Elaborato del corso Machine Learning e Data Mining

di **Landi Federico** | mat. 713227

AA 2020/2021

Il dataset analizzato riguarda pazienti che sono stati sottoposti ad un intervento chirurgico al torace a causa di un carcinoma del polmone. Il carcinoma del polmone è la neoplasia con il maggior tasso di incidenza e di mortalità nel mondo ed è causa, ogni anno, di oltre 1 milione di morti.

Lo scopo dell'elaborato è stato quello di trovare un modello che sia in grado di classificare correttamente i pazienti che dopo un anno dall'operazione al torace siano sopravvissuti o deceduti (sopravvissuti=TRUE, deceduti=FALSE).

Maggiori dettagli su come sono state implementate le varie tecniche si possono trovare sul notebook Jupiter dell'elaborato.

Analisi Dati

Come prima cosa, è stata effettuata una rinominazione delle features sostituendo le sigle iniziali con sigle più comprensibili ed intuitive. Dopo di che si è proceduto a verificare la tipologia dei dati messi a disposizione del dataset e la presenza di eventuali valori nulli (non presenti). Successivamente, dopo aver ricavato alcuni grafici informativi, si è proceduto a controllare eventuali valori fuori scala e sono state rivelate anomalie in alcuni dati della feature FEV1. Inizialmente si è pensato di rimuovere del tutto i campioni contenenti i valori non corretti ma successivamente, dopo aver notato una forte correlazione tra le features FEV1 e FVC, si è adottato un regressore lineare per correggere i dati FEV1 grazie ai valori corretti di FVC. Infine i dati non numerici sono stati resi numerici grazie alla funzione LabelEncoder().

Previsione

Dopo aver suddiviso i dati in Training Set e Test Set, è stato utilizzato un semplice Decision Tree per avere un'idea di quanti fossero buoni i dati contenuti nel dataset. Il punteggio ottenuto è abbastanza basso (0.69) e guardando la matrice di confusione si nota che il classificatore riesce ad individuare i casi FALSE ma classifica scorrettamente i TRUE. Questa disparità è dovuta alla maggiore presenza di casi FALSE che di casi TRUE nel dataset originario. Per cercare di ottenere prestazioni migliori si è provato a cercare quale peso dare alle variabili TRUE rispetto alle variabili FALSE per aumentare il punteggio F1, cercando così di bilanciare i TP e TN. Il bilanciamento dei dati (che ha visto dare maggior risalto hai dati TRUE) non ha però migliorato la matrice di confusione ed il punteggio non ha subito miglioramenti rilevanti. (*nota: al posto del punteggio F1 si è provato a valutare anche il punteggio F2, utilizzando la funzione 'make_scorer ' con parametro 'f-beta*

score' pari a 2 ma il risultato privilegiava troppo i casi FALSE portando ad una bassa accuratezza del modello)

Data la disparità tra i dati TRUE e FALSE si è provato ad applicare 2 tecniche di data augmentation: SMOTE (o meglio una sua variante, denominata SMOTENC) e Adasyn. Inizialmente la prima tecnica è stata erroneamente utilizzata sull'intero dataset compromettendo il risultato finale: aumentando sia il training set che il test set è logico che i dati artificiali vengano classificati correttamente. Si è così proceduto ad aumentare solo i dati del Training set con il metodo Adasyn. Purtroppo anche in questo caso i risultati non hanno avuto miglioramenti.

Successivamente sono state adottate altre tecniche di machine learning più avanzate, come ExtraTrees, SVM, XGboost e Stacking ma tutte quante hanno portato ad un modello che non riesce a classificare le classi TRUE, classificandole tutte come FALSE. Si è notato che più è potente l'algoritmo di previsione e più il sistema effettua una previsione sbilanciata verso i FN. Un miglioramento si è notato solo con una Neuran network ma in quel caso si è peggiorata drasticamente la classificazione dei FALSE e in generale i grafici sulle prestazioni ottenuti dalla rete sono fortemente randomici.

Data la tendenza degli algoritmi di classificazione avanzati di sbilanciare la classificazione verso i TN si è ritornato ad adottare un Decision Tree ma questa volta provando ad eliminare le feature cercando di rendere meno specifico l'albero. Attraverso la funzione SequentialFeatureSelector() messa a disposizione da Mlxtend è stato possibile utilizzare solo 3 features non solo mantenendo lo stesso numero di classificazioni TN ma migliorando del doppio la classificazione dei TP. Utilizzando un DT con tutte le feature si ottiene con cross_val_score() un punteggio di 0.75 mentre con DT con solo 3 features il punteggio è di 0.77.

Un ulteriore prova è stata effettuata attraverso l'impiego dell'algoritmo XGBoost ma questa volta dando maggior rilevanza hai dati TRUE rispetto ai casi FALSE grazie al parametro 'scale_pos_weight' che in questo caso è stato impostato pari a 700. Il modello risultante è il migliore di quelli trovati finora dal punto di vista della F1 score garantendo una buona previsione dei casti TRUE.

Giusto per mostrare meglio i risultati ottenuti, vene presentata sottostante una tabella che mostra i punteggi ottenuti con i 2 approcci più promettenti utilizzano 3 metriche differenti ed una cross validation con cv=10. Vengono inoltre riportati i punteggi ottenuti con tali algoritmi senza l'aggiunta o modifica di parametri

Modello	Accuracy	Precision	F1 score	F2 score
Decision Tree (all features)	0.776	0.20	0.145	0.151
Decision Tree (3 features + weighted class)	0.755	0.17	0.216	0.209
XGBoost	0.837	0.30	0.162	0.144
XGBoost (weighted class)	0.774	0.30	0.299	0.325

Come si può notare dalla tabella, il miglior risultato trovato è stato ottenuto grazie all'algoritmo XGBoost con classi pesate. Rispetto all'algoritmo XGboost standard si è perso qualche punto percentuale per quanto riguarda l'accuracy ma ne ha giovato il punteggio F2 permettendo così al modello di riuscire a classificare meglio anche le istanze di classe TRUE che faticavano ad essere riconosciute da tutti gli altri modelli.

Come ultima prova è stata creata una pipeline che prevede l'utilizzo dell'algoritmo di clustering KMean() e di un semplice Decision Tree. In questo caso i risultati sono simili a quelli ottenuti in precedenza con l'utilizzo di sole 3 features.

Conclusioni

I risultati ottenuti sono in linea con quanto trovato su diversi papers[1, 2] e sulla pagina Kaggle del dataset.

La difficolta maggiore non è stata tanto nel cercare un modello che garantisse il miglior punteggio dal punto di vista dell'accuratezza ma nell' individuare un modello che non andasse in overfitting e che potesse classificare un numero sufficiente di TP. Il compromesso è stato raggiunto in questo caso utilizzando 2 differenti approcci. Nel primo approccio si è impiegato un Albero di decisione che utilizzasse solo 3 delle 16 features presenti nel dataset e che desse maggior peso ai dati di classe TRUE rispetto a quelli di classe FALSE. Nel secondo approccio, quello più promettente, si è impiegato l'algoritmo XGBoost con classi pesate, permettendo anche in questo caso di dare più importanza alle istanze di classe TRUE rispetto a quelle di classe FALSE.

Solo un articolo di quelli ricercati è riuscito ad ottenere un punteggio superiore: gli autori hanno utilizzato una procedura custom che prevede l'utilizzo di 3 differenti tecniche di clustering (k-means clustering, fuzzy c-means clustering e mean shift clustering) nella fase di preprocessing. Purtroppo il paiper non è sempre chiaro nei passaggi ed adottare tale tecnica è risultato arduo.