# Twitter Botnet Detection using Machine Learning
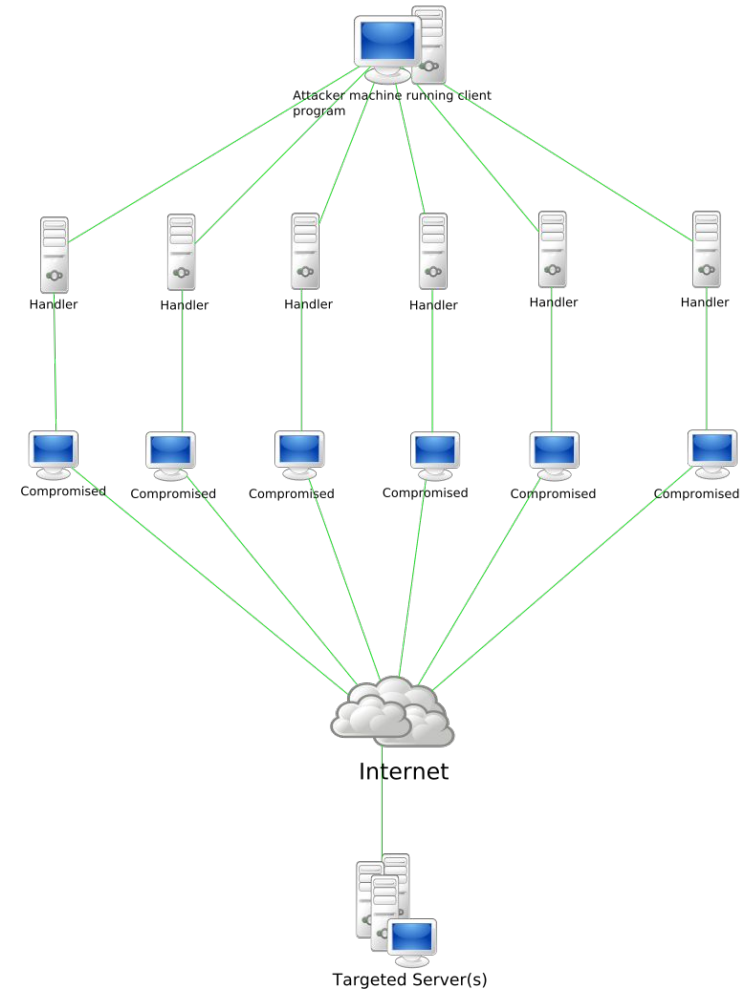
Name:              Shreyas Nikam
Roll no:           31241
Seminar Guide:     Prof. R. A. Kulkarni

# Introduction:

- A botnet is nothing more than a string of connected computers coordinated together to perform a task.

- The owner can control the botnet using command and control (C&C) software.

- Botnets are increasingly rented out by cyber criminals as commodities for a variety of purposes.

# Common tasks executed by botnets include:

- Using your machine's power to assist in distributed denial-of-service (DDoS) attacks to shut down websites.
- Emailing spam out to millions of Internet users.
- Generating fake Internet traffic on a third-party website for financial gain.
- Replacing banner ads in your web browser specifically targeted at you.
- Pop-ups ads designed to get you to pay for the removal of the botnet through a phony anti-spyware package

# Motivation:

❖ Twitter botnets have been an area of interest for security experts and the average user of the platform for some time now.
❖ As the number of botnet attacks has been increasing, it is very difficult to find devices without any vulnerability.
❖ Bots can influence public opinion. This is certainly an extremely powerful tool, and as with most powerful tools, there is the possibility that it will be used for malicious or less ethical purposes at some point.
❖ Hence there is a need of botnet detection systems that can classify between real users and identify the lurking botnets.

# Literature Survey:

| No. | Title | Description | Findings | Limitations | Dataset |
|---|---|---|---|---|---|
| 1. | "Detection of Botnet Activity via Machine Learning" | Describes a simple taxonomy for data ex-filtration techniques. | Proposed a solution to the problem of detecting botnet activity using a machine learning approach. | Lot of false-positives, more fine-tuning required | Publicly available dataset found at the University of New Brunswick |
| 2. | "It's All in a Name: Detecting and Labeling Bots by Their Name" | Uses random string detection applied to users' names to filter Twitter streams for bot accounts. | This technique is able to easily filter accounts that are likely bot accounts. | Restrictions on data sharing, and not all bots are the same. | Own dataset |
| 3. | "Botnet Campaign Detection on Twitter" | A novel approach to detecting bots on twitter in near real-time. | Used K-means clustering to classify botnets in real time. | Optimization and refinement required. | Combination of 13 datasets |
| 4. | "An empirical comparison of botnet detection methods" | Compares the output of three different botnet detection methods. | Simplified comparison of BClus, CAMNEP and Bothunter methods. | Need for a comparison methodology and a proper error metric. | Own dataset |

# Problem Definition:

❖ Apply various machine learning techniques and algorithms to build a Twitter Botnet Detector.

❖ Most of the existing methods focus on numerical-statistical data such as count of followers, friends, statuses, and other boolean values like whether the user has default profile picture and whether the account is verified.

❖ In this seminar, I try to process the screen name of the users to further enhance the results of the algorithms.

# Data Exploration:

The dataset consists of 20 columns and 2797 rows.

Also, as it can be seen from the diagram below, the dataset is well-balanced.
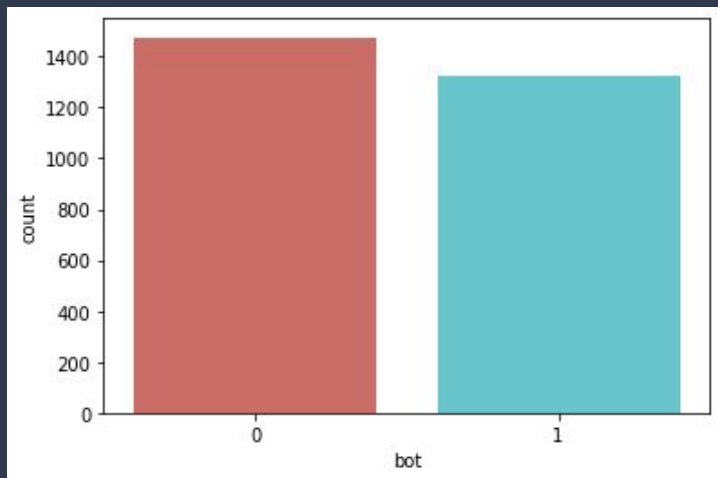


```
In [14]: dataset.columns.values
Out[14]:
array(['id', 'id_str', 'screen_name', 'location', 'description', 'url',
       'followers_count', 'friends_count', 'listedcount', 'created_at',
       'favourites_count', 'verified', 'statuses_count', 'lang', 'status',
       'default_profile', 'default_profile_image', 'has_extended_profile',
       'name', 'bot'], dtype=object)

In [15]: dataset.head()
Out[15]:
            id         id_str  ...                  name bot
0  1.638039e+08  1.638039e+08  ...               Camisha   1
1  7.710000e+17  7.710000e+17  ...                  ___    1
2  8.400000e+17  8.400000e+17  ...            9738166924   1
3  7.710000e+17  7.710000e+17  ...  1 347-615-3461holley   1
4  7.610000e+17  7.610000e+17  ...        (917) 615-7513   1

[5 rows x 20 columns]

In [16]:
```

| Index | id | id_str | screen_name | location | description | url | followers_count | friends_count | listedcount | created_at | favourites_count | verified | statuses_count | lanq | status | default_profile | default_profile_imaqe | has_extended |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1.6... | 1.6... | 1lovetake... | | | | 0 | 9 | 0 | Wed Jul... | 0 | False | 0 | en | {} | True | True | False |
| 1 | 7.7... | 7.7... | 1pRsHsia8... | 33 | | | 4 | 102 | 0 | Thu Sep... | 2 | False | 0 | en | {} | True | False | True |
| 2 | 8.4e +17 | 8.4e +17 | 2UvyAyaFb... | | | | 2 | 97 | 0 | Fri Mar... | 0 | False | 0 | en | {} | True | True | False |
| 3 | 7.7... | 7.7... | 3461holley | | | | 6 | 109 | 0 | Tue Aug... | 0 | False | 2 | en | { ... | True | True | False |
| 4 | 7.6... | 7.6... | 917_7513 | Sports | | | 0 | 54 | 0 | Thu Aug... | 7 | False | 18 | en | { ... | True | True | False |
| 5 | 1.9... | 1.9... | alertmess... | UK | The Neiq... | htt... | 43869 | 4786 | 69 | Mon Sep... | 125 | False | 120082 | en | { ... | False | False | False |
| 6 | 2.6... | 2.6... | AmishFarm... | Lanca... | We invit... | htt... | 119 | 165 | 5 | Thu Jul... | 22 | False | 115 | en | { ... | False | False | False |
| 7 | 2.7... | 2.7... | AnayaJarr... | | | | 3 | 21 | 0 | Sun Sep... | 0 | False | 0 | en | {} | True | True | False |
| 8 | 2.4... | 2.4... | Attack_On... | Attack o... | | | 567 | 0 | 2 | Wed Mar... | 0 | False | 16733 | ja | { ... | False | False | False |
| 9 | 7.0... | 7.0... | AutoFollo... | All o... | Follow a... | htt... | 10832 | 2673 | 42 | Fri Jul... | 2413 | False | 34971 | en | { ... | False | False | False |
| 10 | 1.1... | 1.1... | Axelcarp2... | Nothing ... | | | 70 | 354 | 8 | Fri Jan... | 60 | False | 5074 | es | { ... | False | False | False |
| 11 | 2.9... | 2.9... | b2a681397... | | | | 0 | 15 | 0 | Sat Dec... | 0 | False | 0 | en | {} | True | True | False |
| 12 | 3.1... | 3.1... | Bedr0ckLe... | Londo... | We are o... | htt... | 970 | 1642 | 15 | Tue May... | 145 | False | 976 | en | { ... | False | False | False |
| 13 | 1.5... | 1.5... | blue_trac... | San D... | _____... | | 392 | 638 | 22 | Tue Aug... | 110 | False | 4530 | en | { ... | True | False | False |
| 14 | 2.4... | 2.4... | brunoairah | exo-l... | unfair__ | | 340 | 440 | 2 | Sat Mar... | 3058 | False | 18570 | en | { ... | False | False | True |
| 15 | 1.3... | 1.3... | BTCMaster | Miami... | http://t... | htt... | 1811 | 15 | 262 | Fri Apr... | 20 | False | 29998 | en | { ... | False | False | False |
| 16 | 7.6... | 7.6... | CambsPoke... | Cambr... | Get noti... | | 407 | 2 | 0 | Sun Aug... | 1 | False | 32573 | en | { ... | False | False | False |
| 17 | 3.3... | 3.3... | Circaener... | Coler... | Our aim ... | htt... | 957 | 2826 | 39 | Sat Jun... | 3255 | False | 4413 | en | { ... | False | False | False |
| 18 | 1.8... | 1.8... | ComicBook... | Everythi... | | | 109901 | 85639 | 650 | Thu Sep... | 3550 | False | 455 | en | { ... | True | False | False |
| 19 | 2.6... | 2.6... | DanielStu... | Pro foot... | | | 1961954 | 420 | 4731 | Fri Mar... | 879 | True | 1389 | en | { ... | True | False | False |
| 20 | 7.9... | 7.9... | Defendant... | Every Mo... | htt... | | 1132 | 8 | 7 | Wed Nov... | 32 | False | 656 | en | { ... | True | False | True |

# Observations:



observations_y - DataFrame

| bot | id | id_str | followers_count | friends_count | listedcount | favourites_count | verified | statuses_count | default_profile | default_profile_image | has_extended_profil |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 5.5086e+16 | 5.5… | 1.87617e+06 | 7647.13 | 5941.84 | 3266.63 | 0.436… | 10683.1 | 0.250678 | 0.0243902 | 0.194444 |
| 1 | 2.52337e+17 | 2.5… | 13458.8 | 1611.37 | 120.141 | 660.081 | 0.006… | 29249.2 | 0.656321 | 0.0878123 | 0.0635882 |

Using recursive feature elimination, the columns that were selected as good predictors of outcome are as follows:

[ 'followers_count', 'friends_count', 'listedcount', 'has_extended_profile', 'default_profile_image', 'default_profile', 'statuses_count', 'verified', 'favourites_count' ]

# Innovation:

Using 'screen_names' and 'name' of the account to predict the probability of the account being a bot and adding this probability as a column to predict the overall outcome.

**Algorithm used:** Logistic Regression

Accuracy ~65%

Acceptable, since these are only used to predict probability and further used to evaluate the model.

(SVM implementation accuracy ~63%)

```
Algorithm:
    for each str in list of names:
        calculate alphabets
        calculate digits
        calculate special characters
        calculate has_bot_substring
    return new columns[alpha_count, digit_count,
                       sp_chars, has_bot_substr]

scale the new columns between (0,1)
create instance of LogisticRegressionClassifier
classifier.fit()
classifier.predict_probability()
add probability as new column to dataset
```

# Algorithms used and comparison:

- Logistic Regression
- SVC
- Decision Tree Classifier
- Random Forest Classifier
- Gaussian Naive Bayes Classifier
- Artificial Neural Network

| Algorithm: | Accuracy-precision-recall-F1 |
|---|---|
| Logistic Regression | 0.738 - 0.81 - 0.74 - 0.72 |
| SVC | 0.644 - 0.79 - 0.64 - 0.59 |
| Decision Tree Classifier | 0.862 - 0.86 - 0.86 - 0.86 |
| Random Forest Classifier | 0.886 - 0.89 - 0.89 - 0.89 |
| Gaussian NB Classifier | 0.655 - 0.78 - 0.65 - 0.61 |
| ANN | 0.867 - 0.87 - 0.87 - 0.87 |

# Logistic Regression:

- Used to predict the probability of a categorical dependent variable.
- The dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.).
- Predicts $P(Y=1)$ as a function of X.
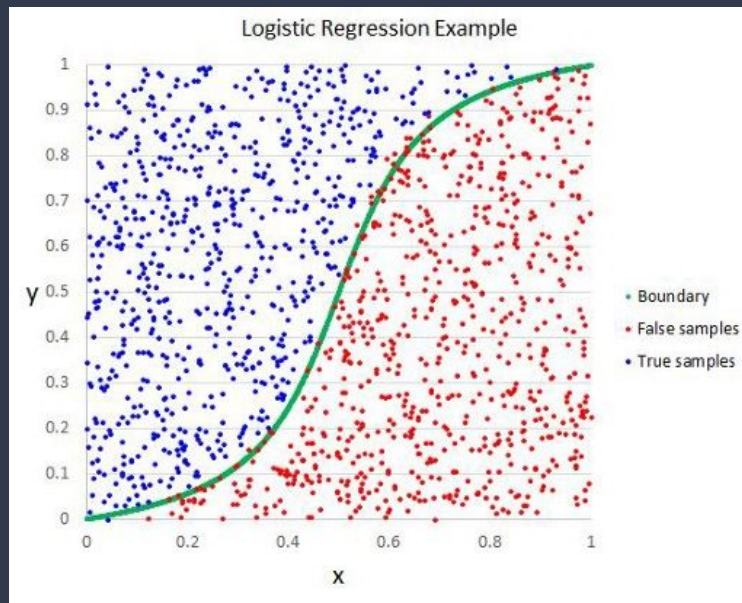
# SVM:

- Each data item is plotted as a point in an n-dimensional space with the value of each feature being the value of a particular coordinate.
- Classification is performed by finding the optimal hyperplane that differentiates the two classes very well.



Logistic Regression Example

- Boundary
- False samples
- True samples



Optimal hyperplane

Maximum margin
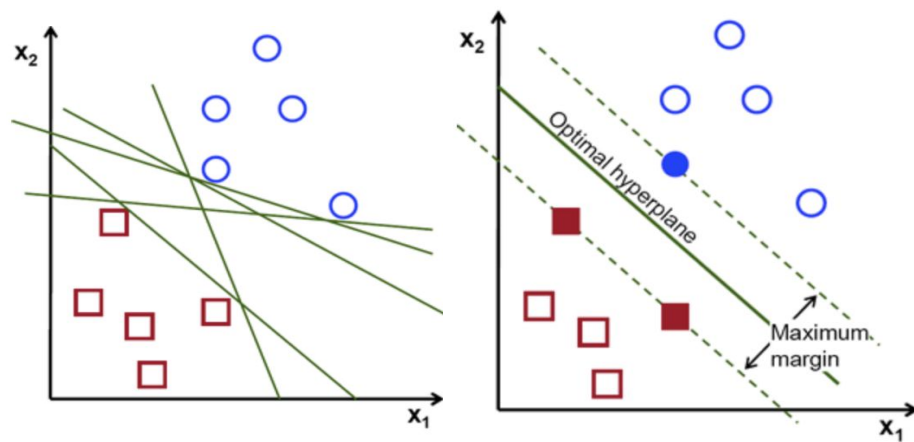
# Decision Tree Classifier:

- Repetitive division of the working area (plot) into sub part by identifying lines.
- At every stage, select the one that gives best information gain.
- When information gain is 0, the division stops.
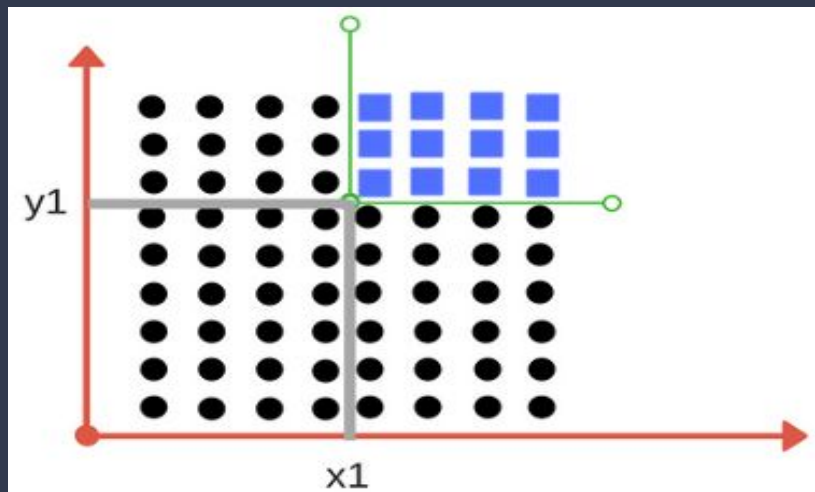


# Random Forest Classifier:

- Creates a set of decision trees from a randomly selected subset of training set.
- Then, it aggregates the votes from different decision trees to decide the final class of the test object.
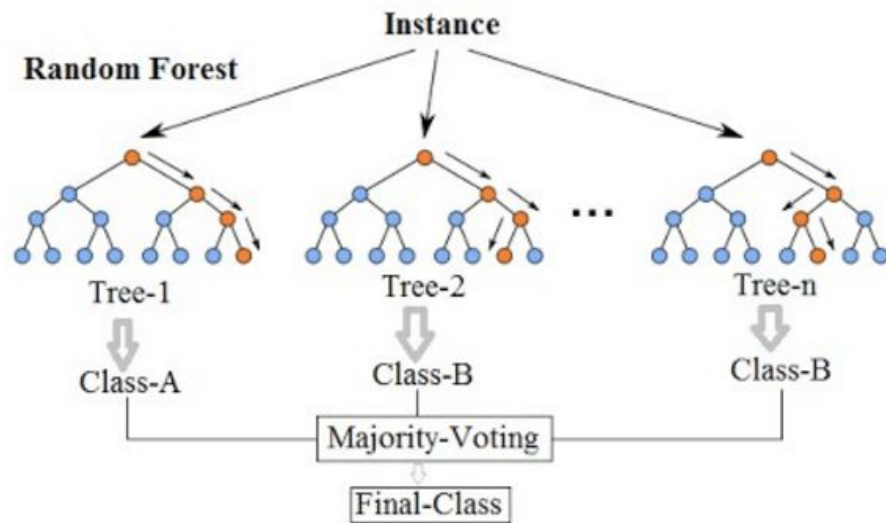
# Gaussian Naive Bayes:

- Helps us to find the probability of a hypothesis, given our prior knowledge.
- Describes the probability of an event, based on prior knowledge of conditions that might be related to the event

# Artificial Neural Network:

- An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain.
- Each connection, like the synapses in a biological brain, can transmit a signal to other neurons.
- An artificial neuron that receives a signal then processes it and can signal neurons connected to it.



**Likelihood**
How probable is the evidence given that our hypothesis is true?

**Prior**
How probable was our hypothesis before observing the evidence?

$$P(H \mid e) = \frac{P(e \mid H)\,P(H)}{P(e)}$$

**Posterior**
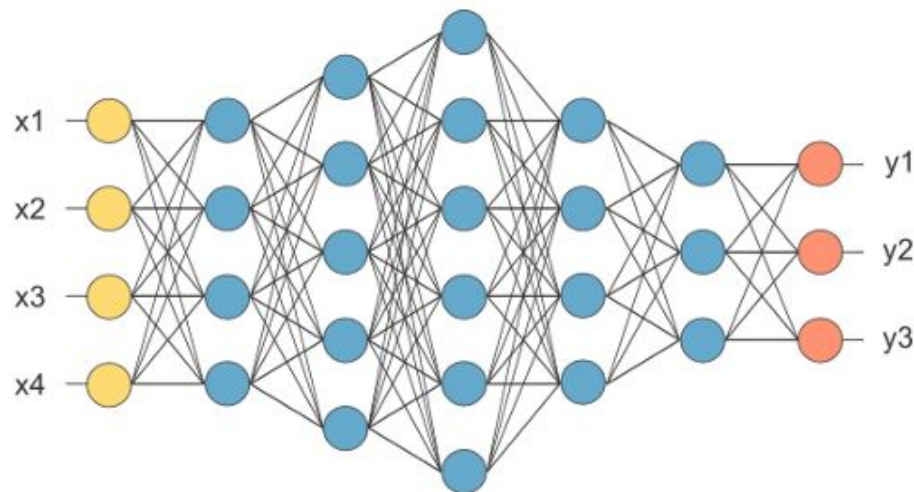How probable is our hypothesis given the observed evidence? (Not directly computable)
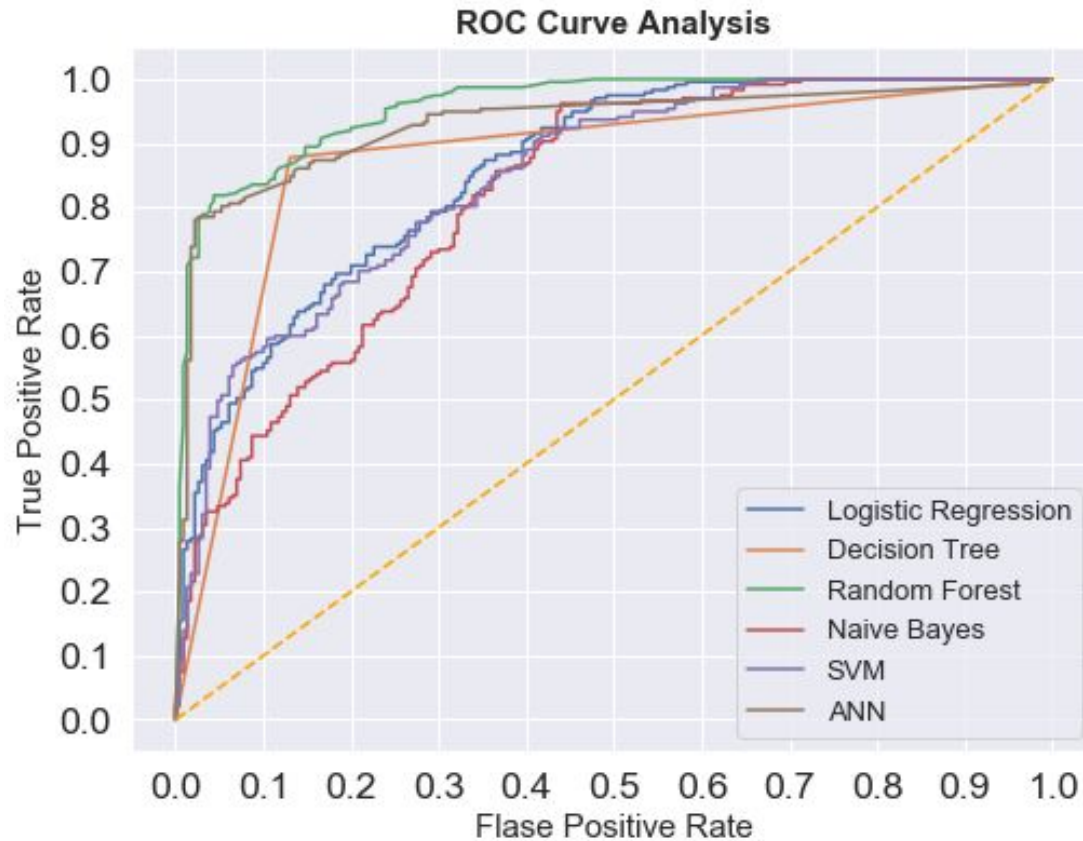
**Marginal**
How probable is the new evidence under all possible hypotheses?
$P(e) = \sum P(e \mid H_i)\,P(H_i)$
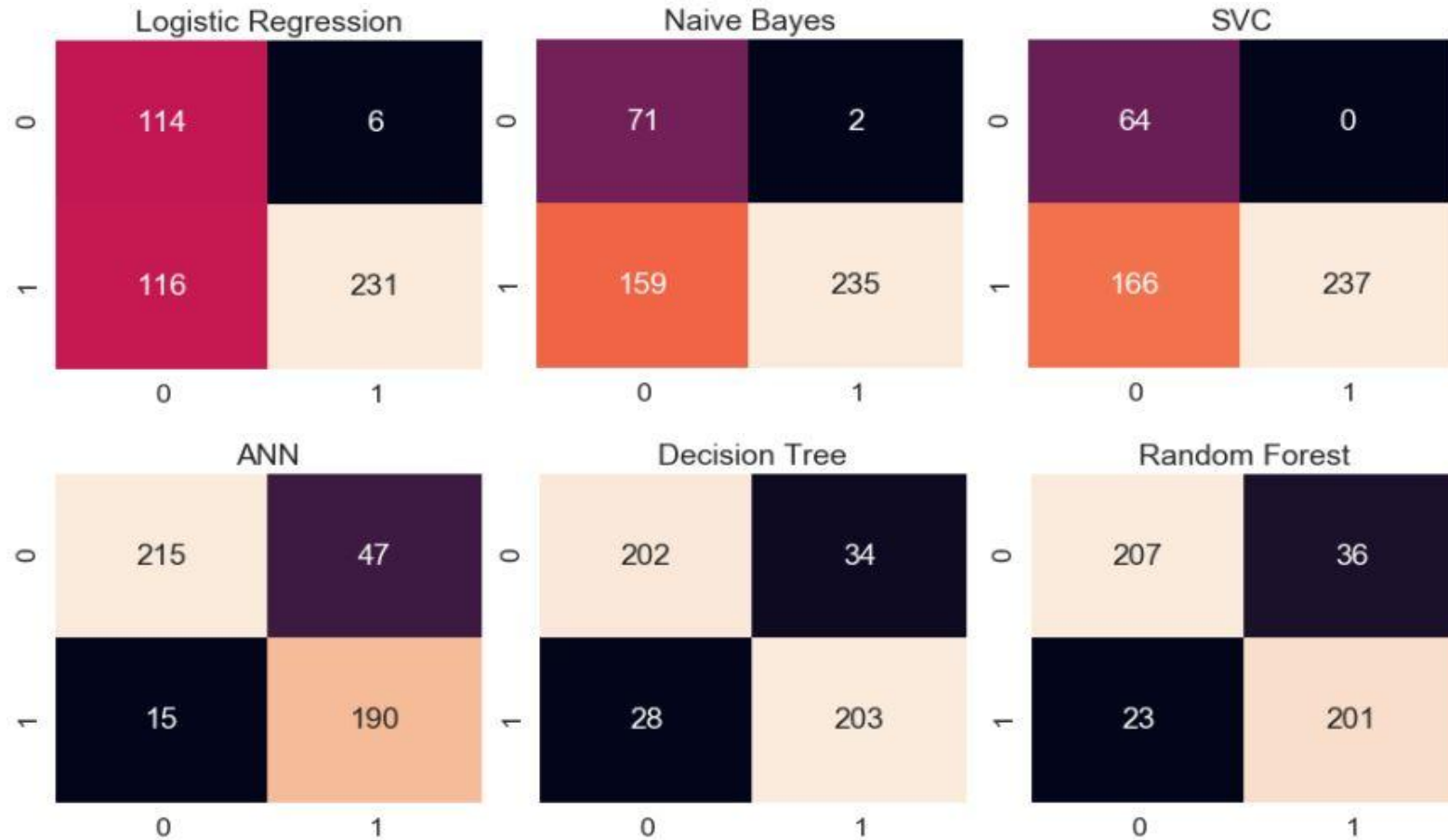
Confusion matrices of each model

ROC Curve Analysis

Comparison of ROC curves of all models.

# Conclusion:

❖ Thus, I was able to increase the accuracy of the models by 2-3% by using string detection applied to user names to filter twitter streams for bot accounts.

❖ Also, I was able to implement different machine learning algorithms on the dataset and provide a comprehensive study on comparison of the results for the same.

# Future Scope:

❖ This model was only implemented to detect bots from Twitter dataset. The model can be developed further for the detection of actual botnets on Twitter stream in real time and report these accounts for suspension and/or deletion. All the attributes need to be further researched and refinements need to be done.

# References:

[1] Jeremy D. Fields, "Botnet Campaign Detection on Twitter", Department of Computer Science, SUNY Polytechnic, Marcy, NY, 2018.

[2] Diogo Jeronimo, "Detection of Botnet Activity via Machine Learning", Instituto Superior Tecnico, Lisboa,     Portugal, November 2018.

[3] Beskow, David & Carley, Kathleen, "Its All in a Name: Detecting and Labeling Bots by Their Name",  Computational and Mathematical Organization Theory. 25. 10.1007/s10588-018-09290-1.

[4] Hoang XD, Nguyen QC, "Botnet Detection Based On Machine Learning Techniques Using DNS Query Data", *Future Internet*. 2018; 10(5):43.

[5] S. Gacia, M. Grill, J. Stiborek, A. Zunino, "An empirical comparison of botnet detection methods", Czech Technical University, Prague, May 2014.