Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea in informatica

Rilevazione di transazioni fraudolente nell'e-commerce basata sull'apprendimento automatico

Relatore:
Chiar.mo Prof.
Roberto Amadini

Presentata da:
Luigi Manieri

Correlatore:
Fernando Falcone

Sessione IIIa
Anno Accademico 2021-2022

Indice

Elenco delle figure

Capitolo 1

Introduzione

Il mondo del commercio elettronico è in costante crescita e ha portato molti vantaggi per i consumatori, tra cui la comodità dello shopping online da qualsiasi luogo e in qualsiasi momento. Tuttavia, questo mercato in rapida espansione ha anche attirato l'attenzione di truffatori che cercano di sfruttare sistemi e-commerce vulnerabili attraverso ordini fraudolenti. Questi ordini possono avere gravi conseguenze economiche per le aziende, comprese le perdite finanziarie dirette e la diminuzione della fiducia dei clienti.

Per affrontare questo problema, le applicazioni di e-commerce hanno adottato diversi metodi per identificare e prevenire gli ordini fraudolenti, tra cui l'analisi manuale dei dati e l'utilizzo di sistemi di verifica automatizzati. Tuttavia, questi metodi hanno i loro limiti e possono essere migliorati attraverso l'utilizzo di tecniche di machine learning.

Uno degli approcci più comuni per il rilevamento delle frodi è la classificazione, in cui gli algoritmi di apprendimento automatico sono addestrati su un insieme di dati etichettati. In questo caso la classificazione è binaria, in quanto le transazioni possono essere fraudolente (label 1) e non fraudolente (label 0), quindi in un problema di questo tipo l'algoritmo prende in input dei dati di cui già conosce l'etichetta, e cerca di dedurre in base alle features la probabilità che una nuova transazione sia fraudolenta o no, basandosi sulle etichette dei dati su cui è stato addestrato. Un'altra tecnica comune è

il clustering, che utilizza algoritmi di apprendimento non supervisionato per raggruppare le transazioni in gruppi simili senza sapere a quale classe appartengono (non conosce le etichette) e quindi identificare quelli che potrebbero essere fraudolenti.

In questa tesi esplorerò il problema degli ordini fraudolenti nell'e-commerce, gli impatti economici che hanno sulle aziende e i metodi attuali utilizzati per scovarli. Analizzerò anche i modelli di machine learning e come questi possono essere utilizzati per identificare gli ordini fraudolenti con maggiore precisione e come valutarli in questo contesto. L'obiettivo è fornire una comprensione più profonda del problema degli ordini fraudolenti nell'e-commerce e analizzare metodi per affrontarlo attraverso l'utilizzo di tecnologie avanzate.

Per fare ciò, utilizzerò un modello di machine learning per fare delle predizioni su un dataset di transazioni open source, per capire quali sono i fattori chiave per ottenere buone prestazioni, con l'obiettivo di massimizzare i guadagni e ridurre al minimo i costi. Dopo aver analizzato diversi modelli, verranno valutate le performance del miglior modello trovato su un dataset di transazioni di un'azienda di fashion cliente dell'azienda per cui lavoro, con lo scopo di realizzare un prodotto appetibile e vendibile da proporre a questo cliente e in caso di successo anche ad altri.

Capitolo 2

Preliminari

In questo capitolo si parlerà di alcune conoscenze preliminari per comprendere al meglio il problema della fraud detection, perché è importante analizzarlo sia per le aziende che per gli utenti finali, e infine le soluzioni proposte e dei metodi per definirne la qualità.

2.1 Le transazioni fraudolente

Le transazioni fraudolente nell'e-commerce possono rappresentare una grande sfida per le aziende che operano online, poiché possono causare perdite finanziarie, danni alla reputazione e una riduzione della fiducia dei clienti.

Le transazioni fraudolente sono transazioni illegali effettuate da qualcun altro per conto dell'utente, impossessandosi dei dati della carta di credito o del conto per un tornaconto personale [1].

Kumar Sadineni nel suo studio[1] presenta alcuni dei modi in cui si possono effettuare delle transazioni fraudolente:

- Skimming - Questa tecnica viene utilizzata per recuperare i dettagli sulla banda magnetica della carta di credito attraverso un dispositivo elettronico.
- Phishing - Questa tecnica viene utilizzata per ottenere i dati della carta di credito attraverso trappole in cui vengono inviate delle e-mail come

se fossero da parte di funzionari della banca e in cui vengono richiesti i dati privati della carta.

- Frode "Card Not Present" - Questa tecnica utilizza il numero di carta e la data di scadenza per effettuare transazioni senza utilizzare esplicitamente la carta.
- Acquisizione dell'account - Si verifica quando un'informazione riservata viene condivisa con un estraneo e questi la usa per ottenere vantaggi personali.
- Carte catturate durante la spedizione - Questo accade quando una nuova carta di credito viene intercettata durante la spedizione.
- Perdita della carta - Quando una carta viene smarrita e cade nelle mani sbagliate, c'è la possibilità che le informazioni confidenziali vengano rubate.
- Siti web falsi - che catturano l'attenzione dei clienti dando loro l'impressione che il sito web sia autentico e li spingono ad acquistare prodotti online inviando i dati della carta di credito.

Figura 2.1: Schema dei possibili tipi di frode identificati da Kumar Sadineni nel suo studio.

Per prevenire questo tipo di frode, le aziende possono adottare misure di sicurezza appropriate, come la crittografia SSL per proteggere le informazioni sensibili dei clienti durante la trasmissione e tecnologie anti-frode avanzate per identificare e prevenire le transazioni potenzialmente fraudolente. Queste tecnologie possono includere l'utilizzo di sistemi di apprendimento automatico per identificare dei pattern tra le transazioni e individuare quelle anomale e potenzialmente sospette.

È anche importante che le aziende monitorino costantemente le loro attività di e-commerce per identificare eventuali anomalie o transazioni sospette e prendere misure appropriate per prevenire ulteriori perdite.

Inoltre, le aziende possono adottare politiche rigorose per la gestione dei reclami e dei rimborsi in caso di frode, al fine di garantire che i clienti siano trattati equamente e che i loro interessi vengano tutelati.

Infine, è fondamentale che le aziende informino i clienti sulle loro misure di sicurezza e su come possono proteggere le loro informazioni personali e finanziarie durante gli acquisti online. Questo può aiutare a rafforzare la fiducia dei clienti nel sito web e nella marca, oltre a ridurre il rischio di frodi future.

2.2 Impatto economico

Le transazioni fraudolente hanno un impatto significativo sull'economia dell'e-commerce. Questo tipo di frode può causare perdite finanziarie dirette per le aziende che operano online, sia a causa delle transazioni effettuate illegalmente con le informazioni dei clienti che a causa dei costi associati alla gestione dei reclami e dei rimborsi.

Le frodi e-commerce possono anche avere un impatto negativo sulla reputazione delle aziende e sulla fiducia dei clienti. I clienti che hanno subito una frode o che temono di diventare vittime di una frode possono essere meno propensi a effettuare acquisti online, il che può influire negativamente sulle vendite e sulla crescita delle aziende.

Inoltre, le frodi e-commerce possono aumentare i costi per le banche e le società emittenti di carte di credito, che devono gestire i reclami e i rimborsi e prendere misure per prevenire ulteriori perdite.

Infine, le transazioni fraudolente possono anche avere un impatto negativo sulle agenzie governative e sulla società in generale, poiché queste frodi possono richiedere risorse pubbliche per la loro indagine e perseguimento.

In sintesi, le transazioni fraudolente nell'e-commerce hanno un impatto negativo su diverse parti interessate, dalle aziende alle banche e alle agenzie governative, e possono influire negativamente sulla crescita economica e sulla fiducia dei consumatori nell'acquisto di beni e servizi online.

Di seguito è riportata una figura che rappresenta in modo chiaro l'impatto economico delle frodi sia per gli utenti finali sia per le aziende, evidenziando il numero di vittime annuali e la quantità di denaro che è stata persa dalle aziende a causa delle frodi.

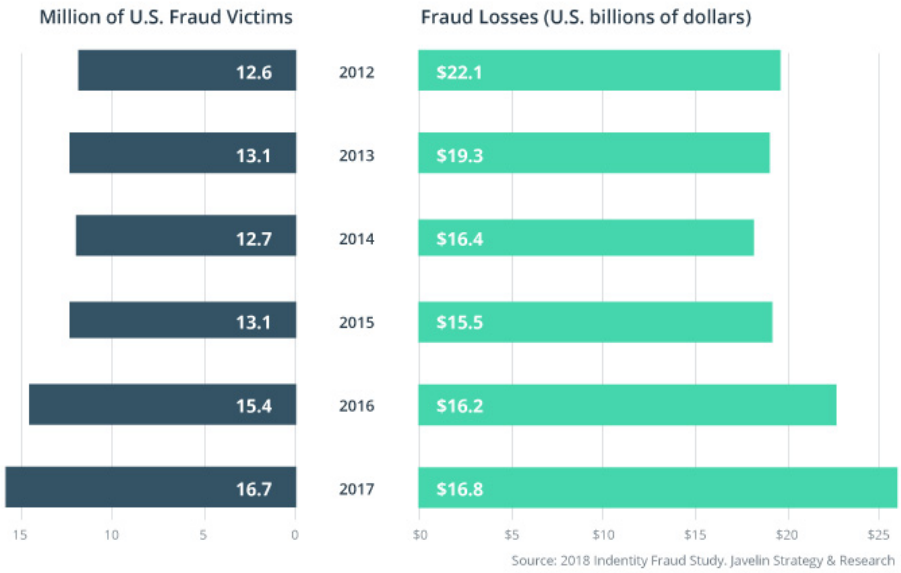

Figura 2.2: Grafico che rappresenta anno per anno il numero di persone coinvolte in frodi elettroniche e la quantità di denaro perso dai soggetti coinvolti nelle frodi negli USA

Per questo è importante per le aziende che operano nel settore e-commerce intraprendere azioni per contrastare la continua evoluzione delle frodi elettroniche, quindi lo sviluppo di sistemi di fraud detection diventa fondamentale per restare competitivi nel business.

Le aziende che operano nell'e-commerce sono da anni attente a questo tipo di problema, infatti l'investimento del settore finanziario in sistemi di sicurezza avanzati per proteggere i clienti ha impedito oltre 1,8 miliardi di sterline di frodi non autorizzate nel 2019, tuttavia i criminali hanno comunque rubato con successo oltre 1,2 miliardi di sterline attraverso frodi e truffe

[2], il che ci suggerisce che la ricerca sta andando avanti così come le tecniche utilizzate dai criminali, ragion per cui è di fondamentale importanza continuare a migliorare i sistemi di prevenzione.

2.3 L'apprendimento automatico nella rilevazione delle frodi

La fraud detection, o la rilevazione delle frodi, è una disciplina che mira a identificare comportamenti anomali e transazioni sospette all'interno di grandi quantità di dati. In passato, la fraud detection si basava principalmente sull'utilizzo di regole predefinite e modelli statistici per individuare comportamenti sospetti. Tuttavia, con l'aumento della quantità di dati disponibili e l'evoluzione delle tecnologie, i metodi di fraud detection stanno diventando sempre più sofisticati.

Negli anni precedenti sono state utilizzate tecniche di verifica manuale delle frodi, come il campionamento di scoperta, cioé la raccolta di campioni casuali senza alcuna idea preconcetta, su cui vengono svolte delle indagini manuali. Successivamente la fraud detection è stata computerizzata e automatizzata, con metodi di data mining che coinvolgono tecniche statistiche, matematiche, intelligenza artificiale e tecniche di apprendimento automatico per estrarre e identificare informazioni utili e conseguente conoscenza da grandi banche dati [3].

Una delle principali tendenze nella fraud detection è l'utilizzo di tecniche di apprendimento automatico. I modelli di apprendimento automatico possono analizzare grandi quantità di dati e identificare pattern e relazioni non evidenti, che potrebbero essere utilizzati per individuare comportamenti fraudolenti. Inoltre, i modelli di apprendimento automatico possono essere addestrati continuamente sui nuovi dati per migliorare la loro capacità di rilevare le frodi.

Un possibile algoritmo di rilevamento delle frodi sulle carte di credito consiste nell'identificare le transazioni con un'alta probabilità di essere una

frode, sulla base di modelli storici di frode, secondo lo schema illustrato nella figura 2.3. In generale, il principio dell'apprendimento dovrebbe basarsi sul comportamento dell'utente specifico e sui modelli di spesa dei truffatori e non sui vettori della frode, cioé i metodi con cui la frode viene attuata (elencati nella sezione 2.1) [4].

Per questo problema sono stati utilizzati con successo diversi sistemi di rilevamento basati su tecniche di machine learning, in particolare: reti neurali, apprendimento bayesiano, regole di associazione, modelli ibridi, support vector machine, peer group analysis, random forest, discriminant analysis, analisi delle reti sociali [5].

Figura 2.3: Schema del funzionamento di un sistema di fraud detection basato su algoritmi di apprendimento automatico

Gli algoritmi di apprendimento supervisionato vengono addestrati sui dati fraudolenti e non fraudolenti, al fine di identificare i pattern e le relazioni che caratterizzano le frodi, secondo lo schema della figura 2.4. Nell'apprendimento supervisionato, il modello deve quindi conoscere le etichette dei dati su cui viene addestrato, per poter identificare dei pattern che contraddistinguono una classe piuttosto che un'altra. Una volta addestrati, questi modelli possono essere utilizzati per analizzare nuovi dati e cercare di identificare i comportamenti sospetti.

Figura 2.4: Schema del funzionamento di un algoritmo di apprendimento automatico supervisionato

Gli algoritmi di apprendimento non supervisionato vengono utilizzati per individuare pattern e relazioni nascosti all'interno dei dati, senza la necessità di avere dati già etichettati come "frode" o "non frode", quindi si possono utilizzare quando non si conoscono a priori le etichette delle transazioni. Questi algoritmi possono essere utilizzati per identificare cluster di transazioni sospette o per generare nuove feature da utilizzare nei modelli di apprendimento supervisionato.

In realtà con il clustering, più che rilevare cluster di transazioni fraudolente, la ricerca si è concentrata sulla *Outlier Detection*, cioè nel rilevamento di quelle che sono le anomalie nei dati delle transazioni, che si manifestano solo in quelle fraudolente. In questo modo si sono costruiti modelli che rilevano gli outliers nelle transazioni e la bontà del modello si può valutare andando a verificare che gli outliers individuati coincidano con le frodi [6].

Nella figura 2.5 si può osservare il possibile output di un algoritmo non supervisionato.

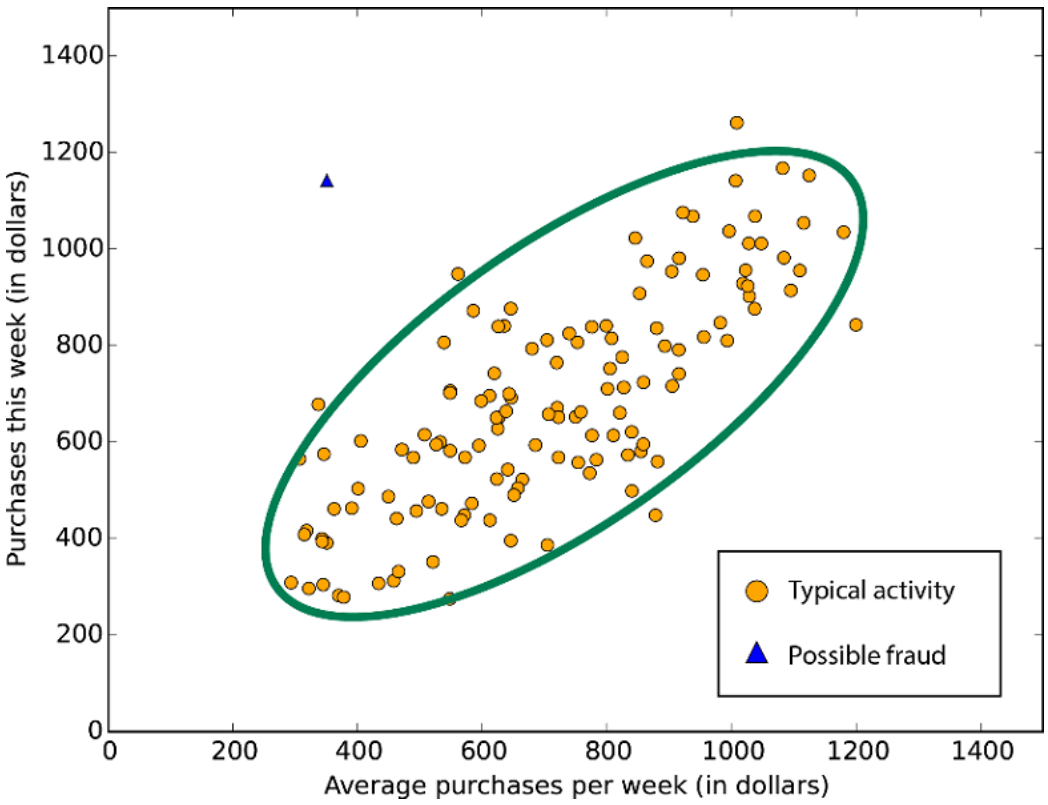

Figura 2.5: Schema del possibile output di un algoritmo di apprendimento automatico non supervisionato

Infine, gli algoritmi di reinforcement learning vengono utilizzati per addestrare i modelli attraverso la ricezione di feedback sull'azione intrapresa. Ad esempio, un modello potrebbe essere addestrato per prendere determinate azioni (come la segnalazione di una transazione sospetta) in base ai dati di input, e quindi ricevere feedback sull'azione intrapresa, come ad esempio se la transazione segnalata è stata effettivamente una frode o meno. In base a questo feedback, il modello può adattare la sua strategia per migliorare la sua capacità di rilevare le frodi.

Negli ultimi anni si è passati poi a un utilizzo combinato di queste diverse tecniche che ha prodotto notevoli passi in avanti [7]. Inoltre, si sta assistendo a una maggiore collaborazione tra le aziende e le agenzie governative per con-

dividere informazioni e dati sui casi di frode, al fine di migliorare la capacità di prevenire e rilevare le frodi. Ciò sta portando a una maggiore efficacia nella lotta contro la frode e a una maggiore protezione per i consumatori.

In futuro, si prevede che con queste tecnologie innovative e la crescente quantità di dati generati dai dispositivi connessi ci saranno ulteriori sviluppi nel campo della rilevazione delle frodi.

2.4 Valutazione del modello

Bisogna inoltre definire i criteri di valutazione per misurare le performance di un modello di fraud detection, ponendo l'attenzione sui costi che possono derivare da eventuali classificazioni errate: quanto costa classificare erroneamente come fraudolenta una transazione non fraudolenta? E quanto il viceversa?

Di seguito parleremo delle metriche utilizzate per valutare le performance di un modello di fraud detection. É utile innanzitutto introdurre il concetto di matrice di confusione.

Matrice di confusione

La matrice di confusione può essere utile per visualizzare alcune tra le metriche più comuni per quanto riguarda la valutazione di un modello di machine learning, tra cui quelle che andremo ad analizzare di seguito. In particolare, la matrice di confusione confronta le previsioni del modello con le classi effettive degli esempi nel set di dati di test e riporta il numero di volte in cui il modello ha previsto correttamente o erroneamente ciascuna classe.

La matrice di confusione è organizzata in una tabella 2x2, dove le righe rappresentano le classi reali e le colonne rappresentano le previsioni del modello. In particolare, la matrice di confusione riporta i seguenti valori:

- True Positives (TP): il numero di esempi classificati correttamente come positivi dal modello.

- False Positives (FP): il numero di esempi classificati erroneamente come positivi dal modello.
- True Negatives (TN): il numero di esempi classificati correttamente come negativi dal modello.
- False Negatives (FN): il numero di esempi classificati erroneamente come negativi dal modello.

TP	FP
FN	TN

Ad esempio, se in un set di 100 ordini, il modello classifica correttamente 80 ordini come non fraudolenti e 10 ordini come fraudolenti, ma erroneamente classifica 5 ordini non fraudolenti come fraudolenti e 5 ordini fraudolenti come non fraudolenti, la matrice di confusione sarebbe:

	Predetti non fraudolenti	Predetti fraudolenti
Ordini non fraudolenti	80	5
Ordini fraudolenti	5	10

La matrice di confusione può essere utilizzata per calcolare la precisione, la recall e l'accuratezza del modello, nonché altre metriche come la F1-score, che combinano la precisione e la recall.

2.4.1 Precisione

La precisione è una metrica comunemente utilizzata per valutare le prestazioni di un modello di machine learning. Essa misura la frazione di previsioni positive fatte dal modello che sono corrette, ovvero il numero di veri positivi (TP) diviso per la somma di veri positivi e falsi positivi:

$$Precisione = \frac{TP}{TP + FP} \tag{2.1}$$

La precisione è una metrica importante quando è importante minimizzare i falsi positivi, ovvero quando una previsione positiva errata potrebbe avere conseguenze negative.

È importante notare che la precisione da sola potrebbe non essere sufficiente per valutare le prestazioni di un modello in modo accurato, specialmente quando si hanno classi molto sbilanciate, come nel nostro caso.

Ad esempio, se si ha un set di dati con una classe positiva che rappresenta solo il 10% del totale, un modello che prevede sempre la classe negativa avrà una precisione del 90%, ma sarà in realtà molto inefficiente. In generale, la precisione è una metrica utile per valutare le prestazioni di un modello di machine learning, ma deve essere considerata insieme ad altre metriche per avere una valutazione completa delle sue prestazioni.

2.4.2 Recall

La recall, nota anche come sensibilità o tasso di positivi veri, è un'altra metrica importante utilizzata per valutare le prestazioni di un modello di machine learning, specialmente quando si hanno classi sbilanciate. La recall misura la percentuale di istanze positive che sono state correttamente identificate dal modello.

La recall è calcolata come il rapporto tra il numero di istanze positive correttamente identificate (TP) e il numero totale di istanze positive presenti nel set di dati (TP + FN):

$$Recall = \frac{TP}{TP + FN} \tag{2.2}$$

Ad esempio, se un modello di classificazione binaria ha correttamente identificato 80 istanze positive su un totale di 100 istanze positive, la sua recall sarebbe del 80%.

La recall è particolarmente importante quando si hanno classi sbilanciate, ovvero quando una classe è molto più comune dell'altra. In questo caso, la precisione potrebbe non fornire un'indicazione accurata delle prestazioni del modello. Ad esempio, se si ha un set di dati in cui la classe positiva

rappresenta solo il 10% del totale, un modello che prevede sempre la classe negativa avrà una precisione del 90%, ma una recall del 0%. Ciò significa che il modello non ha identificato correttamente alcuna istanza della classe positiva.

2.4.3 ROC

La ROC (Receiver Operating Characteristic) è un'altra metrica utilizzata per valutare le prestazioni di un modello di classificazione binaria. La ROC si basa sulla curva ROC, che rappresenta la relazione tra il tasso di falsi positivi (FPR) e il tasso di positivi veri (TPR) del modello al variare della soglia di decisione, cioé la soglia di probabilità oltre il quale un'istanza viene classificata come positiva o negativa.

Da notare che il TPR coincide con la sensibilità, quindi rappresenta la proporzione di istanze positive correttamente identificate dal modello rispetto al totale delle istanze positive, mentre il FPR rappresenta la proporzione di istanze negative erroneamente identificate dal modello rispetto al totale delle istanze negative.

La curva ROC è costruita tracciando il TPR sulle ordinate e il FPR sulle ascisse al variare della soglia di decisione (threshold). Un modello che effettua previsioni casuali avrebbe una curva ROC che segue la diagonale del grafico, mentre un modello perfetto avrebbe una curva ROC che passa per il punto (0, 1), ovvero un TPR del 100% e un FPR del 0%, come è possibile osservare nelle figure di seguito.

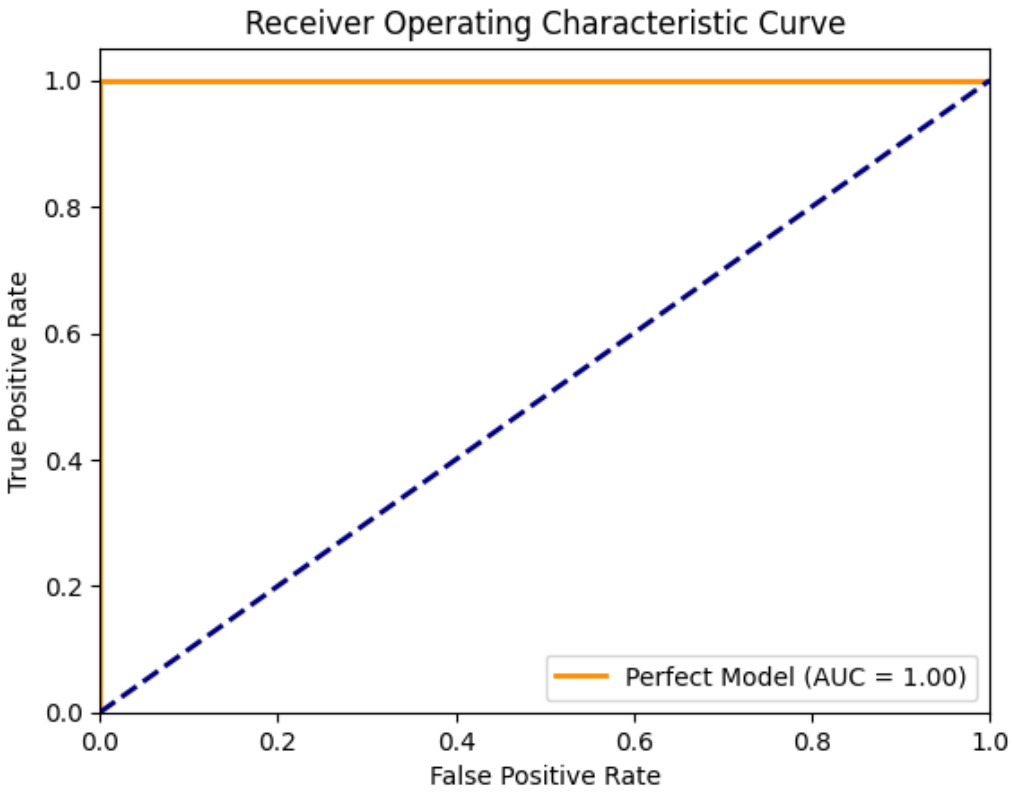

Figura 2.6: Grafico della curva ROC di un modello perfetto

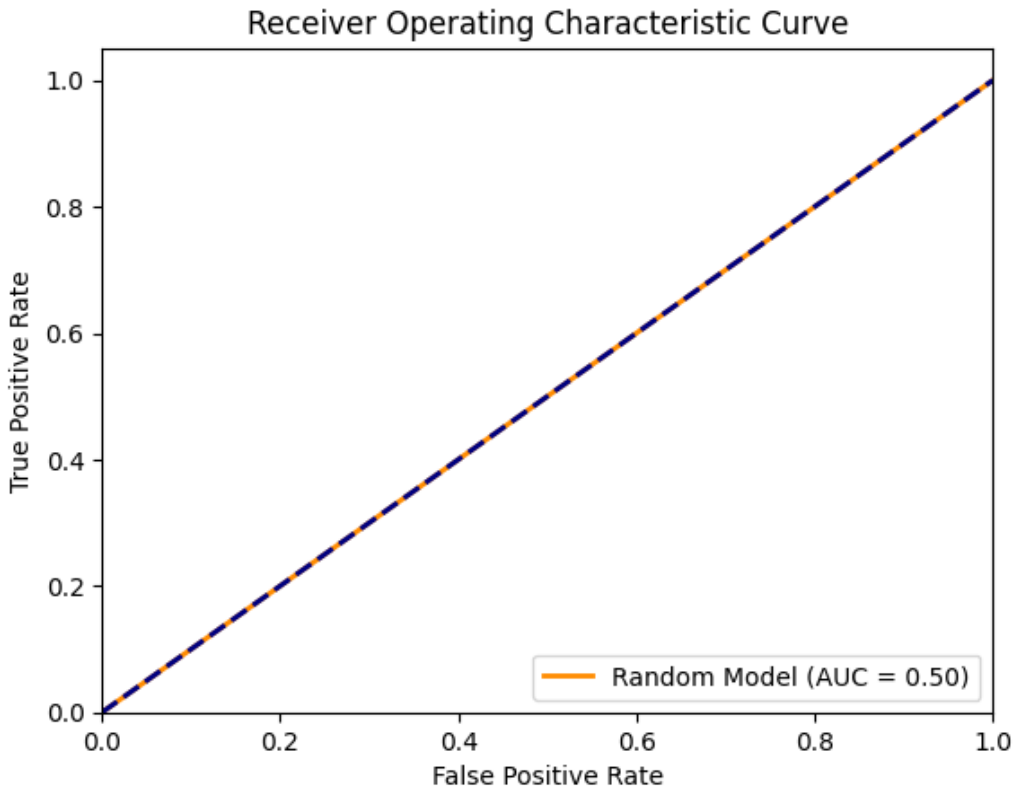

Figura 2.7: Grafico della curva ROC di un modello casuale

Il valore numerico della ROC è dato dall'area sotto la curva ROC (AUC), che rappresenta la probabilità che il modello classifichi correttamente una istanza positiva scelta a caso rispetto a una istanza negativa scelta a caso. L'AUC può assumere valori compresi tra 0 e 1, dove un valore di 0,5 indica che il modello effettua previsioni casuali, mentre un valore di 1 indica che il modello è perfetto.

Due vantaggi principali dell'utilizzare l'AUC come metrica sono che:

- AUC è scale-invariant. Misura il ranking delle previsioni, anziché i valori assoluti
- AUC è classificazione soglia-invariante: misura la qualità delle previsioni del modello, indipendentemente dalla soglia di classificazione

scelta.

2.4.4 F1 Score

L'F1 score (o F-measure) è una metrica di valutazione delle performance di un modello di classificazione binaria. È una combinazione della precisione (precision) e della recall (sensibilità). L'F1 score combina queste due metriche in modo tale da avere un singolo valore che rappresenti il bilanciamento tra la precisione e la recall del modello. L'F1 score viene calcolato come la media armonica della precisione e della recall, ed è definito come:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{2.3}$$

Il valore dell'F1 score varia da 0 a 1, dove un valore di 1 rappresenta la performance migliore possibile. Un modello con un F1 score elevato indica che il modello ha una buona capacità di classificare correttamente gli esempi positivi e negativi.

L'F1 score è particolarmente utile quando il dataset ha una distribuzione sbilanciata delle classi, dove una classe può essere presente in modo molto meno frequente dell'altra. In questi casi, l'F1 score può fornire una valutazione più accurata e generale del modello.

2.4.5 Analisi dei costi

Queste metriche utilizzate in maniera complementare, ci danno una buona misura delle prestazioni di un modello di classificazione per la detection delle frodi. Avere un buon valore di precisione, è necessario per minimizzare gli errori di classificazione in generale: sia classificare erroneamente un'istanza negativa come positiva, sia il viceversa, ha un costo.

Nel primo caso, bisogna considerare tutte le spese dovute all'assistenza del cliente la cui transazione è stata classificata in modo incoretto, mentre nel secondo caso i costi sono maggiori perché una transazione fraudolenta non è stata identificata, quindi i costi coincidono con il totale della transazione.

Per questo, è importante avere un buon valore di recall (TPR) minimizzando i falsi positivi in quanto più costosi.

Inoltre, poiché l'AUC rappresenta la probabilità che il modello classifichi correttamente una istanza positiva scelta in maniera casuale rispetto a una istanza negativa scelta in maniera casuale, un buon valore di AUC per un modello di fraud detection è indicazione di ottime prestazioni, perché vuol dire che il modello sta generalizzando correttamente.

Per questi motivi nella realizzazione del modello abbiamo cercato di massimizzare i risultati riportati da queste metriche, poiché appunto buone performance misurate in questo modo coincidono con il minimizzare i costi relativi alle frodi.

Capitolo 3

Esperimenti sul dataset open source

In questo capitolo andremo ad analizzare come abbiamo sviluppato il modello di fraud detection, a partire dalla scelta del dataset fino alla valutazione e validazione del modello, passando per i passaggi necessari allo sviluppo.

Per prima cosa, analizzerò nel dettaglio il dataset utilizzato e le trasformazioni applicate ai dati da dare in pasto all'algoritmo, successivamente analizzerò la scelta e la valutazione del modello.

3.1 Il Dataset

Il dataset utilizzato per l'addestramento del modello è stato preso dalla competizione "IEEE-CIS Fraud Detection" ed è un dataset messo a disposizione pubblicamente sul sito Kaggle.com. Per ogni transazione bancaria, contiene informazioni riguardanti:

- il totale della transazione
- la carta di credito utilizzata (tipo di carta, emittente bancario, paese, ecc.)
- gli indirizzi di spedizione e fatturazione

- i domini delle email (sia dell'acquirente che del ricevente)
- informazioni sui prodotti acquistati
- data della transazione
- informazioni su:
  - counting (es. quanti indirizzi sono associati alla carta)
  - match (es. sui nomi o sugli indirizzi)
  - timedelta (es. giorni passati dall'ultima transazione)

Ogni riga del dataset contiene una label che indica se la transazione è stata classificata dagli organi competenti come fraudolenta oppure no (colonna "isFraud").
Per fare supervised learning è necessario che le label vengano apposte in modo corretto, e in questo caso le informazioni sulla classificazione della transazione sono fornite dal gestore di metodi di pagamento Vesta, una delle principali società di servizi di pagamento al mondo.

Il dataset è diviso in due parti, la prima contenente tutte le informazioni sopra citate, la seconda invece informazioni riguardanti l'identità, cioè una serie di informazioni sul dispositivo utilizzato per la transazione. Queste non sono state utilizzate per la nostra analisi per mancanza di risorse, ma possono essere prese in considerazione in futuro per migliorare le performance.

3.1.1 Preprocessing dei dati

Prima di poter utilizzare il dataset, è stato necessario apportare delle modifiche. Ad esempio, è stato necessario modificare alcune features (colonne) per renderle compatibili con il formato richiesto dai modelli della libreria Scikit Learn, quindi la codifica di alcuni campi da Stringa a Float, così come la scelta del metodo per la gestione dei valori N/A (features che non sono state valorizzate per alcune transazioni), sono state alcune delle manipolazioni effettuate.

One Hot Encoder

La libreria Sci-kit Learn mette a disposizione diversi strumenti per il preprocessing dei dati, tra questi vi è il One Hot Encoder (OHE), che ha fatto al caso nostro per la codifica dei campi di tipo stringa che non potevano essere tralasciati nell'analisi (es. gli indirizzi).

Questi campi si potevano codificare mediante un numero intero identificativo, cioé associando un numero ad ogni possibile valore che assume la stringa nel dataset, tuttavia è stato scelto il OHE in quanto le stringhe in questione (es. gli indirizzi) erano tra le feature più interessanti nell'analisi, e in questo modo si è evitato il rischio che il modello non riuscisse ad interpretare correttamente il loro significato.

Ciò che l'encoder fa attraverso la sua funzione principale fit_transform è prendere tutti i valori che assume la feature da codificare e generare un array di colonne una per ognuno di questi valori, inserendo poi, in ogni riga del dataset e per ogni feature (colonna) generata, 1 se la riga aveva quel valore per quella feature, 0 altrimenti.

Es. Data la seguente tabella:

Code	Cibo	Bevanda
0	Pizza	Cocacola
1	Pasta	Acqua

Vogliamo codificare con il OHE le features "Cibo" e "Bevanda". I valori che assume la feature "Cibo" sono "Pizza" e "Pasta", mentre per "Bevanda" abbiamo "Acqua" e "Cocacola". Il OHE prende in input una tabella di questo tipo e ritorna una tabella del tipo

Code	Pizza	Pasta	Acqua	Cocacola
0	1	0	0	1
1	0	1	1	0

Gestione degli N/A values

In questo caso, per quanto riguarda gli N/A values sono state utilizzate diverse tecniche per la gestione, tra cui il SimpleImputer fornito da sklearn, che permette di sostituire i valori N/A con:

- la media tra i valori presenti
- la mediana tra i valori presenti
- il valore più frequente tra quelli presenti

Queste tecniche hanno portato risultati molto simili tra loro, per cui è stato effettuato anche un tentativo sostituendo i valori N/A con un valore di default scelto in modo arbitrario (nel caso specifico è stato scelto il valore 0). Anche questa volta, i risultati sono stati simili, per cui si è deciso di proseguire con questa strategia in quanto meno costosa a livello di risorse, poiché il lavoro svolto dal SimpleImputer è stato sostituito da una semplice funzione per sostituire (in place) i valori N/A con il valore arbitrario 0.

3.1.2 Oversampling del dataset

Un problema fronteggiato durante lo sviluppo del modello è stato quello per cui il numero di istanze positive (isFraud=1), cioè quelle fraudolente, presenti nel dataset erano molto meno numerose di quelle negative (non fraudolente). Questo ha portato nei primi esperimenti a casi di overfitting, cioé una situazione che si verifica quando il modello predice correttamente sui dati su cui viene allenato ma performa male quando gli vengono forniti dati nuovi.
Questo è giustificato dal fatto che se in un dataset abbiamo solo l'1% di label positive (e 99% negative) il modello rispondendo sempre in modo negativo risponderà bene il 99% delle volte, per cui avrà un buon valore di precisione anche se in realtà sta performando decisamente male (cioé non dice mai che è un ordine fraudolento). Per ovviare a questo problema, è stata utilizzata una strategia di oversampling del dataset, cioè un ribilanciamento, che consiste

nell'inserire nel dataset una serie di istanze già presenti, moltiplicandole in modo tale che il nuovo rapporto tra istanze positive/negative sia esattamente del 50%. Questo è stato fatto con la classe RandomOversampler della libreria ImbalancedLearning.
Applicare questa tecnica sia ai dati di training che a quelli di test ha portato ad un grande incremento delle performance del modello.

3.2 Approcci e risultati

Dopo la scelta del dataset e il preprocessing dei dati, si passa alla scelta e implementazione del modello di machine learning più adeguato. Il problema di stabilire se una transazione può essere o meno fraudolenta è un problema di classificazione: basandosi su ciò che ha appreso, il modello prende in input una transazione e decide la probabilità che questa sia fraudolenta oppure no, se la probabilità raggiunge una certa soglia allora classifica la transazione come fraudolenta.

3.2.1 Modelli addestrati

Durante i vari esperimenti sono stati addestrati diversi modelli e ne sono state valutate le performance, per poi arrivare alla scelta definitiva, cioè quello che offriva le prestazioni migliori. La scelta dei modelli da addestrare si è basata sulla letteratura, come ad esempio lo studio *Detection of Fraudulent Transactions in Credit Card using Machine Learning Algorithms* [1]. Per tutti gli algoritmi, sono state utilizzate le tecniche di preprocessing dei dati citate nella sezione 3.1.1.

Decision tree

Un Decision Tree è un modello di classificazione che si basa su una serie di decisioni prese in sequenza per classificare un'osservazione in una o più categorie. In pratica, il modello crea un albero di decisione con nodi che rap-

presentano domande a cui è possibile rispondere con "sì" o "no" e foglie che rappresentano le categorie di output. Il processo di creazione dell'albero di decisione consiste nella selezione della variabile più significativa per dividere l'insieme di dati in due parti in modo che la varianza all'interno di ogni parte sia minima. Questo processo viene ripetuto iterativamente fino a quando non si ottiene un albero completo.

Di seguito sono riportati i risultati ottenuti da questo algoritmo.

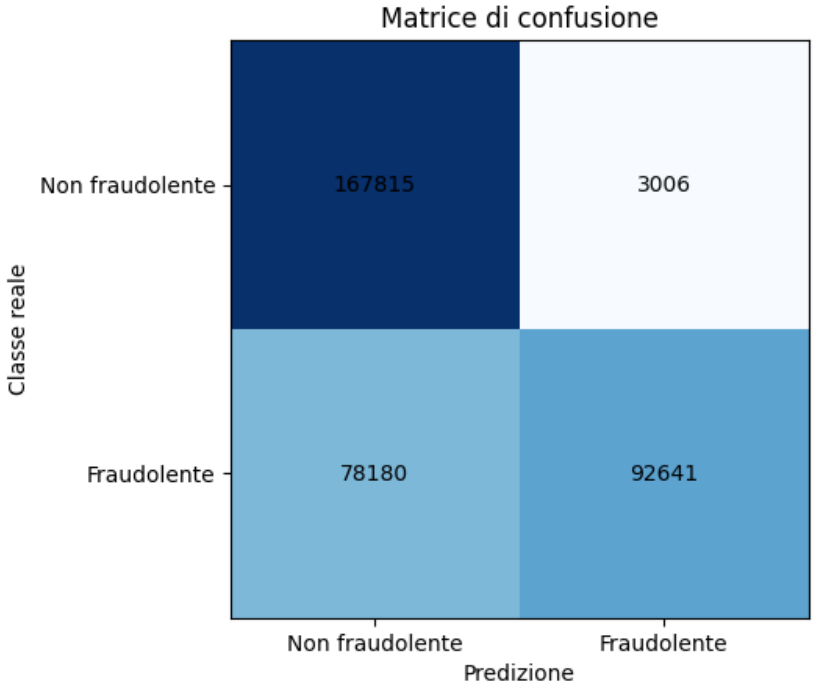

Figura 3.1: Decision tree: Matrice di confusione

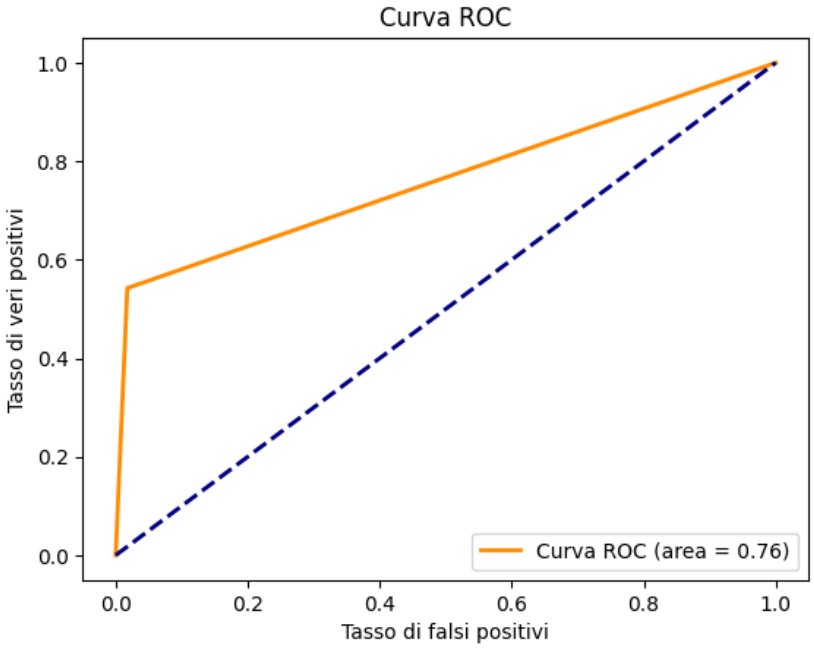

Figura 3.2: Decision tree: Curva ROC

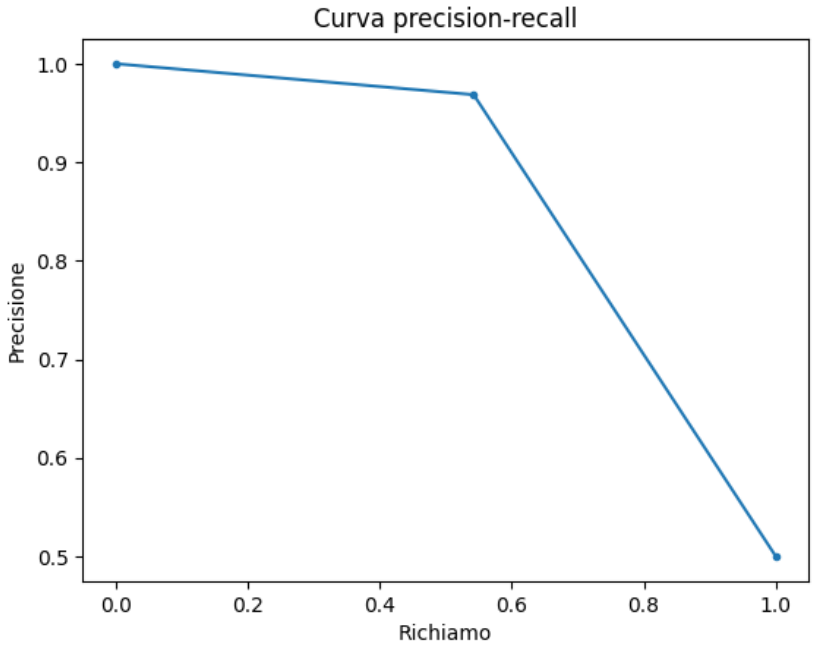

Figura 3.3: Decision tree: Curva precision-recall

Random forest

La Random Forest è invece una tecnica di apprendimento automatico basata sull'aggregazione di più alberi decisionali (in genere con la stessa profondità) che vengono costruiti su sottoinsiemi casuali e indipendenti del dataset di training. Ogni albero della Random Forest viene addestrato su un sottoinsieme dei dati e delle variabili di input, in modo da ridurre il rischio di overfitting e di migliorare la generalizzazione. Una volta addestrati tutti gli alberi, il modello aggrega le previsioni di ogni albero per stabilire una previsione finale.

La Random Forest è in genere più accurata rispetto a un singolo Decision Tree perché combina le previsioni di più alberi e riduce gli errori, ma è anche più complessa e richiede più tempo e risorse computazionali per l'addestramento e la predizione.

La random forest è quindi un modello cosiddetto "ensemble", cioé una tecnica di apprendimento automatico che combina i risultati di più modelli di apprendimento per ottenere una previsione più accurata rispetto a un singolo modello.

Di seguito i risultati ottenuti:

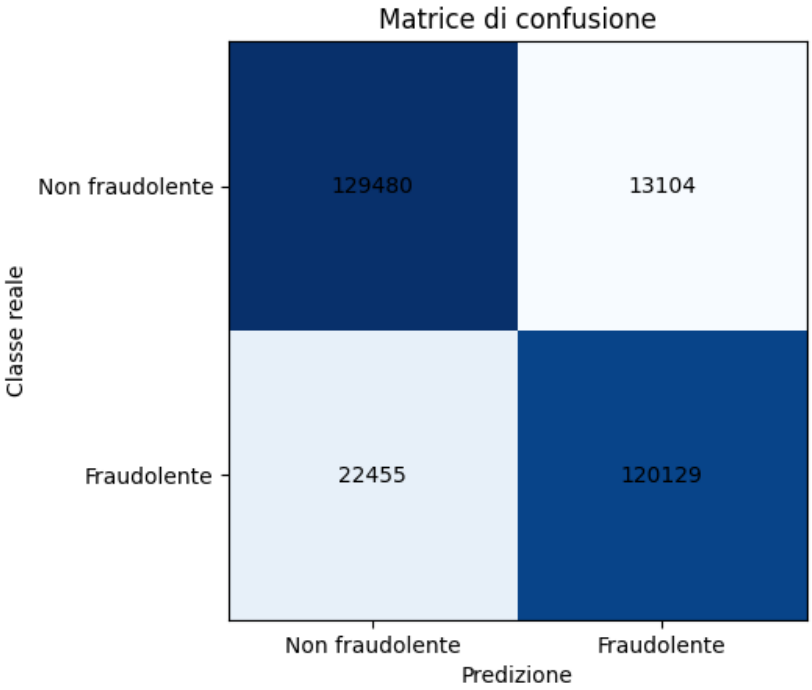

Figura 3.4: Random forest: Matrice di confusione

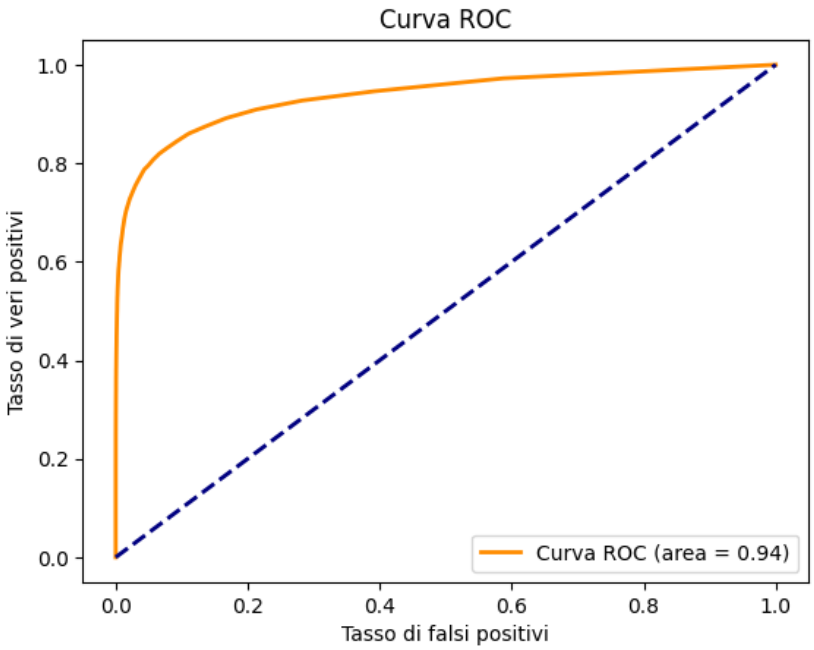

Figura 3.5: Random forest: Curva ROC

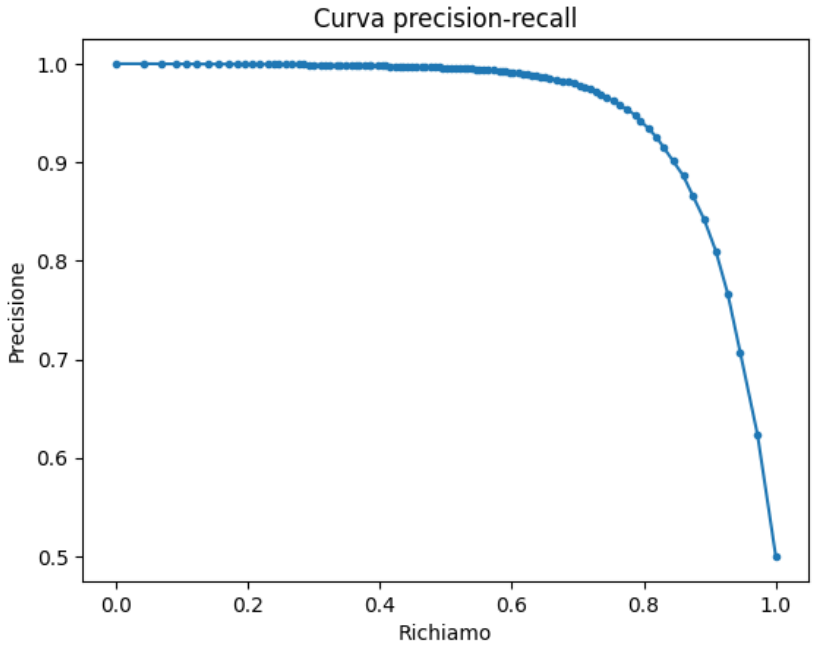

Figura 3.6: Random forest: Curva precision-recall

Gradient boosting

Poiché i metodi ensemble per questo tipo di problema sembravano performare meglio, sono state valutate le performance anche dell'algoritmo gradient boosting, un algoritmo ensemble capace di selezionare automaticamente le variabili più rilevanti per la previsione da dataset sia con tante ch con poche features. Il gradient boosting classifier è un algoritmo di machine learning che costruisce un insieme di modelli di decisione deboli e li combina per creare un modello più forte. Ad ogni iterazione, viene addestrato un nuovo modello sui dati originali insieme agli errori residui dei modelli precedenti, fino a quando non si raggiunge un certo numero di modelli o un criterio di arresto. Alla fine, tutti i modelli sono combinati insieme per creare un modello di classificazione più forte.

Queste sono le performance del Gradient boosting:

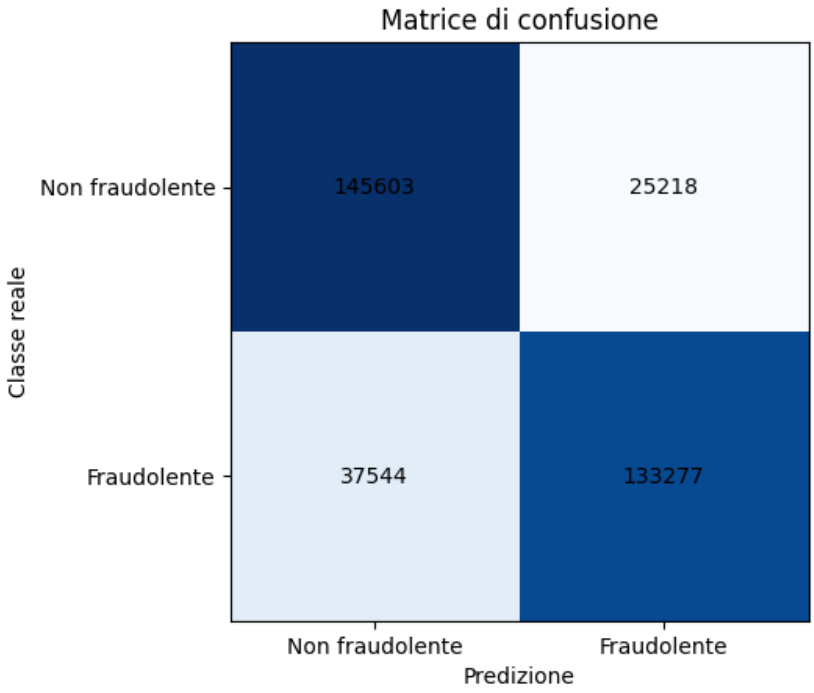

Figura 3.7: Gradient boosting: Matrice di confusione

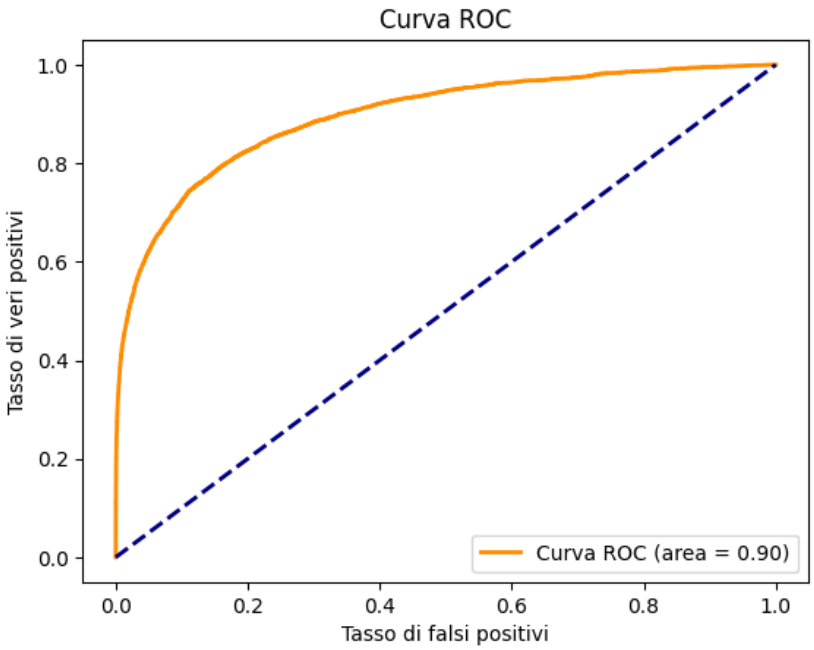

Figura 3.8: Gradient boosting: Curva ROC

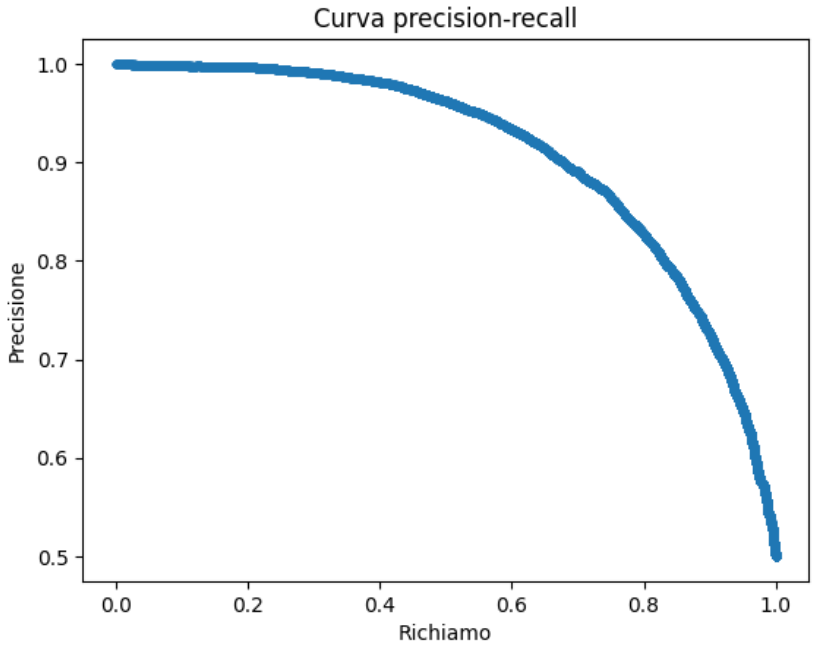

Figura 3.9: Gradient boosting: Curva precision-recall

Adaptive boosting

Infine, è stato addestrato anche un modello di adaptive boosting, un algoritmo ensemble semplice e veloce, quindi adatto a dataset di grandi dimensioni, che si basa sul dare dei pesi ai campioni utilizzati dai singoli modelli e utilizzare questi pesi per dare una valutazione complessiva basata sulle capacità di classificazione del modello. L'Adaptive Boosting (AdaBoost) è un algoritmo di boosting che addestra una sequenza di modelli di decisione deboli su differenti sottoinsiemi di dati. I modelli di decisione deboli sono poi combinati in un modello di classificazione più forte mediante una votazione pesata. L'AdaBoost è utile per la classificazione e la regressione, ma è sensibile alla presenza di outlier nei dati e al rumore.

Di seguito, le performance:

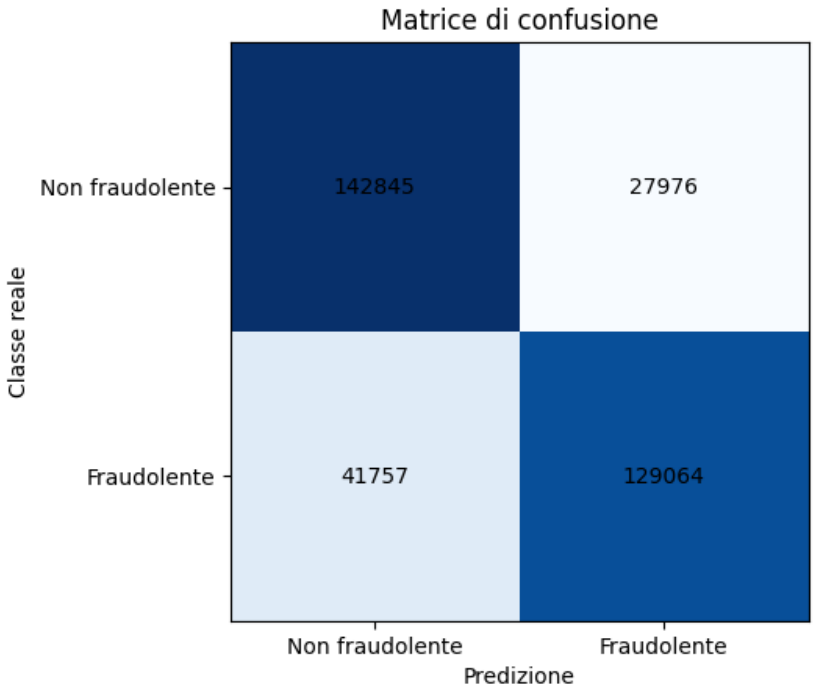

Figura 3.10: Adaptive boosting: Matrice di confusione

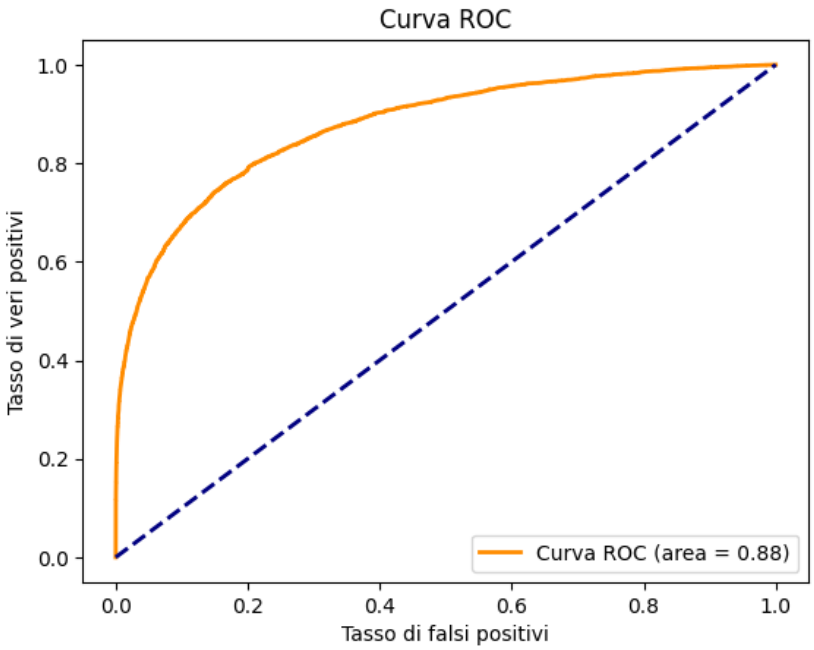

Figura 3.11: Adaptive boosting: Curva ROC

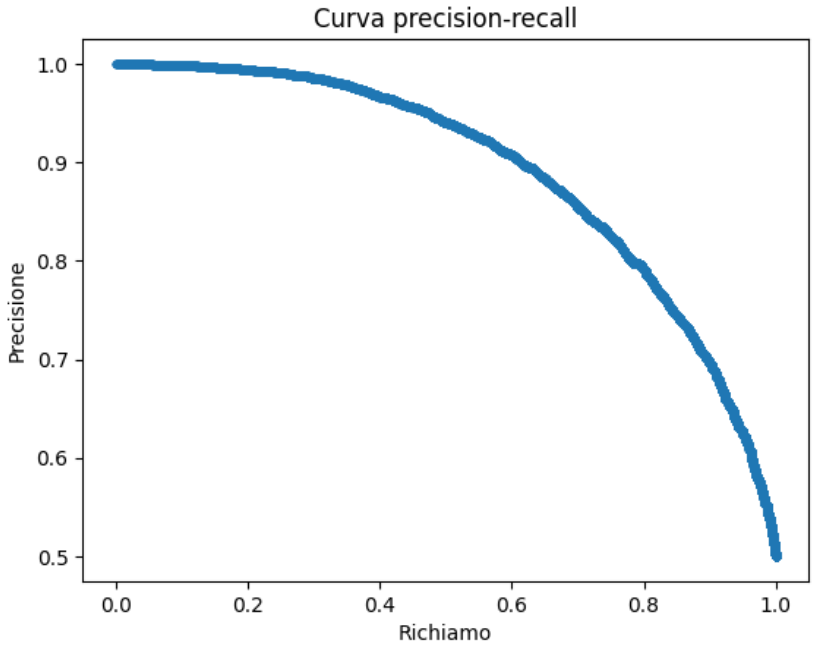

Figura 3.12: Adaptive boosting: Curva precision-recall

3.2.2 Analisi dei risultati

Di seguito è riportata una tabella riassuntiva delle performance dei modelli, che permette di visualizzare e comparare i risulitati ottenuti dagli esperimenti.

Modello	AUC	Precision	Recall	F1
Decision Tree	0.76	0.97	0.54	0.70
Random Forest	0.94	0.80	0.85	0.87
Gradient boosting	0.90	0.84	0.78	0.81
Adaptive boosting	0.88	0.82	0.75	0.78

Tabella 3.1: Performance dei modelli di machine learning addestrati

Come si può vedere nella figura, l'algoritmo Random forest è quello che ha assicurato le performance migliori in quasi tutte le metriche di valutazione, anche se con una precisione leggermente più bassa. Come già anticipato, nel problema della fraud detection è più importante classificare correttamente le istanze positive (quindi avere un buon valore di richiamo) anche a discapito di classificare alcune istanze negative in modo erroneo (precisione più bassa), in quanto i costi sono decisamente maggiori.

I fattori che hanno portato al successo dell'algoritmo Random forest comprendono il fatto che riesce a tenere conto dello storico delle transazioni legate ad un utente per individuare le anomalie nelle nuove transazioni, il che è considerato uno dei fattori più influenti per la determinazione delle transazioni fraudolente, come analizzato nello studio condotto da R. Sailusha [8].

Questo modello verrà successivamente utilizzato per portare avanti un esperimento su dei dati reali, che sarà spiegato nel capitolo successivo.

3.2.3 Considerazioni durante l'addestramento

Durante questi esperimenti, il dataset è stato diviso in training set (70%) e test set (30%), ed è stato possibile grazie al parametro *random_state* far variare in modo casuale le istanze che ad ogni round di addestramento fa-

cevano parte di un insieme o dell'altro. Durante l'addestramento si è poi analizzato come le variazioni di alcuni parametri si riflettono nelle variazioni delle performance.

Tuning dei parametri

Ogni modello andava configurato per ottenere il massimo delle prestazioni, quindi è stato importante configurare i parametri di inizializzazione, come ad esempio:

- n_estimators: il numero di modelli più semplici utilizzati dai metodi ensemble
- class_weight: il peso dato ad ognuna delle classi del dataset, nel nostro caso 0 (non fraudolento) e 1 (fraudolento)
- threshold: in italiano "soglia", si riferisce a un valore di soglia usato per distinguere le classi in un problema di classificazione binaria

Il parametro n_estimators è stato fissato al valore di default (=100), in quanto è stato osservato che aumentare il questo valore coincideva con un rallentamento delle performance e nessun evidente incremento della precisione delle predizioni del modello.

Il parametro class_weight è stata una prima soluzione al problema discusso nella sezione 3.1.3, ossia il numero ridotto di istanze positive rispetto a quelle negative. Impostando il valore su 'balanced', il modello assegna un peso alle istanze di una classe in proporzione alla frequenza con cui comparivano nel dataset. Tuttavia il problema sopra citato è stato poi risolto completamente attraverso il ribilanciamento, che ha prodotto risultati migliori.

Anche il threshold è stato un parametro fondamentale per arrivare alla costruzione di un buon modello. Essenzialmente esso rappresenta la soglia di probabilità oltre il quale un'istanza viene classificata come positiva o negativa. Un threshold molto alto, andrà a influenzare la sensibilità del modello a classificare istanze positive, in quanto tenderà ad essere più conservativo,

mentre un valore troppo basso porta ad una diminuzione della precisione, perché il processo tenderà troppo facilmente a classificare le istanze come positive, anche se in realtà non lo sono. Nel nostro caso, abbassare il valore del threshold è stato un passaggio fondamentale per ottenere un buon valore di richiamo, anche se a costo di un leggero decremento osservato nella precisione.

Durante i vari esperimenti inoltre, è stato sempre tenuto conto della possibilità di overfitting, una situazione in cui il modello performa bene sui dati su cui è addestrato ma non riesce ad adattarsi a dati nuovi, per cui per garantire più varietà sia nella fase di addestramento che in quella di test, sono stati utilizzati a giro tre valori diversi di random_state, che in base al valore selezionato inseriva delle istanze diverse nel train e test set (in modo casuale).

Capitolo 4

Esperimento su dati reali

Adesso che abbiamo un modello che performa bene su dati di transazioni generiche, proviamo ad applicare lo stesso algoritmo ad un caso reale. L'esperimento è stato portato avanti sui dati degli ordini di una famosa azienda di fashion italiana, che esporta il suo marchio in tutto il mondo nel settore dell'abbigliamento di lusso. Questi dati sono stati dati in pasto all'algoritmo di Random Forest sviluppato precedentemente, dopo un'attenta analisi ed elaborazione delle feature delle transazioni in questione. Successivamente, il modello è stato valutato e validato secondo i criteri analizzati nella sezione 2.4. Lo scopo finale è quello di realizzare un prodotto che funzioni in un sistema reale, con l'ambizione di produrre valore sia per l'azienda sia per il cliente, integrandolo nel normale flusso di esecuzione del suddetto sistema e-commerce.

4.1 I dati

Come anticipato, i dati utilizzati per lo studio sono frutto di un'estrapolazione ed elaborazione di una serie di informazioni relative agli ordini effettuati sul sistema e-commerce dell'azienda cliente.

4.1.1 Estrazione dei dati

Il dataset è stato estratto attraverso uno script scritto nel linguaggio beanshell, un linguaggio di scripting Java-based, sul sistema e-commerce basato sul software SAP (anch'esso basato su Java), che mette a disposizione una sezione per eseguire script mentre il sistema è online. In questo modo è stato possibile interrogare direttamente il database ed è stato possibile scegliere le feature più interessanti per questo tipo di analisi, attraverso i metodi Java per l'accesso agli oggetti del database.

I dati relativi alle transazioni sono già etichettati dal sistema, sulla base di alcune verifiche effettuate dal gestore di metodi di pagamento Adyen e di altre effettuate a mano dagli operatori del sistema (quindi successive al completamento della transazione). Le etichette sono quindi da considerare affidabili, per cui è stato possibile fare dell'apprendimento supervisionato.

Feature engineering

I dati relativi agli ordini contengono moltissime informazioni, alcune delle quali fondamentali per questo tipo di analisi, altre invece completamente superflue. E' stato quindi importante capire quali potevano essere quelle interessanti ai fini dell'addestramento del modello, sulla base delle ricerche effettuate nella letteratura relativa. Di seguito sono elencate le principali informazioni relative ad ogni ordine che ho deciso di tenere in considerazione per l'analisi:

- Numero dell'ordine - identificativo
- Etichetta - fraudolento o non fraudolento
- Lingua e valuta
- Indirizzo di spedizione - paese e codice postale
- Indirizzo di pagamento - associato alla carta
- Indirizzo di fatturazione

- Numero di prodotti acquistati
- Totale dell'ordine
- Numero medio di prodotti acquistati dal relativo utente
- Totale medio degli ordini effettuati dall'utente
- Numero totale di ordini dell'utente - utile per considerare i nuovi clienti
- Tipo e stato del pagamento
- Tipo di carta utilizzata
- Store da cui è stato effettuato l'ordine - negozio online US, IT, SP, RU, ecc.
- Dominio della mail dell'utente

Come si può notare, le features estratte sono concettualmente simili a quelle presenti nel dataset open source, proprio perché la letteratura suggerisce di cercare in questo tipo di caratteristiche delle anomalie che potrebbero portare un modello ad etichettare una transazione come sospetta.

4.1.2 Preprocessing

Anche in questo caso è stato necessario pre-processare i dati prima di darli in input all'algoritmo. E' stato utilizzato di nuovo il OneHotEncoder (OHE) per la gestione dei valori di tipo Stringa, che non sono adatti ai modelli di machine learning. Questa volta però a differenza del dataset utilizzato per lo sviluppo del modello, è entrata in gioco la problematica relativa alle prestazioni: i valori stringa presenti nel dataset erano molto più variegati, e siccome il OHE crea una nuova feature per ogni valore diverso presente nella feature da codificare (che indica se la entry aveva quel valore prima della codifica o no), il dataset risultante aveva più di 7000 features, un aumento notevole rispetto alle del dataset originale (20 features circa).

I dati quindi erano molto più complessi da elaborare, infatti all'inizio degli esperimenti nessuna delle esecuzioni riusciva a terminare per mancanza di risorse. Per questo motivo, è stato impossibile effettuare un oversampling del train set, che invece è stato effettuato solo sul test set affinché le valutazioni fossero effettuate su un buon insieme e quindi restassero veritiere. Tuttavia anche riducendo le operazioni effettuate sui dati, alcune volte l'esecuzione (preprocessing, addestramento del modello e predizione) riusciva a terminare e a mostrare dei risultati, mentre altre volte si interrompeva per mancanza di risorse, generalmente se si avevano altri processi attivi contemporaneamente (es. il browser).

Il passaggio fondamentale è stato notare che il OHE salvava gli 0 e gli 1 corrispondenti ad ogni riga e ad ogni feature codificata in formato float64, cioè numeri a virgola mobile (64 bit), anche se questo non era necessario, per cui imponendo al OHE di codificare i valori in formato int8, cioé numeri interi a 8 bit, c'è stato un grande aumento nelle prestazioni grazie ad un utilizzo delle risorse notevolmente minore. Infine, visto che era stato osservato precedentemente che la gestione dei valori N/A con un metodo piuttosto che un altro non si rispecchiava in un miglioramento/peggioramento delle performance dell'algoritmo, è stato scelto ancora di sostituire quei valori con il valore di default 0.

4.2 Analisi dei risultati

In questa sezione andremo ad analizzare nel dettaglio i risultati ottenuti dagli esperimenti condotti sul caso di studio reale dell'azienda di fashion presentata in precedenza. Ricordiamo che i dati su cui sono stati effettuati i test consistevano di un sottoinsieme del dataset di partenza, che è stato successivamente bilanciato per avere un buon numero sia di istanze fraudolente che non fraudolente.

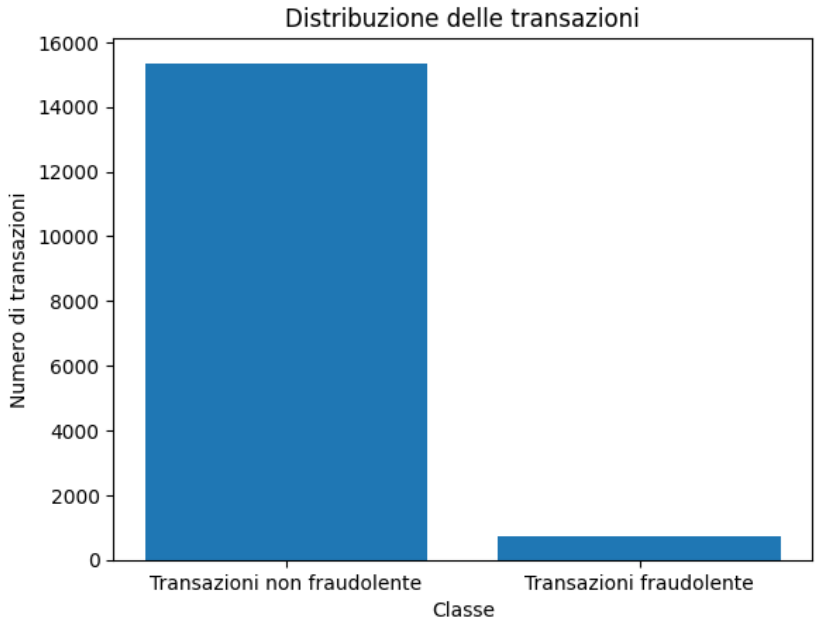

Figura 4.1: Distribuzione delle transazioni del test set prima del ribilanciamento

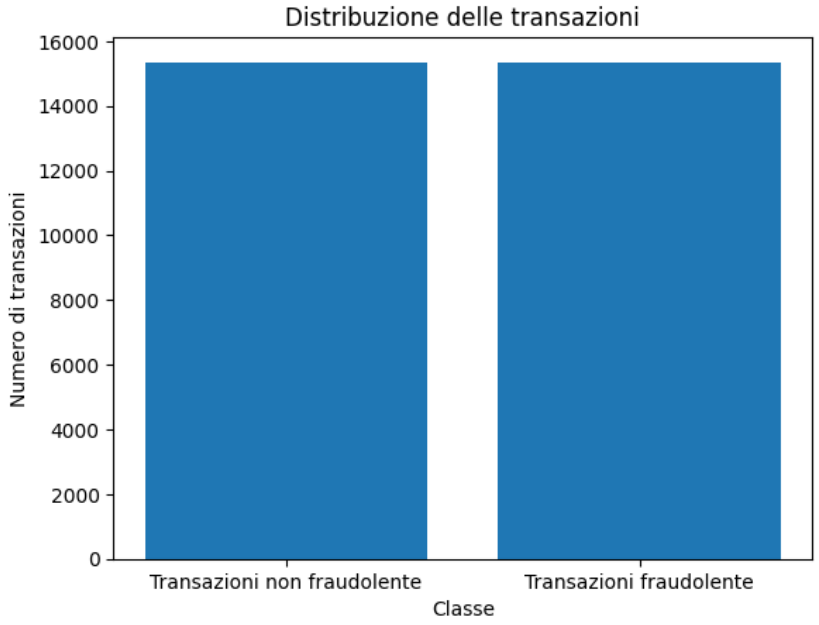

Figura 4.2: Distribuzione delle transazioni del test set dopo il ribilanciamento

Come si può notare da questi grafici, prima del bilanciamento le istanze positive non erano abbastanza rappresentate, quindi il rischio di fare overfitting era molto alto (in quanto un modello rispondendo sempre "no" poteva avere comunque un valore di precisione superiore al 90%). Dopo il bilanciamento, oltre ad aumentare il numero totale di istanze del test set, è aumentata la rappresentatività della classe dei fraudolenti, ragion per cui si può considerare sufficiente la generalità delle valutazioni così come la bontà dei risultati.

4.2.1 Matrice di confusione

Come già analizzato in precedenza, la matrice di confusione è un'ottima metrica per misurare le performance di un modello, basandosi su quante istanze classifica correttamente e quante invece no. Di seguito viene riportata la matrice di confusione del modello applicato ai dati dell'azienda di fashion sopra citata.

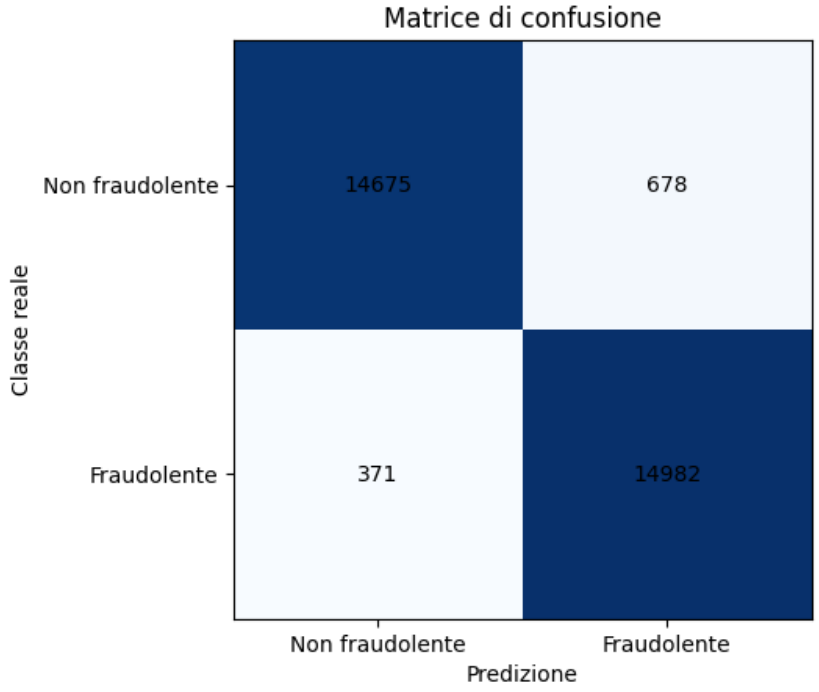

Figura 4.3: Matrice di confusione del modello

Ricordiamo che i riquadri della matrice rappresentano

- in alto a sinistra, le istanze non fraudolente classificate correttamente (TN)
- in alto a destra, le istanze non fraudolente classificate erroneamente come fraudolente (FN)
- in basso a sinistra, le istanze fraudolente classificate erroneamente come non fraudolente (FN)

- in basso a destra, le istanze fraudolente classificate correttamente (TP)

Per dire che un algoritmo di machine learning performa bene è dunque necessario che la maggior parte delle istanze si trovino sulla diagonale principale della matrice, poiché su essa abbiamo il numero di veri positivi e veri negativi, mentre sulla diagonale opposta abbiamo i falsi positivi e negativi.

Dalla nostra matrice di confusione, si evince che sul totale delle istanze, cioè circa 30.000 (la somma dei quattro riquadri della matrice), divise perfettamente a metà tra fraudolente e non, l'algoritmo riesce a predire correttamente 14.675 istanze negative, sbagliando solo in 678 casi; mentre osserviamo che solo 371 istanze fraudolente sono state etichettate come non fraudolente, mentre 14.982 sono state riconosciute correttamente.

4.2.2 Curva ROC

La curva ROC come esplorato nella sezione 2.4.3 è una metrica utile per valutare la generalità di un modello, e il suo valore numerico è dato dall'AUC, l'area al di sotto della curva. In pratica, la ROC è una funzione che riassume le possibili prestazioni di un detector [9].

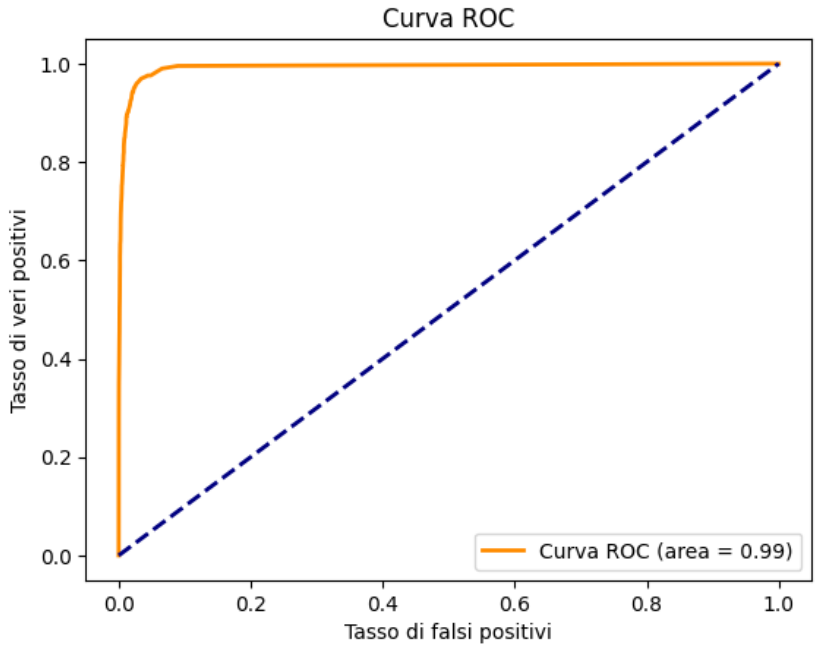

Figura 4.4: Grafico della curva ROC del modello

Come detto in precedenza, un modello con una curva ROC che tocca l'angolo in alto a sinistra, è un modello che performa quasi perfettamente, predicendo con ottima precisione sia le istanze positive che quelle negative. Questo perché l'area sotto la curva (ovvero il valore AUC) raggiunge il suo massimo in quel punto, infatti quest valore è stato successivamente calcolato ed è risultato pari a 0.99.

4.2.3 Curva precision-recall

A differenza della ROC, la curva precision-recall raggiunge il suo valore massimo (cioé massimizza l'area al di sotto della curva) quando tocca l'angolo in alto a destra. La curva precision-recall è un grafico che rappresenta la precisione (PPV) rispetto al richiamo (TPR) del modello al variare della soglia di classificazione. La precisione (PPV) è il rapporto tra i veri positivi (TP) e il numero totale di classificazioni positive effettuate dal modello (TP + FP), mentre il richiamo (TPR) è la proporzione di veri positivi (TP) rispetto al totale di casi positivi (TP + FN).

La curva precision-recall rappresenta quindi la capacità del modello di identificare correttamente i casi positivi e di evitare di etichettare erroneamente i casi negativi come positivi al variare della soglia di classificazione. Un modello perfetto si posiziona nell'angolo in alto a destra del grafico, con una precisione e un richiamo pari a 1. Un modello che non riesce a distinguere tra le due classi ha una curva precision-recall che segue la linea orizzontale corrispondente alla proporzione di casi positivi nel dataset. In generale, come per la ROC, maggiore è l'area sotto la curva, migliore è la performance del modello.

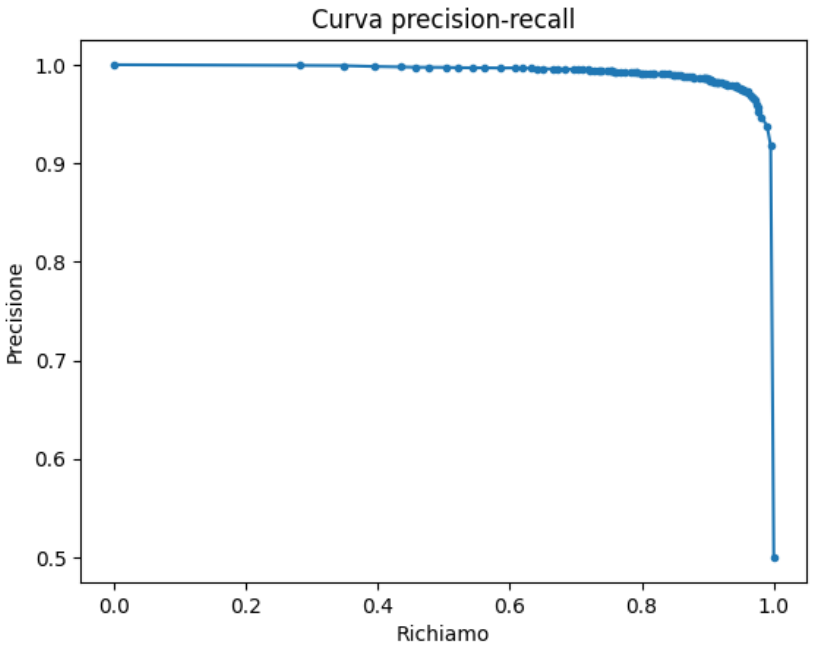

Figura 4.5: Grafico della curva precision-recall del modello

Come si evince dalla figura sopra, il si avvicina molto all'angolo in alto a destra e riesce quasi a massimizzare l'area sotto la curva, entrambi ottimi indicatori del fatto che il modello riesca a generalizzare molto bene.

4.2.4 F1 Score

Lo score F1, che come abbiamo visto dipende direttamente dai valori di precisione e recall, è stata una metrica aggiuntiva utilizzata soprattutto per confermare i risultati evidenziati con le altre metriche. Il valore del F1 score del modello è stato calcolato essere circa 0.97. Il modello quindi anche in questa metrica ha raggiunto un valore molto vicino a 1 sia sulle istanze positive che negative, come era giusto aspettarsi guardando le altre metriche. Ciò è confermato dal fatto che i valori di precisione e recall erano anch'essi molto vicini a 1, quindi l'F1 score ha confermato anche da questo punto di vista la buona capacità di predizione del modello.

4.2.5 Riassunto dei risultati

Nella figura 4.6, riportata di seguito, possiamo vedere un confronto delle metriche del modello sui dati reali e quelli open source, osservando che la Random forest ha performato leggermente meglio sui dati reali. Nonostante ci si potesse aspettare che fare previsioni sui dati provenienti dal mondo reale fosse più complesso e quindi in generale che le performance del modello sarebbero peggiorate, in questo capitolo abbiamo osservato un miglioramento dei risultati sotto tutti gli aspetti, il che ci porta a sottolineare ancora una volta l'importanza dei dati, delle informazioni e delle feature più rilevanti per il problema in questione e più in generale delle caratteristiche specifiche del dominio di applicazione, durante lo sviluppo di un modello di machine learning.

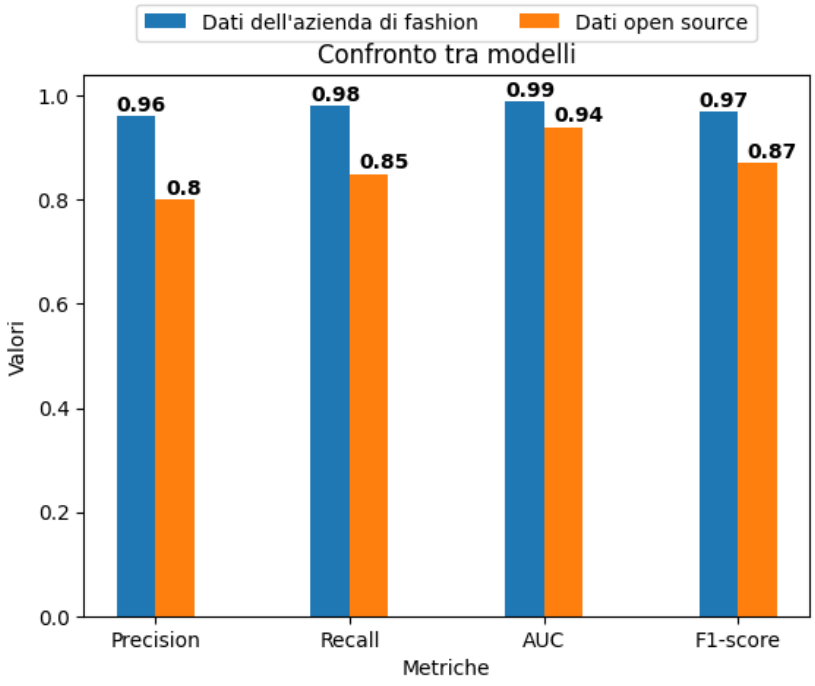

Figura 4.6: Grafico a barre delle metriche del modello su entrambi i dataset su cui è stato addestrato e testato

Capitolo 5

Conclusioni

In conclusione, la ricerca ha dimostrato che l'analisi e la prevenzione degli ordini fraudolenti sono cruciali per le aziende che operano in settori dove le transazioni online rappresentano una parte importante delle attività. L'uso di modelli di machine learning può offrire un grande supporto alle aziende per identificare in modo efficace e rapido le transazioni sospette, riducendo i rischi di perdite economiche e di danno alla reputazione.

L'analisi svolta ha inoltre evidenziato l'importanza della qualità dei dati nell'addestramento dei modelli di machine learning per la fraud detection. Il preprocessing dei dati, ovvero la pulizia, l'elaborazione e la trasformazione dei dati in modo da renderli adatti all'addestramento del modello, rappresenta una fase fondamentale della pipeline di machine learning.

In particolare, è importante la selezione delle features rilevanti per il problema di fraud detection e la riduzione della dimensionalità dei dati, in modo da evitare problemi di overfitting e migliorare l'efficienza del modello. Infatti, è importante avere accesso a quelle che sono le features più interessanti per il dominio del problema, ciò che non è fondamentale è solo rumore, che può portare a delle inaccuratezze.

L'esperimento sui dati dell'azienda di fashion ha dimostrato che l'addestramento del modello sui dati specifici può portare a risultati ancora migliori, grazie alla presenza di informazioni e caratteristiche specifiche del dominio

di applicazione. Infine, la valutazione delle performance del modello tramite metriche come precision, recall, roc auc e f1 score ha permesso di quantificare l'accuratezza dei risultati e di comprendere l'importanza di una corretta scelta del modello e delle tecniche di valutazione.

Abbiamo potuto osservare come il modello riesca a generalizzare molto bene e quindi a fare predizioni molto precise sulle transazioni, tuttavia come già detto in precedenze, prima di diventare un vero e proprio prodotto, ci sono delle considerazioni che vanno fatte per rendere il modello adatto ad operare nel mondo reale.

Ad esempio, bisognerebbe effettuare dei test con dei dati nuovi e generati dal sistema e verificare che le performance non vadano al di sotto della soglia di guadagno, così come analizzare come si comporta con la detection real time. Inoltre, si potrebbe introdurre un metodo di cross validation, una tecnica di valutazione dei modelli di machine learning che prevede di suddividere il dataset in k parti uguali, addestrare il modello su k-1 parti e valutarlo sulla parte rimanente (fold di test), ripetendo il processo k volte, con lo scopo di generalizzare meglio ed evitare overfitting.

Un possibile miglioramento ulteriore sarebbe quello di considerare i costi effettivi dovuti alle classificazioni erronee (utilizzando ad esempio delle matrici di costo), che non sono stati presi in considerazione in questa analisi ma che possono essere molto utili per calibrare meglio i parametri, sempre nell'ottica generale di ridurre al minimo i costi e massimizzare i guadagni.

Un ulteriore problema può essere legato alla scalabilità: come sottolineato in precedenza l'utilizzo delle risorse è stato un fattore importante da tenere in considerazione; si può pensare infatti di apportare delle ottimizzazioni al codice per abbattere il problema delle prestazioni e rendere quindi il modello fruibile su qualunque tipo di sistema, senza requisiti di potenza di calcolo, e affinché resti tale anche se dovessero aumentare la quantità o complessità dei dati.

Tra gli sviluppi futuri di questo progetto c'è sicuramente quello di inserire questo tipo di analisi all'interno del sistema e-commerce per cui è stato rea-

lizzato, in modo tale che ogni volta che un ordine viene effettuato vengano estratte le feature necessarie per la predizione, date in pasto al modello e venga ritornato un valore che indica se si può procedere o se la transazione è sospetta e quindi deve essere rifiutata. Per fare ciò si potrebbe considerare di integrare nel processo di completamento di un ordine sul sistema e-commerce un passaggio aggiuntivo in cui:

1. Vengono estratte le feature selezionate durante la fase di feature engineering - questo si potrebbe fare integrando nella parte Java la creazione di un oggetto con le features desiderate
2. l'oggetto creato viene esportato e dato in pasto al modello che deve essere deployato in modo tale da essere accessibile al sistema e-commerce - devono essere garantite delle interfacce di accesso al modello e definito un metodo di restituzione del risultato dell'elaborazione
3. la risposta (positiva o negativa) del modello viene memorizzata nel sistema che in base a quella decide se proseguire con l'ordine o bloccare la transazione perché potenzialmente sospetta; eventualmente si possono introdurre delle azioni successive di verifica ulteriore.

L'utente registrato sul sistema acconsente al trattamento dei dati personali, quindi non si pongono problemi di privacy nell'aggiunta di questo passaggio ulteriore.

In sostanza, sono tante le considerazioni da fare per sviluppare un prodotto commerciabile a partire da questo studio, ciò che è certo è che avere un sistema di fraud detection funzionante per proteggersi dalle frodi online sta diventando di vitale importanza per tutte le aziende che operano nel settore e-commerce.

In definitiva, la ricerca ha fornito utili indicazioni per le aziende che desiderano proteggere la propria attività dagli ordini fraudolenti, tramite l'utilizzo di strumenti basati sull'intelligenza artificiale. Tuttavia, è importante sottolineare che la fraud detection è un problema in continua evoluzione, e

che è necessario continuare a migliorare e adattare i modelli di machine learning per far fronte alle nuove tecniche di frode e alle continue evoluzioni del mercato.

Bibliografia

[1] P. K. Sadineni, "Detection of fraudulent transactions in credit card using machine learning algorithms," in *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pp. 659–660, 2020.

[2] L. Wei, M. Peng, and W. Wu, "Financial literacy and fraud detection——evidence from china," *International Review of Economics and Finance*, vol. 76, pp. 478–494, 2021.

[3] A. Z. Aisha Abdallah, Mohd Aizaini Maarof, "Fraud detection system: A survey," *Journal of Network and Computer Applications*, vol. 68, pp. 90–113, 2016.

[4] S. Mittal and S. Tyagi, "Performance evaluation of machine learning algorithms for credit card fraud detection," in *2019 9th International Conference on Cloud Computing, Data Science and Engineering (Confluence)*, pp. 320–324, 2019.

[5] A. S. Alejandro Correa Bahnsen, Djamila Aouada and B. Ottersten, "Feature engineering strategies for credit card fraud detection," *Expert Systems With Applications*, vol. 51, p. 134–142, 2016.

[6] K. Chaudhary, J. Yadav, and B. Mallick, "A review of fraud detection techniques: Credit card," *International Journal of Computer Applications*, vol. 45, no. 1, pp. 39–44, 2012.

[7] F. Carcillo, Y.-A. Le Borgne, O. Caelen, Y. Kessaci, F. Oblé, and G. Bontempi, "Combining unsupervised and supervised learning in credit card fraud detection," *Information sciences*, vol. 557, pp. 317–331, 2021.

[8] Y. Zhu, D. Xi, B. Song, F. Zhuang, S. Chen, X. Gu, and Q. He, "Modeling users' behavior sequences with hierarchical explainable network for cross-domain fraud detection," in *Proceedings of The Web Conference 2020*, WWW '20, (New York, NY, USA), p. 928–938, Association for Computing Machinery, 2020.

[9] R. Sailusha, V. Gnaneswar, R. Ramesh, and G. R. Rao, "Credit card fraud detection using machine learning," in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 1264–1270, 2020.

Ringraziamenti

Questo traguardo non sarebbe stato possibile senza alcune persone che mi sono state vicine, chi solo da qualche anno, chi da tutta la vita, per questo vorrei condividere con loro la grande gioia che provo per questo successo.

Vorrei ringraziare i miei genitori, Rocco e Maria, per non avermi mai fatto mancare nulla, per avermi sempre dato la possibilità di esprimermi al meglio, e per avermi insegnato ad alzare sempre l'asticella, a dare sempre il massimo e a non mollare davanti alle difficoltà.

Ringrazio le mie sorelle Miriam e Jlenia, i rispettivi compagni Mattia e Davide e i miei marmocchi Gabriele e Giorgia, per tutti i bellissimi momenti passati insieme, per essere stati sempre dalla mia parte e non avermi mai fatto mancare il loro sostegno e il loro appoggio, e per la gioia che continuano a portare nella mia vita.

Ringrazio nonna Costanza, una presenza fondamentale nella mia vita, per tutto l'affetto che non smette mai di dimostrarmi e per tutti gli insegnamenti che mi ha dato e continua a darmi ogni giorno, e gli altri nonni che avrei tanto voluto avere qui con me in questo momento di grande gioia, ma che so che mi guardano e sono orgogliosi di me.

Ringrazio tutti gli altri parenti, cugini e cugine, zii e zie, per essere un'unica grande famiglia, sempre unita, sempre presente, sempre insieme nei momenti di gioia così come in quelli di difficoltà.

Vorrei poi ringraziare Irene, che mi è stata sempre vicina e mi ha sempre aiutato e sostenuto con grande entusiasmo, per aver gioito con me nei momenti belli e per avermi sempre supportato in quelli più difficili, per non

avermi mai fatto perdere la fiducia in me stesso e per non mancare mai di farmi sentire apprezzato. Che questo traguardo sia solo il primo di una lunga serie da poter festeggiare insieme.

Ringrazio Alessia e Nicola, e vabbè anche Giorgia dai, gli amici di una vita, quelli che ognuno dovrebbe avere al proprio fianco, per tutte le giornate e le nottate passate a chiacchierare, per tutte le volte che non mi hanno fatto mancare la loro presenza quando ne avevo bisogno.

Ringrazio Giovanni, che in poco tempo è diventato un compagno di vita, ne abbiamo passate tante insieme e spero di poterne passare altrettante.

Ringrazio il klan, un gruppo di persone unico, in poco tempo sono diventati miei fratelli e hanno contribuito a rendere questo percorso universitario un'avventura fantastica. Sono sicuro che ognuno di loro avrà una vita piena di successi e so che ci saremo sempre l'uno per l'altro per festeggiarli.

Vorrei infine ringraziare tutti gli amici di Venosa (e dintorni), per non avermi mai fatto mancare il proprio affetto ogni volta che scendo, e per ricordarmi ogni volta che un posto per tornare ce l'ho e ce l'avrò sempre.