High Performance & Scalable Analytics, NO-SQL Big Data Platforms
Project Report

Giovanni Cusumano Martina Formichini Giuseppe Minardi
Gilda Pepe Carmela Sgroi

Aprile 2021

Introduzione

Il dataset scelto si compone di due file csv (fake.csv e true.csv) contenenti una lista di articoli con notizie fake e vere, in un range temporale che va dal 2015 al 2018. Le notizie fake sono state collezionate dall'autore originario attraverso siti che sono stati classificati da Politifact (una organizzazione statunitense di fact-checking) e Wikipedia come siti non affidabili. Il file csv, contentente notizie vere, è stato invece ottenuto dall'autore originario attraverso il crawling degli articoli da `Reuters.com`.

Le features presenti in entrambi i dataset sono elencate nella tabella 1.

Tabella 1: Features presenti nel dataset.

Feature	Tipo
Title	String
Text	String
Subject	String
Date	DateTime

L'obiettivo che ci siamo posti con questo lavoro è quello di costruire un classificatore che sia in grado di distinguere una notizia veritiera da una fake.

Data preparation

Nelle fasi preliminari di Data preparation abbiamo riscontrato delle criticità nella funzione `spark.read.load` nell'eseguire il parsing dei campi testuali dei dataset. Questi, infatti, nel nostro caso includono stringhe molto "sporche", contenenti numerose tag HTML e caratteri non ASCII. A seguito di queste problematicità, abbiamo ritenuto opportuno effettuare una fase di early data processing sfruttando la libreria Pandas.

Per prima cosa, quindi, abbiamo aperto con Pandas i due file csv e creato una nuova colonna classe, assegnando il valore stringa *Real News* alle notizie vere e il valore *Fake News* a quelle false. Grazie a ciò, abbiamo potuto unire i due dataframe senza perdere l'informazione riguardante la veridicità delle news.

A questo punto abbiamo sistemato le features *title* e *text*, utilizzando **Beautiful Soup** per fare lo stripping delle tag HTML, portando i testi così ottenuti in minuscolo e sostituendo,

infine, i caratteri non ASCII con spazi vuoti. Per quanto concerne le date, abbiamo eliminato le righe con lunghezza maggiore di 18 caratteri in quanto notizie contenenti solamente spam. Infine abbiamo salvato il file csv e abbiamo effettuato il load con pyspark.

Data understanding

Classe e categoria

Dopo esserci assicurati della non presenza di "null values" abbiamo potuto iniziare il vero e proprio processo di data understanding. Per prima cosa abbiamo raggruppato per classe, contando il numero di notizie vere e false e rappresentando graficamente il risultato ottenuto.

Il dataset presenta un totale di 42,380 news; le notizie vere sono in tutto 18,909 mentre quelle false 23,471. Nel grafico seguente abbiamo plottato le proporzioni delle due classi. In figura 1 abbiamo plottato la distribuzione di news vere e false, notando che le news fake sono circa il 5% in più rispetto a quelle vere, pertanto il dataset risulta essere relativamente bilanciato.

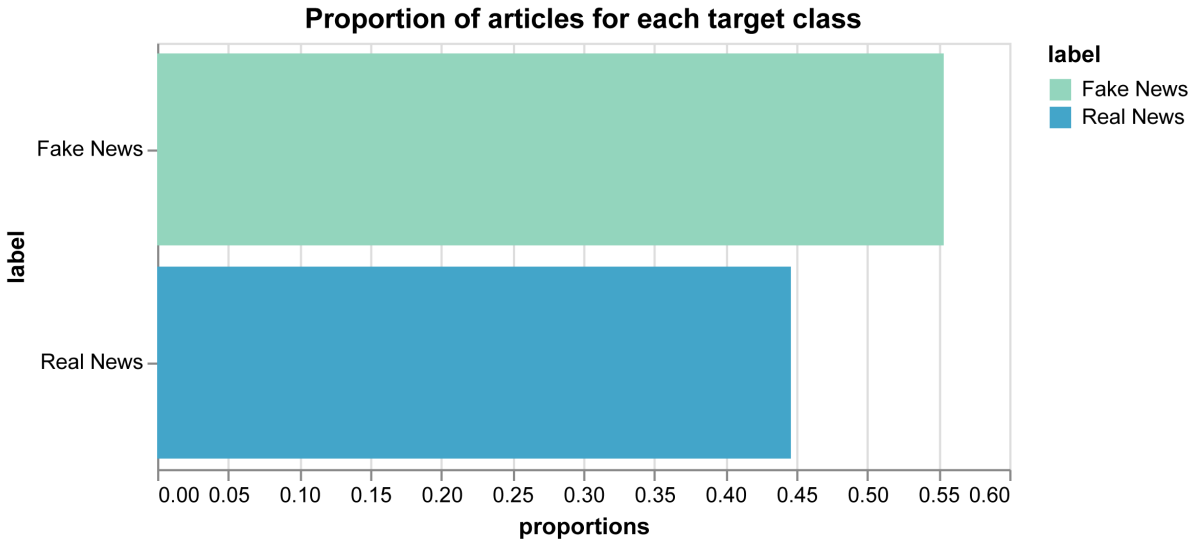
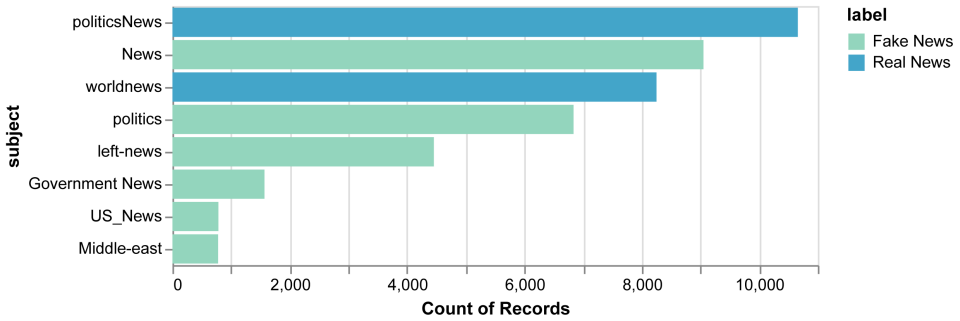


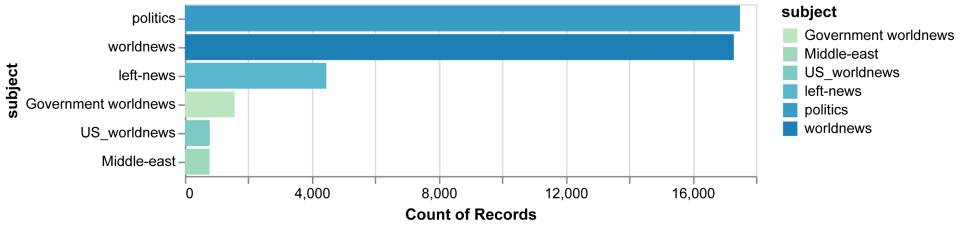
Figura 1: Numero di notizie vere e di notizie false

Abbiamo quindi osservato la distribuzione delle categorie (variabile *subject*, figura 2) tra le notizie vere e quelle false. A questo punto è emersa una differenza di etichettatura tra i due dataset originari (fake.csv e real.csv): nella categoria *Real News*, infatti, esistono due unici valori *politicsNews* e *worldnews*, mentre per quanto riguarda le notizie false è presente un numero maggiore di categorie, nello specifico: *News*, *politics*, *left news*, *Government news*, *US_news* e *Middle-east* (questa divisione può essere notata in figura 2a). Per poter meglio mettere a confronto i dati, in questa prima fase di data exploration, abbiamo deciso di accorpare le categorie dello stesso tipo (*politicsNews* e *politics*; *worldnews* e *News*). Le categorie così ottenute sono: *worldnews, politics, left-news, Government worldnews, US_worldnews, Middle_east* (figura 2b).

Poiché alcune subject degli articoli sono composte interamente da fake news e altre da real news, lasciare tale variabile nel dataset avrebbe inserito un bias troppo grande nel nostro



(a) Distribuzione originaria



(b) Distribuzione dopo feature engeneering

Figura 2: Distribuzione delle news per subject

classificatore finale. Abbiamo quindi deciso di non tenere in considerazione la categoria per l'allenamento dei modelli.

Analisi esplorativa del testo

Per poter condurre le successive analisi, abbiamo unito testo e titolo creando una nuova stringa e inserendola in una nuova colonna. La decisione di unire le due colonne è stata presa per evitare di effettuare due *TF-IDF* separati per titolo e testo. Con l'utilizzo di **nltk** abbiamo costruito una funzione per eliminare le stopwords e lemmatizzare. In questo modo abbiamo preparato il testo per la data exploration.

Anzitutto abbiamo analizzato la lunghezza media degli articoli di ciascuna classe. Quello che abbiamo osservato è che non c'è una differenza significativa in lunghezza fra le due classi (in media 249 per le notizie false e 238 per quelle vere). Abbiamo in seguito plottato la distribuzione della lunghezza degli articoli.

Come si può notare in figura 3, non ci sono significative differenze nella distribuzione; tuttavia, emerge che le notizie vere sono generalmente più corte rispetto alle fake. Per questa scarsa rilevanza abbiamo deciso di non tenere in considerazione la lunghezza degli articoli per il processo di classificazione. A questo punto, abbiamo esaminato le parole più frequenti per ciascuna delle due classi (a tal scopo abbiamo applicato un filtro per ogni classe). Osservando i risultati (figura 4), le parole significativamente più frequenti nelle notizie vere sembrano essere simili a quelle delle notizie false. Infatti, ad eccezione della parola *trump*, che compare nelle fake news in misura decisamente maggiore rispetto agli articoli con le notizie vere, anche nelle fake è preponderante la presenza di parole quali *president*, *people* e *state*. In aggiunta, tra le parole più frequenti figurano anche *obama* e *clinton*.

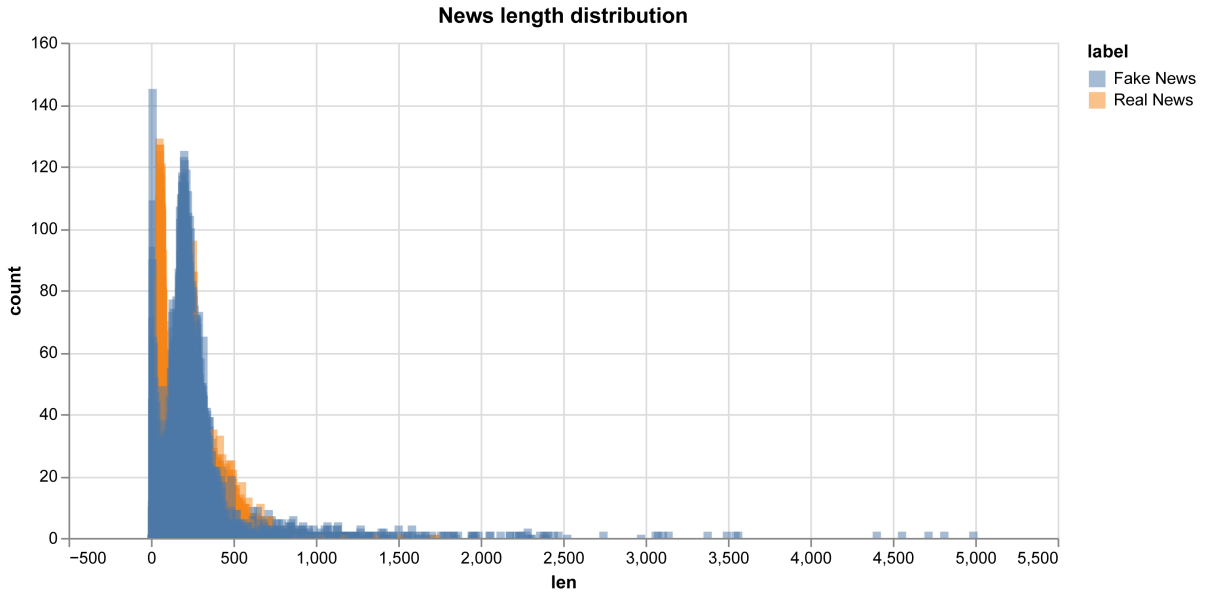


Figura 3: Distribuzione della lunghezza delle notizie.

Abbiamo, anche in questo caso, plottato i risultati per averne una più chiara visualizzazione. In ogni caso, in questa prima fase esplorativa, non è possibile notare a colpo d'occhio una differenza determinante fra la classe delle notizie vere e quella delle fake.

I risultati ottenuti, inoltre, non sorprendono considerando che le notizie sono state ottenute dall'autore del dataset da siti americani e che, come emerso dalle analisi precedenti, la metà delle notizie presenti sono di tipo politico.

Date

Al fine di estrarre il mese e gli anni dalla feature *DateTime* abbiamo creato un parser per le date. Plottando la distribuzione delle date è stato quindi possibile vedere come le notizie vere facciano riferimento agli anni 2016 e 2017, con un picco significativo tra ottobre e dicembre 2017. Le notizie false, al contrario, nonostante presentino lo stesso picco nei mesi sopraccitati, sono maggiormente presenti in tutti gli altri mesi, con valori che retrocedono nel tempo fino ad aprile 2015. Questa differente distribuzione è emersa in maniera ancora più evidente nel grafico normalizzato (figura 5a). In aggiunta abbiamo plottato anche la distribuzione delle subject normalizzata per anno (figura 5b).

Anche per le date, come nel caso delle subject, la differente distribuzione potrebbe inserire un bias nel processo di classificazione finale, inoltre ridurrebbe la generalizzazione dei classificatori che non riuscirebbero ad analizzare news successive al 2018 o antecedenti al 2015. Per queste ragioni abbiamo deciso di non includere la data nel nostro classificatore.

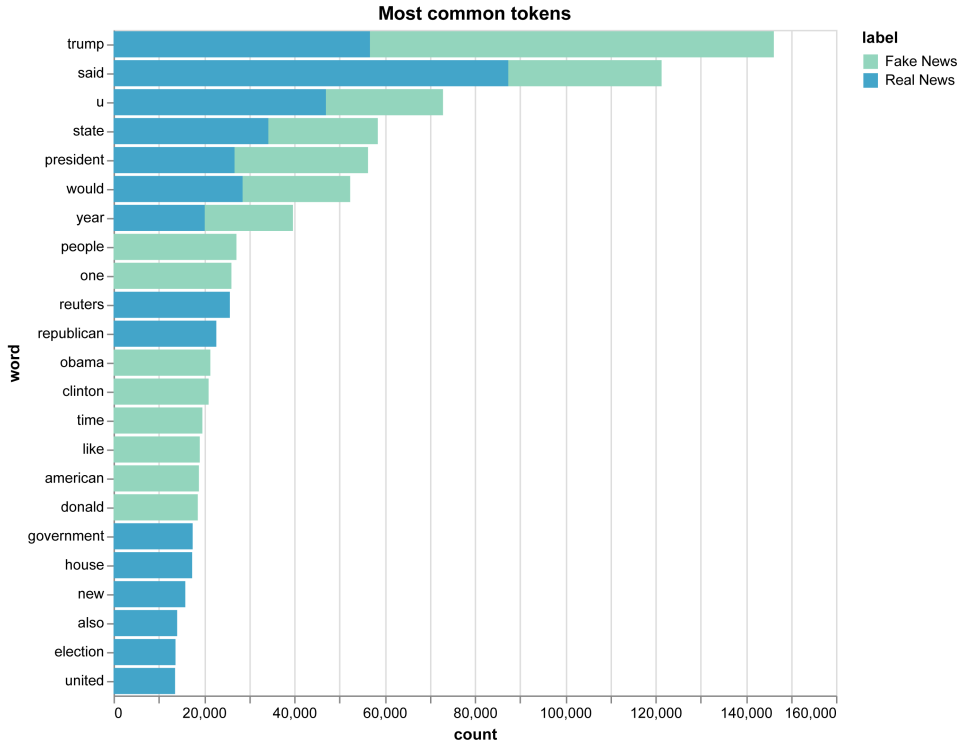
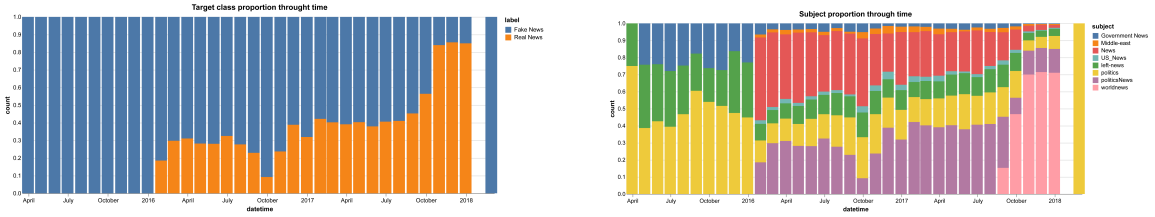


Figura 4: Parole più frequenti divise per appartenenza alle notizie fake e true.

CLASSIFICAZIONE

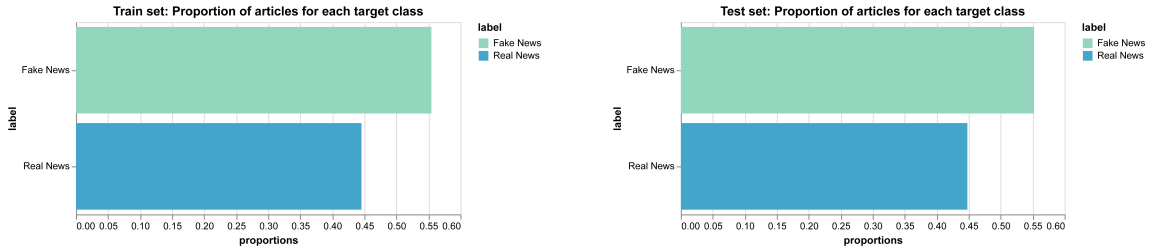
Per la parte di classificazione abbiamo suddiviso il dataset in train e test set con una ripartizione in 70%–30% e, per assicurarci che le proporzioni fra le classi siano state mantenute, in figura 6 è possibile osservare che la proporzione fra le classi nel test e nel train set sono uguali a quelle del dataset completo in figura 6. Quindi, per processare il dataset abbiamo innanzitutto tokenizzato i vari testi (che, come si ricorderà, sono composti da testo e titolo) per poi, successivamente, creare il TF-IDF. A questo punto, abbiamo usato come classificatore una Random Forest con 20 alberi. Non abbiamo usato un numero alto di stimatori perché l'algoritmo è risultato estremamente lento; tuttavia, anche con un numero non elevato il nostro classificatore ha raggiunto una



(a) Distribuzione di notizie Fake e vere nel tempo

(b) Distribuzione della variabile "subject" nel tempo

Figura 5: Proporzioni di label class e subject nel tempo



(a) Proporzione della variabile target nel train set **(b)** Proporzione della variabile target nel test set

Figura 6: Proporzioni del train e del test set a confronto

accuratezza dell'85%[1]. Per migliorare il nostro task di classificazione abbiamo dunque deciso di usare un Gradient–Boosted Tree Classifier. Il Gradient–Boosted Trees è lo stato dell'arte per la classificazione di dati tabulari; abbiamo rilevato, infatti, che pur mantenendo un numero basso di iterazioni (`maxIter=10`), il modello ha raggiunto il 99.5% di accuratezza. Avendo ottenuto una accuratezza così alta, non abbiamo ritenuto necessario eseguire la model selection con una Grid–Search Cross-Validation per trovare i migliori iperparametri per i nostri algoritmi.

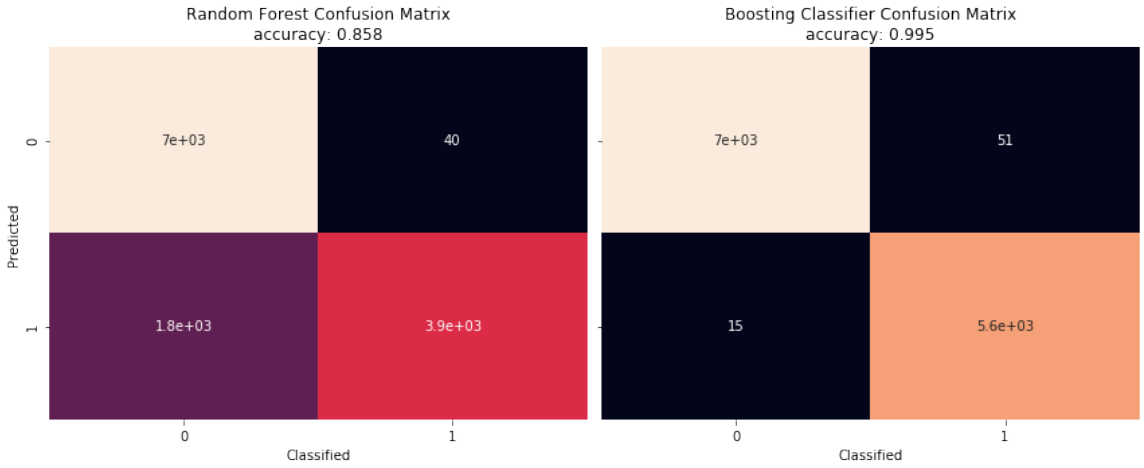


Figura 7: A sinistra: Confusion matrix della Random Forest; a destra: Confusion matrix del Gradient–BoostedTrees.

Nella figura 7 è possibile visualizzare la *confusion matrix* di entrambi i classificatori. Si può osservare, quindi, che la Random Forest ha avuto problemi a classificare correttamente le fake news, mentre il Gradient–BoostedTrees ha mostrato una performance nettamente migliore.

[1]Eventuali discrepanze con i risultati visibili nel codice allegato per l'esame sono da imputare al basso numero di alberi usati per la foresta e alla natura stocastica dell'algoritmo. Per avere una stima migliore avremmo dovuto effettuare una *cross–validation* ma i limiti temporali non ci hanno permesso di approfondire questo aspetto.