Decision Tree e Random Forest a confronto

Sofia Galante

February 5, 2021

1 Introduzione

Lo scopo di questo elaborato è mettere a confronto due diversi classificatori: il **Decision Tree** e la **Random Forest** e di far notare come il secondo sia migliore rispetto al primo.

Il Decision Tree è un classificatore potente e semplice, costituito da un albero in cui:

1. ogni nodo corrisponde ad un test su un singolo attributo;
2. gli archi rappresentano le possibili risposte ad un test.

Durante la fase di **train**, si genera l'albero utilizzando un training set. Nella fase di **test**, il Decision Tree viene utilizzato per predire la classe degli esempi presenti in un test set.

La Random Forest è una collezione di Decision Tree. Durante la fase di **train**, l'algoritmo genera un numero predefinito di alberi andando a dividere il training set in modo casuale per ogni diverso albero. Quando si utilizza la Random Forest per predire una classe di un esempio, nella fase di **test**, l'algoritmo compie l'azione su tutti gli alberi presenti nella collezione e, infine, sceglie come risultato finale il valore che è stato ritornato più volte.

2 Esperimento svolto

L'esperimento consiste nell'osservare e mettere a confronto le *accuracy* ottenute con l'utilizzo di un Decision Tree e di una Random Forest. In particolare, si sono svolti 10 test su 4 dataset diversi (per osservare come le differenze di prestazioni dei due classificatori fossero indipendenti dalla scelta del dataset). Si precisa che:

1. Durante la fase di train degli alberi, è stata utilizzata come **funzione di calcolo dell'impurità** di un dataset la funzione **Entropia** in quanto questa risulta essere la più precisa tra quelle da noi studiate.
2. Negli esperimenti svolti, la Random Forest è costituita da 10 alberi e ogni albero viene allenato con un training set costituito dal 40% di esempi e considerando solo il 60% degli attributi.

3. Ogni dataset è stato diviso in un set di train (80%) e uno di test (20%).

3 Documentazione del codice

Il codice è stato diviso in 5 diversi file python.

1. *node.py* → Questo file contiene la classe $Node$, utilizzata per costruire la struttura dell'albero. La classe presenta 3 attributi e 2 metodi (non contando il costruttore):
   (a) *nodeId* è l'identificatore del nodo (cioè l'attributo su cui si svolge il test);
   (b) $leaf$ è un valore booleano che indica se il nodo è una foglia ($True$) o no ($False$);
   (c) *childern* è una lista di figli del nodo;
   (d) $add\_child(self, y, v)$ aggiunge un figlio y a *children* e indica con v la risposta al test che il nodo ($self$) rappresenta.
2. *decisiontree.py* → Questo file contiene la classe $DecisionTree$. Questa classe contiene un singolo attributo e 6 metodi (non contando il costruttore):
   (a) *root* contiene la radice dell'albero;
   (b) $train(self, D, X, Y)$ & $\_do\_train(self, D, X, Y)$ addestrano il DecisionTree utilizzando il dataset D;
   (c) $predict(self, D)$ & $\_do\_predict(self, tree, D)$ utilizzano l'albero per predire la classe degli esempi nel dataset D;
   (d) $\_cost(self, D, Y)$ & $\_gain(self, D, X, Y)$ sono metodi che vengono utilizzati durante l'addestramento e servono a determinare su quale attributo compiere il test (e quindi quale nodo aggiungere nell'albero).
3. $randomforest.py$ → Questo file contiene la classe $RandomForest$, con i suoi 4 attributi e 2 metodi (non contando il costruttore):
   (a) *num_trees* è il numero di alberi di cui è composta la Random Forest;
   (b) *max_X* & *max_samples* rappresentano la percentuale di attributi che la Random Forest utilizza per addestrare i Decision Tree di cui è composta e la percentuale del numero di esempi considerati;
   (c) *trees* è una lista contenente tutti gli alberi della Random Forest;
   (d) $train(self, D, X, Y)$ addestra i Decision Tree della Random Forest con un sottoinsieme di D (determinato dagli attributi sopra descritti);
   (e) $predict(self, D)$ utilizza gli alberi per predire le classi di D. Il risultato finale viene scelto attraverso un "voto per maggioranza".
4. *test.py* → Questo file contiene i due metodi usati per svolgere i test:

(a) $test(Datasets, X, Y)$ divide i dataset in training set e test set secondo la percentuale impostata e in modo che la divisione mantenga la stessa proporzione delle classi in entrambi i set. Salva inoltre i risultati del test (accuracy dei due algoritmi e media e variazione standard di quest'ultima) su file csv;

(b) $accuracy(y\_true, y\_pred)$ calcola l'accuracy dell'algoritmo corrispondente mettendo a confronto le classi predette con quelle corrette.

5. *datasets.py* → Questo file, infine, contiene tutti i parametri modificabili per svolgere diversi esperimenti e il metodo necessario per importare i dataset ($get\_datasets()$) nel programma.

4 Presentazione e analisi dei risultati sperimentali

	DT - Accuracy	RF - Accuracy
Test1	0.4553	0.6911
Test2	0.4228	0.7805
Test3	0.374	0.7642
Test4	0.4146	0.7724
Test5	0.4715	0.8211
Test6	0.3902	0.7724
Test7	0.4146	0.8049
Test8	0.4228	0.8374
Test9	0.4309	0.813
Test10	0.3984	0.8293

DT - Mean	DT - STD	RF - Mean	RF - STD
0.4195	0.0275	0.7886	0.0408

Table 1: Dataset 1 - Results

	DT - Accuracy	RF - Accuracy
Test1	0.9419	0.9419
Test2	0.8953	0.907
Test3	0.9535	0.9302
Test4	0.8953	0.9535
Test5	0.9651	0.9767
Test6	0.9302	0.9186
Test7	0.9884	0.9884
Test8	0.8953	0.9535
Test9	0.9302	0.9651
Test10	0.9302	0.9419

DT - Mean	DT - STD	RF - Mean	RF - STD
0.9326	0.0298	0.9477	0.024

Table 2: Dataset 2 - Results

	DT - Accuracy	RF - Accuracy
Test1	0.8743	0.8063
Test2	0.8482	0.7435
Test3	0.7696	0.7853
Test4	0.7853	0.7592
Test5	0.8743	0.8482
Test6	0.8168	0.7801
Test7	0.8586	0.801
Test8	0.822	0.8534
Test9	0.8534	0.7696
Test10	0.8063	0.8063

DT - Mean	DT - STD	RF - Mean	RF - STD
0.8309	0.0347	0.7953	0.0338

Table 3: Dataset 3 - Results

	DT - Accuracy	RF - Accuracy
Test1	0.5179	0.6786
Test2	0.6071	0.6607
Test3	0.75	0.7321
Test4	0.625	0.7321
Test5	0.6071	0.6964
Test6	0.4464	0.6786
Test7	0.6429	0.7321
Test8	0.5536	0.625
Test9	0.5714	0.6786
Test10	0.5	0.6964

DT - Mean	DT - STD	RF - Mean	RF - STD
0.5821	0.0807	0.6911	0.033

Table 4: Dataset 4 - Results

Come si può vedere dalle tabelle, i risultati sperimentali confermano le aspettative iniziali: nella maggior parte dei test svolti l'algoritmo che utilizza il Decision Tree ha portato un'accuracy minore dell'algoritmo che utilizza la Random Forest.
Si nota inoltre che la Random Forest non ha solo una media migliore, ma anche una variazione standard più piccola di quella del Decision Tree.