Università degli Studi di Padova
Dipartimento di Scienze Statistiche
Corso di Laurea Triennale in

Statistica per le Tecnologie e le Scienze

Analisi statistica delle relazioni tra le lingue

Relatore: Prof.ssa Manuela Cattelan
Dipartimento di Scienze Statistiche

Laureando: Giorgio Nagy
Matricola n. 1231405

Anno Accademico 2021/2022

Indice

Introduzione

La linguistica è lo studio scientifico del linguaggio verbale umano e delle sue strutture [19].
Gli esseri umani sono da sempre stati affascinati dalla molteplicità di lingue che esistono nel mondo, domandandosi il perché della loro diversità, somiglianza e talvolta stranezza. Oggi la linguistica insegna che certe lingue presentano caratteristiche in comune perché appartengono alla medesima famiglia, ovvero discendono da un antenato comune, o perché sono geograficamente vicine.
La presente tesi si pone l'obiettivo di studiare le relazioni tra le lingue del mondo con un approccio quantitativo e statistico, indagando se sia ragionevolmente possibile classificarle a partire dai loro attributi.
Per fare questo ci si è avvalsi di due risorse: WALS e PHOIBLE. Si tratta di due progetti messi a disposizione dall'Istituto Max Planck per la scienza della storia umana.
Il *Word Atlas of Language Structures* (WALS) è un esteso database di proprietà strutturali (fonologiche, grammaticali e lessicali) di lingue raccolte da materiali descrittivi come, ad esempio, grammatiche di riferimento. Contiene informazioni relative a 2662 lingue del mondo e riguardo a 192 caratteristiche, dove ciascuna caratteristica è una variabile qualitativa che può assumere due o più valori [15].
PHOIBLE è una raccolta di dati che contiene 3020 sistemi fonologici provenienti da 2186 lingue distinte [32]. Un sistema fonologico è costituito dai fonemi posseduti da una lingua, dove un fonema è un'unità di suono in grado di distinguere due parole in una particolare lingua [36].
La tesi è organizzata come segue. Nel Capitolo 1 vengono descritte le procedure di creazione e manipolazione dei datasets di riferimento. A seguire nel Capitolo 2 vengono condotte alcune analisi esplorative atte a sintetizzare i dati di partenza e ad indagare la loro natura e codifica. Successivamente nel Capitolo 3 vengono adattati dei metodi di *clustering*, col fine di verificare la ragionevolezza dei raggruppamenti che

si ottengono. In seguito viene presentata l'implementazione di alcune reti filogenetiche, ottenute partendo da un allineamento di sequenze. Si è fatto uso a questo proposito di *SplitsTree*, un programma messo a disposizione dall'Università di Tübingen che ha lo scopo di creare reti filogenetiche senza radice da sequenze di dati molecolari [25]. Questo tipo di grafi è generalmente utilizzato nell'ambito della genetica, e la presente tesi si domanda se sia ragionevole applicarli anche allo studio delle relazioni tra le lingue del mondo. Si procede infine alla costruzione di alcuni alberi di classificazione con l'obiettivo di determinare i fonemi in grado di discriminare maggiormente fra lingue appartenenti a famiglie ed aree linguistiche diverse. Viene inoltre valutata la potenza predittiva degli alberi di classificazione in questo contesto.

Per condurre le analisi dei dati necessarie ad ottenere risultati e grafici sono stati usati i seguenti software:

1. R (versione 4.1.3),
2. Python (versione 3.10.1),
3. SplitsTree4 (versione 4.18.3).

Il codice è riportato nell'Appendice.

Capitolo 1

Creazione dei datasets

In questo capitolo viene fornita una descrizione delle risorse sfruttate per ottenere i datasets necessari a condurre le analisi esposte nella presente tesi. Viene inoltre illustrata la struttura dei datasets elaborati e le variabili che li compongono. Sono stati considerati sia dati legati a proprietà strutturali, sia dati relativi a sistemi fonologici appartenenti a lingue di tutto il mondo.

1.1 Analisi delle caratteristiche nelle lingue

Con **caratteristica** si intende una proprietà strutturale astratta grazie alla quale una lingua può essere messa a confronto con altre lingue. Alcuni esempi di caratteristiche sono il numero di vocali presenti nella lingua, il modo in cui viene indicata la pluralità nei nomi, la presenza o meno di un tempo futuro nel sistema verbale.

1.1.1 WALS

I linguisti hanno a lungo lavorato con mappe che mostrano l'estensione geografica delle lingue nel mondo, e il primo atlante linguistico mondiale completo è stato divulgato nel 1994 [33, 1, 29]. Per oltre un secolo, hanno anche prodotto atlanti incentrati sulla distribuzione geografica di attributi linguistici nei dialetti di una lingua. La peculiarità del *World Atlas of Language Structures* (WALS) sta nel fatto che si tratta del primo atlante di caratteristiche che sia stato sviluppato su una scala mondiale. La sua prima pubblicazione avviene nel 2005 sotto forma di libro con CD-ROM annesso, da parte della Oxford University

Press. Originariamente contiene 160 mappe relative alla distribuzione geografica di proprietà linguistiche strutturali in tutto il mondo [9]. Mentre gli atlanti linguistici dialettali evidenziano la distribuzione sul territorio di attributi come certi suoni in comune, o certe parole (si veda ad esempio l'Atlante Linguistico Italiano [10]), WALS contiene solo proprietà astratte del sistema linguistico che hanno il vantaggio di poter essere confrontate fra ogni lingua del mondo. La branca della linguistica che studia le diverse lingue del mondo alla ricerca di fenomeni strutturali comuni è chiamata tipologia.
WALS fornisce un contributo significativo all'ambito della tipologia areale, un campo della linguistica che cerca di stabilire se particolari distribuzioni geografiche siano il risultato di contatto fra lingue contigue. Consta di 99 capitoli, ciascuno contenente una o più caratteristiche, per un totale di 192 caratteristiche strutturali.
WALS è una pubblicazione dell'Istituto Max Planck per la scienza della storia umana [15].

1.1.2 Datasets *caratteristiche* e *valori*

I datasets di WALS sono disponibili in una repository GitHub all'indirizzo *https://github.com/cldf-datasets/wals*. La versione più recente, e quella usata nella presente tesi, è quella del 7 luglio 2022. I dati sono in formato CLDF: si tratta di un format impiegato per lo scambio di dati interlinguistici. I dati veri e propri sono contenuti in uno o più file in formato csv. In particolare si è interessati ai files languages.csv, values.csv, parameters.csv e codes.csv, reperibili all'interno della cartella *cldf*.
Una volta scaricati, questi quattro dataset possono essere manipolati tramite R [34]. Poiché i datasets languages.csv e codes.csv contengono caratteri speciali come rispettivamente *á, ä, ã, é* ed *í* per il nome di alcune lingue e ṽ: per il nome di alcune caratteristiche, diventa necessario caricarli in R specificando la codifica `encoding = "UTF-8"`, affinché tutti i caratteri siano letti correttamente.
Il dataset *valori* è costituito dalle prime 5 variabili di values.csv. Ha 76475 righe ed è un dataset in formato lungo. Il dataset *caratteristiche* è costruito a partire dai datasets languages.csv e *valori*. Ha 2660 righe e 202 colonne. Ogni riga corrisponde ad una lingua. Le sue prime 10 variabili sono una rielaborazione di languages.csv e contengono informazioni generali riguardo a ciascuna lingua. Le altre 192 variabili sono le vere e proprie caratteristiche strutturali e sono ottenute

trasformando *valori* da formato lungo in formato largo. Si è scelto di escludere le lingue Fiote e Jiarong in quanto non contenevano informazioni relative ad alcuna proprietà. Infine il nome delle variabili relative alle caratteristiche viene cambiato perché sia uguale al codice alfanumerico di WALS combinato con il nome esteso delle variabili (preso da parameters.csv), e parimenti i valori dei livelli delle variabili sono sostituiti con la loro descrizione completa (presa da codes.csv).

Variabili del dataset *valori*

La variabile `Parameter_ID` assegna un codice alfanumerico ad ognuna delle 192 possibili caratteristiche. La variabile `Language_ID` associa a ciascuna lingua il codice WALS corrispondente. `ID` funge da identificatore univoco per il dataset *valori*; è formata dalla combinazione delle variabili `Parameter_ID` e `Language_ID`, ed indica la presenza di una specifica caratteristica in una determinata lingua. La variabile `Value` fornisce il valore assunto dalla caratteristica, in quanto ogni caratteristica è un fattore a più livelli. Infine `Code_ID` è la combinazione di `Parameter_ID` e `Value`.

In Tabella 1.2 vengono descritte brevemente le variabili del dataset *valori*.

#	Nome	Descrizione
1	`ID`	Combinazione di `Parameter_ID` e `Language_ID`
2	`Language_ID`	Codice WALS della lingua
3	`Parameter_ID`	Codice della caratteristica
4	`Value`	Valore assunto dalla caratteristica
5	`Code_ID`	Combinazione di `Parameter_ID` e `Value`

Tabella 1.1: Descrizione delle variabili presenti nel dataset *valori*.

Variabili del dataset *caratteristiche*

Ciascuna lingua è identificata tramite un codice di tre lettere chiamato `Wals_code`, creato appositamente per questa applicazione. Sono in-

clusi anche altri due sistemi per classificazione dei linguaggi: il codice ISO 639-3 (`Iso_code`) e il codice Glottolog (`Glottocode`). Si tratta di standard internazionali che associano a ciascun linguaggio un codice identificativo unico, ma che in questo dataset possiedono dei valori mancanti [20].
Molte lingue sono conosciute in letteratura sotto nomi differenti. La variabile `Nome` indica il nome con il quale la lingua è attualmente conosciuta, poiché i nomi più vecchi sono spesso considerati offensivi dalle comunità dove si parla la lingua. `Latitudine` e `Longitudine` forniscono le coordinate geografiche che determinano l'ubicazione della lingua. Solitamente si tratta della capitale o del centro abitato più grande in cui è parlata la lingua. La variabile `Famiglia` indica la famiglia linguistica di appartenenza, dove due lingue appartengono alla stessa famiglia linguistica se sono derivanti da un antenato in comune. Il `Genus` (anche detto ramo) è un gruppo di lingue filogeneticamente correlate all'interno di una famiglia linguistica. Ad esempio l'italiano fa parte del ramo Italico della famiglia delle lingue Indo-Europee, mentre il tedesco appartiene al ramo Germanico, l'irlandese al ramo Celtico ed il croato al ramo Slavico. La variabile `Macroarea` assume il valore di una delle sei macroregioni possibili: Eurasia, Africa, America del Nord, America del Sud, Australia e Papunesia. La Papunesia costituisce una regione per sé a causa della varietà e del numero elevato di lingue che vi sono parlate. `Nazione` mostra le nazioni in cui è parlata ogni lingua, mediante delle sigle formate da due lettere per nazione [9].
Per quanto riguarda le 192 variabili concernenti le caratteristiche strutturali, esse non sono altro che fattori a più livelli. Possono essere inoltre suddivise in alcune categorie. In questa sezione si farà riferimento alla variabile mediante il codice alfanumerico della caratteristica, evitando di scrivere il nome completo per brevità. Così ad esempio con `1A` si indicherà la variabile `1A.Consonant.Inventories`. Dunque, le variabili da `1A` a `19A` sono perlopiù caratteristiche relative alla fonologia. Da `20A` a `25B` troviamo caratteristiche riguardanti la morfologia sintattica, e cioè atte a descrivere la posizione in cui si collocano solitamente i morfemi in una frase. Da `26A` a `63A` vi sono caratteristiche grammaticali di nomi ed aggettivi, mentre da `64A` a `73A` vi sono le caratteristiche grammaticali del sistema verbale. Le variabili da `74A` a `100A` hanno a che vedere con l'ordine degli elementi della frase e con l'allineamento morfosintattico. Da `101A` a `128A` trovano posto altre caratteristiche grammaticali, mentre da `129A` a `138A` vi sono caratteristiche lessicali. Infine da `139A` a `144Y` si possono osservare caratteri-

stiche che descrivono come sono formate le negazioni nella lingua [15]. In Tabella 1.1 vengono descritte brevemente le variabili del dataset *caratteristiche*.

#	Nome	Descrizione
1	`Wals_code`	Codice assegnato da WALS alla lingua
2	`Iso_code`	Codice ISO 639-3
3	`Glottocode`	Codice Glottolog
4	`Nome`	Nome (in inglese) della lingua
5	`Latitudine`	Latitudine della lingua
6	`Longitudine`	Longitudine della lingua
7	`Genus`	Ramo della famiglia linguistica
8	`Famiglia`	Famiglia linguistica di appartenenza
9	`Macroarea`	Regione di appartenenza
10	`Nazione`	Nazione di appartenenza
11	`1A`	Caratteristiche strutturali
...	`...`	
192	`144Y`	

Tabella 1.2: Descrizione delle variabili presenti nel dataset *caratteristiche*.

1.2 Analisi dei fonemi nelle lingue

Con **fonema** si intende un'unità di suono dotata di valore distintivo, ossia una unità che può produrre variazioni di significato se scambiata con un'altra unità [36]. Ne è un esempio la serie **p**are, **b**are, **t**are, **d**are, **c**are, **g**are. Due parole che si differenziano per un solo fonema costituiscono una coppia minima. L'insieme dei fonemi di una lingua costituisce il suo sistema fonologico; in altre parole si tratta di tutti i suoi suoni che possono provocare cambiamenti di significato.
In una lingua ad un singolo fonema possono corrispondere più foni, ovvero più realizzazioni specifiche diverse. Un fonema infatti è un'entità

astratta che viene espressa concretamente da un fono. Foni e fonemi sono solitamente rappresentati tramite l'Alfabeto Fonetico Internazionale (o IPA dall'inglese International Phonetic Alphabet), ed è pratica comune scrivere i foni fra parentesi quadre [ ] e i fonemi fra barre oblique // [2]. In italiano, si considerino i foni [r], [ʀ] e [ʁ]. La prima è una vibrante alveolare (la variante comunemente incontrata in italiano), la seconda è una vibrante uvulare (la cosiddetta erre moscia), la terza è una fricativa uvulare sonora (la cosiddetta erre francese). Tutte e tre vengono articolate in modi diversi dal parlante, hanno caratteristiche fisiche differenti e vengono percepite diversamente da chi ascolta; tuttavia, in italiano corrispondono al fonema /r/. Si dice allora che [r], [ʀ] e [ʁ] sono allofoni del fonema /r/, in quanto [rana], [ʀana] e [ʁana] sono tutte pronunce accettabili della parola *rana* [36]. Al contrario in una lingua come l'armeno usare /ʁ/ al posto di /r/ può cambiare completamente il significato della parola [16]. Per esempio *taṙ*, che significa *lettera*, viene pronunciato /tɑr/ con la erre italiana, mentre *tał*, che significa *poema*, viene pronunciato /tɑʁ/ con la erre francese. Quelle che per un parlante italiano sono realizzazioni diverse dello stesso suono, per un parlante armeno costituiscono due fonemi diversi. In armeno quindi *taṙ* e *tał* formano una coppia minima.
Si noti come l'interesse dell'analisi dei fonemi sia la pronuncia delle parole e non il modo in cui vengono scritte secondo l'ortografia della specifica lingua. Infatti, lo stesso fonema può equivalere a simboli diversi o viceversa lo stesso simbolo può equivalere a fonemi diversi. Ad esempio nelle parole italiane **c**ane e **ch**icco simboli diversi hanno la stessa pronuncia, mentre in **c**ane e **c**esto lo stesso simbolo ha pronunce diverse.
D'ora in poi ci si occuperà solo dei fonemi delle lingue, e non dei loro allofoni.

1.2.1 PHOIBLE

PHOIBLE è una base di dati contenente sistemi fonologici e tratti distintivi che sono stati estratti e compilati da fonti primarie e altre basi di dati. Attualmente la versione 2.0 del 2019 conta di 3020 sistemi che includono 3183 unità provenienti da 2186 lingue distinte. PHOIBLE contiene informazioni anche riguardo agli allofoni dei fonemi in molte lingue, ma questi non saranno oggetto di analisi nella presente tesi.
Alcune lingue in PHOIBLE hanno più di una rappresentazione poiché fonti diverse possono non essere d'accordo sul numero e/o l'identità

dei fonemi della lingua. I due principi che guidano lo sviluppo di PHOIBLE sono:

1. essere fedeli alla descrizione della lingua nel documento fonte,
2. rappresentare tutti i dati in maniera coerente tramite la codifica dell'Alfabeto Fonetico Internazionale in Unicode (uno standard per la codifica della maggior parte dei sistemi di scrittura nel mondo).

PHOIBLE è una pubblicazione dell'Istituto Max Planck per la scienza della storia umana [32].

1.2.2 Il dataset *fonemi*

I datasets di PHOIBLE sono disponibili in una repository GitHub all'indirizzo *https://github.com/phoible/dev*. Nello specifico si è interessati al file phoible.csv, reperibile all'interno della cartella *data*. Per questioni di efficienza lo si è scaricato con estensione .txt.
Analogamente a quanto visto prima, anche phoible.txt deve essere caricato su R tramite la codifica `encoding = "UTF-8"`, in quanto contiene caratteri dell'Alfabeto Fonetico Internazionale che sarebbero altrimenti letti in maniera errata [34].
phoible.csv è un dataset in formato lungo, con 105488 righe e 48 colonne. Le uniche sue variabili di interesse per le analisi condotte nella presente tesi sono `InventoryID` e `Glottocode`. La variabile `InventoryID` fornisce un codice diverso per ogni sistema fonologico distinto mentre `Glottocode` è il codice Glottolog [20]. Grazie al codice Glottolog siamo in grado di collegare questi dati alle informazioni del dataset *caratteristiche*. Innanzitutto si procede col rimuovere tutti i sistemi fonologici con `Glottocode` non univoco (si rimane quindi con un sistema fonologico per lingua) e tutti i fonemi che non siano fra i 100 più comuni a livello mondiale. A questo punto si trasforma il dataset ottenuto in formato largo, e si rimuovono tutte le lingue con `Glottocode` non corrispondente alla variabile omonima del dataset *caratteristiche*. Nell'eventualità che avvenga il contrario, vale a dire che *caratteristiche* abbia delle lingue con `Glottocode` duplicati (può succedere se sono state rilevate varianti o dialetti della medesima lingua), viene mantenuta solamente la lingua con meno valori mancanti. In questo modo si è ottenuto il dataset *fonemi*: esso ha 1325 righe e 101 colonne. Le unità statistiche sono tutte le lingue con `Glottocode`

presente nella lista dei codici Glottolog della variabile omonima del dataset *caratteristiche*. Possiede poi una variabile `Nome` contenente i nomi delle lingue, e 100 variabili che rappresentano i 100 fonemi più comuni a livello mondiale. Ognuna di queste variabili è un fattore che assume valore 1 se un fonema fa parte del sistema fonologico di una lingua e 0 altrimenti.

Variabili del dataset *fonemi*

La variabile `Nome` in *fonemi* assume gli stessi valori della variabile `Nome` del dataset *caratteristiche*. I nomi delle 100 variabili relative alla presenza dei fonemi sono caratteri dell'Alfabeto Fonetico Internazionale, e rappresentano ciascuno un fonema. Ognuna di queste variabili è una variabile dicotomica che assume valore 1 se il fonema è presente nella lingua e 0 se è assente. In altre parole ciascuna casella può essere considerata come una variabile dicotomica Y_{ij} tale che:

$$Y_{ij} = \begin{cases} 1, & \text{se il fonema } f_j \text{ è presente nella lingua } l_i, \\ 0, & \text{altrimenti.} \end{cases}$$

con $i = 1, ..., 1325$ e $j = 1, ..., 100$.

1.3 Unione dei datasets *caratteristiche* e *fonemi*

È possibile unire i datasets *caratteristiche* e *fonemi*, a patto di considerare solo le lingue presenti in entrambi i dataset. Le unità statistiche diminuiscono dalle 2660 di *caratteristiche* alle 1325 di *fonemi*. Le variabili totali diventano $202 + 100 = 302$. Il dataset così ottenuto ha dunque 1325 righe e 302 colonne, e da qui in poi verrà indicato con *unione*. L'utilità di questo dataset consiste nel fatto che permette di condurre analisi relative ai fonemi delle lingue, tenendo al contempo in considerazione i valori assunti da covariate come le variabili `Famiglia` o `Macroarea`.
Il dataset *famiglie* trae appunto vantaggio da queste informazioni; *famiglie* è infatti un sottoinsieme del dataset *fonemi*, costruito a partire dal dataset *unione*. Esso è costituito da 240 lingue appartenenti a 3 famiglie linguistiche diverse: 80 sono lingue indoeuropee (parlate

in un territorio che va dall'Europa all'India, e di cui fa parte l'italiano), 80 sono lingue niger-kordofaniane (parlate in Africa) e 80 sono lingue pama-nyunga (parlate in Australia). Per ciascuna lingua sono rilevate 100 variabili che corrispondono ai 100 fonemi più comuni a livello mondiale. Il dataset *famiglie* pertanto possiede 240 righe e 100 colonne.

Capitolo 2

Analisi esplorative

Nel corso di questo capitolo si vedranno alcune analisi esplorative dei datasets descritti nel capitolo precedente.

2.1 Distribuzione delle caratteristiche nel mondo

Come anticipato già nella sezione 1.1.1 della presente tesi, i dati procurati da WALS sono particolarmente utili per visualizzare il modo con cui le proprietà delle lingue si ripartiscono nel mondo. Per avere una visione d'insieme si è soliti rappresentare le informazioni su delle mappe. Questo è facilmente ottenibile disegnando un planisfero e marcando con dei pallini le lingue per le quali è rilevata una certa caratteristica. I pallini saranno poi colorati in base alle varie modalità assunte dalla variabile in questione.
Vi è una certa variabilità rispetto al grado con cui le caratteristiche esibiscono distribuzioni riconducibili a modelli areali. In certe mappe emergono chiaramente delle divisioni geografiche fra le lingue del mondo, mentre in altre la situazione può essere ben più complessa.
Quando si incontrano delle aree geografiche dove le lingue sono simili tra di loro, le possibili spiegazioni sono tre:

1. le somiglianze sono dovute al fatto che le lingue appartengono alla stessa famiglia linguistica, ovvero discendono tutte da uno stesso antenato comune. Questo è il caso per le lingue romanze (altrimenti dette neolatine). Italiano, spagnolo, francese, portoghese e romeno derivano dal latino e per questo sono **geneticamente correlate**. I tratti in comune sono perciò spiegabili come elementi ereditati congiuntamente dalle varie lingue della famiglia. Il

dataset *caratteristiche* contiene 180 famiglie linguistiche separate. Esistono anche lingue isolate per le quali non è dimostrata la parentela con altre lingue del mondo, come ad esempio la lingua basca;

2. le somiglianze sono emerse per via della prossimità geografica. Esistono numerosi casi di aree geografiche, anche relativamente piccole, come i Balcani o l'America Centrale, dove molte proprietà sono condivise a causa dello scambio avvenuto nel tempo fra le lingue. In questo contesto si parla di **area linguistica** e le lingue in questione possono essere anche geneticamente non imparentate, o esserlo solo lontanamente;

3. le somiglianze sono da ritenersi almeno parzialmente dovute al caso. Soprattutto quando si considerano caratteristiche con soltanto due livelli, è molto facile che si creino aree geografiche dove una delle due modalità predomina mentre l'altra sottosta. In altre parole due lingue possono condividere degli aspetti anche se non vi è nessun tipo di connessione fra di loro [9].

Seguono ora tre esempi di mappe con distribuzioni di caratteristiche, riguardanti variabili appartenenti al dataset *caratteristiche*. I grafici con le mappe sono stati ottenuti tramite le librerie di R *maps* [4] e *ggplot2* [40]. Nell'Appendice C della presente tesi è fornito il codice R per creare anche tre files html interattivi, relativi alle mappe esposte di seguito. Per ottenerli sono state usate le librerie *leaflet* [8] e *htmlwidgets* [39].

2.1.1 Ordine del complemento oggetto e del verbo

La mappa in Figura 2.1 mostra l'ordine dominante del complemento oggetto lessicale e del verbo. In pratica viene studiato l'ordine con cui si dispongono normalmente il complemento oggetto ed il verbo di una frase. Ad esempio nella frase *il cane insegue il gatto*, *cane* è il soggetto, *insegue* è il verbo e *gatto* è il complemento oggetto. L'italiano è una lingua di tipo VO (Verbo-Oggetto) poiché il verbo precede il soggetto, e infatti come si può vedere in Figura 2.1 è contrassegnata con un pallino azzurro. Al contrario il turco è una lingua di tipo OV (Oggetto-Verbo), in quanto il complemento oggetto solitamente precede il verbo, come nella frase *Mehmedi gördüm* , che significa *Ho visto Mehmet* , ed è contrassegnata con un pallino verde [38]. Esistono

però anche lingue con entrambi gli ordini e dove nessun ordine è dominante, come ad esempio la lingua tedesca, e che sono contrassegnate con un pallino rosso.
La Figura 2.1 evidenzia una chiara distribuzione dei due ordini di verbo e complemento oggetto fra le lingue del mondo. In Europa, in Africa settentrionale e nella penisola arabica la maggioranza delle lingue è di tipo VO, mentre in gran parte dell'Asia si riscontra il tipo OV. Tuttavia nel sud-est asiatico predomina nuovamente l'ordine VO, con l'eccezione della Papua Nuova Guinea dove si rilevano prevalentemente lingue con ordine OV. L'ordine VO prevale anche nell'Africa subsahariana, con alcuni raggruppamenti di lingue di tipo OV ad est e ovest. In America centrale e sulla costa orientale dell'America settentrionale si osservano lingue di tipo VO, mentre il resto dell'America settentrionale e meridionale preferisce l'ordine di tipo OV. Le lingue dove nessun ordine è dominante sono particolarmente comuni in Australia [14].
In Tabella 2.1 viene riportata la distribuzione di frequenza delle lingue per ciascun valore assunto dalla variabile *Ordine del complemento oggetto e del verbo*, in ordine decrescente.

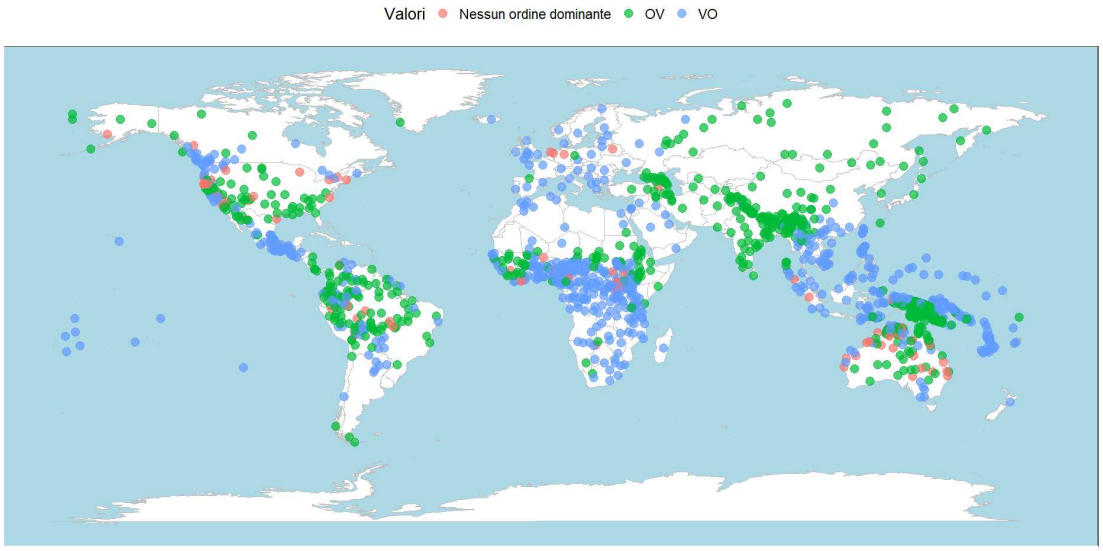

Figura 2.1: Mappa della distribuzione nel mondo dell'ordine del complemento oggetto e del verbo nella frase.

Ordine del complemento oggetto e del verbo	**Lingue**
OV	712
VO	705
Nessun ordine dominante	101

Tabella 2.1: Distribuzione di frequenza delle lingue per ciascun valore assunto dalla variabile *Ordine del complemento oggetto e del verbo*.

2.1.2 Articoli determinativi

La mappa in Figura 2.2 mostra informazioni relative agli articoli determinativi. Un articolo determinativo è un morfema che accompagna nomi e che codifica determinatezza e specificità. Normalmente precede il nome a cui si riferisce, ed è questo il caso dell'inglese *the* e dell'italiano *il*. Gran parte delle lingue in Europa occidentale, America centrale, Africa equatoriale e nell'Oceano Pacifico sfruttano questo sistema. Un'altra possibile soluzione è l'uso di un suffisso determinativo, come ad esempio in romeno, albanese o nelle lingue scandinave. Alcune lingue, soprattutto negli Stati Uniti, usano aggettivi dimostrativi come *questo* o *quello* in funzione di articoli determinativi. Ciononostante la maggior parte delle lingue del mondo non possiede nessun articolo determinativo o indeterminativo, come ad esempio le lingue slave. Lingue di questo tipo si concentrano soprattutto in Asia e America meridionale. Infine in una striscia che va dalla Turchia all'India si possono trovare lingue dove sono presenti articoli indeterminativi come *un* ma non articoli determinativi [13].
In Tabella 2.2 viene riportata la distribuzione di frequenza delle lingue per ciascun valore assunto dalla variabile *Articoli determinativi*, in ordine decrescente.

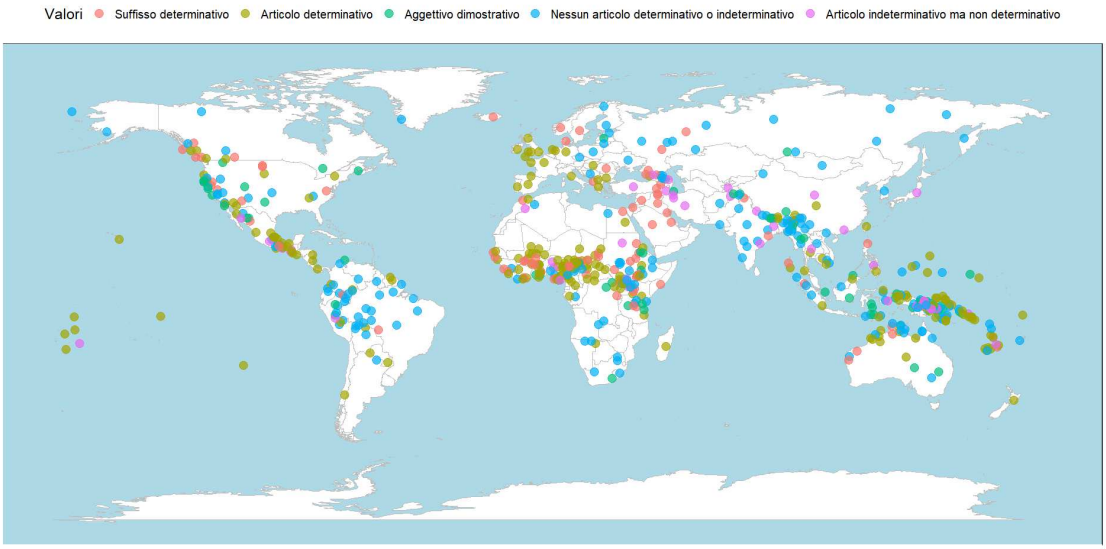

Figura 2.2: Mappa della distribuzione nel mondo degli articoli determinativi.

Articoli determinativi	**Lingue**
Articolo determinativo	216
Nessun articolo determinativo o indeterminativo	198
Suffisso determinativo	92
Aggettivo dimostrativo	69
Articolo indeterminativo ma non determinativo	45

Tabella 2.2: Distribuzione di frequenza delle lingue per ciascun valore assunto dalla variabile *Articoli determinativi*.

2.1.3 Codifica dei plurali nei nomi

La mappa in Figura 2.3 mostra i metodi con cui le lingue del mondo indicano la pluralità dei nomi. In sostanza questo è ottenuto in due modi; il primo (e il più comune) consiste nell'esplicitare la pluralità cambiando la forma morfologica del nome. Ciò si esprime tramite un suffisso in italiano e nella prevalenza delle lingue dell'Eurasia, dell'A-

merica e dell'Australia: singolare *cavall-o*, plurale *cavall-i* Altre soluzioni per segnalare il plurale nei nomi sono l'uso di prefissi (comune nelle lingue Bantu dell'Africa meridionale), il mutamento della radice del nome, l'uso di un tono diverso, il raddoppiamento della parola o un misto fra due o tre di queste soluzioni.
Il secondo modo consiste nell'esplicitare la pluralità mediante una parola separata che compare altrove nella frase. Ad esempio in lingua hawaiana la parola *mau* svolge la stessa funzione del plurale in italiano, ma è una parola a sé stante che modifica il nome e che può comparire dovunque all'interno della frase. Dunque in *lua mau i'a* (*due pesci*), *lua* significa *due*, *mau* mostra la pluralità e *i'a* significa *pesce* [12].
In Tabella 2.3 viene riportata la distribuzione di frequenza delle lingue per ciascun valore assunto dalla variabile *Codifica dei plurali nei nomi*, in ordine decrescente.

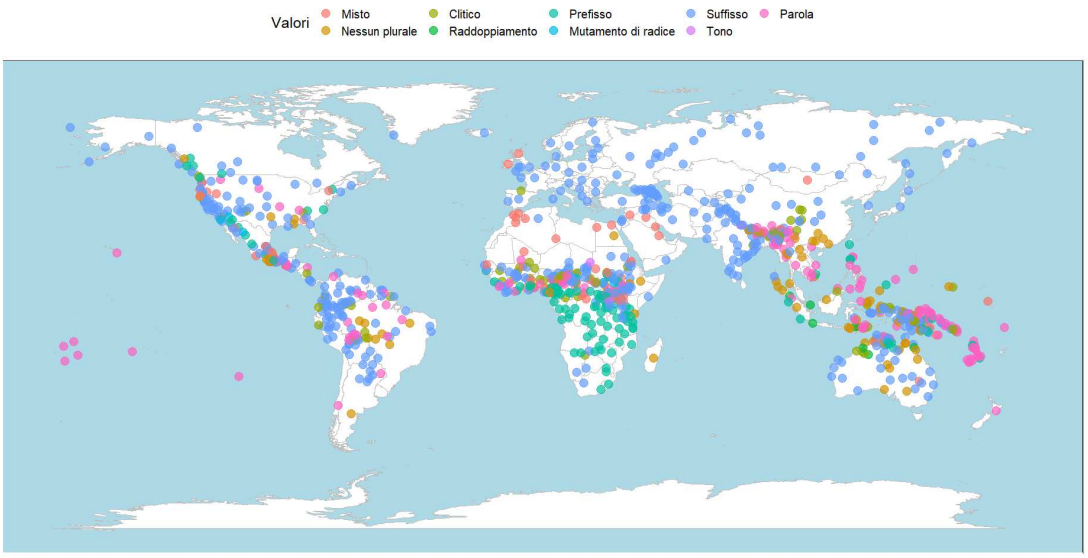

Figura 2.3: Mappa della distribuzione nel mondo dei modi in cui viene codificata la pluralità nei nomi.

Codifica dei plurali nei nomi	Lingue
Suffisso	513
Parola	170
Prefisso	126
Nessun plurale	98
Clitico	81
Misto	60
Raddoppiamento	8
Mutamento di radice	6
Tono	4

Tabella 2.3: Distribuzione di frequenza delle lingue per ciascun valore assunto dalla variabile *Codifica dei plurali nei nomi*.

2.2 Distribuzione dei fonemi nel mondo

Spesso è possibile individuare distribuzioni particolari anche per i fonemi. Non è insolito infatti che il sistema fonologico di lingue geograficamente vicine tenda ad assomigliarsi, o che condividano dei fonemi insoliti.
Viene ora presentato un esempio di una mappa con la distribuzione di alcuni fonemi nel mondo [4, 40]. Come prima è disponibile un file html interattivo che è associato alla mappa seguente [8, 39].
La mappa in Figura 2.6 mostra il modo con cui si ripartiscono nel mondo i fonemi /e/ ed /o/.

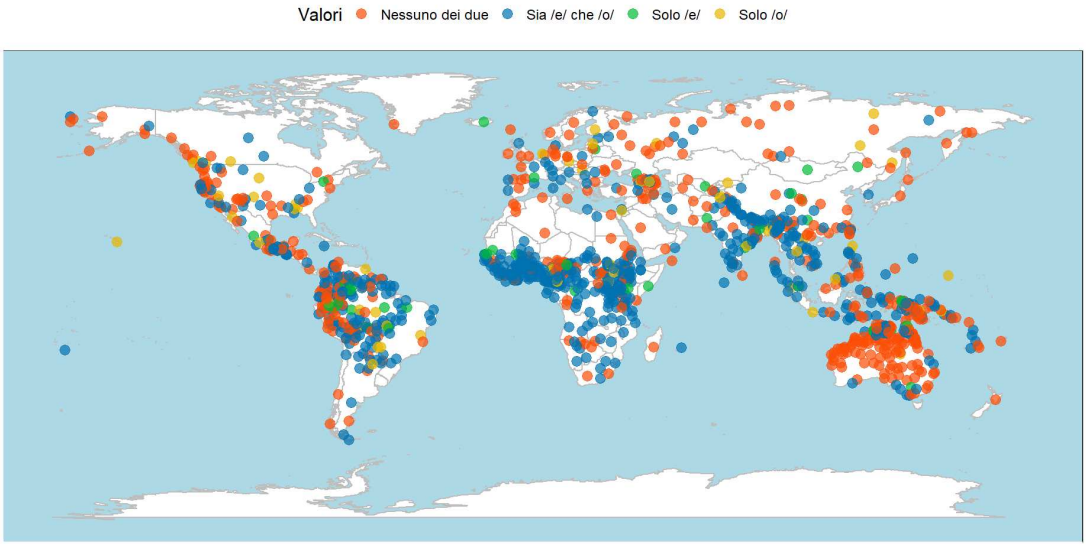

Figura 2.4: Mappa della distribuzione nel mondo dei fonemi /e/ ed /o/.

Nella lingua italiana il fonema /e/ corrisponde al suono della **e** in c**e**ra ed il fonema /o/ alla **o** di p**o**mo. Salta subito all'occhio il fatto che le lingue che distinguono fra entrambi i fonemi si concentrino in Africa subsahariana, in America meridionale, nel subcontinente indiano, nel sud-est asiatico e in parte dell'Europa. Diversamente sono assenti nel resto dell'Europa e particolarmente in Australia, area quest'ultima dove sono più frequenti sistemi vocalici formati dai soli tre fonemi /a i u/. Come si può notare le lingue dove è presente solo un fonema e non l'altro sono nettamente più rare. Se una lingua possiede il fonema /e/ è molto probabile che possieda anche il fonema /o/ e viceversa. Il motivo è che le lingue sono naturalmente portate a preferire la simmetria nel modo in cui organizzano i loro fonemi. Generalmente una lingua con un inventario fonologico asimmetrico tende ad essere meno stabile e a cambiare più velocemente rispetto ad una con un inventario simmetrico.
In Tabella 2.4 viene riportata la distribuzione di frequenza delle lingue per la presenza o l'assenza dei fonemi /e/ ed /o/, in ordine decrescente.

Fonemi /e/ ed /o/	Lingue
Sia /e/ che /o/	691
Nessuno dei due	500
Solo /o/	73
Solo /e/	61

Tabella 2.4: Distribuzione di frequenza delle lingue per la presenza o l'assenza dei fonemi /e/ ed /o/.

2.3 Distribuzione dei valori mancanti in *caratteristiche*

Per ammissione dei suoi stessi autori, il database WALS mostra delle limitazioni non triviali dal punto di vista statistico [21].
Nonostante renda direttamente disponibile per la prima volta un'enorme mole di informazioni di tipo linguistico, uno dei principali problemi che lo affligge è la grande quantità di valori mancanti, specie per quanto riguarda le 192 variabili relative alle caratteristiche strutturali. D'altronde tale limitazione può essere superata solo raccogliendo nuovi dati, ovvero svolgendo del lavoro sul campo con i parlanti delle lingue, il che costituisce indubbiamente la parte più costosa della creazione del database. Difatti se si considera il sottoinsieme del dataset *caratteristiche* che riguarda le variabili relative alle caratteristiche strutturali, esso può essere visto come una matrice di dimensione 2660 per 192 (2660 lingue e 192 caratteristiche). Tuttavia delle $2660 \cdot 192 = 510720$ celle disponibili solo 76475 sono occupate. In altre parole, solo per il $\frac{76475}{510720} \approx 15\%$ delle celle è stato rilevato un valore. Conseguentemente ben l'85% delle celle sarà costituito da valori mancanti: una proporzione altissima! Tutto questo si traduce in un'elevata sparsità complessiva del dataset *caratteristiche*.
È possibile concentrarsi sui valori mancanti anche condizionatamente alle caratteristiche o alle lingue. In quanto alle caratteristiche, il numero di valori mancanti per ognuna delle 192 caratteristiche può assumere valori da 0 (la caratteristica è rilevata per ogni lingua del database) a un massimo di 2659.

In Figura 2.5 è possibile osservare un grafico a barre del numero di valori mancanti per ciascuna delle caratteristiche, ordinate in senso crescente in base al numero di valori mancanti riscontrati per essa.

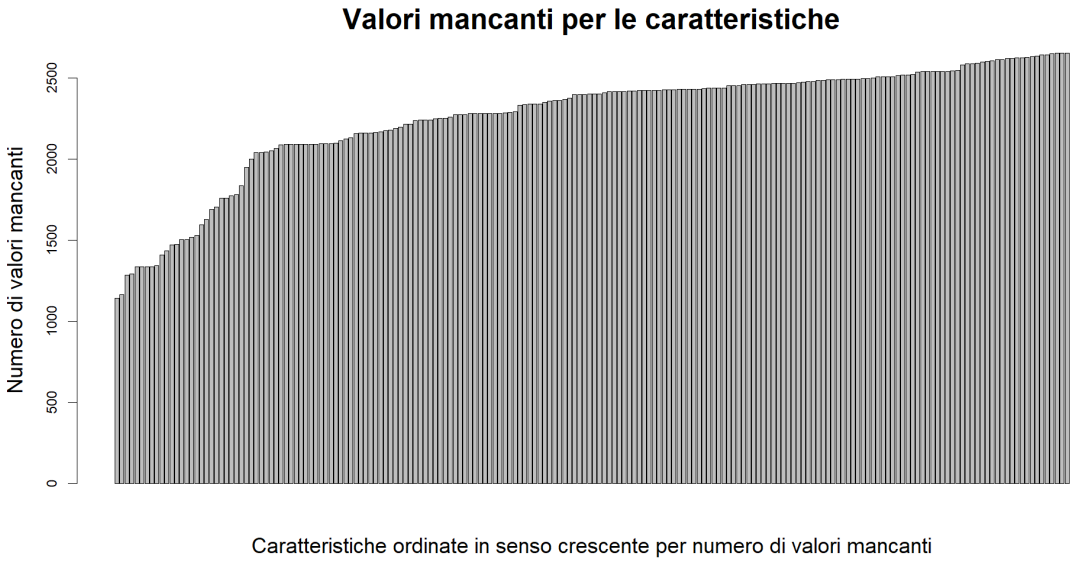

Figura 2.5: Grafico a barre del numero di valori mancanti per le caratteristiche.

Persino la proprietà con meno valori mancanti ne possiede 1142 su 2660. Analogamente il numero mediano di valori mancanti per le proprietà è 2403, e la media è 2262; si tratta di numeri molto elevati. Per giunta la proprietà con più valori mancanti ne possiede 2655 su 2660, il che significa che esiste una caratteristica rilevata solo per 5 lingue.
Per quanto riguarda invece i valori mancanti presenti per ciascuna delle 2660 lingue, la situazione è leggermente migliore. Ognuna delle 2660 lingue può avere da 0 (se tutte le caratteristiche sono state rilevate per lingua) a 191 valori mancanti.
In Figura 2.6 è possibile osservare un grafico a barre del numero di valori mancanti per ciascuna delle caratteristiche, ordinate in senso crescente in base al numero di valori mancanti riscontrati per essa.

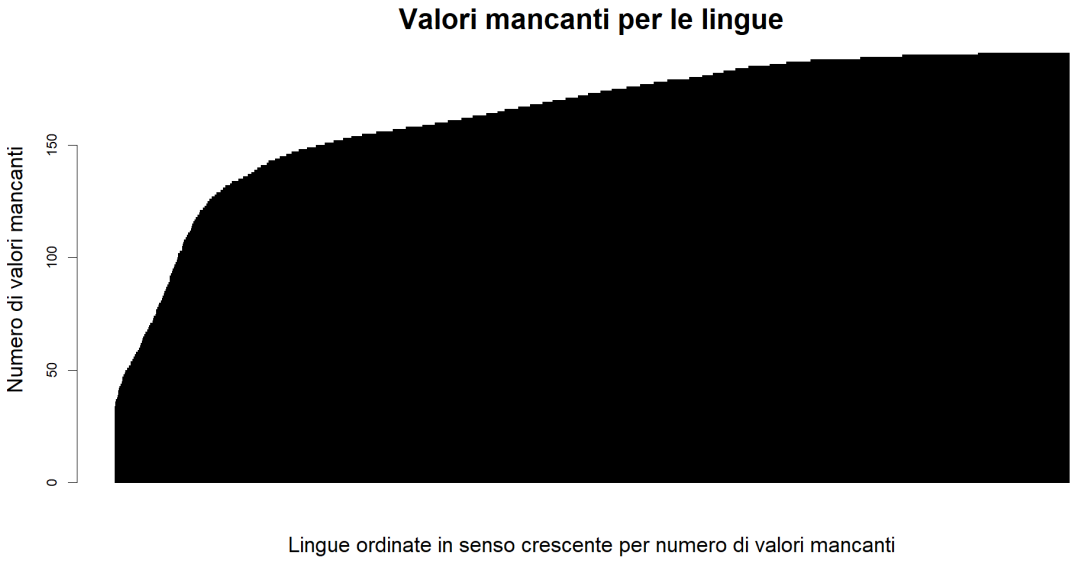

Figura 2.6: Grafico a barre del numero di valori mancanti per le lingue

La lingua con meno valori mancanti è l'inglese, con un minimo di 33. Il numero mediano di valori mancanti per le lingue è 173 e la media è 163.2. Le lingue con più valori mancanti ne possiedono 191 su 192, il che significa che esistono lingue per le quali è stata rilevata solamente una caratteristica.

2.4 Creazione di sottoinsiemi privi di valori mancanti

Per far fronte alla problematica esposta nella sezione 2.3 si è ritenuto opportuno creare dei sottoinsiemi dei dati originari tali da essere privi di valori mancanti.
Come primo passo si individua un sottoinsieme di partenza che funga da base per il sottoinsieme finale. Una scelta ragionevole consiste nel partire dalle 100 lingue con meno valori mancanti per tutte le 192 proprietà strutturali. Ci si accorge che solo 21 proprietà possiedono 0 valori mancanti per ognuna delle 100 lingue. Questo costituisce già un possibile sottoinsieme privo di valori mancanti, tuttavia si desidera mantenere un numero più elevato di proprietà strutturali. Si decide quindi di allentare la condizione, e di considerare le proprietà che abbiano un numero di valori mancanti inferiore a una certa soglia. Se tale soglia è fissata pari a 10, le proprietà che soddisfano la condizione

risultano essere 97 in totale. Il numero di lingue che non hanno valori mancanti in corrispondenza di queste 97 proprietà risulta essere uguale a 30. È possibile verificare che all'aumentare delle proprietà strutturali considerate, diminuisce il numero delle lingue che si possono tenere, e viceversa all'aumentare delle lingue considerate, diminuisce il numero delle proprietà che si possono conservare affinché il sottoinsieme ricavato sia privo di valori mancanti. Tramite le operazioni appena descritte è stato ottenuto il sottoinsieme di *caratteristiche* che si desiderava, avente 30 lingue come righe e 97 caratteristiche come colonne. Da qui in poi il dataset risultante verrà chiamato ***lingue***. All'interno dell'Appendice A è possibile consultare i nomi delle lingue, opportunamente tradotti in italiano, assieme alle caratteristiche selezionate per *lingue*.
Per estrarre dai dati originari dei sottoinsiemi diversi si possono fare variare sia il sottoinsieme iniziale, sia la soglia del numero massimo di valori mancanti presenti nelle proprietà. Ad esempio selezionando le 200 lingue con meno valori mancanti e scegliendo una soglia pari a 17 si ottiene un sottoinsieme di 100 lingue e 52 caratteristiche.

Capitolo 3

Modelli per la classificazione delle lingue

Il presente capitolo si propone di analizzare le relazioni tra le lingue con un approccio prettamente modellistico. Vengono pertanto presentati tre metodi atti alla classificazione delle lingue. Per il primo sono stati adattati dei metodi di *clustering* (raggruppamento) al dataset *famiglie*, un sottoinsieme del dataset *fonemi*. Il secondo consiste nell'uso di reti filogenetiche, ed è stato applicato sia al dataset *lingue*, sia al dataset *fonemi*. Il terzo è basato sull'impiego di alberi di classificazione, ed è stato adoperato soltanto per il dataset *fonemi*.

3.1 *Clustering*

Il dataset *famiglie* contiene informazioni relative a 240 lingue appartenenti a 3 famiglie linguistiche diverse, in cui 80 sono lingue indoeuropee, 80 sono lingue niger-kordofaniane e 80 sono lingue pama-nyunga. Per ciascuna lingua sono rilevate 100 variabili binarie, dove ogni variabile assume valore 1 se un fonema fa parte del sistema fonologico di una lingua e 0 altrimenti. Ci si domanda allora se lingue appartenenti alla medesima famiglia linguistica tendano ad avere sistemi fonologici simili, e se sia quindi ragionevole raggrupparle in base alla famiglia linguistica di appartenenza. Per fare questo, si procede con l'esaminare dei metodi di raggruppamento, o *clustering* [27], ed a valutare la bontà delle analisi ottenute. In generale, un raggruppamento è sensato se mostra omogeneità all'interno dei gruppi ed eterogenei-

tà tra i gruppi. Il *clustering* è effettuato sulla base di similarità, che in molti approcci è concepita in termini di distanza (dissimilarità) in uno spazio multidimensionale. Le informazioni necessarie sono misure relative alla somiglianza tra gli elementi o dati dai quali si possano calcolare similarità. La scelta della misura di similarità più idonea per il calcolo della distanza dipende dalla natura delle variabili (discrete, continue, binarie), dalla scala di misurazione (nominale, ordinale), e dalla conoscenza dell'ambito di applicazione [27].
Sia p il numero di variabili a disposizione. Una funzione a valori reali $d(P,Q)$ è una misura di distanza valida fra due punti arbitrari p-dimensionali P e Q con coordinate $P = (x_1, x_2, ..., x_p)$ e $Q = (y_1, y_2, ..., y_p)$ se soddisfa le seguenti proprietà

$$\begin{aligned} &d(P,Q) = d(Q,P) && \text{(simmetria)},\\ &d(P,Q) \geqslant 0 && \text{(non-negatività)},\\ &d(P,P) = 0 \;\forall\; P && \text{(identità)}. \end{aligned}$$

In aggiunta è una metrica se, dato R un qualsiasi punto intermedio,

$$\begin{aligned} &d(P,Q) = 0 \text{ se e solo se } P = Q,\\ &d(P,Q) \leqslant d(P,R) + d(R,Q) && \text{(disuguaglianza triangolare)}. \end{aligned}$$

La distanza più nota è quella Euclidea. Essa è definita come

$$\begin{aligned} d_E(P,Q) &= \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_p - y_p)^2} = \\ &= \sqrt{\sum_{k=1}^{p} (x_k - y_k)^2}. \end{aligned}$$

Quando possibile, è consigliabile usare distanze *vere* per raggruppare le unità statistiche, ovvero distanze che soddisfino le proprietà delle misure di distanza. D'altro canto, la maggior parte degli algoritmi di *clustering* accetta anche distanze che non rispettano necessariamente la disuguaglianza triangolare.
Nel momento in cui le unità non possono essere rappresentate da misurazioni p-dimensionali sensate, si è soliti confrontare coppie di unità

in base alla presenza o all'assenza di certe caratteristiche. Unità simili hanno più caratteristiche in comune rispetto a unità dissimili. Questo è il caso presente nel dataset *famiglie*, in cui le 100 variabili sono di tipo binario. In questo contesto la distanza Euclidea elevata al quadrato fornisce un conteggio del numero di discrepanze. Una discrepanza si verifica quando due unità assumono valori opposti per una variabile. Una distanza elevata corrisponde a molte discrepanze, ossia a due unità dissimili.
Infatti se x_{ij} è il valore (1 o 0) della j-esima variabile binaria per l'i-esima unità e x_{kj} il valore (di nuovo, 1 o 0) della j-esima variabile per la k-esima unità, $j = 1, 2, ..., p$, allora

$$(x_{ij} - x_{kj})^2 = \begin{cases} 0, & \text{se } x_{ij} = x_{kj} = 1 \text{ o } x_{ij} = x_{kj} = 0, \\ 1, & \text{se } x_{ij} \neq x_{kj}. \end{cases}$$

La distanza Euclidea elevata al quadrato risulta essere uguale a

$$\sum_{j=1}^{p} (x_{ij} - x_{kj})^2.$$

Tuttavia, questo tipo di distanza pesa in maniera uguale gli abbinamenti 1-1 e 0-0. In alcuni casi, una corrispondenza 1-1 è un indicatore di similarità più forte rispetto ad una corrispondenza 0-0. Pertanto, può essere ragionevole escludere le accoppiate 0-0 dal calcolo della distanza. Un indice di similarità che opera questa decisione è la distanza di Jaccard [34]. Essa è pari a $\frac{\#\text{ coppie con un solo 1}}{\#\text{ coppie con almeno un 1}}$, o equivalentemente al rapporto fra le discrepanze e fra le discrepanze più le corrispondenze escludendo gli abbinamenti 0-0. Per esprimere più chiaramente la formulazione di questa distanza, è utile arrangiare le frequenze delle corrispondenze e delle discrepanze per due unità i e k nella forma di una tabella di contingenza, come in Tabella 3.1.

		Unità k		
		1	0	Totale
Unità i	1	a	b	$a+b$
	0	c	d	$c+d$
	Totale	$a+c$	$b+d$	$a+b+c+d$

Tabella 3.1: Tabella di contingenza per due variabili binarie.

La distanza di Jaccard fra due unità i e k è data da $d_J(i,k) = \frac{b+c}{a+b+c}$. Raramente è possibile esaminare tutti i raggruppamenti disponibili per trovare quello migliore, in quanto il numero di partizioni di n unità in g gruppi cresce rapidamente. A causa di questo problema, sono stati sviluppati molti algoritmi per il *clustering* che individuano raggruppamenti *ragionevoli* senza dover elaborare tutte le configurazioni possibili. Gli algoritmi di *clustering* si dividono in due categorie principali:

1. metodi gerarchici,

2. metodi di partizione.

I primi organizzano i dati in sequenze di partizioni nidificate, e possono procedere attraverso una serie di fusioni o divisioni in successione. I metodi di *clustering* gerarchico agglomerativo partono dalle singole osservazioni. Pertanto, all'inizio la numerosità dei cluster corrisponde a quella delle osservazioni. Al primo passo le osservazioni più simili sono raggruppate fra di loro, e ad ogni passo successivo i gruppi risultanti vengono fusi fra di loro in base alle loro similarità.
I secondi determinano il partizionamento dei dati in un numero di gruppi prestabilito in modo da ridurre il più possibile la dispersione all'interno del singolo cluster e, viceversa, di aumentare la dispersione tra un cluster e un altro [27].

3.1.1 Metodi gerarchici

Di seguito vengono elencati i passaggi di un algoritmo di *clustering* gerarchico agglomerativo per raggruppare n unità:

1. si parte con n cluster, ciascuno contenente un'unica unità, e una matrice simmetrica $n \times n$ di distanze (o similarità) $\mathbf{D} = \{d_{ik}\}$;

2. si cerca la coppia di cluster più vicini (simili) all'interno della matrice di distanze. La distanza fra i due cluster U e V più simili fra di loro viene indicata con d_{UV};

3. si uniscono i cluster U e V. Il cluster risultante viene indicato con (UV). Si procede ad aggiornare gli elementi della matrice di distanze rimuovendo le righe e le colonne che corrispondono ai cluster U e V e aggiungendo una riga e una colonna che fornisce le distanze fra il cluster (UV) e i cluster rimanenti;

4. si ripetono i passaggi 2 e 3 per un totale di $n - 1$ volte. Tutte le unità faranno parte di un singolo cluster dopo che l'algoritmo è terminato. Si tiene nota della composizione dei cluster e i livelli (distanze o similarità) per i quali avvengono le unioni.

Ad ogni ripetizione del passaggio 2, si trova la più piccola distanza in $\mathbf{D} = \{d_{ik}\}$ e si uniscono i corrispondenti cluster U e V, formando il cluster (UV). Ad ogni ripetizione del passaggio 3, si indica con $d_{(UV)W}$ la distanza fra (UV) e un qualsiasi altro cluster W. Il modo con cui viene determinata $d_{(UV)W}$ influenza la divisione in gruppi dei passi successivi. I metodi più usati sono i seguenti:

1. legame singolo (*single linkage*). La distanza fra due cluster è determinata dalla distanza fra i due membri, uno appartenente a ciascun cluster, che sono più vicini tra di loro. La distanza fra (UV) e un qualsiasi altro cluster W è definita come

   $$d_{(UV)W} = \min\{d_{UW}, d_{VW}\},$$

   dove le quantità d_{UW} e d_{VW} sono le distanze fra i membri più vicini dei cluster U e W e dei cluster V e W, rispettivamente;

2. legame completo (*complete linkage*). La distanza fra due cluster è determinata dalla distanza fra i due membri, uno appartenente a ciascun cluster, che sono più distanti tra di loro. La distanza fra (UV) e un qualsiasi altro cluster W è definita come

   $$d_{(UV)W} = \max\{d_{UW}, d_{VW}\},$$

   dove le quantità d_{UW} e d_{VW} sono le distanze fra i membri più distanti dei cluster U e W e dei cluster V e W, rispettivamente;

3. legame medio (*average linkage*). La distanza fra due cluster è data dalla distanza media fra tutte le coppie di unità dove un membro appartiene a ciascun cluster. La distanza fra (UV) e un qualsiasi altro cluster W è definita come

$$d_{(UV)W} = \frac{\sum_i \sum_k d_{ik}}{N_{(UV)} \cdot N_W},$$

dove d_{ik} è la distanza fra l'elemento i nel cluster (UV) e l'elemento k nel cluster W, e $N_{(UV)}$ e N_W sono il numero di unità nei cluster (UV) e W, rispettivamente;

4. metodo di Ward. Consiste nel scegliere il raggruppamento successivo che minimizza la perdita di informazione. L'informazione è definita come somma delle devianze interne ai cluster $ESS_{tot} = ESS_1 + ... + ESS_k$, dove k è il numero di cluster. Ad ogni passaggio dell'analisi, viene considerata l'unione di ogni possibile coppia di cluster, e i due cluster la cui combinazione risulta nell'aumento minore di ESS (perdita di informazione minima) sono uniti. Se G_j rappresenta il cluster j-esimo, allora la j-esima componente di ESS è definita come

$$ESS_j = \sum_{x_i \in G_j} (\mathbf{x}_i - \overline{\mathbf{x}}_{G_j})^\intercal \cdot (\mathbf{x}_i - \overline{\mathbf{x}}_{G_j}),$$

dove $\overline{\mathbf{x}}_{G_j}$ rappresenta la media calcolata sulle sole osservazioni nel cluster G_j.

Analogamente ad altre procedure di *clustering*, anche i metodi gerarchici sono sensibili agli *outliers*. Inoltre un elemento collocato in maniera errata all'inizio della procedura non può essere riallocato una volta che l'algoritmo è terminato.

3.1.2 Metodi di partizione

I metodi di partizione richiedono che il numero di cluster venga specificato prima della procedura di *clustering*. Il metodo di partizione più noto è quello delle k-medie (k-*means*). Nella sua versione più semplice, esso consiste in 3 passaggi:

1. si suddividono le osservazioni in k gruppi di partenza o si identificano k centroidi che rappresentino il nucleo dei cluster;

2. si assegna ogni osservazione al cluster con il centroide più vicino. La distanza è calcolata mediante la distanza Euclidea. Successivamente si ricalcola il centroide del cluster che riceve l'osservazione e il centroide del cluster che la cede;

3. si ripete il passaggio 2 finché i cluster non vengono più modificati.

3.1.3 Risultati del *clustering*

Metodi gerarchici

A fronte del fatto che il dataset *famiglie* è composto da 100 variabili binarie, è stato ritenuto opportuno usare come misura di distanza la distanza di Jaccard.
Sono stati adattati tutti e 4 i metodi gerarchici illustrati nella sezione 3.1.1 della presente tesi. L'unico a mostrare dei risultati soddisfacenti è il metodo di Ward, mentre gli altri metodi tendono a raggruppare tutte le lingue all'interno di un'unica macro-classe. In Tabella 3.2 vengono confrontate le effettive famiglie linguistiche di appartenenza con i cluster creati dal metodo di Ward, fissando il numero di cluster desiderati uguale a 3. Per indicare le famiglie sono state usate le seguenti abbreviazioni: *IE = Indoeuropea*, *NC = Niger-Kordofaniana*, *PM = Pama-Nyunga*.

Famiglia	**Gruppo**			
	1	2	3	Totale
IE	0	8	72	80
NC	0	69	11	80
PM	79	0	1	80
Totale	79	77	84	240

Tabella 3.2: Matrice di confusione per il *clustering* gerarchico condotto usando il metodo di Ward.

Le lingue indoeuropee sono state raggruppate nel terzo cluster, le lingue niger-kordofaniane nel secondo e le lingue pama-nyunga nel primo. Nel complesso, solamente 20 osservazioni non sono raggruppate

correttamente, pari all'incirca all'8.3% del totale. La famiglia che è stata raggruppata meglio è quella delle lingue pama-nyunga. Le lingue indoeuropee e niger-kordofaniane sono anch'esse raggruppate correttamente per la maggior parte, sebbene tendano ad essere confuse le une per le altre. I risultati indicano che i sistemi fonologici delle lingue pama-nyunga parlate in Australia sono simili tra di loro e al contempo diversi rispetto ai sistemi fonologici delle altre due famiglie.
Le analisi sono state realizzate tramite la libreria di R *cluster* [31].

Metodi di partizione

Per verificare se anche i metodi di partizione generano dei cluster che rispecchiano la suddivisione in famiglie delle lingue del dataset *famiglie*, si è applicato il *k-means*. Come prima cosa si prova a vedere quale sia il numero di cluster suggerito dai dati. Per fare questo si osserva la percentuale di devianza spiegata che si ottiene usando gruppi con dimensioni diverse. La chiave è cercare di capire a partire da quale punto l'incremento marginale della devianza possa venire trascurato. Il grafico in Figura 3.1 mostra la percentuale di devianza spiegata facendo variare da 1 a 10 il numero di partizioni effettuate. Il numero di centroidi di partenza è stato fissato uguale a 5.

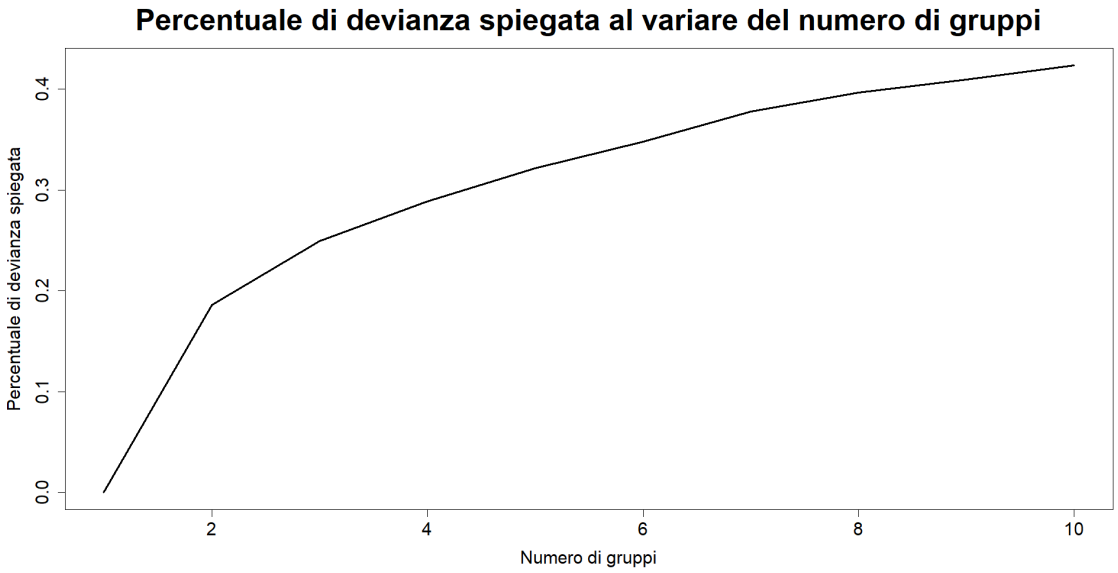

Figura 3.1: Grafico della percentuale di devianza spiegata al variare del numero di gruppi.

Dal grafico in Figura 3.1 risulta opportuno rappresentare i dati tramite 3 gruppi.
In Tabella 3.3 vengono confrontate le famiglie linguistiche di appartenenza con i cluster individuati dal *k-means* per 3 gruppi.

Famiglia	**Gruppo**			
	1	2	3	Totale
IE	0	77	3	80
NC	0	9	71	80
PM	78	2	0	80
Totale	78	88	74	240

Tabella 3.3: Matrice di confusione per il *k-means*.

Il *clustering* fornisce dei risultati apprezzabili pure in questa situazione. La proporzione di lingue che non sono state raggruppate correttamente ammonta a $\frac{14}{240} \approx 0.058$, ovvero circa al 5.8% del totale. La famiglia che forma il raggruppamento migliore è nuovamente quella delle lingue pama-nyunga. Nel complesso, l'uso della distanza Euclidea da parte del *k-means* non ha influito negativamente sull'individuazione dei cluster.
Per poter visualizzare meglio i risultati ottenuti tramite il *k-means*, in Figura 3.2 è fornito un grafico dei gruppi sul piano delle prime due componenti principali. Ciascuna lingua viene rappresentata con segni grafici differenti a seconda della famiglia a cui appartiene, mentre i cluster sono suddivisi in base al colore e delimitati da una linea di contorno.

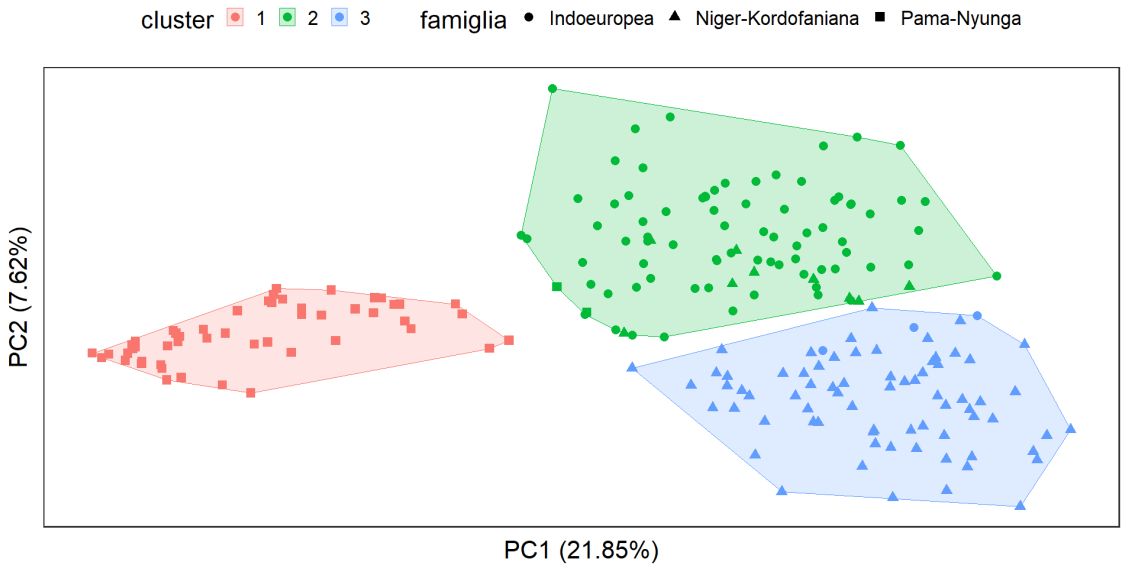

Figura 3.2: Famiglia di appartenenza e raggruppamento del *k-means*.

I cluster relativi alle lingue indoeuropee e niger-kordofaniane sono particolarmente vicini fra di loro e per questo motivo alcuni dei loro elementi vicini al confine appartengono alla famiglia linguistica sbagliata. Al contrario, il cluster riguardante le lingue pama-nyunga dimostra di essere sufficientemente distante dagli altri due affinché le sue osservazioni siano per la quasi totalità raggruppate correttamente. Per ottenere il grafico in Figura 3.2 è stata usata la libreria di R *ggfortify* [23].

3.2 Reti filogenetiche

Nelle scienze biologiche, si indica con unità tassonomica o *taxon*, plurale ***taxa***, un oggetto o un organismo la cui storia evolutiva è di interesse [24]. I taxa appartenenti ad uno stesso raggruppamento sono distinguibili morfologicamente da altri taxa per almeno una caratteristica comune e vengono tradizionalmente fatti discendere tutti da un progenitore comune. La storia evolutiva di un raggruppamento di taxa è solitamente rappresentata con dei diagrammi chiamati alberi filogenetici [25]. Ciascun nodo (o biforcazione) con dei discendenti indica l'antenato comune più recente dei soggetti che si trovano ai nodi successivi, e la radice dell'albero è l'antenato comune di tutti i taxa esaminati. L'uso di rappresentazioni di questo tipo si adatta bene alla schematizzazione delle relazioni fra le specie del mondo, in quanto tut-

ta la vita sul pianeta Terra fa parte di un singolo albero filogenetico. Questo è dovuto al fatto che ogni essere vivente discende da un unico organismo primordiale vissuto miliardi di anni fa.
È interessante notare come da questo punto di vista vi siano alcune somiglianze peculiari fra genetica e linguistica. Invero, analogamente alle specie animali, anche le lingue possono essere ricondotte a delle famiglie, chiamate appunto *famiglie linguistiche*. L'idea di base è che le differenze fra le lingue del mondo si possono riportare a concetti analoghi alle mutazioni genetiche riscontrabili in natura fra diverse generazioni di una specie. A questo proposito si osservi la similarità fra l'albero filogenetico di 12 specie di primati [22] e l'albero linguistico della famiglia delle lingue indoeuropee, costruito in base ad un'analisi filogenetica incentrata sui vincoli di provenienza [7], in Figura 3.3.

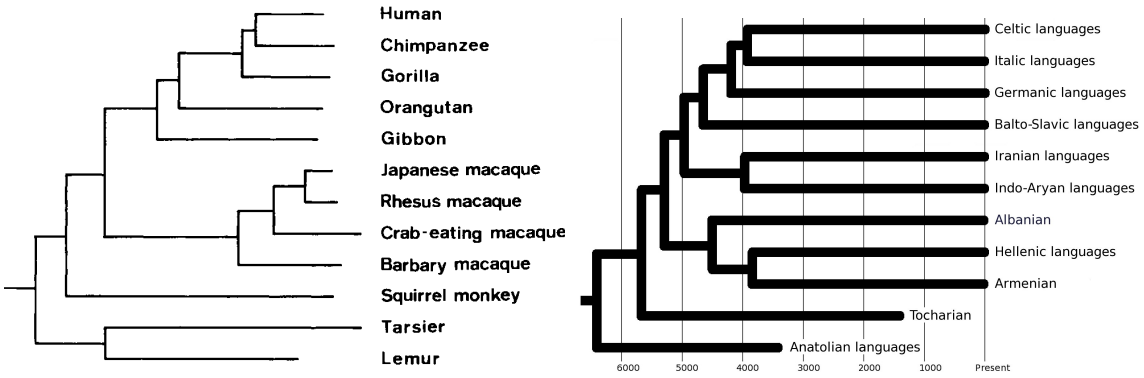

Figura 3.3: Albero filogenetico per 12 specie di primati (a sinistra) e albero linguistico delle lingue indoeuropee (a destra).

Nella filogenetica, tuttavia, questo tipo di relazione non spiega tutte le interconnessioni evolutive presenti tra differenti taxa, ma solo una parte. La ricerca attuale propone una visione integrata dell'evoluzione consolidando le relazioni filogenetiche strutturate ad albero con altri e differenti meccanismi [11]. Risultano perciò adeguate nel contesto di riferimento le reti filogenetiche.
Una **rete filogenetica** è un qualsiasi grafo usato per visualizzare relazioni di tipo evolutivo fra individui, taxa, geni, cromosomi, genomi o specie [26]. Le reti filogenetiche sono utilizzate quando si è in presenza di eventi reticolari come l'ibridizzazione, il trasferimento di geni in senso orizzontale, la ricombinazione o la perdita di geni. Differiscono dagli alberi filogenetici in quanto modellizzano esplicitamente reti densamente interconnesse, attraverso l'aggiunta di nodi ibridi (nodi con due genitori) invece di usare solo nodi gerarchici, ciascuno con

un unico genitore. Gli alberi filogenetici sono pertanto un sottoinsieme delle reti filogenetiche [5, 6]. In altre parole, le reti filogenetiche ammettono la possibilità che taxa appartenenti allo stesso raggruppamento possano non derivare tutti da un antenato comune. Difatti le mutazioni riscontrabili nei taxa non sono sempre riconducibili ad una discendenza diretta dal progenitore, ma possono essere dovute all'influenza esercitata sui taxa dall'ambiente o anche da altri taxa.
In conseguenza di quanto detto, è possibile fare la seguente considerazione: visto che modelli come gli alberi filogenetici sono applicabili non solo a esseri viventi, ma anche alle lingue parlate in tutto il mondo, è ragionevole assumere che lo stesso sia vero pure per le reti filogenetiche. Inoltre l'implementazione delle reti filogenetiche per la rappresentazione delle relazioni tra le lingue risulta essere particolarmente opportuna per una serie di motivi.
Innanzitutto, i linguisti concordano nell'affermare che le lingue non hanno tutte un'origine comune, vale a dire che non esiste un'unica famiglia linguistica che comprenda tutte le lingue parlate nel mondo. Al contrario è più probabile che il linguaggio abbia avuto origine indipendentemente nel tempo in varie parti del mondo. A causa di questo il dataset *caratteristiche* contiene 180 famiglie linguistiche separate e 83 lingue isolate, lingue queste ultime per le quali non è stata dimostrata la parentela con nessun altro idioma. Si dice che le lingue che appartengono alla medesima famiglia linguistica sono geneticamente correlate. Dunque in un tale contesto un modello come l'albero filogenetico può essere applicato solo ad un livello *locale*, vale a dire internamente a una singola famiglia linguistica. Oltretutto è completamente inadatto nel momento in cui si vogliano studiare lingue che non siano correlate geneticamente fra di loro.
Pertanto il vantaggio delle reti filogenetiche è che consentono di rappresentare relazioni anche quando le lingue esaminate non sono necessariamente correlate geneticamente. Di conseguenza esse offrono un modo per tenere conto di fenomeni come le aree linguistiche, nelle quali le somiglianze fra le lingue che le compongono sono emerse per via della prossimità geografica. Pertanto l'utilizzo di reti filogenetiche è particolarmente efficace nell'individuare eventuali aree linguistiche in un territorio o a livello mondiale.
Infine, adattare alle lingue un modello solitamente adoperato in bioinformatica è giustificato dal fatto che anche le lingue, ugualmente agli esseri viventi, mutano ed evolvono nel corso del tempo, stringono relazioni di parentela con lingue provenienti da altre famiglie e possono

persino dare vita a delle *lingue figlie*. Da questo punto di vista, si può pensare che l'insieme di attributi e regole che caratterizzano una lingua formi una sorta di *genoma linguistico*, grazie al quale si possono studiare relazioni di parentela e di vicinanza fra lingue.
Questa strada è già stata percorsa da alcuni autori nell'ambito di famiglie linguistiche di dimensione ridotta [17]. La presente tesi si propone di estendere l'analisi a un livello che comprenda lingue provenienti da molteplici famiglie linguistiche e parlate in ogni regione del mondo.

3.2.1 SplitsTree

SplitsTree è un programma gratuito usato per generare reti filogenetiche senza radice a partire da vari tipi di dati, quali ad esempio allineamenti di sequenze, matrici di distanze e insiemi di alberi. In bioinformatica, un allineamento di sequenze è una procedura con cui vengono messe a confronto ed allineate due o più sequenze primarie di aminoacidi, DNA o RNA. Lo scopo è identificare delle regioni di similarità che possano essere il risultato di relazioni di tipo funzionale, strutturale o evolutivo fra le sequenze degli organismi. Più in generale consiste nell'allineare un insieme di taxa come se fossero le righe di una matrice, e confrontarli in base alle loro caratteristiche.
L'obiettivo di SplitsTree è quello di fornire un metodo per elaborare reti filogenetiche. Come suggerito dal nome del programma, esso è basato sul concetto matematico di ***split***. Ad esempio, si consideri il seguente allineamento di sequenze binarie, costituito da 4 taxa:

```
a 010011010110
b 100001011110
c 011001101110
d 010001101111
```

Uno split è una partizione dell'insieme dei taxa in due gruppi disgiunti e non-vuoti [6]. Ciascuna colonna non costante dell'allineamento definisce uno split fra i taxa con valore 0 e con valore 1. Ad esempio, la prima colonna partiziona i taxa in due raggruppamenti $\{a, c, d\}$ e $\{b\}$, e genera quindi lo split $\frac{\{a,c,d\}}{\{b\}}$, equivalentemente esprimibile come $\frac{\{b\}}{\{a,c,d\}}$. Al contrario la quarta colonna non definisce alcuno split perché i caratteri sono costanti. Più split ci sono fra due taxa, più essi risultano essere diversi.
SplitsTree sfrutta un algoritmo agglomerativo chiamato *Neighbor-Net*

per generare una rete filogenetica a partire da un allineamento di sequenze. Si procede ora ad illustrare il funzionamento di Neighbor-Net.

Processo agglomerativo

L'algoritmo ha inizio assegnando un nodo ad ogni taxon. Ad ogni iterazione, si seleziona una coppia di nodi e si aspetta fino a quando uno dei due nodi non forma nuovamente una coppia con un altro nodo. A questo punto si procede a sostituire i tre nodi collegati con due nodi collegati da un arco, e così facendo riduciamo la matrice delle distanze. Se vi è ancora un nodo collegato ad altri due, si fa una seconda agglomerazione e riduzione. Si continua poi all'iterazione successiva. Ogni volta che si agglomera si sostituiscono tre nodi con due. Questo metodo agglomerativo genera una collezione di split che non può essere rappresentata da un albero filogenetico. La procedura continua fino a che non rimangono solo due (o tre) nodi liberi. Il prodotto finale dell'algoritmo Neighbor-Net è una collezione di split *circolari*. Formalmente, una collezione di split dell'insieme dei taxa X è circolare se esiste un ordinamento $x_1, x_2, ..., x_n$ dei taxa tale che ciascun split è della forma $\{x_i, x_{i+1}, ..., c_j\}|X - \{x_i, ..., x_j\}$ per i e j che soddisfino $1 \leqslant i \leqslant j < n$. Graficamente, gli split circolari emergono quando si pongono attorno a un cerchio i taxa e si considerano gli split ottenuti tagliando il cerchio lungo una linea [6].

Formule di selezione

Il metodo Neighbor-Net è determinato dalla formule usate per selezionare nodi per l'agglomerazione e dalla formula usata per ridurre la matrice delle distanze a seguito di ogni agglomerazione. Si supponga di avere n nodi rimanenti. All'inizio dell'algoritmo, nessuno dei nodi avrà vicini già assegnati a loro. In seguito, alcune coppie di nodi saranno state identificate come vicine, ma non agglomerate. Occorre tenere conto di queste relazioni di vicinanza quando si selezionano nodi per l'agglomerazione.
Le relazioni di vicinanza raggruppano gli n nodi in $C_1, C_2, ..., C_m$ cluster, con $m \leqslant n$, alcuni dei quali contengono un singolo nodo e altri che contengono una coppia di nodi vicini. La distanza $d(C_i, C_j)$ fra due cluster è la media delle distanze fra gli elementi in ciascun cluster:

$$d(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \cdot \sum_{x \in C_i} \sum_{y \in C_j} d_{xy},$$

dove con $|C_i|$ e $|C_j|$ si indicano le numerosità dei cluster C_i e C_j, rispettivamente.
La selezione di nodi vicini procede in due passi. Prima è necessario trovare la coppia di cluster che minimizza la formula

$$Q(C_i, C_j) = (m-2) \cdot d(C_i, C_j) - \sum_{\substack{k=1 \\ k \neq i}}^{m} d(C_i, C_k) - \sum_{\substack{k=1 \\ k \neq j}}^{m} d(C_j, C_k).$$

Si supponga che C_{i^*} e C_{j^*} siano due cluster che minimizzano $Q(C_i, C_j)$. Il secondo passo consiste nel scegliere quale nodo $x_i \in C_{i^*}$ e quale nodo $x_j \in C_{j^*}$ siano da rendere vicini. I cluster C_{i^*} e C_{j^*} contengono ciascuno o uno o due nodi. Se questi cluster fossero separati in nodi individuali si otterrebbero $m + |C_{i^*}| + |C_{j^*}| - 2$ cluster in totale. Si indichi con $\hat{m}$ l'espressione $m + |C_{i^*}| + |C_{j^*}| - 2$. Per mantenere la consistenza, questo valore $\hat{m}$ sostituisce m nella definizione di $Q(C_i, C_j)$ quando si selezionano particolari nodi *all'interno* di cluster. In altri termini, si selezionano i nodi $x_i \in C_{i^*}$ e $x_j \in C_{j^*}$ che minimizzano

$$\hat{Q}(x_i, x_j) = (\hat{m}-2) \cdot d(x_i, x_j) - \sum_{\substack{k=1 \\ k \neq i}}^{\hat{m}} d(x_i, C_k) - \sum_{\substack{k=1 \\ k \neq j}}^{\hat{m}} d(x_j, C_k).$$

Formule di riduzione della distanza

Si supponga che il nodo y abbia due vicini, x e z. Nel passo agglomerativo di Neighbor-Net, x, y e z sono sostituiti da due nuovi nodi u e v.
Le distanze da u e v ad un altro nodo a sono calcolate mediante le formule di riduzione

$$\begin{aligned} d(u, a) &= \alpha \cdot d(x, a) + \beta \cdot d(y, a), \\ d(v, a) &= \beta \cdot d(y, a) + \gamma \cdot d(z, a), \\ d(u, v) &= \alpha \cdot d(x, y) + \beta \cdot d(x, z) + \gamma \cdot d(y, z), \end{aligned}$$

dove α, β e γ sono numeri reali non-negativi con $\alpha + \beta + \gamma = 1$.
Di default si usa $\alpha = \beta = \gamma = \frac{1}{3}$.
Le formule di selezione e la formula di riduzione della distanza garantiscono la consistenza dal punto di vista statistico del metodo Neighbor-Net [6].

Risultato di Neighbor-Net

La rappresentazione grafica della rete filogenetica generata da SplitsTree è chiamata *splits graph*. Gli splits graph generati da Neighbor-Net sono sempre planari, ossia possono essere sempre visualizzati in uno spazio a due dimensioni.
In sostanza se due taxa sono simili, ovvero se i loro allineamenti di sequenze si assomigliano, allora essi saranno fisicamente vicini nella rete filogenetica costruita da Neighbor-Net. In questo senso si può dire che Neighbor-Net, più che una tecnica di classificazione, sia una tecnica di *raggruppamento*. Una volta adattata una rete filogenetica ai dati, sarà compito dell'utente determinare se l'algoritmo ha individuato delle classi chiaramente definite o meno, in base alla sua conoscenza sul tema.
SplitsTree è messo a disposizione dall'Università di Tübingen. Attualmente la versione più recente, e quella usata nella presente tesi, è la 4.18.3 del 23 luglio 2022 [25].
SplitsTree usa un formato per file estensibile e modulare chiamato NEXUS. È ampiamente adottato in bioinformatica, e registra informazioni riguardo a taxa e caratteri morfologici e molecolari [30].

3.2.2 Creazione di input per Splistree

Per poter fornire dei dati a SplitsTree, bisogna prima creare un allineamento di sequenze a partire da dati di tipo linguistico, siano essi relativi alle caratteristiche o ai fonemi delle lingue che si vogliono esaminare. Questo si può ottenere manipolando tramite R [34] i datasets *lingue* e *unione*, rispettivamente riguardanti le caratteristiche e i fonemi. Una volta fatto questo, si procede con il manipolare la matrice dell'allineamento di sequenze così ottenuta mediante Python, in modo tale da creare un file di testo con estensione `.txt` scritto in maniera conforme al formato NEXUS. Quest'ultimo verrà infine dato in input a SplitsTree.

Input a partire dai dati relativi alle caratteristiche

Per poter creare un allineamento di sequenze derivato dalle caratteristiche del dataset *caratteristiche*, ci si è avvalsi del dataset *lingue*. Come illustrato nella sezione 2.4 della presente tesi, *lingue* è un sottoinsieme del dataset *caratteristiche* privo di valori mancanti.
I taxa dell'allineamento risultante sono costituiti dalle 30 lingue del dataset *lingue*. Ciascuna delle sue colonne è una variabile dicotomica che assume valore 1 se la lingua assume una certa modalità per una caratteristica, e 0 altrimenti. Pertanto la lunghezza delle sequenze è data dalla somma del numero dei livelli delle 97 caratteristiche di *lingue*, la quale risulta essere pari a 380. La matrice formata da 0 e 1 così ottenuta è un allineamento di sequenze binarie. Il passaggio successivo consiste nel creare due file di testo con estensione `.txt` contenenti rispettivamente l'allineamento di sequenze e il nome delle 30 lingue. Il programma Python reperibile nell'Appendice D della presente tesi prende in input i due file di testo e una volta eseguito, li combina per formare un unico file in formato NEXUS. La sintassi del formato NEXUS richiede che vengano prima specificati i taxa (in questo caso i nomi delle 30 lingue) e che venga poi fornito l'allineamento di sequenze. È importante prestare attenzione anche alla rimozione di caratteri speciali che non sono accettati da SplitsTree, quali ad esempio *á, ä, ã, é* ed *í* per il nome di alcune lingue.
Il risultato finale di queste operazioni può essere fornito in input a SplitsTree per generare una rete filogenetica. Nell'Appendice B della presente tesi è possibile osservare l'input per SplitsTree così ottenuto. Per questioni di spazio, sono stati riportati solo 30 dei 380 split.

Input a partire dai dati relativi ai fonemi

In questa occasione si è deciso di testare ulteriormente l'abilità dell'algoritmo Neighbor-Net nel discriminare fra famiglie linguistiche e ad individuare eventuali aree linguistiche. Per realizzare questo sono stati somministrati a SplitsTree due datasets appositamente creati. Il primo consiste in 6 famiglie linguistiche, per ognuna delle quali sono state considerate 5 lingue dal dataset *unione*, in modo tale da creare un campione di dimensione $6 \cdot 5 = 30$ lingue. Il secondo è il dataset *famiglie*, che è formato da 240 lingue ripartite in 3 famiglie linguistiche di 80 lingue ciascuna. Visto il gran numero di *labels* che si generano nella rete filogenetica risultante in quest'ultimo caso, si è deciso di assegnare ad ogni lingua un'etichetta formata da una delle sigle `ie`,

`nc` o `pm` se una lingua appartiene rispettivamente alla famiglia delle lingue indoeuropee, niger-kordofaniane o pama-nyunga, assieme a un identificatore numerico che va da 1 a 240. In entrambi i casi le colonne dei relativi allineamenti sono costituiti dalle 100 variabili del dataset *unione* che contengono informazioni riguardo ai 100 fonemi più comuni a livello mondiale. Infatti, come osservato nella sezione 1.2.2 della presente tesi, le variabili relative ai fonemi sono già delle variabili dicotomiche che assumono valore 1 se il fonema è presente nella lingua e 0 se è assente. In altre parole esse costituiscono già un allineamento di sequenze. Arrivati qui, la procedura è analoga a quella illustrata nella sezione precedente: si stampa su un file di testo l'allineamento di sequenze e tramite Python si crea un file in formato NEXUS. A questo punto non resta che passare il file in input a SplitsTree.

3.2.3 Risultati delle reti filogenetiche per *caratteristiche*

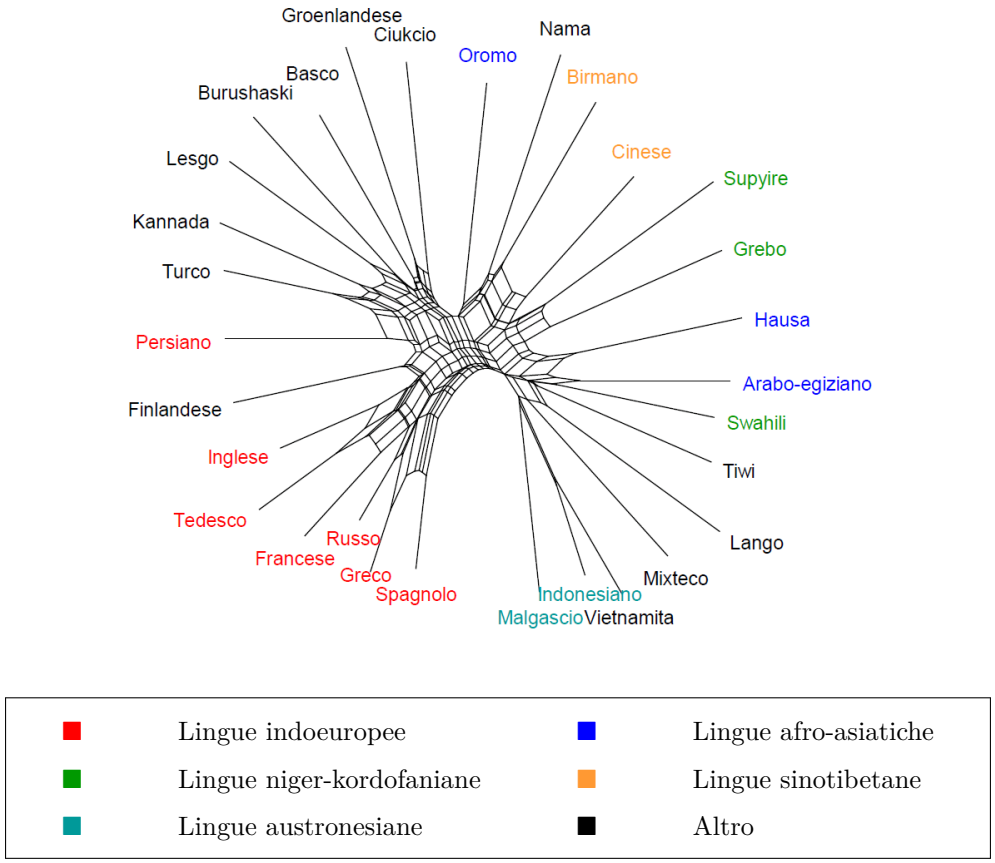

Figura 3.4: Rete filogenetica costruita a partire dalle caratteristiche del dataset *lingue*, colorata in base alla famiglia linguistica di appartenenza.

Ci si appresta ora ad interpretare la rete filogenetica in Figura 3.4, la quale è basata sull'allineamento di sequenze creato a partire dal dataset *lingue*. Come indicato nella legenda posta qui sopra, le lingue sono state colorate in base alla famiglia linguistica di cui fanno parte. SplitsTree è stato in grado di discriminare complessivamente bene fra le varie famiglie linguistiche. In particolare, si può vedere come le lingue indoeuropee siano tutte concentrate nel settore in basso a sinistra, formando peraltro un ramo ben distinto dal resto delle lingue, fatta eccezione per il persiano. Infatti, il persiano è parlato in medio oriente e sebbene sia una lingua indoeuropea, ha subito per molti secoli l'influenza di lingue di ceppo turcico, come il turco. Al contrario il finlandese, pur essendo una lingua uralica, viene comunque raggruppato insieme alle lingue indoeuropee. Questo è dovuto

al fatto che il finlandese fa comunque parte dell'area linguistica europea; per questo motivo, nonostante non faccia parte della famiglia delle lingue indoeuropee, condivide ugualmente numerose caratteristiche con loro, e questo fa sì che SplitsTree lo ponga vicino ad esse. Per motivi analoghi, anche lo swahili è posto più vicino all'arabo egiziano che alle altre lingue niger-kordofaniane, per via della forte influenza che la lingua araba ha storicamente avuto sullo swahili. È possibile che il Cinese mandarino sia stato raggruppato accanto alle lingue niger-kordofaniane Supyire e Grebo in quanto tutte e tre possiedono i cosiddetti toni, ossia dei tratti prosodici caratterizzati dalla variazione dell'altezza del suono di una sillaba.
In Figura 3.5 è rappresentata la stessa rete filogenetica di Figura 3.4, tuttavia ora le lingue sono colorate in base alla macroarea in cui sono parlate.

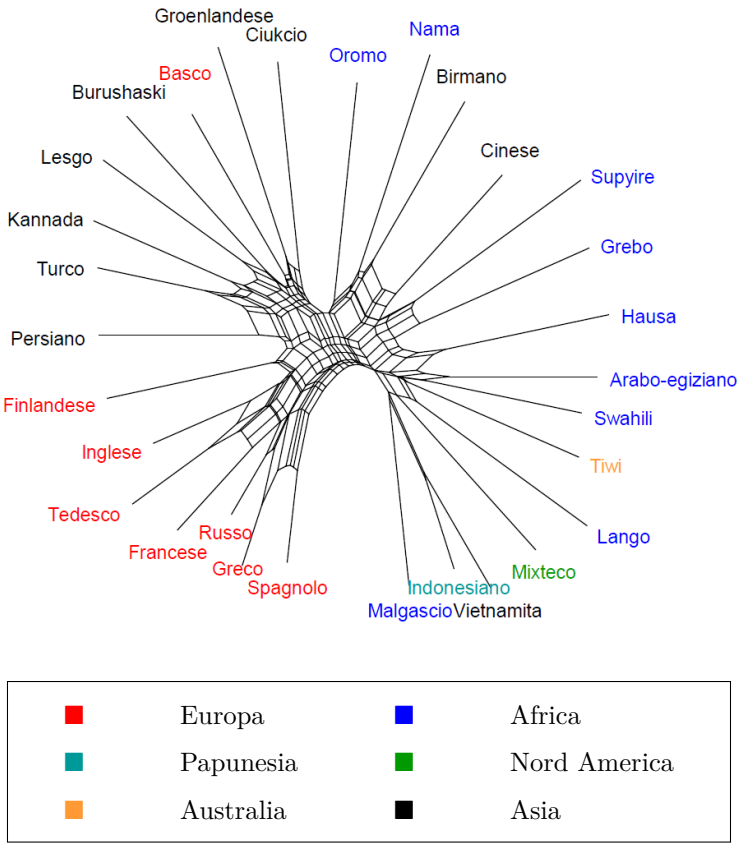

Figura 3.5: Rete filogenetica costruita a partire dalle caratteristiche del dataset *lingue*, colorata in base alla macroarea di appartenenza.

All'interno di Figura 3.5 è chiaramente osservabile l'area linguistica europea. Essa è indicata con il colore rosso ed è ubicata nella parte

in basso a sinistra della rete filogenetica. Si noti come la lingua basca sia considerevolmente lontana dalle altre lingue parlate in Europa, nonostante sia parlata in aree geografiche situate nel nord della Spagna e nell'estremo sud-ovest della Francia. Ciò si deve al fatto che si tratta di una lingua isolata, che possiede delle caratteristiche che la differenziano da tutte le altre lingue parlate in Europa. Il malgascio e l'indonesiano vengono collocati uno vicino all'altro in quanto sono entrambe lingue austronesiane, come è visibile anche in Figura 3.4. Questo sebbene siano parlati rispettivamente in Madagascar ed in Indonesia, che come si può vedere in Figura 3.5 sono addirittura nazioni situate in continenti diversi. Pure buona parte delle lingue parlate in Africa viene raggruppata nella zona sulla destra della rete filogenetica.

3.2.4 Risultati delle reti filogenetiche per *fonemi*

Rete con 6 famiglie linguistiche

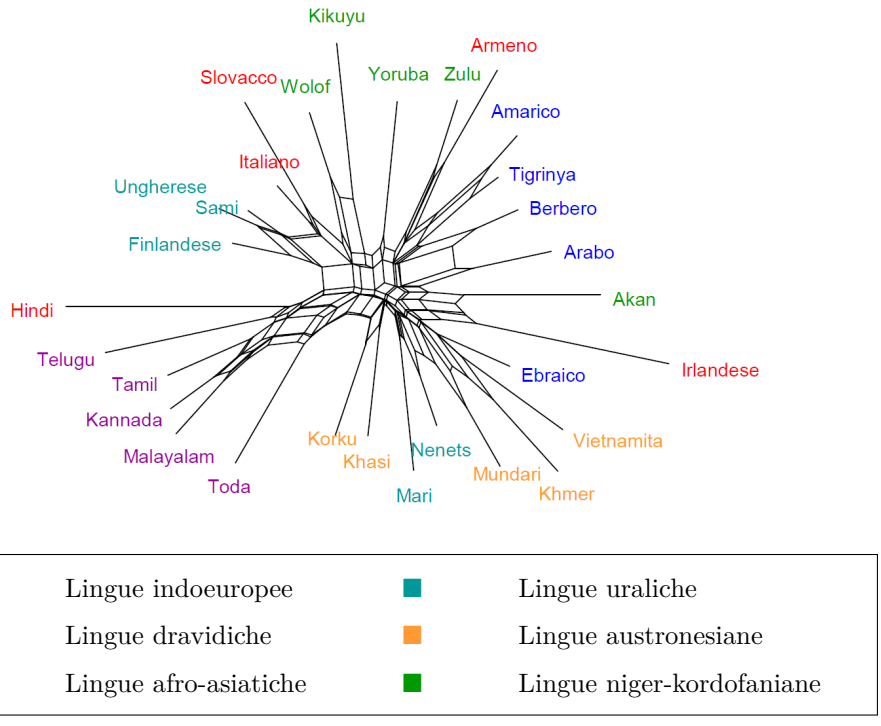

Figura 3.6: Rete filogenetica costruita a partire dai fonemi del dataset *unione*, colorata in base ad una di 6 famiglie linguistiche di appartenenza.

La rete filogenetica in Figura 3.6 mostra che SplitsTree è capace di individuare con efficacia la famiglia di appartenenza di una lingua anche

a partire dai suoi fonemi; suggerisce inoltre che il sistema fonologico di una lingua sia influenzato non solo dalla famiglia linguistica di cui fa parte, ma anche dall'area linguistica in cui è parlata. Si considerino a questo proposito le lingue collocate nella parte in basso a sinistra della rete. Come è possibile vedere, le lingue dravidiche (colorate in viola e parlate nella metà inferiore del subcontinente indiano) sono state tutte raggruppate una vicina all'altra. Si osservi che nonostante la lingua Hindi sia una lingua indoeuropea e le lingue Korku e Khasi siano lingue austronesiane, esse sono posizionate di fianco alle lingue dravidiche poiché fanno tutte parte dell'area linguistica del subcontinente indiano. Le lingue indoeuropee coprono un territorio vasto ed eterogeneo che si estende, come è suggerito dal nome, dall'India all'Europa. Quest'estensione geografica, nonché il contatto con altri popoli e lingue, ha determinato una notevole differenziazione dal punto di vista fonologico. Questo aspetto è rispecchiato dal fatto che nella rete filogenetica in Figura 3.6 le lingue indoeuropee sono tutte considerevolmente distanziate fra di loro, fatta eccezione per l'italiano e lo slovacco.

Rete con 3 famiglie linguistiche

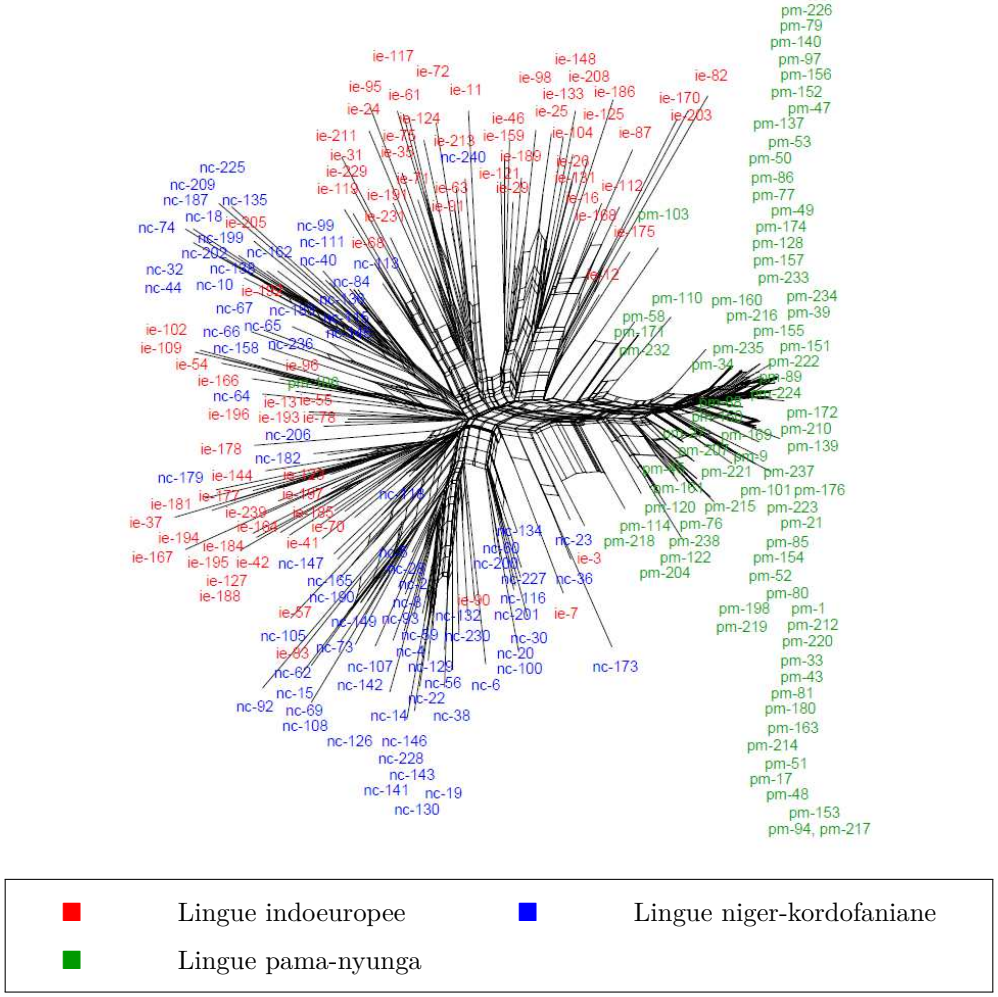

Figura 3.7: Rete filogenetica costruita a partire dai fonemi del dataset *unione*, colorata in base ad una di 3 famiglie linguistiche di appartenenza.

La rete filogenetica in Figura 3.7 è stata creata a partire dal dataset *famiglie*. Il raggruppamento effettuato da SplitsTree lascia qualcosa a desiderare per le lingue indoeuropee e le lingue niger-kordofaniane, che sembrano venire ripartite in due sottogruppi ciascuna, mentre risulta essere particolarmente buono per le lingue pama-nyunga. Queste ultime sono infatti tutte concentrate in un ramo isolato della rete: si tratta di un forte indicatore del fatto che esse sono geneticamente correlate fra di loro e che discendono da un antenato comune esistito piuttosto recentemente (meno di 5000 anni fa). Nella rete filogenetica in Figura 3.7 sono colorate in verde, come indicato in legenda, e si trovano nella parte a destra della rete. A fronte della densità di taxa

all'interno del ramo, le loro etichette sono impilate in verticale dal SplitsTree per mancanza di spazio.

3.3 Alberi di classificazione

I risultati ottenuti dalle analisi illustrate nelle sezioni 3.1 e 3.2 della presente tesi (*clustering* e reti filogenetiche, rispettivamente) suggeriscono che è possibile raggruppare le lingue del mondo in maniera sensata. Si è interessati a questo punto a valutare l'efficacia di una tecnica di apprendimento supervisionato che consenta di trarre dai dati a disposizione informazioni per classificare e fare previsioni riguardo a delle covariate di interesse, come ad esempio la famiglia linguistica di appartenenza o la macroarea in cui è parlata una lingua.
Per fare questo è necessario individuare una regola che consenta di classificare un'unità statistica in una di K categorie a disposizione, sfruttando le informazioni relative a p variabili esplicative rilevate. Per costruire tale regola di classificazione, si assume di conoscere la categoria di appartenenza per un insieme di n osservazioni.
Si procede adesso a definire un possibile approccio basato su una tecnica di tipo non parametrico, chiamata *albero di classificazione* (*classification tree*, abbreviato anche con CT). Si tratta di un modello predittivo che descrive una struttura ad albero dove i nodi terminali equivalgono alle classificazioni predette per la variabile risposta a partire dai valori delle variabili esplicative, e ogni nodo interno rappresenta una variabile in corrispondenza della quale è stato effettuato uno *split*, cioè una suddivisione di una regione in delle sotto-regioni. Per indicare un nodo terminale, si usa spesso il termine *foglia*.
Per definizione dunque, Y è una variabile casuale qualitativa categoriale con K livelli. Y funge da variabile risposta, dove ciascun suo livello rappresenta una classe di appartenenza. Si indicano con $y_1, ..., y_n$ le classi di appartenenza per gli elementi del campione, e con n_k il numero di unità possedute dalla k-esima classe, per $k = 1, ..., K$. Infine, si denota con $x = (x_1, ..., x_p)$ l'informazione relativa alle p variabili esplicative.
La probabilità che un'osservazione con caratteristiche x faccia parte della classe k è data dalla funzione $p_k(x) = \mathbb{P}\{Y = k|X = x\}$. Poiché Y ha K classi, $p(x)$ assume valori nel simplesso K-dimensionale, ossia i suoi valori sono probabilità $p_1(x), ..., p_K(x)$ che sommano a 1. L'idea alla base di un albero di classificazione è di approssimare $p_k(x)$ me-

diante una funzione a gradini, imponendo che le J suddivisioni dello spazio $x = (x_1, ..., x_p)$ avvengano con tagli paralleli agli assi coordinati. Così facendo richiediamo che le $R_1, ..., R_J$ regioni che si vanno a formare siano dei rettangoli p-dimensionali con i lati paralleli agli assi coordinati. L'approssimazione di $p_k(x)$ avviene tramite

$$\hat{p}_k(x) = \sum_{j=1}^{J} P_{jk} \cdot \mathbb{1}(x \in R_j),$$

dove P_{jk} rappresenta la probabilità che $Y = k$ nella regione R_j per $j = 1, ..., J$, mentre $\mathbb{1}(x \in R_j)$ è la funzione indicatrice della regione, tale che

$$\mathbb{1}(x \in R_j) = \begin{cases} 1, & \text{se } x \in R_j, \\ 0, & \text{altrimenti.} \end{cases}$$

Per stimare i P_{jk} si usa la media aritmetica

$$\hat{P}_{jk} = M(y_i = k : x_i \in R_j) = \frac{1}{n_{jk}} \cdot \sum_{i \in R_j} \mathbb{1}(y_i = k),$$

che altri non è che la frequenza relativa degli elementi appartenenti alla classe k e presenti nella regione R_j, ovvero per i quali $y_i = k$ e $x_i \in R_j$. Questo rende $\hat{P}_j$ un vettore K-dimensionale, i cui valori stimano le componenti $p_1(x), ..., p_K(x)$ [3].
Il modo con cui le osservazioni x_i si distribuiscono nelle regioni R_j è strettamente legato a eventuali previsioni che si vogliono fare riguardo a soggetti per i quali la classe di appartenenza sia ignota. Infatti, dalla definizione di $\hat{P}_{jk}$ deriva che per individuare la classe di assegnazione di un'osservazione per la quale non si conosce la classe di appartenenza, ci si affida a scegliere la classe che compare più frequentemente nella regione in cui si trova l'osservazione [18].
Per determinare le regioni R_j, occorre fissare un qualche criterio obiettivo. Il criterio di riferimento è la devianza, che viene minimizzata tramite un approccio subottimale di *ottimizzazione passo a passo*, nel senso che si costruisce una sequenza di approssimazioni via via più precise, e per ognuna di esse si minimizza la devianza relativa al passaggio dall'approssimazione attuale alla precedente. Pertanto, questa

è una procedura di *ottimizzazione miopica*. Pur non garantendo la minimizzazione globale della devianza, essa mantiene comunque una complessità computazionale ridotta [3].
Il processo di crescita dell'albero di classificazione inizia con $J = 1$ e $R_J = \mathbb{R}^p$, e procede iterativamente per un numero di cicli. A ogni passo, si divide in due una delle regioni R_j precedentemente costruite. In questa maniera, anche la porzione di dati che appartengono alla regione R_j viene divisa a metà. L'algoritmo di crescita di un albero di classificazione fa sì che il punto di suddivisione sia ricercato in corrispondenza della variabile che porta ad una riduzione maggiore della devianza. Di conseguenza, le variabili utilizzate per gli splits possono anche essere considerate le più importanti nel determinare la variabile risposta.
La devianza D è definita come

$$D = 2n \cdot \sum_{j=1}^{J} \frac{n_j}{n} \cdot Q(\hat{P}_j),$$

dove i termini $Q(\cdot)$ sono chiamati misure (o indici) di *impurità* in quanto indicano il grado con cui gli elementi di una certa foglia non sono omogenei rispetto alla variabile risposta. Si osservi che la devianza D, a meno della costante $2n$, è una media di indici di impurità pesati per la numerosità relativa delle foglie. Nella definizione originale della devianza, l'impurità viene quantificata usando come indice di impurità l'entropia

$$Q(P_j) = -\sum_{k=1}^{K} P_{jk} \cdot \log P_{jk},$$

ma essa può essere sostituita da altre misure di impurità. Fra le alternative possibili, una variante comune è l'indice di Gini

$$Q(P_j) = \sum_{k=1}^{K} P_{jk} \cdot (1 - P_{jk}).$$

È importante tenere a mente che se si lascia operare senza restrizione l'algoritmo di crescita, l'albero cresce fino a che $J = n$, ovvero fino

a quando non si ottiene un albero con n foglie. L'albero risultante farebbe quindi corrispondere una regione R_j ad ognuno degli n valori osservati di x. Per evitare questo si usano vari parametri di controllo che fanno in modo che la crescita non continui indefinitamente. Essi consistono nel fissare dei criteri che devono essere rispettati perché venga effettuato un nuovo split. I più usati sono il numero minimo di osservazioni necessarie in un nodo, il numero minimo di osservazioni necessarie in ciascun nodo terminale, il valore minimo di cui si deve ridurre la devianza.
Infine, quando si ha a disposizione un numero N di osservazioni sufficientemente grande, si è soliti fare crescere l'albero di classificazione a partire da un *insieme di addestramento* (anche chiamato *insieme di stima* o *training set*) costituito da $n < N$ osservazioni per le quali si conosce la classe di appartenenza; si procede poi a verificare la sua efficacia tramite un *insieme di verifica* (anche chiamato *test set*) formato da $N - n$ osservazioni, per i quali si assume che la classe di appartenenza sia ignota. Solitamente si sceglie n in modo tale che $\frac{n}{N} = 0.75$. L'uso di un insieme di verifica è una delle strategie usate per scongiurare il pericolo che emerga il problema del *sovradattamento* (*overfitting*). Esso si presenta quando le relazioni empiriche apprese sono dovute a fluttuazioni aleatorie che valgono solo nell'insieme di addestramento ma non in generale, per l'intera popolazione. Questo renderebbe il modello inadatto ad effettuare previsioni su nuovi dati. Dunque l'albero di classificazione va valutato su un insieme di verifica disgiunto dall'insieme di addestramento.

3.3.1 Metodo di costruzione

Ci si appresta a definire due applicazioni per gli alberi di classificazione. Nel primo caso la variabile di classificazione usata è la macroarea in cui è parlata una lingua, mentre nel secondo caso consiste nella famiglia linguistica di appartenenza.

Alberi per previsioni sulla macroarea

Il dataset che è stato costruito per implementare l'analisi si chiama *albero_macroarea*. Si tratta di una rielaborazione del dataset *unione*, ottenuta selezionando la variabile relativa alla `Macroarea` e le variabili che indicano la presenza o meno dei 100 fonemi più comuni a livello mondiale. La prima svolge il ruolo di variabile risposta, mentre le se-

conde costituiscono 100 variabili esplicative, ognuna delle quali è una variabile dicotomica che assume valore 1 se il fonema è presente nella lingua e 0 se è assente. Poiché non tutte le librerie di R supportano i caratteri dell'Alfabeto Fonetico Internazionale, i nomi delle variabili sono stati cambiati perché siano i numeri che vanno da 1 a 100. Il dataset *albero_macroarea* così ottenuto ha dunque 1325 righe e 101 colonne. In questo caso non si è ritenuto opportuno adattare un albero di classificazione al dataset *lingue* per via dell'esiguità del numero delle sue osservazioni (appena 30), la quale avrebbe reso ardua la creazione di un insieme di addestramento e di verifica a partire da esse.
La variabile risposta Y_1 per la quale si desiderano classificare le osservazioni del dataset *albero_macroarea* è pertanto definita come

$$Y_1 = \begin{cases} 1, & \text{se la lingua è parlata in Africa } (AF), \\ 2, & \text{se la lingua è parlata in Eurasia } (EA), \\ 3, & \text{se la lingua è parlata in America del Nord } (NA), \\ 4, & \text{se la lingua è parlata in America del Sud } (SA), \\ 5, & \text{se la lingua è parlata in Australia } (AU), \\ 6, & \text{se la lingua è parlata in Papunesia } (PA). \end{cases}$$

In Tabella 3.4 viene inoltre riportata la distribuzione di frequenza delle lingue per ognuna delle 6 classi, in ordine decrescente.

Macroarea	**Numero**
Africa	361
Eurasia	320
America del Sud	221
Australia	155
Papunesia	143
America del Nord	125

Tabella 3.4: Distribuzione di frequenza delle lingue per ciascun valore assunto dalla variabile *Macroarea* del dataset *unione*.

La parola Papunesia è una combinazione fra *Papua* (Nuova Guinea) e *Austronesia*, e si riferisce ad un'area geografica che include le isole del sud-est Asiatico e l'Oceania, escludendo l'Australia. La distinzione fatta fra Australia e Papunesia è dovuta alla considerevole diversità linguistica che è riscontrabile in queste regioni. In particolare, la Papua Nuova Guinea è la nazione con più lingue al mondo, con più di 820 lingue native, che rappresentano ben il 12% del totale a livello mondiale. Tuttavia la maggior parte di esse ha meno di 1000 parlanti [20].
Sono stati adattati due alberi di classificazione: uno usando come misura di impurità l'indice di Gini, e un altro ricorrendo all'entropia.
I parametri di controllo degli alberi sono stati stabiliti manualmente in maniera tale da evitare il sovradattamento del modello e garantire comunque una sufficiente precisione nei risultati.
Per realizzare le analisi sono state usate le librerie di R *tidyverse* [34, 41], *rsample* [35], *rpart* [37], *rattle* [42] e *yardstick* [28].

Alberi per previsioni sulla famiglia

In questa occasione si riprende in mano il dataset *famiglie*: difatti, il dataset *albero_famiglia* sul quale sono state condotte le analisi è una rielaborazione del dataset *famiglie*, ottenuta aggiungendo la variabile `Famiglia`, che rappresenta la famiglia linguistica di appartenenza. Essa funge da variabile risposta per la classificazione delle lingue di *albero_famiglia*, ed è definita come

$$Y_2 = \begin{cases} 1, & \text{se la lingua è indoeuropea } (IE), \\ 2, & \text{se la lingua è niger-kordofaniana } (NC), \\ 3, & \text{se la lingua è pama-nyunga } (PM). \end{cases}$$

In Tabella 3.5 viene riportata la distribuzione di frequenza delle lingue per ognuna delle 3 classi.

Famiglia	Numero
Indoeuropea	80
Niger-Kordofaniana	80
Pama-Nyunga	80

Tabella 3.5: Distribuzione di frequenza delle lingue del dataset *famiglie* per la famiglia linguistica di appartenenza.

Anche in questo caso si sono considerati due alberi di classificazione diversi: uno basato sull'indice di Gini, e un altro sull'entropia.

3.3.2 Risultati degli alberi di classificazione

Alberi per previsioni sulla macroarea

I risultati degli alberi di classificazione sono riportati nelle matrici di confusione in Tabella 3.6 e in Tabella 3.7, nelle quali vengono affiancate le risposte effettive con le previsioni fatte dall'albero. Le previsioni sono state fatte sull'insieme di verifica, di numerosità 332, pari a un quarto delle osservazioni a disposizione. Per indicare le macroaree presenti nelle tabelle seguenti, sono state usate le sigle già descritte nella definizione della variabile risposta Y_1. Le sigle sono state ordinate alfabeticamente, in maniera tale da rispecchiare l'ordine utilizzato da R.

Previsione	**Risposta effettiva**						
	AF	AU	EA	NA	PA	SA	Totale
AF	70	1	15	1	2	3	92
AU	0	38	0	0	0	0	38
EA	14	0	53	11	4	2	84
NA	1	0	1	10	0	0	12
PA	9	2	7	3	14	2	37
SA	2	0	3	11	8	45	69
Totale	96	41	79	36	28	52	332

Tabella 3.6: Matrice di confusione per l'albero di classificazione costruito usando come misura di impurità l'indice di Gini, e come variabile risposta la *macroarea*.

Previsione	**Risposta effettiva**						
	AF	AU	EA	NA	PA	SA	Totale
AF	66	1	17	0	1	3	88
AU	9	40	7	0	2	5	63
EA	14	0	50	10	3	2	79
NA	0	0	0	5	0	0	5
PA	2	0	1	3	13	2	21
SA	5	0	4	18	9	40	76
Totale	96	41	79	36	28	52	332

Tabella 3.7: Matrice di confusione per l'albero di classificazione costruito usando come misura di impurità l'entropia, e come variabile risposta la *macroarea*.

Le stime numeriche mostrate di seguito sono state approssimate alla seconda cifra decimale.
Il tasso di corretta classificazione è pari a 0.69 per l'albero costruito misurando l'impurità con l'indice di Gini, e a 0.64 per l'albero costruito usando l'entropia. Questo ci fa preferire il primo dei due, in quanto classifica correttamente una proporzione maggiore di osservazioni dell'insieme di verifica.
Si esamina ora il *recupero* (precisione), ovvero la proporzione di lingue di ciascuna macroarea che è stata classificata correttamente. Equivalentemente, si tratta della probabilità di classificare correttamente una lingua condizionatamente alla macroarea. Per ottenere questa misura si divide il numero di lingue classificate correttamente per il totale di colonna. I valori di precisione a cui si perviene per gli alberi basati sull'indice di Gini e sull'entropia sono simili fra di loro, come si può vedere in Tabella 3.8

Impurità	**Precisione**					
	AF	AU	EA	NA	PA	SA
Gini	0.73	0.93	0.67	0.28	0.5	0.87
Entropia	0.69	0.98	0.63	0.14	0.46	0.77

Tabella 3.8: Precisione ottenuta per gli alberi di classificazione costruiti usando come misure di impurità l'indice di Gini e l'entropia, e come variabile risposta la *macroarea*.

Le maggiori difficoltà nella classificazione si riscontrano per le lingue parlate in America del Nord e in Papunesia. In entrambi i casi la precisione è inferiore o uguale a 0.5, il che indica che la previsione restituita dall'albero di classificazione per le due classi è peggiore del lancio di una moneta. Le ragioni che rendono le lingue appartenenti a queste due macroaree così difficili da classificare sono molteplici; innanzitutto, va notato che come osservabile in Tabella 3.4, America del Nord e Papunesia costituiscono le classi meno numerose all'interno del campione considerato. Un numero limitato di lingue su cui fare allenare l'albero può causare una prestazione non ottimale nell'insieme di stima. Inoltre, vale la pena di osservare che l'America del Nord e la Papunesia sono due fra le regioni del mondo a maggiore diversità linguistica. In altre parole, l'elevata densità e numerosità di lingue

presente su di un territorio complica l'individuazione di variabili che possano *identificare* con certezza la macroarea.
Al contrario le altre macroaree mostrano una precisione nettamente più elevata. Salta subito all'occhio l'Australia, che esibisce un grado di precisione persino superiore al 90%. Il motivo di questo notevole risultato è da attribuirsi al fatto che le lingue parlate in Australia sono al contempo molto simili fra di loro dal punto di vista fonologico, ma anche diverse dal resto delle lingue del mondo.
Si passa infine ad esaminare le variabili che influiscono maggiormente sulla scelta della classe di assegnazione. La variabile che rappresenta la presenza del fonema /ɳ/ è uno dei principali strumenti che consentono all'albero di individuare le lingue appartenenti alla macroarea Australiana. Il simbolo ɳ rappresenta una consonante nasale alveolopalatale, un suono raro a livello mondiale ma particolarmente comune in Australia. La presenza dei fonemi /f/, /a/, /˥/ (un tono), /ɲ/ (come la **gn** di **gn**occhi) e /ɡb/ sembra essere invece caratteristica nelle lingue appartenenti alla macroarea africana. La conclusione è coerente con la realtà: i toni occorrono solitamente in lingue parlate in Africa e nell'estremo oriente, e il fonema /ɡb/ è specificamente frequente in Africa. D'altro canto, l'assenza dei fonemi /ɳ/, /f/, /ŋ/ e la presenza del fonema /ɾ/ (una monovibrante alveolare, simile se non identica alla erre italiana fra due vocali) suggeriscono che una lingua sia parlata in America del Sud.
Per tutte e tre le macroaree appena considerate, il percorso (*path*) dal nodo radice al nodo foglia è molto corto, poiché è formato al massimo da 4 split. Viceversa, il percorso che collega il nodo radice ai nodi foglia con le macroaree America del Nord e Papunesia è considerevolmente più lungo, e questo diminuisce la bontà delle previsioni effettuate su di esse.

Alberi per previsioni sulla famiglia

Le matrici di confusione in Tabella 3.9 e in Tabella 3.10 riportano i risultati degli alberi di classificazione per il dataset *albero_famiglia*. L'insieme di verifica usato ha numerosità uguale a 60, che corrisponde a un quarto di 240.

Previsione	**Risposta effettiva**			
	IE	NC	PM	Totale
IE	19	3	0	22
NC	1	17	1	19
PM	1	0	18	19
Totale	21	20	19	60

Tabella 3.9: Matrice di confusione per l'albero di classificazione costruito usando come misura di impurità l'indice di Gini, e come variabile risposta la *famiglia*.

Previsione	**Risposta effettiva**			
	IE	NC	PM	Totale
IE	17	4	0	21
NC	3	16	1	20
PM	1	0	18	19
Totale	21	20	19	60

Tabella 3.10: Matrice di confusione per l'albero di classificazione costruito usando come misura di impurità l'entropia, e come variabile risposta la *famiglia*.

Il tasso di corretta classificazione è pari a 0.90 per l'albero costruito misurando l'impurità con l'indice di Gini, e a 0.85 per l'albero costruito usando l'entropia. Ancora una volta, si è portati a preferire l'albero costruito misurando l'impurità tramite l'indice di Gini, in quanto classifica correttamente una proporzione leggermente maggiore di osservazioni dell'insieme di verifica.
Pure la precisione rispecchia un'ottima classificazione delle lingue rispetto alla famiglia linguistica di appartenenza, come si può vedere in Tabella 3.11.

Impurità	**Precisione**		
	IE	NC	PM
Gini	0.90	0.85	0.95
Entropia	0.81	0.80	0.95

Tabella 3.11: Precisione ottenuta per gli alberi di classificazione costruiti usando come misure di impurità l'indice di Gini e l'entropia, e come variabile risposta la *famiglia*.

Si noti come anche in questa occasione le lingue pama-nyunga vengono classificate correttamente oltre il 90% delle volte, analogamente a quanto avvenuto quanto si sono classificate le lingue usando come variabile di classificazione la macroarea. Questo risultato suggerisce che le lingue pama-nyunga formano una classificazione ben definita sia dal punto linguistico che territoriale. In generale la classificazione rispetto alla famiglia linguistica di appartenenza sembra fornire dei risultati migliori rispetto alla classificazione rispetto alla macroarea. Questo potrebbe essere dovuto al fatto che solitamente le lingue di una famiglia sono tutte collocate all'interno della stessa macroarea, mentre una macroarea contiene al suo interno decine di famiglie linguistiche diverse. Inoltre, la variabile `Macroarea` è costituita da 6 categorie, mentre la variabile `Famiglia` del dataset *albero_famiglia* ne possiede solo 3. Un numero maggiore di categorie comporta un numero maggiore di errori possibili. Data una variabile categoriale a K livelli, i tipi di errori che si possono commettere sono $K \cdot (K-1)$. Dunque mentre per `Macroarea` essi sono $6 \cdot (6-1) = 30$, per `Famiglia` ammontano a $3 \cdot (3-1) = 6$, esattamente 5 volte di meno.
Le variabili che influiscono maggiormente sulla scelta della classe di assegnazione sono le seguenti. L'assenza dei fonemi /b/ e /f/ consente di identificare una lingua come appartenente alla famiglia pama-nyunga. Al contrario se essi sono presenti assieme anche ai fonemi /kp/, /ɳ/ (un tono) e /ɓ/ (una consonante implosiva), allora la lingua in questione è niger-kordofaniana. Infine, la presenza di /b/, /f/ e /x/ (come il suono di **ch** nella parola tedesca Na**ch**t), e l'assenza di /kp/, /ɳ/ e /ɓ/, sono caratteristiche delle lingue indoeuropee.

Conclusioni

Nel corso della presente tesi sono stati esaminati alcuni approcci di analisi statistica aventi lo scopo di indagare le relazioni presenti tra le lingue.
Si è proceduto inizialmente a descrivere i datasets di riferimento, ottenuti rielaborando delle masse di dati reperite da repositories online. Le operazioni di creazione ed elaborazione dei datasets sono risultate particolarmente laboriose, in quanto è stato necessario combinare informazioni da più fonti diverse. Si è inoltre fornita una breve introduzione alle tematiche collegate allo studio delle lingue, con lo scopo di rendere il lettore più familiare con certi concetti specifici alla linguistica.
In seguito, sono state effettuate alcune analisi esplorative per investigare la distribuzione di una selezione di attributi nelle lingue del mondo. Questo ha permesso di verificare che spesso gli attributi si distribuiscono formando dei raggruppamenti, i quali possono dare vita a delle aree linguistiche costituite da lingue geograficamente vicine che si influenzano a vicenda. In aggiunta, si è confermato che lingue appartenenti alla stessa famiglia linguistica tendono ad avere attributi in comune e ad assomigliarsi.
Considerando uno dei datasets creati, si è fatto notare come i dati di tipo linguistico siano sovente piagati da un'alta frequenza di valori mancanti, i quali possono incidere pesantemente sulla qualità dei risultati ottenuti. Per far fronte a questa problematica, si è definita una procedura per la realizzazione di sottoinsiemi che siano privi di valori mancanti. I datasets così creati sono formati solamente da osservazioni complete, tuttavia è presente anche una considerevole riduzione nella numerosità delle unità statistiche.
In seguito si è focalizzata l'attenzione sull'impiego di tecniche atte a raggruppare e classificare le lingue del mondo. Come prima cosa sono state implementate delle tecniche di *clustering*. Sono stati presi in considerazioni metodi gerarchici e metodi di partizione. Dai risultati

emerge che i raggruppamenti ottenuti in tale maniera rispecchiano le varie famiglie linguistiche di appartenenza delle lingue.
In seguito si è delineato un possibile approccio basato su delle reti filogenetiche. Questo è stato fatto anche in vista della nozione che tali modelli possono descrivere opportunamente non solo le relazioni che esistono fra lingue geneticamente correlate, ma pure fra lingue che sono soltanto geograficamente vicine e che possono fare parte della stessa area linguistica. Da questo punto di vista le reti filogenetiche costituiscono una valida alternativa ai classici alberi filogenetici, i quali richiedono il vincolo di discendenza da un antenato comune. I risultati confermano che le reti filogenetiche sono un modello adatto alla rappresentazione delle lingue, e che il fare parte della stessa area linguistica gioca un ruolo equamente rilevante all'appartenenza alla medesima famiglia linguistica, per quanto riguarda la formazione di raggruppamenti fra lingue. In questo senso, potrebbe essere interessante introdurre più strumenti di natura statistica per valutare la bontà delle reti filogenetiche ottenute, nonché condurre alcune verifiche d'ipotesi.
Arrivati a questo punto, ci si è concentrati sull'uso di alcuni alberi di classificazione con il fine di fare previsioni riguardo alla macroarea e alla famiglia di una lingua, a partire dalle informazioni relative ai suoi fonemi. Per verificare la bontà degli alberi costruiti sull'insieme di addestramento, si è finto fatto uso di un insieme di verifica. Dai risultati si evince che certi fonemi sono notevolmente più importanti per effettuare previsioni rispetto ad altri. In aggiunta, mentre per certe macroaree fare previsioni tramite l'albero risulta essere molto conveniente, per altre macroaree le previsioni ottenute mediante l'albero sono decisamente peggiori. Per provare a ridurre l'errore di classificazione potrebbe essere opportuno a questo proposito adattare una *random forest*.
In conclusione, si è constatato come i dati a disposizione consentano di descrivere in modo apprezzabile le relazioni fra le lingue. Tuttavia, occorre tenere a mente che, per quanto potrebbe piacere, le lingue non sono oggetti immutabili e immuni dai cambiamenti. Le loro storie evolutive sono determinate da interazioni reciproche costanti e da una miriade di altri fattori. Una futura modellizzazione potrebbe concentrarsi proprio sull'integrare più modi per tenere conto delle influenze che le lingue hanno le une sulle altre.

Bibliografia

[1] Asher R.; Moseley C.; Darkes G. (2007). *Atlas of the World's Languages* in Atlas of the world's languages. Numero Bd. 10. Routledge.

[2] Association I. P. (1999). *Handbook of the International Phonetic Association: A guide to the use of the International Phonetic Alphabet*. Cambridge University Press.

[3] Azzalini A.; Scarpa B. (2012). *Data analysis and data mining: An introduction*. OUP USA.

[4] Becker R. A.; Wilks A. R.; Brownrigg R.; Minka T. P.; Deckmyn A. (2021). *maps: Draw Geographical Maps*. Versione pacchetto R 3.4.0.

[5] Bryant D.; Moulton V. (2002). Neighbornet: an agglomerative method for the construction of planar phylogenetic networks. In *International workshop on algorithms in bioinformatics*, pp. 375–391. Springer.

[6] Bryant D.; Moulton V. (2004). Neighbor-net: an agglomerative method for the construction of phylogenetic networks. *Molecular biology and evolution*, **21**(2), 255–265.

[7] Chang W.; Hall D.; Cathcart C.; Garrett A. (2015). Ancestry-constrained phylogenetic analysis supports the indo-european steppe hypothesis. *Language*, pp. 194–244.

[8] Cheng J.; Karambelkar B.; Xie Y. (2022). *leaflet: Create Interactive Web Maps with the JavaScript 'Leaflet' Library*. Versione pacchetto R 2.1.0.

[9] Comrie B.; Dryer M. S.; Gil D.; Haspelmath M. (2013). Introduction In *The World Atlas of Language Structures Online*. A cura di Dryer M. S., Haspelmath M. Max Planck Institute for Evolutionary Anthropology, Leipzig.

[10] Cugno F.; Rivoira M.; Ronco G. (2020). L'atlante linguistico italiano (ali). *Romance Philology*, **74**(2).

[11] Dagan T.; Martin W. (2006). The tree of one percent. *Genome biology*, **7**(10), 1–7.

[12] Dryer M. S. (2013a). Coding of nominal plurality. In *The World Atlas of Language Structures Online*. A cura di Dryer M. S., Haspelmath M. Max Planck Institute for Evolutionary Anthropology, Leipzig.

[13] Dryer M. S. (2013b). Definite articles. In *The World Atlas of Language Structures Online*. A cura di Dryer M. S., Haspelmath M. Max Planck Institute for Evolutionary Anthropology, Leipzig.

[14] Dryer M. S. (2013c). Order of object and verb. In *The World Atlas of Language Structures Online*. A cura di Dryer M. S., Haspelmath M. Max Planck Institute for Evolutionary Anthropology, Leipzig.

[15] Dryer M. S.; Haspelmath M., (A cura di) (2013). *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

[16] Dum-Tragut J. (2009). Armenian. *Armenian*, pp. 1–758.

[17] Edmondson J.; A J. (2011). Notes on the subdivisions in kra. *Journal of the Guangxi University of Nationalities*, **14**, 8–14.

[18] Gareth J.; Witten D.; Hastie T.; Tibshirani R. (2013). *An introduction to statistical learning: with applications in R*. Springer.

[19] Genetti C. (2018). *How languages work: An introduction to language and linguistics*. Cambridge University Press.

[20] Hammarström H.; Forkel R.; Haspelmath M.; Bank S., (A cura di) (2022). *Glottolog 4.6*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

[21] Haspelmath M. (2009). The typological database of the world atlas of language structures. *The Use of Databases in Cross-Linguistic Studies*, p. 283.

[22] Hayasaka K.; Gojobori T.; Horai S. (1988). Molecular phylogeny and evolution of primate mitochondrial dna. *Molecular Biology and Evolution*, **5**(6), 626–644.

[23] Horikoshi M.; Tang Y. (2018). *ggfortify: Data Visualization Tools for Statistical Analysis Results*.

[24] Huson D. H. (2005). Ismb-tutorial: Introduction to phylogenetic networks. *Center for Bioinformatics, Tubingen University, Sand*, **14**, 72075.

[25] Huson D. H.; Bryant D. (2006). Application of phylogenetic networks in evolutionary studies. *Molecular biology and evolution*, **23**(2), 254–267.

[26] Huson D. H.; Scornavacca C. (2011). A survey of combinatorial methods for phylogenetic networks. *Genome biology and evolution*, **3**, 23–35.

[27] Johnson R. A.; Wichern D. W. (2007). *Applied Multivariate Statistical Analysis*. Pearson Education. Sesta edizione.

[28] Kuhn M.; Vaughan D. (2021). *yardstick: Tidy Characterizations of Model Performance*. Versione pacchetto R 0.0.9.

[29] Lewis P. M. (2009). *Ethnologue: Languages of the World*. SIL International, Dallas, 16ª edizione.

[30] Maddison D. R.; Swofford D. L.; Maddison W. P. (1997). Nexus: an extensible file format for systematic information. *Systematic biology*, **46**(4), 590–621.

[31] Maechler M.; Rousseeuw P.; Struyf A.; Hubert M.; Hornik K. (2022). *cluster: Cluster Analysis Basics and Extensions*. Versione pacchetto R 2.1.3.

[32] Moran S.; McCloy D., (A cura di) (2019). *PHOIBLE 2.0*. Max Planck Institute for the Science of Human History, Jena.

[33] Moseley C.; Asher R. (1994). *Atlas of the World's languages*. Routledge, London.

[34] R Core Team (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

[35] Silge J.; Chow F.; Kuhn M.; Wickham H. (2021). *rsample: General Resampling Infrastructure*. Versione pacchetto R 0.1.1.

[36] Simone R. (1990). *Fondamenti di linguistica*. Numero 9. Laterza Bari.

[37] Therneau T.; Atkinson B. (2022). *rpart: Recursive Partitioning and Regression Trees*. Versione pacchetto R 4.1.16.

[38] Underhill R. (1976). *Turkish Grammar*. Massachusetts Institute of Technology (MIT) Press, Cambridge.

[39] Vaidyanathan R.; Xie Y.; Allaire J.; Cheng J.; Sievert C.; Russell K. (2021). *htmlwidgets: HTML Widgets for R*. Versione pacchetto R 1.5.4.

[40] Wickham H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. Versione pacchetto R 3.3.6.

[41] Wickham H.; Averick M.; Bryan J.; Chang W.; McGowan L. D.; François R.; Grolemund G.; Hayes A.; Henry L.; Hester J.; Kuhn M.; Pedersen T. L.; Miller E.; Bache S. M.; Müller K.; Ooms J.; Robinson D.; Seidel D. P.; Spinu V.; Takahashi K.; Vaughan D.; Wilke C.; Woo K.; Yutani H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, **4**(43), 1686.

[42] Williams G. J. (2011). *Data Mining with Rattle and R: The art of excavating data for knowledge discovery*. Use R! Springer.

Sitografia

Pagina principale del World Atlas of Language Structures (WALS)
https://wals.info/

Pagina principale di PHOIBLE
https://phoible.org/

Pagina github da cui si possono scaricare i datasets di WALS
https://github.com/cldf-datasets/wals

Pagina github da cui si possono scaricare i datasets di PHOIBLE
https://github.com/phoible/dev

Cross-Linguistic Data Formats
https://cldf.clld.org/

Sito ufficiale dello standard ISO 639-3 per la nomenclatura delle lingue
https://iso639-3.sil.org/

Fonte dell'albero linguistico della famiglia delle lingue indoeuropee usato nella sezione 3.1
https://commons.wikimedia.org/wiki/
File:IndoEuropeanLanguageFamilyRelationsChart.jpg

Appendice A

Appendice: dataset *lingue*

Lista dei nomi delle lingue selezionate per il dataset *lingue*:

1. Arabo-egiziano,
2. Birmano,
3. Basco,
4. Burushaski,
5. Cinese,
6. Ciukcio,
7. Finlandese,
8. Francese,
9. Grebo,
10. Greco,
11. Groenlandese,
12. Hausa,
13. Indonesiano,
14. Inglese,
15. Kannada,

16. Lango,
17. Lesgo,
18. Malgascio,
19. Mixteco,
20. Nama,
21. Oromo,
22. Persiano,
23. Russo,
24. Spagnolo,
25. Supyire,
26. Swahili,
27. Tedesco,
28. Tiwi,
29. Turco,
30. Vietnamita.

Lista delle caratteristiche selezionate per la tabella *lingue*:

1. 1A.Consonant.Inventories,
2. 2A.Vowel.Quality.Inventories,
3. 3A.Consonant-Vowel.Ratio,
4. 4A.Voicing.in.Plosives.and.Fricatives,
5. 5A.Voicing.and.Gaps.in.Plosive.Systems,
6. 6A.Uvular.Consonants,
7. 7A.Glottalized.Consonants,
8. 8A.Lateral.Consonants,
9. 9A.The.Velar.Nasal,
10. 10A.Vowel.Nasalization,
11. 11A.Front.Rounded.Vowels,
12. 12A.Syllable.Structure,
13. 13A.Tone,
14. 18A.Absence.of.Common.Consonants,
15. 19A.Presence.of.Uncommon.Consonants,
16. 20A.Fusion.of.Selected.Inflectional.Formatives,
17. 21A.Exponence.of.Selected.Inflectional.Formatives,
18. 21B.Exponence.of.Tense-Aspect-Mood.Inflection,
19. 22A.Inflectional.Synthesis.of.the.Verb,
20. 23A.Locus.of.Marking.in.the.Clause,
21. 24A.Locus.of.Marking.in.Possessive.Noun.Phrases,
22. 25A.Locus.of.Marking:.Whole-language.Typology,
23. 25B.Zero.Marking.of.A.and.P.Arguments,

24. 26A.Prefixing.vs..Suffixing.in.Inflectional.Morphology,
25. 28A.Case.Syncretism,
26. 29A.Syncretism.in.Verbal.Person/Number.Marking,
27. 30A.Number.of.Genders,
28. 31A.Sex-based.and.Non-sex-based.Gender.Systems,
29. 32A.Systems.of.Gender.Assignment,
30. 33A.Coding.of.Nominal.Plurality,
31. 35A.Plurality.in.Independent.Personal.Pronouns,
32. 39A.Inclusive/Exclusive.Distinction.in.Independent.Pronouns,
33. 40A.Inclusive/Exclusive.Distinction.in.Verbal.Inflection,
34. 44A.Gender.Distinctions.in.Independent.Personal.Pronouns,
35. 45A.Politeness.Distinctions.in.Pronouns,
36. 48A.Person.Marking.on.Adpositions,
37. 49A.Number.of.Cases,
38. 50A.Asymmetrical.Case-Marking,
39. 51A.Position.of.Case.Affixes,
40. 58A.Obligatory.Possessive.Inflection,
41. 58B.Number.of.Possessive.Nouns,
42. 59A.Possessive.Classification,
43. 65A.Perfective/Imperfective.Aspect,
44. 66A.The.Past.Tense,
45. 67A.The.Future.Tense,
46. 68A.The.Perfect,
47. 69A.Position.of.Tense-Aspect.Affixes,

48. 70A.The.Morphological.Imperative,
49. 71A.The.Prohibitive,
50. 72A.Imperative-Hortative.Systems,
51. 73A.The.Optative,
52. 75A.Epistemic.Possibility,
53. 77A.Semantic.Distinctions.of.Evidentiality,
54. 78A.Coding.of.Evidentiality,
55. 79A.Suppletion.According.to.Tense.and.Aspect,
56. 79B.Suppletion.in.Imperatives.and.Hortatives,
57. 80A.Verbal.Number.and.Suppletion,
58. 81A.Order.of.Subject,.Object.and.Verb,
59. 82A.Order.of.Subject.and.Verb,
60. 83A.Order.of.Object.and.Verb,
61. 85A.Order.of.Adposition.and.Noun.Phrase,
62. 86A.Order.of.Genitive.and.Noun,
63. 87A.Order.of.Adjective.and.Noun,
64. 88A.Order.of.Demonstrative.and.Noun,
65. 89A.Order.of.Numeral.and.Noun,
66. 90A.Order.of.Relative.Clause.and.Noun,
67. 92A.Position.of.Polar.Question.Particles,
68. 93A.Position.of.Interrogative.Phrases.in.Content.Questions,
69. 95A.Relationship.between.the.Order.of.Object.and.Verb. and.the.Order.of.Adposition.and.Noun.Phrase,
70. 96A.Relationship.between.the.Order.of.Object.and.Verb. and.the.Order.of.Relative.Clause.and.Noun,

71. 97A.Relationship.between.the.Order.of.Object.and.Verb.
    and.the.Order.of.Adjective.and.Noun,
72. 98A.Alignment.of.Case.Marking.of.Full.Noun.Phrases,
73. 99A.Alignment.of.Case.Marking.of.Pronouns,
74. 100A.Alignment.of.Verbal.Person.Marking,
75. 101A.Expression.of.Pronominal.Subjects,
76. 102A.Verbal.Person.Marking,
77. 103A.Third.Person.Zero.of.Verbal.Person.Marking,
78. 104A.Order.of.Person.Markers.on.the.Verb,
79. 107A.Passive.Constructions,
80. 108A.Antipassive.Constructions,
81. 108B.Productivity.of.the.Antipassive.Construction,
82. 109A.Applicative.Constructions,
83. 109B.Other.Roles.of.Applied.Objects,
84. 111A.Nonperiphrastic.Causative.Constructions,
85. 112A.Negative.Morphemes,
86. 113A.Symmetric.and.Asymmetric.Standard.Negation,
87. 114A.Subtypes.of.Asymmetric.Standard.Negation,
88. 116A.Polar.Questions,
89. 136A.M-T.Pronouns,
90. 136B.M.in.First.Person.Singular,
91. 137A.N-M.Pronouns,
92. 137B.M.in.Second.Person.Singular,
93. 143A.Order.of.Negative.Morpheme.and.Verb,

94. 143E.Preverbal.Negative.Morphemes,

95. 143F.Postverbal.Negative.Morphemes,

96. 143G.Minor.morphological.means.of.signaling.negation,

97. 144A.Position.of.Negative.Word.With.Respect.to.Subject,. Object,.and.Verb.

Appendice B

Appendice: esempio di input per SplitsTree

```
#nexus

BEGIN Taxa;
DIMENSIONS ntax=30;
TAXLABELS

[1] 'Arabo-egiziano'
[2] 'Birmano'
[3] 'Basco'
[4] 'Burushaski'
[5] 'Cinese'
[6] 'Ciukcio'
[7] 'Finlandese'
[8] 'Francese'
[9] 'Grebo'
[10] 'Greco'
[11] 'Groenlandese'
[12] 'Hausa'
[13] 'Indonesiano'
[14] 'Inglese'
[15] 'Kannada'
[16] 'Lango'
[17] 'Lesgo'
[18] 'Malgascio'
[19] 'Mixteco'
[20] 'Nama'
```

```
[21] 'Oromo'
[22] 'Persiano'
[23] 'Russo'
[24] 'Spagnolo'
[25] 'Supyire'
[26] 'Swahili'
[27] 'Tedesco'
[28] 'Tiwi'
[29] 'Turco'
[30] 'Vietnamita'

;

END;[Taxa]

BEGIN Unaligned;
DIMENSIONS ntax=30;
FORMAT


       datatype=STANDARD
       missing=?
       symbols="0 1 2 3 4 5 6 7 8 9"

labels=left
;

MATRIX
'Arabo-egiziano'     0100010000001000101001000100 10,
'Birmano'            1000000101000100010000100 10010,
'Basco'              0010010000001010000101001001 00,
'Burushaski'         0001010000001010000101000100 10,
'Cinese'             1000000101000100010001000100 10,
'Ciukcio'            0001010000001010001000101010 00,
'Finlandese'         0100000010100000110001000100 10,
'Francese'           0100100000100010010001000100 10,
'Grebo'              1000001001000100010001000100 10,
'Greco'              0100010000001010000101000100 10,
'Groenlandese'       0100010000001010001001001010 00,
'Hausa'              1000001001000100010001000100 10,
```

```
'Indonesiano'        01001000000100100100010010010,
'Inglese'            01001000001000100100010010010,
'Kannada'            01000100001000100100010010011,
'Lango'              01000100000010010010011000100,
'Lesgo'              10000001010001000100010010010,
'Malgascio'          10001000010001000100010010010,
'Mixteco'            01000000101000100100010010010,
'Nama'               01001000000101000100010010010,
'Oromo'              01000100001000010100010010010,
'Persiano'           01000100000010001010010010010,
'Russo'              01001000001000100100010010010,
'Spagnolo'           01000100000010100010100010010,
'Supyire'            10001000010001000100010010010,
'Swahili'            01000100000010100010010010010,
'Tedesco'            01001000001000100100010010010,
'Tiwi'               01000100000010100000101010010,
'Turco'              01000100001000001100010010010,
'Vietnamita'         10000001010001000100010010010;
END;[Unaligned]
```

Appendice C

Appendice: codice R

In questa Appendice viene riportato il codice R che è stato utilizzato per svolgere le analisi descritte nei capitoli precedenti.

```r
# caricamento dei datasets
languages <- read.delim("languages.csv", header = TRUE, sep = ",",
                        encoding = "UTF-8")
values <- read.csv("values.csv", header = TRUE)
parameters <- read.csv("parameters.csv", header = TRUE)
codes <- read.delim("codes.csv", header = TRUE, sep = ",",
                    encoding = "UTF-8")

# creazione del dataset valori
valori <- values[1:5]
dim(valori) # 76475 per 5

# creazione delle prime 10 variabili del dataset caratteristiche
caratteristiche <- data.frame(languages[c(1, 7, 6, 2, 4, 5, 10, 8,
                                          3, 15)])
colnames(caratteristiche) <- c("Wals_code", "Iso_code", "Glottocode",
                               "Nome", "Latitudine", "Longitudine",
                               "Genus", "Famiglia", "Macroarea",
                               "Nazione")

# creazione delle altre 192 variabili del dataset caratteristiche
valori.largo <- reshape(data = valori[c(2, 3, 5)],
                        idvar = "Language_ID",
                        timevar = "Parameter_ID",
                        direction = "wide")
colnames(valori.largo) <- gsub("Code_ID.", "", colnames(valori.largo))
valori.largo <- valori.largo[order(match(colnames(valori.largo),
                                         parameters$ID))]
valori.largo <- valori.largo[c(ncol(valori.largo),
                               1:(ncol(valori.largo)-1))]
colnames(valori.largo) <- c(colnames(valori.largo[1]),
                      paste0(colnames(valori.largo[-1]), ".",
                             gsub(" ", ".", parameters$Name)))
for(i in 1:nrow(codes)) {
  valori.largo[valori.largo == codes$ID[i]] <- codes$Name[i]
}

```

```r
# rimozione delle lingue Fiote e Jiarong
languages$ID[!(languages$ID %in% unique(valori$Language_ID))]
vuoto <- which(!(languages$ID %in% unique(valori$Language_ID)))
caratteristiche <- cbind(caratteristiche[-vuoto,], valori.largo[-1])

dim(caratteristiche) # 2660 lingue per 192 caratteristiche
```

Codice C.1: Creazione dei datasets *caratteristiche* e *valori*.

```r
# caricamento del dataset
phoible <- read.delim("phoible.txt", header = TRUE, sep = ",",
                      encoding = "UTF-8")
dim(phoible)

# Glottocode non è un codice univoco
length(unique(phoible$InventoryID))
length(unique(phoible$Glottocode))

# rimozione delle lingue con Glottocode duplicato
pho <- phoible[phoible$InventoryID %in%
                 phoible$InventoryID[!duplicated(
                 phoible$Glottocode)], ]

# 100 fonemi più frequenti a livello mondiale
suoni <- head(sort(table(pho$Phoneme), decreasing = TRUE), 100)
suoni
# simboli IPA per i 100 fonemi
nomi_suoni <- names(suoni)
nomi_suoni
# frequenza relativa dei 100 suoni più comuni a livello mondiale
round(suoni/2176, 3)

# rimozione dei fonemi che non sono fra i 100 più comuni
pho.1 <- pho[pho$Phoneme %in% nomi_suoni,]

# trasformazione da formato lungo in formato largo
pho.1.largo <- as.matrix(xtabs(~Glottocode + Phoneme,
                               pho.1, sparse = TRUE))
pho.1.largo <- pho.1.largo[, order(match(colnames(pho.1.largo), nomi_
    suoni))]

# codici Glottocode delle lingue in comune fra i datasets
    caratteristiche e pho.1.largo
gc <- rownames(pho.1.largo)[rownames(pho.1.largo) %in%
                      caratteristiche$Glottocode]

# creazione preliminare del dataset fonemi
fonemi <- pho.1.largo[rownames(pho.1.largo) %in% gc, ]

# individuazione delle lingue con Glottocode duplicati nel dataset
    caratteristiche
caratteristiche.1 <- caratteristiche[caratteristiche$Glottocode %in%
gc,]
rownames(caratteristiche.1) <- 1:nrow(caratteristiche.1)
dupl <- unique(caratteristiche.1$Glottocode[
  duplicated(caratteristiche.1$Glottocode)])

```

```r
# individuazione delle lingue duplicate con meno valori mancanti
meno_NA <- 0
for(i in 1:length(dupl)) {
  meno_NA[i] <- names(which.min(rowSums(is.na(
    caratteristiche.1[caratteristiche.1$Glottocode == dupl[i], ]))))
}
caratteristiche.1[meno_NA, 4] # lingue che vengono mantenute
gruppo_di_interesse <- which(caratteristiche.1$Glottocode %in% dupl)
length(gruppo_di_interesse)

# uguale a caratteristiche.1 ma senza i Glottocode duplicati
caratteristiche.2 <- caratteristiche.1[-c(gruppo_di_interesse[
!(gruppo_di_interesse %in% meno_NA)]), ]

# creazione definitiva del dataset fonemi
fonemi <- fonemi[order(match(rownames(fonemi), caratteristiche.2$
    Glottocode)), ]
fonemi <- as.data.frame(cbind(caratteristiche.2$Nome, fonemi))
colnames(fonemi) <- c("Nome", colnames(fonemi)[-1])
rownames(fonemi) <- caratteristiche.2$Wals_code
dim(fonemi) # 1325 per 101

# creazione del dataset unione
unione <- cbind(caratteristiche.2, fonemi[-1])
dim(unione) # 1325 per 302

# creazione del dataset famiglie
ie <- unione$Nome[which(unione$Famiglia == "Indo-European")]
nc <- unione$Nome[which(unione$Famiglia == "Niger-Congo")]
pm <- unione$Nome[which(unione$Famiglia == "Pama-Nyungan")]

p <- 80
set.seed(777)
lingue_famiglie <- c(ie[sample(length(ie), p)],
                     nc[sample(length(nc), p)],
                     pm[sample(length(pm), p)])

condizione <- which(unione$Nome %in% lingue_famiglie)
famiglie <- unione[condizione, 203:ncol(unione)]

famiglie <- as.data.frame(sapply(as.data.frame(famiglie), as.numeric))
rownames(famiglie) <- unione$Nome[condizione]
dim(famiglie) # 240 per 100
```

Codice C.2: Creazione dei datasets *fonemi*, *unione* e *famiglie*

Il codice R riportato di seguito consente di creare quattro files html interattivi, relativi alle mappe illustrate nel Capitolo 2 della presente tesi.

```r
### Ordine del complemento oggetto e del verbo

table(caratteristiche$`83A.Order.of.Object.and.Verb`)

variabile.1 <- data.frame(na.omit(caratteristiche[c(4, 5, 6, 100)]))
colnames(variabile.1) <- c("Nome", "Latitudine", "Longitudine",
                           "Valori")
```

```
variabile.1$Valori <- as.factor(variabile.1$Valori)
levels(variabile.1$Valori) <- c("Nessun ordine dominante", "OV", "VO")

# mappa interattiva
library(leaflet)

colori.1 <- colorFactor(c("#F8766D", "#00BA38", "#619CFF"),
                        domain = levels(variabile.1$Valori))

mappa.1 <- leaflet(variabile.1) %>%
  addTiles() %>%
  addCircleMarkers(lng = ~Longitudine, lat = ~Latitudine,
                   label = ~paste0("Nome: ", Nome, "  Valore: ",
                                   Valori),
                   color = ~colori.1(Valori),
                   radius = 10,
                   stroke = FALSE, fillOpacity = 0.5)
mappa.1

# salvataggio della mappa come un html
library(htmlwidgets)
saveWidget(mappa.1, "ordine del complemento oggetto e del verbo.html",
    selfcontained = TRUE)

# mappa statica
library(maps)
library(ggplot2)

mondo <- map_data("world")

# plot di base con ggplot2
base <- ggplot() + coord_fixed() +
  xlab("") + ylab("")

# aggiunta della mappa al plot di base
base_mondo <- base + geom_polygon(data=mondo, aes(x=long, y=lat,
                                                  group=group),
                                  colour="gray", fill="white")

# pulizia della mappa e rimozione di elementi indesiderati
pulizia <-
  theme_bw(base_size=15) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_rect(fill = 'light blue',
                                        colour = 'light blue'),
        axis.line = element_line(colour = "white"),
        legend.position="top",
        axis.ticks=element_blank(),
        axis.text.x=element_blank(),
        axis.text.y=element_blank())

base_mappa <- base_mondo + pulizia

# aggiunta di punti alla mappa con colori dipendenti dai valori
mappa_colorata.1 <-
```

```
  base_mappa +
  geom_point(data=variabile.1,
             aes(x=Longitudine, y=Latitudine, colour=Valori), size=4,
    alpha=I(0.7))

mappa_colorata.1


### Articoli determinativi

table(caratteristiche$`37A.Definite.Articles`)

variabile.2 <- data.frame(na.omit(caratteristiche[c(4, 5, 6, 50)]))
colnames(variabile.2) <- c("Nome", "Latitudine", "Longitudine",
                           "Valori")
variabile.2$Valori <- as.factor(variabile.2$Valori)
levels(variabile.2$Valori) <- c("Suffisso determinativo", "Articolo
    determinativo",
                                "Aggettivo dimostrativo",
                                "Nessun articolo determinativo o
    indeterminativo",
                                "Articolo indeterminativo ma non
    determinativo")

# mappa interattiva

colori.2 <- colorFactor(c("#00BF7D", "#A3A500", "#E76BF3", "#00B0F6", "
    #F8766D"),
                        domain = levels(variabile.2$Valori))

mappa.2 <- leaflet(variabile.2) %>%
  addTiles() %>%
  addCircleMarkers(lng = ~Longitudine, lat = ~Latitudine,
                   label = ~paste0("Nome: ", Nome, "  Valore: ",
                                   Valori),
                   color = ~colori.2(Valori),
                   radius = 10,
                   stroke = FALSE, fillOpacity = 0.5)
mappa.2

# salvataggio della mappa come un html
saveWidget(mappa.2, "articoli determinativi.html", selfcontained = TRUE
    )

# mappa statica

# pulizia della mappa e rimozione di elementi indesiderati
pulizia.2 <-
  theme_bw(base_size=11) +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_
    blank(),
        panel.background = element_rect(fill = 'light blue', colour = '
    light blue'),
        axis.line = element_line(colour = "white"), legend.position="
    top",
        axis.ticks=element_blank(), axis.text.x=element_blank(),
```

```
        axis.text.y=element_blank())

base_mappa.2 <- base_mondo + pulizia.2

# aggiunta di punti alla mappa con colori dipendenti dai valori
mappa_colorata.2 <-
  base_mappa.2 +
  geom_point(data=variabile.2,
             aes(x=Longitudine, y=Latitudine, colour=Valori), size=4,
    alpha=I(0.7))

mappa_colorata.2


### Codifica dei plurali nei nomi

table(caratteristiche$`33A.Coding.of.Nominal.Plurality`)

variabile.3 <- data.frame(na.omit(caratteristiche[c(4, 5, 6, 46)]))
colnames(variabile.3) <- c("Nome", "Latitudine", "Longitudine",
                           "Valori")
variabile.3$Valori <- as.factor(variabile.3$Valori)
levels(variabile.3$Valori) <- c("Misto", "Nessun plurale", "Clitico",
                                "Raddoppiamento", "Prefisso", "Mutamento
    di radice",
                                "Suffisso", "Tono", "Parola")

# mappa interattiva

colori.3 <- colorFactor(c("#93AA00", "#F8766D", "#00B9E3", "#D39200", "
    #DB72FB", "#00C19F", "#00BA38", "#619CFF", "#FF61C3"),
  #c("yellow", "green", "purple", "gray", "navy", "orange",
                        #"dark green", "red", "black"),
                      domain = levels(variabile.3$Valori))

mappa.3 <- leaflet(variabile.3) %>%
  addTiles() %>%
  addCircleMarkers(lng = ~Longitudine, lat = ~Latitudine,
                   label = ~paste0("Nome: ", Nome, "  Valore: ",
                                   Valori),
                   color = ~colori.3(Valori),
                   radius = 10,
                   stroke = FALSE, fillOpacity = 0.5)
mappa.3

# salvataggio della mappa come un html
saveWidget(mappa.3, "codifica dei plurali nei nomi.html", selfcontained
    = TRUE)

# mappa statica

# aggiunta di punti alla mappa con colori dipendenti dai valori
mappa_colorata.3 <-
  base_mappa +
  geom_point(data=variabile.3,
             aes(x=Longitudine, y=Latitudine, colour=Valori), size=4,
```

```
    alpha=I(0.7)) +
  scale_color_manual(values = c("#F8766D", "#D39200", "#93AA00", "#00
    BA38",
                                "#00C19F", "#00B9E3", "#619CFF", "#
    DB72FB", "#FF61C3"))

mappa_colorata.3


### Distribuzione dei fonemi nel mondo

table(fonemi[,15], fonemi[,17])
s1 <- 216
s2 <- 218
variabile.4 <- data.frame(cbind(na.omit(unione[c(4, 5, 6)]),
                                rep(NA, nrow(unione))))
colnames(variabile.4) <- c("Nome", "Latitudine", "Longitudine",
                           "Valori")
variabile.4$Valori[c(unione[s1] == 1 & unione[s2] == 1)] <- "Sia /e/
    che /o/"
variabile.4$Valori[c(unione[s1] == 1 & unione[s2] == 0)] <- "Solo /e/"
variabile.4$Valori[c(unione[s1] == 0 & unione[s2] == 1)] <- "Solo /o/"
variabile.4$Valori[c(unione[s1] == 0 & unione[s2] == 0)] <- "Nessuno
    dei due"

variabile.4$Valori <- as.factor(variabile.4$Valori)

# mappa interattiva

colori.4 <- colorFactor(c("#FC4E07", "#0072B2", "#00BA38", "#E7B800"),
                        domain = levels(variabile.4$Valori))

mappa.4 <- leaflet(variabile.4) %>%
  addTiles() %>%
  addCircleMarkers(lng = ~Longitudine, lat = ~Latitudine,
                   label = ~paste0("Nome: ", Nome, "  Valore: ",
                                   Valori),
                   color = ~colori.4(Valori),
                   radius = 10,
                   stroke = FALSE, fillOpacity = 0.5)
mappa.4

# salvataggio della mappa come un html
saveWidget(mappa.4, "distribuzione dei fonemi nel mondo.html",
    selfcontained = TRUE)

# mappa statica

# aggiunta di punti alla mappa con colori dipendenti dai valori
mappa_colorata.4 <-
  base_mappa +
  geom_point(data=variabile.4,
             aes(x=Longitudine, y=Latitudine, colour=Valori), size=4,
    alpha=I(0.7)) +
  scale_color_manual(values = c("#FC4E07", "#0072B2", "#00BA38", "#
    E7B800"))
```

```
210
211 mappa_colorata.4
```

Codice C.3: Analisi esplorative per la distribuzione delle caratteristiche e dei fonemi nel mondo.

```
x <- caratteristiche[complete.cases(caratteristiche),]
dim(x) # c'è un problema con i valori mancanti: sono troppi

# sottoinsieme del dataset caratteristiche che riguarda le variabili relative alle caratteristiche strutturali
data <- caratteristiche[-c(1:10)]

# caselle con valori mancanti
sum(is.na(data))
# caselle disponibili
nrow(data) * (ncol(data))
# proporzione di caselle con valori mancanti
sum(is.na(data)) / ((nrow(data)) * (ncol(data)))

# NA per variabili
k <- rep(0, ncol(data))
for(i in 1:(ncol(data))) {
  k[i] <- sum(is.na(data[i]))
}
names(k) <- 1:(ncol(data))
sort(k)
summary(k)

barplot(sort(k), ylab = "Numero di valori mancanti",
        xlab = "Caratteristiche ordinate in senso crescente per numero di valori mancanti",
        main = "Valori mancanti per le caratteristiche",
        names.arg = FALSE,
        cex.lab = 1.5, cex.main = 2)

# NA per lingue
w <- rep(0, nrow(data))
for(i in 1:nrow(data)) {
  w[i] <- sum(is.na(data[i,]))
}
names(w) <- caratteristiche[,1]
summary(w)
barplot(sort(w), ylab = "Numero di valori mancanti",
        xlab = "Lingue ordinate in senso crescente per numero di valori mancanti",
        main = "Valori mancanti per le lingue",
        names.arg = FALSE,
        cex.lab = 1.5, cex.main = 2)
```

Codice C.4: Distribuzione dei valori mancanti nel dataset *caratteristiche*.

```
n <- 100
head(sort(w), n) # 100 lingue con meno valori mancanti per le 192 caratteristiche
```

```

migliori <- which(w <= sort(w)[n]) # indici delle migliori n lingue
val.man <- matrix(rep(0, n*ncol(data)), nrow = n)
for(i in 1:n) {
  val.man[i,] <- is.na(data[migliori[i],])
}

# somma per colonna di val.man
som.col <- colSums(val.man) # numero di valori mancanti posseduti dalle 192 caratteristiche per le migliori n lingue
sum(som.col == 0)

# indici delle caratteristiche con meno di 10 valori mancanti
indice <- which(som.col <= 10)

# somma per riga in corrispondenza delle caratteristiche selezionate
som.rig <- rowSums(val.man[,indice])

# sottoinsieme con 30 lingue e 107 variabili, di cui 97 caratteristiche prive di valori mancanti
lingue <- caratteristiche[migliori[which(som.rig == 0)], c(1:10, indice + 10)]
dim(lingue)
```

Codice C.5: Creazione di sottoinsiemi privi di valori mancanti.

```
library(cluster)

table(famiglie[1,], famiglie[2,])
(18+11)/(18+11+13)
dist(famiglie[1:2,], method="binary") # uguale

dm <- dist(famiglie, method="binary") # indice di Jaccard
head(dm)
length(dm)

k <- 3

hcc <- hclust(dm, method="complete")

hcs <- hclust(dm, method="single")

hca <- hclust(dm, method="average")

hcw <- hclust(dm, method="ward.D2")


# verifica della bontà dei metodi

famiglia <- unione$Famiglia[condizione]
famiglia <- as.factor(famiglia)
levels(famiglia) <- c("Indoeuropea", "Niger-Kordofaniana", "Pama-Nyunga")
table(famiglia)

table(famiglia, cutree(hcc, k=k), dnn=c("famiglia", "gruppo"))
table(cutree(hcc, k=k))
```

```r

table(famiglia, cutree(hcs, k=k), dnn=c("famiglia", "gruppo"))
table(cutree(hcs, k=k))

table(famiglia, cutree(hca, k=k), dnn=c("famiglia", "gruppo"))
table(cutree(hca, k=k))

table(famiglia, cutree(hcw, k=k), dnn=c("famiglia", "gruppo"))
table(cutree(hcw, k=k))

# il metodo migliore è quello di Ward
```

Codice C.6: Analisi dei gruppi tramite metodi gerarchici.

```r
set.seed(777)
ce <- 5 # 5 centroidi di partenza

km1 <- kmeans(famiglie, centers=1, nstart=ce)
km2 <- kmeans(famiglie, centers=2, nstart=ce)
km3 <- kmeans(famiglie, centers=3, nstart=ce)
km4 <- kmeans(famiglie, centers=4, nstart=ce)
km5 <- kmeans(famiglie, centers=5, nstart=ce)
km6 <- kmeans(famiglie, centers=6, nstart=ce)
km7 <- kmeans(famiglie, centers=7, nstart=ce)
km8 <- kmeans(famiglie, centers=8, nstart=ce)
km9 <- kmeans(famiglie, centers=9, nstart=ce)
km10 <- kmeans(famiglie, centers=10, nstart=ce)

var.spieg <- c(km1$betweenss/km1$totss,
               km2$betweenss/km2$totss,
               km3$betweenss/km3$totss,
               km4$betweenss/km4$totss,
               km5$betweenss/km5$totss,
               km6$betweenss/km6$totss,
               km7$betweenss/km7$totss,
               km8$betweenss/km8$totss,
               km9$betweenss/km9$totss,
               km10$betweenss/km10$totss)

plot(1:10, var.spieg, type="l", lwd=3,
     xlab = "Numero di gruppi",
     ylab = "Percentuale di devianza spiegata",
     main = "Percentuale di devianza spiegata al variare del numero di gruppi",
     cex.lab = 1.5, cex.main = 2.5)

# si sceglie di usare 3 gruppi

# tabella di classificazione

table(famiglia, km3$cluster)

library(ggfortify)
pulizia.f <-
  theme_bw(base_size=25) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
```

```
        panel.background = element_rect(fill = 'white',
                                        colour = 'white'),
        legend.position="top",
        axis.ticks=element_blank(),
        axis.text.x=element_blank(),
        axis.text.y=element_blank())

# plot dei gruppi sul piano delle prime due componenti principali

gruppi <- autoplot(km3, data=as.data.frame(cbind(famiglie, famiglia)),
         shape="famiglia", frame=TRUE, size=4)
gruppi + pulizia.f
```

Codice C.7: Analisi dei gruppi tramite metodi di partizionamento.

```
# creazione di un vettore di identificatori per i parametri
nomi <- colnames(lingue[-c(1:10)])
nomi.1 <- substring(nomi, 1, regexpr("[.]", nomi))
par_id <- gsub(pattern = '.$', replacement = '', nomi.1)

# selezione della porzione di valori che contiene i dati, e creazione
    di un allineamento di sequenze costituito da una tabella di 1 e 0
condizione.1 <- (valori$Language_ID %in% lingue$Wals_code) &
  (valori$Parameter_ID %in% par_id)
dati.1 <- as.matrix(xtabs(~Language_ID[condizione.1] +
                          Code_ID[condizione.1],
                        valori, sparse = TRUE))
rowSums(dati.1) # somma per riga uguale per ogni lingua

# allineamento di sequenze
write.table(dati.1, "Lingue.txt")

# nomi delle lingue presenti nell'allineamento
write.table(lingue$Nome, "Nomi_Lingue.txt")
```

Codice C.8: Creazione di un allineamento di sequenze per le caratteristiche a partire dal dataset *lingue*.

```
# nomi delle lingue necessarie per l'analisi
lingue_fonemi <- c("Italian", "Slovak", "Hindi", "Armenian (Eastern)",
    "Irish",
        "Arabic (Modern Standard)", "Hebrew (Modern)", "Berber (Middle
    Atlas)", "Tigrinya", "Amharic",
        "Zulu", "Kikuyu", "Yoruba", "Wolof", "Akan",
        "Hungarian", "Finnish", "Mari (Hill)", "Nenets", "Saami (
    Northern)",
        "Khmer", "Vietnamese", "Khasi", "Mundari", "Korku",
        "Telugu", "Kannada", "Malayalam", "Tamil", "Toda")

# selezione del sottoinsieme del dataset unione che contiene le
    informazioni relative ai fonemi delle lingue presenti nella lista
    lingue_fonemi
condizione.2 <- which(unione$Nome %in% lingue_fonemi)
dati.2 <- cbind(unione$Nome[condizione.2],
              unione[condizione.2, 203:ncol(unione)])
rownames(dati.2) <- unione[condizione.2, 1]
```

```

# allineamento di sequenze
write.table(dati.2[-1], "Lingue_Fonemi.txt")

# nomi delle lingue presenti nell'allineamento
write.table(dati.2[,1], "Nomi_Lingue_Fonemi.txt")
```

Codice C.9: Creazione di un allineamento di sequenze per i fonemi a partire dal dataset *unione*.

```
sigle <- famiglia
levels(sigle) <- c("ie", "nc", "pm")
sigle <- paste(sigle, 1:nrow(famiglie))
rownames(famiglie) <- unione[condizione, 1]

# allineamento di sequenze
write.table(famiglie, "Lingue_Fonemi.1.txt")

# nomi delle lingue presenti nell'allineamento
write.table(sigle, "Nomi_Lingue_Fonemi.1.txt")
```

Codice C.10: Creazione di un allineamento di sequenze per i fonemi a partire dal dataset *famiglie*.

```
albero_macroarea <- as.data.frame(cbind(unione$Macroarea, unione
    [203:302]))

for(i in 1:(ncol(albero_macroarea))) {
  albero_macroarea[,i] <- as.factor(albero_macroarea[,i])
}

colnames(albero_macroarea) <- c("Macroarea", 1:(ncol(albero_macroarea)
    -1))
legenda.1 <- t(cbind(colnames(fonemi[-1]), 1:(ncol(albero_macroarea)-1)
    ))

table(unione$Macroarea, exclude = NULL)

library(tidyverse)
library(rsample)
library(rpart)
library(rattle)
library(yardstick)
set.seed(123)
split <- initial_split(albero_macroarea)
train <- training(split)
test <- testing(split)

# Stima dell'albero di regressione

# Misura di impurità: indice di Gini
set.seed(123)
albero_macroarea.1 <- rpart(Macroarea ~ ., data = train,
             method = "class",
             control = rpart.control(minbucket = 10,
                                     cp = 0.01))
```

```

plot(albero_macroarea.1)
text(albero_macroarea.1)

# Misura di impurità: entropia di Shannon
albero_macroarea.2 <- rpart(Macroarea ~ ., data = train,
              parms = list(split = "information"),
              method = "class",
              control = rpart.control(minbucket = 10,
                                      cp = 0.01))

plot(albero_macroarea.2)
text(albero_macroarea.2)

# confronto fra le previsioni
pred.1 <- predict(albero_macroarea.1, newdata = test, type = "class")
pred.2 <- predict(albero_macroarea.2, newdata = test, type = "class")

par(mfrow=c(1,2))
plot(albero_macroarea.1, main = "Albero con indice di Gini")
plot(albero_macroarea.2, main = "Albero con entropia")
par(mfrow=c(1,1))

# studio delle previsioni

# matrice di confusione per l'indice di Gini
matr_conf.1 <- with(test, table(pred.1, Macroarea))
matr_conf.1

accuratezza.1 <- sum(diag(matr_conf.1)) / sum(matr_conf.1)
recupero.1 <- diag(matr_conf.1) / colSums(matr_conf.1)

# matrice di confusione per l'entropia
matr_conf.2 <- with(test, table(pred.2, Macroarea))
matr_conf.2

accuratezza.2 <- sum(diag(matr_conf.2)) / sum(matr_conf.2)
recupero.2 <- diag(matr_conf.2) / colSums(matr_conf.2)

# confronto dell'accuratezza
cbind(accuratezza.1, accuratezza.2)

# confronto del recupero
cbind(recupero.1, recupero.2)
```

Codice C.11: Creazione del dataset *albero_macroarea* e costruzione degli alberi di classificazione.

```
albero_famiglia <- as.data.frame(cbind(unione$Famiglia[condizione],
    famiglie))

for(i in 1:(ncol(albero_famiglia))) {
  albero_famiglia[,i] <- as.factor(albero_famiglia[,i])
}

colnames(albero_famiglia) <- c("Famiglia", 1:(ncol(albero_famiglia)-1))
```

```r
legenda.2 <- t(cbind(colnames(fonemi[-1]), 1:(ncol(albero_famiglia)-1))
    )

set.seed(123)
split <- initial_split(albero_famiglia)
train <- training(split)
test <- testing(split)

# Stima dell'albero di regressione

# Misura di impurità: indice di Gini
set.seed(123)
albero_famiglia.1 <- rpart(Famiglia ~ ., data = train,
                   method = "class",
                   control = rpart.control(minbucket = 5,
                                           cp = 0.01))

plot(albero_famiglia.1)
text(albero_famiglia.1)

# Misura di impurità: entropia di Shannon
albero_famiglia.2 <- rpart(Famiglia ~ ., data = train,
                   parms = list(split = "information"),
                   method = "class",
                   control = rpart.control(minbucket = 5,
                                           cp = 0.01))

plot(albero_famiglia.2)
text(albero_famiglia.2)

# confronto fra le previsioni
pred.1 <- predict(albero_famiglia.1, newdata = test, type = "class")
pred.2 <- predict(albero_famiglia.2, newdata = test, type = "class")

par(mfrow=c(1,2))
plot(albero_famiglia.1, main = "Albero con indice di Gini")
plot(albero_famiglia.2, main = "Albero con entropia")
par(mfrow=c(1,1))

# studio delle previsioni

# matrice di confusione per l'indice di Gini
matr_conf.1 <- with(test, table(pred.1, Famiglia))
matr_conf.1

accuratezza.1 <- sum(diag(matr_conf.1)) / sum(matr_conf.1)
recupero.1 <- diag(matr_conf.1) / colSums(matr_conf.1)

# matrice di confusione per l'entropia
matr_conf.2 <- with(test, table(pred.2, Famiglia))
matr_conf.2

accuratezza.2 <- sum(diag(matr_conf.2)) / sum(matr_conf.2)
recupero.2 <- diag(matr_conf.2) / colSums(matr_conf.2)

# confronto dell'accuratezza
```

```
63 cbind(accuratezza.1, accuratezza.2)
64
65 # confronto del recupero
66 cbind(recupero.1, recupero.2)
```

Codice C.12: Creazione del dataset *albero_famiglia* e costruzione degli alberi di classificazione.

Appendice D

Appendice: codice Python

Programma per la creazione di input per SplitsTree

```
import numpy as np
f = open("input_SplitsTree.txt", "w") # nome del file che viene creato
  # file di testo con matrice di 0 e 1 per la presenza delle
    caratteristiche nelle lingue
l = open("Lingue.txt").readlines()
  # elenco numerato con i nomi delle lingue
n = open("Nomi_Lingue.txt").readlines()


print("#nexus\n\nBEGIN Taxa;\nDIMENSIONS ntax=%d;\nTAXLABELS\n"
      %(len(n)-1), end="\n", file = f)


nomi = []
# elenco dei taxa, ovvero dei nomi delle lingue
for i in range(1, len(n)):
    x = n[i][5:-2]
    print("[%d]" %i, end=" ", file=f)
    if i >= 100:
        x = x.replace(" ", "", 1)
    # sostituzione dei caratteri non supportati da SplitsTree
    schema = [" ", "(", ")", "\"", "\'", "=", "|", "á", "à", "ã",
              "â", "ä", "é", "É", "è", "ê", "ë", "í", "ì", "ó",
              "ò", "ô", "ö", "ú", "ù", "ü", "ñ"]
    sostituto = ["-", "", "", "", "", "", "", "a", "a", "a", "a",
                 "a", "e", "E", "e", "e", "e", "i", "i", "o", "o",
                 "o", "o", "u", "u", "u", "n"]
    for j in range(len(schema)):
        x = x.replace(schema[j], sostituto[j])
    print("\'" + x + "\'", end="\n", file=f)
    nomi.append(x)


```

```
# informazioni per SplitsTree
print("\n;\n\nEND;[Taxa]\n\nBEGIN Unaligned;", end = "", file=f)
print("\nDIMENSIONS ", end = "", file=f)
print("ntax=%d;\nFORMAT\n\n\n       " %(len(n)-1), end = "", file=f)
print("datatype=STANDARD\n       ", end = "", file=f)
print("missing=?\n       ", end = "", file=f)
print("symbols=\"0 1 2 3 4 5 6 7 8 9\"\n       ", end="\n", file=f)
print("labels=left\n;\n\nMATRIX", end="\n", file=f)


# matrice di 0 e 1 che forma l'allineamento di sequenze
for i in range(1, len(l)):
    print("\'" + nomi[i-1] + "\'", end=" ", file=f)
    if i != len(l)-1:
        print(l[i][5:].replace(" ",
              "").replace("\"", "").rstrip("\n")
              + ",", end="\n", file=f)
    else:
        print(l[i][5:].replace(" ",
              "").replace("\"", "").rstrip("\n")
              + ";", end="\n", file=f)
print("END;[Unaligned]", end="", file=f)


f.close()
```

Codice D.1: Creazione di input per SplitsTree.

Ringraziamenti

Desidero ringraziare sinceramente il mio Relatore, la Professoressa Manuela Cattelan, per avermi accompagnato durante questo lavoro di tesi con estrema disponibilità.
Ringrazio i miei genitori Stefano e Judit per il sostegno che mi hanno dimostrato e per la grande pazienza che hanno avuto con me in questi mesi di studio. Un grazie a mio fratello Carlo e al resto delle famiglie Nagy e Szövérfi per avermi seguito lungo questo percorso.
Ringrazio i miei compagni di corso per essermi stati vicini, in particolare Pietro, Francesco e Francesco, Marco, Augusto, Leonardo, Giuseppe, Sophie Grace, Raul e Federico.
Infine ringrazio il mio amico Giacomo.