UNIVERSITÁ POLITECNICA DELLE MARCHE

FACOLTÁ DI ECONOMIA GIORGIO FUÁ

Corso di Laurea Magistrale in Data Science per l'Economia e le Imprese

Interclasse LM 56 - LM 91

Progettazione e sviluppo di un sistema per la raccolta e l'analisi dei dati bibliometrici relativi ai docenti delle università italiane

Design and development of a system for the gathering and analysis of biblometric data of Italian professors

Relatore:
Prof. Domenico Potena

Tesi di Laurea di:
Federica Volpi

Correlatore:
Prof. Diego D'Adda

Anno Accademico 2021 - 2022

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

L'obiettivo del progetto di ricerca descritto nelle successive pagine è quello di verificare se i meccanismi di citazioni e references nelle pubblicazioni accademiche sono mutati nel tempo.

La scelta di questo argomento è stata motivata dall'introduzione del sistema di abilitazione scientifica nazionale con la Legge 240 del 30 Dicembre 2010 [2]. Questa, come recita l'art. 16 della Legge, "attesta la qualificazione scientifica che costituisce requisito necessario per l'accesso alla prima e alla seconda fascia dei professori".

Al fine di determinare l'abilitazione sono stati introdotti degli indicatori che variano a seconda del Settore Scientifico-Disciplinare (o SSD) di appartenenza dei candidati e hanno lo scopo di valutare analiticamente,

grazie a dei valori-soglia, i titoli e le ricerche scientifiche.

Per alcuni SSD, chiamati settori bibliometrici, gli indicatori su cui si basa l'abilitazione derivano in parte dalla bibliometria, disciplina che applica metodi matematici e statistici allo studio della comunicazione scientifica. Come riportato nell'articolo 2 del DM del 08/08/2018, n.589 "Definizione valori - soglia degli indicatori di impatto della produzione scientifica" [1], gli indicatori di riferimento per questi SSD sono:

- Numero di articoli pubblicati nelle banche dati internazionali "Scopus" (il più grande database internazionale bibliografico e multidisciplinare) e "Web of Science" (banca dati citazionale in ambito accademico) rispettivamente, nei dieci e cinque anni prima per la prima e seconda fascia;

- numero di citazioni ricevute da pubblicazioni nelle banche dati internazionali "Scopus" e "Web of Science", rispettivamente, nei quindici e dieci anni precedenti per la prima e seconda fascia;

- l'h-index, o indice di Hirsch, calcolato sulla base del numero di citazioni ricevute con riferimento al numero di articoli pubblicati, rispettivamente, nei quindici e dieci anni precedenti per la prima e seconda fascia.

Gli indicatori a cui fanno riferimento i settori non bibliometrici sono:

- numero di articoli su riviste scientifiche dotate di ISSN e di contributi in volumi dotati di ISBN pubblicati, rispettivamente, nei dieci anni e cinque anni precedenti, per professori di prima e seconda fascia;
- numero di articoli su riviste appartenenti alla classe A pubblicati, rispettivamente, nei quindici anni e dieci anni precedenti, per professori di prima e seconda fascia;
- numero di libri a uno o più autori dotati di ISBN e pubblicati, rispettivamente, nei quindici anni e dieci anni precedenti, per professori di prima e seconda fascia.

É evidente che gli indicatori bibliometrici, basati sul numero delle citazioni, derivano dall'idea che un elevato numero di citazioni rispecchi l'impatto che gli autori citati hanno sugli autori citanti, considerando quindi l'articolo in questione come una pubblicazione di valore.
Nonostante ciò, l'uso di questi indicatori (anche il numero di pubblicazioni) può portare a comportamenti strategici finalizzati principalmente al superamento delle soglie, come documentato da M. Sabeer nel caso delle autocitazioni [3].

Il presente lavoro ha come obiettivo di studio quello di approfondire i comportamenti citazionali degli autori in alcuni settori scientifico-disciplinari. Le informazioni necessarie per effettuare questa tipologia di analisi sono state estratte dal database di Scopus, più ampio di Web of Science, tramite codice identificativo del singolo docente. In particolare, i dati di interesse riguardano le pubblicazioni, le citazioni accumulate negli anni per pubblicazione, l'elenco di titoli nelle references e i loro i rispettivi autori.

Nel Capitolo 2, *Estrazione dell'identificativo Scopus*, si presenteranno i settori scientifico-disciplinari presi in esame e la procedura utilizzata per ottenere il codice identificativo di Scopus da associare a ciascun docente.
Nel Capitolo 3, *Estrazione dei dati relativi a pubblicazioni e references*, verranno descritti in dettaglio gli step effettuati per l'ottenimento dei dati da Scopus e per la costruzione del database.
Nel Capitolo 4, *Analisi empirica*, si presentano alcune analisi e conclusioni tratte dai dati ottenuti e la visualizzazione a grafo dei dati.

Capitolo 2

Estrazione dell'identificativo Scopus

In questo capitolo verranno inizialmente forniti dettagli sugli SSD su cui si è scelto di focalizzarsi. In seguito l'attenzione si sposterà sulla creazione di una procedura di estrazione del codice identificativo di Scopus associato a ogni docente. Successivamente al tuning dei parametri della procedura, il capitolo si concluderà con la presentazione delle combinazioni di parametri ottimali.

2.1 Descrizione dei dati

Al fine di ottenere i dati da utilizzare per l'analisi si è optato per prendere in esame i docenti di tre settori scientifico-disciplinari differenti:

- ING-IND/17, settore che studia le metodologie ed i criteri generali che presiedono alla pianificazione, progettazione, realizzazione e gestione degli impianti industriali;
- ING-IND/35, che tratta aspetti progettuali, economici, organizzativi e gestionali in campo ingegneristico;
- SECS-P/06, settore dell'economia applicata.

Di questi, ING-IND/17 e ING-IND/35 sono settori bibliometrici mentre SECS-P/06 è un settore non bibliometrico.
La scelta di analizzare due settori bibliometrici e uno non bibliometrico è motivata dal fatto che potrebbero emergere comportamenti citazionali differenti, considerato il fatto che gli SSD sottostanno a indicatori diversi.
In aggiunta, sono stati scelti i tre SSD sopra descritti in quanto:

- ING-IND/35 e SECS-P/06 sono due settori molto simili a livello di contenuti ma solo uno di essi è bibliometrico;

- ING-IND/17 è un settore che si trova nella stessa area disciplinare (area 9) di ING-IND/35 e si può considerare storicamente a esso vicino.

Una simile analisi che ha messo a confronto settori bibliometrici e non è stata presentata anche nell'articolo "Self-citations as strategic response to the use of metrics for career decisions" di M. Seeber, M. Cattaneo, M. Meoli, P.Malighetti [3].

L'elenco dei docenti strutturati, i.e. professori ordinari, associati e ricercatori (sia a tempo inteterminato che determinato), in ciascuno dei tre settori, per gli anni dal 2001 al 2021, è stato ricavato da dati pubblici forniti dal Ministero dell'Istruzione, Università e Ricerca, contenenti i campi descritti in Tabella 2.1.[1]

Come si può osservare dalla Tabella 2.1, al suo interno non è presente il codice identificativo di Scopus, fondamentale per poter ricavare le informazioni sulle pubblicazioni. Inizialmente si è deciso di aggiungere una colonna "id" contenente un indice progressivo per individuare univocamente l'autore. In seguito, è stata implementata una procedura che,

[1] É importante sottolineare che le informazioni sullo stesso docente nel corso degli anni possono variare (e.g. per un cambio di SSD o Università). Per l'analisi che seguirà nel Capitolo 4, l'SSD diventa nel tempo invariante, anche per chi è uscito dal sistema o ha effettuato un cambio di settore in quanto il numero di docenti per cui questo accade è da considerarsi basso.

Feature	**Descrizione**
Fascia	e.g. ordinario, ricercatore, associato non confermato
Nome	Cognome e nome del singolo
Genere	Maschio o Femmina
Ateneo	Ateneo di appartenenza
Facoltà	Facoltà dell'ateneo
SSD	Settore scientifico-disciplinare
SC	Settore concorsuale
Dipartimento	Dipartimento di afferenza all'interno della facoltà
d_statale	1 = statale, 0 = non statale
Anno	Anno di riferimento dei dati

Tabella 2.1: Descrizione delle variabili nei file del MIUR

grazie all'utilizzo di alcuni API (application programming interface) di Scopus descritti nelle pagine seguenti, permette di estrarre l'id dell'autore a partire dal suo nome e cognome. L'API utilizzato per estrarre il codice identificativo di ogni autore è Author Search. Questo rappresenta un'interfaccia che permette di effettuare ricerche nel database degli autori in Scopus, ricevendo in risposta un file JSON contenente dettagli sui profili corrispondenti alla *query* di ricerca, fatta sulla base del nome e del cognome.

Le informazioni contenute nel JSON che si è scelto di estrarre sono le seguenti:

- dc_identifier, identificatore Scopus;
- surname, cognome dell'autore;

- name, nome dell'autore;
- subject areas, insieme delle prime tre tematiche dei *paper* sulla base del numero di pubblicazioni fatte in ogni area;
- affiliation id, id dell'ateneo di appartenenza.

Coerentemente a quanto supposto, effettuare una ricerca sulla base delle generalità dell'autore ha fatto sorgere criticità per quanto riguarda:

- Nomi composti, poichè Scopus potrebbe aver memorizzato l'autore con solo una parte del nome, non ottenendo risultati in output;
- casi di omonimia o quasi omonimia per cui può essere restituito in output più di un singolo profilo.

Nomi Composti Se la richiesta all'API Author Search non restituisce alcun profilo, è probabile che si stia effettuando una *query* con un nome composto. In questo caso, la scelta è stata quella di effettuare una richiesta per il cognome e ogni parte del nome, come riportato nel seguente esempio.

Richiesta con nome composto Si supponga che la chiamata all'API per "Maria Sole Bianca Luisa Brioschi" restituisca un JSON vuoto, le richieste che verranno fatte in seguito sono:

1. Brioschi + Maria
2. Brioschi + Sole
3. Brioschi + Bianca
4. Brioschi + Luisa

In questo modo è possibile verificare se uno tra tutti i risultati in output sia effettivamente quello desiderato.

Casi di omonimia Scegliendo di effettuare richieste all'API sulla base delle generalità del singolo, in molti casi, l'output della richiesta non sarà composto da un solo autore ma da tutti i profili con quel nome e cognome o simili. Il risultato della chiamata all'API Author Search, infatti, include ogni possibile variazione del nome dell'autore inserito nella *query* di ricerca, in quanto non è detto che nelle pubblicazioni ci si riferisca all'autore sempre allo stesso modo (si pensi all'utilizzo del nome puntato). Per esempio, nel caso si volesse cercare l'autore John Smith, nome molto comune, i risultati includono anche nomi come David John Smith e John R. Smith.

Ciò implica la necessità di dover determinare quale, tra i tanti identificativi in output, sia quello corretto.

Inizialmente si era ipotizzato di utilizzare l'affiliazione dell'autore così da scegliere il record per cui questo campo coincideva con l'ateneo di appartenenza (memorizzato nei file del MIUR). Tuttavia, questo approccio ha riscontrato le seguenti problematiche:

- potrebbero esserci casi di omonimia all'interno dello stesso ateneo;
- si correrebbe il rischio di perdere quei profili che, seppur corretti, hanno cambiato affiliazione recentemente;
- Scopus non possiede nomi univoci per le affiliazioni e sono presenti anche diversi livelli di dettaglio (e.g. UNIVPM, Università Politecnica delle Marche, Politechnic University of Marche, Dipartimento di Ingegneria dell'Informazione, ...).

Per i motivi sopra elencati si è deciso di non effettuare una disambiguazione dei risultati sulla base delle affliazioni.

2.2 Procedura di disambiguazione dei risultati

La strada intrapresa è stata quella dell'assegnazione di un punteggio ad ogni record in output, sulla base del valore delle sue features. In questo

modo, più il punteggio associato all'individuo sarà alto, più è probabile che quel profilo sia quello corretto. In aggiunta, per aumentare la robustezza dalla procedura e la correttezza della scelta finale degli identificativi Scopus, si è optato per ripeterla per i tre SSD (settore scientifico-disciplinare) su 20 anni consecutivi (dal 2001 al 2021). Una volta ottenuti gli id con punteggio più alto per ogni autore in input, è stato possibile confrontare i risultati tra i diversi anni e assegnare ad ogni autore l'id che è risultato più volte essere quello corretto.

Il dataset dal quale si ottengono i nomi e cognomi per le richieste all'API Author Search è composto da alcuni campi dei file del Ministero con l'aggiunta di due colonne:

- anno, riguardante all'anno di riferimento del singolo record;
- is_duplicate, variabile dummy che permette di verificare se il record corrispondente è duplicato ed è quindi presente un altro individuo nello stesso SSD e nello stesso anno con ugual nome e cognome, casistica non è possibile disambiguare i risultati.

Di seguito si riportano alcuni record esemplificativi della tabella "dati_miur", a partire dalla quale sono state effettuate le richieste all'API di Scopus.

id	nome	cognome	fascia	genere	ateneo	facolta	SSD	sc	anno	is_duplicate
1294	Arrigo	Pareschi	Ordinario	M	BOLOGNA	Ingegneria	ING-IND/17	09/B2	2011	0
1295	Augusto	Bianchini	Ricercatore non confermato	M	BOLOGNA	Ingegneria II	ING-IND/17	09/B2	2011	0
1296	Marco	Gentilini	Associato confermato	M	BOLOGNA	Ingegneria	ING-IND/17	09/B2	2011	0
1297	Emilio	Ferrari	Ordinario	M	BOLOGNA	Ingegneria	ING-IND/17	09/B2	2011	0
1298	Cristina	Mora	Ricercatore	F	BOLOGNA	Ingegneria	ING-IND/17	09/B2	2011	0

Tabella 2.2: Esempio dei dati della tabella dati_miur

Per ogni record della tabella "dati_miur", dunque per ogni SSD e per ogni anno dal 2001 al 2021, si è proceduti con una chiamata all'API Author Search. L'insieme dei risultati delle richieste all'API sono stati memorizzati in una tabella chiamata "dati_scopus_per_score" che, per ogni record in risposta da Scopus, memorizza alcune informazioni della Tabella 2.2 (id_originale, SSD, anno) insieme a:

- *dc_identifier*, codice identificativo Scopus dell'autore;
- cognome e nome dell'autore;
- *subject_area*, feature che rappresenta le principali aree tematiche delle pubblicazioni fatte dall'autore (fino a un massimo di tre) insieme al corrispondente quantitativo;
- *affiliation_id*, identificativo Scopus dell'affiliazione dell'autore.

Nella Tabella 2.3 si riportano alcune righe esemplificative che rappresentano i risultati ottenuti dalla chiamata all'API per l'autrice Rita Gamberini (id originale 1950).

dc_identifier	surname	name	subject_area	affiliation_id	id_originale	SSD	anno
56230793900	Gamberini	Rita	ENGI:92,BUSI:69,ENVI:44	60004591	1950	ING-IND/17	2015
6603009210	Gamberini	Rita	MEDI:3		1950	ING-IND/17	2015
7003805356	Gamberini	Maria Rita	MEDI:86,BIOC:25,IMMU:4	127298005	1950	ING-IND/17	2015

Tabella 2.3: Esempio dei dati della tabella dati_scopus_per_score

Come è possibile notare, per la singola autrice, come anche per tanti altri, Scopus ha individuato più risultati, in questo caso tre, distinguibili da un diverso codice identificativo (*dc_identifier*). Di conseguenza, l'obiettivo principale è risultato essere l'assegnazione del codice identificatore Scopus corretto per ogni individuo in ciascun anno. Questo è stato effettuato automaticamente da una procedura che si basa su tre punteggi:

- score per il nome, che permette di valutare la somiglianza tra il nome originale e quelli delle risposte ottenute da Scopus;
- score per il cognome, analogo al precedente;
- score per le subject areas, secondo il quale, più una subject area è frequente all'interno delle risposte di Scopus per un dato SSD, maggiore sarà il suo score (interpretabile come frequenza relativa).

Nelle seguenti pagine si presenteranno le costruzioni degli score di cui sopra utilizzando come esempio i dati nella Tabella 2.3, quindi per il settore scientifico-disciplinare ING-IND/17, nell'anno 2015 e per l'autrice Rita Gamberini.

2.2.1 Score per nome e cognome

I due punteggi sono stati costruiti sulla base dell'Edit Distance. Quest'ultima, a partire da due stringhe (e.g. "xxxx" e "xxxy"), restituisce la distanza tra di esse in termini di numero di lettere da rimuovere e inserire in modo da rendere la prima stringa uguale alla seconda. Nel caso delle due stringe nell'esempio l'Edit Distance è pari a 2 poichè, nella seconda stringa, sarebbe necessario:

- rimuovere la "y";
- aggiungere la "x".

Nel caso specifico della creazione dello score da assegnare ai nomi e i cognomi dei risultati di Scopus, l'Edit Distance è stata utilizzata per determinare quanta differenza ci fosse tra il nome (o cognome) dell'autore originale e quelli restituiti dalla chiamata all'API Author Search. Questa distanza è stata normalizzata per la massima lunghezza delle stringhe prese in considerazione, così da poter confrontare i valori ottenuti per stringhe di lunghezza differente e avere un valore che può variare nell'intervallo chiuso e limitato $[0, 1]$. Gli estremi di questo intevallo rappresentano le seguenti situazioni:

- quando la distanza normalizzata è 1 vuol dire che le stringhe coincidono;
- quando assume valore 0 vuol dire che non ci sono alcune lettere in comune tra le due stringhe.

Sono stati calcolati due score, ed_nome ed ed_cognome, così come segue. Detto x il nome originale e y uno dei nomi dei risultati in output ricevuti dal database di Scopus, si ha:

$$ed\_nome = \frac{max(len(x), len(y)) - editDistance(x, y)}{max(len(x), len(y))}.$$

Per il calcolo dello score del cognome si è ragionato in modo analogo. L'algoritmo utilizzato per il calcolo dei punteggi da assegnare al nome e al cognome di ogni dc_identifier è riassunto nell'Algoritmo 1.

Algoritmo 1 Calcolo degli score di nome e cognome per l'SSD i e l'anno j

```
1: X_miur ← "dati_miur" where "SSD" = i and "anno" = j
2: for ogni autore in X_miur do
3:     n ← nome dell'autore
4:     c ← cognome dell'autore
5:     id ← id originale dell'autore
6:     T ← "dati_scopus_per_score" where "id_originale" = id
7:     for ogni dc_identifier in T do
8:         n_dc ← nome dell'dc_identifier
9:         c_dc ← cognome del dc_identifier
10:        n_max ← max(n, n_dc)
11:        c_max ← max(c, c_dc)
12:        ed_nome = (n_max − Edit Distanct(n,n_id)) / n_max        ▷ Score del nome
13:        ed_cognome = (c_max − Edit Distance(c,c_dc)) / c_max     ▷ Score del cognome
```

A scopo esemplificativo, in Tabella 2.4 si riportano i valori degli score per il nome e cognome degli autori Scopus nella Tabella 2.3.

dc_identifier	surname	name	cognome originale	nome originale	ed_cognome	ed_nome
56230793900	Gamberini	Rita	Gamberini	Rita	1,0	1,0
6603009210	Gamberini	Rita	Gamberini	Rita	1,0	1,0
7003805356	Gamberini	Maria Rita	Gamberini	Rita	1,0	0,4

Tabella 2.4: Esempio di valori di ed_nome e ed_cognome

Come è possibile osservare, i valori degli score del nome e cognome sono pari a 1 per gli id 56230793900 e 6603009210 mentre nel caso del nome dell'autrice 7003805356 (Maria Rita), si nota uno scostamento dal nome dell'autrice per la quale è stata effettuata la richiesta (Rita). Questa differenza, oltre ad essere facilmente visibile a livello visivo, si evidenzia

anche dallo score ed_nome che assume un valore pari a 0,4. Il calcolo effettuato è stato il seguente:

- i caratteri da rimuovere da "Maria Rita" per ottenere la stringa originale "Rita" sono 6, inclusi gli spazi;
- la stringa più lunga contiene 10 caratteri.

Di conseguenza, lo score per il nome è pari a $\frac{10-6}{10} = 0,4$.

2.2.2 Score per le subject areas

La costruzione degli score per le subject areas segue due step distinti. Inizialmente, per ogni SSD e anno considerato, viene interrogato Scopus tramite API per ottenere le risposte di tutti i nomi presenti nel settore corrispondente e si considerano tutte le risposte ricevuto. In seguito sono state considerate tutte le subject areas presenti tra le risposte di Scopus, calcolandone la frequenza relativa di ognuna. In questo modo, le percentuali di ciascuna area tematica rispetto al totale rapprestano il peso che viene associato alla subject.
Questo passaggio è visibile dalla Tabella 2.5 nella quale è possibile individuare:

- subject Area;

- occ, numero di volte in cui quella corrispondente subject area compare all'interno dei risultati di Scopus;
- tot_occ, numero totale delle occorrenze delle subject area;
- rate, frequenza relativa della presenza della subject area sul totale, calcolata come $\frac{occ}{tot\_occ} \cdot 100$.

Sulla base dei valori ottenuti si è optato per considerare come valide solamente una percentuale delle aree tematiche (e.g. il primo 70% tra tutte le subject areas in ordine decrescente per *rate*). Nel caso particolare della Tabella 2.5, mantenendo il primo 70% delle subject areas si terrebbero in considerazione le prime 19, assegnando 0 alle restanti.

In seguito, sono state ricalcolate le frequenze relative delle subject areas considerate rilevanti allo step precedente. Questo passaggio è stato effettuato in maniera analoga a quello del calcolo di *rate* nella Tabella 2.5, con la differenza sostanziale che il totale delle occorrenze è stato calcolato sommando quelle delle subject areas rilevanti. A partire dall'esempio 2.5, i pesi finali delle subject areas sono diversi da 0 solo per 19 materie, come visibile dalla Tabella 2.6.

A questo punto, ad ogni record ricevuto in output da Scopus è stato affiancato un peso cumulativo per tutte le subject areas ad esso associate.

sub	**occ**	**tot_occ**	**rate**
ENGI	207	1127	18,37 %
BUSI	148	1127	13,13 %
MEDI	142	1127	12,60%
COMP	86	1127	7,63%
BIOC	74	1127	6,57%
DECI	59	1127	5,24%
ENVI	50	1127	4,44%
SOCI	40	1127	3,55%
AGRI	38	1127	3,37%
ENER	35	1127	3,11%
MATH	32	1127	2,84%
PHYS	27	1127	2,40%
MATE	26	1127	2,31%
IMMU	20	1127	1,77%
CHEM	17	1127	1,51%
EART	17	1127	1,51%
NEUR	16	1127	1,42%
PHAR	14	1127	1,24%
NURS	13	1127	1,15%
DENT	11	1127	0,98%
MULT	11	1127	0,98%
ARTS	10	1127	0,89%
CENG	10	1127	0,89%
HEAL	9	1127	0,80%
ECON	7	1127	0,62%
PSYC	6	1127	0,53%
VETE	2	1127	0,18%

Tabella 2.5: Pesi iniziali delle subject areas: ING-IND/17, 2015

Questo valore si può pensare come una media dei punteggi delle subject, pesata per il numero di *paper* scritti per ogni subject, ottenendo, per il

sub	**occ**	**tot_occ**	**rate**
ENGI	207	1061	19,51%
BUSI	148	1061	13,95%
MEDI	142	1061	13,38%
COMP	86	1061	8,11%
BIOC	74	1061	6,97%
DECI	59	1061	5,56%
ENVI	50	1061	4,71%
SOCI	40	1061	3,77%
AGRI	38	1061	3,58%
ENER	35	1061	3,3 %
MATH	32	1061	3,02%
PHYS	27	1061	2,54%
MATE	26	1061	2,45%
IMMU	20	1061	1,89%
CHEM	17	1061	1,6 %
EART	17	1061	1,6 %
NEUR	16	1061	1,51%
PHAR	14	1061	1,32%
NURS	13	1061	1,23%
DENT	-	-	0
MULT	-	-	0
ARTS	-	-	0
CENG	-	-	0
HEAL	-	-	0
ECON	-	-	0
PSYC	-	-	0
VETE	-	-	0

Tabella 2.6: Pesi finali delle subject areas: ING-IND/17, 2015

record j-esimo, il seguente score:

$$peso\_sub_j = \frac{\sum_i n\_doc_{subject_i} \cdot weight_{subject_i}}{\sum_i n\_doc_{subject_i}} \cdot 10. \qquad (2.1)$$

Nella formula (2.1) si trovano:

- $n\_doc_{subject_i}$, che rappresenta il numero di documenti pubblicati dall'autore j nella subject area i-esima;
- $weight_{subject_i}$, il peso della subject area i-esima calcolato come frequenza relativa delle subject areas più "rilevanti";
- una moltiplicazione per 10, così da dare maggior rilevanza alle subject areas durante la procedura di disambiguazione.

In Tabella 2.7 si riportano i pesi associati alle subject areas calcolari con i valori nella Tabella 2.6 per tre risultati di Scopus come esempio. In

dc_identifier	**surname**	**name**	**subject areas**	**peso_sub**
56230793900	Gamberini	Rita	ENGI:92,BUSI:69,ENVI:44	1,446
6603009210	Gamberini	Rita	MEDI:3	1,338
7003805356	Gamberini	Maria Rita	MEDI:86,BIOC:25,IMMU:4	1,159

Tabella 2.7: Esempio di pesi cumulativi per dc_identifier

particolare, il valore di *peso_sub* per l'identificatore 56230793900 è stato ottenuto utilizzando il seguente calcolo:

$$\frac{0,1951 \cdot 92 + 0,1395 \cdot 69 + 0,0471 \cdot 44}{92 + 69 + 44} \cdot 10 = 1,446.$$

La procedura di assegnazione del valore di *peso_sub* ad ogni dc_identifier per ciascun anno e SSD è stata riassunta dall'Algortimo 2.

Algoritmo 2 Calcolo dello score associato alle subjects per l'SSD i e l'anno j

X ← "dati_scopus_per_score" **where** "SSD" = i **and** "anno" = j
n ← numero totale delle subjects (con ripetizioni)
Rel_subj ← percentuale di subjects da mantenere **in** X
for `ogni subjects in X` **do**
    n_sub ← numero delle volte in cui subjects è presente nei risultati
    score_1 = $\frac{n\_sub}{n}$
T_sub ← tabella con subjects e score_1
Ordinare T_sub per score_1 in modo decrescente
for `ogni subjects not in top Rel_sub % di T_sub` **do**
    score_1 = 0
n_relevant_subjects ← subjects con score_1 > 0
for `ogni subjects in T_sub` **do**
    if score_1>0 **then**
        score_2 = $\frac{n\_sub}{n\_relevant\_subjects}$
    else
        score_2 = 0
for `ogni id in X` **do**
    $n\_doc_{subject_i}$ ← numero documenti scritti per subject i-esima
    peso_sub= $\frac{\sum_i n\_doc_{subject_i} \cdot peso\_2_{subject_i}}{\sum_i n\_doc_{subject_i}} \cdot 10$

2.2.3 Score totale

Lo score totale è stato ottenuto dalla somma dei tre punteggi descritti nelle sotto-sezioni 2.2.1 e 2.2.2. Riferendosi sempre ai dati nella tabella 2.3, si è arrivati al calcolo dello score totale sommando i valori negli score presentati nelle tabelle 2.4 e 2.7, ricavando il seguente risultato.

dc_identifier	surname	name	peso_sub	ed_cognome	ed_nome	score totale
56230793900	Gamberini	Rita	1,0	1,0	1,446	3,446
6603009210	Gamberini	Rita	1,0	1,0	1,338	3,338
7003805356	Gamberini	Maria Rita	1,0	0,4	1,159	2,559

Tuttavia, per poter cominciare ad escludere alcuni risultati, lo score complessivo è stato calcolato solo per quei record il cui score del nome e del cognome è risultato essere maggiore di un valore soglia (da determinare) mentre ai restanti è stato assegnato 0.

Algoritmo 3 Calcolo dello score totale per l'SSD i e l'anno j

```
1: X_miur ← "dati_miur" where "SSD" = i and "anno" = j
2: soglia ← valore sotto il quale scartare i risultati
3: for ogni autore in X_miur do
4:     id ← id originale dell'autore
5:     T ← "dati_scopus_per_score" where "id_originale" = id
6:     for ogni id in T do
7:         if ed_nome≥ soglia and ed_cognome ≥ soglia then
8:             score_totale = ed_nome + ed_cognome + peso_sub
9:         else
10:            score_totale = 0
```

Inoltre, per quei record la cui chiamata all'API Author Search ha restituito un solo risultato, questo sarà considerato come un match esatto.

2.2.4 Normalizzazione con la doppia sigmoide

Per poter confrontare diversi score totali escludendo così, per ogni docente, quei dc_identifier che non sono effettivamente rilevanti, si è optato

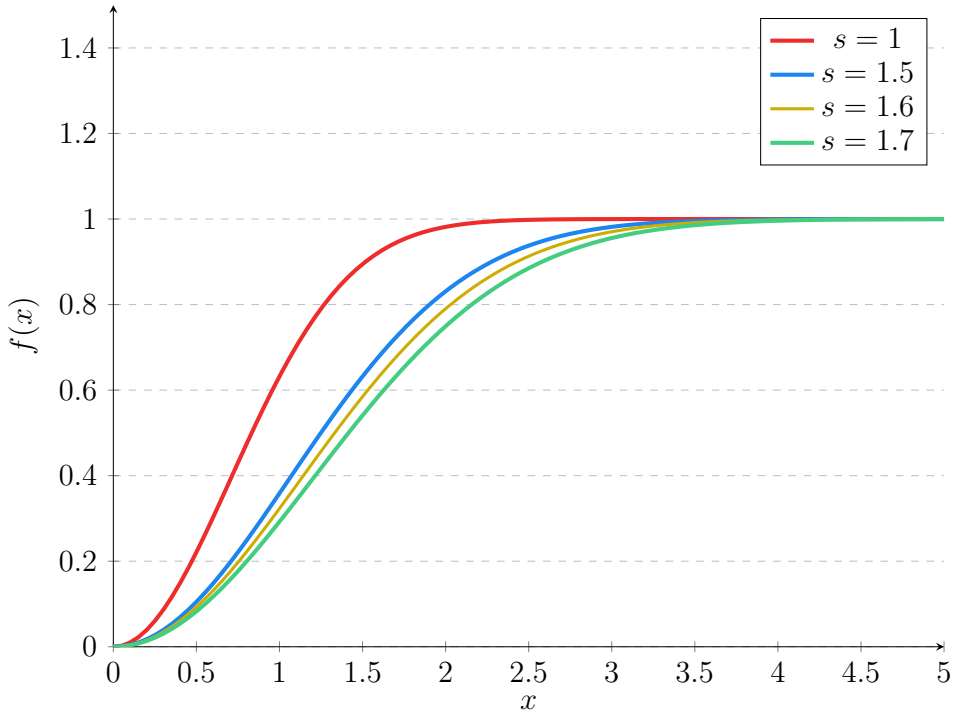
per normalizzare i valori così da limitarli ad un range che va da 0 a 1. La scelta è ricaduta nell'utilizzo di una doppia sigmoide con centro 0 e s (*steepness*) da determinare. In particolare, il valore normalizzato dello score totale per l'i-esimo dc_identifier sarà dato da:

$$ds_i = 1 - e^{-\left(\frac{st_i}{s}\right)^2}. \tag{2.2}$$

Nella formula (2.2) si trovano:

- ds_i, valore della doppia sigmoide per l'i-esimo dc_identifier;
- st_i, valore dello score totale per l'i-esimo dc_identifier;
- s, pendenza della funzione che può essere considerata come un parametro da far variare sulla base della performance della procedura.

La pendenza della funzione è un parametro fondamentale che permette di determinare la rapidità con la quale i valori crescono e, di conseguenza, quanto sia la distanza tra i valori trasformati a partire da quelli originali. Di seguito viene proposto un grafico contenente la funzione doppia sigmoidea al variare di alcuni valori del parametro *s*.



Grazie all'utilizzo della doppia sigmoide, è stato possibile effettuare un confronto tra gli score totali normalizzati e determinare una soglia sopra la quale considerare i dc_identifier ancora plausibili come risultati corretti.

Nel caso dell'esempio della Tabella 2.3, i valori dello score totale normalizzati con la doppia sigmoide avente $s = 1,6$ sono riportati in Tabella 2.8. Supponendo che la soglia sotto la quale i dc_identifier non si considerano più plausibili sia pari a 0.93, otterremo i seguenti risultati:

Dc_identifier	Surname	Name	Score totale	Score normalizzato	Decisione
56230793900	Gamberini	Rita	3,446	0,9903	Keep
6603009210	Gamberini	Rita	3,338	0,9871	Keep
7003805356	Gamberini	Maria Rita	2,559	0,9285	Discard

Tabella 2.8: Score totali per l'id originale 1950

Per cercare di ottenere un dc_identifier univoco per individuo, limitando quindi i risultati ambigui come nel caso della Tabella 2.8, si è proceduto con il confrontare le differenze tra i valori dello score totale non normalizzato, per tutti gli identificatori associati al singolo id originale. L'idea fondamentale dietro questo step sta nel fatto che pesi molto simili tra loro non permettono in assegnare con certezza un *dc_identifier* univoco ad un individuo. Dunque, se la differenza tra due score per lo stesso individuo risulta essere maggiore di un valore da determinare, allora il record con score più basso verrà escluso dall'analisi, altrimenti verranno mantenuti entrambi i *dc_identifier*. Per poter rappresentare al meglio questo passaggio si supponga che la differenza tra due valori debba essere maggiore di 0.1 per poterla considerare significativa, per i risultati nella Tabella 2.8 si è ragionato come segue.

Per ogni dc_identifier è stata calcolata la differenza tra il suo valore dello score totale e quello dei dc_identifier rimanenti, ottenendo l'output in Tabella 2.9.

dc_identifier	id confrontato	Differenza	Decisione sull'id confrontato
56230793900	6603009210	$0,108$	discard
6603009210	56230793900	$-0,108$	keep

Tabella 2.9: Differenza tra gli score totali

Osservando l'ultima colonna della Tabella 2.9 si nota che, nel 2015, l'unico dc_identifier plausibile per essere quello corretto da associare all'autrice Rita Gamberini (id_originale: 1950) è 56230793900, che corrisponde infatti al record con score totale più elevato (3,446).

2.2.5 Assegnazione di un id univoco a ciascun individuo

A questo punto della procedura, per ogni SSD e per ogni anno si avrà un dc_identifier assegnato o un risultato "ambiguo" qualora la procedura non sia stata in grado di individuare un solo id tra le diverse risposte di Scopus.

Per poter concludere la disambiguazione dei risultati e, dunque, ottenere un singolo id per ogni individuo nei dataset del MIUR si utilizzano gli output derivanti dalla procedura sopra descritta eseguita per ogni anno, dal 2001 al 2021. Ciò è stato fatto perché i docenti che compongono gli SSD non è detto che rimangano fissi ogni anno e di conseguenza variano i

pesi che le subject areas assumono, portando ad assegnazioni che possono variare annualmente.

Tramite un raggruppamento per SSD e, al suo interno, per nome e cognome, si sceglierà quella risposta (*dc_identifier* o ambiguo) la cui frequenza relativa negli anni risulta essere superiore a una soglia da determinare.

Si supponga che, ad esempio, le assegnazioni dei *dc_identifier* dal 2001 al 2021 per l'autrice Rita Gamberini siano i seguenti.

Nome	**Cognome**	**Anno**	**Dc_identifier assegnato**
Rita	Gamberini	2001	56230793900
Rita	Gamberini	2002	56230793900
Rita	Gamberini	2003	56230793900
Rita	Gamberini	2004	6603009210
Rita	Gamberini	2005	56230793900
Rita	Gamberini	2006	6603009210
Rita	Gamberini	2007	56230793900
Rita	Gamberini	2008	6603009210
Rita	Gamberini	2009	56230793900
Rita	Gamberini	2010	56230793900
Rita	Gamberini	2011	6603009210
Rita	Gamberini	2012	7003805356
Rita	Gamberini	2013	56230793900
Rita	Gamberini	2014	7003805356
Rita	Gamberini	2015	56230793900
Rita	Gamberini	2016	56230793900
Rita	Gamberini	2017	56230793900
Rita	Gamberini	2018	6603009210
Rita	Gamberini	2019	56230793900
Rita	Gamberini	2020	56230793900
Rita	Gamberini	2021	56230793900

Tabella 2.10: Assegnazione dei dc_identifier negli anni

A partire da questi risultati esemplificativi, si prosegue con il calcolo delle frequenze relative di ogni identificatore (numero di volte in cui compare l'id sul numero totale degli anni), ottenendo i risultati sotto riportati.

Se la soglia di accettazione del *dc_identifier* fosse pari al 50%, allora all'autrice Rita Gamberini verrà associato l'identificatore Scopus

Dc_identifier assegnato	**Frequenza relativa**
56230793900	66,7%
6603009210	23,8%
7003805356	9,5%

56230793900.

2.2.6 Stuttura del database per la procedura

Per poter memorizzare i dati originali del MIUR negli anni 2001 - 2021, i risultati ottenuti dalle richieste all'API Author Search, gli score associati ad ogni *dc_identifier* e le assegnazioni fatte nel corso degli anni dalla procedura, è stato scelto di creare un database relazione al fine di costruire un collegamento logico tra le tabelle utilizzate. Come DBMS è stato scelto MySQL.

Le tabelle presenti nel database sono:

- dati MIUR, i cui campi sono visibili dalla tabella 2.2;
- dati Scopus per *score*, già presentate nella tabella 2.3;
- affiliations Scopus, che contiene le informazioni sulle affiliazioni degli autori nella tabella precedente;
- subjects, contiene informazioni sulle aree tematiche trattate da ogni autore Scopus insieme con il numero di *paper* pubblicati per area;

- df scores finali, tabella che contiene gli score associati a ogni *dc_identifier* e informazioni sulla decisione di scelta o scarto del singolo;
- ambigui, contenente gli id_originali per cui i risultati non sono stati disambiguati dalla procedura e gli identificatori di Scopus per esso plausibili;
- match finali definitivi, tabella che memorizza, per ogni SSD e anno, il *dc_identifier* di Scopus o il risultato ambiguo per ogni autore originale della tabella dati MIUR.

Di seguito, in figura 2.1, è possibile visualizzare lo schema del database con chiavi primarie, esterne e vincoli di integrità referenziale.

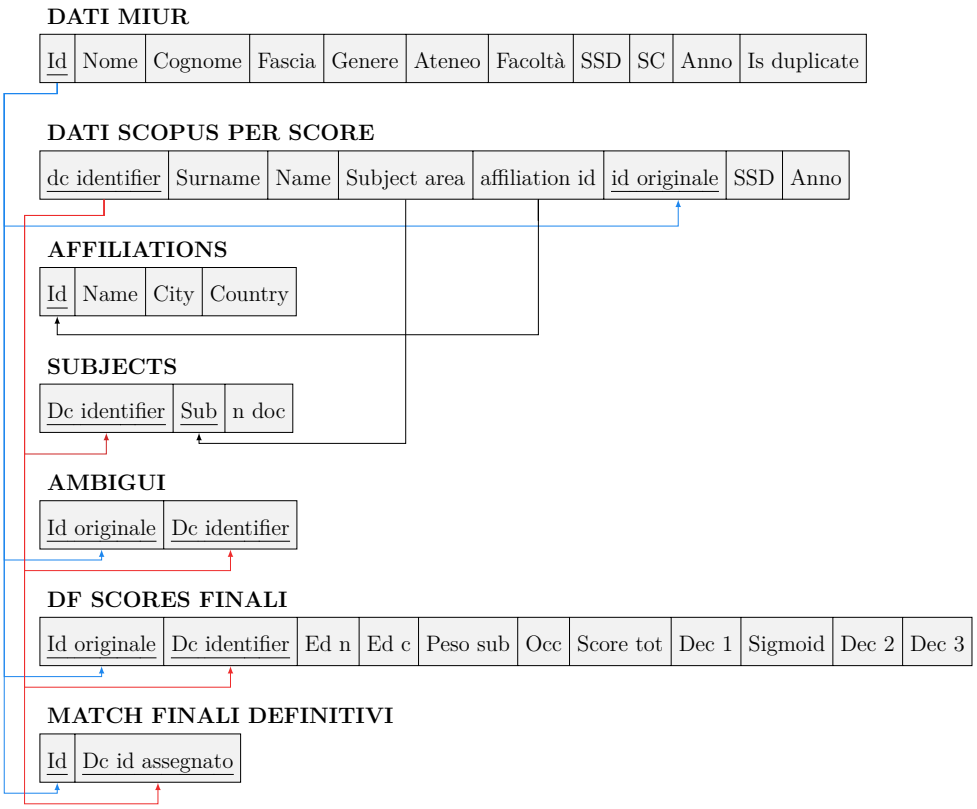


Figura 2.1: Schema del database "Assegnazione_identificativi_Scopus"

2.3 Tuning dei parametri

Al fine di ottenere una procedura di disambiguazione delle risposte che sia sufficientemente precisa, sono stati confrontati gli identificatori ottenuti per i tre SSD negli anni dal 2001 al 2021 con quelli corretti assegnati manualmente, così da poter calcolare il numero di errori commessi, numero di risultati ambigui e l'accuratezza.

Questa tecnica permette di individuare il valore ottimale dei parametri

da utilizzare nella disambiguazione di altri SSD. La valutazione è stata fatta sulla base di metriche di precisione quali:

- numero di assegnazioni corrette;
- numero di assegnazioni errate;
- numero di record per cui non è stato possibile ottenere un id univoco e per i quali la disambiguazione dovrà essere effettuata manualmente;
- *accuracy*, calcolata come numero di assegnazioni corrette su numero di assegnazioni totali effettuate.

I parametri da determinare sono 7:

1. RelSubj, percentuale sotto la quale si assegna alla *subject* un punteggio pari a 0;
2. EdNome, valore dell'*edit distance* per lo scostamento tra il nome effettivo e quelli dei risultati di Scopus
3. EdCognome, valore dell'*edit distance* per lo scostamento tra il cognome effettivo e quelli dei risultati di Scopus;
4. SteepnessSigmoid, valore di *steepness* della sigmoide;

5. Alpha, valore della doppia sigmoide sotto la quale il risultato di Scopus non viene considerato accettablie;

6. Beta, valore soglia corrispondente alla differenza tra gli score di diversi *dc_identifier* riferiti allo stesso autore. Sotto questo valore soglia i risultati non si considerano più accettabili;

7. Rate, percentuale riferita a quante volte avviene la singola assegnazione di un id, utilizzato nella parte finale della procedura.

2.3.1 Stima delle tempistiche

La scelta del *range* di valori sul quale far variare i parametri sopra descritti è stata presa sulla base dei tempi di esecuzione del codice (circa 20 minuti per singola combinazione di parametri).

Per non ottenere un costo computazionale eccessivamente alto, si è optato per far variare ogni parametro sulla base di una griglia di tre valori per 6 parametri e due valori per un parametro (rate). Di seguito si riportano i valori scelti per parametro.

Il totale delle combinazioni dei parametri, considerato il numero di valori che essi possono assumere, è pari a $3^6 \cdot 2$, ossia 1458. Il tempo massimo stimato per il tuning dei parametri, quindi, è circa di 20 giorni e 12 ore.

Parametro	**Valori assunti**
relSubj	90, 80, 70
edNome	0.5, 0.3, 0.1
edCognome	0.5, 0.3, 0.1
steepnessSigmoid	1.7, 1.6, 1.5
alpha	0.6, 0.5, 0.4
beta	0.2, 0.1, 0.05
rate	0.6, 0.5

Tabella 2.11: Griglie di valori dei parametri

L'algoritmo 4, descritto nella pagina successiva, rappresenta la procedura utilizzata per la disambiguazione dei risultati estratti da Scopus e il tuning dei parametri in essa presenti. Il codice è visibile in Appendice A.

Algoritmo 4 Tuning dei parametri nella tabella 2.11

Collegamento al database "Assegnazione_identificativi_Scopus".
for `ogni combinazione di parametri in tabella 2.11` **do**
    for `ogni SSD in lista dei tre settori analizzati` **do**
        for `ogni anno in range dal 2001 al 2021` **do**
            X ← tabella dati di Scopus e originali
            count ← numero riposte di Scopus per ogni autore
            Eseguire algoritmo 2 ▷ Pesi delle subject_areas
            Eseguire algoritmo 1 ▷ Score per il nome e cognome
            Eseguire algoritmo 3 ▷ Score totali
            Prima esclusione di alcuni risultati
            for `ogni record in X` **do**
                if Score totale = 0 **and** count > 1 **then**
                    Si esclude
                else if Score totale > 0 **and** count > 1 **then**
                    Si considera ancora valido
                else
                    Essendo l'unico risultato, si considera esatto
            Normalizzazione con la doppia sigmoide
            for `ogni record in X` **do**
                score_double_sigmoid $\leftarrow \left(1 - \exp^{-\frac{score\ totale}{steepness\ sigmoid}}\right)^2$
            for ogni dc_identifier **in** X **do**
                if score_double_sigmoid $< \alpha$ **then**
                    Si esclude
            for ogni id originale ancora ambiguo **in** X **do**
                for ogni dc_identifier in risposta id originale **as** i **do**
                    for ogni dc_identifier diverso da i **as** j **do**
                        D ← differenza tra lo score totale di i e j
                        if D è maggiore di β **then**
                            si esclude j dall'analisi
            Si assegnano le decisioni finali
            for ogni id originale **in** X **do**
                if count dei dc_identifier ancora plausibili > 1 **then**
                    decisione = ambiguo
                else if count dc_identifier plausibili = 1 **then**
                    decisione = dc_identifier rimasto
Decisione finale: risultato che si ottiene il il rate% delle volte dopo un **group by** per SSD e un conteggio delle assegnazioni.

Una volta concluso il tuning dei parametri è stato necessario individuare le migliori combinazioni di questi ultimi per ciascun settore. Ciò può essere fatto considerando le diverse misure di performance dell'algoritmo.

Si è optato per scegliere quelle combinazioni di parametri che massimizzano l'accuratezza della procedura in ciascun settore. Questa scelta è stata motivata dal fatto che è preferibile minimizzare gli errori commessi poichè implicherebbero la presenza di identificativi errati da utilizzare per estrarre dati che non sono effettivamente rilevanti. In aggiunta, per quanto i risultati ambigui corrispondano a un lavoro di individuazione dell'id Scopus da fare a mano, una volta trovato il match esatto, questi saranno dati sicuramente corretti e ciò porterebbe a un aumento dell'*cluster*.

I risultati ottimali per ciascun settore scientifico-disciplinare sono riportati in Tabella 2.12 [2].

[2]Quando non viene esplicitato il valore del parametro, eventualità indicata con il simbolo "-", questo implica che i diversi valori testati non comportano una variazione nella *accuracy* dell'algoritmo.

Configurazioni ottimali dei parametri			
	ING-IND/17	**ING-IND/35**	**SECS-P/06**
Accuracy	93,47%	95,42%	96,95%
RelSubj	90	-	-
EdNome	0,5	0,1	0,3/0,1
EdCognome	-	0,1	-
SteepnessSigmoid	-	-	-
Alpha	-	-	-
Beta	0,2	0,2	0,2
Rate	0,5	0,5	0,5
Esatti	186	271	286
Sbagliati	13	13	9
Ambigui	30	24	9

Tabella 2.12: Parametri ottimali della procedura di disambiguazione per ciascun settore

Dalla tabella 2.12 si può notare che i parametri che sembrano non influenzare la performance della procedura realizzata per i tre SSD sono la pendenza della doppia sigmoide e il valore di quest'ultima sotto il quale il risultato non si considera più accettabile (alpha). Infatti la percentuale di *accuracy* rimane molto elevata per qualsiasi valore (tra quelli testati) di "SteepnessSigmoid" e "Alpha".

Avendo a disposizione solamente i dati corretti di tre settori scientifico-disciplinari non si possiedono sufficienti informazioni per poter suddividerli in *cluster* e assegnare a ognuno di essi un *set* di parametri. Nonostante ciò, sembrerebbe che ING-IND/35 e SECS-P/06 possano usare lo

stesso settaggio di parametri e ottenere il livello più alto di accuratezza.

Capitolo 3

Estrazione dei dati relativi a pubblicazioni e references

In questo capitolo verranno descritti i passi effettuati per l'estrazione dei dati da Scopus e la costruzione del database da utilizzare per l'analisi che seguirà.

Una volta assegnato un singolo identificativo Scopus, tramite Scopus API, a ciascun docente degli SSD scelti per l'analisi, sono state effettuate richieste al database di Scopus per ottenere informazioni sulle pubblicazioni di ognuno di essi.

I dati che si è scelto di memorizzare riguardano:

- identificativo (ID) della pubblicazione

- titolo della pubblicazione
- data di pubblicazione
- autori
- info sulla tipologia di pubblicazione e sulla fonte in cui è stata pubblicata
- numero di citazioni per anno
- *papers* nelle references
- autori dei *paper* nelle references

La procedura di estrazione delle informazioni descritta nelle seguenti pagine permette di memorizzare i dati all'interno di un database relazionale, realizzato con MySQL, chiamato "dati_scopus".
Le API utilizzati sono state tre:

- Scopus Search API, interfaccia per effettuare ricerche nel database dei *paper* in Scopus. É stato utilizzato per ottenere i dati su tutti *paper* pubblicati da ogni autore e le citazioni negli anni degli stessi.
- Citation Overview API, interfaccia per effettuare richieste al fine di ottenere informazioni sulle citazioni dei *paper* di Scopus negli anni.

- Abstract Retrieval API, interfaccia per ricavare informazioni sulle references di ogni *paper* e relative agli autori di ciascuna reference.

3.1 Informazioni sui papers

La procedura di estrazione dei dati è partita con l'estrazione dei *paper* di ogni autore e delle sue citazioni negli anni. La tabella al centro del procedimento è "aut_ministeriale", in cui sono memorizzati informazioni sui docenti dei tre settori disciplinari analizzati (ING-IND/17, ING-IND/35, SECS-P/06), insieme con il codice identificativo di Scopus individuato dalla procedura descritta nel Capitolo 2. in Tabella 3.1 si riporta il record esemplificativo che riprende l'esempio utilizzato nel capitolo precedente e al quale si farà riferimento nel corso di questo capitolo.

Nome	**Cognome**	**Genere**	**Fascia**	**Ateneo**	**SSD**	**SC**	**au_identifier**
Rita	Gamberini	Ordinario	F	MODENA e REGGIO EMILIA	ING-IND/17		56230793900

Tabella 3.1: Record della tabella *aut miniseriale* a titolo esemplificativo

Inizialmente, per ogni *dc_identifier* della tabella "aut_ministeriale" è stata effettuata una richiesta all'API Scopus Search con vista (numero di campi in output) completa. La specificazione della vista, che in questo

caso presenta due opzioni (standard e completa), permette di ottenere anche informazioni sul numero totale di citazioni di ogni pubblicazione. L'output della chiamata all'API Scopus Search è un file JSON dal quale sono stati estratti alcuni campi e memorizzati nella tabella "papers". Gli attributi della tabella sono descritti in Tabella 3.2. L'ultima feature della

Feature	**Descrizione**
Id documento	Codice identificativo di Scopus del *paper*
Titolo	Titolo della pubblicazione, se presente
Data	Data di pubblicazione del *paper*
publicationName	Nome della rivista in cui è stato pubblicato il *paper*
scourceType	Tipologia di rivista (e.g. journal, conference)
sourceSubType	Sotto tipologia della pubblicazione (e.g. articolo, conference *paper*)
n_citation	Numero di citazioni complessive
is_reference	0 = non è una reference, 1 = è una reference

Tabella 3.2: Descrizione delle features nella tabella *papers*

Tabella 3.2 è una variabile dicotomica che permette di distinguere i *paper* degli autori nei settori analizzati dalle references di questi ultimi. Grazie ad essa è possibile determinare eventuali *paper* che fanno riferimento a pubblicazioni prodotte da un autore nello stesso settore, poichè il valore di *is_reference* di quest'ultimo sarà pari a 0. Nella Tabella 3.3 si riportano 3 delle 105 pubblicazioni dell'autrice Rita Gamberini individuate tramite richieste all'API Scopus Search.

In aggiunta, per ogni *paper* ottenuto dalla chiamata all'API Scopus Search, vengono memorizzati anche autori e co-autori in una tabella chiamata

id_documento	Titolo	Data	publicationName	sourceType	sourceSubType	n_citation	is_reference
84975511688	Hydrolysable PBS-based poly(ester urethane)s thermoplastic elastomers	2014-01-01	Polymer Degradation and Stability	Journal	Article	33	0
84872454317	An automated picking workstation for healthcare applications	2013-01-23	Computers and Industrial Engineering	Journal	Article	10	0
61849123224	A fuzzy multi-attribute model for risk evaluation in workplaces	2009-05-01	Safety Science	Journal	Article	106	0

Tabella 3.3: Esempio di pubblicazioni nella tabella *papers*

"aut_scopus". Quest'ultima contiene informazioni su tutti gli autori delle pubblicazioni, anche non appartenenti agli SSD analizzati. Le *features* presenti nella tabella riguardano solamente:

- identificatore dell'autore;
- cognome;
- nome;
- identificativo dell'affiliazione dell'autore.

Il numero totale di pubblicazioni dei docenti nei settori ING-IND/17, ING-IND/35 e SECS-P/06 ricavate dalle richieste Scopus sono 20.435. Per quanto riguarda gli autori delle pubblicazioni nella Tabella 3.3 questi sono stati inseriti nella Tabella "aut_scopus" e sono i seguenti. La *relationship* che lega un *paper* ai suoi autori è stata tradotta in un'ulteriore tabella chiamata "authorship" che contiene due campi:

dc_identifier	Cognome	Nome	Affiliazione
23010276000	Rimini	Bianca	60004591
24722923600	Mora	Cristina	60028218
35812213600	Piccinini	Paolo	60004591
55119667100	Gigli	Matteo	60028218
55828149363	Grassi	A.	60004591
56155547500	Fabbri	Martina	60004591
56230793900	Gamberini	Rita	60004591
6701460954	Lotti	Nadia	60028218
7003595956	Prati	Andrea	60015466
7003816201	Munari	Andrea	60028218
7004328488	Gazzano	Massimo	60021199
7006870483	Cucchiara	Rita	60004591

Tabella 3.4: Esempio di pubblicazioni nella tabella *papers*

- Id dell'autore;
- Id del documento pubblicato.

In questo modo, per ciascun *paper* saranno presenti tante coppie (id autore, id documento) quanti sono i suoi autori.

3.1.1 Time Series delle citazioni

Una volta memorizzate le informazioni sul singolo *paper* e sui suoi autori si è ritenuto utile estrarre il numero di citazioni della pubblicazioni suddiviso per anno. Infatti, il valore complessivo delle citazioni di una pubblicazione non permetterebbe di estrapolare *insights* così rilevanti ri-

spetto a quelli che possono essere tratti dallo stesso dato disaggregato lungo la dimensione temporale.

I dati sull'evoluzione nel tempo delle citazioni del singolo *paper* sono stati ottenuti da chiamate all'API Abstract Retrieval. Quest'ultimo permette di ricavare le cosiddette *citation matrix* delle pubblicazioni dalle quali è stata costruita la tabella "citations". Essa memorizza al suo interno il numero di citazioni di un *paper* per ogni anno a partire dalla sua pubblicazione, come nell'esempio in Tabella 3.5.

id_documento	**anno**	**n_citations**
85019120764	2019	9
85019120764	2020	21
85019120764	2021	17
85019120764	2022	8

Tabella 3.5: Descrizione delle features nella tabella *citations*

3.1.2 Algoritmo e meccanismo di gestione degli errori

La procedura di estrazione dei dati da Scopus tramite chiamate agli API è stata lanciata su una macchina virtuale in cui risiede il database "dati_scopus". L'algoritmo che riassume i passaggi descritti nelle precedenti pagine è descritto dall'Algortimo 5.

Algoritmo 5 Estrazione dei dati sulle pubblicazioni

```
1: for ogni autore in tabella aut miniseriale do Effettuare richiesta all'API Scopus Search
2:     for ogni paper in risposta alla richiesta do
3:         if paper in tabella papers then
4:             continue
5:         else
6:             Inserire paper in tabella papers
7:             for ogni autore in autori del paper do
8:                 if autore in tabella aut scopus then
9:                     continue
10:                else
11:                    Inserire dettagli autore in tabella aut scopus
12:            Effettuare richiesta all'API Citation Overview per paper
13:            Inserire dettagli citazioni in tabella citations
```

Inoltre, si è ritenuto fondamentale mettere in atto un meccanismo in grado di memorizzare fino a quale autore è avvenuto il download delle informazioni.

Inizialmente è stata aggiunta una colonna alla tabella "aut_ministeriale" contenente un numero progressivo per individuare univocamente gli autore.

In seguito è stata costruita una tabella ausiliaria che permette di memorizzare gli autori e l'id progressivo per i quali sono già stati ottenuti i dati. Ciò fornisce, in caso di spegnimento o riavvio della macchina, la possibilità di far ripartire la procedura dall'ultimo autore memorizzato nella tabella ausiliara, così da non effettuare richieste ridondanti alle

API.

3.2 Dati sulle references

Una volta ottenute le informazioni su tutti i *papers*, si è passati ad utilizzare il loro codice identificativo per poter ricavare l'elenco delle pubblicazioni inserite nelle references. L'API utilizzato in questo step dello scaricamento dei dati è stato l'Abstract Retrieval API che, con la view "ref", permette di richiedere tutte le references di un *paper* e la lista dei loro autori.

Per ogni pubblicazione dei docenti nei settori ING-IND/17, ING-IND/35 e SECS-P/06 i *papers* all'interno della loro bibliografia sono stati inseriti nella tabella "papers" e i loro autori nella tabella "aut_scopus", se non già presenti. La relazione che lega un *paper* alla sua bibliografia è stata rappresentata nella tabella "references" che contiene i seguenti campi:

- Id_documento, identificatore del documento scritto da uno degli autori nei tre settori presi in esame;
- Id_documento_citato, identificatore del documento nelle references.

Di conseguenza, per ogni *paper* saranno presenti tanti record quante sono le pubblicazioni inserite all'interno della bibliografia. La Tabella 3.6 contiene alcune delle references del *paper* 85019120764 ed è stata ottenuta tramite *join* tra la tabella "references" e "paper" ponendo come condizioni:

- Id_documento della tabella "references" uguale a 85019120764;
- il campo id_documento_citato di "references" uguale al campo id_documento di "papers".

id_documento	titolo	data	publicationName	sourceType	sourceSubType	n_citation	is_reference
77952561844	From strategy to business models and onto tactics	01/04/2010				1080	1
355662	Value creation in e-business	01/06/2001				2882	1
77952581340	Business models, business strategy and innovation	01/04/2010				3609	1
77952568109	Business models: A discovery driven approach	01/04/2010				596	1
77952564431	Business models as models	01/04/2010				685	1

Tabella 3.6: Esempio di references del paper 85019120764

In Tabella 3.6, i campi "publication name", "source type" e "source sub type" sono vuoti perchè si tratta di informazioni sulle pubblicazioni non presenti nel file JSON in risposta alla chiamata all'API Abstract Retrieval.

Nel caso del download delle references la procedura seguita è descritta

Algoritmo 6 Estrazione dei dati sulle references

```
1: for ogni pubblicazione in tabella papers do Effettuare richiesta all'API Abstract Retrieval con view = ref
2:     for ogni reference in risposta alla richiesta do
3:         Inserire (id pubblicazione, id reference) in tabella references
4:         if references in tabella papers then
5:             continue
6:         else
7:             Inserire reference in tabella papers con is_reference = 1
8:             for ogni autore in autori della reference do
9:                 if autore in tabella aut scopus then
10:                    continue
11:                else
12:                    Inserire dettagli autore in tabella aut scopus
```

dall'Algoritmo 6 e il codice si trova in Appendice B.

Anche per effettuare il download dei dati delle references è stato implementato in meccanismo di gestione degli errori. Ciò è stato fatto utilizzando un'identificativo progressivo (da 1 a 20435, numero di *paper* ottenuti) associato alle pubblicazioni degli autori nei tre settori in analisi, ottenute come descritto nella sezione 3.1. Ogni volta che vengono estratte le references di un *paper*, il suo identificativo progressivo viene inserito in una tabella.

In questo modo, in caso di errore, è possibile rilanciare la procedura di scaricamento dei dati a partire dall'ultimo *paper* preso in esame.

3.3 Struttura del database

Tutti i dati menzionati precedentemente sono stati memorizzati in differenti tabelle collegate tra loro a formare un database. Questo database, chiamato "dati scopus", contiene le seguenti tabelle:

- Autori ministeriale, conentente le informazioni degli docenti nei settori in analisi, come nella Tabella 2.1;
- Autori Scopus, contenente le informazioni sugli tutti autori delle pubblicazioni e delle references;
- *papers*, contenente i dati descritti nella Tabella 3.2;
- Authorship, che rappresenta la relationship esistente tra autore e *paper*;
- Citations, che contiene le citazioni dei *papers* negli anni;
- References, che rappresenta la relationship esistente tra un *paper* e le sue references.

La figura 3.1 rappresenta lo schema logico del database "dati scopus".

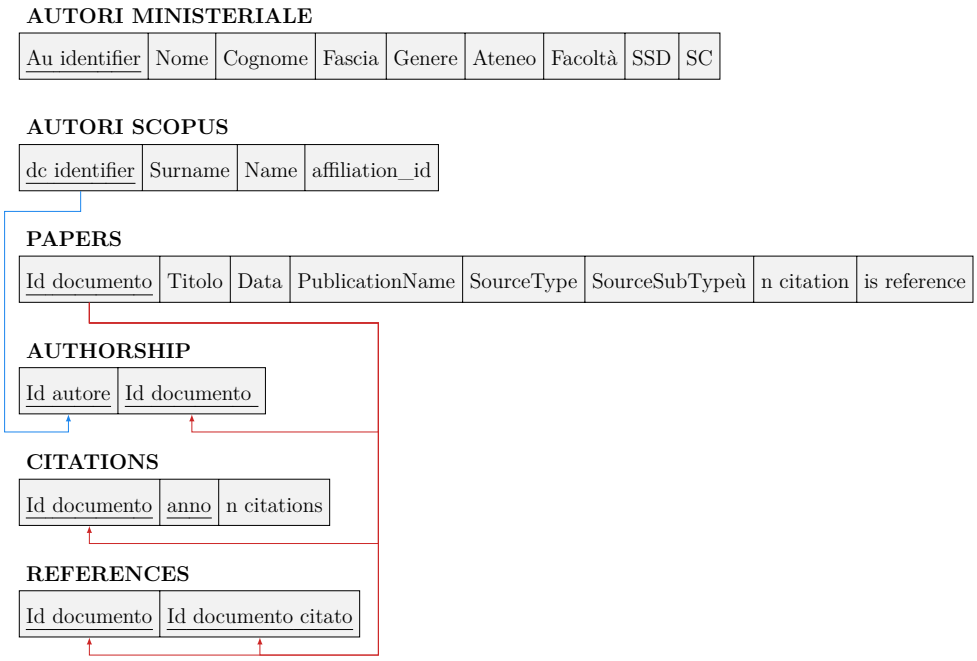


Figura 3.1: Schema del database “Dati Scopus”

Capitolo 4

Analisi empirica

L'analisi svolta nel presente capitolo si pone l'obiettivo di esplorare i dati estratti da Scopus con un focus particolare sull'analisi temporale delle tendenze citazionali delle pubblicazioni, settori scientifico-disciplinari e singoli autori. In particolare si affronteranno i seguenti temi:

- numero di pubblicazioni per docente fatte negli anni in ogni SSD;
- numero di references delle pubblicazioni e andamento di questo numero negli anni in ogni SSD;
- auto-citazioni delle pubblicazioni, degli autori e dei settori scientifico-disciplinari.

In seguito si presenterà una visualizzazione a grafo dei docenti che sono tra loro collegati da archi orientati che rappresentano la citazione nel quale, quindi, due docenti saranno legati tra loro solamente se uno di essi ha citato l'altro in una delle sue pubblicazioni.

4.0.1 Numerosità degli SSD e pubblicazioni per autore

Si è proceduti inizialmente con una descrizione della composizione dei singoli SSD, in particolare della numerosità. Ciò è stato fatto per poter avere un'idea di come il settore si è espanso nel tempo e poter quindi rapportare dati come "pubblicazioni effettuate annualmente" al numero di docenti nel settore, ottenendo una quantità confrontabile tra diversi anni.

I dati utilizzati per la costruzione del grafico che seguirà sono quelli della tabella "dati_miur" contenuta nel database "Scopus". Le informazioni utilizzate tra quelle disponibili sono state:

- identificatore univoco del docente;
- settore di appartenenza;
- anno di riferimento.

Per ottenere il numero di docenti di ogni SSD in ogni anno è stato effettuato un raggruppamento per "Anno di appartenenza", utilizzando un *count* degli identificatori come funzione di aggregazione ottenendo una tabella strutturata come quella che segue:

SSD	**Anno**	**Numero docenti**
ING-IND/17	2018	169
ING-IND/35	2018	235
SECS-P/06	2018	173
ING-IND/17	2019	176
ING-IND/35	2019	236
SECS-P/06	2019	181
ING-IND/17	2020	181
ING-IND/35	2020	249
SECS-P/06	2020	200

Tabella 4.1: Numero di individui negli SSD dal 2018 al 2020

Il grafico realizzato a partire dalle informazioni è presentato in Figura 4.1.

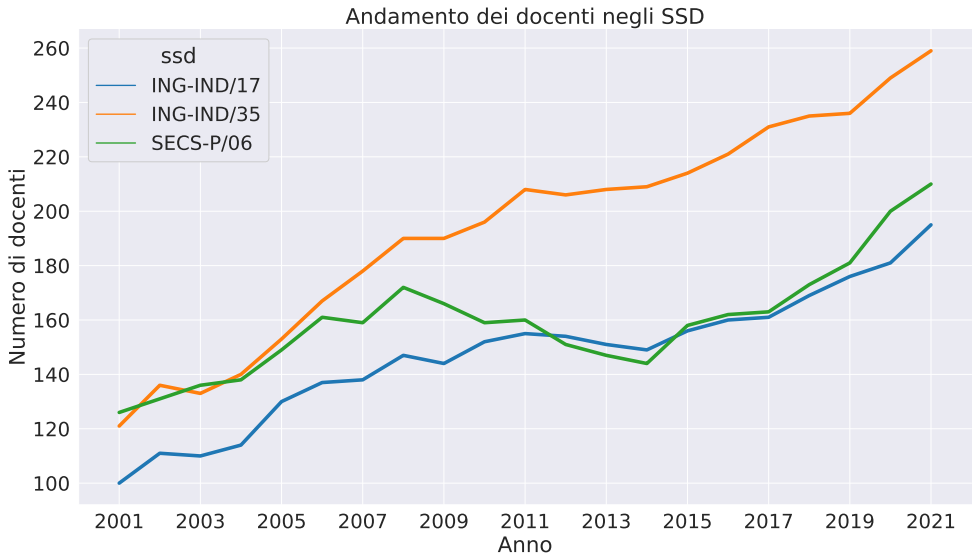


Figura 4.1: Andamento della dimensione dei settori nel tempo

In Figura 4.1 è possibile notare che il settore disciplinare che ha avuto una crescita più rapida in termini di numerosità è ING-IND/35. I due settori restanti invece, nonostante vi sia stato un divario tra le loro curve fino al 2009, si sono riallineati e tendono ad essere più simili a livello di numero di docenti che li compongono.

Le successive analisi si concentreranno sulle tipologie di pubblicazioni fatte da ogni SSD nel tempo, sul numero di pubblicazioni fatte dai docenti (rapportato alla numerosità del settore) e sulle citazioni fatte dagli autori nelle pubblicazioni.

I dati utilizzati, estratti dal database a seguito di una *query*, contengono le seguenti informazioni:

- Id della pubblicazione;
- tipologia (articolo, *conference paper*, ...);
- id dell'autore;
- SSD dell'autore – se presente;
- anno di pubblicazione,
- id del documento citato,
- id dell'autore citato,
- SSD dell'autore citato – se presente.

In questo modo è stato possibile ottenere i collegamenti tra ciascun autore della pubblicazione e ogni autore del *paper* citato. In particolare, i dati ottenuti sono strutturati come in Tabella 4.2.

Id documento	Tipologia	Id autore	SSD	Anno	Id documento citato	Id autore citato	SSD citato
84920784150	Article	15768608600	ING-IND/35	2015	84885192991	15726654000	ING-IND/17
84920784150	Article	15768608600	ING-IND/35	2015	84885192991	57192050380	ING-IND/17
84920784150	Article	15768608600	ING-IND/35	2015	84885192991	7101680357	ING-IND/17
84920784150	Article	15768608600	ING-IND/35	2015	84885192991	15768608600	ING-IND/35
84920784150	Article	7101680357	ING-IND/17	2015	84885192991	15726654000	ING-IND/17
84920784150	Article	7101680357	ING-IND/17	2015	84885192991	57192050380	ING-IND/17
84920784150	Article	7101680357	ING-IND/17	2015	84885192991	7101680357	ING-IND/17
84920784150	Article	7101680357	ING-IND/17	2015	84885192991	15768608600	ING-IND/35

Tabella 4.2: Tabella centrale dell'analisi con collegamenti tra autori e pubblicazioni che si citano

Inizialmente si è deciso di mostrare come le tipologie delle pubblicazioni

si siano evolute nel tempo in ciascun settore e ciò è stato fatto tramite un conteggio per anno dei *paper* per ogni SSD e tipologia. Il risultato ottenuto è stato riassunto nelle Figure 4.2, 4.3, 4.4.

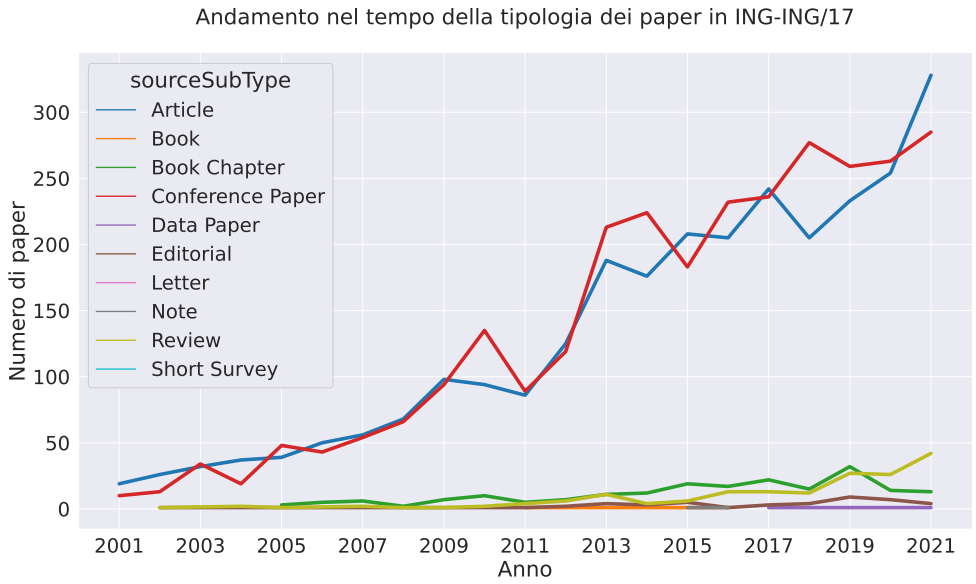


Figura 4.2: Numero di pubblicazioni per tipologia nel tempo in ING-IND/17

Dalle Figure 4.2, 4.3 e 4.4 si nota subito una suddivisione dei settori in due gruppi. Infatti, per quanto gli articoli siano la tipologia di pubblicazione più diffusa in tutti e tre i settore, solamente ING-IND/17 ha pubblicato una quantità significativa e sempre crescente di *conference papers*.

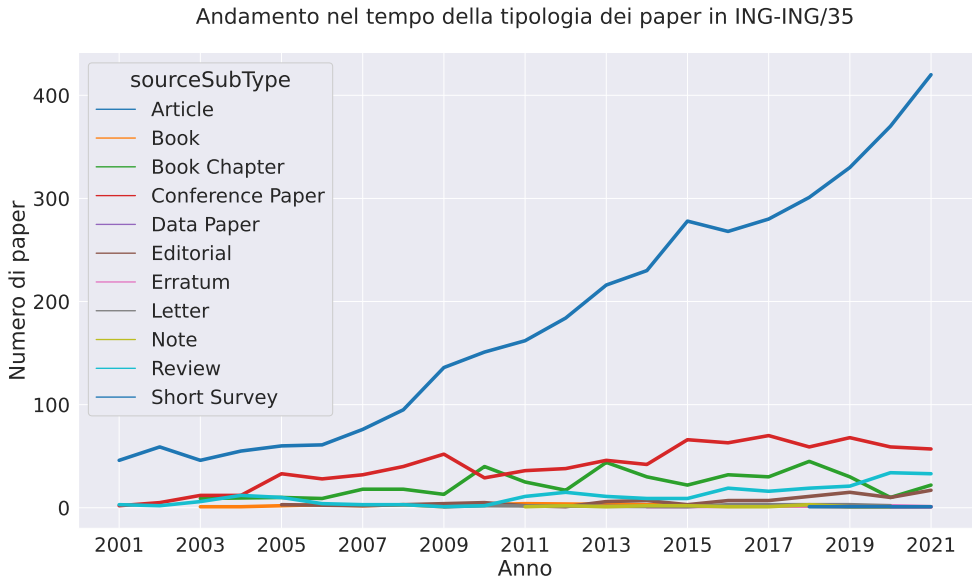


Figura 4.3: Numero di pubblicazioni per tipologia nel tempo in ING-IND/35

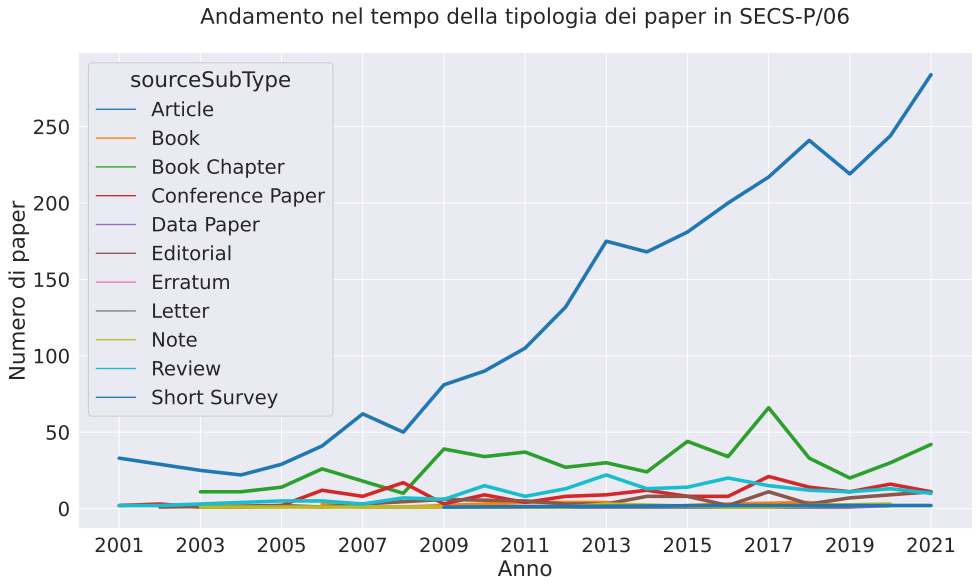


Figura 4.4: Numero di pubblicazioni per tipologia nel tempo in SECS-P/06

É da sottolineare il fatto che, a seconda della tipologia della pubblicazione, varia solitamente anche il numero di references di quest'ultima e, di norma, un articolo possiede molte più references di un *conference paper*.

In seguito il focus si è spostato sul calcolo del numero di pubblicazioni effettuate nel corso del tempo da ciascun settore scientifico-disciplinare. In particolare, il numero assoluto di pubblicazioni annuali è stato rapportato alla numerosità del settore in quel particolare anno, così da ottenere il numero di pubblicazioni per autore.

I dati utilizzati sono stati ottenuti a seguito di un'aggregazione per "SSD" e "Anno", utilizzando come funzione di aggregazione un *distinct count* degli id dei *paper* pubblicati. Alla tabella ottenuta è stata affiancata la colonna "paper per autore", calcolata come segue: il numero di *paper* scritti per autore del settore scientifico disciplinare i nell'anno t è dato da

$$paper\,per\,docente_{i,t} = \frac{numero\,di\,paper_{i,t}}{numero\,di\,docenti_{i,t}}.$$

Alcune righe esemplificative del risultato ottenuto sono riportate in Tabella 4.3. Il risultato completo, per tutti gli anni e i settori, è stato riassunto con il grafico in Figura 4.5.

Anno	SSD	Numero di paper	Paper per docente
2001	ING-IND/17	29	0.29
2001	ING-IND/35	51	0.42
2001	SECS-P/06	37	0.29
2002	ING-IND/17	41	0.37
2002	ING-IND/35	66	0.49
2002	SECS-P/06	35	0.27
2003	ING-IND/17	67	0.61
2003	ING-IND/35	74	0.56
2003	SECS-P/06	41	0.30

Tabella 4.3: Numero di pubblicazioni per docente negli SSD per gli anni 2001-2003

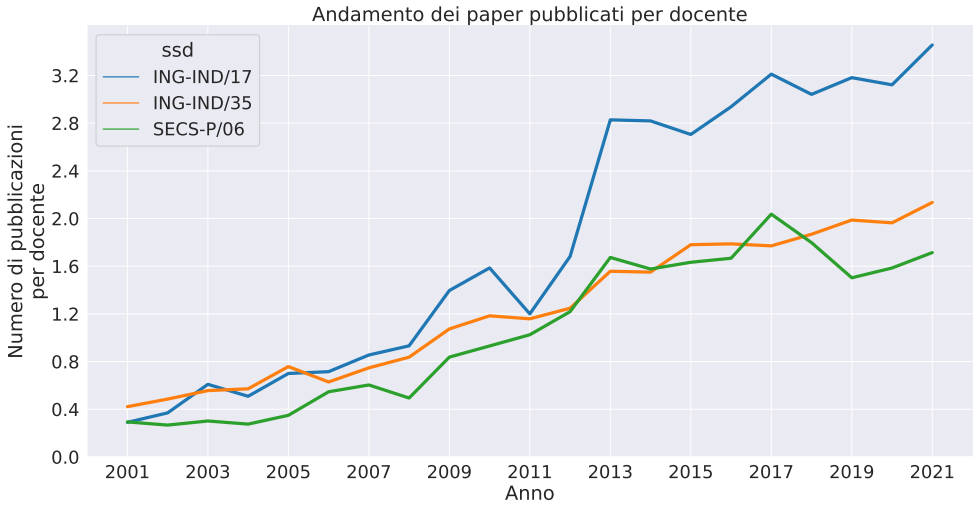


Figura 4.5: Andamento delle pubblicazioni per autore

A partire dalla Figura 4.5 possono essere fatte diverse considerazioni. In primo luogo si può notare che il numero di *paper* pubblicati per docente ha un andamento simile in tutti i settori fino al 2011.

Negli anni seguenti il settore che si discosta maggiormente per più pub-

blicazioni per docente è ING-IND/17, che subisce una rapida crescita in pochi anni.

In questo caso i settori che tendono ad essere più simili sono SECS-P/06 e ING-IND/35, per quanto ciò non avvenga in termini di numero di autori che vi appartengono, come mostrato in Figura 4.1.

Il settore che vede più pubblicazioni per docente e che ha più che raddoppiato questo numero in due anni (da 1.2 *paper* per autore nel 2011 a 2.8 *paper* per autore nel 2013) è ING-IND/17, pur essendo il settore meno numeroso dei tre. Dalla Tabella 4.4, contente i valori che generano il rapporto graficato in Figura 4.5 tra gli anni 2011 e 2013, è possibile notare che il numero di docenti presenti in ING-IND/17 resta pressochè costante e ciò che fa crescere il rapporto è il numero delle pubblicazioni.

Anno	**SSD**	**Numero di paper**	**Numero di autori**
2011	ING-IND/17	186	155
2012	ING-IND/17	259	154
2013	ING-IND/17	427	151

Tabella 4.4: Numero di paper e autori di ING-IND/17 per gli anni 2011-2013

Il numero di *paper* infatti è in forte aumento sia nel 2012 sia nel 2013, anno durante il quale diventa 1.64 volte maggiore dell'anno precedente. Una crescita così notevole delle pubblicazioni fatte potrebbe essere do-

vuta in primis a un aumento di produttività (non necessariamente di qualità), come anche all'aumento della collaborazione sia intra-settoriale (tra docenti dello stesso SSD), sia extra-settoriale (come collaborazioni con aziende, ricercatori di altri SSD o ricercatori non strutturati).

Per verificare a quale delle due casistiche potrebbe adattarsi maggiormente alla situazione presentata in Figura 4.5, si è partiti con l'analisi del numero di autori per pubblicazione.

Il grafico in Figura 4.6 rappresenta l'andamento del numero medio di autori per *paper*.

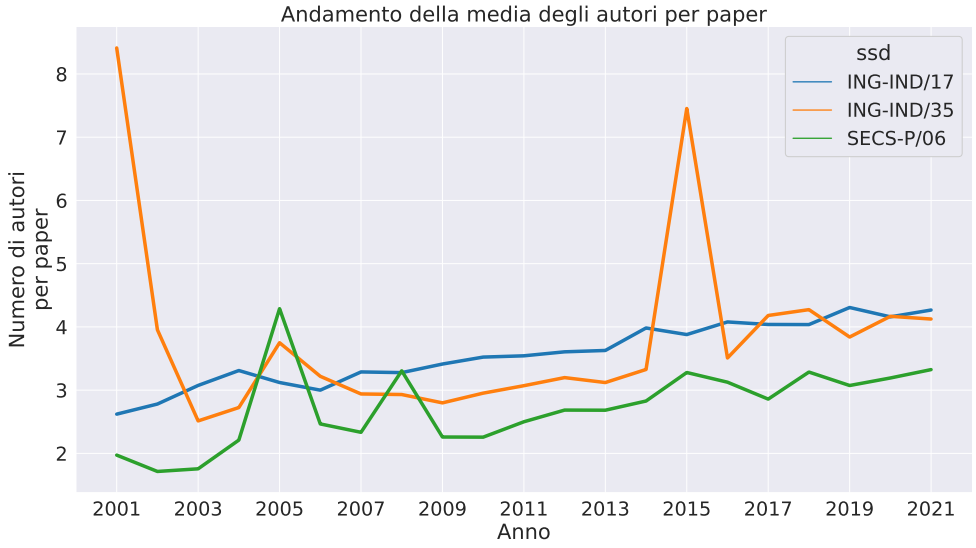


Figura 4.6: Andamento della media del numero di autori per pubblicazione

Il primo fenomeno che si nota osservando la Figura 4.6 è il picco della curva di ING-IND/35 in corrispondenza del 2015. Ciò è dovuto al fatto

che in questo SSD, nel 2015 15 pubblicazioni su 381 sono state scritte da oltre 100 autori, istanze che possono essere sicuramente considerate anomale.

In aggiunta, focalizzando l'attenzione sulla curva di ING-IND/17 si può notare come questa è sempre crescente, ad indicare una collaborazione maggiore nella stesura delle pubblicazioni nel corso del tempo.

Rilevante è anche l'aumento della mediana del numero di autori per pubblicazione (non riportata in figura), oltre che della media, riscontrato sempre in ING-IND/17 a cavallo tra il 2012 e il 2013, che passa da 3 a 4.

Per poter verificare se l'aumento degli autori per *paper* sia la conseguenza di una maggiore collaborazione tra docenti, è stato utilizzato il grafico del numero medio di docenti negli anni, presentato in Figura 4.7.

Dalla Figura 4.7 si può notare che, mentre in ING-IND/35 e SECS-P/06 il numero medio di autori dello stesso SSD per *paper* è molto variabile in tutti e 20 gli anni, la curva di ING-IND/17 subisce una decrescita a partire dal 2011. Questo va a sottolineare ancor di più il fatto che l'aumento della produttività del settore non è indice di una maggior collaborazione interna ma di produzioni con autori esterni (e.g. aziende).

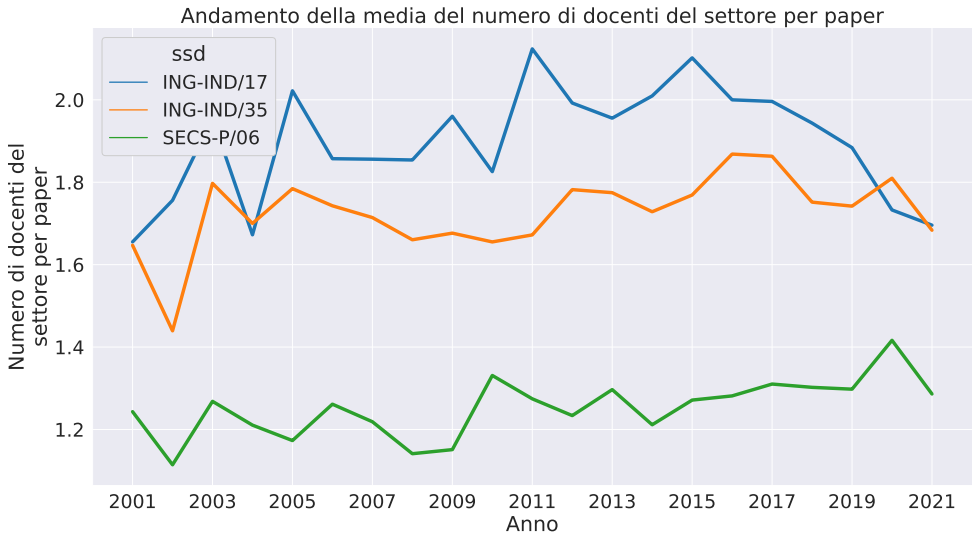


Figura 4.7: Andamento della media del numero di docenti del settore per paper

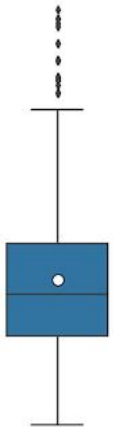
4.1 Analisi di references e autocitazioni delle pubblicazioni

4.1.1 Le references

In seguito il focus si è spostato sulle references delle pubblicazioni, con lo scopo di individuare eventuali cambiamenti nei pattern citazionali dei settori e, di conseguenza, degli autori nel corso degli anni.

Al fine di avere un'idea della distribuzione delle citazioni delle pubblicazioni negli anni in ciascun settore sono stati realizzati del box-plot (per ogni anno e SSD) che contengono le seguenti informazioni:

- Il primo segmento dal basso della scatola centrale colorata rappresenta il primo quartile (25%), anche detto *lower quartile* o Q1
- Il segmento centrale è la mediana (50%) o Q2
- Quello più in alto corrisponde al 75-esimo percentile, anche detto *upper quartile* o Q3



- I baffi invece sono individuati tramite una funzione del primo e terzo quartile, in particolare:
  - Quello più in basso è calcolato come "$Q1 - 1.5 \cdot IQR$", dove IQR rappresenta il range interquartile (75% - 25%)
  - Quello più in alto è calcolato come "$Q3 + 1.5 \cdot IQR$"
- I restanti punti individuano gli "outliers", quelle osservazioni che ricadono al di fuori dell'intervallo chiuso e limitato $[Q1 - 1.5 \cdot IQR, Q3 + 1.5 \cdot IQR]$.
- Il cerchio bianco rappresenta la media

Per la realizzazione del box-plot è stato inizialmente necessario calcolare il numero di references di ogni *paper*. Questo è stato fatto tramite un raggruppamento della Tabella 4.2 per "id documento", "SSD" e "Anno",

così da ottenere le citazioni di ogni pubblicazione (che si ripetono per gli SSD degli autori). La funzione di aggregazione utilizzata è un *distinct count* degli id dei documenti citati. La Tabella 4.5 funge da esempio del risultato ottenuto.

id_documento	**SSD**	**Anno**	**Citazioni totali**
85030773727	SECS-P/06	2021	55
85044972050	ING-IND/35	2021	69
85050115648	SECS-P/06	2021	59
85054898907	ING-IND/17	2021	67
85058140742	ING-IND/17	2021	50
85062280908	SECS-P/06	2021	66

Tabella 4.5: Esempio: calcolo numero di citazioni dei paper

La distribuzione dei dati nel tempo suddivisi per SSD è riassunta dai box-plot riportati nelle Figure 4.8, 4.9, 4.10[1].

Dai grafici è possibile osservare che per tutti e tre i settori la distribuzione è particolarmente variabile fino al 2011, anno successivo al quale si nota una leggera crescita della media delle references. Permane la similitudine tra i settori SECS-P/06 e ING-IND/35 che hanno entrambi una media che cresce meno rapidamente nel tempo rispetto a ING-IND/17, pur avendo valori tendenzialmente più elevati.

[1]Per l'asse y dei box-plot si è scelto l'utilizzo di una scala logaritmica poichè, pur essendo presenti outliers con valori molto elevati, permette di avere una visualizzazione più compresibile dei dati.

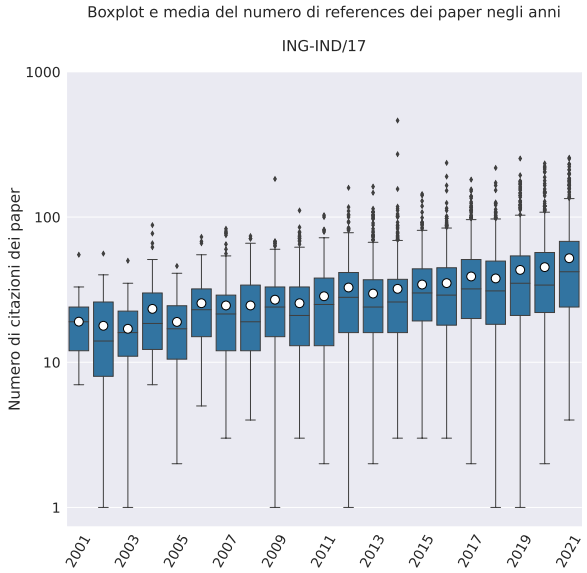


Figura 4.8: Distribuzione delle references dei paper negli anni e per ING-IND/17

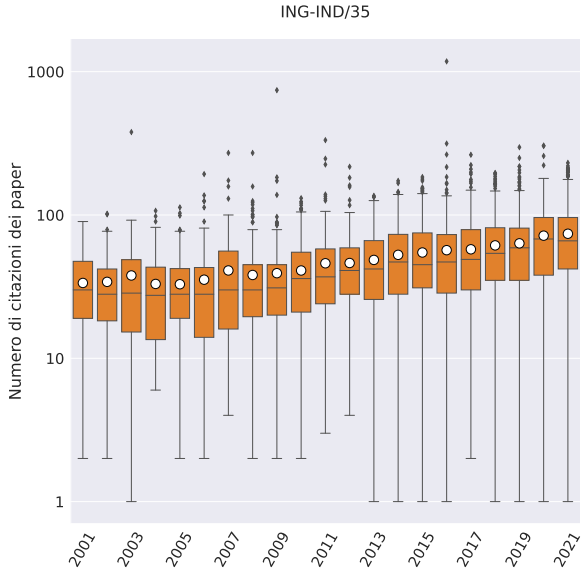


Figura 4.9: Distribuzione delle references dei paper negli anni e per ING-IND/35

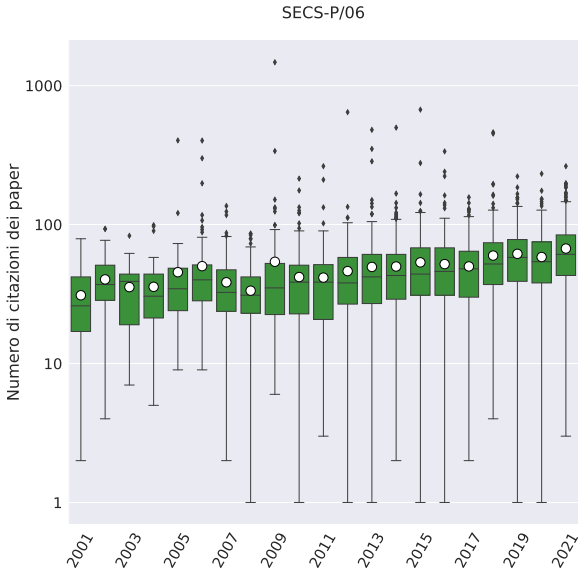


Figura 4.10: Distribuzione delle references dei paper negli anni e per SECS-P/06

In aggiunta, questi due settori, oltre a presentare un maggior numero di outliers in termini di citazioni, contengono osservazioni anomale particolarmente rilevanti, tra cui *paper* con oltre 1000 references.

Al fine di visualizzare numericamente come è variato il numero di references delle pubblicazioni nel tempo, nelle Figure 4.11, 4.12 e 4.13 si presentano delle *frequency tables* (in termini percentuali) del numero di references suddivise per settore in 3 anni differenti: 2010, 2015, 2020. La scelta di questi tre anni è motivata dal fatto che gli indici bibliometrici di valutazione della ricerca per alcuni settori sono stati introdotti a fine 2010; in questo modo è possibile verificare se la loro introduzione abbia

portato a un cambiamento nelle abitudini citazionali degli autori.

Per facilitare la lettura della tabella, il numero delle citazioni è stato suddiviso in 20 intervalli equispaziati. Saranno quindi presenti informazioni da leggere come segue: "nell'anno 2010 (Frigura 4.11), il 97,17% delle pubblicazioni del settore ING-IND/17 possiede un numero di references che ricade nell'intervallo [1,72)".

Figura 4.11: Frequency table relativa delle references dei *paper* del 2010

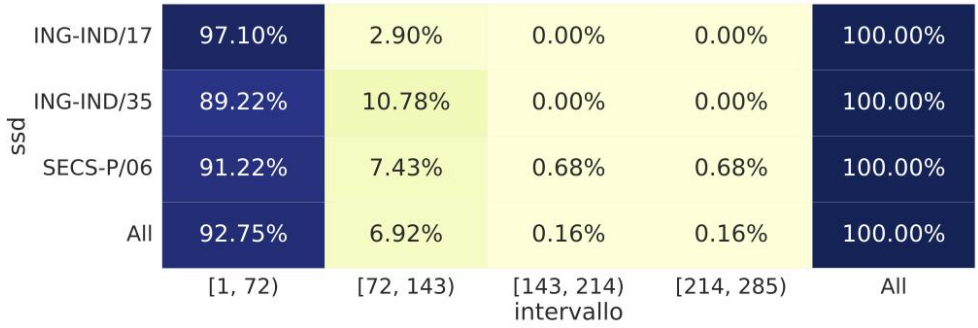


Figura 4.12: Frequency table relativa delle references dei paper del 2015

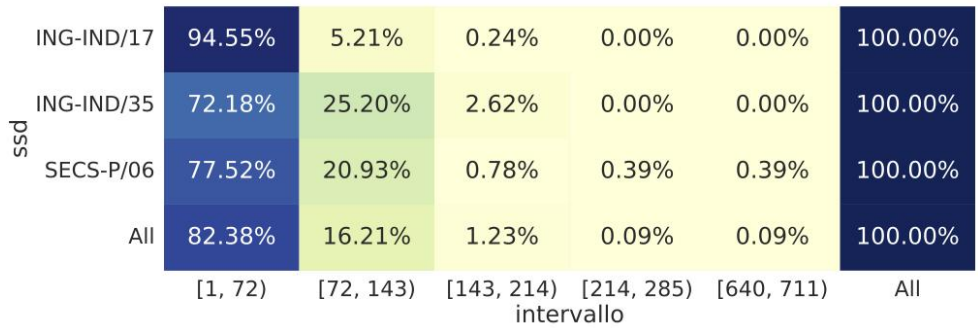
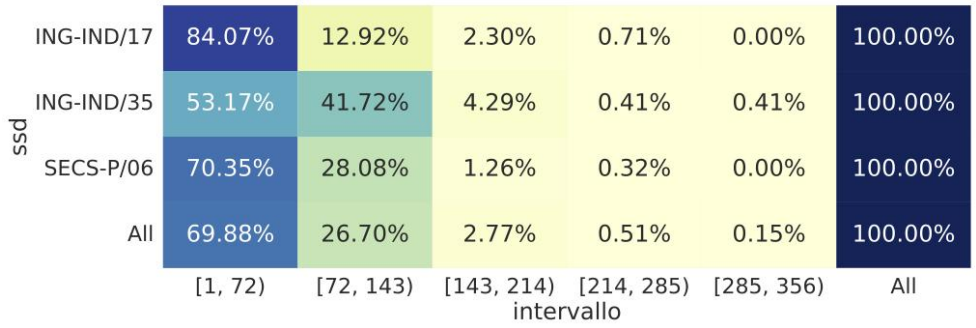


Figura 4.13: Frequency table relativa delle reference dei paper del 2020



Si può notare che tra il 2010 (Figura 4.11) e il 2015 (Figura 4.12) la percentuale di pubblicazioni con più references aumenta notevolmente in tutti e tre i settori (come anche complessivamente, di oltre 10 punti percentuali), avendo anche una piccola parte dei *paper* che ricade tra le 640 e le 710 references.

Nel 2020, Figura 4.13, non sono presenti *paper* con un numero di references così elevato ma i dati tendono a distribuirsi in maniera più uniforme nei diversi intervalli. È notevole il cambiamento che si evidenzia nel settore ING-IND/35 che passa da avere il 25% dei *paper* nel secondo intervallo (tra le 72 e 142 references) ad averne il 41.72%.

Nonostante ciò, l'aumento delle references dei *paper* nel tempo potrebbe essere dovuto anche al miglioramento dei sistemi di *information retrieval* per la costruzione delle references, che rendono questo lavoro più rapido

e meno costoso.

Per poter osservare l'andamento delle references per pubblicazione è stato realizzato un grafico temporale del rapporto tra numero di references fatte nell'anno (sommando le references di ogni *paper*) diviso in numero di pubblicazioni effettuate in quell'anno per ogni SSD. Questo coincide con l'andamento della media nei box-plot (Figure 4.8, 4.9 e 4.10) e ne facilita la visualizzazione, in quanto la curva non è influenzata dalla scala logaritmica dell'asse y. Il grafico ottenuto è presentato in Figura 4.14.

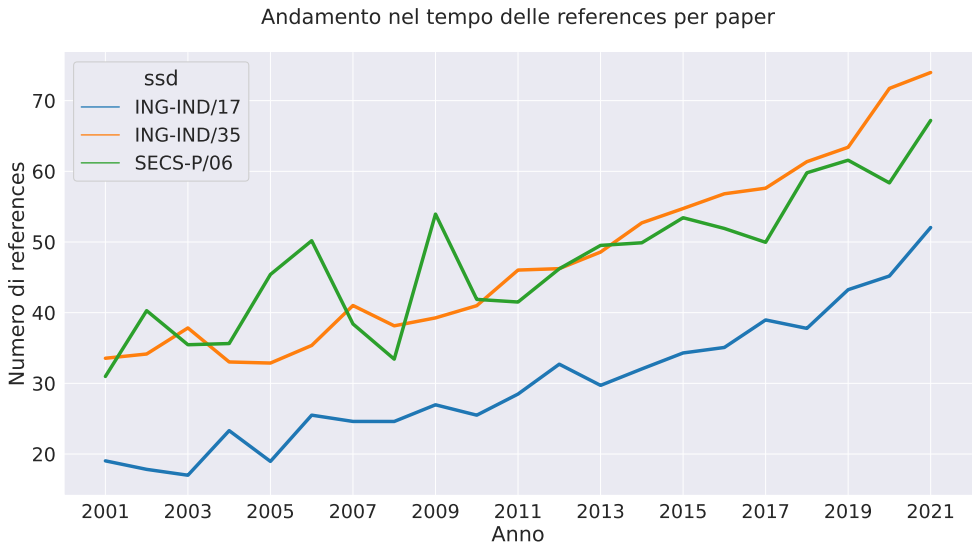


Figura 4.14: Andamento del numero medio di references per paper

Come già osservato dai box-plot, si può notare che l'andamento delle citazioni per *paper* sembra essere simile in tutti i tre settori, con la differenza che le curve di SECS-P/06 e ING-IND/35 tendono ad avere

valori molto simili e più alti rispetto all'altro settore. Il fatto che ING-IND/17 si trovi in un range più basso di references per *paper* potrebbe essere dovuto all'elevato numero di conference *papers* in questo settore (Figura 4.2) che, come già detto precedentemente, possiedono in media meno references di altre tipologie di pubblicazioni, o anche ad articoli più contenuti (nella lunghezza e nello scope) che necessitano di meno references.

4.1.2 Le autocitazioni

Oltre che focalizzarsi sulle citazioni fatte (references delle pubblicazioni), si è ritenuto significativo analizzare anche il numero di auto-citazioni. L'auto-citazione viene definita rispetto a tre livelli di analisi:

- A livello di *paper*;
- a livello di autore;
- a livello di SSD.

L'auto-citazione nei *paper* è definita come una citazione a una pubblicazione scritta da almeno uno tra gli autori del *paper* citante.
L'auto-citazione a livello di autore invece corrisponde all'eventualità che un autore cita se stesso.

L'auto-citazione nel settore rappresenta quella citazione che un autore fa a un altro del suo stesso settore.

La scelta è ricaduta nel raffigurare queste informazioni non in termini assoluti ma rapportati alle citazioni totali fatte dal *paper* (settore o autore). Ciò ha consentito di ottenere una misura relativa che rappresenta quante sono le auto-citazioni rispetto al totale delle citazioni e verrà chiamata *indice di inwardness*.

Per calcolare il numero di autocitazioni delle pubblicazioni e degli autori si è partiti con il creare una nuova colonna alla tabella originale e assegnare ad essa i seguenti valori:

- 1 se l'id dell'autore del *paper* è uguale all'id dell'autore citato;
- 0 altrimenti.

Alcune righe del risultato ottenuto sono riportate in Tabella 4.6.

Id documento	Dc identifier	SSD	Id documento citato	Id autore citato	SSD citato	Anno	Auto-citazione
33746817053	6701373353	ING-IND/17	37109613	6701373353	ING-IND/17	2006	1
33746817053	6701373353	ING-IND/17	37109613	36850960900	ING-IND/17	2006	0
33746817053	6701373353	ING-IND/17	37109613	36785734900	ING-IND/17	2006	0
33746817053	36850960900	ING-IND/17	37109613	6701373353	ING-IND/17	2006	0
33746817053	36850960900	ING-IND/17	37109613	36850960900	ING-IND/17	2006	1
33746817053	36850960900	ING-IND/17	37109613	36785734900	ING-IND/17	2006	0
33746817053	36785734900	ING-IND/17	37109613	6701373353	ING-IND/17	2006	0
33746817053	36785734900	ING-IND/17	37109613	36850960900	ING-IND/17	2006	0
33746817053	36785734900	ING-IND/17	37109613	36785734900	ING-IND/17	2006	1

Tabella 4.6: Esempio: creazione colonna autocitazioni

Citazioni a pubblicazioni scritte dagli autori del paper

Tramite la Tabella 4.6 è stato possibile calcolare il numero di autocitazioni dei *paper* tramite due aggregazioni.

- La prima è stata un raggruppamento per "id doumento" e "id documento citato", utilizzando come funzione di aggregazione il *massimo* della colonna "auto-citazioni". Ciò ha permesso di ottenere una tabella che nella colonna "auto-citazioni" ha 1 se il documento citato contiene autori del documento citante o 0 altrimenti.

- La seconda aggregazione è stata un raggruppamento per "id documento" e la *somma* come funzione di aggregazione della feature numerica "auto-citazioni", in modo da avere, per ogni documento, il numero di *paper* citati che sono stati scritti da autori del documento di partenza.

La Tabella 4.7 rappresenta il risultato finale, affiancato al numero di citazioni totali.

id documento	SSD	Anno	References	Citazioni a paper di autori
85133091370	ING-IND/17	2014	26	1
85133086028	ING-IND/17	2014	17	0
85133074479	ING-IND/17	2014	34	1
85133047286	ING-IND/17	2014	25	0
85133023565	SECS-P/06	2021	63	2
85132767583	SECS-P/06	2020	42	1

Tabella 4.7: Dati per l'indice di inwardness del paper

Il rapporto percentuale tra "citazioni a *paper* di autori" e "references", dunque l'indice di *inwardness* del paper è stato graficato a seguito di un raggruppamento per "Anno" e "SSD" utilizzando la *media* dell'*inwardness* come funzione di aggregazione. Il risultato ottenuto è riportato in Figura 4.15.

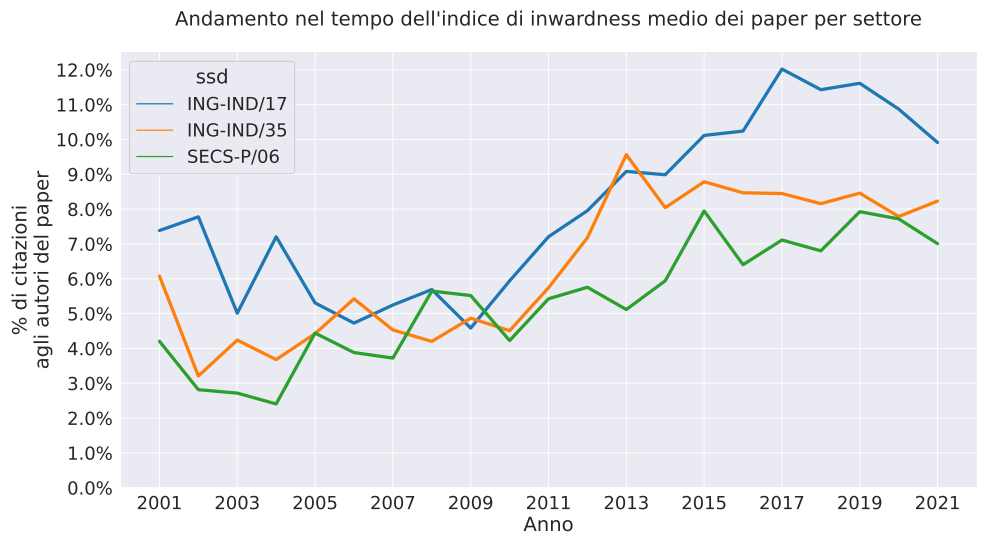


Figura 4.15: Andamento della media dell'indice di inwardness dei paper

Si è deciso di presentare anche l'andamento della mediana dell'indice di *inwardness*. La scelta dell'utilizzo della mediana è motivata dall'elevato numero di *outliers* presenti nei dati, visibili anche dai box-plot. La mediana è meno sensibile alla presenza di *outliers* in quanto li considera come singole osservazioni, contrariamente alla media che tiene conto anche del valore della feature in questione. Il grafico ottenuto è visibile in Figura 4.16.

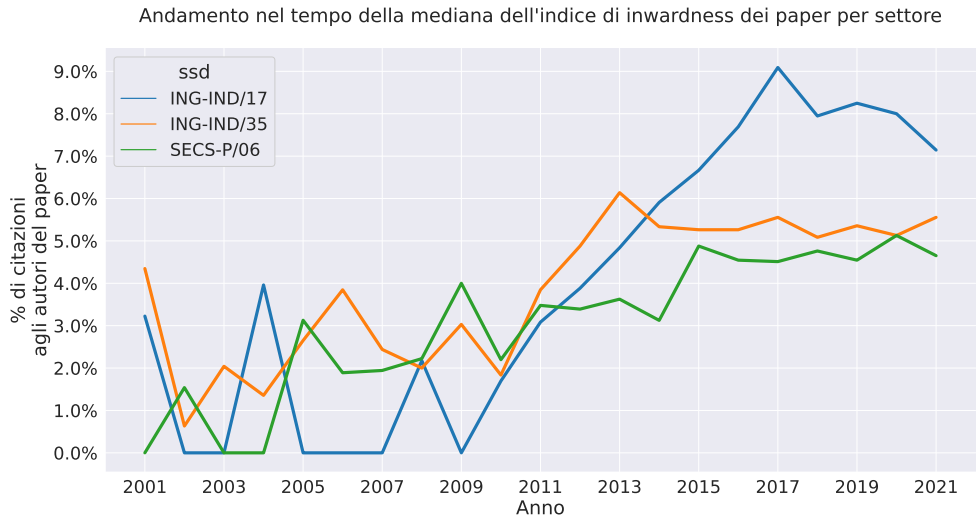


Figura 4.16: Andamento della mediana dell'indice di inwardness dei paper

Dalle Figure 4.15 e 4.16 si può subito notare che per tutti e tre i settori la situazione è particolarmente instabile fino al 2010. Dal 2011 in poi si osserva una crescita rilevante dell'indice di *inwardness* nel settore ING-IND/17 che raggiunge una media del 12% nel 2017. Come nel grafico

4.14, si osserva una somiglianza tra i due restanti settori. Infatti, pur essendoci stata una crescita notevole anche in ING-IND/35 tra il 2011 e il 2013, in seguito la situazione si è stabilizzata, raggiungendo gli stessi livelli di SECS-P/06. La situazione di plateau che interessa i settori ING-IND/35 e SECS-P/06 a seguito del 2015 implica una crescita delle citazioni agli autori del *paper* direttamente proporzionale a quella delle references, visibile in Figura 4.14.

È però interessante confrontare la crescita improvvisa della curva in entrambi i settori bibliometrici (ING-IND/17 e ING-IND/35) nei due anni successivi alla riforma con la risposta ridardataria riscontrata nel settore bibliometrico (Figura 4.16). Si potrebbe ipotizzare che ciò sia dovuto alla riforma stessa e dunque all'introduzione, solo per alcuni settori (tra cui ING-IND/17 e ING-IND/35), di indici bibliometrici come sistema di valutazione della ricerca, sulla quale vengono basati gli avanzamenti di carriera.

La curva di SECS-P/06, invece, subisce un crescita rilevante solo successivamente, a cavallo tra il 2014 e il 2015. Questo potrebbe essere conseguenza dell'esito delle tornate di abilitazione avvenute nel 2012 e 2013. Si può supporre che a seguito degli esiti sia stata notata la rilevanza delle citazioni come criterio ai fini delle valutazioni.

Un altro aspetto da evidenziare in Figura 4.15 è la decrescita della curva di ING-IND/17 a seguito del 2017. Le references dei *paper* in questo settore fino al 2021 rimangono comunque in crescita (vedi Figura 4.14) e di conseguenza il ridursi dell'indice di *inwardness* dei *paper* potrebbe essere motivato da:

- numero medio di citazioni agli autori della pubblicazione costante;
- numero medio di citazioni agli autori decrescente e, quindi, un effettiva modifica nel comportamento dei docenti nel settore.

In Tabella 4.8 si riportano i valori medi annuali del numero di citazioni agli autori per il settore ING-IND/17 dal 2017 al 2021.

SSD	**Anno**	**Citazioni agli autori (media)**
ING-IND/17	2017	3,86
ING-IND/17	2018	3,43
ING-IND/17	2019	3,86
ING-IND/17	2020	3,95
ING-IND/17	2021	4,27

Tabella 4.8: Media delle citazioni agli autori per ING-IND/17 negli anni 2017-2021

Osservando i valori della Tabella 4.8 si può notare che le citazioni agli autori rimangono pressocchè costanti negli anni (tranne piccole variazio-

ni) e dunque non sembra esserci una cambiamento del comportamento dei docenti in questo settore.

Autocitazioni degli autori

In seguito è stato calcolato l'indice di *inwardness* specifico per il singolo autore, quindi la percentuale di autocitazioni sul totale delle citazioni fatte dall'autore negli anni.

Per ottenere i dati utili alla realizzazione del grafico è stato effettuato un raggruppamento della Tabella 4.6 per "dc_identifier", "SSD", "Anno", "id documento" utilizzando come funzione di aggregazione per la variabile "autocitazioni" la *somma*. Questo ha permesso di ottenere il numero di autocitazione fatte da un autore per ogni documento, affiancando a ciò il numero di citazioni totali del documento.

I dati sono stati nuovamente aggregati per "dc_identifier", "SSD", "Anno" al fine di ottenere in numero di autocitazioni e il numero di citazioni complessive annuali, così come riportato in Tabella 4.9.

dc identifier	SSD	Anno	Auto-citazioni	Citazioni fatte
6503864967	ING-IND/17	2013	1	8.0
6503864967	ING-IND/17	2014	11	322.0
6503864967	ING-IND/17	2015	2	88.0
6503864967	ING-IND/17	2016	7	184.0
6503864967	ING-IND/17	2017	9	177.0
6503864967	ING-IND/17	2018	14	265.0
6503864967	ING-IND/17	2019	43	467.0
6503864967	ING-IND/17	2020	8	221.0
6503864967	ING-IND/17	2021	6	207.0

Tabella 4.9: Dati per l'indice di inwardness dell'autore

Per poter visualizzare l'andamento dell'indice di *inwardness* degli autori (rapporto percentuale tra "autocitazioni" e "citazioni fatte") negli anni per SSD sono state considerate sia la media sia la mediana[2] di questi valori e sono stati realizzati i grafici presentati nelle Figure 4.17 e 4.18.

Le Figure 4.17 e 4.18 rispecchiano la situazione precedente e sottolineano ancor di più la crescita dell'indice di *inwardness* degli autori nel settore ING-IND/17 a seguito della riforma a cavallo tra il 2010 e il 2011.
Anche in ING-IND/35 la percentuale di autocitazioni degli autori è stata in crescita fino al 2013, stabilizzandosi in seguito. Permane comunque quella che potrebbe essere considerata un'immediata risposta all'intro-

[2]La scelta di riportare anche il grafico della mediana è stata presa, come nel caso precedente, onde evitare che eventuali *outliers* influenzino eccessivamente l'andamento dei valori non considerati anomali.

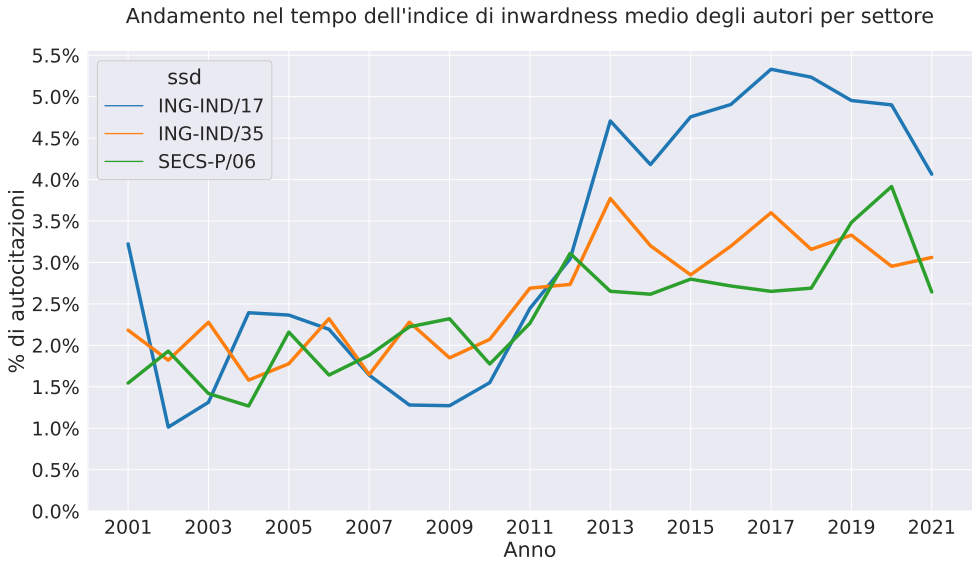


Figura 4.17: Andamento della media dell'indice di inwardness degli autori

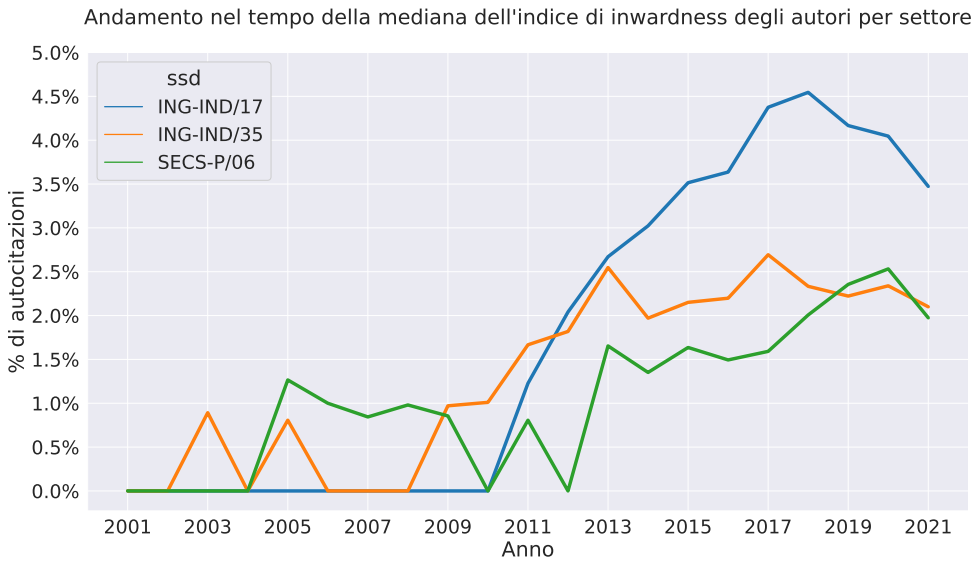


Figura 4.18: Andamento della mediana dell'indice di inwardness degli autori

duzione degli indici bibliometrici da parte dei settori disciplinari coinvolti, comportamento non riscontrato invece in SECS-P/06 che vede

un'instabilità nell'indice fino al 2012 e una crescita immediata nel 2013.

Citazioni interne ai settori scientifico-disciplinari

L'ultimo punto di vista sotto il quale è stato analizzato l'indice di *inwardness* è quello del settore disciplinare.

Per il calcolo delle citazioni interne al settore si è proceduto come segue.

A partire dalla Tabella 4.2, è stata creata un'ulteriore colonna "citazioni interne al settore" che, a differenza della tabella 4.6 contiene:

- 1 se l'SSD dell'autore citante è uguale all'SSD dell'autore citato;
- 0 altrimenti.

A questo punto sono state effettuate due aggregazioni successive:

- La prima è stata un raggruppamento per "id documento", "SSD", "Anno", "id documento citato" con funzione di aggregazione *massimo* per la feature "citazioni interne al settore", così da ottenere:
  - 1 se il *paper* citato contiene autori dello stesso SSD;
  - 0 altrimenti.
- La seconda è stata un raggruppamento per "id documento", "SSD", "Anno" con funzione di aggregazione *somma* per le "citazioni interne al settore" in modo tale da avere il numero di volte che un

SSD ha citato *paper* scritti da autori dello stesso SSD per ogni documento.

Dopo aver affiancato a questo risultato la colonna contenente le *references* totali del *paper*, è stato calcolato l'indice di *inwardness* del singolo *paper* per ogni SSD come il rapporto tra il numero di citazioni a *paper* interni al settore e il numero di *references* totali. Al fine di ottenere un unico valore per ciascun anno e SSD è stato effettuato un raggruppamento per queste due variabili con funzione di aggregazione *media* per l'indice di "*inwardness*".

La percentuale di citazioni interne al settore stesso, è stato rappresentato graficamente in Figura 4.19.

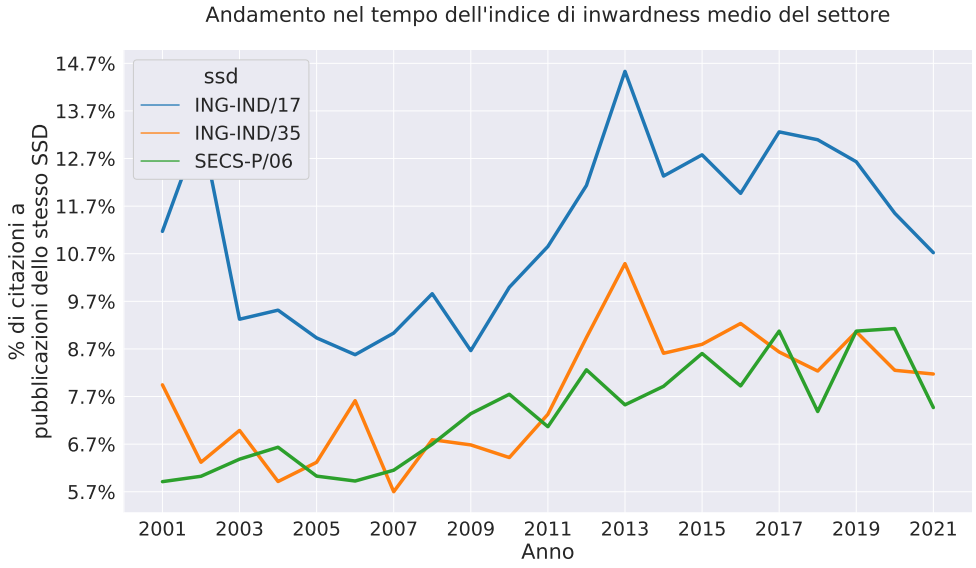


Figura 4.19: Andamento dell'indice di inwardness degli SSD

Quest'ultimo grafico non fa che andare a conferma delle ipotesi fatte precedentemente in quanto, per i due settori ING-IND/17 e ING-IND/35 si osserva una rapida crescita dell'indice di *inwardness* a seguire del 2010, fenomeno non riscontrato nell'unico settore non bibliometrico preso in analisi. I valori di SECS-P/06, per quanto crescenti nel tempo, non subiscono variazioni rilevanti. Ciò potrebbe essere motivato dal fatto che questo è un settore meno numeroso rispetto agli altri e più eterogeneo, il che comporta maggiori citazioni verso gli esterni.

Un'informazione da mettere a confronto con la Figura 4.19 è la percentuale di citazioni fatte a pubblicazioni dello stesso SSD al di fuori di quelle che sono state scritte da autori del *paper*. Questo corrisponde a scorporare dai valori presenti nella Figura 4.19 le pubblicazioni che costituiscono autocitazioni. Il grafico che ne deriva è presentato in Figura 4.20.

Questo grafico sottolinea ancor di più il fatto che per il settore ING-IND/17 a seguire del 2010 è diminuita la percentuale di citazioni verso gli autori esterni in quanto quelle intra-settoriali aumentano del 2% in due anni (dal 2011 al 2013). La medesima osservazione non ha la stessa rilevanza per il settore ING-IND/35 che vede sì una crescita a seguito dell'anno 2010, ma senza picchi rilevanti. Focalizzando l'attenzione

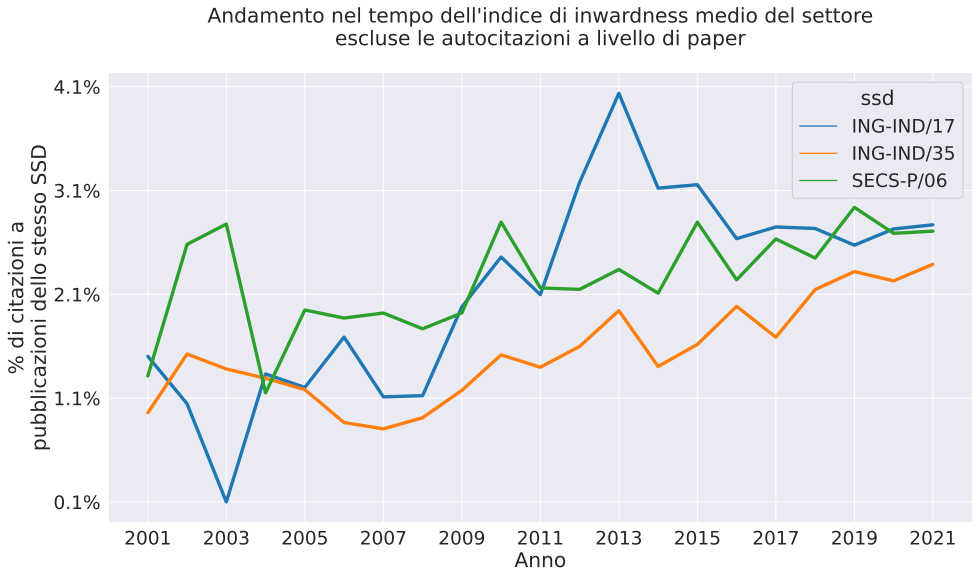


Figura 4.20: Andamento dell'indice di inwardness degli SSD, escluse le autocitazioni a livello di paper

sul settore SECS-P/06, per quanto i valori siano più elevati rispetto a ING-IND/35, non si osservano *trend* crescenti così evidenti quanto i due restanti settori.

4.2 Costruzione del network

Con i dati a disposizione estratti dal database di Scopus è possibile ricavare i collegamenti tra gli autori (sulla base delle citazioni) generando una rete. Infatti, effettuando un raggruppamento della Tabella 4.2 per "Id autore", "Anno" e "Id autore citato" utilizzando come funzione di aggregazione il *conteggio*, è possibile ottenere informazioni quali il numero

di volte in cui un autore cita un altro in un determinato anno. Sono stati esclusi sia i collegamenti tra co-autori, che rappresentano l'eventualità in cui un autore cita in una pubblicazione un *paper* di un altro autore della pubblicazione di partenza, sia le autocitazioni.
L'insieme di tutti i collegamenti tra gli autori ha permesso la creazione di una rete che si allarga nel corso del tempo in quanto aumentano gli individui coinvolti e le citazioni cumulative fatte. La rete è stata raffigurata tramite un grafo orientato interattivo che ha le seguenti caratteristiche:

- i nodi rappresentano gli autori e sono etichettati con l'identificativo Scopus ad essi associato;
- il colore dei nodi rappresenta il settore scientifico-disciplinare di appartenenza;
- lo spessore degli archi indica il numero di citazioni effettuate dall'autore citante a quello citato, più l'arco è spesso e maggiore è il numero di citazioni fatte.

É stato realizzato un grafo per ogni anno tra il 2001 e il 2021, considerando per ciascuno di essi il numero di citazioni cumulative, ossia fatte fino all'anno in questione.

Per semplificare la visualizzazione del grafo e limitare la quantità di informazioni presentata (cercando di escludere dati poco rilevanti), si è deciso di rappresentare il collegamento tra due autori solamente se il peso dell'arco è maggiore o uguale a 20 e, dunque, se il numero di citazioni fino a un dato anno tra l'autore citante e quello citato è maggiore o uguale a 20. Questa soglia può essere variata ottenendo livelli di dettaglio del grafo differenti.

Le Figure 4.21, 4.22 e 4.23 rappresentano i grafi in tre anni differenti: 2010, 2015, 2020. Nei grafi, la legenda coincide con quella dei grafici precedenti:

- il blu rappresenta ING-IND/17;
- l'arancione rappresenta ING-IND/35;
- SECS-P/06 è indicato dal colore verde.

Figura 4.21: Grafo citazionale fino al 2010

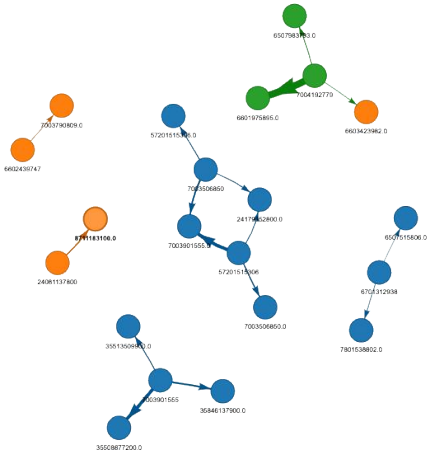


Figura 4.22: Grafo citazionale fino al 2015

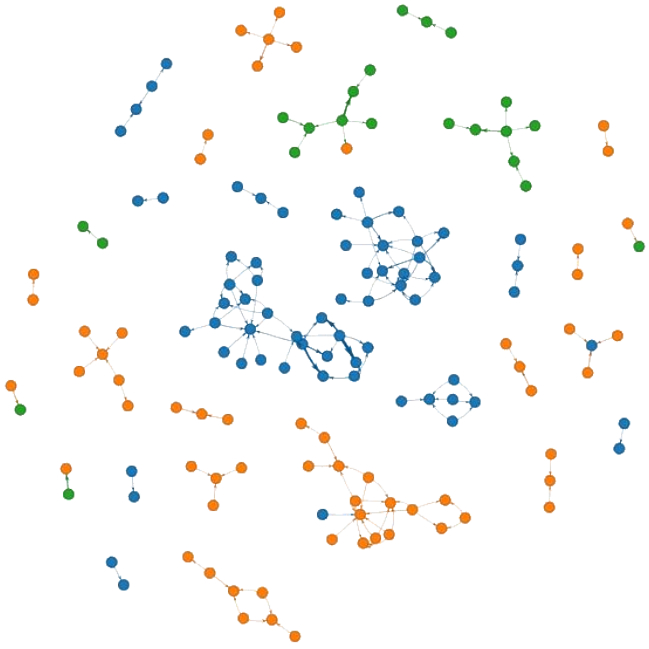
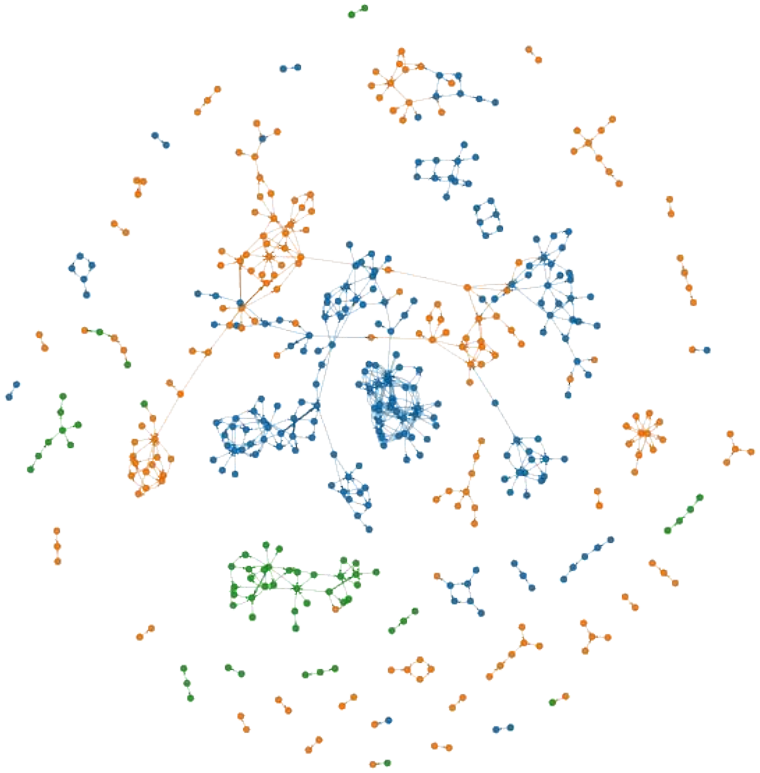


Figura 4.23: Grafo citazionale fino al 2020



Dalle Figure 4.21, 4.22 e 4.23 è possibile osservare la crescita della rete nel tempo e la creazione di *cluster* citazionali, ossia di gruppi di autori che si citano a vicenda. Di conseguenza, la rappresentazione a grafo delle informazioni ricavate da Scopus, utilizzando una soglia fissa per gli archi, consente di individuare gruppi di autori che si citano tra loro e si ampliano con gli anni, facendo anche risaltare i legami che si instaurano tra autori di settori scientifico-disciplinari differenti.

Al fine di sottolineare la crescita effettiva del numero di citazioni tra un anno e l'altro sono stati realizzati i grafi ottenuti utilizzando una soglia variabile che utilizza come soglia il valore minimo del peso degli archi che si trovano nel primo 0.6%, ordinati in maniera decrescente. In questo modo, la complessità crescente dei grafi indica un aumento rilevante del peso delle connessioni tra i nodi. In particolare, le soglie degli archi dei grafi per gli anni 2010, 2015 e 2020 sono:

- 20 per il 2010;
- 29 per il 2015;
- 38 per il 2020.

I risultati ottenuti sono quasi analoghi a quelli riportati nelle Figure 4.21, 4.22 e 4.23 a indicare che, pur facendo aumentare la soglia nel corso degli anni, si nota ancora una rilevante crescita del grafo.

Capitolo 5

Conclusione e sviluppi futuri

Le analisi descrittive e preliminari suggeriscono, quindi, un cambiamento nel comportamento citazionale dei docenti dei settori bibliometrici, sebbene da queste non sia possibile supportare l'evidenza di un legame causale con l'introduzione dell'ASN (Abilitazione Scientifica Nazionale). Focalizzando l'attenzione sui diversi indici di *inwardness* (a livello di *paper*, di autore e di settore), così come definiti nella sotto-sezione 4.1.2, risultano evidenti sia la crescita delle curve di ING-IND/17 e ING-IND/35 (entrambi settori bibliometrici) subito dopo l'introduzione della Legge 240 del 30 Dicembre del 2010 [2], sia la risposta ritardata di SECS-P/06, settore non bibliometrico. Di conseguenza, è possibile ipotizzare che ciò rappresenti una risposta strategica all'introduzione degli indici bibliome-

trici come metrica di valutazione della ricerca ai fini dell'abilitazione a professore ordinario, di prima e seconda fascia.

Lo sviluppo di analisi causali attraverso l'utilizzo di modelli di *impact/-policy analysis* è una promettente direzione di ricerca "aperta" dal lavoro in oggetto.

Di seguito vengono presentati alcuni sviluppi futuri che potrebbero essere intrapresi in conseguenza del del progetto di ricerca descritto nelle pagine precedenti.

Ottimizzazione della procedura di disambiguazione

Utilizzo dei co-autori Per migliorare l'accuratezza dell'algoritmo di assegnazione del codice Scopus a ogni docente, si potrebbero utilizzare i co-autori degli identificativi ambigui. In questo modo se uno o più co-autori di un identificativo coincidono con autori già correttamente identificati, allora è probabile che quello sia l'identificativo corretto, in quanto le collaborazioni tra docenti dello stesso SSD sono molto frequenti.

Utilizzo delle *past affiliations* Una tra le possibili strade da percorrere per raffinare la procedura di disambiguazione dei risultati è quella dell'utilizzo delle affiliazioni passate degli autori. Come presentato nel Capitolo 2, Sezione 2.1, l'affiliazione dell'autore presente nelle risposte di

Scopus non è stata utilizzata ai fini della disambiguazione per problemi di ambiguità dell'affiliazione stessa (ad esempio più di un identificativo e nome per la stessa affiliazione) e di corrispondenza con l'affiliazione del docente originale. Attuando però una procedura di disambiguazione delle affiliazioni Scopus al fine di associarle a quelle dei docenti di partenza, sarebbe possibile sfruttare questo campo. Ciononostante, questo non permetterebbe di superare il problema di un eventuale cambio recente di affiliazione dei professori; ciò si potrebbe risolvere estraendo da Scopus anche le affiliazioni passate degli autori, così da possedere in aggiunta dati storici.

Peso della subject area MED Nella procedura di disambiguazione dei risultati, in tutti settori scientifico-disciplinari presi in esame, alla subject area MED (*Medicine*) viene sempre dato molto peso, nonostante non sia un'area tematica così rilevante nei tre SSD. Ciò avviene perchè, come mostrato dal Professor Marek Kwiek nel seminario di *Research Infrastructure for Science and Innovation Policy Studies*, circa il 40% degli autori su Scopus appartengono a MED. Questo comporta un elevato numero di identificativi plausibili appartenenti a quell'area tematica e una conseguente maggior probabilità di commettere errori. Essendo una

subject area molto presente tra i risultati, questa peserà notevolmente nel calcolo dello score complessivo degli identificativi ad essa collegati, rischiando quindi un'assegnazione errata.
Una soluzione potrebbe essere quella di assegnare manualmente un peso inferiore qualora si volessero individuare gli identificativi di autori non inerenti al settore.

Ampliamento degli SSD utilizzati come test Al fine di poter assegnare la stessa combinazione di parametri della procedura a più settori scientifico-disciplinari, sarebbe utile aumentare il numero di SSD utilizzati per il calcolo delle *performance* della procedura stessa. L'utilizzo di più settori, infatti, permetterebbe una suddivisione in *cluster* sulla base delle caratteristiche intrinseche del settore (aree tematiche, numerosità,...) e sarebbe possibile verificare se le combinazioni di parametri ottimali di ogni SSD coincidono per quelli che si trovano nello stesso gruppo.

Ulteriori analisi

A partire dalle informazioni disponibili e dalle analisi già effettuate, si potrebbe approfondire la struttura a grafo dei dati al fine di individuare eventuali possibili *club citazionali*, gruppi di autori che hanno instaurato tra loro collegamenti che potrebbero essere considerati strategici, i.e.

finalizzati a incrementare le performance bibliometriche attraverso pratiche collaborative, collusive o "di scambio". Sarebbe interessante verificare se a seguito del 2010 la presenza di gruppi citazionali si intensifichi sia in termini di numero di gruppi, ma anche di coesione tra il gruppo stesso. Per fare ciò bisognerebbe utilizzare un algoritmo di individuazione di sotto-grafi più frequenti (con determinate caratteristiche topologiche) e verificare l'orientamento degli archi che legano i nodi del sotto-grafo stesso.

Appendice A

Codice Python della procedura di disambiguazione e tuning dei parametri

```
import pandas as pd
import sqlalchemy as sql
from sqlalchemy.types import Integer, String, Float, Boolean
import re
import numpy as np
from typing import List, Union
def calculate_distance(
    string_a: str, string_b: str, calculate_ratio: bool = False
```

```
) -> Union[int, float]:

    # Make sure string_a and string_b are string types

    assert type(string_a) is str, "string_a must be of type str"

    assert type(string_b) is str, "string_b must be of type str"


    # Initialize matrix of zeros

    rows: int = len(string_a) + 1

    cols: int = len(string_b) + 1

    distance: np.ndarray = np.zeros((rows, cols), dtype=int)


    # Populate matrix of zeros with the indeces of each character of both strings

    for i in range(1, rows):

        for j in range(1, cols):

            distance[i][0] = i

            distance[0][j] = j


    # Iterate over the matrix to compute the cost of deletions, insertions and/or
        substitutions

    for col in range(1, cols):

        for row in range(1, rows):

            if string_a[row - 1] == string_b[col - 1]:

                cost = 0 # If the characters are the same in the two strings in a
                    given position [i,j] then the cost is 0

            else:
```

```
                # In order to align the results with those of the Python
                    Levenshtein package, if we choose to calculate the ratio
                # the cost of a substitution is 2. If we calculate just distance,
                    then the cost of a substitution is 1.
                if calculate_ratio:
                    cost = 2
                else:
                    cost = 1
            distance[row][col] = min(
                distance[row - 1][col] + 1,  # Cost of deletions
                distance[row][col - 1] + 1,  # Cost of insertions
                distance[row - 1][col - 1] + cost,
            )  # Cost of substitutions


    if calculate_ratio:
        # Computation of the Levenshtein Distance Ratio
        ratio: float = ((len(string_a) + len(string_b)) - distance[row][col]) / (
            len(string_a) + len(string_b)
        )
        return ratio
    else:
        # This is the minimum number of edits needed to convert string a to string
            b
        return distance[row][col]
```

```
################

engine = sql.create_engine('mysql://root:Teabreak%4097@127.0.0.1/scopus')

connection = engine.connect()

metadata = sql.MetaData()

relSubj = [90, 80, 70]

edNome = [0.5, 0.3, 0.1]

edCognome = [0.5, 0.3, 0.1]

steepnessSigmoid = [1.7, 1.6, 1.5]

alpha = [0.6, 0.5, 0.4]

beta = [0.2, 0.1, 0.05]

rate = [0.6, 0.5]

settori =['ING-IND/17','ING-IND/35','SECS-P/06']

anni = np.arange(2001,2022)

cnt = 0

for subject_rate in relSubj:

    for parametro_nome in edNome:

        for parametro_cognome in edCognome:

            for sigmoide in steepnessSigmoid:

                for parametro_alpha in alpha:

                    for parametro_beta in beta:

                        for parametro_rate in rate:

                            cnt = cnt + 1

                            #if cnt <= paper in cui si      interrotta :

                                continue
```

```
for settore in settori :
    for anno in anni:
        query = """SELECT \
            a.*, b.nome, b.cognome \
            FROM scopus.dati_scopus_per_score as a,
                scopus.dati_miur as b\
            WHERE a.id_originale = b.id and b.ssd =
                %s and b.anno = %s and a.ssd = b.ssd
                and a.anno = b.anno """
        res = engine.execute(query,[settore,int(anno)
            ])
        df = pd.DataFrame(res.fetchall())
        subjects_df = df[['dc_identifier','
            subject_area']]
        subjects_temp = subjects_df['subject_area'].
            str.split(',', expand = True)
        subjects_df.drop(columns=['subject_area'],
            inplace = True)
        subjects_df.reset_index(drop=True)
        subjects_df = pd.concat([subjects_df,
            subjects_temp], axis=1)
        subjects_melted = pd.melt(subjects_df,
            id_vars = 'dc_identifier',
            var_name = 'Attribute',
```

```
    value_name = 'Value')
subjects_melted.dropna(inplace = True)
subjects_melted.drop(columns = ['Attribute'],
    inplace = True)
subjects_melted[['sub', 'n_doc']] =
    subjects_melted['Value'].str.split(':',
    expand=True)
subjects_melted.drop(columns=['Value'],
    inplace=True)


subjects_melted.to_sql(
    'subjects',
    engine,
    if_exists='replace',
    index=False,
    chunksize=100,
    dtype={
        'dc_identifier': String(100),
        'sub':String(10),
        'n_doc':Integer
    }
)


df_scores = pd.DataFrame()
```

```
df_scores['id_originale'] = df['id_originale']
df_scores['dc_identifier'] = df['dc_identifier
    ']
df_scores['ed_nome'] = ''
# edit distance nome
for i in range(len(df)):
    dist = max(len(df.at[i, 'nome'].strip()),
        len(df.at[i, 'name'].strip()))
    fDist = float(dist - calculate_distance(
        df.at[i, 'nome'].strip(), df.at[i, '
        name'].strip())) / float(dist)
    df_scores['ed_nome'][i] = fDist
# edit distance cognome
df_scores['ed_cognome'] = ''
for i in range(len(df)):
    dist = max(len(df.at[i, 'cognome'].strip()
        ), len(df.at[i, 'surname'].strip()))
    fDist = float(dist - calculate_distance(
        df.at[i, 'cognome'].strip(), df.at[i,
        'surname'].strip())) / float(dist)
    df_scores['ed_cognome'][i] = fDist
    print('1')
# weights for subjects+n_papers
print('a')
```

```
query_1 = (" create view
    first_weights_subjects as( \
        with temp as( \
        SELECT *, count(*) over (partition
            by sub) as occ \
        FROM scopus.subjects) \
        select temp.*, rate \
        from temp,( select temp.*, coalesce
            (rate_1, 0) as rate, tot \
        from temp left join (select * from
            (select *, occ/tot*100 rate_1
            \
        from temp b, (select count(*) as
            tot from temp) a \
        having sub != '' )y) z \
        on (temp.dc_identifier = z.
            dc_identifier and temp.sub =
            z.sub)) d \
        where temp.dc_identifier = d.
            dc_identifier and temp.sub =
            d.sub) ")
res_1 = engine.execute(query_1)
query_sub_1 = """select sub, rate
            from first_weights_subjects
```

```
                    group by sub
                    having sub != ''"""
res_sub_1 = engine.execute(query_sub_1)
print('c')
sub_pesi = pd.DataFrame(res_sub_1.fetchall
    ())
print('d')
limit_sub = round(len(sub_pesi)*
    subject_rate/100)
print('e')
query_sub_2 = """create view pesi_subjects as
     (select sub, rate
                    from first_weights_subjects
                    group by sub
                    having sub != ''
                    order by rate desc
                    limit %s)"""
res_sub_2 = engine.execute(query_sub_2,
    int(limit_sub))
print("b")
query_2_1="""create table subj_weights as (
    with temp as (select a.dc_identifier, a.
    sub, a.n_doc, a.occ, coalesce(b.rate, 0)
    as rate
```

```
        from first_weights_subjects a left join
            pesi_subjects b on a.sub = b.sub)
        SELECT *, x.occ/y.n_sub_relevant as
            weight
        FROM temp x, (select count(sub) as
            n_sub_relevant
        from temp where rate > 0) y
        where rate > 0
        )"""
    res_2_1 = engine.execute(query_2_1)
    print("b")
    query_2 = """
        select x.dc_identifier, coalesce(sum(x.
            n_doc*weight)/sum(x.n_doc)*10, 0)
            as peso_sub\
        from first_weights_subjects x left join
            subj_weights y on (x.dc_identifier =
            y.dc_identifier)\
        and x.sub = y.sub\
        group by dc_identifier """
    res_2 = engine.execute(query_2)
    weights_df = pd.DataFrame(res_2.fetchall())
    print("b")
```

```
weights_df.columns = ['dc_identifier_1', '
    peso_sub']
weights_df['dc_identifier_1']=weights_df['
    dc_identifier_1'].astype(str)
df_scores['dc_identifier']=df_scores['
    dc_identifier'].astype(str)
df_scores.set_index(['dc_identifier'], drop=
    False, inplace = True)
weights_df.set_index('dc_identifier_1', drop=
    True, inplace = True)
df_scores['peso_sub']=''
for identifier_1 in df_scores.index:
    for identifier_2 in weights_df.index:
        if identifier_1==identifier_2:
            df_scores.at[identifier_1,'
                peso_sub'] = weights_df.at[
                identifier_2,'peso_sub']
        else:
            continue
df_scores.reset_index(drop = True, inplace =
    True)
print(df_scores)
df_scores.columns = ['id_originale', '
    dc_identifier', 'ed_nome', 'ed_cognome'
```

```
        , 'peso_sub']
    df_scores['n_occ_1'] = df.groupby('
        id_originale')['id_originale'].transform('
        count')
    df_scores['score_tot'] = ''
    for i in range(len(df_scores)):
        if pd.isnull(df_scores.at[i, 'peso_sub']):
            df_scores['score_tot'][i] = 0
        elif (df_scores.at[i, 'ed_nome'] >=
            parametro_nome and df_scores.at[i, '
            ed_cognome'] >=
            parametro_cognome):
            df_scores['score_tot'][i] = (float(
                df_scores.at[i, 'ed_nome'])+
                                    float(
                                        df_scores
                                        .at[i
                                        , '
                                        ed_cognome
                                        '])) +

                                        float
                                        (
                                        df_scores
```

```
                                                    .at[i,
                                                     ,
                                                    peso_sub
                                                    ']))
        else:
            df_scores['score_tot'][i] = 0
    #df_scores.drop('dc_indentifier_1', axis = 1,
     inplace = True)
    df_scores['decisione_nome'] = ''
    for i in range(len(df_scores)):
        if (df_scores.at[i, 'score_tot'] == 0
            and df_scores.at[i,'n_occ_1']>1):
            df_scores['decisione_nome'][i] = 'no'
        elif df_scores.at[i, 'score_tot'] > 0
            and df_scores.at[i,'n_occ_1']>1:
            df_scores['decisione_nome'][i] = 'si'
        else:
            df_scores['decisione_nome'][i] = '
                unica'
    df_scores['score_double_sigmoid'] = ''
    for i in range(len(df_scores)):
        df_scores['score_double_sigmoid'][i] =
            float(1 - np.exp(-((df_scores.at[i, '
            score_tot'] / sigmoide) ** 2)))
```

```
print("c")
df_scores.to_sql(
    'df_scores',
    engine,
    if_exists = 'replace',
    index = False,
    chunksize = 100,
    dtype = {
        'id_originale': Integer,
        'dc_identifier': String(100),
        'ed_nome': Float,
        'ed_cognome': Float,
        'peso_sub': Float,
        'n_occ_1': Integer,
        'score_tot': Float,
        'decisione_nome': String(5),
        'score_double_sigmoid': Float
    })
print("b0")
query_3 = """drop view scopus.
    first_weights_subjects;"""
res_3 = engine.execute(query_3)
print("b1")
```

```
query_drop_pesi = """drop view pesi_subjects
    """
res_drop_pesi = engine.execute(
    query_drop_pesi)
print("b")

query_4 = """create table temp as (\
                SELECT df_scores.*, if(
                    df_scores.decisione_nome
                     = 'unica', 'unica', if(b.
                    dc_identifier is not null
                    , 'no','si')) as
                    decisione_sigm\
                FROM df_scores left join (
                    SELECT *\
                                from
                                    df_scores
                                    \
                                where
                                    score_double_sigmoid
                                    <
```

```
                                %
                                s
                                and

                                decisione_nome
                                !='
                                unica
                                ')

                                b
                                \
                            on df_scores.
                                dc_identifier
                                 = b.
                                dc_identifier
                                 and
                                df_scores
                                .
                                id_originale
                                 = b.
                                id_originale
                                )"""
```

```
res_4 = engine.execute(query_4, float(
    parametro_alpha))


query_10 = """create table temp_2 as (with
    temp_1 as (SELECT distinct (y.
    dc_identifier), y.id_originale\
                                FROM temp x
                                    , temp y\
                                where x.
                                    dc_identifier
                                     <> y.
                                    dc_identifier
                                     and x.
                                    id_originale
                                    =y.
                                    id_originale
                                    \
                                and x.
                                    score_tot
                                     - y.
                                    score_tot
                                     >= %s\
                                and y.
                                    decisione_sigm
```

```
= 'si')\
select temp.*, if(temp.
decisione_nome = 'unica', '
unica', if(temp.
decisione_sigm = 'no', 'no',if
(temp_1.dc_identifier is not
null, 'no', 'si'))) as
decisione_diff_score_tot\
from temp left join temp_1 on
temp.dc_identifier = temp_1.
dc_identifier\
and
temp
.
id_originale
=
temp_1
.
id_originale
)
"""
```

```
res_10 = engine.execute(query_10, float(
    parametro_beta))



query_13 = """drop table temp"""
res_13 = engine.execute(query_13)
query_14 = """drop table df_scores"""
res_14 = engine.execute(query_14)

query_15 = """select * from temp_2"""
res_15 = engine.execute(query_15)
final_table_weights = pd.DataFrame(res_15.
    fetchall())
final_table_weights.to_sql(
    'df_scores_finali',
    engine,
    if_exists = 'append',
    index = False,
    chunksize = 100,
    dtype = {
        'id_originale': Integer,
        'dc_identifier': String(100),
        'ed_nome': Float,
        'ed_cognome': Float,
```

```
        'peso_sub': Float,
        'n_occ_1': Integer,
        'score_tot': Float,
        'decisione_nome': String(5),
        'score_double_sigmoid': Float,
        'decisione_sigmoid': String(5),
        'decisione_diff_score_tot': String(5)
    })

query_17 = """drop table temp_2"""
res_17 = engine.execute(query_17)
query_18 = """create view temp as (select
    id_originale, count(
    decisione_diff_score_tot) as n_si
from df_scores_finali
where decisione_diff_score_tot = 'si'
group by id_originale
having n_si > 1)"""
res_18 = engine.execute(query_18)
query_19 = """create table finale as (select
    df_scores_finali.*, coalesce(temp.
    id_originale,0) as ambiguo
from df_scores_finali left join temp on
    df_scores_finali.id_originale = temp.
```

```
        id_originale)"""
res_19 = engine.execute(query_19)
query_28 = """alter table  finale
add column assegnazione varchar(10)"""
res_28 = engine.execute(query_28)
query_20 = """update finale
set  assegnazione = 'si' where
        decisione_diff_score_tot = 'unica'"""
res_20 = engine.execute(query_20)
query_21 = """update finale
set  assegnazione = 'si' where
        decisione_diff_score_tot = 'si' and
        ambiguo = 0"""
res_21 = engine.execute(query_21)
query_22 = """update finale
set  assegnazione = 'no' where
        decisione_diff_score_tot = 'no'"""
res_22 = engine.execute(query_22)
query_23 = """update finale
set  assegnazione = 'ambiguo' where
        decisione_diff_score_tot = 'si' and
        ambiguo != 0"""
res_23 = engine.execute(query_23)
query_24 = """alter table  finale
```

```
drop column ambiguo"""
res_24 = engine.execute(query_24)
#query_25 = """drop table df_scores_finali"""
#res_25 = engine.execute(query_25)
query_26 = """select id_originale,
    dc_identifier, nome, cognome
from finale, dati_miur
where assegnazione = 'ambiguo' and finale.
    id_originale = dati_miur.id"""
res_26 = engine.execute(query_26)
ambiguo = pd.DataFrame(res_26.fetchall())
if len(ambiguo) == 0:
    ambiguo = pd.DataFrame(columns =['
        id_originale', 'dc_identifier', 'nome', '
        cognome'], index = ['0'])
    for i in range(4):
        ambiguo.iat[0,i] = None
ambiguo.to_sql(
    'ambigui',
    engine,
    if_exists = 'replace',
    index = False,
    chunksize = 100,
    dtype = {
```

```
            'id_originale': Integer,
            'dc_identifier': String(40),
            'nome': String(50),
            'cognome': String(50)
        }
    )

query_27 = """create view match_esatti as (
 select id_originale, dc_identifier
from finale
where assegnazione='si')"""
res_27 = engine.execute(query_27)
query_28 = """create table match_finali_def
    as(
 select dati_miur.*, match_esatti.
    dc_identifier as dc_identifier
from dati_miur left join match_esatti on
    dati_miur.id = match_esatti.id_originale
where dati_miur.anno = %s and dati_miur.
    ssd = %s) """
res_28 = engine.execute(query_28, [int(anno),
     settore])
query_34 = """create table match_finali as(
```

```
select match_finali_def.*, ambigui.
    dc_identifier as ambiguo
from match_finali_def left join ambigui on
    match_finali_def.id = ambigui.
    id_originale)"""
res_34 = engine.execute(query_34)
query_35 = """alter table match_finali
add column dc_identifier_assegnato varchar
    (30)"""
res_35 = engine.execute(query_35)
query_36 = """update match_finali
set dc_identifier_assegnato = dc_identifier
    where dc_identifier is not null"""
res_36 = engine.execute(query_36)
query_37 = """update match_finali
set dc_identifier_assegnato = 'ambiguo'
    where ambiguo is not null"""
res_37 = engine.execute(query_37)
query_38 = """update match_finali
set dc_identifier_assegnato = 'not exists'
    where dc_identifier is null and ambiguo
    is null"""
res_38 = engine.execute(query_38)
query_39 = """alter table match_finali
```

```
drop column ambiguo"""
res_39 = engine.execute(query_39)
query_40 = """alter table match_finali
drop column dc_identifier"""
res_40 = engine.execute(query_40)
query_41 = """select distinct *
from match_finali"""
res_41 = engine.execute(query_41)
match_finali_definitivi = pd.DataFrame(
    res_41.fetchall())
match_finali_definitivi.to_sql(
    'match_finali_definitivi',
    engine,
    if_exists = 'append',
    index = False,
    chunksize = 100,
    dtype = {
        'id': Integer,
        'nome': String(50),
        'cognome': String(50),
        'fascia': String(225),
        'genere': String(1),
        'ateneo': String(100),
        'facolt ': String(225),
```

```
            'ssd': String(20),
            'sc': String(10),
            'anno': Integer,
            'is_duplicate': String(10),
            'dc_identifier_assegnato': String(40)
        })
query_29 = """drop view match_esatti"""
res_29 = engine.execute(query_29)
query_30 = """drop view temp"""
res_30 = engine.execute(query_30)
query_31 = """drop table finale """
res_31 = engine.execute(query_31)
query_32 = """drop table subjects """
res_32 = engine.execute(query_32)
query_33 = """drop table subj_weights"""
res_33 = engine.execute(query_33)
query_42 = """drop table match_finali_def"""
res_42 = engine.execute(query_42)
query_43 = """drop table match_finali"""
res_43 = engine.execute(query_43)


####CONFRONTO DATI REALI
```

```
query_confronto = """create table confronto_actual as
    (
with temp as(select a.*, b.au_identifier, count(a.
    dc_identifier_assegnato) as tot
from scopus.match_finali_definitivi a right join
scopus.aut_ministeriale b on a.nome=b.nome and a.
    cognome=b.cognome
and a.ssd = b.ssd
group by a.ssd, a.nome, a.cognome)
 select a.*, b.au_identifier, count(a.
    dc_identifier_assegnato)/tot as rate
from scopus.match_finali_definitivi a, temp b
        where a.nome = b.nome and a.cognome = b.
            cognome
        group by a.nome, a.cognome, a.
            dc_identifier_assegnato
        having rate > %s)"""
res_confronto = engine.execute(query_confronto, float
    (parametro_rate))


query_add_column = """alter table confronto_actual
        add column confronto int"""
res_add_column = engine.execute(query_add_column
    )
```

```
query_match = """update confronto_actual
        set confronto = 1 where
            dc_identifier_assegnato = au_identifier
            """
res_match = engine.execute(query_match)
query_unmatched = """alter table confronto_actual
        add column confronto_sbagliato int"""
res_unmatched = engine.execute(query_unmatched)
query_unmatched = """update confronto_actual
        set confronto_sbagliato = 1 where
            dc_identifier_assegnato != au_identifier
            and
        dc_identifier_assegnato!='ambiguo' and
            dc_identifier_assegnato != 'not exists'"""
res_unmatched = engine.execute(query_unmatched)
query_add_column = """alter table confronto_actual
        add column ambiguo int"""
res_add_column = engine.execute(query_add_column
    )
query_unmatched = """update confronto_actual
        set ambiguo = 1 where
            dc_identifier_assegnato ='ambiguo'"""
res_unmatched = engine.execute(query_unmatched)
```

```
query_tabella = """SELECT ssd, sum(confronto) as
    esatti, sum(confronto_sbagliato) as sbagliati, sum(
    ambiguo) as ambigui,
        sum(confronto)/(sum(confronto)+sum(
            confronto_sbagliato)) as accuracy
        FROM scopus.confronto_actual
        group by ssd"""
res_tabella = engine.execute(query_tabella)
tab = pd.DataFrame(res_tabella.fetchall())
tab['relSubj'] = subject_rate
tab['edNome'] = parametro_nome
tab['edCognome'] = parametro_cognome
tab['steepnessSigmoid'] = sigmoide
tab['alpha'] = parametro_alpha
tab['beta'] = parametro_beta
tab['rate'] = parametro_rate

query_drop = """drop table confronto_actual"""
res_drop = engine.execute(query_drop)
tab.reindex(columns = ['relSubj', 'edNome','
    edCognome','steepnessSigmoid','alpha', 'beta', '
    rate', 'ssd', 'esatti', 'sbagliati','ambigui','
    accuracy'])
tab.to_sql(
```

```
                    'tabella_parametri',
                    engine,
                    if_exists = 'append',
                    index = False,
                    chunksize = 100,
                    dtype = {
                        'relSubj': Float,
                        'edNome': Float,
                        'edCognome': Float,
                        'steepnessSigmoid': Float,
                        'alpha': Float,
                        'beta': Float,
                        'rate': Float,
                        'esatti': Integer,
                        'sbagliati': Integer,
                        'ambigui': Integer,
                        'accuracy': Float
                    })
query_17 = """drop table match_finali_definitivi"""
res_17 = engine.execute(query_17)
query_17 = """drop table ambigui"""
res_17 = engine.execute(query_17)
query_17 = """drop table df_scores_finali"""
res_17 = engine.execute(query_17)
```

Appendice B

Codice Python per l'estrazione dei dati da Scopus e creazione del database

```
import pandas as pd
import requests
import time
import json
import re
import numpy as np
import sqlalchemy as sql
from sqlalchemy.types import Integer, String, Float, Date, Boolean
```

```
#########extract values

def extract_values(obj, key):

    """Pull all values of specified key from nested JSON."""

    arr = []


    def extract(obj, arr, key):

        """Recursively search for values of key in JSON tree."""

        if isinstance(obj, dict):

            for k, v in obj.items():

                if isinstance(v, (dict, list)):

                    extract(v, arr, key)

                elif k == key:

                    arr.append(v)

        elif isinstance(obj, list):

            for item in obj:

                extract(item, arr, key)

        return arr


    results = extract(obj, arr, key)

    return results

###########


engine = sql.create_engine('mysql://root:Teabreak%4097@127.0.0.1/dati_scopus')

connection = engine.connect()
```

```
metadata = sql.MetaData()

dati= """

    select *

    from aut_ministeriale

    where n_autore >= 308

    """

res = engine.execute(dati)

aut = pd.DataFrame(res.fetchall(), columns = res.keys())

####################

for j,i in zip(aut['au_identifier'], aut['n_autore']):

    check_progressivo = """select au_identifier

    from autori_progressivo_1

    where au_identifier=%s"""

    res_progressivo = connection.execute(check_progressivo, j)

    if len(pd.DataFrame(res_progressivo.fetchall()))==0:

        query_insert = """INSERT INTO autori_progressivo_1

                                        VALUES (%s,%s);"""

        connection.execute(query_insert, [(j,i)])

    tot_papers = 1

    n_papers = 0

    while tot_papers > n_papers:

            req_ref = requests.get(f"http://api.elsevier.com/content/search/

                scopus?view=complete&query=au-id%28{int(j)}%29&start={

                n_papers}",
```

```
                    headers = {
                        'Accept': 'application/json',
                        'X-ELS-APIKey': '907
                            dbb591095128d9e05bdac0bde295e',
                        'X-ELS-insttoken': '22126
                            a7d7202e035c6550398cfeb8d74'
                    }
                    )
req_ref.encoding='utf-8'
req_ref.encoding.encode('utf-8', errors = 'replace')
data_doc = req_ref.json()
ndoc = data_doc['search-results']['opensearch:itemsPerPage']
temp_doc = data_doc['search-results']['entry']
n_papers += int(ndoc)
tot_papers = int(extract_values(data_doc, 'opensearch:totalResults')
    [0])
for doc in range(len(temp_doc)):
    id_doc = pd.DataFrame(extract_values(temp_doc[doc], 'dc:
        identifier'), columns = ['id_documento'])
    id_doc.iat[0, 0] = re.split(':', id_doc.iat[0, 0])[1]
    query_check = """SELECT id_documento, is_reference
            FROM papers_1
            WHERE id_documento = %s"""
    res = connection.execute(query_check, int(id_doc.iat[0,0]))
```

```
check_1 = pd.DataFrame(res.fetchall(), columns = res.keys())
if len(check_1) != 0 and check_1.at[0, 'is_reference'] == 0:
    continue
if len(check_1)== 0:
    if 'dc: title ' in temp_doc[doc]:
        name = pd.DataFrame(extract_values(temp_doc[doc], 'dc:
            title'), columns = ['titolo'])
    else:
        name = pd.DataFrame([None], columns = ['titolo'])
    if 'prism:coverDate' in temp_doc[doc]:
        date = pd.DataFrame(extract_values(temp_doc[doc], '
            prism:coverDate'), columns = ['data'])
    else:
        date = pd.DataFrame([None], columns = ['data'])
    if 'prism:publicationName' in temp_doc[doc]:
        pubName = pd.DataFrame(extract_values(temp_doc[doc],
            'prism:publicationName'), columns =['
            publicationName'])
    else:
        pubName = pd.DataFrame([None],
                               columns = ['publicationName']
                               )
    if 'prism:aggregationType' in temp_doc[doc]:
        source_type = pd.DataFrame(extract_values(temp_doc[
```

```
            doc], 'prism:aggregationType'), columns=['sourceType'
            ])
    else:
        source_type = pd.DataFrame([None], columns = ['
            sourceType'])
    if 'subtypeDescription' in temp_doc[doc]:
        subType = pd.DataFrame(extract_values(temp_doc[doc],
            'subtypeDescription'), columns = ['sourceSubType'])
    else:
        subType = pd.DataFrame([None], columns = ['
            sourceSubType'])
    if 'citedby-count' in temp_doc[doc]:
        citation  = pd.DataFrame(extract_values(temp_doc[doc],
            'citedby-count'), columns = ['n_citation'])
    else:
        citation  = pd.DataFrame([None], columns = ['n_citation'
            ])
    ref  = pd.DataFrame([0], columns = ['is_reference'])
    nuovo_paper = pd.concat([id_doc, name, date, pubName,
        source_type, subType, citation, ref],
                                axis  = 1
                                )
    nuovo_paper.to_sql(
        'papers_1',
```

```
        engine,
        if_exists = 'append',
        index = False,
        chunksize = 100,
        dtype = {
            'id_documento': String(50),
            'titolo': String(225),
            'data': Date,
            'publicationName': String(225),
            'sourceType': String(50),
            'sourceSubType': String(50),
            'n_citation': Integer,
            'is_reference': Integer
        }
    )
    auth_doc = temp_doc[doc]['author']
    for auth in range(len(auth_doc)):
        au_autore = extract_values(auth_doc[auth], 'authid')
        query_check = """SELECT *
                                    FROM aut_scopus_1
                                    WHERE dc_identifier = %s
                                        """
        res = engine.execute(query_check, au_autore)
        check = pd.DataFrame(res.fetchall())
```

```
if len(check) != 0:
    pass
else:
    au_autore = extract_values(auth_doc[auth], 'authid')
    nome_autore = extract_values(auth_doc[auth], 'given
        -name')
    cognome_autore = extract_values(auth_doc[auth], '
        surname')
    if 'afid' in auth_doc[auth]:
        affil = auth_doc[auth]['afid'][0]
        affiliation_autore = extract_values(affil, '$')
    else:
        affiliation_autore = [None]

    values = (au_autore, cognome_autore, nome_autore,
        affiliation_autore)
    query_insert = """INSERT INTO aut_scopus_1 (
        dc_identifier, surname, name, affiliation_id)
                                VALUES (%s, %s, %s, %s);"""
    connection.execute(query_insert, [values])

author = pd.DataFrame(extract_values(auth_doc[auth], '
    authid'), columns = ['id_autore'])
nuovo_autore = pd.concat([author, id_doc], axis = 1)
```

```
        nuovo_autore.to_sql(
        'authorship_1',
        engine,
        if_exists = 'append',
        index = False,
        chunksize = 100,
        dtype = {
            'id_autore': String(50),
            'id_documento': String(50)}
        )
    cit_req = requests.get(
        f"https://api.elsevier.com/content/abstract/citations?
            scopus_id={id_doc.iat[0,0]}&date={int(re.split('-',
            date.iat[0,0])[0]    )}-2022",
        headers = {
            'Accept': 'application/json',
            'X-ELS-APIKey': '907
                dbb591095128d9e05bdac0bde295e',
            'X-ELS-insttoken': '22126
                a7d7202e035c6550398cfeb8d74'
        }
        )
    data_cit = cit_req.json()
    citation_matrix = pd.DataFrame(columns = ['id_documento',
```

```
        'anno', 'n_citations'])
    cit_1 = pd.DataFrame(columns = ['id_documento', 'anno', '
        n_citations'], index = range(1))
    matrix = data_cit['abstract-citations-response']['
        citeColumnTotalXML']['citeCountHeader']
    if type(matrix['columnHeading']) == str:
        cit_1.at[0, 'id_documento'] = id_doc.iat[0, 0]
        cit_1.at[0, 'anno'] = int(matrix['columnHeading'])
        cit_1.at[0, 'n_citations'] = int(matrix['columnTotal'])
        citation_matrix = pd.concat([citation_matrix, cit_1])
        citation_matrix.reset_index()
        citation_matrix.to_sql(
            'citations_1',
            engine,
            if_exists = 'append',
            index = False,
            chunksize = 100,
            dtype = {
                'id_documento': String(50),
                'anno': Integer,
                'n_citations': Integer
            }
        )
    else:
```

```
            for i in range(len(matrix['columnHeading'])):
                cit_1.at[0, 'id_documento'] = id_doc.iat[0,0]
                cit_1.at[0, 'anno'] = int(extract_values(matrix['
                    columnHeading'][i], '$')[0])
                cit_1.at[0, 'n_citations'] = int(extract_values(
                    matrix['columnTotal'][i], '$')[0])
                citation_matrix = pd.concat([citation_matrix, cit_1])
                citation_matrix.reset_index()
            citation_matrix.to_sql(
                'citations_1',
                engine,
                if_exists = 'append',
                index = False,
                chunksize = 100,
                dtype ={
                    'id_documento': String(50),
                    'anno': Integer,
                    'n_citations': Integer
                }
            )

        if len(check_1) != 0 and check_1.at[0, 'is_reference'] == 1:
            if 'prism:coverDate' in temp_doc[doc]:
                date = pd.DataFrame(extract_values(temp_doc[doc], '
```

```
            prism:coverDate'), columns = ['data'])
    else:
        date = pd.DataFrame([None], columns = ['data'])
    if 'prism:publicationName' in temp_doc[doc]:
        pubName = pd.DataFrame(extract_values(temp_doc[doc],
             'prism:publicationName'), columns =['
            publicationName'])
    else:
        pubName = pd.DataFrame([None],
                               columns = ['publicationName']
                               )
    if 'prism:aggregationType' in temp_doc[doc]:
        source_type = pd.DataFrame(extract_values(temp_doc[
            doc], 'prism:aggregationType'), columns=['sourceType'
            ])
    else:
        source_type = pd.DataFrame([None], columns = ['
            sourceType'])
    if 'subtypeDescription' in temp_doc[doc]:
        subType = pd.DataFrame(extract_values(temp_doc[doc],
            'subtypeDescription'), columns = ['sourceSubType'])
    else:
        subType = pd.DataFrame([None], columns = ['
            sourceSubType'])
```

```
query_update = """update papers_1
set publicationName = %s, sourceType = %s, sourceSubType =
    %s, is_reference = 0
where id_documento = %s"""
res_update = engine.execute(query_update, [pubName.iat
    [0,0], source_type.iat[0,0],
                                        subType.iat[0,0],
                                            id_doc.iat
                                            [0,0]])
cit_req = requests.get(
    f"https://api.elsevier.com/content/abstract/citations?
        scopus_id={id_doc.iat[0, 0]}&date={int(re.split('-',
        date.iat[0, 0])[0])}-2022",
    headers = {
        'Accept': 'application/json',
        'X-ELS-APIKey': '907
            dbb591095128d9e05bdac0bde295e',
        'X-ELS-insttoken': '22126
            a7d7202e035c6550398cfeb8d74'
    }
)
data_cit = cit_req.json()
citation_matrix = pd.DataFrame(columns = ['id_documento',
    'anno', 'n_citations'])
```

```
cit_1 = pd.DataFrame(columns = ['id_documento', 'anno', '
    n_citations'], index = range(1))
matrix = data_cit['abstract-citations-response']['
    citeColumnTotalXML']['citeCountHeader']
if type(matrix['columnHeading']) == str:
    cit_1.at[0, 'id_documento'] = id_doc.iat[0, 0]
    cit_1.at[0, 'anno'] = int(matrix['columnHeading'])
    cit_1.at[0, 'n_citations'] = int(matrix['columnTotal'])
    citation_matrix = pd.concat([citation_matrix, cit_1])
    citation_matrix.reset_index()
    citation_matrix.to_sql(
        'citations_1',
        engine,
        if_exists = 'append',
        index = False,
        chunksize = 100,
        dtype = {
            'id_documento': String(50),
            'anno': Integer,
            'n_citations': Integer
        }
    )
else:
    for i in range(len(matrix['columnHeading'])):
```

```
            cit_1.at[0, 'id_documento'] = id_doc.iat[0, 0]
            cit_1.at[0, 'anno'] = int(extract_values(matrix['
                columnHeading'][i], '$')[0])
            cit_1.at[0, 'n_citations'] = int(extract_values(
                matrix['columnTotal'][i], '$')[0])
            citation_matrix = pd.concat([citation_matrix, cit_1])
            citation_matrix.reset_index()
        citation_matrix.to_sql(
            'citations_1',
            engine,
            if_exists = 'append',
            index = False,
            chunksize = 100,
            dtype = {
                'id_documento': String(50),
                'anno': Integer,
                'n_citations': Integer
            }
        )
```

Bibliografia

[1] *DM 08/08/2018 n.589 Definizione valori - soglia degli indicatori di impatto della produzione scientifica.*

[2] *LEGGE 30 dicembre 2010, n. 240 (in G.U. n. 10 del 14 gennaio 2011 - Suppl. Ord. n.11 - in vigore dal 29 gennaio 2011) - Norme in materia di organizzazione delle università, di personale accademico e reclutamento, nonché delega al Governo per incentivare la qualità e l'efficienza del sistema universitario.*

[3] Marco Seeber et al. "Self-citations as strategic response to the use of metrics for career decisions". In: *Research Policy* 48.2 (2019). Academic Misconduct, Misrepresentation, and Gaming, pp. 478–491. ISSN: 0048-7333. DOI: https://doi.org/10.1016/j.respol.2017.12.004. URL: https://www.sciencedirect.com/science/article/pii/S004873331730210X.