Sommario

[Introduzione al problema 1](#_Toc166072083)

[Analisi funzionale 4](#_Toc166072084)

[Profilo assicurativo 4](#_Toc166072085)

[Coperture assicurative 6](#_Toc166072086)

[Coperture selezionate 7](#_Toc166072087)

[I sistemi di raccomandazione 8](#_Toc166072088)

[Sistemi basati sul contenuto 9](#_Toc166072089)

[Sistemi collaborativi 10](#_Toc166072090)

[Sistemi memory Based (neighborhood based) 11](#_Toc166072091)

[Sistemi model based 13](#_Toc166072092)

[Sistemi ibridi 14](#_Toc166072093)

[Valutazione di un sistema 14](#_Toc166072094)

[Considerazioni in ambito assicurativo 15](#_Toc166072095)

[Soluzione implementata 15](#_Toc166072096)

[Acquisizione dati 16](#_Toc166072097)

[Analisi dei dati 19](#_Toc166072098)

[Addestramento classificatore 21](#_Toc166072099)

[Valutazione del classificatore 21](#_Toc166072100)

[Popolamento matrice users-items 23](#_Toc166072101)

[Addestramento sistema di raccomandazione 24](#_Toc166072102)

[Architettura 24](#_Toc166072103)

[Conclusioni e sviluppi futuri 33](#_Toc166072104)

# Introduzione al problema

Il progetto presentato prevede l’analisi del sistema emissivo polizze di un cliente assicurativo di **Fincons Group S.p.A.**, lavorando sul sistema in questione ho notato delle peculiarità del sistema che andrebbero migliorate.

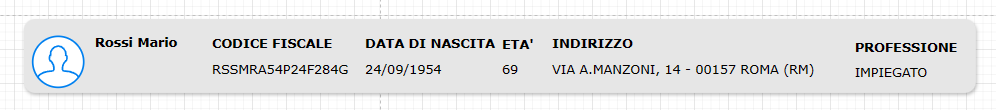
L’attuale sistema emissivo delle polizze assicurative risulta essere farraginoso, di poco impatto e anonimo. L’immagine mostra la pagina utilizzata dagli agenti per l’emissione delle polizze.

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

Pagina iniziale emissione polizze

L’agente assicurativo non ha a disposizione una panoramica esauriente del cliente che sta gestendo, per visualizzare il profilo del cliente deve aprire la pagina del profilo completo, senza avere la possibilità di visualizzare contemporaneamente le sue informazioni e le coperture assicurative che può proporgli. Nell’immagine che segue vengono mostrati i dati del cliente a disposizione dell’agente nella pagina di emissione.



Dati Cliente

L’agente assicurativo visualizza le coperture assicurative che è possibile vendere al cliente suddivise per tipologia prodotto. Le coperture assicurative raggruppate secondo questo criterio non sono personalizzate; quindi, l’agente non ha alcun supporto riguardo le coperture assicurative che sarebbe opportuno proporre al cliente. Se le coperture assicurative di una tipologia sono numerose non è possibile visualizzarle tutte contemporaneamente nella stessa schermata. Nell’immagine vengono mostrate le coperture assicurative presenti in pagina secondo il raggruppamento descritto.

Immagine che contiene testo, schermata, Policromia, software

Descrizione generata automaticamente

Elenco coperture assicurative

La pagina descritta è stata realizzata utilizzando in parte tecnologie legacy, quindi ogni volta che l’utente interagisce con la pagina essa viene ricaricata, senza garantire una buona interazione fornendo all’utente un’esperienza di navigazione limitata e deludente.

Dall’analisi eseguita sono emersi le problematiche riguardo l’utilizzo dell’applicativo:

* Durante l’emissione delle polizze assicurative, è possibile visualizzare solo una minima parte dei dati relativi al profilo cliente.
* Le coperture assicurative sono visualizzate indipendentemente dal cliente selezionato.
* Le coperture assicurative sono raggruppate in base alla tipologia di prodotto e non in base all’interesse del cliente.
* L’applicazione fornisce una scarsa interattività ed un’esperienza utente deludente.

# Analisi funzionale

Per risolvere i problemi descritti nel capitolo precedente si propone la realizzazione di una nuova dashboard.

La sua progettazione ruota intorno al cliente ed è stata pensata per essere utilizzata dagli agenti assicurativi. Sono state previste tre sezioni:

* Profilo assicurativo del cliente
* Sezione coperture assicurative
* Sezione beni selezionati

Il cruscotto progettato avrà l’aspetto mostrato nell’immagine:

Immagine che contiene testo, schermata, software, numero

Descrizione generata automaticamente

Cruscotto cliente

## Profilo assicurativo

Il profilo del cliente è stato pensato per rendere disponibili all’agente sempre le informazioni fondamentali, su richiesta le altre informazioni. L’agente avrà subito a disposizione un grafico riassuntivo con leggenda delle coperture assicurative, il nominativo, ed i suoi recapiti principali. L’immagine mostra un esempio appena descritto.

Immagine che contiene testo, schermata, software, Carattere

Descrizione generata automaticamente

Profilo assicurativo del cliente

Nella schermata visualizzata sono mostrati:

* Il nominativo del cliente nell’intestazione della pagina.
* Sulla sinistra un grafico riassuntivo con leggenda delle coperture attive. *Il colore delle coperture presenti nel grafico corrisponde al colore definito per la categoria di polizza.*
* Sulla destra i dati essenziali del cliente, *nominativo, indirizzo principale o di residenza, il telefono principale, l’indirizzo email principale.*

Cliccando sul pulsante *espandi* presente in basso a destra verrà visualizzata la il profilo completo del cliente. È prevista una scheda per ogni tipologia di dato *anagrafica, indirizzo, recapiti,* professione; cliccando su di uno delle schede presenti verranno visualizzati i dati corrispodenti.

Immagine che contiene testo, numero, Carattere, schermata

Descrizione generata automaticamente

Profilo completo cliente

Cliccando sul pulsante *espandi* presente a destra nella sezione *coperture assicurative* verrà visualizzato un report riassuntivo contenente le polizze stipulate dal cliente. L’immagine mostra un esempio del report appena descritto:

Immagine che contiene testo, schermata, software, Pagina Web

Descrizione generata automaticamente

Coperture assicurative

Nel report per ogni copertura visualizzata ci sono i dati:

* Codice univoco polizza
* Tipologia polizza
* Dettaglio bene assicurato
* Data stipulazione contratto
* Data scadenza contratto
* Importo premio totale

La sezione descritta permette all’agente di avere una panoramica dell’attuale situazione assicurativa del cliente.

## Coperture assicurative

La seconda parte della dashboard è incentrata sulle coperture assicurative che verranno proposte al cliente raggruppate in:

* **Coperture suggerite**: sono le coperture assicurative che vengono suggerite all’agente per proporle direttamente al cliente.
* **Coperture più vendute**: sono le coperture più vendute calcolate rispetto ad una statistica eseguita in base alle vendite dell’anno precedente a quello in corso.
* **Altre coperture**: sono le coperture escluse dai precedenti raggruppamenti.

Ogni copertura è rappresentata tramite una *card*, contenente i dati:

* Icona del prodotto
* Nome del prodotto
* Categoria del prodotto
* Descrizione del prodotto
* Spunta attraverso la quale è possibile selezionare o deselezionare il prodotto

Il colore utilizzato per rappresentare l’icona, il nome e la categoria del prodotto è il colore definito per la categoria.

Immagine che contiene testo, Carattere, linea, schermata

Descrizione generata automaticamente  
Card relative alle coperture assicurative

## Coperture selezionate

Il terzo raggruppamento presente in fondo alla pagina può essere considerato una bussola per l’utente.

Essa contiene dei segnaposti uno per ogni copertura selezionata, ogni segnaposto contiene:

* Icona del prodotto
* Nome del prodotto

Per rappresentare il colore di sfondo è stato utilizzato il colore definito per la categoria.

Immagine che contiene testo, schermata, Carattere, Marchio

Descrizione generata automaticamente

Coperture selezionate

# I sistemi di raccomandazione

L’obbiettivo di un sistema di raccomandazione[[1]](#endnote-1) consiste nel generare suggerimenti significativi per l’utente del sistema che possano essere di suo interesse. *I libri suggeriti da Amazon oppure i film su Netflix, rappresentano un chiaro esempio del funzionamento di un sistema di raccomandazione*. L’architettura del sistema dipende dal dominio applicativo, dai dati disponibili e dalle modalità con le quali l’utente interagisce con l’applicativo. *Su Netflix dopo aver visto un film è possibile esprimere una valutazione di gradimento*. Le valutazioni che esprime l’utente vengono memorizzate all’interno di una matrice *users-items*.

Immagine che contiene numero, schermata, parole crociate, calendario

Descrizione generata automaticamente

Matrice Users-Items

La matrice users-items rappresenta le preferenze degli utenti *(n in questo caso 5)* per gli items del sistema *(m in questo caso 6)*. Ogni cella **rij** *con i che va da 1 a 5 e j che va da 1 a 6* indica la valutazione dell’utente *(i)* per l’item *(j)*. La matrice users-items è una matrice sparsa, l’obbiettivo di un sistema di raccomandazione è calcolare la valutazione di un generico utente per gli items sui quali non ha espresso alcuna valutazione.

Ci sono diversi approcci ai sistemi di raccomandazione che si possono classificare in:

* Content based recommending: nei sistemi basati sul contenuto gli *items* che vengono suggeriti all’utente hanno un contenuto simile rispetto a quelli a cui l’utente in passato ha dato una valutazione positiva oppure ha acquistato.
* Collaborative Filtering Methods: nei sistemi collaborativi gli *items* suggeriti all’*utente* sono basati sulle valutazioni complessive degli utenti.
* Hybrid approaches: nei metodi ibridi vengono combinate le strategie viste sia per l’approccio basato sul contenuto che quelle basate sui sistemi collaborativi.

Immagine che contiene testo, schermata, linea, Carattere

Descrizione generata automaticamente

Tipologia sistemi di raccomandazione

## Sistemi basati sul contenuto

I sistemi di raccomandazione basati sul contenuto prevedono la creazione di una base di conoscenza creata in base agli elementi che si vogliono proporre al cliente.

Un esempio di base di conoscenza *per una videoteca* potrebbe essere definito come:

* Titolo film
* Descrizione trama
* Cast
* Regista
* Genere

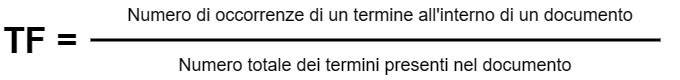
I campi proposti rappresentano un sotto insieme di una possibile base di conoscenza reale; ogni campo a sua volta può essere collegato ad altre basi di conoscenza. Il genere potrebbe essere descritto come:

* Titolo genere
* Descrizione genere
* Generi collegati
* Registi che seguono il genere
* Film del genere

La base di conoscenza creata verrà rappresentata come un grafo, dove i nodi rappresentano le entità ed le associazioni rappresentano la relazione che intercorre tra le entità collegate.

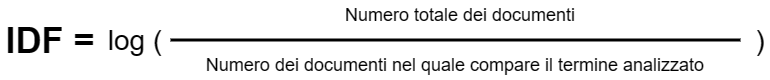
Dopo aver creato e popolato la base di conoscenza bisogna misurare la similarità tra gli elementi; essa viene calcolata utilizzando il **TF-IDF[[2]](#endnote-2)**, Il TF-IDF è costituito dal *TF - Term Frequency* e dall’*IDF - Inverse Document Frequency*.

Il **Term Frequency** di un termine viene calcolato come il rapporto tra *“il numero di volte che quel termine compare all’interno di un documento”* ed *“il numero totale dei termini presenti all’interno del documento”*.



Formula per il calcolo di TF – *Term Frequency*

L’**Inverse Document Frequency** di un termine viene calcolato come il *logaritmo* del rapporto tra *“il numero totale dei documenti”* ed *“il numero di documenti nei quali compare il termine analizzato”*.



Formula per il calcolo di IDF – Inverse Document Frequency

Il **TF-IDF** viene calcolato come il prodotto degli elementi presentati.

Immagine che contiene testo, Carattere, bianco, Elementi grafici

Descrizione generata automaticamente

Formula per il calcolo di TF-IDF

L’utilizzo del TF-IDF permette di non dover eliminare le *“stop words*[[3]](#endnote-3)*”* in quanto il TF-IDF calcolato per esse risulta 0 e quindi non rilevante.

Le raccomandazioni calcolate utilizzando l’approccio *Content Based Methods* prevedono di calcolare la similarità tra le coperture sottoscritte dal cliente e le coperture presenti all’interno della base di conoscenza.

Per calcolare la similarità è necessario rappresentare la base di conoscenza in forma numerica; per forma numerica s’intende creare un vettore relativo ad ogni copertura assicurativa presente nella base di conoscenza. Ogni vettore è costituito dalle caratteristiche individuate della copertura assicurativa rappresentata utilizzando gli **embeddings**[[4]](#endnote-4), la similarità viene misurata calcolando il coseno[[5]](#endnote-5) dell’angolo creato fra l’embedding relativo alla copertura assicurativa sottoscritta dal cliente e quelle presente nella base di conoscenza, se il risultato è “1” rappresenta la massima similarità *(tendono ad essere sovrapposti)* mentre se è “0” rappresenta la minima similarità *(tendono ad essere ortogonali)*.

## Sistemi collaborativi

I sistemi collaborativi lavorano sulla collezione delle preferenze sugli items indicate dagli utenti per un determinato dominio applicativo, esplorando le similarità presenti fra le valutazioni esistenti. *Per fare un confronto nella vita reale i sistemi collaborativi si basano sul concetto “intuitivo” di passaparola, es. una persona chiede ai propri amici o conoscenti cosa pensano dell’ultimo film uscito al cinema, oppure l’indicazione su quale ristorante scegliere.*

I sistemi collaborativi possono essere suddivisi in due ulteriori categorie:

* Metodo basato sul vicinato *(neighborhood based)* chiamato anche metodo basato sulla memoria *(memory based method)*.
* Metodo basato sul modello.

Immagine che contiene testo, schermata, linea, Carattere

Descrizione generata automaticamente

Suddivisione – sistemi collaborativi

### Sistemi memory Based (neighborhood based)

Le tecniche dei sistemi basati sul vicinato prevedono di calcolare la similarità dell’utente attivo utilizzando un sottoinsieme di utenti. Viene calcolata una media pesata delle loro valutazioni ed utilizzata per la stima della predizione relativa all’utente attivo. I passaggi di questo approccio possono essere schematizzati nell’algoritmo:

1. Assegnare un peso a tutti gli utenti rispettando la similarità con l’utente attivo.
2. Seleziona *k-utenti* che hanno la similarità più alta con l’utente attivo, *comunemente chiamati vicinato (neighborhood)*.
3. Calcolare la predizione attraverso una media ponderata tra le valutazioni del vicinato.

Per quanto riguarda il passaggio (1), si definisce *wa,u* la misura di similarità tra l’utente attivo *a* ed un utente generico *u*. La formula più diffusa per il calcolo della similarità è il *coefficiente di correlazione di Pearson*.

Immagine che contiene testo, Carattere, linea, bianco

Descrizione generata automaticamente

Formula per il calcolo del Coefficiente di correlazione di Pearson tra utente *a* ed utente *u*

Gli elementi presenti nel coefficiente di correlazione di Pearson sono identificati:

* I: rappresenta l’insieme delle preferenze definite dagli utenti.
* ru,i: è la valutazione dell’utente *u* rispetto all’item *i*.
* ra,i: è la valutazione dell’utente attivo *a* rispetto all’item *i*.
* : indica la media delle valutazioni eseguite dall’utente *u*.
* ****: indica la media delle valutazioni eseguite dall’utente attivo *a*.

Nel passaggio (3) le predizioni vengono calcolate come una media ponderata dello scarto relativo alle valutazioni dei vicini rispetto alla media delle loro valutazioni.

Immagine che contiene testo, Carattere, linea, bianco

Descrizione generata automaticamente

Formula per il calcolo dele predizioni

La formula relativa alle predizioni è definita come segue:

* pa,i: è la previsione per l’utente *a* relativa all’elemento *i*.
* wa,u: è la similarità tra l’utente *a* e l’utente *u*.
* K: è il vicinato, formato dagli utenti simili.

La similarità calcolata attraverso il coefficiente di correlazione di Pearson misura la distanza tra due variabili linearmente dipendenti. Ci sono diverse alternative per il calcolo della similarità, anche se studi empirici[[6]](#endnote-6) hanno dimostrato che il coefficiente di correlazione di Pearson offre prestazioni migliori.

L’approccio basato sul vicinato risulta essere poco scalabile in quanto la ricerca di utenti simili ha un’elevata complessità computazionale. Linden[[7]](#endnote-7) propone un approccio *collaborativo item-to-item*, in questo approccio vengono accoppiati gli item simili tra loro partendo dagli item valutati dall’utente. Seguendo questo approccio il coefficiente di correlazione di Pearson viene modificato ottenendo:

Immagine che contiene testo, Carattere, linea, bianco

Descrizione generata automaticamente

Formula del Coefficiente di correlazione di Pearson relativo all’approccio *item to item*

Gli elementi presenti in questa versione del coefficiente di correlazione di Pearson sono:

* U: rappresenta l’insieme degli utenti che hanno valutato sia l’item *i* che l’item *j*.
* ru,i: è la valutazione dell’utente *u* relativo all’item *i*.
* : è la media delle valutazioni degli utenti rispetto all’item *i*.
* : è la media delle valutazioni degli utenti rispetto all’item *j*.

La predizione della valutazione sull’item *i* da parte dell’utente attivo si ottiene eseguendo una media ponderata:

Immagine che contiene Carattere, testo, bianco, linea

Descrizione generata automaticamente

Formula per il calcolo delle predizioni per l’approccio collaborativo *item to item*

### Sistemi model based

I sistemi basati sul modello forniscono le raccomandazioni in base alle statistiche sulle preferenze degli utenti. Un prima approccio[[8]](#endnote-8) ai sistemi collaborativi suggerisce di trattare il problema come se fosse un problema di classificazione, quindi viene costruito un classificatore per ogni utente attivo rappresentando gli elementi *(items)* come vettori di features[[9]](#footnote-1) e le valutazioni dell’utente come le labels[[10]](#footnote-2).

L’approccio che viene maggiormente utilizzato fa uso dei *fattori latenti e della fattorizzazione di matrici*. Le tecniche di questo approccio si basano su nozioni statistiche che calcolano la similarità fra gli utenti, o fra gli elementi. Il modello a fattori latenti presume che la similarità tra utenti ed items sia indotta da fattori nascosti presenti nella struttura dati. *Es. un utente che valuta un film in modo positivo, probabilmente valuterà in modo positivo film che appartengono allo stesso genere o dello stesso regista.*

Nella fattorizzazione di matrici sono presenti un’ampia serie di tecniche utilizzate con successo che prevedono la rappresentazione simultanea degli utenti e degli item come un insieme di vettori di features sconosciute *(vettori colonna)* *wu, hi ∈ ℜk* , lungo *k* dimensioni latenti *(nascoste)*. Queste caratteristiche *(features)* approssimano la valutazione dell’utente *u* relativa all’elemento *i* attraverso il prodotto *wuThi*, approssimando il rating *ru,i* noto rispettando la funzione di costo *(loss function)[[11]](#endnote-9)*.

La squared loss è la scelta predefinita per la funzione di costo, nel caso in cui la funzione obbiettivo è minimizzata.

Immagine che contiene testo, Carattere, bianco, calligrafia

Descrizione generata automaticamente

Formula Squared Loss Function

Gli elementi presenti nella *squared loss function* sono:

* W = [w1…wn]T : è una matrice di rango *n \* k*.
* H = [h1…hm] : è una matrice di rango m \* k.
* L : identifica l’insieme di coppie *user-item* per cui il *rating[[12]](#footnote-3)* è noto.

Nei sistemi reali la matrice *R n \* m* contenente le valutazioni degli utenti *(users)* sugli elementi *(items)* sarà una matrice sparsa di grandi dimensioni, quindi si può ridurre il problema fattorizzando la matrice *R* , la fattorizzazione di matrici è un’operazione che permette di ridurre una matrice sparsa di grandi dimensioni nel prodotto di matrici dense di dimensioni più piccole.

Immagine che contiene schermata, design, Rettangolo, diagramma

Descrizione generata automaticamente  
Fattorizzazione di matrici

La Netflix Prize[[13]](#endnote-10) è stata vinta da un gruppo di ricerca, il quale ha utilizzato per eseguire la fattorizzazione di matrici il metodo a decomposizione a valori singolari troncata *(truncated SVD)[[14]](#endnote-11)* di *R*.

Immagine che contiene Carattere, bianco, Elementi grafici, testo

Descrizione generata automaticamente

Fattorizzazione di matrici a valori singolati

Questo metodo permette di rappresentare *R* come il prodotto delle matrici U, ∑, VT *(la trasposta di V)*,*R = U∑VT*. U è una matrice ortogonale di rango m \* m, VT è una matrice ortogonale n \* n, ∑ è una matrice diagonale a valori crescenti σ1 ≥ σ2 ≥ · · · ≥ σp, p = min{m, n}, conosciuta come valori singolari di A.

## Sistemi ibridi

I sistemi ibridi utilizzano tecniche che appartengono sia ai sistemi collaborativi che ai sistemi basati sul contenuto e calcolano le raccomandazioni usando entrambi gli approcci. Le implementazioni dei sistemi ibridi prevedono il calcolo delle predizioni sia con l’approccio basato sul contenuto che utilizzando l’approccio collaborativo separatamente e poi successivamente vengono combinate le previsioni ottenute da entrambi gli approcci, attraverso l’utilizzo di medie pesate.

## Valutazione di un sistema

Un modello di apprendimento automatico viene valutato attraverso l’utilizzo di metriche che confrontato i valori predetti per il set di test con i valori reali. La misura più diffusa è il MAE *(Mean Absolute Error)[[15]](#endnote-12)*. Essa è definita come la sommatoria della differenza del valore assoluto tra il valore predetto e la valutazione media per l’utente *u* relativo all’item *i*.

Immagine che contiene Carattere, testo, bianco, linea

Descrizione generata automaticamente  
Formula del MAE – *Mean Absolute Error*

Una metrica correlata molto utilizzata è la RMSE *(Root Mean Squared Error)*, la quale pone maggiore enfasi sul maggiore errore assoluto.

Immagine che contiene testo, Carattere, linea, bianco

Descrizione generata automaticamente  
Formula RMSE – *Root Mean Squared Error*

Queste metriche considerano gli items tutti uguali tra loro mentre per i sistemi di raccomandazione è di fondamentale importanza hanno lo svantaggio

## Considerazioni in ambito assicurativo

Prima di descrivere le implementazioni effettuate per realizzare il sistema di raccomandazione è necessario puntualizzare alcuni aspetti relativi al dominio preso in esame, *l’ambito assicurativo*, per il quale è necessario utilizzare degli accorgimenti.

Le peculiarità dell’ambito assicurativo che possono essere individuate sono:

* **Cardinalità ridotta**: l’offerta assicurativa può contenere un numero limitato di prodotti rispetto all’offerta delle piattaforme e-commerce o delle piattaforme di streaming on-line. Questo è un aspetto che non può essere risolto in quanto è una caratteristica intrinseca del dominio applicativo.
* **Affidabilità**: un libro, la visione di un film sono prodotti di uso comune che vengono acquistati giornalmente o settimanalmente. Le coperture assicurative vengono modificate raramente pertanto è necessario avere un elevato livello di affidabilità da parte del cliente. Per affrontare questo aspetto è necessario implementare un sistema che abbia un’elevata affidabilità garantendo all’utente l’effettiva possibilità di proporre al cliente elementi che possano essere realmente di suo interesse.
* **Vincoli**: quando si sottoscrive un’assicurazione ci sono diversi vincoli che vanno rispettati. Per risolvere questo aspetto è indispensabile rappresentare i vincoli esistenti nel mondo assicurativo creando una base di conoscenza, alcuni degli strumenti attraverso cui è possibile rappresentare un grafo sono RDF[[16]](#endnote-13) - *Resource Description Framework*, OWL[[17]](#endnote-14) - *Web Ontology Language* utilizzato per la rappresentazione di ontologie[[18]](#endnote-15), Neo4j[[19]](#endnote-16) per la rappresentazione di basi di dati a grafo.
* **Assenza di un rating esplicito**: nell’ambito assicurativo non è possibile chiedere un valore di gradimento esplicito al cliente e risulta difficile calcolarlo in modo implicito. Un approccio per la risoluzione di questa problematica può essere rappresentato dall’utilizzo di un metodo semi-supervisionato. In questo caso il rating viene calcolato da un algoritmo di apprendimento automatico.

Nel progetto presentato i problemi affrontati sono l’affidabilità e l’assenza di rating esplicito.

# Soluzione implementata

L’immagine mostra l’architettura implementata per il sistema di raccomandazione realizzato.

Immagine che contiene testo, diagramma, linea, Diagramma

Descrizione generata automaticamente  
Architettura sistema di raccomandazione

Il sistema di raccomandazione è stato implementato seguendo un metodo semi-supervisionato. Il termine semi-supervisionato riguarda le valutazioni relative ai prodotti assicurativi in quanto non sono state acquisite dal cliente, il quale esprimendo il proprio criterio di gradimento in merito al servizio offerto oppure attraverso la compilazione di un questionario di gradimento; ma sono state definite come una funzione di probabilità calcolata da un algoritmo di apprendimento automatico, in seguito è stato addestrato il modello di raccomandazione sulle valutazioni calcolate in precedenza. Per calcolare le valutazioni è stato addestrato un algoritmo di classificazione, il quale per ogni cliente calcola la prossima polizza assicurativa che il cliente potrebbe sottoscrivere lo score ottenuto verrà impostato come valutazione nella matrice *users-items*. I passi definiti per la definizione del processo sono:

* Acquisizione dati.
* Analisi dei dati.
* Addestramento del modello di classificazione.
* Valutazione del classificatore.
* Popolamento della matrice *users-items*.
* Addestramento del modello di raccomandazione.
* Valutazione del sistema di raccomandazione.
* Rilascio modello.

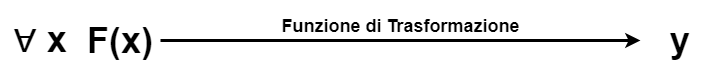
## Acquisizione dati

Il primo passo consiste nell’acquisizione e nell’analisi dei dati su cui eseguire l’addestramento del modello di classificazione per popolare la matrice *users-items*. I dati che vengono acquisiti dallo storage *(file in formato csv)* sono:

|  |  |
| --- | --- |
| **Nome Campo** | **Tipologia** |
| Sesso | Carattere |
| Data di Nascita | Data |
| Stato Civile | Stringa |
| Coniuge a Carico | Stringa *(true o false)* |
| Numero Figli | Numerico |
| Numero Figli a Carico | Numerico |
| Professione | Stringa |
| Reddito (RAL) | Numerico |
| Tipo Impiego | Stringa *(Lavoratore dipedente/Libero professionista)* |
| Provincia di Residenza | Stringa *(Sigla provincia)* |
| Polizza Stipulata | Nome polizza |

Dati Cliente presenti nello Storage

Su questi dati vengono eseguite le fasi di pulizia e trasformazione preliminare dei dati che daranno origine alle *features* utilizzate dal classificatore. La fase di trasformazione del dato avviene definendo per i campi coinvolti delle funzioni di trasformazione dato.

  
Definizione funzione di trasformazione dati

Le funzioni di trasformazione che sono state definite sono:

* **Estrazione anno e mese di nascita**: la funzione prende in input la data di nascita del cliente e ritorna in output il mese e l’anno di nascita del cliente.
* **Estensione stato civile**: la funzione prende in esame lo stato civile del cliente e ritorna lo stato civile ed un valore booleano che identifica se il cliente è single oppure in coppia. La definizione del flag *“is-single”* è data dalla tabella:

|  |  |
| --- | --- |
| **Stato Civile** | **Is Single** |
| Celibe/Nubile | True |
| Separato | True |
| Divorziato | True |
| Coniugato | False |
| Convivente | False |

Tabella di conversione stato civile

* **Conversione coniuge a carico**: conversione del valore presente per coniuge a carico da stringa a booleano, se è presente un valore *null* o *vuoto* viene impostato *false*.
* **Conversione numero figli**: converte il valore presente in numerico, se è presente un valore *null* o *vuoto* viene impostato a 0.
* **Validazione numero figli a carico**: converte il valore presente in numerico, verifica che il valore calcolato sia minore o uguale del valore calcolato, se è presente un valore *null* o *vuoto* o fallisce la validazione viene impostato a 0.
* **Determinazione fascia di reddito**: viene determinata la fascia di reddito andando a confrontare il reddito acquisito con gli estremi delle fasce di reddito presenti. Le fasce di reddito definite per la determinazione della fascia di reddito di appartenenza del cliente sono:

|  |  |  |
| --- | --- | --- |
| **Fascia Reddito** | **Limite Reddito Minimo** | **Limite Reddito Massimo** |
| Reddito Basso | 0 | 15000 |
| Reddito Medio Basso | 15001 | 28000 |
| Reddito Medio | 28001 | 55000 |
| Reddito Medio Alto | 55001 | 75000 |
| Reddito Alto | 75001 | 120000 |
| Reddito Molto Alto | 120001 | Nessun Limite |

Tabella di Conversione da Reddito (RAL) a Fascia Reddito

* **Conversione tipo impiego**: viene definito un nuovo campo booleano *IsFreelancer* il quale viene valorizzato con true se il campo *Tipo Impiego* vale *“Libero professionista”* con false se vale *“Lavoratore dipendente”*.
* **Definizione tipo reddito**: viene definito un nuovo campo *Tipo Reddito* che definisce il tipo reddito del cliente ottenuto in base alla professione ed al valore del campo tipo impiego. Nella tabella sono mostrati i valori previsti per questa tipologia di campo ed i criteri applicati per la conversione:

|  |  |
| --- | --- |
| **Descrizione tipo reddito** | **Criterio applicato** |
| Reddito da Lavoro Dipendente | Se Tipo Impiego vale *Lavoratore dipendente* |
| Reddito da Lavoro come Libero Professionista | Se Tipo Impiego vale *Libero professionista* |
| Nessun Reddito | Se professione corrisponde ad uno dei seguenti valori: *Casalinga, Studente, Disoccupato* |
| Altro Reddito | Se professione corrisponde ad uno dei seguenti valori: *Pensionato, Bracciante, Religioso, Società, Ente Privato, Ente Pubblico* |

Tabella per la determinazione del tipo reddito

* **Estensione residenza**: partendo dalla sigla della provincia viene aggiunto un nuovo campo contenente il nome della regione a cui appartiene la provincia in cui risiede il cliente.
* **Estensione delle polizze sottoscritte**: partendo dal nome della polizza sottoscritta dal cliente vengono aggiunti i campi:
  + Identificativo polizza *–* è un numero progressivo che identifica univocamente il tipo polizza.
  + Codice polizza – è una stringa alfanumerica che identifica univocamente il tipo polizza.
  + Descrizione polizza – rappresenta la descrizione del prodotto.

Dopo aver eseguito le trasformazioni descritte le *features* del cliente saranno:

|  |  |
| --- | --- |
| **Nome Campo** | **Tipologia Campo** |
| Mese Nascita | Numerico |
| Anno Nascita | Numerico |
| Stato Civile | Stringa |
| Is Single | Booleano |
| Coniuge Carico | Booleano |
| Numero Figli | Numerico |
| Numero Figli Carico | Numerico |
| Professione | Stringa |
| Fascia Reddito | Stringa |
| Tipo Impiego | Stringa |
| Is Freelancer | Booleano |
| Provincia Residenza | Stringa |
| Regione Residenza | Stringa |
| Nome Polizza | Stringa |
| Identificativo Polizza | Numerico |
| Codice Polizza | Stringa |
| Descrizione Polizza | Stringa |

Campi utilizzati per l’addestramento del classificatore

Le trasformazioni eseguite sono state necessarie per generalizzare i dati in modo da permettere al classificatore di riuscire a calcolare maggiori similarità tra i clienti e le polizze sottoscritte, un’eccessiva generalizzazione avrebbe avuto come effetto collaterale la perdita di accuratezza del sistema.

## Analisi dei dati

Sono stati acquisiti dal sistema *641743* dati relativi a clientiche hanno sottoscritto le polizze. Prima di procedere ad eseguire l’addestramento del modello è necessario capire la distribuzione dei dati che viene mostrata dalla tabella e dal grafico che seguono:

|  |  |
| --- | --- |
| **Nome Polizza** | **Numero Clienti** |
| R.C.A. | 574047 |
| INFORTUNI | 64399 |
| MULTIGARANZIA ABITAZIONE | 2415 |
| MALATTIA | 768 |
| R.C. DIVERSI | 78 |
| A.R.D. | 22 |
| GLOBALE FABBRICATI | 8 |
| INCENDIO/FURTO | 4 |
| TUTELA GIUDIZIARIA | 2 |

Tabella con Distribuzione dei dati

Grafico che mostra la distribuzione delle polizze acquistate dai clienti

Dai report mostrati è possibile notare che la distribuzione dei dati è fortemente sbilanciata verso l’R.C.A., questa è una situazione comunque reale in quanto all’interno di un nucleo familiare è possibile che ci siano più veicoli ed ognuno di essi deve avere l’assicurazione R.C.A. Inoltre, considerando il numero totale di clienti coinvolti e la distribuzione probabilmente ci sono clienti che hanno sottoscritto più di una polizza. I dati ottenuti vengono salvati su di una tabella del database *(CustomerLearningFeatures)*.

I clienti vengono raggruppati per caratteristiche simili escludendo i dati delle polizze sottoscritte, i campi in base a cui viene eseguito il raggruppamento sono:

|  |  |
| --- | --- |
| **Nome Campo** | **Tipologia Campo** |
| Sesso | Carattere |
| Mese Nascita | Numerico |
| Anno Nascita | Numerico |
| Stato Civile | Stringa |
| Single | Booleano |
| Coniuge Carico | Booleano |
| Numero Figli | Numerico |
| Numero Figli Carico | Numerico |
| Professione | Stringa |
| Fascia Reddito | Stringa |
| Tipo Reddito | Stringa |
| Freelancer | Booleano |
| Provincia Residenza | Stringa |
| Regione Residenza | Stringa |

Tabella di raggruppamento clienti

Ai campi presenti in tabella viene aggiunto un ulteriore campo calcolato che verrà chiamato *CustomerId* esso è un contatore *(campo numerico)* che viene incrementato di un’unità per ogni occorrenza del cliente individuata, una volta terminato il calcolo del *CustomerId* per tutti i clienti vengono aggiornati i dati. Il calcolo del *CustomerId* è un calcolo che avviene in modo incrementale. Di seguito viene mostrato lo pseudo-codice per il calcolo descritto. Pseudo-codice per il calcolo del CustomerId:

Immagine che contiene testo, Carattere, schermata, algebra

Descrizione generata automaticamente

Pseudo-codice per il calcolo del CustomerId

L’immagine mostra un esempio del risultato dell’applicazione dell’algoritmo per il calcolo del *CustomerId*.

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

Esempio di dati salvati sulla tabella *CustomerLearningFeatures*

## Riduzione cardinalità

Dopo aver eseguito le funzioni di trasformazione sui dati è possibile notare all’interno della tabella la presenza di occorrenze duplicate. Ci sono dei clienti aventi le stesse caratteristiche e quindi lo stesso *CustomerId* che hanno sottoscritto la stessa polizza, questa situazione si verifica in quanto ci potrebbero essere due clienti con le stesse caratteristiche che hanno sottoscritto la stessa polizza presso la stessa assicurazione oppure lo stesso cliente che ha rinnovato lo stesso contratto per più anni, nello scenario analizzato l’anno di stipulazione/rinnovo del contratto non è disponibile quindi per contenere il consumo di risorse le righe duplicate verranno rimosse in questo modo i dati potrebbero avere una distribuzione più omogenea e potrebbe aumentare la probabilità anche per quelle polizze la cui presenza risulta essere infinitesimale rispetto all’insieme.

|  |  |
| --- | --- |
| **Nome Polizza** | **Numero Clienti Duplicati** |
| R.C.A. | 59448 |
| MULTIGARANZIA ABITAZIONE | 42 |
| INFORTUNI | 1532 |
| MALATTIA | 8 |

Tabella riassuntiva dei clienti duplicati

La tabella mostra i clienti duplicati per tipologia di polizza, in totale sono circa un decimo dell’intero insieme.

Immagine che contiene testo, schermata, Carattere, algebra

Descrizione generata automaticamente  
Pseudo-codice per la riduzione della cardinalità delle features dei clienti

|  |  |
| --- | --- |
| **Nome Polizza** | **Numero Clienti** |
| R.C.A. | 541776 |
| A.R.D. | 22 |
| R.C. Diversi | 78 |
| MULTIGARANZIA ABITAZIONE | 2394 |
| GLOBALE FABBRICATI | 8 |
| INFORTUNI | 63619 |
| MALATTIA | 764 |
| INCENDIO/FURTO | 4 |
| TUTELA GIUDIZIARIA | 2 |

Tabella della distribuzione dei clienti dopo la riduzione

Grafico distribuzione clienti dopo riduzione

## Valutazione del modello

ML.NET permette di costruire un modello attraverso l’utilizzo di una GUI *(AutoML)*, esegue il training utilizzando diversi algoritmi e poi suggerisce quello che presenta le prestazioni migliori. In ML.NET sono state definite delle metriche per la valutazione dei modelli, i modelli di classificazione multi-classe vengono valutati attraverso le seguenti metriche:

* **Micro-accuracy**: L'accuratezza micro-media aggrega i contributi di tutte le classi per calcolare la metrica media. Corrisponde alla percentuale di istanze stimate correttamente. La micro-media non tiene conto dell'appartenenza a una classe. Essenzialmente, ogni coppia campione-classe contribuisce nello stesso modo alla metrica di accuratezza. L’obbiettivo di questa media è quello di tendere a 1.0.
* **Macro-accuracy**: L'accuratezza macro-media corrisponde all'accuratezza media a livello di classe. Viene confrontata l'accuratezza per ogni classe e l'accuratezza macro-media è la media di queste accuratezze. Essenzialmente, ogni classe contribuisce nello stesso modo alla metrica di accuratezza. Alle classi di minoranza viene assegnato un peso uguale a quello delle classi più grandi. La metrica della macro-media assegna lo stesso peso a ogni classe, indipendentemente dal numero di istanze di tale classe contenute nel set di dati. L’obbiettivo di questa metrica è quello di tendere a 1.0.
* **Log-loss**: La perdita logaritmica misura le prestazioni di un modello di classificazione in cui l'input della stima è un valore di probabilità compreso tra 0,00 e 1,00. Questa metrica aumenta quando la probabilità stimata devia dall'etichetta effettiva. L’obbiettivo di questa metrica è quello di tendere a 0.0.
* **Log-loss reduction**: La riduzione della perdita logaritmica può essere interpretata come un vantaggio del classificatore rispetto alla stima casuale. Questa metrica può assumere valore compreso tra 1.00 e 0.00, i valori che tendono a 1.00 sono stime migliori rispetto a quelle che tendono a 0.00.

I modelli di raccomandazione utilizzano le stesse metriche utilizzate per i modelli di regressione visto che entrambi producono un valore numerico. I modelli di regressione producono un valore numerico che rappresenta l’output di una generica funzione, i modelli di raccomandazione producono in output un numero compreso tra un intervallo *,valore di classificazione,* di valori *(es. una valutazione da 1 a 5)* oppure produce in output una raccomandazione *si/no* oppure *0/1*. Le metriche utilizzate per valutare i modelli di regressione o di raccomandazione sono:

* **R-Squared**: R-squared (R2)[[20]](#endnote-17) o coefficiente di determinazione rappresenta la potenza predittiva del modello come valore compreso tra -inf e 1,00. 1,00 significa corrispondenza perfetta e la corrispondenza può essere arbitrariamente insufficiente, quindi i punteggi possono essere negativi. Il punteggio 0,00 significa che il modello indovina il valore previsto per l'etichetta. Un valore R2 negativo indica che l'adattamento non segue la tendenza dei dati e il modello esegue prestazioni peggiori rispetto all'ipotesi casuale. Ciò è possibile solo con modelli di regressione non lineare o regressione lineare vincolata. R2 misura il grado di prossimità dei valori dei dati di test effettivi ai valori stimati. L’obbiettivo di questa metrica è tendere a 1.0.
* **Absolute-loss**: Absolute-loss[[21]](#endnote-18) o errore assoluto medio misura la prossimità delle stime ai risultati effettivi. Corrisponde alla media di tutti gli errori del modello, dove un errore del modello è la distanza tra il valore di etichetta stimato e quello corretto. Questo errore di stima viene calcolato per ogni record del set di dati di test. Infine, viene calcolato il valore medio per tutti gli errori assoluti registrati. L’obbiettivo di questa metrica è tendere a 0.0.
* **Squared-loss**: La perdita quadratica[[22]](#endnote-19) o l'errore quadratico medio (MSE), detto anche deviazione quadratica media (MSD), indica come chiudere una linea di regressione a un set di valori di dati di test prendendo le distanze dai punti alla linea di regressione (queste distanze sono gli errori E) ed elevandoli al quadrato. La quadratura assegna più peso alle differenze maggiori. Questa metrica assume sempre valori positivi, il suo obbiettivo è tendere a 0.0.
* **RMS-loss**: RMS-loss[[23]](#endnote-20) o radice dell'errore quadratico medio, anche detto radice della deviazione quadratica media, misura la differenza tra i valori stimati da un modello e i valori osservati nell'ambiente del modello. RMS-loss è la radice quadrata di Squared-loss e ha la stessa unità come etichetta, simile a absolute-loss ma assegnando più peso alle differenze maggiori. La radice dell'errore quadratico medio viene comunemente usata in climatologia, previsioni e analisi di regressione per verificare i risultati sperimentali. Questa metrica misura l’accuratezza del modello, assume sempre valori positivi ed il suo obbiettivo è tendere a 0.0.

## Addestramento classificatore

Per eseguire l’addestramento dei modelli è stata utilizzata ML .NET[[24]](#endnote-21) la libreria di machine learning sviluppata da Microsoft su tecnologia .NET Core. I test relativi all’addestramento sono stato eseguiti utilizzando la classe *ModelBuilder[[25]](#footnote-4)* e la GUI per eseguire il test e l’addestramento dei modelli in modalità visuale.

Quando si esegue l’addestramento del modello vanno seguiti i passi:

* Selezione la sorgente dati *(in questo caso la connessione al database sql server)*.
* Selezionare la tabella o la vista contenente i dati su cui eseguire l’addestramento.
* Specificare le opzioni per le colonne che rappresentano le features
  + È possibile includere o ignorare la feature.
  + La feature può essere impostata come continua o categorica.
* Identificare la colonna che rappresenta la label da predire.

L’immagine mostra una configurazione possibile per l’addestramento del classificatore.

## Valutazione del classificatore

La configurazione minima per eseguire l’algoritmo di classificazione è data dal 60 % dei clienti acquisiti, tale valore è il minimo consentito che consente di avere una copertura minima su tutti i prodotti. Nell’immagine viene mostrato un report con grafico relativo alla distribuzione dei dati per un’estrazione degli stessi al 60 %.

Immagine che contiene testo, schermata, numero, Carattere

Descrizione generata automaticamente

Di seguito viene riportato il report ottenuto in seguito all’addestramento del modello di classificazione.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algoritmo** | **Micro Accuracy** | **Macro Accuracy** | **Durata** | **Iterazione** |
| SdcaMaximumEntropyMulti | 0,9956 | 0,4000 | 3,6 | 0 |
| FastTreeOva | 0,9999 | 0,8000 | 4,2 | 1 |
| FastForestOva | 0,9987 | 0,6000 | 4,4 | 2 |
| SdcaLogisticRegressionOva | 0,9022 | 0,2000 | 8,3 | 3 |
| LightGbmMulti | 0,9999 | 0,8000 | 2,0 | 4 |
| LbfgsMaximumEntropyMulti | 0,9999 | 0,8000 | 1,6 | 5 |
| FastForestOva | 0,9987 | 0,6000 | 3,9 | 6 |
| LbfgsLogisticRegressionOva | 0,9999 | 0,8000 | 2,8 | 7 |
| LightGbmMulti | 0,8526 | 0,4717 | 1,8 | 8 |
| FastTreeOva | 1,0000 | 0,9167 | 17,9 | 9 |
| LbfgsMaximumEntropyMulti | 1,0000 | 1,0000 | 6,7 | 10 |
| SdcaMaximumEntropyMulti | 0,9022 | 0,2000 | 3,3 | 11 |
| FastForestOva | 0,9999 | 0,7500 | 21,1 | 12 |
| SdcaLogisticRegressionOva | 0,9022 | 0,1667 | 40,3 | 13 |
| LightGbmMulti | 0,9998 | 0,6531 | 5,3 | 14 |
| FastTreeOva | 1,0000 | 1,0000 | 36,9 | 15 |
| LbfgsLogisticRegressionOva | 1,0000 | 1,0000 | 14,1 | 16 |
| SdcaMaximumEntropyMulti | 0,9956 | 0,4000 | 2,1 | 17 |
| LbfgsMaximumEntropyMulti | 1,0000 | 1,0000 | 22,0 | 18 |

Tabella riassuntiva delle valutazioni dei modelli di classificazione

L’algoritmo di classificazione selezionato è stato il *LbfgsMaximumEntropyMulti*.

## Popolamento matrice users-items

Il primo approccio seguito per popolare la matrice *users-items* ha previsto l’inserimento per ogni *CustomerId* calcolato di un *“1”* per le polizze che il cliente aveva sottoscritto *“0”* altrimenti, lo pseudo-codice che segue mostra l’algoritmo appena descritto.

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente  
Pseudo-codice per popolamento matrice *users-items*

La matrice *users-items* ottenuta come risultato è una matrice sparsa fortemente sbilanciata per le polizze *R.C.A.* come è possibile vedere dall’immagine.



Matrice *users-items*

È possibile popolare la matrice *users-items* utilizzando il classificatore. L’algoritmo prevede di acquisire tutti i *CustomerId* salvati e per ognuno di essi calcola la probabilità che venga sottoscritta una polizza, se la predizione corrisponde alla polizza corrente alla coppia *(CustomerId-IdentificativoPolizza)* viene associato lo *score* calcolato dall’algoritmo di classificazione altrimenti viene assegnato 0. Il popolamento della matrice *users-items* è un processo incrementale. L’immagine che segue mostra lo pseudo-codice relativo all’algoritmo appena descritto.

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente  
Algoritmo di popolamento Matrice Users-Items

L’esecuzione dell’algoritmo indicato permette di ottenere una matrice *users-items* densa, questo è un passaggio fondamentale nella definizione dell’algoritmo di raccomandazione in quanto inserendo la probabilità solo in corrispondenza delle polizze effettivamente sottoscritte dai clienti si otterrebbe come risultato una matrice sparsa e fortemente sbilanciata per le polizze R.C.A. quindi il sistema di raccomandazione fornirà come suggerimento le polizze R.C.A. con una probabilità del 100 % le altre polizze non verrebbero considerate. L’immagine mostra un esempio della matrice *users-items* dopo aver eseguito l’algoritmo.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Esempio della matrice users-items

## Addestramento sistema di raccomandazione

Una volta popolato la matrice *users-items* è possibile eseguire l’addestramento del sistema di raccomandazione. ML .NET utilizza come algoritmo di raccomandazione la *MatrixFactorization[[26]](#footnote-5)* , i passi che vanno seguiti per addestrare il modello in questo caso sono:

* Selezione la sorgente dati *(in questo caso la connessione al database sql server)*.
* Selezionare la tabella o la vista contenente i dati su cui eseguire l’addestramento.
* Selezionare la colonna che rappresenta lo *UserId*.
* Selezionare la colonna che rappresenta lo *ItemId*.
* È possibile anche definire le colonne che rappresenteranno le features ele opzioni per esse.
  + È possibile includere o ignorare la feature.
  + La feature può essere impostata come continua o categorica.
* Selezionare la colonna che rappresenta il *Rating* da predire.

Nell’immagine viene mostrata una possibile configurazione del modello di sistema di raccomandazione.

# Architettura

L’immagine mostra l’architettura implementata per la gestione della piattaforma, in seguito verranno descritti i flussi implementati relativi ai vari componenti che costituiscono la piattaforma.

Immagine che contiene testo, diagramma, schermata, Piano

Descrizione generata automaticamente

Architettura Dashboard

Avendo implementato una PoC *(Proof of Concept)* ci sono degli elementi che sono stati simulati attraverso dei Worker Service[[27]](#endnote-22). I worker service implementati sono:

* **Gestione Clienti**: worker service che si occupa dell’acquisizione di nuovi clienti.

Immagine che contiene testo, schermata, linea, diagramma

Descrizione generata automaticamente  
Worker Service – Gestione Clienti

* **Gestione Polizze**: worker service che si occupa dell’emissione di nuove polizze sui clienti attivi e riattiva i clienti disattivati.

Immagine che contiene testo, schermata, Parallelo, diagramma

Descrizione generata automaticamente

Worker Service – Emissione Polizze

* **Gestione Polizze Scadute**: worker service che si occupa di cancellare le polizze scadute e la disattivazione dei clienti.

Immagine che contiene testo, Parallelo, schermata, numero

Descrizione generata automaticamente  
Worker Service – Eliminazione Polizze

* **Gestione Classificatore**: worker service che si occupa di acquisire le caratteristiche cliente e le polizze che ha sottoscritto dallo storage. Dopo procede a processarle ed a salvarle sul database (sql). Al termine del processo di acquisizione ed elaborazione dati provvede ad eseguire l’addestramento del modello di classificazione *(classificatore)*.

Immagine che contiene testo, diagramma, Parallelo, linea

Descrizione generata automaticamente  
Worker Service – Gestione Classificatore

* **Gestione Sistema di Raccomandazione**: worker service che si occupa di addestrare il sistema di raccomandazione, calcolando il rating che i vari clienti hanno dato alle polizze come score della previsione eseguita con il classificatore.

Immagine che contiene testo, Parallelo, diagramma, linea

Descrizione generata automaticamente  
Worker Service – Gestione Sistema di Raccomandazione

I worker service che simulano gli applicativi della piattaforma assicurativa dopo aver aggiornato i dati sul database provvedono ad inviare un messaggio su di una coda RabbitMQ[[28]](#endnote-23), i messaggi inviati vengono gestiti da processi serverless[[29]](#endnote-24) in ascolto sulla coda, i quali provvedono ad aggiornare le varie viste presenti su MongoDB i processi in questione sono stati implementati tramite delle Azure Function[[30]](#endnote-25). I messaggi gestiti sono:

* Inserimento nuovo cliente
* Eliminazione cliente
* Inserimento nuova polizza
* Eliminazione polizza

Di seguito verranno illustrati i diagrammi che descrivono il flusso gestito dalle Azure Function.

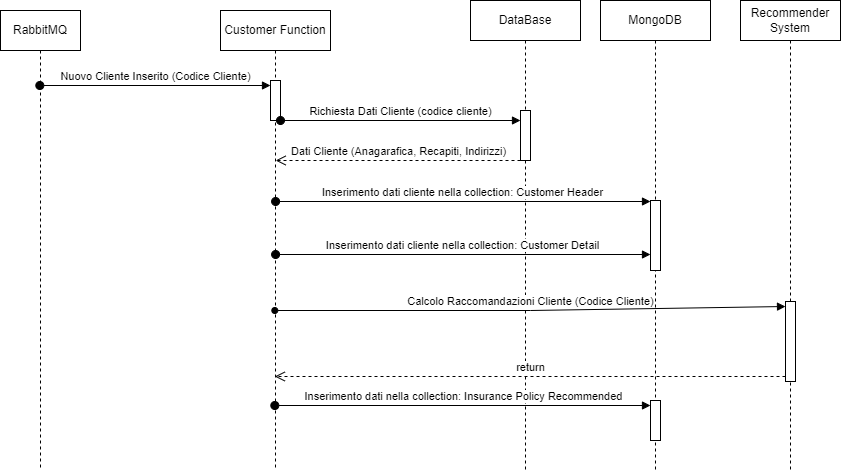
  
Azure Function – Inserimento Nuovo Cliente

Immagine che contiene testo, schermata, linea, Carattere

Descrizione generata automaticamente  
Azure Function – Eliminazione Cliente

Immagine che contiene testo, linea, diagramma, schermata

Descrizione generata automaticamente  
Azure Function – Inserimento Nuova Polizza

Immagine che contiene testo, schermata, linea, diagramma

Descrizione generata automaticamente  
Azure Function – Eliminazione Polizza

L’agente assicurativo interagisce con la dashboard il cui front-end è stato implemento utilizzando *Angular[[31]](#footnote-6)* mentre il back-end è stato implementato come una Web Api utilizzando *dotNET-7[[32]](#footnote-7)* nel diagramma che segue viene mostrato il flusso d’interazione dell’utente con la dashboard.

Immagine che contiene testo, documento, Parallelo, bianco e nero

Descrizione generata automaticamente  
Interazione Dashboard

# Conclusioni e sviluppi futuri

Il progetto presentato vuole fornire un’idea del supporto che l’impiego di tecnologie come il machine learning possono fornire al settore assicurativo.

In questo caso ho implementato un sistema di raccomandazione utilizzando un approccio collaborativo semi-supervisionato.

Il sistema implementato lavora off-line e i suggerimenti calcolati per il cliente vengono salvati su *MongoDB*: in questo modo l’utente della dashboard ha un’applicazione reattiva, dinamica e e aggiornata realtime. Inoltre, evitando di calcolare le raccomandazioni ad ogni richiesta dell’utente, si evita un consumo di risorse il più delle volte superfluo. Per l’implementazione del sistema di raccomandazione è fondamentale addestrare il classificatore con una notevole base di dati eterogenea e consistente in modo da ottenere un sistema che accresca sempre più la sua accuratezza. L’approccio semi-supervisionato rispetto all’approccio supervisionato riduce le prestazioni del sistema in quanto vengono utilizzati dei valori che non sono stati acquisiti dagli utenti *(es. tramite intervista o questionario)* ma calcolati da un algoritmo di apprendimento automatico.

Gli sviluppi futuri che possono essere eseguiti sono molteplici.

* **Ampliare le features**: inserendo i dati dei beni assicurati, le garanzie acquistate, dati estratti da questionari compilati dal cliente *(es. questionario sulla salute)*, solvibilità del cliente e metodi di pagamento utilizzati. In questo modo il classificatore, collezionando una mole di dati maggiore, fornirà delle predizioni più accurate.   
  Se le features sono troppe da gestire è possibile applicare un algoritmo di clustering in modo da creare gruppi di clienti omogenei. Questa riduzione però potrebbe ridurre le prestazioni, quindi in fase di analisi e test dei modelli bisogna valutare se è il caso di applicare un algoritmo di clustering oppure utilizzare meno features in modo da ridurre la frammentazione dei clienti naturalmente.
* **Utilizzare un approccio ibrido**: sarebbe possibile creare una base di conoscenza assicurativa indicando le peculiarità di ogni polizza, *es. garanzie obbligatorie, garanzie facoltative, vincoli e clausole, modalità di pagamento ammesse, caratteristiche dei beni che possono essere assicurati*. In questo modo sarebbe possibile calcolare la similarità tra i prodotti assicurativi venduti, *utilizzando il TF-IDF descritto in precedenza*, integrando l’algoritmo collaborativo implementato.
* **Creazione di pacchetti assicurativi**: nella soluzione proposta l’algoritmo calcola la probabilità che un cliente possa sottoscrivere o meno una determinata polizza ed in seguito calcola i suggerimenti per il cliente. Inserendo le garanzie acquistate fra le caratteristiche del cliente ed implementando la base di conoscenza in modo da rappresentare le relazioni presenti fra le garanzie ed i vincoli esistenti sulle garanzie, possono essere creati dei pacchetti assicurativi su misura per ciascun cliente. Si potrebbe inoltre calcolare la probabilità che un cliente possa acquistare una determinata garanzia in futuro. Il sistema, quindi, andrebbe a suggerire al cliente dei pacchetti assicurativi personalizzati composti dalla polizza che si suggerisce di sottoscrivere indicando le garanzie che possono essere acquistate.

1. **Recommender Systems**, Prem Melville and Vikas Sindhwani [↑](#endnote-ref-1)
2. [**TF-IDF — Term Frequency-Inverse Document Frequency – LearnDataSci**](https://www.learndatasci.com/glossary/tf-idf-term-frequency-inverse-document-frequency/), Fatih Karabiber, [↑](#endnote-ref-2)
3. [**Stop word - Wikipedia**](https://en.wikipedia.org/wiki/Stop_word), Wikipedia [↑](#endnote-ref-3)
4. [**Getting Started With Embeddings (huggingface.co)**](https://huggingface.co/blog/getting-started-with-embeddings), Omar Espejel [↑](#endnote-ref-4)
5. [**Coseno - Wikipedia**](https://it.wikipedia.org/wiki/Coseno), Wikipedia [↑](#endnote-ref-5)
6. John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, Madison, WI, July 1998 [↑](#endnote-ref-6)
7. Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing,

   7(1):76–80, 2003. [↑](#endnote-ref-7)
8. Daniel Billsus and Michael J. Pazzani. Learning collaborative information filters. In Proceedings of the Fifteenth International Conference on Machine Learning (ICML-98), pages 46–54, Madison, WI, 1998. Morgan Kaufmann [↑](#endnote-ref-8)
9. Feature: le caratteristiche che identificano un elemento *(item)* [↑](#footnote-ref-1)
10. Label: l’etichetta valore su cui eseguire la predizione [↑](#footnote-ref-2)
11. [Loss Function](https://en.wikipedia.org/wiki/Loss_function) [↑](#endnote-ref-9)
12. Rating: valutazione di un utente su di un elemento [↑](#footnote-ref-3)
13. [Netflix Prize](https://en.wikipedia.org/wiki/Netflix_Prize) [↑](#endnote-ref-10)
14. [Decomposizione a valori singolari (SVD)](https://it.wikipedia.org/wiki/Decomposizione_ai_valori_singolari) [↑](#endnote-ref-11)
15. [MAE - Mean Absolute Error](https://en.wikipedia.org/wiki/Mean_absolute_error) [↑](#endnote-ref-12)
16. [RDF - Resource Description Framework](https://www.w3.org/RDF/) [↑](#endnote-ref-13)
17. [OWL - Web Ontology Language](https://www.w3.org/OWL/) [↑](#endnote-ref-14)
18. [Ontologia Informatica](https://it.wikipedia.org/wiki/Ontologia_(informatica)) [↑](#endnote-ref-15)
19. [Neo4j - Docs](https://neo4j.com/docs/) [↑](#endnote-ref-16)
20. [R-Squared (R2)](https://en.wikipedia.org/wiki/Coefficient_of_determination) [↑](#endnote-ref-17)
21. [Mean absolute error](https://en.wikipedia.org/wiki/Mean_absolute_error) [↑](#endnote-ref-18)
22. [Mean squared error](https://en.wikipedia.org/wiki/Mean_squared_error) [↑](#endnote-ref-19)
23. [Root mean square deviation](https://en.wikipedia.org/wiki/Root_mean_square_deviation) [↑](#endnote-ref-20)
24. [ML .NET](https://learn.microsoft.com/it-it/dotnet/machine-learning/how-does-mldotnet-work) [↑](#endnote-ref-21)
25. [ModelBuilder](https://learn.microsoft.com/it-it/dotnet/api/microsoft.entityframeworkcore.modelbuilder?view=efcore-8.0) [↑](#footnote-ref-4)
26. [MatrixFactorizationTrainer](https://learn.microsoft.com/en-us/dotnet/api/microsoft.ml.trainers.matrixfactorizationtrainer?view=ml-dotnet-preview) [↑](#footnote-ref-5)
27. [Worker Service](https://www.w3.org/TR/service-workers/) [↑](#endnote-ref-22)
28. [RabbitMQ](https://en.wikipedia.org/wiki/RabbitMQ) [↑](#endnote-ref-23)
29. [Serverless](https://en.wikipedia.org/wiki/Serverless_computing) [↑](#endnote-ref-24)
30. [Azure Functions](https://learn.microsoft.com/en-us/azure/azure-functions/functions-overview?pivots=programming-language-csharp) [↑](#endnote-ref-25)
31. [Angular](https://angular.io/docs) [↑](#footnote-ref-6)
32. [dotNET-7](https://learn.microsoft.com/it-it/dotnet/core/whats-new/dotnet-7) [↑](#footnote-ref-7)