




LINK GITHUB: https://github.com/GamalElmetkanany/AIRBNB_MILANO.git

RELAZIONE PROGETTO ICON 23-24

DOCENTE: NICOLA FANIZZI

ANALISI MULTIDIMENSIONALE DI DATI AIRBNB A MILANO

A CURA DI: AGNESE D'ADDABBO (758308) - GAMAL ELMETKANANY (758332)
A.DADDABBO20@STUDENTI.UNIBA.IT G.ELMETKANANY@STUDENTI.UNIBA.IT



Indice

1.INTRODUZIONE AL PROGETTO	
2.IMPLEMENTAZIONE DEL SISTEMA	
2.1 Strumenti utilizzati.....	
2.2 Impiego degli Approfondimenti Metodologici.....	
2.3 Struttura progetto.....	
3.FUNZIONAMENTO.....	
3.3 PREPROCESSING DEI DATI.....	
3.3.1 Cleaning del DataSet.....	
3.3.2 Gestione delle Amenities.....	
3.3.3 Preprocessing per la Belief Network.....	
3.3.4 Preprocessing per il Clustering.....	
4. MOTIVAZIONI SULLA SCELTA DEGLI APPROFONDIMENTI METODOLOGICI.....	
4.1 Scelta del clustering.....	
4.2 Scelta algoritmo K-Means.....	
4.3 Scelta Belief Network.....	
5. ANALISI DELLE PREFERENZE DEGLI UTENTI E PREDIZIONE DELLA QUALITA' DELLE STRUTTURE MEDIANTE BELIEF NETWORK E VARIABLE ELIMINATION	
6. DEFINIZIONE DI UNA BASE DI CONOSCENZA IN LOGICA DI PRIMO ORDINE PER LA RAPPRESENTAZIONE E L'INTERROGAZIONE DEI DATI	
7. CONCLUSIONI.....	

1. INTRODUZIONE AL PROGETTO

Il settore dell'hospitality, in costante evoluzione e caratterizzato da una crescente complessità, richiede approcci innovativi nell'analisi dei dati al fine di potenziare l'esperienza degli utenti.



Per tale motivo, siamo propensi a sviluppare una Base di Conoscenza volta all'esplorazione e all'ottimizzazione del vasto set di dati fornito da *"Inside Airbnb"*. Tale iniziativa si pone l'obiettivo di condurre un'analisi dettagliata e trasparente sulle attività di Airbnb in diverse città nel panorama internazionale. Nel contesto specifico del nostro studio, ci siamo concentrati sulla città di Milano, in quanto desideriamo fornire un contributo significativo alla comprensione del panorama degli affitti a breve termine in questa città. Attraverso l'applicazione di metodologie avanzate di apprendimento automatico, intendiamo non solo offrire una verifica dettagliata di attributi fondamentali quali il numero di letti e bagni, ma anche esplorare aspetti più complessi, tra cui la disponibilità di strutture per la preparazione di cibi.

2. IMPLEMENTAZIONE DEL SISTEMA

2.1 Strumenti Utilizzati

Al fine di condurre il nostro Caso di Studio in maniera efficace e svolgere un'analisi approfondita del set di dati, abbiamo deliberato di utilizzare i seguenti strumenti:

la scelta del linguaggio di programmazione per lo sviluppo del nostro progetto è ricaduta su **Python**. Questa selezione è stata motivata dalla vasta disponibilità di librerie e dalla facilità d'uso, con particolare attenzione a quelle strettamente rilevanti per i nostri obiettivi di ricerca. Tra le librerie chiave impiegate, figurano:

1. **Pandas:** Utilizzata per la gestione del dataset e le attività di preprocessing, Pandas è stata cruciale per la manipolazione efficiente e la preparazione dei dati per le fasi successive del progetto.
2. **PySwip:** Questa libreria è stata impiegata per interfacciarsi con gli strumenti dell'applicativo *SWI-Prolog*. Ciò è stato fondamentale per la costruzione e l'interrogazione della nostra **Knowledge Base**, integrando efficacemente la programmazione in Python con le funzionalità di Prolog.
3. **Scikit-learn (Sklearn):** Abbiamo fatto uso di *Scikit-learn* per la parte relativa al **clustering** dei dati. Questa libreria ci ha fornito gli strumenti necessari per implementare metodi avanzati di analisi dei dati, in particolare quelli legati al clustering.
4. **Pgmpy:** Utilizzata per l'inferenza probabilistica, Pgmpy è stata fondamentale per la costruzione e l'analisi della **Belief Network**. Ci ha permesso di modellare relazioni probabilistiche complesse tra le diverse caratteristiche delle stanze.

In aggiunta a queste librerie Python, abbiamo anche incorporato l'utilizzo di **Weka**, un'applicativo specializzato, per apprendere la struttura della Belief Network dai dati. Questo approccio si è reso necessario in quanto la struttura della rete non era conosciuta a priori e richiedeva un processo di apprendimento automatico.

2.2 Impiego degli Approfondimenti Metodologici

Knowledge Base(KB):

sistema organizzato di informazioni che rappresenta la conoscenza di un particolare dominio o campo di studio in un formato strutturato. Nel nostro caso è stato utilizzato per rendere facilmente accessibili i dettagli delle stanze. L'utente potrà interrogare la KB attraverso query mirate, ottenendo risposte accurate e pertinenti in tempo reale. Questa funzionalità sarà fondamentale per chiunque stia cercando una sistemazione specifica in base a requisiti particolari.

Clustering:

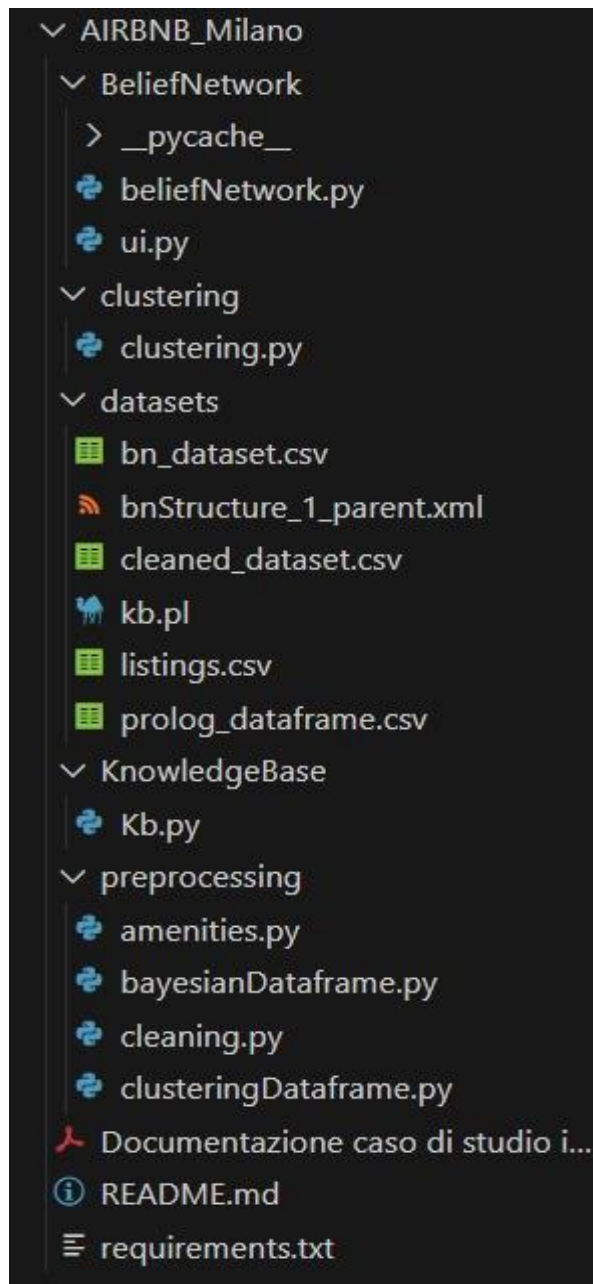
tecnica di apprendimento il cui obiettivo principale è quello di raggruppare insieme di oggetti simili tra loro. Nel nostro caso consente il raggruppamento di stanze simili, semplificando la ricerca per gli utenti. Questo approccio rende più efficace l'analisi di vasti insiemi di dati, consentendo di individuare pattern e tendenze altrimenti difficili da identificare.

Belief Network:

modello probabilistico che serve a rappresentare relazioni tra diverse variabili casuali, infatti per noi è stato cruciale per identificare correlazioni nascoste tra le caratteristiche delle stanze. Questo ci permetterà di calcolare probabilità a posteriori sulla qualità della stanza, basate su preferenze espresse dagli utenti. L'obiettivo finale è fornire raccomandazioni personalizzate e informazioni dettagliate per facilitare la scelta dell'alloggio.

In sintesi, quindi, la combinazione di Knowledge Base, Clustering e Belief Network promette di fornire agli utenti uno strumento potente per esplorare, valutare e selezionare le opzioni di alloggio più adatte alle loro esigenze, migliorando significativamente l'esperienza nel settore dell'hospitality a Milano.

2.3 Impiego degli Approfondimenti Metodologici



Package BeliefNetwork:

- Il modulo “*bielefNetwork*” carica una struttura di rete bayesiana da un file XML. Effettua inferenze sulla rete utilizzando il metodo di eliminazione delle variabili e restituisce i risultati in base alle preferenze specificate come input.

- Il modulo “*ui*” implementa un'interfaccia utente per interrogare una Knowledge Base tramite query Prolog e condurre inferenze su una rete bayesiana usando la libreria **pgmpy**. Gli utenti possono inserire query Prolog o specificare preferenze tramite attributi discreti e booleani per ottenere risultati statistici basati su una rete bayesiana predefinita.

Package clustering:

- Il modulo “clustering” legge un file CSV contenente dati. Esegue il clustering utilizzando l'algoritmo K-Means con un numero specificato di cluster e iterazioni, e aggiunge le etichette dei cluster al dataframe. Infine salva il dataframe aggiornato su un file CSV.

Package KnowledgeBase:

- Il modulo “*kb*” crea un file Knowledge Base Prolog (**kb.pl**) utilizzando i dati contenuti in un file CSV, includendo informazioni sulle caratteristiche delle stanze e definendo regole basate su tali dati (per esempio per determinare la disponibilità di prenotazioni, le caratteristiche delle stanze e i criteri di prezzo).

Package datasets:

In questo package è sono presenti i file CSV contenenti i dati dopo le varie fasi di preprocessing, il file KnowledgeBase il quale contiene regole, query e caratteristiche delle stanze e la struttura della rete Bayesiana ottenuta scegliendo un solo nodo genitore per ciascuna variabile figlia

Package preprocessing:

- Il modulo “*amenities*” definisce una funzione per creare una matrice di booleani che rappresenta la presenza o l'assenza di determinate comfort in un dataframe di annunci immobiliari, utilizzando una soglia per filtrare i comfort meno comuni. Inoltre, fornisce funzioni ausiliarie per manipolare i dati e pulire le colonne del dataframe.
- Il modulo “*bayesianDataframe*” prende un dataframe contenente dati sugli annunci immobiliari e applica una serie di trasformazioni per prepararlo all'uso in una belief network, incluso il calcolo di una matrice di presenza/assenza dei comfort più comuni. Le colonne non necessarie vengono eliminate, i valori delle colonne vengono discretizzati per creare categorie, e infine vengono effettuate alcune operazioni di pulizia e concatenazione dei dati per ottenere un dataframe pronto per l'analisi.
- Il modulo “*cleaning*” esegue il preprocessing di un dataset contenente dati sugli annunci immobiliari a Milano, comprese operazioni come la rimozione di colonne non necessarie, la gestione dei valori mancanti, la discretizzazione di alcune variabili, la trasformazione di valori booleani, e la creazione di nuove variabili. Inoltre, prepara il dataset per l'uso in una belief network e per l'analisi di clustering, salvando i risultati in file CSV separati.
- Il modulo “*clusteringDataframe*” esegue il preprocessing per il clustering di un dataset, includendo la creazione di una matrice di comfort, la rimozione di colonne non numeriche e non necessarie, l'applicazione della PCA (Principal Component Analysis) per ridurre le dimensioni, e la normalizzazione dei dati. Il risultato è un dataframe pronto per l'analisi di clustering.

FUNZIONAMENTO

Ora illustreremo nel dettaglio il processo che ha portato al compimento del nostro progetto.

3.3 PREPROCESSING DEI DATI

La fase iniziale del processo è stata il **Preprocessing dei dati**, ovvero una serie di operazioni atte a pulire e organizzare le informazioni in modo tale che fossero utilizzabili per le analisi successive.

La riduzione del dataset a informazioni rilevanti, la gestione delle amenities, la costruzione di un dataset per la belief network e il clustering sono passaggi chiave che hanno contribuito a rendere i dati più comprensibili e adatti all'analisi. Il risultato è un dataset strutturato e ottimizzato, pronto per essere esplorato attraverso varie tecniche di analisi e modellazione.

```
## Esecuzione del codice

**Importante eseguire i run nell'ordine in cui sono posti almeno per la prima volta**<br>

Fase di preprocessing <br>

<code> python preprocessing/cleaning.py ./datasets/listings.csv </code>

Creazione dei clusters <br>

<code> python clustering/clustering.py ./datasets/cleaned_dataset.csv [number of clusters] [number of iterations] </code>

Creazione Knowledge Base <br>

<code> python KnowledgeBase/Kb.py ./datasets/prolog_dataframe.csv </code>

User Interface per porre query al sistema<br>

<code> python BeliefNetwork/ui.py </code>
```

Ma ora analizziamo meglio le operazioni svolte in questa prima fase.

3.3.1 CLEANING DEL DATASET

Abbiamo iniziato con la pulizia del dataset, eliminando le colonne considerate poco rilevanti ai fini del nostro studio. Questa fase è stata fondamentale per ridurre il rumore nei dati e concentrarci solo sulle informazioni più rilevanti. Successivamente, ci siamo concentrati sulle tipologie principali di proprietà, eliminando righe che non contenevano il valore *'bedrooms'*. Un'attenzione particolare è stata posta nella modifica di caratteri problematici e nella gestione di valori booleani, trasformando, ad esempio, i valori 'TRUE' in 1.

Per affrontare i dati mancanti, abbiamo adottato una strategia di riempimento utilizzando *Simple Imputer*, metodo per la gestione dei valori mancanti che vengono sostituiti con altri valori calcolati sulla base di strategie. Nel nostro caso la strategia adottata per il rimpiazzamento di questi valori, è

stata la media degli altri valori presenti nella stessa colonna, questo perché si stava parlando di feature numeriche (ex. Altezza dell'edificio, prezzo della casa, etc). Per quanto riguarda le feature categoriche (ex. Colore, categoria, etc.), la strategia adottata è stata quella della sostituzione del valore più frequente presente sempre nella stessa colonna. Infine, abbiamo introdotto la colonna '*is_center*' per identificare le stanze collocate nelle zone centrali della città, basandoci su una lista creata attraverso una mappa di Milano. Questa colonna è stata utile, ad esempio, se la localizzazione in città ha un impatto sul prezzo delle stanze.

3.3.2 GESTIONE DELLE AMENITIES

La colonna '*amenities*' è stata successivamente affrontata, quella riguardante quindi i servizi offerti dalla struttura. Dopo un'operazione testuale, abbiamo eliminato le colonne correlate a un numero di stanze inferiore a un certo threshold (30% delle strutture totali). Questa operazione ha permesso di semplificare ulteriormente il dataset, concentrando l'analisi su un sottoinsieme più significativo di dati. Successivamente, abbiamo costruito una matrice da aggiungere al nostro dataframe, rendendo più efficiente la gestione di tali informazioni.

3.3.3 PREPROCESSING PER LA BELIEF NETWORK

La costruzione di un dataset utilizzabile per la belief network ha richiesto ulteriori operazioni. Abbiamo eliminato colonne non funzionali all'inferenza probabilistica, eseguendo casting ed operazioni testuali, convertendoli quindi in formati più appropriati per l'analisi. La procedura più significativa è stata la discretizzazione di varie colonne, rendendo i valori testuali, per migliorare la comprensibilità, quindi ad esempio la variabile di tipo intero '*prezzo*', verrà discretizzata in intervalli non più interi, ma testuali come ad esempio '*basso*', '*medio*', '*alto*'. Utilizzando il metodo di inferenza "Variable Elimination", abbiamo definito intervalli per suddividere il dataframe in modo omogeneo.

3.3.5 PREPROCESSING PER IL CLUSTERING

L'ultimo preprocessing è stato dedicato al clustering. Abbiamo selezionato le principali feature per valutare la similarità tra le camere, come prezzo, numero di recensioni e rating della struttura. L'applicazione di Principal Component Analysis (PCA), tecnica utilizzata per ridurre la dimensionalità dei dati e MinMaxScaler, metodo utilizzato per scalare le feature in modo che siano comprese in un intervallo, ha ottimizzato la gestione delle features. Dopo il clustering, abbiamo definito il dataset da utilizzare per la Knowledge Base, introducendo una colonna con il cluster relativo per ciascuna stanza all'interno del dataset ottenuto dall'operazione di cleaning.

4 MOTIVAZIONE DELLA SCELTA APPROFONDIMENTI METODOLOGICI

4.1 SCELTA DEL CLUSTERING

Nel corso della nostra analisi, abbiamo adottato la tecnica del clustering per identificare e raggruppare stanze che condividono caratteristiche simili. Questo approccio ci consente di creare dei cluster, ossia insiemi di esempi che sono più simili tra loro rispetto ad altri esempi nel dataset. Il clustering può essere di due tipi: hard clustering e soft clustering. Nel nostro caso, abbiamo optato per il primo tipo, che prevede un'assegnazione statica di ogni esempio ad una classe di appartenenza, utilizzando l'algoritmo **k-means**.

4.2 SCELTA ALGORITMO K-MEANS

L'algoritmo K-Means riceve in input un insieme di esempi di addestramento e il numero desiderato di classi k in cui dividere i dati, associando ad ogni classe un esempio tramite una funzione di assegnazione. Per ogni feature X_j , viene creata una funzione X_{bj} che fornisce una predizione del valore di quella feature per ogni classe.

L'obiettivo principale dell'algoritmo K-Means è minimizzare l'errore quadratico, che è definito come la somma dei quadrati delle differenze tra le predizioni delle feature e i valori reali delle feature per ogni esempio.

$$\sum_{e \in Es} \sum_{j=1}^n (\hat{X}_j(classe(e)) - X_j(e))^2.$$

Figura 1: definizione dell'errore quadratico

La scelta del numero ottimale di classi k è cruciale e può essere determinata osservando la curva dell'errore quadratico in funzione del numero di cluster. Si cerca il "gomito" della curva, il punto in cui l'aggiunta di ulteriori cluster non fornisce un miglioramento significativo, infatti sebbene chiediamo all'utente di specificare il numero di cluster e il numero di iterazioni, abbiamo utilizzato il "metodo del gomito" per determinare il numero ottimale di cluster, questa scelta è stata effettuata perché questo metodo ha la capacità di fornire una guida empirica e intuitiva per la selezione sul numero di cluster più appropriato al contesto dell'algoritmo k-means. Questo metodo consiste nel tracciare un grafico dell'errore in funzione del numero di cluster e individuare il punto in cui l'errore diminuisce significativamente.

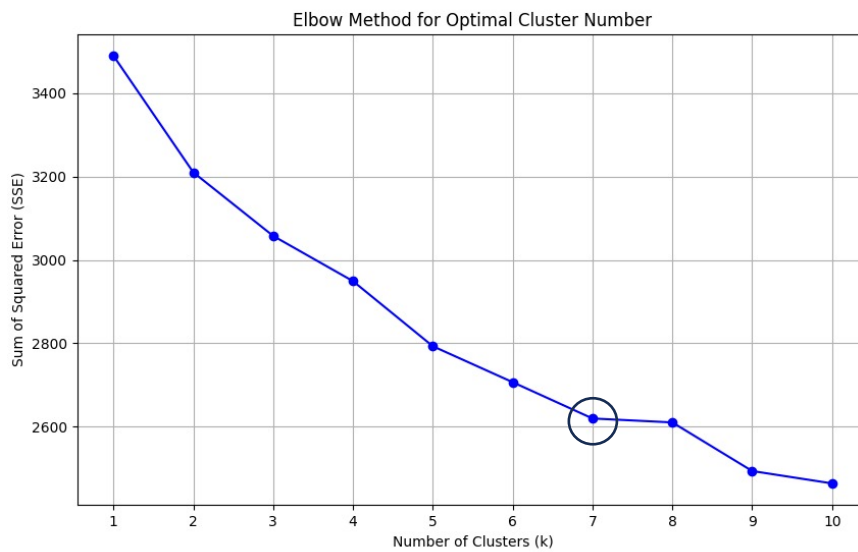


Figura 2: k ottimale ottenuto con il metodo del gomito

Nonostante la curva possa essere irregolare, abbiamo identificato 7 come il numero ideale di cluster poiché si trova subito dopo un cambio significativo di inclinazione.

Infine, ecco i valori delle coordinate dei centroidi, che rappresentano le medie delle features principali considerate nell'applicazione del k-means su 7 cluster e con 100 iterazioni.

Host is superhost	0.035	0.036	0.019	0.045	0.004	0.045	0.014
Is center	0.097	0.096	0.782	0.097	0.157	0.099	0.098
Host response rate	0.095	0.012	0.012	0.010	0.011	0.009	0.011
Number of reviews	0.104	0.075	0.088	0.118	0.877	0.117	0.101
Price	0.479	0.642	0.487	0.367	0.645	0.442	0.682
Review scores rating	0.373	0.251	0.381	0.373	0.306	0.400	0.593

È importante notare che abbiamo considerato solo le features numeriche poiché il k-means opera su questo tipo di dati. I valori potrebbero sembrare insignificanti, ma ciò è dovuto all'operazione di scaling che abbiamo eseguito in precedenza per standardizzare le features.

4.3 SCELTA BELIEF NETWORK

Nel nostro progetto, abbiamo deciso di utilizzare la Belief Network per analizzare i dati e comprendere le relazioni tra le diverse features. Questo approccio ci consente di visualizzare in modo chiaro e strutturato come le varie caratteristiche sono correlate tra loro. Inoltre, la realizzazione della Belief Network ci offre la possibilità di eseguire interrogazioni basate sull'inferenza probabilistica. Questo significa che possiamo ottenere informazioni più approfondite sui dati e fare previsioni sulla base delle relazioni probabilistiche identificate nella rete. In sostanza, l'utilizzo della Belief Network ci consente di esplorare e comprendere meglio la complessità dei nostri dati, fornendo una base solida per le nostre analisi e decisioni future.

Una struttura concettuale di notevole rilievo è rappresentata dalla rete bayesiana. Tale modello, esemplificato attraverso un grafico direzionato aciclico, delinea una distribuzione di probabilità congiunta su un insieme di variabili, alcune delle quali interdipendenti. La rete visualizza in modo chiaro le connessioni tra le caratteristiche, offrendo una serie di probabilità condizionate. Questo ordinamento delle caratteristiche si basa su una distinzione tra variabili genitore e figlie, dove i genitori costituiscono il minimo insieme di predecessori di una variabile figlia, garantendo che gli altri predecessori siano condizionalmente indipendenti dalla variabile figlia stessa, tenuto conto dei genitori.

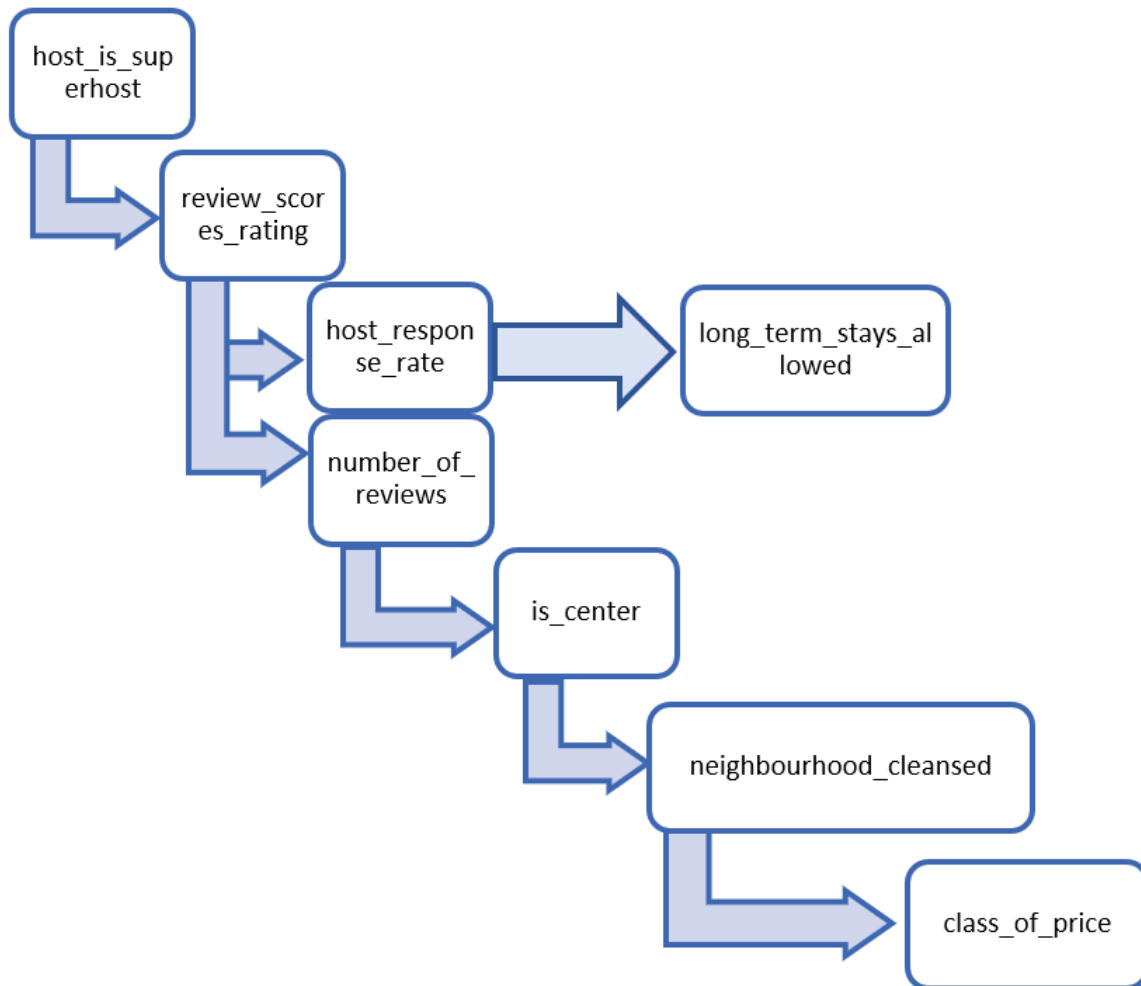
$$P(X_i | X_1, \dots, X_{i-1}) = P(X_i | \text{parents}(X_i))$$

Figura 3: probabilità condizionata

Nel contesto del nostro lavoro, la struttura della rete bayesiana non era predefinita e abbiamo dovuto apprenderla dai dati per ottenere l'ordinamento ottimale. Abbiamo impiegato l'applicativo *Weka* per questa operazione, il quale offre diverse opzioni di configurazione:

- **Per l'algoritmo di ricerca**- Per l'algoritmo di ricerca, abbiamo optato per il "Hill Climber", utilizzando lo score type "AIC" (Akaike Information Criterion). Abbiamo deciso di utilizzare un solo genitore per ciascun nodo, il che risultante in un classificatore Naive Bayes.
- **Per lo stimatore**, abbiamo utilizzato il "SimpleEstimator" con ($\alpha = 0.5$), un parametro che incide sull'apprendimento delle Conditional Probability Tables (CPT) e rappresenta un conteggio iniziale per ciascun valore. Abbiamo mantenuto il valore predefinito in quanto non avevamo informazioni sulle distribuzioni dei dati o eventuali pseudo-conteggi.

Abbiamo perfezionato il processo di selezione della struttura utilizzando il metodo della **k-fold cross validation**. Abbiamo diviso casualmente gli esempi di training in 10 fold. Questo ci ha permesso di riutilizzare gli esempi sia per il training che per la convalida, garantendo così l'utilizzo completo dei dati per il training. In questa struttura, ogni variabile figlia è considerata in relazione a un solo genitore. Di seguito sono riportati i punteggi ottenuti attraverso questo approccio.



Notiamo come la classe di prezzo dipenda dal quartiere, e come gli host più reattivi siano coloro che permettono lunghe permanenze.

La seguente tabella presenta i **log score** ottenuti utilizzando diversi criteri di selezione del modello. I log score forniscono una misura della bontà di un modello probabilistico, valutando quanto bene il modello si adatta ai dati e al contempo evita la complessità eccessiva. Valori più alti di log score indicano un modello migliore in termini di capacità predittiva e nell'attenzione dell'uso di risorse.

LogScore Bayes	-171125.5019531362
LogScore BDeu	-174482.15321895957
LogScore MDL	-173890.53465616005
LogScore ENTROPY	-170899.0085764058
LogScore AIC	-171554.0085764058

La seguente tabella di classificazione offre una visione più dettagliata delle prestazioni del modello nella classificazione delle istanze in diverse categorie. Le metriche di valutazione incluse sono:

Precision, *Recall* e *F-Measure*.

La *precision* indica la proporzione di istanze correttamente classificate come appartenenti a una determinata classe rispetto a tutte le istanze classificate come appartenenti a quella classe. *Recall*, invece, misura la proporzione di istanze di una classe correttamente classificate rispetto a tutte le istanze effettivamente appartenenti a quella classe. Infine, *F-Measure* è la media armonica di Precision e Recall e fornisce una misura complessiva delle prestazioni del modello.

Classe di qualità	Precision	Recall	F-Measure
Low Rating	0,624	0,766	0,688
Top Rating	0,557	0,161	0,250
Nice Rating	0,542	0,516	0,529
Good Rating	0,660	0,880	0,754
Media	0,603	0,616	0,580

La seguente matrice di confusione, invece, visualizza in modo dettagliato le prestazioni del modello, mostrando il numero di istanze correttamente e erroneamente classificate per ciascuna classe di output. Le righe rappresentano le classi reali, mentre le colonne rappresentano le classi predette dal modello. Gli elementi diagonali della matrice rappresentano il numero di istanze correttamente classificate, mentre gli elementi al di fuori della diagonale indicano gli errori di classificazione.

Classificato come →	Low Rating	Top Rating	Nice Rating	Good Rating
Low Rating	2825	190	351	320
Top Rating	777	320	323	568
Nice Rating	792	1	847	0
Good Rating	130	63	41	1721

Dopo aver completato il processo di apprendimento della struttura della Belief Network, gli utenti hanno l'opportunità di formulare query basate sulle loro preferenze personali riguardanti varie proprietà delle strutture, come ad esempio il quartiere di ubicazione, lo status di "superhost" dell'host e la disponibilità di diversi servizi. Queste preferenze utente vengono utilizzate come evidenze per l'algoritmo di **Variable Elimination**, dove la variabile da predire diventa il "review scores rating", ovvero la media delle recensioni rilasciate dagli utenti per la struttura in questione.

In pratica, ciò significa che le preferenze espresse dall'utente vengono utilizzate per calcolare la probabilità che le camere presenti nel dataset abbiano un determinato livello di qualità. Questo permette agli utenti di valutare con una certa affidabilità la probabilità che una determinata struttura soddisfi le loro aspettative di qualità e possa piacer loro.

La probabilità calcolata sarà quindi $P(\text{review_scores_rating} | \text{preferenze utente})$

la seguente query la seguente query cerca alloggi che sono gestiti da 'superhost', si trovano nel quartiere "Duomo", rientrano nella fascia di prezzo *media* e offrono la connessione *Wi-Fi*.

```
neighbourhood_cleansed = duomo, class_of_price = medium, host_is_superhost = True, wifi = True
```

```
neighbourhood_cleansed = duomo, class_of_price = medium, host_is_superhost = True, wifi = True
RATING      PROBABILITY
top_rating   0.2259
nice_rating  0.5039
good_rating  0.0567
low_rating   0.2135
```

5 DEFINIZIONE DI UNA BASE DI CONOSCENZA IN LOGICA DI PRIMO ORDINE PER LA RAPPRESENTAZIONE E L'INTERROGAZIONE DEI DATI

La creazione di una base di conoscenza (Knowledge Base o KB) in logica di primo ordine è un processo fondamentale per rappresentare la conoscenza riguardante un particolare dominio all'interno di una macchina. Questo processo prevede diversi passaggi:

- 1. Decidere il dominio da rappresentare:** Il dominio può includere aspetti del mondo reale, immaginario o astratto, come numeri e insiemi.
- 2. Selezione delle proposizioni atomiche:** Il progettista deve identificare le proposizioni atomiche che meglio rappresentano il mondo in questione.
- 3. Assiomatizzazione del dominio:** Si definiscono le proposizioni che sono considerate vere nell'interpretazione del dominio. Queste proposizioni costituiscono gli assiomi della KB.
- 4. Definizione delle query:** Vengono definite le query che consentono di verificare se specifiche proposizioni sono conseguenze logiche della KB, ovvero se sono vere in tutti i modelli della KB.

Il sistema, a differenza del progettista, non comprende il significato dei simboli, ma è in grado di determinare se una proposizione è una conseguenza logica degli assiomi presenti nella KB. Il progettista, d'altro canto, interpreta il risultato ottenuto dal sistema in base all'interpretazione intesa della KB.

Nel nostro progetto, abbiamo scelto di rappresentare la conoscenza utilizzando una base di conoscenza in *Prolog*, che si basa sulla logica del primo ordine. Prima di definire la KB, il dataset è stato sottoposto a una fase di preprocessing tramite il *file cleaning.py*. Successivamente, abbiamo selezionato le features da assiomatizzare come fatti nella KB e abbiamo aggiunto regole che consentono all'utente di sottomettere query più complesse.

ESEMPI QUERY

1. L'utente tramite suddetta query può sapere quali sono le stanze che rispettano un determinato intervallo di prezzo. In particolare, può inserire 5 classi di prezzo:

- *top_level* $prezzo > 675$
- *expensive* $prezzo > 200$ and $prezzo \leq 675$
- *medium* $prezzo > 55$ and $prezzo \leq 200$
- *affordable* $prezzo > 40$ and $prezzo \leq 50$
- *economy* $prezzo \leq 40$

OUTPUT :	
	Room
0	178992
1	397875
2	430787
3	494974
4	698687
5	709463
6	761868
7	802843
8	881892
9	1039269
10	1052451
...	

price_range(Room, "affordable")

2. Questa query cerca tutte le camere (Result) che possono ospitare almeno 4 persone e hanno un prezzo inferiore o uguale a \$100.

bookable_rooms(Result,4,100)

OUTPUT: Result	
	23986
	46536
	88130
	94233
	138071
	227032
...	

3. Questa regola definisce che Result sarà una lista di tutte le camere (X) che soddisfano la condizione definita dalla regola big_room(X).

big_rooms(Result)

OUTPUT: Result
763503
1932739
2174955
3557461
4240968
...
...

4. L'utente tramite suddetta query può verificare se due stanze sono simili sulla base del cluster a loro associato. In particolare, se i cluster delle stanze identificate da X e Y risultano uguali, allora le stanze saranno simili.

similar_rooms(X,Y) :- cluster(X,C),cluster(Y,D), C = D.

similar_rooms(40470,55055)
true

5. Questa query ricerca per identificare tutte le camere che sono dotate di una cucina. Se una stanza offre questo servizio, verrà considerata come risultato positivo della query.

amenities(Room,"kitchen")

OUTPUT:

	Room
0	23986
1	40470
2	46536
3	55055
4	59226
...	...
8829	53762510
8830	53764507
8831	53808297
8832	53819768
8833	53830099

6. La regola `stay_range(Room, 2)` definisce che una stanza (Room) soddisfa il criterio se ha un intervallo di permanenza minimo di 2 notti. Quindi, la query chiede di trovare tutte le camere che soddisfano questo criterio.

<code>stay_range(Room, 2)</code>	
OUTPUT:	
	Room
0	23986
1	59226
2	77958
3	79696
4	88130
...	...
6175	53638345
6176	53653590
6177	53749126
6178	53762510
6179	53824628

7. questa query cerca se l'abitazione con ID 46536 è situata nel centro. Se l'abitazione è nel centro, la query restituirà un risultato positivo, altrimenti restituirà un risultato negativo.

<code>is_center(room)</code>
<code>is_center(46536)</code> <code>false</code>

8. questa query cerca se l'host dell'abitazione con ID 23986 ha verificato la sua identità utilizzando il servizio Jumio. Se la verifica è stata effettuata tramite Jumio, la query restituirà un risultato positivo, altrimenti restituirà un risultato negativo

<code>host_verifications(room, "jumio")</code>
<code>host_verifications(23986, "jumio")</code> <code>true</code>

Attraverso l'utilizzo di queste query e l'analisi delle probabilità calcolate, gli utenti sono in grado di prendere decisioni informate e di scegliere la struttura che meglio si adatta alle loro preferenze e alle loro aspettative di qualità. Inoltre, questo approccio fornisce una metodologia basata su evidenze probabilistiche per supportare le decisioni degli utenti nell'ambito della valutazione della qualità delle strutture disponibili.

6 ANALISI E OTTIMIZZAZIONE DEI SERVIZI DI AIRBNB A MILANO ATTRAVERSO TECNICHE DI APPRENDIMENTO AUTOMATICO

In conclusione, possiamo riaffermare che la nostra ricerca si è focalizzata sull'analisi e l'ottimizzazione dei servizi offerti da Airbnb nella zona di Milano, avvalendoci di una serie di tecniche avanzate di apprendimento automatico.

Abbiamo iniziato definendo una base di conoscenza utilizzando la logica di primo ordine, che ci ha permesso di strutturare e organizzare le informazioni relative alle diverse proprietà registrate su Airbnb. Questo approccio ha fornito una solida fondazione per l'analisi dei dati e la scoperta di pattern interessanti.

Successivamente, abbiamo adottato tecniche di apprendimento supervisionato in forma probabilistica, come le belief network, per comprendere le relazioni complesse tra le varie caratteristiche delle camere e dei servizi offerti. Ciò ci ha consentito di effettuare interrogazioni sofisticate sulla nostra base di conoscenza e di ottenere informazioni dettagliate sui servizi, il numero di letti, i bagni e altro ancora, migliorando così la qualità complessiva dell'esperienza dell'utente.

Inoltre, abbiamo impiegato tecniche di apprendimento non supervisionato, come il clustering con l'algoritmo k-means, per identificare camere simili e raggrupparle in base alle loro caratteristiche comuni. Questo ci ha permesso di fornire raccomandazioni personalizzate agli utenti, migliorando ulteriormente l'esperienza di ricerca e prenotazione delle camere su Airbnb.

Per ampliare ulteriormente la nostra analisi, abbiamo considerato l'implementazione di tecniche di elaborazione del linguaggio naturale (NLP) e di part-of-speech tagging per analizzare in modo più approfondito le descrizioni delle camere e identificare le parole chiave rilevanti. Questo avrebbe migliorato la comprensione del contenuto delle descrizioni e consentito una ricerca più accurata da parte degli utenti.

Infine, abbiamo evidenziato l'importanza di questo tipo di studi per migliorare i servizi offerti da Airbnb, enfatizzando l'inclusione di fattori di "user experience" nella nostra analisi. Questo ci ha permesso di andare oltre le semplici caratteristiche delle camere e di considerare anche le preferenze e le abitudini degli utenti, contribuendo così a migliorare l'esperienza complessiva di prenotazione.

La nostra ricerca ha dimostrato l'efficacia delle tecniche di apprendimento automatico nell'analisi e nell'ottimizzazione dei servizi di Airbnb a Milano, offrendo nuove prospettive e possibilità di miglioramento per il settore dell'ospitalità digitale.