Andrea Augello
Department of Engineering, University of Palermo, Italy

Alberi di decisione

Introduzione

Introduzione

- Gli alberi di decisione sono un metodo di apprendimento supervisionato
- Sono utilizzati per la classificazione e la regressione
- Sono un modello di apprendimento interpretabile e facilmente visualizzabile
- Utilizzano semplici regole di decisione inferite dai dati di addestramento

Vantaggi e svantaggi

- Vantaggi
  - Semplice da capire e interpretare
  - Richiede poca preparazione dei dati
  - Può gestire sia dati numerici che categorici
  - Non necessita di assunzioni sulle distribuzioni dei dati
  - Può gestire problemi multi-classe
  - Richiede poche risorse per l'inferenza

Vantaggi e svantaggi

- Vantaggi
  - Semplice da capire e interpretare
  - Richiede poca preparazione dei dati
  - Può gestire sia dati numerici che categorici
  - Non necessita di assunzioni sulle distribuzioni dei dati
  - Può gestire problemi multi-classe
  - Richiede poche risorse per l'inferenza
- Svantaggi
  - Sono facilmente soggetti all'overfitting
  - Possono essere instabili, piccole variazioni nei dati possono portare a grandi variazioni nella struttura dell'albero
  - Necessitano di classi bilanciate
  - L'output non è continuo, ma a step

Come funziona?

Come funziona?

- All'arrivo di un vettore di input, l'albero di decisione esegue una serie di test per determinare la classe di appartenenza.
- Le decisioni sono tipicamente binarie, del tipo "*feature* $x_i \leq \alpha$?" quindi ogni nodo dell'albero ha due figli. (Esistono anche altri tipi di alberi)
- Divisioni successive dello spazio delle feature creano regioni corrispondenti alle classi. (Non tutte le feature vengono necessariamente utilizzate)

Algoritmo di inferenza

Funzione PREDICI (x, albero)
Output: Classe predetta

nodo $\leftarrow$ radice dell'albero
while nodo non è una foglia **do** # Non ha una label associata
    if $x[nodo.feature] \leq nodo.soglia$ **then**
        nodo $\leftarrow$ figlio sinistro del nodo
    else
        nodo $\leftarrow$ figlio destro del nodo
    end if
end while
return etichetta associata al nodo

Algoritmo di addestramento

Funzione ADDESTRA ($dataset$, $features$)
Output: Albero di decisione

if STOPCONDITION **then**
    return Nodo foglia con classe più appropriata
end if
$feature, soglia \leftarrow$ SCEGLISPLIT($dataset$, $features$)
for $e \in Dataset$ **do**
    if $e[feature] \leq soglia$ **then**
        $dataset_1 \leftarrow e$
    else
        $dataset_2 \leftarrow e$
    end if
end for
$nodo \leftarrow$ NUOVONODO($feature$, $soglia$)
$nodo.figlio_1 \leftarrow$ ADDESTRA($dataset_1$, $features$)
$nodo.figlio_2 \leftarrow$ ADDESTRA($dataset_2$, $features$)
return $nodo$

Scelte implementative

Ad ogni step di divisione dell’albero ci sono delle scelte da fare:

- Come si sceglie la feature da utilizzare per lo split successivo?
- Come si sceglie il valore di soglia?
- Come si determina quando fermarsi?
- Come si determina la classe di un nodo foglia?

Esistono diverse strategie per rispondere a queste domande, che portano a diverse implementazioni di alberi di decisione.

Scelta di feature e soglie

Soglie

- Per ogni feature x_i, ogni possibile valore di soglia α può determinare uno split differente.
- Potenzialmente quindi ci sono infiniti split possibili tra cui scegliere.
- In pratica ci si limita ad i valori di soglia che corrispondono ai punti medi tra due valori consecutivi di x_i osservati nel dataset.
- I possibili punti di scelta ad ogni nodo sono quindi dati dal numero di feature moltiplicato per la cardinalità del dataset.

Come scegliere quale tra questi punti di scelta utilizzare?

Scelta di feature e soglie

Criterio di split

- Ogni divisione di un nodo genera due discendenti.
- È ragionevole voler scegliere la divisione che genera i discendenti più omogenei possibile rispetto alla divisione originale.

Scelta di feature e soglie

Criterio di split

- Ogni divisione di un nodo genera due discendenti.
- È ragionevole voler scegliere la divisione che genera i discendenti più omogenei possibile rispetto alla divisione originale.
- Due i criteri più utilizzati:

Indice di Gini

$$Gini = 1 - \sum_{i=1}^{n} p_i^2$$

Entropia di Shannon

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i$$

- La diminuzione dell'impurità di un nodo è data dalla somma pesata delle impurità dei nodi figli.

$$\Delta I = I_{parent} - \frac{N_1}{N} I_1 - \frac{N_2}{N} I_2$$

Scelta di feature e soglie

Criterio di split

- Ogni divisione di un nodo genera due discendenti.
- È ragionevole voler scegliere la divisione che genera i discendenti più omogenei possibile rispetto alla divisione originale.
- Due i criteri più utilizzati:

Indice di Gini

$$Gini = 1 - \sum_{i=1}^{n} p_i^2$$

Entropia di Shannon

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i$$

- La diminuzione dell'impurità di un nodo è data dalla somma pesata delle impurità dei nodi figli.

$$\Delta I = I_{parent} - \frac{N_1}{N} I_1 - \frac{N_2}{N} I_2$$

- Tra i possibili punti di scelta si sceglie quello che massimizza la diminuzione dell'impurità.

Quando fermarsi?

- Threshold di impurità
- Numero minimo di esempi in un nodo
- Profondità massima dell’albero

Cosa fare se un nodo foglia non è puro?

- Assegnare la classe più frequente

Un esempio manuale

Un esempio manuale

Dataset di esempio

- Due feature x_1 e x_2
- Tre classi
- 10 esempi

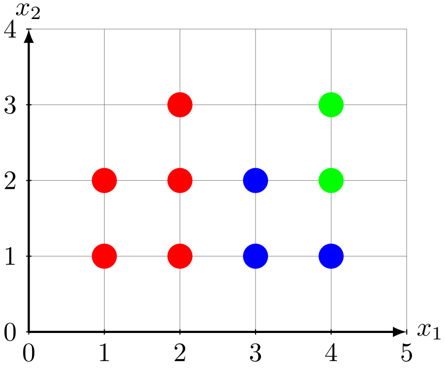
x_1	x_2	y
1	1	1
1	2	1
2	1	1
2	2	1
2	3	1
3	1	2
3	2	2
4	1	2
4	2	3
4	3	3

Un esempio manuale

Dataset di esempio

- Due feature x_1 e x_2
- Tre classi
- 10 esempi

x_1	x_2	y
1	1	1
1	2	1
2	1	1
2	2	1
2	3	1
3	1	2
3	2	2
4	1	2
4	2	3
4	3	3

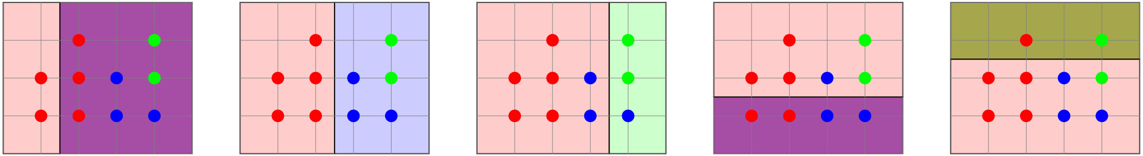


Inidice di Gini iniziale:

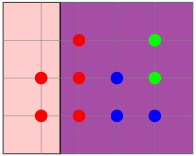
$$1 - \left(\frac{5}{10}\right)^2 - \left(\frac{3}{10}\right)^2 - \left(\frac{2}{10}\right)^2 = 0.62$$

Un esempio manuale

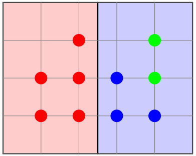
Possibili split:



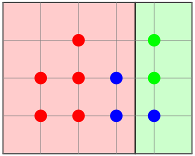
Possibili split:



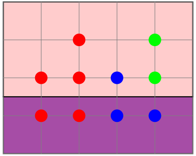
$\Delta I = 0.62 - \frac{2}{10} \cdot 0 - \frac{8}{10} \cdot 0.66 = \mathbf{0.09}$



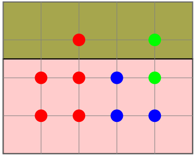
$\Delta I = 0.62 - \frac{5}{10} \cdot 0 - \frac{5}{10} \cdot 0.48 = \mathbf{0.38}$



$\Delta I = 0.62 - \frac{7}{10} \cdot 0.40 - \frac{3}{10} \cdot 0.44 = \mathbf{0.21}$

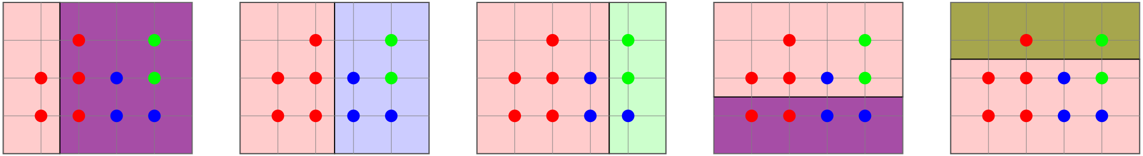


$\Delta I = 0.62 - \frac{6}{10} \cdot 0.61 - \frac{4}{10} \cdot 0.5 = \mathbf{0.05}$



$\Delta I = 0.62 \frac{2}{10} \cdot 0.5 - \frac{8}{10} \cdot 0.59 = \mathbf{0.05}$

Possibili split:



$\Delta I =$
$0.62 - \frac{2}{10} \cdot 0 - \frac{8}{10} \cdot 0.66 =$
$\mathbf{0.09}$

$\Delta I =$
$0.62 - \frac{5}{10} \cdot 0 - \frac{5}{10} \cdot 0.48 =$
$\mathbf{0.38}$

$\Delta I =$
$0.62 - \frac{7}{10} 0.40 - \frac{3}{10} \cdot 0.44 =$
$\mathbf{0.21}$

$\Delta I =$
$0.62 - \frac{6}{10} \cdot 0.61 - \frac{4}{10} \cdot$
$0.5 = \mathbf{0.05}$

$\Delta I =$
$0.62 \frac{2}{10} \cdot 0.5 - \frac{8}{10} \cdot 0.59 =$
$\mathbf{0.05}$

Il maggior guadagno di informazione si ottiene con lo split sulla feature x_0 con soglia $\alpha = 2.5$.

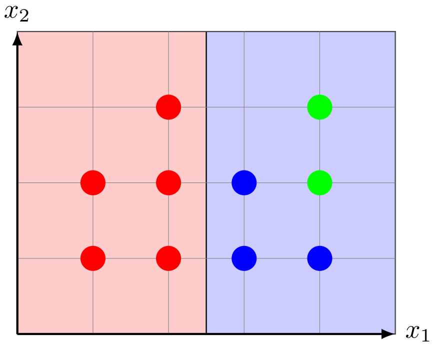
Un esempio manuale

Primo split:

- Scegliamo la feature x_0
- Scegliamo il valore di soglia $\alpha = 2.5$

Effetto:

- Inidice di Gini iniziale: 0.62
  $G = 1 - \left(\frac{5}{10}\right)^2 - \left(\frac{3}{10}\right)^2 - \left(\frac{2}{10}\right)^2 = 0.62$
- Indice di Gini nodo sinistro: $G_1 = 0$
- Indice di Gini nodo destro:
  $G_2 = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0.48$
- $\Delta I = 0.62 - \frac{5}{10} \cdot 0 - \frac{5}{10} \cdot 0.48 = 0.38$



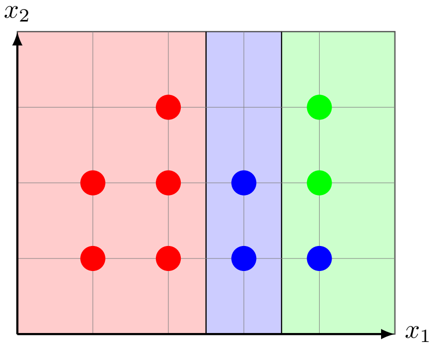
Un esempio manuale

Secondo split (sul figlio destro, il sinistro è puro):

- Scegliamo la feature x_0
- Scegliamo il valore di soglia $\alpha = 3.5$

Effetto:

- Indice di Gini iniziale: $G = 0.48$
- Indice di Gini nodo sinistro: $G_1 = 0$
- Indice di Gini nodo destro: $G_2 = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.66$
- $\Delta I = 0.48 - \frac{3}{5} \cdot 0 - \frac{2}{5} \cdot 0.66 = 0.22$



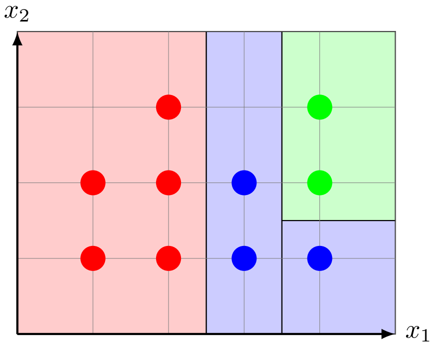
Un esempio manuale

Ultimo split:

- Scegliamo la feature x_1
- Scegliamo il valore di soglia $\alpha = 1.5$

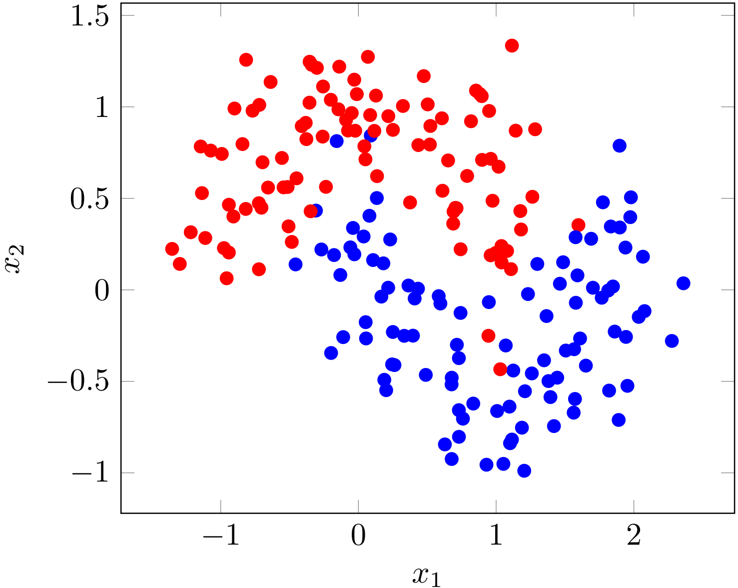
Effetto:

- Indice di Gini iniziale: $G = 0.66$
- Indice di Gini nodo sinistro: $G_1 = 0$
- Indice di Gini nodo destro: $G_2 = 0$
- $\Delta H = 0.66 - \frac{1}{3} \cdot 0 - \frac{2}{3} \cdot 0 = 0.66$



Implementazione

Dataset



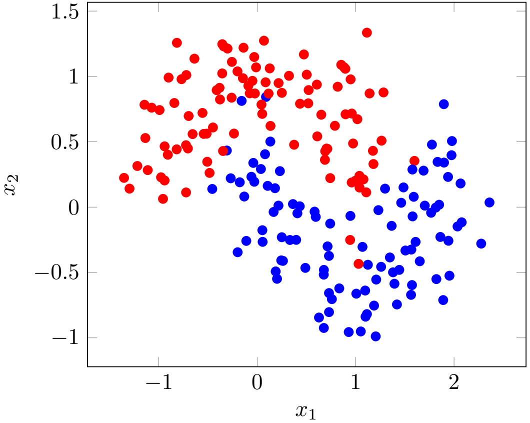
Implementazione

```python
import utils
from manual_tree import DecisionTree
def main():
    X,y = utils.load_data("dataset.dat")
    tree = DecisionTree(max_depth=3)
    tree.fit(X, y)
    utils.classification_stats(y, tree(X))
    utils.plot(X,y, tree, pause=True)

    for depth in range(1,9):
        tree = DecisionTree(                # Creazione del classificatore
                max_depth=depth)            # Profondità massima
        tree.fit(X, y, quiet=True)          # Addestramento
        print(f"Depth: {depth}")
        utils.classification_stats(y, tree(X))
        utils.plot(X,y, tree, title=f"Depth: {depth}")

if __name__ == "__main__":
    main()
```

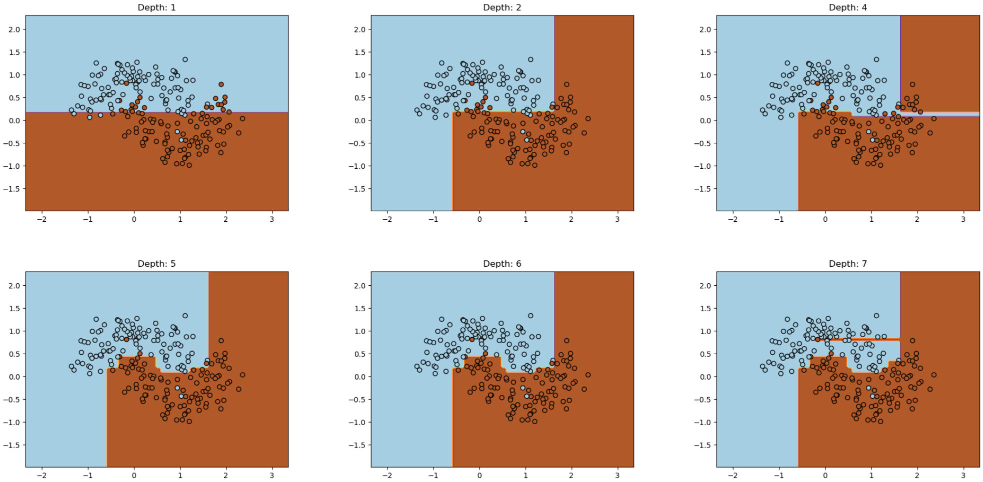
Implementazione

Sistema di decisione multilivello:

```
x_1 <= 0.20
|     x_0 <= -0.59
|     |     class: 0
|     |     class: 1
|     x_0 <= 1.64
|     |     class: 0
|     |     class: 1
```



Visualizzare lo spazio delle feature



Valutare la bontà della classificazione

- Accuratezza: $\frac{TP+TN}{TP+TN+FP+FN}$
- Precisione: $\frac{TP}{TP+FP}$
- Recall: $\frac{TP}{TP+FN}$
- F1: $2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$

Non rimane che implementare l'albero

```
class Node():
        def __init__():
        def __str__():
        def _forward():
        def _gini():
        def _best_split():
        def _fit():

class DecisionTree(Node):
        def __init__(self, max_depth=3):
        def fit(self, X, y):
        def __call__():
```