TOR VERGATA
UNIVERSITÀ DEGLI STUDI DI ROMA

Macroarea di Ingegneria
Dipartimento di Ingegneria Civile e Ingegneria Informatica

Corso di Machine Learning

A.A. 2024/2025

Autoencoder per Intrusion Detection

Martina Lupini

Michele Tosi

Indice

- Problema & Dataset
- Pre-processamento
- Esplorazione dataset
- Architettura modello
- Tuning iperparametri & Addestramento
- Scelta threshold
- Valutazioni finali

Problema & Dataset

Il seguente progetto pone come obiettivo la costruzione di un autoencoder per il rilevamento di **flussi anomali** nel traffico di rete. Il dataset di riferimento è il dataset **NSL-KDD**, utilizzato come benchmark nei problemi di intrusion detection.

Il dataset è composto da tutte features numeriche ad eccezione di **protocol_type**, **service**, **flag_type** e **label**. Non presenta valori mancanti.

```
Data columns (total 42 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   duration                     125973 non-null  int64
 1   protocol_type                125973 non-null  object
 2   service                      125973 non-null  object
 3   flag                         125973 non-null  object
 4   src_bytes                    125973 non-null  int64
 5   dst_bytes                    125973 non-null  int64
 6   land                         125973 non-null  int64
 7   wrong_fragment               125973 non-null  int64
 8   urgent                       125973 non-null  int64
 9   hot                          125973 non-null  int64
 10  num_failed_logins            125973 non-null  int64
 11  logged_in                    125973 non-null  int64
 12  num_compromised              125973 non-null  int64
 13  root_shell                   125973 non-null  int64
 14  su_attempted                 125973 non-null  int64
 15  num_root                     125973 non-null  int64
 16  num_file_creations           125973 non-null  int64
 17  num_shells                   125973 non-null  int64
 18  num_access_files             125973 non-null  int64
 19  num_outbound_cmds            125973 non-null  int64
 20  is_host_login                125973 non-null  int64
 21  is_guest_login               125973 non-null  int64
 22  count                        125973 non-null  int64
 23  srv_count                    125973 non-null  int64
 24  serror_rate                  125973 non-null  float64
 25  srv_serror_rate              125973 non-null  float64
 26  rerror_rate                  125973 non-null  float64
 27  srv_rerror_rate              125973 non-null  float64
 28  same_srv_rate                125973 non-null  float64
 29  diff_srv_rate                125973 non-null  float64
 30  srv_diff_host_rate           125973 non-null  float64
 31  dst_host_count               125973 non-null  int64
 32  dst_host_srv_count           125973 non-null  int64
 33  dst_host_same_srv_rate       125973 non-null  float64
 34  dst_host_diff_srv_rate       125973 non-null  float64
 35  dst_host_same_src_port_rate  125973 non-null  float64
 36  dst_host_srv_diff_host_rate  125973 non-null  float64
 37  dst_host_serror_rate         125973 non-null  float64
 38  dst_host_srv_serror_rate     125973 non-null  float64
 39  dst_host_rerror_rate         125973 non-null  float64
 40  dst_host_srv_rerror_rate     125973 non-null  float64
 41  label                        125973 non-null  object
dtypes: float64(15), int64(23), object(4)
```

Pre-processamento

- **OHE** delle features categoriche
- **Separazione** delle istanze anomale e non anomale
- **Rimozione** delle etichette
- **Divisione** in training (70 %), validation (15%) e test set (15%)
- Applicazione ***MinMaxScaler()***

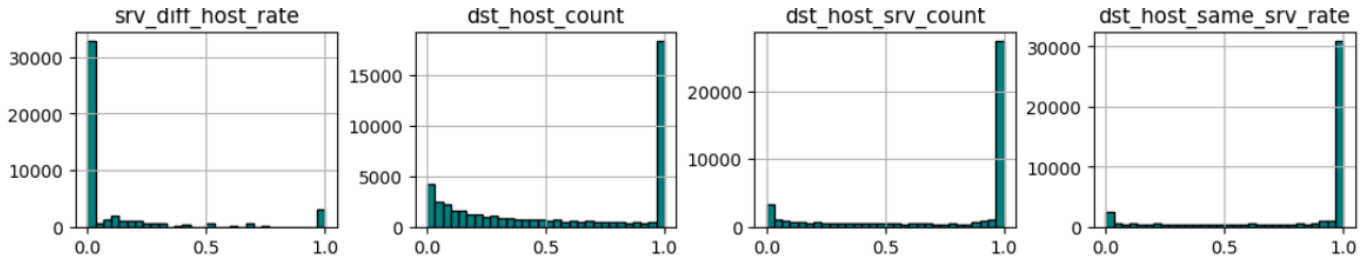
Esplorazione dataset

Andando ad analizzare le distribuzioni delle feauture notiamo che *wrong_fragment* assume sempre valore 0.

Statistiche descrittive:

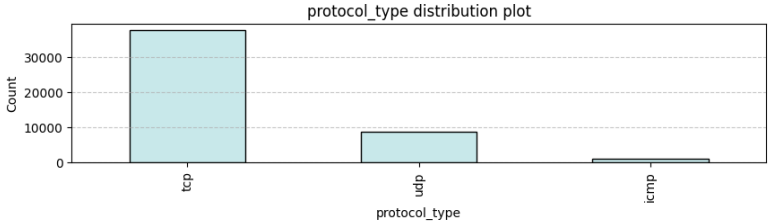
	duration	src_bytes	dst_bytes	land	wrong_fragment	\
count	47140.000000	4.714000e+04	4.714000e+04	47140.000000	47140.0	
mean	163.279317	1.314623e+04	4.416887e+03	0.000106	0.0	
std	1291.684593	4.638773e+05	7.015225e+04	0.010298	0.0	
min	0.000000	0.000000e+00	0.000000e+00	0.000000	0.0	
25%	0.000000	1.320000e+02	1.050000e+02	0.000000	0.0	
50%	0.000000	2.340000e+02	3.790000e+02	0.000000	0.0	
75%	0.000000	3.240000e+02	2.065000e+03	0.000000	0.0	
max	40504.000000	8.958152e+07	7.028652e+06	1.000000	0.0	

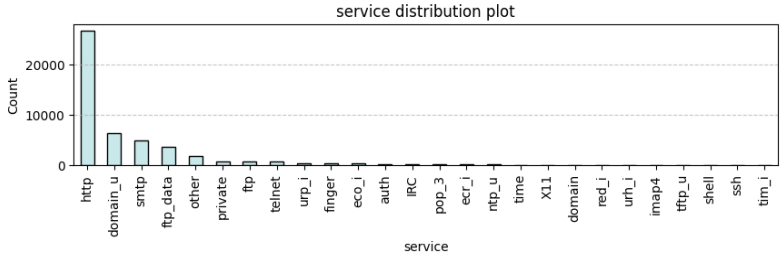
Se grafichiamo i valori assunti dalle features per effetto dello scaler tutti i valori sono compresi tra [0,1].

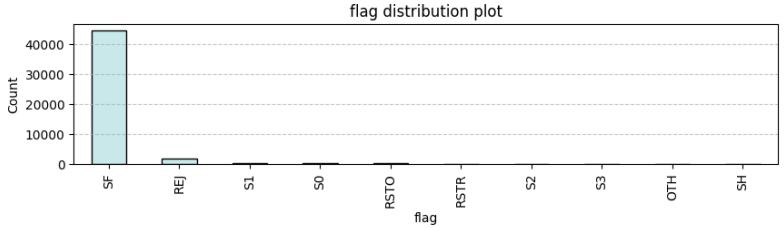


Esplorazione dataset

Alcuni valori non sono mai assunti dalle features categoriche del training set suggerendo che sono tipici di istanze anomale.

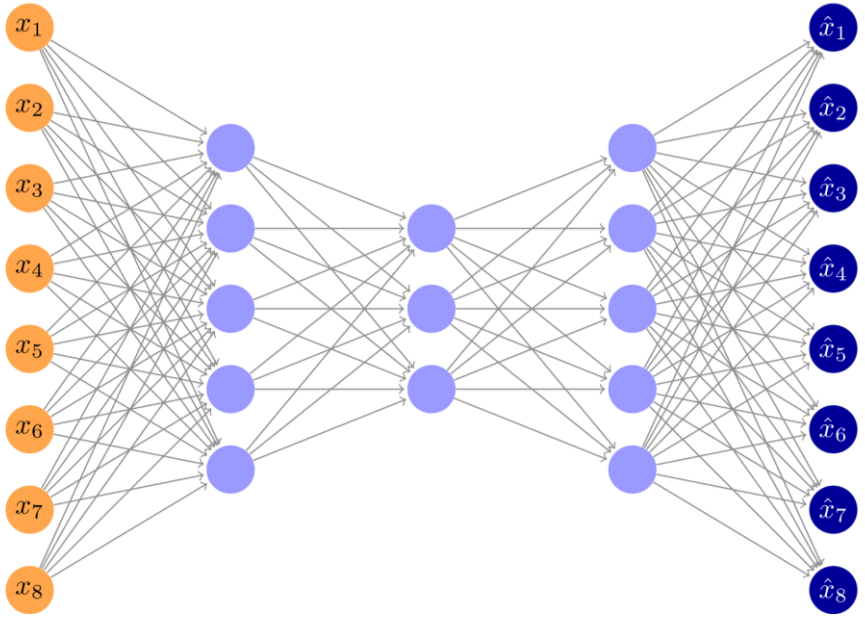






Architettura del modello

- **Architettura simmetrica**
- **ReLU**
- **Numero di livelli e unità** non fissati
- **Adam** come algoritmo di ottimizzazione
- **MSE** come funzione di perdita



Tuning iperparametri & Addestramento

Per scegliere il numero di livelli e di unità per livello si utilizza l'algoritmo **Hyperband**. La configurazione restituita è la seguente:

Model: "sequential_1"

Layer (type)	Output Shape	Param #
flatten_1 (Flatten)	(None, 122)	0
dense_4 (Dense)	(None, 250)	30,750
dense_5 (Dense)	(None, 125)	31,375
dense_6 (Dense)	(None, 250)	31,500
dense_7 (Dense)	(None, 122)	30,622

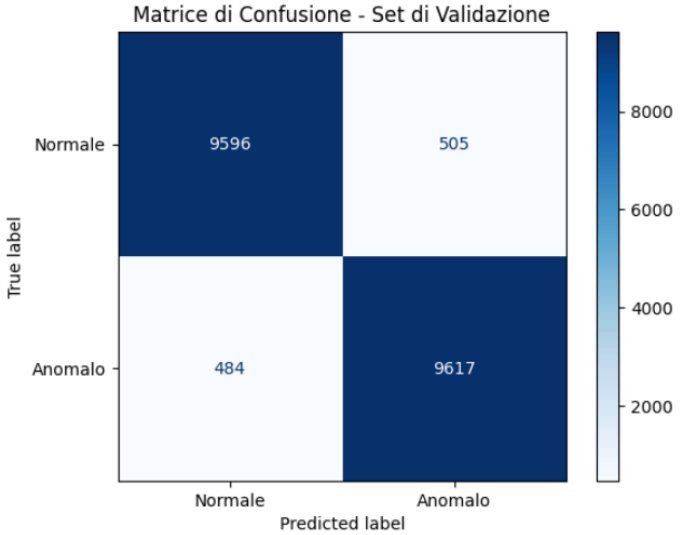
Total params: 124,247 (485.34 KB)
Trainable params: 124,247 (485.34 KB)
Non-trainable params: 0 (0.00 B)

Durante l'addestramento si utilizza **Early Stopping** per aumentare le abilità di generalizzazione del modello.

Threshold 95° Percentile

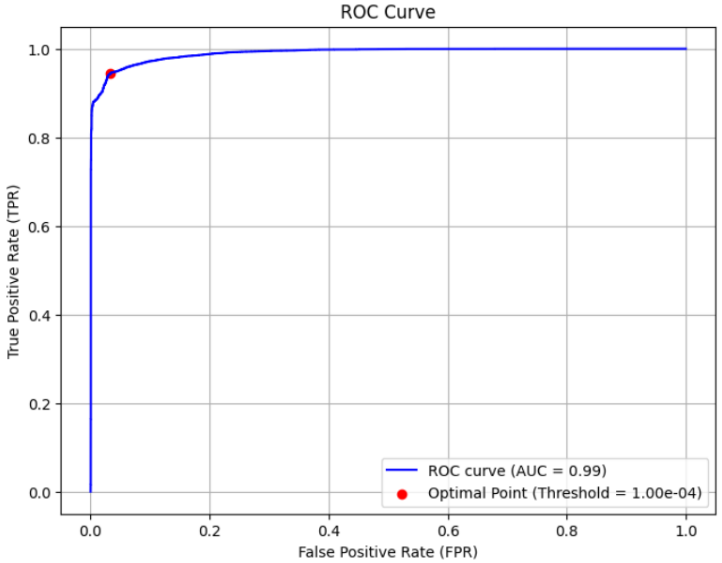
Calcoliamo ora la threshold in modo tale che il **95% dei dati non anomali** cada al di sotto di tale soglia.

```
La threshold percentile è: 7.409553983966119e-05
Report di classificazione sul set di validazione con threshold percentile :
              precision    recall  f1-score   support

           0       0.95      0.95      0.95     10101
           1       0.95      0.95      0.95     10101

    accuracy                           0.95     20202
   macro avg       0.95      0.95      0.95     20202
weighted avg       0.95      0.95      0.95     20202
```



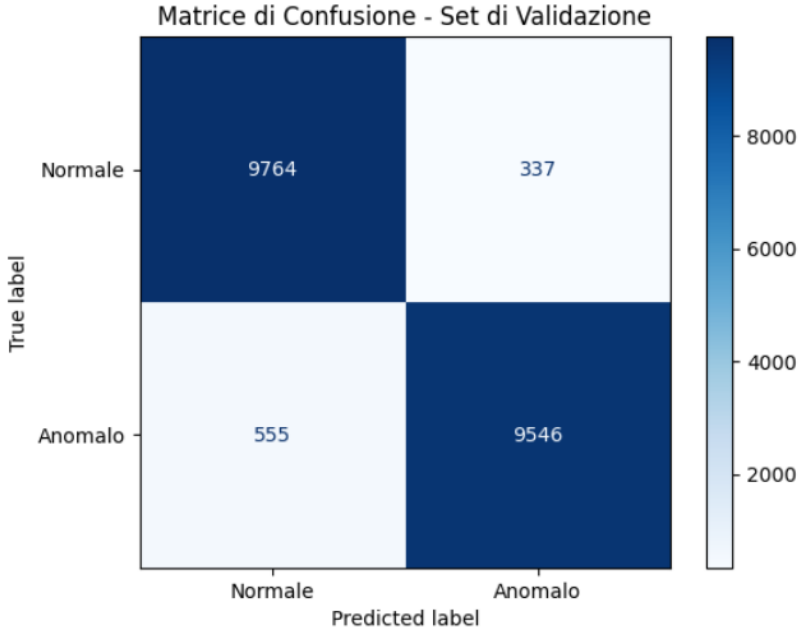
Threshold Youden’s Index

L'indice di Youden rappresenta il punto della **curva ROC** che massimizza la **differenza tra il TPR e il FPR**. La curva ROC viene tracciata valutando il TPR e il FPR per diverse thresholds.



Threshold Youden’s Index

```
La threshold ROC è: 0.00010043981154384735
Report di classificazione sul set di validazione con threshold ROC :
              precision    recall  f1-score   support

           0       0.95      0.97      0.96     10101
           1       0.97      0.95      0.96     10101

    accuracy                           0.96     20202
   macro avg       0.96      0.96      0.96     20202
weighted avg       0.96      0.96      0.96     20202
```



Threshold Vettoriale

A seguito dell'addestramento viene calcolato il **vettore di errore di ricostruzione** per ciascuna feature. Esso è composto dal massimo valore di errore per ciascuna componente.

Algorithm 1 Proposed Autoencoder Threshold Calculation

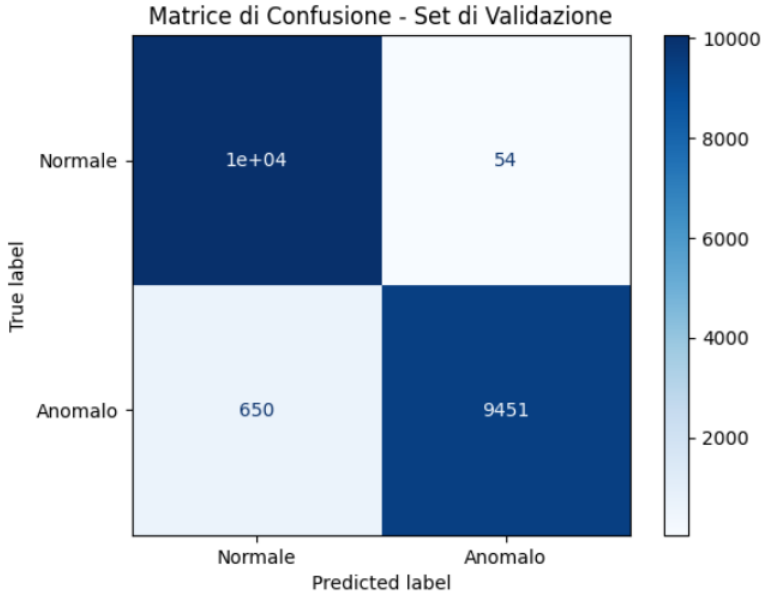
$X \leftarrow$ Data of a specific class
$n_{samples} \leftarrow$ Number of X samples
$n_{features} \leftarrow$ Number of X features
$AE \leftarrow$ Trained autoencoder with X data
$th \leftarrow (0, 0, ..., 0_{n_{features}})$
for $i = 1$ to $n_{samples}$ **do**
  $\hat{X}_i \leftarrow AE(X_i)$
  $(r_1, r_2, ..., r_{n_{features}}) \leftarrow RE(X_i, \hat{X}_i)$
  $th \leftarrow max((th_1, th_2, ..., th_{n_{features}}), (r_1, r_2, ..., r_{n_{features}}))$
end for

Dal paper: https://www.researchgate.net/publication/366852853_Practical_autoencoder_based_anomaly_detection_by_using_vector_reconstruction_error

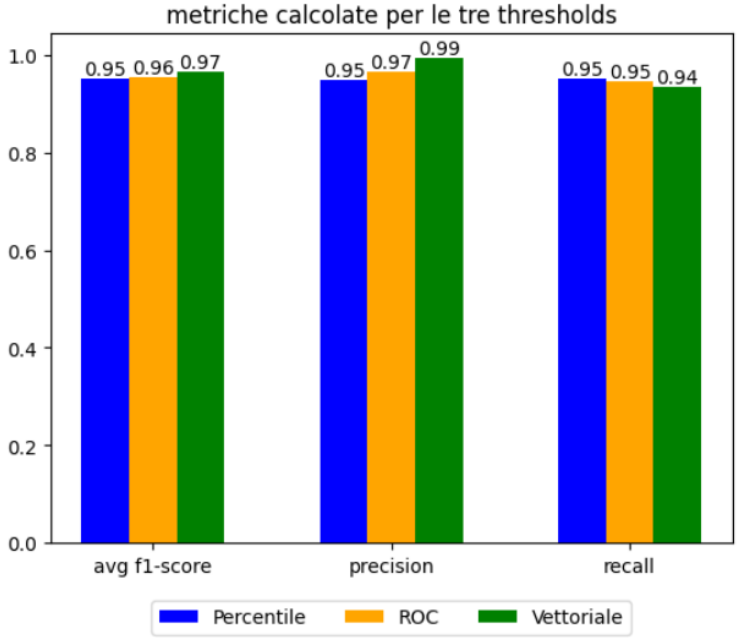
Threshold Vettoriale

Report di classificazione sul set di validazione con threshold vettoriale:

	precision	recall	f1-score	support
0	0.94	0.99	0.97	10101
1	0.99	0.94	0.96	10101
accuracy			0.97	20202
macro avg	0.97	0.97	0.97	20202
weighted avg	0.97	0.97	0.97	20202



Valutazione finale



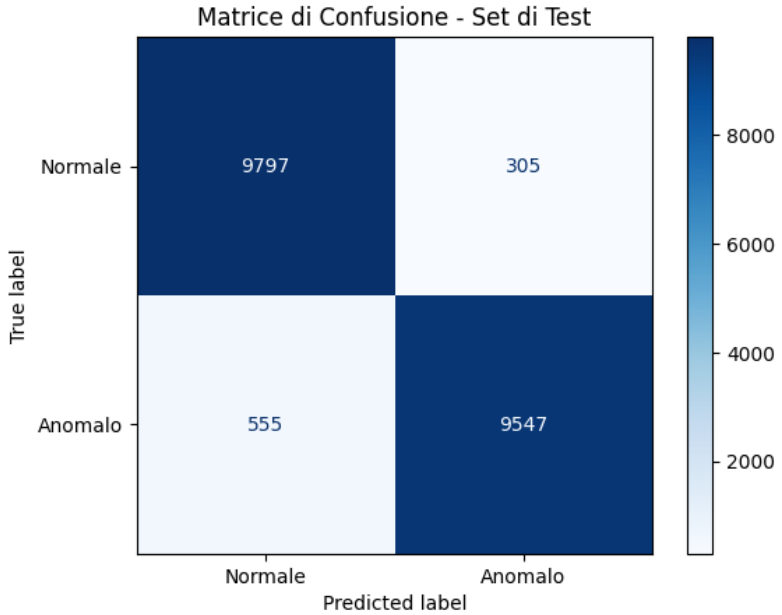
Nel nostro caso, è prioritario identificare correttamente gli attacchi (anomalie), anche a costo di accettare un numero più elevato di falsi positivi. Ciò implica dare maggiore importanza alla **recall**, poiché non vogliamo classificare attacchi come normali (falsi negativi).
La threshold **ROC** sembra essere la scelta migliore, garantisce una buona capacità di identificazione degli attacchi mantenendo un bilanciamento accettabile con precisione e F1-score

Valutazione finale

Il risultato finale sul **test set** del modello scelto è:

Report di classificazione sul set di test con threshold ROC :

	precision	recall	f1-score	support
0	0.95	0.97	0.96	10102
1	0.97	0.95	0.96	10102
accuracy			0.96	20204
macro avg	0.96	0.96	0.96	20204
weighted avg	0.96	0.96	0.96	20204



Grazie per l’attenzione!

Michele Tosi, Martina Lupini
Università degli Studi di Roma “Tor Vergata”
Facoltà di Ingegneria