DEBATE NIGHT-SOCIALBOTS

THE AUSTRALIAN NATIONAL UNIVERSITY

Abhinav Pandey u6724645

Harpragaas Singh u6424131

1. INTRODUCTION

Serious concerns have been raised about the role of 'socialbots' in manipulating public opinion and influencing the outcome of elections by retweeting partisan content to increase its reach. Here we analyze the role and influence of socialbots on Twitter by determining how they contribute to retweet diffusions. The study describes a way to predict the botscore of all the users and then do classification to identify bots from the given dataset

2. DATA PREPROCESSING

2.1 Problems with the Dataset

Data from the real world is never perfect and the data set used in this paper is not an exception. The features of the data that represents various features of the tweet and the user contained several forms of improper and incomplete data. Following are the classes of problems observed:

1. Missing Attribute Values :

   Data contains mostly numeric and categorical attributes where each attribute had missing values.
   Thus it was concluded that because of missing values the data has to be *transformed* properly.

2. Improper types:

   For an efficient and meaningful data analysis, The attribute values must be of same type and should be consistent.

2.2 Data Preprocessing Techniques

There were two major objectives for using data preprocessing techniques:

a) To learn more about the nature of the data

b) To solve the above mentioned problems in our data

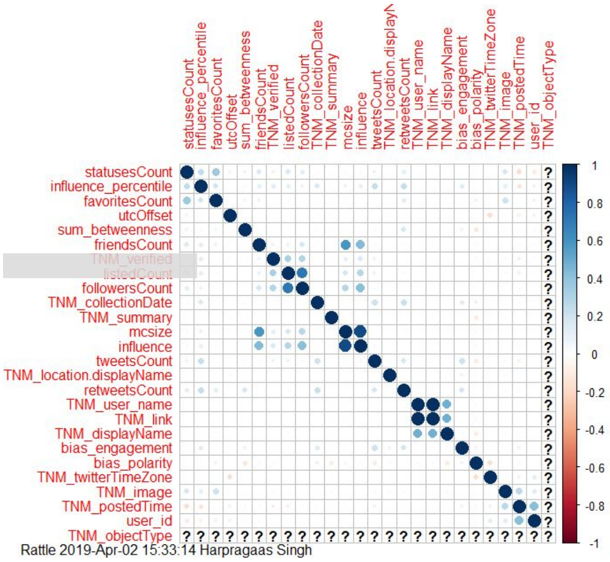


Fig 1 Correlation between all the features

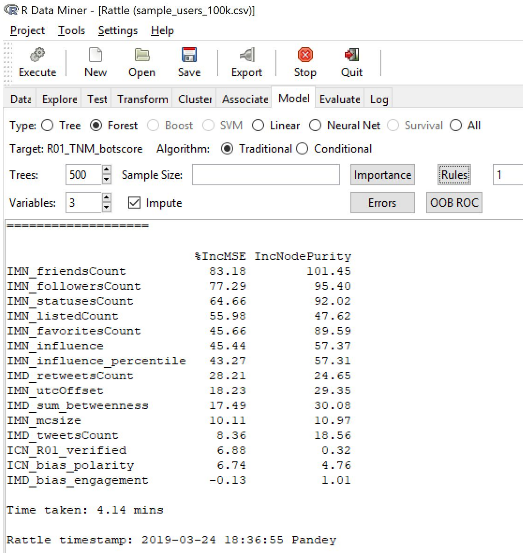
In the Fig1, we observe and compare the strength of correlation between variables of dataset and observe the importance of each feature to decide which feature to be considered for calculating the bot_score,*regression* and predicting whether the user is a bot or not,*classification.*

To solve the problems of the dataset and select the features that will be used for the study, the following approach was used:

Categorical features can only take on a limited and usually a fixed string of value thus the categorical features were ignored. The verified is converted to binary where True is set to 1 False is set to 0 and NA is set to 0. For features, *tweets_count, retweet_scount , bias_engagement and sum_betweenness* were imputed such that the missing values in this feature were set to median while for the remaining features missing values were set to mean of the dataset. The target variable,*botscore* was transformed to numeric and scaling it's value between 0 and 1.

2.3 Feature Importance

Rattle GUI is used in order to determine the importance of the features described above. The features were used to model a Random Forest with 500 trees and 3 variables in GUI. The model gave a Variable Importance table that will be used to further filter out the variables needed for the target variable.


R Data Miner - [Rattle (sample_users_100k.csv)]
Project Tools Settings Help
Execute New Open Save Export Stop Quit
Data Explore Test Transform Cluster Associate Model Evaluate Log
Importance Rules Errors OOB ROC

Type: ○ Tree ● Forest ○ Boost ○ SVM ○ Linear ○ Neural Net ○ Survival ○ All

Target: R01_TNM_botscore Algorithm: ● Traditional ○ Conditional

Trees: 500 Sample Size: 1

Variables: 3 ☑ Impute

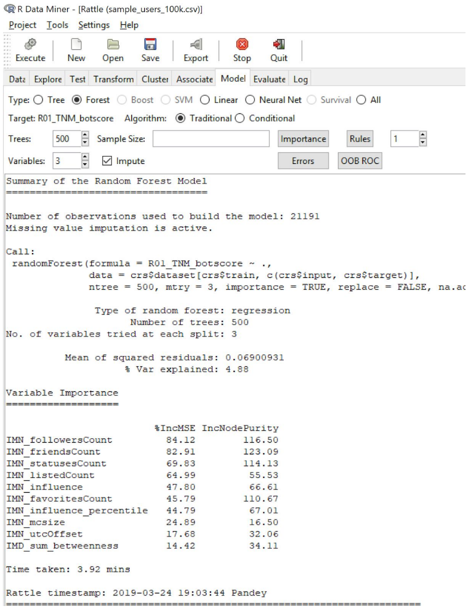
```
===================


                          %IncMSE IncNodePurity
IMN_friendsCount            83.18        101.45
IMN_followersCount          77.29         95.40
IMN_statusesCount           64.66         92.02
IMN_listedCount             55.98         47.62
IMN_favoritesCount          45.66         89.59
IMN_influence               45.44         57.37
IMN_influence_percentile    43.27         57.31
IMD_retweetsCount           28.21         24.65
IMN_utcOffset               18.23         29.35
IMD_sum_betweenness         17.49         30.08
IMN_mcsize                  10.11         10.97
IMD_tweetsCount              8.36         18.56
ICN_R01_verified             6.88          0.32
ICN_bias_polarity            6.74          4.76
IMD_bias_engagement         -0.13          1.01

Time taken: 4.14 mins

Rattle timestamp: 2019-03-24 18:36:55 Pandey
```

Fig 2 : Important Features

In Fig 2 we determine the feature importance based on IncMSE, where higher the IncMSE more important is the corresponding attribute. Further, the variable with least importance is ignored and the model is rebuilt again to determine the importance of the remaining features.

The following features *bias_engagement, bias_polarity and verified* were removed in the same order to check for the feature importance.

Fig 3 displays the most important features that were later used to model regressor and classifier, using the train function from the caret package, for determining the *botscore* and determining *bot_or_not*.


R Data Miner - [Rattle (sample_users_100k.csv)]
Project Tools Settings Help
Execute New Open Save Export Stop Quit
Data Explore Test Transform Cluster Associate Model Evaluate Log
Importance Rules Errors OOB ROC

Type: ○ Tree ● Forest ○ Boost ○ SVM ○ Linear ○ Neural Net ○ Survival ○ All

Target: R01_TNM_botscore Algorithm: ● Traditional ○ Conditional

Trees: 500 Sample Size: 1

Variables: 3 ☑ Impute

```
Summary of the Random Forest Model
==================================

Number of observations used to build the model: 21191
Missing value imputation is active.

Call:
 randomForest(formula = R01_TNM_botscore ~ .,
              data = crs$dataset[crs$train, c(crs$input, crs$target)],
              ntree = 500, mtry = 3, importance = TRUE, replace = FALSE, na.ac

               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 3

          Mean of squared residuals: 0.06900931
                    % Var explained: 4.88

Variable Importance
===================

                          %IncMSE IncNodePurity
IMN_followersCount          84.12        116.50
IMN_friendsCount            82.91        123.09
IMN_statusesCount           69.83        114.13
IMN_listedCount             64.99         55.53
IMN_influence               47.80         66.61
IMN_favoritesCount          45.79        110.67
IMN_influence_percentile    44.79         67.01
IMN_mcsize                  24.89         16.50
IMN_utcOffset               17.68         32.06
IMD_sum_betweenness         14.42         34.11

Time taken: 3.92 mins

Rattle timestamp: 2019-03-24 19:03:44 Pandey
======================================================================
```

Fig 3 : Final Important Features

3. REGRESSION TASK

3.1 REGRESSION MODELS

While several Data Mining Models are available for the task, the **Random Forest** model is used for the regression task.The reason for selecting the model is, Random Forest improves on the concept of Decision Tree.

3.1.1 Decision Tree vs Random Forest

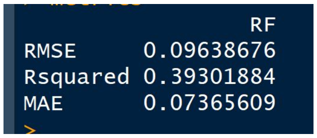
In the Decision Tree model,the data is split into smaller data groups based on features of the data. The splits are chosen depending on the purity measure. At each node, the information gain has to be maximised. However one single decision tree has high variance, *tends to overfit*, by bagging the variance is averaged away, giving the majority vote. Random Forest are nothing but bagged Decision Tree models.

3.1.2 Neural Networks vs Random Forest

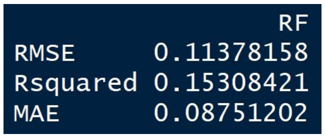
Neural networks are used to learn and discover inherent features in objects that will be helpful in their prediction,it may require a lot of data making it computationally expensive in comparison to Random Forest.

3.2 OBSERVATIONS

In the regression task, Random Forest model performed statistically better than Neural Network model and Decision Tree model with an RMSE value of **0.09638** as opposed to **0.1137** of the Decision Tree model and with an R-squared value of 0.3930 as opposed to 0.003 of the Neural Network Model



```
                 RF
RMSE     0.09638676
Rsquared 0.39301884
MAE      0.07365609
>
```



```
                 RF
RMSE     0.11378158
Rsquared 0.15308421
MAE      0.08751202
```

(a) (b)

Fig 4: Showing RMSE values for (a)Random Forest and (b) Decision Tree

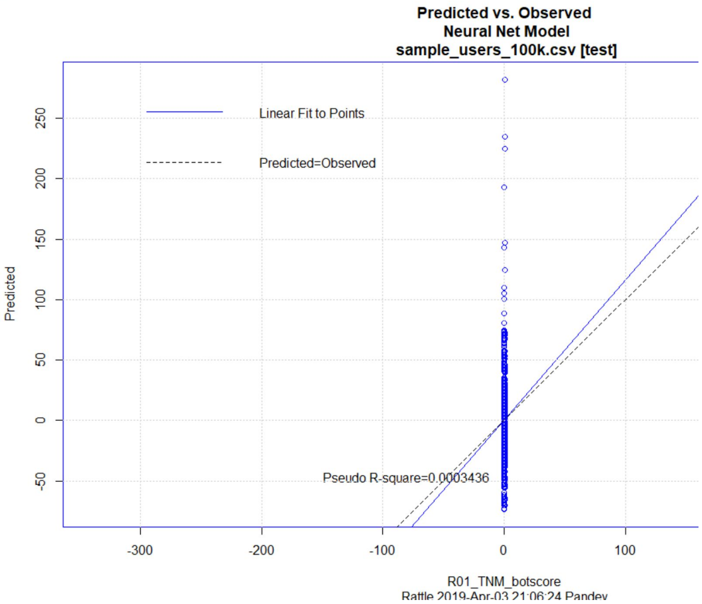


Fig 4 and Fig 5 depicts the performance analysis of the models

Fig 5: pseudo R squared value for Neural Network(from Rattle GUI)

4. CLASSIFICATION TASK

4.1 CLASSIFICATION MODELS

To predict *is_bot* for the above dataset the method for Random Forest used for regression task was used. The ***ranger*** method used to implement the Random Forest is available for both Regression and Classification models.
However, the result obtained from the model was not optimal. The potential reason for the same was discovered to be due to imbalance data,*that is,* the data belonging only to a few features.

4.1.1 Extreme Gradient Boosting

Extreme Gradient Boosting or ***XGBoost*** is one of the fasted implementations of gradient boosted trees.
It does this by tackling one of the major inefficiencies of gradient boosted trees: considering the potential loss for all possible splits to create a new branch.XGBoost tackles this inefficiency by looking at the distribution of features across all data points in a leaf and using this information to reduce the search space of possible feature splits.

4.1.2 Extreme Gradient Boosting vs Random Forest

XGBoost build trees one at a time, where each new tree helps to correct errors made by previously trained tree. With addition of every new tree, the model becomes even more expensive. Thus XGBT training takes longer time because of the fact are trees built sequentially. However benchmark results have shown *XGBoost are better learners than Random Forests.*
The results obtained showed some improvement but still were not optimal.Thus,the dataset was explored again to check for any imbalance as discussed above.

4.1.3 Down-Sampling

In the above dataset to predict whether a user is bot or not it is observed that the percentage of bots user in comparison to the percentage of human users is minimal. Thus a trivial classifier would be the one in which the bots are ignored, thus giving us a good accuracy but it wouldn't predict the presence of bots and would have a 100% rate of false negatives. Thus in order to solve the above problem Downsampling is used.
The results obtained after downsampling and using XGBoost gives a better performance in comparison to Random Forests and XGBoost(*without downsampling)*.

4.2 OBSERVATIONS

MODEL NAME	BALANCED ACCURACY	PRECISION	RECALL	F-MEASURE
Decision Tree('rpart')	0.5640	0.56	0.13	0.21
Random Forest('ranger')	0.5962	0.74	0.20	0.31
XGBoost(with downsampling)	0.81327	0.17	0.78	0.27

Tabel 1 : Comparisons between different Classification Models

From Table 1, it is observed that the XGBoost model outperforms the other two. In the above table the higher recall for XGBoost model indicates the result of total relevant results correctly classified by the model.

5. CONCLUSION AND FUTURE WORK

In this study, the data is used to predict the botscore using the Random Forest model and to classify is_bot feature using the XGBoost model. The same preprocessed features are used for training and testing both the models. The study indicates the performance of different models we trained for the same.

In order to observe a difference in the performance for the selected models some important categorical features viz. ***link,user_name,displayName and summary*** were used to train and test the model again, however, there was no improvement in the performance.

The above study also shows that in data analysis there is no model that is best suited for a type of problem. The "No Free Lunch" theorem states the same,"*there is no one model that works best for every problem*".

The assumption of a good model for one problem may not hold true for another problem, so it is common to use multiple models for data analysis.

6. **REFERENCES**

1. Marian-Andrei Rizoiu and Timothy Graham and Rui Zhang and Yifei Zhang and Robert Ackland and Lexing Xie, #DEBATENIGHT: The Role and Influence of Socialbots on Twitter During the 1st 2016 U.S. Presidential Debate, 2018
2. Paulo Cortez, Data Mining with Neural Networks and Support Vector Machines using the R/rminer Tool,2010
3. Andy Liaw and Matthew Wiener, Classification and Regression by randomForest, 2002
4. https://github.com/topepo/caret
5. Blog.echen.me. (2019). *Choosing a Machine Learning Classifier*. [online] Available at: http://blog.echen.me/2011/04/27/choosing-a-machine-learning-classifier/ [Accessed 3 Apr. 2019].