10/01/2024

MA 513 – Hands on machine learning for Cybersecurity

Proposition 1 : Keystroke dynamic information

Enzo SINACOLA – Wassim EL YACOUBI – Théo DOMINGUES – Paul DESBOIS
CDI - AERO 5

Table des matières

Introduction

User authentication is a crucial problem in the field of computer security, as it allows to control the access to sensitive information and resources. Traditional methods of authentication, based on passwords or PIN codes, are often vulnerable to attacks such as phishing, brute force, or keylogging. To enhance the security of authentication, it is possible to use biometric techniques, which exploit the physical or behavioural characteristics unique to everyone.

Among the biometric techniques, keystroke dynamics is a method that verifies or even tries to determine the identity of the person who produces the keystrokes, by analysing the timing information of the keystrokes, such as the duration of each key press or the interval between two consecutive key presses. Keystroke dynamics has several advantages, such as the ease of implementation, the low cost, and the discretion.

In this project, we will study keystroke dynamics as a method of user authentication, and as a tool for anomaly detection. We will first propose a small tutorial on keystroke dynamics and explain how it can be used to protect sensitive information and assets, as well as to detect suspicious or malicious behaviours. Then, we will base ourselves on the dataset 'StrongPasswordData.csv', which contains keystroke data from 51 subjects, and propose different classifier models for keystroke detection. We will describe each machine learning technique used, and the motivation of our choice. We will also evaluate and compare the performance and the robustness of the different models and discuss the advantages and disadvantages of each approach.

I. Subject chosen

Proposition 1 : Keystroke dynamic information

As a securing user authentication, Keystroke dynamic information could be used to verify or even try to determine the identity of the person who is producing the keystrokes. The techniques used to do this vary widely in sophistication and range from statistical techniques to artificial intelligence approaches like neural networks.

Propose a small tutorial about the Keystroke dynamic information and how it can be a way for protecting sensitive information and assets as well as an anomaly detection tool. Then, based on the dataset *"StrongPass-wordData.csv"*, analyse the latter and propose different classifier models for keystroke detection. Describe every used machine learning technique and the motivation behind its use.

The document is about **keystroke dynamics authentication**, which is a biometric technique that verifies the identity of a user based on how they type a password or a text. The authors explain the main concepts, challenges, and applications of this technique, as well as review the existing methods

and datasets. They also present their own contributions, such as a new benchmark dataset, a new evaluation protocol, and a new fusion scheme for keystroke dynamics. The document aims to provide a comprehensive overview of the state-of-the-art and the future directions of keystroke dynamics authentication.

Keystroke Dynamics Concepts

The authors define the keystroke dynamics as the analysis of the timing information of the keystrokes, such as the duration of each key press (**dwell time**) and the interval between two consecutive key presses (**flight time**). They also introduce the notions of **static** and **dynamic** keystroke dynamics, depending on whether the user types a fixed or a free text. They discuss the advantages and disadvantages of each approach, and the trade-offs between security and usability.

Keystroke Dynamics Challenges

The authors identify the main challenges that affect the performance and the reliability of keystroke dynamics authentication, such as the variability of the typing behaviour, the influence of external factors (e.g., keyboard, environment, mood, etc.), the presence of impostors and attackers, and the lack of standards and benchmarks. They also propose some possible solutions and directions to address these challenges, such as the use of adaptive and robust classifiers, the incorporation of additional features and modalities, the design of secure and user-friendly interfaces, and the development of common evaluation frameworks and datasets.

Keystroke Dynamics Applications

The authors present some of the existing and potential applications of keystroke dynamics authentication, such as the enhancement of password-based systems, the protection of online services and transactions, the monitoring of computer users and activities, and the identification of authors and languages. They also provide some examples and references of real-world implementations and experiments of keystroke dynamics in various domains and scenarios.

Keystroke Dynamics Methods and Datasets

The authors review the existing methods and datasets for keystroke dynamics authentication, and classify them according to different criteria, such as the type of keystroke dynamics (static or dynamic), the type of features (timing or pressure), the type of classifiers (statistical or machine learning), and the type of evaluation (verification or identification). They also compare and analyse the results and the performance of the different methods and datasets and highlight the strengths and weaknesses of each approach.

Keystroke Dynamics Contributions

The authors present their own contributions to the field of keystroke dynamics authentication, which include:

- A new benchmark dataset, called **GREYC keystroke**, that contains keystroke data from 133 subjects typing a fixed password and a free text, with different keyboards and sessions. The dataset also includes demographic and biographic information, as well as pressure features measured by a special keyboard.

- A new evaluation protocol, called **one vs. one**, that allows a fair and realistic comparison of different keystroke dynamics methods and datasets, by considering the variability of the typing behaviour and the impostor distribution.
- A new fusion scheme, called **weighted majority voting**, that combines the decisions of multiple classifiers based on their confidence levels, and improves the accuracy and the robustness of keystroke dynamics authentication.

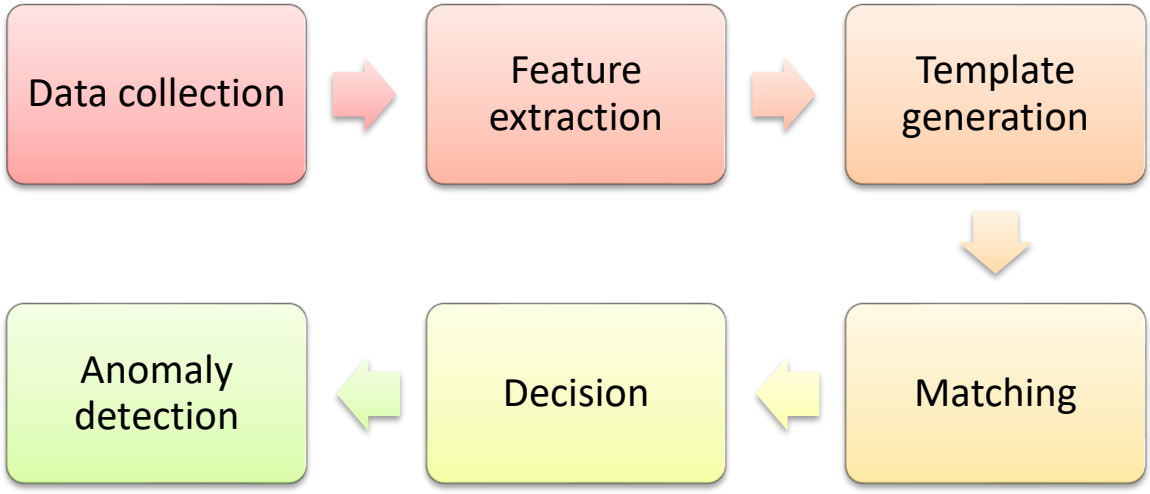
The authors demonstrate the effectiveness and the relevance of their contributions by conducting extensive experiments and comparisons with other methods and datasets, and by reporting promising results and insights.

Conclusion

The document concludes by summarizing the main points and findings of the document, and by outlining the future work and perspectives of keystroke dynamics authentication. The authors emphasize the potential and the benefits of keystroke dynamics as a biometric technique, and the need for further research and development to overcome the existing challenges and limitations, and to explore new applications and opportunities.

II. Tutorial for Keystroke dynamic information

Keystroke dynamics is a biometric technique that uses the timing and pressure of each key press and release to identify or verify a user based on their typing behaviour. Unlike other biometric techniques, such as fingerprint or iris recognition, keystroke dynamics does not require any special hardware or sensors and can be easily integrated into existing systems that use keyboards, such as computers, laptops, or mobile devices. Keystroke dynamics can also provide continuous authentication, which means that the system can monitor the user's typing behaviour throughout the session and detect any anomalies or changes that may indicate an impostor or a security breach.



The basic steps of keystroke dynamics are as follows:

- Data collection: The system collects keystroke data from users who want to enrol in the system. The keystroke data consists of the timing and pressure of each key press and release, as well as the sequence of keys typed by the user. The system can also use additional sensors, such as accelerometer and orientation sensors, to capture more information about the typing behaviour.

- Feature extraction: The system extracts feature from the keystroke data, such as the duration of each key press (hold time), the time between two consecutive key presses (latency), the time between pressing and releasing a key (dwell time), the time between releasing a key and pressing another key (flight time), and the n-graphs of consecutive keys (e.g., digraphs, trigraphs, etc.). These features represent the characteristics of the user's typing behaviour and can be used to distinguish different users.

- Template generation: The system builds a biometric template for each user based on the extracted features. The biometric template is a representation of the user's typing behaviour that can be used for comparison with new keystroke data. The system can use various techniques to create a template, such as statistical methods (e.g., mean, standard deviation, etc.), machine learning methods (e.g., neural networks, support vector machines, etc.), or hybrid methods (e.g., combining statistical and machine learning methods).

- Matching: The system compares the biometric template of a user with the keystroke data of a new typing session to determine the similarity or dissimilarity between them. The system can use different metrics to measure the similarity or dissimilarity, such as Euclidean distance, Manhattan distance, Mahalanobis distance, etc. The system can also use different modes of operation, such as verification mode (where the user claims an identity and the system verifies it) or identification mode (where the system determines the identity of the user from a set of enrolled users).

- Decision: The system decides whether to accept or reject the user based on a predefined threshold or a dynamic threshold that adapts to the user's typing behaviour over time. The system can also use different criteria to make the decision, such as false acceptance rate (FAR), false rejection rate (FRR), equal error rate (EER), or receiver operating characteristic (ROC) curve.

- Anomaly detection: The system detects any anomalies in the keystroke data, such as changes in the typing rhythm, errors, corrections, pauses, etc. The system can use different methods to detect anomalies, such as statistical methods (e.g., z-score, outlier detection, etc.), machine learning methods (e.g., clustering, classification, etc.), or hybrid methods (e.g., combining statistical and machine learning methods). The system can also use different actions to respond to anomalies, such as alerting the user, requesting additional verification, or terminating the session.

Keystroke dynamics can be used for protecting sensitive information and assets, as well as an anomaly detection tool, by providing a reliable and convenient way of authenticating and verifying users based on their typing behaviour. Keystroke dynamics can also enhance the security and usability of existing systems that use passwords or other authentication methods, by adding an additional layer of protection or reducing the frequency of password prompts. Keystroke dynamics can also prevent unauthorized access or impersonation by detecting any deviations or inconsistencies in the user's typing behaviour that may indicate an attack or a compromise.

III. Presentation of our models

1. Dataset analysis

For this tutorial, we'll use the **'StrongPasswordData.csv'** dataset. This dataset contains information about keystroke dynamics, including key press and release times. Let's explore different machine learning models to detect keystroke patterns.

This dataset is composed of several columns such as "*subject*" that we will use as the target for the model. We will drop "*sessionIndex*" and "*rep*" since they are not used or mentioned in the research paper used to help us. The rest are the features used to train and test our models.

The features are a succession of columns showed like this :

- H (Hold Time):

Definition: The duration for which a key is held down.

Example: H.i represents the hold time for the key 'i'.

- DD (Dwell Time or Flight Time):

Definition: Dwell time is the time between the release of one key and the press of the next key. Flight time is the time taken to move from one key to the next without releasing the first key.

Example: DD.i.e represents the dwell time between the keys 'i' and 'e'.

- UD (Up-Down Time):

Definition: The time interval between the release of one key and the press of the same key again.

Example: UD.i.e represents the up-down time for the key 'i' between the keys 'i' and 'e'.

These timing features are crucial in creating a unique profile for everyone's typing behaviour. Analysing the variations in hold times, dwell times, and up-down times can contribute to the development of a reliable keystroke dynamics-based authentication system.

2. Machine Learning Models used

Support Vector Machines (SVM)

- Motivation: SVM is known for its effectiveness in handling high-dimensional data and is well-suited for binary classification tasks like keystroke detection
- Approach: SVM constructs a hyperplane that best separates data points into different classes, maximizing the margin between them.

Random Forest Classifier (RFC)

- Motivation: RFC is an ensemble learning method that combines multiple decision trees, providing robustness and reducing overfitting.
- Approach: RFC builds multiple decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

K-Nearest Neighbors (KNN)

- Motivation: KNN is a simple, yet powerful, algorithm that can be used for pattern recognition tasks.
- Approach: KNN classifies new data points based on the majority class of their k-nearest neighbors. It works well when the decision boundary is non-linear.

3. Models' tunning

Since we want to get the best results, we will try to use for each model, so we keep the best parameters according to the algorithm.

To optimize the performance of each model, we will use GridSearch, a hyperparameter tuning technique:

- SVM Tuning: Explore different kernel functions (linear, polynomial, radial basis function) and tune the regularization parameter.
- RFC Tuning: Adjust the number of trees, maximum depth, and other hyperparameters to enhance performance.
- KNN Tuning: Optimize the number of neighbours and explore different distance metrics for KNN.

IV. Results

The details on how we managed to study, split, and train/test our data are explained on the notebook file "**Projet_Ma513_Sujet1_ELYACOUBI_SINACOLA_DOMINGUES_DESBOIS.ipynb**".

What we are interested here is the results given by each model, let's have a look for our 3 models.

1. SVM

After the GridSearch we found that hyperparameters should be :

Argument	Value
C	10
gamma	auto
kernel	rbf

The score obtained during training and testing for this combination :

Dataset	Performance (overall)
Training	88.7 %
Testing	90.0 %

The classification report is given in the notebook for more details, but we can see that for some subjects, it performs very well, even almost perfectly, but some of the subjects are a little too hard to predict for the model. In average, it seems alright. Meaning we may have some features that the model didn't manage to link correctly to the targets.

2. RFC

After the GridSearch we found that hyperparameters should be :

Argument	Value
max_depth	None
min_samples_leaf	1
min_samples_split	2
n_estimators	200

The score obtained during training and testing for this combination :

Dataset	Performance (overall)
Training	93.8 %
Testing	94.0 %

The classification report is given in the notebook for more details, but we can see now that it performs better overall for all targets, even perfectly for some of them. Which we could be reassured about since RFC is a robust model for that kind of study.

3. KNN

After the GridSearch we found that hyperparameters should be :

Argument	Value
n_neighbors	8
p	1 (Manhattan distance)
weights	distance

The score obtained during training and testing for this combination :

Dataset	Performance (overall)
Training	91.2 %
Testing	92.0 %

The classification report is given in the notebook for more details. It looks like this model is currently in between RFC and SVM in terms of performance only by looking the report. But we will need more indicators to say that.

Overall, we have now :

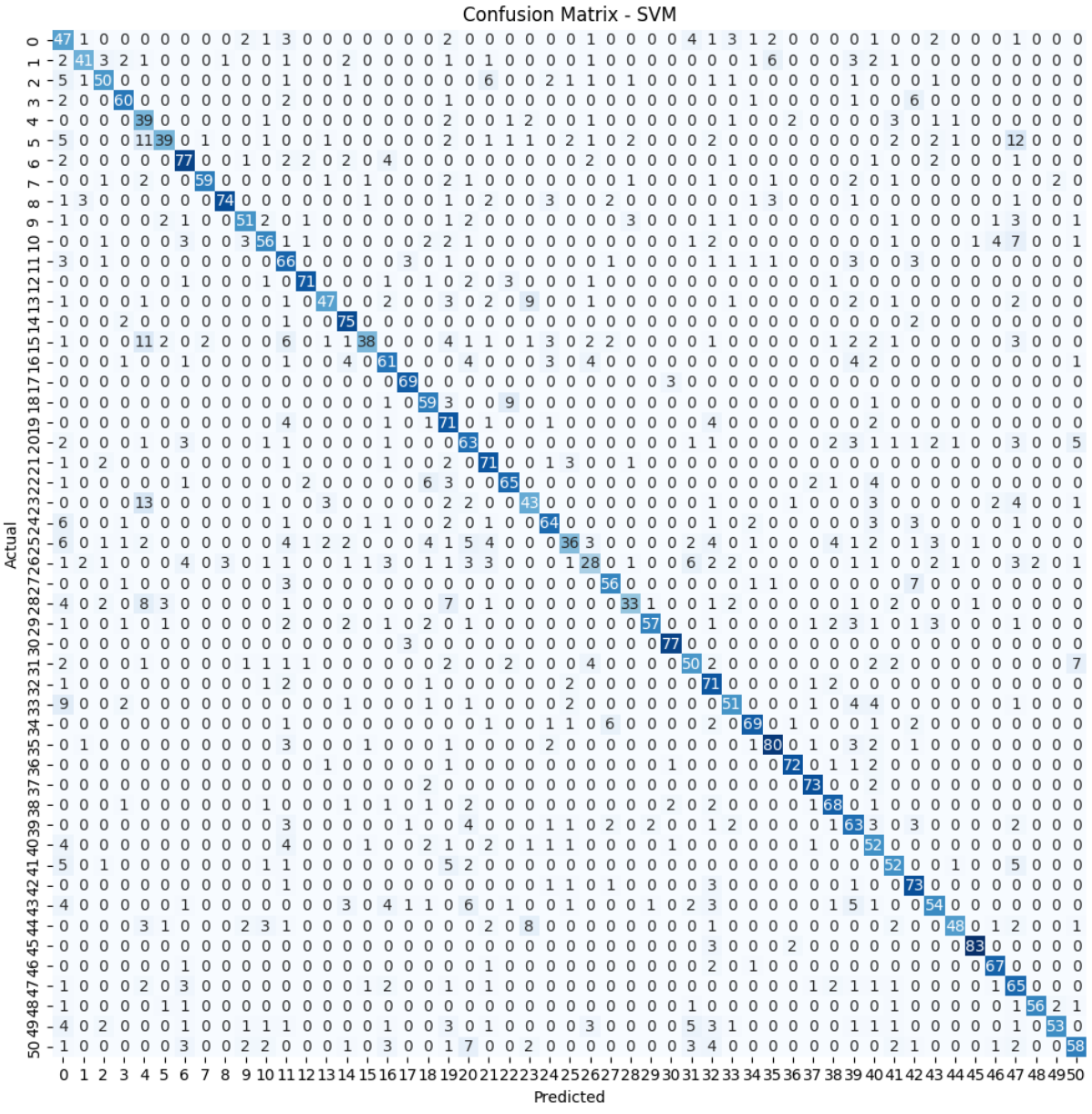
	Training	Testing
SVM	88.7 %	90 %
RFC	93.8 %	94 %
KNN	91.2 %	92.0 %

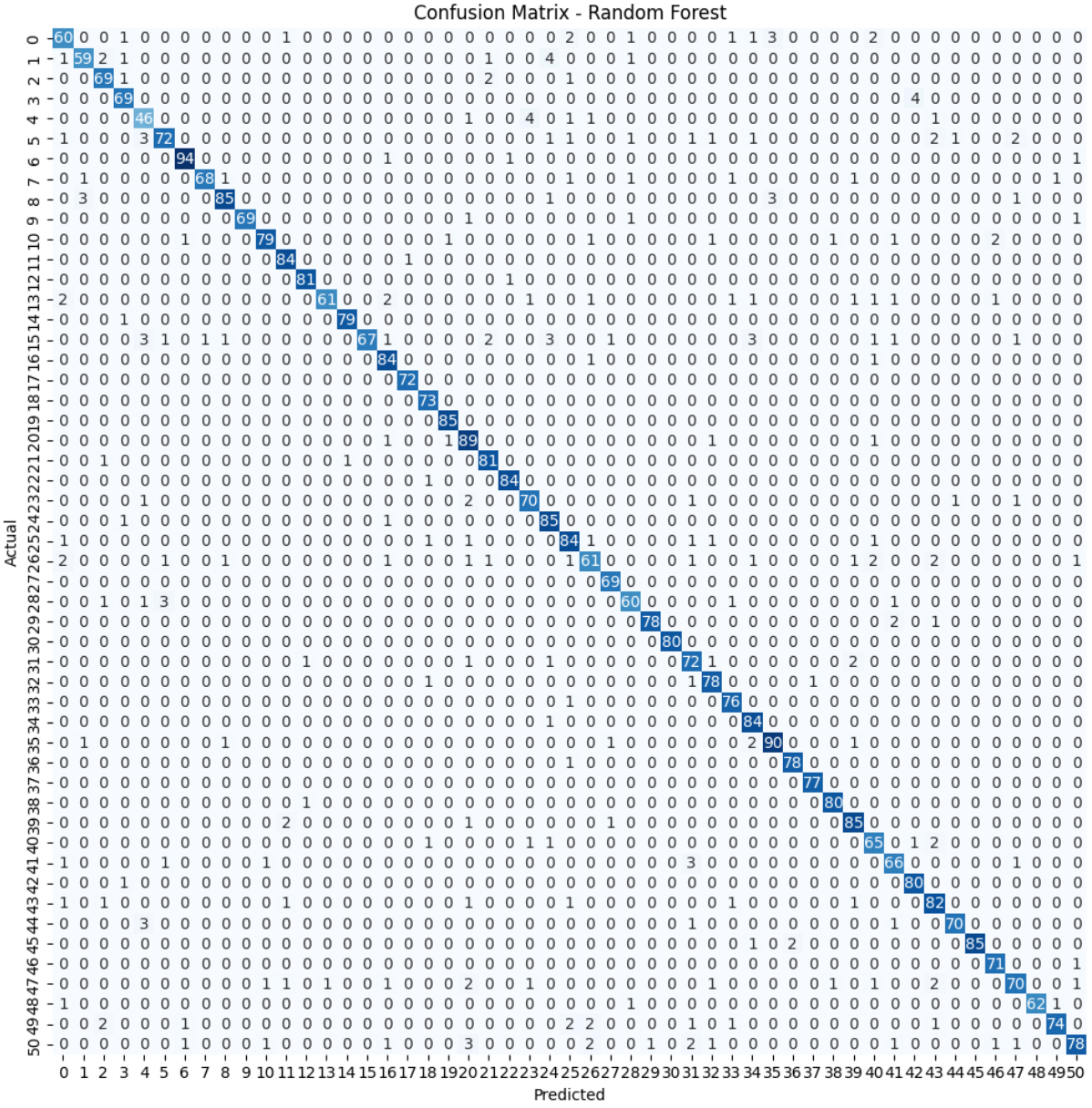
4. Performance analysis

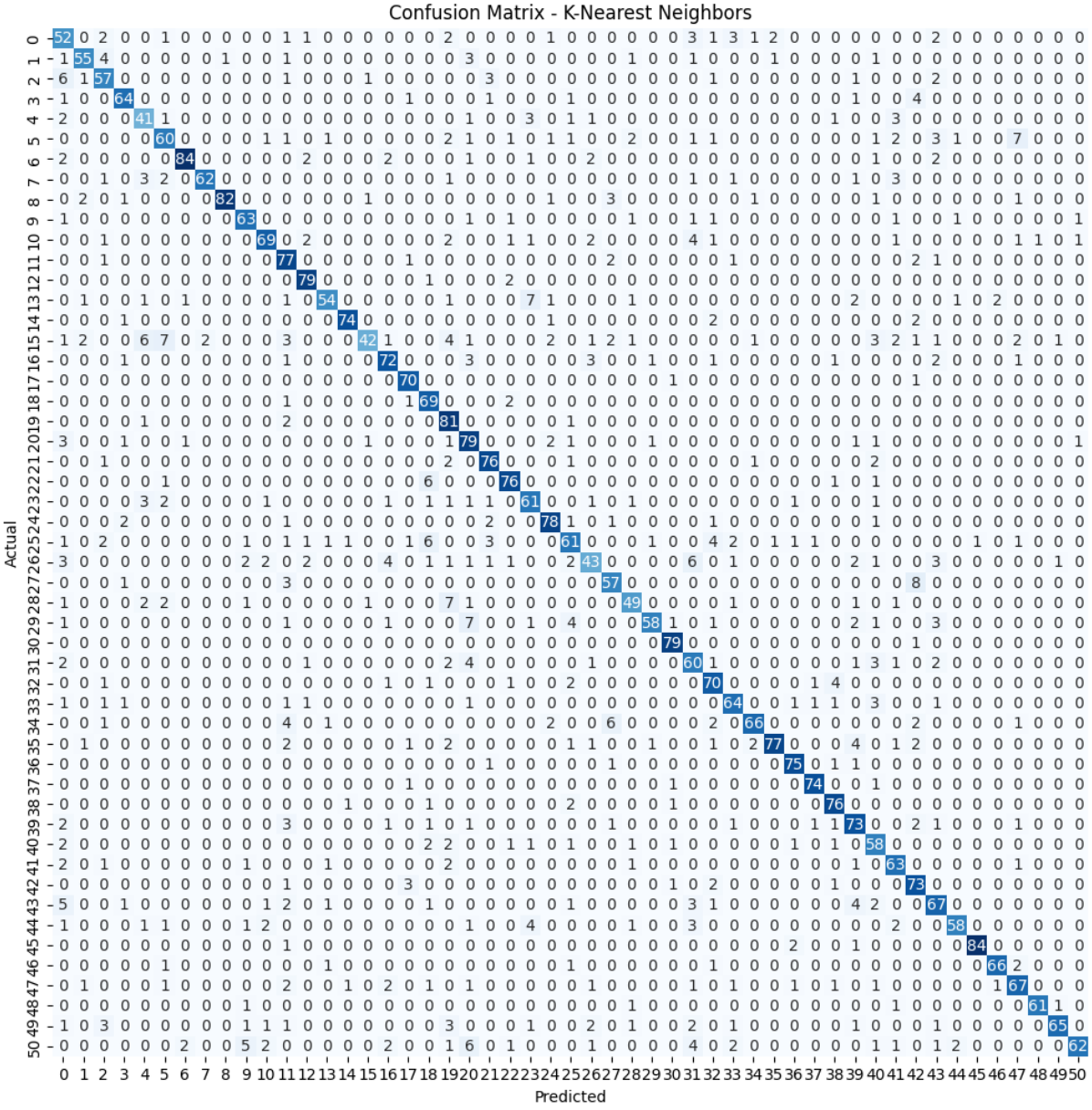
Having some score for testing and prediction is great, but we can also have some insight about what happened for some models to have an idea of what we can change to get better results.

1. Confusion matrix :

If we have a look at the confusion matrix for the 3 models we get :




Confusion Matrix - Random Forest
Actual
Predicted



What we want for the models is to have the greatest values for cells in the diagonal, meaning we are tending to have no false-positives or false-negatives.

This is what we got for the RFC model, very few cells that are not in the diagonal are different to 0, which is what we could expect from this model with the score precision before.

However, for the SVM model, we got a lot of false-positives and false-negatives for some target, meaning that the model really failed to connect features with the targets during training.

The last model KNN performed well but slightly underperformed regarding RFC model since we still have some issues for some targets, but still less than the SVM model.

In terms of false-positives and false-negatives, we can sort models like that : **RFC > KNN > SVM**, like for the accuracy scores form training and testing.

2. Indicators

Confusion matrices are great to have a look at the overall performance, but we want to know if we can have an idea about why the model performed well or not.

For that we will use 3 indicators : Cohen's Kappa (CK), Hamming loss (HL) and Matthew's Correlation Coefficient (MCC).

These are the results we got, the details are in the notebook, on how we performed that analysis :

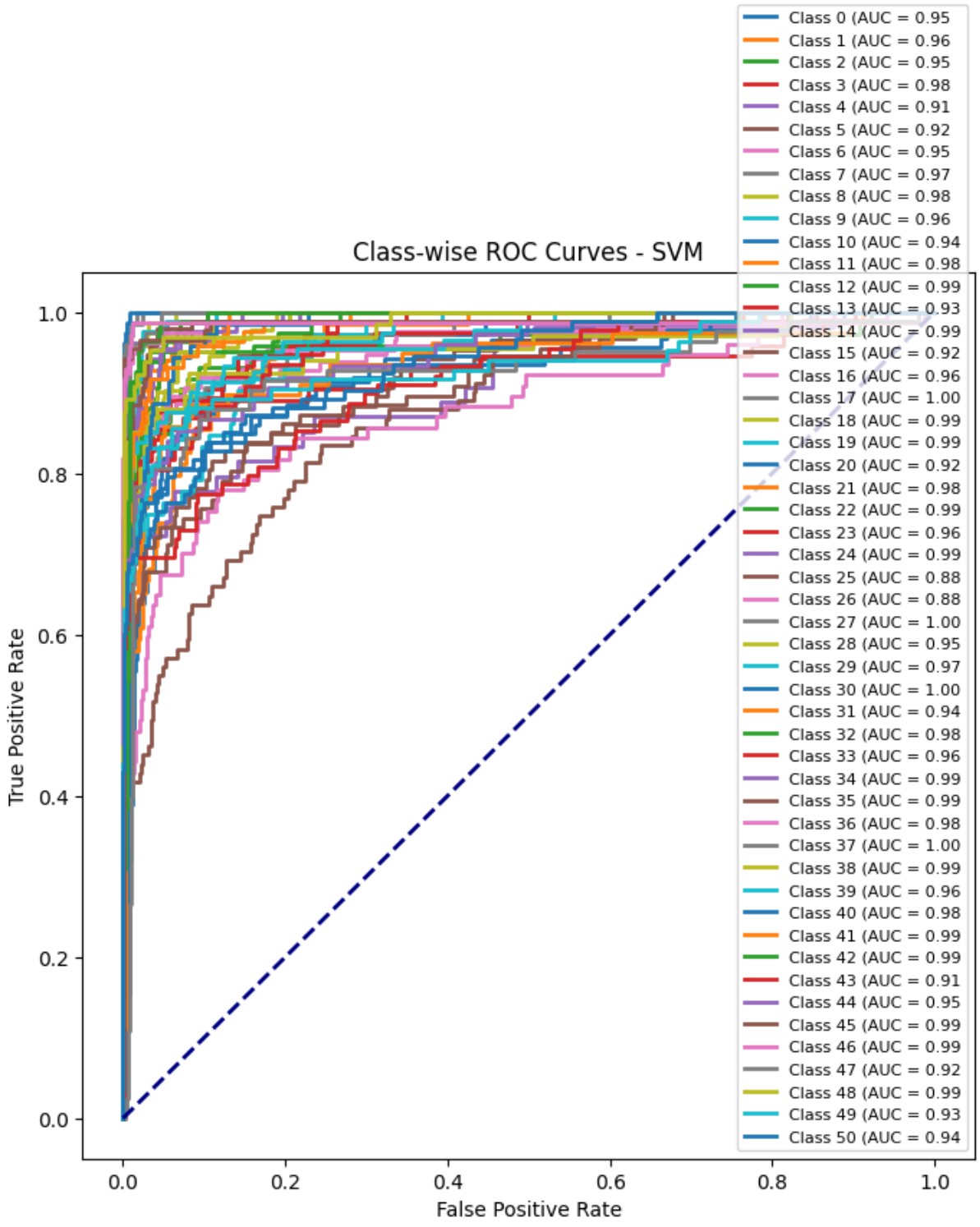
	CK	HL	MCC
SVM	**0.73**	**0.26**	**0.73**
RFC	**0.93**	**0.06**	**0.94**
KNN	0.83	0.17	0.83

RFC outperformed the other models, since the SVM model struggles to keep up the pace.

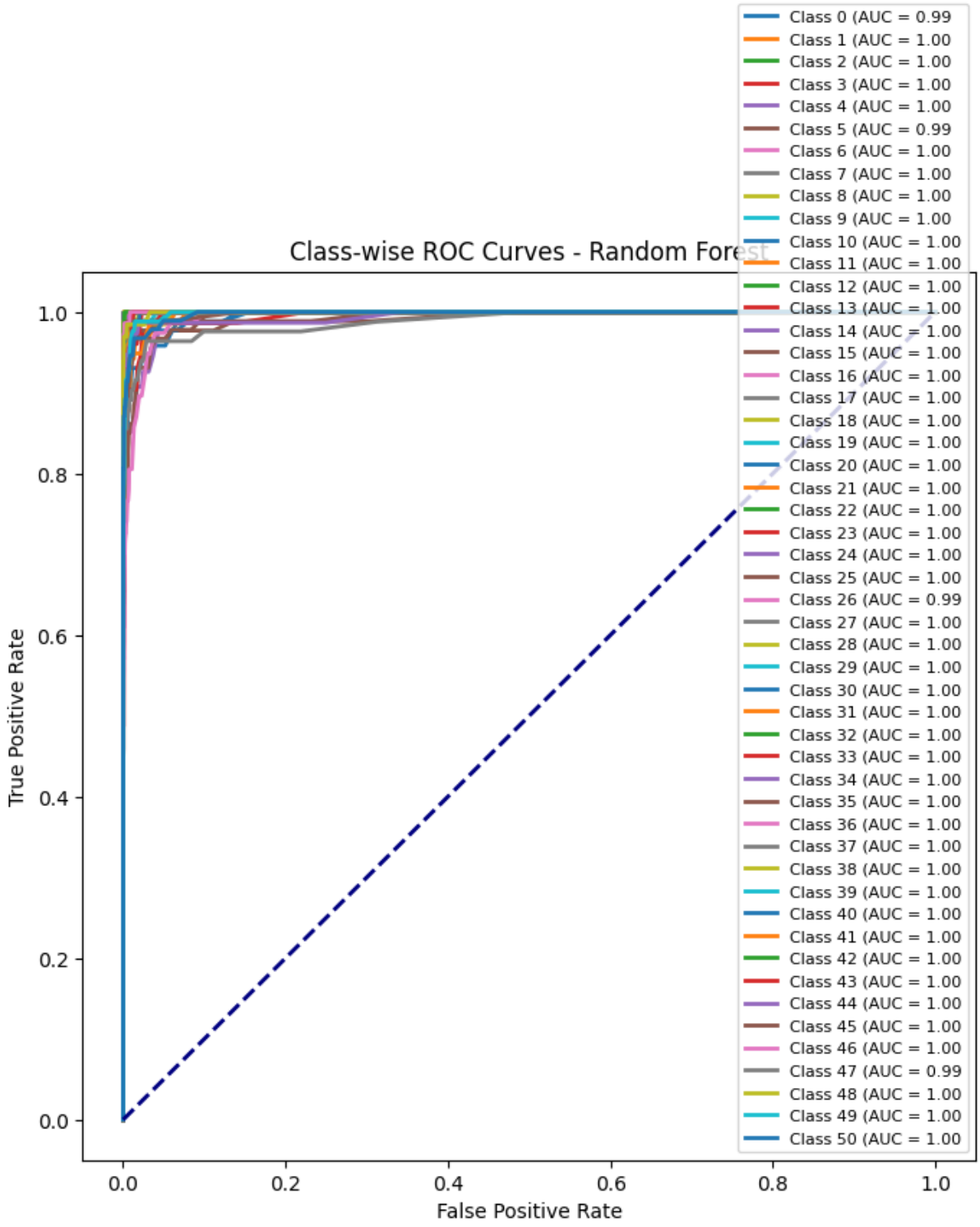
3. Roc curves class-wise

What we can also do is to plot class-wise roc curves to watch for each model what class may be failing in the prediction : the more the curves are up-left positioned on the graph means the prediction is less failing in predictions.

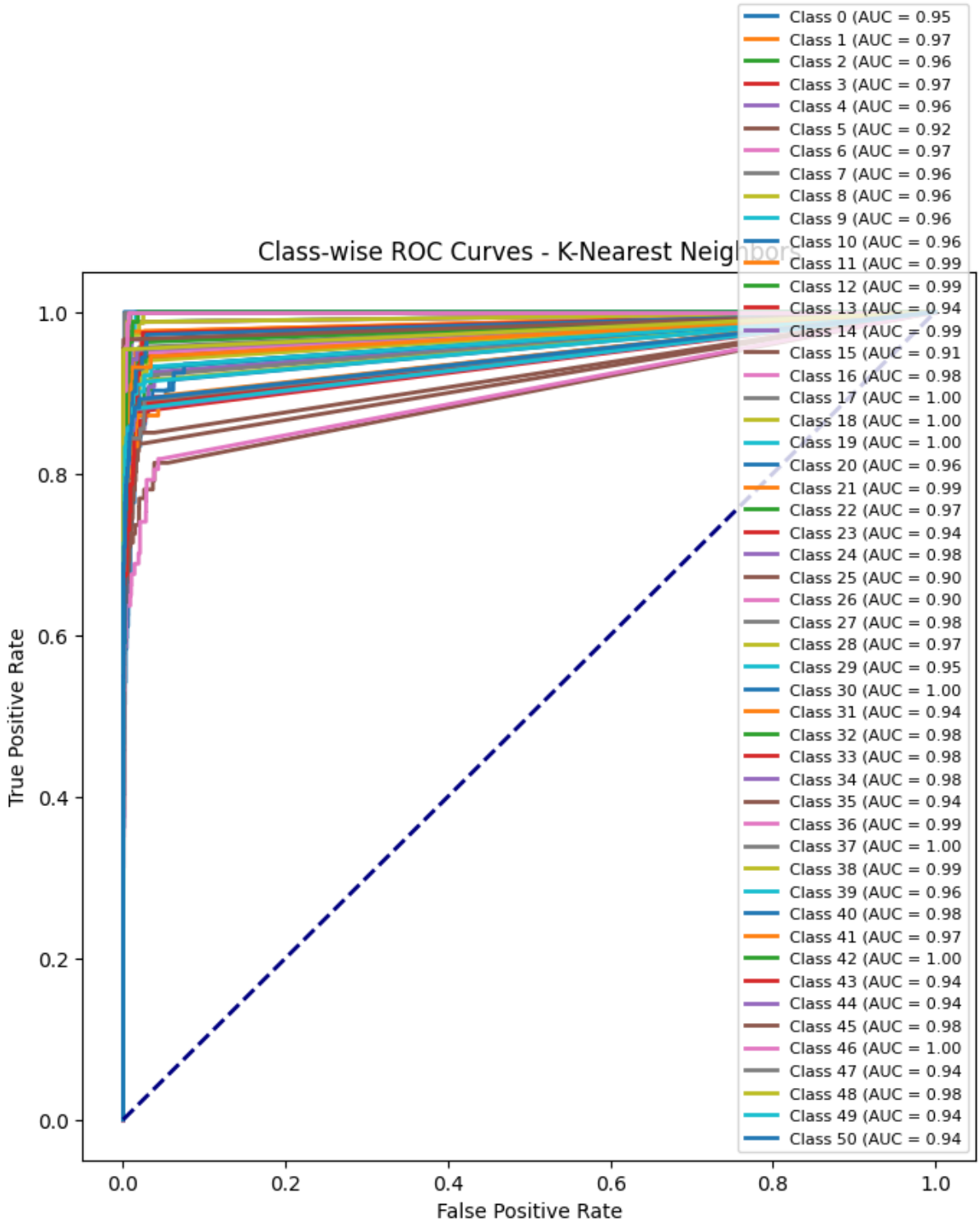
- SVM :



- RFC :


Class-wise ROC Curves - Random Forest
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Class 0 (AUC = 0.99
Class 1 (AUC = 1.00
Class 2 (AUC = 1.00
Class 3 (AUC = 1.00
Class 4 (AUC = 1.00
Class 5 (AUC = 0.99
Class 6 (AUC = 1.00
Class 7 (AUC = 1.00
Class 8 (AUC = 1.00
Class 9 (AUC = 1.00
Class 10 (AUC = 1.00
Class 11 (AUC = 1.00
Class 12 (AUC = 1.00
Class 13 (AUC = 1.00
Class 14 (AUC = 1.00
Class 15 (AUC = 1.00
Class 16 (AUC = 1.00
Class 17 (AUC = 1.00
Class 18 (AUC = 1.00
Class 19 (AUC = 1.00
Class 20 (AUC = 1.00
Class 21 (AUC = 1.00
Class 22 (AUC = 1.00
Class 23 (AUC = 1.00
Class 24 (AUC = 1.00
Class 25 (AUC = 1.00
Class 26 (AUC = 0.99
Class 27 (AUC = 1.00
Class 28 (AUC = 1.00
Class 29 (AUC = 1.00
Class 30 (AUC = 1.00
Class 31 (AUC = 1.00
Class 32 (AUC = 1.00
Class 33 (AUC = 1.00
Class 34 (AUC = 1.00
Class 35 (AUC = 1.00
Class 36 (AUC = 1.00
Class 37 (AUC = 1.00
Class 38 (AUC = 1.00
Class 39 (AUC = 1.00
Class 40 (AUC = 1.00
Class 41 (AUC = 1.00
Class 42 (AUC = 1.00
Class 43 (AUC = 1.00
Class 44 (AUC = 1.00
Class 45 (AUC = 1.00
Class 46 (AUC = 1.00
Class 47 (AUC = 0.99
Class 48 (AUC = 1.00
Class 49 (AUC = 1.00
Class 50 (AUC = 1.00

- KNN :



We can rapidly see that, as predicted with the other indicators, RFC has the best curves for each class. We can also see that for some features, KNN and SVM are struggling to make the difference between subjects.

Receiver Operating Characteristic (ROC) curves are graphical representations commonly used to assess the performance of binary classification models, such as those used in keystroke dynamics

or other security-related tasks. The primary objective of ROC curves is to visualize the trade-off between sensitivity and specificity across different classification thresholds.

Here are the key objectives and what to expect in each case when analyzing ROC curves:

- **Sensitivity (True Positive Rate):**

Objective: Sensitivity measures the proportion of actual positive instances correctly identified by the model.

Expectation: As sensitivity increases, the model becomes better at detecting positive instances, resulting in a curve that approaches the upper-left corner of the ROC space.

- **Specificity (True Negative Rate):**

Objective: Specificity measures the proportion of actual negative instances correctly identified by the model.

Expectation: As specificity increases, the model becomes better at avoiding false alarms for negative instances, leading to a curve that approaches the upper-right corner of the ROC space.

- **Trade-off Between Sensitivity and Specificity:**

Objective: ROC curves provide a visual representation of the trade-off between sensitivity and specificity across different classification thresholds.

Expectation: The curve illustrates that as sensitivity increases, specificity may decrease, and vice versa. A model that perfectly balances sensitivity and specificity would have a curve that hugs the top-left corner, indicating high performance.

- **Area Under the ROC Curve (AUC-ROC):**

Objective: AUC-ROC quantifies the overall performance of a classification model by measuring the area under the ROC curve.

Expectation: A higher AUC-ROC value (closer to 1) indicates better overall model performance. A model with an AUC-ROC of 0.5 suggests random guessing, while an AUC-ROC of 1 indicates perfect classification.

- **Random Classifier Expectation:**

Objective: The diagonal line $(y = x)$ on the ROC space represents the performance of a random classifier.

Expectation: A model's ROC curve should be above the diagonal line. If it lies on the diagonal, the model performs no better than random chance.

- **Perfect Classifier Expectation:**

Objective: The upper-left corner of the ROC space represents the performance of a perfect classifier.

Expectation: While achieving a perfect classifier is rare in practice, a good model strives to have a ROC curve that is as close to the upper-left corner as possible.

In a certain way, the 3 models tend to be perfect classifier since we have almost all upper-left curves. However, we tend to have better results again with RFC, then KNN and finally SVM.

V. Conclusion and openings

In this project, we dug into the realm of keystroke dynamics as a method for user authentication and anomaly detection. The motivation behind this exploration was to enhance traditional authentication methods, susceptible to various attacks, by leveraging biometric techniques. Keystroke dynamics, focusing on the timing information of key presses, offers advantages such as ease of implementation, low cost, and discretion.

The tutorial provided insights into the basic steps of keystroke dynamics, including data collection, feature extraction, template generation, matching, decision-making, and anomaly detection. This biometric technique can serve as a robust tool for protecting sensitive information and assets, as well as detecting suspicious or malicious behaviors through continuous authentication.

The analysis centred on the 'StrongPasswordData.csv' dataset, featuring keystroke dynamics information from 51 subjects. Three machine learning models—Support Vector Machine (SVM), Random Forest Classifier (RFC), and K-Nearest Neighbors (KNN)—were employed for keystroke detection. A thorough examination of each model's performance, including training and testing results, confusion matrices, and class-wise ROC curves, provided a comprehensive understanding of their strengths and weaknesses.

Results indicated that RFC outperformed the other models, achieving a testing performance of 94%. The confusion matrix and additional indicators, such as Cohen’s Kappa, Hamming loss, and Matthew’s Correlation Coefficient, further supported RFC's superior performance. The ROC curves class-wise analysis revealed that RFC excelled in distinguishing between subjects.

In conclusion, keystroke dynamics, when harnessed with machine learning models, emerges as a promising avenue for user authentication and anomaly detection. RFC, with its robust performance, stands out as a preferred model in this context. However, continuous exploration and research are essential to further refine and advance keystroke dynamics authentication, addressing challenges and uncovering new opportunities in the realm of cybersecurity. The journey continues towards more secure and reliable authentication methods in the ever-evolving landscape of computer security.