2021

Password Strength Classification

Abhay Chaudhary, Mohammed Habeeb, Motiki Pavan Anudeep, Anagani Pavan Krishna

Vellore Institute of Technology, abhay.19bce7290@vitap.ac.in

Password Strength Classification

by
Abhay Chaudhary -19BCE7290
Mohammed Habeeb -19BCE7583
Motiki Pavan Anudeep -19BCD7067
Anagani Pavan Krishna -19BCD7209

A Case study report was submitted in partial fulfilment of the requirements
for the degree of Bachelors of Technology
in
Computer Science and Engineering

Department of Computer Science and
Engineering
Vellore Institute of Technology

2021

DECLARATION

I, at this moment, declare that this project entitled *"Password Strength Classification"* was submitted by me during the period of DECEMBER In Partial Fulfillment For The Award Of Degree Of Computer Science And Engineering At Vellore Institute Of Technology.

I also declare that this project is the result of my team effort. Therefore, it was not submitted earlier to any other person/university to award any other degree or diploma, either in whole or partial.

PLACE: VIT-AP

DATE: December 18, 2021

Abhay Chaudhary -19BCE7290

Mohammed Habeeb -19BCE7583

Motiki Pavan Anudeep -19BCD7067

Anagani Pavan Krishna -19BCD7209

ACKNOWLEDGEMENT

I express my sincere gratitude and wholehearted thanks to my Respected guide Dr Gopikrishnan S, Department of Computer Science And Engineering, Vellore Institute Of Technology Andhra Pradesh, who has guided us throughout the study helped us to complete the project successfully.

I want to acknowledge my sincere thanks to the entire Department of Computer Science And Engineering staff, friends, Team and parents, and everyone who helped me complete the project successfully.

I also express my heartfelt regards to my parents, guide, friends, and Team, who have directly or indirectly helped complete the project work.

Abhay Chaudhary -19BCE7290

Mohammed Habeeb -19BCE7583

Motiki Pavan Anudeep -19BCD7067

Anagani Pavan Krishna -19BCD7209

CERTIFICATION OF ORIGINALITY

This is to certify that the project entitled *"Password Strength Classification"* Submitted by Abhay Chaudhary and his Team on their work and carried out under my supervision and guidance. It is recommended that the candidate may now be evaluated for his project work by the institute.

Dr Gopikrishnan S

(Professor)

SUMMARY

We used various machine learning models to classify the passwords depending on their strength in our dataset. We can reasonably claim that SVM (radial) is the best method, with the highest accuracy and F1 score, followed by SVM and the Naive Bayes classifier. On the other hand, random Forest lags because it successfully classifies two classes but fails to categorize one. One of the reasons for this is that we utilize the party random forest library, which reduces the dataset to run compared to the standard libraries.

The same concept may be used for any group of passwords. We've also built a function that accepts the user's input and calculates the strength using these techniques, bolstering our claims.

TABLE OF CONTENTS

Abstract - Passwords can be used to access certain information, an account, a computer system, or a secure location. A single user can have several password-protected accounts. According to research, individuals prefer to use similar passwords for several accounts with minimal variation. Once a single password is discovered, it may be used to access various accounts. This article examines password security, taking a thorough look at what goes into creating a strong password and the difficulties of cracking one. The following parts cover related studies and demonstrate the many facets of password security, neglected weaknesses, and the relevance of passwords that are commonly overlooked visually and mathematically. This document helps to reduce the danger posed by individuals looking to reveal sensitive digital information.

Index Terms—Password, Security, Password Strength.

1. INTRODUCTION

Since the emergence of technology, passwords have become a significant element of people's lives. Since the dawn of technology, they have played a critical role in safeguarding our digital lives online. In a world where information security is becoming increasingly important, every organization's requirement for user access control is vital. Verifying a user's identity and granting appropriate access is known as authentication. Authentication may be accomplished using various techniques, such as cards or biometric characteristics such as iris scanners and fingerprint scanners. The difficulty with these approaches is that smartcards and other similar items might be lost or stolen, and biometric scanners are expensive and require additional resources that aren't always accessible, particularly in houses. Password authentication, the most prevalent type of authentication, is generally what a person knows. Passwords are the most frequent authentication method since they are free and easy to use, but they may also be the most dangerous if not set up correctly. Most passwords do not necessitate using a separate hardware device. Thus there is no cost to the company in purchasing one. Physical hardware devices may be utilized by anybody who has them, making them more likely to be misplaced or stolen. Password systems also do not need much hardware or computing power to function. Because users are already accustomed to using passwords to access, these authentication methods seldom require substantial training.

Passwords, unlike biometric information, may be quickly changed if they are hacked. However, if implemented with suitable rules and processes, passwords may give significant security benefits to both companies and people. A password is a secret that is remembered. Usually, a string of characters to verify a user's identity.

A party retains the secret termed the claimant, while the party validating the claimant's

identity is called the verifier, according to the NIST Digital Identity Guidelines. The verifier can infer the claimant's identity if the claimant successfully shows password knowledge to the verifier using an established authentication methodology. The efficiency of a password in repelling guessing and brute-force assaults is measured by its password strength. Therefore, the length and complexity of a password are crucial. The size of a password is merely the number of individual characters used in its construction, whereas complexity refers to the number of characters used in its production. The absolute risk of a security compromise is reduced when strong passwords are used.

2. RELATED WORK

Significant research has been done on passwords, security, authentication methods, and options beyond passwords. More secure alternatives to passwords exist. But as Herley et al. stated in their paper, there are several barriers to moving beyond passwords, such as diversity of requirements, user reluctance and usability, individual control of end-user systems etc. Today, alphanumeric passwords are still the most common mode of authentication; hence the focus rests on improving the security of passwords and their authentication. Halderman et al. bypass the need to remember multiple passwords for different accounts by using a strengthened hash function to generate high entropy passwords when they are needed. A single short master password protects these passwords. Udi Manber implemented two salts to prevent guessing attacks on passwords protected with one- way functions. So far, most of the existing research focuses on security management and storage of passwords. Passwords have long been used in the I.T. sector. They've been employed for various objectives, the most common of which safeguard someone's identity or data.

Many research articles have been published on this subject, and research has been ongoing for quite some time. Every study focused on a method for improving current research. For example, "Password Strength Prediction Using Supervised Machine Learning Techniques" and "A New Multimodal Approach for Password Strength Estimation—Part II: Experimental Evaluation" are two articles published in IEEE on this topic. To get better outcomes, these articles employed a variety of strategies. The dataset we chose can be found on Kaggle.com. The credentials utilized in the research came from the internet breach of 000webhost. The password's strength was determined using a tool developed by Georgia Tech University called PARS, which includes all commercial password metres.

All of the passwords were supplied to the device, and it generated new files for each commercial password strength metre. The passwords were stored in files with an additional column indicating their strength as determined by commercial password strength metres. Twitter, Microsoft, and battle are among the commercial password strength algorithms employed. Rather than rules, it is solely dependent on machine learning; only credentials that have been classified as weak, medium, or firm by all three strength metres were stored. This indicates that all passwords were weak, medium, or firm.

There were 3 million passwords, but only 0.7 million passwords remained after the intersection of all commercial metre classes. Because only those passwords were utilized that were highlighted in a specific category by all three algorithms, the decrease was achieved.

3. ASPECTS OF PASSWORD SECURITY

There are several factors of password security to consider. How passwords are saved is one of them. In addition, password storage must be secure to safeguard passwords against harmful attempts. Passwords can be stored in various ways, including plain text, hashing, salted hashing, and rainbow tables. It's also worth thinking about whether humans or computers produce the passwords. Computer-generated passwords are more random than human-generated passwords. Finally, theft of passwords is also a concern to consider. Social engineering, brute force, keylogging, and other methods can steal passwords. The subsections that follow go through the many components of password security.

A. Password Storage

A password can be made up of characters, numbers and special characters. Passwords are primarily case sensitive. Passwords can be entirely numeric. Passcodes are often used as PINs (Personal Identification Numbers) in ATMs and Net banking operations. Passwords are stored online in several ways. Some are much more secure than others, and some are vulnerable to attacks. The following section lists a few of the most popular ways.

Plain Text Passwords – This is the simplest form of storing a password. Somewhere on the site's server, there is a database that stores passwords and usernames in plain text. For example, if the password is _PassText321', the password is stored as _PassText321 'in the database. This is the worst form of storing passwords in terms of security. If the site is hacked and the passwords are stored in human-readable form, all the passwords are immediately compromised. The hacker can read all the passwords with virtually no extra effort.

Encrypted Passwords – Many sites store an encrypted form of the password in the database on their server. Encryption uses a unique key to convert the password into a random text string. The advantage is that the hacker cannot obtain the passwords without the key. All that can be obtained are the random encrypted strings. The disadvantage is that the key is often stored on the same server where the passwords are. So if the server is hacked and the key is retrieved, all the passwords can be decrypted and compromised. In addition, the very fact that encryption is reversible, i.e. a message can be coded and decoded, poses a security threat.

Hashed Passwords – Hashing is a function that will turn the password into a random long string of letters and numbers. The advantage of hashes over encryption is that hashes are irreversible. Once the password is hashed, there exists no algorithm to change it back to the original password. The hacker would have to hash several combinations one-by-one to see which hash matches with the one stored on the server. One way to do this is rainbow tables, which are computationally very fast. Hackers can also use a brute force attack, where every possible combination of letters and numbers are tried, hashed and matched with the hash retrieved from the database. This method can take a very long time and largely depends on how powerful the machine is. However, today, computers have become high-speed, and brute force attacks like John The Ripper can crack passwords quite efficiently. Different hashing algorithms like MD5, SHA-1, SHA-256, and SHA-512 exist.

Salted Hashes – To make hashes more secure, _salt 'can be added to the hash. A random string of characters is either prefixed or postfixed to the password before hashing it. Every password has a different salt. Even if the salts are stored on the database, cracking the passwords using a rainbow table will be very complicated as the salted passwords are long, complex, and unique. Salted hashes can be brute-forced, but the time taken is significantly longer. Using two salts, one public and one private can protect the password against offline attacks.

B. Human Generated Passwords Vs Randomly Generated Passwords

Passwords can be either human-generated or random generated. A random number generator generates a random string of numbers with characters from a pre-defined character set. Each character in the character set has the same probability of being chosen. A pseudorandom number generator (PRNG) generates a random sequence and has applications in cryptography. However, PRNG numbers are not random because they are generated from a small set of initial values. This set is Password Security: An Analysis of Password Strengths and Vulnerabilities called the PRNG's state, and a truly random seed is included within it.

Human-generated passwords are never really random. However, human-generated passwords are usually easy to remember. Humans choose passwords that are usually similar to some element of their lives, like addresses, birthdates, names of relatives, or words commonly used in everyday life. Passwords like _abcdefg 'or _123456 'are also commonly used. It is hard to remember so many different passwords with people possessing multiple accounts. So, most opt for using short, easy-to-remember passwords. This makes human-generated passwords more vulnerable and easy to guess. It has also been noted that web users tend to reuse their passwords. If a single password becomes known, more than one account will be compromised. Since most passwords are human-generated, individual users must make sure the passwords are strong and secure.

C. Password Theft

Passwords can be leaked in several ways. An attacker can hack into the site's database, storing the user credentials and uncover a vast number of passwords. Thefts can also occur on a personal level. A user can write down the password somewhere, making its way to malicious hands. Or a user can set a specific and obvious password that is easy to guess. Social engineering, phishing or keyloggers can also compromise passwords. Finally, passwords can commonly be uncovered by brute-forcing or offline dictionary attacks.

4. PASSWORD STRENGTH

A brute force attack tries every possible combination in a given character set and matches it against the original password. So more the number of possible combinations, the more time it takes for the algorithm to generate the guesses. On average, almost half of the combinations are tried before striking on the right one. Therefore, the longer it takes to break a password, the stronger it is. So it is logical to conclude that the greater the length of a password, the better it can stand against a brute force attack.

Let the length of the password that is to be cracked be
N. Let the password consist of only lower case alphabets. This forms the character set. The possible candidates for each character of the password are 26. Let the character set consist of k characters for a more generic case. Then the number of possible passwords can be N^k. So, the password length can increase by either increasing N or by increasing k.

If the password length is 6 and it is made up of only the lower case alphabets, then the number of possible passwords is 26^6, 308915776. If it were made of upper and lower case characters, the character set size would be 52, and the possibilities would be 52^6, which is 1.9770 x 10^{10}. If the password size is 7, the possibilities will become 26^7 and 52^7.

To prove that a longer password is more difficult to break than a shorter password, user-entered passwords were hashed and then brute-forced. An MD5 hash function first hashed passwords. Once the password is hashed, the combinations are created for a fixed length. Next, every combination is hashed using the same MD5 hash function and compared to the original password's hash. When a match is found, the function terminates. The word whose hash matched the original hash is the correct password. In the worst-case scenario, the code will test every single combination before finding a match. Finally, the time taken for each password to be broken is calculated and tabulated.

A. Numeric Tests

The first tests run were for 5 letter passwords. Then, time to break a single password was calculated, and the test was repeated for one hundred different passwords consisting of only lower case alphabets from a-z. The next set of tests was for 6 letter passwords. Again, the time required to break a single password was calculated, and the test was repeated for one hundred different passwords from a character set of lower case alphabets, a-z. The table shows 20 of the test results. As seen from table 1, the time required to break a six-letter password is much higher than a five-letter password. And it is also clear from the table that there is more or less a uniform increase in the time. As calculated graphically, the average time increase is 26.

B. Alphanumeric Tests

The next set of tests was run for calculating the time to break 6 alphanumeric letter passwords. Twenty passwords were tested for this. The alphanumeric passwords were compared to twenty randomly selected 6 letters alphabetical passwords, and their graphs were computed, which shows how much the password strengthens by adding to its character set. For alphanumeric passwords, the character set becomes 36. Hence for a 6 letter alphanumeric password, the number of possibilities are 36^6= 2176782336, and for a 6 letter alphabetic password, the number of possibilities is 26^6=308915776.

C. Multiple Case Tests

The next set of tests was run to calculate the time to brute force passwords comprised of both upper and lower case alphabets. The character set for multi-case passwords is 52. Twenty random passwords each of 6 letters were tested. These were compared to twenty lower case passwords, and their graphs were computed. For each 6 letters multi-case password, the number of possibilities is 52^6=19770609664, and for a 6 letter alphabetic password, the number of possibilities is 26^6=308915776. The graphical results corroborate that increasing the character set strengthens the password by a significant amount.

5. PROBLEM DEFINITION

The most important words or phrases used to protect someone's identity and data are passwords. They serve as your first line of defence against illegal access to your accounts and personal data. In addition, your computer will be safer from hackers and lousy malware if you use a strong password.

However, judging the strength of a password simply by glancing at it is challenging. Furthermore, determining a password's strength based on some guidelines is not recommended since it might be deceptive because various password metres employ different criteria to determine its strength. As a result, we used a database of existing passwords with specified strengths based on settings provided by commercial password metres. We constructed a machine learning model to predict if a password is weafounddium, or firm based on this data. Instead of using rules, we used machine learning to make our forecast.

Then, to determine the optimal ML algorithm, we compared numerous ML methods. We have included a feature that allows users to enter their passwords and retrieve the strength based on these algorithms. The main goal of our research is to determine the strength of a password and classify it as Strong, Weak, or Medium using machine learning methods (2,1,0). We compared the most prominent machine learning techniques for our dataset in addition to this classification. We've also introduced a feature that allows users to input a password and check it for strength using these methods. Finally, we must fulfil specific secondary objectives to achieve our core objectives, including preprocessing, comprehending the data, applying the appropriate models, and assessing the findings.

6. Methodology/Procedure

1) Gathering Data:

Kaggle.com was used to obtain the data. The passwords we utilized in our investigation came from the internet breach of 000webhost. The strength of the passwords in the dataset is pre-determined using a technology called PARS developed by Georgia Tech University, which includes all commercial password metres.

Twitter, Microsoft, and battle are among the commercial password strength algorithms employed.

a. There are eight columns in the dataset

b. The first column is called 'Id,' and it is unique.

c. The passwords are in the second column.

d. The password length is listed in the third column.

e. The percentage of capital, lowercase, numerals, and special characters in the password is represented in the fourth, fifth, sixth, and seventh columns, respectively.

f. The password's strength is shown in the last column, ranging from 0-Weak to 1-Medium to 2-Strong.

2) Preprocessing:

There were roughly 7 lakh rows and eight columns in our dataset. We chose a sample of 2 lakh rows for our study because the number of rows was enormous, and we didn't have the physical capacity to analyze the entire dataset. The Sr No column and the column holding the passwords themselves were deleted from our dataset. We deleted the password column since the type and quantity of characters in the password, not itself, determines our strength. Our second difficulty was that our dataset had no missing values. We added a function that generated missing values in the dataset at random. We have about 10% missing values in our dataset after that. We chose to mean to handle missing data since it gave us the best results, and we used mode for the Strength (Class) column because mean couldn't be used because it would establish a new class on its own. For our algorithms to work, we needed to modify the type of columns.

We changed the Strength column to factor for the algorithms and the length column to numeric from Integer. The next step was to see if any of the columns had a high association. We looked for a heatmap for this and couldn't locate any. After that, we moved on to the outliers. For this, we utilized the outlier's library. Outliers were deleted from all of the columns. The data has to be normalized next. Before normalization, we separated the data into train and test data in a 75:25 ratio. We then used the scale function to standardize the data. Finally, the caTools library was used to divide the data.

3) ML Models:

We started with a linear kernel Support Vector Machine (SVM). Then, we utilized the e1071 library. To analyze the data, we created a confusion matrix. After that, we switched to SVM with a radial kernel. The passwords were then categorized using a nave Bayes classifier. Then we moved on to Random Forest, for which we created a decision tree and a confusion matrix using the party library. The data were then compared and analyzed.

- Support Vector Machine (both linear and radial kernels): SVMs have supervised learning models that examine classification and regression analysis data. They come with related learning algorithms.

- Naïve Bayes Classifier: The Bayes' Theorem is used to create a set of classification algorithms known as Naive Bayes classifiers. It is a family of algorithms that share a similar idea, namely that each pair of characteristics being categorized is independent of the others.

- Random Forest: A random forest is a meta estimator that employs averaging to increase predicted accuracy and control over-fitting by fitting many decision tree classifiers on various sub-samples of the dataset. After that, we visualized our confusion matrix to make it easier to comprehend. We also designed a tool for the end-user to input a password and rapidly checked its strength.

7. Results and Discussion

```
'data.frame':   669640 obs. of  8 variables:
 $ X                  : int  0 1 2 3 4 5 6 7 8 9 ...
 $ Password           : chr  "kzde5577" "kino3434" "visi7k1yr" "megzy123" ...
 $ Length             : int  8 8 9 8 11 16 8 8 12 8 ...
 $ X.No.of.Upper.Case: num  0 0 0 0 0 ...
 $ X.No.of.Lower.Case: num  0.5 0.5 0.778 0.625 0.909 ...
 $ X.No.of.Numbers    : num  0.5 0.5 0.2222 0.375 0.0909 ...
 $ X.No.of.Spl.Chars : num  0 0 0 0 0 0 0 0 0 0 ...
 $ Strength           : int  1 1 1 1 1 2 1 1 1 1 ...
```

Fig 1: Structure of the original dataset

```
  X          Password Length X.No.of.Upper.Case X.No.of.Lower.Case X.No.of.Numbers X.No.of.Spl.Chars Strength
1 0          kzde5577      8              0.0000          0.5000000      0.50000000                 0        1
2 1          kino3434      8              0.0000          0.5000000      0.50000000                 0        1
3 2         visi7k1yr      9              0.0000          0.7777778      0.22222222                 0        1
4 3          megzy123      8              0.0000          0.6250000      0.37500000                 0        1
5 4       lamborghin1     11              0.0000          0.9090909      0.09090909                 0        1
6 5 AVYq1lDE4MgAZfNt     16              0.5625          0.3125000      0.12500000                 0        2
```

Fig 2: First 6 rows of the original dataset.

```
       X              Password             Length       X.No.of.Upper.Case X.No.of.Lower.Case X.No.of.Numbers X.No.of.Spl.Chars     Strength
 Min.   :     0   Length:669640      Min.   :  1.00   Min.   :0.00       Min.   :0.00       Min.   :0.00      Min.   :0          Min.   :0.00
 1st Qu.:167322   Class :character   1st Qu.:  8.00   1st Qu.:0.00       1st Qu.:0.50       1st Qu.:0.19      1st Qu.:0          1st Qu.:1.00
 Median :334895   Mode  :character   Median :  9.00   Median :0.00       Median :0.67       Median :0.30      Median :0          Median :1.00
 Mean   :334779                      Mean   :  9.99   Mean   :0.05       Mean   :0.61       Mean   :0.33      Mean   :0          Mean   :0.99
 3rd Qu.:502185                      3rd Qu.: 11.00   3rd Qu.:0.00       3rd Qu.:0.78       3rd Qu.:0.40      3rd Qu.:0          3rd Qu.:1.00
 Max.   :669639                      Max.   :220.00   Max.   :1.00       Max.   :1.00       Max.   :1.00      Max.   :1          Max.   :2.00
 NA's   :67419                       NA's   :66887    NA's   :66859      NA's   :66938      NA's   :66698     NA's   :66840      NA's   :67500
```

Fig 3: Summary of the dataset after creating Na's

```
   X          Password Length X.No.of.Upper.Case X.No.of.Lower.Case X.No.of.Numbers X.No.of.Spl.Chars Strength
1  0          kzde5577      8              0.0000          0.5000000      0.50000000                 0        1
2 NA          kino3434     NA              0.0000          0.5000000      0.50000000                 0        1
3  2              <NA>      9              0.0000          0.7777778      0.22222222                 0        1
4  3          megzy123     NA              0.0000          0.6250000      0.37500000                 0        1
5  4       lamborghin1     11                  NA          0.9090909      0.09090909                 0        1
6  5 AVYq1lDE4MgAZfNt     16              0.5625          0.3125000      0.12500000                 0       NA
```

Fig 4: First six rows after inserting N.A.'s

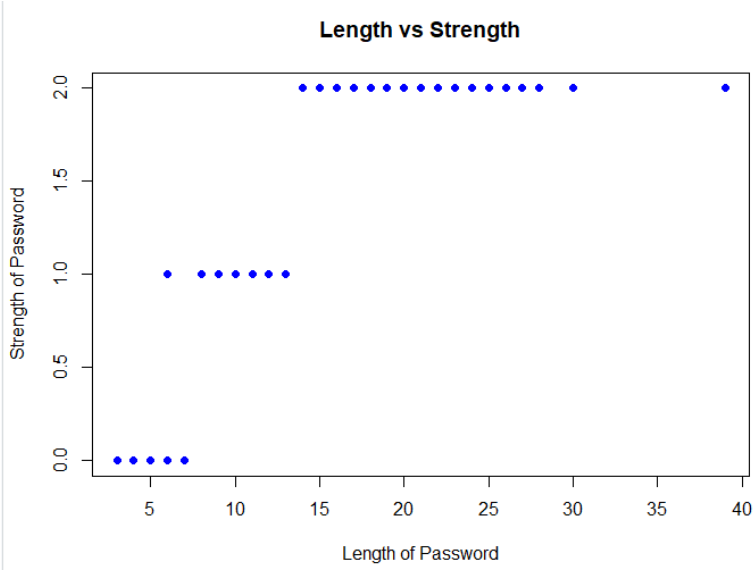


Fig 5: Basic Analysis from Dataset- Length vs Strength of the Password

We can see that as the password length increases, the strength also tends to increase. Also, our dataset has more passwords in the substantial region lengthwise.

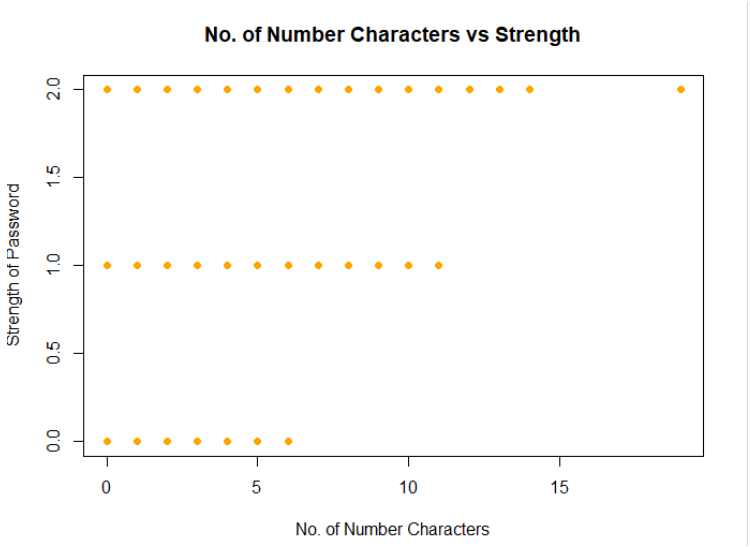


Fig 6: Basic Analysis from Dataset - No. of Number Characters vs Strength of the Password

The number of characters does not play as significant a part in determining strength as the length, but as the number of characters grows, so does the strength.

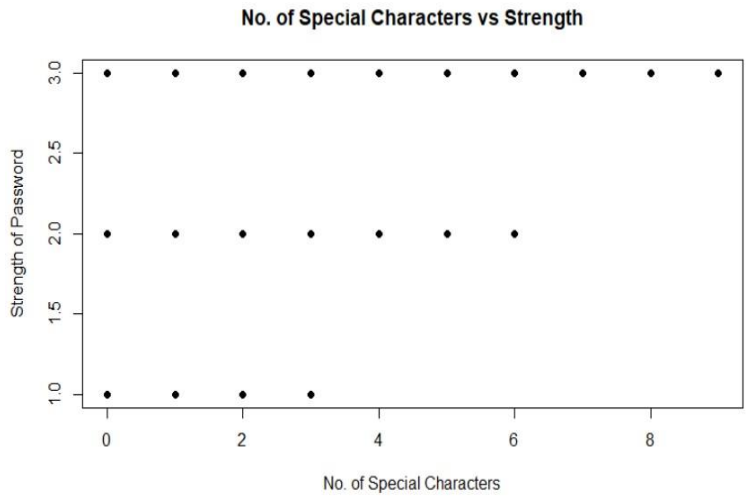


Fig 7: Basic Analysis from Dataset – Number of Special Characters vs Strength of the Password

An increase in special characters also increases the strength of the password.

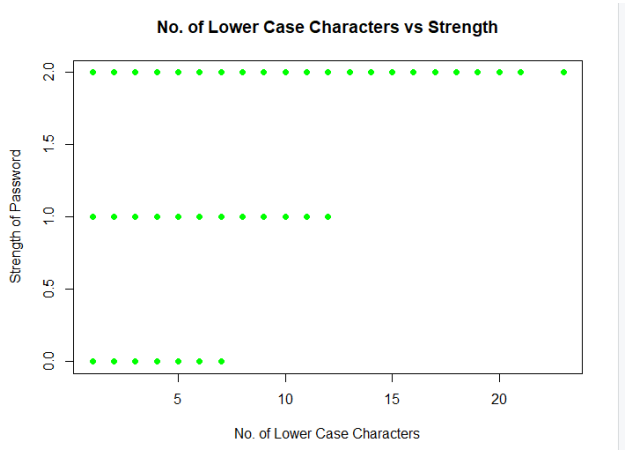


Fig 8: Basic Analysis from Dataset – No. of Lower Case Characters vs Strength of the Password

The more the number of lower case characters, the higher the strength.

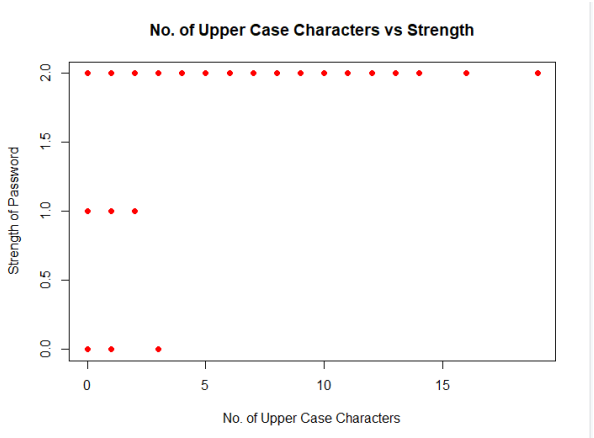


Fig 9: Basic Analysis from Dataset – No. of Upper Case Characters vs Strength of the Password

The collection contains many more items with a large number of upper-case characters. As the number of people grows, so does their power.

Also, we discovered that length is not the only criterion for determining strength since many passwords with more minor special characters, upper case, lower case, and other characters have high strength. We can deduce that both the length and the combination of different characters are equally important.

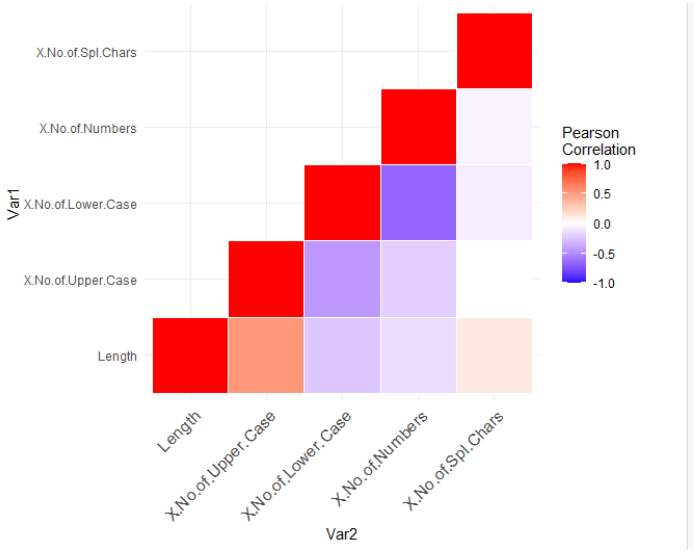


Fig 10: Correlation Analysis

No two columns are highly correlated with each other.

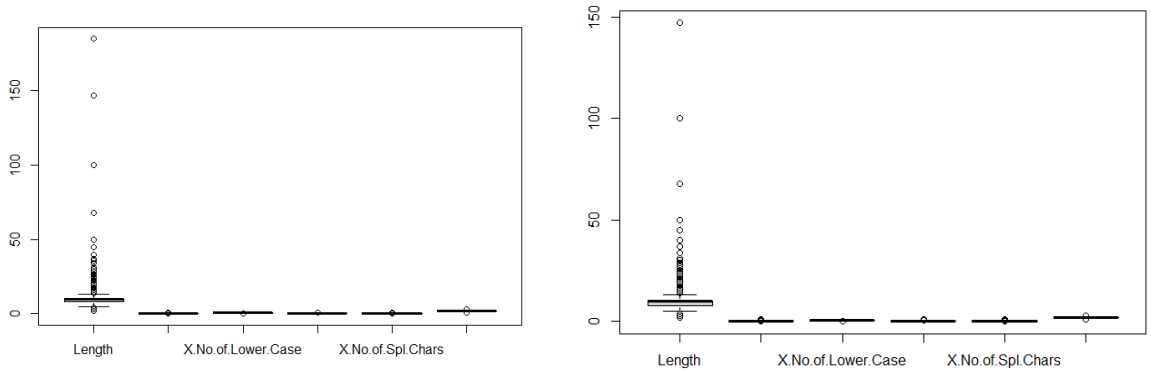


Fig 11: Boxplot Before and After Outliers

Outliers Detected and Removed

Length	1
Upper case	4
Lower Case	3329
No of numbers	11
Special characters	1

8. Analysis from MLmodels

Confusion Matrix SVM:

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1     2
         0  5261   611     0
         1   648 36501   594
         2     7   586  4955

Overall Statistics

               Accuracy : 0.9502
                 95% CI : (0.9483, 0.9522)
    No Information Rate : 0.7668
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.8705

 Mcnemar's Test P-Value : 0.04317

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity            0.8893   0.9682   0.8930
Specificity            0.9859   0.8917   0.9864
Pos Pred Value         0.8959   0.9671   0.8931
Neg Pred Value         0.9849   0.8952   0.9864
Prevalence             0.1203   0.7668   0.1129
Detection Rate         0.1070   0.7424   0.1008
Detection Prevalence   0.1194   0.7677   0.1128
Balanced Accuracy      0.9376   0.9300   0.9397
```

Confusion Matrix SVM (Radial):

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1     2
         0  5211   594     0
         1   702 36487    88
         2     3   617  5461

Overall Statistics

               Accuracy : 0.9592
                 95% CI : (0.9575, 0.961)
    No Information Rate : 0.7668
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8956

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity            0.8808   0.9679   0.9841
Specificity            0.9863   0.9311   0.9858
Pos Pred Value         0.8977   0.9788   0.8980
Neg Pred Value         0.9837   0.8981   0.9980
Prevalence             0.1203   0.7668   0.1129
Detection Rate         0.1060   0.7422   0.1111
Detection Prevalence   0.1181   0.7582   0.1237
Balanced Accuracy      0.9335   0.9495   0.9850
```

Confusion Matrix Naïve Bayes:

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1     2
         0  4808   552     1
         1  1005 36247   376
         2   103   899  5172

Overall Statistics

               Accuracy : 0.9403
                 95% CI : (0.9381, 0.9424)
    No Information Rate : 0.7668
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8452

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity            0.8127   0.9615   0.9321
Specificity            0.9872   0.8795   0.9770
Pos Pred Value         0.8968   0.9633   0.8377
Neg Pred Value         0.9747   0.8742   0.9912
Prevalence             0.1203   0.7668   0.1129
Detection Rate         0.0978   0.7373   0.1052
Detection Prevalence   0.1090   0.7654   0.1256
Balanced Accuracy      0.9000   0.9205   0.9545
```

Confusion Matrix Random Forest:

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1     2
         0  2612   438     3
         1  3131 33681    27
         2   173  3579  5519

Overall Statistics

               Accuracy : 0.8505
                 95% CI : (0.8473, 0.8536)
    No Information Rate : 0.7668
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.623

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity           0.44151   0.8934   0.9946
Specificity           0.98980   0.7246   0.9140
Pos Pred Value        0.85555   0.9143   0.5953
Neg Pred Value        0.92835   0.6741   0.9992
Prevalence            0.12033   0.7668   0.1129
Detection Rate        0.05313   0.6851   0.1123
Detection Prevalence  0.06210   0.7493   0.1886
Balanced Accuracy     0.71566   0.8090   0.9543
```

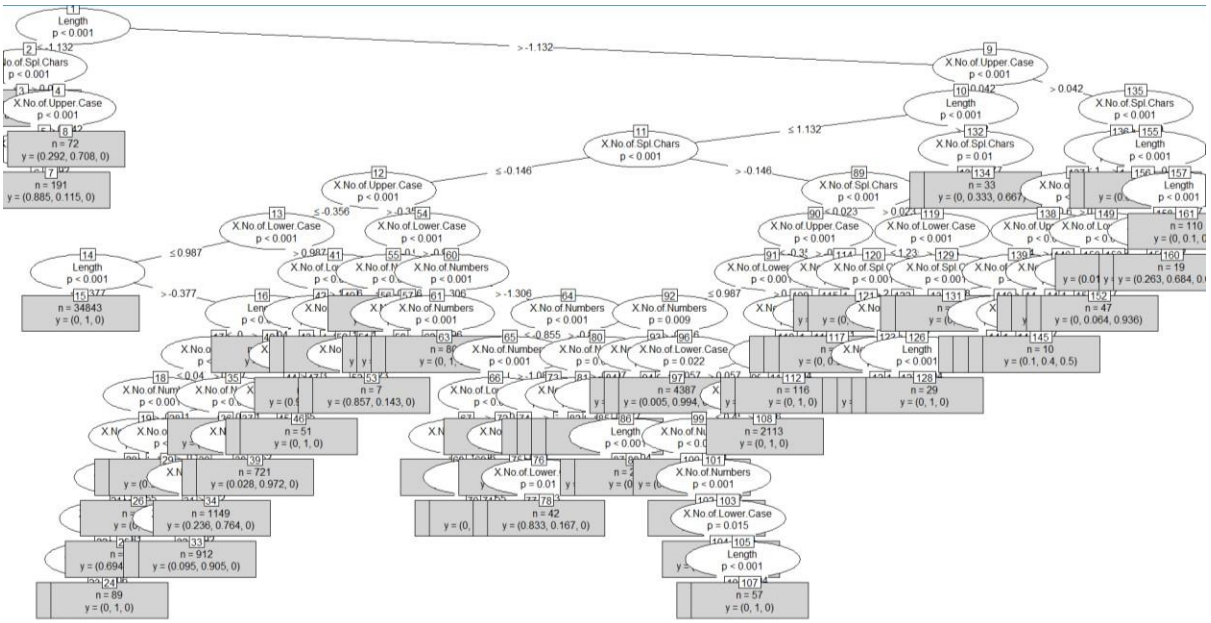
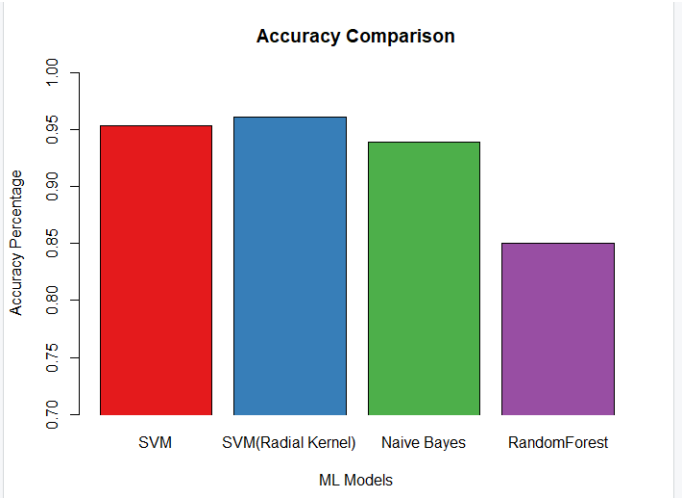


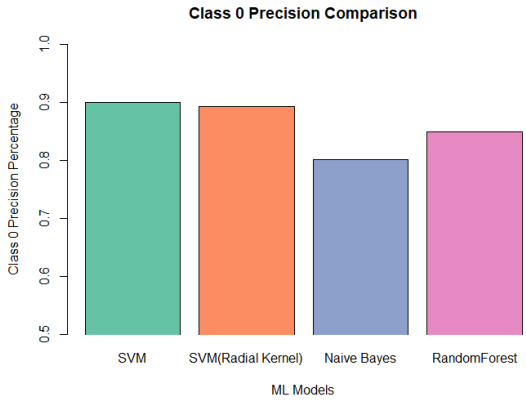
Fig 12: Decision Tree Using Random Forest

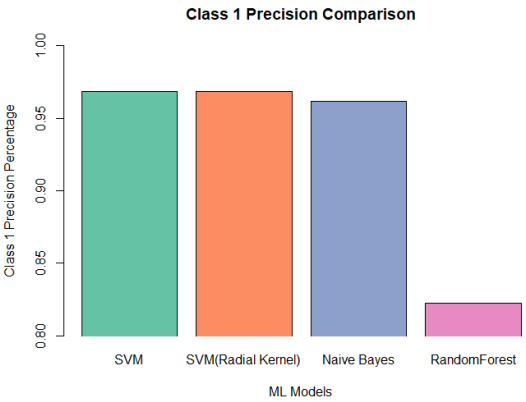
Accuracy:

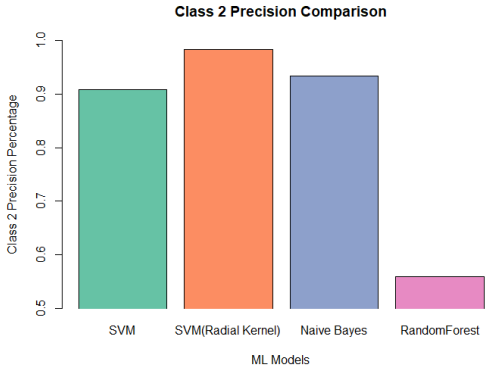


When we compare the accuracy of all the methods, we find that SVM – radial has the best accuracy, followed by SVM and nave Bayes, all of which are close to 95 per cent. The accuracy of random forest is 85 per cent. The SVM model should be used in this case.

Precision:



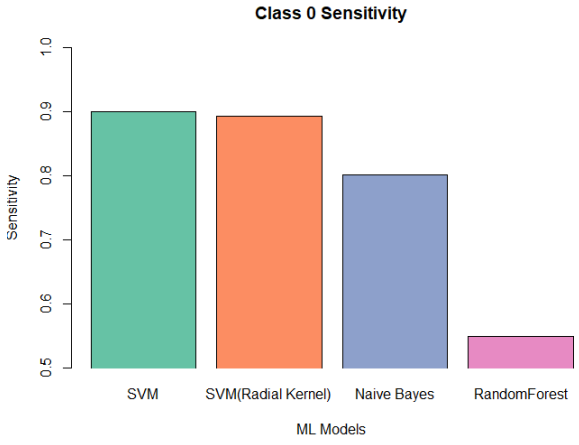


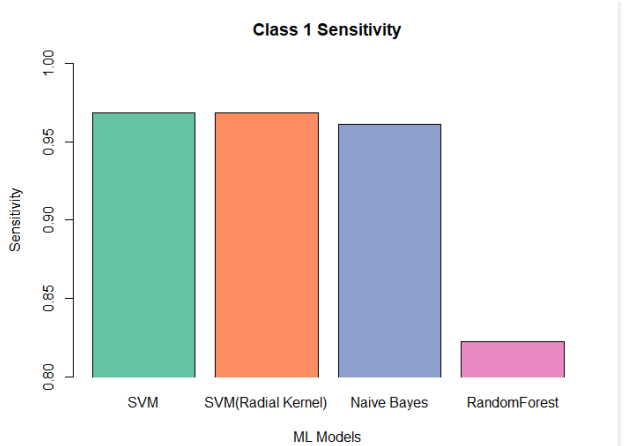


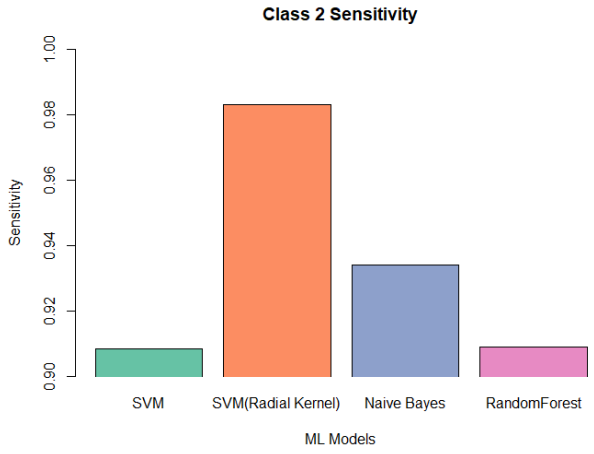
Precision (also known as positive predictive value) refers to the percentage of relevant results older than a certain age. For all of the classes, we can observe that SVM (both) have a good level of accuracy. For classes 1 and 2, Nave Bayes has high accuracy. However, for class 0, it fails. Only class 0 has a high level of accuracy, whereas the rest of the classes fail.

All of the algorithms have good accuracy, except for random forest for class 2, which has a lot of false positives.

Recall:



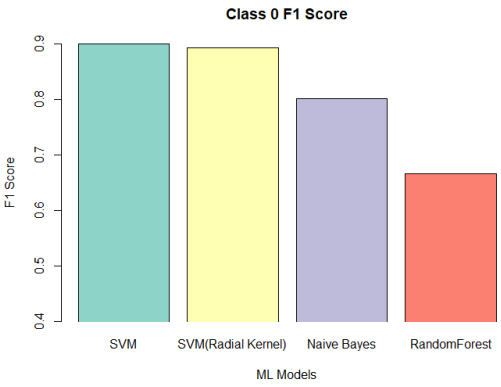


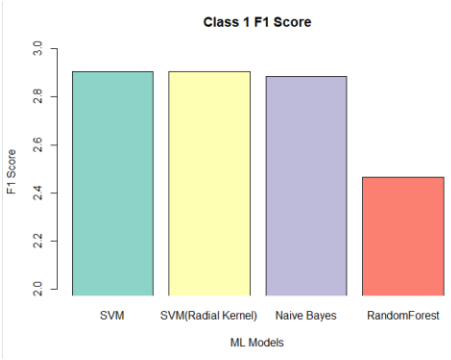


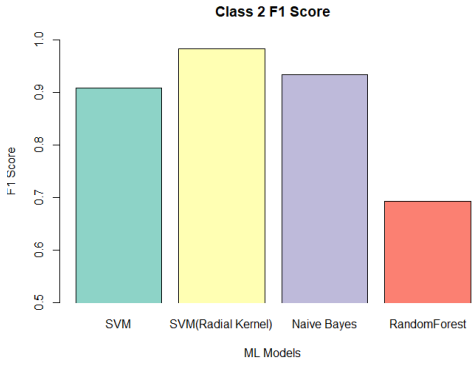
Recall (also called Sensitivity) refers to the precentage of total relevant results correctly classified by the algorithm.

SVM radial is better than the rest as it shows high recall for all the classes. SVM linear shows consistent recall for all classes at 0.9. Naïve Bayes and random forest show comparatively low recall than others.

F1 Score

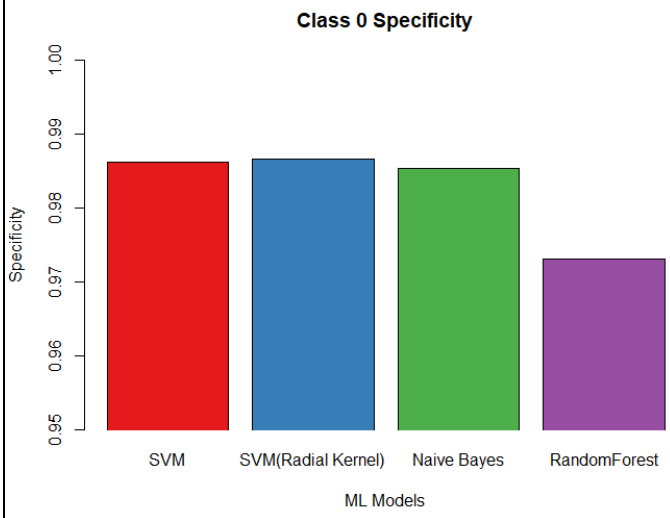


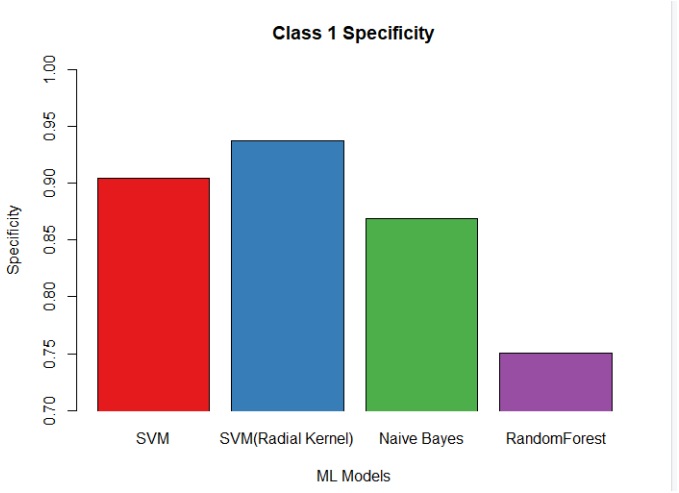


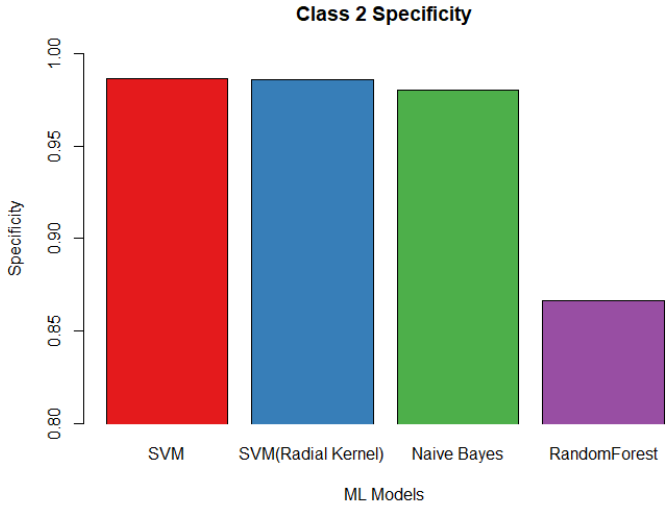


It is the harmonic mean of Precision and Recall and gives a better measure of the incorrectly classified cases than the Accuracy Metric.

Since we have imbalanced classes, the F1 score is a better metric. Now we can clearly say that SVM(radial) is better. SVM linear is better for binary classification and fails for multiclass


Class 0 Specificity
Specificity
1.00
0.99
0.98
0.97
0.96
0.95
SVM
SVM(Radial Kernel)
Naive Bayes
RandomForest
ML Models


Class 1 Specificity
Specificity
1.00
0.95
0.90
0.85
0.80
0.75
0.70
SVM
SVM(Radial Kernel)
Naive Bayes
RandomForest
ML Models


Class 2 Specificity
Specificity
1.00
0.95
0.90
0.85
0.80
SVM
SVM(Radial Kernel)
Naive Bayes
RandomForest
ML Models

User Prediction Function:

```
> length = nchar(str)
> upp=str_count(str, "[A-Z]")
> loo=str_count(str, "[a-z]")
> num=str_count(str, "[0-9]")
> spec=length-(upp+loo+num)
> vec<-c(length,upp/length,loo/length,num/length,spec/length)
> new_df=data.frame(vec)
> new_df=transpose(new_df)
> names(new_df)[1] <- "Length"
> names(new_df)[2] <- "X.No.of.Upper.Case"
> names(new_df)[3] <- "X.No.of.Lower.Case"
> names(new_df)[4] <- "X.No.of.Numbers"
> names(new_df)[5] <- "X.No.of.Spl.Chars"
> new_df[] = scale(new_df)
> print("Prediction form SVM")
[1] "Prediction form SVM"
> y_pred = predict(classifier_svm, newdata = new_df)
> print(y_pred)
factor(0)
Levels: 0 1 2
> print("Prediction form Kernel SVM")
[1] "Prediction form Kernel SVM"
> y_pred = predict(classifier_ksvm, newdata = new_df)
> print(y_pred)
factor(0)
Levels: 0 1 2
> print("Prediction form Naive Bayes")
[1] "Prediction form Naive Bayes"
> y_pred = predict(classifier_nb, newdata = new_df)
> print(y_pred)
[1] 1
Levels: 0 1 2
> print("Prediction form Random Forest")
[1] "Prediction form Random Forest"
> y_pred = predict(rf, newdata = new_df)
> print(y_pred)
[1] 1
Levels: 0 1 2
```

For password

“abhAy238” Strength

given by SVM:0

SVM(radial) :0

Naïve Bayes :1

Random Forest :1

9. Conclusion and Future Scope

We used various machine learning models to classify the passwords depending on their strength in our dataset. We can reasonably claim that SVM (radial) is the best method, with the highest accuracy and F1 score, followed by SVM and the Naive Bayes classifier. On the other hand, random Forest lags because it successfully classifies two classes but fails to categorize one. One of the reasons for this is that we utilize the party random forest library, which reduces the dataset to run compared to the standard libraries.

The same concept may be used for any group of passwords. We've also built a function that accepts the user's input and calculates the strength using these techniques, bolstering our claims. This is an area where further research can be done. More categorization methods can be used, and the results can be compared. Newer strategies, such as ensemble, can help us achieve more accuracy, precision, recall, and, as a result, a higher F1 score. These approaches can also replace rule-based password metres because they are far more practical. We can avoid data breaches and assaults and make the world safer if we make our passwords stronger.

10. References

[1] https://www.kaggle.com/bhavikbb/password-strength-classifier-dataset

[2] http://en.wikipedia.org/wiki/Password

[3] F.Bergadano, B.Crispo, G.Ruffo, "Proactive password checking with decision trees", Proc. of the 4th ACM conference on computer and communications security, Zurich, Switzerland, 1997, pp 67-77.

[4] Giancarlo Ruffo, Francesco Bergadano, "EnFilter: A Password Enforcement and Filter Tool Based on Pattern Recognition Techniques", Springer Berlin / Heidelberg, 1611-3349 (Online), Volume 3617/2005.

[5] Vijaya MS, Jamuna KS, Karpagavalli S," Password Strength Prediction using Supervised Machine Learning Techniques", IEEE, 978-1-4244-5321-4, pp 401-405, 2009.

[6] Lan H. Witten, Eibe Frank, "Data Mining – Practical Machine Learning Tools and Techniques," 2nd Edition, Elsevier, 2005.

[7] John Shawe-Taylor, Nello Cristianini, "Support Vector Machines and other kernel-based learning methods", 2000, Cambridge University Press, U.K.

[8] Vapnik V.N," Statistical Learning Theory", J.Wiley & Sons, Inc., 1998, New York.

[9] Soman K.P, Loganathan R, Ajay V, "Machine Learning

[10] with SVM and other Kernel Methods", 2009, PHI, India.

[11] https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2

[12] https://ieeexplore.ieee.org/abstract/document/5376606

[13] https://www.ukessays.com/essays/information-technology/security-benefits-of- passwords-information-technology-essay.php

[14] https://medium.com/@faizann20/machine-learning-based-password-strength- classification-7b2a3c84b1a6