Information Security Lab Project Report

Submitted in Partial Fulfilment of requirements for the Award of

Degree of Bachelor of Technology in Computer Science and Engineering

Submitted by

Manan Gupta (21BCP328)

Dhvani Chokshi (21BCP329)

Dwija Panchal (21BCP333)

Purva Patel (21BCP348)

Submitted To

Dr. Rutvij Jhaveri

Department of Computer Science and Engineering

School of Technology

Pandit Deendayal Energy University, Gandhinagar

September 2023

TABLE OF CONTENTS

Information Security LAB

Password Strength Checker

Introduction :

In an era of increasing cyber threats, password security is a critical component of safeguarding digital assets and sensitive information. Weak passwords can lead to unauthorized access, data breaches, and security vulnerabilities. The project aims to address this issue by developing a responsible and ethical tool for assessing and improving password security.

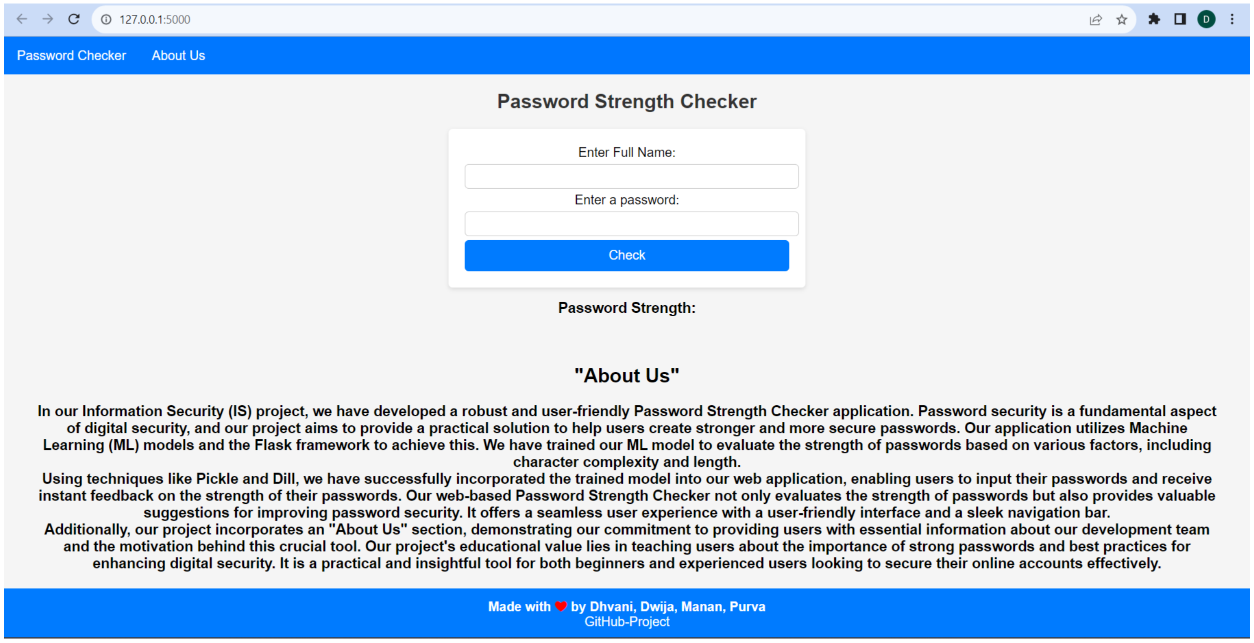
The Password Strength Checker project aims to develop a robust and secure tool for assessing and enhancing the password security of digital assets. In an era of increasing cyber threats, protecting sensitive data and accounts is of paramount importance. This project addresses the critical need for evaluating and strengthening password security. The project scope includes the design and implementation of a password cracking tool capable of testing the strength of passwords in a controlled environment. It does not involve any malicious or unauthorized activities. The project will follow a responsible and ethical approach, adhering to best practices in information security and ethical hacking. The tool will only be used for legitimate security testing and not for unauthorized purposes. This project addresses a critical aspect of information security by developing a responsible and ethical tool for assessing and improving password security. It aligns with the growing need to protect digital assets in an increasingly connected and vulnerable world.

Problem Statement :

Development of a Responsible Password Strength Checker and Optimizer.

The objective of this project is to design and implement a Password Strength Checker and Optimizer tool that can evaluate the strength of passwords used to protect digital assets. The tool will operate within the boundaries of responsible and ethical hacking principles, ensuring that all assessments are conducted with proper authorization and consent.

Project In Detail:



Password Strength Checker

Password Strength:

"About Us"

In our Information Security (IS) project, we have developed a robust and user-friendly Password Strength Checker application. Password security is a fundamental aspect of digital security, and our project aims to provide a practical solution to help users create stronger and more secure passwords. Our application utilizes Machine Learning (ML) models and the Flask framework to achieve this. We have trained our ML model to evaluate the strength of passwords based on various factors, including character complexity and length.
Using techniques like Pickle and Dill, we have successfully incorporated the trained model into our web application, enabling users to input their passwords and receive instant feedback on the strength of their passwords. Our web-based Password Strength Checker not only evaluates the strength of passwords but also provides valuable suggestions for improving password security. It offers a seamless user experience with a user-friendly interface and a sleek navigation bar.
Additionally, our project incorporates an "About Us" section, demonstrating our commitment to providing users with essential information about our development team and the motivation behind this crucial tool. Our project's educational value lies in teaching users about the importance of strong passwords and best practices for enhancing digital security. It is a practical and insightful tool for both beginners and experienced users looking to secure their online accounts effectively.

Made with ❤ by Dhvani, Dwija, Manan, Purva
GitHub-Project

Fig:01 Password strength checker website's homepage

Password strength checker and optimizer involves assessing the strength of a password based on various criteria and suggesting improvements.

Passwords are a common method of authentication. In cybersecurity, authentication ensures that users are who they claim to be. Weak passwords can compromise the effectiveness of authentication, making it easier for unauthorized individuals to gain access.

Strong passwords play a crucial role in controlling access to systems, networks, and data. The project addresses the importance of access control by helping users and organizations implement more secure password policies.

Vulnerability Assessment: The project conducts vulnerability assessments related to passwords. It identifies weaknesses in password practices and offers recommendations for improvement. This aligns with the broader concept of vulnerability assessment in cybersecurity, where vulnerabilities are identified and mitigated to enhance overall security.

The project begins by defining its goals and objectives, emphasizing the responsible and ethical assessment of password security. It establishes the scope, which includes designing and implementing a password cracking tool for controlled security testing.

The project team developed a password cracking tool, which is capable of testing the strength of passwords in a controlled environment. The tool has features that allow it to evaluate passwords based on various criteria, including length, character types, and complexity.

With authorization in place, the tool is used to assess the strength of passwords. This assessment includes checking for various factors such as:

Length: Ensuring passwords meet a minimum length requirement.

Complexity: Evaluating the presence of digits, uppercase and lowercase letters, and special characters.

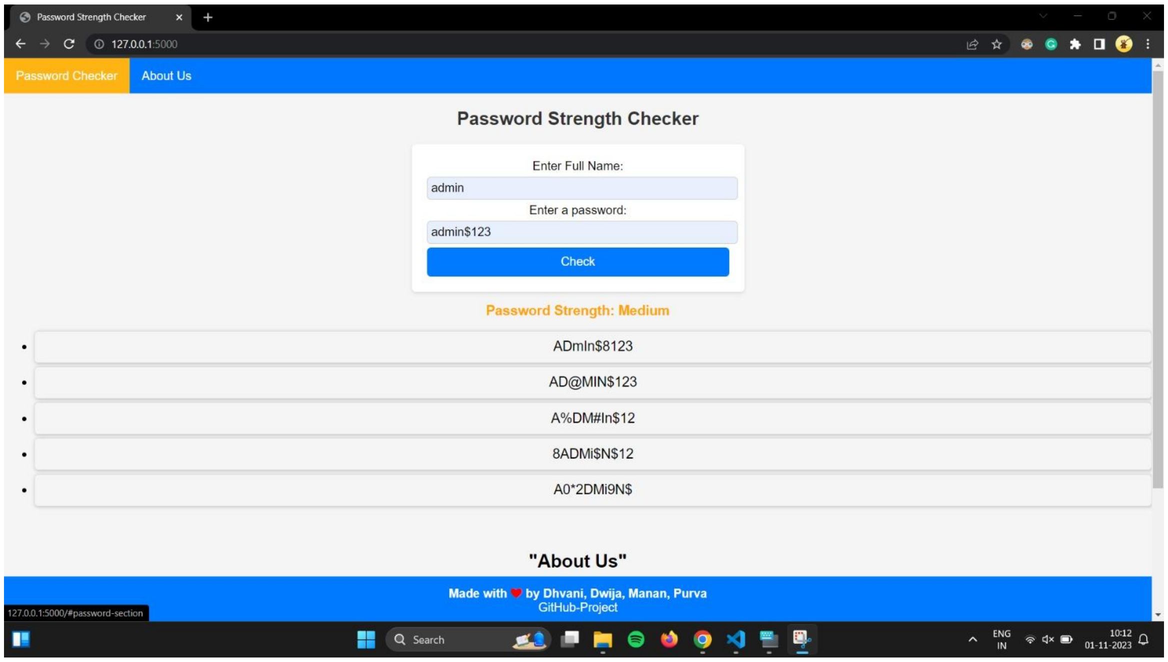
Resistance to dictionary attacks: Checking if passwords are susceptible to common word-based attacks.

Brute-force resistance: Assessing whether passwords can withstand automated guessing attempts.

The tool provides assessment results, indicating the strength of each password tested. For weak passwords, the tool offers recommendations for improvement. These recommendations may include suggestions to use longer passwords, add special characters, or avoid easily guessable patterns.

Working Of Our Password Strength Checker and optimizer tool:

- Password entered by user is of medium strength. So our tool suggest some strong password for the user.



Password Strength Checker

Enter Full Name: admin

Enter a password: admin$123

Password Strength: Medium

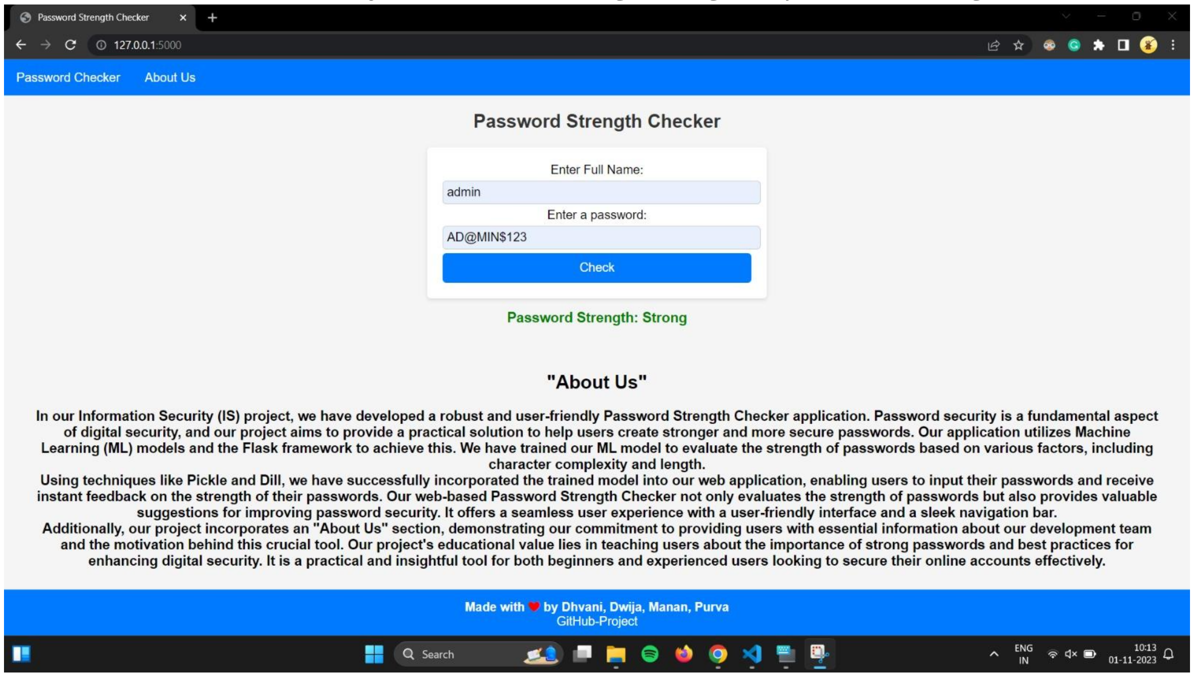
- ADmIn$8123
- AD@MIN$123
- A%DM#In$12
- 8ADMiN12
- A0*2DMi9N$

"About Us"

Made with ❤ by Dhvani, Dwija, Manan, Purva
GitHub-Project

Fig:02 When the password is of medium strength

- Password entered by the user is strong enough to process for login.



Password Strength Checker

Enter Full Name: admin

Enter a password: AD@MIN$123

Password Strength: Strong

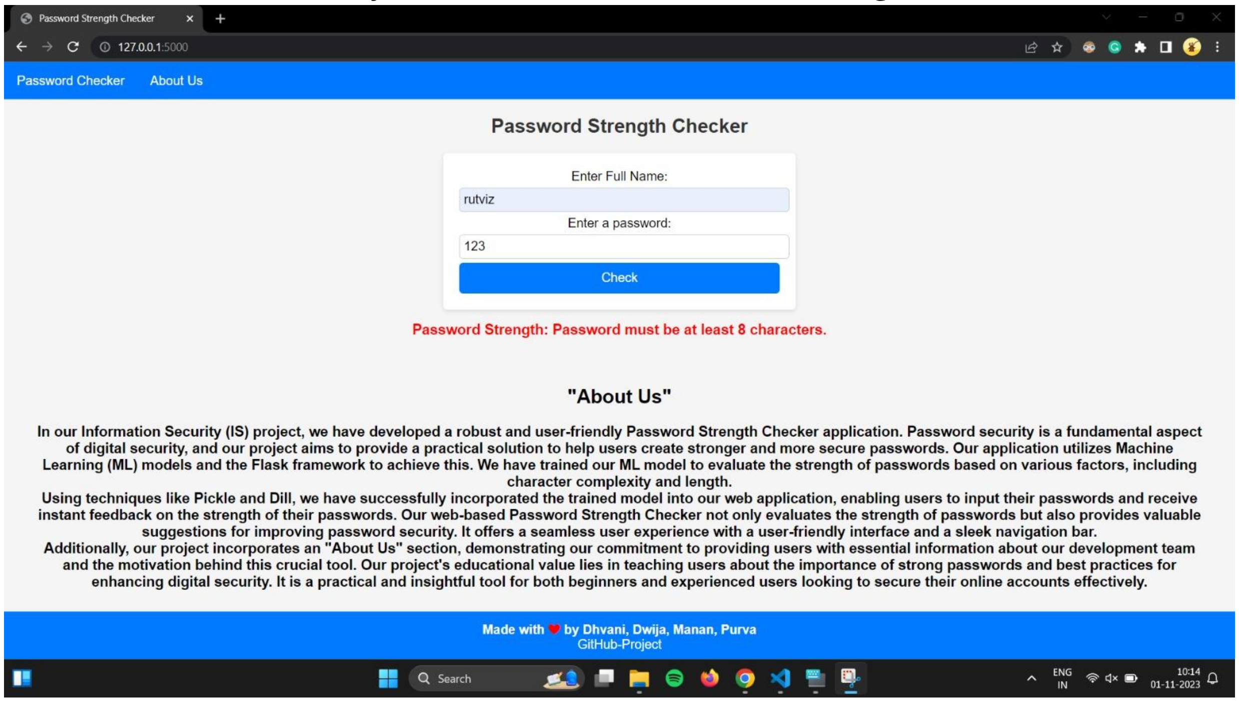
"About Us"

In our Information Security (IS) project, we have developed a robust and user-friendly Password Strength Checker application. Password security is a fundamental aspect of digital security, and our project aims to provide a practical solution to help users create stronger and more secure passwords. Our application utilizes Machine Learning (ML) models and the Flask framework to achieve this. We have trained our ML model to evaluate the strength of passwords based on various factors, including character complexity and length.
Using techniques like Pickle and Dill, we have successfully incorporated the trained model into our web application, enabling users to input their passwords and receive instant feedback on the strength of their passwords. Our web-based Password Strength Checker not only evaluates the strength of passwords but also provides valuable suggestions for improving password security. It offers a seamless user experience with a user-friendly interface and a sleek navigation bar.
Additionally, our project incorporates an "About Us" section, demonstrating our commitment to providing users with essential information about our development team and the motivation behind this crucial tool. Our project's educational value lies in teaching users about the importance of strong passwords and best practices for enhancing digital security. It is a practical and insightful tool for both beginners and experienced users looking to secure their online accounts effectively.

Made with ❤ by Dhvani, Dwija, Manan, Purva
GitHub-Project

Fig:03 password entered by user is strong

- Password entered by the user should be of minimum length of 8 characters.



Password Strength Checker

Enter Full Name: rutviz

Enter a password: 123

Check

Password Strength: Password must be at least 8 characters.

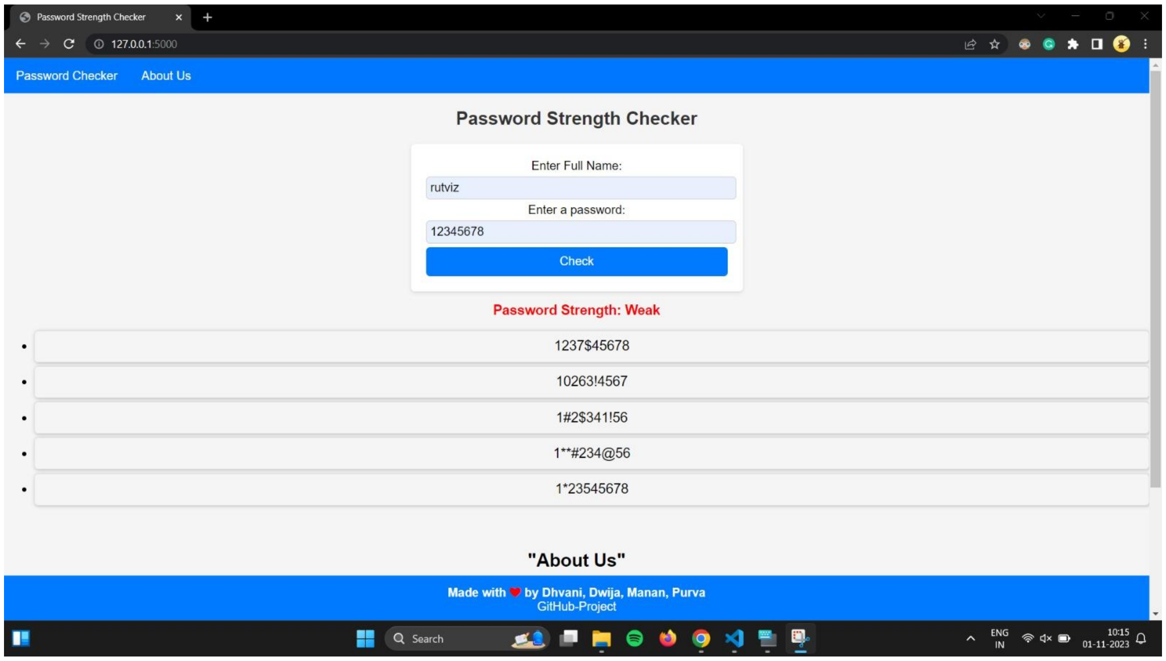
"About Us"

In our Information Security (IS) project, we have developed a robust and user-friendly Password Strength Checker application. Password security is a fundamental aspect of digital security, and our project aims to provide a practical solution to help users create stronger and more secure passwords. Our application utilizes Machine Learning (ML) models and the Flask framework to achieve this. We have trained our ML model to evaluate the strength of passwords based on various factors, including character complexity and length.
Using techniques like Pickle and Dill, we have successfully incorporated the trained model into our web application, enabling users to input their passwords and receive instant feedback on the strength of their passwords. Our web-based Password Strength Checker not only evaluates the strength of passwords but also provides valuable suggestions for improving password security. It offers a seamless user experience with a user-friendly interface and a sleek navigation bar.
Additionally, our project incorporates an "About Us" section, demonstrating our commitment to providing users with essential information about our development team and the motivation behind this crucial tool. Our project's educational value lies in teaching users about the importance of strong passwords and best practices for enhancing digital security. It is a practical and insightful tool for both beginners and experienced users looking to secure their online accounts effectively.

Made with ❤ by Dhvani, Dwija, Manan, Purva
GitHub-Project

Fig:04 Password should be of minimum 8 characters

- Password entered by user is too weak . So the tool suggest stronger passwords using random module of python.



Password Strength Checker

Enter Full Name: rutviz

Enter a password: 12345678

Check

Password Strength: Weak

- 1237$45678
- 10263!4567
- 1#2$341!56
- 1**#234@56
- 1*23545678

"About Us"

Made with ❤ by Dhvani, Dwija, Manan, Purva
GitHub-Project

Fig:05 Password is too weak

Technologies Used:

1. **Python:** The code is written in Python, a versatile and widely used programming language known for its simplicity and readability.
2. **Pandas:** Pandas is a popular data manipulation library in Python. You use it to read and manipulate the dataset, perform data cleaning, and conduct exploratory data analysis (EDA).
3. **NumPy:** NumPy is used for numerical operations and array manipulations. It's often used in conjunction with Pandas for data analysis.
4. **Seaborn and Matplotlib:** These are Python libraries for data visualization. You use Seaborn and Matplotlib to create various plots and visualizations, such as count plots for password strength distribution.
5. **Scikit-Learn (sklearn):** Scikit-Learn is a comprehensive library for machine learning and data analysis. In your code, you use Scikit-Learn for various tasks, including feature extraction and model training.
6. **XGBoost:** XGBoost is an optimized and efficient gradient boosting library. You use it to build a machine learning model for password strength prediction.
7. **Dill:** Dill is a Python library for serializing and deserializing Python objects. You use it to save the trained XGBoost model and the TfidfVectorizer object to disk for later use.
8. **TfidfVectorizer:** TfidfVectorizer is part of Scikit-Learn and is used to convert text data into numerical features based on the Term Frequency-Inverse Document Frequency (TF-IDF) method.
9. **Flask :** For backend side Python’s Flask library has been used.
10. **HTML,CSS:** For frontend HTML and CSS have been used.

Model Description:

The most important factor in analyzing the strength of the passwords using machine learning is collecting a meaningful dataset that contains a huge amount of passwords with strength as weak, medium, and strong. The dataset contains weak, medium, and strong passwords with labelling as 0, 1, and 2, respectively.

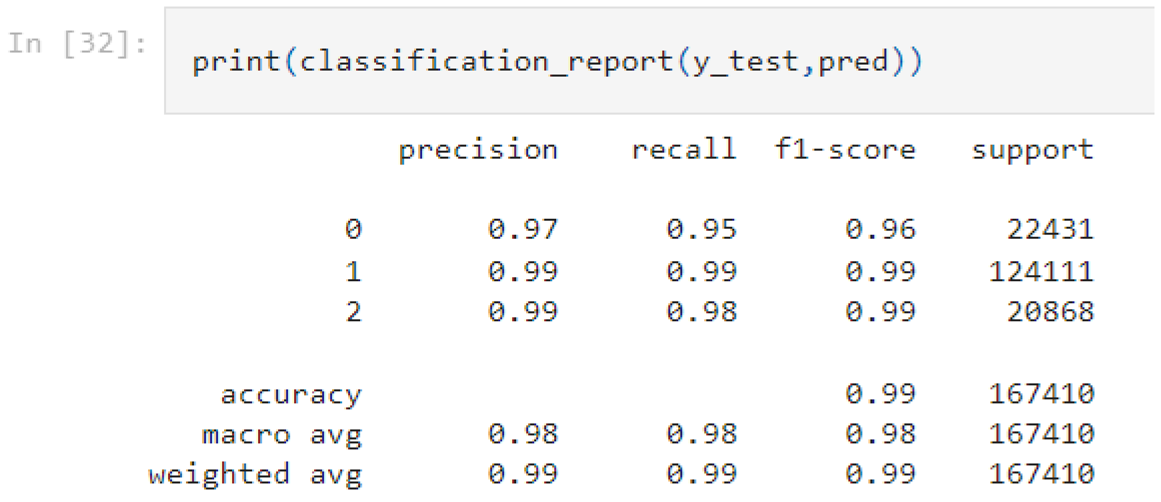
Here's the code's workflow according to the project:

- Data Loading and Preprocessing:The code starts by loading a dataset from a CSV file ('data.csv') that contains password data. It then performs data cleaning, handling any bad lines and missing values in the dataset.
- Data Exploration:The code explores the dataset by displaying the first few rows, the last few rows, the shape (number of rows and columns), and a count plot of the password strengths to understand the data distribution.
- Feature Extraction:The feature extraction process involves converting the password strings into numerical features. This is done using the Term Frequency-Inverse Document Frequency (TF-IDF) method. The TfidfVectorizer from Scikit-Learn is used to transform the password strings into numerical vectors.
- Training-Testing Split:The dataset is split into training and testing sets using the train_test_split function from Scikit-Learn. This allows for model training and evaluation on different subsets of the data.
- Model Selection and Training:The XGBoost algorithm is selected as the model for password strength prediction. XGBoost is an efficient and optimized gradient boosting algorithm commonly used for classification tasks. An instance of the XGBoost classifier (xgb.XGBClassifier) is created, and it is trained on the training data (X_train and y_train) using the .fit() method.
- Model Evaluation: The trained XGBoost model is used to make predictions on the testing data (X_test) using the .predict() method. Performance metrics for classification, such as confusion matrix and classification report, are computed and printed using Scikit-Learn's functions. These metrics provide insights into how well the model is performing in predicting password strengths.

- Model Serialization: The trained XGBoost model and the TfidfVectorizer are serialized and saved to files using the Dill library. This allows you to store the trained model and vectorizer for later use without the need to retrain the model.
- Predicting Password Strength: A sample password (temp_pswd) is provided, and the TfidfVectorizer is used to transform it into a numerical vector. The XGBoost model is then used to predict the password strength of the sample password.
- Model Interpretation: The code calculates and displays the predicted class label for the sample password (xgb_classifier.predict(temp_pswd)) and the predicted class probabilities (xgb_classifier.predict_proba(temp_pswd)). This provides information about the model's confidence in its predictions.
- In summary, the XGBoost model is trained to predict the strength of passwords based on their character compositions and other features extracted using TF-IDF. It is evaluated using performance metrics, serialized for future use, and used to predict the strength of a sample password. This machine learning model is designed to assess and classify the strength of passwords, which is essential for password security in various applications.

To summarize:

- Data is loaded from a CSV file ('data.csv') containing password data which has nearly 60,00,000 passwords and their strength.
- Data preprocessing includes handling missing values and visualizing the distribution of password strengths.
- The dataset is split into features (X) and labels (Y), where X represents the password data, and Y represents the corresponding strength labels.
- The password data is transformed using TfidfVectorizer to create numerical features based on character frequency.
- XGBoost is used to train a machine learning model for password strength prediction.
- The trained model and TfidfVectorizer are saved to disk using Dill for future use.



```
In [32]: print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.95      0.96     22431
           1       0.99      0.99      0.99    124111
           2       0.99      0.98      0.99     20868

    accuracy                           0.99    167410
   macro avg       0.98      0.98      0.98    167410
weighted avg       0.99      0.99      0.99    167410
```

Fig:06 Output of model

Overall, this code represents a typical workflow for building and training a machine learning model to predict password strength, using Python libraries and machine learning techniques. It aligns with the project's goal of assessing and improving password security in a controlled and responsible manner.

Time Complexity Analysis:

Data Loading and Preprocessing:

Computational Complexity: Loading data from a CSV file and performing basic preprocessing, such as handling missing values, has a linear computational complexity, typically **O(n)**, where 'n' is the number of data records.

Data Exploration:

Computational Complexity: Simple data exploration tasks, such as displaying the first/last few rows, checking the shape, and creating basic visualizations, have relatively low computational complexity.

Feature Extraction (TF-IDF):

Computational Complexity: Calculating the TF-IDF features for passwords involves tokenizing each password and creating a sparse matrix. The computational complexity is influenced by the number of passwords and the average password length. It can be roughly approximated as **O(n * m),** where 'n' is the number of passwords and 'm' is the average password length.

Training-Testing Split:

Computational Complexity: Splitting the dataset into training and testing sets using train_test_split has a linear computational complexity, typically **O(n)**, where 'n' is the size of the dataset.

Model Training (XGBoost):

Computational Complexity: Training the XGBoost model has a complexity that depends on the number of trees (iterations) and the depth of each tree. The complexity is typically

O(d * n * T), where 'd' is the maximum depth of a tree, 'n' is the number of samples, and 'T' is the number of trees.

Model Evaluation:

Computational Complexity: Evaluating the model using performance metrics like a confusion matrix and classification report has a linear computational complexity, typically **O(n)**, where 'n' is the number of predictions.

Model Serialization:

Computational Complexity: Serializing and saving the model and vectorizer is typically a fast operation and does not significantly contribute to the overall complexity of the project.

Password Strength Prediction:

Computational Complexity: Predicting the strength of a password using the trained model involves transforming the password into a numerical vector using the TF-IDF vectorizer and making predictions with the XGBoost model. The computational complexity is typically

O(m + d), where 'm' is the average password length and 'd' is the depth of the XGBoost trees.

In terms of algorithmic complexity, the main factors affecting this project's performance are the size of the dataset, the complexity of the machine learning model (XGBoost), and the dimensionality of the TF-IDF feature vectors.

Future Scope :

The Password Strength Checker and Optimizer project has a promising future scope in the field of information security due to the ever-evolving landscape of cybersecurity threats and the ongoing need for stronger security measures. Here are some future prospects and potential areas of growth for this project:

1. Integration with MFA technologies to provide a comprehensive security solution that combines strong passwords with additional authentication factors.
2. Developing real-time password assessment capabilities that can assess passwords as they are created or changed, providing immediate feedback to users.
3. Expanding the project's capabilities to assist organizations in conducting security audits and ensuring compliance with data protection regulations, such as GDPR and HIPAA.
4. Integrating with password management tools to provide a seamless user experience, including generating, storing, and assessing passwords securely.
5. Enhancing password recovery processes with secure and user-friendly methods, such as biometric authentication or trusted device verification.

Conclusion:

In this project work we made a password strength detection and analysis system by implementing multiple machine learning approaches on a web application. The main target of our system was mainly to force the users to choose strong passwords with complex patterns and without any known or common patterns in order to make it difficult for hackers and crackers to crack the passwords. The foremost step that we took to deal with this issue was getting a dataset that contains numerous passwords with different patterns. The machine learning models that we implemented performed well whereas the best results were evaluated by XGboost with an accuracy of 99%. Thus user can check his password strength and if it is weak then our tool suggests a stronger password.

Link to our project:

https://github.com/manan152003/info-sec_project

References:

- https://www.ijert.org/research/real-time-password-strength-analysis-on-a-web-application-using-multiple-machine-learning-approaches-IJERTV9IS120146.pdf
- https://thecleverprogrammer.com/2022/08/22/password-strength-checker-with-machine-learning/