Report Tecnico

Progetto: Changes in Latitudes, Changes in Attitudes
Master Big Data Analytics & Social Mining, Università di Pisa

Leonardo Catalano, Luca Fusar Imperatore, Belinda Katherine Graux Bonilla,
Maria Iacono, Giovanni Pievani Trapletti

15 luglio 2022

1 Scraping e Dataset

OBIETTIVO: Creare il nostro dataset estraendo, da un noto sito di recensioni, i dati relativi a tutte le recensioni per i mesi di Giugno, Luglio e Agosto di tutte le strutture presenti sul sito per le regioni Toscana, Veneto e Puglia, oltre alle informazioni relative alle strutture recensite (indirizzo, valutazione media, misure anti-covid implementate, tipologia di struttura ecc). In assenza di API messe pubblicamente a disposizione per scopi di ricerca da parte del sito esaminato, lo scraping è stato effettuato mediante scripts scritti in Python sfruttando prevalentemente le librerie *Selenium*[1] e *BeautifulSoup*[2]. Il lavoro di scraping dei dati si è svolto in tre fasi:

1. Un primo script, ricevuto in input l'URL della pagina del sito di recensioni contenente la lista di strutture di accoglienza per una data regione, scaricava il codice html della pagina per poi estrarne tutti gli URL delle strutture ivi contenuti accompagnati da due codici univoci identificativi: uno per la struttura ed uno per la zona geografica, entrambi ricavati dall'URL della struttura. Il risultato finale è un csv contenente gli URL di tutte le strutture presenti sul sito per quella regione.

2. Un secondo script che, ricevuto in input il csv generato al passo uno, visitava (grazie a Selenium) tutti gli URL ivi contenuti per scaricare le informazioni relative all'albergo ("hotel info") e salvarle in un altro csv.

3. Infine, un terzo script che, procedendo allo stesso modo del precedente, salvava tutte le recensioni (in qualunque lingua) presenti su ogni struttura salvata nel primo csv insieme ad alcune informazioni (non sensibili) relative al recensore.

Abbiamo effettuato un primo pre-processing dei dati, sempre sfruttando script Python, atto a unire in un unico csv alcune tabelle che erano separate (ovviamente per ragioni di tempo gli script sono stati eseguiti in parallelo da diverse macchine) ed a controllare che nessuna struttura fosse stata "persa per la strada". Il risultato finale è stato costituito da tre tabelle per ogni regione: una con tutti gli URL, una con tutte le *hotel info* ed infine una con tutte le recensioni. I campi contenenti l'ID univoco della struttura e quello della località geografica fungono da *foreign keys* per mettere fra loro in relazione le tabelle all'occorrenza.

Si ottiene in tal modo un dataset complessivamente composto di 8.640 strutture ricettive, alberghiere e non alberghiere (fatte salve le c.d. "case vacanza", trattate dalla piattaforma esaminata in sede separata): di queste, 2.935 sono in Veneto, 2.870 in Toscana, 2.835 in Puglia. Per ciascun record sono riportate le informazioni presenti nel sito riguardanti indirizzo, tipologia di struttura, valutazione media, misure anti-covid implementate, ecc.

Le recensioni complessivamente scaricate sono 361.334: 106.061 riferite al Veneto, 125.018 alla Toscana, 130.255 alla Puglia. Quanto all'anno di pubblicazione, 98.312 risalgono al 2017, 90.036 al 2018, 73.943 al 2019, 48.179 al 2020 e 50.864 al 2021 (il calo che si osserva già prima della pandemia è probabilmente dovuto alla presenza di piattaforme concorrenti: non si ritiene comunque che questo andamento infici la possibilità di effettuare valide analisi sugli effetti del Covid.

[1] https://www.selenium.dev/documentation/webdriver/
[2] https://www.crummy.com/software/BeautifulSoup/bs4/doc/

2 Analisi Mobilità e provienza:

2.1 Preprocessing

- Pulizia dei campi della provenienza dell'utente e della posizione della struttura

OBIETTIVO: ricondurre tutti i dati relativi a luoghi geografici inseriti dagli utenti (in testo libero) e dalla piattaforma a liste di due elementi con il nome esatto della località (città, comune) e lo Stato di appartenenza. Si sono esplorati due possibili metodi:

- Metodo veloce, con minore precisione: mediante l'incrocio con dataset tabellari con i nomi di migliaia di città italiane ed estere, identificare tramite corrispondenza tra stringhe la città specificata sulla piattaforma e quindi associare il relativo stato. I dataset sono stati integrati dall'utilizzo della libreria Python *Pycountry*[3] e *Us*[4]. Tale metodo, tuttavia, non consente di identificare località indicate dall'utente in maniera imprecisa o quelle indicate dalla piattaforma stessa con frazioni anziché Comuni.

- Metodo meno veloce, con alta precisione: tramite la libreria *GeoPy*[5] sono state eseguite chiamate API a *OpenStreetMap*[6] specificando il campo di provenienza inserito dall'utente e la posizione della struttura per ricavarne l'indirizzo esatto e le relative coordinate geografiche. Uno script a parte si è reso necessario per estrarre il comune dall'indirizzo ritornato come stringa da GeoPy. Per far questo è stato usato un dataset di Istat con i nomi di tutti i comuni italiani[7].

Con il secondo metodo, quello effettivamente impiegato nella nostra analisi, è stato possibile pulire i campi di provenienza dell'utente e della posizione della struttura, riportandoli per ogni recensione alla medesima forma "Comune, Italia" per le località italiane e "Stato" per le località estere. Non era necessario per le nostre analisi tenere informazione della città di provenienza dell' utente estero; si sono però tenute le coordinate geografiche della città al fine di calcolare accuratamente la distanza percorsa per il viaggio. In questo modo è stato possibile identificare l'esatto Comune di ubicazione per tutte le strutture considerate, e identificare il luogo di provenienza dell'utente in praticamente tutti i casi in cui questo dato è stato inserito nella recensione (241957 su 241253 recensioni in cui il dato è stato fornito). I Paesi di provenienza sono stati poi raggruppati, mediante join con un dataset tabellare, nelle tre categorie "Italia", "Resto d'Europa", "Resto del mondo".

2.2 Calcolo della distanza di ogni viaggio

OBIETTIVO: stimare la distanza che si assume sia stata percorsa dal recensore in occasione del viaggio, calcolata come la distanza in linea d'aria tra le coordinate geografiche della città di provenienza dell'utente e quelle della struttura nella quale e' avvenuto il soggiorno e quindi la recensione.

- METODO: È stata adoperata l'apposita funzione di GeoPy.

2.3 Grafico delle mappe delle regioni

OBIETTIVO: Suddividere il territorio regionale in base alla "categoria turistica prevalente" di ciascun Comune come definita da Istat[8], e fornirne una rappresentazione grafica, indicando la ripartizione percentuale del numero di recensioni per ogni categoria anno dopo anno attraverso una scala cromatica e widget interattivo.

- METODO: La puntuale identificazione del Comune di appartenenza di ogni hotel, descritta nel punto 2.1, ha consentito di ricondurre ciascuna struttura (con le relative recensioni) ad una delle categorie turistiche sopra menzionate. A quel punto,i dati territoriali di ogni comune sono stati scaricati dal portale Istat in formato shapefile, sono stati lavorati in Python tramite la libreria *GeoPandas*[9] con lo scopo di generare aggregati per ogni categoria a tali aggregati

[3] https://pypi.org/project/pycountry/
[4] https://pypi.org/project/us/
[5] https://pypi.org/project/geopy/
[6] https://www.openstreetmap.org/
[7] https://www.istat.it/it/archivio/6789
[8] https://www.istat.it/it/archivio/247191
[9] https://geopandas.org/en/stable/

sono stati associati il numero di recensioni totali e la relativa ripartizione percentuale. . A seguito dell'elaborazione, i dati sono stati esportati come file Geojson e Topojson. Le mappe interattive sono state infine realizzate tramite Altair[10].

Si precisa che, per quanto attiene le categorie turistiche, rispetto alla tassonomia ISTAT sono state operate le seguenti ridenominazioni e/o accorpamenti:

(a) ai "Comuni con vocazione montana" (qui, più brevemente, "Montagna"), sono stati accorpati anche i "Comuni a vocazione montana e con vocazione culturale, storica, artistica e paesaggistica";

(b) ai "Comuni con vocazione marittima" (qui, più brevemente, "Mare"), sono stati accorpati anche i "Comuni a vocazione marittima e con vocazione culturale, storica, artistica e paesaggistica";

(c) ai "Comuni a vocazione culturale, storica, artistica e paesaggistica" (qui, più brevemente, "Cultura"), sono stati accorpati anche i "Comuni a vocazione culturale, storica, artistica e paesaggistica e altre vocazioni";

(d) per i "Comuni del turismo lacuale" ed i "Comuni del turismo termale", si è adottata la terminologia più concisa di "Lago" e "Terme";

(e) sono stati ricondotti alla categoria "Altro" i "Comuni turistici non appartenenti ad una categoria specifica", gli "Altri comuni turistici con due vocazioni", i "Comuni non turistici".

3 Classifica dell'apprezzamento di una struttura

OBIETTIVO: calcolare un indicatore per stilare, anno per anno, una classifica della struttura più apprezzata in ciascuna Regione METODO: L'"apprezzamento" è dato dalla moltiplicazione del voto medio delle recensioni ottenuto dalla struttura in un determinato anno per il rapporto tra il numero delle recensioni da essa ricevute in un determinato anno e il totale delle recensioni per le strutture di quella regione nel medesimo anno.

4 Andamento delle recensioni per tipologia di struttura

Per la specifica analisi, sono state accorpate nell'unica categoria "Hotel" le tipologie descritte dalla piattaforma come "hotel(s)", "small hotel(s)" e "special hotel(s)"; alla categoria "Resort" i "resort(s)" e gli "special resort(s)", alla categoria "BB/Guesthouse" i "BB(s)", gli "special BB(s)", le "guesthouse(s)"; alla categoria "Farmhouse" le "farmhouse(s)" ed i "ranch(es)"; alla categoria "Boutique hotel" i "boutique hotel(s)" e i "lodge(s)".

5 Text Mining Recensioni

5.1 Pre-processing

- Per motivi di privacy e data-ethics sono state rimosse dal dataset delle recensioni le informazioni relative all'utente (nickname, provenienza, ecc) mantenendo soltanto i campi relativi a testo e titolo della recensione ed all' anno in cui è stata effettuata (oltre che ovviamente gli ID relativi alla struttura interessata ed all'area geografica).

- Alla tabella è stata aggiunta una colonna con l'informazione relativa alla lingua in cui è stata scritta la recensione. La lingua è stata riconosciuta automaticamente sfruttando la libreria Python *langdetect*[11].

5.2 Wordclouds

OBIETTIVO: Come prima analisi esplorativa per quanto riguarda la parte di text mining creare delle wordcloud per ogni anno preso in analisi, in modo da vedere se ci sono parole prevalenti diverse in anni diversi. Il tool utilizzato è la libreria Python WordClouds[12]. Le wordclouds sono

[10] https://github.com/altair-viz/altair
[11] https://pypi.org/project/langdetect/
[12] https://amueller.github.io/word_cloud/

state generate prima separatamente per ogni regione e poi a livello aggregato. L'approccio è stato il seguente:

- Per prima cosa sono state selezionate solo le 245.137 recensioni in italiano sfruttando l'attributo *language* precedentemente generato. È stato poi creato un dizionario le cui chiavi erano gli anni presenti nel dataset e i valori le recensioni italiane relative a quell'anno. Le recensioni sono state quindi *tokenizzate*, lemmatizzate e sono state eliminate le stopwords. Per tutti questi passaggi sono stati utilizzati i tool forniti dalla libreria Python per l' NLP *NLTK*[13]

- Dopo un primo tentativo di generazione delle wordcloud basandosi sulla semplice frequenza che, come previsto, non ha dato risultati significativi (infatti il dataset contiene molte di quelle che potremmo chiamare "stopwords di dominio" come 'albergo', 'hotel', 'recensione' che risultavano ovviamente essere le parole più frequenti in ogni anno), si è scelto di tentare un approccio leggermente più raffinato.

- Sono quindi state implementate manualmente delle funzioni per il calcolo del TF-IDF. Le wordclouds sono quindi state generate in base al valore di TF-IDF delle parole calcolato considerando le recensioni complessive come *corpus* e le recensioni relative ad un certo anno come singolo documento. Questo permette di far emergere le parole che hanno contraddistinto un certo anno di recensioni. Quello che emerge negli anni precedenti al covid è la presenza di molto rumore, le parole con alto TF-IDF spesso hanno comunque una frequenza relativamente bassa, tuttavia si può notare la presenza di alcuni eventi caratteristici di quell'anno, spesso termini relativi a concerti ('Robbie Williams', che nel 2017 ha fatto un tour in Italia; 'Aeresmith', 'Visarno', la location dove si tengono i concerti del Firenze Rocks, ecc.) o a tour operator, cooperative agricole ecc. presenti sul territorio in questione. Come sperato però gli anni del covid non lasciano adito a dubbi: quasi tutte le parole nelle wordclouds sono relative alla sfera semantica della pandemia ed, in questo caso, anche la frequenza grezza dei termini è piuttosto alta.

5.3 Emotion Analysis

OBIETTIVO: Aggiungere un ulteriore attributo al nostro dataset che caratterizzi la valenza emotiva di ogni recensione, l'idea è di avere una nuova colonna con, per ogni riga, un vettore contenente lo score delle emozioni associate alla recensione. Sono stati tentati due approcci: uno basato su un *lexicon* sviluppato dall'Università di Pisa: itEM[14], l'altro basato su un tool sviluppato dall'università Bocconi: FEEL-it[15]; quest'ultimo incorpora in una libreria sia un corpus di tweets annotati per Sentiment/Emotion Analysis che un transformer pre-addestrato sull'italiano per generare i word embedding, fornendo quindi un modello pre-addestrato in grado di effettuare sia sentiment che emotion analysis su testo italiano quasi *out of the box*. Per entrambe le analisi sono state utilizzate solo le recensioni in italiano (che sono comunque largamente la maggioranza del dataset).

- Il primo tentativo è stato effettuato con itEM. Le recensioni sono state tokenizzate per parola e pre-processate in modo simile a quanto fatto per le wordcloud. Successivamente sono stati creati i *word embedding* sfruttando l'implementazione di *word2vec* contenuta nella libreria *Gensim*[16]. Infine, calcolando la matrice di similarità fra i word embeddings del nostro corpus e i seeds annotati del lexicon, abbiamo ottenuto lo score emotivo delle singole parole del corpus che è stato poi utilizzato, sfruttando una funzione implementata da noi *ad hoc*, per calcolare lo score emotivo dell'intera recensione. Le emozioni che itEM è in grado di identificare sono: JOY, SADNESS, ANGER, FEAR, TRUST, DISGUST, SURPRISE, ANTICIPATION.

- Nel secondo caso la procedura è stata più semplice. Il modello pre-addestrato era disponibile su *Hugging Faces* ed era dotato di un API piuttosto semplice. Sfruttando l'API abbiamo quindi processato le recensioni ottenendo anche in questo caso un vettore con lo score emotivo di ogni recensione. Le emozioni che FEEL-it è in grado di identificare sono: JOY, SADNESS, FEAR e ANGER.

[13] https://www.nltk.org/
[14] https://github.com/Unipisa/ItEM
[15] https://huggingface.co/MilaNLProc/feel-it-italian-emotion
[16] https://radimrehurek.com/gensim/

Abbiamo quindi proceduto ad un'analisi qualitativa dei risultati per determinare quale delle due metodologie avesse ottenuto un risultato migliore attestando delle performance nettamente migliori nel caso di FEEL-it, che sembrava attribuire alle recensioni score molto più corretti. Abbiamo attribuito questa superiorità del secondo approccio principalmente a due fattori: in primo luogo itEM sembra essere maggiormente pensate per annotare singole parole piuttosto che frasi, visto anche l'utilizzo di un modello *context-independent* per il word embedding come *word2vec*, a differenza di FEEL-it che si basa su un modello *context dependent* come BERT, da cui la necessità di creare noi una funzione per calcolare lo score emotivo della recensione sulla base degli score delle singole parole. In secondo luogo è possibile che i seeds annotati contenuti in itEM fossero di natura molto diversa rispetto ai termini del nostro corpus; FEEL-it, invece, è addestrato su un corpus di tweets risalenti al 2020 [17], quindi il modello non solo è già "avvezzo" al linguaggio informale dei social ma è anche venuto in contatto con termini relativi alla pandemia, il che costituisce un grosso valore aggiunto vista la natura della nostra analisi. Sono quindi alla fine stati utilizzati gli score calcolati mediante la libreria FEEL-it.

5.4 Topic Extraction

OBIETTIVO: Estrarre i topic prevalenti all'interno del dataset di recensioni e studiarne l'evoluzione nell'arco temporale in esame. È stata utilizzata una libreria Python per il *topic modelling*: BERTopic[18]. In breve, BERtopic genera gli embedding per le frasi che riceve in input sfruttando una rete neurale basata sui transformer, la dimensionalità del vettore restituito dalla rete viene quindi ridotta mediante l'algoritmo UMAP e dunque il vettore viene dato in input all'algoritmo di clustering HDBSCAN che clusterizza i documenti. La topic extraction per i cluster avviene quindi sfruttando il c-TF-IDF che ci permette di ottenere le parole più rilevanti per ogni cluster. L'approccio da noi seguito è stato il seguente:

- Sono stati fatti numerosi tentativi sperimentando sia diversi iper-parametri, sia dataset diversi. Infatti si è provato a far girare il modello sia sulle recensioni in italiano ed inglese (largamente le due lingue prevalenti all'interno del dataset, circa l'86% delle recensioni sono in una di queste due lingue) che su quelle solo in italiano (circa il 67% del dataset), usando rispettivamente un *embedding model* pre-addestrato multilingue ed uno per l'italiano. Alla fine i risultati più soddisfacenti sono stati ottenuti a partire dalle recensioni solo in italiano.
- BERTopic prende in input frasi, è stato necessario quindi tokenizzare per frase le recensioni (fornire al modello la recensione intera avrebbe comportato una grande perdita di informazione visto che gran parte del testo sarebbe stato troncato al momento dell'embedding). Questo però ci ha messo di fronte a problemi di memoria in quanto il nostro dataset era adesso costituito da più di un milione di frasi e, pur sfruttando la RAM extra offerta da COOLAB+, è stato necessario ridurre la dimensione del dataset. Il dataset è quindi stato ridotto di circa il 40%. Si è però cercato in questa fase di preservare le recensioni contententi parole chiave ai fini della nostra analisi come: 'covid', 'pandemia', ecc per non perdere informazioni preziose. Inoltre, dal momento che il numero di recensioni presenti nel dataset non era lo stesso per ogni anno (con un trend discendente dal 2017 al 2021) si è scelto di ribilanciare il dataset in modo tale che il numero di recensioni presenti per ogni anno fosse uguale al numeri di recensioni del 2021.

Alla fine BERtopic ci ha permesso di riconoscere 100 topic prevalenti all'interno del nostro dataset di recensioni, un sottoinsieme di essi (quelli più interessanti ai fini della nostra analisi) è stato quindi studiato nella sua evoluzione temporale pre e post avvento della pandemia.

6 Altro

Le altre analisi di natura esplorativa o di statistica descrittiva sono state fatte utilizzando prevalentemente Pandas[19] e, occasionalmente per le analisi preliminari, Excel.

[17] https://aclanthology.org/2021.wassa-1.8.pdf
[18] https://maartengr.github.io/BERTopic/index.html
[19] https://pandas.pydata.org/