

**Model for password security**

**Matteo Gianvenuti**

**Donato Bruno**

Summary

[Problem description 3](#_Toc174529064)

[Technologies used 3](#_Toc174529065)

[Dataset 3](#_Toc174529066)

[Stratification 3](#_Toc174529067)

[Features 4](#_Toc174529068)

[Accuracy 4](#_Toc174529069)

[Bibliography 5](#_Toc174529070)

# Problem description

Identify and create the best classifier model to determine the strength of a password. In practice, the model must establish which class the password belongs to with a good accuracy, choosing between five classes: "Too weak", "Weak", "Moderate", "Strong", "Very strong".

## Technologies used

We used the library scikit-learn which is an efficient tool for machine learning in Python. We tested different classifier models from the library to understand which is the most efficient to choose it.

* Gradient Boosting Classifier (ensemble): It is an additive model, which adds decision trees one at a time, trying to improve the model step by step. Based on negative gradient of the loss function.[[1]](#GradientBoostingClassifier) (An ensemble model combines predictions from multiple models to improve overall performance over that achieved with a single model).
* Random Forest Classifier (ensemble): It is based on a set of independent decision trees, trained on different subset of the training set. It uses averaging to improve the predictive accuracy and control over-fitting.[[2]](#RandomForestClassifier)
* K Neighbors Classifier (neighbors): A classifier implementing the k-nearest neighbors vote. The classification is computed from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point.[[3]](#KNeighborsClassifier)
* Support Vector Classifier (svm): It is a technique based on the Support Vector Machines, which try to find a hyperplane that best separates classes in a dataset.[[4]](#SVC)

## Dataset

We decided to split out dataset in the classical two part with the following dimension: 70% training set and 30% test set. This is because the other main option, 80% training set and 20% test set, caused overfitting. Furthermore, a test set that is too small may not be representative for a reliable performance evaluation, especially since our dataset consists of only 195 examples. All other options appeared too unbalanced based on experimental results.

### Stratification

This is one of the most important parts in the data handling. The *stratification* consists in maintain the dataset balanced also after the split. This means maintaining the same number of examples for each class in both the training set and the evaluation set. This prevents the model from learning more about one class or more than other classes.

## Features

## Accuracy

We have tested and evaluated all these models to find the best one for our task.

|  |  |  |
| --- | --- | --- |
| Model | Acc. with stratification | Acc. without stratification |
| Gradient Boosting Classifier | 75% | 68% |
| Random Forest Classifier | 75% | 64% |
| K Neighbors Classifier | 71% |  |
| Support Vector Classifier |  | 66% |

We decide to use the Random Forest Classifier for the logical simplicity and the high accuracy. To improve accuracy, we also experimented with the “GridSearchCV” class from scikit-learn. This class allows us to systematically search through multiple hyperparameter combinations by splitting the dataset into several folds and training the model on each fold. It then evaluates the performance of each model configuration to find the best one. This approach was unsuccessful because the dataset was too small, which led to a worsening of performance.

# Bibliography

[1] <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html#sklearn.ensemble.GradientBoostingClassifier>.

[2] <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>.

[3] <https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>.

[4] <https://scikit-learn.org/stable/modules/svm.html#svm-classification>.