

# Water Quality Prediction

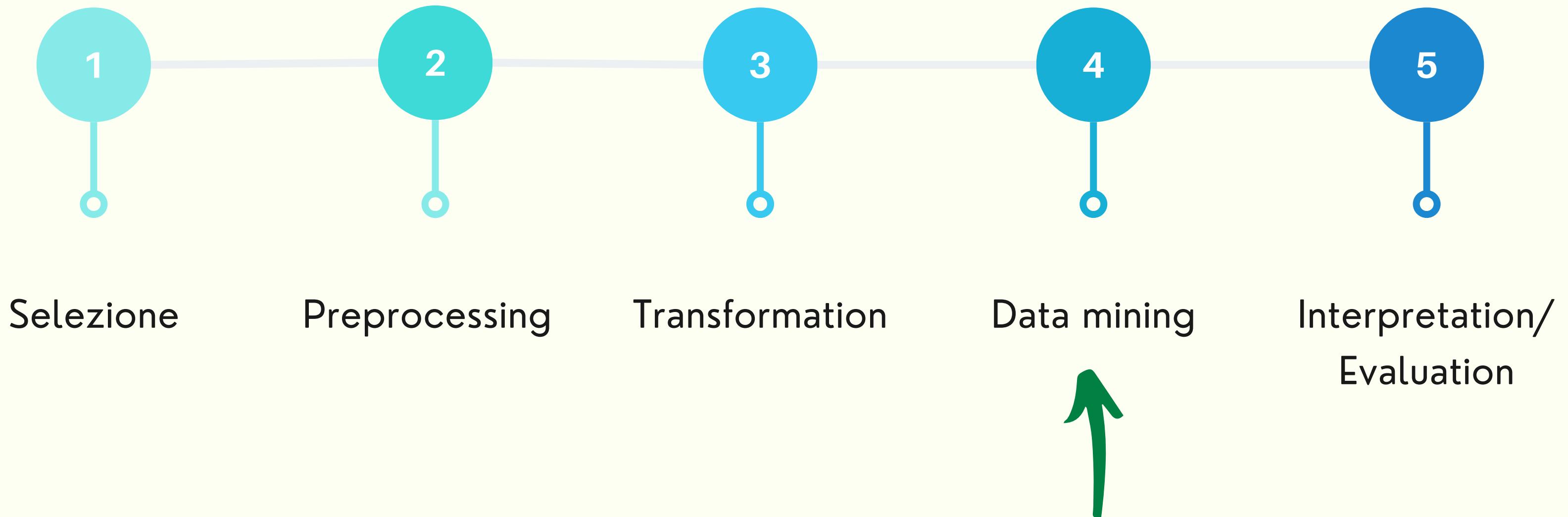


Progetto del corso Data Mining



Giuseppe Centraco  
227591

# Scoperta della conoscenza



# Quali sono gli obiettivi?

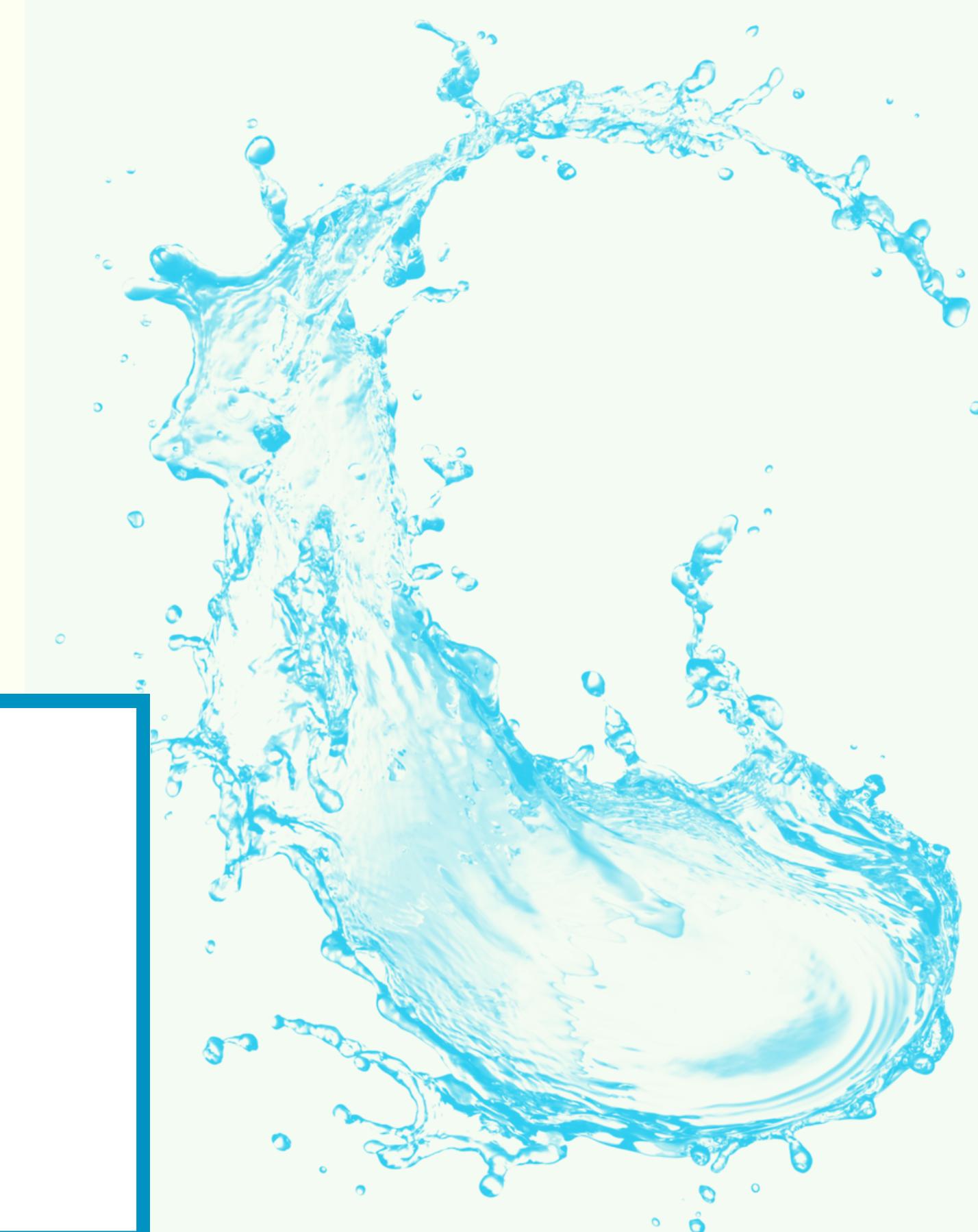
## → Water Quality Prediction

Sviluppo di modelli predittivi per la **classificazione** (binaria) della potabilità dell'acqua facilitando la presa di decisioni informate per la gestione delle risorse idriche.



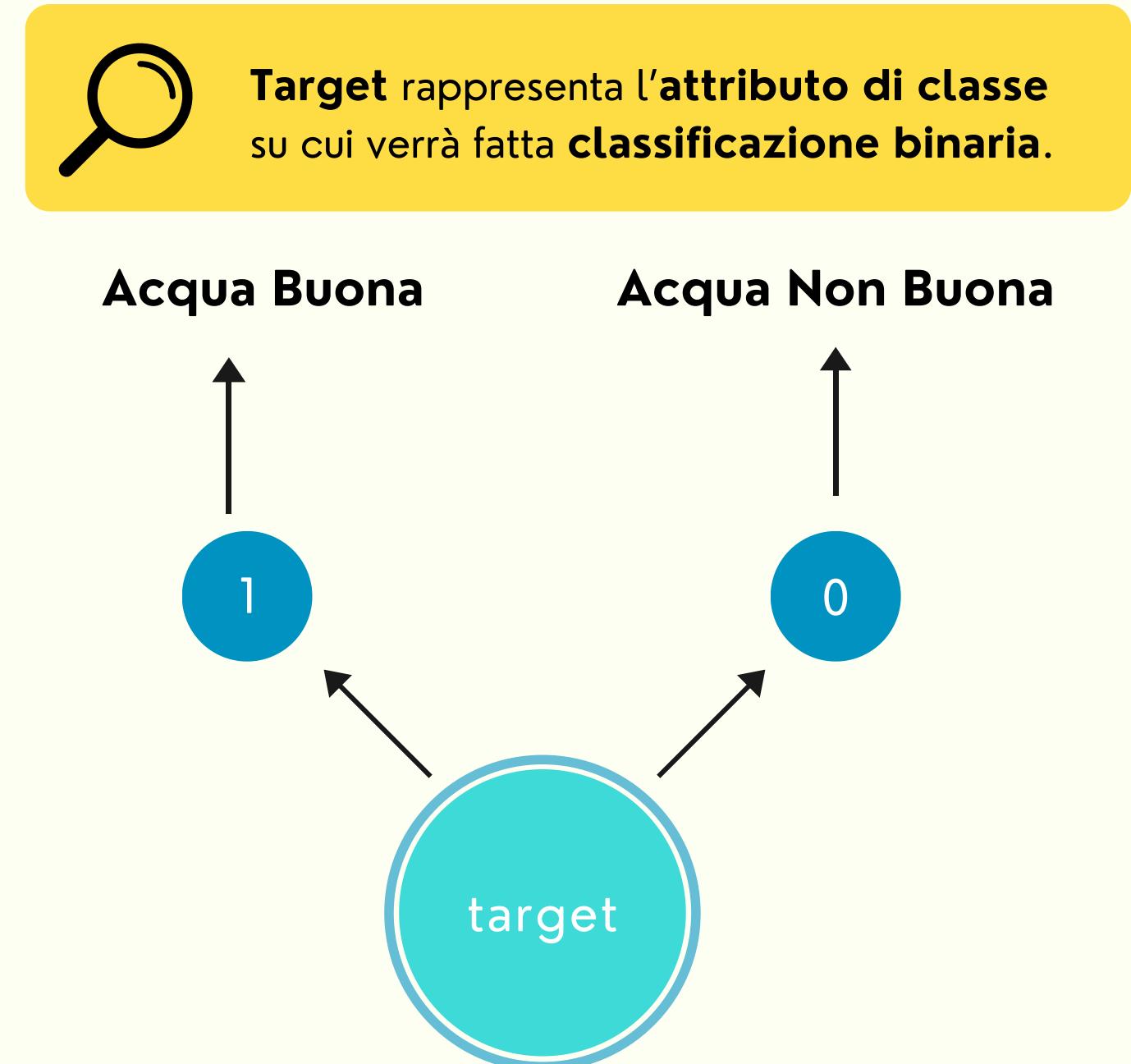
## INFORMAZIONI DATASET

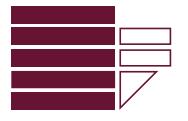
- Fonte: **Kaggle**
- Numero tuple: **1048575**
- Numero attributi: **24**
- Tipologia di attributi: **int, float, object**



# Quali sono gli attributi del dataset?

- **index** (int64)
- **ph** (float64)
- **iron** (float64)
- **nitrate** (float64)
- **chloride** (float64)
- **lead** (float64)
- **zinc** (float64)
- **color** (object)
- **turbidity** (float64)
- **fluoride** (float64)
- **copper** (float64)
- **odor** (float64)
- **sulfate** (float64)
- **conductivity** (float64)
- **chlorine** (float64)
- **manganese** (float64)
- **total dissolved solids** (float64)
- **source** (object)
- **water temperature** (float64)
- **air temperature** (float64)
- **month** (object)
- **day** (float64)
- **time of day** (float64)
- **target** (int64 - binario 0 o 1)





# Preprocessing

Tutti i dati sono stati  
preprocessati ed è stata  
fatta Data Visualization.

**1 - Attributi inutili**

**2 - Tuple duplicate**

**3 - Analisi su singoli attributi**

**4 - Trasformazione dei dati**

**5 - Bilanciamento**

**6 - Campionamento**

**7 - Analisi outlier**

\*Il tutto è stato reso possibile grazie alle librerie di supporto Python.

# Preprocessing (1) - Attributi inutili e Tuple duplicate

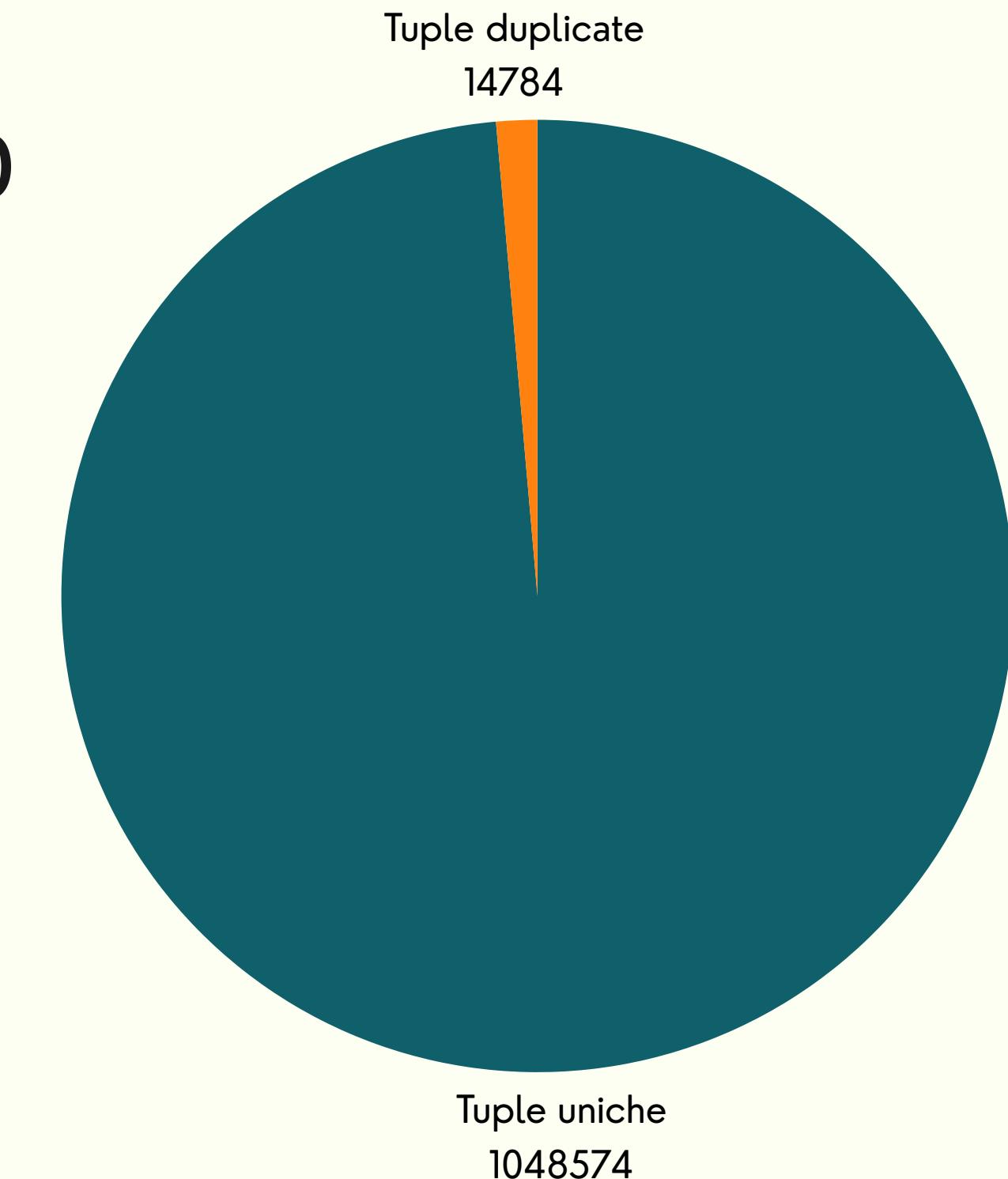
- Rimozione dell'attributo **index** (irrilevante)
- Identificazione e rimozione delle **tuple duplicate**(\*)



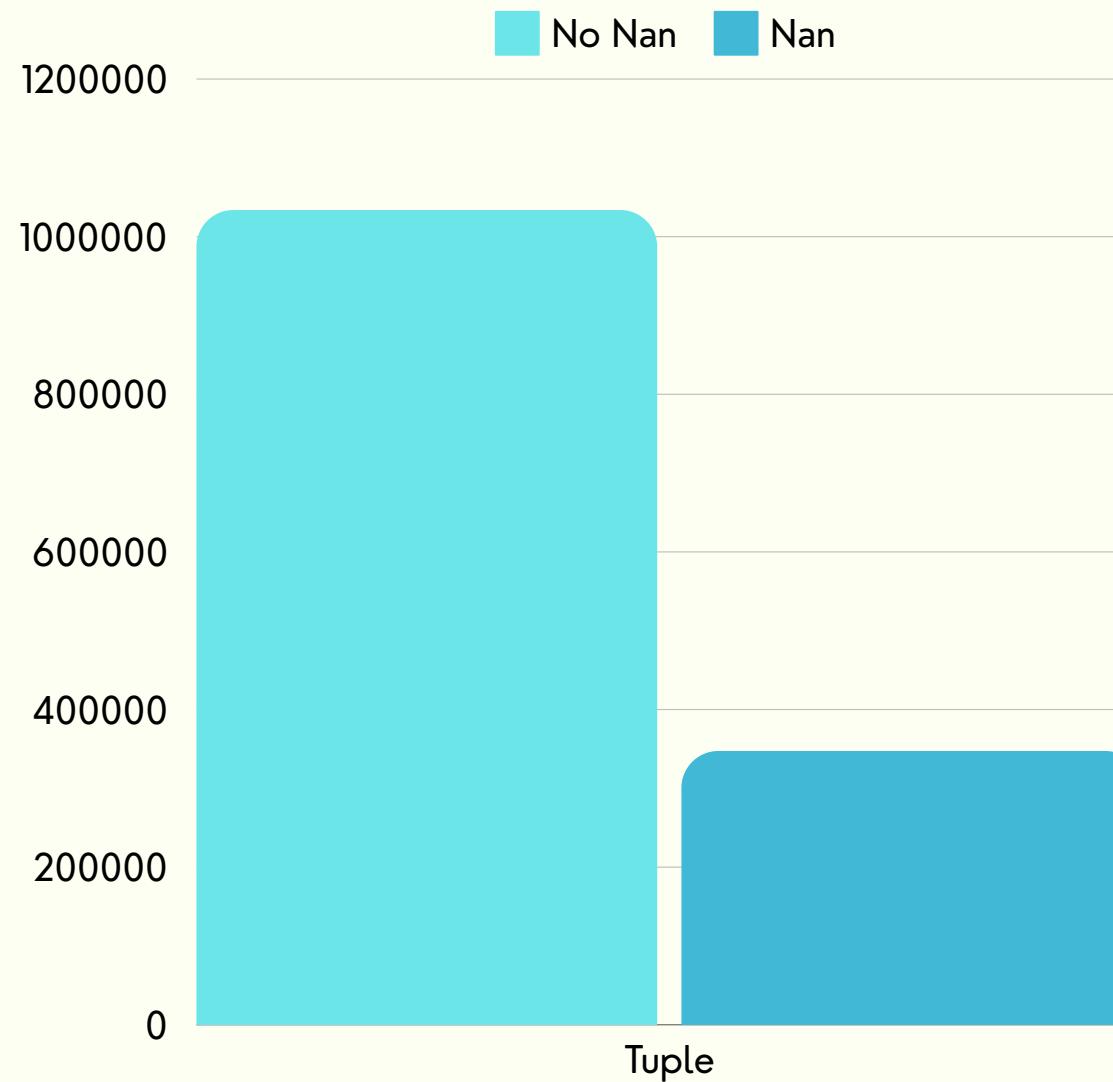
 \*non essendoci più un attributo identificativo (index) **14784** tuple sono state rilevate come duplicate.

→ **Soluzione: banale drop!**

 Tuple uniche  Tuple duplicate



## Preprocessing (2) - Valori Nan



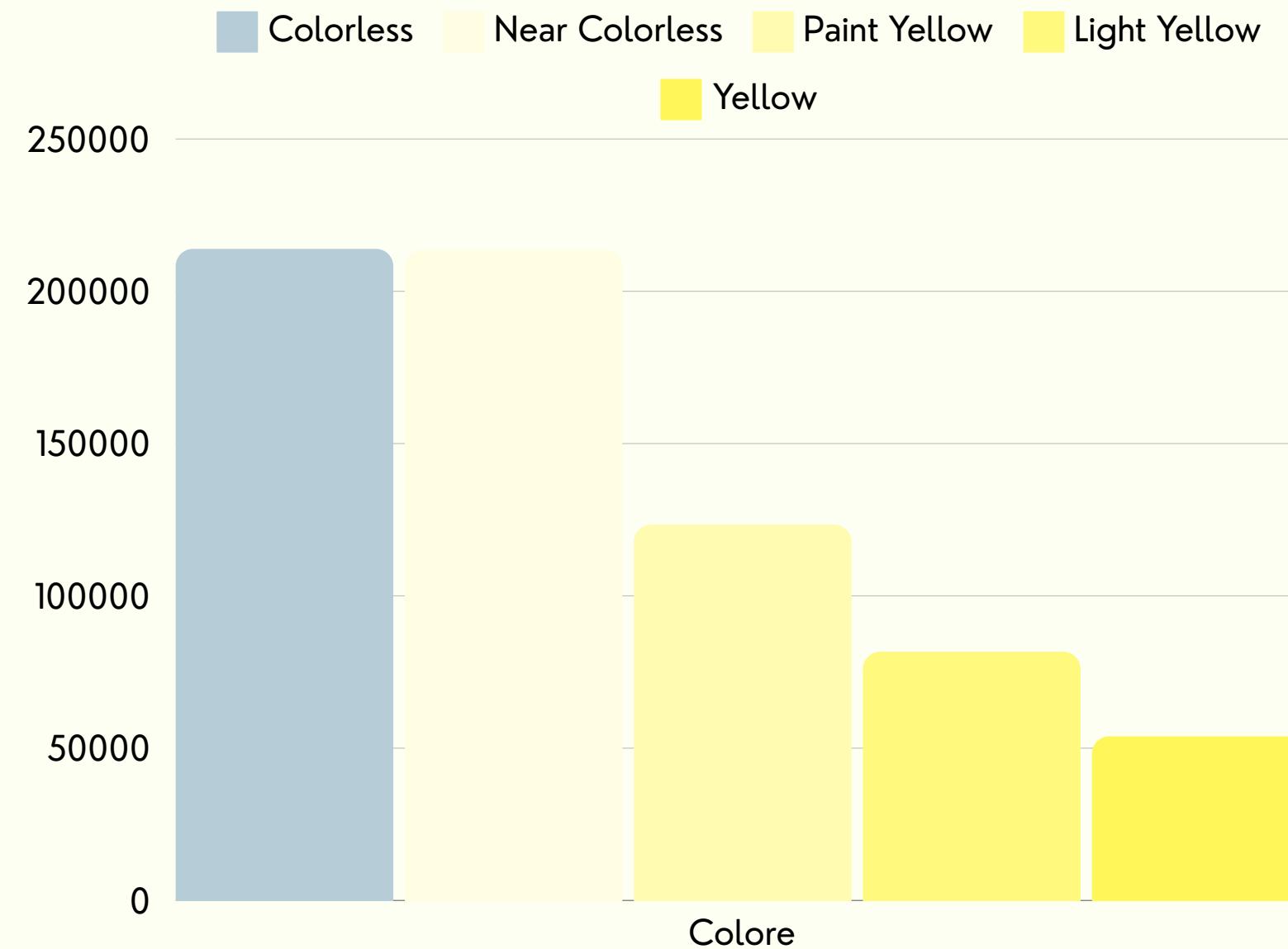
Come gestire i valori **Nan**?

Senza duplicati il dataset contiene più di un milione di tuple. Quindi, la presenza di valori Nan è stata risolta con un banale **drop** di tutte le tuple contenenti almeno un valore nullo.  
Dimensione dataset dopo il drop:  
**686413**

Non era meglio gestire i valori con un **imputer**?

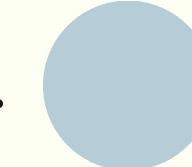
1. Bassa relazione tra i dati per avere buoni risultati con Iterative Imputer;
2. Costo computazionale elevato per eseguire Imputer basato su RandomForest.

# Data Visualization (1) - Come si comporta l'attributo color?

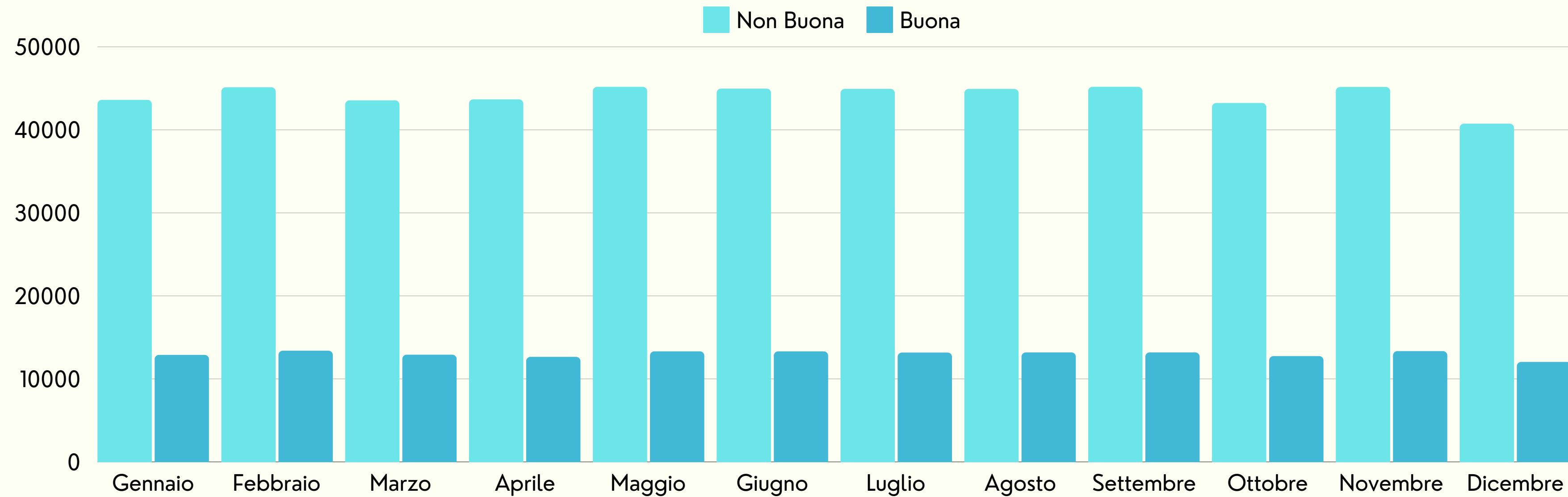


Non ci sono **valori inaspettati** oppure **anomali**.

\*a scopo dimostrativo è stato utilizzato il grigio per indicare Colorless.



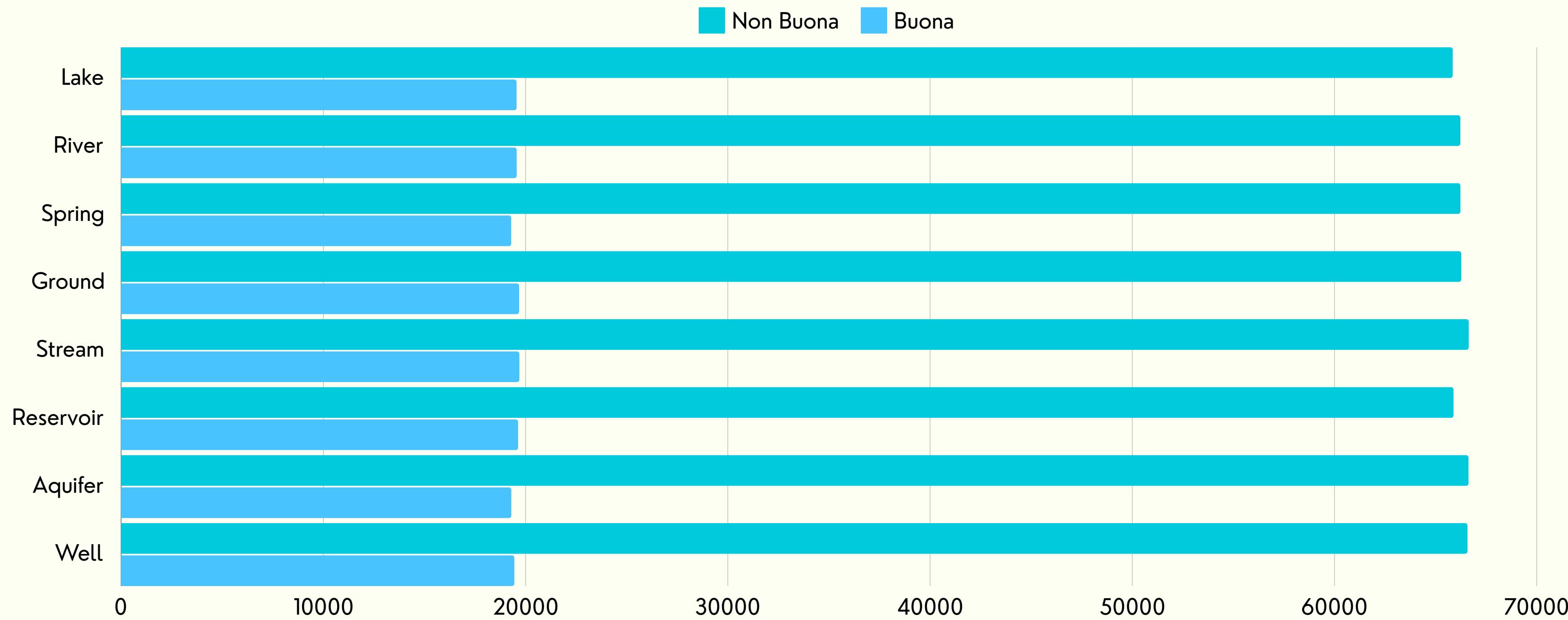
## Data Visualization (2) - Come cambia la qualità dell'acqua in base al mese?



Questo tipo di analisi offre degli spunti preziosi sulle possibili **influenze stagionali e ambientali**. Le variazioni climatiche, gli agenti atmosferici e le attività umane che si susseguono in vari periodi dell'anno possono avere un impatto significativo sulla **qualità** delle risorse idriche.

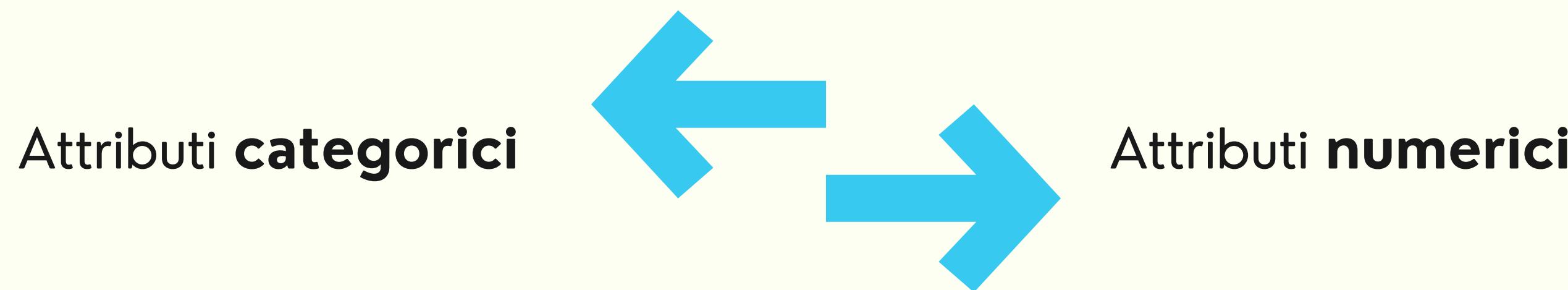
## Data Visualization (3) - Quanto incide la sorgente dell'acqua?

Diverse sorgenti, quali fiumi, laghi o pozzi, possono presentare variazioni significative nei parametri di qualità.



## Preprocessing (3) - Come sono stati trasformati i dati?

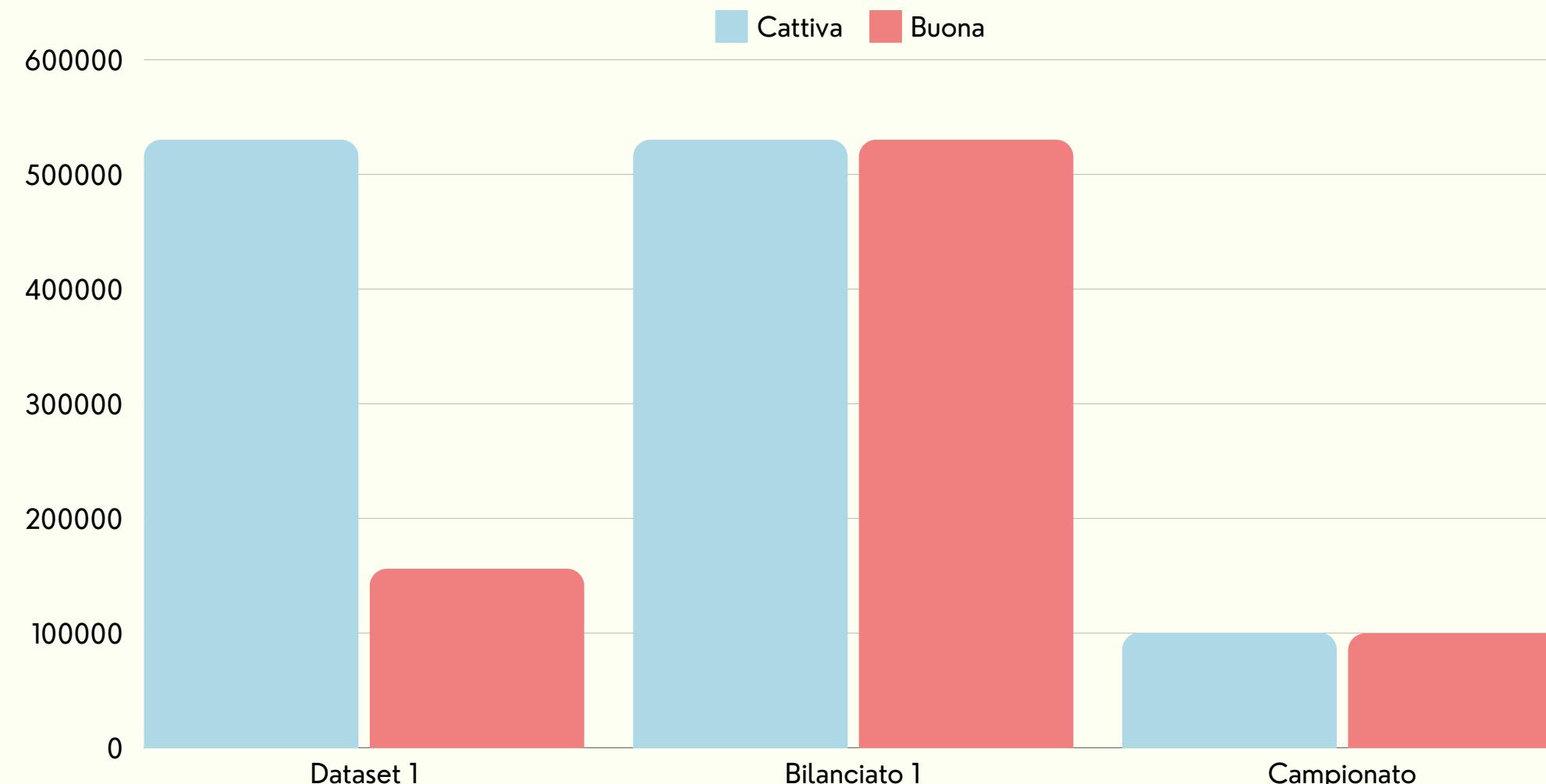
La trasformazione degli attributi è avvenuta tramite il metodo **encode\_categorical(ds)**, il quale internamente utilizza un LabelEncoder.



I dati poi sono stati scalati tramite uno **StandardScaler** in modo tale da garantire che ogni caratteristica contribuisca equamente al modello, consentendo di ottenere prestazioni più robuste e interpretabili. Utile soprattutto nell'analisi degli outlier in scale di attributi diversi possono influenzare il calcolo della distanza.

## Preprocessing (4) - Analisi outlier

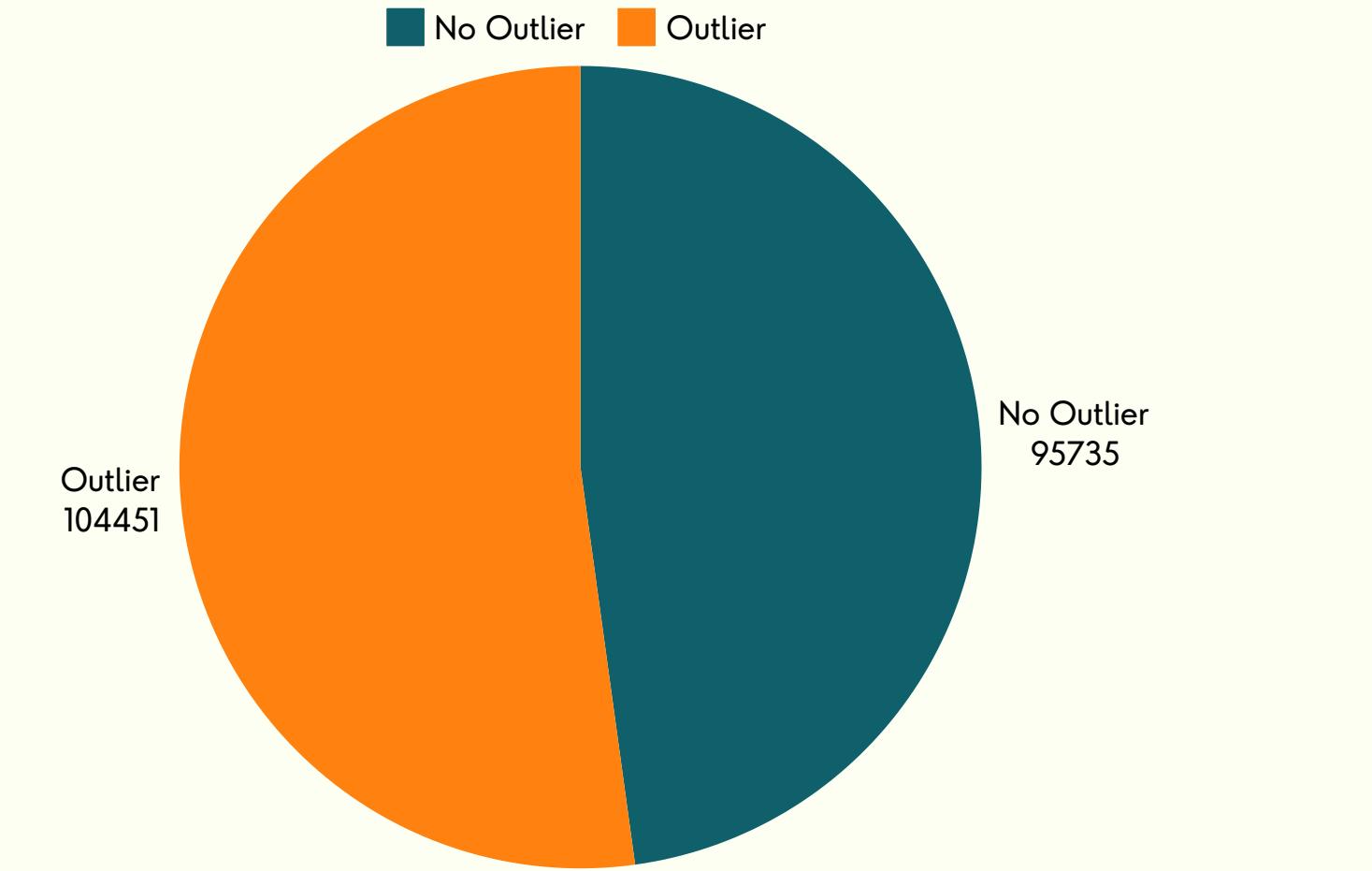
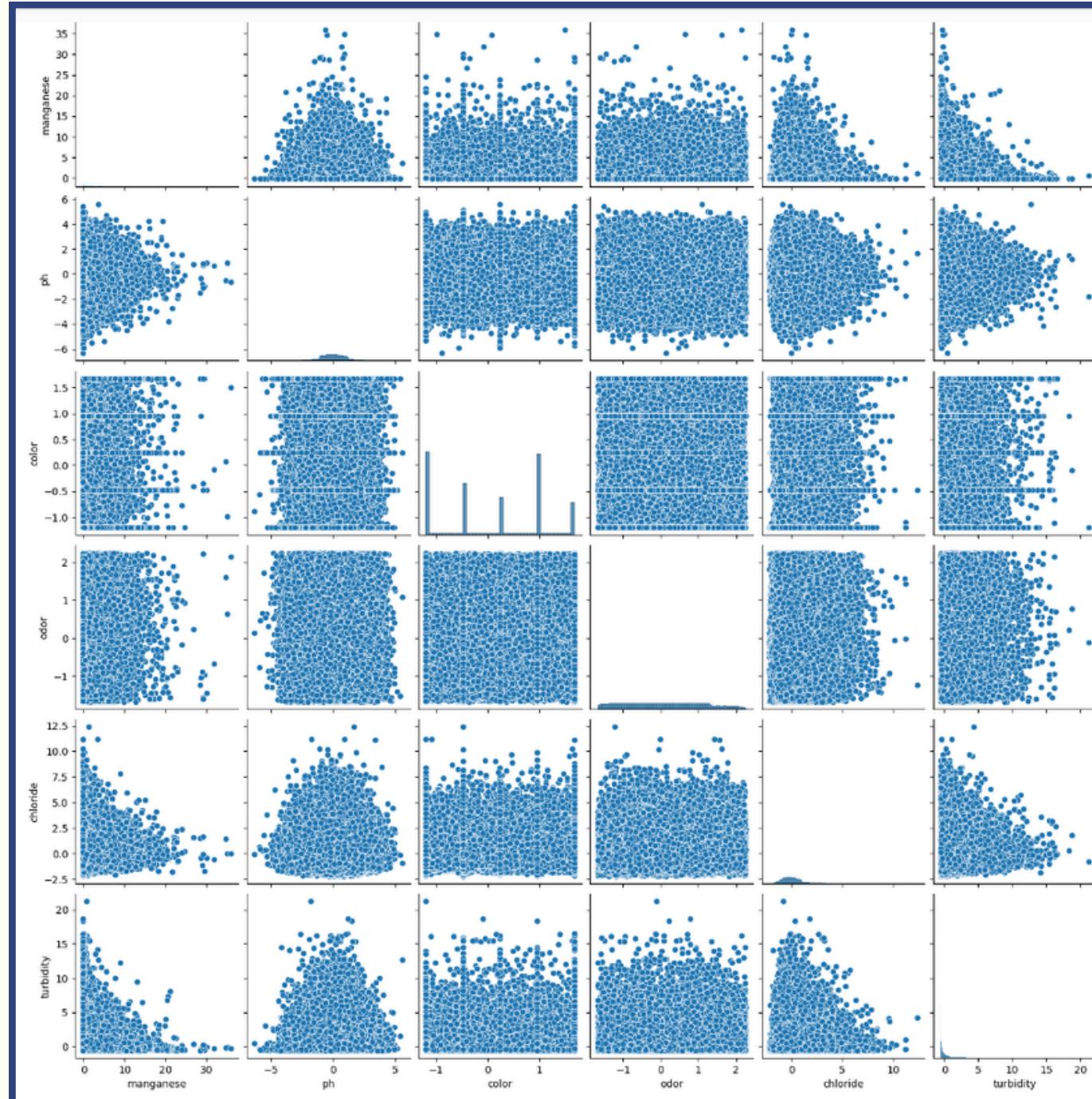
Gli outlier sono delle osservazioni che si **discostano** marcatamente da tutte le altre e possono provocare errori nelle analisi. L'algoritmo DBSCAN per rilevare gli outlier richiede enormi capacità computazionali per l'esecuzione su un numero grande di tuple. Quindi, il dataset è stato prima **bilanciato (Bilanciato1)** e **campionato (Campionato)** prima di applicare DBSCAN.



- **Dataset 1** = dataset allo stato attuale
- **Bilanciato 1** = dataset bilanciato tramite  
`balance_dataset_combine(df_scale  
d, 'target', 1, 1)`
- **Campionato** = applicazione del campionamento semplice (numero random di tuple pari a 200186) sul dataset bilanciato 1

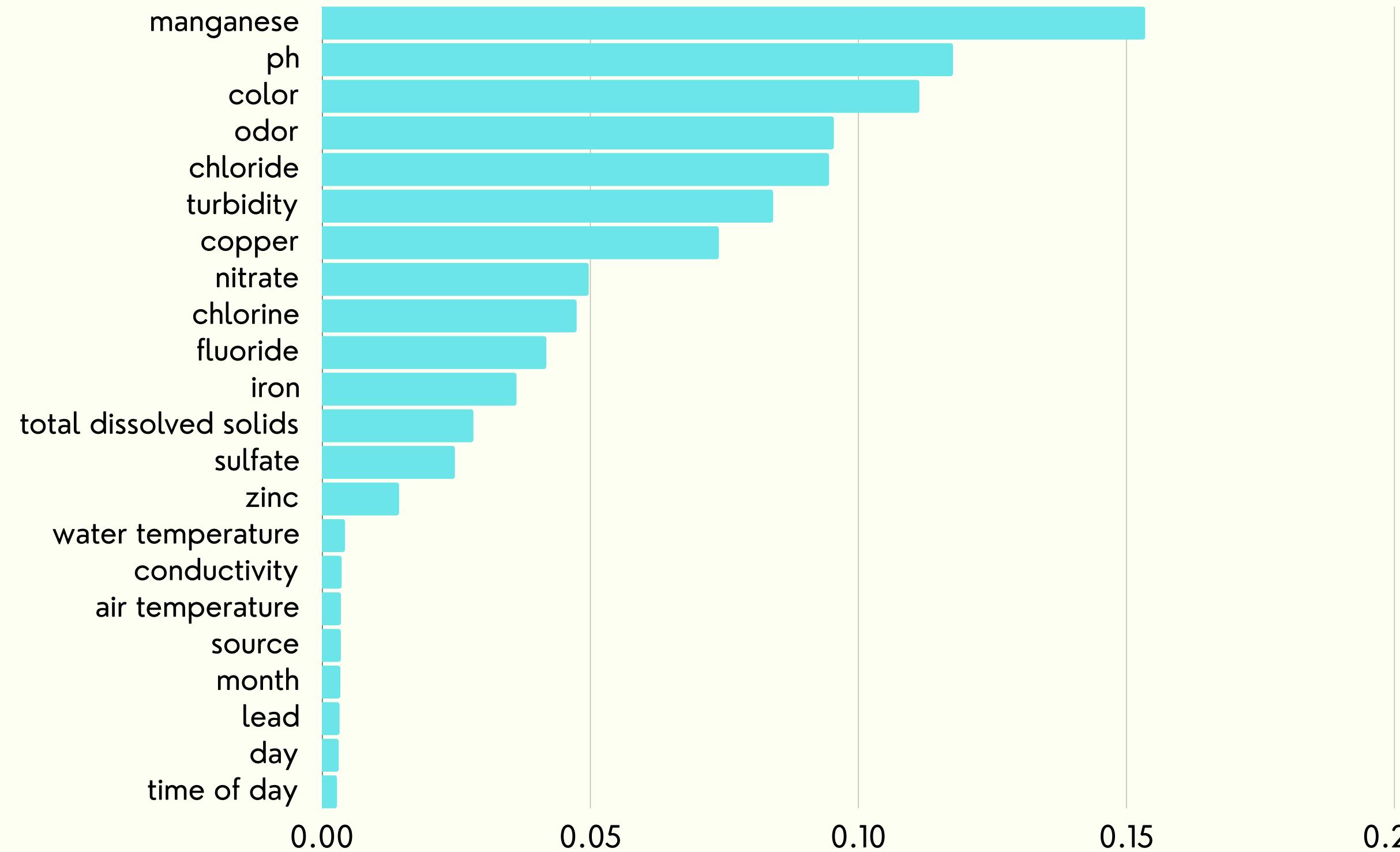
# Ci sono outlier nel dataset?

- Ricerca outlier tramite **DBSCAN**.



- Le **104451** tuple outlier rilevate sono state rimosse (per lo stesso motivo della rimozione dei valori Nan);
- Il dataset è stato ri-bilanciato tramite il metodo **balance\_dataset(df\_no\_outliers, 'target')**, il quale esegue undersampling della classe maggioritaria.

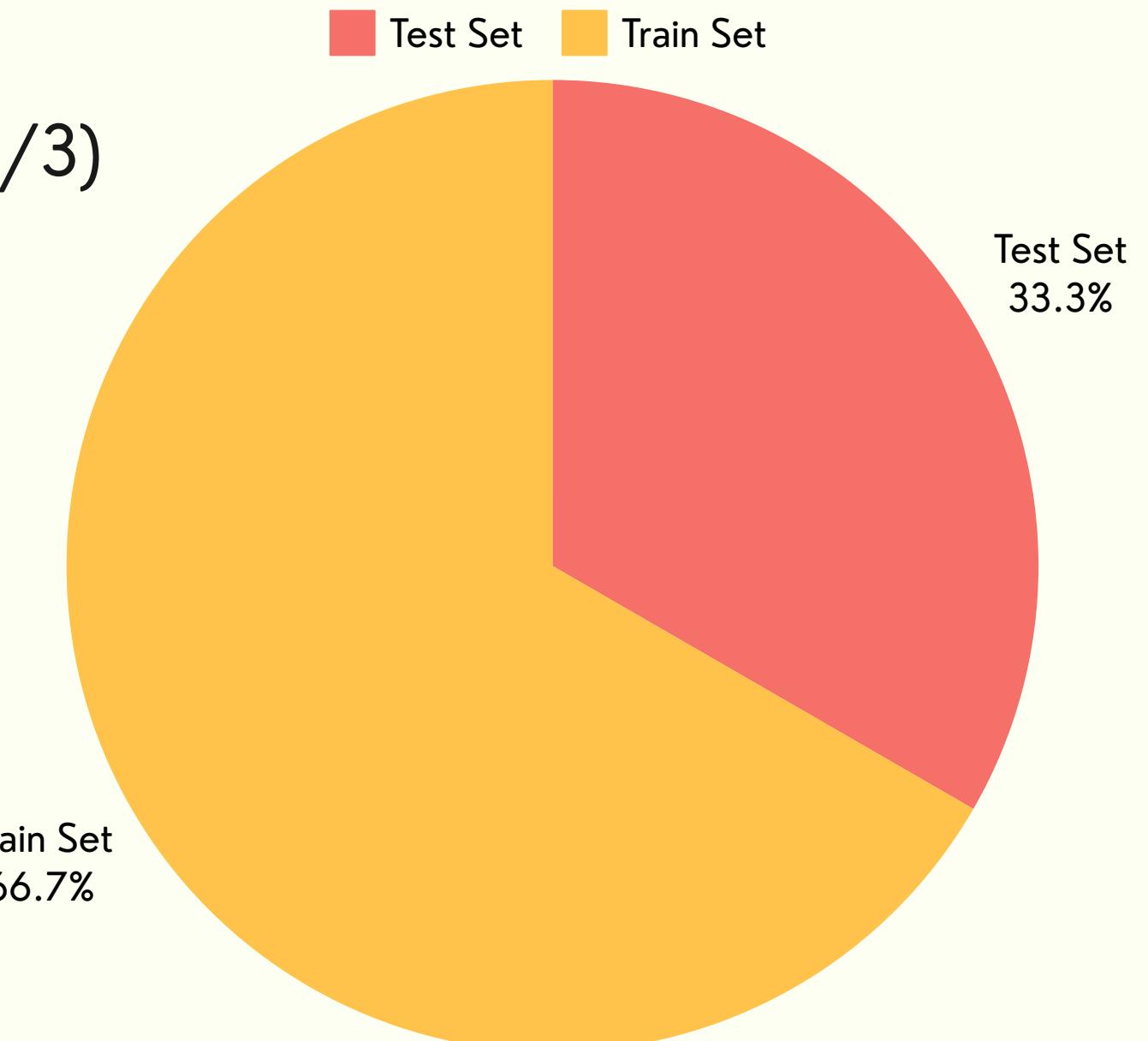
## Preprocessing (5) - Qual è il grado di importanza degli attributi?



- La **feature map** è stata calcolata tramite RandomForest.
- Gli attributi "month", "time of day" e "day" non sono rilevanti ai fini dell'analisi, quindi sono stati droppati.

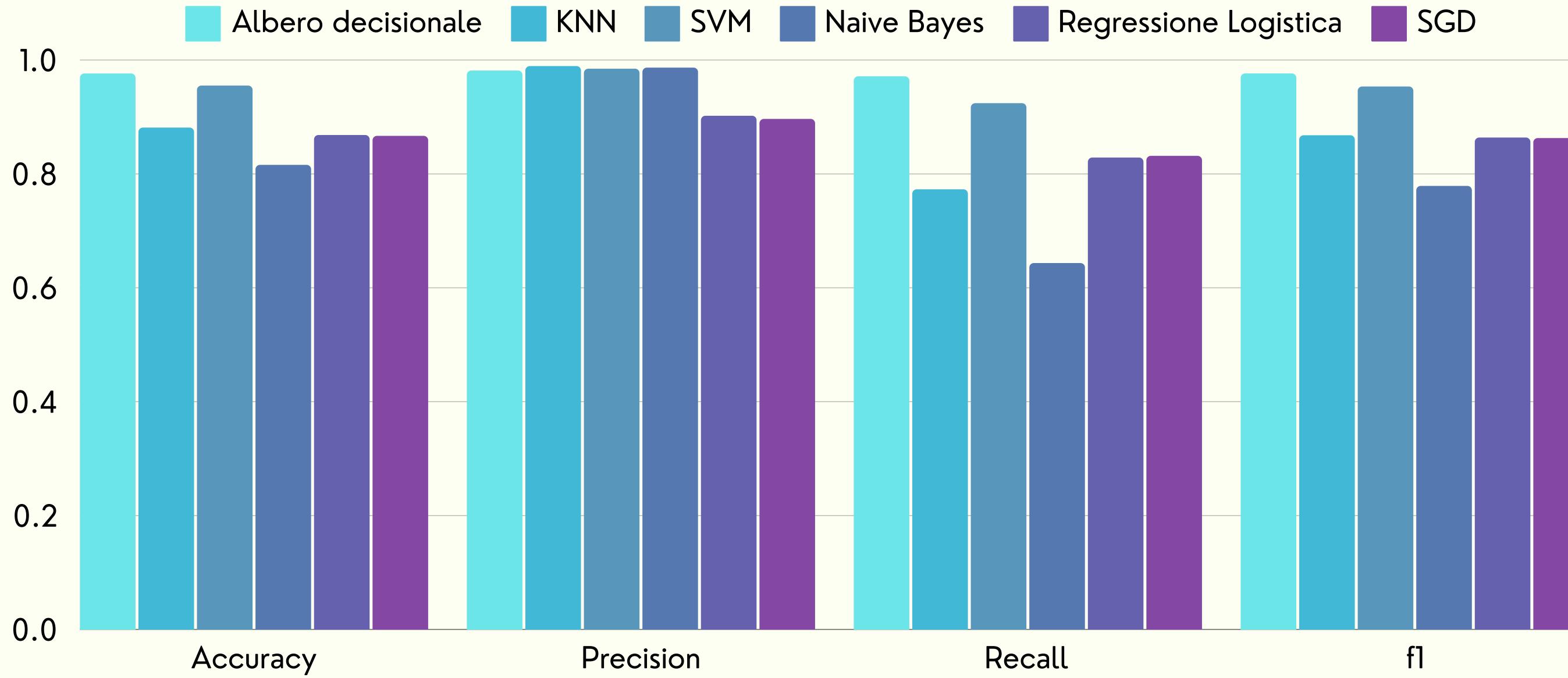
# Classificatori

- Terminata la fase di preprocessing, il dataset ha le seguenti caratteristiche:
  - 76060 tuple
  - 20 attributi
- Il dataset è stato partizionato in **train set** (2/3) e **test set** (1/3)
- Che tipo di **classificatori** sono stati utilizzati?
  - Di base (anche con la versione GridSearchCV)
  - Ensemble
  - Reti neurali
- Le metriche utilizzate per valutare i modelli sono:  
**accuracy, precision, recall, f1-score**



# Risultati classificatori di base

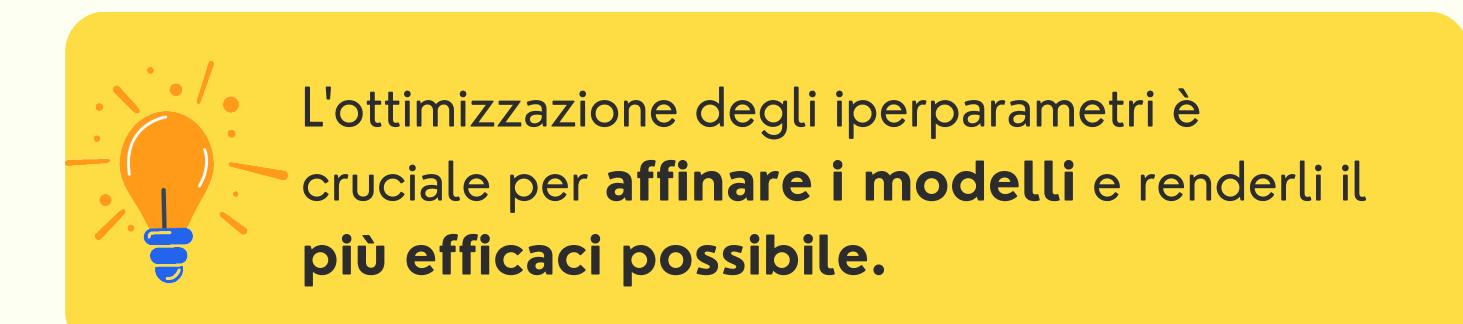
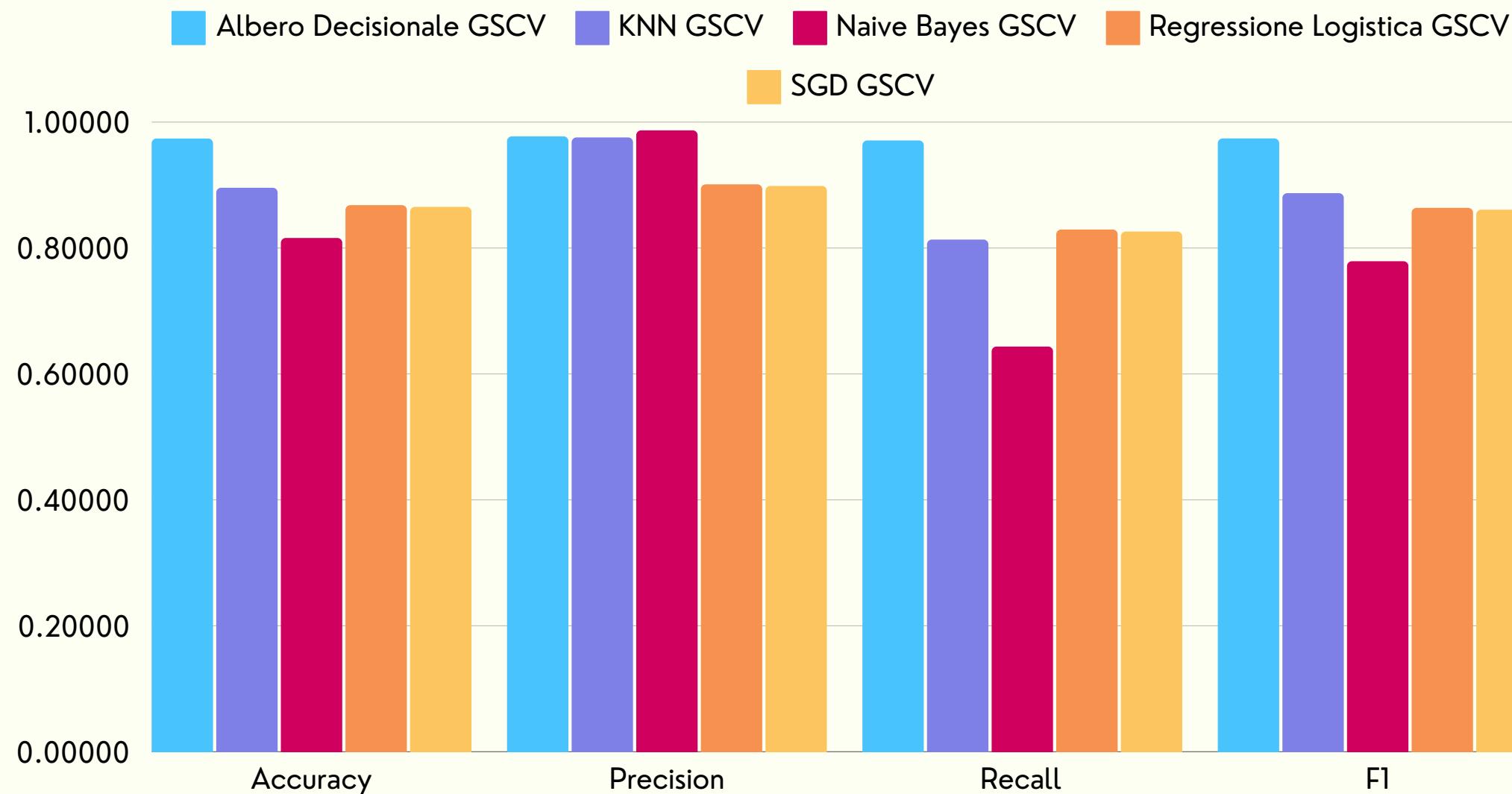
Questi modelli offrono un'introduzione intuitiva e diretta ai principi di classificazione, pur essendo relativamente semplici da implementare e interpretare.



L'albero decisionale  
è il modello migliore  
tranne per la metrica  
Precision.

# Risultati classificatori di base GridSearchCV

Per migliorare ulteriormente le prestazioni dei modelli di base, è stata applicata l'**ottimizzazione degli iperparametri** mediante l'uso di GridSearchCV. Permette di esplorare una vasta gamma di configurazioni e di **identificare quelle che offrono le migliori prestazioni sul dataset di training**. L'ottimizzazione degli iperparametri è cruciale per affinare i modelli e renderli il più efficaci possibile.

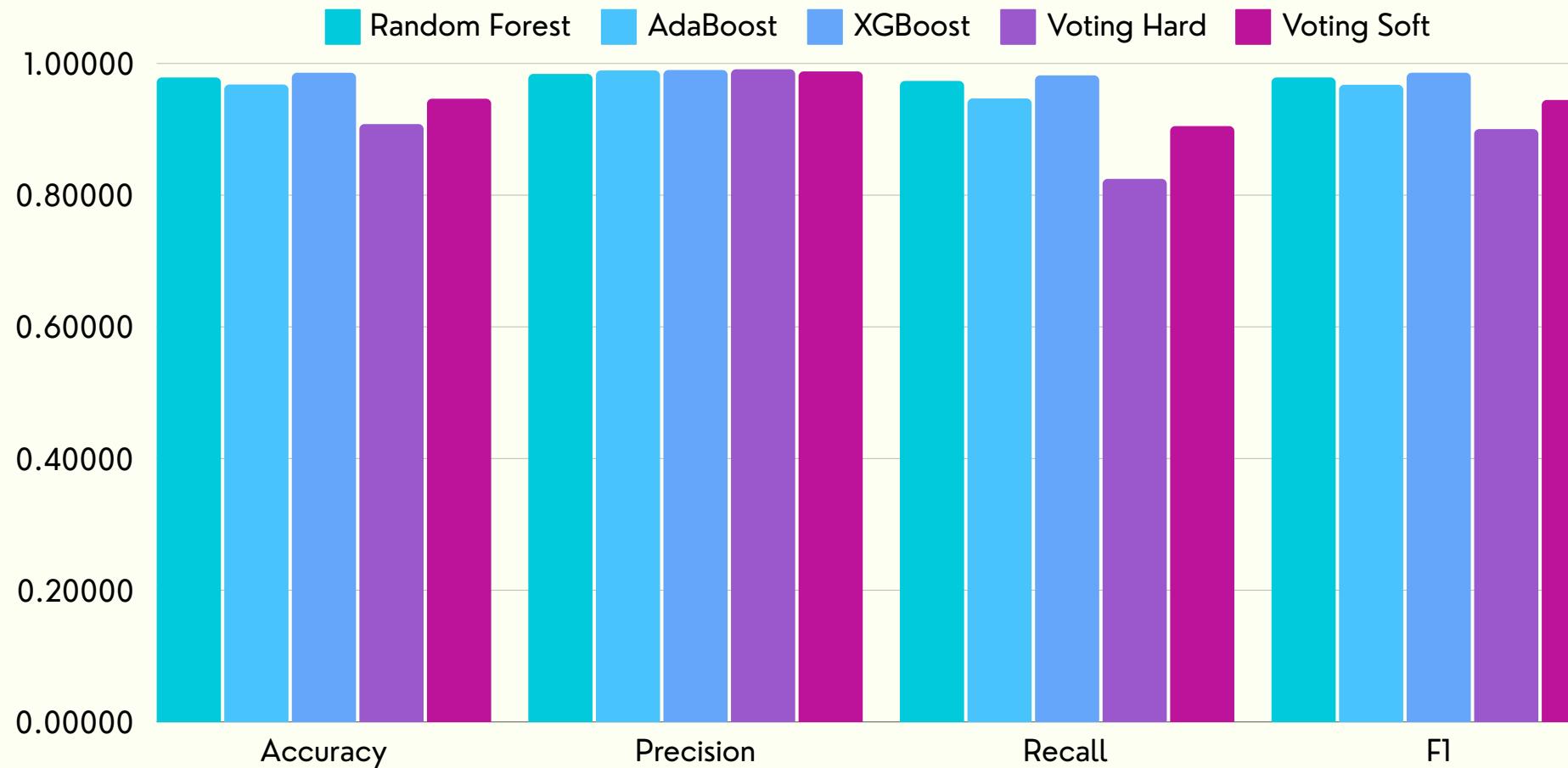


- SVM non è stato ottimizzato per questioni computazionali;
- Piccoli miglioramenti, in particolare per il classificatore KNN.

# Risultati classificatori ensemble

Sono modelli noti per la loro capacità di **migliorare significativamente la performance di classificazione** aggregando i risultati di più modelli di base:

- **Bagging**, consiste nella costruzione di più modelli indipendenti su sottoinsiemi diversi dei dati di addestramento e poi aggrega le loro previsioni;
- **Boosting**, consente di combinare più modelli deboli per realizzare un modello forte e accurato dopo varie iterazioni;
- **Voting**, consiste nel combinare le previsioni provenienti da più modelli per fare una previsione finale. È un approccio di ensemble learning che aggrega i risultati di diversi classificatori base e predice l'output basandosi sulla maggioranza dei voti.



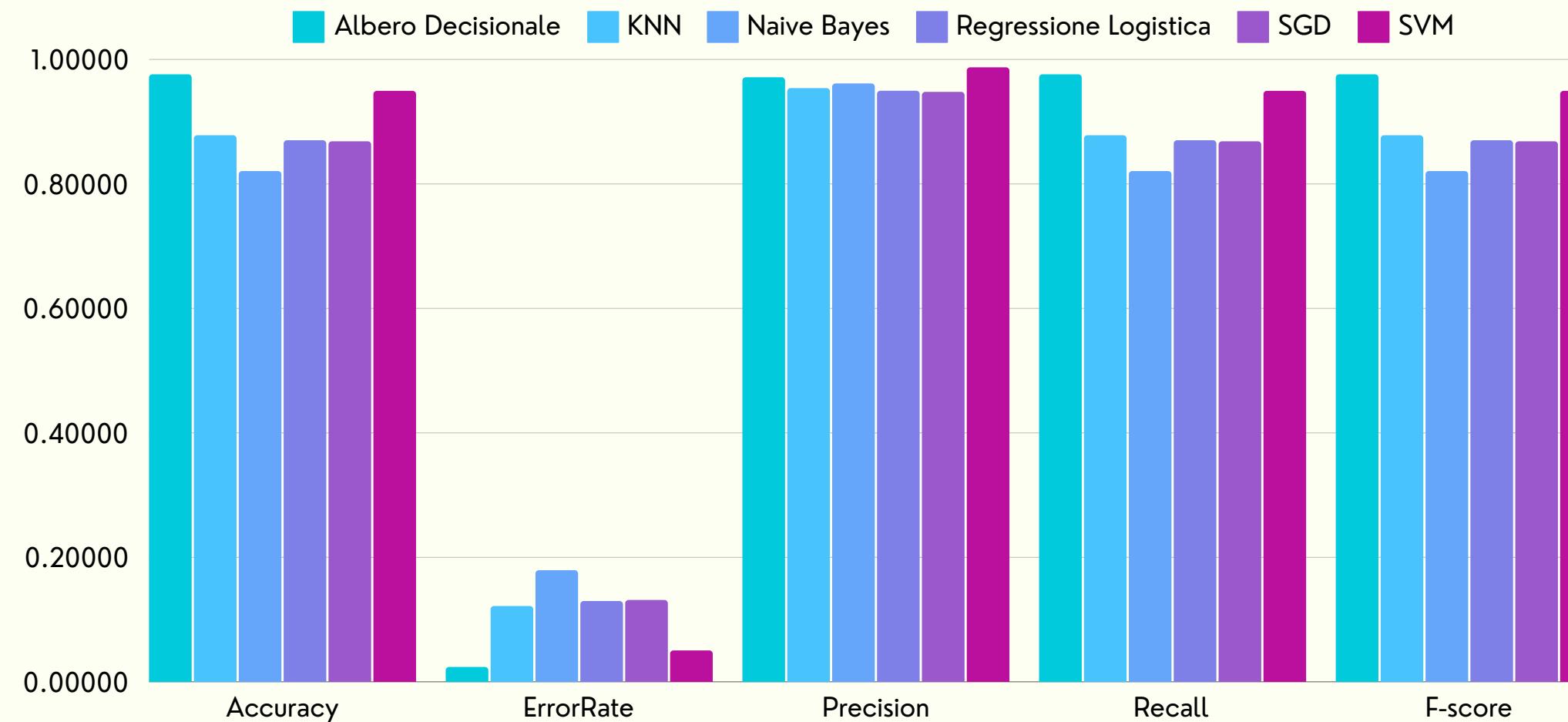
RandomForest rientra nella categoria dei modelli bagging, mentre XGBoost e AdaBoost nel boosting.

Il voting è quello che performa peggio.

# Risultati cross validation

La cross-validation è una tecnica utilizzata per valutare l'abilità di generalizzazione di un modello di data mining su un set di dati indipendente, assicurando che il modello sia robusto e performante su dati non visti durante l'addestramento.

- Il training set viene diviso in un **numero predefinito k di parti o "fold"** di dimensioni uguali;
- Per ogni iterazione (in totale sono k) della cross-validation **viene selezionato uno dei k fold come test di validazione del modello addestrato sui restanti k-1 fold**. A questo punto si memorizzano le performance del modello relative all'iterazione corrente;
- Al termine delle k iterazioni si calcola la **media delle performance registrate in ogni iterazione** per fornire una stima complessiva della performance del modello.



**k = 10 # fold della cross validation**

I risultati della cross-validation, confermano che si è evitato il problema della dipendenza da una particolare partizione.

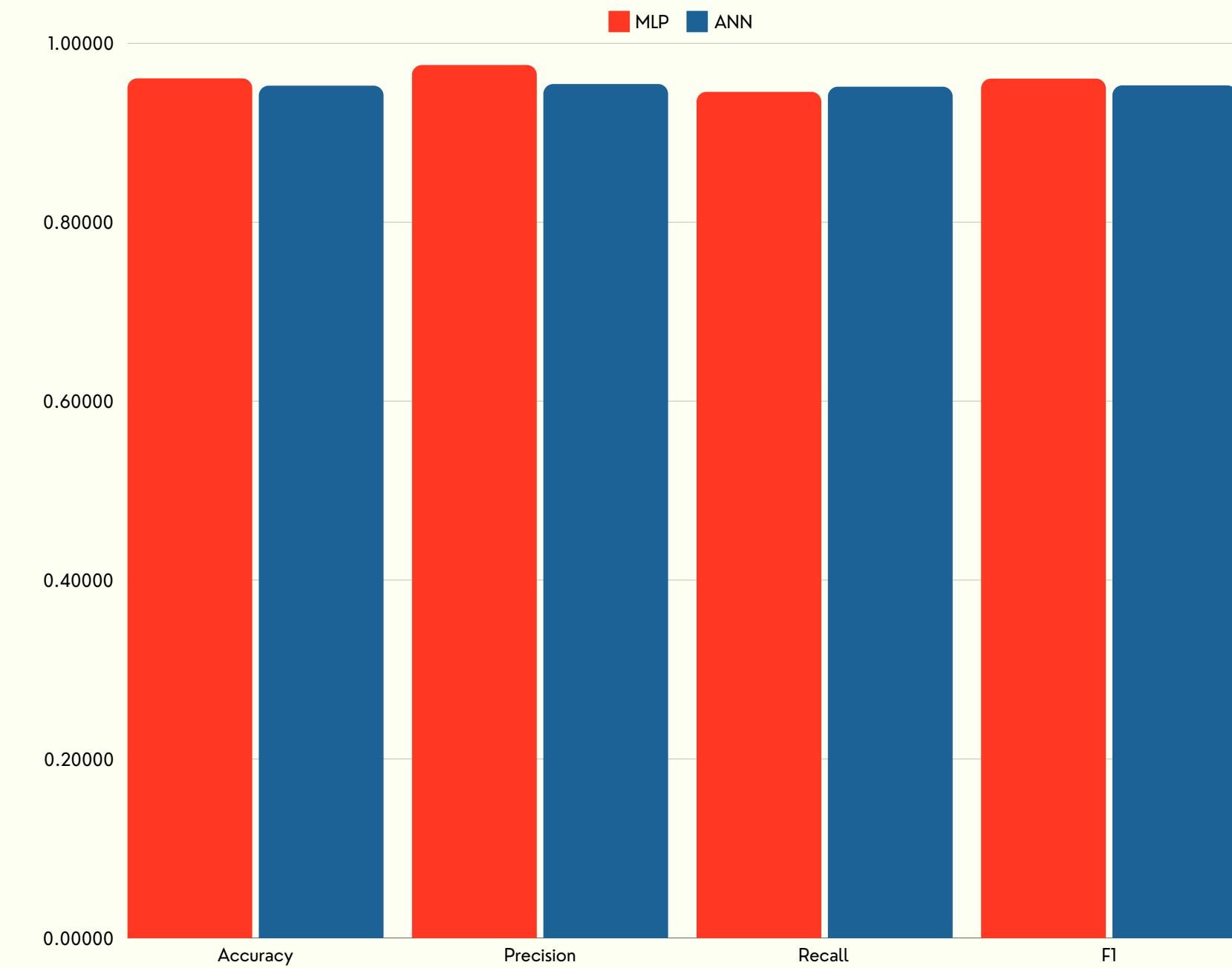
È possibile notare grandi miglioramenti soprattutto per il classificatore Naive Bayes.

# Risultati reti neurali

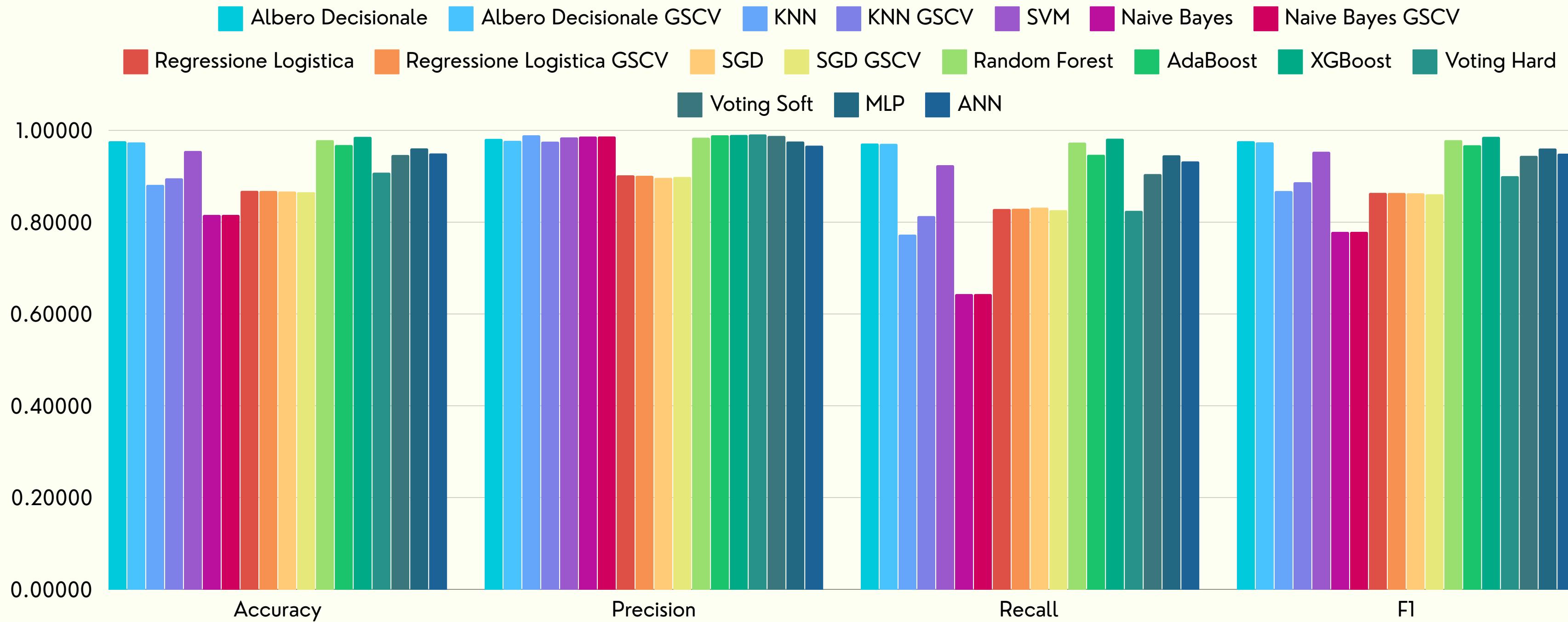
È stata implementata una rete neurale custom (ANN) con le librerie Keras e Tensorflow per confrontarla con una rete MLP (singolo livello nascosto con 100 neuroni).

## Quali sono le caratteristiche della ANN?

- Modello sequenziale con **5 strati** (1 input, 3 nascosti, 1 output);
- Strati nascosti con 64 e 128 neuroni e **attivazione ReLU**;
- Strato di output con **attivazione sigmoid** per classificazione binaria;
- Compilazione con perdita **binary\_crossentropy**, ottimizzatore **SGD**, e metriche di **Recall**, **Precision** e **BinaryAccuracy**;
- Normalizzazione dei dati con **StandardScaler**;
- **Early stopping** basato su recall per prevenire l'overfitting;
- Addestramento per **massimo 500 epoch**, batch size di 1024;
- Learning rate di default (0.001).



# Valutazioni generali delle performance (test set)



# Conclusioni

Tra i problemi affrontati:

- gestione Nan, duplicati, attributi inutili, outlier
- dataset sbilanciato
- implementazione dei modelli

Tutti i modelli **GridSearchCV** sono ottimizzati per la metrica **Recall**:

- Modello migliore: **XGBoost** con **0.981903**
- Modello peggiore: **Naive Bayes** con **0.643433**