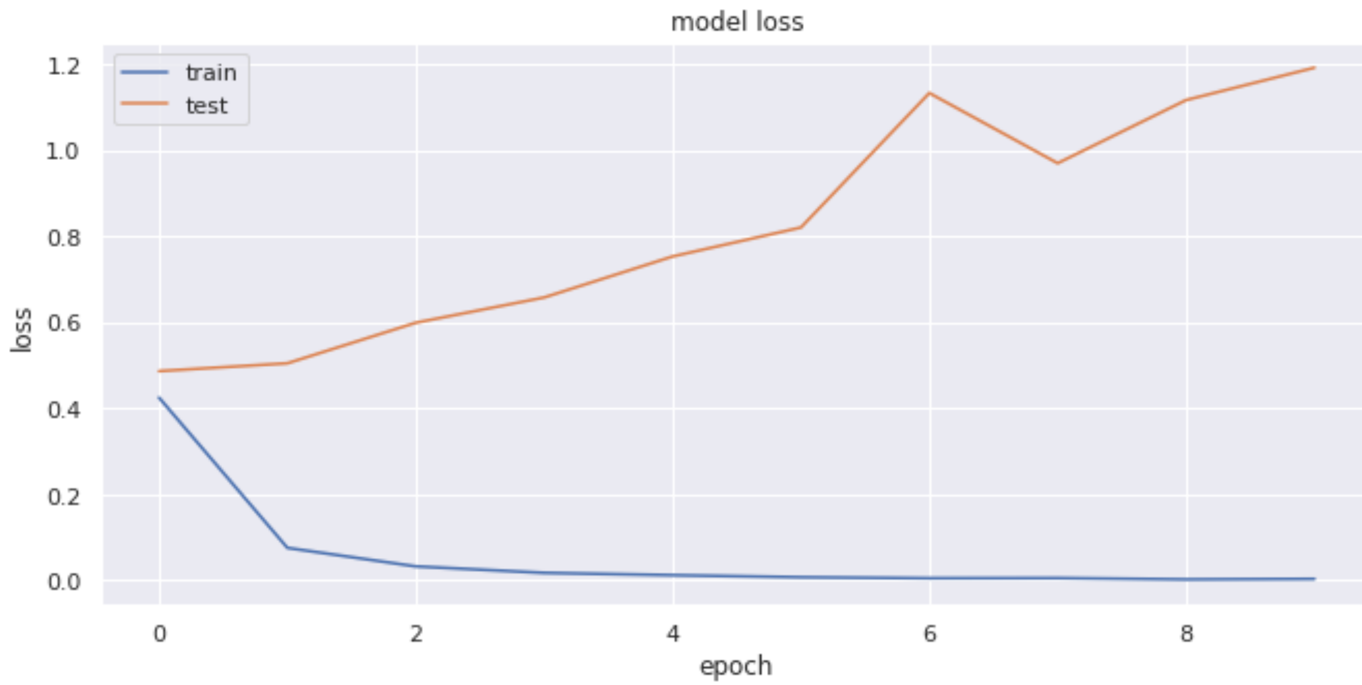
```
import matplotlib.pyplot as plt

plt.plot(history_glove.history['accuracy'])
plt.plot(history_glove.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```




```
Train_results = model.evaluate(X_train, y_train)
Test_results = model.evaluate(X_test, y_test)
print(f'Train accuracy: {Train_results[1]*100:0.2f}')
print(f'Test accuracy: {Test_results[1]*100:0.2f}')
```

```
846/846 [==============================] - 25s 29ms/step - loss: 0.0208 - accu
363/363 [==============================] - 9s 26ms/step - loss: 0.5710 - accu
Train accuracy: 99.46
Test accuracy: 77.67
```

```
import matplotlib.pyplot as plt

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```




```
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, Embedding, MaxPooling1D, Conv1D, Flatter
model1 = Sequential([
  Embedding(vocab_size + 1, embedding_dimension, weights = [embedding_matrix],
            name="embedding"),
  Dropout(0.5),
  Conv1D(64, 5, activation='relu'),
  MaxPooling1D(pool_size=4),
  LSTM(20, return_sequences=True),
  LSTM(20),
  Dropout (0.5),
  Dense(512),
  Dropout(0.5),
  Dense(256),
  Dense(1, activation='sigmoid')
```

```
])
```

```
model1.summary()
```

```
Model: "sequential_23"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding (Embedding)        (None, None, 100)         11376200  
_________________________________________________________________
dropout_34 (Dropout)         (None, None, 100)         0         
_________________________________________________________________
conv1d_52 (Conv1D)           (None, None, 64)          32064     
_________________________________________________________________
max_pooling1d_51 (MaxPooling (None, None, 64)          0         
_________________________________________________________________
lstm_7 (LSTM)                (None, None, 20)          6800      
_________________________________________________________________
lstm_8 (LSTM)                (None, 20)                3280      
_________________________________________________________________
dropout_35 (Dropout)         (None, 20)                0         
_________________________________________________________________
dense_60 (Dense)             (None, 512)               10752     
_________________________________________________________________
dropout_36 (Dropout)         (None, 512)               0         
_________________________________________________________________
dense_61 (Dense)             (None, 256)               131328    
_________________________________________________________________
dense_62 (Dense)             (None, 1)                 257       
=================================================================
Total params: 11,560,681
Trainable params: 11,560,681
Non-trainable params: 0
_________________________________________________________________
```

```
model1.compile(optimizer='adam',
              loss=tensorflow.keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

```
history1 = model1.fit(X_train,
    y_train,
    validation_data=(X_test, y_test),
    epochs=5)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-54ddc46fb392> in <module>()
----> 1 history1 = model1.fit(X_train,
      2     y_train,
      3     validation_data=(X_test, y_test),
      4     epochs=5)

NameError: name 'model1' is not defined
```

SEARCH STACK OVERFLOW

```
import matplotlib.pyplot as plt

plt.plot(history1.history['accuracy'])
plt.plot(history1.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

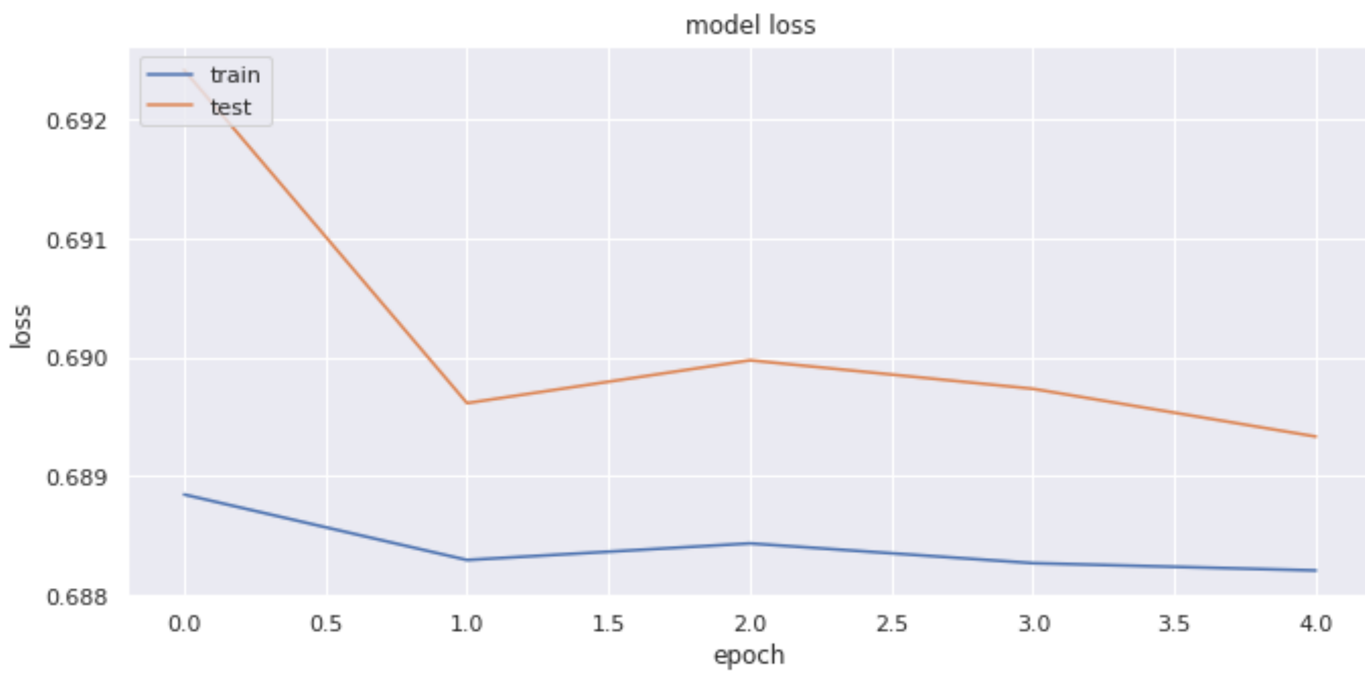
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-a9ba30a72a32> in <module>()
      1 import matplotlib.pyplot as plt
      2
----> 3 plt.plot(history1.history['accuracy'])
      4 plt.plot(history1.history['val_accuracy'])
      5 plt.title('model accuracy')

NameError: name 'history1' is not defined
```

SEARCH STACK OVERFLOW

```
Train_results = model1.evaluate(X_train, y_train)
Test_results = model1.evaluate(X_test, y_test)
print(f'Train accuracy: {Train_results[1]*100:0.2f}')
print(f'Test accuracy: {Test_results[1]*100:0.2f}')
```

```
846/846 [==============================] - 258s 304ms/step - loss: 0.6881 - ac
363/363 [==============================] - 103s 285ms/step - loss: 0.6893 - ac
Train accuracy: 55.05
Test accuracy: 54.37
```

```
import matplotlib.pyplot as plt

plt.plot(history1.history['loss'])
plt.plot(history1.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```



```
model2 = Sequential([
  Embedding(vocab_size + 1, embedding_dimension, weights = [embedding_matrix],
            name="embedding"),
  LSTM(20, return_sequences=True),
  LSTM(20),
  Dropout (0.5),
  Dense(512),
  Dropout(0.5),
  Dense(256),
  Dense(1, activation='sigmoid')
])
```

```
model2.summary()
```

```
Model: "sequential_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 100)         11376200
_________________________________________________________________
lstm_5 (LSTM)                (None, None, 20)          9680
_________________________________________________________________
lstm_6 (LSTM)                (None, 20)                3280
_________________________________________________________________
dropout_11 (Dropout)         (None, 20)                0
_________________________________________________________________
dense_12 (Dense)             (None, 512)               10752
_________________________________________________________________
dropout_12 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_13 (Dense)             (None, 256)               131328
_________________________________________________________________
dense_14 (Dense)             (None, 1)                 257
=================================================================
Total params: 11,531,497
Trainable params: 11,531,497
Non-trainable params: 0
_________________________________________________________________
```

```
model2.compile(optimizer='adam',
              loss=tensorflow.keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

```
history2 = model2.fit(X_train,
    y_train,
    validation_data=(X_test, y_test),
    epochs=5)
```

```
Epoch 1/5
141/846 [====>.........................] - ETA: 1:44:47 - loss: 0.6921 - accur
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-78-51217dda1542> in <module>()
      2     y_train,
      3     validation_data=(X_test, y_test),
----> 4     epochs=5)

6 frames
/usr/local/lib/python3.7/dist-packages/tensorflow/python/eager/execute.py in
quick_execute(op_name, num_outputs, inputs, attrs, ctx, name)
     58     ctx.ensure_initialized()
     59     tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle, device_name,
op_name,
---> 60                                         inputs, attrs, num_outputs)
     61   except core._NotOkStatusException as e:
     62     if name is not None:

KeyboardInterrupt:
```

Next Step

Although the accuracies were high in both models, the validation accuracies did not improve much and were lower which shows both models were overfitting. The next step would be to improve the architecture of the neural networks to see if the validation accuracies improve