Identifying Twitter troll accounts functioning in the Mueller investigation using Machine Learning

Authors: Aman Wagadre, Manisha Chaurasia, Piyusha Sane, Rishabh Kumar, Tushar Rathi

Introduction

The Mueller investigation held during the 2016 U.S. Presidential Election revealed 200,000 tweets that were linked to “malicious activity”. These accounts were linked to Russia which then led to the discovery of large networks of accounts that pushed provocative tweets in order to stir up negative feelings about one of the candidates involved. Further investigation showed the use of Russian bots besides manually controlled accounts involved in the incident. This paper works towards developing new methods/ideas for discovering twitter troll accounts using Machine Learning as a medium.

The goal for these troll accounts is generally to spread hostility, doubt and trepidation. These accounts have been found to impersonate people on either end of the political spectrum, e.g. activists of the Black Lives Matters movement as well as ardent Donald Trump supporters. Datasets for these accounts are available and can be leveraged to study certain characteristics of these accounts via Machine Learning.

Python 2.75 is used to achieve the analysis required to segregate the data into readable and computable format before applying several mathematical models in order to learn the data.

Problem Statement

2,425,909 tweets collected between March 28 and April 5 targeting the Mueller incident was used as a dataset. This data was unclassified test data. Besides these accounts, a list of twitter troll accounts released during the same period by Twitter was used as classified data for troll accounts.

The basic idea here was to classify these accounts based on their twitter activity as troll accounts or normal accounts. Each tweet had features like: ids, followers count, statuses count, time zone, language, verification status, friends count, listed count, number of tweets, text of these tweets, retweets etc. but this dataset was unclassified. The accounts extracted from the NBC troll accounts data also had similar features in addition to the fact that these were classified data.

The datasets had accounts that could be categorized as:

1. Automated bots that mostly spam hashtags and retweet links to Kremlin-aligned articles.

2. Accounts operated by professional social media trolling operations, mostly in Russia.

3. Americans, mostly alt-right adherents, who post and retweet the same material as the bots and professional trolls.

Identification of important features given as well as additional features which could be engineered were important aspects of the problem. This required a concatenation of the two datasets followed by identification of the training, validation and test datasets for Machine Learning.

Methods

The data had to be concatenated into a single dataset using the two acquired. This was done using the common parameter, user id. For each account id the features accompanying it were: followers count, statuses count, time zone, language, verification status, number of active rate, all the tweets associated with the account during the incident and number of retweets. Activity rate was computed using the number of tweets during the Mueller incident. Different tweets under the same account were clubbed together to get separate databases of tweets for each profile required for the semantic features. The former mentioned features were clubbed as profile features.

Since the model used was a mathematical model, the features had to be modified in order to affect the model parameters. A categorical approach was followed to transform the data into readable and computable data. For e.g. languages (English, Russian, Spanish etc.) were given different numbers like 1, 2 and 3. Similarly verification status came under binary classification (1- verified; 0- non-verified). The purpose of our model was to recognize the importance of each of the features given to it in order to come up with a substantial way to spot troll accounts.

Developing a classified training and validation dataset was done through careful identification of non-troll accounts which were then coupled with the troll accounts dataset obtained from NBC. This was achieved through classification of verified accounts as non-troll accounts.

Model

The objective was to differentiate between the troll twitter accounts and real twitter accounts by using a Multi-layer perceptron. Each of the feature extracted from the twitter dataset were converted to numerical values by using categorical segregation. The features were divided mainly into two parts 1) Profile feature 2) Semantic features. A total of 9 profile features and the average value of part of speech tags for all the 36 tags were used as the input values of the multilayer perceptron.

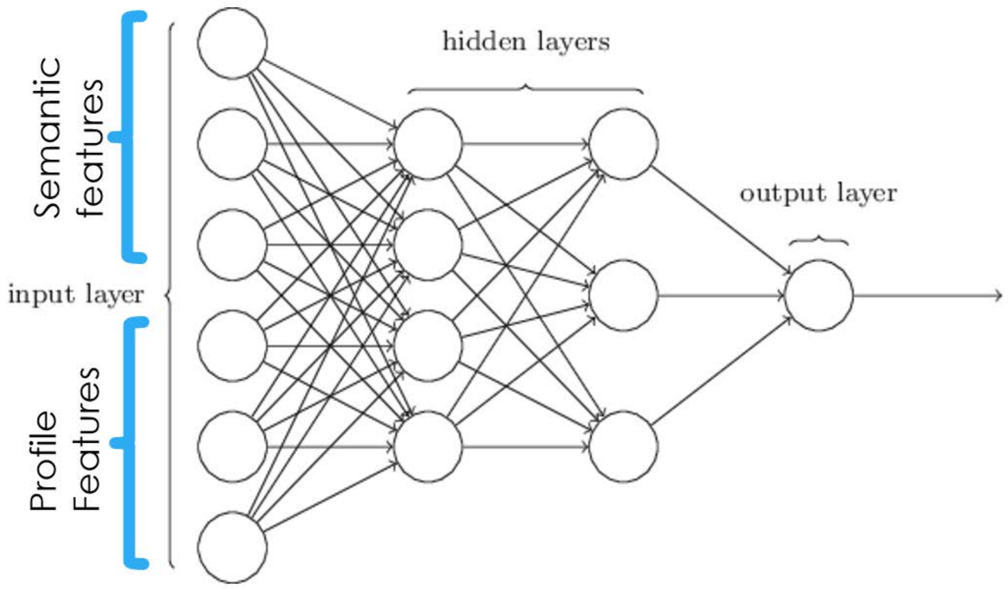

Fig.1 Multilayer Perceptron

Combination of these two features can be amalgamated with our model to get weights for each feature. Hyperparameters like number of hidden layers and the number of activation units can be optimized using comparison of test and training errors.

Conclusion

The main work done up to now was the extraction of features which can be used to identify troll accounts by using there profile and semantic features. The Multilayer perceptron model proposed for this problem is yet to be used on this dataset. The number of hidden layers and hidden units will only be determined after the tuning of this model for this particular dataset.

Appendix

```
from sklearn.neural_network import MLPClassifier

import csv

import numpy as np

from sklearn.metrics import classification_report,confusion_matrix

from textblob import TextBlob



### This model only input the profile and semantic features of troll profiles

### Profiles and semantic features of real accounts has to be appended to the X (input vector) and Y
(output vector {0 for fake accounts and 1 for real accounts})


### Input of profile features from csv file

inputprofilecsv="introllprof.csv"

read2=csv.reader(open(inputprofilecsv,newline=""),delimiter=",")

i=0

X=[]

for row2 in read2:

        i+=1

        if i==1:

                continue

        else:

                row2=np.array(row2)

                inp2 = row2.astype(np.float)

                inp2=list(inp2)

                X.append(inp2)


### Input of tweet semantic feature from csv file

inputtweetscsv="introlltweets.csv"

read=csv.reader(open(inputtweetscsv,newline=""),delimiter=",")
```

```
j=0
for row in read:
        j=j+1
        if j==126:
                break
        else:
                moditags=["CC","CD","DT","EX","FW","IN","JJ","JJR","JJS","LS","MD","NN",
                "NNS","NNP","NNPS","PDT","POS","PRP","PRP$","RB","RBR","RBS","RP","SYM",
                "TO","UH","VB","VBD","VBG","VBN","VBP","VBZ","WDT","WP","WP$","WRB"]
                tagcount=np.zeros(37,dtype=float)
                tagcoutn=list(tagcount)
                str=TextBlob(row[0])
                t=str.tags
                sent=str.sentiment.polarity
                for i in t:
                        ind=moditags.index(i[1])
                        tagcount[ind]=tagcount[ind]+1
                tagcount[36]=sent
                tagcount=list(tagcount)
                e=0
                while e!=37:
                        X[j-1].append(tagcount[e])
                        e=e+1


# X=[[0.0,0.0],[1.0,1.0]]
Y=np.zeros(j-1)
# Y=np.random.randint(2, size=i-1)
# Y=np.random.randint(2, size=2)
clf = MLPClassifier(solver='lbfgs', alpha=1e-5,hidden_layer_sizes=(30,30), random_state=1)
```

```
print(clf.fit(X,Y))
```