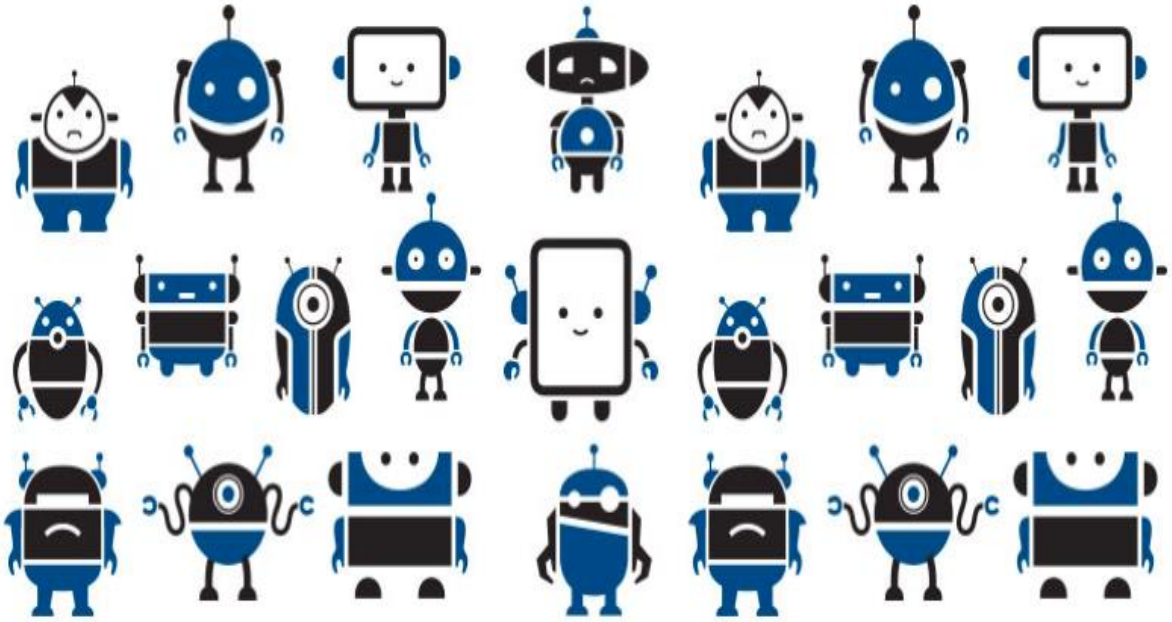


Bots in the Net: Applying Machine Learning to Identify Social Media Trolls

PREPARED BY

VIBHU DAVE

PARAMJEET SINGH

ANKIT JAISWAL

VASAM SHIVANI

SANDHYA MADDULA

INDEX

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Trolls and bots have a huge and often unrecognized influence on social media. They are used to influence conversations for commercial or political reasons. They can push their content to the top of people's news feeds, search results, and shopping carts. Some say they can even influence presidential elections. In order to maintain the quality of discussion on social sites, it has become necessary to screen and moderate community content.

Trolls are dangerous online because it's not always obvious when you are being influenced by them or engaging with them.

Bots are computer programs posing as people. They can amplify the effect of trolls by engaging or liking their content en masse, or by posting their own content in an automated fashion. They will get more sophisticated and harder to detect in the future. Bots can now create entire paragraphs of text in response to text posts or comments.

So in this project we are working with the reddit dataset to identify bots and trolls. In this project we will use the machine learning models to identify suspicious posts and comments.

In the first part of the project data cleaning is performed because the organized data may be incomplete, contain duplicates or errors. So data cleaning is the process to prevent and correct these errors. And then Data Analysis is performed in which various data analysis techniques are used to understand, interpret and derive the conclusions. Machine learning models like Decision Tree Classifier and Logistic Regressions are used to identify the relation among data variables.

In the second part of the project, deployment of the model is done to verify the results.

2. PROBLEM STATEMENT

Social media is becoming an increasingly attractive target to spread disinformation. For our project, we will implement and compare different machine learning techniques to classify whether tweets were posted by trolls or as bots. On a "Reddit" dataset, we apply our classifier to build a live classifier that can aid social media companies in moderating the content.

3. REDDIT DATASET

Trolls and bots are widespread across social media, and they influence us in ways we are not always aware of. Trolls can be relatively harmless, just trying to entertain themselves at others' expense, but they can also be political actors sowing mistrust or discord. While some bots offer helpful information, others can be used to manipulate vote counts and promote content that supports their agenda. We will work on how machine learning can help protect our communities from abuse.

We will focus on Reddit because users often complain of trolls in political threads. It's easier for trolls to operate thanks to anonymous posting. Operatives can create dozens or hundreds of accounts to simulate user engagement, likes and comments.

We will compile a dataset of 267036 rows and 22 columns to identify the bots and the trolls in the data downloaded from the Reddit website and the data is already preprocessed.

banned_by	no_follow	link_id	gilded	author	author_verified	author_comment_karma	author_link_karma	num_comments	created_utc	score	over_18	body	downs	is_submitter	num_reports	controversiality	quarantine
NaN	True	t3_2l5szg	0	ADHDbot	False	-6	1	1.0	1415026958	1	False	As per the rules in the side bar, yes or no qu...	0	False	NaN	0	False
NaN	True	t3_2l61gs	0	ADHDbot	False	-6	1	1.0	1415031687	1	False	Meme and image posts are not allowed on this s...	0	False	NaN	0	False
NaN	True	t3_2l7ma8	0	ADHDbot	False	-6	1	1.0	1415060465	1	False	As per the rules in the side bar, yes or no qu...	0	False	NaN	0	False

Fig i. Reddit Dataset

4. DECISION TREE CLASSIFIER

Decision Tree Classifier is a simple and widely used classification technique. It applies a straight forward idea to solve the classification problem. Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record. Each time it receives an answer, a follow-up question is asked until a conclusion about the class label of the record is reached.

Once the decision tree has been constructed, classifying a test record is straightforward. Starting from the root node, we apply the test condition to the record and follow the appropriate branch based on the outcome of the test. It then leads us either to another internal node, for which a new test condition is applied, or to a leaf node. When we reach the leaf node, the class label associated with the leaf node is then assigned to the record.

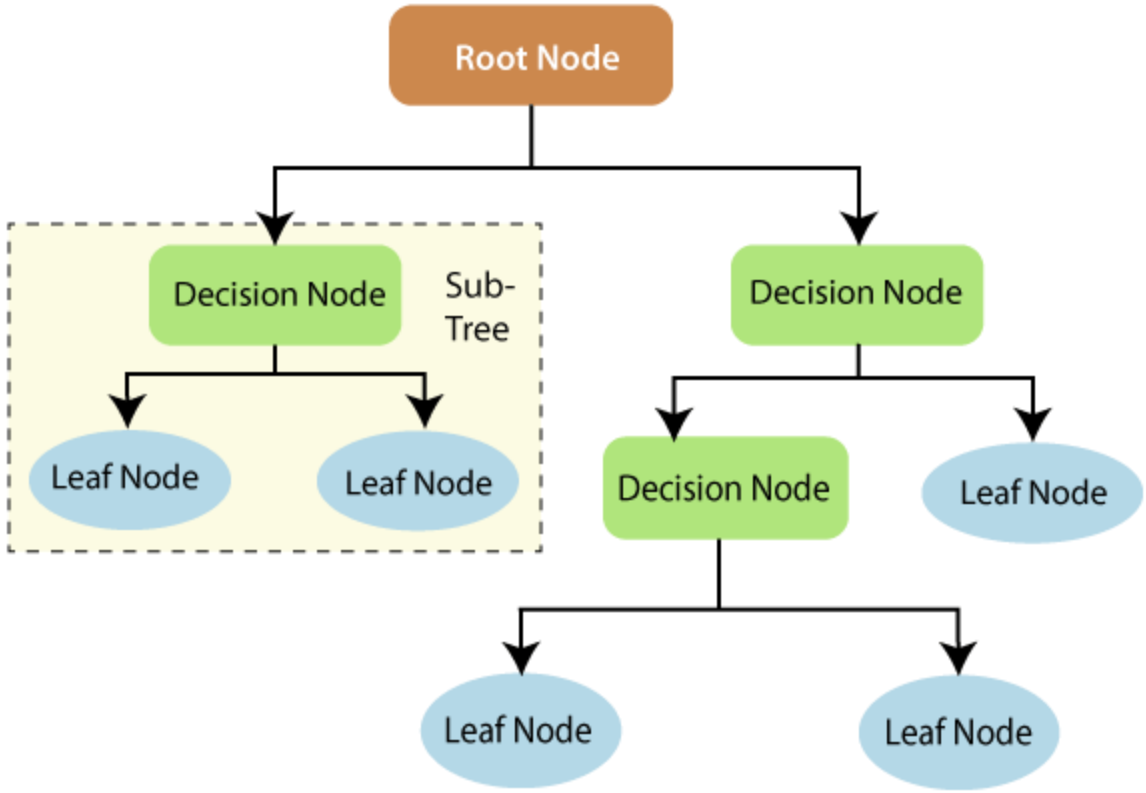


Fig ii. Decision Tree Classifier

5. DESIGN AND ANALYSIS OF REDDIT DATA

Data was preprocessed to filter the tweets. Therefore, “Bots” and “Trolls” columns were regarded as the target variables. The correlation between the columns were found to observe the highly correlated ones with the target variables. The other columns which have no much influence with the target variables or possessing null values were dropped.

This process can be done in python by importing the basic libraries that include numpy, pandas, matplotlib and seaborn. The dataset is then read into the file and correlation can be found through pandas library.

a. EXPLORATORY DATA ANALYSIS (EDA)

The information of the dataset can be read in the python file as what type a particular variable is. This helps in finding out the numeric and categorical variables in the dataset. Also, the null columns can be found.

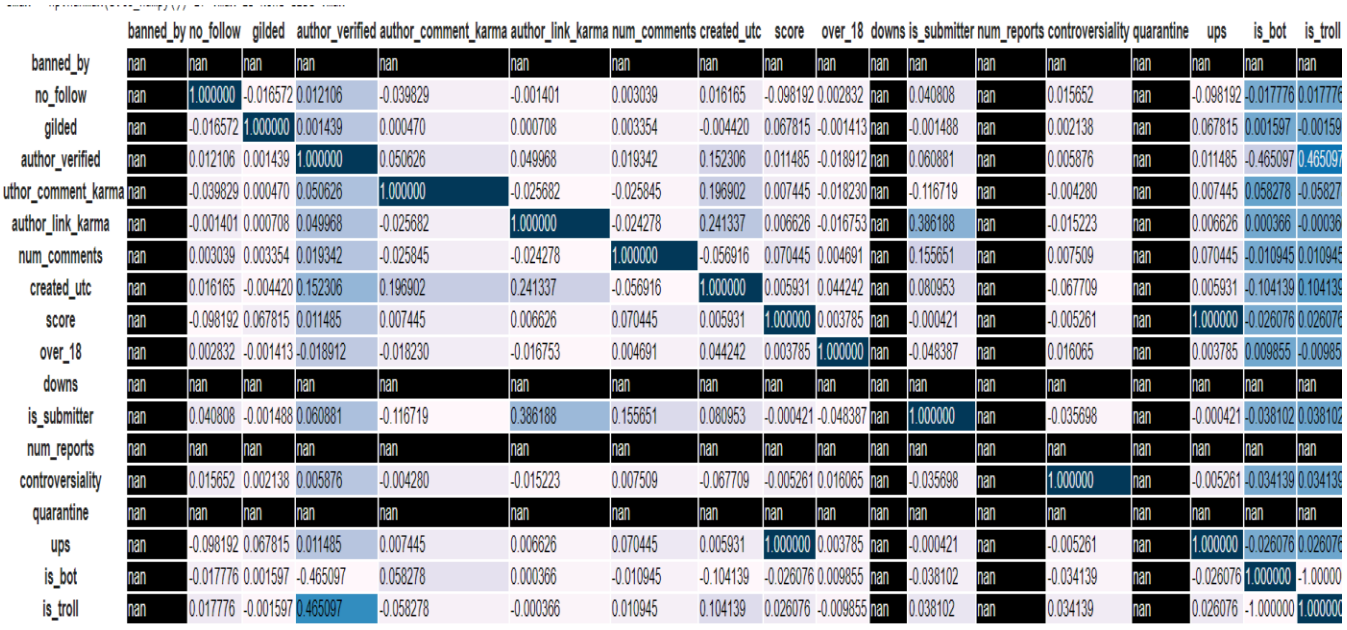
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 267036 entries, 0 to 267035
Data columns (total 22 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   banned_by             0 non-null       float64
 1   no_follow             267036 non-null  bool   
 2   link_id               267036 non-null  object 
 3   gilded                267036 non-null  int64  
 4   author                267036 non-null  object 
 5   author_verified       267036 non-null  bool   
 6   author_comment_karma  267036 non-null  int64  
 7   author_link_karma     267036 non-null  int64  
 8   num_comments          266950 non-null  float64
 9   created_utc           267036 non-null  int64  
 10  score                 267036 non-null  int64  
 11  over_18               267036 non-null  bool   
 12  body                  267035 non-null  object 
 13  downs                 267036 non-null  int64  
 14  is_submitter          267036 non-null  bool   
 15  num_reports           0 non-null       float64
 16  controversiality      267036 non-null  int64  
 17  quarantine            267036 non-null  bool   
 18  ups                   267036 non-null  int64  
 19  is_bot                267036 non-null  bool   
 20  is_troll              267036 non-null  bool   
 21  recent_comments       267036 non-null  object 
```

Fig iii. Information of the datatypes of the variables

b. CORRELATION

The statistical relationship between two variables is referred to as their correlation. There may be complex and unknown relationships between the variables in your dataset. It is important to discover and quantify the degree to which variables in the dataset are dependent upon each other. This knowledge can help better prepare the data to meet the expectations of machine learning algorithms, such as linear regression, whose performance will degrade with the presence of these interdependencies.

Correlation coefficients quantify the association between variables or features of the dataset. These statistics are of high importance and python has great tools to calculate them. Among them was the Pandas library and that is fast, comprehensive and well-documented.



	banned_by	no_follow	gilded	author_verified	author_comment_karma	author_link_karma	num_comments	created_utc	score	over_18	downs	is_submitter	num_reports	controversiality	quarantine	ups	is_bot	is_troll
banned_by	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
no_follow	nan	1.000000	-0.016572	0.012106	-0.039829	-0.001401	0.003039	0.016165	-0.098192	0.002832	nan	0.040808	nan	0.015652	nan	-0.098192	-0.017776	0.017776
gilded	nan	-0.016572	1.000000	0.001439	0.000470	0.000708	0.003354	-0.004420	0.067815	-0.001413	nan	-0.001488	nan	0.002138	nan	0.067815	0.001597	-0.00159
author_verified	nan	0.012106	0.001439	1.000000	0.050626	0.049968	0.019342	0.152306	0.011485	-0.018912	nan	0.060881	nan	0.005876	nan	0.011485	-0.465097	0.465097
uthor_comment_karma	nan	-0.039829	0.000470	0.050626	1.000000	-0.025682	-0.025845	0.196902	0.007445	-0.018230	nan	-0.116719	nan	-0.004280	nan	0.007445	0.058278	-0.05827
author_link_karma	nan	-0.001401	0.000708	0.049968	-0.025682	1.000000	-0.024278	0.241337	0.006626	-0.016753	nan	0.386188	nan	-0.015223	nan	0.006626	0.000366	-0.00036
num_comments	nan	0.003039	0.003354	0.019342	-0.025845	-0.024278	1.000000	-0.056916	0.070445	0.004691	nan	0.155651	nan	0.007509	nan	0.070445	-0.010945	0.010945
created_utc	nan	0.016165	-0.004420	0.152306	0.196902	0.241337	-0.056916	1.000000	0.005931	0.044242	nan	0.080953	nan	-0.067709	nan	0.005931	-0.104139	0.104139
score	nan	-0.098192	0.067815	0.011485	0.007445	0.006626	0.070445	0.005931	1.000000	0.003785	nan	-0.000421	nan	-0.005261	nan	1.000000	-0.026076	0.026076
over_18	nan	0.002832	-0.001413	-0.018912	-0.018230	-0.016753	0.004691	0.044242	0.003785	1.000000	nan	-0.048387	nan	0.016065	nan	0.003785	0.009855	-0.00985
downs	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
is_submitter	nan	0.040808	-0.001488	0.060881	-0.116719	0.386188	0.155651	0.080953	-0.000421	-0.048387	nan	1.000000	nan	-0.035698	nan	-0.000421	-0.038102	0.038102
num_reports	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
controversiality	nan	0.015652	0.002138	0.005876	-0.004280	-0.015223	0.007509	-0.067709	-0.005261	0.016065	nan	-0.035698	nan	1.000000	nan	-0.005261	-0.034139	0.034139
quarantine	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
ups	nan	-0.098192	0.067815	0.011485	0.007445	0.006626	0.070445	0.005931	1.000000	0.003785	nan	-0.000421	nan	-0.005261	nan	1.000000	-0.026076	0.026076
is_bot	nan	-0.017776	0.001597	-0.465097	0.058278	0.000366	-0.010945	-0.104139	-0.026076	0.009855	nan	-0.038102	nan	-0.034139	nan	-0.026076	1.000000	-1.00000
is_troll	nan	0.017776	-0.001597	0.465097	-0.058278	-0.000366	0.010945	0.104139	0.026076	-0.009855	nan	0.038102	nan	0.034139	nan	0.026076	-1.000000	1.000000

Fig iv. Correlation between all the variables in the dataset

Correlation could be positive, meaning both variables move in the same direction, or negative, meaning that when one variable's value increases, the other variables' values decrease. Correlation can also be neutral or zero, meaning that the variables are unrelated. If the correlated featured variables are not picked out, they may end up with predicting the biased results.

From the dataset, we observe that the "Score" and "Ups" columns are highly positively correlated with "+1" value. "Is_bot" and "Is_troll" columns are highly negatively correlated with "-1" value. Therefore, one of them can be dropped. "Is_bot" and "Is_troll" columns are interdependent on each other and so, one of them can be considered as the target variable. (As here, the prediction is to find out whether it is a troll or a bot).
"Banned_by" and "num_reports" columns have 0 non-null values. So we drop those columns thereby removing all the missing values from the dataset.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 266949 entries, 0 to 267035
Data columns (total 17 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   no_follow             266949 non-null  bool
 1   link_id               266949 non-null  object
 2   gilded                266949 non-null  int64
 3   author                266949 non-null  object
 4   author_verified       266949 non-null  bool
 5   author_comment_karma  266949 non-null  int64
 6   author_link_karma     266949 non-null  int64
 7   num_comments          266949 non-null  int64
 8   created_utc           266949 non-null  int64
 9   over_18               266949 non-null  bool
 10  body                  266949 non-null  object
 11  downs                 266949 non-null  int64
 12  is_submitter          266949 non-null  bool
 13  controversiality      266949 non-null  int64
 14  quarantine            266949 non-null  bool
 15  is_bot                266949 non-null  bool
 16  recent_comments       266949 non-null  object
dtypes: bool(6), int64(7), object(4)
memory usage: 26.0+ MB
```

Fig v. Variables information after removing correlated variables

The final variable information can be as shown in the above image.

	no_follow	link_id	gilded	author	author_verified	author_comment_karma	author_link_karma	num_comments	over_18	downs	is_submitter	controversiality	quarantine
0	True	t3_2l5szg	0	ADHDbot	False	-6	1	1	False	0	False	0	False
1	True	t3_2l61gs	0	ADHDbot	False	-6	1	1	False	0	False	0	False
2	True	t3_2l7ma8	0	ADHDbot	False	-6	1	1	False	0	False	0	False
3	True	t3_2l7t5h	0	ADHDbot	False	-6	1	1	False	0	False	0	False
4	True	t3_2l900k	0	ADHDbot	False	-6	1	1	False	0	False	0	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
267031	True	t3_96p3s7	0	haiku_robot	True	159092	1	10	False	0	False	0	False
267032	True	t3_1w5fy8	0	tipmoonbot2	False	2	1	10	False	0	False	0	False
267033	True	t3_1w5fy8	0	tipmoonbot2	False	2	1	10	False	0	False	0	False
267034	True	t3_1w6etv	0	tipmoonbot2	False	2	1	26	False	0	False	0	False
267035	True	t3_1w6etv	0	tipmoonbot2	False	2	1	26	False	0	False	0	False

266949 rows × 13 columns

Fig vi. Columns remaining after removing correlated columns

c. ONE HOT ENCODING

In this dataset, we encounter columns that contain numbers of no specific order of preference. The data in the column usually denotes a category or value of the category and also when the data in the column is label encoded. This confuses the machine learning model. To avoid this, the data in the column should be One Hot encoded.
It refers to splitting the column which contains numerical categorical data to many columns depending on the number of categories present in that column. Each column contains “0” or “1” corresponding to which column it has been placed.

In this dataset, “is_bot” and “is_troll” variables are correlated with each other. If “is_bot” is considered “0”, the other will be “1” i.e they are one hot encoded.

d. SCALING AND NORMALISATION

Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling. Standardization of datasets is a common requirement for many machine learning estimators implemented in scikit-learn; they might behave badly if the individual features do not more or less look like standard normally distributed data.

From the dataset, variables like “num_comments” or other variables has higher values. So in our model, this variable may have greater influence to other. For standardization of data, we use scikit learn library to import “Standard Scaler”.

6. TRAIN-TEST SPLIT

The train-test split procedure is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modelling problem. It can be used for classification or regression problems and can be used for any supervised learning algorithm.
The procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model and is referred to as the training dataset. The second subset is not used to train the model. Instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values. This second dataset is referred to as the test dataset.

Train Dataset: Used to fit the machine learning model.
Test Dataset: Used to evaluate the fit machine learning model.

After the processing part of the Reddit dataset, we continue with the "train test split" procedure of data. Therefore, we split our processed Reddit data into 80:20 of train and test data. This can be done by applying "train_test_split" function from the scikit learn library in python.

from sklearn.model_selection import train_test_split

7. DECISION TREE APPLICATION

After the train-test split function is performed, Decision tree algorithm is applied.

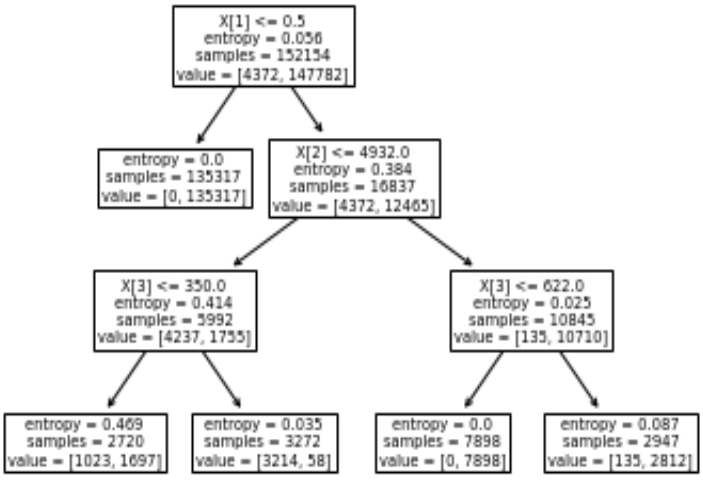


Fig vii. Decision Tree Classifier

a. CONFUSION MATRIX

Confusion matrix is the table that is often used to describe the performance of a classification model on a set of test data for which the true values are known.

Confusion Matrix is a useful machine learning method which allows us to measure **Recall, Precision, Accuracy, and AUC-ROC curve**. Below

given is an example to know the terms True Positive, True Negative, False Negative, and True Negative. For Example, if it is True Positive: You projected positive and its turn out to be true.

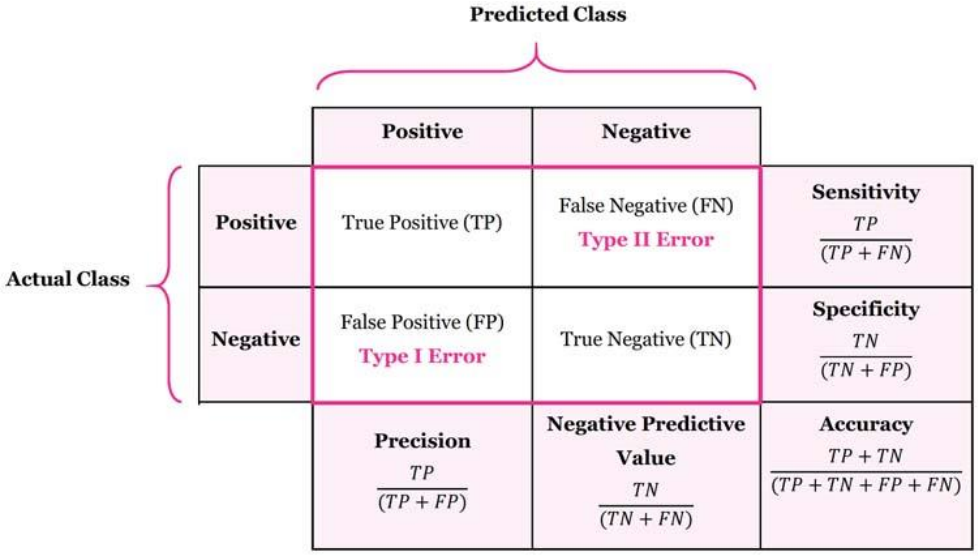


Fig viii. Confusion matrix

Confusion matrix for the applied decision tree algorithm can be shown in the below image.

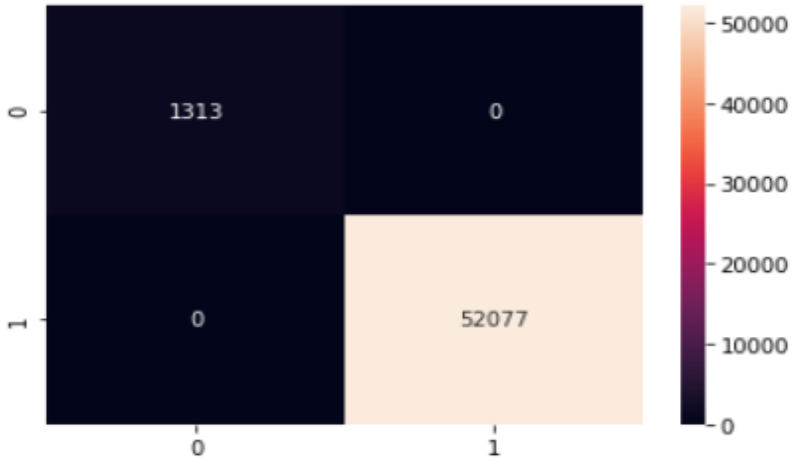


Fig ix. Confusion matrix for Decision Tree

From the confusion matrix, there are 1313 true negative and 52077 true positive and zero false negative and zero false positive. So there may be a chance of overfitting of data.

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1313
1	1.00	1.00	1.00	52077
accuracy			1.00	53390
macro avg	1.00	1.00	1.00	53390
weighted avg	1.00	1.00	1.00	53390

Fig x. Classification Report for Decision Tree

8. LOGISTIC REGRESSION APPLICATION

Logistic Regression is used for the classification problems. This is a predictive analysis algorithm and is based on the concept of probability. The hypothesis of logistic regression tends to limit the cost function between 0 and 1.

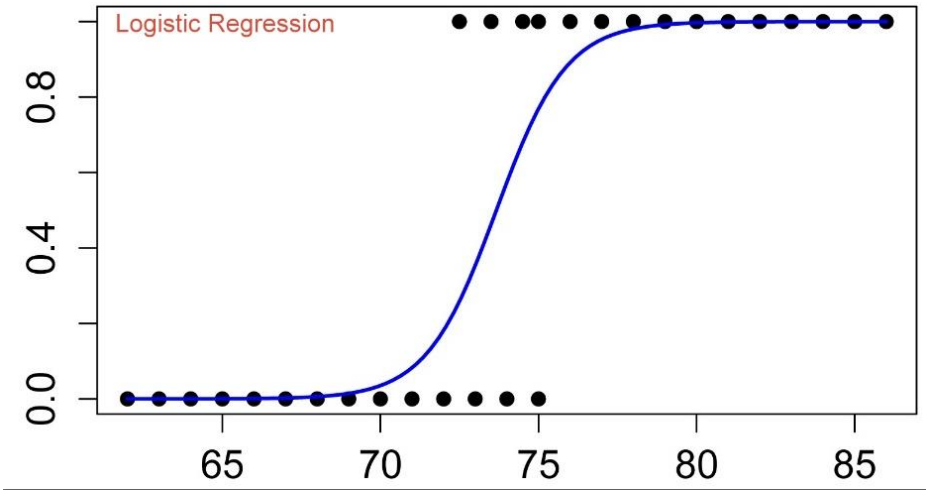


Fig xi. Example of the Logistic Regression

Logistic regression algorithm is performed on the splitted train test data. The confusion matrix is obtained after applying this algorithm and is compared with that of the Decision tree algorithm.

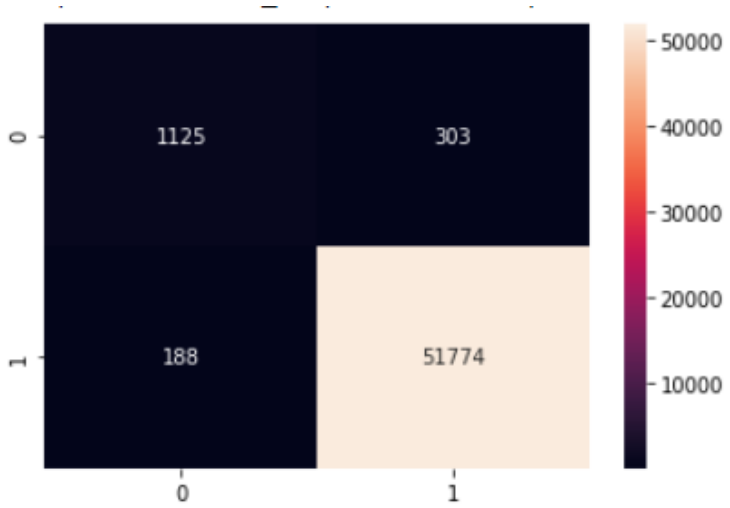


Fig xii. Confusion Matrix for Logistic Regression

From the confusion matrix there are 1125 true negative and 51774 true positive and 303 false negative and 188 false positive. So we got 0.99 accuracy.

	precision	recall	f1-score	support
0	0.86	0.79	0.82	1428
1	0.99	1.00	1.00	51962
accuracy			0.99	53390
macro avg	0.93	0.89	0.91	53390
weighted avg	0.99	0.99	0.99	53390

Fig xiii. Classification Report for Logistic Regression

9. CROSS VALIDATION

Cross-validation is a technique for evaluating machine learning models by training several ML models on subsets of the available input data and

evaluating them on the complementary subset of the data. Cross-validation is used to detect overfitting, ie, failing to generalize a pattern.

The Cross Validation technique is applied on the Reddit dataset. This is done by importing scikit-learn library into python. Stratified K-Fold cross validation technique is applied on both decision tree and logistic regression models. ROC-AUC score of Decision tree is found to be 0.99496 and that of Logistic regression is 0.74384.

```
]  1 from sklearn.model_selection import cross_val_score,StratifiedKFold,cross_val_predict
   2
   3 # Set up our K-fold cross-validation
   4 kf = StratifiedKFold(n_splits=10,)
   5
   6 decision_tree = DecisionTreeClassifier()
   7 logistic_reg = LogisticRegression(max_iter=1000)
   8
   9 # Train our models using KFold cv
  10 dtree_score = cross_val_score(decision_tree,feature, label, cv = kf,scoring='roc_auc')
  11 lr_score = cross_val_score(logistic_reg,feature, label, cv = kf,scoring='roc_auc')
  12
  13 # Print the mean of each array of scores
  14 print("Decision Tree:", np.mean(dtree_score),"Logistic Regression:", np.mean(lr_score))

Decision Tree: 0.9949610982059358 Logistic Regression: 0.7438434383261796
```

Fig xiv. Cross Validation Technique

10. BALANCING OF DATASET

An imbalanced classification problem is an example of a classification problem where the distribution of examples across the known classes is biased or skewed. The distribution can vary from a slight bias to a severe imbalance where there is one example in the minority class for hundreds, thousands, or millions of examples in the majority class or classes.

Imbalanced classifications pose a challenge for predictive modelling as most of the machine learning algorithms used for classification were

designed around the assumption of an equal number of examples for each class. This results in models that have poor predictive performance, specifically for the minority class. This is a problem because typically, the minority class is more important and therefore the problem is more sensitive to classification errors for the minority class than the majority class.

From our dataset, the values of the target variable, “is_bot” is not balanced. True value of “is_bot” is 260382 and False value is 6567. This shows that there is skewness in the data and a need to balance the dataset.

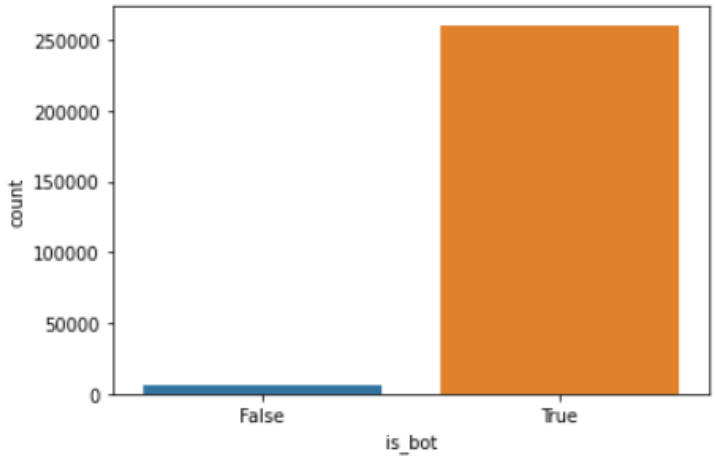


Fig xv. Imbalanced data of the Target variable “is_bot”

To balance the dataset we consider random values of “true” from “is_bot” variable and is exactly equal to “false” value.

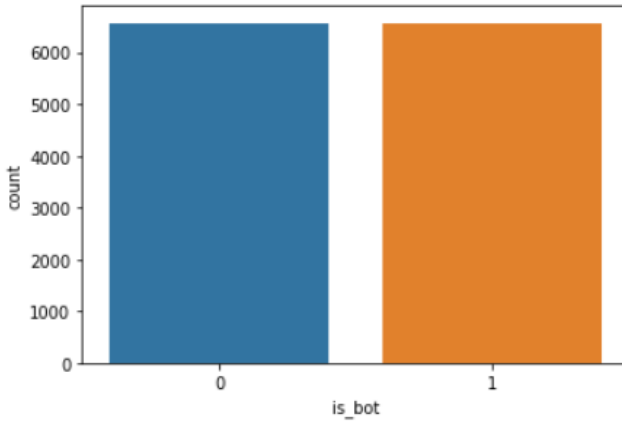


Fig xvi. Balanced data of the Target variable “is_bot”

For the balanced data, we perform the same operations of applying Decision Tree and Logistic regression models.

a. DECISION TREE MODEL

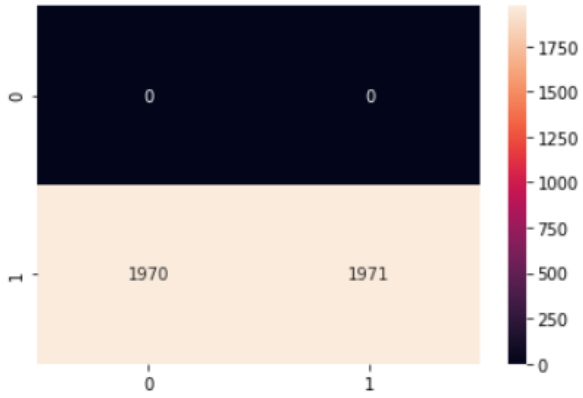


Fig xvii. Confusion matrix for Balanced Decision Tree model

The confusion matrix for the model predicts “True” for all the inputs ie., as “is_bot”. This shows that the performance of the model is not good.

	precision	recall	f1-score	support
0	0.00	0.00	0.00	0
1	1.00	0.50	0.67	3941
accuracy			0.50	3941
macro avg	0.50	0.25	0.33	3941
weighted avg	1.00	0.50	0.67	3941

Fig xviii. Classification Report for Balanced Decision Tree model

b. LOGISTIC REGRESSION MODEL

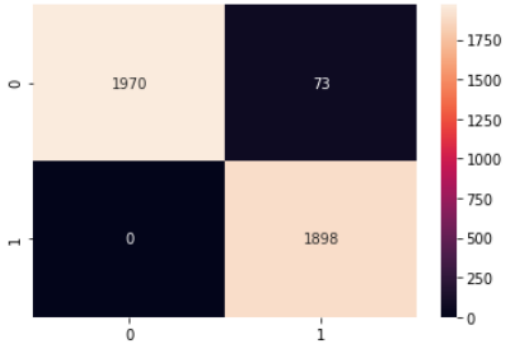


Fig xix. Confusion matrix for Balanced Logistic Regression model

From the above obtained confusion matrix, there are 1970 true negatives and 1898 true positives and 73 false negatives and 0 false positives. This gives an accuracy of 0.98. The other details can be observed from the below classification report.

	precision	recall	f1-score	support
0	1.00	0.96	0.98	2043
1	0.96	1.00	0.98	1898
accuracy			0.98	3941
macro avg	0.98	0.98	0.98	3941
weighted avg	0.98	0.98	0.98	3941

Fig xx. Classification Report for Balanced Logistic Regression model

c. CROSS VALIDATION

Cross Validation technique is applied for the balanced dataset by importing the scikit-learn library. The stratified K-Fold Cross Validation technique is applied on both the decision tree and logistic regression models. ROC-AUC score of Decision tree is 0.971080 and that of Logistic regression 0.90158

```
1 dtree_score1 = cross_val_score(decision_tree,feature1, label1, cv = kf,scoring='roc_auc')
2 lr_score1 = cross_val_score(logistic_reg,feature1, label1, cv = kf,scoring='roc_auc')
3
4 # Print the mean of each array of scores
5 print("Decision Tree:", np.mean(dtree_score1),"Logistic Regression:", np.mean(lr_score1))
```

```
Decision Tree: 0.9710806697108068 Logistic Regression: 0.9015879041202683
```

Fig xxi. Application of cross validation technique

11. DEPLOYMENT

Once the solution for the machine learning model is obtained, we can showcase it and make it accessible for others. Hence, the final stage of the machine learning lifecycle is to deploy that model.

As per the founders of Streamlit, it is the fastest way to build data apps and share them. It is a recent model deployment tool that simplifies the entire model deployment cycle and let's deploy our models quickly. It is a simple, quick, and interpretable model deployment tool.

For machine learning models developed, the source code is not sent directly to the client or end customer. An interface is created in which the client can give the inputs and obtain the output. The **Troll-bot predictor** model is deployed with the help of **Streamlit** and **Github**. Final app was created with the help of streamlit and deployed using the same. We will be having the following files for the model deployment in GitHub:

- **App.py** – In this file, we have the implementation of our model. The actual web implementation is done and codes are written in this file. We have streamlit feature of marking down the HTML codes with the help of steramlit.markdown().

- **Ml.py** – In this file, the required coding part of the model is done. We have used Linear Regression for the final deployment. It contains two variables X and Y which we had used for training of the model. By importing the pickle library into python, a pickle file is generated for the machine learning model.

- **Model.pkl** – By importing the pickle library into python, a pickle file is generated for the machine learning model.

- **Requirements.txt** – This contains all the essential libraries that are required for the deployment. In this project, Streamlit (0.73.1),

pandas (1.1.3), NumPy (1.19.2), scikit-learn (0.23.2) are the basic libraries that are used.

- **Setup.sh** – This helps us to set up the initial level of app deployment. This file at will tell to run requirements.txt initially. This will check whether the required libraries are downloaded or not and if the files are not their this will install the libraries.

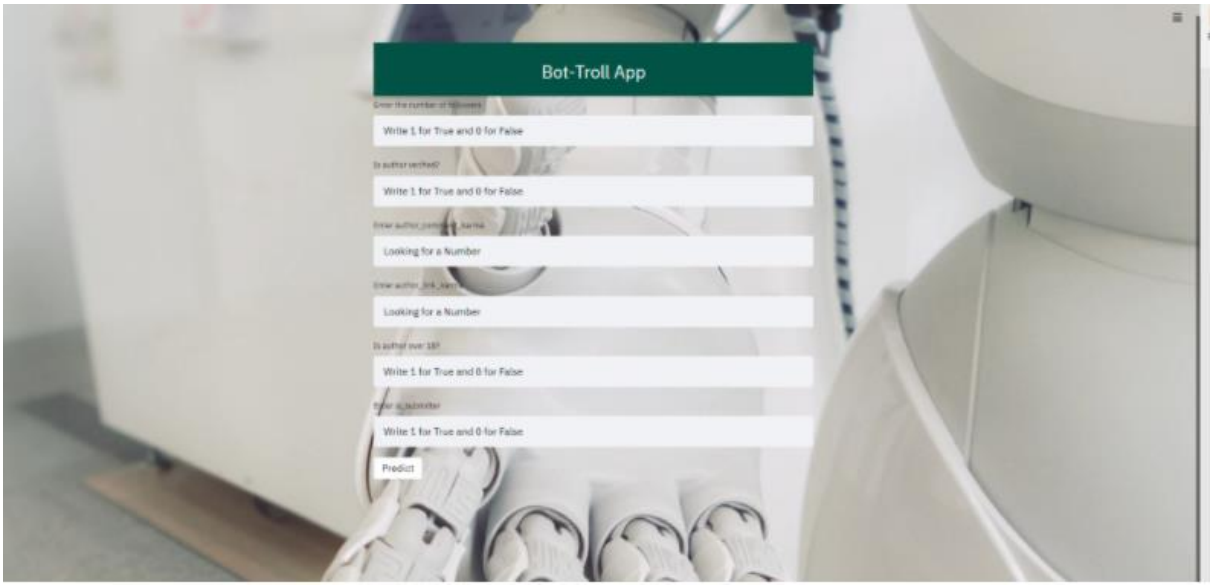


Fig xxii. Deployed Troll – Bot Prediction Model

12. TESTING

In the very starting we have tried our classifier on different types of machine learning models. We have collected accuracies of different models. The models used in this project are Logistic Regression, Decision tree and random forest.

For Random forest, Decision tree and Logistic regression we are getting an accuracy of 99%, 100 and 92% respectively.

Decision Tree:

	precision	recall	f1-score	support
False	0.98	0.74	0.85	2168
True	0.99	1.00	1.00	72774
accuracy			0.99	74942
macro avg	0.99	0.87	0.92	74942
weighted avg	0.99	0.99	0.99	74942

Logistic Regression:

	precision	recall	f1-score	support
False	0.00	0.00	0.00	2168
True	0.97	1.00	0.99	72774
accuracy			0.97	74942
macro avg	0.49	0.50	0.49	74942
weighted avg	0.94	0.97	0.96	74942

Decision Tree: 1.0 Logistic Regression: 0.921410087361137

Fig. Performance for Decision Tree and Logistic Regression

Classification report:

	precision	recall	f1-score	support
False	1.00	1.00	1.00	2168
True	1.00	1.00	1.00	72774
accuracy			1.00	74942
macro avg	1.00	1.00	1.00	74942
weighted avg	1.00	1.00	1.00	74942

Accuracy Score:
0.9999733126951509

Fig. Performance for Random Forest

Fig xxiii. Performance metrics for the three models

Out of the results obtained, we can observe that the random forest and decision tree models are facing the overfitting problem over the data. Hence, Logistic regression model is opted for our final deployment.

We have performed testing for the different inputs. In the deployed app we have 5 variables as the input. These variables are “does the user have followers or not”, “is author verified”, “Is author over 18”, “is author a submitter or not” etc.

S.No	Test Case ID	Test Input	Expected Result	Actual Result
1.	T101	0, 1, 1354, 4, 1, 1	0	0
2.	T102	0, 1, 10807, 1, 1, 1	0	0
3.	T103	0, 0, 258, 14611, 1, 1	1	0
4.	T104	0, 0, 3275, 45367, 1, 1	1	0
5.	T105	0, 1, 20013, 1, 1, 1	0	0

Table i. Test results for different inputs

The app is finally built by using Streamlit. Different test cases were given to get the final output and they are recorded as in the above table. The app is predicting whether it is a troll or a bot effectively with an accuracy of 84% and this is done by using the applied logistic regression model.

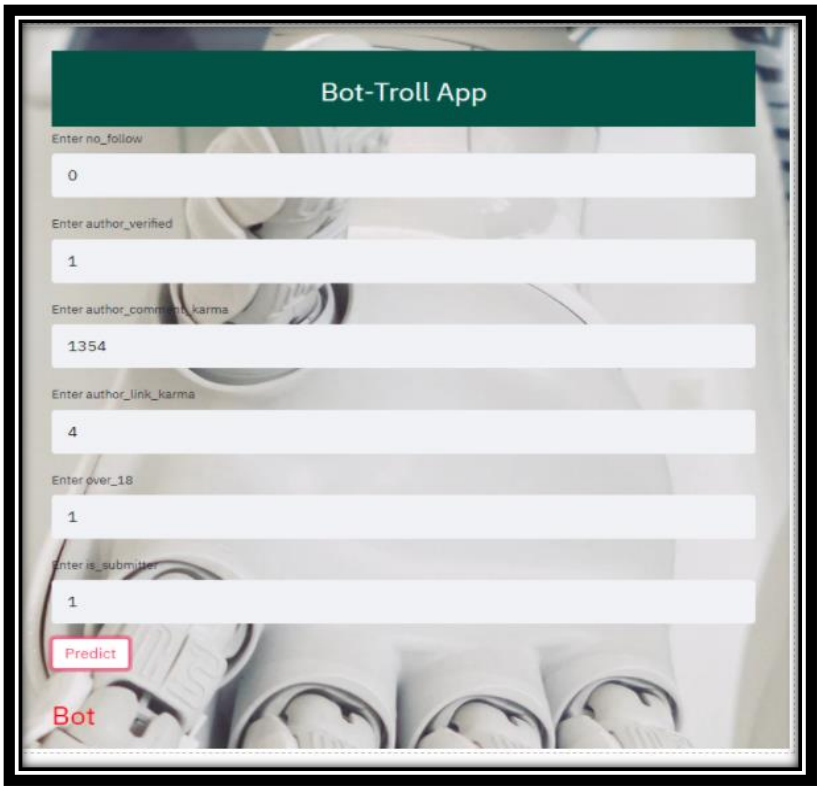


Fig xxiv. Troll – Bot Prediction app using Streamlit