Sistema basato su conoscenza e apprendimento automatico per il turismo

Francesco Musci
721269
f.musci10@studenti.uniba.it
https://github.com/muscif/icon2122

Progetto per l'esame di Ingegneria della Conoscenza
A.A. 2021/2022

Indice

1 Introduzione

Un agente intelligente è un sistema che agisce in un certo ambiente capace di utilizzare le sue conoscenze ed abilità per perseguire le sue preferenze ed obiettivi. Lo scopo del progetto è realizzare un prototipo di tale sistema nell'ambito del turismo.

Un sistema di questo tipo può raccogliere conoscenza da un'ontologia ed usarla per l'apprendimento e per la costruzione di nuova conoscenza. Un'ontologia, in informatica, rappresenta la conoscenza in un certo dominio con dati strutturati in maniera formale.

Nel Capitolo 2 verranno descritti l'ontologia e gli strumenti utilizzati.

Nel Capitolo 3 verrà proposto un sistema di apprendimento supervisionato che, basandosi su indicatori geografici e demografici, predice la capacità turistica di ciascun comune italiano.

Nel Capitolo 4 verrà proposto un sistema basato su conoscenza che, basandosi su dati turistici di una città, suggerirà all'utente un alloggio, un ristorante e un'attrazione da visitare in quella città.

Il progetto è scritto in Prolog e Python. Le librerie di Python rilevanti usate sono OSMPythonTools e scikit-learn.

2 Ontologia

L'ontologia utilizzata è OpenStreetMap[1]. Si tratta di uno strumento collaborativo per la mappatura di caratteristiche geografiche. Gli elementi fondamentali di OpenStreetMap sono i nodi, le relazioni, le vie.

Elemento	Descrizione
Nodo	Punto definito da id e coordinate geografiche.
Relazione	Aggregagazione di elementi (anche altre relazioni).
Via	Lista di nodi collegati tra loro a formare un poligono.

Il quarto elemento fondamentale è il tag. Tutti gli elementi possono avere dei tag. Un tag consiste di due elementi, una chiave e un valore. Sono questi tag che costituiscono la vera e propria ontologia di OpenStreetMap. I tag sono disponibili sul sito[2] o mediante OSMOnto[3] (in formato OWL).

I dati possono essere estratti in XML direttamente dal sito principale (i dati riguarderanno l'area visualizzata), da GeoFabrik[4] o tramite API. Il metodo scelto è stato

[1] https://www.openstreetmap.org/about/
[2] https://wiki.openstreetmap.org/wiki/Map_features
[3] https://wiki.openstreetmap.org/wiki/OSMonto
[4] https://download.geofabrik.de/

quest'ultimo, in quanto i dati in XML o di GeoFabrik sono dei download di massa di tutti dati riguardanti una certa area; usando le API si possono scremare a monte i dati necessari. Le API utilizzate sono Nominatim e Overpass.

Nominatim[5][6] è un servizio di geocoding (ottenere delle coordinate geografiche a partire da un indirizzo o un nome di luogo) e reverse geocoding (ottenere l'indirizzo o luogo a partire da delle coordinate geografiche). In questo progetto viene utilizzato per ottenere l'id di un'area, dato in input il suo nome. Questo id verrà poi utilizzato con Overpass.

Overpass[7][8] è un servizio che consente di interrogare OpenStreetMap come se si stesse interrogando un database; infatti, Overpass dispone di un query language apposito. Dato in input l'id di un'area (preso da Nominatim), il tipo di elemento che si sta cercando e i tag, si otterrà l'elenco di tutti gli elementi di quel tipo con quei tag nell'area richiesta.

Le query sono effettuate usando python con la libreria OSMPythonTools, che mette a disposizione le API di Nominatim e Overpass. Segue un esempio di query:

```
id = nominatim.query("Roma").areaId()
q = overpassQueryBuilder(area=id, elementType="node", selector='"amenity"="bar"')
res = overpass.query(q)
```

Questa query restituisce tutti i bar di Roma.

3 Apprendimento supervisionato

L'apprendimento è la capacità di un agente di migliorare il suo comportamento basandosi sull'esperienza.

Nell'apprendimento supervisionato, le istanze di un dataset vengono partizionate in input features e target features. L'obiettivo è prevedere il valore delle target features di istanze non note basandosi sulle input features e sull'esperienza acquisita dalle istanze in cui le target feature sono note.

Una tipologia di apprendimento supervisionato è la regressione. Consiste nel predire il valore di ciascuna istanza delle target feature. Questo valore appartiene al dominio continuo; invece, per i valori discreti, si fa riferimento agli algoritmi di apprendimento supervisionato di classificazione.

[5] https://wiki.openstreetmap.org/wiki/Nominatim
[6] https://nominatim.openstreetmap.org/
[7] https://wiki.openstreetmap.org/wiki/Overpass_API
[8] https://overpass-turbo.eu/

3.1 Obiettivo

L'articolo "Using OpenStreetMap Data and Machine Learning to Generate Socio-Economic Indicators"[9] usa i dati OpenStreetMap con l'apprendimento supervisionato per generare indicatori socio-economici. Prendendo ispirazione da questo articolo, si possono raccogliere dati da OpenStreetMap per prevedere il turismo in un comune, misurato in numero di strutture ricettive.

Per ciascun comune, le input feature sono le seguenti:

Input feature	Descrizione
Occupazione	Occupati sul totale della popolazione (%)
Reddito	Reddito imponibile totale (€)
Cibo	Numero di ristoranti, bar e fast food
Istruzione	Numero di scuole, asili e biblioteche
Natura	Numero di foreste, prati e corsi d'acqua
Tempo libero	Numero di luoghi per il tempo libero
Turismo	Numero di strutture ricettive

I dati sul reddito imponibile e il numero di posti letto sono presi dal database Istat[10]. I dati sull'occupazione per comune sono presi da uno studio de "Il Sole 24 Ore"[11]. Le feature sono state scelte tra le map feature di OpenStreetMap[12].

Il numero di strutture ricettive nel dataset Istat non è riportato nel 12% dei comuni. L'obiettivo è tentare di prevedere questo numero addestrando il modello sul restante 88% dei valori.

3.2 Metodologia

Inizialmente è necessaria una fase di preprocessing, in cui si predispone il dataset affinché l'apprendimento avvenga in modo ottimale.

Per rendere l'apprendimento ottimale, occorre normalizzare e scalare gli attributi numerici. Tutte le feature sono scalate e normalizzate in base alla popolazione del comune (eccetto "Occupazione" che, essendo in percentuale, ne tiene già conto).

Dopo questa fase iniziale di preprocessing, le prime righe del dataset e la heatmap delle correlazioni si presentano così:

[9] https://www.mdpi.com/2220-9964/9/9/498
[10] http://dati.istat.it/
[11] https://www.infodata.ilsole24ore.com/2019/04/16/39185/
[12] https://wiki.openstreetmap.org/wiki/Map_features

Occupazione	Reddito	Cibo	Istruzione	Natura	Tempo libero	Turismo
0.626996	0.769005	-0.214931	-0.416853	-0.178448	-0.131865	0.000331
0.871771	1.022510	0.094709	-0.416853	-0.100418	0.077145	-0.058229
0.743866	0.110689	0.066628	0.083687	-0.124910	-0.122953	-0.062347
-0.882706	-0.382707	0.247532	-0.416853	-0.179710	-0.159582	-0.122374
-0.877477	-0.828957	1.168103	-0.416853	9.332535	-0.210264	0.295995

Figura 1: Prime righe del dataset.

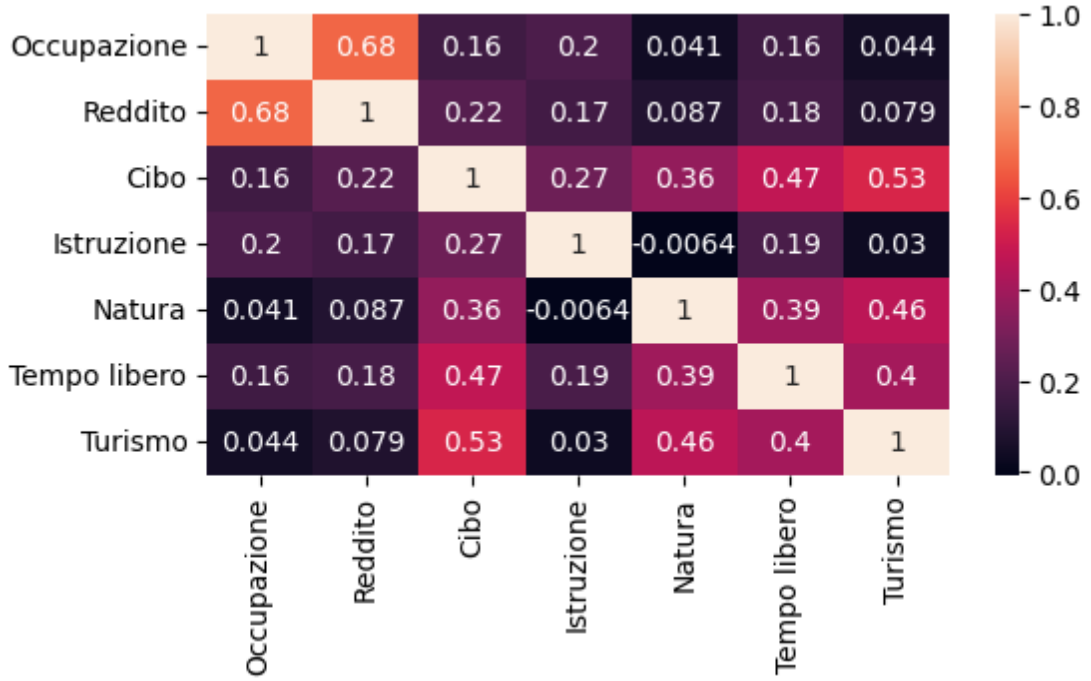

Figura 2: Heatmap delle correlazioni.

Non sembrano esserci correlazioni significative con la target feature, se non una lieve con "Cibo".

Una volta ottenuto il dataset, è necessario addestrare i modelli, che vanno divisi in train e test.

Un problema dell'apprendimento supervisionato è quello dell'overfitting, ovvero: il modello impara regolarità apparenti e correlazioni spurie presenti nel training set, per

cui non impara a predire dati, bensì si sovradatta al training set.

È importante che questo problema venga risolto con la tecnica del *k*-fold cross-validation. Questa tecnica prevede che il training set venga suddiviso in *k* partizioni; il training viene effettuato su *k*-1 partizioni e la validazione quella rimanente. Questo processo viene ripetuto *k* volte per tutti le partizioni, in modo tale che ciascuna partizione venga usata almeno una volta come validation set. Alla fine si otterrà un modello migliore che andrà poi valutato con il test set.

3.3 Risultati

I regressori testati sono la LinearRegression, il RandomForestRegressor, il GradientBoostingRegressor e il DecisionTreeRegressor.

Le metriche scelte sono il Mean Absolute Error (MAE), il Mean Squared Error (MSE) e il Max Error (ME).

Il MAE è la media della somma del valore assoluto delle differenze tra il valore reale e il valore predetto su ogni esempio:

$$L_1 = \sum_{e \in Es} \sum_{Y \in T} \left| Y(e) - \hat{Y}(e) \right|$$

$$MAE = \frac{L_1}{|Es|}$$

è tanto migliore quanto più è vicino a 0.

Il MSE è la media della somma del quadrato delle differenze tra il valore reale e il valore predetto su ogni esempio:

$$L_2 = \sum_{e \in Es} \sum_{Y \in T} \left(Y(e) - \hat{Y}(e) \right)^2$$

$$MSE = \frac{L_2}{|Es|}$$

anche in questo caso il valore è tanto migliore quanto più è vicino a 0, ma con la differenza che gli errori più gravi hanno più peso.

Il ME è semplicemente l'errore peggiore (in termini di L_1).

$$L_\infty = \max_{e \in Es} \max_{Y \in T} \left| Y(e) - \hat{Y}(e) \right|$$

Di seguito, i risultati ottenuti:

Regressore	MAE	MSE	ME
Linear Regression	0.598346	0.561939	3.071198
Random Forest	0.528524	0.484362	2.642037
Gradient Boosting	0.513211	0.441634	2.472786
Decision Tree	0.686238	0.907059	3.410840

Come è possibile vedere, non ci sono grandi variazioni tra i modelli. Il miglior modello è il Gradient Boosting secondo tutte le metriche.

Il modello può essere valutato anche valutando i valori predetti sugli esempi mancanti dal dataset Istat.

Provincia	Comune	Occupazione	Reddito	Popolazione	Cibo	Istruzione	Natura	Tempo libero	Turismo
LO	Abbadia Cerreto	56.382979	3473755	275	3	0	4	0	9.773417
BS	Acquafredda	58.051690	19295002	1518	0	0	0	0	11.571316
CR	Acquanegra Cremonese	63.938974	16680806	1123	3	0	0	1	46.062339
VV	Acquaro	47.980501	16469684	1891	0	0	10	0	15.422963
CL	Acquaviva Platani	43.280977	6689931	891	1	0	0	0	10.094445

Figura 3: Valori di turismo predetti per gli esempi non noti.

Si è scelto di confrontare le medie di ciascun attributo.

	Noti	Non noti
Occupazione	60.630282	58.713755
Reddito	138076982.114611	25572085.239421
Cibo	18.939254	1.791759
Istruzione	2.007676	0.360802
Natura	7.818422	2.741648
Tempo libero	4.524616	0.788419
Popolazione	9628.920990	2082.586860
Turismo	59.568789	35.890977

Figura 4: Medie dei valori noti e non noti.

Si riscontrano grandi differenze tra i valori noti e non noti. Una possibile spiegazione è che i comuni con valori non noti siano comuni poco conosciuti e/o poco popolati e questo influisce sul reddito e il numero di luoghi presenti su OpenStreetMap. Infatti, la media delle strutture ricettive predette è proporzionale agli altri parametri, dimostrando la bontà del modello addestrato.

4 Sistema basato su conoscenza

Un sistema basato su conoscenza (KBS) è un sistema che usa la conoscenza di un dominio per risolvere problemi. Questa conoscenza può essere declinata in due modi: fatti e regole. I fatti sono ciò che la base di conoscenza (KB) dà per vero (ipotesi); le regole descrivono il modo in cui la KB deve ragionare sui fatti.

In questo progetto è utilizzata una KB per un recommender system per il turismo. Il sistema chiede in input all'utente le sue preferenze e le utilizza per suggerirgli un alloggio, un ristorante e un'attrazione. I fatti della knowledge base consistono in un elenco di alloggi, ristoranti e attrazioni ottenuti da OpenStreetMap (un codice in Python raccoglie questi dati e li converte in atomi Prolog). Le regole sono le regole di inferenza logica per determinare quali alloggi sono adatti alle preferenze dell'utente.

```
write("Benvenuto nel sistema di raccomandazione per il turismo!\n"),
write("> Dove desideri andare?\n1. Berlino\n2. Londra\n3. Parigi\n4. Milano\n"),
read(DestinationInput),
(
    DestinationInput == 1 -> Destination = berlino;
    DestinationInput == 2 -> Destination = londra;
    DestinationInput == 3 -> Destination = parigi;
    DestinationInput == 4 -> Destination = milano
),
consult(Destination),
```

Inizialmente, viene chiesto all'utente dove vuole andare. A quel punto viene consultato il corrispondente file contenente i fatti per quella destinazione. Vengono poste all'utente varie domande sulle sue preferenze. Infine, tra i luoghi che rispondono alle esigenze dell'utente, viene selezionato casualmente un alloggio; il ristorante e l'attrazione sono quelli più vicini (in linea d'aria) all'alloggio scelto.

Per ciascuno di questi luoghi, gli attributi sono i seguenti:

Alloggio	Ristorante	Attrazione
Nome	Nome	Nome
Latitudine	Latitudine	Latitudine
Longitudine	Longitudine	Longitudine
Telefono	Telefono	Telefono
Sito	Sito	Sito
E-mail	E-mail	E-mail
Stelle	Cucina	Quota d'ingresso
Accesso a Internet	Asporto	Accessibilità
Accessibilità	Domicilio	
	Orario di apertura	
	Accessibilità	

I luoghi hanno tutti un nome e almeno un modo per contattarli (quello inserito dall'utente sarà lo stesso per i tre luoghi). Per gli alloggi, è necessario che sia noto il numero di stelle.

Segue un esempio di atomo Prolog di un alloggio.

```
accommodation(
    "Hotel Steglitz International",
    52.4562628,
    13.3212357,
    "+49 30 790050",
    "https://www.si-hotel.com/",
    "info@si-hotel.com",
    4,
    "true",
    "true"
).
```

Segue un esempio di regola. Per i ristoranti, l'attributo "Cucina" contiene i vari tipi di cucina che il ristorante offre separati da un punto e virgola (ad es. "japanese;sushi"). Poiché sarebbe difficile partizionare la stringa mediante espressioni regolari in Prolog, un modo banale per cercare i ristoranti che servono la cucina preferita dall'utente è quello di cercare la sottostringa. Con un linguaggio logico come Prolog, il risultato si ottiene in questo modo:

```
write("\n> Inserisci il tipo di cucina che vorresti mangiare.\n"),
read(InputCuisine),
restaurant(Name, _, _, _, _, _, Cuisine, _, _, _, _),
substring(InputCuisine, Cuisine).
```

Questo atomo fa sì che Prolog cerchi tutte le assegnazioni della variabile Cuisine nell'atomo restaurant che rendono questo intero atomo vero.

Alla fine, vengono suggeriti all'utente un alloggio, un ristorante e un'attrazione. Segue un esempio di output.

```
ALLOGGIO
Nome: Hotel Steglitz International
Telefono: +49 30 790050
Sito: https://www.si-hotel.com/
E-mail: info@si-hotel.com
Stelle: 4
Accesso a Internet: true
Accessibilità: true

RISTORANTE
Nome: Peter Pane
Telefono: +49 30 76722130
Sito: false
E-mail: schloss@peterpane.de
Cucina: burger
Asporto: false
Domicilio: true
Orario di apertura: Su-Th 12:00-22:30; Fr,Sa 12:00-23:30
Accessibilità: true

ATTRAZIONE
Nome: Helmut Newton Foundation
Telefono: +49 30 3186 4856
Sito: http://www.helmut-newton.de/
E-mail: info@helmut-newton-foundation.org
Quota d'ingresso: true
Accessibilità: true

Grazie!
```

5 Conclusione

In questo progetto sono state mostrate delle possibili applicazioni di tecniche di apprendimento supervisionato e sistemi basati su conoscenza nell'ambito del turismo.

L'obiettivo del sistema basato su conoscenza era costruire un recommender system che potesse raccomandare un viaggio a un utente.

L'obiettivo dell'apprendimento supervisionato era predire il numero di strutture ricettive nei comuni in cui quel dato non è presente. L'esperimento ha dato risultati soddisfacenti.

Una criticità del sistema che si presta ad estensioni è che le feature di OpenStreetMap vengono prese come conteggi di attributi, nascondendo il fatto che molti di questi hanno più senso se interpretati come area. Nel sistema attuale, per esempio, un piccolo prato e una grande foresta hanno lo stesso peso. Non è stato possibile implementare questa funzione per via di difficoltà tecniche.