Password Strength Prediction using GBM, KNN & SVM Algorithms

Yogesh Anandhakumar
MSc Cyber Security
National College of Ireland
Dublin, Ireland
x23167998@sudent.ncirl.ie

***Abstract*— Password is a secret or personal key, which helps to ensure that only authorized users are allowed to access our data, systems, etc. The main objective of this research is to predict and strengthen passwords by using three machine learning algorithms such as Gradient Boosting Machines (GBM), K-Nearest Neighbors (KNN), and Support Vector Machines (SVM). This is done by using a dataset from Kaggle that includes a variety of passwords labeled with the following attributes: "Has Lowercase, Has Uppercase, Has Special Character, Length, Strength". First, the model has been trained and tested with the above-mentioned training dataset. Then the model was tested once again with a secondary dataset, which has password and strength only. I believe that this research will contribute to the advancement of the field of cybersecurity by providing valuable insights on how to strengthen passwords and improve authentication techniques.**

Keywords—Password Strength, Security, Machine Learning, GBM, KNN & SVM.

I. INTRODUCTION

Passwords are the primary means of user authentication, fund transfers, social media platforms, protecting sensitive data and critical systems. Because of its easy implementation, attackers will try to get access to data or accounts which are private for someone. Many reports says that the weak passwords or usual password patterns like Date of Birth, name, mobile numbers, etc., are being cracked easily. However, there are a lot of issues facing in password strength tests nowadays. So this study will help users to predict their password strength and secure their data and accounts from attackers [1].

A. Motivation

Conventional techniques, which depends on elementary principles such as minimum length and character kinds, are readily evaded by skilled adversaries armed with sophisticated instruments and extensive password dictionaries. Modern password cracking techniques like dictionary attacks and brute-force methods are complicated, and rule based solutions finds it difficult to keep up. Since hacking techniques and user behavior patterns are always evolving, current systems are not flexible enough to keep up. Rejecting weak passwords without offering concrete suggestions for improvement, is bad for both user comprehension and password hygiene. This research explores various machine learning algorithm's potential for identifying & improving password strength. Our goal in investigating this research issue is to understand more about the actual capacity of machine learning algorithms to safeguard user accounts and information. This study will also examine how machine learning algorithms are capable of surpassing conventional techniques in terms of accuracy. My personal goal is to fully utilize machine learning for evaluating password strength, so that we all can use the internet in a safer manner.

B. Objectives

The main objective of this research is to predict the strength of passwords using three machine learning algorithms namely – Gradient Boosting Machines (GBM), K-Nearest Neighbors (KNN), and Support Vector Machines (SVM). This research contributes to strengthen the passwords by:

- Using a dataset from Kaggle which has passwords, and supporting labels like strength, length, has lowercase, has numeric & has uppercase.
- Train the three models – GBM, KNN & SVM.
- Evaluate the performance of the three models
- Again test the trained model with a secondary dataset from Kaggle.
- Evaluate each model's overall performance and determine which is the best.

II. RESEARCH QUESTION

“ How accurate are the three machine learning models, Gradient Boosting Machines (GBM), K-Nearest Neighbors (KNN), and Support Vector Machines (SVM), in predicting the password strength? ”

III. RELATED WORK

A. Password Strength Prediction using Supervised Machine Learning Techniques

In this paper, the authors used several machine learning techniques such as Supervised Vector Machines (SVM), Decision Tree Classifier (DT), Naïve Bayes Classifier (NB), and Multilayer perceptron (MLP), which is a positive aspect as these methods shows effectiveness in classification tasks. They have created 10,000 random passwords using PC Tools password generator and they have selected more features for their model, nearly 27 features and they have password strength types as Very weak, weak, good, strong, and very strong, labelled as 1,2,3,4, & 5. Their approach of feature extraction helped in improving classification effectiveness and computational efficiency. They have used WEKA and SVM Ligh for the password strength analysis. The models accuracy were Naïve Bayes as 73.6, DT as 87.2, MLP as 89.8 and SVM as 98.3. Then they compared the their trained model with Google Password meter and Microsoft Password Checker to evaluate the performance of their trained model. The negative aspect in this paper was they used random 10,000 passwords for training the model, which may not be able to predict real-time passwords strengths accurately [2].

B. Password Strength Analysis and its Classification by Applying Machine Learning Based Techniques

In this paper, the authors used several machine learning techniques such as Logistic regression (LR), Decision tree

(DT), Random Forest Classifier (RF), Naïve Bayes (NB), XG Boost (XGB), Support Vector Machine (SVM), and Multilayer perceptron (MLP), which is a positive aspect as many variety of models were employed. They used a Kaggle dataset which has 80,000 random passwords, where 12.35% are strong, 74.29% are medium, 13.36% are weak in total. They have also used TF-IDF vectorizer for feature extraction, which is an another one positive aspect, which can increase the accuracy of the classification models. Then they trained the models and got the outputs of accuracy, LR as 81.79, DT as 84.48, RF as 85.11, NB as 74.29, XGB as 99.70, SVM as 81.10, and MLP as 95.18. The negative aspect in this study is achieving the efficient and timely output by applying these models on the real-world passwords might be a key challenge [2].

C. *Performance Analysis of Machine Learning Algorithms for Password Strength Check*

In this paper, the authors used several machine learning techniques such as Decision Tree (DT), Logistic Regression (LR), Random Forest Classifier (RF) and K-Nearest Neighbors (KNN), which is a positive aspect as many variety of models were employed. They used a Kaggle dataset which has 669640 passwords, which has only two columns, Password & Strength. They trained the model and got the outputs as accuracy, RF as 98%, DT as 97%, LR as 82% and KNN as 83%. The negative aspect in this study is the selection of dataset, because it has only two labels, password and strength, which doesn't have any extra features [3].

D. *Enhancing Password Security via Supervised Learning*

In this paper, the authors used several machine learning techniques such as Logistic Regression (LR), Naïve Bayes (NB), Decision tree (DT), and Convolutional neural network (CNN), which is a positive aspect as many variety of models were used. The dataset has 670000 passwords, which they have collected from different websites via web scrapping. They trained the model and got the outputs as accuracy, LR as 70%, NB as 27%, DT as 40%, and CNN as 37%. The negative things in this study is, the accuracy of majority of the models was so low, so they could have employed few other classifier models also [4].

E. *Machine-Learning-Based Password-Strength-Estimation Approach for Passwords of Lithuanian Context*

In this paper, the authors proposed a machine learning approach to estimate the strength of Lithuanian user passwords, and the dataset of 110000 passwords, which was a combination of both international passwords (English) and Lithuanian language passwords. The use of Lithuanian passwords in dataset, which is a positive aspect as there are no cracking tools are available for Lithuanian passwords. This model is focused on non-English passwords strength prediction. They have obtained the output as accuracy, 76% as International passwords and 79% as Lithuanian passwords [5].

F. *Real Time Password Strength Analysis on a Web Application Using Multiple Machine Learning Approaches*

In this paper, the author analyzed the password strength in real time on a web application using multiple machine learning techniques, such as Decision Tree (DT), Naïve Bayes (NB), Linear Regression (LR), Random Forest (RF), and Neural Network (NN). They used a dataset of 700000 passwords, with three types of strength such as Strong, Medium, and Weak. Also he used TF-ID Vectorizer for extracting features of the dataset. Then he trained th e model with 70% of dataset and tested the model with the rest 30% of the dataset. Then he implemented the model in a web application, where webpage is created using HTML5 and CSS3, back-end functions with JavaScript and back-end DB with PHP. While the model gave output as accuracy, DT as 99%, NB as 87%, LR as 89%, RF as 95% and NN as 92%. Then after implementing the trained model in the web application, they created 250 users, with similar usernames and different pattern passwords. Then he performed brute force attacks, reverse brute force attacks, and dictionary attacks on the website using Burp Suite. The negative aspect is, even after implementing the model, the attacks can crack strong passwords. So he suggested to add systems like MFA, OTP verifications, etc. Also he concluded as the model can be used for forcing users to strengthen passwords, if they are using less secure passwords [6].

G. *Password Strength: An Empirical Analysis*

In this paper, the authors gave a comprehensive analysis on password strength using three different datasets, Italian (IT), Finnish (FI), and MySpace (MS) datasets. Then he performed password guessing, Dictionary attack, Mangling using dictionaries and probabilistic context free grammars, and Markov Chain-based attack for measuring the resilience of passwords. The use of different datasets and the password resilience measuring methods are the positive aspect here, as they are different approach and very effective, which may help in real world passwords attacks. The negative aspect on this study was limited usage of attacks, which may not help in securing passwords over other attacks [7].

H. *Proactive Password Strength Analyzer Using Filters and Machine Learning Techniques*

In this study, the authors suggested a submodule for access control scheme, by proposing a framework to proactively analyze the strength of passwords, where filters and Support Vector Machines (SVM) are employed. The framework was having 3 filters, where the filter 1 is verifying the password is empty or same as username, filter 2 is verifying the password among common passwords list, and the filter 3 performs a dictionary attack to crack the password. Then after all filters, features are selected and tested with SVM model, which is already trained by a 10000 passwords of 8 character length. Finally, the model categorizes passwords as very weak, weak, good, strong, and very strong based on its strength. The proposed framework was a positive aspect, as it has multiple filters and a machine learning model to categorize the passwords strength. The negative aspect in this framework was utilizing limited passwords, if they have used more than 100000 passwords for training the model, so that might have helped in categorizing the passwords in real time also [8].

I. *Increase Security by Analyzing Password Strength using Machine Learning*

In this study, the authors proposed a framework for analyzing password strength. The framework has nine major steps, data collection, text processing, feature extraction, TF-IDF, data preparation, model building, model evaluation, model testing, if the model is okay then model description or again to the model building process. They collected 100000 plain text passwords, text processed using NLP, selected features, calculated the frequency of characters by TF-IDF scores, preprocessed the data, trained the model (logistic

regression with multi-class classification), evaluated the model, and finally tested the model. The positive aspect in this study was the model was trained and tested with a good number of passwords, which may really help in the real-world access control mechanisms [9].

IV. Choice of Methods

I have selected Gradient Boosting Machines (GBM), K-Nearest Neighbors (KNN), and Support Vector Machines (SVM) algorithms in this research for predicting password strength. Each of these algorithms have different unique strengths and can be used very effectively based on our dataset and requirements.

A. Gradient Boosting Machine (GBM):

This algorithm is very effective in tasks like predicting, where it can handle data's like numeric and category, very well. This is widely used for classification and regression tasks. This model works good for scenarios like handling various types of data, complex non-linear relationships, high dimensional, classification, regression, imbalanced data, and feature handling. But this is very prone to over fitting and improper tuning. Also boosting may result in overfitting due to noise, but it has mechanisms like subsampling and tree constraints to solve the issue and make the model good for various data types.

B. K-Nearest Neighbors (KNN)

This algorithm is easy to use and understand, which does not make assumptions based on the data, and also works very effectively for real world data. This is helpful in situation where decision boundaries are not regular, for both classification tasks and regression tasks. We can fine-tune the model by modifying the distance metric as we require based on the data. It also works very efficiently for smaller datasets, because it can predict using less training data's [3].

C. Support Vector Machines Classifiers (SVM)

This algorithm is very effective and accurate for high-dimensional features data's, where each character will be treated as separate feature. The kernel trick allows us to avoid computationally expensive operations by finding an appropriate kernel function which is equal to the high dimension space. By including kernel function, we can turn SVM into a powerful tool for classification or regression problems. SVM gives good generalization performance with different kernel functions to fit the data based on the type of data [8].

Combining the strengths of these three algorithms could actually cover a pretty broad spectrum of data behaviors and patterns, resulting in a robust solution. For example, SVM can handle high-dimensional data, where KNN can adjust to change very fast, without the necessity of retraining being done. The overfitting problem can be eliminated by using cross-validation during the training phase. To handle high memory, the algorithm in KNN can be reduced in dimensionality using techniques like Principal Component Analysis (PCA). Every algorithm has few set of hyperparameters to fine-tune, and grid search or even randomized search, which can result in high efficiency or accuracy rate.

V. Methodology and Implementation

Below is the architecture of the model we used in this research,

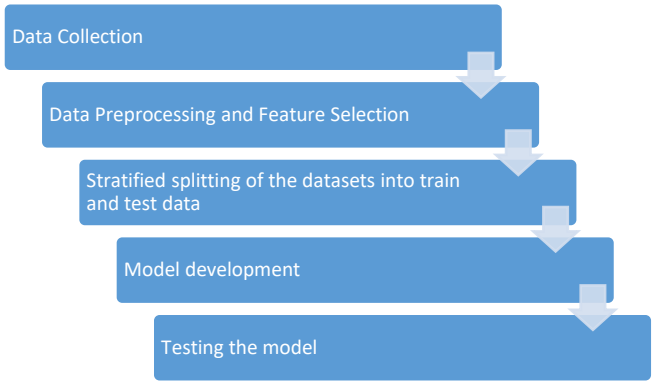

Fig. 1. Model Architecture

Here we have used a two datasets in this research. First is the training dataset from Kaggle, which has 10,000 passwords and columns such as Password, Has Lowercase, Has Uppercase, Has Special Character, Length, and Strength. Second is the testing dataset from Kaggle, which has 6,69,879 passwords and columns such as password and strength. In this research we have made two experiments.

Exp 1 – data collection, preprocessing, feature selection, stratified splitting of dataset, building model, testing the model with train data, and again testing the trained model with a secondary dataset.

Exp 2 – data collection, preprocessing, balancing, feature selection, stratified splitting of dataset, building model, testing the model with train data, testing the model with training dataset fully, again testing the model with a secondary dataset, hyper parameter tuning of the model, and again testing the tuned model with a secondary dataset.

A. Experiment 1: Without balancing and Tuning.

a) Data Collection:

- Initially the Pandas and NumPy libraries are imported.
- Then the training dataset has been imported as **"df"**.

b) Data Preprocessing

- Then its shape, columns, types, descriptive statistics, null values, value counts has been checked.
- Then distribution graph for the dataset based on password strength has been checked by importing seaborn and matplotlib.pyplot libraries.

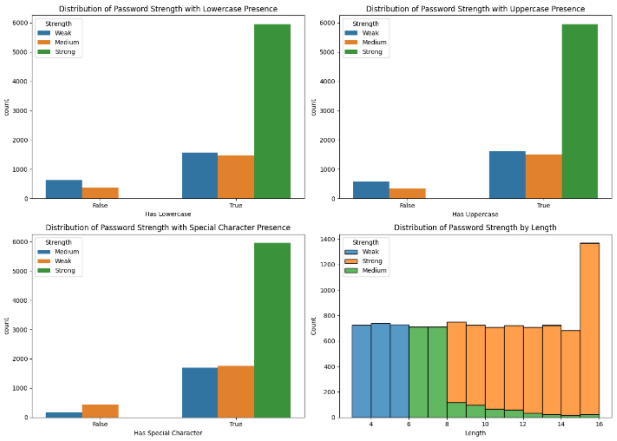

Fig. 2. Distribution graph

- Then feature selection has been made and also a correlation matrix has been created for better understanding of the features dependency in the dataset.

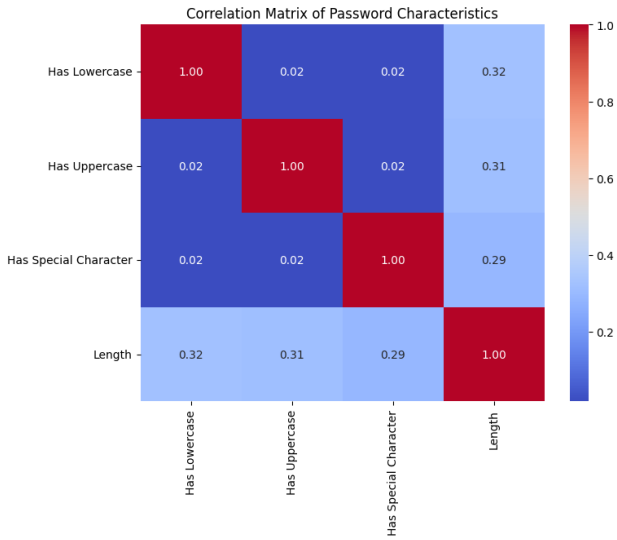

Fig. 3. Correlation matrix

- Then stratified splitting of the testing dataset has been made by importing train_test_split & LabelEncoder.

c) Modelling

- Then the three models has been initialized, modelled, trained, evaluated, and printed the result output by importing SVC, KneighbnorsClassifier, GradientBoostingClassifier, and classification report.

B. *Experiment 2: With balancing and Tuning*

a) Data Collection:

- Initially the Pandas and NumPy libraries are imported.
- Then the training dataset has been imported as **"df"**.

b) Data Balancing and Preprocessing

- Then the dataset has been equally balanced.
- Then its shape, columns, types, descriptive statistics, null values, value counts has been checked.
- Then distribution graph for the dataset based on password strength has been checked by importing seaborn and matplotlib.pyplot libraries.

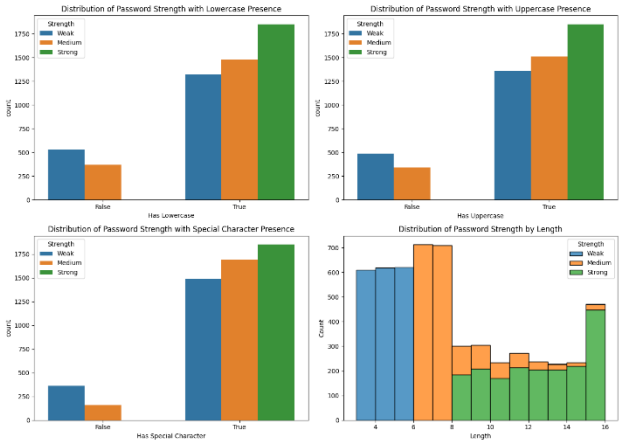

Fig. 4. Distribution graph

- Then feature selection has been made and also a correlation matrix has been created for better understanding of the features dependency in the dataset.

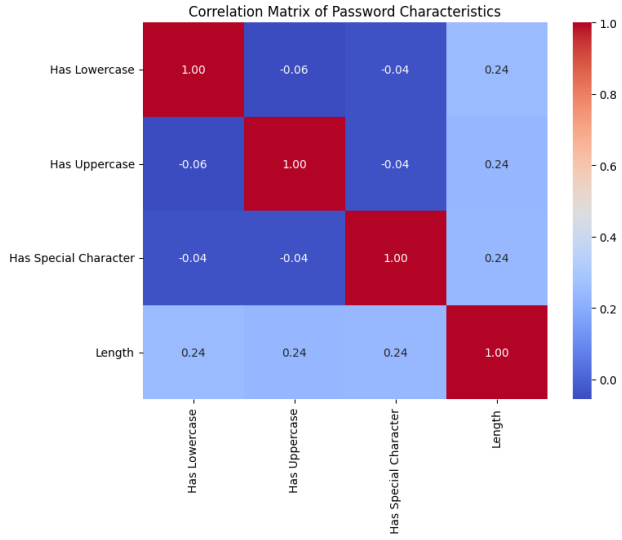

Fig. 5. Correlation matrix

- Then stratified splitting of the testing dataset has been made by importing train_test_split & LabelEncoder.

c) Modelling

- Then the three models has been initialized, modelled, trained, evaluated, and printed the result output by importing SVC, KneighbnorsClassifier, GradientBoostingClassifier, and classification report.

d) Testing the trained model performance with same training dataset

- Then imported the model and tested the model with the same training dataset fully. The result has been listed below with a confusion matrix for each model.

TABLE I. RESULT

Models	Accuracy
SVM	100%
KNN	100%
GBM	100%

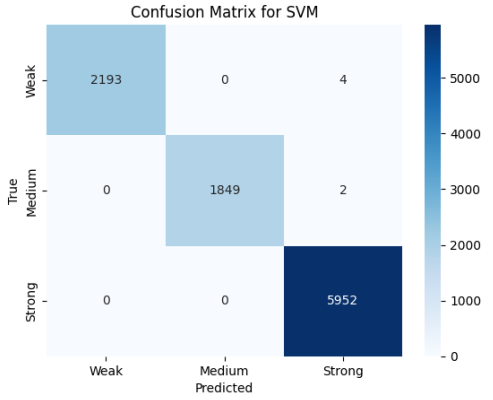

Fig. 6. Confusion matrix for SVM

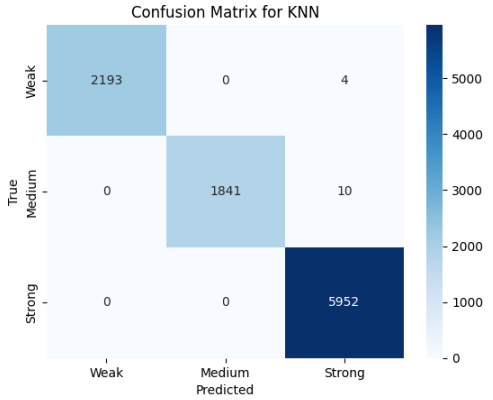

Fig. 7. Confusion matrix for KNN

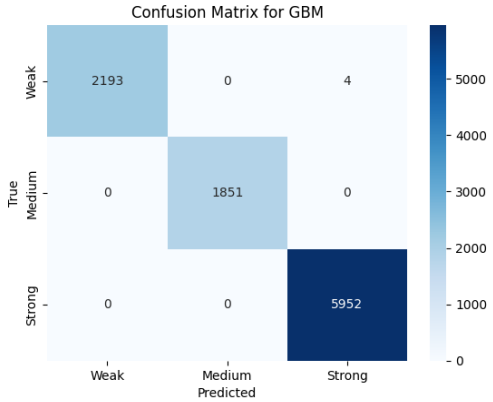

Fig. 8. Confusion matrix for GBM

e) Again testing the model with a secondary dataset

- Then to evaluate the trained model, a random 1,00,000 passwords of the secondary dataset has been tested and printed the result output with a confusion matrix for each model.

TABLE II. RESULT

Models	Accuracy
SVM	83%
KNN	84%
GBM	75%

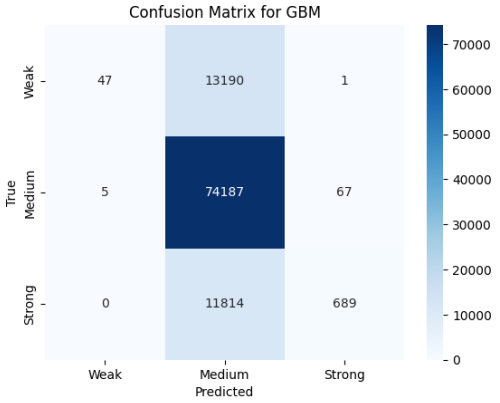

Fig. 9. Confusion matrix for SVM

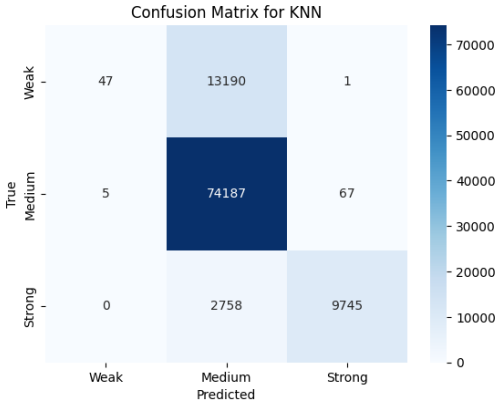

Fig. 10. Confusion matrix for KNN

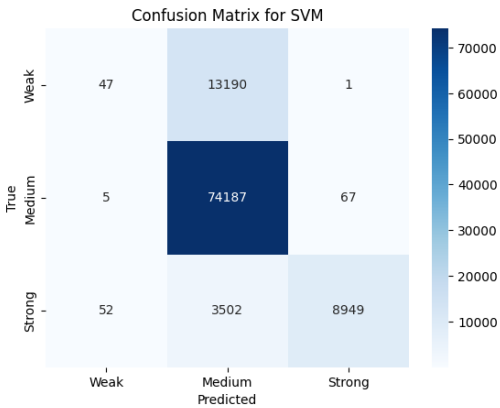

Fig. 11. Confusion matrix for GBM

f) Hyper-parameter tuning

- As the results are not satisfactory, hyper parameter tuning has been done. So, we are importing the GridSearchCV, KNeighborsClassifier, SVC, and GradientBoostingClassifier from sklearn, then defining the classifiers, setting the hyperparameter grids, initializing the grid search, fitting the models, printing the best parameters & scores, and accuracy of each model.

```
KNN - Best Parameters: {'algorithm': 'auto', 'n_neighbors': 3, 'weights': 'distance'}
KNN - Best Cross-Validation Score: 0.9986491553420688
SVM - Best Parameters: {'C': 2, 'gamma': 'scale', 'kernel': 'rbf'}
SVM - Best Cross-Validation Score: 0.9995498537024533
GBM - Best Parameters: {'learning_rate': 0.05, 'max_depth': 3, 'n_estimators': 50}
GBM - Best Cross-Validation Score: 0.9995500562429698
KNN - Test Accuracy: 1.0
SVM - Test Accuracy: 1.0
GBM - Test Accuracy: 1.0
```

Fig. 12. Hyper-parameter tuning result

VI. Evaluation

A. *Experiment 1: Without balancing and Tuning.*

a) Data evaluation:

- Then to evaluate the trained model, a random 1,00,000 passwords of the secondary dataset has been tested and printed the result output.

TABLE III. Result

Models	Accuracy
SVM	86%
KNN	86%
GBM	75%

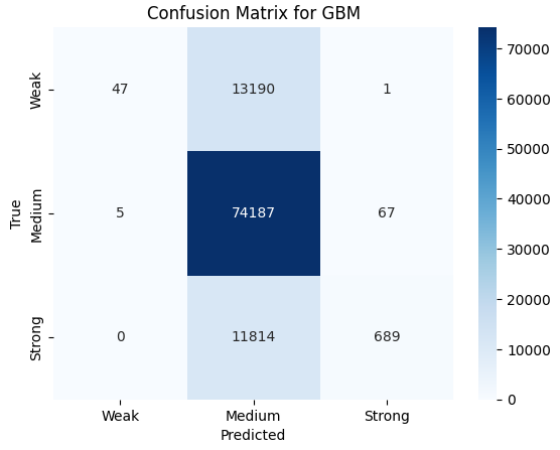

Fig. 13. Confusion matrix for SVM

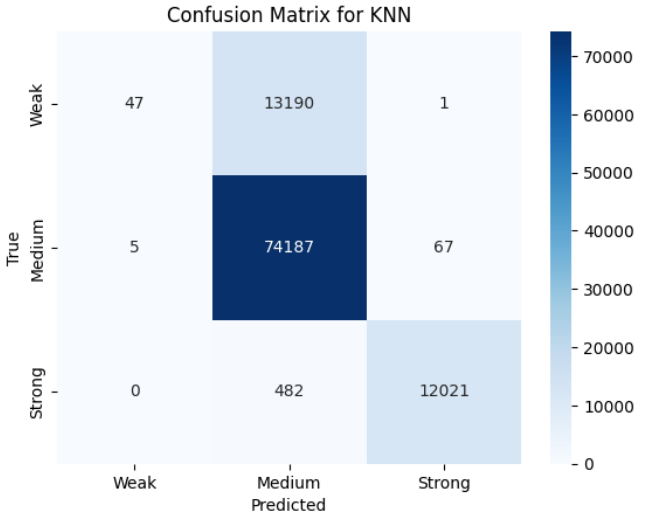

Fig. 14. Confusion matrix for KNN

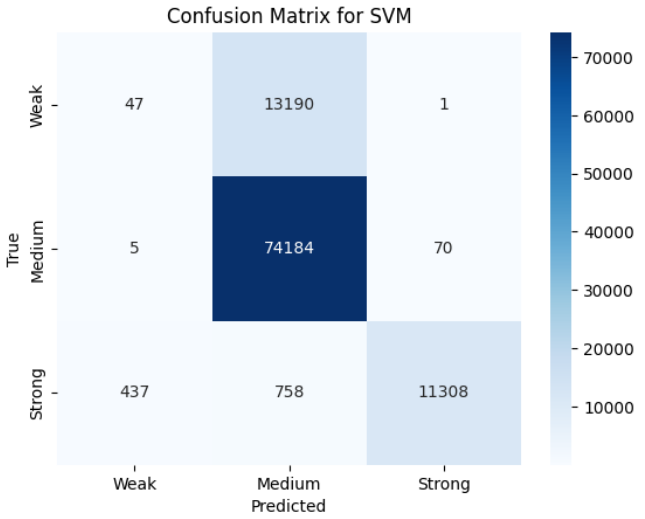

Fig. 15. Confusion matrix for GBM

B. *Experiment 2: With balancing and Tuning*

a) Data evaluation:

- Then to evaluate the tuned model, a random 1,00,000 passwords of the same secondary dataset has been tested and printed the result output.

TABLE IV. Result

Models	Accuracy
SVM	75%
KNN	75%
GBM	75%

VII. Conclusion and Futurework

Based on the above two experiment results, the "Experiment 1" has the better performance. Also all the three models are overfitted to the training dataset, so that's why the performance of models were 100% while testing the training dataset. The three models (SVM, KNN, GBM) results of testing the secondary dataset are,

TABLE V. RESULT

Models		Precision	Recall	F1-Score	Accuracy
SVM	Weak	84%	100%	91%	86%
	Medium	99%	90%	95%	
	Strong	10%	0%	1%	
KNN	Weak	84%	100%	92%	86%
	Medium	99%	96%	98%	
	Strong	90%	0%	1%	
GBM	Weak	75%	100%	86%	75%
	Medium	91%	6%	10%	
	Strong	90%	0%	1%	

Based on the above mentioned result, KNN model is the best performance model for predicting password strength, because of its high precision and recall for "Weak" and "Medium" categories. This shows its ability to accurately predict most weak and medium passwords, which will be more beneficial in maintaining a secure access control mechanisms. However, all the models are not able to predict strong passwords. If I had time, I would have made further tuning the model or adding more possible features in feature selection process, which might result in effectively predicting the strong passwords too. Also I would have handled the class imbalance by using SMOTE technique, where it will help in the models sensitivity to strong passwords.

REFERENCES

[1] PCMAG, 'Definition of password | PCMag', PCMAG. Accessed: May 04, 2024. [Online]. Available: https://www.pcmag.com/encyclopedia/term/password

[2] S. Sarkar and M. Nandan, 'Password Strength Analysis and its Classification by Applying Machine Learning Based Techniques', 2022 2nd International Conference on Computer Science, Engineering and Applications, ICCSEA 2022, 2022, doi: 10.1109/ICCSEA54677.2022.9936117.

[3] R. Divya, Gaganashree, S. B. Devamane, V. Dharshini, and S. Deepika, 'Performance Analysis of Machine Learning Algorithms for Password Strength Check', Proceedings - 2023 International Conference on Computational Intelligence for Information, Security and Communication Applications, CIISCA 2023, pp. 338–343, 2023, doi: 10.1109/CIISCA59740.2023.00071.

[4] Y. Alkurdi, M. Al Fayoumi, A. Al-Badarneh, and Q. A. Al-Haija, 'Enhancing Password Security via Supervised Learning', 2023 International Conference on Information Technology: Cybersecurity Challenges for Sustainable Cities, ICIT 2023 - Proceeding, pp. 256–259, 2023, doi: 10.1109/ICIT58056.2023.10226141.

[5] J. Machado et al., 'Machine-Learning-Based Password-Strength-Estimation Approach for Passwords of Lithuanian Context', Applied Sciences 2023, Vol. 13, Page 7811, vol. 13, no. 13, p. 7811, Jul. 2023, doi: 10.3390/APP13137811.

[6] U. Farooq, 'Real Time Password Strength Analysis on a Web Application Using Multiple Machine Learning Approaches', International Journal of Engineering Research & Technology (IJERT), 2020, Accessed: Mar. 01, 2024. [Online]. Available: www.ijert.org

[7] M. Dell'Amico, P. Michiardi, and Y. Roudier, 'Password strength: An empirical analysis', Proceedings - IEEE INFOCOM, 2010, doi: 10.1109/INFCOM.2010.5461951.

[8] Suganya G, Karpgavalli S, and Christina V, 'Proactive Password Strength Analyzer Using Filters and Machine Learning Techniques', Int J Comput Appl, vol. Volume 7, No.14, no. October 2010, pp. 0975–8887, 2010.

[9] A. Asaduzzaman, D. D'Souza, M. R. Uddin, and Y. Woldeyes, 'Increase Security by Analyzing Password Strength using Machine Learning', pp. 32–37, Apr. 2024, doi: 10.1109/ECTIDAMTNCON60518.2024.10479995.