Marco Piazza 829588
Relazione assignment n°1 Advanced Machine Learning:

Predicting default payments with Fully-connected NNS

The dataset contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005.

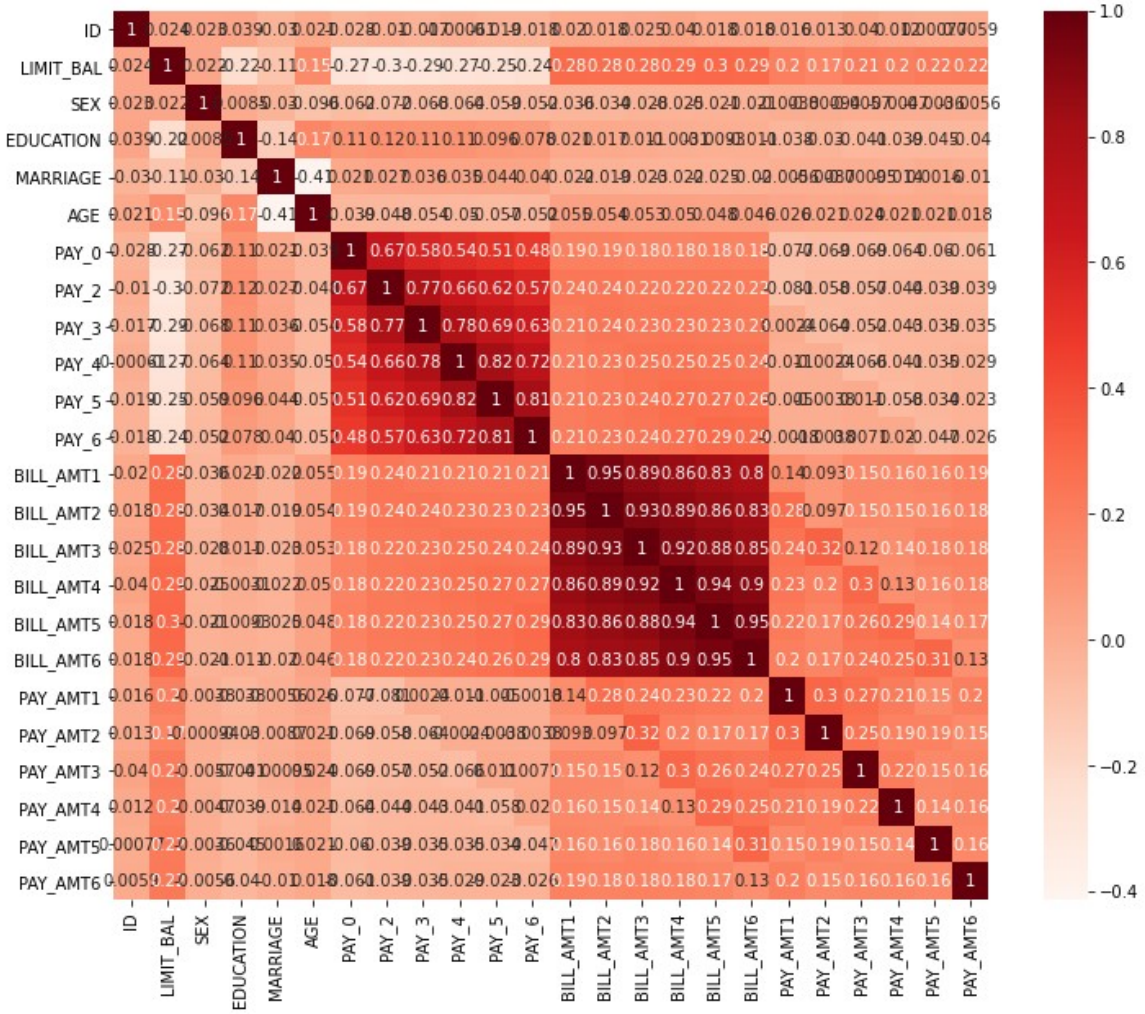
Sezione1: Inspecting the data

Il dataset presenta 24000 osservazioni con 24 feature. Analizzando i valori presenti nelle varie colonne sono state riscontrate alcune incongruenze rispetto ai valori presentati nel dizionario del dataset:

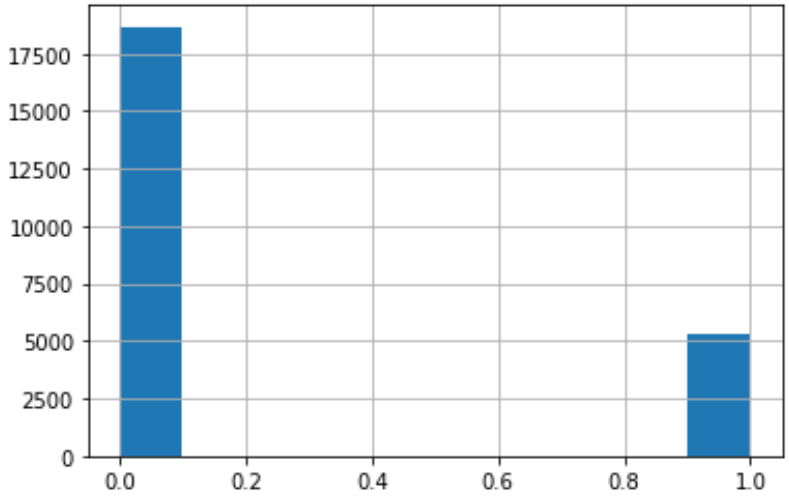
- il campo *education* contiene 10 valori 0, che sono stati accorpati nel valore 4 ("*others*")
- il campo *marriage* contiene 45 valori 0, che sono stati accorpati nel valore 3 ("*others*")

Inoltre sulla feature *education* è stato deciso di accorpare i valori 5 e 6 (di cui non si conosce il significato) con valore 4 (others). Il numero di elementi del dataset modificati risulta minimale rispetto al numero totale di elementi, per questo motivo si pensa che il modello non risulti particolarmente influenzato da queste modifiche.

Successivamente si è passati ad un analisi delle feature, in particolare l'analisi della distribuzione delle feature non fornisce particolari informazioni, in quanto quasi tutte presentano una distribuzione normale. E' stato però osservato che alcuni gruppi di feature (PAY0 – PAY6, BILL_AMT1 – BILL_AMT6 e PAY_AMT1 – PAY_AMT6) presentano una distribuzione molto simile. È stato quindi deciso di approfondire la relazione tra queste e analizzando la correlazione è risultata una forte correlazione tra le fature PAY e BILL_AMT.



L'ultima analisi ha riguardato la distribuzione della variabile target. Come si può vedere dal grafico riportato c'è uno sbilanciamento tra i due valori con circa 18000 istanze contro 5000.



Sezione2:

Preparing the data

Considerando quanto presentato sopra si è deciso di procedere secondo due approcci: il primo considerato un approccio più vicino al machine learning tradizionale, quindi è stato deciso di effettuare una feature selection su PAY e BILL_AMT; il secondo invece è stato un approccio più tipicamente deep, il dataset è stato mantenuto in forma originale, perché una rete della giusta complessità può avere la capacità di trovare tutte le relazioni tra i dati. Inoltre in questa fase sono state effettuate le operazioni tipiche e necessarie per l'utilizzo di una rete neurale:

- standardizzazione delle feature
- cast delle feature a tipo float

Inoltre sono stati provati alcuni approcci di balancing dei dati agendo sul dataset:

- oversampling: SMOTE
- undersampling: undersampling basato sui centroidi e random undersampling

Sezione 3: Building the model and training results

Sezione 3.1: The model

Seguendo la strategia di un doppio approccio sono stati quindi costruiti due modelli diversi: il modello allenato sul dataset ottenuto dalla feature selection più semplice in termini di layers e neuroni, mentre l'approccio deep più complesso. Per ottenere entrambi i modelli sono state fatte varie prove scegliendo gli hyperparametri e la forma della rete in modo da ottimizzare le performance ed evitare l'overfitting.

La rete costruita secondo il primo approccio prevede solo 2 hidden layers:

- Hidden layer 64 neuroni
- Hidden layer 32 neuroni

È stata utilizzata *relu* come funzione di attivazione per gli hidden layer, mentre *sigmoid* per l'ultimo layer (in quanto problema di classificazione binaria). Come ottimizzatore è stato scelto *sgd* con learning rate pari a 0.01. La dimensione scelta del batch è stata di 64 e il modello è stato allenato per 50 epoche. Come funzione di loss è stata scelta la *binary cross-entropy*. Si è deciso di non applicare nessuna strategia di regolarizzazione, in quanto il modello costruito è già semplice di per sé, mentre come balancing l'approccio più performante è risultato l'utilizzo di pesi diversi sulle classi (0.65 per la classe di minoranza e 0.35 per la classe di maggioranza).

La seconda rete risulta essere invece più complessa e prevede 4 hidden layers:

- Hidden layer 512 neuroni
- Hidden layer 256 neuroni (0.3 di dropout)
- Hidden layer 64 neuroni (0.2 dropout)
- Hidden layer 32 neuroni (0.1 dropout)

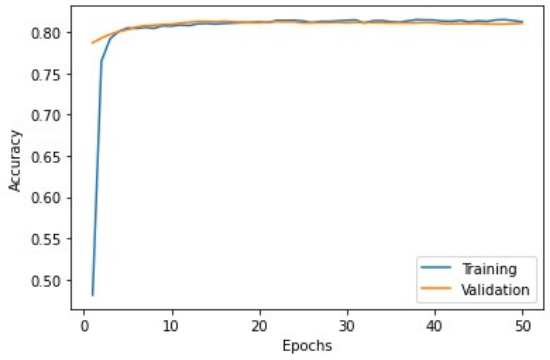
Anche in questo caso è stata scelta *relu* come funzione di attivazione per gli hidden layer e *sigmoid* per il layer di output. Come ottimizzatore è scelto *sgd* con learning rate pari a 0.01. La dimensione del batch è stata 128 e il modello è stato allenato per 50 epoche. Come funzione di loss è stata scelta la *binary cross-entropy*. E' stato scelto di applicare il dropout come tecnica di regolarizzazione applicandolo in maniera decrescente: 0.3, 0.2 e 0.1. La strategia di balancing scelta è stata la medesima del modello precedente.

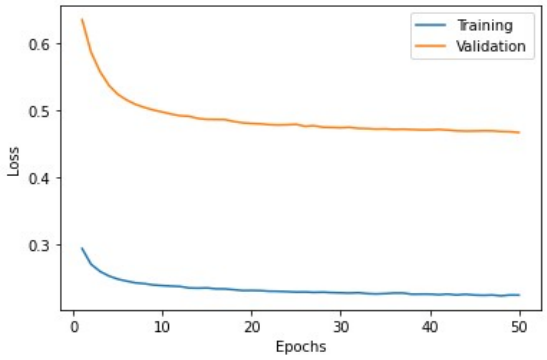
Per il secondo modello sono state provate varie strategie di regolarizzazione: L1, L2 e dropout. La scelta è ricaduta sul dropout perché è risultato quello che ha massimizzato le performance generali del modello. La regolarizzazione di tipo L1 è risultata troppo penalizzante e ha abbassato troppo le performance del modello, mentre abbassando il valore di ƛ si creava un modello troppo simile all'originale. Questo effetto è risultato anche con l'utilizzo della regolarizzazione di tipo L2, seppur in misura minore della precedente.
Sia per quanto riguarda la strategia di balancing che le strategia di regolarizzazione la scelta è stata fatta dopo aver effettuato vari tentativi e scegliendo quelle che riuscissero a massimizzare le performance, senza però aumentare l'overfitting sui dati di training e al tempo stesso non introducesse una sorta di underfitting.

	Hidden Lay.	Attivazione	Epoche	Batch size	Loss	Ottimizzatore	Regolarizzazione	Balancing
Modello 1	64 - 32	relu sigmoid	+50	64	binary cross-entropy	sgd (0.01)		pesi: 0.65 – 0.35
Modello 2	512 – 256 – 64 - 32	relu sigmoid	+50	128	binary cross-entropy	sgd(0.01)	droput: 0,3 – 0,2 – 0,1	pesi: 0.65 – 0.35

Sezione 3.2: Training Results and validation of the model
I risultati ottenuti su entrambi i modelli in fase di training hanno dimostrato come l'approccio di tipo deep sia molto più performante, questo si è dimostrato sia in fase di allenamento che in fase di predizione. Per questo motivo si è deciso di riportare solo i risultati ottenuti dal secondo modello, inoltre è stato consegnato solo il codice che si riferisce allo stesso modello.

Per quanto riguarda il training si è riusciti ad ottimizzare il modello, in quanto la curva dell'accuracy del training set quasi coincide con la curva del validation set, quindi il modello ha performance buone sia su dati già visti, che su dati mai visti. Questo è stato ottenuto solo grazie all'inserimento del dropout, perché con questo numero di epoche il modello senza regolarizzazione andava in una situazione di overfitting. L'andamento della funzione di loss risulta simile per entrambi le fasi tuttavia è più alta per il validation in maniera costante, ma avendo l'accuracy uguale non è stato trovato un motivo, forse potrebbe essere legata all'utilizzo dei pesi nel training o alla regolarizzazione tramite dropout.





La validazione del modello ha permesso di confermare come questo approccio ha permesso di migliorare la qualità della predizione del modello. Infatti l'accuracy totale è rimasta pressochè invariata, ma sono aumentati i valori di precision e recall sulle classi di minoranza, aumentando anche i valori di questi ultimi per il modello nel suo complesso e aumentando anche il valore di F1-score. Questo conferma come il modello accetti qualche errore in più sulla classe di maggioranza a patto di avere una migliore predizione su quella di minoranza, che corrisponde al comportamento desiderato.

	Precision	Recall	F1-score	Accuracy
1 (minoranza)	0,44	0,61	0,51	
0 (maggioranza)	0,92	0,85	0,88	
Totale	0,61	0,43	0,51	0,81

Come si può vedere anche dalla tabella con i risultati le prestazioni del modello sono migliorate di molto sulla classe di minoranza, ma questo non ha inficiato le quelle sulla classe di maggioranza.
L'introduzione del dropout ha permesso inoltre di limitare il fenomeno dell'overfitting, infatti trainando il modello senza quest'ultima miglioravano le performance in fase di training, peggiorando però quando si è andato a validare il modello. Inoltre veniva accentuato il problema della predizione sulla classe di minoranza, segno che magari la distribuzione delle feature nel training set non è coerente con quella reale del problema.
In conclusione è quindi possibile affermare che il modello proposto possa essere considerato migliore degli altri analizzati per qualità della predizione e capacità di generalizzazione del problema. Inoltre tra le varie strategie di balancing provate il peso assegnato alle classi è quello che ha permesso un miglioramento maggiore delle performance totali del modello (quindi senza troppa perdita sulla classe di maggioranza). Infine tra le forme di regolarizzazione proposte quella che ha fornito i risultati migliori è stata dropout. Con l'utilizzo di regolarizzazione L1/L2 è risultato che con valori di λ troppo piccoli non si riuscisse a generalizzare abbastanza il modello, mentre con valori troppo grandi il modello risultasse in underfitting.

Sezione 4: Make Predictions on the provided test set
Per il task di previsione sul test è stato scelto di utilizzare il modello deep, costruito secondo quanto spiegato sopra. I risultati attesi possono essere in linea con quelli ottenuti dal durante la fase di predizione sul validation set. Si considera possa esserci un piccolo calo trascurabile dovuto al fatto che i dati non visti possano differire da quelli visti nel training, ma si considera che il modello abbia una buona capacità di adattamento per mantenere buone performance.

Le predizioni sono state effetuate con il modello deep. Nel codice non sono state incluse tutte le prove effettuate per ottenere il modello considerato più performante e neanche il modello uno, in quanto come discusso le performance sono state lontane dall'altro modello proposto.