UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Dipartimento di Scienze Fisiche, Informatiche e Matematiche

Corso di Laurea in Informatica

Raccolta ed Analisi di Dati Relativi a Pubblicazioni Scientifiche e Relative Conferenze

Nicolay Osalchuk

Tesi di Laurea

Relatore:
Prof. Riccardo Martoglia

Correlatori:
Prof. Luca Bedogni
Prof. Giacomo Cabri

Anno Accademico 2020/2021

RINGRAZIAMENTI

Ringrazio il prof. Riccardo Martoglia, il prof. Luca Bedogni e il prof. Giacomo Cabri per la loro disponibilità, per gli indispensabili consigli e per avermi fornito ogni materiale utile alla stesura dell'elaborato.

Un ringraziamento speciale va anche alla mia famiglia e ai miei amici che in questi anni mi hanno sempre sostenuto e sopportato.

PAROLE CHIAVI

Analisi dati
Pandas
Python
Conferenza
Citazione

INDICE

Introduzione **8**

1. Data Science **10**
1.1 Cos'è la data science 10
1.1.2 Il data scientist 10
1.2 Big data 11
1.2.2 Crescita dei volumi di dati 12
1.3 Data analysis 13
1.3.2 Raccolta e pulizia dei dati 14
1.3.3 Elaborazione dei dati 14
1.3.4 L'analisi 14

2. Tecnologie e strumenti **16**
2.1 Python 16
2.2 Pandas 17
2.3 Seaborn 18
2.4 URLlib 18
2.5 BeautifulSoup 18
2.6 NumPy 19
2.7 Jupyter notebook 19

3. Progettazione **21**
3.1 Obiettivo 21
3.2 Pulizia dei dati 22
3.3 Elaborazione dei dati 23
3.3.2 L'evoluzione delle conferenze 24
3.4 Estrazione dei dati turistici 29
3.4.2 Pulizia dati turistici 30
3.4.3 Elaborazione dei dati turistici 31
3.4.4 Informazioni turistiche sugli stati 33
3.5 Correlazione 34
3.5.2 Correlazione per città 35
3.5.3 Correlazione per stato 35
3.5.4 Correlazione per singola conferenza 36
3.6 Possibili implementazioni future 36

4. Implementazione **38**
4.1 Pulizia dati 38
4.2 Elaborazione dati sulle conferenze 39
4.2.2 L'evoluzione delle conferenze 39
4.3 Estrazione dei dati turistici 44

4.3.2 Informazioni turistiche sugli stati 46
4.4 Correlazione 47
4.4.2 Correlazione per singola conferenza 48

5. Risultati 51
5.1 Classifiche 51
5.2 Risultati della correlazione 54
5.3 Analisi conferenze 60

Conclusioni 66

Introduzione

L'evoluzione tecnologica e informatica nel corso degli ultimi anni ha portato allo sviluppo di molti sistemi per la raccolta di grandi quantità di dati e di conseguenza è nata la necessità di analizzare le informazioni ottenute, con il fine di estrarre quelle più utili che suggeriscono conclusioni e supportino le decisioni strategiche aziendali e non solo. Inevitabilmente sono nate tante tecnologie con lo scopo di facilitare la raccolta, la manipolazione e l'analisi dei dati.

Nel presente elaborato si analizzeranno alcuni dati relativi a pubblicazioni scientifiche partendo da un dataset con più di 2 milioni di articoli di conferenza.
La prima fase fondamentale è la pulizia dei dati, che consiste nell'individuare e eliminare i dati 'sporchi'. Dopodiché segue una fase di elaborazione delle informazioni, quindi partendo dal dataset principale vengono individuati ed estratti i dati più utili per poi essere raccolti in diverse strutture dati, principalmente dataframe, che verranno poi riutilizzati in seguito per lo scopo finale. Una volta elaborati i dati sulle conferenze, si andrà ad estrarre il luogo di ogni edizione. Nella fase successiva si andrà a raccogliere dati turistici relativi alle città e stati per verificare se esiste un correlazione tra il successo di una conferenza e il livello di attrazione turistica del luogo dove si è tenuta.

La struttura di questa tesi è organizzata in 5 capitoli il cui contenuto verrà di seguito illustrato. Nel primo capitolo troviamo una breve introduzione al data science, questo può essere utile per farsi un'idea di cosa vuol dire fare data analytics.
Il capitolo 2, invece, descrive in modo approfondito le tecnologie e strumenti utilizzati, in particolare il linguaggio di programmazione utilizzato e le sue librerie.
Il capitolo 3 tratta tutte le fasi di progettazione, quindi si andrà a discutere nel dettaglio ogni passaggio fatto, specificando da quali dati si è partiti, le operazioni eseguite e quali risultati si è ottenuto. In questo capitolo, perciò, verranno illustrati alcuni esempi dei diversi dataframe utilizzati specificando per ognuno di loro il modo in cui hanno contribuito a raggiungere l'obiettivo finale.
Il capitolo 4 invece è quello più tecnico perché tratta tutta la parte di implementazione. Verranno perciò mostrate e commentate le principali parti di codice, in particolar modo quelle funzione citate nel capitolo 4.
Infine il capitolo 5, quello dedicato ai risultati, dove si potranno osservare i grafici ottenuti e di cui si è discusso nei precedenti capitoli, in particolar modo i grafici che mostrano i risultati della correlazione. Inoltre saranno analizzate più nello specifico alcune conferenze e le sue edizioni confrontando quelle migliori con le peggiori e commentando il grafico dell'andamento delle citazioni nel tempo.

Parte I

Il caso di studio

Capitolo 1

Data Science

Prima di iniziare a parlare del progetto è necessario conoscere alcuni concetti che sono alla base del lavoro svolto. Nel seguente capitolo dunque seguirà una breve introduzione alla data science, big data e data analysis.

1.1 Cos'è la data science

La data science è una disciplina abbastanza nuova in quanto si basa sulla collezione di grandi quantità di dati, attività che in passato era abbastanza difficile se non impossibile da svolgere per via dei limiti tecnologici e informatici. Questi settori, però, soprattutto negli ultimi anni sono stati soggetti di una grande evoluzione che hanno portato alla nascita di tanti sistemi che immagazzinano, in modo automatizzato, una grande mole di dati. Oggi il mondo è sommerso da informazioni di ogni tipo che acquisiscono più o meno valore in base a come vengono utilizzate. In ogni contesto una corretta decisione deve prendere in considerazione tutti gli elementi e i dati rilevanti a disposizione e per questo motivo nasce e acquista sempre più importanza la data science.

Possiamo definire la scienza dei dati[1] come l'insieme di principi metodologici e di tecniche multidisciplinari volto a interpretare ed estrarre conoscenza dai dati attraverso la relativa fase di analisi da parte di un esperto.
Il cuore della data science, ovviamente, sono i dati, ma affinché i risultati finali siano affidabili e precisi è necessario che i dati raccolti siano corretti e puliti. Per questo motivo chi si occupa dei dati deve avere una formazione matematica, informatica e statistica in quanto i metodi della scienza dei dati si basano su tecniche provenienti da varie discipline.
Nella gestione di un'azienda elaborare una strategia di business affidabile ed efficace è fondamentale e non può prescindere dai dati. Ad oggi un'azienda può facilmente accedere a una grande quantità di dati ma c'è bisogno di chi analizza ed estrae informazioni ed inevitabilmente nel corso degli ultimi anni si è diffusa sempre di più la figura del data scientist perché nonostante molte operazioni sui dati vengono realizzate dall'algoritmo rimane fondamentale il lavoro umano svolto prima, tanto che il ruolo del data scientist è stato definito dalla Harvard Business Review come la professione più attraente del XXI secolo.

1.1.2 Il data scientist

Il data scientist[2] è una figura professionale che ricopre un ruolo chiave nell'analisi dei dati. Essi infatti sono esperti in grado di ricavare insight da enormi quantità di dati, strutturati e non strutturati, allo scopo di aiutare a definire o soddisfare esigenze specifiche e obiettivi aziendali. Il ruolo del data scientist nell'analisi dei dati sta diventando sempre più importante in quanto le aziende si affidano sempre più a Big Data e analytics a supporto dei processi decisionali. Il compito principale di un data scientist è l'analisi dei dati, un processo che inizia con la raccolta dei dati e termina con le decisioni aziendali basate sui risultati finali. Ogni settore ha il proprio patrimonio di big data da analizzare, a partire dai e-commerce, dove i data scientist possono aiutare le aziende a migliorare il servizio clienti, riconoscere le tendenze e sviluppare servizi o prodotti, fino ai social-network per definire campagne pubblicitarie mirate o nel settore della sanità per migliorare i servizi sanitari e garantire sicurezza e conformità.

Come avrete notato nel descrivere cos'è la data science e chi è il data scientist è impossibile non citare big data a data analysis, due concetti fondamentali che andiamo ad approfondire nelle sezioni successive.

1.2 Big data

In statistica e informatica, la locuzione inglese big data[3] indica genericamente una raccolta di dati informativi così estesa in termini di volume, velocità e varietà da richiedere tecnologie e metodi analitici specifici per l'estrazione di valore o conoscenza. Il termine è utilizzato dunque in riferimento alla capacità di analizzare ovvero estrapolare e mettere in relazione un'enorme mole di dati eterogenei, strutturati e non strutturati, grazie a sofisticati metodi statistici e informatici di elaborazione, allo scopo di scoprire i legami tra fenomeni diversi (ad esempio correlazioni) e prevedere quelli futuri.
Non esiste una soglia di riferimento prestabilita in termini di dimensione oltre la quale è lecito parlare di Big Data, ma il volume dei dati deve essere correlato alla capacità del sistema di acquisire le informazioni così come arrivano dalle differenti sorgenti, quindi, un sistema diventa big quando aumenta il volume dei dati e allo stesso tempo aumenta la velocità/flusso di informazioni che il sistema deve poter acquisire e gestire per secondo.
In uno studio del 2001, l'analista Douglas Laney aveva definito il modello di crescita come tridimensionale, ovvero che con il passare del tempo aumentano volume, velocità e varietà dei dati. In molti casi questo modello è ancora valido, nonostante esso sia stato successivamente esteso.
Nel modello chiamato delle 3V, le tre parole chiavi hanno questo significato:

1. Volume: ovvero la quantità di dati raccolti ogni secondo da diverse fonti, come sensori, social network o database.
2. Varietà: si riferisce alla diverse tipologie di dati che vengono raccolti, prima dell'avvento dei big data infatti, i dati raccolti era tutti omogenei ma per avere risultati più precisi è necessario tenere in considerazione ogni tipo di dato.
3. Velocità: si riferisce alla velocità con cui i nuovi dati vengono generati.

Con il passare del tempo sono state introdotte altre V:

4. Veridicità: nasce in seguito per il continuo aumento della velocità e della varietà di dati. Considerando infatti la grande velocità con cui vengono raccolti i dati e la loro varietà è possibile che ci sia qualche errore di lettura, per questo motivo può risultare utile associare ad ogni dato raccolto un indice di veridicità in modo che un dato corretto sia più significativo di una lettura sporca.
5. Valore: si riferisce alla capacità di trasformare i dati in valore.

1.2.2 Crescita dei volumi di dati

Per capire quanto i big data sono ritenuti interessanti dalle aziende e quanto hanno investito negli ultimi anni su questa tecnologia, per la gestione e l'analisi dei dati, è sufficiente osservare il grafico nella figura 1.1 che indica la crescita del volume di dati che ci circonda dal 2005 ad oggi.

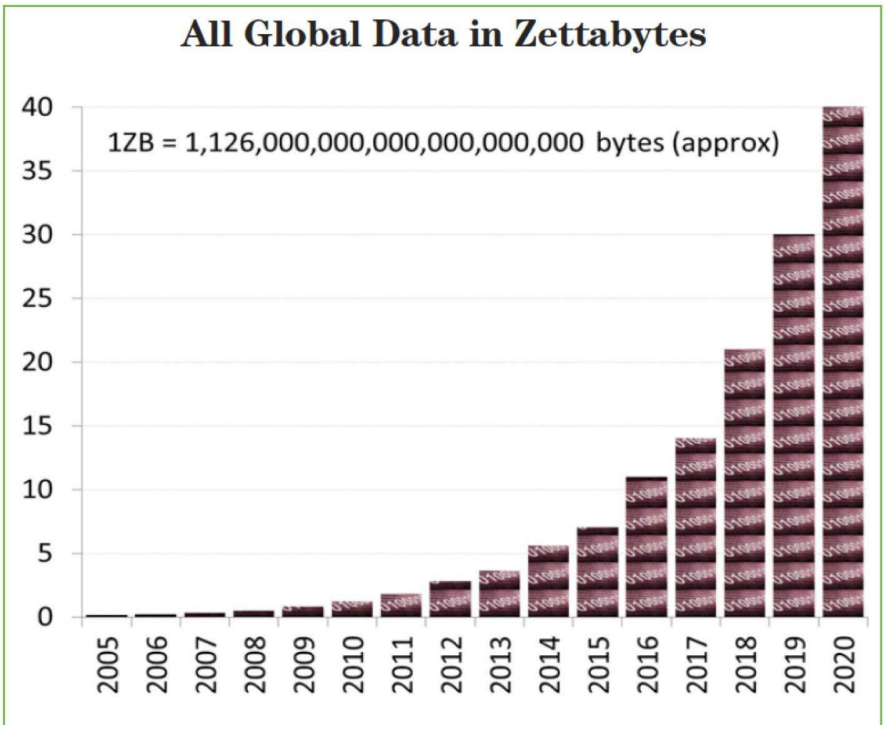


Figura 1.1 all global data[1]

Salta subito all'occhio la crescita esponenziale ma per rendersi veramente conto della quantità di dati raccolti ogni minuto bisogna soffermarsi un attimo sulla figura 1.2 che mostra solo alcuni dei dati che generiamo ogni qualvolta che utilizziamo uno smartphone o un qualsiasi dispositivo collegato ad internet.

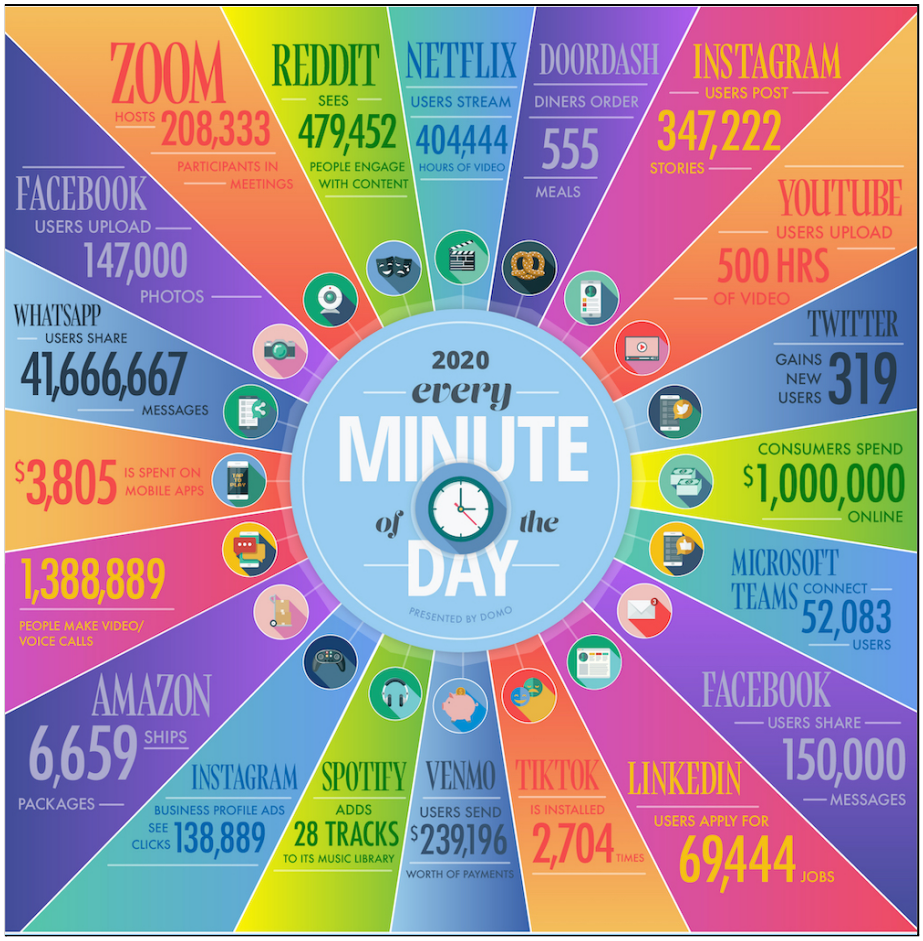


Figura 1.2 data never sleep[2]

Ogni volta che guardiamo un film su Netflix, accediamo ad un social network per pubblicare un post, facciamo un acquisto su un e-commerce vengono salvate delle informazioni, ma non solo, infatti oggi siamo circondati di sensori e telecamere che possono raccogliere migliaia di dati ogni secondo. Tutte queste informazioni raccolte valgono poco se non utilizzate nel modo corretto, per questo motivo diventa fondamentale saper fare l'analisi dei dati che avremo modo di approfondire nella sezione successiva.

1.3 Data analysis

L'analisi dei dati[4] è un processo fondamentale per la data science che comporta la pulizia, trasformazione, e modellazione di dati con il fine di evidenziare informazioni che

suggeriscano conclusioni e supportino le decisioni strategiche aziendali. Questo processo si può dividere in diversi fasi interattive, in quanto il risultato delle fasi successive possono portare a rivedere le fasi precedenti.

1.3.2 Raccolta e pulizia dei dati

La prima fase fondamentale è la raccolta dei dati che consiste nel raccogliere più informazioni possibili da diversi tipi di fonti come sensori, telecamere, file excel, database o semplici fogli di testo. Raccogliere diversi tipi di informazioni permette di avere più dati da analizzare e quindi risposte più precise, ma bisogna aggiungere una fase di elaborazione per organizzare i dati per l'analisi. Organizzare significa inserire dati eterogenei in una struttura dati comune per tutti in modo da facilitare l'accesso per la successiva esplorazione.
Dopo aver strutturato i dati, molto spesso è necessario una pulizia che è un processo di prevenzione e correzione di eventuali errori per migliorare la qualità delle informazioni da utilizzare per la successiva analisi. I dati possono contenere errori, essere duplicati o incompleti per via della grande varietà e velocità con cui vengono acquisiti. Ci sono diversi tecniche che permettono di individuare molti tipi di errori. Una volta ottenuto un set di dati pulito si può passare allo step successivo.

1.3.3 Elaborazione dei dati

Durante l'elaborazione vengono effettuati una serie di azioni per trasformare ed estrarre i dati in un formato output appropriato per un utilizzo successivo. Spesso infatti i dati non sono subito pronti per essere utilizzati, a volte sono necessarie alcune di queste operazioni:

- validazione: per garantire che i dati forniti siano corretti;
- conversione: quando è necessario convertire i dati in un altro formato o lingua;
- ordinamento: a volte è necessario disporre gli elementi in una sequenza ordinata;
- riepilogo: vengono utilizzate per riassumere una serie di osservazioni;
- aggregazione: è un alto livello di dati che vengono acquisiti combinando dati individuali[5];
- rapporti: informazioni calcolate, dati di riepilogo o elenchi di dettagli;
- classificazione: ovvero separare i dati in varie categorie.

Durante questa fase, quindi, possono essere generati alcuni statistiche descrittive come la media o la mediana, i dati possono essere raggruppati o ordinati in base ad alcuni criteri e mostrati in formato grafico o tabellare, oppure possono essere dati in pasto ad algoritmi di machine learning o intelligenza artificiale per generare un output.

1.3.4 L'analisi

L'ultima fase è l'analisi che consente di trasformare i dati elaborati in informazioni preziose che supportino le decisioni strategiche aziendali. Si possono distinguere principalmente due tipologie di analisi:

1. analisi descrittiva: risponde a domande tipo "cos'è successo?" o "quante volte?", quindi rappresenta il passato e il presente;
2. analisi predittiva: nasce grazie agli avanzamenti del Machine Learning e delle tecniche di Intelligenza Artificiale e consente di rispondere a domande quali "è più possibile che si verifichi l'evento A o l'evento B?" o "quante sono le possibilità che si verifichi l'evento C?". Insomma, è in grado di fare previsioni simulando situazioni o scenari possibili aiutando quindi a prendere decisioni o a prevedere risultati.

Da questi due tipi di analisi sta venendo fuori una terza tipologia, l'analisi prescrittiva che attraverso l'utilizzo di sistemi di machine learning e intelligenza artificiale, addestrati ed utilizzati in modo diverso rispetto alle analisi predittive, sono in grado di suggerire comportamenti o azioni per arrivare ad un risultato o far in modo che si verifichi un determinato evento.

Il processo di analisi è senza dubbio la parte più importante di un progetto di data science, e ci si aspetta che restituisca informazioni precise e affidabili per questo motivo è spesso necessario tornare indietro alle fasi precedenti per raccogliere altri dati o per correggere errori che sono venuti fuori solo nell'ultima fase. Il processo finisce con l'interpretazione e la visualizzazione dei dati ottenuti in output. Si può scegliere il modo di comunicare l'analisi tramite grafici, tabelle o semplicemente a parole.

Capitolo 2

Tecnologie e strumenti

In questa seconda parte andremo ad analizzare gli strumenti e le tecnologie utilizzate, in particolare vedremo il linguaggio di programmazione Python e alcune sue librerie utili al fine del progetto.

2.1 Python

Nella sezione 1.1 parlando di data science abbiamo detto che chi si occupa dei dati deve avere una formazione matematica, informatica e statistica in quanto i metodi della scienza dei dati si basano su tecniche provenienti da varie discipline. Una delle competenze informatiche che un data scientist deve sicuramente avere è la padronanza di un linguaggio di programmazione per la manipolazione dei dati. Un linguaggio di programmazione[6] in informatica è un linguaggio dotato di un insieme di regole per scrivere programmi per computer, ovvero un insieme di istruzioni che a partire da un insieme di dati di input, applicando una funzione di trasformazione descritta, appunto, dalle istruzioni, produca un insieme di dati di output. In un linguaggio di programmazione si possono distinguere:

- l'insieme delle parole chiavi: che sono un insieme di parole che hanno un significato particolare per chi deve tradurle in istruzioni eseguibili o deve eseguirle, e a ciascuna delle quali corrisponde un'azione ben definita;
- i caratteri speciali: un insieme di caratteri con significati particolari;
- un insieme di regole sintattiche: un modo di mettere assieme le parole così da formare frasi per indicare le azioni che devono essere compiute.

Oggi abbiamo a disposizione tantissimi linguaggi di programmazione, ognuno ha i suoi pro e contro e le proprie caratteristiche, per questo motivo è importante scegliere quello più adatto per l'attività che si vuole svolgere.
Per l'analisi dei dati i linguaggi di programmazione più indicati sono R, Python e SAS, dei quali andremo ad analizzare alcuni aspetti.
SAS è uno strumento di analisi dei dati molto utilizzato soprattutto nelle grandi società. R è un linguaggio di scripting che offre un'ampia varietà di tecniche per l'elaborazione statistica a grafica. Python, invece, è un linguaggio di uso generico, molto utilizzato in particolare nella comunità della data science. Il primo aspetto fondamentale da tenere in considerazione è il costo, infatti mentre Python e R sono completamente gratuiti, per utilizzare SAS è necessario un abbonamento dal costo non indifferente. Per quanto riguarda l'apprendimento hanno tutti una curva di apprendimento rapida, in particolare python è noto per la sua semplicità nel mondo della programmazione. Tutte e tre i linguaggi hanno buone capacità di gestione dei dati mentre R e Python hanno capacità grafiche molto avanzate a differenza di SAS che nonostante di recente abbia migliorato la sua capacità grafica non si avvicina al livello degli altri due a causa anche della documentazione non adeguata, ma in compenso ha un servizio di

assistenza clienti su cui R e Python non possono contare, ma possono comunque ovviare il problema grazie alla grande community online. Come già detto le grandi aziende utilizzando principalmente SAS che resta il leader di mercato a livello globale, ma R e Python sono le opzioni migliori per start-up e le richieste di lavoro in cui vengono richiesti questi linguaggi sono in grande aumento negli ultimi anni come mostra il grafico nella figura 2.1.

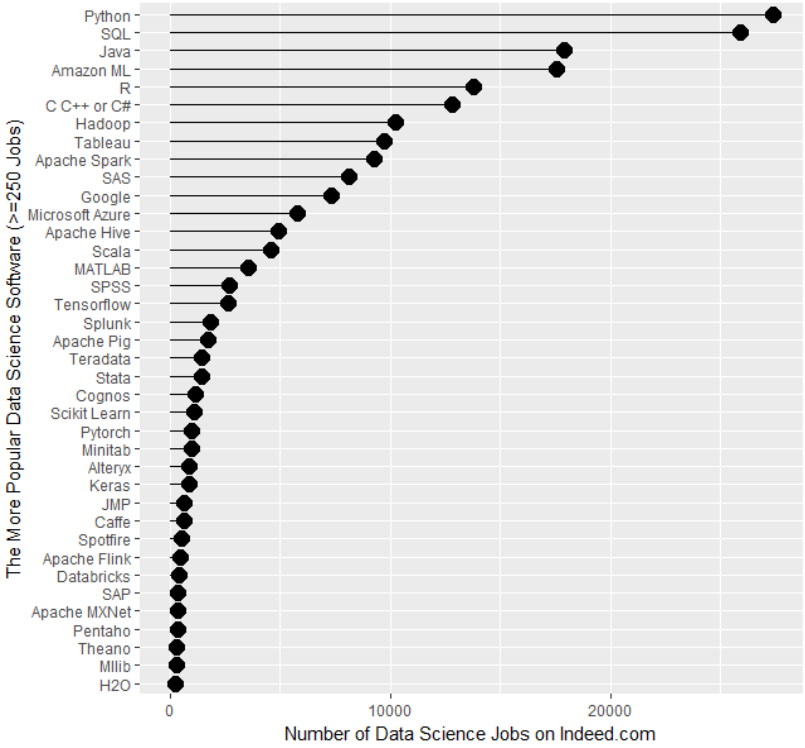


Figura 2.1 number of data science jobs[3]

La scelta finale è ricaduta su python in quanto ha un'attiva community con vasta selezione di librerie e risorse che ha senso usare con tecnologie emergenti come data science. Nelle prossime sezioni vedremo alcune delle librerie utilizzate in questo progetto.

2.2 Pandas

Pandas[7] è una libreria software scritta per il linguaggio di programmazione Python per la manipolazione e l'analisi dei dati. In particolare, offre strutture dati e operazioni per manipolare tabelle numeriche e serie temporali.
Pandas gestisce principalmente due tipi di strutture dati, series e dataframe. Il modo migliore per pensare alle strutture dati di pandas è come contenitori flessibili per dati di dimensioni inferiori. Ad esempio, dataframe è un contenitore per Series e Series è un contenitore per scalari.
Pandas fornisce tantissimi funzionalità per l'elaborazione dei dati, tra cui:

- gestione dei dati mancanti, fondamentale quando si ha che fare con grandi quantità di dati raccolti molto velocemente;
- modificare la dimensione del dataframe inserendo o eliminando colonne;
- funzione "group by" che permette di raggruppare dati con caratteristiche in comune;

- possibilità di convertire un dataframe in una struttura dati di Python;
- possibilità di unire due data set in uno;
- strumenti di I/O per il caricamento di file tipo CSV, excel, basi di dati e altro;
- possibilità di ordinare il dataframe;
- semplicità nell'esecuzione di operazioni di indicizzazione e aggregazione di dati;
- semplicità nella visualizzazione dei risultati delle operazioni;
- semplicità nell'esecuzione di operazioni numeriche e statistiche.

Pandas è in grado di gestire molte altre funzionalità che non sono state citate, non a caso è considerato uno degli strumenti più potenti per l'analisi dei dati.
Possiamo definire pandas il cuore del progetto, infatti quasi tutte le operazioni sui dati sono state fatte utilizzando le varie funzioni che mette a disposizione, dalla pulizia dei dati al calcolo della correlazione.

2.3 Seaborn

Seaborn è un libreria per creare grafici statistici in Python che si integra perfettamente con le strutture dati di pandas. Dietro le quinte seaborn utilizza matplotlib, un'altra libreria di Python per la creazione di visualizzazioni statiche, animate e interattive. Nel modulo seaborn sono presenti diverse funzionalità, facili da usare, per rappresentare graficamente i dati. Una corretta rappresentazione grafica dei dati permette spesso di estrarre informazioni che difficilmente si sarebbero potute ottenere attraverso una normale rappresentazione tabellare.
Nel progetto si è fatto ampio uso di seaborn per rappresentare i risultati finali ottenuti.

2.4 URLlib

Urllib è una libreria di python tramite la quale puoi accedere a siti web, scaricare e analizzare dati ed eseguire qualsiasi richiesta GET e POST. Urllib è composto dai seguenti moduli: request, error, parse e robotparser. Il modulo utilizzato per il progetto è request che viene utilizzato principalmente per l'apertura e il recupero di URL. Una volta recuperata la pagina desiderata si può accedere al contenuto e ai metadati della pagina. Nel progetto il modulo request è stato utile per accedere al "content-length", ovvero la dimensione della pagina richiesta. Più avanti vedremo di quali pagini e per quale motivo.

2.5 BeautifulSoup

BeautifulSoup è una libreria di Python che permette di fare web scraping . Il web scraping è una tecnica che consiste nell'estrazione di informazioni da una pagina web in maniera

automatizzata. Questo può essere fatto sia attraverso un software o con un linguaggio di programmazione.
Insieme a beautifulSoup spesso si utilizza la libreria requests che serve per eseguire richieste HTTP in Python. Una volta ottenuta la pagina web è possibile estrarre le informazioni di cui si ha bisogno. Nel progetto la libreria è stata utilizzata per estrarre informazioni sulle conferenze analizzate.

2.6 NumPy

NumPy[8] è una libreria open source per il linguaggio di programmazione Python, che aggiunge supporto a grandi matrici e array multidimensionali insieme a una vasta collezione di funzioni matematiche di alto livello per poter operare efficientemente su queste strutture dati. La libreria NumPy consente di lavorare con vettori e matrici in maniera più efficiente e veloce di quanto non si possa fare con le liste e le liste di liste (matrici). NumPy mette a disposizione un gran numero di funzioni tra cui:

- mean(): calcola la media di un vettore o matrice;
- sum(): calcola la somma di un vettore o matrice;
- std(): calcola la deviazione standard;
- min(): trova il minimo nel vettore o matrice;
- max(): trova il massimo;
- ndim: dimensione del vettore o matrice.

NumPy viene utilizzato nel progetto spesso per estrarre il minimo o il massimo da una matrice o per calcolare la media ma soprattutto per il calcolo della correlazione.

2.7 Jupyter notebook

Jupyter notebook è un'applicazione web open source che permette di creare e condividere documenti che contengono sia codice che testo narrativo. Viene utilizzato per molti tipi di attività come pulizia, trasformazione e visualizzazione dei dati, machine learning e molto altro. Jupyter supporta oltre 40 linguaggi di programmazione, tra questi ovviamente Python.
Questo notebook risulta molto utile nell'analisi dei dati perché permette di eseguire codice e allo stesso tempo di commentare i risultati ottenuti.
Per il progetto jupyter si è rivelato molto utile perché mi ha permesso di unire in un unico documento la parte di elaborazione con la parte di analisi dei dati ottenuti dopo l'elaborazione, senza dover ogni volta eseguire il codice in un ambiente di sviluppo per poi riportare i risultati in un documento di testo tipo word. Una volta concluso il lavoro il documento può essere esportato in formato pdf, html o LaTeX e condiviso con chiunque.

Parte II

Progettazione, implementazione e risultati

Capitolo 3

Progettazione

In questo capitolo vengono descritte tutte le fasi dell'analisi, dalla pulizia e elaborazione dei dati, all'estrazione di altre informazioni fino alla fase finale di correlazione. Andremo però innanzitutto a definire l'obiettivo che finora è stato solo accennato nell'introduzione.

3.1 Obiettivo

L'obiettivo dello studio è stabilire se esiste una correlazione tra il successo di una conferenza e il luogo dove si è tenuta. Ha senso pensare che una conferenza fatta in una città turistica abbia maggiore successo rispetto ad una città meno attraente da questo punto di vista, e se sì in quale misura? L'analisi condotta risponderà a questa domanda ma prima sono necessarie alcune precisazioni. Innanzitutto dobbiamo definire come misuriamo il successo di una conferenza e per farlo dobbiamo introdurre il dataset di partenza dal quale è iniziata l'analisi. Lo studio è partito da una raccolta di due milioni e mezzo di articoli di conferenza, e per ognuno di essi sono stati raccolti principalmente l'anno di pubblicazione, la conferenza a cui appartiene e il numero di citazioni ricevute. Proprio il numero di citazioni è il parametro che abbiamo scelto per misurare il successo, infatti per ogni conferenza abbiamo raccolto tutti gli articoli ed è stata calcolata la media delle citazioni e questo numero ne determinava la qualità. Per raggiungere l'obiettivo sono state affrontate diverse fasi che possiamo descrivere attraverso un diagramma a blocchi:

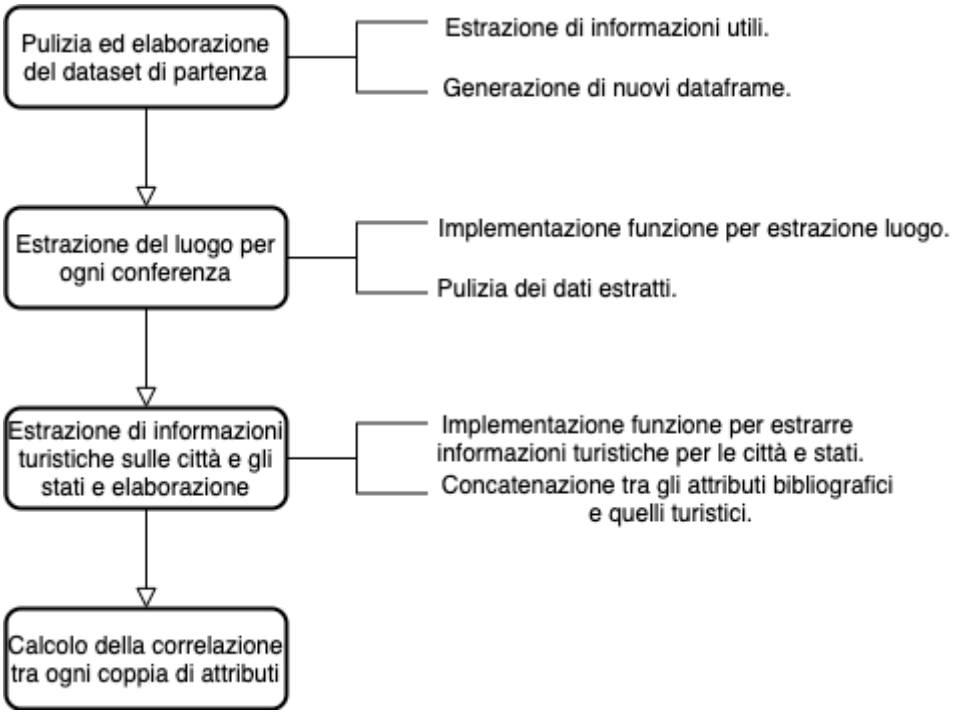


diagramma 3.1

3.2 Pulizia dei dati

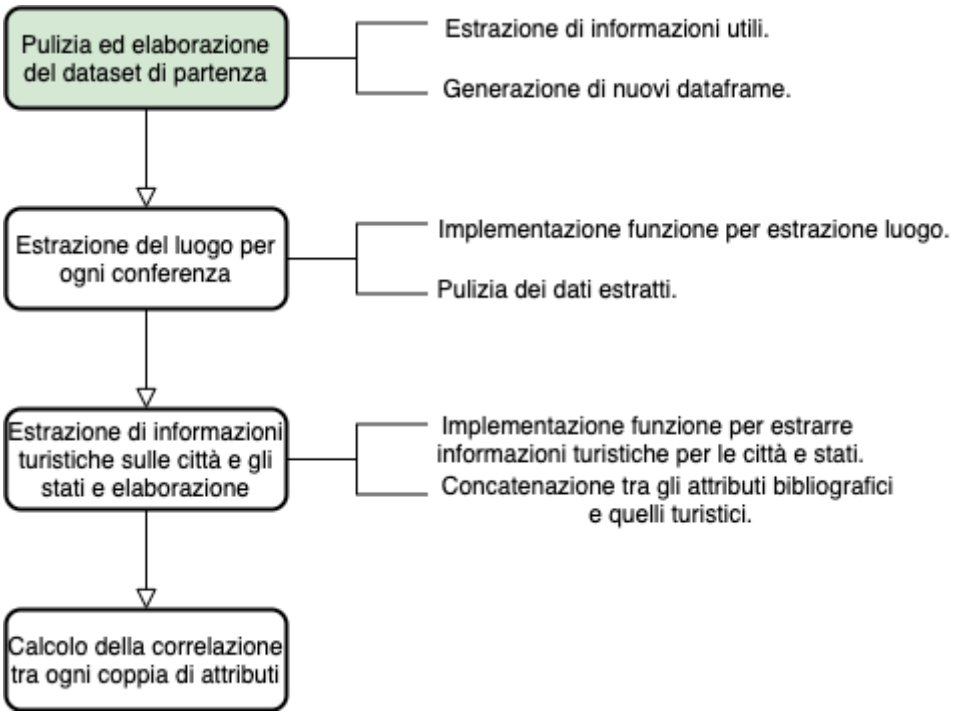


diagramma 3.2

Come abbiamo già spiegato nella sezione 1.3.2 la pulizia dei dati è quasi sempre necessaria nel caso in cui i dati vengono raccolti in modo automatizzato come nel nostro caso. Prima abbiamo detto che per ogni articolo sono stati raccolti anno di pubblicazione, numero di citazioni e la conferenza a cui appartiene, quindi il primo passo della pulizia è controllare che queste tre informazioni siano state acquisite correttamente andando ad eliminare eventualmente le righe con valori errati o incompleti evitando in questo modo di lavorare con dati sporchi che possono influire in modo negativo sul risultato dell'analisi.
Fare la pulizia dei dati non significa eliminare solamente i dati errati, è possibile infatti che durante l'elaborazione ci si accorga che alcuni delle informazioni con cui si sta lavorando non vadano bene, per esempio, nel nostro caso abbiamo raccolto informazioni su conferenza a partire dagli anni '60 al 2020 ma dopo una attenta valutazione ci siamo resi conto che per ottenere un risultato più preciso e affidabile era necessario eliminare gli articoli dal '60 al '83 in quanto non avevamo a disposizione un numero sufficiente di informazioni da analizzare. Infatti mentre per gli anni più recenti è stato possibile raccogliere più di centomila articoli di conferenza, per l'intervallo di tempo che va dal 1960 al 1983 non è stato altrettanto facile ottenere un numero sufficiente di dati. D'altronde nella sezione 1.2 parlando dei big data avevamo visto nella figura 1.1 la crescita esponenziale del volume di dati che ci circonda, quindi non ci dovrebbe stupire il fatto che sia più facile accedere a informazioni su eventi accaduti più recentemente.
Discorso diverso invece per gli articoli più recenti del 2017, i quali sono stati eliminati per non farci trarre in inganno dalla gioventù. Essendo, infatti, le citazioni il parametro scelto per misurare il successo di una conferenza, è meglio non considerare per lo studio gli articoli

troppo recenti i quali potrebbero ricevere ancora tante citazioni e di conseguenza i dati utilizzati per l'analisi si rivelerebbero obsoleti.

3.3 Elaborazione dei dati

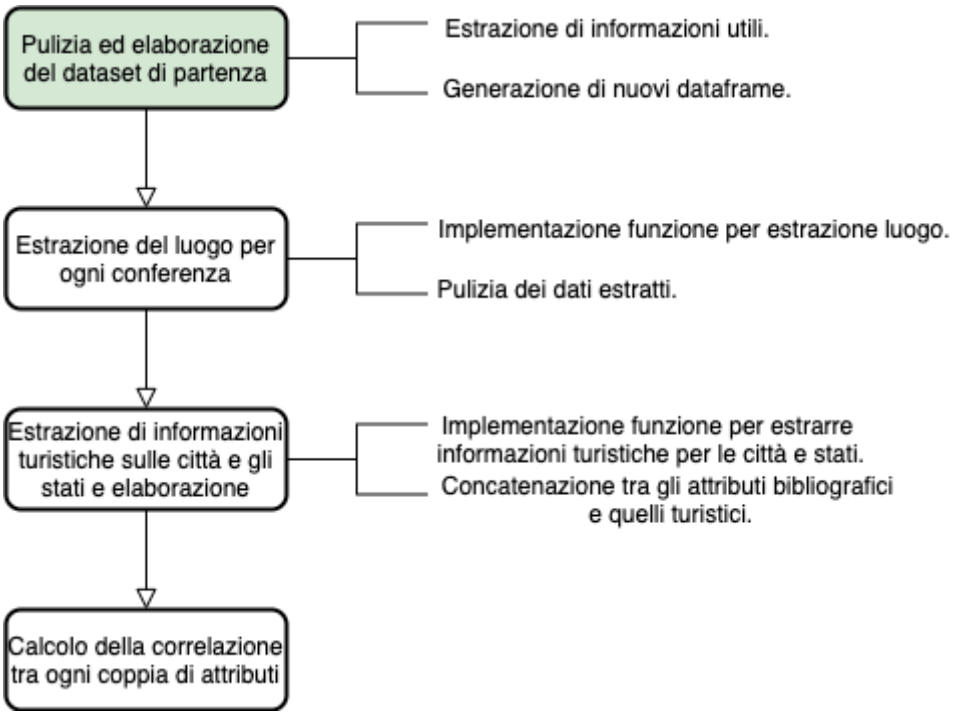


diagramma 3.3

Come abbiamo già visto nella breve introduzione fatta nel paragrafo 1.3.3, elaborare i dati significa classificare, ordinare, aggregare, generare informazioni statistiche e molto altro.
Le operazioni eseguite sui dati di partenze sono diverse, da un semplice ordinamento alla generazione di un nuovo dataframe, con lo scopo di estrarre le informazioni più utile e utilizzarle alla fine per la correlazione e l'analisi.
Partendo dal dataset di cui abbiamo parlato nella sezione 3.1, una delle prime informazioni calcolate è stato il numero di articoli raccolti per ogni anno, in modo da individuare a partire da quale anno avevamo abbastanza materiale per l'analisi. Dopodiché la prima idea è stata quella di fare una classifica delle migliori e delle peggiori conferenze, ma per fare ciò, visto che il dataset di partenza era una raccolto di articoli è stato necessario raggrupparli. Il risultato è il seguente dataframe dove l'indice è la conferenza e per ognuna è stato calcolato il numero di citazioni totali, il numero di articoli e le citazioni medie:

	CIT TOT	CIT AVG	N.ARTICLES
conf/cscw/2000	8471	162.90	52
conf/mobicom/2000	3566	127.35	28
conf/crypto/2001	3947	119.60	33
conf/wmcsa/1999	1429	119.08	12

dataframe 3.1

Dal nuovo dataframe è stato possibile estrarre sia in modo tabellare che grafico la top dieci e la flop dieci, sia in base alle citazioni totali che quelle medie. Qualcuno potrebbe pensare che queste due classifiche sono simile, ma non è così in verità, infatti alcune conferenze sono composte anche da più di 1000 articoli, quindi sono più predisposte ad avere un numero di citazioni totali alto ma non per forza un alto numero di citazioni medie. Nella top dieci per citazioni medie, infatti, troviamo conferenze con un numero di articoli non superiore al 50.

3.3.2 L'evoluzione delle conferenze

Un fattore importante da tenere in considerazione è il fatto che una conferenza può avere diverse edizioni, per esempio molte si tengono ogni anno, perciò, oltre a mettere a confronto le diverse conferenze dove oltre al luogo anche l'argomento incide sul successo, può risultare estremamente utile allo scopo finale analizzare l'evoluzione delle singole edizioni che invece trattano argomenti simili. Per prima cosa quindi è stato creato un nuovo dataframe, come fatto in precedenza, stavolta però senza appunto fare distinzione tra le varie edizioni, calcolando il numero di citazioni totali, il numero di articoli e le citazioni medie per ogni conferenza indipendentemente da quante volte si è tenuta. La tabella ottenuta è la seguente:

	CIT TOT	CIT AVG	N.ARTICLES
adaptive	2074	86.41	24
diau	789	71.72	11
banff	226	41.87	5
crypto	59794	41.87	1428

dataframe 3.2

Partendo dal nuovo dataframe si va ad estrarre anche stavolta la top dieci e la flop dieci sia in base alle citazioni medie che totali. Per semplicità fino alla fine di questo paragrafo il termine evoluzione verrà usato per indicare l'insieme delle edizioni di una particolare conferenza.
Il lato più interessante dell'evoluzione è seguire l'andamento nel tempo delle singole edizioni, calcolando per ogni anno le citazioni totali e le citazioni medie in questo modo possiamo confrontare gli anni migliori con i peggiori per vedere se effettivamente il luogo può aver influito.
Durante questa fase sono stati creati diversi dataframe, andiamo perciò a descrivere per ognuno il punto di partenza e arrivo:

- df_avg_cit_evolution: molto spesso per generare un nuovo dataframe si passa attraverso un dictionary, ovvero una struttura dati di Python, questo perché pandas mette a disposizione delle funzioni di aggregazione che permettono di raggruppare i dati secondo parametri comuni e calcolare la somma o la media di alcuni attributi, come può essere le citazioni totali o medie, per poi, appunto, salvare i dati in questa struttura. In questo caso, partendo dal solito dataset di partenza e raggruppando per conferenza, abbiamo ottenuto un dizionario dove la chiave era la sigla che identifica la conferenza e il valore era a sua volta un altro dizionario così composto: l'anno dell'edizione come chiave e il numero di citazioni medie come valore. Un esempio del risultato ottenuto può essere questo: crypto:{1985: 30, 1986: 25, …, 2001: 120, …, 2017: 10}. Partendo da questa struttura dati è stato facile generare un dataframe dove l'indice era la chiave del dizionario più esterno, quindi la sigla della conferenza, le colonne erano le chiavi del dizionario più interno, quindi gli anni delle varie edizioni tenute, e il valore della cella il numero di citazioni medie ottenute in quell'anno per quella conferenza. Il dataframe generato è il seguente:

	...	2008	2009	2010	2011	2012	2013	...
3dic		NaN	2.46	1.13	0.46	NaN	0.74	
3dui		2.21	2.69	3.79	2.41	2.5	1.82	
ACISicis		1.64	0.86	0.97	0.92	0.67	0.59	
ACMace		2.62	1.9	1.89	3.37	1.92	1.94	

dataframe 3.3

Ovviamente non tutte le conferenza si sono tenute ogni anno, per questo motivo alcuni valori sono NaN.

- df_cit_evolution: la stessa operazione è stata fatta due volte, questa volta però le informazioni calcolate sono state le citazioni totali. Ecco un esempio del dataframe:

	...	2008	2009	2010	2011	2012	2013	...
3dic		NaN	200	75	50	NaN	66	
3dui		82	97	129	87	125	115	
ACISicis		181	177	144	62	74	59	
ACMace		291	187	70	317	129	150	

dataframe 3.4

- df_evolution: questo dataframe oltre a essere una fusione dei precedenti due, calcola altre tre informazioni: il numero di edizioni, il numero di citazioni totali cumulate nel corso degli anni e il numero di citazioni medie totali calcolato a partire dai primi due. Il risultato ottenuto è quindi una tabella dove l'indice è come sempre la conferenza mentre le colonne, che sono 69, sono formate nel seguente modo: 1985-AVG(quindi citazioni medie ottenute nel edizione del 85), 1985-CIT(citazioni totali), 1986-AVG, 1986-CIT e così via fino al 2017, oltre alle tre informazioni sopracitate. Di seguito un esempio del dataframe:

	...	2016-AVG	2016-CIT	2017-AVG	2017-CIT	AVG CIT	TOT CIT	EDITIONS
SIGIR		5.17	1212	3.56	908	12.34	48720	33
SIGMOD		5.11	1329	3.23	721	15.94	57509	33
STACS		0.1	6	0.03	2	3.72	6621	33
ICRA		2.34	1624	1.47	1174	2.69	55742	33

dataframe 3.5

- df_over_and_under_avg_cit: partendo da df_evolution e in particolare dalla colonna citazioni medie totali è stato generato un nuovo dataframe, anche stavolta passando per un dizionario, che divide in due colonne le varie edizioni per ogni conferenza, mettendo nella colonna di sinistra gli anni che hanno avuto più successo rispetto alla media e nella colonna di destra quelli che hanno ricevuti un numero di citazioni inferiore alla media totale. Per esempio ipotizziamo che una conferenza abbia tenuto 4 differenti edizioni, con i seguenti risultati: nel 2010 ha ottenuto 20 citazioni medie, nel 2011 30, 10 nel 2012 e 40 nel 2013, questo vuol dire che la media totale è di 25 citazioni e quindi le colonne del dataframe appena generato saranno così formate: nella colonna edizioni sopra la media troviamo l'anno 2011 e 2013, mentre le edizioni del 2010 e 2012 andranno nell'altra colonna. Inoltre viene calcolata una terza informazioni, ovvero la percentuale di edizioni sotto la media che in questo caso sarebbe stata del 50%. Ecco un estratto del dataframe:

	Sotto la media	Sopra la media	% Sotto la media
VAST	2003, 2004, 2005, 2006 2007, 2008, 2009,2010...	2001	92,85%
CCIA	2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015...	2002	90,90%
EGVE	2000, 2001, 2002, 2003, 2004, 2005, 2008, 2009...	2006, 2007	88.88%
EGH	1986, 1987,1988, 1989, 1990, 1991, 1992, 1993...	2011, 2013, 2015, 2017	87.1%

dataframe 3.6

- df_min_and_max: l'idea è quella di individuare l'anno migliore e peggiore sia a livello di citazioni medie che a livello di citazioni totali. Perciò da df_avg_cit_evolution e df_tot_cit_evolution sono stati estratti per ogni conferenza l'anno e il numero di citazioni della peggiore e della migliore edizione. Inoltre nella tabella è riportato anche la differenza tra il numero di citazioni ottenute nell'anno che ha avuto maggior successo e quello che ha avuto minor successo, in questo modo si può ordinare per differenza e analizzare il luogo dell'edizione migliore rispetto alla peggiore. Ecco un esempio:

	MIN AVG	MAX AVG	MIN CIT	MAX CIT	DIF AVG	DIF CIT
CSCW	1986 (4.14)	2000 (162.9)	1986 (145)	2000 (8471)	158.76	8326
MOBICOM	2017 (1.23)	2000 (127.35)	1996 (156)	2000 (3566)	126.11	3410
ECOOP	2017 (0.24)	1997 (61.52)	2017 (12)	1997 (1415)	61.28	1403
CCS	2017 (3.04)	1993 (62.56)	1994 (275)	2013 (3726)	59.51	3451

dataframe 3.7

- df_ratio: partendo dal numero di citazioni medie per ogni anno estratti dal df_avg_cit_evolution, e dalla media totale che possiamo estrarre dal df_evolution siamo andati a creare un nuovo dataframe che calcola il rapporto tra le citazioni medie annuali e quelle totali. Se per esempio riprendiamo la conferenza con quattro edizioni che abbiamo ipotizzato prima, con le seguenti citazioni medie per anno: 20 nel 2010, nel 2011 30, 10 nel 2012 e 40 nel 2013 l'eventuale riga del df_ratio sarà: 0.8 nel 2010, ovvero 25 diviso 20, 1.2 nel 2011, 0.4 nel 2012 e 1.6 nel 2016. Ecco un estratto del dataframe:

	...	2009	2010	2011	2012	2013	...	tot avg
3dic		2.84	1.31	0.53	NaN	0.85		0.86
3dui		1.2	1.69	1.07	1.11	0.81		2.24
ACISicis		1.1	1.24	1.18	0.86	0.76		0.78
ACMace		0.74	0.73	1.31	0.75	0.75		2.56

dataframe 3.8

Ricordo che tutte le informazioni ottenute durante questa fase saranno utilizzate per l'analisi finale. Per esempio i primi due dataframe sono stati utilizzati all'interno di una funzione che data la sigla di una conferenza in input disegna il grafico dell'andamento nel tempo delle citazioni medie o totali. Da df_over_and_under_avg_cit grazie alla percentuale calcolata, posso andare ad estrarre ed analizzare gli estremi, ovvero quelle con la percentuale più alta o più bassa, perché potenzialmente sono le più significative al nostro fine. Grazie a df_ratio, invece, possiamo facilmente individuare le edizioni che sono andate due, tre o quattro volte meglio rispetto alla media totale.
Le parti di codice più significative e i grafici saranno illustrati e commentati nei capitoli successivi.

Nella sezione 1.3 abbiamo detto che le varie fasi della data analysis sono interattive in quanto il risultato delle fasi successive possono portare a rivedere le fasi precedenti ed è quello che è accaduto durante lo studio dell'evoluzione delle conferenze, infatti abbiamo dovuto fare un'altra operazione per la pulizia dei dati per eliminare le conferenze che si sono tenute solo una volta in quanto non aveva senso includerle nell'analisi dell'evoluzione.

3.4 Estrazione dei dati turistici

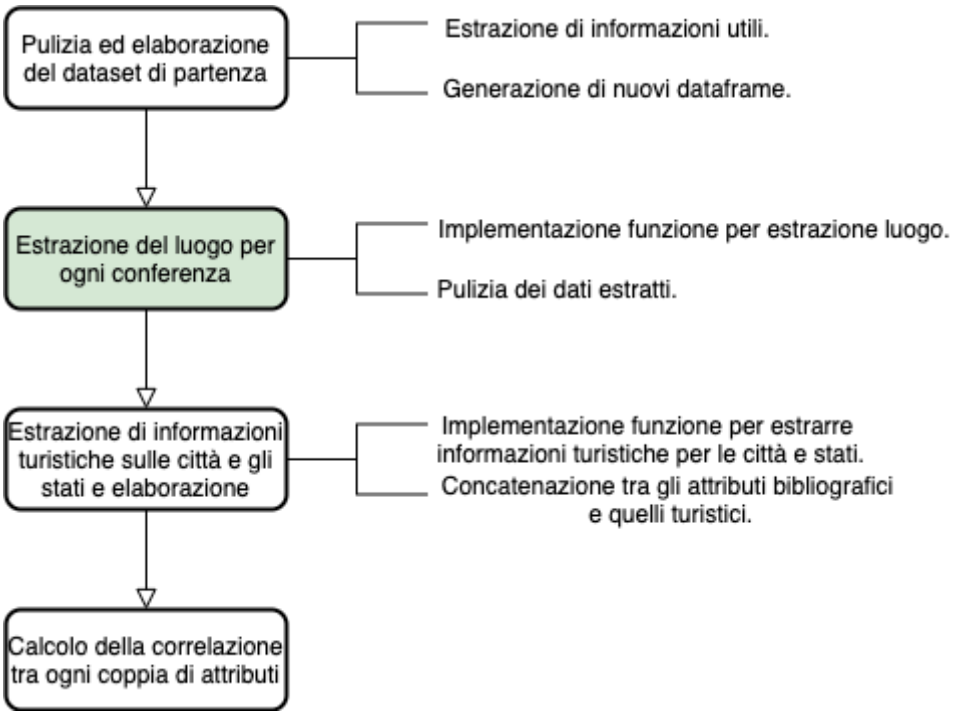


diagramma 3.4

Questa fase è un'altro esempio di come il processo di analisi è interattivo, infatti dopo aver raccolto, pulito ed elaborato i dati sulle conferenze ora bisogna fare lo stesso con le informazioni turistiche.
Per raggiungere il nostro obiettivo è necessario conoscere il luogo dove si sono tenute le conferenze che abbiamo analizzato fino ad ora. Dobbiamo perciò estrarre questa informazione partendo dalla sigla che identifica la conferenza e l'anno in cui si è tenuta. Per farlo utilizziamo la libreria BeautifulSoup di Python che è stata introdotta nella sezione 2.5. E' stata implementata una funzione chiamata get_place, che vedremo nel capitolo dedicato all'implementazione, la quale dato in input il link di una conferenza restituisce il luogo dove si è tenuta. Il link viene generato a partire da un prefisso al quale viene concatenato la sigla e l'anno della conferenza. Una volta data in input, la stringa viene letta da urlopen, una funzione della libreria Urllib di cui abbiamo parlato nel paragrafo 2.4, che come suggerisce il nome apre l'url e permette di fare web scraping e quindi estrarre le informazioni di cui abbiamo bisogno.
Estrarre in modo automatizzato informazioni non è mai facile soprattutto nel caso in cui i dati con cui si lavora sono eterogenei, con questo metodo infatti sono riuscito a raccogliere il luogo di più di 23000 conferenze su quasi 30000. Il motivo è semplice ed è dovuto al fatto che tutti i link vengono generati tutti nello stesso modo a partire dallo stesso prefisso, il che va bene per gran parte di loro ma non per tutti. Apportando alcune modifiche all'algoritmo utilizzato, in particolare al modo in cui viene generato l'url, è stato possibile estrarre il luogo di altre 2000 conferenze circa. Ecco un piccolo esempio del dataframe:

	city	state
conf/cscw/2000	Philadelphia	USA
conf/adaptive/2007	Paris	France
conf/eurocrypt/2004	Interlaken	Switzerland
conf/wmcsa/1999	New Orleans	USA

dataframe 3.9

3.4.2 Pulizia dati turistici

Per l'ennesima volta ci troviamo ad affrontare la fase della pulizia dei dati, ovviamente necessaria per il fatto che i dati sono stati estratti tramite una funziona non precisa al 100%, che può aver quindi raccolto dati incompleti o completamente errati. Bisogna però precisare che è praticamente impossibile scrivere un algoritmo che vada bene per tutte le 30000 conferenze, quindi si cerca di implementare una funzione che vada bene per la maggior parte dei casi.
Una volta terminata l'esecuzione, i dati vengono salvati temporaneamente in un dictionary, che è facilmente convertibile in un dataframe, il quale si può esportare in formato csv grazie alle funzioni messe a disposizione da pandas. Prima di creare il dataframe però bisogna eliminare sicuramente le conferenze per cui non è stato possibile estrarre il luogo, oltre a quelle per cui il luogo estratto è per qualche motivo errato.

3.4.3 Elaborazione dei dati turistici

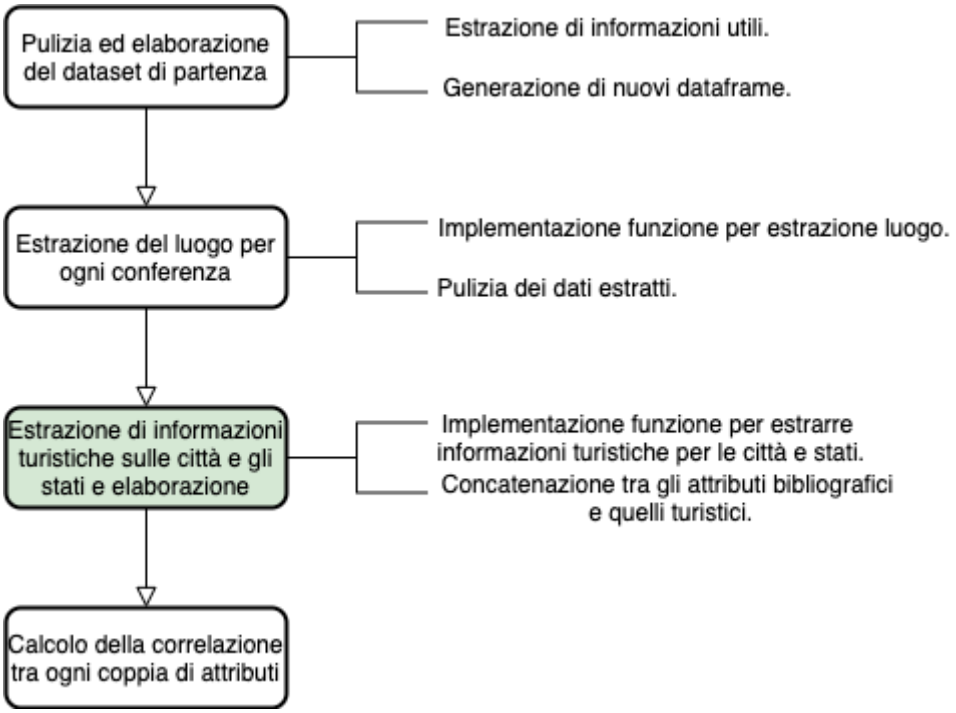


diagramma 3.5

Ora che abbiamo le informazioni che ci servivano salvate in un dizionario possiamo creare un dataframe con la sigla e l'anno della conferenza come indice, e città e stato come colonne, in questo modo per ogni edizione sarà possibile, in modo semplice e veloce, estrarre il posto dove si è tenuta.

Arrivati a questo punto sarebbe molto utile unire le informazioni sulle conferenze, ottenute attraverso l'elaborazione, con i dati turistici appena estratti. Pandas mette a disposizione diverse funzionalità per combinare i dataframe, tra cui merge, join e concat. Proprio quest'ultima è la funzione che abbiamo utilizzato, perché permette di specificare lungo quale asse avverrà la concatenazione, i casi sono due:

- la tabella A e la tabella B hanno le stesse colonne ma diversi indici, il risultato sarà quindi un'unica tabella in cui sono presenti tutte le righe sia della tabella A che B;
- la tabella A e la tabella B hanno diverse colonne ma stessi indici, il risultato sarà quindi una nuova tabella in cui sono presenti sia le colonne della tabella A che B.

I dataframe che vogliamo concatenare hanno gli stessi indici, quindi ricadiamo nel secondo caso. Il risultato finale è una tabella dove sono presenti le citazioni medie e totali calcolate in precedenza e le informazioni appena estratte sul luogo della conferenza diviso in due colonne, città e stato, come mostra la seguente tabella:

	avg cit	tot cit	n.articles	city	state
conf/cscw/2000	162.9	8471	52	Philadelphia	USA
conf/adaptive/2007	86.41	2074	24	Paris	France
conf/eurocrypt/2004	109.3	3935	36	Interlaken	Switzerland
conf/wmcsa/1999	119.08	1429	12	New Orleans	USA

dataframe 3.10

A partire da questo dataframe è possibile generare una nuova struttura dove sono presenti il numero di edizioni, numero di citazioni totali e medie per ogni città, in questo modo:

	avg cit	tot cit	editions
Beijing	168.7	61913	368
Paris	190.33	62430	328
San Francisco	274.26	82004	299
Barcelona	203.89	53420	263

dataframe 3.11

Da qui si può stabilire una classifica dei posti in cui le conferenza hanno maggiore successo. Per raggiungere il nostro obiettivo però ci manca ancora qualcosa, infatti ora che abbiamo una classifica delle città in base alle citazioni medie non ci resta che stilare una classifica basata sull'attrazione turistica e mettere a confronto i risultati ottenuti. Purtroppo trovare un indice che misuri in modo preciso e affidabile l'attrazione turistica di una città è pressoché impossibile perché dovrebbe tenere conto di tantissimi fattori, tra cui il numero di turisti ogni anno, quanto il posto è facilmente raggiungibile e accessibile, la qualità dei servizi di trasporto pubblico, e tanti altri. Per ottenere un simile parametro sarebbe necessario condurre uno studio per ogni singola città, il che costerebbe tantissimo tempo, perciò la nostra scelta è ricaduta su un criterio più facile da estrarre che sicuramente non è sempre affidabile ma comunque significativo. Il parametro scelto è la dimensione della pagina wikipedia (di seguito detto anche "swp"), perché ci si aspetta che una città con un'alta attrazione turistica abbia un swp maggiore rispetto ad una città con bassa attrazione.
Anche in questo caso, come per l'estrazione dei luoghi in cui si sono tenute le conferenze, è necessario implementare una funzione che vada ad estrarre in modo automatizzato la lunghezza della pagina wikipedia per ogni città in cui si è tenuta una conferenza tra quelle analizzate. Quindi a partire dal prefisso "https://en.wikipedia.org/wiki/", al quale si va ad aggiungere il nome della città, è possibile recuperare la pagina e quindi accedere ai metadati da dove si può estrarre lo swp. Il risultato di questa operazione è semplicemente una tabella

con due colonne: nome città e la dimensione della sua pagina wikipedia. Una volta recuperata la dimensione della pagina di tutte le città prese in considerazione, possiamo andare a concatenare questa informazione al dataframe generato poco fa dove per ogni città è stato calcolato citazione medie, totali e numero di edizioni, in questo modo abbiamo un'unica tabella che contiene tutte le informazioni necessarie per la correlazione e di conseguenza basterà invocare la funzione corr() di pandas e mostrare il risultato ottenuto graficamente. Il dataframe utilizzato per la correlazione è il seguente:

	avg cit	tot cit	editions	swp
Beijing	168.7	61913	368	179383
Paris	190.33	62430	328	218583
San Francisco	274.26	82004	299	177857
Barcelona	203.89	53420	263	135424

dataframe 3.12

3.4.4 Informazioni turistiche sugli stati

Come fatto per le città, anche per gli stati sono stati raccolte informazioni turistiche, in questo caso però è stato possibile recuperare dati più precisi, in particolare sono stati individuati i seguenti cinque indici:

1. TTCI: Travel & Tourism Competitiveness Index (min 1, max 7)[9]
2. TSI: Tourist service infrastructure (min 1, max 7)[10]
3. CRBT: Cultural resources and business travel (min 1, max 7)[11]
4. GPI: Ground and port infrastructure (min 1, max 7)[12]
5. SI: Sum index, è semplicemente la somma dei 4 precedenti indici

Partendo dal dataframe che contiene sia le citazioni che il luogo della conferenza e raggruppando per paese è stato possibile calcolare il numero di citazioni medie, totali e numero di edizioni per ogni stato, cosi come fatto per le città. Dopodiché a questo dataframe sono stati concatenati i cinque indici appena raccolti ottenendo così una tabella che ha come colonne avg_cit (citazioni medie), tot_cit (citazioni totali), editions (numero di edizioni), TTCI, TSI, CRBT, GPI e sum index e il nome dello stato come indice, come mostrato nella figura sottostante:

	avg cit	tot cit	editions	TTCI	TSI	CRBT	GPI	sum index
USA	249.77	1539133	6167	5.3	6.6	4.7	4.9	21.5
China	130.44	223458	1714	4.9	3.5	7.0	3.9	19.3
Germany	124.61	168601	1356	5.4	5.9	6.5	5.7	23.5
Italy	150.48	187807	1256	5.1	6.0	6.5	4.7	22.3

dataframe 3.13

A questo punto siamo pronti per calcolare l'indice di correlazione e relativo p-value per ogni coppia di attributi, quindi l'attributo bibliografico e l'attributo turistico.

3.5 Correlazione

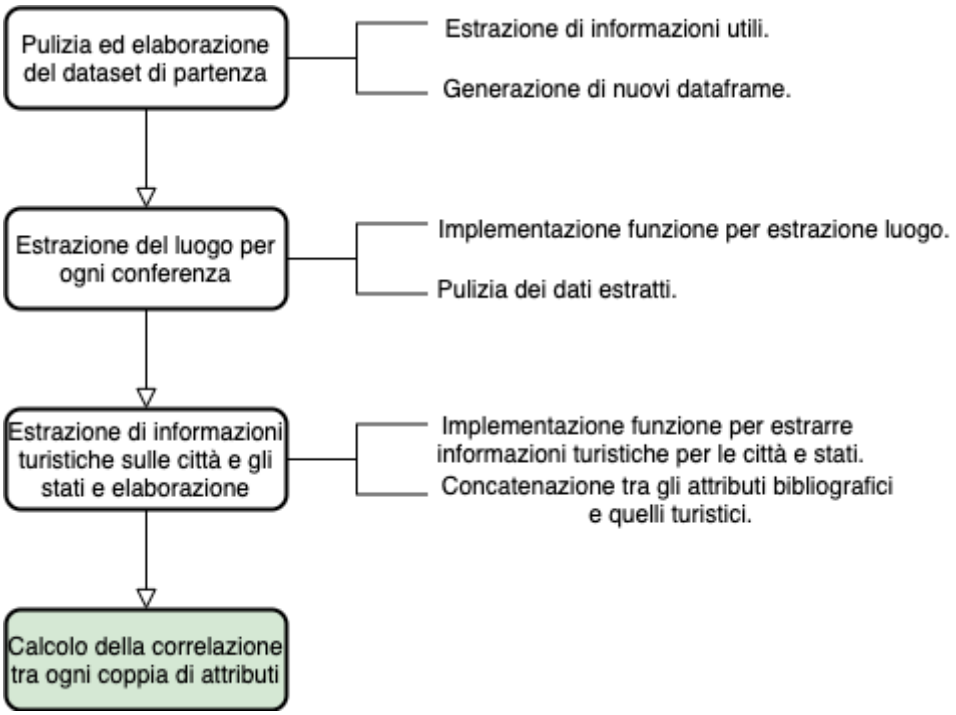


diagramma 3.6

La correlazione[13] è una misura statistica che esprime la relazione lineare tra due variabili ed è molto usata per descrivere semplici relazioni senza dover parlare di causa ed effetto. Il coefficiente di correlazione, è usato per quantificare la forza del rapporto tra variabili ma non è in grado di verificare la presenza o l'effetto di altre variabili che non siano le due prese in esame. La correlazione viene descritta mediante un valore che non è dotato di un'unità di misura specifica, chiamato coefficiente di correlazione, compreso tra -1 e +1 e denotato da r. La significatività statistica è indicata tramite un p-value. Pertanto, le correlazioni in genere

vengono scritte utilizzando due numeri fondamentali: r e p che possono essere interpretati in questo modo:

- Più r si avvicina a zero, più la correlazione lineare è debole;
- Un valore r positivo è indice di una correlazione positiva, in cui i valori delle due variabili tendono ad aumentare in parallelo;
- Un valore r negativo è indice di una correlazione negativa, in cui il valore di una variabile tende ad aumentare quando l'altra diminuisce;
- Il p-value è la prova che, in base a quanto osservato sul campione, possiamo concludere in maniera significativa che il coefficiente di correlazione della popolazione è diverso da zero.

Le correlazioni vengono rappresentati tramite grafici perciò di seguito verrà solamente descritte le operazioni di correlazione fatte mentre i risultati saranno illustrati nei capitoli successivi.

3.5.2 Correlazione per città

Per calcolare il coefficiente di correlazione sono stati utilizzati 3 diverse funzioni: Pearson, Kendall e Spearman.
A partire dai dataframe creati in precedenza è stato facile calcolare il coefficiente di correlazione per ogni coppia di attributi, grazie alla funzione corr di pandas che però non restituisce il p-value, quindi è stato necessario integrarla con le funzioni kendalltau, pearsonr, e spearmanr della libreria stats del modulo scipy che invece restituiscono anche il p-value. Gli attributi presi in considerazione sono il numero di citazioni totali, numero di citazioni medie, numero di edizioni per ogni conferenza e la dimensione della pagina wikipedia di ogni città. Una volta ottenuti i risultati della correlazione possiamo rappresentarli graficamente grazie alla funzione heatmap di seaborn che è una mappa termica dove ogni cella rappresenta il valore di correlazione tra l'attributo lungo l'asse x con quello lungo asse y.
Dopo aver calcolato il coefficiente prendendo in considerazione tutte le città, restringiamo il campo prima alle sole città che hanno ospitato almeno 15 edizioni differenti, e dopo a quelle che ne hanno ospitato almeno 50 e ricalcoliamo il coefficiente di correlazione. Ovviamente più edizioni si sono svolte in una città più è affidabile il risultato ottenuto. Come vedremo dopo il coefficiente tra l'attributo citazioni medie e l'attributo swp, ovvero quello più significativo per il nostro obiettivo, cresce all'aumentare della soglia minima di edizioni svolte in una determinata città per essere inclusa nel calcolo.

3.5.3 Correlazione per stato

Come abbiamo visto nel paragrafo 3.4.4 sono state raccolte informazioni turistiche anche per quanto riguarda lo stato e quindi è stata calcolata la correlazione anche utilizzando gli attributi TTCI, TSI, CRBT, GPI e la somma di questi quattro indici. Anche in questo caso

sono stati utilizzati tutte tre le funzioni messe a disposizione da pandas: Kendall, Pearson e Spearman.

3.5.4 Correlazione per singola conferenza

Nel calcolare il coefficiente di correlazione non possiamo dimenticarci dell'evoluzione, di cui abbiamo parlato nel paragrafo 3.3.2. Infatti è stata implementata una funzione che per ogni edizione di una conferenza estraeva, dai dataframe creati in precedenza, le citazioni medie e totali, lo swp della città in cui si è tenuta tale edizione e il TTCI e SI (ovvero TTCI + TSI + CRBT + GPI) dello stato. Per ognuna di queste informazioni sono state create delle liste ordinate che venivano poi passate a coppie, per esempio lista citazioni medie e lista swp, alle solite tre funzioni Pearson, Kendall e Spearman le quali ne calcolavano il coefficiente tra la coppia di attributi passate in input. Il risultato finale sono due dataframe, uno per la correlazione degli attributi bibliografici, quindi citazioni medie e totali, con gli attributi turistici che riguardano la città e l'altro con gli attributi turistici che riguardano lo stato. Il dataframe ha quidi la sigla della conferenza come indice e nelle colonne la coppia di attributi tra cui è stata fatta la correlazione, per esempio citazioni medie-swp o citazioni totali-TTCI, e per ogni coppia il relativo p-value tra parentesi. Per ogni colonna è stata calcolata la media e la moda ed infine i dataframe sono stati esportati come csv grazie alla funzione di pandas to_csv. Il risultato è il seguente:

	swp - avg cit	swp - tot cit	editions
ACNS	0.29 (0.28)	0.26 (0.34)	15
ECAL	0.78 (0.00)	0.63 (0.01)	15
MM	0.29 (0.00)	0.17 (0.08)	33
SPIN	0.64 (0.00)	0.73 (0.00)	19

dataframe 3.14

3.6 Possibili implementazioni future

L'analisi condotta è un'analisi descrittiva, ma come abbiamo visto nella sezione 1.3.4 oltre a quella descrittiva esiste l'analisi predittiva e proprio su questo si potrebbe basare le eventuali implementazioni future. A partire dai risultati ottenuti si potrebbe rispondere a domande del tipo: "Dato un determinato argomento e una città, quante citazioni medie potrebbe avere questa conferenza?", oppure "Dove posso tenere la mia conferenza se voglio raggiungere 100 citazioni medie per articolo?". Inoltre integrando i dati che già abbiamo con altri dati si

potrebbe individuare il luogo più adatto in base all'argomento della conferenza oppure al periodo dell'anno.
Resta comunque difficile dire a priori se un edizione avrà più o meno successo in base al luogo, questo perché ovviamente il posto è solo uno dei tantissimi fattori che influisce sulla qualità. Per esempio una conferenza che tratta l'argomento pandemie, oggi probabilmente avrebbe molto più successo rispetto alla stessa conferenza fatta 10 anni prima, a prescindere dal luogo.

Capitolo 4

Implementazione

In questo capitolo saranno illustrate e commentate le parti di codice più significative e i dataframe generati.

4.1 Pulizia dati

```
df = df[df['CIT']>-1]
df = df[df['crossref']!='']
df = df[df['year']<str(2018)]
```

Per prima cosa siamo andati ad eliminare gli articoli per cui non è stato possibile estrarre il numero di citazioni, dopodiché quelli per cui non è stato possibile definire a quale conferenza appartiene e infine quelli pubblicati dopo il 2017, quindi a partire dal 2018.

```
df_groupby_year = df.groupby(['year']).CIT.count()
print(df_groupby_year)
```

Per ogni anno calcoliamo il numero di articoli:

```
1960          9
1961         26
1962         16
1963         13
1964         42
1965         51
1966        157
1967        242
1968        287
1969        298
1970        259
1971        296
1972        560
1973        721
1974       1075
1975        950
1976       1180
1977        824
1978       1246
1979       1318
1980       1611
1981       1594
1982       1766
1983       1763
1984       2469
1985       2514
1986       2999
1987       3637
1988       5548
1989       6709
1990       8380
1991       8842
1992       9888
1993      11273
1994      15418
1995      16394
1996      17783
1997      20165
1998      22893
1999      25119
2000      28977
2001      32061
2002      38030
2003      44533
2004      57289
2005      69860
2006      81346
2007      86890
2008      91347
2009      97888
2010     101580
2011     114023
2012     117218
2013     123676
2014     128874
2015     132486
2016     130834
2017     136989
```

E' stato individuato il 1985 come anno di partenza, gli articoli precedenti sono stati perciò eliminati:

```
df = df[df['year']>str(1984)]
```

4.2 Elaborazione dati sulle conferenze

```
top10 = {v:k for v,k in df.groupby(['crossref']).CIT.sum().
 ↪sort_values(ascending=False).head(10).items()}
top10_plot = sns.barplot(x=list(top10.values()), y=list(top10.keys()))
```

A partire dal dataset di partenza e raggruppando per conferenze siamo andati ad estrarre la top dieci che poi è stata rappresentato graficamente con seaborn(sns). Nello stesso mondo ma ponendo sort_values(ascending=True) siamo andati ad estrarre la flop dieci.

```
single_conference_cit = df.groupby(['crossref']).CIT.sum().
 ↪sort_values(ascending=False)
single_conference_count_article = df.groupby(['crossref']).CIT.count().
 ↪sort_values(ascending=False)


single_conference_avg_cit = {key:(int(val)/
 ↪int(single_conference_count_article[key])) for key,val in␣
 ↪single_conference_cit.items()}

single_conference = {k:
 ↪[v,single_conference_avg_cit[k],single_conference_count_article[k]] for k,v␣
 ↪in single_conference_cit.items()}

df_single_conference = pd.DataFrame(data=single_conference.
 ↪values(),index=single_conference.keys(), columns=['CIT TOT', 'CIT AVG', 'N.
 ↪ARTICLES'])

df_single_conference = df_single_conference.sort_values('CIT AVG',␣
 ↪ascending=False)
```

Script per la creazione del dataframe 3.1.

4.2.2 L'evoluzione delle conferenze

```
df[['conf','cod', 'anno']] = df.crossref.str.split("/",expand=True,)

conference_tot_cit = df.groupby(['cod']).CIT.sum()
conference_tot_article = df.groupby(['cod']).CIT.count()
avg_cit_for_conference = {key:(int(val)/int(conference_tot_article[key])) for␣
↪key,val in conference_tot_cit.items()}

conference = {k:[v,avg_cit_for_conference[k],conference_tot_article[k]] for k,v␣
↪in conference_tot_cit.items()}

df_conference = pd.DataFrame(data=conference.values(), index=conference.keys(),␣
↪columns=['CIT TOT', 'CIT AVG', 'N.ARTICLES'])
df_conference = df_conference.sort_values(['CIT AVG'], ascending=False)
```

Script per la creazione del dataframe 3.2.

In modo simile sono stati generati altri dataframe di cui abbiamo discusso nella sezione 3.3.2:

- codice per la creazione del *df_avg_cit_evolution*:

```
lst_dataframes = []
for conference in keys:
    single_conference = {}
    n_editions = 0
    for k,v in conference_tot_cit[conference].items():
        single_conference[str(k)] = v/conference_tot_article[conference][k]
        n_editions +=1
    if n_editions>1:
        lst_dataframes.append(pd.DataFrame(data=[single_conference.values()],␣
↪index = [conference], columns=single_conference.keys()))
avg_cit_evolution = pd.concat(lst_dataframes)
avg_cit_evolution = avg_cit_evolution.reindex(sorted(avg_cit_evolution.
↪columns), axis=1)
avg_cit_evolution.head(10)
```

Script per la creazione del dataframe 3.3.

- codice per la creazione del *df_cit_evolution*:

```
lst_dataframes = []
for conference in keys:
    single_conference = {}
    n_editions = 0
    for k,v in conference_tot_cit[conference].items():
        single_conference[str(k)] = v
        n_editions += 1
    if n_editions >1:
        lst_dataframes.append(pd.DataFrame(data=[single_conference.values()],␣
↪index = [conference], columns=single_conference.keys()))
cit_evolution = pd.concat(lst_dataframes)
cit_evolution = cit_evolution.reindex(sorted(cit_evolution.columns), axis=1)
cit_evolution.head(10)
```

Script per la creazione del dataframe 3.4.

- codice per la creazione di *df_evolution*:

```
lst_dataframes = []
conference_tot_cit = df.groupby(['cod','year']).CIT.sum()
conference_tot_article = df.groupby(['cod','year']).CIT.count()
keys = df.groupby(['cod']).CIT.count().keys()
for conference in keys:
    single_conference = {}
    tot_cit = 0
    tot_article = 0
    n_editions = 0
    for k,v in conference_tot_cit[conference].items():
        single_conference[str(k)+'-CIT'] = v
        single_conference[str(k)+'-AVG'] = v/
 ↪conference_tot_article[conference][k]
        tot_cit += v
        tot_article += conference_tot_article[conference][k]
        n_editions += 1
    single_conference['tot_cit'] = tot_cit
    single_conference['avg_cit'] = tot_cit/tot_article
    single_conference['n_editions'] = n_editions
    lst_dataframes.append(pd.DataFrame(data=[single_conference.values()], index␣
 ↪= [conference], columns=single_conference.keys()))
df_conf_evolution = pd.concat(lst_dataframes)
df_conf_evolution = df_conf_evolution.reindex(sorted(df_conf_evolution.
 ↪columns), axis=1)
```

Script per la creazione del dataframe 3.5.

- codice per la creazione di *df_over_and_under_avg_cit*:

```
over_and_under_avg_cit = {}
for row,col in avg_cit_evolution.iterrows():
    maggiori = []
    minori = []
    for k,v in col.items():
        if not math.isnan(v):
            if(v >= df_conference.at[row,'CIT AVG']):
                maggiori.append(k)
            else:
                minori.append(k)
    x = {}
    x['confSottoLaMedia']= minori
    x['confSopraLaMedia']= maggiori
    over_and_under_avg_cit[row] = x
```

Script per la creazione del dataframe 3.6.

- codice per la creazione di *df_min_and_max*:

```
year_with_min_avg_cit = {k:v for k,v in avg_cit_evolution.idxmin(axis=1).
 ↪items()}
df_year_with_min_avg_cit = pd.DataFrame(data=year_with_min_avg_cit.values(),␣
 ↪index=year_with_min_avg_cit.keys(), columns=['MIN AVG-year'])

min_avg_cit = {k:v for k,v in avg_cit_evolution.min(axis=1).items()}
df_min_avg_cit = pd.DataFrame(data=min_avg_cit.values(), index=min_avg_cit.
 ↪keys(), columns=['MIN AVG-value'])

year_with_min_cit = {k:v for k,v in cit_evolution.idxmin(axis=1).items()}
df_year_with_min_cit = pd.DataFrame(data=year_with_min_cit.values(),␣
 ↪index=year_with_min_cit.keys(), columns=['MIN CIT-year'])

min_cit = {k:v for k,v in cit_evolution.min(axis=1).items()}
df_min_cit = pd.DataFrame(data=min_cit.values(), index=min_cit.keys(),␣
 ↪columns=['MIN CIT-value'])

year_with_max_avg_cit = {k:v for k,v in avg_cit_evolution.idxmax(axis=1).
 ↪items()}
df_year_with_max_avg_cit = pd.DataFrame(data=year_with_max_avg_cit.values(),␣
 ↪index=year_with_max_avg_cit.keys(), columns=['MAX AVG-year'])

max_avg_cit = {k:v for k,v in avg_cit_evolution.max(axis=1).items()}
df_max_avg_cit = pd.DataFrame(data=max_avg_cit.values(), index=max_avg_cit.
 ↪keys(), columns=['MAX AVG-value'])

year_with_max_cit = {k:v for k,v in cit_evolution.idxmax(axis=1).items()}
df_year_with_max_cit = pd.DataFrame(data=year_with_max_cit.values(),␣
 ↪index=year_with_max_cit.keys(), columns=['MAX CIT-year'])

max_cit = {k:v for k,v in cit_evolution.max(axis=1).items()}
df_max_cit = pd.DataFrame(data=max_cit.values(), index=max_cit.keys(),␣
 ↪columns=['MAX CIT-value'])

dif_cit = {k:(max_cit[k]-v)  for k,v in min_cit.items()}
df_dif_cit = pd.DataFrame(data=dif_cit.values(), index=dif_cit.keys(),␣
 ↪columns=['DIF CIT'])
dif_avg_cit = {k:(max_avg_cit[k]-v)  for k,v in min_avg_cit.items()}
df_dif_avg_cit = pd.DataFrame(data=dif_avg_cit.values(), index=dif_avg_cit.
 ↪keys(), columns=['DIF AVG CIT'])

df_min_max = pd.concat([df_year_with_min_avg_cit, df_min_avg_cit,␣
 ↪df_year_with_max_avg_cit, df_max_avg_cit,
```

Script per la creazione del dataframe 3.7.

- codice per la creazione di *df_ratio*:

```
lst_dataframes = []
for conference in keys:
    single_conference = {}
    n_editions = 0
    for k,v in conference_tot_cit[conference].items():
        avg_cit = v/conference_tot_article[conference][k]
        if(df_conf_evolution.at[conference,'avg_cit']!=0):
            single_conference[k] = avg_cit/df_conf_evolution.
 ↪at[conference,'avg_cit']
        else:
            single_conference[k] = 0
        n_editions +=1
    single_conference['tot avg'] = df_conf_evolution.at[conference,'avg_cit']
    if n_editions>1:
        lst_dataframes.append(pd.DataFrame(data=[single_conference.values()],␣
 ↪index = [conference], columns=single_conference.keys()))
df_ratio = pd.concat(lst_dataframes)
df_ratio = df_ratio.reindex(sorted(df_ratio.columns), axis=1)
df_ratio.head(10)
```

Script per la creazione del dataframe 3.8.

Per rappresentare graficamente l'andamento nel tempo delle conferenze sono state implementate le seguenti due funzioni, di cui abbiamo parlato nella sezione 3.3.2:

```
def draw_avg_cit_graph(conf_name):
    plot = sns.barplot(x=list(dict_avg_cit_conference[conf_name].keys()),␣
 ↪y=list(dict_avg_cit_conference[conf_name].values()))
    plot.set_xticklabels(plot.get_xticklabels(), rotation=75)
    plot.set_title(conf_name)
    plot.set(xlabel='YEAR', ylabel='AVG CIT')
    plot.axhline(df_conference.at[conf_name,'CIT AVG'], ls='--')
    plot.text(0.5,int(df_conference.at[conf_name,'CIT AVG'])+0.2, "total avg")
```

```
def draw_tot_cit_graph(conf_name):
    plot = sns.barplot(x=list(dict_cit_conference[conf_name].keys()),␣
 ↪y=list(dict_cit_conference[conf_name].values()))
    plot.set_xticklabels(plot.get_xticklabels(), rotation=75)
    plot.set_title(conf_name)
    plot.set(xlabel='YEAR', ylabel='CIT')
```

I grafici restituiti da queste funzioni saranno illustrati nel capitolo successivo.

4.3 Estrazione dei dati turistici

Funzione *get_place* introdotta nel paragrafo 3.4:

```
def get_place(link):
    html = urlopen(link)
    bsh = BeautifulSoup(html.read(), 'html.parser').h1
    try:
        return str(bsh).split(':')[1].split('<')[0]
    except:
        return None
```

Script per estrazione luogo conferenza.

Funzione che genera il link per ogni conferenza da passare in input a *get_place*:

```
link = 'https://dblp.org/db/'
d = {}
for x in list(df_single_conference.index.values):
    try:
        s = str(x).split('/')
        url = link + s[0] + '/' + s[1] + '/' + s[1]+s[2] + '.html'
    except:
        pass
    try:
        d[x] = get_place(url)
    except:
        d[x] = None
```

Script per la creazione del dataframe 3.9

.

Concatenando il dataframe 3.9 con il 3.1 otteniamo il dataframe 3.10:

```
df_single_conference_whit_place = pd.concat([df_single_conference,␣
 ↪df_place_of_conf],axis=1)
df_single_conference_whit_place.head(10)
```

Script per la creazione del dataframe 3.10.

```
df_sum_cit_by_place = df_single_conference_whit_place.groupby('city')['CIT␣
↪TOT'].sum()
df_count_cit_by_place = df_single_conference_whit_place.groupby('city')['city'].
↪count()
df_mean_cit_by_place = df_single_conference_whit_place.groupby('city')['CIT␣
↪TOT'].mean()
dict_cit_by_city = {}
for k,v in df_sum_cit_by_place.items():
    if not(k == 'None'):
        dict_cit_by_city[k] = [df_mean_cit_by_place[k], v,␣
↪df_count_cit_by_place[k]]
dict_cit_by_city = {k: v for k, v in sorted(dict_cit_by_city.items(),␣
↪key=lambda item: item[1], reverse=True)}
df_cit_by_city = pd.DataFrame(data = dict_cit_by_city.values(), index =␣
↪dict_cit_by_city.keys(), columns = ['avg cit', 'tot cit', 'editions'])
df_cit_by_city = df_cit_by_city.sort_values('editions', ascending=False)
df_cit_by_city.head(10)
```

Script per la creazione del dataframe 3.11.

Ora non ci resta che estrarre la dimensione della pagina wikipedia delle città in cui si sono svolte delle conferenze che stiamo analizzando:

```
size_wikipedia_page = {}
for city in list(df_cit_by_city.index.values):
    if city not in dict_swp.keys():
        page = requests.head('https://en.wikipedia.org/wiki/'+ city)
        c=0
        while page.headers.get ( "content-length" ) == None:
            if c<4:
                page = requests.head('https://en.wikipedia.org/wiki/'+ city)
            else:
                break
            c+=1
        size_wikipedia_page[city]= page.headers.get("content-length")
    else:
        size_wikipedia_page[city]= dict_swp[city]

df_swp = pd.DataFrame(data = size_wikipedia_page.values(), index =␣
↪size_wikipedia_page.keys(), columns = ['swp'])
df_avg_cit_by_place = pd.concat([swp, df_swp],axis=1).set_index('city')
df_avg_cit_by_place.to_csv(path_file +'/swp.csv')
```

Script per estrazione swp.

Concatenando le informazioni appena estratte con il dataframe 3.11 otteniamo il dataframe 3.12.

4.3.2 Informazioni turistiche sugli stati

Come fatto per le città anche per gli stati siamo andati a raggruppare e calcolare citazioni totali, medie e numero di edizioni per ogni stato:

```
df_sum_cit_by_country = df_single_conference_whit_place.groupby('state')['CIT␣
↪TOT'].sum()
df_count_cit_by_country = df_single_conference_whit_place.
↪groupby('state')['city'].count()
df_mean_cit_by_country = df_single_conference_whit_place.groupby('state')['CIT␣
↪TOT'].mean()
dict_cit_by_country = {}
for k,v in df_sum_cit_by_country.items():
    if not(k == 'None'):
        dict_cit_by_country[k] = [df_mean_cit_by_country[k], v,␣
↪df_count_cit_by_country[k]]
dict_cit_by_country = {k: v for k, v in sorted(dict_cit_by_country.items(),␣
↪key=lambda item: item[1], reverse=True)}
df_cit_by_country = pd.DataFrame(data = dict_cit_by_country.values(), index =␣
↪dict_cit_by_country.keys(), columns = ['avg cit', 'tot cit', 'editions'])
df_cit_by_country = df_cit_by_country.sort_values('avg cit', ascending=False)
```

Ottenendo il seguente dataframe:

	avg cit	tot cit	editions
USA	249.77	1539133	6167
China	130.44	223458	1714
Germany	124.61	168601	1356
Italy	150.48	187807	1256

A questo andiamo a concatenare le informazioni turistiche che abbiamo raccolto per ogni stato ottenendo il dataframe 3.13.

4.4 Correlazione

Per il calcolo della correlazione è stata utilizzata la funzione *corr()* di pandas ma non restituiva il p-value e quindi è stato necessario implementare delle funzioni che per ogni metodo di correlazione calcolasse anche il p-value:

```
def kendall_pval(x,y):
    return stats.kendalltau(x,y)[1]

def pearsonr_pval(x,y):
    return stats.pearsonr(x,y)[1]

def spearmanr_pval(x,y):
    return stats.spearmanr(x,y)[1]
```

funzioni per calcolo del p-value.

Ora che abbiamo tutti gli strumenti per calcolare la correlazione non ci resta che invocare la funzione direttamente sui dataframe:

```
corr_pearson = df_cit_and_swp.corr(method= 'pearson')
p_value = df_cit_and_swp.corr(method= pearsonr_pval)

mask = np.triu(np.ones_like(corr_pearson, dtype=float))

f, ax = plt.subplots(figsize=(9, 7))

cmap = sns.diverging_palette(230, 15, as_cmap=True)

sns.heatmap(corr_pearson, mask=mask, cmap=cmap, vmax=1, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5}, annot=True,␣
 ↪annot_kws = {'va': 'bottom'})

sns.heatmap(corr_pearson, mask=mask, cmap=cmap, vmax=1, center=0,
            square=True, linewidths=.5, cbar = False, annot = p_value,␣
 ↪annot_kws = {'va': 'top'})
```

Script1 per calcolo della correlazione.

Una volta calcolato il coefficiente di correlazione e il relativo p-value i dati vengono rappresentati graficamente utilizzando la funzione *heatmap* di seaborn.
Così come per le città è stata calcolata la correlazione anche utilizzando il dataframe 3.13:

```
corr_spearman = df_cit_and_country_index.corr(method= 'spearman')
p_value = df_cit_and_country_index.corr(method= spearmanr_pval)

mask = np.triu(np.ones_like(corr_spearman, dtype=float))
f, ax = plt.subplots(figsize=(11, 9))
cmap = sns.diverging_palette(230, 15, as_cmap=True)

sns.heatmap(corr_spearman, mask=mask, cmap=cmap, vmax=1, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5}, annot=True,␣
 ↪annot_kws = {'va':'bottom'})
sns.heatmap(corr_pearson, mask=mask, cmap=cmap, vmax=1, center=0,
            square=True, linewidths=.5, cbar = False, annot=p_value, annot_kws␣
 ↪= {'va':'top'})
```

Script2 per calcolo della correlazione.

Sia per le città che per gli stati sono stati calcolati i coefficienti utilizzando tutte tre i metodi a disposizione, quindi Spearman, Kendall e Pearson.

4.4.2 Correlazione per singola conferenza

Come già detto nella sezione 3.5.4 la correlazione è stata calcolata anche per singola conferenza:

```
df_year = pd.read_csv(path_file + '/year.csv').set_index('crossref')
def conference_city_corr(conference):
    d_swp = {}
    d_avg_cit = {}
    d_tot_cit = {}
    d = {}
    for crossref,v in df_place_of_conf.iterrows():
        for column,val in v.items():
            if(val==conference):
                if df_place_of_conf.at[crossref,'city'] in list (df_swp.index.
↪values):
                    d_swp[crossref] = df_swp.at[df_place_of_conf.
↪at[crossref,'city'], 'swp']
                    try:
                        d_avg_cit[crossref] = avg_cit_evolution.at[conference,␣
↪str(df_year.at[crossref, 'year'])]
                        d_tot_cit[crossref] = cit_evolution.at[conference,␣
↪str(df_year.at[crossref, 'year'])]
                    except KeyError:
                        d['swp-avg_cit'] = 'ERROR: editions < 2'
                        d['swp-avg_cit_pvalue'] = 'ERROR: editions < 2'
                        d['edtions'] = 1
                        return d
    lst_swp = []
    lst_avg_cit = []
    lst_tot_cit = []
    for k,v in d_swp.items():
        if(v != 'None' and d_avg_cit[k] != 'None' and d_tot_cit[k] != 'None'):
            lst_swp.append(v)
            lst_avg_cit.append(d_avg_cit[k])
            lst_tot_cit.append(d_tot_cit[k])
    if len(lst_swp)<2 and len(lst_avg_cit)<2:
        d['swp-avg_cit'] = 'ERROR: len list < 2'
        d['swp-avg_cit_pvalue'] = 'ERROR: len list < 2'
        d['swp-tot_cit'] = 'ERROR: len list < 2'
        d['swp-tot_cit_pvalue'] = 'ERROR: len list < 2'
        d['edtions'] = len(lst_swp)
    elif lst_swp.count(lst_swp[0]) == len(lst_swp):
        d['swp-avg_cit'] = 'ERROR: An input array is constant'
        d['swp-avg_cit_pvalue'] = 'ERROR: An input array is constant'
        d['swp-tot_cit'] = 'ERROR: An input array is constant'
        d['swp-tot_cit_pvalue'] = 'ERROR: An input array is constant'
        d['edtions'] = len(lst_swp)
    else:
        pearson_avg_cit = stats.spearmanr(lst_swp, lst_avg_cit)
        pearson_tot_cit = stats.spearmanr(lst_swp, lst_tot_cit)
        d['swp-avg_cit'] = pearson_avg_cit[0]
```

```
        d['swp-avg_cit_pvalue'] = pearson_avg_cit[1]
        d['swp-tot_cit'] = pearson_tot_cit[0]
        d['swp-tot_cit_pvalue'] = pearson_tot_cit[1]
        d['edtions'] = len(lst_swp)
    return d
```

Funzione che calcola la correlazione di ogni conferenza.

Una volta implementata la funzione non ci resta che dare in input ogni singola conferenza:

```
d = {}
for conf in avg_cit_evolution.index.values:
    d[conf] = conference_city_corr(conf)
lst = []
for k,v in d.items():
    val = d[k].values()
    key = d[k].keys()
    lst.append(pd.DataFrame(data = [val], index = [k], columns = key))
df_conf_corr = pd.concat(lst)
df_conf_corr.to_csv(path_file + '/corr_city_conf_kendall.csv')
```

Script per la creazione del dataframe 3.14

Anche in questo caso la stessa operazione è stata ripetuta per ogni metodo di correlazione sia usando gli attributi che riguardano la città sia per quelli che riguardano lo stato.
Infine per ogni colonna del dataframe è stato calcolato la media e la moda:

```
df_corr_city_conf_pearson.mean()
df_corr_city_conf_pearson.mode()
```

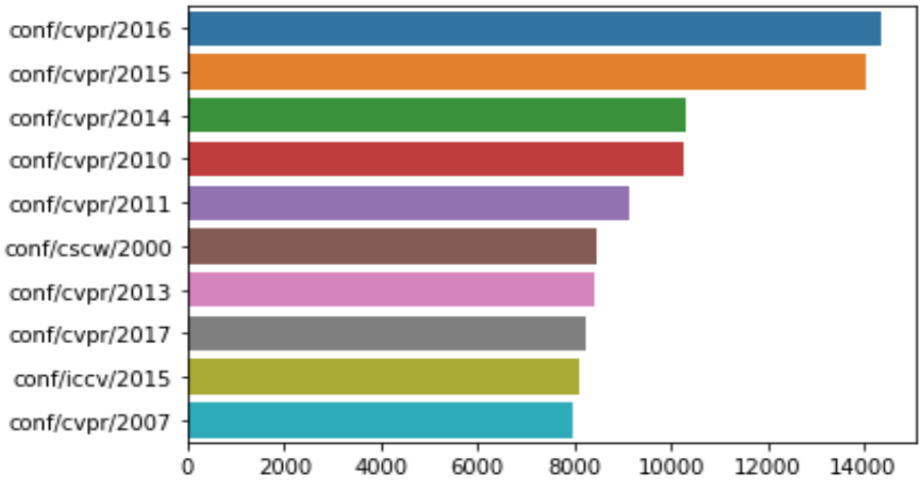
Capitolo 5

Risultati

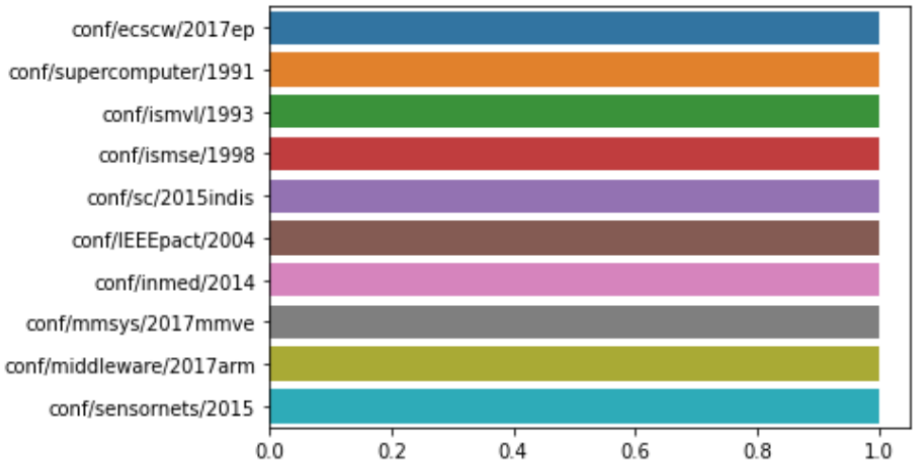
In questo capitolo saranno illustrati i risultati ottenuti, in particolar modo i grafici generati durante la correlazione dei dati.

5.1 Classifiche

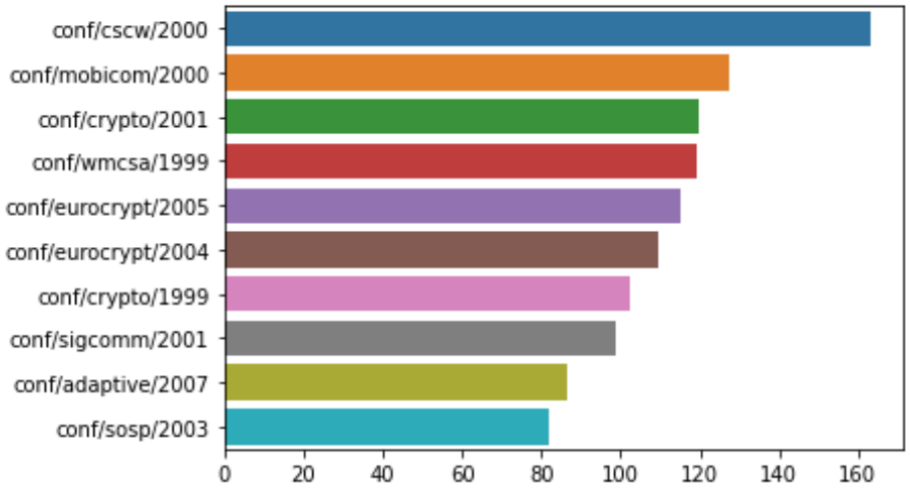
- Le migliori 10 edizioni per numero di citazioni totali:



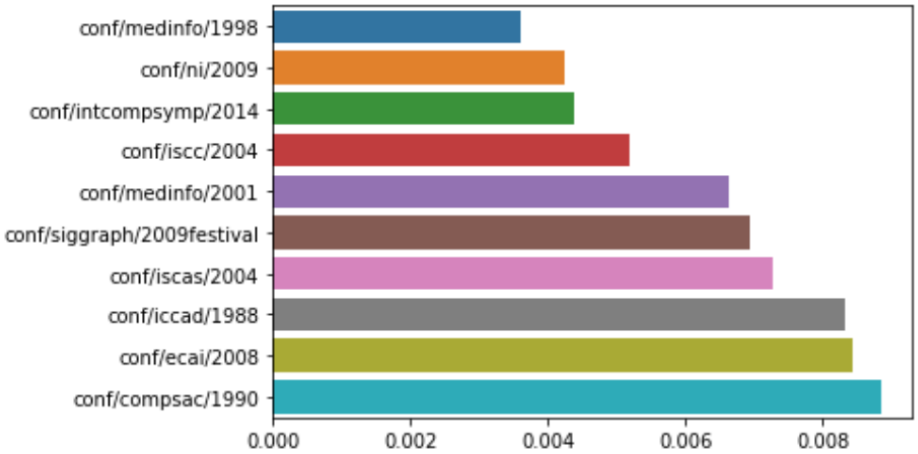
- Le peggiori 10 edizioni per numero di citazioni totali:



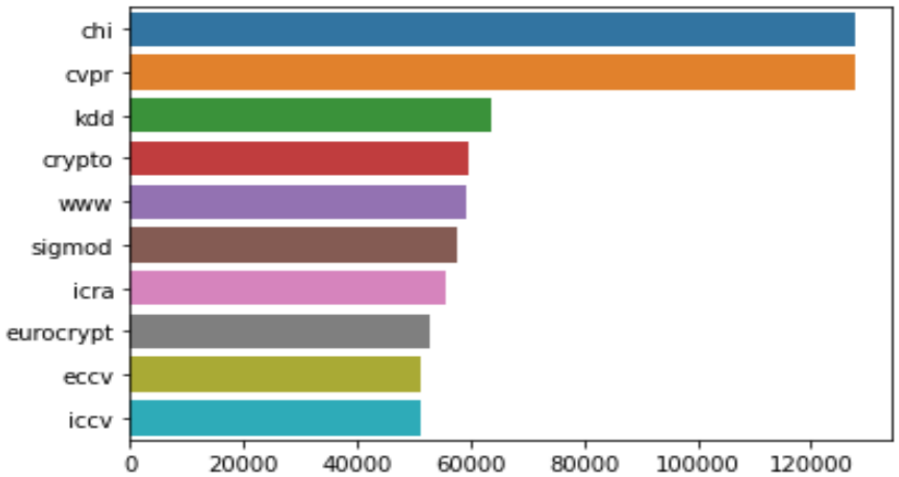
- Le migliori 10 edizioni per numero di citazioni medie:



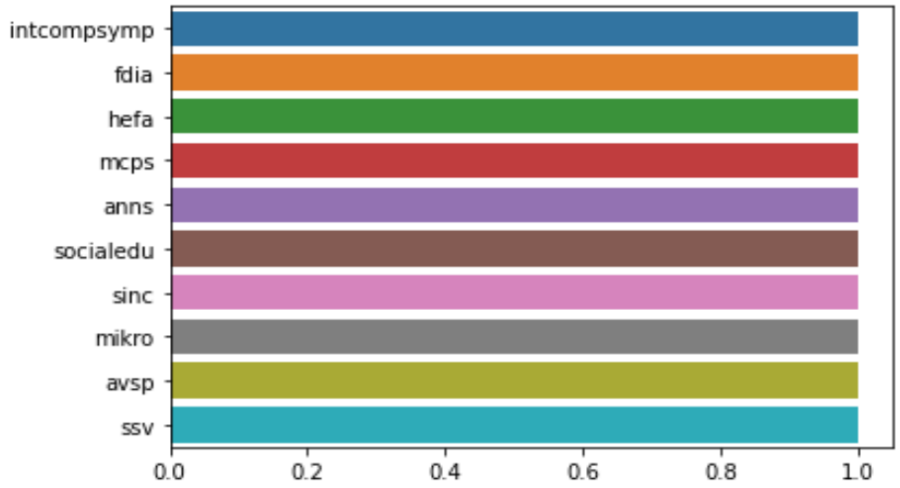
- Le peggiori 10 edizioni per numero di citazioni medie:



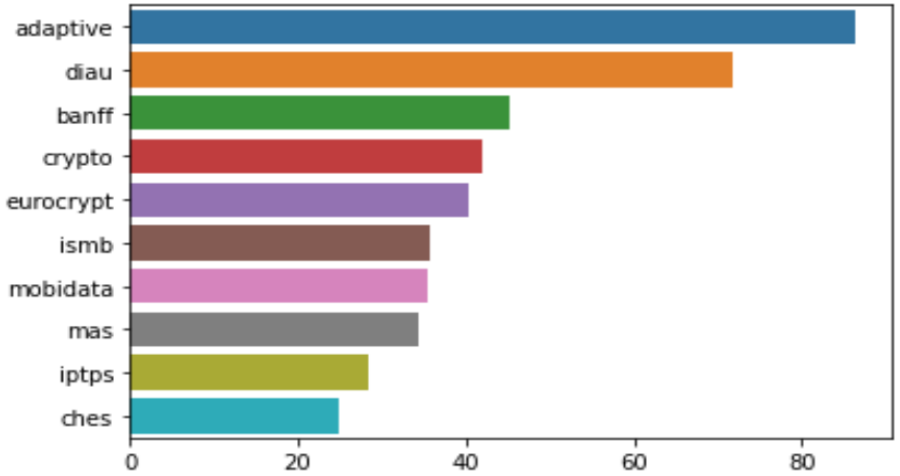
- Le migliori 10 conferenze per numero di citazioni totali:



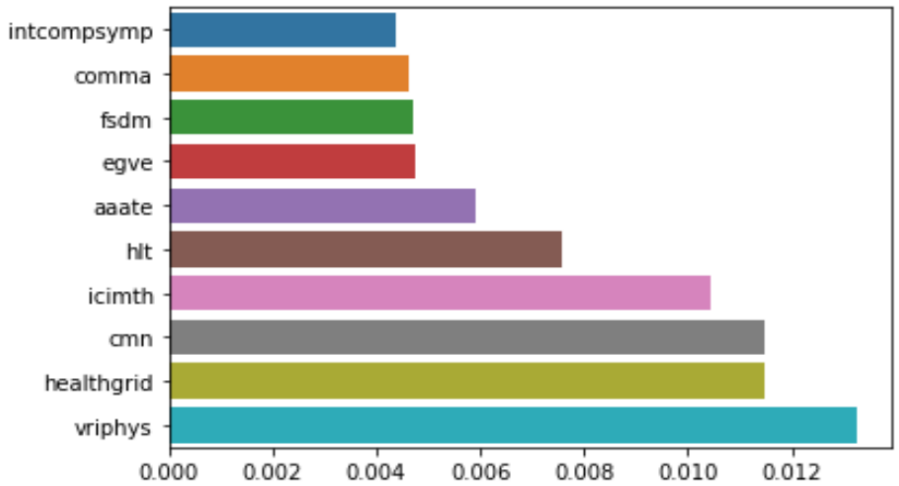
- Le peggiori 10 conferenze per numero di citazioni totali:



- Le migliori 10 conferenze per numero di citazioni medie:



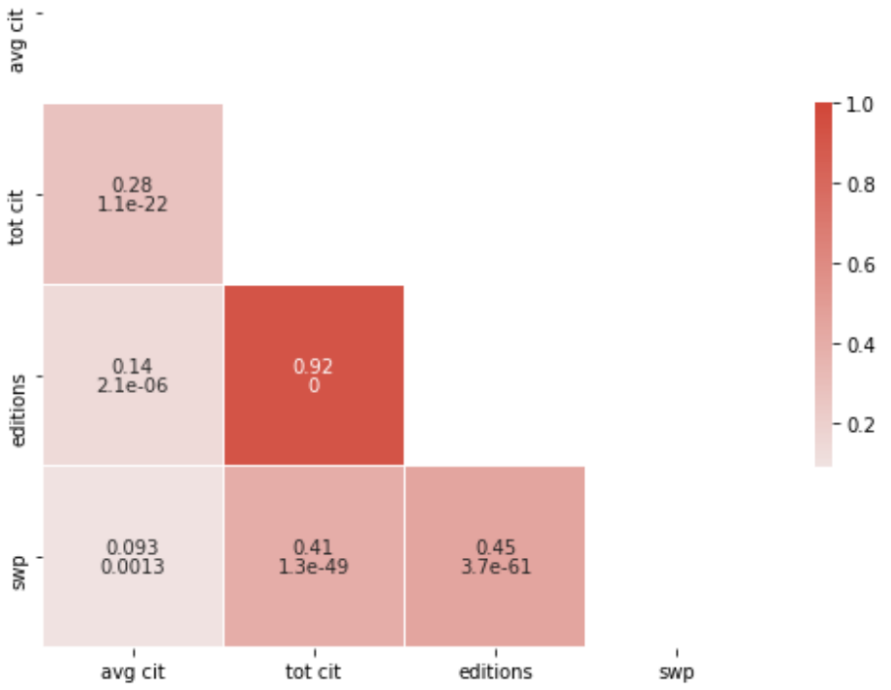
- Le peggiori 10 conferenze per numero di citazioni medie:



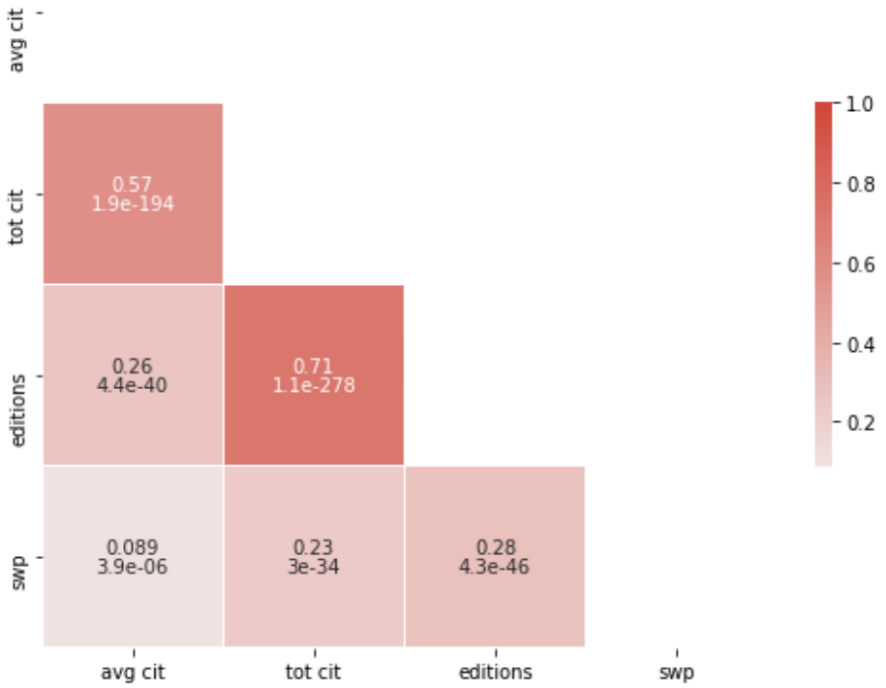
5.2 Risultati della correlazione

Per calcolare la correlazione, come è già stato accennato nel paragrafo 3.5, è stato utilizzato la funzione corr() di pandas. Questa funzione, dato in input un metodo e un dataframe calcola il coefficiente di correlazione per ogni coppia di attributi, quindi per ogni coppia di colonne del dataframe. Come si può vedere nello script nella sezione 4.4 per ogni coefficiente viene calcolato anche il relativo p-value.
Nel grafico il valore in alto è il coefficiente di correlazione, mentre quello appena sotto è il p-value. Per ogni grafico verrà indicato il metodo, il dataframe utilizzato e eventuali dati esclusi dal datafame.

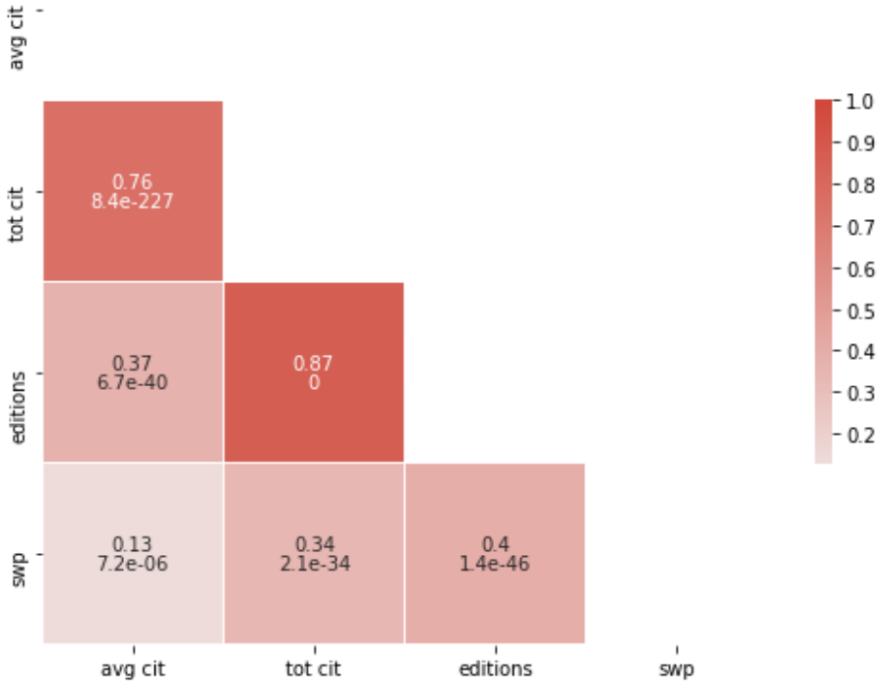
- Correlazione 1 - dataframe: 3.12 - metodo: pearson



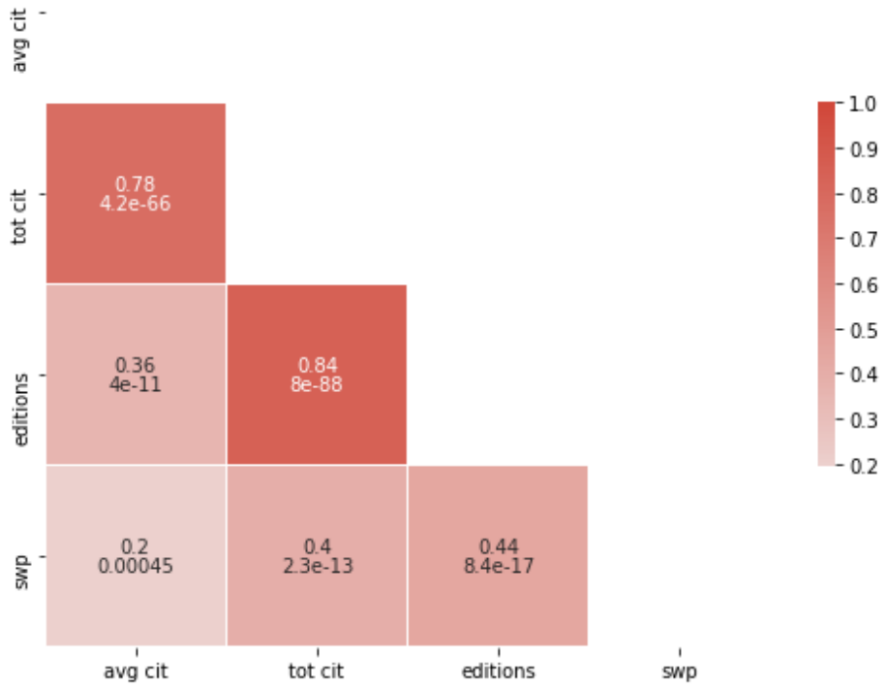
- Correlazione 2 - dataframe: 3.12 - metodo: kendall



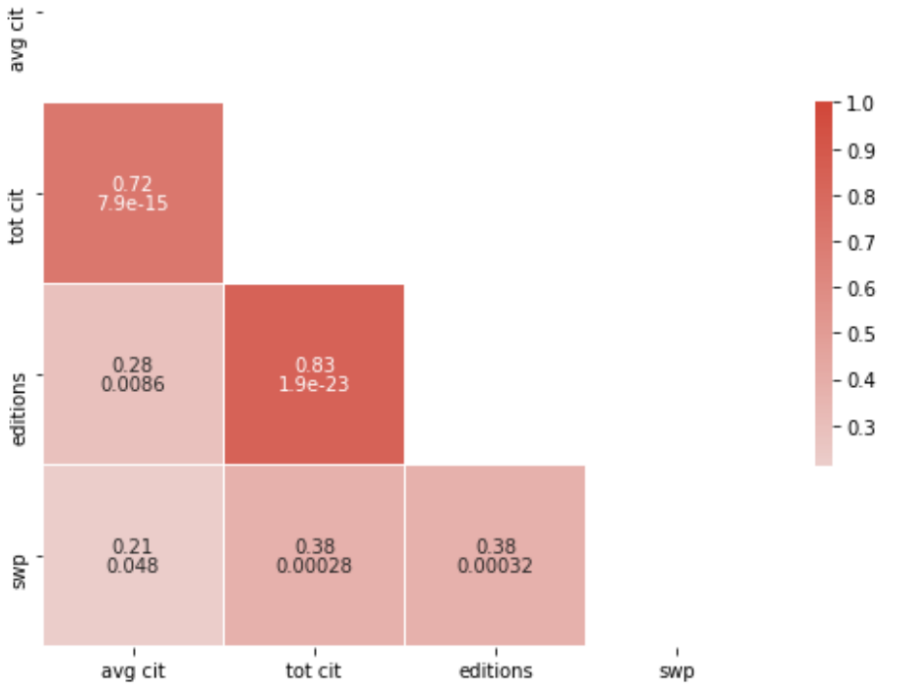
- Correlazione 3 - dataframe: 3.12 - metodo: spearman



- Correlazione 4 - dataframe: 3.12 (solo le città in cui si sono tenute almeno 14 edizioni) - metodo: spearman

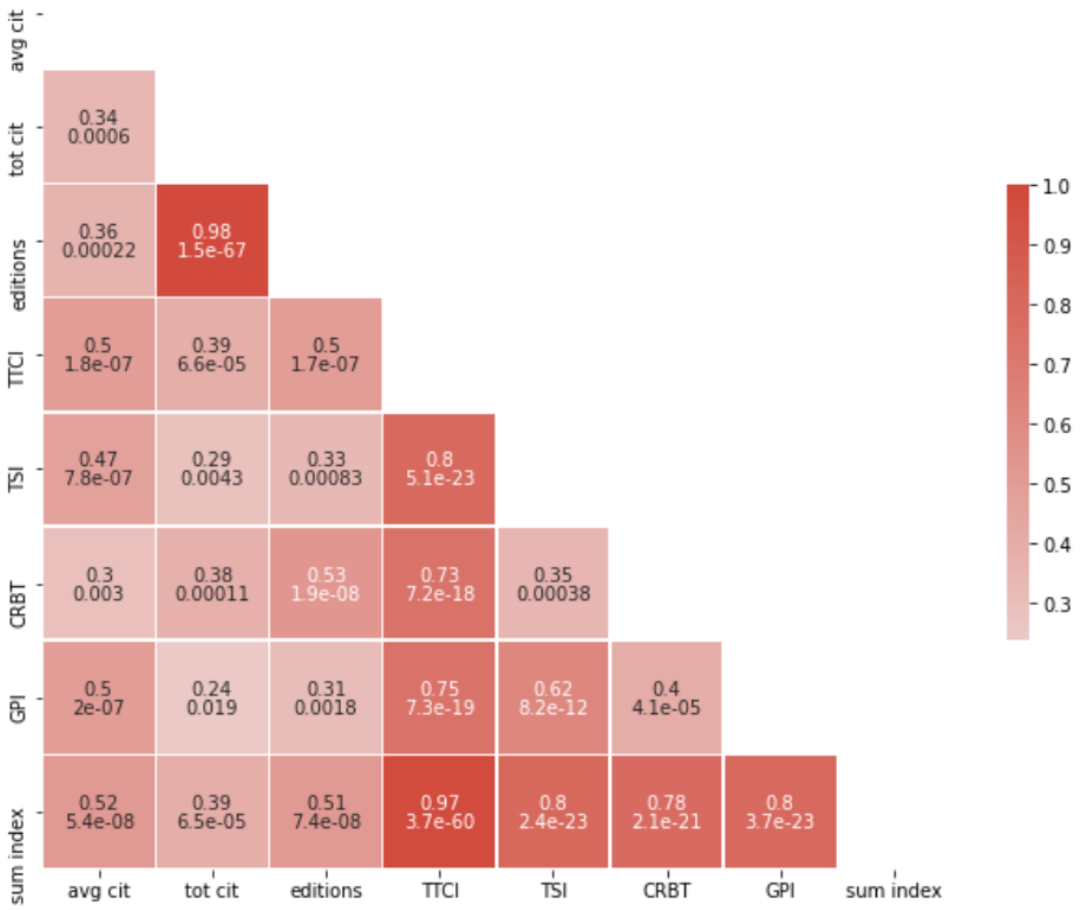


- Correlazione 5 - dataframe: 3.12 (solo le città in cui si sono tenute almeno 57 edizioni) - metodo: spearman

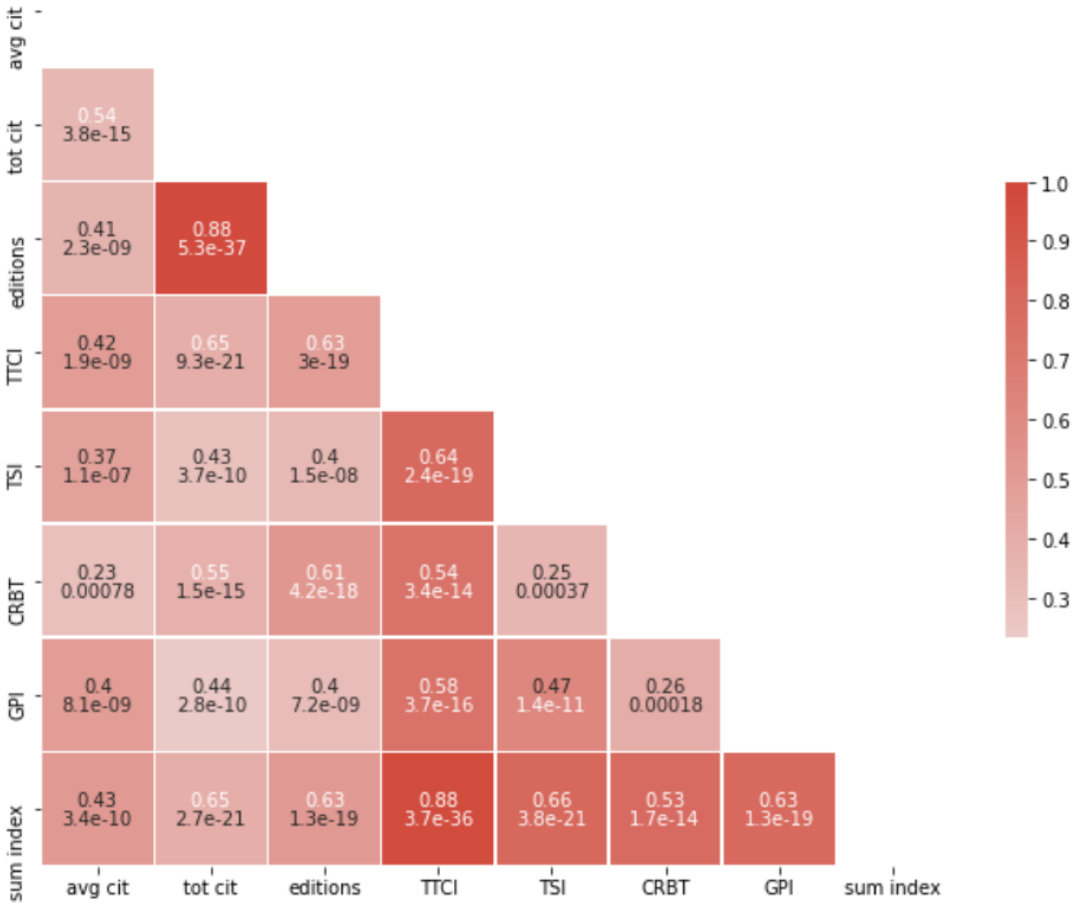


Sono due i coefficienti principali che ci interessano, ovvero il coefficiente di correlazione tra swp e avg cit e tra swp e tot cit. Il fatto che il p-value sia sempre minore di 0.05 è importante e significa che i dati osservati sono statisticamente significativi. Il coefficiente calcolato tra swp e tot cit è intorno allo 0.4, il che vuol dire che abbiamo una buona correlazione tra il numero di citazioni totali e il livello di attrazione turistica della città, in termini di swp, in cui si è svolta la conferenza. Questo significa che in molti casi al crescere delle dimensione della pagina wikipedia crescono anche le citazioni. Il coefficiente calcolato tra swp e avg cit è invece più basso, intorno allo 0.1 e sale fino a 0.2 quando escludiamo le città in cui si sono tenute poche conferenze. Il fatto che le citazioni totali abbiano una buona correlazione mentre le citazioni medie leggermente meno può significare che le conferenze tenuti in grandi città tendono ad avere più articoli il che può impattare in modo positivo sulle tot cit e in modo negativo sulle avg cit. Osservando il coefficiente di correlazione tra swp e editions, intorno allo 0.45, possiamo affermare che nel scegliere il luogo di una conferenza si predilige una città con un alto livello turistico.

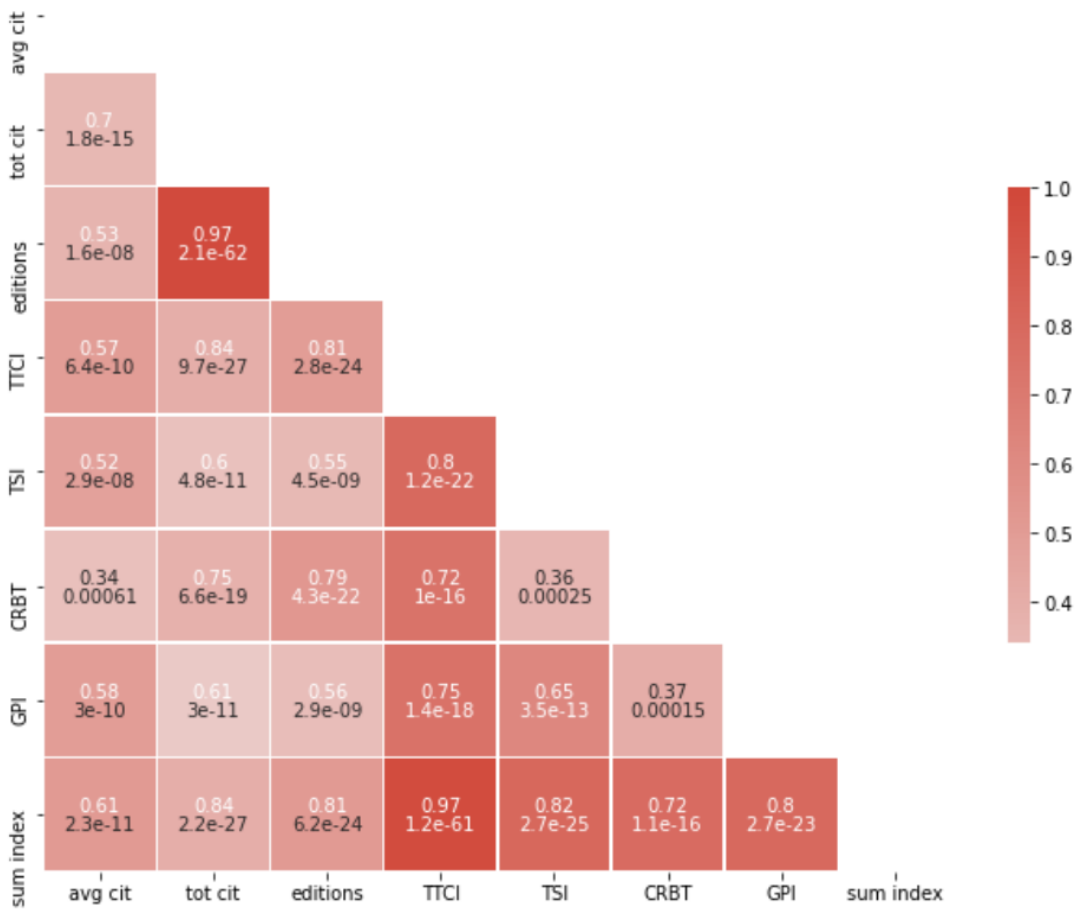
- Correlazione 6 - dataframe: 3.13 - metodo: pearson



- Correlazione 7 - dataframe: 3.13 - metodo: kendall



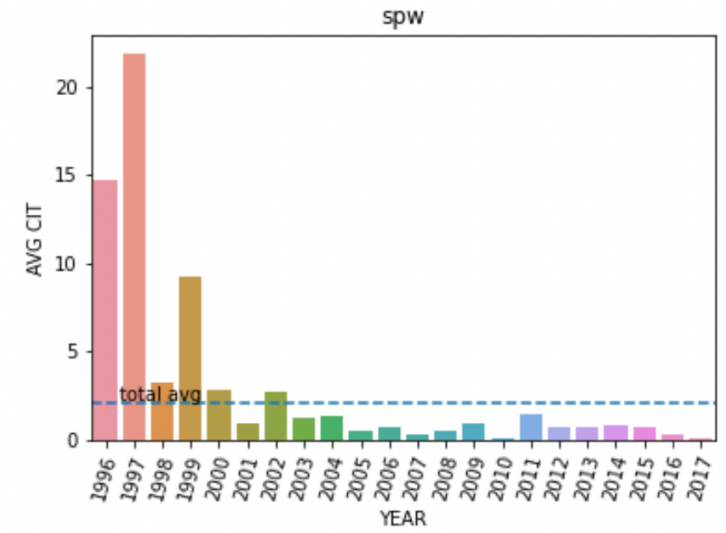
- Correlazione 8 - dataframe: 3.13 - metodo: spearman



In questo caso siamo riusciti ad individuare ed estrarre informazioni più complete e significative per quanto riguarda il livello di attrazione turistica di uno stato, il che è stato impossibile da fare per ogni città. Il p-value è sempre bassissimo, praticamente uguale a zero, quindi possiamo ritenerci soddisfatti dei risultati ottenuti.
Tutti i coefficienti calcolati sono importanti, ma la somma dei 4 indici è probabilmente il più significativo perché è quello che rappresenta meglio il livello di attrazione turistica. In generale i coefficienti ottenuti sono molto più alti rispetto a quelli calcolati utilizzando lo swp, e questo è sicuramente dovuto al fatto di lavorare con indici più precisi. Anche se andiamo a considerare un'area geografica molto più vasta i risultati ottenuti confermano comunque la nostra tesi. Infatti il coefficiente di correlazione tra avg cit e sum index è 0.61 con il metodo spearman, il che vuol dire che in tantissimi casi al crescere del Travel & Tourism Competitiveness Index, Tourist service infrastructure, Cultural resources and business travel e Ground and port infrastructure tendono a crescere anche le citazioni medie. Il coefficiente tra sum index e tot cit arriva addirittura a 0.84, il che indica quasi una perfetta linearità tra i due attributi. Lo stesso vale per la correlazione tra sum index e editions, quindi così come per le città, nel scegliere il luogo di una conferenza si predilige uno stato con un alto livello di attrazione turistica.

5.3 Analisi conferenze

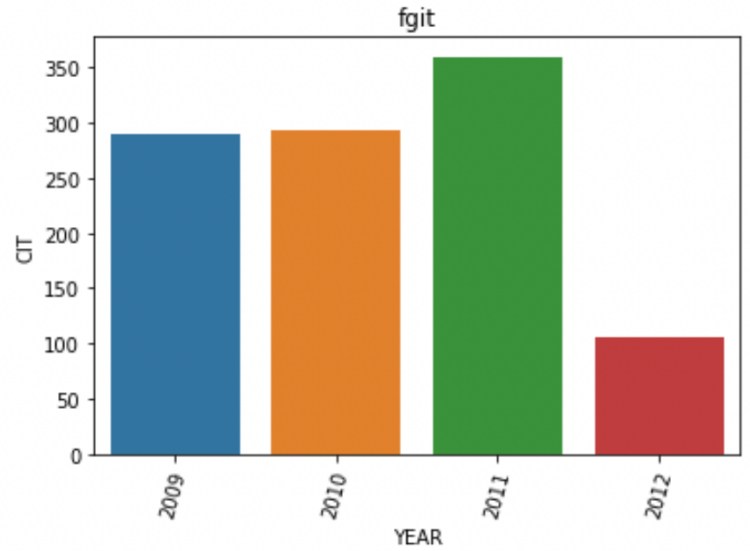
In questo paragrafo andremo ad analizzare più nello specifico alcune conferenze.



Osservando l'andamento delle citazioni medie negli anni salta subito all'occhio l'edizione del 1997. Questa edizione si è tenuta a Parigi in Francia ed è una delle poche insieme a quelle del 2007 e 2016 (entrambe a Brno, Czech Republic) che non si sono organizzate a Cambridge, UK. La dimensione della pagina wikipedia di queste città sono:

- Paris: 218583
- Cambridge: 103502
- Brno: 81748

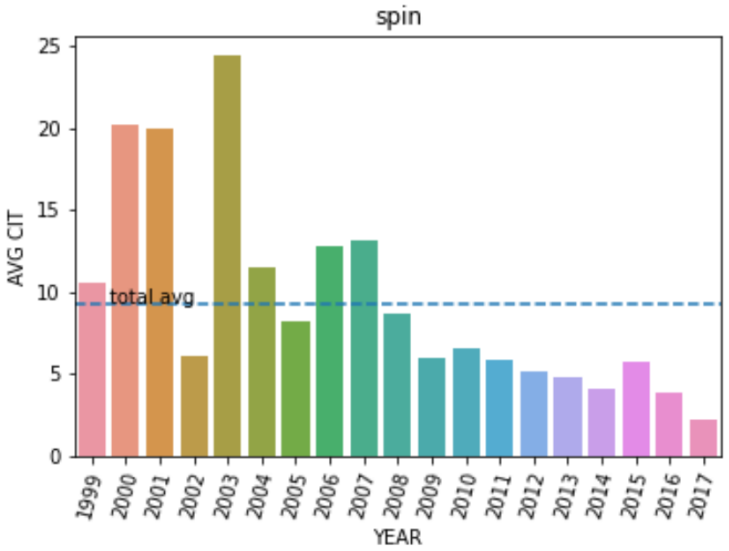
Parigi, tra le tre, è la città più turistica e 'famosa' e infatti l'edizione migliore per citazioni medie si è tenuta proprio in Francia mentre le 2 fatte a Brno sono tra le 4 peggiori.



La Future Generation Information Technology (FGIT) è un'altra conferenza interessante da analizzare nonostante le poche edizioni. Infatti qui le prime 3 edizioni si sono tenute a Jeju Island in Korea mentre l'ultima è stata organizzata a Gangneug sempre in Korea. Le dimensioni delle pagini wikipedia di queste città sono le seguenti:

- Gangneug 6522
- Jeju Island 43542

Anche in questo caso è evidente come le edizioni che si sono tenute nella città con swp più alto hanno avuto più successo.



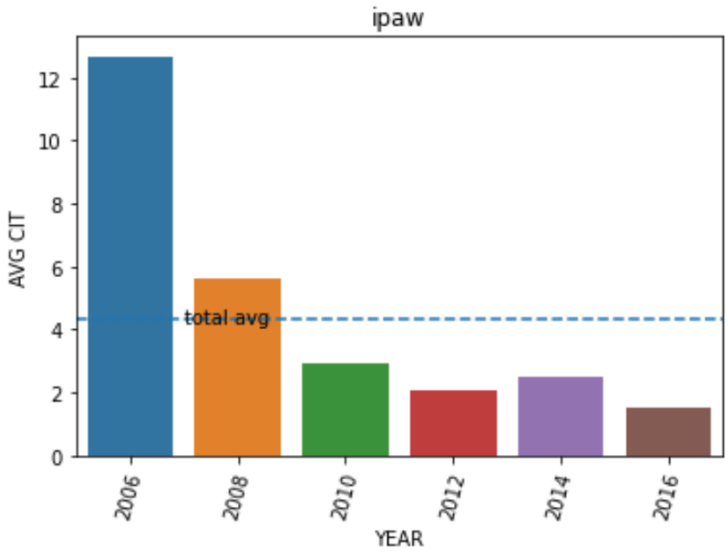
Analizzando questo grafico notiamo subito tre migliori edizioni, ovvero le seguenti:

- 2000: Stanford (115768), CA, USA
- 2001: Toronto (154092), Canada
- 2003: Portland (165786), OR, USA

E' evidente invece come negli ultimi dieci anni sia peggiorata la qualità della conferenza e non a caso anche le città in cui si sono tenute queste conferenze, secondo il nostro parametro, hanno un indice di attrazione minore. Infatti, tralasciando il 2017 che forse è troppo recente e ci può trarre in inganno, queste sono alcune delle ultime edizioni:

- 2011: Snowbird (9225), UT, USA
- 2013: Stony Brook (10041), NY, USA
- 2015: Stellenbosch (34508), South Africa
- 2016: Eindhoven (64028), The Netherlands

Un'altra edizione interessante da analizzare è quella del 2002 perché a differenza delle altre che sono peggiorate di qualità nel tempo questa di trova proprio in mezzo tra le due conferenze migliori. Questa edizione infatti si è tenuta a Grenoble in France che ha dimensione della pagina Wikipedia pari a 77023 e quindi quasi la metà rispetto a Toronto e Portland dove la qualità è più che doppia. Infine ci sono altre due edizioni che spiccano in mezzo alle altre, ovvero quella del 2006 e 2007 che sono state organizzate rispettivamente a Vienna (120905) e Berlino (160847).



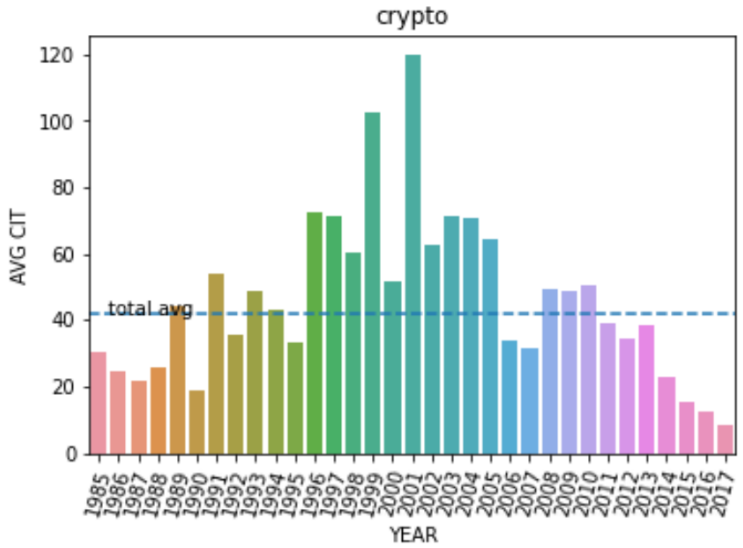
- 2006: Chicago (206303), IL, USA
- 2008: Salt Lake City (140782), UT, USA
- 2010: Troy (135354), NY, USA
- 2012: Santa Barbara (76923), CA, USA
- 2014: Cologne (93960), Germany
- 2016: McLean (15520), VA, USA

Anche in questo caso è evidente come l'edizione nella città più turistica ha nettamente maggior successo rispetto alle altre.
Generalizzando possiamo dire che ogni edizione è leggermente sopra o sotto la media tranne qualcuna che spicca sulle altre in positivo o in negativo. Molto spesso l'edizione che ha avuto più successo è nella città con maggior attrazione turistica. In generale nelle città con dimensione della pagina wikipedia (swp) tra i 50.000 e i 120.000 troviamo edizioni che rispettano la media, mentre molto spesso se una delle conferenze si è tenuta in un luogo con swp maggiore di 150.000 questa spicca sulle altre.

Torniamo ad analizzare le migliori 10 edizioni per citazioni medie:

- cscw 2000: Philadelphia (177336), Pennsylvania, USA
- mobicom 2000: Boston (162528), Massachusetts, USA
- crypto 2001: Santa Barbara (76923), California, USA
- wmcsa 1999: New Orleans (149732), LA, USA
- eurocrypt 2005: Aarhus (142233), Denmark
- eurocrypt 2004: Interlaken (46678), Switzerland
- crypto 1999: Santa Barbara (76923), California, USA
- sigcomm 2001: San Diego (140344), CA, USA
- sosp 2003: The Sagamore (14928), Bolton Landing (Lake George), New York, USA
- mobicom 2002: Atlanta (151558), Georgia, USA

Sei delle top dieci conferenze si sono tenute in città con swp maggiore di 140.000, il che non ci stupisce. Per quanto riguarda The Sagamore non è una città ma un resort su un'isola privata nel Lago di George, quindi nonostante il basso swp è sicuramente una meta molto affascinate per un turista. Un discorso simile si può fare per Interlaken, un comune svizzero di 5592 abitanti, quindi non una grande città ma sicuramente affascinate da visitare, inoltre la Svizzara è nella top dieci dei paesi per TTCI (Travel & Tourism Competitiveness Index). Infine, tutte le edizioni di crypto, non solo quelle qua sopra, sono state organizzate a Santa Barbara quindi non si può valutare quanto il luogo incida sulla qualità.
Il fatto che crypto, mobicom e eurocrypt compaiano più di una volta nella top dieci significa che l'argomento della conferenza è il fattore principale che determina la qualità media delle varie edizioni. Il luogo può incidere in modo positivo o negativo sulla qualità, ma è praticamente impossibile che una conferenza come 'spin' (analizzata in precedenza) con media citazioni totali di tutte le edizioni intorno al 9 arrivi a toccare 100 o 150 citazioni medie solo perché quell'edizione si è tenuta in una città molto turistica. E' possibile, invece, che l'argomento di una conferenza diventi più interessante, perché magari più di attualità, e quindi può avvere una crescita improvvisa, anche in città meno turistiche rispetto alle precedenti edizioni. Può essere una crescita continua che fa salire da 10 a 100 le citazioni medie in 5/10 anni, oppure è semplicemente un 'boom' di quell'anno. Ovviamente può succedere anche il contrario, cioè che un argomento perda d'interesse. Un esempio può essere crypto che nonostante venga organizzata sempre a Santa Barbara, in California, presenta diverse differenza tra le edizioni dal 1996 al 2005 rispetto alle altre, come si può vedere dal grafico sottostante:



Ora andiamo ad estrarre alcuni dati dai dataframe che abbiamo creato soprattutto durante la prima fase di elaborazione. Per esempio utilizzando df_ratio e estraendo quelle con il rapporto più alto possiamo individuare le edizioni che hanno superato le aspettative. Prendiamo le 10 edizioni col rapporto più alto:

- ispw 1985: Coto de Caza (31685), Trabuco Canyon, CA (198721), USA
- sigdoc 1986: Toronto (154092), Ontario, Canada

- ecai 1996: Budapest (161834), Hungary
- spw 1997: Paris (218583), France
- wmcsa 1999: New Orleans (149732), LA (156569), USA
- cscw 2000: Philadelphia (177336), Pennsylvania, USA
- siggraph 2000: New Orleans (149732), LA (156569), USA
- middleware 2001: Heidelberg (71906), Germany
- siggraph 2001: Los Angeles (146569), CA (198721), USA
- ccia 2002: Castellón (9951), Spain

Anche in questo caso il risultato è quello che ci aspettavamo, infatti praticamente tutte le 10 conferenze prese in considerazione si sono tenute o in grandi città turistiche quindi con un alto swp o in uno stato con un alto Travel & Tourism Competitiveness Index come la Spagna o la Germania che sono prime in questa classifica con un TTCI pari a 5.4.

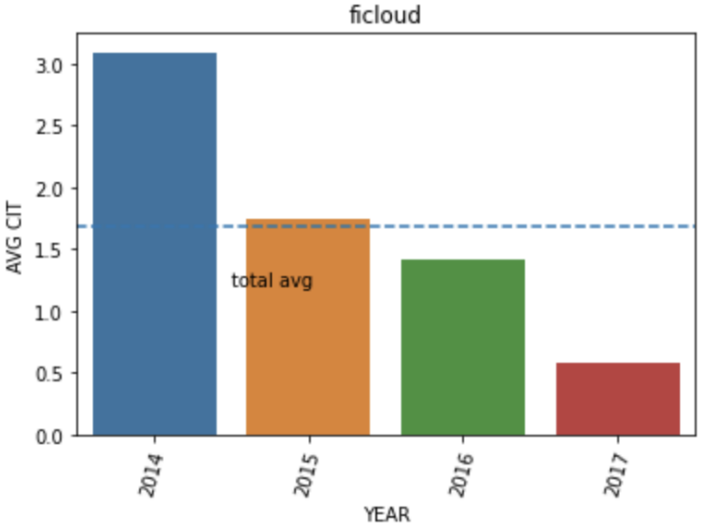
Un'altra idea può essere quella di confrontare l'edizione migliore e la peggiore di alcune conferenze. Utilizzando il dataframe 3.7 si possono estrarre le conferenza con maggior differenza tra l'edizione che ha avuto maggiore successo e quella con minor successo. Ci aspettiamo che le edizioni migliori si siano tenute in posti con alta attrazione turistica. Attenzione però non ci dobbiamo per forza aspettare che l'edizione peggiore sia in un posto con basso swp, questo è vero in parecchi casi ma non sempre.
Le 5 conferenze con maggior differenza sono le seguenti:

- CSCW:
  - peggior edizione: 1986, Austin (163678), TX, USA
  - miglior edizione: 2000,Philadelphia (177336), Pennsylvania, USA
- MOBICOM:
  - peggior edizione: 2017, Snowbird (9225), UT, USA
  - miglior edizione: 2000, Boston (162528), Massachusetts, USA
- WMCSA:
  - peggior edizione: 2006, Semiahamoo Resort (13031), Washington, USA
  - miglior edizione: 1999, New Orleans (149732), LA, USA
- CRYPTO:
  - peggior edizione: 2017, Santa Barbara (76923), CA, USA
  - miglior edizione: 2001, Santa Barbara (76923), CA, USA
- EUROCRYPT:
  - peggior edizione: 1985, Linz (76898), Austria
  - miglior edizione: 2005, Aarhus (142233), Denmark

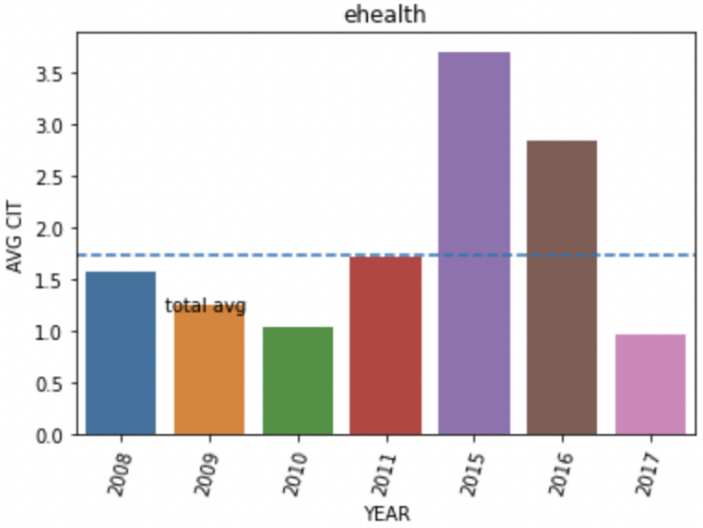
Osservando i risultati si possono notare due cose:

- ci sono esempi come MOBICOM, WMCSA, EUROCRYPT che confermano la nostra tesi, ovvero che le edizioni migliori si tengono in città con alto swp;
- CSCW e CRYPTO invece sono due esempi di come a distanza di 15 anni un argomento può essere più o meno d'attualità, questa può essere una possibile spiegazione della la grande differenza di citazioni nonostante il luogo non cambi o cambi ma resti comunque una città con un alto livello turistico.

Ora come fatto in precedenza andiamo ad analizzare il grafico di alcune conferenze stavolta però ci concentriamo sullo stato in cui si sono tenute e non la città.



- 2014: Spain 5,4
- 2015: Italy 5,1
- 2016: Austria 5
- 2017: Czech Republic 4,3

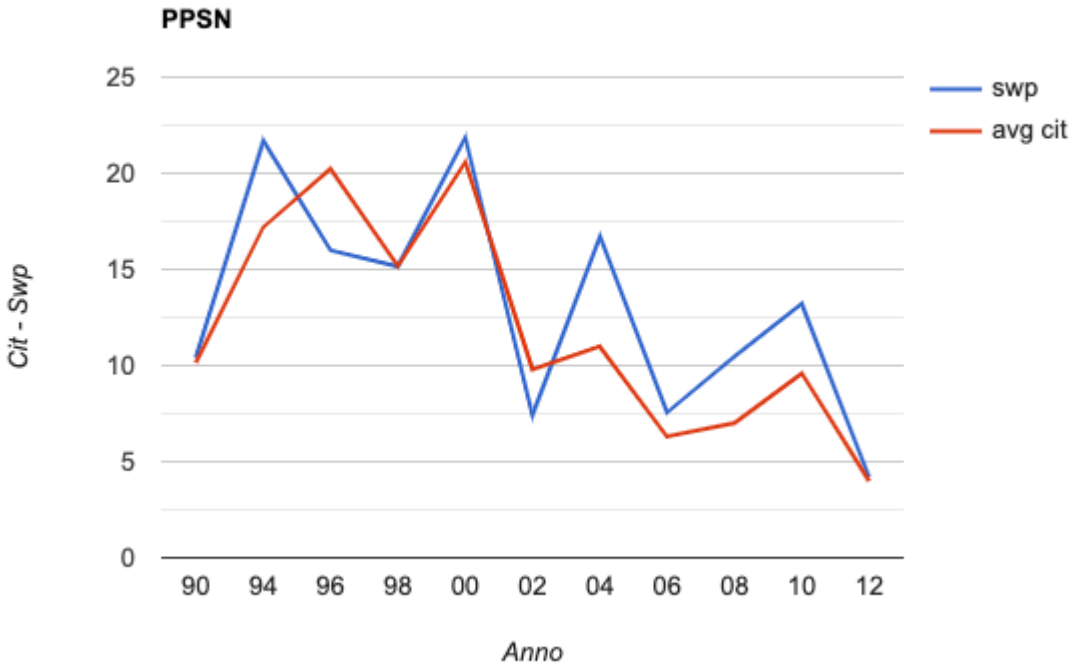


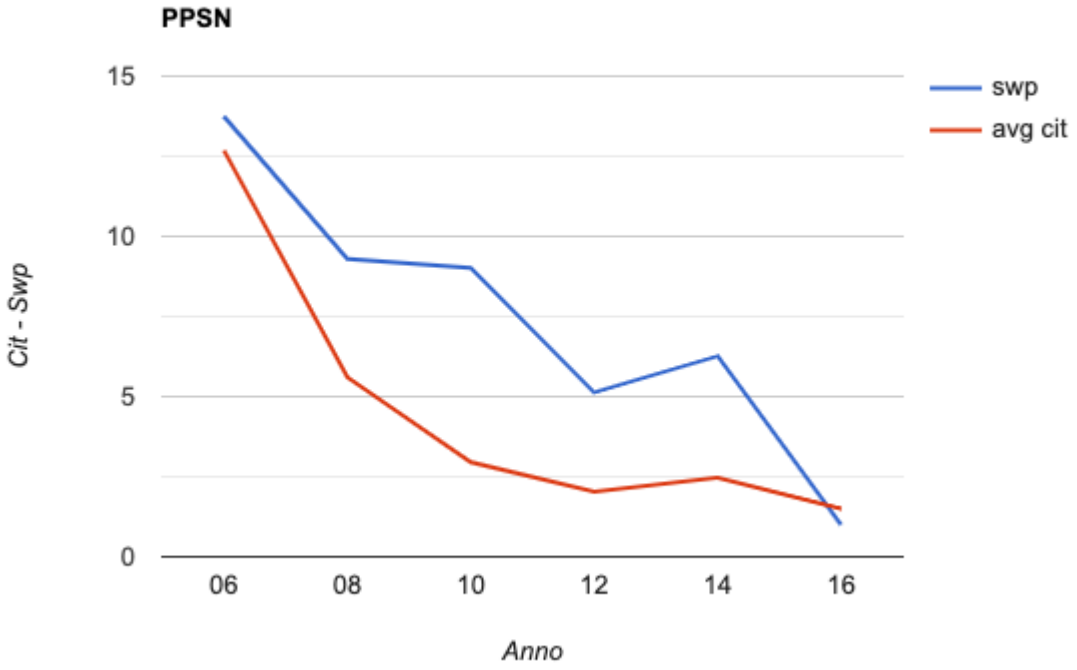
Andiamo a prendere i due anni migliori e i due peggiori, escludendo il 2017 perché potrebbe essere ancora troppo recente.

- 2009: Turkey 4,2
- 2010: Morocco 3,9
- 2015: Italy 5,1
- 2016: Canada 5,1

In entrambi i casi le citazioni medie si alzano nei paesi con un alto indice di turismo.

Un altro aspetto interessante da valutare potrebbe essere l'andamento delle citazioni medie rispetto allo swp della città cui si è tenuta la conferenza, in questo modo:





Possiamo notare come l'andamento delle due variabili prese in considerazione sono molto simili, quasi lineari.

Conclusioni

Il nostro obiettivo è stabilire se esiste una correlazione tra il successo di una conferenza e il luogo dove è stata fatta, abbiamo preso in considerazione circa 30000 edizioni di quasi 4000 conferenze diverse. Dopo aver pulito ed estratto i dati utile sulle varie conferenze abbiamo raccolto il luogo di ognuna di esse. Dal calcolo del coefficiente di correlazione tra gli attributi bibliografici e gli attributi turistici è venuto fuori che c'è una piccola connessione tra il successo di una conferenza e la città dove si è tenuta mentre i risultati ottenuti utilizzando gli indici raccolti per stati hanno completamente confermato la nostra tesi in quanto il coefficiente calcolato tra le citazioni e la somma dei 4 indici oscillano tra lo 0.5 e lo 0.8. Il basso coefficiente di correlazione per le città è dovuto al fatto che non avevamo un indice di attrazione turistica sufficientemente preciso per ognuna, infatti un posto come Honolulu, la capitale delle Hawaii, ha un swp molto inferiore rispetto a Parigi per esempio, questo perché la capitale francese è più grande, più visitata e più facilmente accessibile ma per un turista Honolulu è sicuramente altrettanto affascinante. L'ideale sarebbe avere uno o più valori che misuri in modo preciso il livello di attrazione turistica di ogni città, come è stato fatto per gli stati i quali però si estendono su un'area geografica troppo vasta e quindi non possiamo utilizzare lo stesso parametro per valutare due conferenze, nello stesso paese ma in diverse città. Andando comunque a osservare più da vicino alcune conferenze scelte attentamente abbiamo visto come in verità c'è un legame molto più stretto tra la qualità di una conferenza, in termini di citazioni, e il luogo. Analizzando grafici e tabelle abbiamo visto come molto spesso l'andamento delle citazioni segue l'andamento dello swp e che le migliori conferenze si sono tutte tenute in grandi città turistiche, confermando la nostra tesi. Ovviamente questo non vuol dire che qualsiasi conferenza che si tenga in un luogo con alto indice di attrazione turistica debba andare per forza bene, infatti come già detto in precedenza l'argomento e l'anno sono due fattori determinanti che incidono parecchio sulla qualità ma a parità di questi due la città può influire sul numero di citazioni. Quindi dopo un'attenta analisi possiamo affermare che esiste una correlazione tra il successo di una conferenza e il luogo dove è stata fatta, e infatti tutti i risultati ottenuti vanno in questa direzione. La correlazione con gli indici turistici degli stati sono più che ottimi, così come l'analisi condotta sulle singole conferenze e gli ultimi grafici che descrivono l'andamento delle citazioni rispetto all'andamento dello swp hanno dato i risultati sperati. Anche la correlazione tra le citazioni e lo swp hanno dato buoni risultati, nonostante sia stato dimostrato che il coefficiente potrebbe essere più alto se fosse stato possibile lavorare con un indice di attrazione turistico più preciso che prendesse in considerazione più fattori.

Bibliografia

[1] Data science: https://it.wikipedia.org/wiki/Scienza_dei_dati
[2]Data scientist:
https://www.cwi.it/cio/data-scientist-chi-e-cosa-fa-e-quanto-guadagna_42108239#:~:text=La%20responsabilit%C3%A0%20principale%20di%20un,da%20una%20serie%20di%20fonti.
[3]Big data: https://it.wikipedia.org/wiki/Big_data
[4]Analisi dei dati: https://it.wikipedia.org/wiki/Analisi_dei_dati
[5]Dati aggregati: https://en.wikipedia.org/wiki/Aggregate_data
[6]Linguaggio di programmazione: https://ennebi.solira.org/linprog/pag03.html
[7]Pandas: https://it.wikipedia.org/wiki/Pandas_(software)
[8]NumPy:
https://it.wikipedia.org/wiki/NumPy#:~:text=NumPy%20%C3%A8%20una%20libreria%20open,efficientemente%20su%20queste%20strutture%20dati.
[9]TTCI:
http://reports.weforum.org/travel-and-tourism-competitiveness-report-2019/rankings/#series=TTCI
[10]TSI:
http://reports.weforum.org/travel-and-tourism-competitiveness-report-2019/rankings/#series=TTCI.C.12
[11]CRBT:
http://reports.weforum.org/travel-and-tourism-competitiveness-report-2019/rankings/#series=TTCI.D.14
[12]GPI:
http://reports.weforum.org/travel-and-tourism-competitiveness-report-2019/rankings/#series=TTCI.C.11
[13]Correlazione:
https://www.jmp.com/it_it/statistics-knowledge-portal/what-is-correlation.html#:~:text=La%20correlazione%20descrive%20dati%20che%20cambiano%20insieme&text=Confrontando%20le%20due%20variabili%20con,dell'altitudine%20la%20temperatura%20diminuisce.

Immagini:

[1]all global data in zetabytes: https://www.quora.com/What-is-a-zetabyte
[2]data never sleep: https://www.domo.com/learn/data-never-sleeps-8
[3]more popular data science software:
https://www.kdnuggets.com/2019/06/data-science-jobs-report.html