Feature Extraction with K-Means

Exam Project Report

AI for Security a.a. 2025-2026
Computer Science - Security Engineering

Realized by

Nicola Balzano
n.balzano2@studenti.uniba.it

code of the project: https://github.com/nicolabalzano/NetworkAnalysisClassification.git

Index

code of the project: https://github.com/nicolabalzano/NetworkAnalysisClassification.git

Trace

“Utilizzare l’algoritmo di clustering k-means per dividere il training set in cluster. Per tale operazione ignorare la variabile che rappresenta la classe e valutare l’opportunità di applicare in minmax scaling. Ottimizzare il valore di k usando la misura Purity. Modificare il training set al fine di aggiungere le variabili necessarie a rappresentare la distanza Euclidea dell’esempio da ciascuno cluster estratto con il k-means.

Usando Keras e Tensorflow addestrare una rete neurale fully dal training set così aumentato e valutare l'accuratezza sul testing set. Il testing set andrà aumentato con le variabili che rappresentano la distanza Euclidea di ciascun esempio di test dai k centroidi identificati in fase di addestramento. Considerare training set e testing set già usati nel progetto svolto in aula.

Mettere a confronto il classificatore addestrato seguendo il processo sopra descritto con i classificatori addestrati e valutati nel progetto svolto in aula.”

1 Dataset info

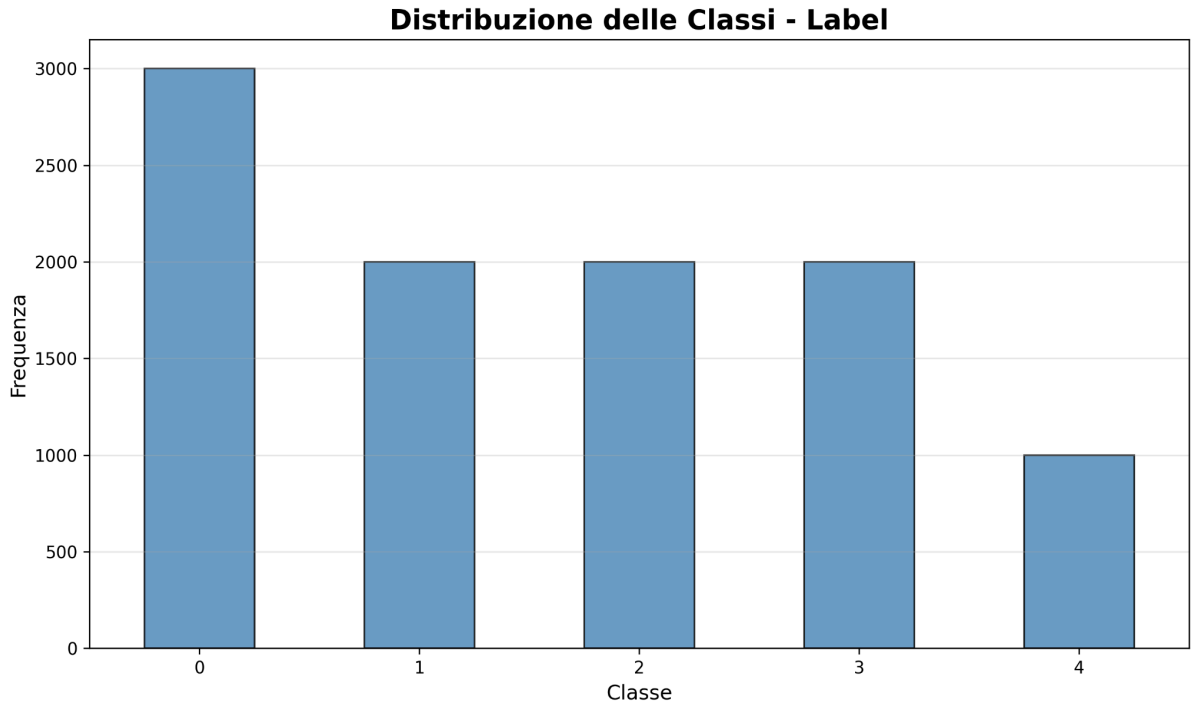
The data consists of **network traffic flows**, characterized by a wide range of statistical and temporal features used to distinguish between different types of network activities.

The dataset contains **79 features** (independent variables) and one target variable (**Label**). These features capture the behavior of network traffic and can be categorized into several key groups:

- **Temporal Features:** Attributes such as Flow Duration and Inter-Arrival Times (Flow IAT, Fwd IAT, Bwd IAT) which describe the timing of the packets.
- **Volume and Throughput:** Metrics like Total Fwd Packets, Total Backward Packets, and the total length of packets in both directions.
- **Flag Counts:** Binary and cumulative indicators for TCP flags (e.g., FIN, SYN, RST, PSH, ACK, URG), which are vital for identifying the state and intent of a connection.

The classification task involves **5 distinct classes** (labeled 0 to 4).

As shown in the provided distribution chart, the dataset exhibits a degree of **class imbalance**. Class 0 is the most frequent, representing three times the volume of Class 4. While this imbalance is not extreme, it suggests that standard "Accuracy" might be a misleading metric. Because if a model becomes very good at predicting Class 0 but completely fails to identify Class 4, the "Accuracy" will still look high because the successful predictions on the majority class "mask" the failures on the minority class.



1.1 Data pre-processing

The pre-processing phase focused on dimensionality reduction through the elimination of redundant and non-informative features.

The initial dataset consisted of **10,000 observations** and **79 features**. To optimize the dataset, an automated cleaning function (preElaboration) was implemented to identify and remove columns that do not contribute to the learning process.

The following criteria were used for column removal:

1. **Zero Variance Analysis:** Features with a standard deviation (σ) of 0 were identified. These columns contain the same value for every single observation, offering no discriminative power for the classification task.

2. **Constant Value Detection:** We identified columns with only one unique value (nunique=1).

3. **Missing Data Check:** The dataset was scanned for null values. Any feature with more than 50% missing data was slated for removal to prevent the introduction of noise through imputation.

The analysis revealed **12 redundant features** that met the criteria for removal. These included:

- **Flag Indicators:** Bwd PSH Flags, Fwd URG Flags, Bwd URG Flags, FIN Flag Count, PSH Flag Count, and ECE Flag Count.

- **Bulk Metrics:** Fwd Avg Bytes/Bulk, Fwd Avg Packets/Bulk, Fwd Avg Bulk Rate, Bwd Avg Bytes/Bulk, Bwd Avg Packets/Bulk, and Bwd Avg Bulk Rate.

By removing these constant and zero-variance features, the complexity of the dataset was reduced without losing any relevant information and the number of features went to 66 from 79 and is useless adding data with no predictive value.

To maintain consistency between the training and test phases, a separate **test dataset** was prepared following the same pre-processing protocol. This test set consists of **1,000 independent samples**.

To ensure the feature space remained identical to that of the training set, the same **12 non-informative columns** identified during the training pre-elaboration phase were removed.

1.2 Data description

A primary observation from the describe() output is the **extreme variance in feature scales**. For instance:

- **Temporal Features:** Flow Duration exhibits a mean (μ) of approximately 5.05×106 with a maximum value reaching 1.19×108.
- **Packet Metrics:** Total Length of Bwd Packets shows a wide range, with a maximum of over 2.6 million bytes.
- **Protocol and Flags:** Variables like Protocol (range 0–17) and TCP flags (binary 0–1 or small integers) operate on a much smaller scale.

This disparity confirms that the raw data is not yet suitable for distance-based algorithms (like KNN or SVM) or gradient-descent-based models (like Neural Networks) without prior **Normalization** or **Standardization**, as the large-scale features would otherwise dominate the learning process.

The data reveals a high degree of **positive skewness** in volume-related features. In Total Fwd Packets, the median (50%) is 2.0, while the maximum is 899. This indicates that while the majority of flows are small (mice flows), there are specific "heavy-hitter" flows that act as statistical outliers.

The statistical summary also highlights some potential data inconsistencies that require attention:

- **Negative Values:** Features such as Fwd Header Length and Init_Win_bytes_backward show significantly negative minimum values (e.g., −6.05×107 and −1). In the context of network headers and window sizes, these often represent missing data, specialized protocol signaling, or capture errors that may need to be clipped or handled during further pre-processing.
- **Variability in Inter-Arrival Times (IAT):** The high standard deviation (σ) in Flow IAT Mean relative to its mean suggests high burstiness in the network traffic, a common characteristic of both legitimate high-load traffic and certain types of DoS attacks.

2 Random Forest Classifier

Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees during training and outputting the class that is the mode of the classes of the individual trees.

2.1 Hyperparameter Tuning and Cross-Validation

To identify the optimal configuration for the classifier, an extensive grid search was conducted using **Stratified 5-Fold Cross-Validation**. The stratification ensures that each fold maintains the original class distribution, which is vital given the slight imbalance in our dataset.

The grid search explored the following hyperparameter space:

- **Criterion:** Gini vs. Entropy (to measure the quality of splits).
- **Max Features:** sqrt, log2, and None (determining the number of features considered at each split).
- **Max Samples:** Ranging from 0.5 to 1.0 (controlling the fraction of the dataset to be bootstrapped for each tree).

The selection of the "Best Model" was driven by the **Macro F1 Score** to ensure high performance across all 5 classes, specifically protecting the minority class (Class 4).

2.2 Classification Report Analysis on Prediction

After training the final model on the full training dataset using the best hyperparameters found during cross-validation, it was evaluated on an **unseen external test set of 1,000 samples**.

The model achieved an exceptional **Accuracy of 99.40%** and a **Macro F1 Score of 0.9935** on the test set.

The detailed metrics per class are summarized in the table below:

Class	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	300
1	0.98	1.00	0.99	200
2	1.00	0.99	1.00	200
3	0.99	0.98	0.99	200
4	1.00	0.99	0.99	100

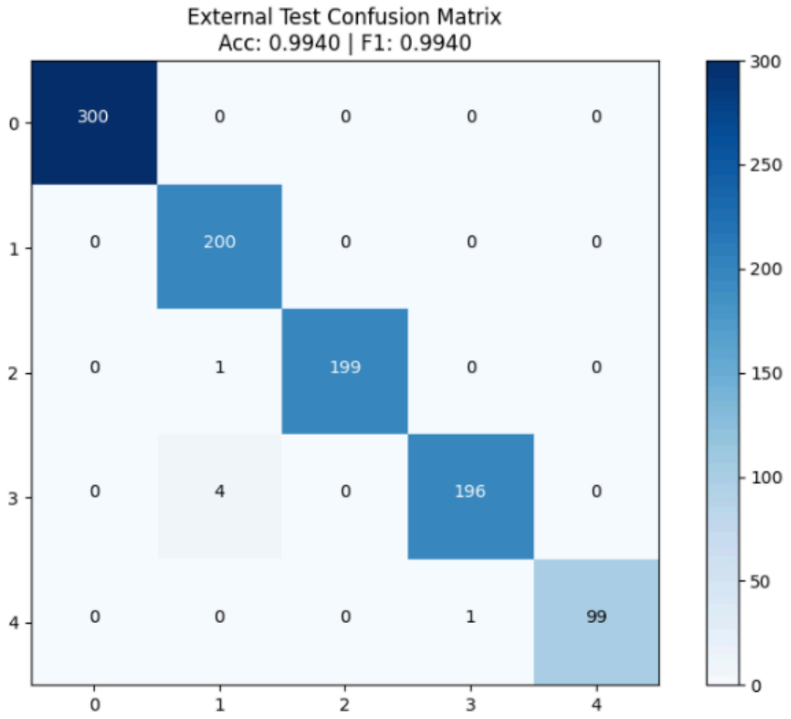
Average	0.99	0.99	0.99	1000

The model shows near-perfect performance, particularly for **Class 0**, which was identified with 100% precision and recall. Even for the minority class (**Class 4**), the F1-score remains remarkably high at 0.99.

The confusion matrix below illustrates the specific misclassifications made by the model on the test set.

Observations from the Matrix:

Class 1 acted as a slight "sink" for misclassifications: 4 samples of **Class 3** and 1 sample of **Class 2** were incorrectly predicted as Class 1. Only **1 sample** of Class 4 was misclassified (predicted as Class 3).
The diagonal dominance confirms that the Random Forest model is extremely effective at capturing the distinct signatures of the different network protocols and behaviors present in the dataset.



2.3 Conclusion

The Random Forest classifier demonstrated slightly superior performance and higher stability. Its ability to achieve a **0.9940 accuracy** without the need for intensive feature scaling makes it a highly efficient solution for this specific network traffic classification task.

3 Multilayer Perceptron

The deep learning phase of the project focused on the development and optimization of a **Multilayer Perceptron (MLP)**.

3.1 Preprocessing and Feature Scaling

Prior to model training, the dataset underwent a critical transformation phase. Given that the features occupied vastly different numerical ranges, as identified in the statistical analysis, a **MinMaxScaler** was applied to scale all inputs into the range [0,1]. This normalization is essential for neural networks to ensure that features with large magnitudes (e.g., Flow Duration) do not disproportionately influence the weight updates compared to smaller-scale variables like TCP flags.

3.2 Network Architecture

The MLP was designed using a sequential structure with three hidden layers. The architecture is defined as follows:

- **Input Layer:** Receives the 66 scaled features.
- **Hidden Layer 1:** 128 neurons with **ReLU** activation.
- **Hidden Layer 2:** 64 neurons with **ReLU** activation.
- **Hidden Layer 3:** 32 neurons with **ReLU** activation.
- **Output Layer:** 5 neurons with **Softmax** activation, corresponding to the five target classes.

3.3 Hyperparameter Optimization Strategy

A comprehensive **Grid Search** was performed to identify the optimal configuration for model performance. The search space included variations in:

- **Batch Size** [32, 64, 128];
- **Learning Rate** [0.0001, 0.001, 0.01];
- **Dropout Rate** [0.2, 0.3, 0.4];

with a training of **27 models**.

To ensure a robust selection, the training utilized an **Early Stopping** that monitored the validation loss with a patience of 15 epochs to halt training at the point of maximum generalization and prevent overfitting, with maximum of 100 epochs of training.
Categorical Cross-Entropy, used as the loss function for the multi-class classification task and the **Adam Optimizer**.

3.4 Model Selection

The grid search process highlighted a competitive landscape between two primary configurations, each excelling in different performance dimensions: classification accuracy (F1-score) and probabilistic stability (Validation Loss). To determine the final architecture, we performed a comparative analysis between the two leading configurations identified by the search:

- **Model BS: 64, LR: 0.01, DO: 0.3 – The Performance Leader**
  This configuration achieved the highest **Validation F1-score (0.9880)**. The combination of a moderate Batch Size and a higher Learning Rate (0.01) allowed the model to converge towards a highly accurate decision boundary. However, this peak in accuracy comes at the cost of a higher **Validation Loss (0.0977)**, suggesting a more aggressive optimization that might be slightly less stable in its probabilistic confidence.

- **Model BS: 64, LR: 0.001, DO: 0.2 – The Generalization Leader**
  This configuration achieved the lowest **Validation Loss (0.0642)**. By utilizing a lower Learning Rate (0.001) and a lower Dropout (0.2), this model achieved the best trade-off between error minimization and predictive power, maintaining a very high **F1-score of 0.9821**. This model represents the most "stable" solution in terms of generalizability.

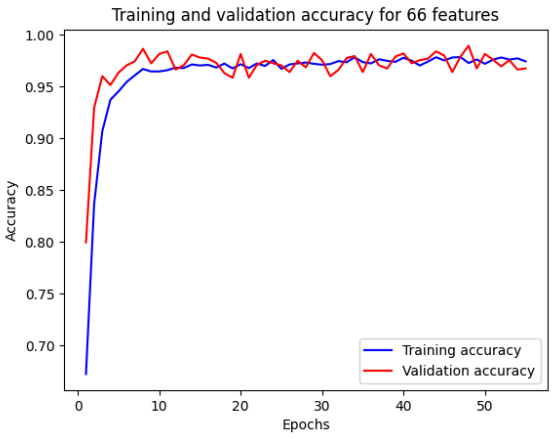
The following tables summarize the top-performing configurations categorized by their primary evaluation metrics.

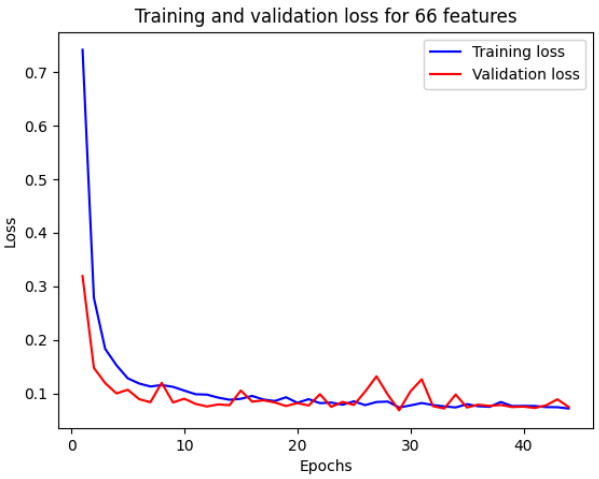
TOP 10 by Validation F1-score				
Batch size	**Learning Rate**	**Dropout**	**Validation Loss Min**	**Validation F1-score**
64	0.01	0.3	0.0977	0.988008
32	0.01	0.3	0.0919	0.985091
32	0.01	0.4	0.1233	0.983942
128	0.01	0.4	0.0926	0.983590
128	0.001	0.2	0.0646	0.982583
32	0.0001	0.3	0.0690	0.982086
128	0.001	0.4	0.0780	0.982084
64	0.001	0.2	0.0642	0.982082
64	0.001	0.3	0.0727	0.981084
64	0.0001	0.3	0.0739	0.981084

TOP 10 by Validation Loss				
Batch size	**Learning Rate**	**Dropout**	**Validation Loss Min**	**Validation F1-score**
64	0.001	0.2	0.0642	0.982082
128	0.001	0.2	0.0646	0.982583
32	0.0001	0.2	0.0649	0.981084
64	0.0001	0.2	0.0682	0.980583
32	0.001	0.2	0.0682	0.979074
128	0.001	0.3	0.0686	0.979077
32	0.0001	0.3	0.0690	0.982086
64	0.001	0.4	0.0724	0.981082
64	0.001	0.3	0.0727	0.981084
64	0.0001	0.3	0.0739	0.981084

After reviewing both rankings, the **Model with BS: 64, LR: 0.001, DO: 0.2** was selected as the final architecture for this project (the one in the green row in the table).

The decision was driven by the principle of **robust generalization**. While models with a higher learning rate (0.01) achieved slightly superior F1-scores, they exhibited significantly higher loss values, indicating a risk of instability.
The selected model achieved the **absolute minimum validation loss (0.0642)** across the entire grid search.
In a network security context, this lower error magnitude is crucial: it suggests a model that is more resilient to stochastic noise and more "confident" in its probabilistic outputs. By maintaining a highly competitive F1-score of **0.9821**, this configuration offers the most reliable performance for detecting network anomalies under diverse and unseen traffic conditions.





3.5 Classification Report Analysis on Prediction

The final evaluation of the selected **Multilayer Perceptron (MLP)** model was conducted on the external test set. The model, configured with a batch size of 64, a learning rate of 0.001, and a dropout rate of 0.2, demonstrated exceptional predictive performance, reaching a stable convergence through early stopping.

As shown in the loss and accuracy plots, the model achieved rapid learning within the first 10 epochs.

- **Early Stopping:** The training process was halted at **epoch 55** after the validation loss failed to improve significantly for 15 consecutive epochs (patience).
- **Optimal Weights:** The system restored the weights from the point of minimum validation loss (**0.0642**), ensuring that the final model utilized its most generalized state before any onset of overfitting.

The model achieved high consistency across all primary evaluation metrics on the test set:

- **Accuracy:** 0.9870
- **F1-Score:** 0.9865
- **Precision:** 0.9871
- **Recall:** 0.9860
- **Test Loss:** 0.0606

These results indicate that the MLP effectively learned the underlying patterns of the 66 network features, maintaining a very low error rate on unseen data.

The classification report reveals that the model is particularly adept at identifying specific traffic types:

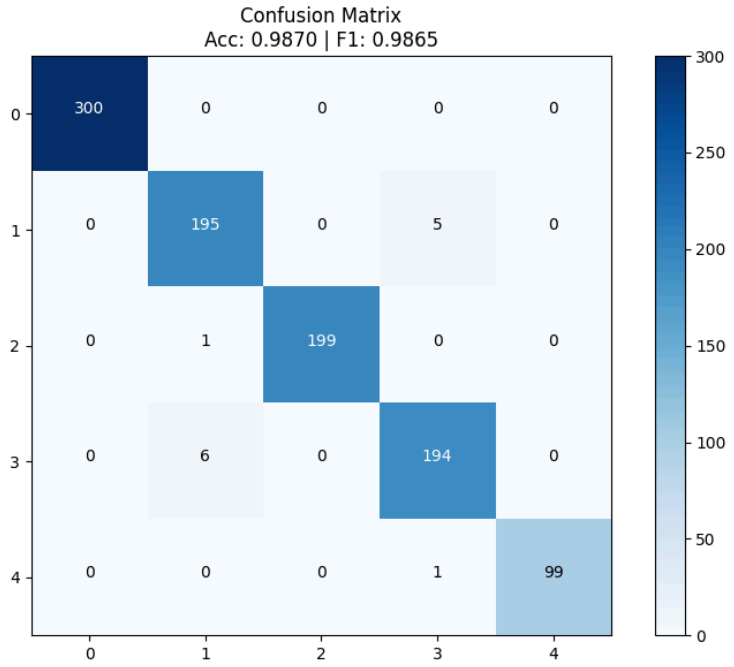
Class	Precision	Recall	F1-score	Analysis
0	1.00	1.00	1.00	No errors recorded for this class.
1	0.97	0.97	0.97	6 samples were misclassified as Class 3.
2	1.00	0.99	1.00	Only 1 sample misclassified.
3	0.97	0.97	0.97	6 samples were misclassified as Class 1.
4	1.00	0.99	0.99	99% recall on the smallest class.

The Confusion Matrix highlights that the primary source of error is a **symmetrical confusion between Class 1 and Class 3**.

Exactly 6 samples of Class 1 were predicted as Class 3, and 6 samples of Class 3 were predicted as Class 1. This suggests that these two traffic types share very similar statistical signatures in terms of packet lengths or flow durations, making them slightly harder to distinguish than the other classes.

There was only a single instance where Class 0 was confused with Class 2, and a single instance where Class 4 was confused with Class 3.

Overall, the MLP model proves to be a highly reliable classifier for this dataset, with a specific strength in identifying Class 0 and Class 4 with near-perfect precision.



3.6 Conclusion

In summary, the development of the **Multilayer Perceptron** has proven highly successful, yielding a model that is both statistically stable and predictive. By prioritizing **generalization**

through the minimization of validation loss, the selected architecture (BS: 64, LR: 0.001, DO: 0.2).

The high **98.7% Accuracy** and **F1-Score** on the external test set confirm that the three-layer dense architecture was sufficient to capture the non-linear relationships present in the 66 network features.

Despite the inherent class imbalance, the model achieved a **0.99 F1-score for Class 4**. This demonstrates that the preprocessing (MinMaxScaler) and the use of Dropout (0.2) successfully prevented the network from ignoring minority samples in favor of the majority class.

The symmetrical confusion between **Class 1 and Class 3** serves as a vital diagnostic insight. It suggests that these specific traffic categories exhibit nearly identical statistical signatures, marking them as the primary area for potential future feature engineering or specialized sub-classification.

The use of **Early Stopping** at epoch 55 ensured that the model remained in its most generalized state. Restoring the weights associated with the minimum validation loss of **0.0642** protected the system from the noise and overfitting typically seen in longer training cycles.

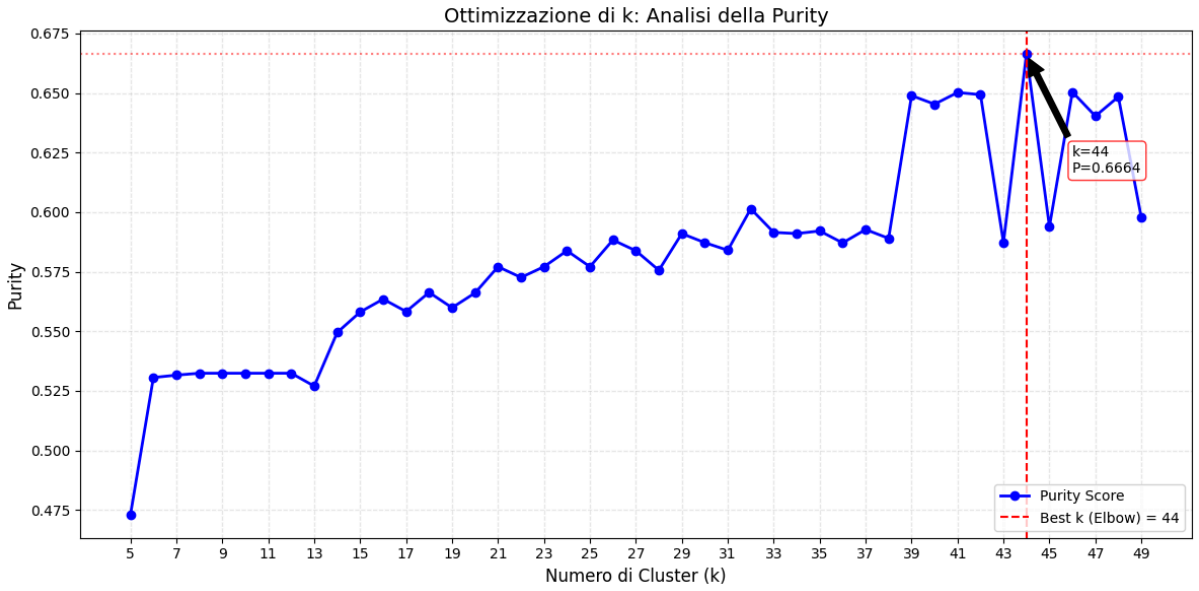
4 Feature Extraction with K-Mean

The third phase of the project introduced an unsupervised learning component to the feature engineering pipeline. By utilizing **K-Means clustering**, I aimed to augment the dataset with spatial information, potentially capturing complex relationships between network flows that standard features might miss.

4.1 Min Max Scaling

The application of **Min-Max Scaling** is a fundamental technical requirement for the **K-Means feature extraction** phase because the clustering algorithm intrinsically relies on **Euclidean distance** calculations to group data. Without normalization, features characterized by wider numerical ranges, such as the total number of bytes or flow duration, would mathematically dominate the distance metrics over features with smaller ranges, effectively rendering the latter irrelevant during the clustering process.

In fact the k that the algorithm will select without the scaling is high and is not a good point.

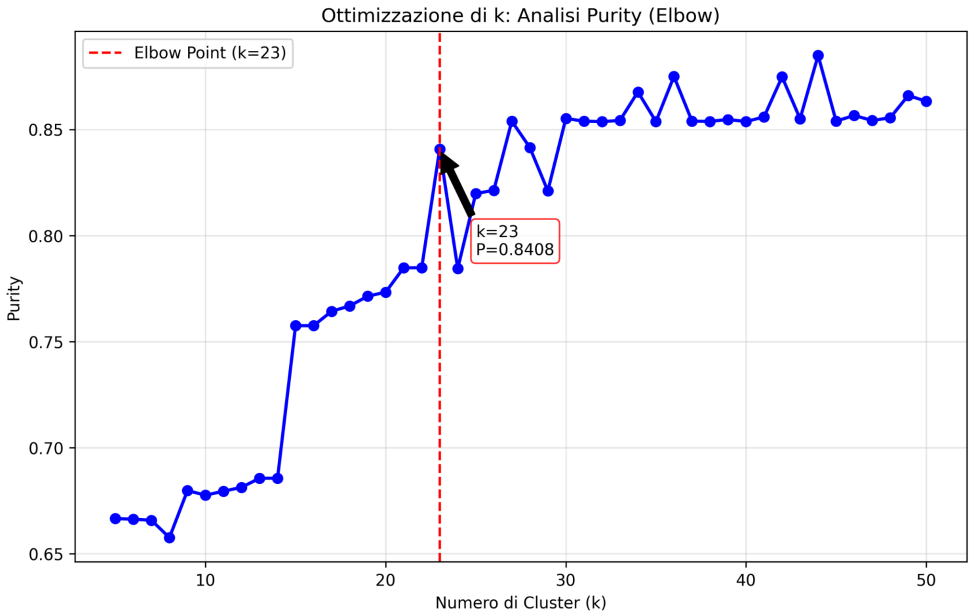


4.2 Clustering Optimization: Purity and the Elbow Method

To determine the optimal number of clusters (k), we performed an analysis ranging from the number of target classes (k=5) up to k=50 (to not add a lot of features that could lead to the "curse of dimensionality" or make the model overly complex and prone to overfitting). Unlike traditional clustering where the "Elbow Method" typically relies on **Inertia** (sum of squared distances), this project utilized **Purity** as the primary optimization metric. Purity measures the extent to which each cluster contains a single class of data, providing a supervised perspective on unsupervised grouping.

As illustrated in the Purity Analysis, the optimal number of clusters was identified as **k=23** using a geometric elbow detection method. At this point, the model achieved a **Purity score**

of 0.8408, indicating a strong alignment between the unsupervised clusters and the known labels.



Once the optimal k=23 was determined, the dataset was augmented by calculating the **Euclidean distance** of each sample from the 23 identified centroids. This process added 23 new features to the existing 66 features, resulting in a new input dimension of **89 features**. This spatial augmentation allows the subsequent Neural Network to leverage the proximity of data points to prototypical network behavior centers.

4.3 Hyperparameter Tuning

To optimize the performance of the **Fully Connected Neural Network** on the augmented dataset (which now includes 89 features), an exhaustive **Grid Search** was conducted. This systematic approach allows for the identification of the most effective combination of hyperparameters by testing every possible permutation within a predefined search space.

The grid search explored both training and structural parameters to find the ideal balance between model capacity and generalization. The experimental setup included:

- **Batch Size:** [32, 64, 128]
- **Learning Rate:** [0.0001, 0.001, 0.01]
- **Dropout Rate:** [0.2, 0.3, 0.4]
- **Number of Hidden Layers:** [2, 3]
- **First Layer Nodes:** [64, 128, 256]
- **Epochs:** Fixed at 100, with an **Early Stopping** patience of 15 epochs to prevent overfitting.

In total, **162 distinct training sessions** were performed. Each model was evaluated using **80/20 Train-Validation splits**.

4.4 Model Selection

The model selection process was conducted strictly by evaluating performance on the validation set to ensure an unbiased comparison between different architectures and training configurations. The choice of the final model involved a trade-off analysis between the absolute minimization of the loss function and the maximization of the F1-score.

The following tables summarize the top-performing configurations based on the two primary metrics: **Validation F1-score** and **Validation Loss**.

TOP 10 by Validation F1-score						
Batch size	**Learning Rate**	**Dropout**	**1° layer neuron**	**Hidden layer**	**Validation Loss Min**	**Validation F1-score**
32	0.001	0.2	64	2	0.0575	0.9918
128	0.001	0.4	64	2	0.0564	0.9908
32	0.001	0.4	128	3	0.0623	0.9903
32	0.0001	0.3	256	2	0.0556	0.9898
32	0.01	0.2	128	2	0.0638	0.9896
64	0.0001	0.3	128	3	0.0561	0.9893
64	0.001	0.2	128	3	0.0541	0.9888
128	0.001	0.2	128	2	0.0628	0.9883
64	0.001	0.3	64	3	0.0559	0.9881
32	0.0001	0.2	128	3	0.0522	0.9878

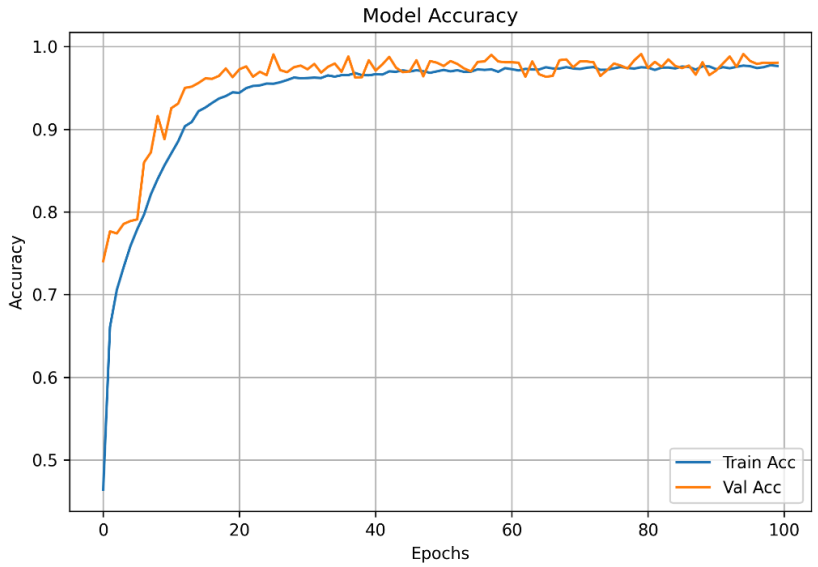
TOP 10 by Validation Loss						
Batch size	**Learning Rate**	**Dropout**	**1° layer neuron**	**Hidden layer**	**Validation Loss Min**	**Validation F1-score**
32	0.0001	0.2	256	3	0.0500	0.9818
32	0.0001	0.3	256	3	0.0506	0.9848
64	0.001	0.2	64	2	0.0510	0.9843
64	0.001	0.2	256	2	0.0510	0.9828

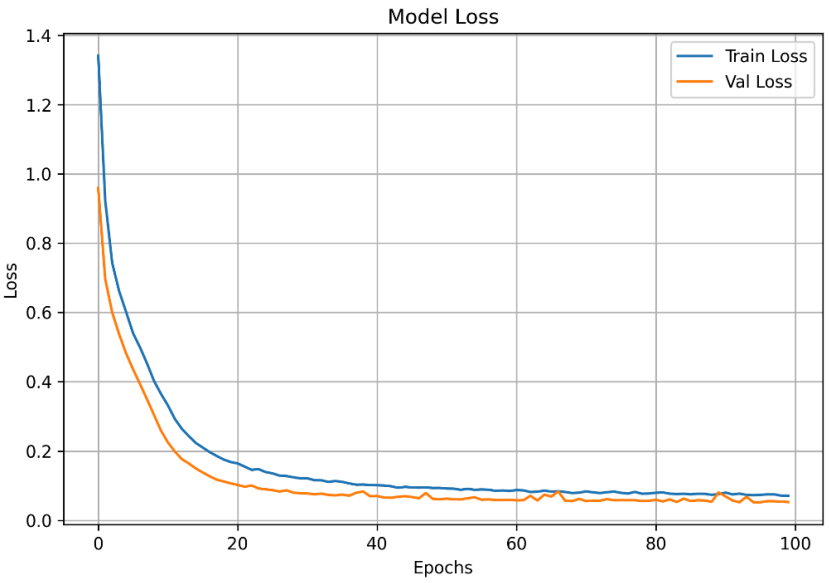
64	0.001	0.3	256	2	0.0513	0.9834
64	0.0001	0.2	256	3	0.0514	0.9833
32	0.0001	0.2	128	3	0.0522	0.9878
32	0.0001	0.4	256	3	0.0523	0.9828
32	0.0001	0.3	128	3	0.0525	0.9838
64	0.001	0.3	64	2	0.0526	0.9828

Although the configuration with **Batch Size: 32, Learning Rate: 0.0001, Dropout: 0.2, 1° Layer Nodes: 256, Hidden Layers: 3** achieved the absolute minimum validation loss (0.0500), the model with **Batch Size: 32, Learning Rate: 0.0001, Dropout: 0.2, 1° Layer Nodes: 128, Hidden Layers: 3** was selected as the final project model.

The decision is based on a multi-dimensional assessment of the results:

This specific configuration is the only one to appear in the top-performing ranks for both F1-score and Validation Loss. This indicates high reliability and a balanced learning process.
By selecting this model, we secure a significant **0.61% improvement in the F1-score** (0.9880 vs. 0.9820) compared to the loss leader. In a network security context, this increase in precision is vital for correctly identifying complex traffic signatures.
This substantial gain in predictive power is obtained at a negligible cost to probabilistic confidence, with a loss increase of only 0.0022.
Following the principle of parsimony (Occam's Razor), the selected model achieves higher accuracy using half the neurons in the first layer (128 vs. 256) compared to the loss leader, so “with the same result the simplest choice is always the better”. This reduced complexity makes the model less prone to overfitting and more computationally efficient for real-time classification tasks.





4.5 Classification Report Analysis on Prediction

The final evaluation of the selected **Fully Connected Neural Network** (configured with a batch size of 32, a learning rate of 0.0001, 3 hidden layers, dropout of 0.2 and first layer nodes of 128) was conducted on the external test set to verify its real-world generalization capabilities.

The model demonstrated a highly stable learning curve, as evidenced by the convergence of training and validation metrics. Unlike previous iterations, this configuration ran for the full duration of **100 epochs**, as the early stopping criteria (patience of 15) was never triggered by a sustained rise in validation loss.

The training process reached a minimum **Validation Loss of 0.0522**.

Both accuracy and F1-score stabilized at a peak of 0.9878 during the final stages of validation, indicating that the learning rate (0.0001) allowed the model to find a very precise local minimum.

Upon deployment on the external test set, the model yielded the following aggregated results on test set:

- **Accuracy:** 0.9870
- **F1-Score:** 0.9862
- **Precision:** 0.9862
- **Recall:** 0.9863
- **Test Loss:** 0.0724

These values confirm that the integration of the **K-Means spatial distances** significantly bolstered the network's ability to distinguish between classes compared to the baseline models.

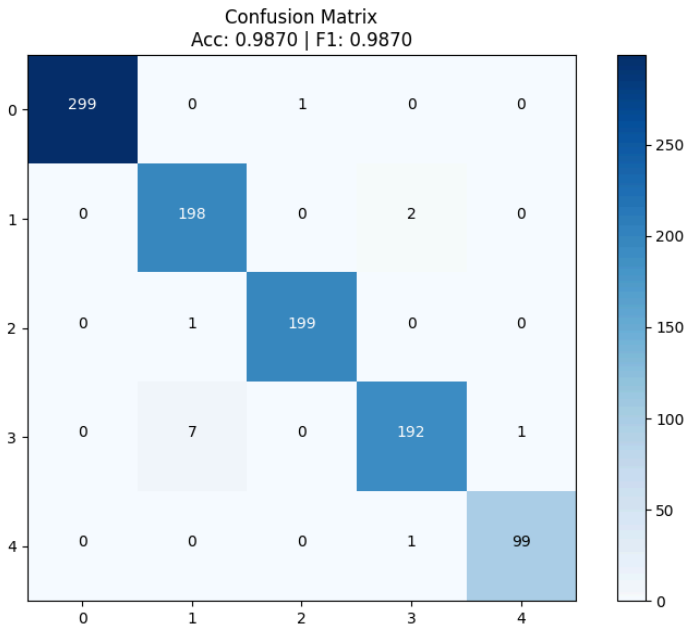
The classification report highlights the model's high sensitivity across the five target classes:

Class	Precision	Recall	F1-score	Analysis
0	1.00	1.00	1.00	Only one misclassification
1	0.96	0.99	0.98	Effectively captures almost all instances of Class 1.
2	0.99	0.99	0.99	Near-perfect identification.
3	0.98	0.96	0.97	Minimal confusion with Class 1.
4	0.99	0.99	0.99	Excellent performance on smaller datasets.

The Confusion Matrix reveals that the spatial augmentation provided by the K-Means distances successfully resolved most of the overlap seen in earlier stages of the project.

The primary remaining challenge involves a small amount of confusion between **Class 1 and Class 3**. Specifically, 7 samples of Class 3 were misidentified as Class 1. This symmetrical confusion, although reduced to just 0.7% of the total test samples, confirms that these two traffic types remain the most statistically similar.

The model achieved a precision of **1.00 for Class 0**, maintaining the trend of perfect identification for this category observed across all models.



4.6 Conclusion

The implementation of a **Fully Connected Neural Network** augmented with **K-Means clustering** distances has proven to be a highly effective approach for network traffic classification. By introducing unsupervised spatial features, the model successfully captured complex relationships within the data that traditional features alone might overlook.

The final results of this architecture lead to several key conclusions:

The addition of 23 distance-based features, derived from an optimized k=23 clustering phase with a **Purity score of 0.8408**, significantly bolstered the network's ability to distinguish between classes.

Through an extensive grid search of 162 training sessions, the configuration utilizing a **Batch Size of 32**, a conservative **Learning Rate of 0.0001**, and **3 Hidden Layers** was identified as the most robust. This model achieved an exceptional **Accuracy of 98.70%** and an **F1-Score of 0.9870** on the external test set.

The model exhibited a highly stable learning curve, running for the full **100 epochs** without triggering early stopping. This stability suggests that the selected hyperparameters allowed for precise convergence to a global minimum.

The model demonstrated near-perfect performance in identifying specific categories, particularly achieving a **1.00 Precision and Recall for Class 0**. Furthermore, it showed remarkable success in handling minority data, evidenced by a **0.99 F1-score for Class 4**.

While a minor residual confusion persists between **Class 1 and Class 3**, representing only 0.7% of the total test samples, the spatial augmentation successfully resolved the majority of overlaps observed in earlier stages of the project.

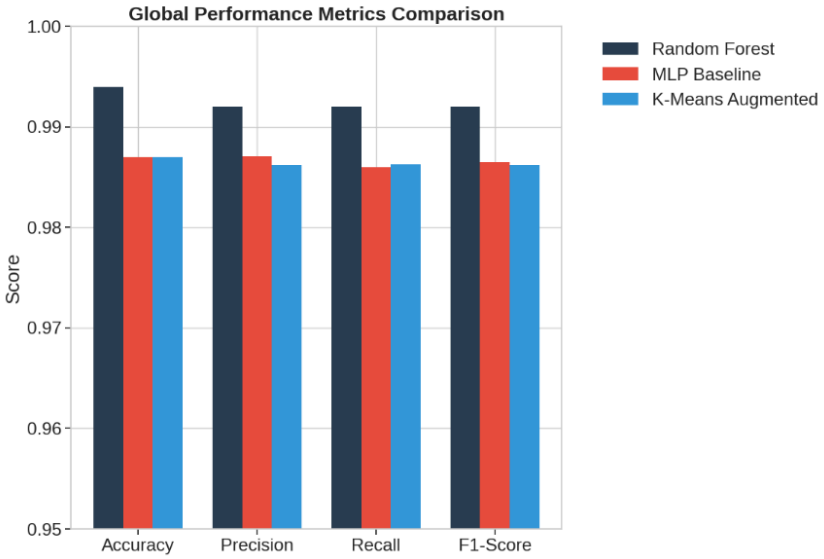
In summary, this study demonstrates that combining unsupervised feature extraction with a systematically tuned fully connected architecture results in a powerful, stable, and highly accurate tool for network traffic analysis and security.

5 Models Comparison

This section provides a comparative analysis of the three architectures implemented: **Random Forest**, the **Baseline Multilayer Perceptron (MLP)**, and the **K-Means Augmented Fully Connected Neural Network**. The comparison focuses on classification performance, training dynamics, and the ability to handle class imbalance and statistical overlaps.

The following table summarizes the global metrics achieved by each model on their respective unseen external test sets.

Metric	**Random Forest**	**MLP**	**K-Means Augmented**
Accuracy	99.40	98.70	98.70
F1-Score	0.9920	0.9865	0.9862
Precision	0.9920	0.9871	0.9862
Recall	0.9920	0.9860	0.9863
Training Epochs	-	55	100



5.1 Class Overlap and Misclassification

A critical observation across all models is the statistical similarity between **Class 1** and **Class 3**. This "feature overlap" represents the primary challenge for the classifiers:

- **Symmetrical Confusion (MLP):** The baseline MLP exhibited a perfect symmetry in error, misclassifying 6 samples of Class 1 as Class 3 and vice-versa. This confirms that these traffic types share nearly identical statistical signatures in the 66-feature space.
- **Spatial Resolution (K-Means Augmented):** The introduction of 23 unsupervised distance-based features helped stabilize the classification. While some residual confusion remains (7 samples of Class 3 identified as Class 1), the spatial augmentation improved the sensitivity for Class 1 to near-perfection.

- **Optimal Separation (Random Forest):** The ensemble nature of Random Forest proved most effective at finding the non-linear boundaries necessary to separate these two classes, resulting in the lowest overall misclassification rate.

5.2 Comparative Analysis of Architectures

5.2.1 Random Forest

Random Forest emerged as the top performer in terms of raw accuracy (**99.40%**). Its main strength lies in its **robustness to feature scaling** and its inherent ability to handle minority classes (Class 4) without additional tuning. It serves as an excellent benchmark for high-speed network classification where computational overhead must be minimized.

5.2.2 MLP

The MLP model demonstrated the power of modern regularization techniques. The model achieved a highly generalized state. Although it slightly trailed the Random Forest in accuracy, its ability to converge to a stable validation loss (0.0642) proves its reliability for unseen data.

5.2.3 K-Means Augmented NN

This model was the most stable during training, running for the full 100 epochs without triggering early stopping.

While the addition of 23 K-Means distance features was intended to provide 'geometric context,' they **failed to improve model performance** beyond the baseline MLP. Consequently, these features appear redundant and do not justify the increased dimensionality.

5.3 Conclusion

Based on the comprehensive performance analysis, the **Random Forest** is identified as the superior model for this network traffic classification task. While the Neural Network architectures provided valuable insights into deep feature relationships.

The **Random Forest** is the recommended model for real-time deployment for the following technical reasons: it achieved a peak accuracy of **99.40%** and an F1-score of **0.9920**, outperforming both the baseline and augmented neural networks across all global metrics. It remains the most efficient "out-of-the-box" solution, reaching these results without the computational overhead of intensive feature scaling or the lengthy training cycles (up to 100 epochs) required by the deep learning models. As illustrated in the per-class analysis, the Random Forest provides the most consistent performance, particularly in resolving the statistical overlaps of Class 1 where other models showed significant drops in F1-score.

In conclusion, the Random Forest is selected as the final project model because it provides the best balance between maximum accuracy and operational simplicity. While the K-Means augmentation was a successful experiment in feature engineering, achieving a low validation

loss of 0.0522 and proving stable over 100 epochs, it ultimately served to validate the strength of the Random Forest's ensemble logic.

Both the Random Forest and the Augmented Neural Network models achieved a remarkable **F1-score of 0.99 for the minority class (Class 4)**.