Alberi decisionali Gestione intelligente delle richieste di assistenza

Prof. Fedeli Massimo

1 Introduzione

In questo documento viene illustrato il funzionamento di un **albero decisionale** attraverso un esempio realistico e vicino all'esperienza quotidiana: la classificazione delle richieste di assistenza tecnica in base alla loro priorità.

L'obiettivo è spiegare come un algoritmo di *machine learning* possa supportare decisioni organizzative, rendendole più rapide, coerenti e motivate.

2 Il problema affrontato

In molti contesti, come scuole, aziende o uffici informatici, arrivano quotidianamente richieste di assistenza. Non tutte hanno la stessa importanza: alcune possono attendere, altre richiedono un intervento immediato.

Il problema consiste nel determinare automaticamente la **priorità** di una richiesta sulla base di alcune informazioni fornite dall'utente.

Ogni richiesta è descritta tramite quattro caratteristiche:

- tipo di problema (software, hardware, rete);
- numero di utenti coinvolti;
- impatto sul servizio;
- urgenza dichiarata.

A queste informazioni è associata un'etichetta, chiamata *Priorità*, che rappresenta la decisione finale da prendere.

3 Classificazione supervisionata

Il problema rientra nella **classificazione supervisionata**. Il modello viene addestrato su un insieme di esempi per i quali la priorità corretta è già nota.

L'algoritmo analizza questi esempi e impara una serie di regole che collegano le caratteristiche della richiesta alla priorità finale.

4 Preparazione dei dati

Il dataset è memorizzato in un file CSV e viene caricato tramite la libreria `pandas`. Poiché le informazioni sono espresse in forma testuale, è necessario trasformarle in valori numerici prima di poterle utilizzare.

Questa operazione viene svolta tramite il **Label Encoding**. Ogni valore testuale viene sostituito da un numero intero.

Ad esempio:

software – 0, hardware 1, rete 2

Per ogni colonna viene salvato il relativo codificatore, così da poter tradurre correttamente anche i dati inseriti successivamente dall'utente.

5 Suddivisione dei dati

Il dataset viene diviso in due parti:

- **training set** (70%), utilizzato per addestrare il modello;
- **test set** (30%), utilizzato per valutarne le prestazioni.

Questa separazione permette di verificare se l'albero decisionale è in grado di generalizzare, cioè di funzionare correttamente anche su dati mai visti prima.

6 L'albero decisionale

Il modello utilizzato è un `DecisionTreeClassifier`. Durante l'addestramento, l'albero costruisce una sequenza di decisioni del tipo:

> Se l'impatto è alto e l'urgenza è alta, allora la priorità è elevata.

A ogni nodo l'algoritmo sceglie la condizione che separa meglio le richieste, riducendo l'**impurità** dei gruppi ottenuti. In questo caso viene utilizzato il criterio di Gini.

7 Profondità dell'albero

La profondità massima dell'albero è limitata a 4 livelli. Questa scelta evita il fenomeno dell'**overfitting**, che si verifica quando il modello impara regole troppo specifiche e perde la capacità di generalizzare.

Un albero con profondità limitata rappresenta un buon compromesso tra:

- capacità di descrivere situazioni diverse;
- semplicità e leggibilità delle decisioni;
- robustezza del modello.

8 Valutazione del modello

Dopo l'addestramento, il modello viene testato sui dati di test. La prestazione viene misurata tramite l'**accuratezza**, cioè la percentuale di previsioni corrette.

Questo valore fornisce un'indicazione quantitativa dell'affidabilità del sistema.

9 Salvataggio e riutilizzo del modello

Una caratteristica importante del programma è il salvataggio del modello addestrato su file. In questo modo:

- l'addestramento non deve essere ripetuto ogni volta;
- il modello può essere utilizzato in momenti diversi;
- il sistema diventa più efficiente.

Insieme al modello vengono salvati anche i codificatori utilizzati per i dati.

10 Previsione di una nuova richiesta

L'utente può inserire manualmente i dati di una nuova richiesta di assistenza. Il programma:

1. codifica i valori testuali;
2. applica il modello addestrato;
3. riconverte il risultato in forma testuale;
4. mostra la priorità prevista.

Questo processo rende evidente come il modello possa essere utilizzato in un contesto reale.

11 Struttura del programma

Il programma è organizzato in funzioni, ciascuna con un compito preciso:

- caricamento e preparazione dei dati;
- addestramento del modello;
- previsione della priorità;
- gestione del menù.

Questa struttura migliora la leggibilità del codice e ne facilita la manutenzione.

12 Conclusione

L'albero decisionale si dimostra uno strumento efficace e facilmente interpretabile per la gestione delle priorità nelle richieste di assistenza.

L'esempio mostra come concetti di intelligenza artificiale possano essere applicati a problemi concreti, mantenendo trasparenza, semplicità e coerenza con il modo umano di prendere decisioni.