Alberi di decisione
Confronto tra Entropia e Errore di Classificazione

Giovanni Cellai

Luglio 2019

Decision tree

In questo progetto è stato implementato l'algoritmo di DecisionTreeLearning
descritto nella sezione 18.3 di Russel Norvig, utilizzando sia l'entropia che l'errore di classificazione come misure di impurità.

L'algoritmo

L'algoritmo prende in ingresso un data set e un valore (*ce*) per cambiare misura di impurità, ritorna l'albero di decisione costruito tramite entropia o errore di classificazione.
L'algoritmo è stato ripreso e adattato dal Repository pubblico.

Data sets

I data sets utilizzati reperiti da UCI sono:
1. Blood Transfusion: size: 510, inputs: 5, classes: 2
2. Haberman: size: 240, inputs: 4, classes: 2
3. Iris: size: 150, inputs: 5, classes: 3 (Stringhe)
4. Mammographic Masses: size: 270, inputs: 6, classes: 2

Tali data sets sono contenuti in file .txt e hanno valori continui.
La grandezza size è sempre un multiplo di 5 per agevolare le operazioni della 5 fold cross validation.

Descrizione classi

Il progetto contiene i seguenti file .py

DataSet.py

La classe contiene la definizione del dataset, essa contiene i seguenti attributi:

```
# examples      : is the list of all examples of the dataset
# target        : is the position of the attribute that indicates the classification of the example
# values        : contains the values of the classification
# inputs        : attributes except the classification
# attributes    : all the attributes
# attrnames     : name of the attributes in the dataset
```

Contiene inoltre il metodo setDataSet che fa il parsing del file .txt e ritorna un oggetto DataSet: per ogni riga crea un example in una lista (semplicemente chiamata examples) e considera le virgole nel
.txt per separare i valori di differenti attributi sulla stessa riga.

DecisionTree.py

La classe definisce la struttura dell'albero di decisione, descritta dagli attributi:
attribute # valore dell'attributo salvato nel nodo
threshold # valore di soglia dell'attributo
attrname # nome dell'attributo o valore di esso
default_child # puntatore a figlio
branches # lista di diramazioni dal nodo
Left = 0 # valore <= della threshold
Right = 1 # valore > della threshold

Sono inoltre presenti funzioni per aggiungere un sotto albero ad un nodo e due funzioni get per ritornare l'attributo e la soglia di un nodo.
Di fondamentale importanza è la funzione GraphViz che visita semplicemente per livelli l'albero (breadth first) e crea automaticamente un file .gv in linguaggio DOT che viene interpretato dalla libreria GraphViz e permette di visualizzarlo in un .pdf.
Durante l'esecuzione verranno creati due pdf, uno per misura d'impurità, per rendere più accessibile l'analisi dell'albero di decisione.

Attenzione: per una seconda esecuzione del Main.py si ricorda di chiudere tali file .pdf per non riscontrare errori di compilazione.

Infine viene definita la classe Leaf che appunto definisce le foglie del nostro albero decisionale.

FiveFold.py

Il file contiene i metodi per attuare il Five Fold Cross Validation, sia per il Classification Error che per l'Entropy.
Il file contiene inoltre il metodo test(fileDataset) che restituisce un grafico per rendere il confronto tra misure di impurità più accessibile.

DecisionTreeLearning.py

Il file contiene le funzioni per creare un decision tree come descritto da Mitchell, switchando a seconda del valore di *ce* (0:entropy, 1:classification error)

Esempio di funzionamento

1. Eseguire il file Main.py
2. Inserire un valore da 1 a 4 per selezionare uno tra i seguenti:
   1) Blood Transfusion 2) Iris 3) Haberman 4) Mammographic Masses

Attenzione: Graphviz è una libreria esterna che, per funzionare, deve essere linkata all'interno del programma:
dobbiamo modificare la riga 03 del file DecisionTree.py

os.environ["PATH"] += os.pathsep + 'C:\Program Files (x86)\Graphviz2.38/bin' #da cambiare al

momento dell'istallazione

sostituendo il percorso dell'eseguibile scaricato precedentemente.
OPPURE (consigliato)
al momento del download della libreria, salvarla in 'C:\Program Files (x86)\Graphviz2.38/bin'

Risultati

I risultati qui riportati sono stati ottenuti da un laptop con processore Intel Core i5-8250U e 8GB di Ram. I test condotti non sono influenzati dalle prestazioni
del calcolatore poiché si considerano soltanto le classificazioni corrette. Questi
risultati potrebbero essere analoghi a quelli su un calcolatore molto più potente.
Il tempo di esecuzione su un dataset è tipicamente inferiore ai 10 secondi.
I seguenti dati sono generiche *run* del programma

Dataset 1:
Entropia
73.72549019607844 % classificate correttamente in media, ovvero 376 classificazioni corrette su 510
Errore di classificazione
71.17647058823529 % classificate correttamente in media, ovvero 363 classificazioni corrette su 510

Dataset 2:
Entropia
93.33333333333334 % classificate correttamente in media, ovvero 140 classificazioni corrette su 150
Errore di classificazione
92.66666666666666 % classificate correttamente in media, ovvero 139 classificazioni corrette su 150

Dataset 3:
Entropia
69.99999999999999 % classificate correttamente in media, ovvero 168 classificazioni corrette su 240
Errore di classificazione
72.91666666666667 % classificate correttamente in media, ovvero 175 classificazioni corrette su 240

Dataset 4:
Entropia
85.90909090909092 % classificate correttamente in media, ovvero 189 classificazioni corrette su 220
Errore di classificazione
85.45454545454547 % classificate correttamente in media, ovvero 188 classificazioni corrette su 220

Analisi e conclusioni finali

Dai risultati precedenti notiamo che i dataset hanno comportamenti e reazioni all'algoritmo differenti tra loro, possiamo però notare che DecisionTreeLearning funziona correttamente, fornendo un tasso di risposta positivo (percentuali sempre maggiori al 70%), particolarmente utilizzando l'entropia.
Notiamo comunque che la differenza tra le due misure di impurità è comunque inferiore ai 3 punti percentuale e vediamo la particolare efficenza dell'entropia soprattutto con dataset di maggiore grandezza.
La misura d'impurità "Errore di classificazione" rimane comunque una scelta valida, in modo particolare per i dataset con pochi dati (vedi risultati Dataset 3).

Concludendo, l'algoritmo DecisionTreeLearning è efficace e consistente utilizzando sia l'entropia che l'errore di classificazione come misure d'impurità, confermando i risultati studiati dalla teoria.