Anselmo Raffaele
846842
r.anselmo@campus.unimib.it

ASSIGNMENT 1
Advanced Machine Learning

Dataset introduction

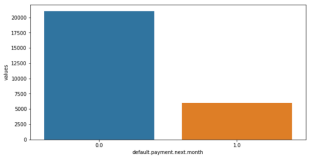
Il dataset contiene 30000 osservazioni riguardanti transazioni fraudolente, 27000 di training con le labels e 3000 di test senza labels.

Data Exploration

Da una breve analisi esplorativa non risultano presenti missing data (a meno della labels del test set), ma alcuni valori delle variabili “MARRIAGE” ,“EDUCATION” e “PAY_X” non sono documentati nelle informazioni disponibili sul dataset.

Unbalanced Class

L'analisi della variabile target ha evidenziato che 5973 clienti su 21027 risultano inadempienti (22.12%) potrebbero pertanto rivelarsi necessarie alcune operazioni preliminari sulla variabile target.

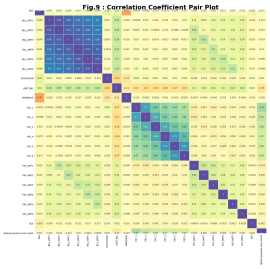


Exploring Variables relation

Correlation Coefficient Pair Plot

Si evidenzia la presenza di “blocchi di features” che condividono buona parte dell’informazione (“BILL_AMTX”, “PAY_X”, “PAY_AMTX”). Potrebbe essere utile operare delle trasformazioni nei dati per rimapparli in un nuovo spazio (e.g. PCA).

Nel Notebook sono presenti esplorazioni tramite grafici più approfonditi.



Data Cleaning

Come notato durante l'analisi preliminare alcune categorie sono etichettate in modo errato e/o non documentate. Vengono pertanto prima di svolgere l’analisi.

Building the Model

Definizione della rete neurale con **due hidden layer** da 50 neuroni (il primo) e 25 (il secondo). Sono state inoltre aggiunti due Layer di Dropout con rate = 0.5 per evitare l'Overfitting.

Per inizializzare i pesi si è scelta la funzione **TruncatedNormal** che genera una distribuzione Normale troncata e forzano la stessa varianza forward/backward tra i layers. É uguale alla *RandomNormal* ad eccezione del fatto che i valori che differiscono della media per due volte la deviazione standard sono eliminati e ridefiniti.
La scelta del set iniziale di pesi è importante perché può determinare la convergenza dell'algoritmo.

In generale si ricerca un set di pesi che riesca a **"rompere la simmetria"** tra le unità.

Inoltre i pesi iniziali assegnati alla prima rete neurale sono salvati in "initial_weights" e sono stati riassegnati a tutti i modelli successivi per migliorare il confronto fra diversi modelli.

I neuroni degli hidden layer sono caratterizzati dalla funzione di attivazione **Rectified Linear Unit (ReLU)**, che è efficiente nella back propagation degli errori e ha la caratteristica di attivare pochi neuroni in momenti diversi rendendo la **rete sparsa** e di facile computazione.

Il layer di output contiene un solo neurone, caratterizzato dalla funzione di attivazione **sigmoid**,che assicura un gradiente forte oltre ad essere molto comune nei problemi di classificazione.

Come ottimizzatore è stato usato un algoritmo con learning rate adattivo, **Adam** (Adaptive Moments), che è una variazione di RMSProp + Momentum:
Il momentum è incorporato direttamente nella stima dei momenti di primo ordine.
Nell'RMSProp il momentum è incluso dopo avere riscalato i gradienti.
Adam aggiunge anche una correzione del bias nei momenti.

La funzione di perdita da ottimizzare è rappresentata dalla **Binary CrossEntropy**. Altre funzioni di perdita come il MAE o il MAE, quando combinate con le unità di output saturano e producono gradienti bassi.

In [48]:
```
def make_model(metrics = METRICS):
 model = keras.Sequential([
     keras.layers.Dense(16, activation='relu',kernel_initializer='TruncatedNormal', input_shape=(23,)),
     keras.layers.Dropout(0.5),
     keras.layers.Dense(64, activation='relu'),
     keras.layers.Dropout(0.5),
     keras.layers.Dense(1, activation='sigmoid'),
 ])
 model.compile(
     optimizer=keras.optimizers.Adam(),
     loss=keras.losses.binary_crossentropy,
     metrics=metrics)

 return model
```

Useful functions

Sono state definite 3 funzioni, **plot_metrics**, **plot_cm**, e **plot_roc** per valutare rispettivamente le performance del modello, la Confusion Matrix e la ROC Curve.

Prima di procedere con la stima dei modelli, si provvede alla standardizzazione del dataset.

Baseline Model

Viene per primo stimato un modello semplice che tuttavia risente della classe sbilanciata. Malgrado l'accuracy sia molto alta, non è un buon indicatore delle performance in questo caso.
Ponendo l'attenzione sull'F-Measure si nota come il classificatore non è efficiente e quindi vanno effettuate delle correzioni per le classi sbilanciate.

Può essere utile concentrarsi sulla diminuizione dei Falsi Negativi, in quanto il loro costo potrebbe essere più elevato rispetto a quello dei Falsi Positivi (nel primo caso "perderemmo" delle transazioni fraudolente, nel secondo probabilmente manderemmo una mail al cliente chiedendo di attenzionare la propria situazione finanziaria)

Weighted Model

Un rimedio utile nel caso di classificazione a classi sbilanciate è rappresentato dall'imposizione di un sistema di pesi che dia più valore alla classe rara nella fase di Training (esistono anche altri metodi come Oversampling ecc..).

Malgrado la ROC non evidenzi particolari miglioramenti si può notare come i valori di precision e recall risultino più bilanciati, a conferma che l'inserimento del sistema di pesi ha aiutato il classificatore

```
loss :  0.5569459783589399
tp :  744.0
fp :  921.0
tn :  3287.0
fn :  448.0
accuracy :  0.7464815
precision :  0.44684684
recall :  0.62416106
auc :  0.77585405
```

Si procede quindi alla stima della classe 'default.payment.next.month' nel Test Set