

**Problem Statement**

In this project, we have used various natural language processing techniques and machine learning algorithms to classify fake news articles using Machine Learning and Deep Learning models.

**Prerequisites**

Python 3.6

Libraries:

* Sklearn (scikit-learn)
* numpy
* scipy
* numpy
* torch
* torchvision
* matplotlib
* tensorflow
* tqdm
* pandas
* numpy
* keras
* sklearn
* nltk

pip install –r requirement.txt

**Dataset Description**

* train.csv: A full training dataset with the following attributes:
  + id: unique id for a news article
  + title: the title of a news article
  + author: author of the news article
  + text: the text of the article; could be incomplete
  + label: a label that marks the article as potentially unreliable
    - 1: unreliable
    - 0: reliable
* test.csv: A testing training dataset with all the same attributes at train.csv without the label.

**File Structure**

.

|

+-- input

++-- data

| +-- train.csv

| +-- test.csv

+-- src

| +-- config.py

| +-- dataset.py

| +-- mlalgorithms.py

| +-- engine.py

| +-- model.py

| +-- predictML.py

| +-- train.py

+-- README.txt

+-- requirements.txt

EDA:

Note: Detailed analysis mentioned in the notebook.

* Read file
* Check null, duplicate and impute/delete
* Feature Creation:
  + Text Length
  + Title Length
  + Text Word Count
  + Title Word Count
* Feature Creation on clean data :
  + clean\_text\_len
  + clean\_text\_word\_count
  + clean\_text\_title\_len
  + clean\_text\_title\_word\_count
* Extract POS tag feature and create other:
  + clean\_text
  + text\_len
  + text\_word\_count
  + text\_unique\_word\_count
  + title\_len
  + title\_word\_count
  + title\_unique\_word\_count
  + VERBRatio
  + NOUNRatio
  + PRONRatio
  + ADJRatio
  + ADVPNRatio
  + ADPRatio
  + CONJRatio
  + DETRatio
  + NUMRatio
  + PRTRatio
  + PUNCRatio
  + ActionCount
  + acronym\_to\_activity\_ratio
  + acronym count
  + num value count
  + is\_len\_range\_1\_500
  + is\_len\_range\_400\_1100
  + is\_len\_range\_22\_80
* Univariate Analysis
  + Distribution of text length
  + Distribution of text word count
  + Distribution of Title length
  + Distribution of Title word count
  + Check dataset is balanced or not
  + Calculate Unigram, Bigram & Trigram of fake and real news and analyze
    - Before processing
    - after processing
  + Process Text and Title of ‘fakenews’, ‘realnews’ dataframe and original data’s dataframe ‘df’
  + Wordcloud graph for ‘fakenews’ and ‘realnews‘
* Bivariate Analysis
  + The distribution of top part-of-speech tags of Text & Title corpus
    - Compare text length with label
    - Compare text word count with label
    - Compare title length with label
    - Compare title word count with label
    - Compare cleaned text word count with label
    - Compare cleaned title word count with label
    - Compare author with label
    - \*compare the records with label which word count is less than 10
    - \*compare the records with label which word count is greater than 100
* Feature Selection
* Run the xgb and extract features which are important

**Train Model**

**Model Parameters:**

* MAX\_LEN = 512
* TRAIN\_BATCH\_SIZE = 64
* TEST\_BATCH\_SIZE = 32
* EPOCHS = 10
* MODEL\_PATH = "../input/model/model.pt"
* TRAINING\_FILE = "../input/data/train.csv"
* NEW\_TRAINING\_FILE = "../input/data/train\_with\_new\_features.csv"
* TESTING\_FILE = "../input/data/test.csv"
* GLOVE="../input/model/glove.6B.300d.txt"
* MODEL\_PATH\_ML = "../input/model"

**Machine Learning**

Execute below command to train machine learning models [Random Forest, Navie Bayes, XGBOOST, Logistic Regression and saves the trained model to MODEL\_PATH\_ML directory

Python mlalgorithms.py

**Deep Learning**

Execute below command to train deep learning models [LSTM,BERT] and saves the model to MODEL\_PATH directory

Python train.py

**Precision:**

**Recall:**