Machine Learning Report
Classification of SSH Connection Logs

Leïna Ben Bella
Cholé Daunas
Rania Azmane

Abstract

This report presents a complete machine learning workflow applied to SSH authentication logs. Two predictive tasks were addressed: binary classification of connection types (Normal vs. Suspect) and multiclass classification of failure types extracted from log messages. The project includes preprocessing, feature engineering, baseline modelling, hyperparameter optimisation, ensemble learning, PCA visualisation, correlation analysis, and tree-based interpretability tools such as Decision Trees, Random Forests and Gradient Boosting.

1 Introduction

SSH authentication logs contain a mixture of structured fields and unstructured text, making them challenging to use directly for machine learning. The goal of this project is to transform these logs into a structured dataset that allows accurate classification of connection behaviours. Two learning problems are investigated: (i) distinguishing normal SSH activity from potentially malicious connections, and (ii) predicting the failure category associated with each authentication attempt. These tasks form the basis of log-based anomaly detection systems widely used in cybersecurity.

2 Data Preprocessing

The preprocessing phase aimed to transform the raw log data into a structured and clean dataset suitable for analysis and modeling. Initially, the Time column, when present, was converted to a datetime format, allowing the extraction of the hour of each event into a new column, Hour. This temporal feature enables the computation of metrics such as the number of connections per IP per hour. In cases where the Time column was missing, a default value of -1 was assigned to the Hour column to avoid missing values.

Next, several features were extracted directly from the text content of each log entry. IP addresses were identified using a regular expression and stored in a dedicated column, with missing values replaced by "0.0.0.0". Usernames were similarly extracted by detecting the word following `user`, with unknown entries replaced by "unknown". Port numbers were captured from the text, converted to numeric values, and missing entries replaced by 0. Additionally, a boolean feature was created to indicate whether an event involved pre-authentication, based on the presence of the `[preauth]` tag.

Event classification was then performed to reduce the complexity of the data. Using the EventId column, events were labeled as either normal "N" or suspect "S" (cf documentation with information about each event). This approach reduced the number of categories from 26 to just 2, simplifying subsequent analysis and machine learning tasks.

Finally, unnecessary columns were removed from the dataset. While the original EventId was discarded after classification, the Content column was retained for reference despite being decomposed into more meaningful features. The final dataset thus included `ip`, `user`, `port`, `preauth`, `Hour`, `conn_per_hour`, `connection_type`, and `Content`.

Overall, this preprocessing pipeline effectively converted raw, unstructured log data into a clean, structured, and enriched dataset that supports both descriptive analysis and predictive modeling.

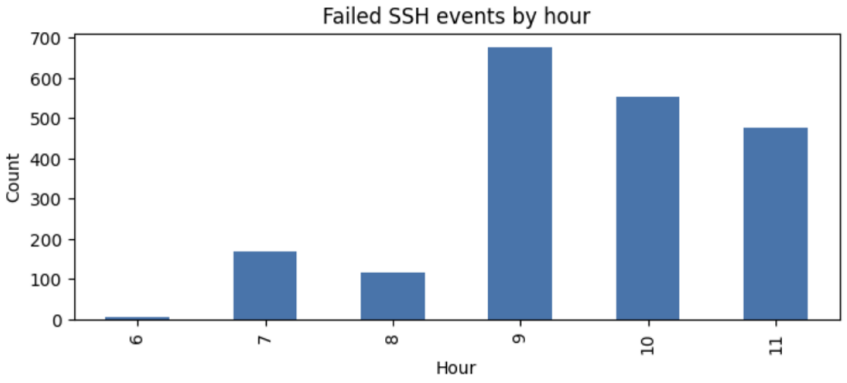
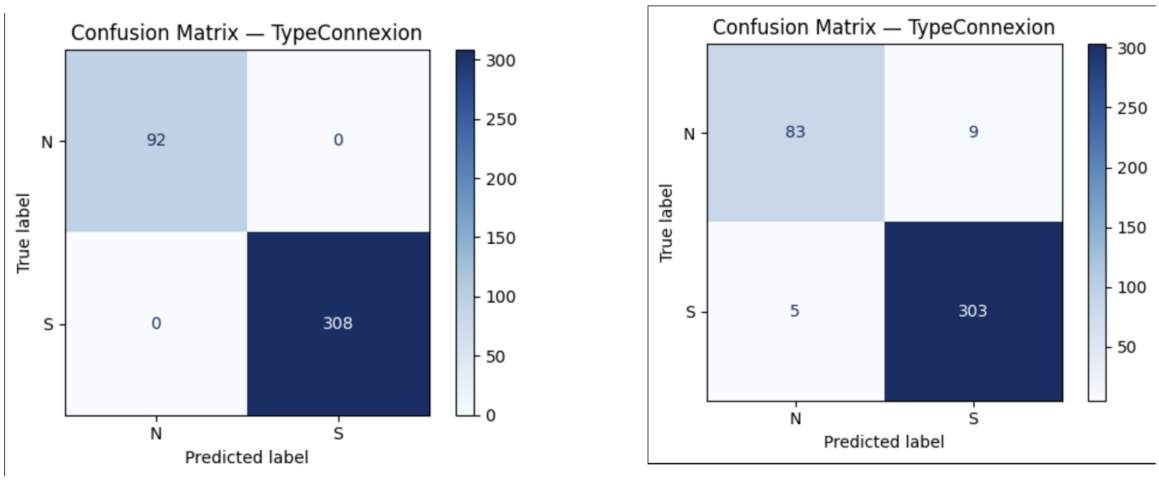


Figure 1: Failed SSH

3 Baseline Models

3.1 Binary Classification: Normal vs. Suspect

A Logistic Regression model was first trained to classify connections as Normal or Suspect. When using all extracted features, including `ip` and `user`, the model achieved extremely high accuracy. However, a closer inspection of these results revealed feature leakage: some usernames and IPs appear exclusively in one class, making prediction artificially easy. Removing these features produced more realistic performance metrics.



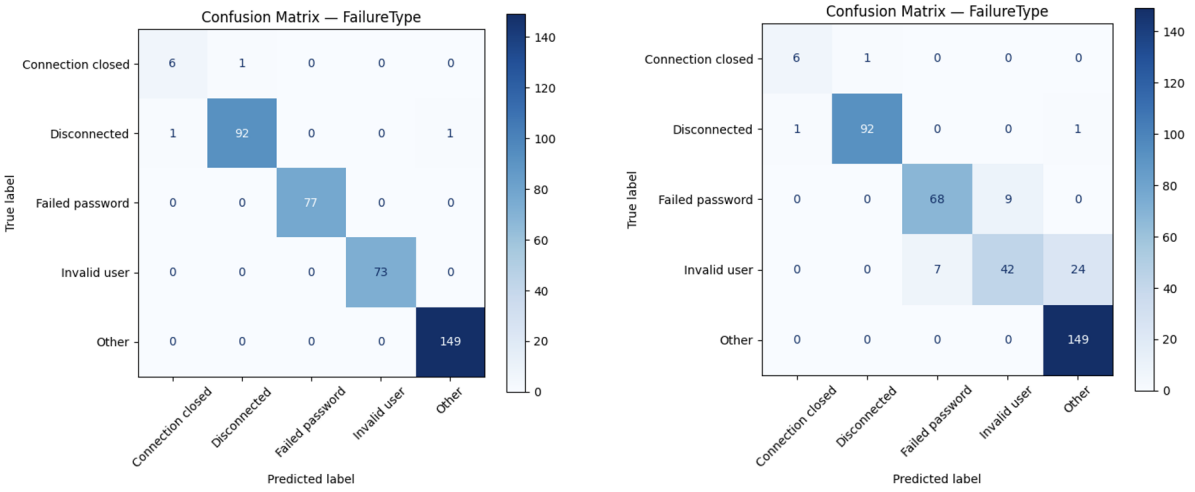
(a) With user and ip (b) Without user and ip

Figure 2: Effect of removing user/ip leakage.

Once identity-based features were removed, the Logistic Regression achieved approximately 96.5% accuracy. The confusion matrix shows that misclassifications occur mainly in borderline cases where behavioural patterns overlap between normal and malicious activity.

3.2 Failure Type Classification

A Linear SVM was trained to predict `FailureType`. Similarly to the previous task, removing the `user` variable improved the fairness of evaluation by eliminating leakage.



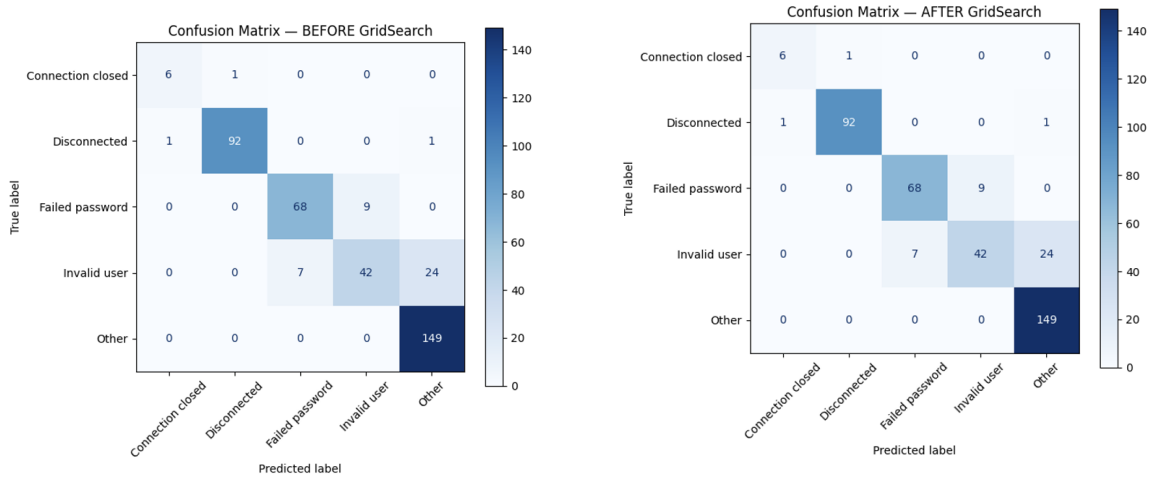
(a) With user (b) Without user

Figure 3: FailureType classification before/after removing user.

The SVM achieved strong performance on the most frequent categories but struggled with rare ones such as Connection closed. Misclassifications often occur between semantically related categories, such as Failed password and Invalid user, reflecting the natural ambiguity present in repeated authentication attempts during attacks.

4 Grid Search Optimization

GridSearchCV was applied to both the FailureType and connection-type mod- els to identify the best hyperparameters using cross-validation. For the FailureType model (Linear SVM), the best parameters were C=1 and max iter=500. The optimization did not improve performance, with overall accuracy remaining at 89%. This is likely due to the small, imbalanced dataset and limited lexical variability. Most classes were already well learned, except "Invalid user", which remains difficult to predict without the user feature.

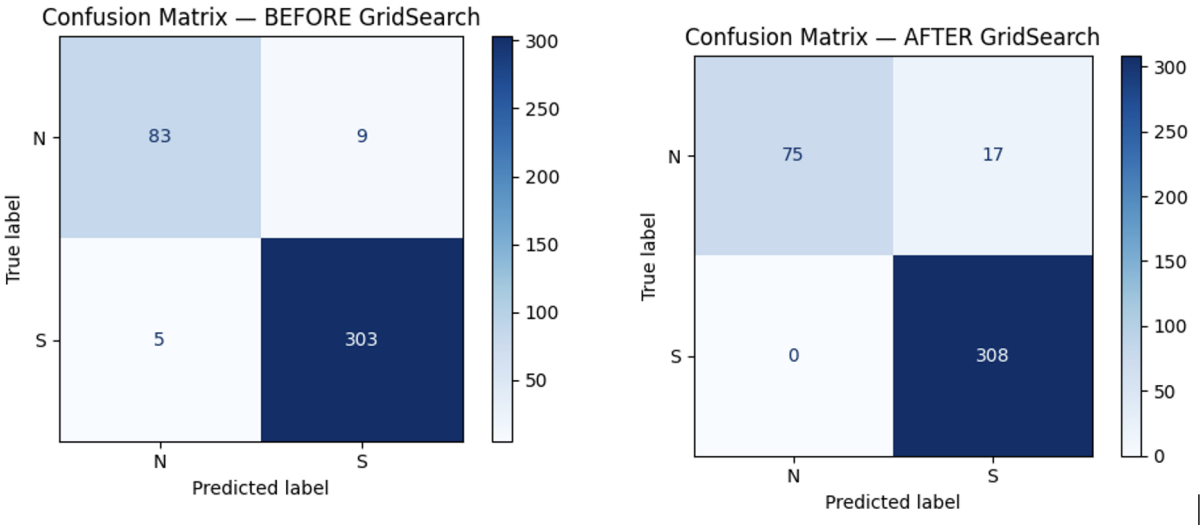


(a) Linear SVM before Grid Search.

(b) Linear SVM after Grid Search.

Figure 4: Confusion matrix BEFORE GridSearchand Confusion matrix AFTER Grid-Search.

For the connection-type model (Logistic Regression), the best parameters were C=0.1 with l2 penalty. After optimization, recall for the suspect class reached 1.0, eliminating false negatives, but precision for the normal class dropped slightly, increasing false positives. Overall, total errors rose from 14 to 17. The optimized model became more sensitive to suspect connections but less precise for normal ones, illustrating a trade-off between class sensitivities.



(a) Logistic Regression before Grid Search.

(b) Logistic Regression after Grid Search.

Figure 5: Confusion matrix BEFORE GridSearchand and Confusion matrix AFTER GridSearch

5 Correlation Heatmap

A correlation heatmap was generated to assess linear relationships among the main numerical and boolean features.

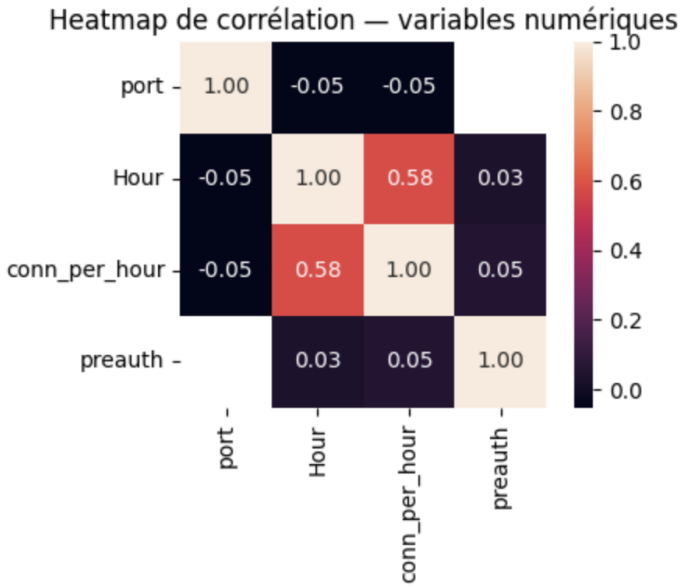


Figure 6: Correlation heatmap of selected features.

The heatmap shows moderate correlation between `Hour` and `conn_per_hour`, which is expected as certain time windows naturally experience more traffic. Other pairwise correlations remain weak, confirming that the engineered features capture complementary aspects of behaviour.

6 PCA Visualisation

To better understand where the classifier makes errors, a PCA was applied to project test samples onto two principal components. This representation helps visualise the structure of the dataset and the location of misclassified points produced by the Voting Classifier.

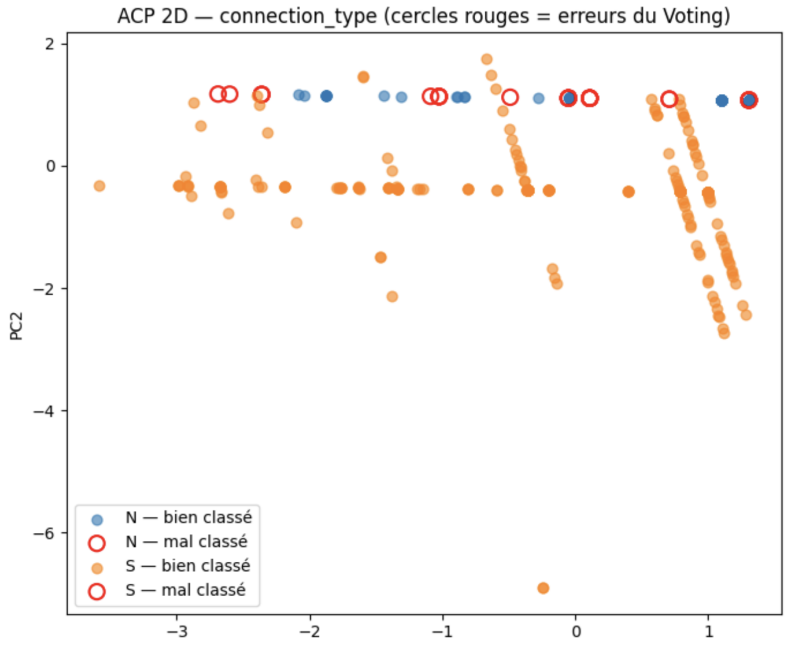


Figure 7: 2D PCA projection with misclassifications highlighted.

The PCA scatterplot reveals that Normal and Suspect points form partially overlapping clusters. Misclassifications (red-circled points) tend to occur near the overlap region, indicating intrinsic ambiguity in those samples rather than model deficiency.

7 Decision Tree Interpretation

A Decision Tree with depth limited to three was trained to obtain an interpretable model. The tree exposes the hierarchical decision logic and reveals which features have the strongest discriminative power.

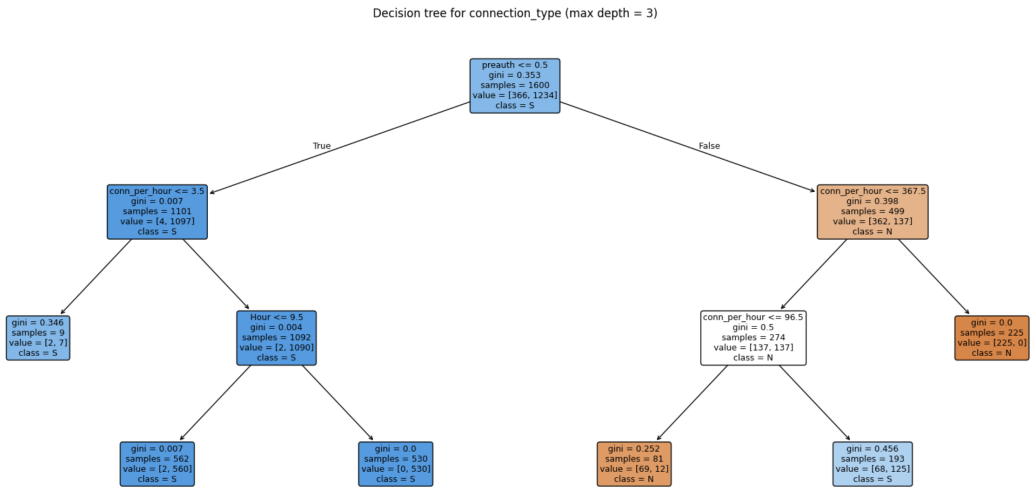


Figure 8: Decision tree for `connection_type` (max depth = 3).

The tree shows that the most important feature is `conn_per_hour`, which is responsible for several high-purity splits distinguishing Suspect from Normal connections. The `preauth` flag appears at the root, highlighting its strong association with suspicious behaviour. The `Hour` feature provides additional refinement only in specific subregions. Despite its shallow depth, the model reaches 95.8% accuracy, confirming that the core behavioural features are highly predictive and easy to interpret.

8 Random Forest and Gradient Boosting

Ensemble tree-based models were also trained for comparison. The Random Forest classifier stabilises predictions by aggregating many decision trees trained on bootstrapped samples, while Gradient Boosting sequentially corrects previous errors.

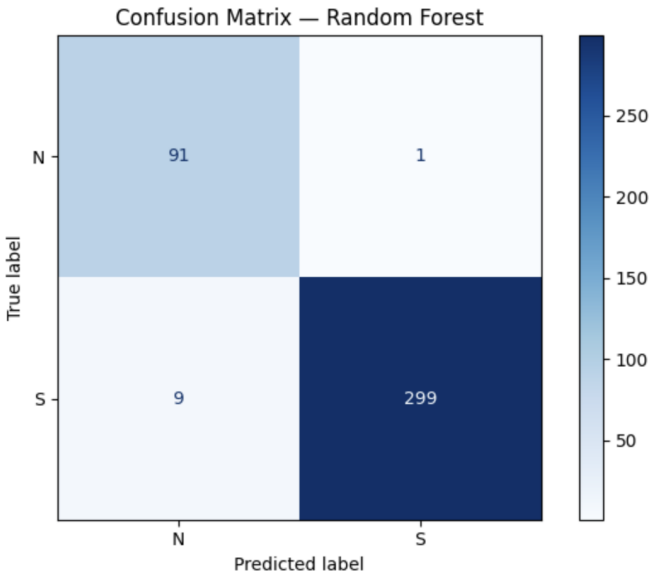


Figure 9: Random Forest confusion matrix.

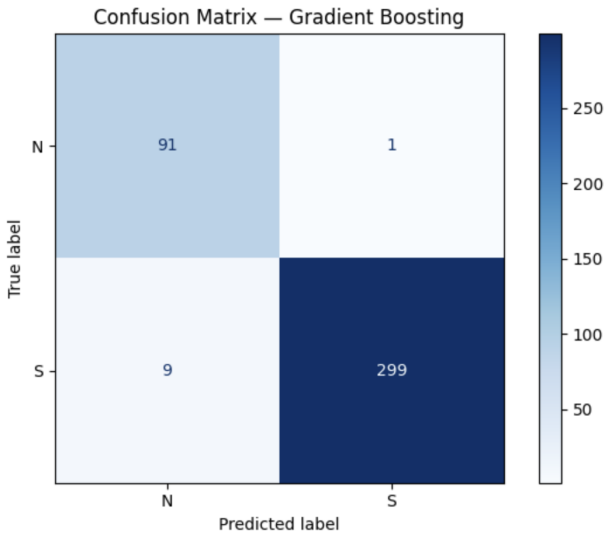
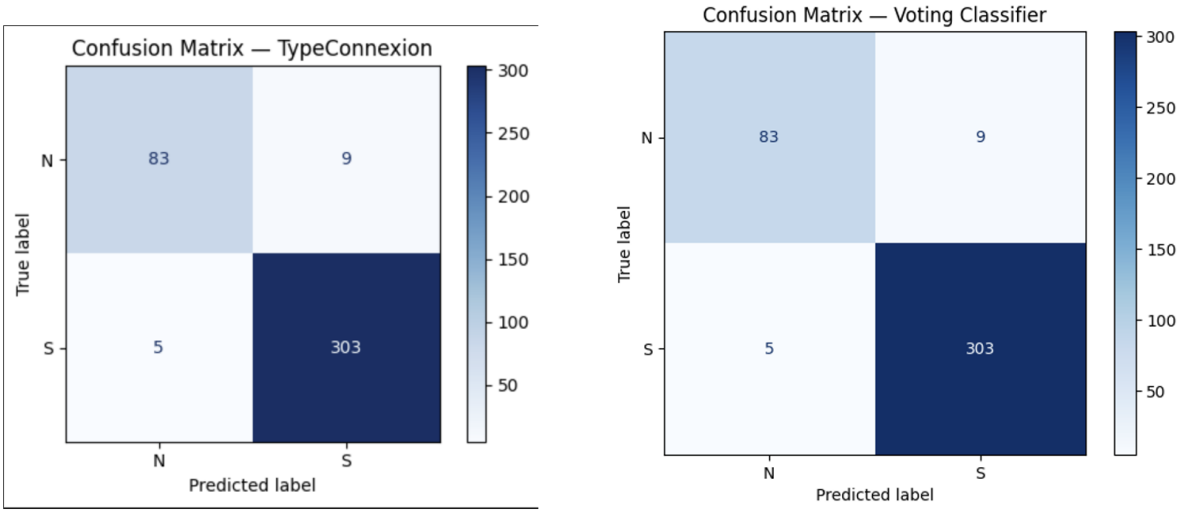


Figure 10: Gradient Boosting confusion matrix.

Both methods achieve similar performance to the logistic baseline, with Boosting offering slightly improved recall in certain cases. This similarity suggests that the engineered features already capture the essential signal, leaving limited room for non-linear models to extract additional structure.

9 Ensemble Modeling with Voting and Bagging

To improve connection type prediction, a Voting Classifier combining a Logis- tic Regression and a Linear SVM was used, with each base model wrapped in a BaggingClassifier. Bagging reduces variance by training multiple estimators on random subsets of the data, while Voting aggregates predictions from the different models. The ensemble achieved the same results as the baseline Logistic Regression, with 96.5% accuracy and similar precision and recall for both classes. This is expected because the dataset is relatively small and simple, and the features are already highly discriminative, making it difficult to improve performance. Furthermore, Logistic Regression is already a stable classifier, so applying Bag- ging does not significantly increase accuracy—it mainly helps reduce variance and improve robustness.



(a) Confusion matrix TypeConnexion

(b) Logistic Regression after Grid Search.

Figure 11: Confusion matrix TypeConnexion and Confusion matrix Voting Classifierh

10 Conclusion

This project demonstrates the full pipeline from raw log processing to advanced machine learning modelling. The engineered features—particularly `conn_per_hour`, `preauth` and time-derived variables—capture essential aspects of SSH connection behaviour. Linear models achieve excellent performance, and tree-based models provide interpretability without sacrificing accuracy. Ensemble methods confirm the robustness of the results. Future work could explore sequential models, anomaly detection frameworks, or transformer-based architectures capable of modelling temporal dependencies in log streams.