Multipli Approcci per la Risoluzione di un Problema di Classificazione Binaria in Presenza di Classi Fortemente Sbilanciate

FEDERICO RAVENDA[1], **ANDREA LONGOBUCCO**[2], **AND GIOVANNI CORNACCHIA**[3]

[1] *Università degli Studi Milano-Bicocca, CdlM in Scienze Statistiche ed Economiche*
[2] *Università degli Studi Milano-Bicocca, CdlM in Scienze Statistiche ed Economiche*
[3] *Università degli Studi Milano-Bicocca, CdlM in Data Science*

Compiled July 10, 2022

La previsione di nuovi churner è diventata una questione critica per i fornitori di servizi di tutto il mondo; in particolare per gli operatori di telecomunicazioni, per i quali acquisire nuovi clienti è fino a 6/7 volte più oneroso rispetto a mantenere quelli esistenti [1]. Per stare al passo con il mercato, vengono fatti notevoli investimenti per sviluppare nuove strategie anti-churn, inclusi modelli di machine learning sempre più utilizzati in questo campo.
Nella presente trattazione saranno studiati diversi approcci nelle fasi di Preprocessing e Modelling per la risoluzione di un problema aziendale tipo.
In particolare nella prima fase del progetto ci si è concentrati sull'ingegnerizzazione di alcune delle features presenti nel dataset e nell'individuazione di quali siano le variabili che presentino segnale maggiore per spiegare il tasso di abbandono.
In una fase successiva si è ipotizzato che ci potesse essere una relazione tra il grado di anomalia di una osservazione e la possibilità di un abbandono; si è deciso, quindi, di convertire il task in un problema di anomaly detection.
Nella terza fase si è fatto ricorso a modelli di Machine Learning e Deep Learning con lo scopo di classificare correttamente l'abbandono di un cliente.

CONTENTS

1. INTRODUZIONE

Negli ultimi anni, la necessità di modellizzare e prevedere il comportamento di un certo cluster di clienti ha ricoperto sempre di più un ruolo preponderante in ambito di **Business Analytics** all'interno delle aziende.
Le decisioni che guidano le politiche di un management efficiente si ritrovano a fare i conti con quelle che sono le reali evidenze derivanti da fenomeni socio-economici.
Statistici e Data Scientists hanno quindi una grossa responsabilità in quanto sta a loro cercare di estrarre *valore* dai dati che hanno a disposizione, attraverso delle analisi ad-hoc.
Una pratica molto comune è quella di cercare di estrarre informazioni dai dati che andranno a supporto delle politiche di **Customer retention**, ovvero a supporto di tutte le attività atte a

trattenere i propri clienti nel tempo.
Poichè lo scopo ultimo delle aziende è quello di massimizzare i propri profitti, è necessario porre al centro delle analisi i clienti stessi; più un cliente acquista beni o servizi dall'impresa (o più è propenso a farlo) e più alto è il suo valore per la stessa. Poichè la perdita di un cliente rappresenta per le imprese la perdita di un certo profitto, risulta di fondamentale importanza per queste indagare su quelle che possano essere le motivazioni che portano la clientela a fare un certo tipo di scelta; per questo motivo, molto spesso, oltre ad indagare le cause riguardanti l'eventuale abbandono, risulta utile cercare di *prevedere* in anticipo quale possa essere il comportamento di un cliente.
É proprio sulla base di quanto descritto fino a questo momento che viene prodotta la seguente trattazione.

A. Il Dataset

Il dataset in esame riguarda la raccolta di una serie di caratteristiche dei clienti di un'azienda. Lo scopo dell'analisi è quello di prevedere se un cliente possa decidere di abbandonare o meno l'acquisto di beni e/o servizi presso l'azienda (e quindi se questo possa essere classificato come un eventuale *"**Churner**"*).
Le variabili di cui si compone il set di dati in questione sono le seguenti[1]:

1. **external_id:** *object*, identificativo alfanumerico dell'utente;
2. **how many_ok_urls:** *integer*, numero di siti che il motore semantico è riuscito ad analizzare;
3. **how many_ko_urls:** *integer*, numero di siti che il motore semantico non è riuscito ad analizzare;

4-10. **os_**:** *dummy*, sistema operativo utilizzato;
11-20. **browser_**:** *dummy*, browser utilizzato;
21-24. **feriale_**:** *float*, frazione di siti visitati in un determinato momento della giornata di un giorno lavorativo (sabato escluso) [variabile normalizzata a 1];
25-28. **weekend_**:** *float*, frazione di siti visitati in un determinato momento della giornata di un giorno del weekend [variabile normalizzata a 1];
29-37. **L**_**:** *float*, frazione di pagine web contenenti un testo dalla lunghezza indicata dal numero di parole [variabile normalizzata a 1];
38-63. **categories_**:** *float*, frazione di siti web visitati appartenenti alla categoria indicata, espressa in percentuale;
64-87. **admants_**:** categorie semantiche ad hoc, i valori risultano tutti nulli;
88-91. **CINEMA; CALCIO ; SPORT ; SKY_FAMIGLIA:** *dummy*, pacchetti di canali Sky scelti dall'utente, sono quattro variabili;
92-96. **FLG_**:** *dummy*, abbonamento Sky scelto dell'utente;
97-100. **STB_**:** *dummy*, tipologia di decoder utilizzato dall'utente;
101. **DATA_RIF:** *date*, data di rilevazione dell'osservazione;
102. **Pdisc:** *dummy*, ***variabile target*** che riguarda gli utenti che hanno cessato l'acquisto di beni o servizi (è uguale a 0 se l'utente non è un churner e uguale a 1 se il cliente è un churner)

Il dataset è composto da **330.586** righe e **102** colonne.
Si è provveduto ad eliminare le variabili sopraelencate che risultano avere solo valori costanti; le colonne che sono state eliminate sono le seguenti: *os_bds, os_osx, browser_android, browser_chromium, browser_edge, browser_ie, categories_emotions* e tutte le variabili *admants*.

[1] si è deciso per semplicità di indicare come ipotetico esempio solo una variabile per macrocategoria

Dopo questa prima fase di screening il dataset ridotto si compone di **72** variabili.

2. EDA

Una primissima esplorazione dei dati fa emergere quella che sembrerebbe essere la problematica principale del task in esame, si ci trova in un contesto di ***classi fortemente sbilanciate***; analizzando la variabile *Pdisc* si può notare che i **churner** sono solo circa il 3% delle osservazioni totali (ovvero solo 9950 osservazioni), mentre il restante 97% sono **non churner**.
Questo è un problema che ricorre abitualmente in task riguardanti la rilevazione di frodi e churner: si tratta di fenomeni in cui una classe domina fortemente l'altra. Questo sbilanciamento può creare grossi problemi di estrazione di informazione "chiara" dai dati.
Per quanto riguarda la variabile *external_id* ci si è potuti rendere conto che alcuni codici identificativi fossero ripetuti; un'analisi dettagliata ha mostrato che vi sono **4.172** osservazioni che presentano copie di codici identificativi già esistenti nel dataset e che alcune di queste copie compaiono anche più di una volta. É però emerso che nessun identificativo che compare più di una volta presenta un cambiamento nella variabile target *Pdisc*, ma piuttosto presenta dei cambiamenti nelle covariate; si può supporre, guardando i dati, che per alcuni clienti sia stata fatta più di una registrazione dai sistemi informativi nel momento in cui questi abbiano aggiunto o tolto un pacchetto o un servizio nel loro abbonamento.
Successivamente ci si è domandati quale potesse essere l'incidenza di abbandono per coloro i quali sono proprietari di tutti i pacchetti di canali (*CINEMA; CALCIO ; SPORT ; SKY_FAMIGLIA*) e per coloro che non ne possiedono neanche uno; mentre si è potuto notare che solo l'1% di clienti che sono proprietari di tutti i pacchetti è un churner, invece è risultato che circa il 18% di clienti che non possiede nessun pacchetto decide di abbandonare (Figura 1).

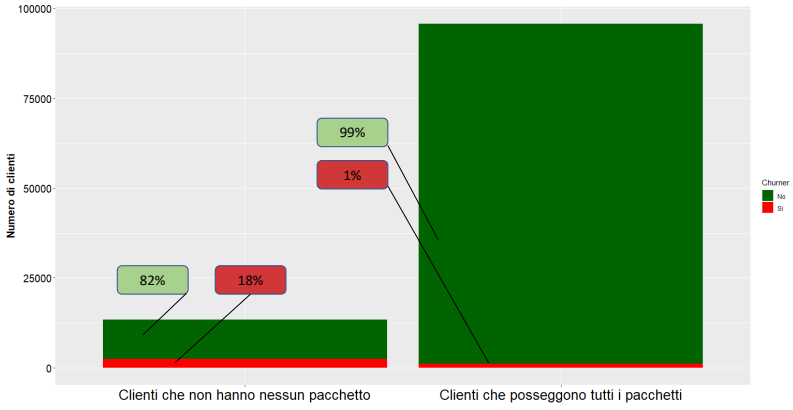

Fig. 1. Il grafico in figura mostra rispettivamente i BarPlot del numero di clienti che non posseggono nessun pacchetto canali e del numero di clienti che possiedono tutti i pacchetti canali, condizionatamente al numero di churner. Si evince che in proporzione i clienti che non posseggono pacchetti tendono ad abbandonare molto più facilmente la compagnia.

Ci si è voluti porre le stesse domande per quanto riguarda gli abbonamenti sottoscritti con l'azienda

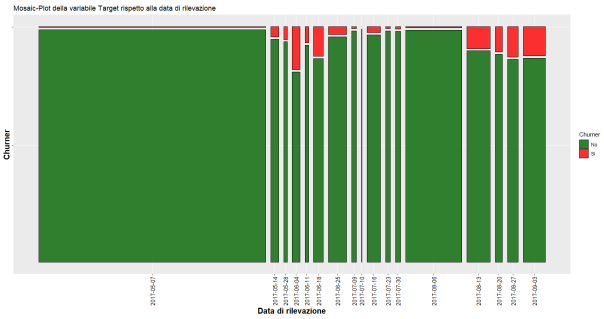

Fig. 2. Il Mosaic-Plot mette in evidenza quello che potrebbe essere un certo andamento stagionale della proporzione totale di churner rispetto alle date di rilevazione. Questo tipo di andamento sembrerebbe suggerire la possibile presenza di segnale nella variabile *DATA_RIF*.

(*FLG_MV; FLG_MYSKYHD ; FLG_HD ; FLG_MYSKY; FLG_SKY_ON_DEMAND*); in questo caso, mentre nessun cliente che ha sottoscritto tutti gli abbonamenti con l'azienda ha deciso di abbandonare (percentuale di non abbandono pari al 100%), invece circa il 14% di coloro che non ne hanno sottoscritto nessuno è un churner.
Emerge dall'analisi esplorativa che vi è una certa tendenza all'abbandono da parte di clienti che hanno meno servizi e/o pacchetti di canali attivi; si può quindi supporre che un cliente che gode di un maggior numero di servizi e/o pacchetti sia maggiormente fidelizzato nei confronti della compagnia rispetto ad altri.
Per quanto riguarda la variabile *DATA_RIF*, che restituisce la data di rilevazione della relativa osservazione, è necessario fare un discorso a parte. L'analisi del dataset ha fatto emergere che le rilevazioni presenti sono state fatte ad intervalli di tempo **irregolari** nei soli mesi di *Maggio, Giugno, Luglio, Agosto e Settembre* dell'anno 2017 (nel mese di *Settembre* inoltre le rilevazioni vengono fatte solo giorno 3). Sembrerebbe essere presente una certa **stagionalità** rispetto al numero di churner per ogni data di rilevazione. Le rilevazioni vengono fatte complessivamente in 17 giorni differenti ed hanno un intervallo temporale che va dal 7 Maggio del 2017 al 3 Settembre del 2017 (si osservi Figura 2).

Ci si è infine resi conto che le variabili relative ai pacchetti di canali attivi (*CINEMA; CALCIO ; SPORT ; SKY_FAMIGLIA*) presentano una tripla modalità (possono essere uguali a 0,1 o 2); tutte le ossevazioni che hanno modalità pari a 2 in un pacchetto presentano lo stesso valore anche per tutti gli altri. Le osservazioni con questa caratteristica sono solo 6 e sono tutti non churner. Probabilmente i valori pari a 2 possono essere dovuti al fatto che i pacchetti siano nominali; è possibile che un singolo cliente possa quindi abilitare più di un pacchetto cedendolo a terzi (ad esempio ai figli).

3. FEATURES ENGINEERING

Gli algoritmi di apprendimento utilizzano i dati di input per produrre previsioni. Molto spesso, tuttavia, i dati forniti potrebbero non essere sufficienti e/o potrebbe esserci molto rumore al loro interno, rendendo così difficoltoso creare un buon modello statistico o di machine learning. È qui che entra in gioco la fase di *Features Engineering*.

A. Creazione di nuove Features

La creazione di nuove variabili dai dati che si hanno a disposizione può migliorare drasticamente il potere predittivo degli algoritmi.
Poichè ci si trova in questo caso in un contesto di classi fortemente sbilanciate è di fondamentale importanza costruire delle variabili che siano in grado di creare valore all'interno dei dati, evidenziando dei possibili pattern con la classe minoritaria.
Sulla base delle considerazioni fatte in fase di ***EDA*** e delle informazioni ottenute, si è provveduto a creare delle nuove variabili:

- **not_one_service**: *variabile dicotomica*. Essa assume valore 0 se i clienti sottoscrivono **almeno un abbonamento** tra *FLG_MV; FLG_MYSKYHD ; FLG_HD ; FLG_MYSKY; FLG_SKY_ON_DEMAND*, valore 1 se **non ne sottoscrivono nessuno**;

- **number_of_services**: *variabile numerica*. Per ogni volta che viene osservato un cliente, la variabile restituisce il numero di abbonamenti che lo stesso ha attivi in quel preciso momento;

- **not_one_package**: *variabile dicotomica*. Essa assume valore 0 se i clienti posseggono **almeno un pacchetto** tra *CINEMA; CALCIO ; SPORT ; SKY_FAMIGLIA*, valore 1 se **non ne posseggono alcuno**;

- **number_of_packages**: *variabile numerica*. Per ogni volta che viene osservato un cliente, la variabile restituisce il numero di pacchetti che lo stesso ha attivi in quel preciso momento;

- **is_duplicated**: *variabile dicotomica*. La variabile restituisce valore 0 se il codice identificativo della variabile *external_id* compare solo una volta nel dataset, valore 1 se compare più di una volta. La nuova variabile dummy andrà a sostituirsi alla variabile *external_id*.

Poichè dall'analisi esplorativa è emerso che la variabile ***DATA_RIF*** presentasse una certa stagionalità rispetto al numero di churner per ogni data di rilevazione, si è deciso si traformarla in modo tale da poterla rendere più facilmente gestibile dai modelli statistici e machine learning. Per fare ciò si è optato per la soluzione del ***Label Encoding***.
Poichè le rilevazioni sono state raccolte in soli 17 giorni differenti, è stato possibile attribuire una modalità che andasse da 0 a 16 per ogni data di riferimento.
Vista la stagionalità presente, si è deciso di tenere la variabile adottando questo tipo di soluzione poichè sarebbe potuta risultare molto utile per la classificazione delle osservazioni presenti nel dataset; in circostanze differenti in cui ,ad esempio, si volesse invece prevedere un eventuale churn futuro sulla base delle caratteristiche di un cliente, questo tipo di soluzione risulterebbe inutilizzabile (un'osservazione futura avrebbe una data, e quindi un'etichetta di encoding, che non è presente nel set di training).

4. MODELLING

A. Features Selection con Stability Selection: Un Metodo Robusto

Utilizzare tutte le variabili presenti nel dataset per la previsione della variabile target potrebbe creare del rumore durante la fase di apprendimento dei modelli, andando così ad inficiare negativamente sulle prestazioni degli algoritmi. In contesti in

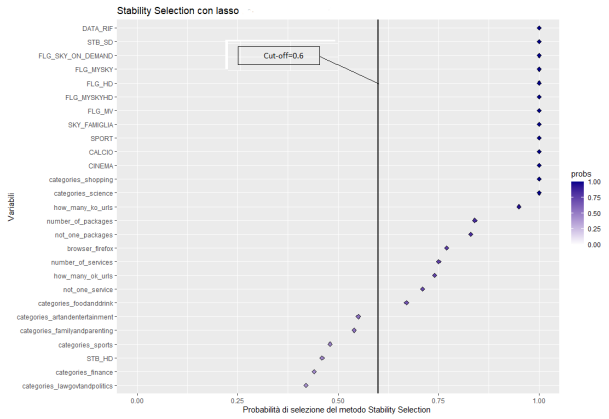

Fig. 3. Il seguente grafico rappresenta la probabilità di selezione del metodo di Stability Selection con lasso; per comodità di visualizzazione si è deciso di zoomare sulle variabili associate al valore di probabilità più elevata.
Il cut-off oltre il quale una variabile è stata considerata importante è stato fissato uguale a 0.6.

cui si ha a che fare con molte covariate è opportuno adoperare delle tecniche di ***Features Selection*** al fine di ridurre la dimensionalità del dataset.
Ipotizzando *sparsità* del segnale nelle covariate, si utilizzano dei metodi che possano selezionare le variabili in cui la presenza di segnale è particolarmente intensa.
Per fare ***Features Selection*** è stato utilizzato un approccio basato su *Lasso*; si è scelto inoltre di affiancare a supporto del lasso uno strumento ***più robusto***: ***"Stability Selection"*** [2]. Applicare il metodo di ***Stability Selection*** al *Lasso* ha permesso di ridurre drasticamente la dimensionalità del dataset (passando da 72 variabili a ***18*** (17 covariate più la variabile risposta *Pdisc*) in cui è stato riscontrato **un segnale molto forte**, come è ben visibile dal grafico in Figura 3, nelle seguenti variabili appartenti al **dataset originale**:
`''DATA_RIF(Label-Encoding)''; ''STB_SD'';`
`''FLG_SKY_ON_DEMAND''; ''FLG_MYSKY''; ''FLG_HD'';`
`''FLG_MYSKYHD''; ''FLG_MV''; ''SKY_FAMIGLIA''; ''SPORT'';`
`''CALCIO''; ''CINEMA''; ''categories_shopping'';`
`''categories_science''; ''how_many_ko_urls''.`
Sono state inoltre selezionate dal metodo le seguenti variabili del dataset originale, pur presentando un segnale minore rispetto a quelle sopra elencate:
`''browser_firefox''; ''how_many_ok_urls'';`
`''categories_foodanddrink''.`

Nel metodo di selezione non è stata inclusa la variabile *External_id*.

Il fatto che alcune variabili del dataset originale abbiano un segnale molto più forte rispetto ad altre può essere molto utile per l'azienda oltre che per la previsione in esame, anche dal punto di vista inferenziale, al fine di dare ulteriore supporto decisionale al management.
Le variabili costruite nella fase di ***Features Engineering*** sono state selezionate tutte quante dal metodo; questo è un buon indicatore di come si sia stati in grado di estrarre valore (ed informazione) dai dati originali.

B. Una Soluzione per le Classi Sbilanciate: SMOTENC

Come evidenziato in precedenza, il dataset in esame è fortemente sbilanciato a sfavore della classe dei *churner*.
I problemi di classificazione fortemente sbilanciati rappresentano una sfida per la modellizzazione predittiva poiché la maggior parte degli algoritmi di Machine Learning utilizzati per la classificazione tendono, per loro natura, a indirizzare le loro previsioni verso la classe maggioritaria. Ciò si traduce in modelli che hanno scarse capacità predittive, in particolare per la classe minoritaria. Questo si rivela essere un grosso problema, poichè la classe minoritaria è, spesso, la classe più importante che si vuole predirre.
Per questo motivo si è provato a bilanciare il dataset utilizzando l'algoritmo `SMOTE-NC` [3]. Si tratta di una tecnica di **sovracampionamento** in grado di generare nuove osservazioni sintentiche sia per variabili nominali che continue (caratteristica particolarmente utile rispetto all'alto numero di dummy presenti nel dataset in esame).
Si è deciso di bilanciare il dataset utilizzando le seguenti proporzioni tra, rispettivamente, classe minoritaria e classe maggioritaria: *20%-80%*, *30%-70%*, *40%-60%*.

C. Anomaly detection

L'anomaly detection è uno strumento che permette di identificare le osservazioni che si discostano dal comportamento della maggior parte di quelle presenti in un set di dati. Dati anomali possono stare ad indicare incidenti critici, come ad esempio un problema tecnico, o potenziali opportunità, come invece ad esempio un cambiamento nel comportamento dei consumatori.
L'ipotesi alla base dell'applicazione di un metodo che andasse ad identificarle, è che ci fosse una qualche relazione tra le osservazioni anomale e quelle labellate come churner.
In questa fase sono state implementate due diverse tecniche nel contesto di anomaly detection:

- `Autoencoder per Outlier Detection`
- `Isolation Forest` [4] [5]

Quest'ultimo metodo si è rivelato particolarmente efficace per identificare i churner. L'algoritmo, infatti, oltre a restituire un risultato puntuale (-1 nel caso di outliers, 1 nel caso di inliers), associa uno score che rappresenta la magnitudine dell'anomalia (nel range (-1, 1)).

In un `Isolation Forest`, i dati sottocampionati casualmente vengono elaborati in una struttura ad albero basata su caratteristiche selezionate casualmente. È meno probabile che i campioni che si trovano nelle foglie più in profondità dell'albero siano anomalie poiché hanno richiesto più tagli per essere isolati. Allo stesso modo, i campioni che finiscono nelle foglie associate ai rami più corti indicano anomalie poiché è stato più facile per l'albero separarli dalle altre osservazioni.
A partire dai risultati restituiti dall'algoritmo si è deciso di valutare la bontà del modello per il task in esame (ovvero valutare se le osservazioni classificate come outliers e inliers coincidono con le osservazioni labellate, rispettivamente, come churner e clienti fedeli), tenendo a mente che Isolation Forest è un metodo non supervisionato che quindi non 'vede' le vere etichette dei dati.

Nella Tabella 1 si osserva come il modello abbia un'accuratezza inferiore rispetto ad un classificatore triviale che classifica tutte le osservazioni come appartenenti alla

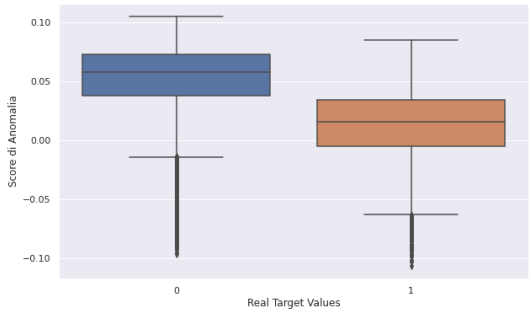

Fig. 4. Il grafico mostra i valori di score risultanti dal modello Isolation Forest condizionatamente alla reale etichetta

classe maggioritaria, tuttavia i valori degli indicatori di BAC e F1-score sono ben superiori rispetto a quelli di un classificatore triviale.

BAC	ACC	P	R	F1
0.629811	0.938068	0.18167	0.301809	0.226813

Table 1. Metriche di riferimento calcolate rispetto agli output di Isolation Forest

Si è deciso successivamente di utilizzare gli score ottenuti in output dal modello come una nuova feature da inserire all'interno del dataset: **"anomalies_score"**. Si è deciso di inserirla poichè, come evidenziato dalla Figura 4, gli score ottenuti sembrano discriminare correttamente le due classi: la scatola blu (valori di score condizionati alla classe dei clienti fedeli) giace al di sopra della scatola arancione (valori di score condizionati alla classe dei clienti churner).

D. Modelli utilizzati

Per la fase di Modelling si è deciso di ricorrere all'utilizzo di modelli e algoritmi in grado di restituire, oltre ad un vettore di previsioni 0-1, un vettore di probabilità (o pseudo-probabilità) che indica la possibilità che un cliente sia un churner. Inoltre si sono preferiti modelli computazionalmente efficienti, in grado di restituire una previsione in tempi brevi.
Di seguito i modelli utilizzati:

- `CatBoost` [6]: *o Categorical Boosting* è una libreria open source generalmente utilizzata per risolvere problemi di classificazione e regressione e ottimizzata per dati eterogenei. La particolarità di questo algoritmo risiede nel fatto che performa particolarmente bene per dataset contenenti un elevato numero di variabili categoriali.
- `LightGBM` [7]: è, anch'esso, un algoritmo basato su gradient boosting, con la caratteristica di essere particolarmente rapido da un punto di vista computazionale. La particolarità di questo modello, a differenza degli altri algoritmi basati su alberi decisionali, risiede nel fatto di costruire l'albero verticalmente, anzichè orizzontalmente. In sostanza gli *splits* avvengono nella foglia di ciascun albero e non nei nodi intermedi. Si tratta di un algoritmo che, così come è stato fatto per il precedente, è stato scelto perchè particolarmente performante su una grande varietà di tasks.
- `Rete Bayesiana Feed Forward` [8]: Si tratta di una semplice rete neurale Feed-Forward in cui i pesi, anzichè essere identificati da un valore puntuale, vengono modellizzati tramite una distribuzione di probabilità. Si è scelto di utilizzare una rete bayesiana poichè queste sono meno soggette a overfitting (modellare i pesi pone dei vincoli).
- `Extra-Trees Random Forest`: Rappresenta una versione randomizzata di Random Forest.
- `Naive Bayes Classifier`

5. RISULTATI

In questa sezione vengono riassunte le performance ottenute a partire dai modelli individuati precedentemente allenati a partire da dataset differenti.
I modelli sono stati allenati sul 70% delle osservazioni presenti nel dataset (training set) e le previsioni sono state invece effettuate sulla rimanente parte. Si è deciso, data la natura fortemente sbilanciata delle classi, di utilizzare un criterio di splitting stratificato (*holdout stratificato 70%-30%*). Per ciascuna tabella si è deciso di valutare le performance degli algoritmi rispetto a metriche differenti (Accuracy, Balanced-Accuracy, F_1 score, Precision, Recall).
Per avere una *baseline* con cui confrontarsi, si è preso come riferimento inizialmente un classificatore *triviale*, ovvero un modello che restituisce come risultati un vettore di elementi appartenenti unicamente alla classe maggioritaria, che nel caso in esame ha un'accuratezza del 96,99% (ma che ha un'accuratezza bilanciata del 50%).

Nella Tabella 2 si fa riferimento alle performance dei modelli selezionati, **senza che sul dataset sia stata effettuata alcuna selezione di variabili.**
Si osserva come i modelli che meglio performano, rispetto alle diverse tipologie di metriche, sono quelli evidenziati in giallo. Mentre Catboost overperforma rispetto agli altri modelli in termini di Accuracy e F1, Il classificatore di Bayes Naive, invece, è quello che riesce a classificare meglio le osservazioni appartenenti alla classe minoritaria (la sensitività è estremamente più alta rispetto a quella degli altri modelli), a discapito però di un'accuratezza sensibilmente inferiore.
In Figura 5 si osserva il grafico delle curve ROC in corrispondenza dei modelli utilizzati (ad eccezione delle reti Bayesiane, che risulta essere il modello che performa peggio).

Nella Tabella 3 si osservano le performance dei diversi modelli allenati sul dataset dimensionalmente ridotto (in cui è stata effettuata **Feature Selection**).
I modelli che hanno raggiunto le performance migliori sono i medesimi considerati sul dataset non ridotto. Si può osservare però come, una volta ridotto il numero delle variabili, i modelli performino meglio sul test set considerato.
Da questo fatto si può evincere che la maggior parte delle features contribuiscono ad introdurre rumore nella fase di apprendimento dei modelli. In Figura 6 si osservano le curve di ROC in corrispondenza dei modelli utilizzati sul dataset dimensionalmente ridotto. Si osserva come l'Area Sotto la Curva

Metrics	CatBoost	lightGBM	Extra-Trees	Naive Bayes	Bayesian NN
BAC	0.593	0.586	0.559	0.634	0.550
Accuracy	0.974	0.973	0.972	0.925	0.972
Precision	0.794	0.740	0.778	0.152	0.781
Recall	0.187	0.174	0.119	0.323	0.101
f1	0.303	0.282	0.206	0.207	0.178

Table 2. Metriche calcolate rispetto ai diversi Modelli allenati sul dataset con tutte le Features

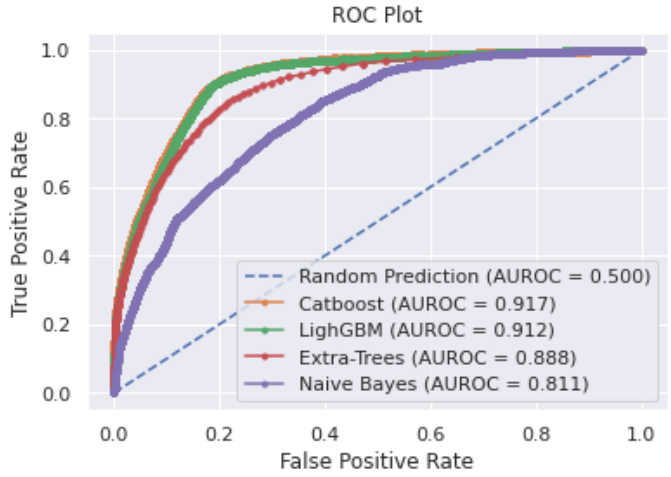

Fig. 5. Il grafico mostra la curva di ROC risultante dai modelli allenati sul dataset senza Feature Selection

Metrics	CatBoost	lightGBM	Extra-Trees	Naive Bayes
BAC	0.597	0.595	0.598	0.673
Accuracy	0.974	0.974	0.969	0.924
Precision	0.735	0.741	0.464	0.174
Recall	0.196	0.192	0.203	0.406
f1	0.310	0.305	0.283	0.243

Table 3. Metriche calcolate rispetto ai diversi Modelli allenati sul dataset in cui sono state selezionate un numero ridotto di variabili

(AUC) sia più basso su questi modelli rispetto ai medesimi con tutte le Features.

Nelle Tabelle 4 e 5 vengono riassunte le performance del classificatore `CatBoost`, quello che meglio ha performato fino ad ora, in presenza di traning set (rispettivamente *senza* e *con* Feature Selection) in cui è stato effettuato ricampionamento (utilizzando le proporzioni tra osservazione nella classe minoritaria e maggioritaria definite in tabella).
Come si osserva dai risultati, aumentando il numero delle osservazioni nella classe minoritaria i valori di Sensitivity aumentano, ma, inevitabilmente, peggiorano le performance di accuratezza globali dell'algoritmo. La scelta di utilizzare ricampionamento avviene, dunque, ad un costo: il modello è in grado di discriminare meglio la classe minoritaria (viene dato focus maggiore su questa poichè il numero di osservazioni aumentano), ma, al tempo stesso, l'algoritmo prevede peggio la classe maggioritaria; in ottica imprenditoriale il prezzo pagato con l'utilizzo del ricampionamento, in questo caso specifico, può risultare molto alto in quanto riconoscere come churners clienti che in realtà non lo sono potrebbe portare le aziende a commettere errori nella pianificazione delle promozioni, andando così ad offrire "più di quanto necessario" a alcuni clienti che in realtà non hanno alcuna intenzione di abbandonare la compagnia.

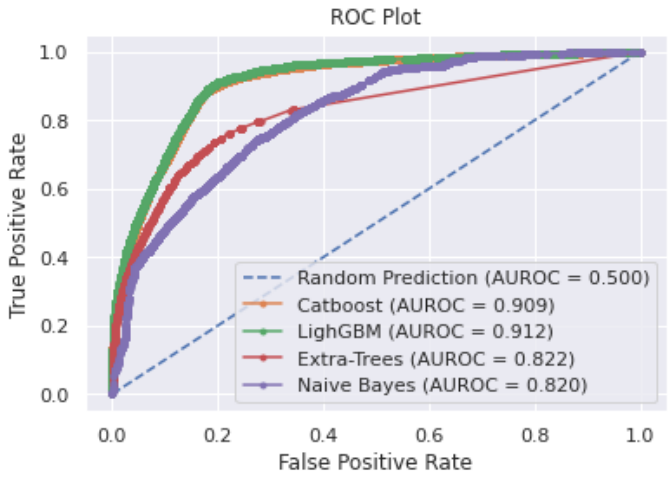

Fig. 6. Il grafico mostra la curva di ROC risultante dai modelli allenati sul dataset con Feature Selection

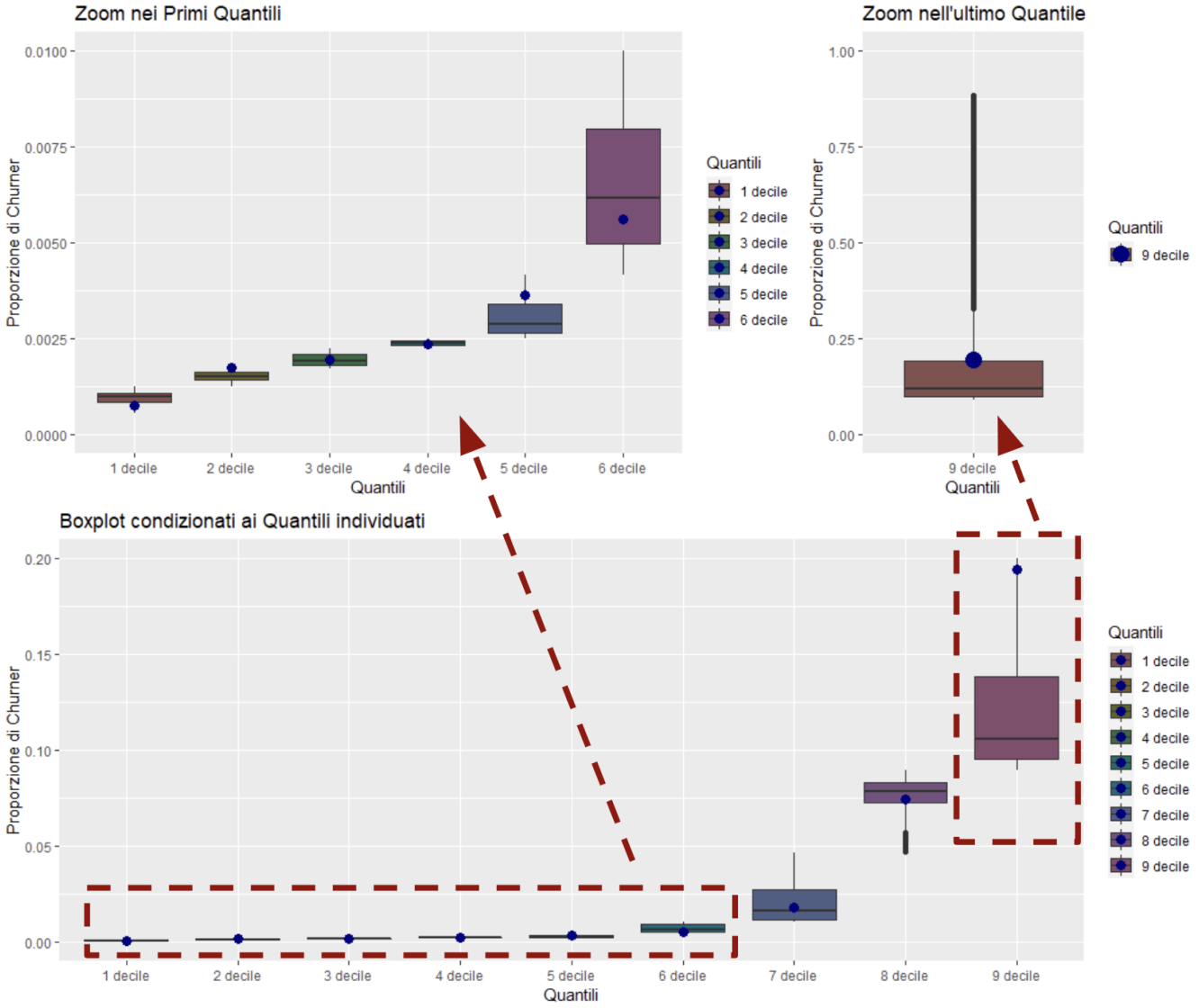

Fig. 7. In basso i boxplot delle probabilità condizionati ai quantili individuati. Per ragioni di visualizzazione si è deciso di zoommare i primi (in alto a sinistra) e l'ultimo (in alto a destra). I punti blu rappresentano la proporzione di churner per ogni decile.

Metrics	20-80%	30-70%	40%-60%
BAC	0.605	0.612	0.618
Accuracy	0.965	0.963	0.960
Precision	0.384	0.335	0.308
Recall	0.220	0.240	0.254
f1	0.280	0.280	0.278

Table 4. Performance del classificatore CatBoost allenato sul dataset ricampionato tramite l'utilizzo di SMOTE-NC sul dataset dimensionalmente non ridotto. In alto si osservano le proporzioni tra classe minoritaria e maggioritaria.

Metrics	20-80%	30-70%	40%-60%
BAC	0.667	0.698	0.720
Accuracy	0.954	0.939	0.924
Precision	0.294	0.232	0.200
Recall	0.361	0.441	0.502
f1	0.324	0.304	0.286

Table 5. Performance del classificatore CatBoost allenato sul dataset ricampionato tramite l'utilizzo di SMOTE-NC sul dataset dimensionalmente ridotto. In alto si osservano le proporzioni tra classe minoritaria e maggioritaria

6. METODO DEI QUANTILI

Molto spesso approcci predittivi *"puri"*, finalizzati alla massimizzazione delle performance, e le comuni metriche, possono non bastare per la risoluzione e la valutazione degli obiettivi prefissati ad inizio analisi. In casi simili a quello in esame, in cui non solo ci si trova in presenza di classi fortemente sbilanciate, ma, inoltre, le covariate ad esse associate presentano per lo più un segnale molto debole, si è cercata una soluzione alternativa per valutare le performance del singolo classificatore.

Sulla base di ciò si è deciso di utilizzare ***il metodo dei quantili***: l'idea alla base di questo approccio è stata quella di dividere la probabilità delle previsioni restituite dal classificatore **CatBoost** in decili e, per ognuno di questi, è stata indicata la rispettiva reale frazione di churners. Infine si è andati a valutare due parametri: il primo è che la percentuale di churners di ogni decile sia progressivamente maggiore rispetto a quella del decile precedente (quindi che ci sia andamento crescente); il secondo parametro è che la proporzione di churners ricada nel range di probabilità fissato nel decile, al fine di valutare se ci sia coerenza tra le previsioni e i dati originali.

Nel caso in esame, guardando i boxplot condizionati ai decili delle probabilità di previsione, si può facilmente evincere che questi parametri vengono soddisfatti entrambi: si può notare un andamento chiaramente crescente e i punti blu (che stanno ad indicare la proporzione di churners per ogni decile) appartengono al range di probabilità fissato (vedi Figura 7).

7. CONCLUSIONI

Complessivamente si può notare che le prestazioni dei modelli che utilizzano il dataset originale siano inferiori a quelle dei modelli che utilizzano quello ridotto in cui è stata fatta *Features Selection*. Si può appurare dai risùltati dei modelli sul dataset ridotto che la fase di preprocessing dei dati sia stata svolta in maniera congrua con quanto evidenziato durante l'EDA e che i dati siano stati parzialmente ripuliti dal rumore. Le nuove variabili ingegnerizzate, infatti, hanno permesso di estrapolare nuovo segnale dal dataset creando nuovi pattern sui quali i modelli sono stati in grado di apprendere migliorandosi. Gli approcci *tree-based* si sono rivelati particolarmente efficaci per la previsione del comportamento dei clienti.

Per le analisi sono stati utilizzati principalemente quattro modelli (selezionati sulla base delle performance ottenute sul test set). I modelli basati su alberi decisionali hanno restituito, come ampiamente analizzato nella precedente sezione, delle performance molto simili tra di loro, ma diverse rispetto al classificatore di Bayes. Il modello finale selezionato, ***CatBoost***, è quello che ha ottenuto i migliori risultati in termini di Accuracy, score F_1 e AUC, ed è risultato in grado di restituire previsioni coerenti con gli obiettivi prefissati ad inizio analisi, soprattutto vista la complessità del problema di classi fortemente sbilanciate e di scarsa correlazione tra covariate e variabile risposta. La previsione in senso stretto e l'identificazione di quelle che sono le caratteristiche principali di un cliente che cessa l'acquisto di beni e/o servizi, possono permettere al management di creare delle strategie aziendali specifiche di customer retention, personalizzabili per clienti singoli o cluster di clienti.
Un'applicazione possbile di quanto trattato nel suddetto studio, potrebbe essere quella di offrire tramite i canali di CRM aziendale delle soluzioni esclusive ad ex clienti o a chi potrebbe

manifestare la volontà di abbandonare l'acquisto di ogni servizio presso l'azienda.

REFERENCES

1. Model-HubSpot, "The ultimate list of marketing statistics for 2022 (source: link)," Innovation **15**, 3.
2. N. Meinshausen and P. Bühlmann, "Stability selection," J. Royal Stat. Soc. Ser. B (Statistical Methodol. **72**, 417–473 (2010).
3. M. Mukherjee and M. Khushi, "Smote-enc: A novel smote-based method to generate synthetic data for nominal and continuous features," Appl. Syst. Innov. **4**, 18 (2021).
4. F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining,* (IEEE, 2008), pp. 413–422.
5. H. John and S. Naaz, "Credit card fraud detection using local outlier factor and isolation forest," Int. J. Comput. Sci. Eng **7**, 1060–1064 (2019).
6. L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," Adv. neural information processing systems **31** (2018).
7. G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," Adv. neural information processing systems **30** (2017).
8. D. T. Chang, "Probabilistic deep learning with probabilistic neural networks and deep probabilistic models," arXiv preprint arXiv:2106.00120 (2021).