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

Statistica per le tecnologie e le scienze

FAKE NEWS DETECTION TRAMITE TEXT MINING E MODELLI STATISTICI

Relatore: Prof. Antonio Canale
Dipartimento di Scienze Statistiche

Laureando: Cecilia Peccolo
Matricola n. 1220253

Anno Accademico 2021/2022

A nonna Ivana,
che con le mie stesse opportunità
sarebbe arrivata a traguardi anche
migliori di questo.

ABSTRACT

Il presente lavoro mira a studiare alcuni approcci statistici per la rilevazione di fake news, per poi confrontarli e arrivare al metodo che giunge alle previsioni migliori tra quelli considerati.

La base di partenza è una collezione di documenti già etichettati, da cui poi si procede in due fasi: la prima prevede l'utilizzo di tecniche di text mining sui testi, in modo da ottenere dei dati strutturati utili per la seconda fase. Qui infatti si usano le informazioni estratte dalla prima per applicare delle procedure di classificazione del singolo testo, utilizzando diversi modelli statistici presentati al capitolo 3. La classificazione è binaria, in quanto un elemento della collezione viene interpretato come "reale" se tutte le parti sono verificabili, o "falso" se anche solo una parte di esso non è vera.

Gli strumenti utilizzati sono stati appresi ed esercitati durante molteplici insegnamenti del corso di studi, con particolare menzione a Analisi dei dati multidimensionali, Modelli statistici 2 e Metodi statistici per i Big Data. Inoltre sono stati introdotti i metodi della Foresta Casuale e del Gradient Boosting non previsti nei corsi frequentati, ritenuto necessario un approfondimento.

Le analisi sono state svolte usando i linguaggi Python e R.

Indice

Capitolo 1

Introduzione

1.1 Fake news

1.1.1 Definizione di fake news

In letteratura esistono molteplici definizioni di "fake news", diverse per ogni ambito considerato: per dare una definizione di ciò che verrà considerata una *notizia falsa* nel presente lavoro si sono considerate diverse fonti ufficiali. Per esempio, in *"Social media and fake news in the 2016 election"*[1], viene data la definizione di "fake news" come notizie che sono facilmente verificabili come false e che sono scritte in modo fuorviante per ingannare il lettore.
Nella presente relazione si userà la seguente definizione basata su quella data nell' articolo citato precedentemente.

Definizione 1 *Una fake news è un articolo di cronaca che è intenzionalmente e verificabilmente falso.*

1.1.2 La questione delle fake news

Il numero di persone attive nel web è aumentato in modo esponenziale negli ultimi anni, portando la società a basare la maggior parte delle proprie conoscenze relative ad un qualsiasi campo della vita quotidiana (in questo caso ci si riferisce in particolare le notizie di attualità) su ciò che si legge nei social media. Questo risultato è sicuramente dovuto alle modalità con cui si reperisce una notizia, più facili e veloci dei metodi tradizionali come la televisione o un quotidiano. Infatti, una ricerca effettuata negli Stati Uniti nel 2016 [2], afferma che il 68% degli adulti statunitensi apprende le notizie esclusivamente dai social, proporzione cresciuta notevolmente rispetto al 2012, in cui era il 49%.

Nonostante i metodi moderni presentino molti vantaggi, la facile reperibilità dell'informazione porta anche allo spargimento più immediato di un enorme numero di notizie non vere, create appositamente per motivi politici o economici: per esempio, nel 2016 è nata in America la teoria complottista "Pizzagate"[3], la quale attaccava diversi funzionari del partito democratico di traffico di esseri umani e abusi su minori. Venne stimato che oltre 1 milione di Tweet legati al tema

fossero fake news, la cui pubblicazione nei social contribuisce in modo sostanziale alla diffusione di una reputazione negativa dello schieramento politico in causa.

Tuttavia, la propaganda contro gli avversari in una campagna elettorale non è un fenomeno nuovo, ma in un epoca in cui il lettore è immerso in una quantità così vasta di notizie che legge e condivide senza accertarsi della loro veridicità, è necessario elaborare dei metodi di riconoscimento automatico sempre più efficienti. Le difficoltà nel riconoscimento di notizie false cominciano proprio nell'automatizzazione dei processi: infatti, le fake news sono elaborate appositamente per essere credibili, rendendone quindi difficile la corretta classificazione basandosi esclusivamente sulle caratteristiche del testo.
Di conseguenza, basandosi sulla struttura testuale, si rischia di classificare come "real" quelle fake news che al loro interno citano pezzi di notizie vere ma decontestualizzate. In questo modo si ottiene una distorsione della realtà mantendendo una struttura sintattica di una notizia reale.

1.1.3 Fake news detection

Nei paragrafi precedenti è stata presentata la definizione teorica di fake news nei social media. Ora si presenta la formulazione matematica teorica alla base dei procedimenti implementati in seguito.
Definiamo la seguente notazione:

- Si chiami a il singolo articolo con un certo contenuto.
- Definiamo inoltre l'insieme degli attributi che descrivono il singolo documento come ϵ.

Codifica matematica di fake news: Date le caratteristiche dell'articolo a, l'obiettivo di un procedimento di fake news detection è quello di predirre se a è fake news o meno, data la funzione $F : \epsilon \to [0, 1]$ come:

$$F(a) = \begin{cases} 1 & \text{se } a \text{ è una fake news} \\ 0 & \text{altrimenti} \end{cases}$$

dove F è la funzione di classificazione di cui verrà trattato nel capitolo 3.

1.2 Text mining

Il text mining è definibile come una tecnologia linguistico/matematica utile per lo studio di una grande quantità di testi, da cui si è in grado di estrarre informazioni latenti che risulterebbero molto costose da ricavare manualmente. Infatti, il concetto fondamentale è quello di riuscire ad estrarre poche informazioni da grandi banche di dati senza doverli elaborare tutti singolarmente.

1.2.1 Differenza tra text mining e data mining

Il data mining rappresenta un insieme di tecniche per estrarre dei pattern o modelli latenti dai dati, così da poterne ottenere informazioni preziose. Nel caso presente, si intendono usare queste procedure per classificare i dati a disposizione, i quali però, essendo testi, non sono utilizzabili per le tecniche di data mining.
Qui entra in gioco il text mining, il quale, come il data mining, cerca di estrarre informazioni utili identificando ed esplorando modelli nei dati: questa volta però le origini dei dati sono proprio i testi.
Dunque, si estrarranno le caratteristiche latenti dai testi tramite il text mining e le si trasformeranno in un formato strutturato utile per l'applicazione dei procedimenti automatici di data mining.

La prima procedura da eseguire per poter trasformare dei documenti liberi in dati strutturati è quella del *pre-processing*.

1.2.2 Elaborazione dei testi

Come già accennato nelle sezioni precedenti, il *fake news detection* è un problema di classificazione binaria dei testi. Per ottenere una struttura dei dati adatto ad applicare tecniche di classificazione, è necessario svolgere in primo luogo i seguenti passaggi, che vanno sotto il nome di *"pre-processing"*.

- Tokenizzazione
  La prima fase è quella di manipolare i dati estraendo un insieme di elementi rilevanti come parole, bigrammi e segni di punteggiatura. La tecnica usata a questo fine è la "tokenizzazione", la quale si occupa di suddividere il contenuto dei testi in token, definibile come sequenza di caratteri circondata dai limitatori, in questo caso gli spazi bianchi. In questa fase si è fatto uso anche della lista di **stop-words** delle parole inglesi: queste sono dei vocaboli definiti appositamente per essere ignorati al momento dell'elaborazione del testo, in quanto congiunzioni o parole che non portano a differenziare i testi perchè parti strutturali del discorso. Come lista di riferimento si è usata quella presente nella libreria NLTK (Natural Language Toolkit) in Python, di cui è disponibile il dizionario alla directory
  "home/pratima/nltk_ data/corpora/stopwords".
  Questa procedura è utile per aumentare la qualità dei dati e per ridurre la quantità di parole da processare, abbassando le risorse computazionali utilizzate.

- Stemming
  Il processo di stemming riduce le parole alla loro radice, togliendone qualsiasi variazione morfologica. Questo procedimento non è stato utilizzato in tutte le fasi dell'estrazione di caratteristiche dai testi in quanto non sempre porta a vantaggi: è risultato favorevole solo nel conteggio dei vocaboli in quanto diminuisce i numeri distinti di tipi termini, incrementando le frequenze solo di alcuni.

1.3 Obiettivi

La ricerca si prefigge di utilizzare le informazioni più significative tra quelle estratte dai testi tramite il text mining, per elaborare dei modelli statistici che classifichino i documenti come *fake* o *real*. Con l'utilizzo di opportune misure di efficacia, i metodi utilizzati verranno confrontati e discussi.

Capitolo 2

Il data set

2.1 Descrizione del data set

I dati presi in considerazione sono stati raccolti ed elaborati dagli autori dell'articolo *"WELFake: Word Embedding Over Linguistic Features for Fake News Detection"*[4]: i documenti sono generalmente notizie di attualità pubblicate da aziende di stampa americane.
Questa collezione è stata creata unendo i seguenti data set:

- **McIntire**: i 6335 documenti presenti sono stati selezionati da migliaia di articoli pubblicati da New York Times, WSJ, Bloomberg, NPR e The Guardian tra il 2015 e 2016 nei mesi antecedenti alle elezioni presidenziali americane.
- **Reuters**: questa collezione di documenti è stata raccolta e formattata nei primi anni 2000. Raccoglie 44898 documenti riguardanti notizie di diverso genere.
- **BuzzFeed Political**: questo data set comprende un campione completo di post pubblicati su Facebook da delle aziende di stampa americane in prossimità delle elezioni presidenziali statunitensi del 2016. Contiene 101 articoli.
- **Kaggle**: il resto dei 20800 documenti sono vari articoli di carattere politico.

Il totale di dati raccolti sono 72134 articoli, con 35028 veri e 37106 falsi.
L'unione di queste quattro collezioni per otterne una di dimensioni così elevate, presenta vantaggi non solo dal punto di vista di *overfitting*, ma anche di riduzione delle limitazioni e della distorsione di un singolo data set.
Inoltre, sono stati selezionati specificatamente questi quattro perchè avevano la stessa struttura: per ogni documento sono stati rilevati un codice seriale identificativo, il titolo, il testo e l'etichetta "FAKE" o "REAL", data manualmente da dei giornalisti. Per facilità d'uso, quest'ultima colonna è stata trasformata in una variabile dicotomica, pari a 1 quando la modalità del documento è "FAKE" e 0 altrimenti.

Dai dati di partenza sono stati eliminati tutti quei documenti privi di testo, ottenendo così una collezione di 71357 documenti, con 36330 notizie false e 35027 notizie reali (figura 2.1).
Per le analisi, si sono estratti casualmente dei documenti per il **data set di train** e per quello di **test**, prendendo il 75% dei dati per l'addestramento e il resto per la validazione: dunque, il primo conterrà 53517 documenti e il secondo 17840.

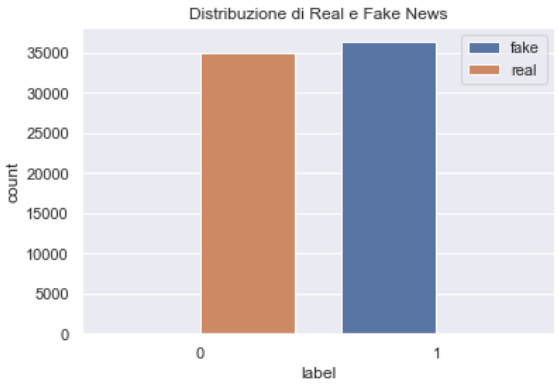

Figura 2.1: Frequenza di real e fake news

2.2 Selezione delle caratteristiche

Come spiegato nel paragrafo 1.2, per automatizzare la procedura di rilevazione di fake news, è necessario estrarre dai documenti delle caratteristiche utilizzabili come dati strutturati per gli algoritmi di classificazione. Di conseguenza, si è pensato a delle possibili caratteristiche estraibili dai testi tramite tecniche di text mining che potessero differenziare fake e real news.

2.2.1 Analisi sintattica

Anzitutto, si è eseguita un'analisi sulla struttura dei testi. Partendo da delle considerazioni esposte nell'articolo *Fake News Detection on Social Media: A Data Mining Perspective*"[5], scritto da dei ricercatori della Michigan State University, per ogni testo si è voluto estrarre:

- Lunghezza del testo, considerando che le fake news, essendo meno argomentabili, abbiano una lunghezza media inferiore. Infatti, se confrontiamo la percentuale di fake e real news nella figura 2.2, si vede che la frequenza di informazioni reali in testi con meno di 250 parole è inferiore alla percentuale di fake. Questa invece diminuisce sempre di più considerando i testi di oltre 1000 parole, come invece rappresentato nella figura 2.3.
- Percentuale di parole corte e lunghe, considerando le prime come inferiori ai 5 caratteri, mentre le seconde come superiori agli 11.

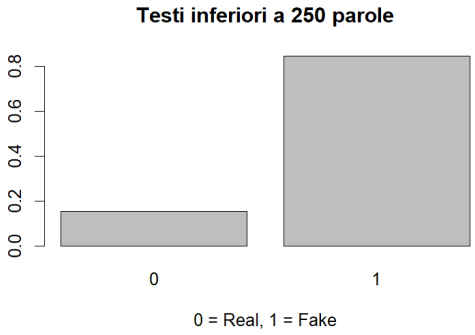

Figura 2.2: Frequenza di fake e real news nei testi con meno di 250 parole

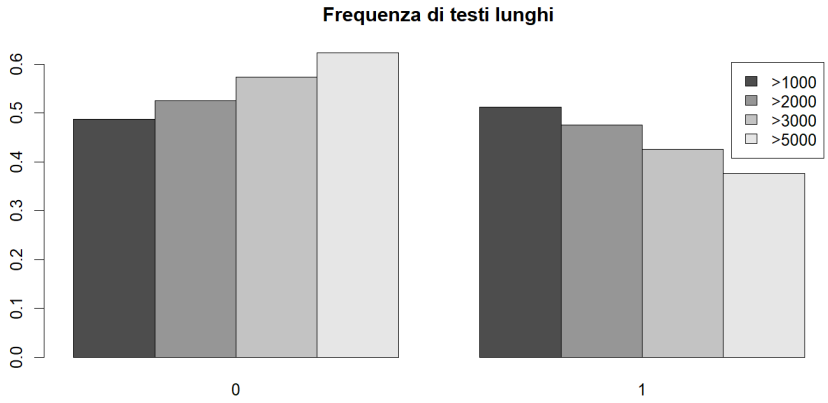

Figura 2.3: Frequenza di 0 (real) e 1 (fake) nei testi con più di 1000, 2000, 3000 e 5000 parole

Dalla rappresentazione delle frequenze di documenti con oltre il 20% di parole superiore agli 11 caratteri (figura 2.4) si nota che la frequenza di parole "lunghe" è nettamente superiore nelle fake news.

- Numero di parole medio per frase
- Numero di segni di punteggiatura
- Dato che le fake news non fanno riferimenti a fatti accaduti realmente, si suppone che non possano essere presenti numerosi citazioni, quindi si è contata anche la presenza di discorsi diretti, contando il numero di ' " ' presenti nei testi. Dai dati infatti vediamo che la frequenza di notizie reali fra i documenti che hanno almeno 1 citazione è decisamente maggiore, come illustrato nella figura 2.5.

Per ottenere una visione bidimensionale dei dati raccolti così da poterne fare considerazioni globali, si è usata l'**analisi delle componenti principali** come strumento di riduzione della dimensionalità. Dal grafico delle componenti principali

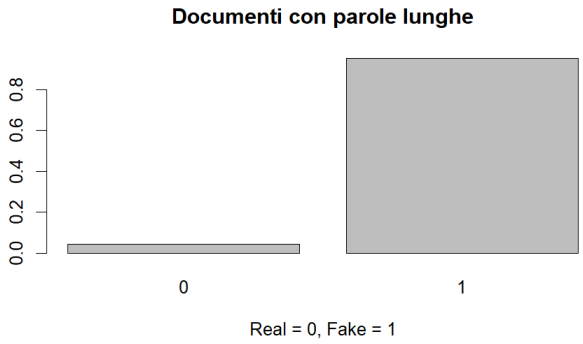

Figura 2.4: Frequenza di real e fake news nei testi con più del 20% di parole con più di 11 caratteri

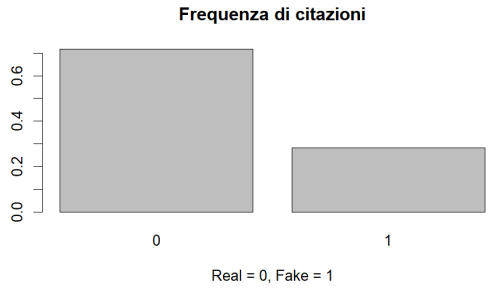

Figura 2.5: Frequenza di real e fake news nei testi con più di 1 citazione

si vede che la prima componente è spiegata da lunghezza del testo, numero di citazioni e segni di punteggiatura, mentre la seconda è principalmente una misura di percentuale di parole "lunghe/corte" e di numero medio di parole per frase. Osserviamo che le variabili con punteggi negativi per la prima e positivi per la seconda sono soprattutto real news, mentre le fake news hanno principalmente punteggi positivi rispetto sia alla prima che la seconda. Quindi le osservazioni con valori alti per le variabili di lunghezza del testo, numero di citazioni, presenza di punteggiatura e percentuale di parole brevi, sono per lo più notizie vere. Il contrario invece per le fake news, le quali però avranno valori superiori per le variabili di lunghezza media di frase e percentuale di parole lunghe.
I risultati sono illustrati nella figura 2.6 .

2.2.2 Bag of words

Dopo l'analisi della struttura sintattica dei documenti, si è svolta un'analisi delle parole più frequenti nelle due tipologie di news considerate. I risultati osservati sono stati ottenuti dopo aver processato i testi: quindi sono state tolte le stop-words e i segni di puteggiatura e si sono uniti i bigrammi più frequenti.
Si può notare (figure 2.7, 2.8) che ci sono delle parole con frequenza molto alta sia nelle notizie reali che in quelle false. Tuttavia, non si esclude che il peso delle

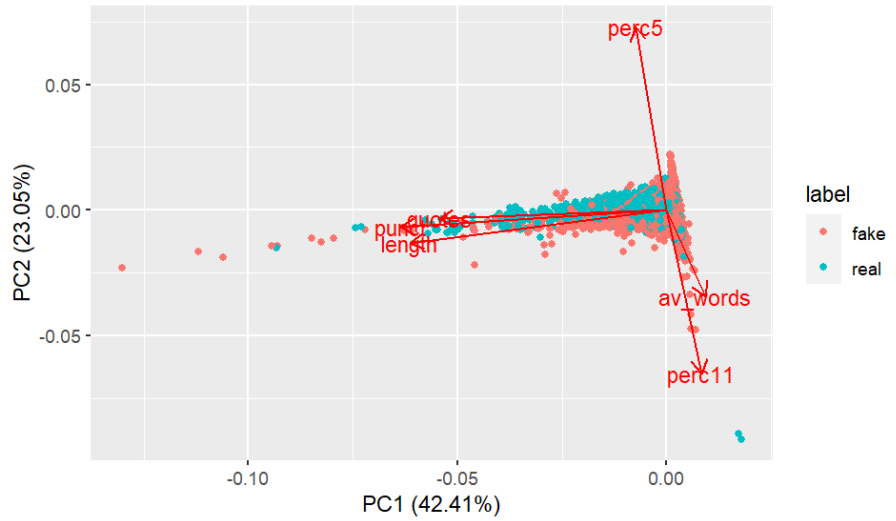

Figura 2.6: Analisi delle componenti principali delle caratteristiche sintattiche dei documenti

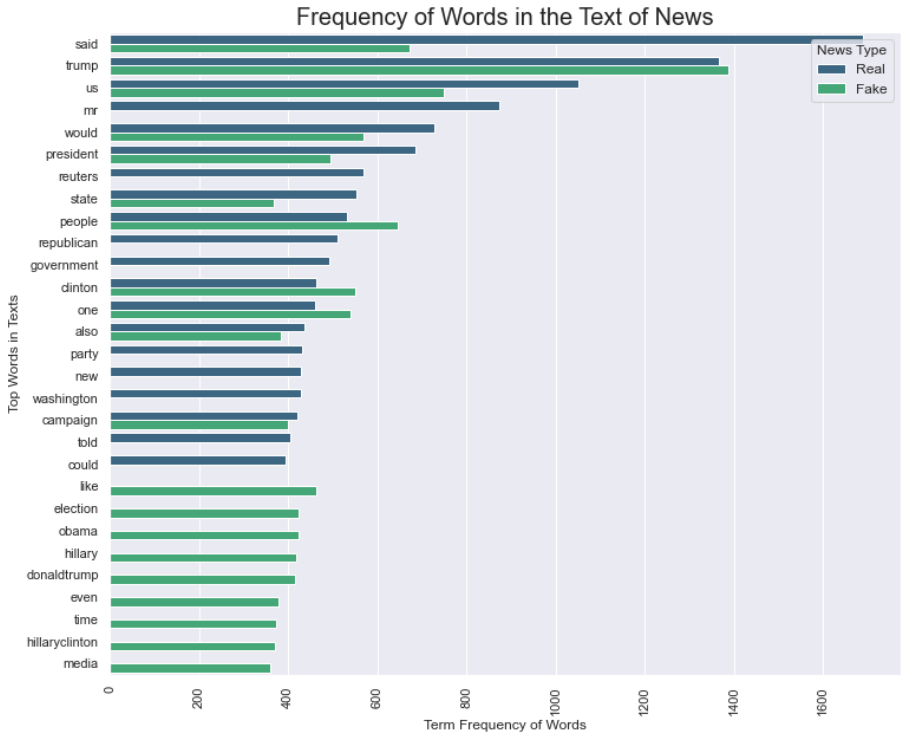

Figura 2.7: Frequenze delle parole più frequenti

Figura 2.8: Wordcloud di Fake e Real news

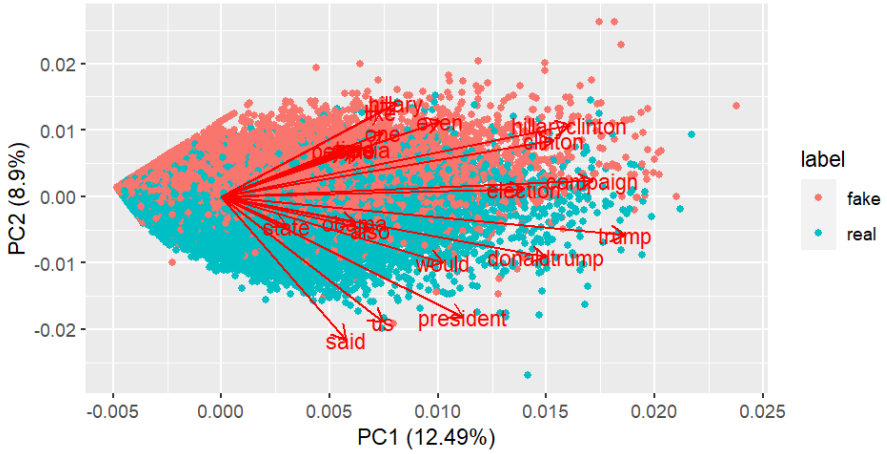

Figura 2.9: Analisi delle componenti principali della *bag-of-words*

parole più frequenti delle fake news calcolato in ogni testo, potrebbe essere una caratteristica importante da estrarre per la classificazione.
Anzitutto si procede con la creazione della *bag-of-words* di riferimento: questa è costituita dalle 20 parole più frequenti nei documenti falsi. Per ottenere i pesi di rilevanza di ogni elemento della bag-of-words nei documenti, si è considerato lo schema di pesatura **TFIDF (*term frequency inverse document frequency*)**: questa quantità non indica solo il numero di volte che un termine appare nel testo, ma anche l'importanza che ha all'interno dell'intera collezione. In particolare, il TFIDF prevede che il peso c_{ij} venga calcolato nel modo seguente:

$$c_{ij} = f_{ij} \log \frac{N}{n_j} \tag{2.1}$$

dove f_{ij} è la frequenza della j-esima parola nell'i-esimo documento e n_j è il numero di documenti in cui appare j fra tutti gli N del data set di train.
Dall'analisi delle componenti principali per le 20 parole più frequenti nelle fake news (figura 2.9), si vede che le osservazioni vengono ben distinte tra fake e real lungo l'asse della seconda componente principale, descritta sopratutto da parole come "said", "president", "hillary" e "U.S."; al contrario termini come, ad esempio, "trump", "campaign", "election" o "clinton", molto frequenti in entrambe le categorie, hanno correlazione quasi nulla con la seconda componente. Tuttavia, la varianza cumulata spiegata dalle due componenti è solo del 20% circa, quindi troppo bassa per poter avere una visualizzazione accurata dei dati originari sul piano bidimensionale.

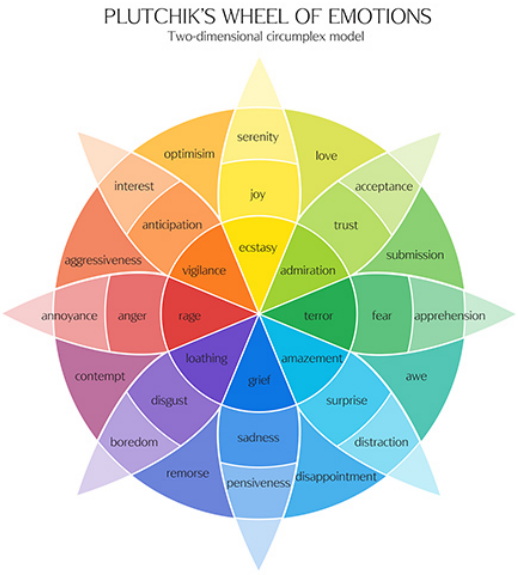

Figura 2.10: La ruota delle emozioni di Robert Plutchik

2.2.3 Analisi delle emozione primarie

Un'altro aspetto interessante da tenere in considerazione nell'analisi, sono le emozioni espresse nei testi. Infatti, le fake news vengono scritte appositamente per catturare l'attenzione del lettore e suscitargli un'emozione forte così da favorire la condivisione dell'articolo: per questi motivi ci si aspetta che le fake news contengano una percentuale di emozioni *forti* alta.
Lo psicologo Robert Plutchik nel libro *"Emotion: a Psychoevolutionary Synthesis"* [13] ha esposto la sua teoria secondo la quale ogni animale prova 8 emozioni primarie: gioia, accettazione, paura, sorpresa, tristezza, disgusto, rabbia e aspettativa. L'idea di Plutchik è che le emozioni possono essere disposte in una "ruota" e che combinando quelle primarie, si ottengono emozioni più complesse, con lo stesso funzionamento della ruota dei colori: per esempio, la delizia è il risultato della combinazione di sorpresa e gioia. Da questa teoria è stata elaborata la *ruota delle emozioni* rappresentata nella figura 2.10.

Nel 2013, Saif M. Mohammad e Peter D. Turney del National Research Council Canada, hanno utilizzato questa teoria per classificare in dizionari ontologici 27000 termini, così da poter predirre la percentuale di ogni emozione espressa nel testo [6].
Recenti aggiornamenti hanno aggiunto anche la classificazione in emozioni negative e emozioni positive, categorie più generiche per poter descrivere anche le parole non allocate nelle 8 emozioni primarie.

Utilizzando i dizionari ontologici citati precedentemente, si è ottenuto per ogni documento della collezione la percentuale delle 10 possibili emozioni. Nella figura 2.11 viene rappresentato il risultato dell'analisi delle componenti principali per una visualizzazione in dimensioni ridotte delle osservazioni.

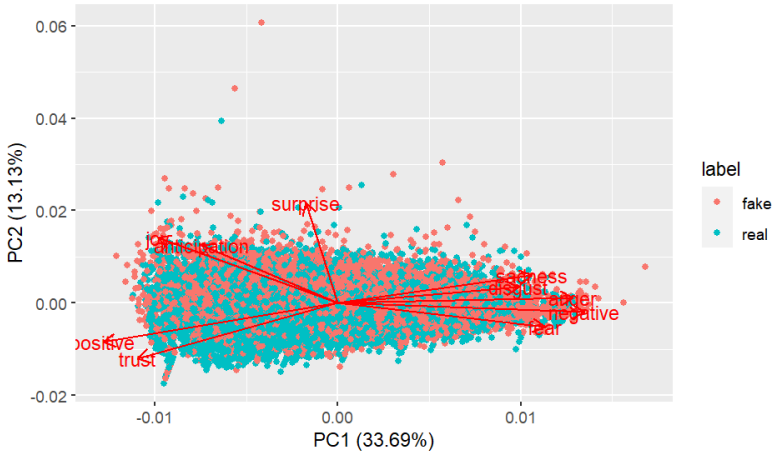

Figura 2.11: Analisi delle componenti principali delle emozioni primarie

La distinzione tra fake e real news risulta più chiara rispetto la prima componente principale, che è correlata positivamente alle emozioni negative come tristezza, rabbia, paura e disgusto: le variabili con punteggi più alti rispetto la prima componente sono principalmente fake news. Emozioni positive come fiducia o gioia, sono correlate negativamente alla prima componente principale, facendo intuire che osservazioni con punteggi alti per questi variabili sono soprattutto notizie reali.

2.2.4 Topic modelling

Come ultima procedura di estrazione delle caratteristiche, si è applicata una procedura di clustering dei testi, in particolare è stato utilizzato il metodo non supervisionato **Latent Dirichlet Allocation**. Questo è un modello probabilistico che si basa sull'assunzione che ogni testo sia considerabile come una mistura di argomenti latenti, dove ogni argomento è caratterizzato da una distribuzione di parole.
Il processo di generazione dei *topic* prevede anzitutto il campionamento dell'argomento per l'i-esimo documento da θ_i dove $\theta_i \sim Dir(\alpha)$ e, condizionatamente al topic selezionato, si campiona la k-esima parola da Ψ_k, con $\Psi_k \sim Dir(\beta)$. I parametri α e β sono dei vettori con, rispettivamente, i k topic e le v parole del dizionario.
Nel data set di train si sono individuati 8 topic, riassumibili con i seguenti titoli: politica estera statunitense, notizie di attualità, la campagna elettorale del 2016, rapporti USA-Russia, parito repubblicano, notizie di cronaca, lo scandalo del 2015 tra i funzionari Hillary Clinton e infine argomenti sociologici generali.
Per ottenere la classificazione anche dei testi nel data set di test, si è utilizzata la probabilità a posteriori:

$$Pr(d_{ij} \mid t_{j^*}, x) \tag{2.2}$$

che definisce la probabilità che l'i-esimo documento del test set riguardi il j-esimo topic, dati i j^* argomenti trovati nel data set di train e le x (con x v-dimensionale) parole del dizionario utilizzato per creare il modello iniziale.

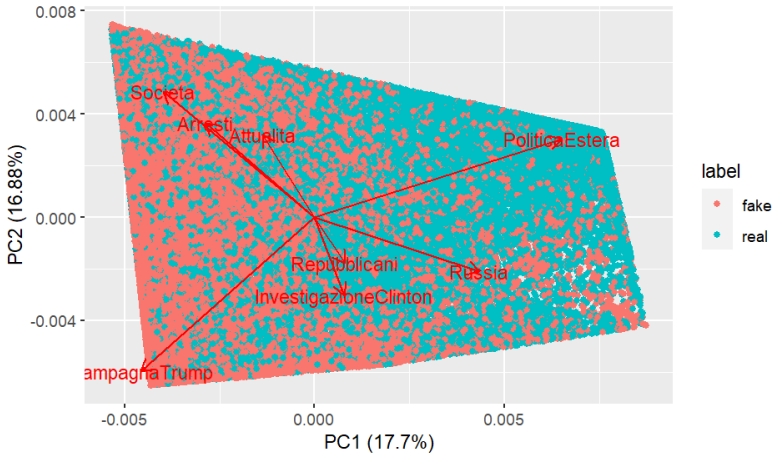

Figura 2.12: Analisi delle componenti principali dei topic latenti

Si è ottenuta l'analisi delle componenti principali della figura 2.12.
Dall'analisi risulta che la discriminazione più evidente tra real e fake news avviene lungo l'asse della prima componente, la quale risulta positivamente correlata con gli argomenti di politica estera e rapporti con la Russia. Le osservazioni con punteggi più alti in corrispondenza di queste variabili sono soprattutto notizie vere, mentre quelle con punteggi maggiori in materia di campagna elettorale di Donald Trump sono quasi esclusivamente fake. Tuttavia, le prime due componenti principali spiegano circa solo il 35% della varianza cumulata. Infatti abbiamo maggiore distinzione se prendiamo in considerazione anche la terza componente, la quale porta a una varianza spiegata cumulata del 50% (figura 2.13).

2.3 Rappresentazione generale

In generale, le variabili rilevate portano ad una buona distinzione di notizia vera o falsa, anche analizzandole a gruppi come nei paragrafi precedenti.
Per avere una rappresentazione grafica complessiva della somiglianza tra documenti, si è utilizzato il metodo **t-SNE** *(Stochastic Neighbour Embedding)*, il quale modella i punti in modo che oggetti vicini nello spazio originale di 44 dimensioni, risultino vicini nello spazio bidimensionale (e viceversa per gli oggetti lontani), cercando di preservarne la struttura di partenza.
I dati sono stati standardizzati in modo da essere confrontabili.

Dalla figura 2.14 vediamo che c'è una buona distinzione delle due categorie rispetto alla seconda coordinata, di conseguenza le osservazioni della categoria "real" avranno valori simili rispetto a questa.
La visibile distinzione delle fake e real news fa intendere che le variabili selezionate siano dei buoni indicatori delle due categorie: questa base di partenza assicura che i modelli adattati successivamente nel Capitolo 4, discriminino efficacemente le osservazioni e giungano a buone previsioni.

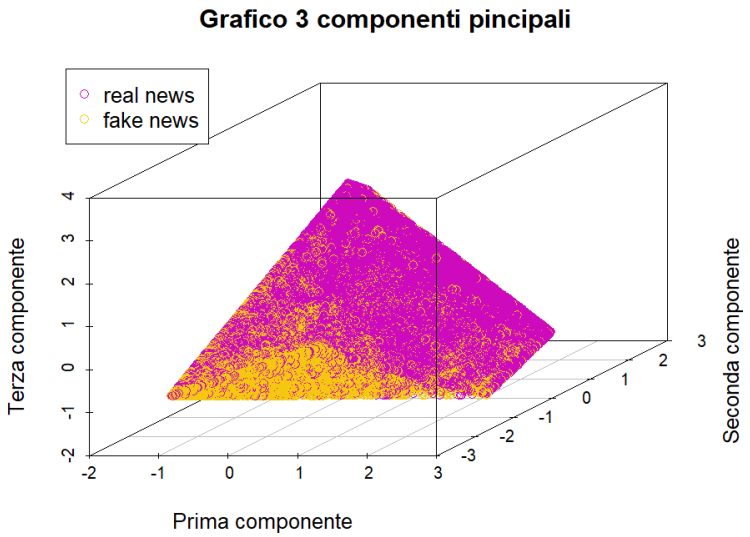

Figura 2.13: Analisi delle componenti principali dei topic latenti

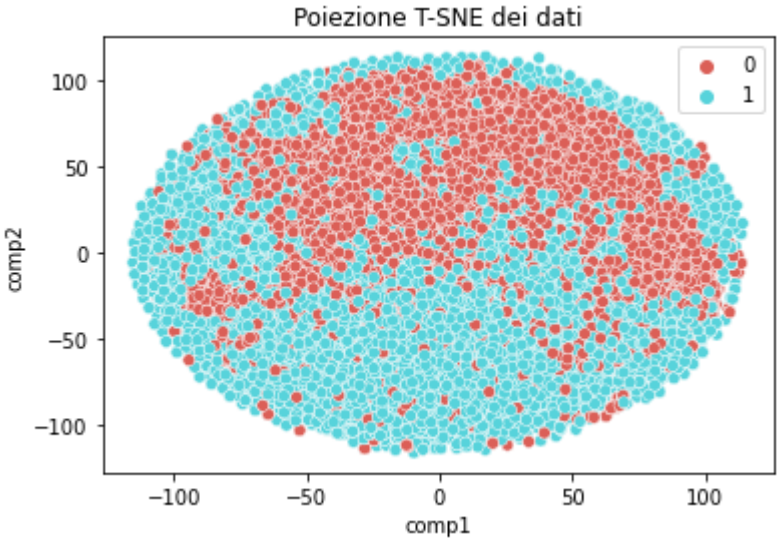

Figura 2.14: Proiezione t-SNE dei dati

Capitolo 3

Metodi

Nel presente capitolo verranno utilizzate le caratteristiche estratte dai documenti nel Capitolo 2 per applicare dei modelli statistici al fake news detection.
Come risposta viene considerata l'etichetta "FAKE" o "REAL" dei documenti come una variabile dicotomica pari a 1 quando la news è falsa e 0 viceversa. Le variabili concomitanti sono in totale 44: come spiegato nel Capitolo 2, per ogni testo è stato registrato la sua lunghezza, la percentuale di parole "corte", la percentuale di parole "lunghe", il numero medio di parole nel testo, il numero di citazioni, il numero di segni di punteggiatura, il punteggio *TFIDF* per ognuna delle 20 parole più frequenti nelle fake news, la percentuale di parole riferite alle 10 emozioni primarie e infine la percentuale di parole riferite agli 8 topic individuati.

3.1 Strumenti di valutazione

Per valutare l'efficacia dei modelli utilizzati per la classificazione delle fake news, è stata utilizzata principalmente la misura di **accuratezza**, la quale viene calcolata utilizzando i seguenti indici:

- **VP (Vero Positivo)**: quando una fake news viene classificata correttamente come tale
- **VN (Vero Negativo)**: quando una real news viene classificata correttamente come tale
- **FP (Falso Positivo)**: quando una notizia predetta come fake in realtà è una notizia vera
- **FN (Falso Negativo)**: quando una notizia classificata come reale è in realtà una fake news

L'accuratezza viene calcolata con la seguente formula:

$$accuratezza = \frac{|\,VP\,| + |\,VN\,|}{|\,VP\,| + |\,VN\,| + |\,FP\,| + |\,FN\,|} \tag{3.1}$$

Un'altra misura di efficacia che verrà utilizzata per il confronto tra modelli è il **punteggio F1**, il quale risulta particolarmente efficace rispetto all'accuratezza

quando ci sono numerosità molto diverse tra categorie. Nel presente lavoro il numero di notizie reali e false sono abbastanza simili, tuttavia è appropriato valutare i modelli con più misure.
Il punteggio F1 è la media armonica di **precisione** e **richiamo**:

$$\text{precisione} = \frac{\mid VP \mid}{\mid VP \mid + \mid FP \mid} \tag{3.2}$$

$$\text{richiamo} = \frac{\mid VP \mid}{\mid VP \mid + \mid FN \mid} \tag{3.3}$$

$$\text{F1 score} = 2 \times \frac{Precisione \times Richiamo}{Precisione + Richiamo} \tag{3.4}$$

Come strumento grafico verrà utilizzato principalmente la **curva ROC***(Receiver Operating Characteristic)* che tiene conto degli indici elencanti in precedenza: infatti, al variare della soglia di separazione tra le due classi, indica i valori di sensibilità (altro modo per chiamare il richiamo della 3.3) e 1 - specificità (3.5).

$$\text{specificità} = \frac{\mid VN \mid}{\mid VN \mid + \mid FP \mid} \tag{3.5}$$

3.2 Modelli

3.2.1 Modello di regressione logistica

Data la variabile risposta dicotomica, lo strumento più semplice e intuitivo da utilizzare per la classificazione è il modello di regressione logistica, il quale assume che le osservazioni provengano dalle variabili casuali Y_i con $Y_i \sim Bin(1, \pi_i)$ e $Y_1, ..., Y_n$ indipendenti. Considerando un esempio teorico, la specificazione della probabilità di successo per l'i-esima osservazione diventa dunque:

$$\pi_i = g^{-1}(\beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip})$$

con $g()$ funzione di legame. Per facilità interpretativa si considererà g() = logit().

Come soglia di separazione per la classificazione nelle due categorie, si considera $\hat{y}_i = 0.5$, allocando quindi l'i-esima osservazione al gruppo "FAKE" se il valore predetto è superiore a questa soglia; se inferiore, viene allocato al gruppo "REAL".

3.2.2 Analisi discriminante

L'analisi discriminante è una procedura che, al contrario della regressione logistica, è pensata appositamente per la classificazione in problemi di regressione con risposta categoriale.
Il punto di partenza è pensare alla distribuzione della variabile casuale p-dimensionale X, come una mistura delle K distribuzioni di probabilità delle K subpopolazioni (nel caso presente, $K = 2$). La densità complessiva è quindi:

$$p(x) = \sum_{k=1}^{K} \pi_k \, p_k(x) \ , \tag{3.6}$$

con $\hat{\pi}_k = \frac{n_k}{n}$, dove n_k è la numerosità della k-esima subpopolazione mentre n è la numerosità totale.
Per ogni soggetto con valori di X noti e pari a x_0, l'analisi discriminante punta a trovare il valore di k in cui allocare l'i-esima osservazione che massimizzi la **funzione discriminante**

$$d_k(x_0) = \log \pi_k + \log p_k(x_0) \ . \tag{3.7}$$

Per poter applicare questa procedura è però necessario conoscere p_k, stimato con due possibili approcci parametrici:

- **Analisi discriminante lineare**: $p_k(x)$ è funzione di densità normale multipla $N_p(\mu_k, \Sigma_k)$ per cui risulta

$$p_k(x) = \frac{1}{(2\pi)^{\frac{p}{2}} det(\Sigma_k)^{\frac{1}{2}}} \exp\{-\frac{1}{2}(x-\mu_k)'\Sigma_k^{-1}(x-\mu_k)\} \tag{3.8}$$

  Viene inoltre assunta omoschedasticità tra gruppi, quindi

$$\hat{\Sigma}_k = \hat{\Sigma} = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)'(x_i - \hat{\mu}_k)$$

  uguale in ogni subpopolazione.

- **Analisi discriminante quadratica**: rimane l'ipotesi distributiva normale per le $p_k(x)$ ma cade l'assunzione di omoschedasticità fra le subpopolazioni, di conseguenza bisogna stimare un $\hat{\Sigma}_k$ per ogni k.

3.2.3 Albero di classificazione

Gli alberi di classificazione applicano a risposte qualitative l'idea alla base degli alberi di regressione: questi mirano ad approssimare una curva di regressione effettuando un partizionamento nello spazio delle p variabili esplicative, in modo da ottenerne una rappresentazione parsimoniosa tramite una funzione a gradini definibile come:

$$\hat{f}(x) = \sum_{j=1}^{J} \hat{c}_j \ I(x \in R_j) \tag{3.9}$$

dove R_j sono le regioni individuate dopo aver effettuato il partizionamento e $\hat{c}_j$ sono le medie di ogni j-esimo gruppo.
La procedura prevede due fasi: la fase di crescita, durante la quale vengono esplorate tutte le possibili suddivisioni per poi effettuare quelle che portano al guadagno maggiore (nel caso degli alberi di regressione, viene inteso come riduzione della devianza all'interno delle regioni) e la fase di potatura, la quale mira ad ottimizzare il numero di foglie presenti introducendo nella funzione obiettivo un parametro di penalizzazione della complessità dell'albero.

Nel caso in esame la risposta è dicotomica, di conseguenza si mira ad approssimare $p(x) = P\{y = 1|x\}$, cioè la probabilità che l'i-esimo documento con caratteristiche

x sia una fake news, tramite una funzione a gradini del tipo

$$\hat{p}(x) = \sum_{j=1}^{J} P_j \, I(x \in R_j) \ , \tag{3.10}$$

dove P_j rappresenta la probabilità che $y = 1$ nella j-esima regione.
Effettuando un nuovo partizionamento, si crea una nuova foglia con associato valore di $\hat{p}(x)$, la quale farà allocare R_j alla classe delle notizie reali se $\hat{p}(x) \leq 0.5$, mentre verrà classificata come fake news se maggiore della soglia di separazione fissata.
Negli alberi di classificazione, non si userà la somma delle devianze all'interno delle regioni come criterio di accostamento per effettuare nuovi tagli, ma un indice di impurità D, il quale corrisponde alla media delle **entropie** Q delle foglie pesate con le loro numerosità:

$$D = 2n \sum_{j}^{J} \frac{n_j}{n} \, Q(\hat{P}_j) \quad \text{con} \quad Q(\hat{P}_j) = - \sum_{k=0,1} P_{jk} \log P_{jk} \ . \tag{3.11}$$

Un'alternativa alla misura di entropia per quantificare l'impurità della regione è l'**indice di Gini**:

$$Q(\hat{P}_j) = \sum_{k=0,1} P_{jk}(1 - P_{jk}) \ . \tag{3.12}$$

La fase di potatura rimane uguale sia in alberi di regressione che classificazione: si definisce la funzione obiettivo

$$C_\alpha(J) = \sum_{j=1}^{J} D_j + \alpha J \ , \tag{3.13}$$

dove α è il **parametro di penalizzazione** della dimensione J dell'albero.
L'obiettivo della fase di potatura è quello di minimizzare C_α: dunque, dopo aver terminato la fase di crescita ed essere giunti a $J{=}n$, si incrementa il valore di α che porta all'esclusione sequenziale di quelle foglie la cui eliminazione comporta il minor incremento di $\sum_j D_j$, ottenendo così l'albero ottimale.

3.2.4 Foresta casuale

Gli alberi di classificazione sono dei metodi molto instabili, in quanto i risultati possono cambiare notevolmente a seconda dell'insieme dei dati di train presi in considerazione.
Una possibile soluzione per migliorare la classificazione è rappresentata dalle foreste casuali: queste infatti sono combinazioni di alberi applicati a una selezione casuale di F variabili esplicative.
L'idea alla base della fase di crescita degli alberi rimane uguale a quella spiegata nel paragrafo 3.2.3 ma, in questo caso, a ogni nuovo nodo non vengono prese in considerazione tutte le possibili suddivisioni delle variabili esplicative per poi scegliere il "taglio" che porti al maggior guadagno: infatti, vengono esplorate solo F variabili scelte casualmente. Inoltre, non è presente la fase di potatura in quanto

il numero ottimale di foglie viene selezionato tramite la combinazione dei diversi alberi.
Dunque, sono presenti due parametri di regolazione da ottimizzare: il numero di F variabili da selezionare per ogni nuova suddivisione, selezionando quella che porta ad un errore di classificazione inferiore, e il numero B di alberi che costituiscano la foresta.
Per ottimizzare ulteriormente i risultati ottenuti, è possibile creare ogni albero della foresta su n campioni **bootstrap** invece che sull'intero dataset: quindi, per ogni B-esima iterazione, vengono campionati n sottoinsiemi dei dati su cui effettuare le suddivisioni sulle K variabili. Per ogni nuovo nodo, viene poi effettuata la media dei risultati ottenuti su ogni n-esimo campione bootstrap, così da ottenere risultati più robusti.

3.2.5 Gradient boosting

Un'alternativa alla Foresta Casuale per aumentare l'efficacia predittiva degli alberi di classificazione è l'algoritmo *Gradient Boosting*, il quale mira all'ottimizzazione della selezione delle variabili.
Alla base della procedura c'è il concetto di **boosting**: questa è una strategia che prevede l'adattamento di più modelli in modo sequenziale, i quali utilizzano lo stesso insieme di dati ma con probabilità di entrata diversa a ogni iterazione, a seconda di come sono state classificate al passo precedente (verrà assegnato un peso maggiore a quelle che portano a un errore di predizione maggiore).
In primo luogo si definisce la **funzione di perdita** considerata, la quale deve essere differenziabile: in questo lavoro si è presa in considerazione la log-verosimiglianza $L(y, F_i)$ della binomiale, dove F_i sono i valori previsti dall'i-esimo albero.
La procedura inizia calcolando il valore della funzione di perdita al primo albero F_1 adattato: da qui, per tutte le M iterazioni, il modello al passo $m+1$ verrà adattato cercando di ridurre la somma di tutte le m funzioni di perdita delle m iterazioni precedenti.
Il nuovo albero risulta:

$$F_{m+1} = F_m - \frac{\partial L(y, F_m(x))}{\partial F_m(x)} \ \gamma_{m+1} \ , \tag{3.14}$$

dove γ_{m+1} è il valore che porta all'ottimizzazione della perdita, in quanto viene calcolato come

$$\hat{\gamma}_{m+1} = argmin \sum_{i=1}^{N} L(y_i, F_m + \gamma_m \, F_{m+1}) \ . \tag{3.15}$$

3.2.6 Scelta degli iperparametri

Come nell'albero di classificazione bisogna regolare il numero massimo di profondità per poter arrivare alle capacità predittive migliori, anche nella *foresta casuale* e nel *gradient boosting* devono essere ottimizzati dei parametri che controllano la precisione del modello: questi vengono chiamati iperparametri.

Nel caso della foresta casuale bisogna regolare: il numero B di alberi presenti nella foresta, il numero di K variabili da considerare per la creazione di un nuovo nodo, l'indice d'impurità, calcolabile con l'entropia (3.13) o l'indici di Gini (3.14), e l'utilizzo o meno dei campioni bootstrap per la costruzione degli alberi.
Per il gradient boosting invece si sono ricercati i valori ottimi per M, definito come numero totale di iterazioni, e il parametro γ di apprendimento, il quale minimizza la funzione di perdita nella creazione del nuovo albero.

Per ottenere gli iperparametri ottimali si è effettuata una **ricerca a griglia**: questa, una volta definiti i possibili valori assumibili da ogni parametro, effettua una ricerca esaustiva esplorando tutte le possibili combinazioni, cercando quella che restituisce il tasso di accuratezza maggiore.
Data la complessità computazionale di tutto il processo, si è voluto introdurre un ulteriore partizionamento dei dati: quindi sono stati presi 21000 documenti dal training set per formare il *validation set*. In questo modo l'aggiustamento degli iperparametri avviene su un data set di train di dimensioni inferiori rispetto a quelle iniziali (ora contiene 32517 testi) e le valutazioni vengono effettuate sul validation set. Il data set di test viene utilizzato per confermare i risultati ottenuti sul validation set.

Per la foresta casuale si indagano 352 possibili combinazioni di parametri:

- B viene ricercato in un intervallo tra 1000 e 3000 di lunghezza 11
- K assume valori in un intervallo di 8 elementi da 1 a 20, il quale comprende i valori di default $\sqrt{p}$ o $\log(p)$, dove p è il numero delle variabili totali (44)
- l'indice di impurità può essere *"entropy"* o *"gini"*
- l'utilizzo o meno del bootstrap regolato dalla presenza dell'input "*bootstrap='True'*".

Invece, per il gradient boosting, si sono esplorate 220 possibili modelli con il valore M che varia in un intervallo di 11 elementi da 1000 a 3000, mentre il tasso di apprendimento può assumere 20 diversi valori tra 0 e 1.

Gli iperparametri considerati si riferiscono agli input richiesti dalle funzioni *"RandomForestClassifier"* e *"GradientBoostingClassifier"* della libreria **sklearn.ensemble**.

Capitolo 4

Applicazione

4.1 Modelli

4.1.1 Modello di regressione logistica

Il primo passo nella fase di applicazione dei modelli è stato adattare il modello additivo di regressione logistica con tutte le variabili concomitanti a disposizione. La formulazione matriciale diventa dunque:

$$logit(\pi) = X\beta \tag{4.1}$$

con X matrice a rango pieno di dimensioni $n \times p$, dove n è il numero di osservazioni nel training set, pari a 53517, e p è il numero di variabili esplicative compresa l'intercetta, quindi 45.
Impostata la soglia di separazione delle due classi a 0.5, il primo modello adattato porta alla seguente matrice di confusione:

Tabella 4.1: Matrice di confusione regressione logistica

Predetti/Osservati	Real	Fake
Real	7435	1043
Fake	1327	8035

con un tasso di accuratezza del 87.02%.

Successivamente, si sono applicate diverse procedure di selezione automatica delle variabili per selezionare il modello preferenziale rispetto ai dati considerati:

- **selezione passo-a-passo**: questa procedura prevede di modificare il modello iniziale togliendo (*selezione all'indietro*) o aggiungendo (*selezione in avanti*) le variabili che portano ad un abbassamento dell'**AIC**(Akaike Information Criterion), fermandosi quando non si possono ottenere altri miglioramenti. Applicando questa procedura automatica, si è selezionato un sottoinsieme di 41 variabili, escludendo solo i punteggi TFIDF dei vocabili "time" e "one", la percentuale di parole riferite all'emozione "joy", e la percentuale di parole inerenti al topic denominato "Politica Estera".
  Il tasso di accuratezza ottenuto è pari a 86.99%, quindi leggermente inferiore al modello iniziale.

- **best subsect selection**: questo algoritmo mira a selezionare il migliore sottoinsieme di variabili tra tutte le combinazioni possibili, non cercando un percorso tra esse.
  Nella Figura 4.1 viene rappresentato come varia l'indice AIC a seconda delle dimensioni del sottogruppo considerato. Questa procedura porta a selezionare il modello con 42 variabili, escludendo anche qui i punteggi riferiti a "time" e "one", e in aggiunta gli scores riferiti all'argomento denominato con "Russia". Il tasso di accuratezza risulta pari a 86.95%.

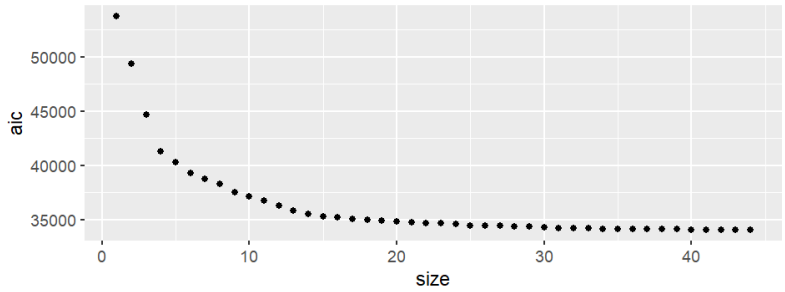

Figura 4.1: AIC al variare della dimensione del sottoinsieme di variabili

I tre modelli raggiungono tassi di accuratezza molto simili, con un leggero vantaggio per il modello completo. Tuttavia, il modello preferibile tra i tre risulta quello selezionato con il metodo *passo-a-passo*, in quanto più parsimonioso:

Tabella 4.2: Confronto tra modelli

	AIC	Accuratezza
Modello completo	34076.89	0.8706
Selezione passo-a-passo	34069.59	0.8705
Best subsect selection	34074.15	0.8704

4.1.2 Analisi discriminante

In questa fase dell'applicazione dei metodi, si sono adattati l'analisi discriminante sia lineare che quadratica, spiegate al capitolo 3.3.2.
Le due procedure, giungono a risultati molto diversi, portando alle seguenti matrici di confusione:

Tabella 4.3: Matrice di confusione LDA

Predetti/Osservati	Real	Fake
Real	7295	1467
Fake	914	8164

Tabella 4.4: Matrice di confusione QDA

Predetti/Osservati	Real	Fake
Real	8026	736
Fake	3360	5718

Notiamo che la tabella 4.3 giunge a risultati simili a quelli della regressione logistica, classificando bene quindi i veri positivi e i veri negativi, con un tasso di accuratezza del 86.67%. L'assunto di eteroschedasticità invece porta a stimare un numero molto elevato di falsi positivi (notizie reali classificate come false), causando infatti una forte diminuzione dell'accuratezza, che risulta 77.04%.

4.1.3 Albero di classificazione

Come introdotto nel capitolo 3.2.3, per arrivare all'albero di classificazione ottimale, non è sufficiente applicare l'algoritmo di crescita, in quanto si giungerebbe a problemi di *overfitting* e a un albero troppo complesso.
Per ottimizzare la procedura di potatura si è voluto regolare il **parametro di complessità** α, dalla (3.15). Infatti, si sono eseguiti diversi esperimenti in cui si è fatto variare il parametro α in tutti i suoi possibili valori, confrontando i risultati di accuratezza ottenuti nel test set.

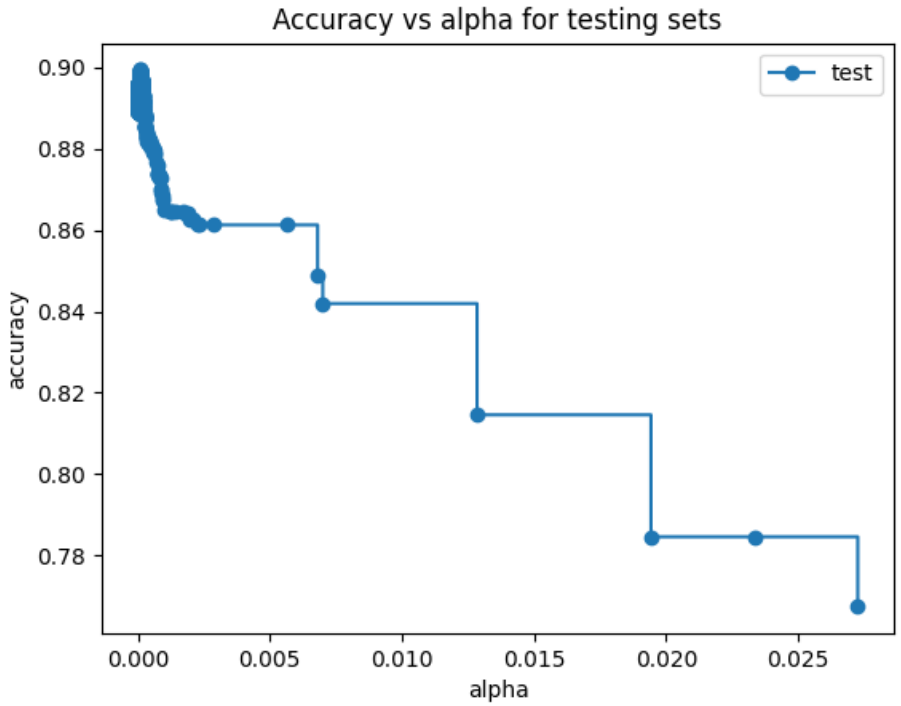

Figura 4.2: Confronto accuratezza e parametro di complessità

Dalla figura 4.2 notiamo che all'aumentare del parametro di complessità diminuisce l'accuratezza, in quanto si penalizza sempre di più un maggiore numero di foglie, portando ad alberi molto potati.
Dunque, si seleziona α dove lo score del test set è maggiore: con questo valore, pari circa a 0.00014, la massima profondità raggiunta dell'albero è 15.
I risultati della figura 4.2 sono stati ottenuti con devianza calcolata tramite l'*indice di Gini*, in quanto parametro di default; tuttavia, per ottenere l'albero che arriva alle previsioni migliori, oltre il parametro di profondità, si desidera modificare anche l'indice di impurità di default con l'*entropia*. Per fare ciò si sono adattati 15 alberi, ognuno con una profondità massima consentita tra 5 e 20 (dato che la massima accuratezza per Gini viene raggiunta con 15) registrando i tassi di accuratezza ottenuti così da poter confrontare i due metodi: nel grafico 4.3 i risultati.

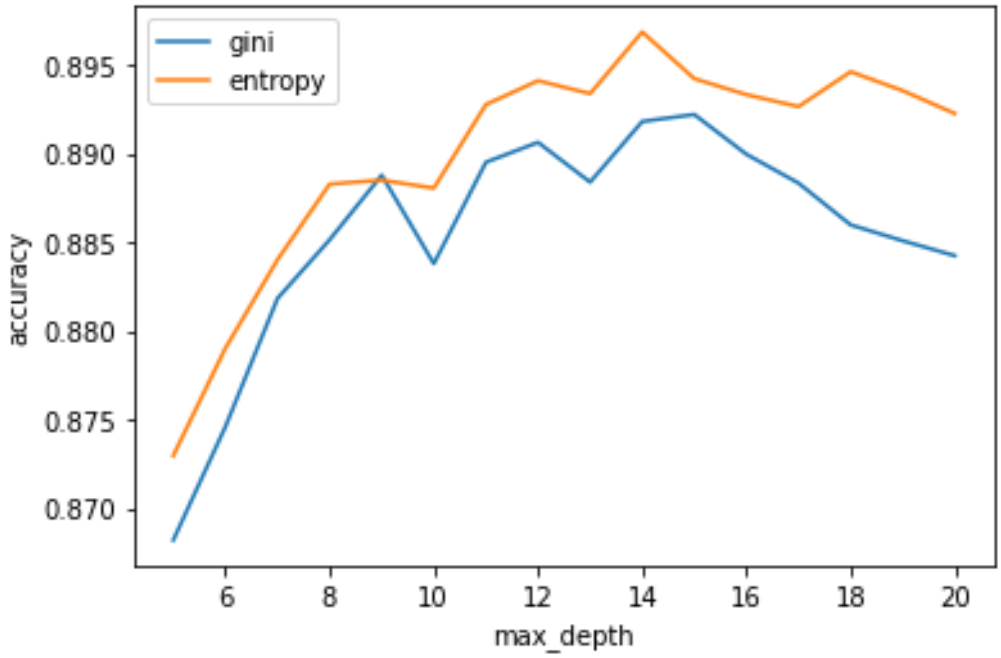

Figura 4.3: Confronto tra accuratezza con entropia e indice di Gini al variare della profondità dell'albero

Come si nota dal grafico, l'albero ottimale si ottiene dunque utilizzando l'entropia come indice di impurità in corrispondenza di un albero di profondità 14.
Impostando questi parametri si ottiene un tasso di accuratezza del 89.50% e la matrice di confusione 4.5.

Tabella 4.5: Matrice di confusione albero di classificazione

Predetti/Osservati	Real	Fake
Real	7682	1080
Fake	793	8285

4.1.4 Foresta casuale

Con l'utilizzo della funzione **sklearn.ensemble.GridSearchCV** si sono ricercate tutte le possibili combinazioni degli iperparametri elencati nel Capitolo 3.2.6.
La configurazione ottima è ottenuta per 2400 alberi presenti nella foresta, costruiti su campioni bootstrap considerando 20 variabili alla costruzione di ogni nuovo nodo. L'indice di impurità utilizzato è quello di entropia.
Adattando la foresta casuale sul data set di train con questa configurazione di iperparametri, si ottiene un tasso di accuratezza del 91.79% e la matrice di confusione 4.6.

Tabella 4.6: Matrice di confusione albero di classificazione

Predetti/Osservati	Real	Fake
Real	7905	857
Fake	608	8470

4.1.5 Gradient boosting

Dopo aver definito gli intervalli di ricerca delle variabili si sono calcolate le prestazioni di tutte le possibili 220 combinazioni di parametri, registrando per ognuna il tasso di accuratezza.
Come classificatore debole si è mantenuto l'albero ottimale ottenuto al 4.1.3, quindi con una profondità massima di 14 *split* e con l'entropia come indice di impurità: fissati questi parametri, si è giunti alla configurazione ottimale con un numero M di iterazioni pari a 1600 e un tasso di apprendimento pari a 0.4.
Il modello adattato con i parametri ottimi raggiunge un tasso di accuratezza del 92.51% con la matrice di confusione 4.7.

Tabella 4.7: Matrice di confusione gradient boosting

Predetti/Osservati	Real	Fake
Real	8038	724
Fake	613	8465

4.2 Confronto dei risultati

Come mostrato nella Tabella 4.8, i tassi di accuratezza e i punteggi F1 registrati per ogni metodo utilizzato raggiungono punteggi simili tra loro e quindi si può affermare di essere giunti a dei risultati robusti.
Il modello che giunge a previsioni migliori è il gradient boosting, seguito dalla foresta casuale. Anche i metodi più semplici sono risultati idonei alla classificazione, tuttavia la quantità da minimizzare in un problema di fake news detection sono i falsi negativi, quindi quelle news classificate come notizie reali ma

Tabella 4.8: Confronto dei risultati

Modello	**Accuratezza**	**F1 score**
Modello di regressione lineare	0.8706	0.8741
Best subsect selection	0.8705	0.8740
Selezione passo-a-passo	0.8704	0.8739
Analisi discriminante lineare	0.8667	0.8727
Analisi discriminante quadratica	0.7704	0.7363
Albero di classificazione	0.8950	0.8984
Foresta casuale	0.9179	0.9204
Gradient boosting	0.9251	0.9268

che in realtà sono fake news. Per esempio l'albero di classificazione giunge a un tasso di accuratezza molto elevato, ma il numero di falsi negativi risulta oltre i 1000. Dunque, il gradient boosting risulta preferibile non solo per l'alto numero di osservazioni classificate correttamente, ma anche per aver il tasso minore di FN.

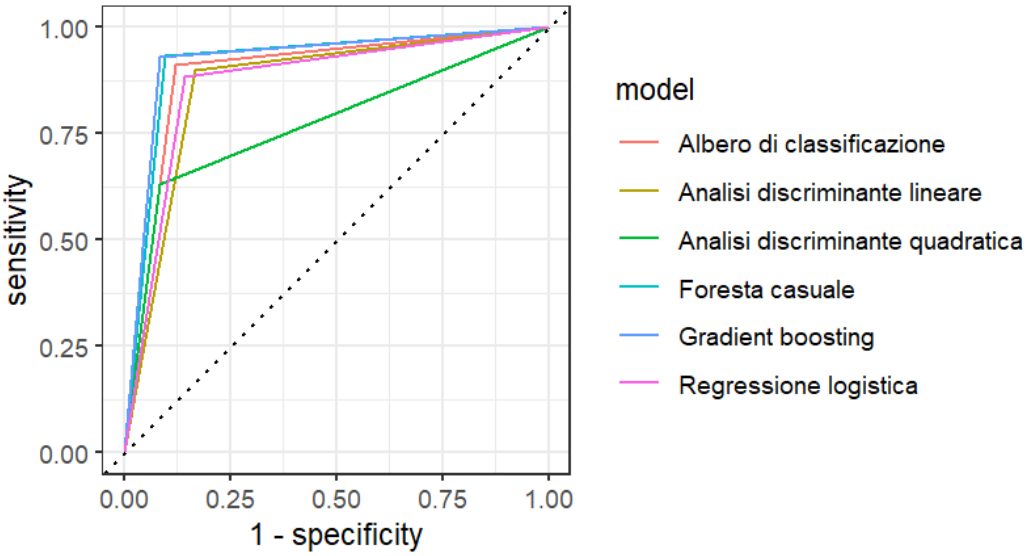

Figura 4.4: Curve ROC

Conclusioni

Tutti i metodi considerati hanno portato a buone capacità predittive, si può dunque giungere alla conclusione che le caratteristiche estratte dai documenti sono risultate dei buoni parametri per discriminare una notizia falsa da una vera.
In particolare, è possibile utilizzare l'output dell'albero di classificazione, della foresta casuale e del gradient boosting per ottenere il punteggio di importanza di ogni variabile; dato che il modello che ottiene previsioni migliori è il gradient boosting verrà presa di riferimento la 4.9, ma notiamo che la 4.10 e la 4.11 giungono a risultati simili.

Tabella 4.9: Gradi di importanza delle prime 10 variabili per il gradient boosting

Nomi variabili	Importanza
Punteggio TFIDF "said"	0.3241
Numero di citazioni	0.1204
Topic "Politica Estera"	0.0825
Topic "Repubblicani"	0.0614
Topic "CampagnaTrump"	0.0445
Topic "Societa"	0.0283
Numero medio di parole per frase	0.0280
Numero segni di punteggiatura	0.0229
Lunghezza testo	0.0121
Topic "Russia"	0.0150
Percentuale di emozione "positive"	0.0147
Punteggio TFIDF "hillary"	0.0139

Notiamo che la variabile più importante è il punteggio TFIDF di ogni documento per la parola "said": si pensi al fatto che in un testo in lingua inglese questa parola può essere messa prima di un discorso diretto o indiretto, anticipando dunque il riferimento a una fonte ufficiale. La presenza di citazioni infatti è il miglior modo per distinguere una notizia vera da una falsa, ipotesi confermata dal fatto che anche la variabile denominata "quotes" che conta i ' " ', presenti in concomitanza di discorsi diretti, è fra quelle con il punteggio di importanza maggiore.
Se guardiamo ai coefficienti del modello di regressione logistica adattato (appendice A), si nota che la stima di massima verosimiglianza sia del coefficiente "said" che "quotes" sono negativi, il che vuol dire che la probabilità che il documento i-esimo sia falso è minore per i documenti che hanno punteggi maggiori in corrispondenza di queste due variabili.

Nelle fake news notiamo invece molte parole per frase con una scarsa presenza di segni di punteggiatura, un numero elevato di parole con meno di 6 caratteri e una lunghezza media dei documenti più elevata: i testi con questa struttura sintattica risultano più semplici e chiari, portando dunque messaggi che colpiscano maggiormente il lettore in quanto più intuitivi.
Un altro aspetto significativo da considerare per distinguere la veridicità di un documento, è l'emozione espressa dalle parole utilizzate: infatti alla nona posizione per punteggio di importanza c'è "positive". Questa variabile esprime la percentuale di parole riferite a emozioni positive presenti nel testo. Il coefficiente stimato nel modello di regressione logistica è negativo, quindi come le citazioni, contribuiscono negativamente alla probabilità che l'i-esima osservazione sia una fake news. L'effetto contrario invece è prodotto dall'emozione "rabbia", la quale è la seconda emozione più "importante".
Per quanto riguarda le variabili rilevate con il topic modelling, dai punteggi di importanza si nota che rivestono un ruolo importante nelle discriminazione delle notizie vere e false: infatti i documenti con un alto numero di parole riferite ai topic denominati come politica estera americana, campagna elettorale di Donald Trump e rapporti degli Stati Uniti con la Russia, sono soprattutto fake news; al contrario invece il topic "Società" sembra entrare negativamente nel modello, indicando maggiore probabilità di avere una notizia reale per i documenti che ne hanno un punteggio alto.

I risultati a cui si è giunti non si possono generalizzare a tutti i problemi di fake news detection, soprattutto per i discorsi riguardanti i topic, in quanto variabili rilevate esclusivamente nei dati in questione; tuttavia, l'elevato numero di documenti di genere così vario presenti nel data set, portano ad un abbassamento della distorsione di un solo insieme di dati e quindi a risultati più robusti.

La minaccia rappresentata dal diffondersi di fake news nel web non è un fenomeno ignorabile e tutti gli utenti di social network o siti d'informazione online devono prestare attenzione a non contribuire all'allargamento del fenomeno. I metodi automatici per far fronte alla questione, come si è analizzato nel presente lavoro, possono portare a risultati molto efficaci, rappresentando dunque una base affidabile: tuttavia, affidarsi a siti ufficiali o verificare una notizia da più fonti, è il modo più semplice per assicurarsi una buona qualità delle informazioni recepite.

Tabella 4.10: Gradi di importanza delle prime 10 variabili per foresta casuale

Nomi variabili	Importanza
Punteggio TFIDF "said"	0.2265
Numero di citazioni	0.1321
Topic "Politica Estera"	0.1232
Topic "Repubblicani"	0.0682
Topic "Campagna Trump"	0.0474
Numero medio di parole per frase	0.0307
Numero di segni di punteggiatura	0.0264
Lunghezza testo	0.0251
Topic "Società"	0.0228
Percentuale di emozione "positive"	0.0159
Punteggio TFIDF "hillary"	0.0153

Tabella 4.11: Gradi di importanza delle prime 10 variabili per albero di classificazione

Nomi variabili	Importanza
Punteggio TFIDF "said"	0.2902
Topic "Politica Estera"	0.1598
Numero di citazioni	0.0956
Topic "Repubblicani"	0.0767
Numero medio di parole per frase	0.0326
Topic "Societa"	0.0269
Lunghezza testo	0.0246
Topic "Campagna Trump"	0.0126
Percentuale di parole "corte"	0.0123
Topic "Russia"	0.0118

Appendice A

Output modello di regressione lineare

Logit Regression Results

Dep. Variable:	label	No. Observations:	53517
Model:	Logit	Df Residuals:	53473
Method:	MLE	Df Model:	43
Date:	Mon, 31 Oct 2022	Pseudo R-squ.:	0.5411
Time:	11:50:41	Log-Likelihood:	-17019.
converged:	True	LL-Null:	-37086.
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
Interc	1.9046	0.081	23.639	0.000	1.747	2.063
length	0.3474	0.019	18.754	0.000	0.311	0.384
perc5	0.4917	0.025	19.355	0.000	0.442	0.542
perc11	0.5792	0.023	25.372	0.000	0.534	0.624
av_words	-1.7602	0.088	-19.981	0.000	-1.933	-1.588
punct	-0.8317	0.026	-32.208	0.000	-0.882	-0.781
quotes	-0.1056	0.021	-4.954	0.000	-0.147	-0.064
fear	-0.0263	0.020	-1.334	0.182	-0.065	0.012
anger	0.2696	0.022	-12.168	0.000	-0.313	-0.226
trust	-0.0480	0.016	-2.993	0.003	-0.079	-0.017
surprise	-0.3305	0.025	-13.155	0.000	-0.380	-0.281
positive	-0.1719	0.022	-7.695	0.000	-0.216	-0.128
negative	-0.1800	0.019	-9.643	0.000	-0.217	-0.143
sadness	0.0520	0.018	2.856	0.004	0.016	0.088
disgust	0.0062	0.018	0.341	0.733	-0.030	0.042
joy	-0.1695	0.019	-8.938	0.000	-0.207	-0.132
anticipation	-0.0386	0.592	-0.065	0.948	-1.199	1.122
PoliticaEstera	0.7158	0.385	1.858	0.063	-0.039	1.471
Attualita	1.4813	0.615	2.409	0.016	0.276	2.686
CampagnaTrump	0.7275	0.341	2.131	0.033	0.058	1.397
Repubblicani	-0.1553	0.493	-0.315	0.753	-1.122	0.811
Russia	0.8182	0.321	2.546	0.011	0.188	1.448

Arresti	0.6018	0.357	1.687	0.092	-0.097	1.301
InvestClinton	0.5204	0.439	1.185	0.236	-0.340	1.381
Società	-0.0791	0.018	-4.324	0.000	-0.115	-0.043
trump	0.0695	0.017	4.130	0.000	0.037	0.102
us	-1.1687	0.019	-61.737	0.000	-1.206	-1.132
said	-0.1162	0.015	-7.611	0.000	-0.086	-0.146
people	-0.0388	0.016	-2.392	0.017	-0.071	-0.007
would	-0.1358	0.017	-8.168	0.000	-0.168	-0.103
clinton	0.0003	0.015	0.019	0.984	-0.029	0.030
one	-0.2444	0.016	-15.320	0.000	-0.276	-0.213
president	0.0421	0.016	2.660	0.008	0.011	0.073
like	0.1524	0.016	9.743	0.000	0.122	0.183
obama	-0.1078	0.014	-7.453	0.000	-0.136	-0.079
election	-0.1336	0.017	-7.918	0.000	-0.167	-0.101
donaldatrump	0.7985	0.041	19.276	0.000	0.717	0.880
hillary	-0.2311	0.015	-15.451	0.000	-0.260	-0.202
campaign	-0.1289	0.014	-8.951	0.000	-0.157	-0.101
also	0.1122	0.016	7.170	0.000	0.082	0.143
even	-0.0103	0.015	-0.707	0.479	-0.039	0.018
time	-0.0925	0.015	-5.985	0.000	-0.123	-0.062
state	0.1525	0.020	7.744	0.000	0.114	0.191
media	-0.0632	0.014	-4.561	0.000	-0.090	-0.036

Number of Fisher Scoring iterations: 7

Appendice B

Applicazione dei metodi

```
import pandas as pd
import numpy as np
from sklearn import metrics
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score

from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis,
      QuadraticDiscriminantAnalysis
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier

from sklearn.model_selection import GridSearchCV
import matplotlib.pyplot as plt
import random

df_train = pd.read_csv('train_modelling.csv')
df_test = pd.read_csv('test_modelling.csv')

y_train = df_train['label']
X_train = df_train.loc[:, df_train.columns != 'label']
y_test = df_test['label']
X_test = df_test.loc[:, df_test.columns != 'label']
random.seed(123)
rnd = random.sample(range(53517), 21000)
X_train1 = X_train.loc[rnd]
y_train1 = y_train.loc[rnd]
X_validation = X_train.drop(rnd)
y_validation = y_train.drop(rnd)

# REGRESSIONE LOGISTICA
model=LogisticRegression()
```

```
glm = model.fit(X_train, y_train)
predictions_glm = glm.predict(X_test)
cm = confusion_matrix(y_test, predictions_glm)
f1_score(y_test, predictions_glm)
accuracy_score(y_test, predictions_glm)

# ANALISI DISCRIMINANTE
LDA = LinearDiscriminantAnalysis()
lda = LDA.fit(X_train, y_train)
QDA = QuadraticDiscriminantAnalysis()
qda = QDA.fit(X_train, y_train)

pred_lda = lda.predict(X_test)
pred_qda = qda.predict(X_test)

cm_lda = confusion_matrix(y_test, pred_lda)
cm_qda = confusion_matrix(y_test, pred_qda)
accuracy_score(y_test, pred_lda)
accuracy_score(y_test, pred_qda)
f1_score(y_test, pred_lda)
f1_score(y_test, pred_qda)

# ALBERO DI CLASSIFICAZIONE
clfs = []
for ccp_alpha in ccp_alphas:
    clf = DecisionTreeClassifier(random_state=0, ccp_alpha=ccp_alpha)
    clf.fit(X_train, y_train)
    clfs.append(clf)

clfs = clfs[:-1]
ccp_alphas = ccp_alphas[:-1]

test_scores = [clf.score(X_test, y_test) for clf in clfs]
ind = test_scores.index(max(test_scores))
train_scores[ind]
ccp_alphas[ind]
clfs[ind].tree_.max_depth

max_depth = []
acc_gini = []
acc_entropy = []
d = list(range(5,21))
for i in d:
    dtree = DecisionTreeClassifier(criterion='gini', max_depth=i,
                                   random_state=0)
    dtree.fit(X_train, y_train)
    pred = dtree.predict(X_test)
```

```
    acc_gini.append(accuracy_score(y_test, pred))
    #
    dtree = DecisionTreeClassifier(criterion='entropy', max_depth=i,
                                   random_state=0)
    dtree.fit(X_train, y_train)
    pred = dtree.predict(X_test)
    acc_entropy.append(accuracy_score(y_test, pred))
    #
    max_depth.append(i)
d[acc_gini.index(max(acc_gini))]
d[acc_entropy.index(max(acc_entropy))]

dtree = DecisionTreeClassifier(criterion='entropy', max_depth=14)
d_fit = dtree.fit(X_train, y_train)
pred_tree = d_fit.predict(X_test)
accuracy_score(y_test, pred_tree)
f1_score(y_test, pred_tree)
cm = confusion_matrix(y_test, pred_tree)

# FORESTA CASUALE
rf_grid = RandomForestClassifier(max_depth=14)
param_grid = {
    'n_estimators': np.linspace(1000, 3000, 11, dtype = int),
    'max_features': [1,2,4,6,7,10,15,20],
    'criterion': ['gini', 'entropy'],
    'bootstrap': ['True', 'False']
}
grid_rf_search = GridSearchCV(estimator = rf_grid, param_grid = param_grid,
                              n_jobs = 2, verbose = 2)
grid_rf = grid_rf_search.fit(X_train1, y_train1)
best_rf_grid = grid_rf_search.best_estimator_
pred_bestrf = best_rf_grid.predict(X_validation)
accuracy_score(y_validation, pred_bestrf)


rf = RandomForestClassifier(n_estimators=2400, criterion='entropy',
                            bootstrap='True', max_depth=14, max_features=20)
rf_fit = rf.fit(X_train1, y_train1)
pred_rf = rf_fit.predict(X_test)
cm = confusion_matrix(y_test, pred_rf)
accuracy_score(y_test, pred_rf)
f1_score(y_test,pred_rf)

importances_rf = rf.feature_importances_

# GRADIENT BOOSTING
gbc=GradientBoostingClassifier(max_depth=14)
```

```
param_grid_gbc = {
    'n_estimators': np.linspace(1000, 3000, 11, dtype=int),
    'learning_rate': np.linspace(0, 1, 11)}
grid_gbc_search = GridSearchCV(estimator=gbc, param_grid=param_grid_gbc,
                               verbose=2)
grid_gbc = grid_gbc_search.fit(X_train1,y_train1)
best_gbc = grid_gbc.best_estimator_
pred_bestgbc = best_gbc.predict(X_validation)
accuracy_score(y_validation, pred_bestgbc)

gbc = GradientBoostingClassifier(n_estimators=1600, learning_rate=0.4,
                                 max_depth=14)
gbc_fit = gbc.fit(X_train1,y_train1)
pred_gbc = gbc_fit.predict(X_test)
accuracy_score(y_test, pred_gbc)
cm = confusion_matrix(y_test, pred_gbc)
```

Appendice C

Selezione del modello di regressione

```
library(caret)

train <- read.csv("train_modelling.csv", row.names = 1)
test <- read.csv("test_modelling.csv", row.names = 1)

accuratezza <- function(y, y_pred){
  m <- confusionMatrix(y, y_pred)
  (m[1,1]+m[2,2]) / (m[1,1]+m[1,2]+m[2,1]+m[2,2])
}


glm1 <- glm(label ~ ., family = binomial, data = train)
pred1 <- predict(glm1, newdata = test, type="response")
acc1 <- accuratezza(test$label, pred1)
confusionMatrix(test$label, pred1)

# STEPWISE
step1 <- stats::step(glm1)
step2 <- stats::step(glm1, direction = "backward")

pred.step <- predict(step1, newdata = test, type="response")
acc2 <- accuratezza(test$label, pred.step)
confusionMatrix(test$label, pred.step)

# BEST SUBSECT SELECTION
library(leaps)
x_train <- as.matrix(train[,-1])
bss <- regsubsets(y = train$label, x=x_train,
                  nvmax = 44)

n.models <- 44
errore <- rep(NA, n.models)
for(j in 1:n.models) {
  c <- coef(bss, id=j)
```

```
  coef_names <- names(c[-1])
  red <- cbind(label=train$label,train[coef_names])
  test_red <- cbind(label=test$label,test[coef_names])
  mod <- glm(label ~ ., family = binomial, data = red)
  errore[j] <- AIC(mod)
}
index <- which.min(errore)
c <- coef(bss, id=index)
coef_names <- names(c[-1])
train_red <- cbind(label=train$label,train[coef_names])
test_red <- cbind(label=test$label,test[coef_names])
glm3 <- glm(label ~ ., family = binomial, data = train_red)
pred3 <- predict(glm3, newdata = test_red, type="response")
acc3 <- accuratezza(test$label, pred3)

# CONFRONTO
AIC(glm1,glm2,glm3)
cbind(acc1,acc2,acc3)
```

Bibliografia

[1] Hunt Allcott and Matthew Gentzkow (2017) *"Social media and fake news in the 2016 election"*, Technical report, National Bureau of Economic Research

[2] Jeffrey Gottfried, Elisa Shearer (2016) *"News Use Across Social Media Platforms 2016"*, Pew Research Centre

[3] *Pizzagate conspiracy theory*, Wikipedia, retrieved on [5]

[4] Pawan Kumar Verma, Prateek Agrawal, Ivone Amorim, Radu Prodan (2021) *"WELFake: Word Embedding Over Linguistic Features for Fake News Detection"*, IEEE Transaction on computational social systems

[5] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang (2017) *"Fake News Detection on Social Media: A Data Mining Perspective"*, ACM SIGKDD Explorations Newsletter

[6] Saif M. Mohammad, Peter D. Turney (2013) *"NRC Emotion Lexicon"*, National Research Council Canada

[7] Maniz Shrestha (2018) *"Detecting Fake News with Sentiment Analysis and Network Metadata"*, Ealham.edu

[8] Magnani Fabio (2011) *"Sviluppo di tecniche di Text Mining per la classificazione semantica di documenti"*, AMSLaurea

[9] Giancarlo Ruffo, Alfonso Semeraro, Anastasia Giachanou, Paolo Rosso (2021) *"Surveying the research on fake news in social media: a tale of networks and language"*, Social and Information Networks (Arxiv)

[10] Harita Reddy, Namratha Raj, Manali Gala, Annappa Basava (2020) *"Text-mining-based Fake News Detection Using Ensemble Methods"*, International Journal of Automation and Computing

[11] B. Scarpa, A. Azzalini (2004) *"Analisi dei dati e data mining"*, Springer

[12] Jerome H. Friedman (2001) *"Greedy function approximation: a gradient boosting machine"*, The Annals of Statistics

[13] R. Plutchik (1980) *"Emotion: A Psychoevolutionary Synthesis"*, Harper & Row

Ringraziamenti

So che una laurea triennale è un traguardo che in molti raggiungono e che probabilmente non sarà l'obiettivo più difficile che dovrò perseguire nella mia vita, ma venendo da un liceo delle scienze umane e essendo entrata al primo anno non sapendo cosa fosse un logaritmo, sono molto soddisfatta di questo risultato, dato tutto l'impegno e la determinazione che mi ci sono voluti per raggiungerlo. Sicuramente però non sarei arrivata fino a qui con le mie sole forze, parte fondamentale del mio percorso sono state tutte le persone che mi hanno aiutata e sostenuta.
Dunque, ci tenevo anzitutto a ringraziare i miei genitori, le mie rocce, che hanno pianto e gioito con me in ogni mio passo, incoraggiandomi sempre a dare il massimo perchè conoscono ciò di cui sono capace. Grazie Papi per i consigli, le frasi di incoraggiamento e gli occhi pieni di felicità quando ti dico un mio successo. Grazie Mamma per i piatti di pesce per "diventare intelligente" le sere prime degli esami, gli abbracci quando andavano male e la pazienza nei periodi di maggiore stress.

Ringrazio il Professor Antonio Canale per essere stata una guida disponibile, paziente e stimolante.
Grazie alla Professoressa Bianco del mio liceo, la quale mi ha sempre insegnato quanto sia importante essere uno studente diligente e furbo, ma soprattutto che il mio valore in quanto studente e persona non si ferma ad un voto.

Un ringraziamento speciale anche a Alessia, Francesca, Sofia e Veronica, che non hanno mai smesso di ricordarmi "chi è il leone". Vorrei trovare singole cose per cui dirvi grazie, ma la verità è che non sarei la persona che sono oggi senza di voi: siete l'unica certezza che voglio avere nella mia vita.

Grazie a Pietro, il mio tutor all'università e nella vita, per tutte le cose preziose che mi hai insegnato di cui farò sempre tesoro.
Grazie a Leonardo, il mio "compagno di banco", per tutti i pomeriggi di studi in cui ci si perdeva in ragionamenti inutilmente approfonditi ma che mi hanno aperto la mente. Non so come avrei fatto senza di te.

Un ringraziamento speciale a Riccardo, Alessia, Elisa, Gianmarco, Leonardo e Sofia: grazie per aver reso i miei anni universitari i miei preferiti di sempre, grazie per gli aperitivi infiniti post studio, per gli aperitivi infiniti invece dello studio, grazie per come mi fate sorridere il cuore, ma sopratutto grazie perchè quando non mi sentivo all'altezza del nostro corso di studi, voi mi facevate sentire sempre nel posto giusto, a casa.