Advanced Machine Learning

Secondo Assignment

Simone Paolo Mottadelli, 820786

Contents

1 Data exploration and data processing

Il dataset contiene informazioni relativamente a pagamenti inadempienti, fattori demografici, dati di credito, cronologia dei pagamenti e estratti conto di clienti di carte di credito a Taiwan da aprile 2005 a settembre 2005.
Il task che si vuole risolvere è un task di classificazione: si vuole predire se il cliente adempirà i pagamenti nel mese successivo.
Il dataset contiene 24 attributi esplicativi, di cui 10 categorici e i rimanenti numerici, e un attributo target binario.
Il primo step di preprocessing che ho deciso di eseguire è stato quello di rimuovere l'attributo *ID* dal dataset, in quanto non significativo per risolvere il problema. Successivamente, ho binarizzato gli attributi categorici, arrivando ad ottenere un dataset costituito da 92 attributi, considerando anche la variabile target. Infine, ho normalizzato le distribuzioni delle variabili numeriche tra 0 e 1, così da avere tutte le variabili in input comprese in tale range.

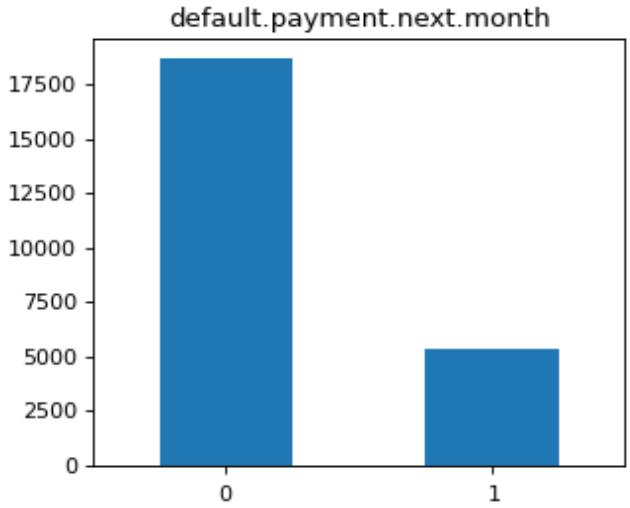

Figure 1: Class distribution

La Figura 1 mostra la distribuzione della variabile target *default.payement.next.month* e, da come si può notare, il dataset è sbilanciato. Per tale ragione, ho deciso di applicare una tecnica di oversampling per bilanciare le due classi nel training set, andando a generare nuove istanze sintetiche della classe di minoranza. In particolare, ho utilizzato una versione di *SMOTE* [1] che riuscisse a trattare anche le variabili binarie che erano state generate dal processo di binarizzazione delle variabili categoriche.

2 Classification

Per risolvere il task di classificazione ho deciso di utilizzare una Feedforward Neural Network con 5 hidden layer, ciascuno costituito rispettivamente da 200, 150, 100, 50 e 25 neuroni. Ho deciso di utilizzare questa architettura complessa con la speranza che venisse incluso il true data generating process e successivamente avrei reso la rete più semplice, cercando di evitare l'overfitting, andando ad applicare una strategia di regolarizzazione.
Per le unità nascoste ho utilizzato la funzione di attivazione $ReLU$, in quanto è una delle funzioni di attivazione più utilizzate al giorno d'oggi, mentre per l'unità di output ho deciso di utilizzare la funzione *sigmoide*, in quanto il task che dobbiamo risolvere è un problema di classificazione binario. La funzione di loss che ho deciso di utilizzare è la *binary cross entropy*.
Ho deciso di utilizzare *Adam* come algoritmo di ottimizzazione perché molto performante utilizzando una batch size di 128.
Ho inizializzato i pesi seguendo la Glorot uniform, mentre ho inizializzato i bias a zero.
Se non utilizzassimo nessuna tecnica di regolarizzazione, ciò che accadrebbe è che il modello andrebbe in overfitting dopo un certo numero di epoche, in quanto, mentre il training error continua a diminuire, il validation error inizia ad aumentare da un certo punto in poi, facendo sì che il modello abbia molta varianza e poco bias. Questo si può vedere dalla Figura 2, la quale mostra che anche la F1-score sul training set continua ad aumentare, mentre sul validation set diminuisce sempre di più.
Inoltre, analizzando i pesi del modello senza applicare nessuna tecnica di regolarizzazione, si può vedere che essi assumono valori molto più grandi rispetto ai valori dei pesi che si otterrebbero applicando delle tecniche di regolarizzazione.

Ho deciso di confrontare due tecniche di regolarizzazione: penalità sulle norme con L2 (con fattore di regolarizzazione di default pari a 0.01) e Dropout (con frazione di unità da omettere pari a 0.5 per ogni layer). Ho applicato la regolarizzazione solo sui pesi, lasciando che i bias non venissero regolarizzati. La Figura 3 mostra i valori della loss function e della F1-score al variare delle epoche per entrambe le strategie di regolarizzazione. Da come si può vedere, la normalizzazione con Dropout ha portato ad ottenere risultati leggermente migliori per quanto riguarda la loss function, ma in termini di F1-score (calcolata considerando la classe 1 come classe positiva) le due strategie sembrano essere molto simili.

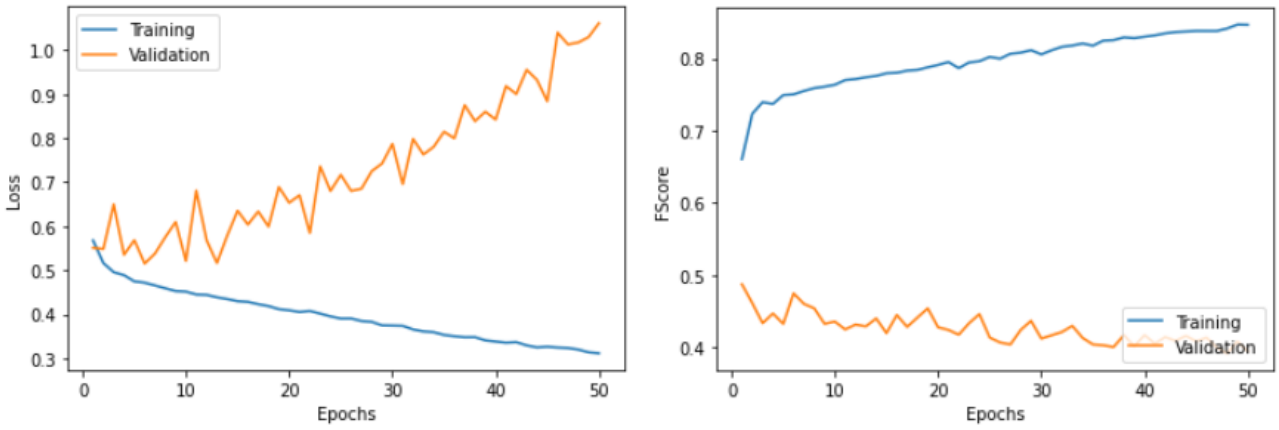

Figure 2: Results without any regularization strategy

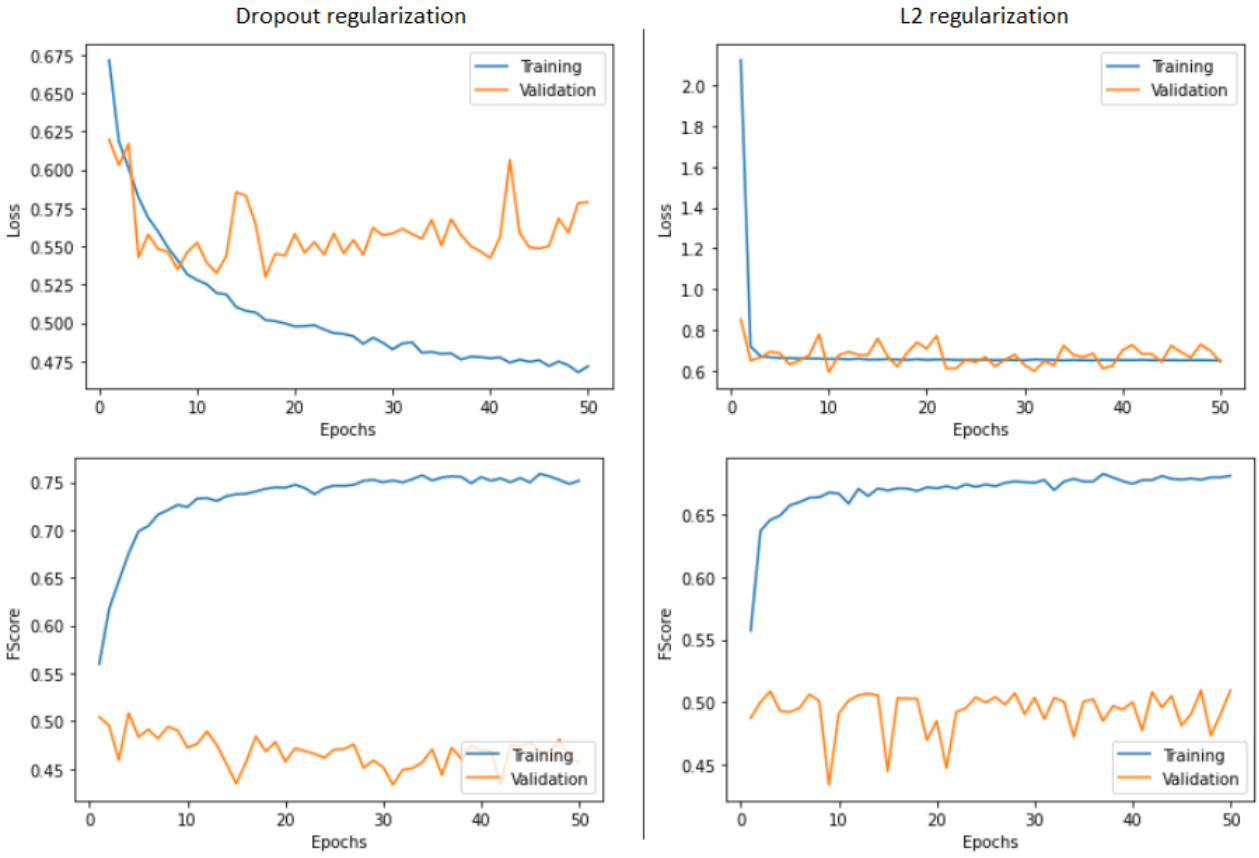

Figure 3: Results of the different regularization strategies

Inoltre, da come si può osservare dai grafici, possiamo notare che applicando la regolarizzazione i modelli sono meno proni ad andare in overfitting.
Analizzando i pesi dei modelli, ho notato che la regolarizzazione con L2 cerca di avere pesi molto più piccoli rispetto ai pesi ottenuti con Dropout.
Infine, ho utilizzato tutto il dataset per costruire una FNN con gli stessi iper-

parametri utilizzati precedentemente e usando Dropout come tecnica di regolarizzazione, fermando l'apprendimento del modello quando l'esecuzione di tre epoche consecutive non avrebbe portato ad ottenere più dei miglioramenti. Ho dunque effettuato le predizioni sul test set non etichettato.

References

[1] Smote nc. `https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.over_sampling.SMOTENC.html`. Accessed: 16.11.2020.