Faculty of Mathematics and Information Science

Warsaw University of Technology

Project ATO – report

Subject: Data Science Workshop

Authors:
Amir Ali, Stanisław Matuszewski, Jacek Czupyt, Patryk Wrona

Warsaw
Summer Semester 2023

Contents

1. Introduction

Nowadays, computers have become mandatory to run a successful business. It is not enough to have isolated computer systems; they need to be networked to facilitate communication with external businesses. This exposes them to the outside world and hacking. System hacking means using computers to commit fraudulent acts such as fraud, privacy invasion, stealing corporate/personal data. Cyber crimes cost many organizations millions of dollars every year. Businesses need to protect themselves against such attacks [DSS21].

Indeed, even machine learning techniques have been applied in many areas of science due to their unique properties like adaptability, scalability, and the potential to rapidly adjust to new and unknown challenges. Cyber security is a fast growing field demanding a great deal of attention because of remarkable progresses in social networks, cloud and web technologies, online banking, mobile environment, smart grid [FS14] .

Imposter syndrome is very real in technology communities, and it has come into focus in cyber-security communities because it is impossible for a single person to be a true expert in everything security related [HCH22]. This has been a challenge in the cyber-secuirty space to identify this physiological behaviour of an individual. Imposter hacking is another type of hacking that has been difficult to track. Due to fact that a same individual can perform the same task differently in a given circumstance, this becomes challenging for the cyber security to identify, if an individual is an imposter.

In this project case, it was deeply discussed about one such case of imposter hacking detection using computer keyboard and mouse behaviour patterns of several individuals. The data set would provide a deep understanding of individuals behaviours of using keyboard and mouse corresponding to the imposters. The main aim of this project is to build an impostor detection system and evaluate it. Furthermore, having used various feature selection methods, it was analysed which features are relevant for such systems – related to mouse or rather to keyboard devices.

In the following chapters of this report, the data provenance and its processing steps, solution architecture, impostor detection and feature selection methodology as well as obtained evaluation results are presented. The report is concluded with a summary of results and possible future improvements in case of larger and scalable, real-time system.

2. Data

In this chapter, the data source, description and processing steps were described within consecutive sections.

2.1. Data source

The data was collected within another project within the *MiNI* faculty. During separate series of experiments, the user patterns were collected and their aggregates using a window of about 10 seconds were provided to the project's team as a text file located in the Github repository [MM23] of the project manager.

2.2. Data description

The data was composed of **21 804** rows and **114** columns. One column contained **event_time**, which was the timestamp of the aggregate in the format *YYYY-MM-DD HH:mm:ss*. Three of the columns were identifiers of a row:

— **data_donor_email** – user identifier
— **session_id** – given user's session number
— **version** – aggregate (observation) *counter* within a given session for selected user

There was 198 unique user identifiers having from 1 up to 800 observations each. Moreover, there were missing values in the provided data. As far as the valuable aggregates are concerned, 46 of the features were related to mouse events, whereas 61 were associated with keyboard, and finally there were 3 features taking into account both devices. There were no impostor labels within the dataset.

2.3. Data processing

2.3.1. Handling Missing Value

Handling missing values in a dataset is an essential step in data preprocessing to ensure accurate and reliable analysis. Based on the information you provided, there were two approaches to handling missing values: removing columns and removing rows.

1. **Remove Column - Having at least 20 percent of Missing Values:** In this approach, columns with a high percentage of missing values are removed from the dataset. According to the information data info, the following 14 features had the most missing values:
   — BOTH DEV M COUNT
   — CLUSTERED KEYS
   — CLUSTERED NR KEYS

— CLUSTERED R KEYS
— DELETE KEY
— DELETE KEYS
— MOUSE CLICK RATIO
— MOUSE DDIR D4
— MOUSE DDIR D5
— MOUSE DDIR D7
— MOUSE DDIR D8
— R DELETE
— R DELETE FREQ
— S KEY TIME MEAN

To determine whether a column should be removed, a threshold of at least 20 percent missing values was used. If a column had more than 20 percent missing values, it was considered for removal. Removing columns with a high percentage of missing values is a common practice to reduce noise and potential inaccuracies in subsequent analyses. It is done because such columns may not provide sufficient information, and imputing a large number of missing values could introduce bias into the data.

2. **Remove Row - If Any Value Is Missing within the Row:**
   10,455 observations (rows) were removed due to missing values. This approach ensures that the remaining data is complete, which can be beneficial for analyses that rely on complete observations.

3. Solution Architecture

This chapter covers the architecture of our 2 approaches – the first one concerns impostor detection systems and the latter refers to the feature selection pipeline.

3.1. Impostor Detection System

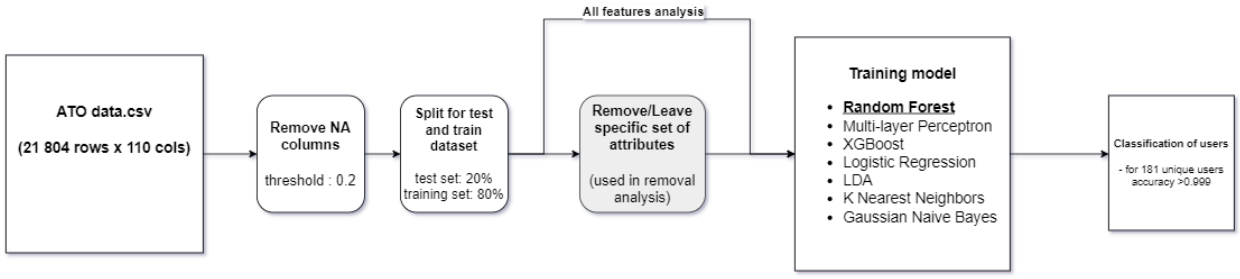

Figure 3.1. Architecture of different scenarios of impostor detection system – including all features, respective groups removal or keeping groups

The architecture diagram of the Impostor System Detection model is a pipeline consisting of six consecutive blocks:

1. ATO *data.csv*: This block represents the input data for the system, which is stored in a CSV file. It contains information and characteristics of each user's activities
2. Remove NA columns (threshold 0.2): At this stage, any columns with a high number of missing values (NA) are removed from the dataset. The threshold of 0.2 indicates that if a column has more than 20% missing values, it will be dropped.
3. Split for test and train set with a ratio of 80/20: The dataset is split into two subsets, one for training the model and the other for testing its performance. The split ratio here is 80% for training and 20% for testing. This ensures that the model is trained on a majority of the data while still having a separate set for evaluation.
4. (Optional) Remove a specific set of attributes: This block allows for the removal of specific attributes from the dataset, what was used in our research in the removal analysis. It gives the option to exclude certain columns that might not be relevant or contain sensitive information for the impostor system detection task.
5. Train a classifier model: This block represents the training phase of the model using a chosen classifier algorithm. Several options are mentioned, such as RandomForestClassifier, XGBoost, LogisticRegression, or MultilayerPerceptron. The choice of the classifier may depend on the specific requirements and performance metrics desired for the impostor system detection task.

6. Classification of users - returning predicted user identifier: Finally, the trained classifier model is used to classify users and detect impostors. The model takes input data for a user and predicts their user identifier based on the learned patterns and features.

3.2. Feature Selection

First, feature selection was based on the variable importances within developed impostor detection systems. Secondly, as an experiment, 3 standard methods of feature selection were used in order distinguish relevant features. The pipeline of feature selection methodology can be investigated in the diagram 3.2. The diagram shows from where come 4 most important feature sets:

— Variable Importance of Random Forest, from a created impostor detection system
— Correlation and High Variance
— VIF selection right after various imputation techniques
— PCA

All features within each of the feature set were then counted and according to the occurrences of respective features, one, the most relevant feature set was obtained (a red one in the evoked diagram 3.2)

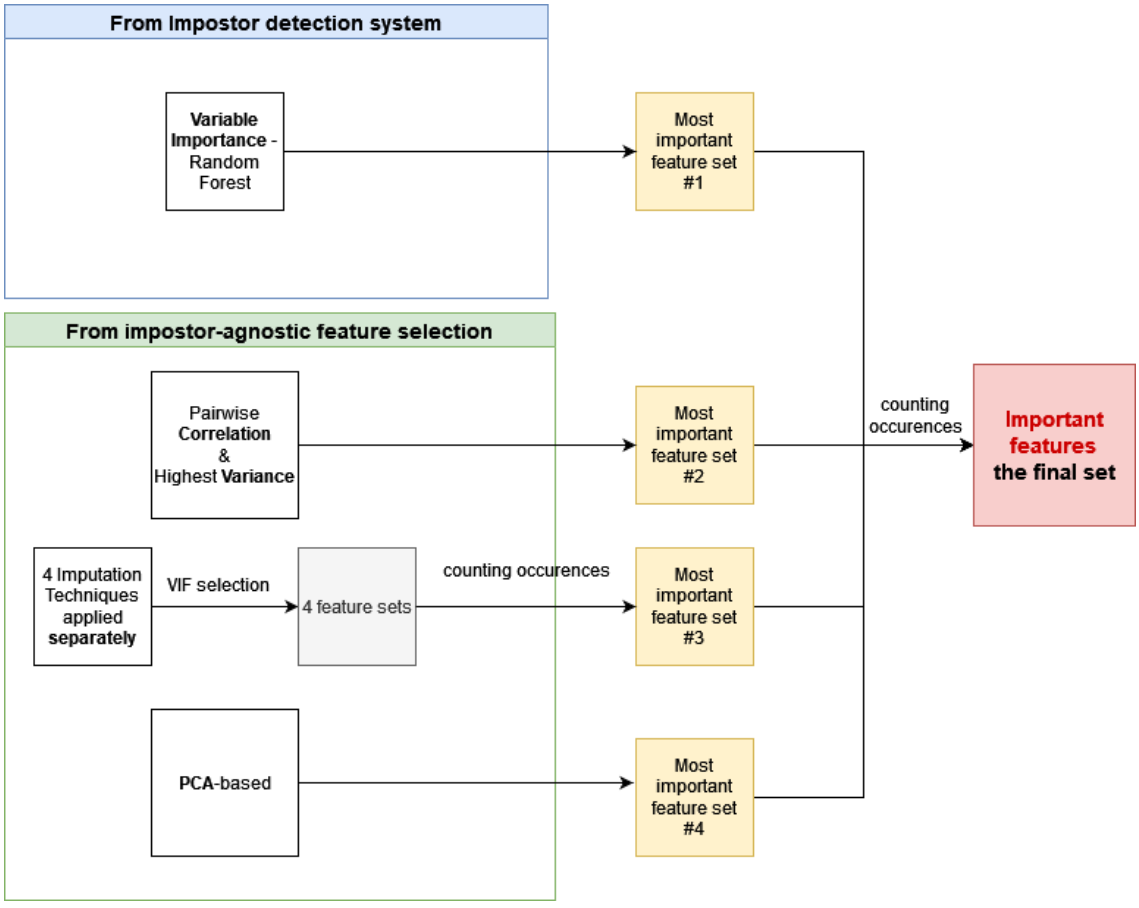

Figure 3.2. Architecture of a feature selection system

4. Impostor Detection System Methodology

4.1. Impostor Detection

The goal of the impostor detection system is to ascertain the authenticity of an account by analysing the user's mouse and keyboard usage patterns against the archived historical data. This comparison aims to identify any disparities that may indicate unauthorized access or usage by an individual other than the account owner.

We utilize a straightforward methodology wherein models are trained to classify users based on the available mouse and keyboard data. We first preprocess the data and remove missing values, as described in a previous chapter. This results in some of the users being eliminated entirely, reducing the number of classes to 180. We than split the data into a training and testing set, and train and compare the performance of multiple different models.

Several of these models offer feature importance extraction techniques, enabling us to leverage them for the purpose of feature selection.

4.2. Removal Analysis

After training a model on whole set of attributes, we conducted a feature removal analysis. The goal of removal analysis is to identify the subset of features that have the most significant impact on the predictive model's performance while excluding irrelevant or redundant features. By removing irrelevant features, the model can become more interpretable, computationally efficient, and less prone to overfitting.

In feature analysis, when considering the removal of a feature from a dataset, we expected two possible scenarios: one where the accuracy of the predictive model will significantly decrease, and another where the accuracy will remain relatively constant as the removed feature is substituted with an equivalent one.

The first scenario arises when the feature under consideration contains valuable information that is critical for accurate predictions. By eliminating such a feature, the model loses access to important patterns within the data. As a consequence, the predictive performance of the model deteriorates, resulting in a notable decrease in accuracy. This outcome underscores the significance of the feature and highlights its contribution towards achieving accurate predictions.

Conversely, the second scenario occurs when the removed feature does not possess unique or substantial information that is crucial for the predictive task at hand. In this situation, the feature may be considered redundant. As a result, the exclusion of this feature does not substantially impact the model's ability to capture the underlying patterns and relationships within the data. To maintain the accuracy levels, an alternative feature that holds similar information or captures the same underlying concept can be introduced as a replacement. This equivalent feature ensures that the model continues to leverage

the relevant information required for accurate predictions, leading to consistent accuracy despite the removal of the original feature.

5. Feature Selection

In this chapter, there are presented the methodology of feature selection methods. First, all applied standard, impostor-agnostic methods were thoroughly described, and finally the methodology involving impostor detection systems can be investigated.

5.1. Standard Method – Correlation Method

The correlation method in feature selection is a technique used to evaluate the linear relationship between variables in a dataset. It helps identify and eliminate highly correlated variables to address issues related to redundancy and multicollinearity. By measuring the correlation coefficient between pairs of variables, we can assess the strength and direction of their relationship.

In this feature selection scenario, a threshold of 70 percent was set to identify variables that were considered highly correlated. Any variables with a correlation coefficient exceeding this threshold were marked for removal. This step is crucial as it simplifies the analysis and helps mitigate the problems associated with multicollinearity, where variables are highly correlated with each other.

Initially, the dataset contained 96 features before applying the feature selection method. After calculating the correlation coefficient for all pairwise combinations of these features, it was found that 63 features exhibited high pairwise correlation, indicating a correlation coefficient exceeding 70 percent with at least one other variable.

By removing one variable from each highly correlated pair, a total of 33 features remained. These 33 features were selected based on their importance and ability to provide meaningful information for the analysis. The goal was to retain the most relevant features while considering their correlation with one another.

Using the correlation method for feature selection, the number of variables was effectively reduced from 96 to 33. This reduction aimed to retain the most informative features while eliminating redundancy caused by high correlation among variables. By doing so, the resulting feature subset is expected to enhance the model's performance and improve the interpretability of the analysis.

5.2. Standard Method – Low Variance Method

The "low variance" method in feature selection involves removing features from a dataset that have minimal variation or limited range of values. It focuses on eliminating features with low variability as they may not provide meaningful information for the modeling task. This technique helps simplify the dataset and improve model performance by retaining features with higher variance and greater potential for discrimination.

In this feature selection scenario, the focus was on removing features with a variance below 25 percent of the maximum possible variance. The objective of this process is

to eliminate features that exhibit little variability across the dataset, as they may not contribute significantly to the modeling task.

Before applying the feature selection method, the dataset initially consisted of 96 features. These features represent the variables or attributes of the dataset.

After applying the feature selection process based on low variance, it was found that 45 features had low variance and were considered for removal. These features exhibited a variance below 25 percent of the maximum possible variance, indicating their limited variability and potential lack of informative power.

As a result of the feature selection process, a reduced feature set was obtained, consisting of 51 features. These 51 features were selected based on their higher variance, indicating they have more meaningful variability and potential significance for the modeling task.

5.3. Standard Method – PCA Method

In this study, we utilized the Principal Component Analysis (PCA) method to select the 30 most important features from a dataset consisting of 96 features. The goal was to reduce the dimensionality of the dataset while retaining the most relevant information.

PCA works by transforming the original features into a new set of orthogonal variables called principal components. These components capture the maximum amount of variation in the data. By selecting a subset of these components, we can effectively represent the dataset with fewer features while preserving the most critical patterns and relationships.

In this case, selecting 30 principal components allowed for a balance between dimensionality reduction and information retention. By choosing this number, we aimed to strike a practical compromise, reducing the complexity of the dataset while still capturing a significant portion of its variability.

The selection of these 30 features was based on their contribution to the captured variance. Each feature had associated loadings or weights on the principal components, indicating their importance. By identifying the features with the highest absolute values of loadings for each principal component, we focused on the most influential and informative features in explaining the dataset's variability.

The specific categorization of the features into mouse-related, keyboard-related, and combined mouse-keyboard features informed the selection process. By including this categorization, we ensured that the selected features represented a diverse range of input device functionalities, capturing the unique aspects of mice, keyboards, and their combined usage.

5.4. Standard Method – Variance Inflation Factor after different Imputation Techniques

As stated in the chapter 2, within the data, there were missing values. In order to impute missing values, there were used 4 imputation methods:

— removing rows with missing values
— most frequent value imputation
— *k nearest neighbors* imputation
— *miceForest* (from python mice library)

After applying each technique, we proceeded in an iterative selection of 30 features using Variance Inflation Factor (VIF), obtaining 4 different sets of 30 features. The final step consisted of counting the occurrences of each feature, and the resulting set contained features occurring in all 4 sets (having count equal to 4)

5.5. Impostor Detection Systems – Selection based on Feature Importances

In our work, we utilized feature extraction techniques from pretrained classification models, with a focus on feature importances derived from our most successful Random Forest model. These feature importances provide valuable insights into the relevance and contribution of each feature towards the classification task.

Unlike classification-agnostic feature extraction methods described in the previous sections, which focus on capturing the overall variance in the dataset or treat multicollinearity, the Random Forest feature importances specifically prioritize the discriminative power of features for classification purposes. By leveraging these importance scores, we gain a deeper understanding of the critical features driving the classification model's predictions.

6. Results

6.1. Impostor Detection Systems

We have compared 10 different classification models originating from the *scikitlearn* python library, the results can be seen in table 6.1. Several of the models performed reasonably well, however the Random Forest classifier surpassed all others, showing an impressive 99.88% accuracy on the test dataset.

6.1.1. Feature Selection From Impostor Detection System

Given the excellent performance of the Random Forest model and its ability to provide more detailed feature importances then most other models, we decided to rely solely on this model for classification based feature selection.

Figure 6.1 presents top 10 most important features according to this model. The first five features are keyboard features, however three mouse features do appear on this graph, which suggests that keyboard data is slightly more valuable for classification. Figure 6.2 shows how effective a Random Forest model is when trained only on the top n most important features. It only takes 6 features to achieve a value of 99% accuracy.

model	acc	tpr	far	frr
RandomForestClassifier	**0.9988**	**0.9988**	**0.0014**	**0.0061**
GradientBoostingClassifier	0.9800	0.9800	0.0181	0.0277
KNeighborsClassifier	0.9411	0.9411	0.0578	0.0734
MLPClassifier	0.9117	0.9117	0.0667	0.0890
LinearDiscriminantAnalysis	0.9034	0.9034	0.0693	0.0566
GaussianNB	0.7878	0.7878	0.1030	0.1008
LogisticRegression	0.5190	0.5190	0.3841	0.4530
MultinomialNB	0.3085	0.3085	0.4876	0.5656
BernoulliNB	0.2532	0.2532	0.6289	0.8015
ComplementNB	0.0580	0.0580	0.7666	0.9725

Table 6.1. Impostor detection model results

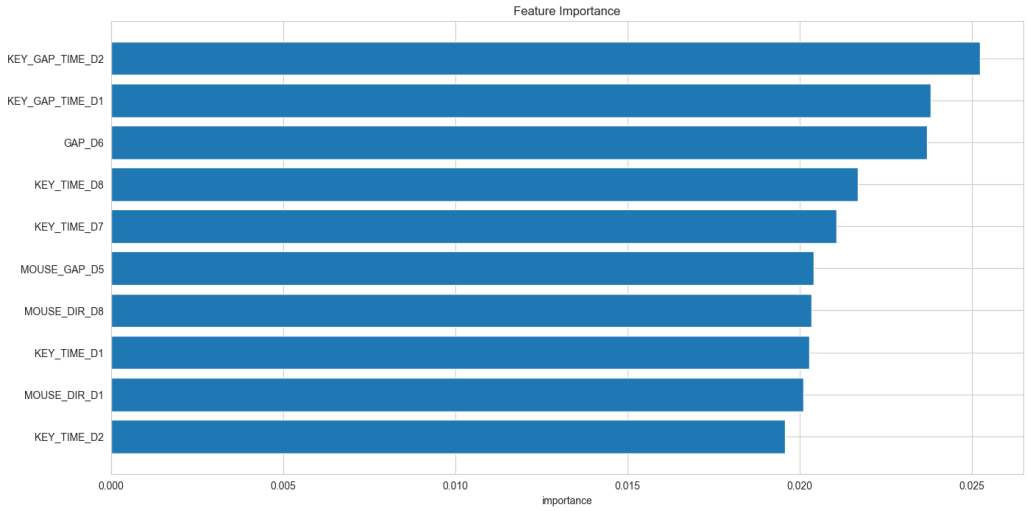

Figure 6.1. Feature importances originating from Random Forest model

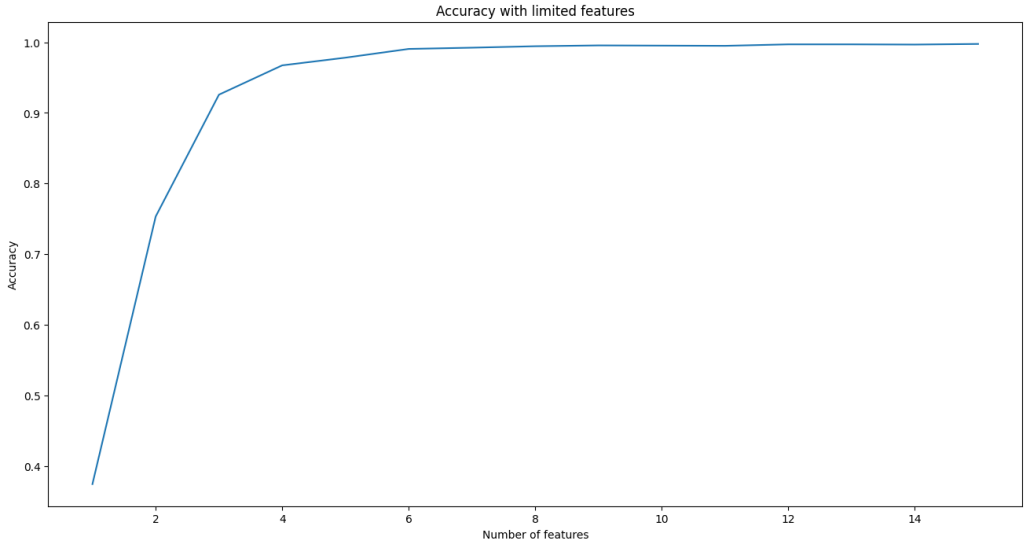

Figure 6.2. Random Forest accuracy when given only the top n features

Some of the features can be grouped together, for example representing deciles of one distribution. We call these "feature groups". In order to estimate the importance of a feature group, the importances of each individual feature were summed together. Figure 6.3 shows importances of the top 10 feature groups. The top five groups clearly outrank the remainder, with two originating from the mouse, and three from the keyboard.

Figure 6.4 shows the accuracies of models trained exclusively on one of these feature groups. It shows we can achieve a 99% accuracy using feature groups from both the mouse and keyboard.

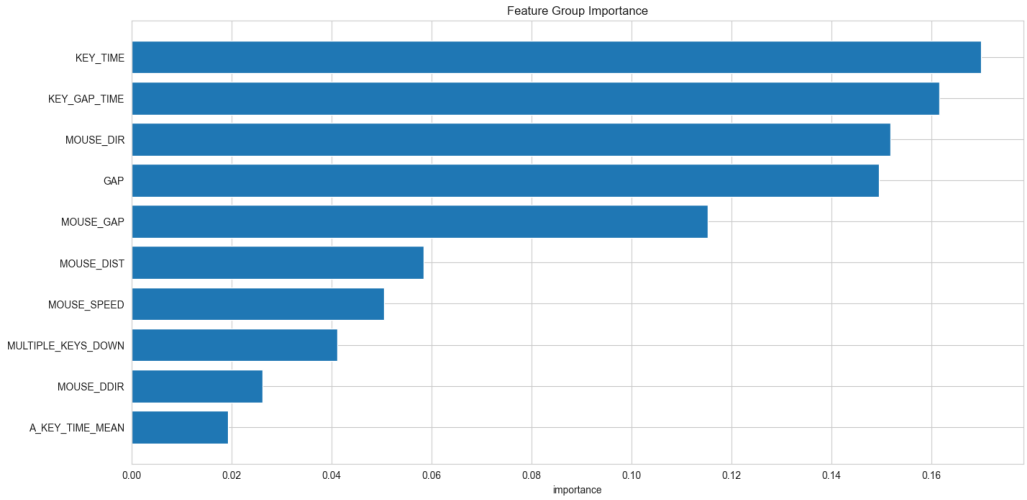

Figure 6.3. Feature group importances originating from Random Forest model

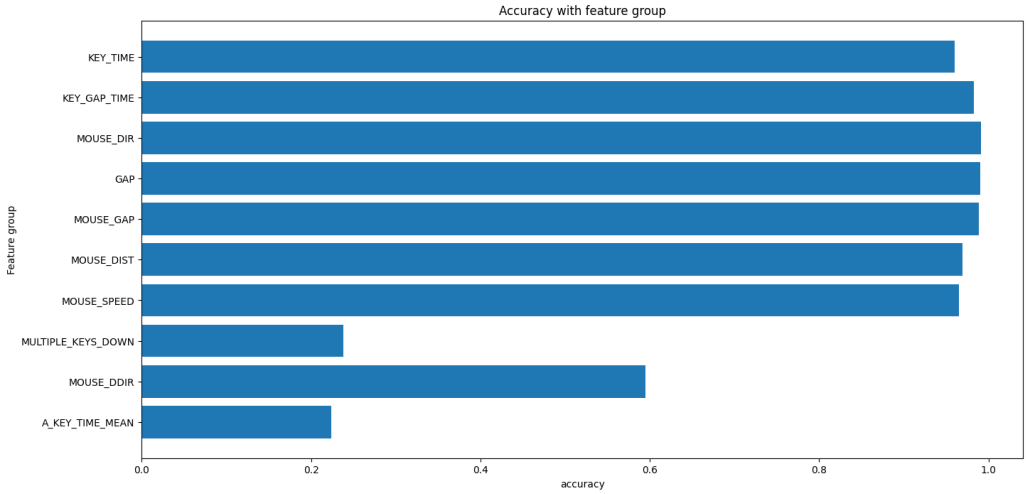

Figure 6.4. Accuracy of a random forest model trained exclusively on a single feature group.

6.1.2. Feature Importance Feature Selection with keyboard and mouse distinction

The figure 6.5 presents the percentage of features related with three groups: keyboard, mouse and both(keyboard and mouse) as a function of the number of features that were left after consecutive removal of the features having the lowest values of variable importance. We can see that having the set of few the most relevant features, there are both keyboard and mouse ones that are relevant as far as the feature importances are concerned. Features related to both devices were quickly removed during the process of selection, but there is no evidence about which – the mouse or keyboard features are more important.

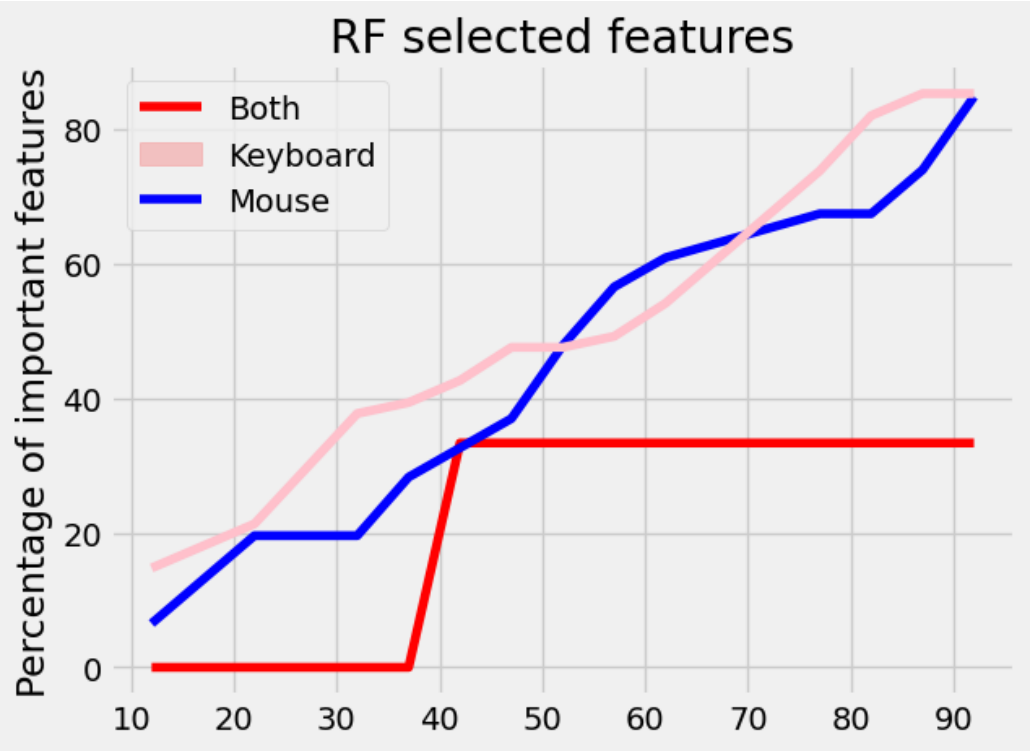

Figure 6.5. Percentage of initial columns left after removing up to selected number of features – Variable Importance Selection, RF - Random Forest

6.2. Removal Analysis

In first approach of removal analysis, we conducted a feature analysis by assessing the impact of removing single variables or their pairs on the evaluation metrics - accuracy and fpr. They found that these individual features or pairs did not have a statistically significant effect on the model's performance. As a result, in next step we decided to remove entire groups of features one by one instead. This approach aimed to capture any collective influence and explore potential redundancy within feature groups.

We observed that even the removal of entire feature groups did not have a statistically significant effect on the evaluation metrics. This finding is visually depicted in the table 6.2. The results of the analysis revealed that the accuracy of the predictive model did not decrease below threshold of 0.998 for any of the removed feature groups. Even when entire groups of features were removed, the model's performance remained remarkably stable. The high accuracy consistently observed throughout the removal of various feature groups highlighted the robustness and resilience of the model to the exclusion of these specific sets of features.

In the last approach, we focused on selectively retaining one group of features at a time while removing the remaining feature groups. The removal of these influential feature groups had a discernible impact on the model's ability to accurately make predictions concerning users. The exact results for this scenario are depicted in table 6.3.

Analysis revealed that among the various groups of features examined, the most influential groups were found to be MOUSE_DIR, MOUSE_GAP, and GAP. These particular feature groups exhibited remarkably high levels of accuracy, reaching approximately 0.99% during the evaluation process.

Removed Group	N of columns	Accuracy	TPR
A KEY TIME MEAN	91	0.99911	0.99911
BOTH DEV C COUNT	91	0.99911	0.99911
GAP	83	0.99941	0.99941
GAP SD	91	0.99941	0.99941
KEY COUNT	91	0.99882	0.99882
KEY GAP TIME	83	0.99911	0.99911
KEY TIME	82	0.99911	0.99911
KEY TIME MEAN	91	0.99911	0.99911
MOUSE COUNT	91	0.99852	0.99852
MOUSE DDIR	90	0.99882	0.99882
MOUSE DIR	83	0.99882	0.99882
MOUSE DIST	83	0.99911	0.99911
MOUSE GAP	83	0.99882	0.99882
MOUSE SPEED	83	0.99911	0.99911
MULTIPLE KEYS DOWN	82	0.99911	0.99911
MULTIPLE KEYS UP	82	0.99852	0.99852

Table 6.2. Feature selection using Random Forest classifier - results after removing one selected group

Left Group	N of columns	Accuracy	TPR
A KEY TIME MEAN	1	0.22431	0.22431
BOTH DEV C COUNT	1	0.10715	0.10715
GAP	9	0.98881	0.98881
GAP SD	1	0.06123	0.06123
KEY COUNT	1	0.01442	0.01442
KEY GAP TIME	9	0.98174	0.98174
KEY TIME	10	0.95819	0.95819
KEY TIME MEAN	1	0.20194	0.20194
MOUSE COUNT	1	0.02619	0.02619
MOUSE DDIR	2	0.59434	0.59434
MOUSE DIR	9	0.99175	0.99175
MOUSE DIST	9	0.97115	0.97115
MOUSE GAP	9	0.98910	0.98910
MOUSE SPEED	9	0.96231	0.96231
MULTIPLE KEYS DOWN	10	0.24109	0.24109
MULTIPLE KEYS UP	10	0.15837	0.15837

Table 6.3. Feature selection using Random Forest classifier - results for selected left groups

Moreover, it was observed that the majority of feature groups with accuracy below 95% consisted of only one or two features. Among the groups with lower accuracy, two exceptions stood out: MULTIPLE_KEYS_DOWN and MULTIPLE_KEYS_UP. These two feature groups consisted of 10 distinct features each. Despite their larger size, these groups exhibited significant variations in accuracy when compared to similar-sized feature groups.

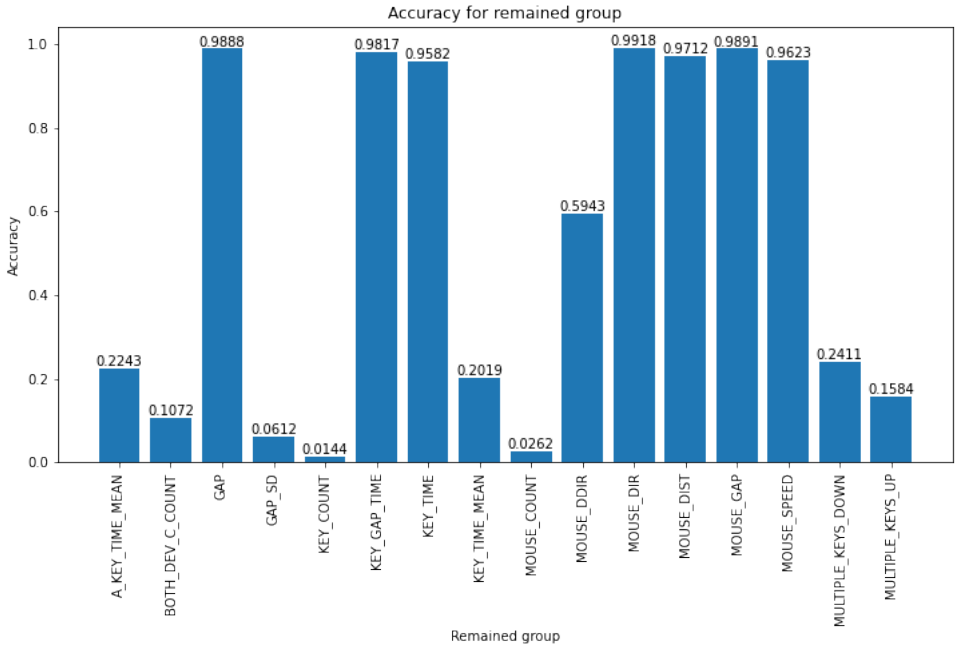

Figure 6.6. Feature selection - Accuracy results after training *RF* classifier only with a selected group

6.3. Standard Methods of Feature Selection

Below are presented the results of standard, impostor-agnostic feature selection methods, thoroughly described in the chapter 5.

6.3.1. Aggregated features from standard methods

The table 6.4 shows the features that were chosen as a result of Feature Selection based on Standard impostor-agnostic methods. It show that according to standard methods, the most relevant features, selected by all four methods were:

— **MOUSE_GAP_D1,3** - 1st and 3rd decile of time between mouse events
— **MULTIPLE_KEYS_UP_D1** - 1st decile of keys released at the same time

6.3.2. Variance Inflation Factor Feature Selection with keyboard and mouse distinction

The figure 6.7 presents the percentage of features related with three groups: keyboard, mouse and both(keyboard and mouse) as a function of the number of features that were left after consecutive removal of the features having the lowest values of Variance Inflation Factor. In fact, the selection was based on *VIF* relying on a linear regression model used on features solely. One can notice that at the beginning phase of the selection, keyboard features were being consequently removed, and mouse features were deemed as less linearly interdependent and thus more relevant. Moreover, having reached a small subset of features, there were still keyboard-related ones which were relevant – that is why below the threshold of 50 left features, usually the next removed one was a mouse-related column. Therefore, at the final stage of the selection, there are both mouse and keyboard features that are essential.

Lp	Feature Name	Occurrence Count
1	MOUSE GAP D1	4
2	MULTIPLE KEYS UP D1	4
3	MOUSE GAP D3	4
4	MOUSE SPEED D3	3
5	MANY KEYS BY KEY PRESSED	3
6	MOUSE DDIR D2	3
7	MOUSE SPEED D5	3
8	MULTIPLE KEYS DOWN	3
9	MULTIPLE KEYS UP	3
10	GAP D1	3
11	MOUSE DIST D3	3
12	KEY TIME	3
13	MOUSE DIST D1	2
14	KEY TIME D9	2
15	MOUSE DIR D5	2
16	MOUSE DDIR D3	2
17	MOUSE DIST D2	2
18	MOUSE DDIR D1	2
19	MULTIPLE KEYS UP D7	2
20	MOUSE DIST D5	2

Table 6.4. Occurrence count of the 20 most important features – all standard feature selection methods merged

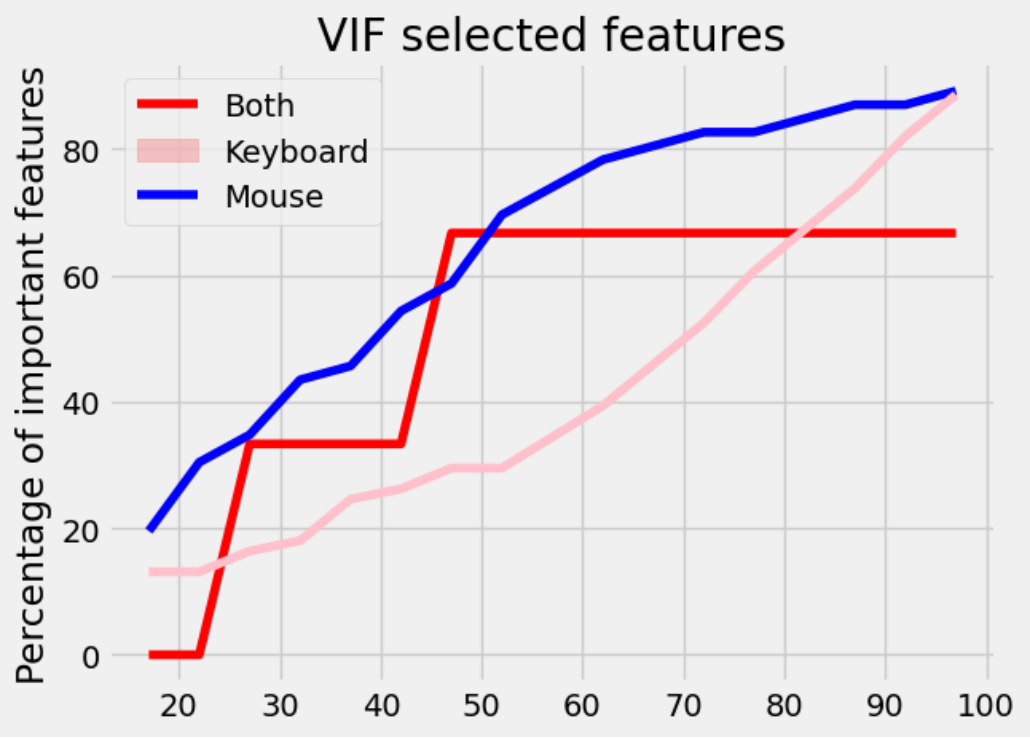

Figure 6.7. Percentage of initial columns left after removing up to selected number of features – VIF selection

7. Conclusions

In conclusion, the prepared classification model demonstrated remarkably good results, surpassing our initial expectations. The results of the feature removal analysis indicate that using a significantly smaller group of features in the Random Forest model holds promise for achieving similar levels of evaluation metrics.

It should be noted that the evaluation metrics reported in the study were obtained from a dataset comprising fewer than 200 users. In many real-life scenarios, the number of users is typically higher, which may potentially impact the performance of the model and lead to a deterioration in the achieved results. The generalizability of the findings to larger user populations should be considered, as the increased complexity and variability introduced by a larger user base can pose challenges to the model's performance. Therefore, it is important to further investigate and validate the model's robustness and predictive capabilities in scenarios with a more extensive user population to ensure reliable performance in practical applications.

Furthermore, the classification model employed in this study necessitates retraining each time a new user needs to be incorporated into the system. This requirement poses a practical challenge, as retraining the model for each new user can be time-consuming and resource-intensive. To address this issue, several potential techniques like incremental learning, transfer learning, or active learning can be considered. Perhaps, in the event of adding new users to the system, it would be more relevant to use Adaptive Random Forest – such a method would permit to a quick adaption to the presence of a new user within a system, not mentioning the fact of being applicable in real-time systems where the data is collected in a form of user event streams.

Finally, according to applied feature selection methods, Random Forest could find value in keyboard features, considering them as comparably important as mouse features. On the other hand, standard methods, based on impostor-agnostic techniques, treated keyboard events as less important up to the point of having a small subset of features, where there were both keyboard and mouse features. Such a finding would imply that a small subset of features, not importantly if related with keyboard or mouse, is enough to sufficiently distinguish users. Nonetheless, if one aims at the highest possible values of all metrics: *Accuracy, True Positive Rate, False Acceptance Rate, False Rejection Rate*, both carefully selected mouse and keyboard features together with tree-based ensembles seems to be the top-performing option, and the study has shown that even 10 of different features are adequate to build an impostor detection system.

Bibliography

[FS14] Vitaly Ford and Ambareen Siraj. „Applications of machine learning in cyber security". In: *Proceedings of the 27th international conference on computer applications in industry and engineering*. Vol. 118. IEEE Xplore Kota Kinabalu, Malaysia. 2014.

[DSS21] Tapadhir Das, Raj Mani ShuklaT, and Shamik Sengupta. „Imposters among us: a supervised learning approach to anomaly detection in IoT sensor data". In: *2021 IEEE 7th World Forum on Internet of Things (WF-IoT)*. IEEE. 2021, pp. 818–823.

[HCH22] Morey J Haber, Brian Chappell, and Christopher Hills. „Imposter Syndrome". In: *Cloud Attack Vectors: Building Effective Cyber-Defense Strategies to Protect Cloud Resources*. Springer, 2022, pp. 413–415.

[MM23] Marcin Luckner Grzegorz Borowik Maciej Grzenda Stanislaw Kazmierczak and Jacek Mandziuk. „ATO data repository". In: *Evaluation of machine learning methods for impostor detection in web applications*. 2023. URL: https://github.com/mluckner/ATO-data.