Assignment 1 (Adavenced Machine Learning) - Default Credit Card Payment

Davide Sangalli - Matricola: 848013

October 21, 2019

Abstract

In questo assignment viene affrontato il problema del Default Credit Card Payment: lo scopo è quello di predirre se un cliente della banca pagherà regolarmente (classe 0 del dataset), oppure andrà in default (classe 1 del dataset)

1 Approccio alla classificazione

1.1 Preprocessing del dataset

Innanzitutto è stata fatta feauture selection sul dataset, per capire quali fossero le features più importanti al fini della predizione di default o meno. Si sono dunque scelte le variabili che risultavano maggiormente correlate con la varaibile di target "default.payment.next.month".
Si è scelto di prendere in considerazione solo le otto variabili più significative, che sono, nell'ordine: PAY_0, PAY_1,PAY_2,PAY_3,PAY_4,PAY_5,PAY_6,LIMIT_BAL e PAY_AMT1.
Questo ranking delle features è stato ottenuto tramite una feature selection univariata effettuata con il metodo "SelectKBest". Si è optato per questo tipo di features selection, per evitare che l'importanza delle features fosse influenzata da un "bias" dovuto al particolare classificatore utilizzato per effettuarla.

Si è dunque modificato il dataset di conseguenza, ed è stato poi preparato per essere utilizzato dalla rete neurale: si sono standardizzate tutte le features e le labels sono state trasformate nella codifica one-hot.

1.2 Tuning della rete neurale

Dall'analisi della variabile di target del dataset, si è notato che c'è la presenza di una class imbalance non trascurabile (le labels "0" sono circa 3,5 volte le labels "1"). Inoltre, per una banca, predirre correttamente se un cliente andrà in default avrà un peso maggiore di predirre correttamente se un cliente pagherà. Da queste osservazioni nasce l'esigenza di utilizzare come misura di scoring

l'*f1_measure*, che oltre alla *recall* (importante per questo problema), tiene conto anche dalla *precision*.
Si è perciò cercato di trovare un set ottimale dei parametri della rete per raggiungere questo scopo. Il tuning è stato implementato tramite la *GridSearchCV*, a cui è stata passata la funzione *create_model*, in cui si cerca di ottimizzare congiuntamente il numero di hidden layers della rete e il numero di neuroni in ogni hidden layer. Si è deciso di provare le seguenti combinazioni di parametri: [8,16],[16,32],[8,16,32],[16,32,64].
Inoltre si è utilizzata una *batch_size* di 32, un numero di epoche pari a 50, una *cross_validation* pari a 3 e *f1* come funzione di scoring. Sono stati inoltre aggiunti i pesi alle classi durante l'esecuzione della *GridSearch*. Per quanto detto sopra, si sono scelti i seguenti pesi:

- *classe 0: peso 1*
- *classe 1: peso 3.56*

La *GridSearch* ha restituito come parametri ottimali [8,16], cioè due hidden layers con, rispettivamente, 8 e 16 neuroni.

1.3 Implementazione della rete neurale

Dopo aver effettuato il tuning, si è proceduto all'implementazione vera e propria della rete, utilizzando i parametri restituiti dalla *GridSearch*, tramite la funzione *build_model*.
In particolare il modello è così costituito:

- un input layer a cui sono state passate le 8 features selezionate in precedenza (quindi *input_shape=(8,)*)
- un primo hidden layer da 8 neuroni con funzione di attivazione *relu*
- un secondo hidden layer da 16 neuroni con funzione di attivazione *relu*
- un layer di output con 2 neuroni (corrispondenti al numero di classi) e con funzione di attivazione *softmax*

Inoltre, gli altri parametri passati alla funzione *compile* sono stati:

- optimizer: *adam*
- loss: *categorical_crossentropy*

Il modello è stato fittato con i dati del traininig set e contemporaneamente validato sui dati del validation set, per avere un riscontro di come la rete neurale si comportava, epoca dopo epoca.

1.4 Conclusioni

Alla fine del ciclo di training, è stato stampato il *classification_report* della rete neurale, sia sul *validation_set*, che sul *training_set*.
Le performance sul validation set sono state soddisfacenti: l'*f1_measure* raggiunge il 55% sulla classe rara (1) e l'85% sulla classe maggioritaria (0).
Un comportamento analogo vale per il training set: *f1_measure* del 54% sulla classe rara (1) e dell'85% sulla classe maggioritaria (0).
Questi dati confermano che il modello è buono e non c'è di fatto overfitting.
Il modello classifica come appartenenti alla classe 1 circa il 28% delle osservazioni del validation set e circa il 27% delle osservazioni appartenenti al training set. Mostra un comportamento analogo anche per quanto riguarda le osservazioni del test set, di cui non sono state fornite le labels (circa il 32% delle istanze viene classificato come appartenente alla classe 1), perciò ci si può ritenere complessivamente soddisfatti da questo modello.