Master's Degree in Data Science

Data Science Lab

***Della Libera Davide** (901638), **Longoni Letizia** (844552)*

Una campagna pubblicitaria guidata dai dati

Indice

Introduzione ed obiettivi

In un mondo sempre più informatizzato e digitale, la *pubblicità online* è diventata uno strumento cruciale per il successo delle aziende e delle loro strategie di business. A tal proposito, il reperimento dei dati di navigazione degli utenti (attraverso appositi sistemi di tracking delle attività dei cookie in rete), combinato con *analisi statistiche* ed approcci di *Intelligenza Artificiale*, permette di massimizzare l'efficacia delle campagne pubblicitarie e di migliorare la ricerca del target di persone a cui sottomettere gli annunci.

Per questo progetto, s'ipotizza una situazione reale nella quale un'azienda richiede di costruire una *strategia di online advertising* basata sui dati rilevati dalle interazioni degli utenti con le pagine web contenenti un certo banner pubblicitario, con lo scopo di ottimizzare la campagna e, di conseguenza, minimizzare i costi inerenti all'inserimento dei messaggi promozionali digitali che reindirizzano al sito internet della compagnia (con i relativi prodotti o servizi offerti).

All'interno del report verrà descritta e definita una metodologia che, partendo da un insieme di dati rilevati sui cookie, punta ad *identificare il target ideale* al quale inviare (o proporre nuovamente) l'advertising, ovvero coloro i quali avranno una probabilità maggiore di cliccare il banner e, auspicabilmente, in seguito acquistare degli articoli offerti dell'azienda.

L'obiettivo è, quindi, quello di studiare un metodo ed implementare uno strumento (riusabile ed adattabile agli specifici contesti) in grado di:

- individuare un insieme di utenti ideali ai quali sottomettere (o risottomettere) la pubblicità;
- predire quali cookie saranno potenziali *"clicker"*, ovvero che cliccheranno con maggiore probabilità sul banner, differenziandoli dagli opposti *"non clicker"*.

In conclusione, verrà proposta una possibile strategia generica per la gestione di una campagna pubblicitaria online basata su dati, tecniche di machine learning ed automazione delle decisioni.

Il dataset di partenza

Il primo aspetto da considerare riguarda i *dati reperiti dal sistema di tracking* in fase di navigazione ed interazione degli utenti (cookie) all'interno delle pagine web contenenti l'annuncio di marketing. Di seguito vengono elencate le informazioni inizialmente a disposizione, con annessa una breve descrizione:

- *Cookie ID*: identificativo univoco associato ad un cookie;
- *Cookie sospetto bot*: il sistema rileva i cookie che potrebbero essere dei bot, quindi dei "falsi utenti";
- *Numero di click*: quante volte il cookie ha cliccato sul banner;
- *Numero di impression*: quante volte il cookie ha visto la pubblicità;
- *Acquisto*: il sistema identifica se il cookie ha o meno comprato il prodotto (o il servizio) associato al banner;
- *Tipo di dispositivo, sistema operativo e browser*: vengono reperiti dei dati generici riguardanti le tecnologie associate al cookie;
- *Momento dell'interazione col banner*: percentuale di interazioni effettuate dai cookie per i diversi momenti della settimana e della giornata, con due livelli di granularità;
- *Lunghezza del testo della pagina*: percentuale di interazioni del cookie effettuate in siti web (contenenti l'advertising) con un certo numero di parole nel testo;
- *Categorie della pagina:* percentuale di interazioni effettuate all'interno di pagine con degli specifici argomenti (categorie). Il sistema di tracking rileva tre diversi livelli di categorie, dalla più generica alla più dettagliata (per esempio, *livello 1*: sport, *livello 2*: calcio, *livello 3*: scarpe con i tacchetti) e crea un'ulteriore classificazione ad-hoc, denominata admants;
- *Sentiment analysis*: il sistema di tracking è in grado di analizzare il sentiment degli utenti che visualizzano il banner pubblicitario, anche in questo caso con due livelli di dettaglio.

Pre-processing sui dati iniziali

Input: 1000 file CSV (zippati) contenenti in media circa *82 osservazioni* (righe), ciascuna rappresentante un cookie con *1416 feature* (colonne).

Output: una nuova versione del dataset con *82'564 righe* e *41 feature* per ciascun cookie.

Il primo fondamentale step da effettuare per poter raggiungere gli obiettivi prefissati, è analizzare ad alto livello i dati a disposizione, cercando eventuali *errori nella fase di reperimento* e *manipolando il dataset* attraverso molteplici tecniche di data management e cleaning, con lo scopo di ottenere un insieme di osservazioni opportunamente strutturato e dal quale poter estrarre del valore.

In questa sezione, verranno approfonditi i diversi step perseguiti durante la *fase di pre-processing*, ponendo il focus sulle variabili a disposizione e sull'associata gestione.

A. Cookie ID (ad_form_id)

Ciascuna osservazione viene identificata univocamente da un numero: a tal proposito, sono stati rilevati degli identificativi negativi, considerati erronei e convertiti in valore assoluto. Non risultano esserci duplicati e, per ottimizzare la gestione delle operazioni future, viene mantenuto come ID *l'indice del dataframe Pandas*;

B. Cookie sospetti bot (suspicious)

In presenza di un bot ci si aspetta solitamente un numero elevato di click sul banner. Per tale motivo, sono stati contati i cookie sospetti con almeno un click, che risultano essere circa il *2%* sul totale dei sospetti "falsi utenti" (*152*). Rilevata la scarsa informazione contenuta in questo campo, viene *eliminata la relativa colonna* dal dataset;

C. Numero di click (clicks) ***e di impression*** (impressions)

È stata rilevata *un'inconsistenza*: nel momento in cui viene effettuato un click, non viene contata l'impression associata, comportamento inaspettato in quanto si assume che per effettuare un click debba esserci la corrispondente precedente visualizzazione del banner. Per ovviare a tale problematica, il *numero di click viene sommato al valore iniziale di impression* dei cookie;

D. Acquisto del prodotto (buy)

Non è presente alcuna osservazione che abbia effettuato almeno un acquisto: viene *eliminata la colonna*;

E. Sistema operativo (os…) ***e browser*** (browser…)

Per definizione, un cookie è associato ad un'unica coppia sistema operativo-browser, proprietà verificata nei dati a disposizioni.

Con lo scopo di *ridurre la dimensionalità* in termini di feature, le *7* colonne associate ad un singolo sistema operativo e le *10* relative al browser, vengono trasformate in *2* attributi: *os_id* e *browser_id*, ciascuno dei quali fa riferimento ad un numero progressivo ed univoco assegnato all'interno degli insiemi dei sistemi operativi e dei browser stessi (esempio, il SO *Android* corrisponde all'ID *0*);

F. Tipo di dispositivo (device_type)

In questo caso, un identificativo univoco era già stato inizialmente assegnato a ciascun tipo di dispositivo. Tuttavia, notando l'assenza di occorrenze di alcuni di essi (esempio, non ci sono cookie provenienti da TV), è stato *riassegnato un ID progressivo* partendo da *0*, eliminando i tipi non presenti;

G. Impression per momento della giornata (time1 e time2…)

In primis, è stato scelto di lavorare solamente con le percentuali di impression per momento della giornata di *livello 1*: mattina, pomeriggio, sera e notte, sia per giornate lavorative che per weekend.

Di seguito si fa riferimento agli *errori* rilevati in fase di analisi, ciascuno con relativo approccio risolutivo:

- valori delle percentuali negativi: conversione in valore assoluto;
- somma delle percentuali per singola osservazione diversa dal *100%*: *normalizzazione nel range* [0,1] sulla base del totale di ciascuna riga (cookie).

È, inoltre, stata rilevata un'altra *inconsistenza nei dati*: il numero di impression per cookie non è compatibile con le distribuzioni di percentuali normalizzate (*ratio*) per momento della giornata (esempio, una sola impression fatta per lo *0.3* alla mattina e per lo *0.7* la sera). In merito al suddetto aspetto, sono state chieste delucidazioni al proprietario del dataset, il quale ha riferito una *problematica in fase di rilevamento delle impression*, che ha portato ad un loro conteggio erroneo.

- ***MIDA: Missing Impressions Detection Algorithm***

Per ovviare al problema del sistema di tracking è stato implementato un *algoritmo ad-hoc*, in grado di *stimare il numero di impression* partendo dalle percentuali normalizzate associate ai diversi momenti della giornata, e mantenendo le proprietà della distribuzione iniziale dei ratio con una certa percentuale di correttezza. Tale metodo parte dall'assunzione che i dati rilevati in relazione al momento della giornata siano corretti, mentre sia erronea la quantità di impression.

Passaggi dell'algoritmo con esempio

Obiettivo: calcolare la stima del numero di impression partendo dalla distribuzione di ratio A.

1) $\boldsymbol{A} = [\, 0.03 \,,\, 0 \,,\, 0.27 \,,\, 0.1 \,,\, 0 \,,\, 0 \,,\, 0.3 \,,\, 0.3 \,]$

Threshold: viene stabilito in partenza un *valore di soglia minimo per un'unità di impression*. Grazie a questo termine è possibile calibrare il *tradeoff* tra numero di impression rilevate ed attinenza alla distribuzione dei ratio. Per esempio, ponendo la soglia uguale a *0.01* (unità minima con arrotondamento a due cifre decimali) verrebbe mantenuta la massima coerenza con la distribuzione di partenza, ma al contempo verrebbero rilevate $\frac{1}{0.01} = 100$ impression, numero elevato rispetto alla media iniziale del dataset (circa *3* per cookie).

Nell'esempio, viene scelto il threshold pari a *0.05*, ponendo i valori minori ad esso uguali a 0.

2) $\boldsymbol{A'} = [\, \mathbf{0} \,,\, 0 \,,\, 0.27 \,,\, 0.1 \,,\, 0 \,,\, 0 \,,\, 0.3 \,,\, 0.3 \,]$

Normalizzazione: la nuova distribuzione viene traslata nuovamente nel range [0,1].

3) $\boldsymbol{A''} = [\, 0 \,,\, 0 \,,\, \mathbf{0.28} \,,\, \mathbf{0.1} \,,\, 0 \,,\, 0 \,,\, \mathbf{0.31} \,,\, \mathbf{0.31} \,]$

Unità di impression: viene definita l'unità ratio di impression, corrispondente al valore minimo della distribuzione.

4) $unit = \mathbf{0.1}$

Stima del numero di impression: calcolando il reciproco dell'unità si trova il numero stimato di impression.

5) $\frac{1}{unit} = \frac{1}{0.1} = \mathbf{10}$

Numero di impression per momento della giornata: moltiplicando i ratio di $\boldsymbol{A''}$ per *10* (a seguito di opportuno arrotondamento) si ottiene il numero di impression per ciascun momento della giornata.

6) $\boldsymbol{A'''} = [\,0\,,0\,,\mathbf{3}\,,\mathbf{1}\,,0\,,0\,,\mathbf{3}\,,\mathbf{3}\,]$

Nuova distribuzione di ratio: A''' viene diviso per il numero totale di impression stimato, trovando la distribuzione finale in termini di percentuali normalizzate.

7) $\boldsymbol{B} = [\,0\,,0\,,0.3\,,0.1\,,0\,,0\,,0.3\,,0.3\,]$

Correttezza dell'algoritmo: l'ultimo passaggio prevede di calcolare la correttezza percentuale rispetto alla distribuzione di ratio iniziale, utilizzando la distanza *D* tra i due array *A* e *B*.

8) $\boldsymbol{D(A,B)} = |0.03 - 0| + |0.27 - 3| = 0.06$

9) $\boldsymbol{Correttezza} = 1 - 0.06 = 0.94 \rightarrow \mathbf{94}\%$

Conclusione: l'algoritmo MIDA ha rilevato 10 impression con una correttezza pari al 94%.

Applicazione di MIDA sul dataset

L'algoritmo MIDA è stato applicato sul dataset con threshold uguale a *0.05*, ottenendo questi risultati:

Risultati MIDA		
Dati	*Dataset iniziale*	*Dataset processato*
Impression	≅ 280 mila	≅ 450 mila
AVG impression	3.5	5.5
Correttezza media	-	97.2%

Tabella 1: risultati di dataset iniziale e dopo MIDA

Per ciascun momento della giornata di *livello 1*, la versione finale delle associate feature corrisponde al numero (intero) di impression effettuate nello specifico arco temporale.

H. Lunghezza del testo della pagina web (L00_50…)

- Sono stati rilevati dei dati erronei nelle percentuali di impression relative alla lunghezza del testo, con valori indefiniti (*NaN*) e pari ad infinito (*inf*): i primi vengono considerati uguali a *0*, mentre i secondi al *100%*;
- *Median replacement*

  Il sistema di tracking in circa *4000* osservazioni non rileva alcun dato in merito a tale caratteristica della pagina. La strategia adottata in questa situazione è quella del *median replacement*, ovvero viene calcolato un array di mediane delle colonne associate alla lunghezza del testo, il quale viene inserito al posto della riga con i dati mancanti;
- Anche in questo caso, si è optato per una *normalizzazione nel range* [0,1];
- *Feature aggregation*

  Analizzando le distribuzioni dei singoli attributi, risulta che oltre il *90%* dei cookie non abbia mai incontrato il banner in pagine web con più di *250 parole*. Per bilanciare la situazione, vengono *aggregate* le colonne relative alla lunghezza del testo maggiore di *250*, ottenendo un'unica feature *L251_more*.
- Infine, partendo dalle distribuzioni di percentuali normalizzate (con opportuno arrotondamento), è stato computato il *numero* (intero) *di impression per ciascuna lunghezza testuale della pagina web*.

***I.* Categorie di argomenti della pagina web** (categories_1, categories_2, categories_3 ed admants)

- Si è deciso di lavorare solamente con le *categorie di livello 1* (ovvero quelle più generiche, in totale *25*) i cui valori percentuali sono stati aggiustati in termini di *NaN* ed *inf*, con la strategia vista in precedenza, e normalizzati nel *range* [0,1].
- La categoria denominata *"emotions"* risulta essere uguale a *0* in tutte le righe del dataset: di conseguenza è stata eliminata in partenza.

- *Mean replacement*

  Viene effettuato un check sulle osservazioni nelle quali il sistema di tracking non ha rilevato alcun dato (circa *4200*): in questo caso la tecnica applicata è stata la *mean replacement* (come nel *punto H*, ma calcolando un array di medie).

- *Gestione della categoria "uncategorized"*

  In circa *24 mila* cookie, una percentuale di siti web visitati viene etichettata come *"uncategorized"* dal sistema di tracking. Per gestire tale aspetto, si è optato per dividere la percentuale normalizzata della suddetta feature e distribuirla a tutte le altre *24 categorie*, eliminando quindi tale variabile ed applicando nuovamente la *normalizzazione nel range* [0,1].

J. Sentiment analysis del cookie (sentiments e feelings)

Il sistema di tracking non è stato in grado di rilevare e fornire dei dati consistenti riguardo la sentiment analysis nella navigazione dei cookie (più del 80% di dati mancanti): per questo motivo tale aspetto non è stato considerato e *le colonne associate sono state eliminate*.

I "nuovi" dati

In seguito, viene riportata una *tabella* rappresentante i dati a disposizione una volta conclusa la fase di pre-processing. Si noti come il precedente ed approfondito lavoro sul dataset, ha portato ad una rilevante *riduzione della dimensionalità* in termini di feature, mantenendo solamente le informazioni utili e necessarie agli obiettivi delineati in partenza e conferendo *interpretabilità, coerenza ed un formato logico ai dati*.

Dataset pre-processato		
1	Sistema operativo	Android (A) BSD (B) iOS (C) Linux (D) OSx (E) Others (F) Windows (G)
2	Browser	Android (A) Chrome (B) Chromium (C) Edge (D) Firefox (E) IE (F) Opera (G) Others (H) Safari (I) Unknown (L)
3	Tipo di dispositivo	Mobile (A) PC (B) Others (C) Tablet (D)
4	Numero di click	-
5	Numero totale di impression	-
6-13	Numero di impression per momento della giornata (sia per giorni lavorativi che per weekend)	Mattina Pomeriggio Sera Notte
14-18	Numero di impression per lunghezza della pagina web (in termini di caratteri)	0 – 50 51 – 100 101 – 250 251+
19-41	Categorie della pagina web (di livello 1)	Arte Sport Business …

Tabella 2: descrizione dataset pre-processato

Analisi esplorativa

Prima di entrare nel dettaglio con la modellazione e gli approcci di machine learning, è stata condotta *un'analisi esplorativa sul dataset pre-processato*, ottenendo una visuale ad alto livello sulle proprietà dei dati.

- ***Click***

Il primo parametro preso in considerazione è il *numero di click effettuati dai cookie*. Come si può notare dal pie chart in *Figura 1*, la percentuale di osservazioni con almeno un click sul banner è 0.33%, valore irrisorio che evidenzia un importante sbilanciamento nel dataset, il quale sfocerà nel *problema della class imbalance* nel contesto della classificazione binaria dei cookie in *"clicker"* e *"non clicker"*.

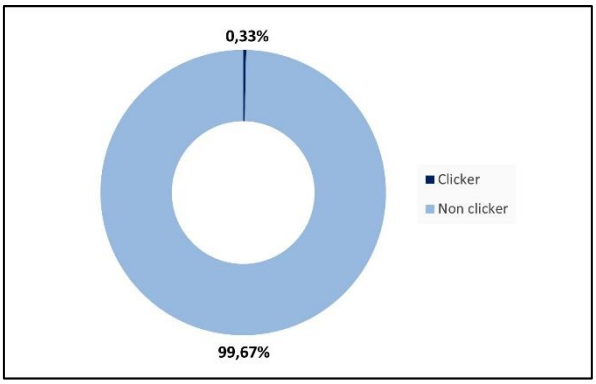

Figura 1: percentuali cookie "clicker" e "non clicker"

- ***Impression***

Analizzando il numero di impression effettuate da ciascun cookie, viene rilevata una media circa pari a *5*, notando una distribuzione concentrata per il 50% tra *2* (primo quartile) e *7* (terzo quartile) e con coda lunga (parecchi outliers e massimo uguale a *245*). Considerando le osservazioni con numero di impression minore o uguale al terzo quartile (*Figura 2*), si noti come ci sia una preponderanza di cookie con quantità di impression compresa nel range [1,3].

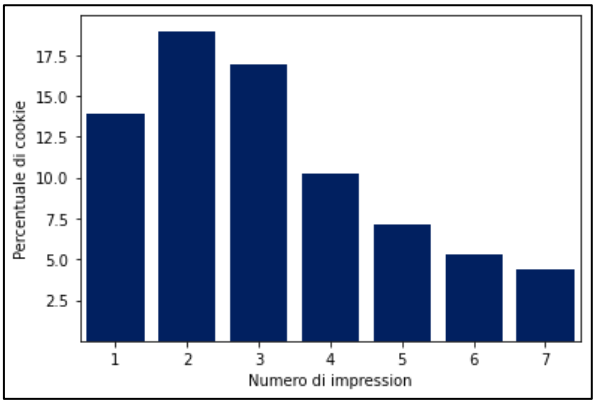

Figura 2: numero di cookie per numero di impression

- ***Impression per momento della giornata***

Come si può osservare nella *Figura 3*, il *67%* delle impression viene rilevato durante i giorni lavorativi. Scendendo nel dettaglio (*Figura 4*), si può concludere che, durante le giornate feriali, le visualizzazioni del banner si concentrino nella fase pomeriggio-sera, mentre nel weekend, oltre ad una percentuale elevata negli orari pomeridiani, si nota un aumento di interazioni durante la notte.

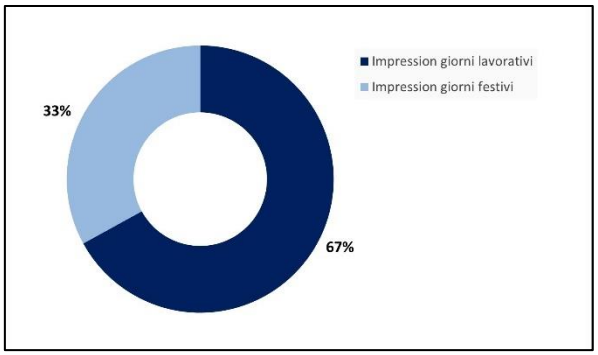

Figura 3: percentuali di impression giorni lavorativi e weekend

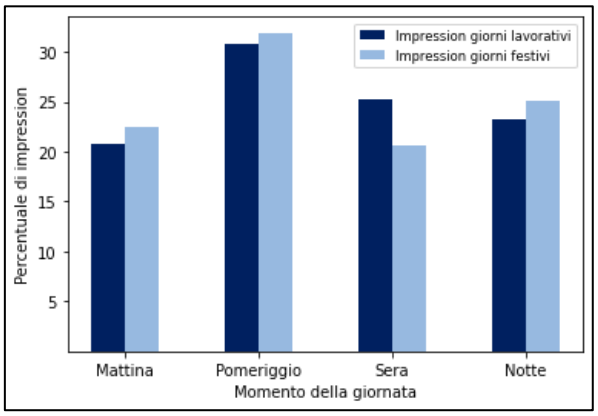

Figura 4: numero di impression per momento della settimana e della giornata

- ***Categorie delle pagine web***

Infine, viene studiata la percentuale delle categorie delle pagine web all'interno delle quali i cookie hanno interagito con il banner, notando come ci siano tre argomenti che spiccano: *arte, tecnologia ed hobby*.

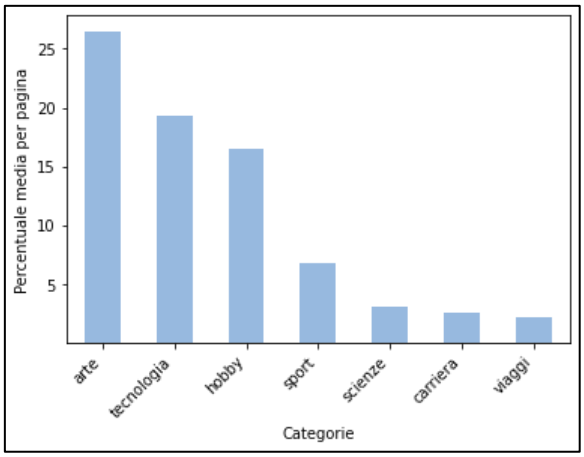

Figura 5: percentuale delle categorie delle pagine web

Pre-processing per machine learning

Input: i dati ottenuti dalla fase di pre-processing. Nel dettaglio: *82'564* righe (cookie) e *41* colonne.

Output: il training ed il test set, entrambi con *54* feature ed una colonna target (diversa per classificazione binaria e regressione, metodologie che verranno applicate in seguito).

Lo scopo di questo step è quello di processare nuovamente i dati in modo tale da renderli utilizzabili in modo effettivo all'interno degli algoritmi di machine learning che saranno l'argomento principale delle successive sezioni. Segue un elenco dei passaggi effettuati, con relativa descrizione:

1) One-hot encoding

Le variabili categoriali (*os_id, browser_id e device_type_id*) sono state trasformate in *array binari*, rendendole quindi di tipo booleano;

2) Creazione della label "clicker"

Obiettivo ML: *classificazione binaria* dei cookie in *"clicker"* (almeno un click) e *"non clicker"* (zero click sul banner).

Alle due classi (positiva e negativa) sono state assegnate le *label 1 e 0*;

3) Computazione dello score (impression e click)

Obiettivo ML: *studio e regressione* su un punteggio assegnato a ciascun cookie, ottenuto combinando il numero di impression e di click effettuati.

Lo *score* è stato calcolato con questo approccio:

- per le *osservazioni "clicker"*, il punteggio è dato dal rapporto tra numero di click e numero di impression. In questo modo, vengono premiati i cookie con più click per impression.

$$\boldsymbol{Score} = \frac{\#\ click}{\#\ impression}$$

- per le *osservazioni "non clicker"*, invece, il valore dello score è dato dalla seguente formula:

$$\boldsymbol{Score} = -\ \frac{\#\ impression - 1}{\#\ impression}$$

  Così facendo, vengono penalizzati i cookie con un numero elevato di impression, considerati i peggiori a cui risottomettere il banner in futuro.

Il metodo appena descritto risulta in un punteggio per cookie compreso nel *range* $[-1, 1]$, dove un valore più elevato rappresenta un'osservazione maggiormente indicata a cui sottomettere nuovamente il banner.

4) Suddivisione in training e test set

Questo passaggio prevede di dividere il dataset in training e test set. La strategia scelta è stata *75/25 (training/test) con stratified sampling* in modo da gestire il problema della class imbalance (classificazione). Si sottolinea, inoltre, che le operazioni che seguono sono state effettuate solamente nel dataset di training;

5) Eliminazione delle osservazioni duplicate

Con lo scopo di evitare eventuali bias in fase di training, come da best practices nel contesto del machine learning, sono state eliminate le osservazioni duplicate;

6) Feature selection

Per rimuovere possibili correlazioni tra le variabili esplicative del dataset, è stato predisposto uno *studio della multicollinearità*, seguendo questa logica:

- computazione della *matrice di correlazione* e rilevamento delle feature con un indice in valore assoluto ≥ 0.7;
- calcolo del *Variance Inflation Factor (VIF)* per l'insieme di variabili ottenuto in precedenza ed eliminazione della feature con *VIF* più elevato, mantenendo una *soglia minima di 5*;
- ripetizione del precedente passaggio con il nuovo insieme di feature fino a che il *VIF* di tutte le variabili rimanenti risulta < 5.

Seguendo tale metodologia, le feature "problematiche" da un punto di vista della correlazione vengono rimosse dal dataset, che conterrà quindi solamente *variabili indipendenti*;

7) Rimozione degli outliers

Con l'obiettivo di bilanciare il dataset sono stati eliminati gli outliers, prendendo in considerazione *separatamente* le due distribuzioni di osservazioni con classe positiva (*clicker*) e negativa (*non clicker*). Il procedimento di rimozione delle osservazioni si basa sulla *distanza di Mahalanobis*:

nello spazio multidimensionale considerato, i punti che superano una certa distanza (threshold) vengono rilevati come outliers e quindi eliminati dalle distribuzioni.

Tale soglia è definita dal punto critico corrispondente al *livello di significatività* $\alpha = 0.1$ (per cookie *"non clicker"*) ed $\alpha = 0.001$ (per i *"clicker"*) nella *distribuzione chi-quadro con n gradi di libertà* uguali al numero di variabili esplicative del dataset.

Per ottenere una soglia coerente utilizzando la distribuzione chi-quadro, i dati sono stati momentaneamente *standardizzati* (distribuzione normale).

Over ed under sampling

Input: il *dataset di training* (sbilanciato) descritto nella sezione precedente. Nel dettaglio: *27'772* osservazioni, di cui *il 99.4%* di classe negativa e lo *0.6%* con label *1* (cookie *"clicker"*).

Output: 3 dataset, ciascuno dei quali ottenuto attraverso una specifica *tecnica di resampling* (ribilanciamento).

Come già constatato in precedenza, il dataset preso in analisi è nettamente sbilanciato (problema della *class imbalance*). Per gestire tale situazione è necessario applicare delle apposite tecniche di ribilanciamento, in modo tale da eliminare (o diminuire) le conseguenti problematiche in fase di applicazione dei modelli di machine learning inerenti alla classificazione binaria (per esempio, l'overfitting).

Entrando nello specifico, sono state utilizzate tre strategie di seguito approfondite:

1) Over sampling

Questo approccio prevede di bilanciare il dataset *incrementando il numero di osservazioni della classe minore* (nel caso esaminato, la classe positiva), generando delle osservazioni sintetiche coerenti con la distribuzione della classe stessa ed aumentando la variabilità.

Nello specifico, è stata utilizzata la tecnica denominata *SMOTE (Synthetic Minority Over-sampling Technique)* la quale prevede la creazione di nuove osservazioni basandosi sull'algoritmo *k-nearest neighbors* (selezione casuale di elementi dalla distribuzione e creazione di nuove istanze interpolando i valori delle feature di ciascun campione selezionato ed i *k* punti più vicini nello spazio multidimensionale).

Nell'applicazione di tale algoritmo, si è deciso di effettuare un over sampling che portasse ad avere il numero di osservazioni con etichetta *1* pari alla *metà* dei cookie *"non clicker"*.

2) Under sampling

In contrapposizione al precedente metodo, l'*under sampling* prevede di ridurre il numero di osservazioni di classe maggiore (in questo caso, la classe negativa *"non clicker"*).

Nel dettaglio, è stata implementata la strategia denominata *NearMiss-1*, che prevede di selezionare un sottoinsieme di osservazioni della classe maggioritaria più simili a quelle della classe minoritaria, ovvero più vicine in termini di distanza nello spazio multidimensionale considerato;

3) Combined sampling (over + under)

Infine, è stato creato un dataset utilizzando una strategia ibrida che unisce over ed under sampling, denominata *SMOTEENN*. In breve, viene prima effettuato un over sampling utilizzando la tecnica *SMOTE* (illustrata in precedenza) e, successivamente, vengono eliminate le osservazioni più distanti dalla distribuzione della classe maggioritaria attraverso *l'algoritmo ENN*.

Si noti come le suddette metodologie si applichino solamente al *training set*, in quanto l'insieme di dati di test deve mantenere la numerosità e le proprietà iniziali (evitare il problema dell'*information/data leakage*).

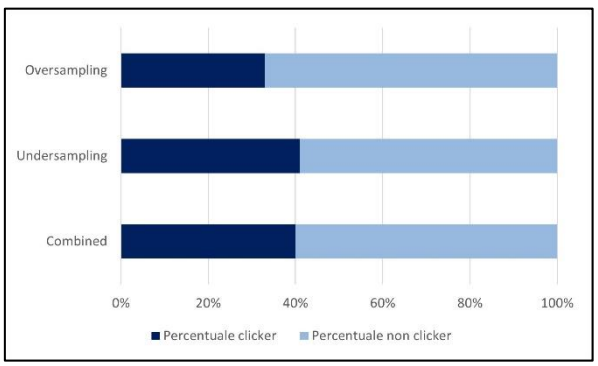

Figura 6: percentuali clicker e non clicker dopo le tecniche di resampling

Classificazione binaria: Random Forest

Il primo obiettivo della ricerca è quello di identificare e separare i cookie *"clicker"* e *"non clicker"*. Per portare a termine suddetto task di *classificazione binaria*, il primo algoritmo di machine learning utilizzato è stato *Random Forest*, tecnica che si basa sulla combinazione di molteplici *decision tree* e che è in grado di catturare interazioni complesse e relazioni non lineari tra dati. Inoltre, un altro aspetto rilevante da sottolineare, è la sua capacità di fornire una misura dell'importanza di ciascuna variabile esplicativa per il raggiungimento del risultato finale.

Di seguito, vengono descritte le peculiarità dell'approccio utilizzato, con un approfondimento riguardante le metriche di valutazione delle performance computate ed i risultati ottenuti:

- ***Dataset di training e testing***

In primis, si noti come la metodologia sia stata applicata a *quattro dataset di training*, con caratteristiche differenti: il dataset *base* (sbilanciato) ed i dataset bilanciati con le tre tecniche precedentemente descritte (*over*, *under* e *combined sampling*). In questo modo, è stato possibile confrontare diversi risultati, con lo scopo di trovare la soluzione migliore.

- ***L'approccio di machine learning***

La pipeline di machine learning perseguita, è definita in questo modo:

1) K-fold Cross Validation

Con lo scopo di rilevare i migliori *iperparametri*, è stata utilizzata la tecnica denominata *Stratified K-fold Cross Validation*, con $k = 3$. In questo modo, è stato possibile confrontare i risultati ottenuti in fase di *validation* con le diverse combinazioni di iperparametri (quali, per esempio, il numero di foreste, il criterio di valutazione e l'utilizzo o meno di pesi per le classi) sui diversi dataset, trovando quella che porta alle performance migliori.

Il *criterio di valutazione* scelto all'interno del ciclo di validation è stato la media della *F1-Score* ottenuta nelle *k* iterazioni effettuate.

Si evidenzia il fatto che questo step sia necessario per definire gli *iperparametri ottimali* che verranno successivamente utilizzati nella fase finale di training (per ciascuno dei quattro dataset).

Un altro aspetto rilevante da riportare riguarda la tecnica di *class weighting*: in tutte le versioni del dataset, le metriche migliori sono state ottenute pesando le osservazioni di classe positiva e negativa sulla base della loro numerosità all'interno del dataset di validation (relazione con class imbalance).

Oltre alla definizione degli iperparametri, la *K-fold Cross Validation* permette di ottenere una *stima dei risultati* che verranno ottenuti in fase di testing, in questo caso relativi alla metrica *F1-Score*, e valutare quindi possibili problemi di over ed underfitting. Nel dettaglio:

K-Fold Cross Validation	
Dataset	*F1-Score*
Base	0.015
Under sampling	0.819
Over sampling	0.976
Combined sampling	0.992

Tabella 3: risultati F1-Score K-Fold CV

Si noti come la *F1-Score* sia una metrica che combina *Precision* e *Recall*, risultando significativa in caso di *class imbalance* e fornendo una valutazione completa delle performance del modello. In questo caso, tre dataset su quattro ottengono risultati ottimi, anche se dovranno essere confermati in fase di testing;

2) Training

Trovati gli iperparametri ottimali della *Random Forest* per ciascun dataset, viene effettuato il *training del modello* con tutto l'insieme delle osservazioni di training (separatamente per ciascun dataset);

3) Testing e performance

I modelli allenati nella fase precedente vengono, infine, applicati al *dataset di testing* ottenuto nella fase di splitting iniziale, il quale mantiene la distribuzione e le proprietà dei dati di input.

In merito alla valutazione delle performance, vengono considerate le seguenti metriche:

- *F1-Score* (già definita in precedenza);
- *Recall*, in grado di valutare le performance del modello sulla base del *rapporto di osservazioni di classe positiva correttamente classificate*. In questo specifico caso di studio, tale valore è fondamentale per comprendere quanti cookie vengono correttamente etichettati come *"clicker"*;
- *decili delle probabilità predette*, per comprendere nel dettaglio come si distribuiscono le probabilità che il modello etichetti le osservazioni con label *1* oppure *0*.

Di seguito, vengono riportati i *risultati* ottenuti in fase di testing con i quattro modelli di *Random Forest*:

Analisi delle metriche

Metriche - Testing		
Dataset	*F1-Score*	*Recall*
Base	0.002	0.015
Under sampling	0.009	0.956
Over sampling	0.003	0.029
Combined sampling	0.007	0.132

Tabella 4: risultati metriche nel testing (Random Forest)

Analizzando la *Tabella 4*, è evidente come le performance del modello in termini di *F1-Score* risultino nettamente insufficienti, soprattutto se confrontate con i risultati ottenuti (ed attesi) in *fase di validation* (*Tabella 3*).

Viene rilevato, infatti, un corposo *overfitting*, giustificato dalla bassa numerosità di osservazioni di classe positiva e dalla *mancanza di pattern significativi* all'interno del dataset, confermati dagli esiti rilevati nonostante l'utilizzo di tecniche specifiche e funzionali di bilanciamento delle classi.

La *Recall* calcolata sul dataset ottenuto dall'*under sampling* risulta essere ottima, sfiorando la perfezione. Tale fenomeno, è però falsato dal fatto che questo specifico modello ha un consistente bias nel classificare i cookie come *"clicker"*, sbagliando l'etichetta per la maggior parte delle osservazioni di classe negativa (si veda l'analisi successiva).

Analisi dei decili delle probabilità predette

Correttezza delle predizioni		
Dataset	*Clicker*	*No Clicker*
Base	2%	96%
Under sampling	96%	27%
Over sampling	3%	95%
Combined sampling	13%	88%

Tabella 5: correttezza delle predizioni (Random Forest)

La *Tabella 5* contiene le percentuali di osservazioni etichettate correttamente per ciascuna delle due classi (positiva/clicker e negativa/no clicker).

Viene confermato il bias del modello con *under sampling*: i cookie *"clicker"* vengono classificati correttamente in quanto la maggior parte delle etichette predette sono di classe positiva. Per questo motivo, il risultato non è da considerarsi affidabile.

Negli altri casi, la *Random Forest* predice con un elevato grado di correttezza le osservazioni *"non clicker"*, ma fallisce gravemente nella classificazione della classe positiva.

Entrando nello specifico, vengono studiati gli andamenti del rapporto di osservazioni classificate correttamente per ciascun decile di probabilità, sia per i *True Positive* che per i *True Negative*.

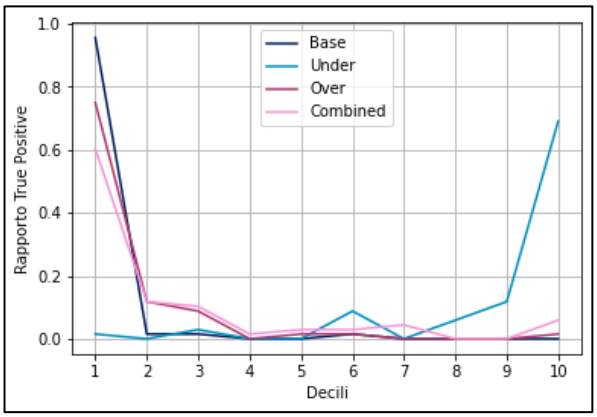

Figura 7: rapporto di True Positive per decile (Random Forest)

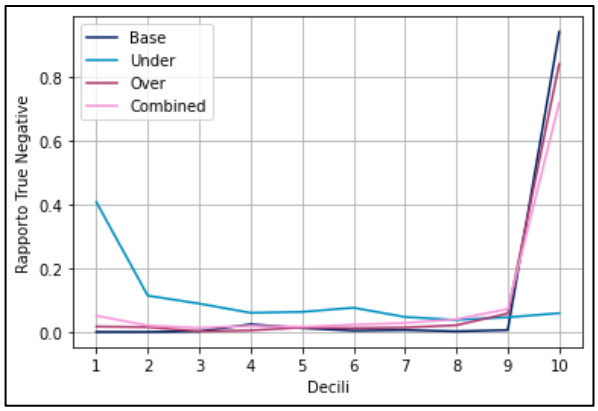

Figura 8: rapporto di True Negative per decile (Random Forest)

Considerando i modelli con *dataset base, over e combined sampled*, il risultato è praticamente identico: la quasi totalità delle osservazioni di classe negativa viene etichettata correttamente e con un grado di certezza oltre il *90%* (*decile 9*), mentre le predizioni relative ai cookie *"clicker"* risultano per lo più gravemente errate. I modelli in questione non riescono ad apprendere come rilevare la classe positiva per la *mancanza di una struttura definita nei dati* (come rilevato in precedenza ed anche nella successiva metodologia).

La *Random Forest* con il *dataset under sampled*, invece, si comporta al contrario a causa del fenomeno descritto sopra.

I risultati ottenuti fino a questo momento non sono soddisfacenti ed è possibile affermare come la strategia *Random Forest* non sia adatta al contesto ed ai dati analizzati.

Classificazione binaria: Mahalanobis

Considerando gli scarsi risultati ottenuti con la tecnica *Random Forest*, approccio "classico" di machine learning del tipo algoritmo-modello, viene testata una logica differente, basata sulla *distanza di Mahalanobis* e pensata appositamente per gestire la classificazione con la problematica della *class imbalance*.

Entrando nel dettaglio, la *distanza di Mahalanobis* misura la somiglianza/dissomiglianza tra un punto ed una distribuzione all'interno di uno spazio multidimensionale, risultando, quindi, effettiva e coerente per un'analisi multivariata come quella posta in essere. A differenza della tradizionale *distanza Euclidea*, la suddetta metrica prende in considerazione la correlazione e la varianza delle variabili, fornendo una misura più accurata ed attinente ai dati.

Formula della distanza di Mahalanobis

$$D^2 = (x - m)^T \cdot C^{-1} \cdot (x - m)$$

- x : vettore dell'osservazione (punto) da confrontare con la distribuzione;
- m : vettore con le medie di ciascuna variabile indipendente (colonne);
- C^{-1} : matrice di covarianza inversa, calcolata sull'insieme delle variabili indipendenti.

Per quanto concerne la *classificazione binaria*, l'idea alla base dell'approccio con *Mahalanobis* è quella di calcolare la distanza di ciascuna osservazione dalle due distribuzioni (classe positiva e classe negativa), ed assegnare la label corrispondente a quella più vicina (simile) al punto considerato. In questo modo, ipotizzando che le distribuzioni abbiano una *struttura definita*, sarà possibile separare correttamente i cookie *"clicker"* e *"non clicker"*, senza la necessità di effettuare un vero e proprio training (con annessa validation) del modello: il vettore *m* e la matrice *C* verranno calcolate con i dati del training set e successivamente utilizzate per computare le somiglianze relative alle osservazioni del dataset di test.

Come effettuato precedentemente con la *Random Forest*, sono stati considerati e valutati separatamente i quattro dataset, ottenendo i seguenti risultati:

Analisi delle metriche

Metriche - Testing		
Dataset	*F1-Score*	*Recall*
Base	0.009	0.398
Under sampling	0.007	0.941
Over sampling	0.007	0.074
Combined sampling	0.011	0.147

Tabella 6: risultati metriche nel testing (Mahalanobis)

Considerando la *Tabella 6*, si nota immediatamente come, anche con questa metodologia, la situazione risulti molto simile alla precedente, con delle performance di *F1-Score* e *Recall* non sufficienti per raggiungere gli obiettivi preposti.

Analisi dei decili delle probabilità predette

Correttezza delle predizioni		
Dataset	*Clicker*	*No Clicker*
Base	40%	73%
Under sampling	94%	14%
Over sampling	7%	94%
Combined sampling	15%	91%

Tabella 7: correttezza delle predizioni (Mahalanobis)

In merito alla percentuale di correttezza nelle predizioni delle osservazioni, nella *versione base* vengono rilevate delle performance maggiormente bilanciate tra classe positiva e negativa, rimanendo però relativamente basse per i cookie *"clicker"*.

Ad ogni modo, l'applicazione *dell'approccio Mahalanobis sul dataset base sbilanciato* risulta essere la soluzione migliore per effettuare la classificazione binaria in questo particolare contesto e con questi specifici dati.

Infine, analizzando l'andamento dei *True Positive* e *True Negative* per decile di probabilità, è possibile notare una maggiore variabilità (e quindi indecisione) nel classificare le osservazioni all'interno dei tre *dataset base, over e combined*, il che conferma nuovamente l'ipotesi relativa alla mancanza di una struttura definita che permetta di separare le due classi dei cookie.

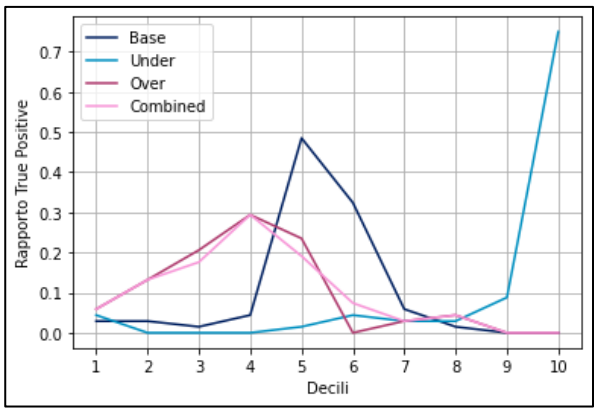

Figura 9: rapporto di True Positive per decile (Mahalanobis)

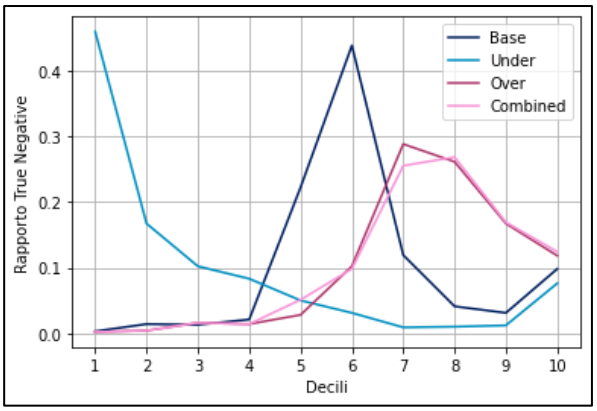

Figura 10: rapporto di True Negative per decile (Mahalanobis)

Classificazione binaria: conclusioni

Dopo aver applicato due algoritmi concettualmente differenti di machine learning su diverse versioni del dataset, ottenute con molteplici tecniche di bilanciamento e gestione della class imbalance, i risultati non sono stati ritenuti soddisfacenti per risolvere il problema posto inizialmente in essere e raggiungere gli obiettivi delineati.

Lo studio delle distribuzioni e le metriche analizzate, portano a concludere che, all'interno dei dati esaminati, *non ci siano pattern rilevanti e definiti che permettano di suddividere i cookie nelle due categorie "clicker" e "non clicker" con un sufficiente grado di correttezza*.

Suddetto esito è (ipoteticamente) imputabile ad una campagna pubblicitaria sostenuta senza delle precise e specifiche linee guida oppure relativa ad una fase iniziale di inserimento semi-randomico del banner nel web. Seguendo questa ipotesi, si sottolinea come le metodologie descritte fino a questo momento, risulteranno maggiormente effettive con il proseguimento della strategia di advertising nel tempo e con il miglioramento nella ricerca del target ideale per il prodotto da pubblicizzare.

Score e regressione

Appurata l'impossibilità di ottenere risultati rilevanti attraverso la classificazione binaria in cookie *"clicker"* e *"non clicker"* con i dati a disposizione, si è optato per una strategia differente, più adatta ad una fase iniziale della campagna pubblicitaria all'interno del web, con l'obiettivo di individuare un insieme ideale di cookie a cui inviare (o mandare nuovamente) il banner pubblicitario.

Nel dettaglio, invece che suddividere le osservazioni in due classi, viene assegnato un punteggio ad ogni cookie, compreso nel *range* $[-1, 1]$, basato sulla relazione tra numero di click ed impression effettuati (come già descritto in precedenza).

Il *principio* alle fondamenta di questa metodologia è il seguente: un cookie con punteggio elevato è più propenso a cliccare il banner e, quindi, sarà sensato inviargli nuovamente la pubblicità (e viceversa). Inoltre, fissando una certa *soglia di score*, sarà possibile analizzare le caratteristiche in comune ad un insieme di osservazioni, con lo scopo, per esempio, di identificare le migliori categorie di siti internet nei quali inserire il banner pubblicitario in futuro.

Per quanto concerne il dataset preso in esame, la distribuzione degli score per cookie viene descritta con il *boxplot* nella *Figura 11*, il quale evidenzia la consistente presenza di *valori bassi* e, quindi, di cookie da scartare in vista degli step futuri.
A tal proposito, è possibile definire una *soglia minima di punteggio* per identificare i cookie "migliori" a cui rimandare l'advertising, per esempio corrispondente a (circa) il valore del terzo baffo del grafico, ovvero -0.5.

Filtrando di conseguenza il dataset, si possono effettuare molteplici *analisi di clustering* per ricercare caratteristiche e pattern condivisi tra le osservazioni dell'insieme di dati considerato, in modo tale da definire delle linee guida per la sottomissione futura del banner nel web.

Nella fattispecie, per esempio, analizzando le *categorie* relative alle pagine web visitate dai cookie "migliori" (filtrati come dichiarato precedentemente) risulta che (circa) il 16% di essi ha incontrato la pubblicità in un sito della *categoria "Arte ed Intrattenimento"*. Nel proseguimento della campagna pubblicitaria potrebbe, quindi, essere sensato inserire il banner in siti web attinenti al genere rilevato.

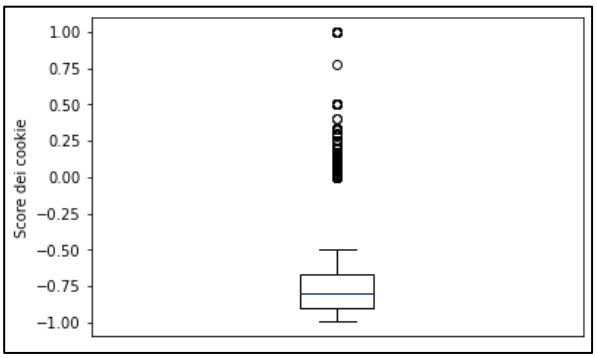

Figura 11: distribuzione degli score per cookie

Infine, si è deciso di lavorare con un algoritmo di regressione, nello specifico il *Random Forest Regression*, con lo scopo di identificare le *feature* che hanno una rilevanza maggiore per ottenere il punteggio. Dopo aver effettuato una fase di training e testing (con *Mean Squared Error* rilevato pari a *0.14*), le variabili più importanti in questi termini sono:

- lunghezza della pagina: *65%*;
- momento della giornata: *10%*;
- categorie "Hobby ed Interessi" e "Tecnologia e Computing": *5%.*

Queste informazioni, combinate con quelle reperite in precedenza, potranno essere funzionali per definire delle strategie pubblicitarie future, con lo scopo di trovare un target sempre più specifico e mirato di cookie.

Conclusione

Al giorno d'oggi, l'analisi dei dati e le tecniche di *Intelligenza Artificiale* sono degli strumenti fondamentali per gestire le campagne pubblicitarie online, sia da un punto di vista dell'identificazione del target di utenti che per ottimizzare i costi di business annessi.

Un aspetto cruciale che emerge dalle analisi effettuate è la necessità di utilizzare specifici approcci in base alle diverse fasi della campagna di marketing:

- in fase iniziale, sarà opportuno optare per uno studio delle caratteristiche e delle interazioni

con le pagine web contenenti il banner, per esempio affidandosi ad una strategia di punteggi per ciascuna osservazione (come descritto nella sezione *"Score e regressione"*). Questo a causa della mancanza di una struttura ben definita nella distribuzione dei dati, dovuta ad un approccio di advertising preliminare semi-randomico;

- successivamente, dopo aver ottenuto una migliore segmentazione degli utenti, si potranno applicare tecniche di classificazione, come quella binaria per la suddivisione dei cookie in *"clicker"* e *"non clicker"*.

In conclusione, per il successo di una campagna pubblicitaria basata sui dati, è necessario combinare le potenzialità della statistica e del machine learning con una processazione dei dati e degli studi sui risultati ottenuti ad-hoc, in modo tale da identificare la strategia ottimale per lo specifico contesto considerato.

Appendice A: una strategia di marketing

Viene riportato un possibile approccio alla campagna pubblicitaria su web (guidata dai dati) che utilizza le tecniche approfondite nel report:

1) Inserire il banner all'interno di pagine web in modo *semi-randomico*, cercando di distribuirlo maggiormente all'interno di siti inerenti al prodotto (categoria della pagina) ed al contempo di variare da un punto di vista delle altre feature (come, per esempio, il momento della giornata), ottenendo un primo insieme di dati da analizzare.
2) In questa fase sarà praticamente impossibile suddividere coerentemente i cookie in *"clicker"* e *"non clicker"* (come provato in precedenza nel paper), quindi, la strada ideale da perseguire è quella che passa per l'*analisi degli score per cookie*. L'insieme delle osservazioni a cui sottomettere il banner nel secondo step possono essere definite in questo modo:
   - dopo aver deciso una soglia minima di punteggio, il banner verrà inviato a tutti i cookie che rispettano suddetto vincolo;
   - inoltre, una seconda parte di pubblicità verrà inviata in modo randomico a nuovi cookie su pagine web con caratteristiche simili alle osservazioni precedentemente rilevate e con maggiore variabilità nelle feature risultate più importanti con la regressione.

   In questo modo, vengono considerati sia dei cookie "target" che, al contempo, delle proprietà dei siti web "ideali" per ottenere il successo della campagna pubblicitaria.
3) Una volta reperiti i dati risultanti dal processo precedente, si potranno testare le metodologie di classificazione e, in base agli esiti negativi o positivi ed agli obiettivi posti in partenza, ripetere il *passaggio 2* oppure ottenere un modello semi-definitivo per distinguere i cookie *"clicker"* e *"non clicker"*.

Appendice B: codice Python

L'implementazione pratica del progetto è stata condotta in linguaggio *Python* (con notebook in formato *ipynb*), seguendo le best practices dell'ingegneria del software e con il principio cardine di creare una componente riusabile, adattabile e facilmente interpretabile grazie alla presenza di commenti dettagliati per ciascuno step.

La cartella che include il codice annesso a questo report è struttura in questo modo:

- **campaign-dataset-split**: cartella contenente i dati di base ricevuti inizialmente, ovvero *1000* file CSV;
- **data**: cartella contenente tutti i dataset processati ed utilizzati durante lo sviluppo;
- **data-preparation**: cartella contenente i file Python per il processing relativo al machine learning ed al ribilanciamento (resampling) del dataset;
- **data-preprocessing**: cartella contenente i file inerenti alla lettura e formattazione iniziale dei dati ed all'intera fase di pre-processing;
- **models**: cartella contenente i notebook relativi ai tre approcci di machine learning utilizzati, ovvero Random Forest, Mahalanobis e Regression;
- **explorative-data-analysis**: un file con il codice per lo studio ad alto livello dei dati e per la creazione dei grafici presenti nel paper.

Bibliografia

Pre-processing sui dati iniziali

Min-Max Normalization
https://en.wikipedia.org/wiki/Feature_scaling

Median and mean replacement
https://vitalflux.com/pandas-impute-missing-values-mean-median-mode

Pre-processing per machine learning

Stratified sampling
https://medium.com/analytics-vidhya/stratified-sampling-in-machine-learning-f5112b5b9cfe

Multicollinearità e VIF
Analisi della multicollinearità
https://online.stat.psu.edu/stat462/node/180/

Variance Inflation Factor (VIF)
https://en.wikipedia.org/wiki/Variance_inflation_factor

One-hot encoding
https://en.wikipedia.org/wiki/One-hot

Distribuzione chi-quadro
https://en.wikipedia.org/wiki/Chi-squared_distribution

Over ed under sampling

Class imbalance
https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning

Tecniche di over ed under sampling
Riassunto
https://medium.com/grabngoinfo/four-oversampling-and-under-sampling-methods-for-imbalanced-classification-using-python-7304aedf9037

SMOTE
https://towardsdatascience.com/smote-fdce2f605729

NearMiss
https://analyticsindiamag.com/using-near-miss-algorithm-for-imbalanced-datasets

SMOTEENN
https://towardsdatascience.com/imbalanced-classification-in-python-smote-enn-method-db5db06b8d50

Data (information) leakage
https://machinelearningmastery.com/data-leakage-machine-learning/

Classificazione binaria: Random Forest

Random Forest
https://en.wikipedia.org/wiki/Random_forest

K-Fold Cross Validation
https://machinelearningmastery.com/k-fold-cross-validation/

Metriche performance classificazione
https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/

Class weighting
https://medium.com/grabngoinfo/balanced-weights-for-imbalanced-classification-465f0e13c5ad

Classificazione binaria: Mahalanobis

Distanza di Mahalanobis
Definizione
https://en.wikipedia.org/wiki/Mahalanobis_distance

Esempio
https://www.machinelearningplus.com/statistics/mahalanobis-distance

Distanza Euclidea e Mahalanobis
https://waterprogramming.wordpress.com/2018/07/23/multivariate-distances-mahalanobis-vs-euclidean/

Score e regressione

Random Forest Regression
https://towardsdatascience.com/random-forest-regression-5f605132d19d