FAKE NEWS DETECTION USING PASSIVE AGGRESSIVE CLASSIFIER

BY:
RASHMI BONGIRWAR
NARASIMHA SIVA SAKETH EMANI

INTRODUCTION

The age of digitalization has come with its set of pros and cons. When we talk about news, earlier they used to be spread only via non digital means, but now major source of information has become social media and other online platforms. It is good because this means wider reach to public but has also become dangerous since there is often spread of fake news too.

MOTIVATION

So, what is the solution for the problem of spread of false information? Try best that it is filtered. This filtering can be done by a few ways, one of which includes big data analysis. That is the aim for us selecting this topic combining it with usage of cloud resources to make the whole project more efficient.

METHOD USED

First, we chose the dataset and decided on performing prediction via classification using passive and aggressive classifier. The aim was to predict correctly whether the given text is real or fake. We chose to use distributed data processing system Apache Spark for Amazon EMR, S3 for storage and retrieval of data.

DATASET

We downloaded huge csv type of dataset having four columns:

- Identification number
- Title of news
- Text
- Label (REAL or FAKE)

Dataset:
https://drive.google.com/file/d/1oC0l_4okLSooKSzcixtMNlzem8H2fG2m/view?usp=sharing

DATASET continued

We chose to use a csv type of format because of its advantages of being structured and availability of headers. We performed pre- processing on the data set which included removal of stop words to make out prediction as accurate as possible.

STORAGE

- The method we used for storage was S3 and uploaded the files on it.
- S3 stores files in a flat organization of containers which are called buckets.
- It is highly scalable.
- Individual S3 objects can store as much as 5 TB of data.

WHY AMAZON S3?

- We chose it over options like NoSQL database because we needed a service that provides easy storage and retrieval. Database creation was not required.
- It was a convenient one stop shop to store our dataset as well as data analysis code which could be accessed easily from anywhere on the web at any time.

DATA PROCESSING SYSTEM

- Distributed data processing system was the heart of our project.
- Amazon EMR enables easy processing of vast amount of data.
- It utilizes a hosted Hadoop framework (the engine of EMR) running on infrastructure of EC2 and S3.
- It allows to instantly perform data- intensive tasks for various types of analysis.

DATA PROCESSING SYSTEM continued

- We chose Apache Spark as our distributed system because of its fast execution over data of any size.
- It is because of in- memory caching feature of Spark.

DATA ANALYSIS

- The data that we analysed is the news articles data.
- Each article is labelled as REAL or FAKE
- Our machine Learning model analyses the data and predicts the credibility of the news

Why EMR?

- The same script can be run on a local machine too
- But the EC2 instance running on a cloud using EMR processing is efficient
- EC2 is scalable
- EC2 allocates resources on demand
- An EMR is further efficient than a single EC2 instance because EMR operates on master-slave nodes
- EMR accesses s3 storage faster than individual EC2

SSH into the EMR instances

- SSH facilitates access to remote systems using secure key
- The key is generated in AWS and used by the local machine as a signature to prove its authenticity.
- Ssh -i ~/Downloads/cloudprojectkey.pem hadoop@ec2-54-186-83-102.us-west-2.compute.amazonaws.com

Confirmation that the connection is made to the remote machine on cloud

```python
# Importing necessary Packages
import numpy as np
import pandas as pd
import itertools
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

# Converting the .csv file to pandas dataframe
df=pd.read_csv('C:/Users/saket/Cloud_Project/news.csv')
df.shape
df.head()

labels=df.label
labels.head()

# dividing the dataset into train and test dataset
x_train,x_test,y_train,y_test=train_test_split(df['text'], labels, test_size=0.2, random_state=7)

# Creating the tfidf vectorizer to convert the news
# articles to vectors for mathematical calculations and deriving relation
tfidf_vectorizer=TfidfVectorizer(stop_words='english', max_df=0.7)

tfidf_train=tfidf_vectorizer.fit_transform(x_train)
tfidf_test=tfidf_vectorizer.transform(x_test)

#Passive aggressive classifier model created and trained based on the training vectors created above
pac=PassiveAggressiveClassifier(max_iter=50)
pac.fit(tfidf_train,y_train)

# Predicting the output of the test data and calculating the accuracy
y_pred=pac.predict(tfidf_test)
score=accuracy_score(y_test,y_pred)
print(y_pred)
print(f'Accuracy: {round(score*100,2)}%')

confusion_matrix(y_test,y_pred, labels=['FAKE','REAL'])
print(str(confusion_matrix))
```

RESULTS

- Files copied to EMR master node instance
  wget https://<url of the file>
- The accuracy of the predictions - 92.82%
- The confusion matrix - [[587 51]
  [43 586]]
- True Positives = 587
- True Negatives = 586
- False positives = 51
- False Negatives = 43
- So, Precision - 0.92
- Recall - 0.93

CONCLUSION

- Understood the impact of EMR on Big data
- Leant the advantages of EMR over normal EC2 instances
- Gained hands on experience in accessing AWS cloud and creating instances using EMR
- Used SSH to access remote machines which helped us learn the basics of SSH and linux

THANK YOU