Università degli Studi di Milano-Bicocca

Corso di Laurea Magistrale in Data Science 2020-2021

Machine Learning

Credit Card Fraud Detection

Alberto Carlone, matr. 726894

Introduzione

Il fenomeno delle frodi attraverso i pagamenti tramite carta di credito è diventato sempre più frequente nel corso degli anni, favoriti dalla digitalizzazione dei metodi di pagamento e dal passaggio a sistemi di e-commerce in luogo dei classici negozi fisici.

Nonostante ciò, si tratta comunque di una decisa minoranza rispetto al totale delle transazioni effettuate e pertanto, in aggiunta agli articolati sistemi architettati per mettere in pratica l'atto criminale in sé, risulta estremamente difficile poter comprendere in anticipo la natura di una transazione.

Da qui nasce la necessità di sviluppare sistemi predittivi per smascherare le transazioni fraudolente prima che sia troppo tardi, generando enormi problemi per il possessore della carta di credito. Per farlo, verrà effettuata una classificazione binaria tra le transazioni fraudolente e non fraudolente, applicando due diversi approcci rispetto al problema dei dataset sbilanciati.

1. Descrizione del Dataset

Il dataset [1] è composto da due giorni di transazioni effettuate da carte di credito nel settembre 2013 da parte di possessori di carte di credito europei.

Il dataset è estremamente sbilanciato in favore delle transazioni non fraudolente in quanto, su 284207 righe, solamente 492 fanno riferimento a transazioni fraudolente, ossia lo 0,17%.

Per ragioni di privacy il dataset è stato fornito direttamente con le variabili di interesse trasformate tramite PCA (Principal Component Analysis) e non è possibile ottenere le informazioni originali. Le uniche due variabili che non hanno subito la trasformazione sono state "Time" e "Amount".

2. Descrizione delle variabili

- **Time** (numerica): rappresenta l'istante in cui è avvenuta la transazione in secondi rispetto alla prima transazione registrata.
- **V1 a V28** (numerica): sono le 28 componenti della trasformazione tramite PCA delle variabili originali.
- **Amount** (numerica): rappresenta l'importo della transazione.
- **Class** (categorica nominale): rappresenta la tipologia della transazione con 1 - fraudolenta e 2 - non fraudolenta.

Pre-Processing

Essendo il dataset già reso disponibile con le variabili trasformate tramite PCA non è stata necessaria una fase di pre-processing particolarmente rilevante. Nell'ipotetica situazione in cui il dataset fosse disponibile con gli attributi originali, avremmo ad esempio potuto effettuare preliminarmente una attribute selection per individuare le variabili di interesse e rendere più veloce l'allenamento dell'algoritmo. È altamente probabile che, rispetto al dataset originale, le variabili derivanti dalla PCA siano inferiori. Si evidenzia infine che l'intero dataset è privo di valori mancanti.

1. Outlier e Valori Anomali

La presenza di outlier può rendere problematico l'allenamento di un e, prima di decidere se rimuoverli o meno, sono stati verificati quanti e per quali classi sono presenti maggiormente.

Essendo il dataset oggetto dello studio basato su un problema di rilevamento di anomalie è plausibile aspettarsi che la maggior parte delle transazioni fraudolente presenti outlier in una delle variabili di input. Ricordando che per definizione un valore è considerato un outlier leggero se è maggiore di Q3+1.5 IQR (o minore di Q1-1.5 IQR) e un outlier forte se è maggiore di

Q3+3 IQR (o minore di Q1-3 IQR) otteniamo che il dataset viene ridotto in questa misura:

Tabella 1

	Class	
	0	1
1.5 IQR	156.136	15
3 IQR	241.413	45

Considerando che eliminando gli outliers oltre 3 IQR il dataset risulterebbe ancora più sbilanciato (le transazioni fraudolente risulterebbero essere solamente lo 0,019%), aumentando la difficoltà di previsione della classe positiva, è stato deciso di mantenere il dato nella sua interezza.

2. Normalizzazione

Come ultimo step prima del training dei modelli selezionati, le variabili (con l'esclusione di *Time*) sono state oggetto di una standardizzazione: le componenti della PCA (le variabili da V1 a V28) avevano già media pari a zero ma una deviazione standard diversa da 1.

$$Z = \frac{X - \mu}{\sigma}$$

In generale, la normalizzazione (o standardizzazione) delle variabili ne riduce l'interpretabilità ma essendo già state oggetto di una PCA questa eventualità non è presente.

Modelli e Misura delle Prestazioni

Per poter prevedere se la transazione è fraudolenta o meno sono stati applicati due diversi approcci, particolarmente validi nel caso di dataset sbilanciati. Nel primo caso è stato mantenuto il dataset sbilanciato considerato una Cost Sensitive Classification, ossia un modello che cerca di minimizzare il "costo" dovuto ad una errata previsione di una delle due classi, penalizzando maggiormente la classe positiva (la classe 1 – fraudolenta); nel secondo caso è stato effettuato invece un oversampling della classe positiva tramite l'algoritmo SMOTE [2].

In entrambi i casi il dataset iniziale è stato diviso in due partizioni tramite uno stratified sampling applicato sulla variabile *Class*: il 67% è stato utilizzato per il training/testing e il restante per la validazione del modello.

Sono stati valutati quindi tre diversi modelli, oggetto inoltre di una ottimizzazione dei parametri (indicati tra parentesi):

- Random Forest (ottimizzando il numero di alberi decisionali)
- J48 (un Decision Tree sviluppato da WEKA, ottimizzando *pruning* e il numero di istanze per foglia)
- Logistic Regression (ottimizzando *ridge*)

Il train/testing dei classificatori è stato effettuato tramite una 5-Fold Cross Validation con stratified sampling della variabile Class.

1. Cost Sensitive Classification

Per il primo approccio è stata utilizzata una Cost Sensitive Classification tramite un nodo KNIME che permette, tra i suoi parametri, di specificare sia il modello da applicare (tra i tre indicati in precedenza) sia la matrice di costo da applicare. Il classificatore è stato quindi applicato tramite una 5-fold Cross Validation minimizzando il costo finale della previsione con la seguente matrice:

Tabella 2

Class	Pred(0)	Pred(1)
0	TN (cost=0)	FP (cost=10)
1	FN (cost=100)	TP (cost=-1)

All'intero processo di Cross Validation è stata quindi applicata un'ottimizzazione dei parametri tramite hillclimbing: una volta ottenuti i valori ottimizzati, sono stati applicati nuovamente al modello per effettuare la validazione sui dati partizionati in precedenza.

2. SMOTE

Per il secondo approccio è stato utilizzato, in fase di allenamento del modello, l'algoritmo SMOTE che permette di rendere il dataset nuovamente bilanciato facendo un oversampling della classe minoritaria. Non lavorando di fatto su una partizione del dataset originale ma su una con dei dati "artificiali" è assolutamente importante che l'algoritmo SMOTE venga applicato esclusivamente nella fase di allenamento in modo che, nella fase di validazione, siano presenti solamente i dati reali.

Per l'allenamento del modello è stato quindi applicato SMOTE durante la 5-fold Cross Validation e, inoltre, l'algoritmo stesso è stato oggetto di una ottimizzazione del parametro relativo ai k-Nearest Neighbor. Una volta ottenuto il numero di kNN ottimale e i parametri ottimizzati relativi al modello utilizzato abbiamo applicato nuovamente questi ultimi al modello, analogamente a quanto fatto per la metodologia precedente.

3. Misure di Performance

Recall

Essendo un dataset estremamente sbilanciato non è possibile valutare in modo corretto le performance dei modelli attraverso la metrica *Accuracy* in quanto, per definizione è strutturata in questo modo:

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP}$$

In base ai dati in nostro possesso potremmo, in linea teorica, avere una *Accuracy* pari al 99,83% prevedendo correttamente solo le classi negative (non fraudolente) e sbagliando tutte le positive.

Una metrica migliore è data dalla *Recall* (detta anche *TPR - True Positive Rate* o *sensitivity*) che viene definita così:

$$Recall = \frac{TP}{TP + FN}$$

Un valore elevato di *Recall* implica che poche classi siano state erroneamente classificate come negative.

Costo

Come già introdotto precedentemente, attraverso una matrice di costo è possibile misurare quanto pesano gli errori di classificazione.

$$Costo = TPcost * TP + TNcost * TN + FPcost * FP + FNcost * FN$$

Un modello con un costo inferiore è da considerarsi più performante.

ROC Curve

La Receiver Operating Characteristic Curve (ROC Curve) è un metodo per visualizzare graficamente le performance di un classificatore indipendentemente dalla distribuzione bilanciata (o sbilanciata) delle classi. Presenta sull'asse delle ascisse il *False Positive Rate* (FPR) mentre sull'asse delle ordinate presenta i *True Positive Rate* (TPR – detta anche Recall). Attraverso la ROC Curve è possibile confrontare il valore dell'area sottesa (*Area Under Curve* – da cui deriva l'acronimo ROC AUC) per stabilire quanto un modello sia più performante di un altro in fase di validazione.

Analisi e Risultati

1. Cost Sensitive Classification

Considerando l'approccio in cui è stato minimizzato il costo della classificazione abbiamo ottenuto i seguenti risultati

Tabella 3

	Recall	Precision	F-Measure
RandomForest	0,846	0,281	0,422
J48	0,698	0,785	0,739
Logistic	0,815	0,41	0,545

Osserviamo che il modello con il valore di Recall più elevato è Random Forest, seguito dalla Logistic Regression e dal Decision Tree J48. Rispetto alla

Logistic Regression, Random Forest ha un valore di Precision molto più basso: ciò significa che, sebbene in maniera ottimale abbia dato più importanza alla classificazione delle classi positive (fraudolente), non sia stato in grado di classificare con altrettanta precisione le classi negative.

Nonostante ciò, non è sorprendente constatare che, considerando la metrica relativa al costo totale della classificazione, Random Forest abbia un costo inferiore alla Logistic Regression

Tabella 4

	Costo
Random Forest	2.713
J48	4.818
Logistic	3.058

Questo è dovuto al fatto che un FN (*Falso Negativo*) in termini di cost matrix è considerato un errore dieci volte più gravoso rispetto ad un FP (*Falso Positivo*).

Figura 1 - ROC Curve per Cost Sensitive Classification

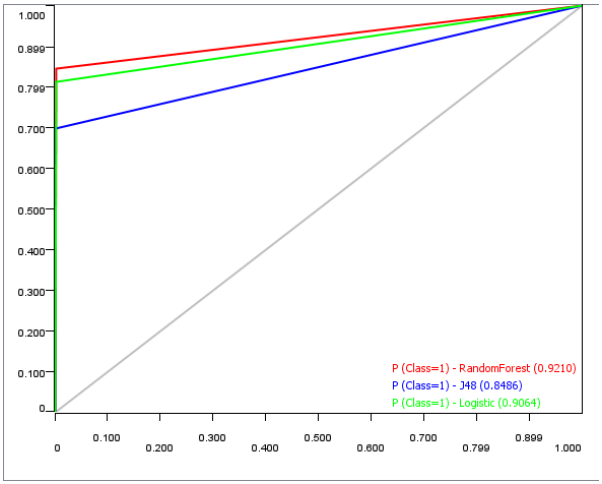

Tabella 5

	AUC
Random Forest	0,9210
J48	0,8486
Logistic	0,9064

Infine, osserviamo attraverso la ROC Curve che il classificatore Random Forest ha una AUC maggiore rispetto agli altri due modelli.

Considerando quindi le tre metriche scelte, il modello con le performance migliore per quanto riguarda la Cost Sensitive Classification risulta essere Random Forest.

2. SMOTE

Analogamente all'analisi precedente osserviamo che il modello con il valore di Recall migliore, ossia Logistic Regression, presenta un valore di Precision decisamente più basso rispetto a Random Forest.

Tabella 6

	Recall	Precision	F-Measure
RandomForest	0,796	0,872	0,832
J48	0,772	0,453	0,571
Logistic	0,877	0,146	0,251

Per lo stesso motivo detto in precedenza, valutando il costo della classificazione, la Logistic Regression ha il valore più basso tra i modelli presi in esame.

Tabella 7

	Costo
Random Forest	3.190
J48	3.726
Logistic	2.687

È importante sottolineare che, a differenza del primo approccio, non è stato utilizzato un modello in cui veniva richiesta la minimizzazione del costo: nonostante questo, il modello ha ottenuto un valore inferiore anche alla Cost Sensitive Classification effettuata con Random Forest, risultata la migliore delle tre.

Figura 2 - ROC Curve per SMOTE

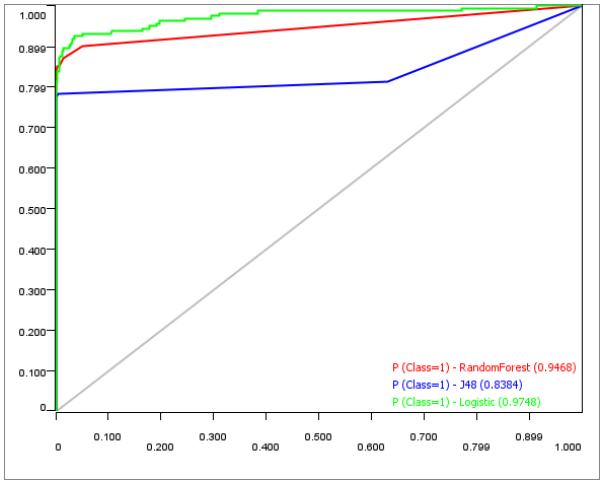

Tabella 8

	AUC
Random Forest	0,9468
J48	0,8384
Logistic	0,9748

Valutando infine la ROC Curve, viene confermato che con l'approccio dell'oversampling SMOTE la Logistic Regression ha una performance migliore avendo un valore della ROC AUC maggiore.

Conclusioni

Considerando i due approcci relativi alla classificazione binaria di un dataset sbilanciato osserviamo che:

- Random Forest risulta essere il modello ottimale utilizzando una Cost Sensitive Classification;
- Logistic Regression risulta essere ottimale rispetto ad un oversampling SMOTE della classe positiva;
- Il Decision Tree J48 non è risultato sufficientemente performante con entrambi gli approcci.

È importante evidenziare però che i classificatori Random Forest e Logistic Regression, in riferimento ai loro valori di Recall e della ROC AUC, non hanno ottenuto valori estremamente differenti: questo può significare che, ad esempio, ottimizzandone maggiormente i parametri avremmo potuto ottenere risultati ancor più simili. Per ottenere risultati migliori una strada alternativa potrebbe essere quella di una rete neurale o altri metodi in grado di pesare in modo più efficiente il costo di una errata classificazione.

Se dovessimo scegliere il classificatore migliore con un'ottica più legata al business, nel caso particolare delle transazioni fraudolente su carta di credito, sarebbe sempre meglio puntare ad un valore minimo di Recall: costa molto più all'azienda (e al cliente che subisce la frode) non riuscire a riconoscere in anticipo una transazione fraudolenta rispetto al segnalare al cliente un falso positivo.

Bibliografia e Sitografia

[1] https://www.kaggle.com/mlg-ulb/creditcardfraud

[2]SMOTE https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume16/chawla02a-html/chawla2002.html