An

Industry Oriented Mini Project Report

On

PASSWORD STRENGTH EVALUATION ALGORITHM

Submitted in partial fulfillment of the Requirements for the award of the degree of

Bachelor of Technology

By

Mounika Bandaru

(20EG105404)

Surya Teja Kancha

(20EG105418)

Manvitha Peddapurapu

(20EG105434)

Under the guidance of

Dr.K.Madhuri

Associate Professor , CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ANURAG UNIVERSITY

VENKATAPUR– 500088

TELANGANA

2023-24

DECLARATION

We hereby declare that the report entitled "**A PASSWORD STRENGTH EVALUATION ALGORITHM**" submitted for the award of Bachelor of Technology Degree is our original work and the report has not formed the basis for the award of any degree, diploma, associateship, or fellowship of similar other titles. It has not been submitted to any other University or Institution for the award of any degree or diploma.

Mounika Bandaru

(20EG105404)

SuryaTeja Kancha

(20EG105418)

Manvitha Peddapurapu

(20EG105434)

CERTIFICATE

This is to certify that the Report entitled "**A Password Strength Evaluation Algorithm**" that is being submitted by **Mounika Bandaru** bearing Hall Ticket Number **20EG105404**, **SuryaTeja Kancha** bearing Hall Ticket Number **20EG105418**, **Manvitha Peddapurapu** bearing the Hall Ticket Number **20EG105434** in partial fulfillment for the award of Bachelor of Technology in Computer Science and Engineering to Anurag University is a record of Bonafide work carried out by them under my guidance and supervision.

The results embodied in this Report have not been submitted to any other University or Institute for the award of any degree or diploma.

Signature of The Supervisor

Dr.K.Madhuri Dean, CSE

Associate Professor

External Examiner 1

External Examiner 2

ACKNOWLEDGMENT

We would like to express our sincere thanks and deep sense of gratitude to project supervisor, **Dr.K.Madhuri,** Associate Professor, Department of CSE for her constant encouragement and inspiring guidance without which this project could not have been completed. Her critical reviews and constructive comments improved our grasp of the subject and led to the fruitful completion of the work. Her patience, guidance and encouragement made this project possible.

We would like to acknowledge our sincere gratitude for the support extended by **Dr. G. Vishnu Murthy**, Dean, Dept. of CSE, Anurag University. We also express our deep sense of gratitude to **Dr. V V S S S Balaram**, Academic coordinator.
Dr. **Pallam Ravi**, Project Co-ordinator and Project review committee members, whose research expertise and commitment to the highest standards continuously motivated me during the crucial stage of our project work.

We would like to express our special thanks to **Dr. V. Vijaya Kumar**, Dean, School of Engineering, Anurag University, for his encouragement and timely support in our B.Tech program.

Mounika Bandaru

(20EG105404)

SuryaTeja Kancha

(20EG105418)

Manvitha Peddapurapu

(20EG105434)

ABSTRACT

Passwords are a part of our daily life. A lot of information is protected and kept securely with these limited characters. It is important to make sure the password we chose is the right one. To determine the strength of a password, a deep analysis is needed. Passwords based on users' personal information are easy for the user to remember, and the user is more likely to give a password based on that.

So, this elaborated analysis shows the behavior of users and their tendency to use personal information. Basic analysis is done as preprocessing on weights of a password and it also shows which algorithm shows more accuracy. The Structure Segmentation Algorithm divides the shared passwords into components based on what type of personal information they are. This algorithm has proven to be consistent.

A Bi Directional Matching Algorithm gives the weight of a particular type of personal information and is flexible enough. A Scoring Algorithm provides strength by giving the score of a specific password. Along with all the algorithms mentioned user feedback is a prime approach. Based on several characteristics User Feedback Algorithm lets the user know what still needs to be given in the password to make it strong.

A Password Strength Evaluation system based on personal information is a productive way to enhance security. It gives an understanding of how passwords work and what is required to give a complex password.

LIST OF FIGURES

LIST OF TABLES

Table of contents

1. INTRODUCTION

There is an increasing importance for information security with the internet as a platform. To identify a user, password authentication is very much needed. Despite having access to both information security and authentication, users often and are very likely to give passwords that are easy for them to remember. These easily memorable passwords are always related to their personal life.

Even in choosing these passwords, the user makes sure they are taken from very basic information. Basic information like the short form of his or her name, birth date, email ID, phone number, and sometimes a specific identity number. So, the strength of a password depends on the type of password the users select.

Due to this it is impossible to select a universal algorithm for strength evaluation. For this kind of password, there needs to be an algorithm that considers personal information.

1.1 MOTIVATION

However, today's password strength evaluation methods do not necessarily consider user's personal information. Not having all aspects seen can lead to serious security vulnerabilities.

Generally, it is not recommended for a strong password to include personal information but if that is what a user needs, a way must be figured out to benefit all of those included. A password strength evaluation algorithm essentially based on the user's personal information would improve the security for the users who want to memorize their password. This algorithm would let the user securely use information.

1.2 EXISTING METHODS

To check for the strength of a password there were a lot of methods that for a general set of passwords. These methods can give a certain result but have their limitations. Most of these methods are not based on personal information. And some of them are related to personal information.

1.2.1 RULE-BASED METHOD

In the rule-based method, the strength of a password is evaluated based on the length and the type of characters used.

For example, let us consider a set of passwords: rhyslarsen - (1)
rudraru@ - (2)
Siya@0508 - (3)

Let us consider the rules of the evaluation:
The minimum length of the password should be equal to or greater than 6.
If similar types of characters are used, then it is a weak password.
If two types of characters are used, then it is a medium password.
If two or more types of characters are used, then it is a strong password.

In (1) the length of the password is 10 which is greater than 6 but the type of characters used are similar in that all are small letters, so the password **rhyslarsen** is classified as weak. In (2) the length of the password is 8 and two types of characters are used which makes it a bit more complicated to crack than (1). But **rudraru@** is still a medium password. In (3) three types of characters are used, and the minimum requirement of length is met so **Siya@0508** is a strong password.
Even though this method is easy and simple, its password is evaluated based on the basic rules. And even if the method categorized the password as strong, it is still a less complex and less secure one.

1.2.2 PATTERN MATCHING METHOD

This method evaluates the strength based on identifying a specific string in a password. It matches the patterns based on the following:

- Semantic matching
- Sequential arrangement characters.
- Keyboard layout
- Weak password dictionary set.

Consider a string ‘**ert**’

And a password- **serty123**

The password consists of the string ‘**ert**’, and the password can be considered weak as ‘**ert**’ is a part of the keyboard layout and can be easily cracked.

This method is more intense than the rule-based method, but the strength evaluation is dependent on how the weights are calculated and is subjective. It could change in a perspective way rather than being universal.

1.2.3 ATTACK ALGORITHM

It refers to attacking a password with any kind of attack algorithm and evaluating a password based on its resisting capacity.

For example, if we take **Brute Force attack**:

Initially, the time taken to crack the password under the attack is calculated. All possible combinations of characters are considered. A strength value is assigned based on this time. Passwords that would take a long time or are impossible to crack are strong and a password that is cracked easily is weak.

Attack algorithms give accurate and practical results. However, the strength evaluation depends on the type of attack algorithm used. The same password may have different strength values according to that. So, this method is not suitable all the time.

1.2.4 STRUCTURE SEGMENTATION ALGORITHM

It is to investigate how users’ password information is included in passwords. The Structure Segmentation algorithm categorizes the given information and divides it into segments. Each type of information is divided accordingly. The algorithm gives back segmented information in the form of a list.

While all the above-mentioned methods do not consider personal information, The Structure Segmentation algorithm is primarily based on private information.

For example,

If the user gives the name: rudra
Birthdate: 20030210
And so, information is given, it segments information as [‘rudra’,’2003’,’02’,10’]

This algorithm considers personal information but does not give anything to determine strength. It is just a basic analysis of the information.

1.2.5 BIDIRECTIONAL MATCHING ALGORITHM

The Bi-Directional Matching Algorithm checks if the password incorporates personal information. It matches the user’s given information with the passwords and returns tags for the password.

If the tag is returned as 0 it means that the password has a name factor in it.
If the tag is returned as 1 then it means that the password has a birthdate factor in it.

For example, if the user gives the name:siya

Birthdate: 19900109

Email: siya@gmail.com

Password: siya@1

Then the tag is returned as 0.
It also gives the length of the password.
The length of the above-given password is returned as 6
The Bi-Directional Matching algorithm still does not give the strength of a password. It only helps in the understanding of the relation between users’ passwords and their personal information.

1.3 PROBLEM STATEMENT AND OBJECTIVE

A password strength evaluation system allows the user to know when their password is weak and when it is strong. It is needed because the user always tries to fill out a password that is easy for them to remember. The existing methods, like the rule-based method, pattern-matching method, and attack algorithm have their drawbacks and work for limited parameters.

The implemented structure Segmentation and Bi-directional algorithms give the relationship between passwords and users' personal information, but they do not provide the strength and feedback to the user.

The proposed method provides an algorithm that considers users' sensitive personal information. It analyses user password structure, and the analysis helps in finding the relationship between password and personal information. As a result, the accuracy and authenticity of strength evaluation results are improved.

2. LITERATURE SURVEY

On Password Strength Measurements: Password Entropy and Password Quality.

Mariam. M. Taha,2013 et.al:

Password strength measurements are done based on password entropy and password quality. In this proposed method, the passwords considered are 8 characters-alphanumeric with special characters. For these kinds of passwords, high or low entropy, and high or low-quality passwords are identified in the analysis phase. It then concludes that high entropy passwords are also high-quality passwords and vice versa. Though this method has its advantages, it has a limited scope and lacks real-world data.

Password Strength Analyzer Using Segmentation Algorithms.

Sivapriya K,2020 et al:

Password strength analyzer using segmentation algorithms. It checks the strength of passwords using segmentation algorithms and runs an analysis to check whether the password is based on user attributes. It then segments the password into meaningful and nonmeaningful substrings and stores the length of meaningful passwords. It provides improved security and is flexible and user-friendly. This analysis is complex and raises privacy concerns.

Password Strength Prediction Using Supervised Machine Learning Techniques.

Vijaya MS,2009 et.al:

Password strength prediction using supervised machine learning algorithms. The proposed algorithm essentially uses Naive Bayes Classification, Support Vector Machine, and Decision Tree. When the results of all the models are compared it came to light that SVM works better than all other methods. Results were also compared with existing password strength-checking tools. Outcomes show that the machine-learning approach has a more efficient way of classifying complex passwords. It gives enhanced security and is considered to have high accuracy and scalability. However, data availability, maintenance, and overfitting are the biggest concerns of this method.

SI No	Author(s)	Method	Advantages	Disadvantages
1	Mariam.M.Taha Taqwa.A.Alhaj Ala.E.Moktar Azza.H.Salim Settana.M.Abdullah	Password entropy. Password quality indicator. A dictionary attack using a second-order Markov model.	Measures password strength in terms of its entropy and PQI at the lowest possible cost. Detects high entropy password that doesn't pass dictionary attack.	Limited scope Generalization of rules.
2	Sivapriya K Deepthi L.R	Segmentation algorithms: Maximum matching algorithm Triangular matrix algorithm Password segmentation algorithm.	Consideration of User Attributes. Optimal segmentation algorithm. Correlation analysis.	Ethical concerns Complexity.
3	Vijaya MS Jamuna KS Karpagavalli S	Machine Learning algorithms: Multilayer perception Decision tree induction Naive Bayes classification Support Vector Machine	Higher accuracy. Handling complex patterns. Feature Importance.	Dependency on training data. Overfitting. Data privacy concerns.

Table2.1: Literature Survey

3. PROPOSED METHOD

This proposed method investigates how the user's personal information is incorporated into their passwords while also measuring a correlation between them. The password strength evaluation method is processed.

Some of the key areas of the system are:

- Basic preprocessing.
- Scoring algorithm.
- User Feedback algorithm.

3.1 BASIC PREPROCESSING

A set of passwords are analyzed initially using logistic regression, XG Boost, Naive Bayes, and Decision Tree for their strengths. Then these algorithms are compared to find out which one is most accurate.

3.1.1 LOGISTIC REGRESSION

Logistic regression is a statistical method used for analyzing a dataset in which there are one or more independent variables that determine an outcome. It is primarily employed for binary classification problems, where the outcome can be one of two possible classes, often represented as 0 and 1. 0 is considered negative while 1 is considered positive. Logistic regression is also extended to handle multi-class classification.

Logistic regression is like linear regression except Logistic regression is used for solving the classification problems whereas Linear Regression is used for solving Regression problems. It is a part of supervised learning and can provide probabilities and classify new data using continuous and discrete datasets. It can easily determine the most effective variables used for the classification.

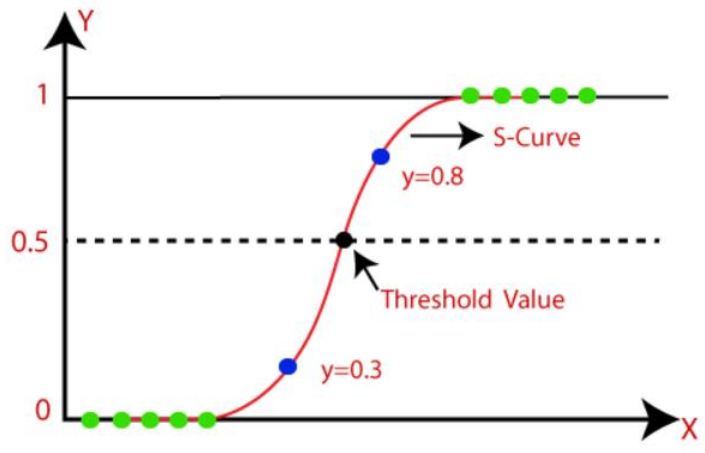


Figure.3.1: Sigmoid function

The above figure represents the logistic function or as it is called Sigmoid function. The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.

The sigmoid function is a mathematical function used to map the predicted values to probabilities. In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tend to be 1, and a value below the threshold value tends to be 0.

For logistic regression, the dependent variable must be categorical in nature and the independent variable should not have multi-collinearity.

Logistic Regression can be classified into three types:

- **Binomial:** In binomial Logistic regression, there can be only two possible types of dependent variables, such as 0 or 1, Pass or Fail, etc.
- **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Logistic regression is widely used in various fields, including finance, healthcare, marketing, and machine learning.

3.1.2 XG BOOST

XGBoost or Extreme Gradient Boosting is a popular and powerful machine learning algorithm that belongs to the class of gradient boosting algorithms. It is widely used for classification and regression tasks and is known for its efficiency, speed, and excellent predictive performance.

It is an optimized distributed gradient boosting library designed for efficient and scalable training of machine learning models. It is an ensemble learning method that combines the predictions of multiple weak models to produce a strong prediction.

In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and these variables are then fed to the second decision tree.

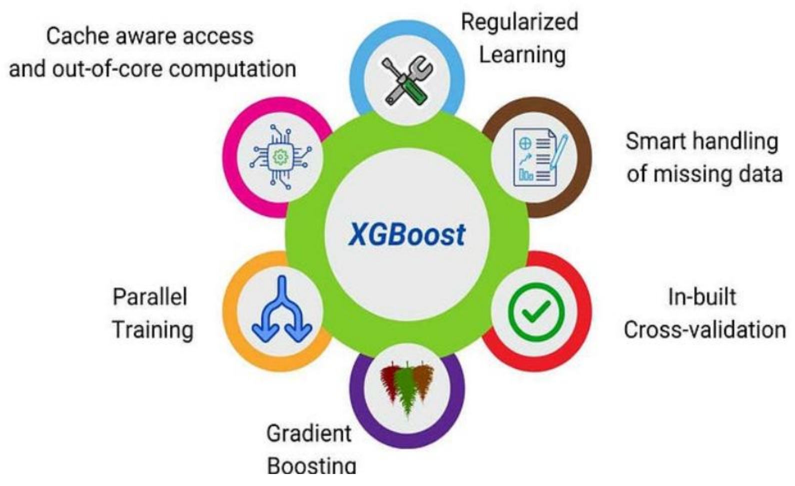


Fig.3.2: XGBoost

There are a lot of reasons to use XGBoost:

1. It has a strong track record of producing high-quality results in various machine learning tasks
2. It is designed for efficient and scalable training of machine learning models, making it suitable for large datasets.
3. It has a wide range of hyperparameters that can be adjusted to optimize performance, making it highly customizable.
4. It has built-in support for handling missing values, making it easy to work with real-world data that often has missing values.

XGBoost is particularly well-suited for structured data and tabular datasets, and it often outperforms other machine learning algorithms due to its powerful ensemble and regularization techniques.

3.1.3 DECISION TREE

A decision tree is a popular and interpretable machine learning algorithm used for both classification and regression tasks. It's a predictive modeling tool that is particularly suited for tasks where the relationship between the features and the target variable is nonlinear and complex. It is a supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical tree structure, which consists of a root node, branches, internal nodes and leaf nodes.

- It is simple to understand as it follows the same process which a human follows while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

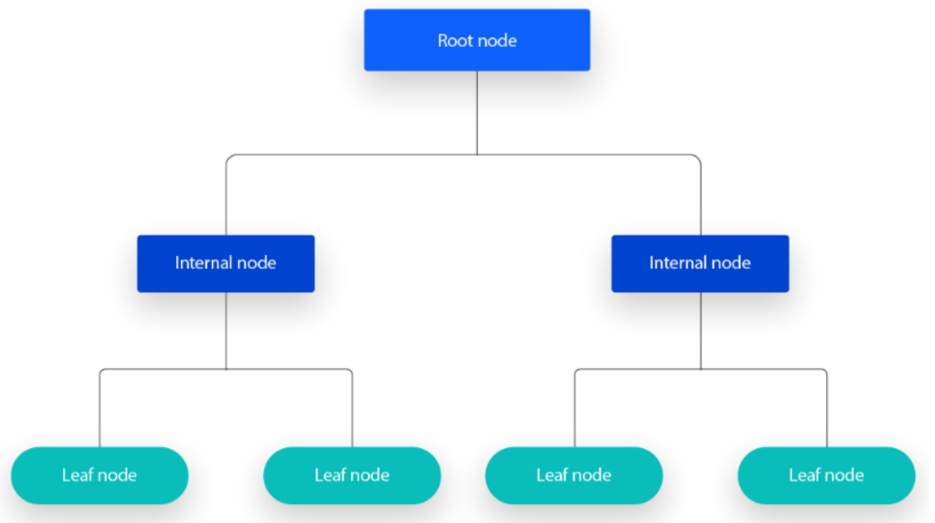


Fig.3.3: Decision Tree

Decision trees are a fundamental building block for many more complex machine learning algorithms, and they are used in various domains, including healthcare, finance, and marketing. While they are relatively simple, they can be powerful tools for solving a wide range of machine learning problems.

3.1.4 NAIVE BAYES

Naive Bayes is a probabilistic machine learning algorithm that is commonly used for classification tasks. It's based on Bayes' theorem and is particularly useful when working with text data, spam detection, and other applications where features are related to the probability of belonging to a particular class.

The fundamental Naive Bayes assumption is that each feature makes an independent or equal contribution to the outcome.

$$P(Y|X) = \frac{P(X \text{ and } Y)}{P(X)}$$

Fig.3.4: Naïve Bayes

In the above equations X and Y are considered to be events. P(Y|X) is the probability of Y occuring before X. P(X and Y) is the probability of both events occuring.

The most popular types of Naive Bayes are:

1. **Gaussian Naïve Bayes (GaussianNB)**: This is a variant of the Naïve Bayes classifier, which is used with Gaussian distributions—i.e., normal distributions—and continuous variables. This model is fitted by finding the mean and standard deviation of each class.

2. **Multinomial Naïve Bayes (MultinomialNB)**: This type of Naïve Bayes classifier assumes that the features are from multinomial distributions. This variant is useful when using discrete data, such as frequency counts, and it is typically applied within natural language processing use cases, like spam classification.

3. **Bernoulli Naïve Bayes (BernoulliNB)**: This is another variant of the Naïve Bayes classifier, which is used with Boolean variables—that is, variables with two values, such as True and False or 1 and 0.

The "naive" in Naive Bayes refers to the assumption that all features used in the classification are independent of each other, which is often not the case in real-world data. Despite this simplifying assumption, Naive Bayes can be surprisingly effective for various tasks, including spam detection, sentiment analysis, and document classification. It is known for its simplicity, speed, and ability to handle high-dimensional data.

The above-mentioned preprocessing methods help initial analysis of passwords and to help in understanding of predefined strengths of a general set of users.

3.2 SCORING ALGORITHM

A scoring algorithm is a mathematical or computational method used to assign scores or rankings to a set of items, data, or individuals based on specific criteria or attributes. They help quantify and compare the quality, relevance, or performance of different elements. The specific details of a scoring algorithm can vary significantly depending on the context and the problem it is designed to solve.

Scoring algorithm can be used to evaluate password strength based on information. The algorithm assigns a score to a password and categorizes them into strength levels. Some of the predefined scoring weights that were considered in implementation are:

Length_weight=2

Lowercase_weight=2

Uppercase_weight=2

Digit_weight=2

Specialchar_weight=2

Before the incrementation the score value is initialized to zero.

If the password has a length of 12 characters or more, it receives a score based on the length_weight. If the password is shorter than 12 characters, a message is added to indicate that the password length is too short.

If the password contains at least one uppercase letter (A-Z), it receives a fixed score based on the uppercase_weight. If the password contains at least one lowercase letter (a-z), it receives a fixed score based on the lowercase_weight.

If the password contains at least one digit (0-9), it receives a fixed score based on the digit_weight.

If the password contains at least one special character (!@#$%^&*(),.?":{}|<>), it receives a fixed score based on the special_char_weight.

Score+= attribute * len(password)

The attributes are the above mentioned criteria.

Then the score is summed up and based on the score the classification is done. For example,

If score < 5 then password is very weak.

If 5 ≤ score < 14 then password is weak.

If 15 ≤ score < 19 then password is moderate.

If 20 ≤ score < 24 then password is strong.

If score > 25 then password is very strong.

It makes it easier for the user to understand their passwords. The design and implementation of a scoring algorithm can significantly impact the quality of decisions and recommendations made in these domains.

3.3 USERFEEDBACK ALGORITHM

The user feedback algorithm provides feedback to the user about the strength of their password and offers suggestions for improvement. this algorithm works within the password strength and evaluate password functions:

3.3.1 SCORING AND CRITERIA EVALUATION

The password is evaluated against various criteria, and points are assigned based on the presence or absence of specific characteristics, such as length, uppercase letters, lowercase letters, digits, and special characters. The score variable accumulates points as the password meets these criteria.

3.3.2. FEEDBACK MESSAGES

The feedback list is used to store messages that provide feedback on the password's strengths and weaknesses. Feedback messages are generated when the password does not meet specific criteria. For example:

If the password is too short, a message like "Password length is too short" is added to the feedback list.

If the password lacks uppercase or lowercase letters, messages like "Include at least one uppercase letter" or "Include at least one lowercase letter" are added.

Similar messages are added if the password lacks digits or special characters.

If personal information is found in the password, a message like "Use personal info in your password in a secure way!" is added.

3.3.3 RETURNING FEEDBACK

The feedback list, along with the calculated score, is returned as part of the function's output. This allows the user to receive specific feedback on their password's strengths and weaknesses.

3.3.4. INCORPORATING FEEDBACK INTO PASSWORD EVALUATION

The evaluate_password function calls the password_strength function to assess the password. It captures the feedback provided by the password_strength function and then categorizes the password's overall strength. The feedback is not just about listing issues but also provides guidance on how to improve the password.

3.3.5. PRESENTING FEEDBACK TO THE USER

The main function, which is the entry point for the user, displays the feedback messages to the user, categorizes the password strength, and offers personalized recommendations based on the feedback messages. It presents the feedback and suggestions in a user-friendly manner.

Overall, this user feedback algorithm ensures that users receive clear and actionable feedback about their password's strength and specific areas where it can be improved. The feedback messages help users understand what aspects of their password need attention, making it easier for them to create more secure passwords.

4. IMPLEMENTATION

4.1 PACKAGES USED

Pandas: It provides data structure and function for efficiently working with structured data as tables.

NumPy: It stands for numerical python and is used for numerical computation. It supports arrays, matrices and mathematical functions.

Seaborn: Seaborn is a data visualization library built on top of Matplotlib, another popular data visualization library in Python. Seaborn provides a high-level interface for creating informative and attractive statistical graphics.

Matplotlib.pyplot: It is used for creating various types of static, animated and interactive visualizations.

Tfidvectorizer: it is used to convert text data into matrix of term frequency and inverse document frequency (TF-IDF) features.

Logistic regression: It is a statistical method used for analyzing a dataset in which there are one or more independent variables that determine an outcome.

XG Boost: It is an optimized distributed gradient boosting library designed for efficient and scalable training of machine learning models

MultinomialNB: It is a popular machine learning algorithm for test classification problems in natural language processing.

DecisionTreeClassifier: it is a classification algorithm provided by scikit-learn, a popular machine learning library in Python. It's part of the decision tree family of algorithms, which are used for both classification and regression tasks.

Re package: Regular expressions are a sequence of characters that define a search pattern. They are used for pattern matching within strings, making it possible to perform tasks like searching for specific words, extracting information from text, or validating input based on specific rules or patterns.

4.2 ATTRIBUTES

read.csv (): It is used to read a csv file.

shape: It returns a tuple, meaning rows and columns.

unique: It is used to retrieve the unique values present in the ‘strength’ column.

figsize(): It set the figure size and specifies the width and height of the figure in inches.

countplot(): It counts the occurrences of each unique value in specific column

np.array(): It converts the data frame into numpy array.

make_chars(): It divides a word into individual characters.

Tfidvectorizer(): It converts text documents into matrix.

fit_transform(): It is used to transform a list of text data into a matrix of TF-IDF features.

personal_info_categories: A dictionary that maps different types of personal information to their corresponding categories.

strength_category: A string attribute representing the strength category of the password, such as "Very Weak," "Weak," "Moderate," "Strong," or "Very Strong.

4.3 VARIABLES

1.length_weight: An integer representing the weightage assigned to the length of the password.

2.uppercase_weight: An integer representing the weightage assigned to the presence of uppercase letters in the password.

3.lowercase_weight: An integer representing the weightage assigned to the presence of lowercase letters in the password.

4.digit_weight: An integer representing the weightage assigned to the presence of digits in the password.

5.special_char_weight: An integer representing the weightage assigned to the presence of special characters in the password.

6.score: An integer representing the cumulative score of the password based on various criteria.

7.feedback: A list containing feedback messages for the user about the password.

8.matched_info: A dictionary that stores any personal information matched within the password.

9.password: A string variable representing the user's input for the password.

10.personal_info: A dictionary that stores various personal information entered by the user.

11.info_category: A string variable representing the category of personal information being checked.

12.info_value: A string variable representing the value of personal information being checked.

4.4 CODES FOR BASIC PREPROCESSING

#import libraries

```
import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.model_selection import train_test_split

import warnings

warnings.filterwarnings("ignore")
```

```
# Read csv file
dataset=pd.read_csv("data.csv",error_bad_lines=False)
# checking shape of the data
dataset.shape
# checking unique values in strength
dataset["strength"]. unique()
plt.figure(figsize= (8,8))
sns.countplot(dataset.strength)
#converting dataset into array
password_new=np.array(dataset)
password_new
#Convert words into characters
def make_chars(inputs):
    characters= []
    for letter in inputs:
        characters.append(letter)
    return characters
make_chars("team 9")
#converting word characters into numerical
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer=TfidfVectorizer(tokenizer=make_chars)

X_=vectorizer.fit_transform(X)
X_.shape
```

```
x_train,x_test,y_train,y_test=train_test_split(X_,y,test_size=0.27, random_state=42)

x_train.shape,x_test.shape
```

#model

```
from sklearn.linear_model import LogisticRegression

import xgboost as xgb

from sklearn.naive_bayes import MultinomialNB

from sklearn.model_selection import cross_val_score

classifier= []

classifier.append(LogisticRegression(multi_class='ovr',n_jobs=-1))

classifier.append(LogisticRegression(multi_class='multinomial',solver='newton-
cg',n_jobs=-1))

classifier.append(xgb.XGBClassifier(n_jobs=-1))

classifier.append(MultinomialNB())

result= []

for model in classifier:

   a=model.fit(x_train,y_train)

   result.append(a.score(x_test,y_test))

result1=pd.DataFrame({'score':result,

                    'algorithms': ['logistic_regr_ovr',

                                 'logistic_regr_mutinomial',

                                 'xgboost','naive bayes']})

result1

import matplotlib.pyplot as plt

import seaborn as sns
```

```
plt.figure(figsize=(8,8))

a=sns.barplot(x='score',y='algorithms',data=result1)

a.set_ylabel('accuracy')

plt.show()
```

4.5 SCORING AND USER FEEDBACK ALGORITHM

It is the combination of both Scoring and User Feedback Algorithm.

```
import re

def password_strength(password, personal_info):

    length_weight = 1

    uppercase_weight = 2

    lowercase_weight = 2

    digit_weight = 2

    special_char_weight = 2

    score = 0

    feedback = []

    matched_info = {}

    if len(password) >= 12:

        score += length_weight * len(password)

    else:

       feedback.append("Password length is too short.")

    if re.search(r'[A-Z]', password):

        score += uppercase_weight
```

```
  else:
        feedback.append("Include at least one uppercase letter.")
    if re.search(r'[a-z]', password):
        score += lowercase_weight
    else:
   feedback.append("Include at least one lowercase letter.")
    if re.search(r'[0-9]', password):
        score += digit_weight
else:
  feedback.append("Include at least one digit.")
     if re.search(r'[!@#$%^&*(),.?":{}|<>]', password):
        score += special_char_weight
    else:
        feedback.append("Include at least one special character.")
    personal_info_categories = {
        "name": "full name",
        "date of birth": "date of birth",
        "Identity number": "identity number",
        "email address": "email",
        "phone number": "phone number"  }
    for info category, info value in personal info. items():
        if info value .lower() in password. lower():
            matched_info[info_category] = info_value
   feedback.append(f"Use personal info in your password in a secure way!")
    return score, feedback, matched_info
def get_strength_category(score):
    if score < 5:
        return "Very Weak"
    elif score < 15:           return "Weak"
    elif score < 20:          return "Moderate"
elif score < 25:
        return "Strong"
    else:
        return "Very Strong"
def evaluate_password(password, personal_info):
    score, feedback, matched_info = password_strength(password, personal_info)
    strength_category = get_strength_category(score)
    return score, strength_category, feedback, matched_info
def main ():
    personal_info = {
        "name": input ("Enter your full name: "),
        "date of birth": input ("Enter your date of birth (YYYYMMDD): "),
        "identity number": input ("Enter your identity number: "),
        "email address": input ("Enter your email address: "),
        "phone number": input ("Enter your phone number: "),
          password = input ("Enter your password: ")
```

```
 score, strength, feedback, matched_info = evaluate_password(password,
personal_info)

    print(f"Password strength: {strength}")

    print(f"Password score: {score}")

    if strength in ["Strong", "Very Strong"]:

        print ("Congratulations! Your password is strong and secure.")

    elif strength == "Moderate":

 print ("Your password is moderately strong.")

    elif strength == "Weak":

        print("Your password is weak. Please consider making it stronger.")

    elif strength == "Very Weak":

        print ("Your password is very weak. Please choose a stronger one.")

    if matched_info:

        print("Personal information found in the password:")

        for info_category, info_value in matched_info.items():

            print(f"- {info_category.capitalize()}: {info_value}")

    if not matched_info:

        print("Personal information not found in the password.")

    if feedback:

        print("Suggestions to improve your password:")

        for suggestion in feedback:

            print(f"- {suggestion}")

if _name_ == "_main_":

    main ()
```

4.6 DATASET

The below mentioned dataset is used as a general dataset for the basic preprocessing. It helps with easy understanding and analysis.

https://www.kaggle.com/datasets/bhavikbb/password-strength-classifier-dataset

5. EXPERIMENT RESULTS

5.1 EXPERIMENT SCREENSHOTS

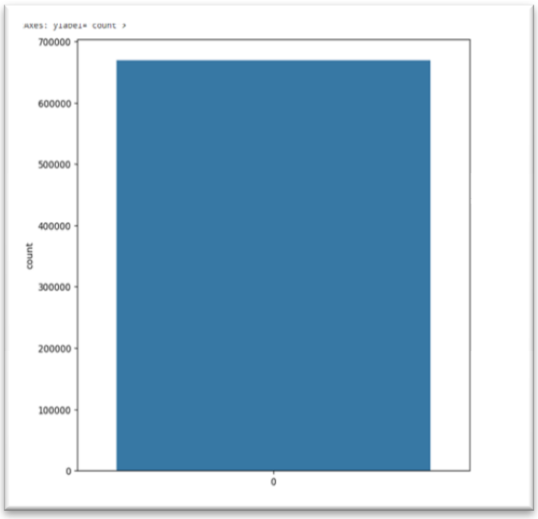
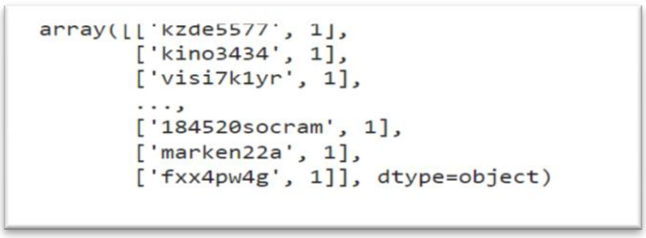


Fig.5.1: Countplot()



```
array([['kzde5577', 1],
       ['kino3434', 1],
       ['visi7k1yr', 1],
       ...,
       ['184520socram', 1],
       ['marken22a', 1],
       ['fxx4pw4g', 1]], dtype=object)
```

Fig.5.2: Dataset to array

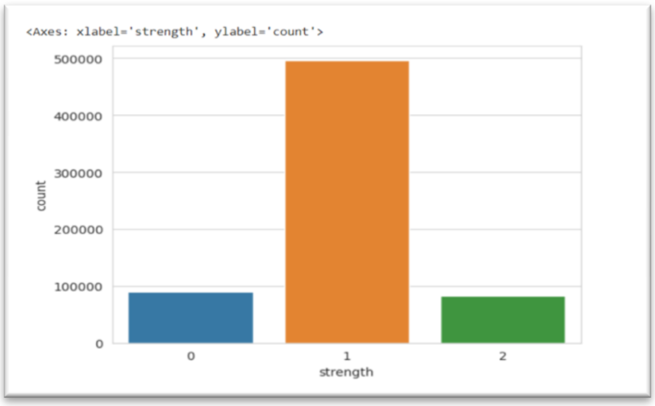
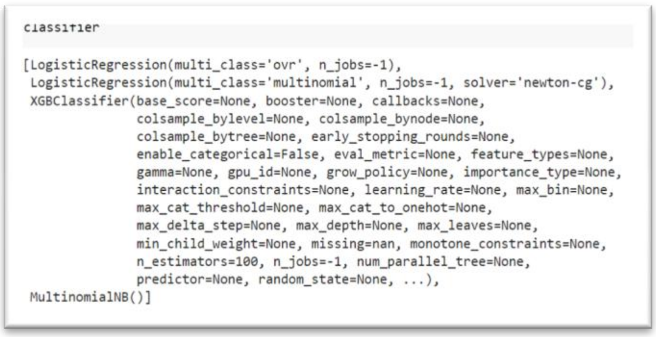
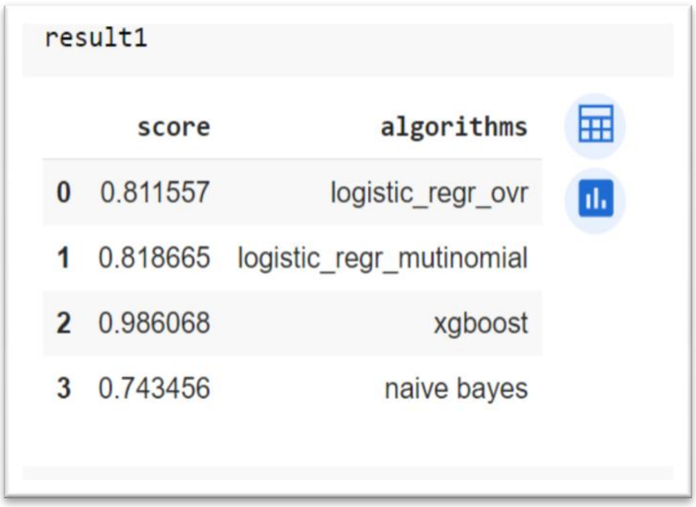


Fig.5.3: Comparison of strength counts



```
classifier

[LogisticRegression(multi_class='ovr', n_jobs=-1),
 LogisticRegression(multi_class='multinomial', n_jobs=-1, solver='newton-cg'),
 XGBClassifier(base_score=None, booster=None, callbacks=None,
               colsample_bylevel=None, colsample_bynode=None,
               colsample_bytree=None, early_stopping_rounds=None,
               enable_categorical=False, eval_metric=None, feature_types=None,
               gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
               interaction_constraints=None, learning_rate=None, max_bin=None,
               max_cat_threshold=None, max_cat_to_onehot=None,
               max_delta_step=None, max_depth=None, max_leaves=None,
               min_child_weight=None, missing=nan, monotone_constraints=None,
               n_estimators=100, n_jobs=-1, num_parallel_tree=None,
               predictor=None, random_state=None, ...),
 MultinomialNB()]
```

Fig.5.4:Classifier



```
result1
```

	score	algorithms
0	0.811557	logistic_regr_ovr
1	0.818665	logistic_regr_mutinomial
2	0.986068	xgboost
3	0.743456	naive bayes

Fig.5.5: Numerical comparison of algorithms

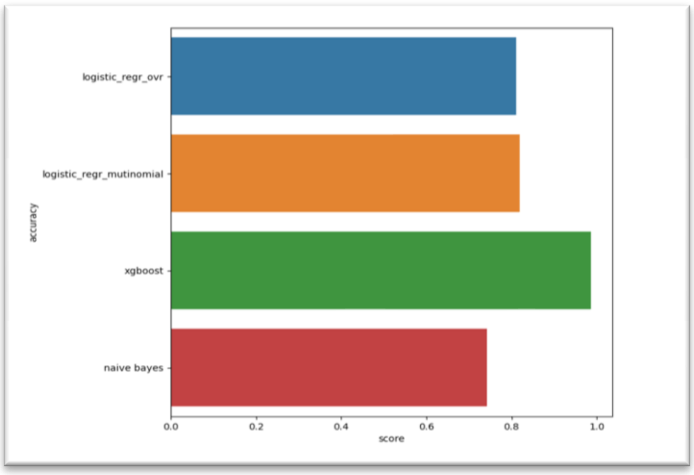


Fig.5.6: Bar graph comparison of algorithms

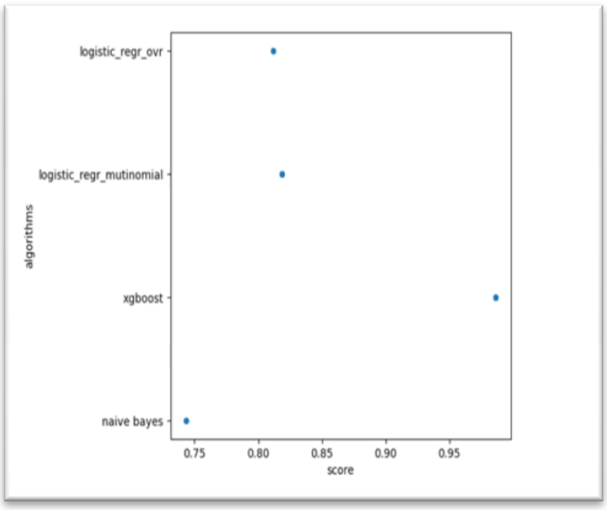
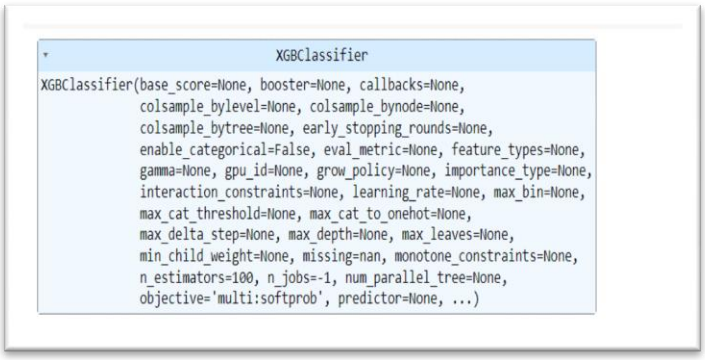
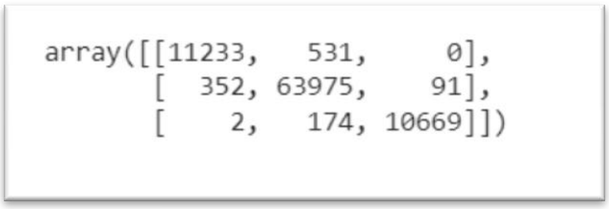


Fig.5.7: Plot comparison of algorithms



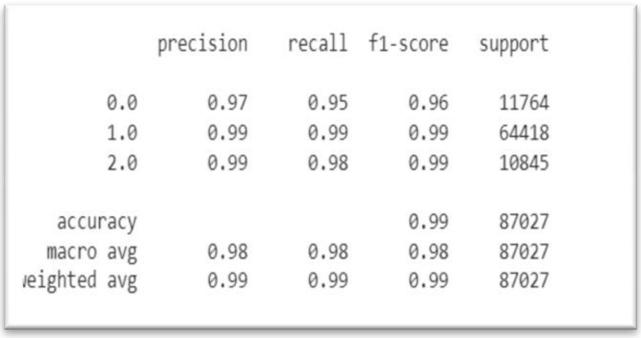
```
XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              n_estimators=100, n_jobs=-1, num_parallel_tree=None,
              objective='multi:softprob', predictor=None, ...)
```

Fig.5.8: Xgb_Classifierfit



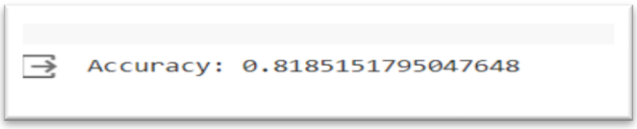
```
array([[11233,   531,     0],
       [  352, 63975,    91],
       [    2,   174, 10669]])
```

Fig.5.9: Confusion Matrix



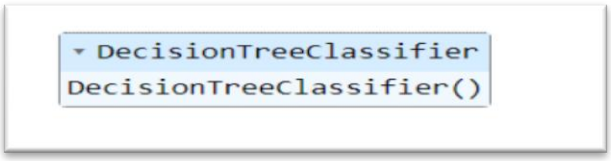
```
              precision    recall  f1-score   support

         0.0       0.97      0.95      0.96     11764
         1.0       0.99      0.99      0.99     64418
         2.0       0.99      0.98      0.99     10845

    accuracy                           0.99     87027
   macro avg       0.98      0.98      0.98     87027
weighted avg       0.99      0.99      0.99     87027
```

Fig.5.10: Classification report



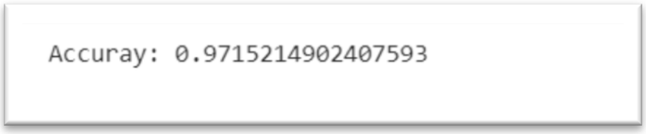
```
Accuracy: 0.8185151795047648
```

Fig.5.11: Accuracy of Logistic Regression



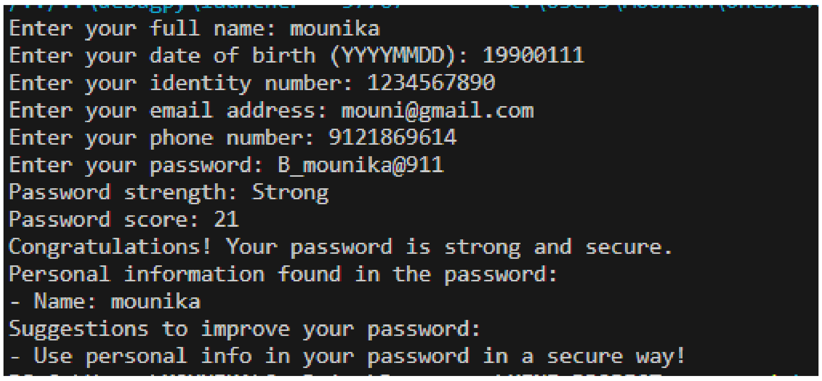
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

Fig.5.12: DecisionTreeClassifier fit



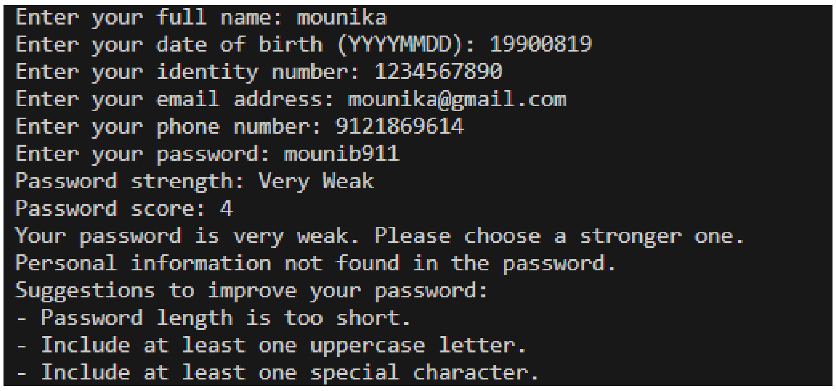
```
Accuray: 0.9715214902407593
```

Fig.5.13: Accuracy of Decision Tree



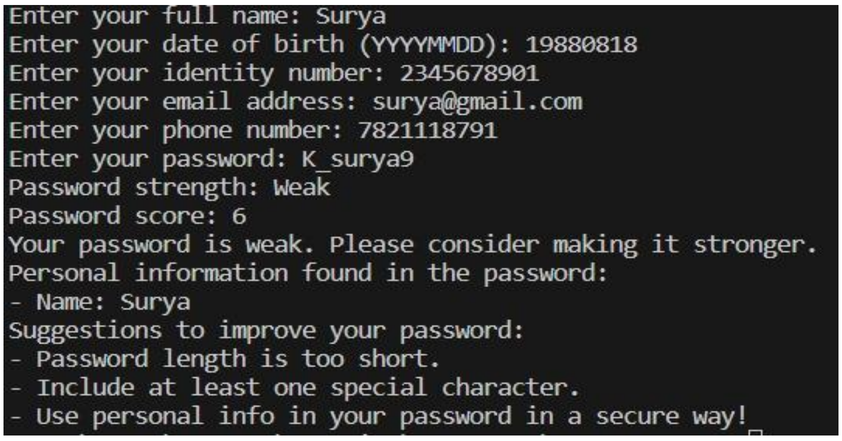
```
Enter your full name: mounika
Enter your date of birth (YYYYMMDD): 19900111
Enter your identity number: 1234567890
Enter your email address: mouni@gmail.com
Enter your phone number: 9121869614
Enter your password: B_mounika@911
Password strength: Strong
Password score: 21
Congratulations! Your password is strong and secure.
Personal information found in the password:
- Name: mounika
Suggestions to improve your password:
- Use personal info in your password in a secure way!
```

Fig.5.14: Strong password



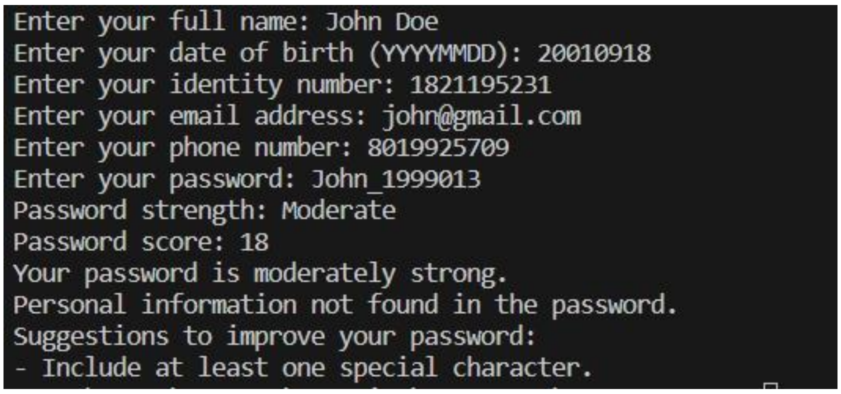
```
Enter your full name: mounika
Enter your date of birth (YYYYMMDD): 19900819
Enter your identity number: 1234567890
Enter your email address: mounika@gmail.com
Enter your phone number: 9121869614
Enter your password: mounib911
Password strength: Very Weak
Password score: 4
Your password is very weak. Please choose a stronger one.
Personal information not found in the password.
Suggestions to improve your password:
- Password length is too short.
- Include at least one uppercase letter.
- Include at least one special character.
```

Fig.5.15: Very weak password



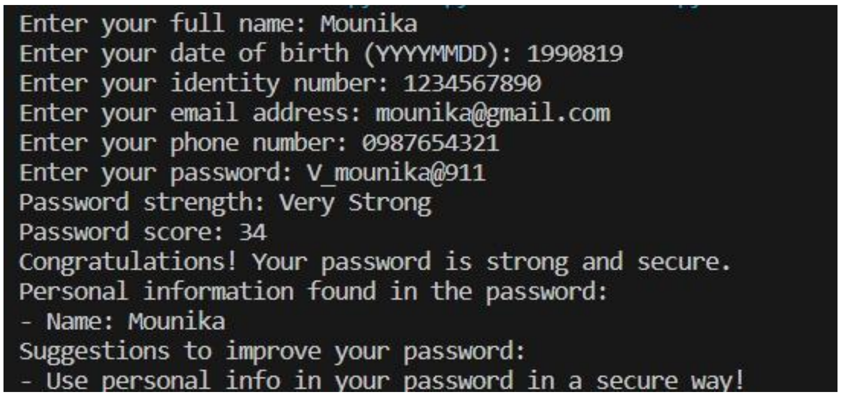
```
Enter your full name: Surya
Enter your date of birth (YYYYMMDD): 19880818
Enter your identity number: 2345678901
Enter your email address: surya@gmail.com
Enter your phone number: 7821118791
Enter your password: K_surya9
Password strength: Weak
Password score: 6
Your password is weak. Please consider making it stronger.
Personal information found in the password:
- Name: Surya
Suggestions to improve your password:
- Password length is too short.
- Include at least one special character.
- Use personal info in your password in a secure way!
```

Fig.5.16: Weak password



```
Enter your full name: John Doe
Enter your date of birth (YYYYMMDD): 20010918
Enter your identity number: 1821195231
Enter your email address: john@gmail.com
Enter your phone number: 8019925709
Enter your password: John_1999013
Password strength: Moderate
Password score: 18
Your password is moderately strong.
Personal information not found in the password.
Suggestions to improve your password:
- Include at least one special character.
```

Fig.5.17: Moderate password



```
Enter your full name: Mounika
Enter your date of birth (YYYYMMDD): 1990819
Enter your identity number: 1234567890
Enter your email address: mounika@gmail.com
Enter your phone number: 0987654321
Enter your password: V_mounika@911
Password strength: Very Strong
Password score: 34
Congratulations! Your password is strong and secure.
Personal information found in the password:
- Name: Mounika
Suggestions to improve your password:
- Use personal info in your password in a secure way!
```

Fig.5.18:Very Strong password

5.2 PARAMETERS

We’ve carried out initial analysis of passwords and their predefined scores with various algorithms and then figured out an algorithm combining Scoring and User Feedback algorithm to give strength of provided password as well as feedback about the password using users’ input.

5.2.1 ACCURACY IN ANALYSIS OF SCORES OF PASSWORDS.

When algorithms, Logistic regression, XG Boost, Naïve Bayes and Decision Tree were applied on the data set containing passwords and their scores to understand the behavior, it was shown that XG Boost and Decision Tree carry out the most accurate analysis compared to the remaining.

5.2.2 STRENGTH EVALUATION OF PASSWPRDS.

The Scoring Algorithm takes the input given by the user and calculates the score of a password. It then gives a score as a result and the level of the password that is it tells whether the password is very weak, weak, moderate or strong.

5.3.3 USER FEEDBACK

User feedback can be achieved by using the User Feedback algorithm. While giving the level of the password the algorithm gives feedback to the user. It gives suggestions to add something more in the password and it even tells if the password given is a good one.

6. DISCUSSION OF RESULTS

S. No	Name	Date Of Birth	Identity Number	E mail	Phone Number
1.	Kiran	21-01-1990	1234567890	kiran_91@gmail.com	7240322299
2.	K_Surya	18-08-1988	2345867901	surya@anurag.edu.in	7821118791
3.	John Doe	19-09-2001	1821119523	johndoe@gmail.com	6300067632
4.	Prudhvi Kiran	29-06-2000	7842373735	prudvi220@gmail.com	6309875421
5.	Suryam	05-05-2005	2030198765	suryam1234@gmail.com	7890654312

Table 6.1: User personal information

The above table represents the user information that will be given to the system before entering the password. Any of these factors may be included in the passwords.

An interactive interface asks for the information of user such as :

1. Name
2. Date of birth
3. Identity number
4. Email
5. Phone number

S.No	Password	Length	Score	Strength	Relation
1.	prudhvi	7	2	Very Weak	No Match
2.	John Doe@	9	6	Weak	The password matches the name.
3.	19900121_Kiran	14	19	Moderate	The password matches the birth date.
4.	Kiran#222065	12	20	Strong	The password matches the name.
5.	SterlingGmail.911	17	25	Very Strong	No Match
6.	kitty	5	2	Very Weak	No Match
7.	ebay44	6	4	Very Weak	No Match
8.	jelly22fish_S	13	19	Moderate	No Match
9.	!usher@911_B	12	20	Strong	No Match
10.	d3ltagamm@..M	13	21	Strong	No Match
11.	suryam@123	10	6	Weak	The password matches the name.
12.	suryoodayam	11	2	Very Weak	No Match
13.	K_surya9	8	6	Weak	The password matches the name.
14.	anurag@2020	11	6	Weak	No Match
15.	AU@20_CSE	9	6	Weak	No Match

Table 6.2: Experiment cases

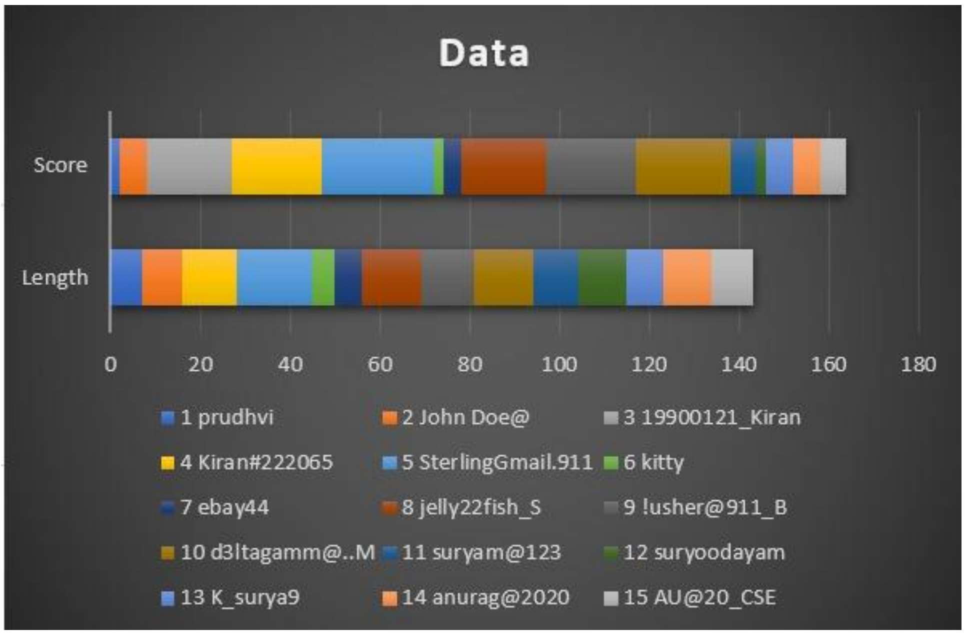


Fig.6.1: Complete overview

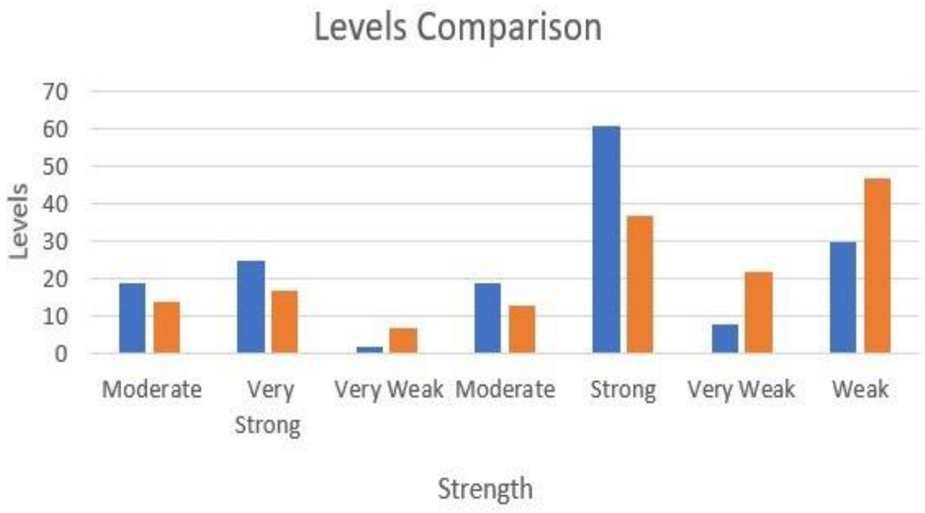


Fig.6.2: Levels Comparison

7. CONCLUSION

This password strength evaluation system provides a basic analysis of passwords, and it explores the behavior of personal information in passwords. It gives the strength of the password chosen and suggests making changes to make the password strong enough. It improves the understanding level and saves time for the user. The combined algorithm of Scoring and User Feedback Algorithms give the user attributes like, score, password level strength and suggestions to improve.

No other evaluation system considers users' personal information, but this system considers and encourages users to include personal information in their password so that they can remember them easily. It is a proper way with is beneficial for the user as they are assured of security. The system helps the user to make their password complex crack from any third party.

The user will be satisfied as he/ she can now use their personal information in a secure way.

8. REFERENCES

[1]Xinchun Cui, Xueqing Li, Yiming Qin, Ding Yong.
A Password Strength Evaluation Algorithm based on Sensitive Personal Information,2020.

[2]Mariam.M. Taha, Taqwa.A.Alhaj, Ala.E.Moktar, Azza.H.Salim, Settana.M.Abdullah.

On Password Strength Measurements: Password Entropy and Password Quality.

[3] Sivapriya K, Deepthi L.R.

Password Strength Analyzer Using Segmentation Algorithms.

[4] Vijaya MS, Jamuna KS, Karpagavalli S.

Password Strength Prediction Using Supervised Machine Learning Techniques.

[5] Bonneau J.

The Science of Guessing: Analyzing an Anonymized Corpus of 70 million Passwords[C]. Security & Privacy. IEEE, 2012.

[6] Kong Xiangqian, Shao Wenwu.

Methods and Application of Password Strength Met[J]. China Computer & Communication, 2019

[7] Chen Ying, Hong Di, Yanan Liu.

Password Strength Evaluation Method Based on Probabilistic Context-Free Grammar[J]. Internet of Things Technologies, 2017