Polizze assicurative per caravan: studio su potenziali clienti

Daniele Ceccarelli (Mat. 864239), Irene Lupino (Mat. 866921), Angelica Barrion (Mat. 790693)

Sommario

È possibile individuare i potenziali acquirenti di polizze assicurative per roulotte?

Abbiamo tentato di rispondere a questa domanda analizzando un dataset che contiene informazioni sui clienti di una compagnia di assicurazione.

Attraverso uno studio di classificazione supervisionata si è cercato di comprendere se determinate variabili possono aiutarci a predire questo genere di acquisti. Il report presenta e compara varie tecniche di classificazione individuando quella migliore.

[1]*Università degli Studi di Milano Bicocca, CdLM Data Science*
[2]*Università degli Studi di Milano Bicocca, CdLM Data Science*
[3]*Università degli Studi di Milano Bicocca, CdLM Data Science*

Sommario

Introduzione

In un mercato in cui la concorrenza è spietata, l'abilità di valutare velocemente il possibile acquirente di una polizza, proponendo un prodotto in linea con le sue caratteristiche, è un fattore assai rilevante. Infatti, questo potrebbe permettere all'agenzia assicurativa di attivare offerte promozionali mirate, riservate ad un gruppo ristretto di potenziali clienti, evitando quindi l'invio di comunicazioni indiscriminate.

Inoltre, il saper riconoscere, all'interno del proprio parco clienti i possibili acquirenti di questa tipologia di polizza, potrebbe evitare l'eventuale migrazione della stessa clientela verso altre assicurazioni.

Caravan Insurance Challenge è il dataset utilizzato per lo studio.

Per lo svolgimento dello studio è stato utilizzata la piattaforma Knime Versione 3.3.1.

Dataset e preprocessing

Dataset

Il set utilizzato è costituito da 87 variabili che includono dati sull'utilizzo del prodotto e dati sociodemografici, derivati dai codici di avviamento postale.

Ogni osservazione corrisponde ad un codice postale. Le variabili che iniziano con M si riferiscono alle statistiche demografiche del codice postale, mentre le variabili che iniziano con P e A si riferiscono alla proprietà del prodotto e alle statistiche assicurative per codice postale.
Per una descrizione più accurata delle variabili, e delle condizioni applicate per costruire il dataset, si rimanda alla piattaforma Kaggle[1].

La variabile target dell'analisi è *CARAVAN.* Nello specifico si considera come 0 (zero) un *non cliente*, ovvero chi non acquista la polizza assicurativa, e 1 (uno) un *cliente*, ovvero chi acquista la polizza.

L'obiettivo dell'analisi è prevedere il possibile cliente, cioè quando la variabile CARAVAN = 1.

Preprocessing

Da una prima analisi esplorativa dei dati non sono risultati valori mancanti.

È stato trovato un solo outlier che per l'attributo number_car_policies ha un valore pari a 12 fuori dal range [0,11], intervallo di valori ammissibili per ogni variabile.

Dato che il dataset contiene 87 attributi si è deciso di eliminare a priori le variabili considerate ridondanti o irrilevanti.

Il dataset è stato originariamente utilizzato per una challenge di datamining; la variabile ORIGIN, contenente i valori train e test, corrispondono rispettivamente ai set di addestramento e valutazione, è stata eliminata perchè non utile ai fini dell'analisi. Successivamente è stata analizzata la correlazione che intercorre tra le variabili. Infatti, l'utilizzo di un numero elevato di variabili rischia di minare l'efficienza del modello a livello statistico. In particolare:

- il modello risulta essere troppo complesso
- vi è difficoltà ad interpretare i dati
- le stime dei parametri possono risultare instabili
- più parametri verranno inseriti, più osservazioni sono necessarie per stimarli

Dall'analisi è risultato che le variabili relative alle proprietà del prodotto assicurativo, in particolare le variabili riguardo i *contributions* e *numbers* sono correlate. Rappresentano, infatti, gli stessi dati: le variabili contributions rappresentano le percentuali in ogni gruppo di clienti per codice postale mentre numbers sono il numero totale delle variabili.

Ad esempio:

> *Contribution car policies* e *number of car polilcies*

sono entrambe variabili relative al numero di polizze assicurative stipulate per auto, una in percentuale e l'altra in numero.

Sono quindi variabili ridondanti. Per questo motivo, e dato che la variabile CARAVAN oggetto di studio è data solo come number, tutte le colonne relative ai contributi assicurativi sono stati esclusi dall'analisi.

Tra gli attributi sono stati individuati Dummy Traps[2] che causano multicollinearità e influenzano negativamente i modelli di machine learning. Sono state quindi eliminate le colonne di seguito elencate, perché considerate Dummy Trap e quindi altamente correlate:

- Customer_main_type
- Roman_catholic
- Living_together
- Singles
- High_level_education
- Entrepreneur
- Social_class_D
- Rented_house
- No_car
- National_Health_Service
- Income_<30000
- Income_30-45000
- Income_45-75000

- Income_75-122000
- Income_>123000

Modelli

Al fine di individuare la tecnica più adatta per lo studio in oggetto, sono stati impiegati diversi modelli di classificazione:

- Modelli euristici: albero di regressione J48, implementato da Weka, Random Forest. Anche se non garantiscono risultati ottimali, permettono di ottenere soluzioni approssimate e ragionevoli.
- Modelli di separazione: Sequential Minimal Optimization utilizzando Polykernel; Multilayer Perceptron, come modello di reti neurali;
- Modelli probabilistici: Naive Bayes (basato sul teorema di Bayes); NBTree che genera un albero decisionale attraverso il classificatore Naive Bayes; A1de.
- Modelli di regressione: Regressione logistica.

Criteri di valutazione

Nell'analisi, per valutare la performance dei vari modelli, sono stati utilizzati i seguenti criteri:

- Accuracy
- Recall
- Precision
- F-measure
- Area Under Curve (AUC) della curva Receiver Operating Characteristic (ROC)

Accuracy

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

Questa misura equivale al rapporto tra il numero di righe classificate correttamente e il numero totale di righe. Attraverso di essa viene valutata l'accuratezza della classificazione.

Con TP e TN vengono indicati i *true positive* e i *true negative*, ovvero le istanze classificate correttamente. Con FP e FN, (rispettivamente *false positive* e *false negative),* il numero di istanze positive e negative classificate erroneamente.

Recall

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

Recall indica la porzione di record positivi correttamente classificati dal modello. Un alto valore di Recall indica una bassa percentuale di record positivi classificati in modo errato.

Precision

$$Precision = \frac{TP}{TP + FP}$$

È il rapporto tra il numero di casi correttamente classificati come positivi e tutti i casi valutati come positivi (includendo anche i FP, ovvero quelli che in realtà sono negativi). L'indicatore Precision è una misura di specificità e indica la probabilità che la classe assegnata si riveli corretta, ignorando se, e quante volte, quella classe non sia stata rivelata come tale.

Un valore alto di Precision determina un numero minore di falsi positivi.

F-measure

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall}$$

Vi è la possibilità che Precision e Recall entrino in conflitto, in quanto è possibile che si costruiscano dei modelli che massimizzino unicamente uno dei due indicatori, registrando, di conseguenza, un contrasto nei risultati. La F-measure, media armonica tra Recall e Precision, consente di fornire un'interpretazione dei due indici più ragionevole.

ROC e AUC

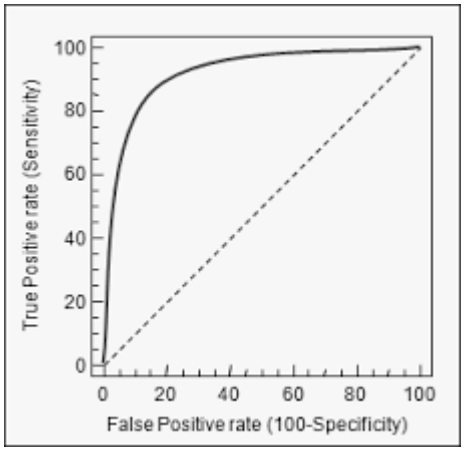

Figura 1 – Example ROC curve from medcalc.org[3]

La curva ROC (Figura 1) consente di rappresentare, sull'asse delle ordinate, la percentuale del numero totale di TP e, sull'asse delle ascisse, la percentuale di FP.

L'area sottesa alla curva di ROC, detta AUC, *Area Under Curve,* assume un ruolo centrale come misura di performance, in quanto consente di definire la qualità del classificatore.

Dataset sbilanciato

Il dataset utilizzato è "unbalanced" in quanto presenta di 9235 righe classificate come "0" e solo 586 classificate come "1" (si veda tabella 1).

Tabella 1 – distribuzione variabile Caravan

Row ID	count
0	9235
1	586

Questo squilibrio può interferire negativamente con le capacità di generalizzazione che un classificatore acquisisce in fase di addestramento.

Nello specifico, il classificatore potrebbe concentrarsi sulla classe maggioritaria dal momento che la maggior parte dei campioni, utilizzati durante l'addestramento, proviene da quella classe. In sostanza, si potrebbe raggiungere un'accuratezza pari al 99% etichettando tutti i casi come *non cliente* (caravan = 0) dando, solo apparentemente, un ottimo risultato. In un contesto del genere, quindi, non solo la stima di un modello di classificazione risulta problematica, ma lo è anche la valutazione della sua accuratezza.

Per affrontare questo problema, è possibile utilizzare differenti approcci:

- Tecniche di Cost-Sensitive Learning, che si basano sull'assegnazione di un costo per errata classificazione. L'obiettivo dell'apprendimento tramite Cost-Sensitive è quindi minimizzare i costi di errata classificazione.
- Tecniche di campionamento, tra cui distinguiamo:
  - undersampling (sotto-campionamento): tecnica che agisce sulla classe prevalente ridimensionandola e creando un suo sottoinsieme.
  - oversampling (sovra-campionamento): tecnica che mira a bilanciare la distribuzione generando nuovi dati a partire dalla classe minoritaria.

Nell'analisi sono stati utilizzati entrambi gli approcci sopra descritti, prediligendo la tecnica Oversampling.

Analisi dataset sbilanciato

Prima di procedere, alla valutazione delle performance dei vari classificatori, è stata eseguita una verifica dei modelli applicati al dataset sbilanciato.

Tabella 2 – Analisi dataset sbilanciato

Row ID	Recall	Precision	F-meas...	Accuracy	AUC
multi_layer	0.026	0.135	0.043	0.932	0.62
A1DE	0.13	0.24	0.168	0.924	0.694
smo_polykernel	0	?	?	0.94	0.5
logistic	0.016	0.3	0.03	0.939	0.697
j48	0.005	0.2	0.01	0.94	0.5
nbtree	0.01	0.25	0.02	0.939	0.635
random_forest	0.031	0.076	0.044	0.92	0.613
naivebayes	0.363	0.117	0.177	0.799	0.665

Esaminando la tabella 2 si nota che

- una Recall tendenzialmente bassa, se confrontata con la Precision (sinonimo di una tendenza da parte dei modelli a classificare maggiormente secondo la classe prevalente).
- Inoltre, valori di F-Measure sono di gran lunga inferiori rispetto ai valori di

Accuracy (probabilmente dovuto al forte sbilanciamento).

- Per quanto riguarda la curva di AUC, i valori ottenuti sono tutti vicini e in alcuni casi (smo_polykernel e j48) pari a 0.5, questo indica che i modelli non sono in grado di distinguere correttamente tra classi positive e classi negative dando così previsioni errate. I modelli A1de, logistic e naivebayes si avvicinano ad un 70% di possibilità di poter distinguere tra classi positive e negative.

Con questa prima analisi si conferma la necessità di bilanciare il dataset.

Cost-sensitive Learning

Questa tecnica permette, attraverso matrici di costo, di individuare il possibile costo che si potrebbe sostenere a causa di classificazioni corrette o errate.
Abbiamo utilizzato due matrici:

Tabella 3 - Matrice di costo 1

Cost Matrix 1	
0	1
5	-1

Tabella 4 - Matrice di costo 2

Cost Matrix 2	
0	1
20	-1

In entrambe le matrici (tabella 3, tabella 4) è stato assegnato un peso maggiore ai falsi negativi, in quanto si assume che la mala classificazione di un positivo (identificato quindi come un negativo), porti alla perdita di un potenziale cliente e quindi ad un mancato guadagno.

Di seguito i risultati ottenuti dalle due matrici:

Tabella 5 – Risultati analisi Matrice di costo 1

Row ID	Recall	Precision	F-meas...	Accuracy	AUC
multi_layer	0.187	0.122	0.148	0.872	0.551
A1DE	0.352	0.133	0.193	0.825	0.603
smo_polykernel	0	?	?	0.94	0.5
logistic	0.244	0.156	0.19	0.876	0.58
j48	0.005	0.2	0.01	0.94	0.502
nbtree	0.052	0.139	0.075	0.924	0.516
random_forest	0.269	0.113	0.159	0.83	0.568
naivebayes	0.472	0.099	0.164	0.713	0.6

Tabella 6 – Analisi Matrice di costo 2

Row ID	Recall	Precision	F-meas...	Accuracy	AUC
multi_layer	0.285	0.111	0.159	0.821	0.57
A1DE	0.596	0.102	0.174	0.664	0.632
smo_polykernel	0	?	?	0.94	0.5
logistic	0.694	0.094	0.165	0.582	0.635
j48	0.995	0.059	0.112	0.061	0.498
nbtree	0.326	0.096	0.148	0.777	0.566
random_forest	0.472	0.095	0.159	0.702	0.594
naivebayes	0.591	0.094	0.163	0.638	0.616

L'applicazione della *Matrice di costo 1* restituisce valori per lo più inferiori rispetto alla *Matrice di costo 2*, la quale penalizza maggiormente i falsi negativi. In particolare, esaminando i dati relativi all'applicazione della seconda matrice, la Recall (per tutti i classificatori) presenta valori più alti. Assumendo che assegnare un costo maggiore ad un FN, abbia portato i modelli a modificare i perimetri con i quali vengono valutati i positivi è possibile notare come la Precision, per la *Matrice di costo 2*, presenti valori inferiori rispetto alla prima.
I valori di AUC si avvicinano allo 0.5 dando quindi evidenza che applicare una matrice di costo non sembri essere una soluzione ottimale per migliorare i modelli del dataset in studio.

Tecniche di campionamento

Entrambe le tecniche precedentemente citate (undersampling e oversampling), presentano pro e contro.

Il dataset contiene 9821 osservazioni. Date le dimensioni del dataset e considerata la bassa percentuale di record identificati come 1 (ovvero la classe *cliente*), si è deciso di utilizzare la tecnica oversampling, in particolare Smote. Sovra-campionando i dati di input in modo casuale, si bilancia la classe attiva vs la classe inattiva.
Abbiamo escluso il metodo undersampling per evitare il rischio di scartare dati potenzialmente utili per il processo di apprendimento. Nello specifico, il dataset iniziale si sarebbe ridotto di oltre la metà.

Classificazione con metodo Holdout

Dopo aver diviso il dataset iniziale in test set e training set, è stato applicato solo a quest'ultimo, il nodo Smote ottenendo un dataset di training bilanciato (si veda Tabella 7).

Tabella 7 – Distribuzione var. Caravan dopo Oversampling

Row ID	count
0	6187
1	6187

Successivamente abbiamo addestrato i modelli con il nuovo training set ottenendo i seguenti risultati:

Tabella 8 – Analisi Modelli con Oversampling

Row ID	Recall	Precision	F-meas...	Accuracy	AUC
multi_layer	0.523	0.098	0.166	0.686	0.643
A1DE	0.135	0.094	0.111	0.871	0.605
smo_polykernel	0.606	0.118	0.198	0.707	0.66
logistic	0.585	0.114	0.191	0.705	0.691
j48	0.197	0.12	0.149	0.866	0.565
nbtree	0.176	0.11	0.136	0.866	0.563
random_forest	0.14	0.134	0.137	0.895	0.62
naivebayes	0.642	0.092	0.16	0.599	0.661

Rispetto alla *Matrice di costi 2* si osserva dalla Tabella 8 che:

- i valori di Recall, ottenuti con smote, sono per lo più inferiori, ad esclusione di Multi_layer, smo_polykernel e naivebayes che registrano, al contrario, un miglioramento.
- l'indice AUC, applicando Smote, ottiene valori superiori per tutti i modelli, ad esclusione di Nbtree,
- Il rapporto tra F-measure e Accuracy risulta invece ancora elevato.

Utilizzare un oversampling per superare lo sbilanciamento del dataset sembra essere un approccio migliore.

Feature Selection e Smote

Fino a questo punto sono stati esclusi dal dataset solo gli attributi ridondanti e irrilevanti. Si procede accostando alla tecnica di oversampling, la Feature Selection con filtro multivariato Cfs-SubsetEval. Sono stati individuati come più significativi:

- Number_of_houses
- Avg_age
- Protestant
- Other_religion
- Married
- Household_without_children
- Medium_level_education
- Farmer
- Skilled_labourers
- Unskilled_labourers
- Social_class_B1
- Social_class_B2
- Home_owners
- 1_car
- 2_cars
- Average_income
- Purchasing_power_class
- Number_of_private_third_party_insurance
- Number_of_car_policies
- Number_of_motorcycle_scooter_policies
- Number_of_tractor_policies
- Number_of_moped_policies
- Number_of_life_insurances
- Number_of_fire_policies
- Number_of_boat_policies
- Number_of_bicycle_policies
- Number_of_social_security_insurance_policies

Nella tabella 9 i risultati ottenuti:

Tabella 9 – Analisi Modelli con Feature Selection e Smote

Row ID	Recall	Precision	F-meas...	Accuracy	AUC
multi_layer	0.456	0.106	0.172	0.739	0.648
A1DE	0.088	0.131	0.105	0.911	0.634
smo_polykernel	0.549	0.108	0.18	0.702	0.63
logistic	0.549	0.106	0.178	0.697	0.676
j48	0.161	0.11	0.13	0.872	0.504
nbtree	0	?	?	0.94	0.654
random_forest	0.098	0.119	0.108	0.903	0.63
naivebayes	0.451	0.1	0.163	0.725	0.648

Vi è un abbassamento dell'indice di Recall su tutti i classificatori rispetto ai risultati ottenuti con il metodo Holdout. Al contrario, la Precision registra valori più alti (al netto di Nbtree). La difficoltà dei modelli di identificare la classe rara è confermata anche da Accuracy, che presenta infatti valori più alti rispetto al metodo Holdout, ad esclusione del classificatore Logistic.
La Feature Selection non sembra apportare significativi miglioramenti ai modelli, probabilmente sono stati eliminati anche attributi utili ai fini della classificazione.
Computazionalmente parlando, si riscontra, invece, un incremento della velocità.

Cross Validation

Per migliorare i risultati ottenuti, si è scelto di ricorrere all'uso del 10 folds cross validation. Questa tecnica prevede la suddivisione del dataset in dieci partizioni tendenzialmente di uguale misura. Ogni partizione viene utilizzata nove volte come training set e una volta come test set. Le medie dei risultati, ottenuti ad ogni iterazione, forniscono gli indici di performance dei classificatori.

Di seguito le performance nella tabella 10.

Tabella 10 – Analisi Modelli con Cross-Validation

Row ID	Recall	Precision	F-meas...	Accuracy	AUC
multi_layer	0.473	0.127	0.2	0.774	0.691
A1DE	0.121	0.145	0.132	0.905	0.678
smo_polykernel	0.594	0.12	0.2	0.716	0.659
logistic	0.597	0.116	0.194	0.704	0.717
j48	0.205	0.131	0.16	0.872	0.535
nbtree	0	?	?	0.94	0.695
random_forest	0.14	0.156	0.148	0.904	0.654
naivebayes	0.536	0.1	0.169	0.686	0.667

Notiamo che per tutti i classificatori vi è un aumento delle performance, in particolare degli indici di Recall, Precision e F-Measure.

Dai risultati ottenuti attraverso la cross validation, identifichiamo i seguenti quattro modelli come utili al fine che ci eravamo posti:

- Logistic,
- smo Polikernel,
- multi_layer
- Naive Bayes.

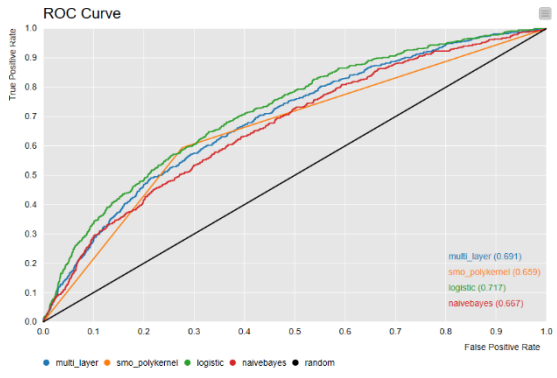

Figura 2 – ROC curve dei modelli ottenuti con cross-validation

Questi modelli infatti hanno un valore AUC che si avvicina a 0.71 quindi con una buona probabilità che il modello possa riconoscere correttamente le classi positive e negative.

Conclusioni

Lo scopo principale era individuare i potenziali acquirenti di polizze assicurative per roulotte.

Per fare questo, dal dataset sono stati eliminati attributi ritenuti ridondanti e irrilevanti. Successivamente è stata fatta un'analisi preliminare senza bilanciare il dataset.

Visti i dati ottenuti, per ovviare allo sbilanciamento, sono state applicate due differenti tecniche, CostSensitive Learning e Oversampling. Quest'ultima ha restituito i risultati migliori. In particolare, ad esclusione di Nbtree, l'indice AUC ottiene valori più elevati, a differenza di Recall per cui l'indice è maggiore solo per i modelli Multi_layer, smo_polykernel e naivebayes.

Rispetto all'analisi iniziale (tab.1 vs tab. 10), si può concludere che, attraverso le varie tecniche utilizzate, quasi tutti i classificatori hanno migliorato i loro indici, in particolare il classificatore multi_layer. L'unico su si riscontra un abbassamento della performance è il classificatore A1de; mentre naive_bayes rimane sostanzialmente invariato.

Tra i modelli utili al nostro studio il modello Logistic risulta il migliore con AUC = 0.71.

Nonostante l'incremento ottenuto dai modelli, c'è ancora l'ipotesi che sia possibile perfezionare i risultati attraverso una diversa Feature Selection e successiva cross validation. Non si esclude, inoltre che, l'applicazione di differenti matrici di costo, possa portare a risultati migliori.

Referenze

[1] Knime, Caravan Insurance dataset
www.kaggle.com/uciml/caravan-insurance-challenge

[2] Kaggle, What is a Dummy Trap and How to avoid it
https://www.kaggle.com/getting-started/149280

[3] Medalc, ROC curve
https://www.medcalc.org/manual/roc-curves.php