UNIVERSITÀ DEGLI STUDI DI MILANO – BICOCCA

Dipartimento di Informatica, Sistemistica e Comunicazione

Corso di Laurea Magistrale in Data Science

TEXT CLASSIFICATION PER INDIVIDUARE GLI ARTISTI A PARTIRE DAL TESTO DELLE LORO CANZONI

Project Team:

Inzadi Greta – 813649

Portaluppi Alessandro – 816090

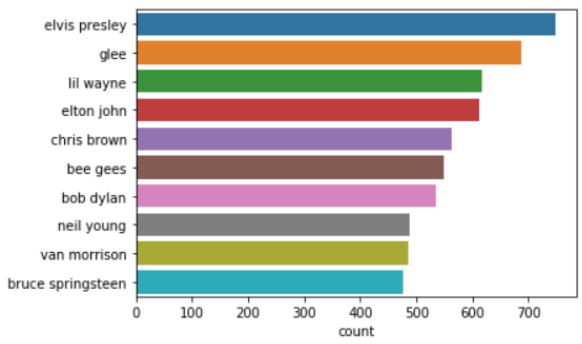
Testa Luca – 816000

Introduzione

A partire da un dataset contente il testo di un numero considerevole di canzoni, delle quali si conoscono l'artista e il genere corrispondenti, si è deciso di implementare una *text classification* per cercare un modello in grado di attribuire l'artista alla canzone in base al testo. Prima di procedere alla valutazione dei vari classificatori, sono state attuate le fasi di *pre-processing* necessarie per permettere ai modelli di utilizzare il testo.

Pre-processing del testo

Come prima cosa, dopo aver importato il dataset, sono stati rimossi dalle canzoni i testi corrispondenti alla dicitura "*instrumental*", dopodiché sono stati mantenuti unicamente quelli scritti in inglese e si è ridotto il dataset a quello contente solo gli artisti con più di 475 canzoni corrispondenti (i dieci più presenti), il tutto per ottenere un miglior risultato in termini di performance degli algoritmi.

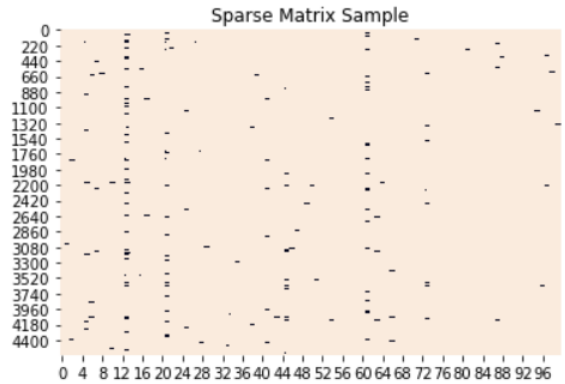


A partire dal testo, sono state ricavate alcune variabili utili ad allenare i vari classificatori nel riconoscimento degli artisti, quali: la lunghezza della canzone, la lunghezza media delle frasi al suo interno e la frequenza della punteggiatura. In seguito, è stata implementata la *POS tagging*, cioè l'assegnazione del rispettivo tag alle parti del discorso in una frase, al fine di estrapolare, per ogni artista, le categorie lessicali più utilizzate nella costruzione delle frasi, in modo tale da individuare una caratteristica del loro stile di scrittura. Per ogni canzone, sono stati conteggiati: nomi, verbi, avverbi, aggettivi, congiunzioni, parole straniere e interiezioni e sono state create variabili che contenessero rispettivamente i valori ottenuti. Si è proceduto alla normalizzazione dei testi, rimuovendo la punteggiatura, i numeri scritti in cifre e riportando tutte le lettere in minuscolo, è stata poi settata la lista delle *stop-word*. Si è calcolata la lunghezza media delle singole parole come altra caratteristica da prendere in considerazione nel modello. Una volta effettuata la tokenizzazione (divisione del testo in unità quanto più corrette sia dal punto di vista sintattico che semantico), dalla lista di *token* sono state rimosse le *stop-word*, e sui *token* rimasti sono state eseguite prima la procedura di lemmatizzazione, per riportare le parole al loro lemma (rimuovendo inflazioni, varianti e, ad esempio, riportando parole al singolare e verbi coniugati al modo infinito), e poi quella di *stemming,* riducendo i termini alla loro radice. Infine, è stato esportato il *dataframe* come file CSV su cui poter andare ad allenare e a testare i vari classificatori, per poi determinare il migliore tra essi.

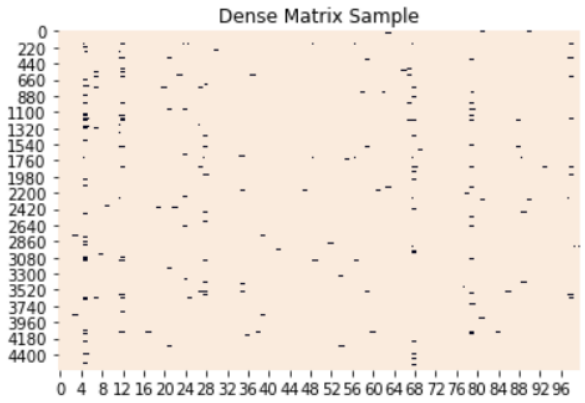
Rappresentazione del testo e classificazione

Si è suddiviso il dataset in *training-set* e *test-set* per valutare le performance dei vari classificatori, è stata identificata la variabile

contenente il nome dell'artista come target e la variabile "*lyrics*" come esplicativa. Si è utilizzato il *tf-idf* ("*term frequency–inverse document frequency*" per il quale il valore di una parola aumenta con l'incremento del conteggio di questa all'interno del testo, ma diminuisce al crescere della stessa nella collezione di testi, in modo da evidenziare le parole che sono più discriminanti tra i testi e più rappresentative degli stessi) come peso da assegnare ai token, presi in considerazione prima come unigrammi e poi come bigrammi e trigrammi (ma poi si è scelto di mantenere solo gli unigrammi, poiché dai test è risultato che con essi si ottenesse un'accuratezza più elevata). Si è utilizzata la funzione *TfidfVectorizer* sul testo processato all'interno del del *training set* per estrarre il vocabolario e costruire la matrice (4608 x 10000) contenente le feature con i rispettivi pesi.



Poiché quest'ultima risultava abbastanza sparsa (problema tipico della rappresentazione tramite *Bag-of-Words* e, in generale, della *text classification*), al fine di eliminare delle colonne e ridurre la dimensionalità della matrice, si è deciso di procedere con la *feature selection* (processo attraverso il quale si sceglie un sottoinsieme di feature rilevanti) basata sul test del Chi Quadrato, scegliendo come soglia un *p-value* pari a 0,95. Si è determinato così se una feature e un artista fossero indipendenti tra loro e solo le feature con un p-value maggiore o uguale alla soglia sono state mantenute. Successivamente si è proceduto a riassegnare i pesi *tf-idf* alle parole, sempre tramite la funzione *TfidfVectorizer*, al fine di ottenere una matrice più densa (4608 x 1526), contente solo le feature statisticamente più significative, su cui andare ad allenare i classificatori.



Gli algoritmi selezionati, trattandosi di un problema *single-label multi-class*, sono stati i seguenti: *Decision Tree*, *Random Forest*, *AdaBoost Classifier*, *Gradient Boosting Classifier*, *XGB Classifier*, *LGBM Classifier*, *KNN*.

Valutazione

Di essi sono stati calcolati l'accuratezza, l'AUC e il tempo di esecuzione.

classifier	accuracy	auc	run_time
DecisionTree	0.416667	0.670430	0.438007
RandomForest	0.549479	0.880982	0.782510
XtremeGradientBoosting	0.131076	0.534144	29.064715
LightGradientBoosting	0.561632	0.894388	6.441276
NaiveBayes	0.509549	0.872898	0.020061
KNN	0.357639	0.681456	0.004000

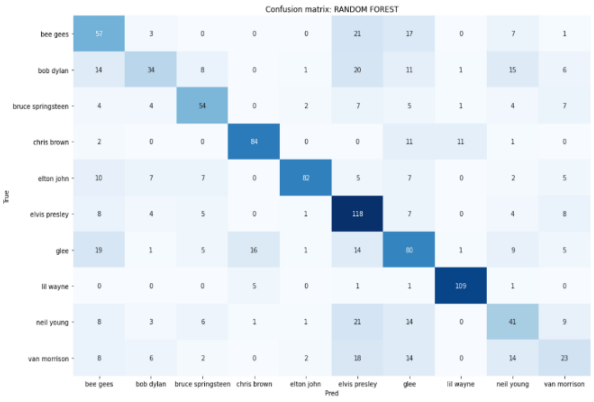
Si è ottenuto che i due con più elevata accuratezza sono il *Random Forest (*0,55) e *l'LGBM Classifier* (0,57). Per questi due sono state ricercate le variabili create a partire dal testo che fossero, rispettivamente, più significative per aumentarne l'accuratezza. Ogni variabile è stata aggiunta singolarmente ai classificatori e si è valutato il suo inserimento basandosi sul valore dell'accuratezza raggiunto con anche il suo contributo: se l'accuratezza era maggiore o uguale a quella ottenuta in precedenza, la variabile era aggiunta alle feature, altrimenti no. Nel primo caso (*Random Forest*) è stato raggiunto lo 0,59

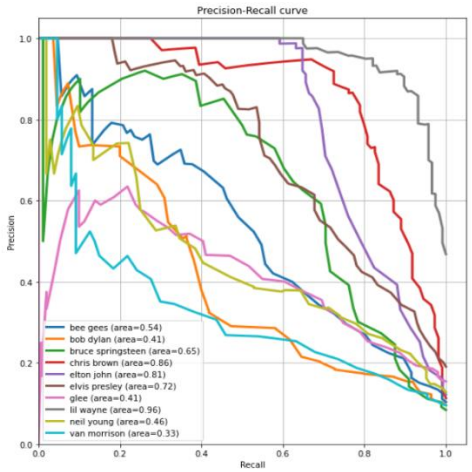
```
Accuracy: 0.59
Auc: 0.89
Detail:
                  precision    recall  f1-score   support

        bee gees       0.44      0.54      0.48       106
       bob dylan       0.55      0.31      0.40       110
bruce springsteen      0.62      0.61      0.62        88
     chris brown       0.79      0.77      0.78       109
      elton john       0.91      0.66      0.76       125
    elvis presley      0.52      0.76      0.62       155
           glee        0.48      0.53      0.50       151
      lil wayne       0.89      0.93      0.91       117
      neil young      0.42      0.39      0.41       104
    van morrison      0.36      0.26      0.30        87

        accuracy                           0.59      1152
       macro avg      0.60      0.58      0.58      1152
    weighted avg      0.60      0.59      0.59      1152
```

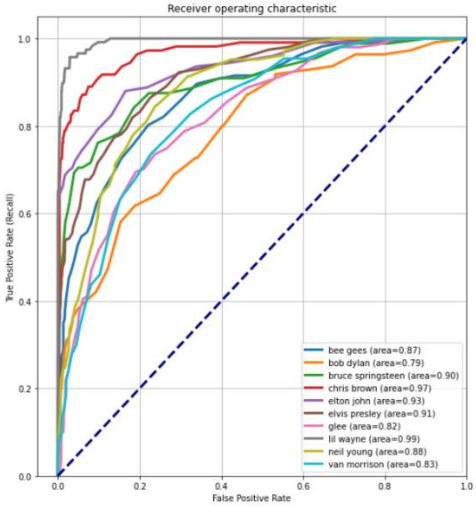
mentre nel secondo (*LGBM Classifier*) lo 0,62.

```
Accuracy: 0.62
Auc: 0.92
Detail:
                  precision    recall  f1-score   support

        bee gees       0.52      0.59      0.56       106
       bob dylan       0.43      0.38      0.40       110
bruce springsteen      0.63      0.59      0.61        88
     chris brown       0.83      0.79      0.81       109
      elton john       0.79      0.66      0.72       125
    elvis presley      0.60      0.67      0.64       155
           glee        0.51      0.54      0.53       151
      lil wayne       0.92      0.93      0.93       117
      neil young      0.47      0.53      0.50       104
    van morrison      0.52      0.45      0.48        87

        accuracy                           0.62      1152
       macro avg      0.62      0.61      0.62      1152
    weighted avg      0.63      0.62      0.62      1152
```

Si è considerata principalmente l'accuratezza (*effectiveness*), poiché si tratta della metrica migliore per valutare i problemi di classificazione del tipo *single-label multi-class*. In tal caso *LGBM Classifier* risulterebbe l'algoritmo migliore, tuttavia in termini di "performance" (*efficiency*, intesa come miglior prestazione nel minor tempo), è da preferire il *Random Forest*, poiché molto più rapido del *LGBM Classifier* e con un'accuratezza di poco più bassa. Per il *Random Forest* sono stati calcolati e rappresentati la matrice di confusione, l'AUC, la curva *Precision-Recall* e la curva di Roc.

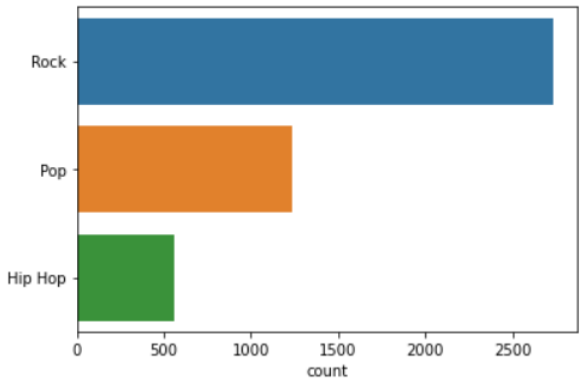






Visualizzazione

Sono stati realizzati dei *wordcloud* in base alle parole più frequenti per genere musicale.



Per il pop si è utilizzata l'immagine delle cuffie, per l'hip-hop del cappellino e per il rock della chitarra.

È stata creata un'applicazione web, attraverso la libreria *streamlit* (nata appunto per generare applicazioni e dashboard interattive con *Python*), per rappresentare nello spazio parole simili tra loro e quantificando questa similarità calcolando la distanza coseno tra i vettori della matrice di *word embedding* (costruita con l'algoritmo *Word2Vec*). L'utente può selezione l'apertura di questa "finestra" e una o più parole *target* per cui andare a visualizzare le parole di contesto (in numero arbitrario tra cinque e venti) in uno spazio bidimensionale o tridimensionale a sua discrezione.

Conclusioni e sviluppi futuri

Il modello migliore per efficacia ed efficienza si è rivelato essere il R*andom Forest.* Tuttavia, ci sarebbero altre vie da percorrere per migliorare le performance, ad esempio testare altri classificatori (come le reti neurali), utilizzare metodi *word embedding*, così come incrementare le canzoni presenti nel training set, in modo tale da allenare maggiormente i classificatori e produrre risultati più soddisfacenti. Ultima, ma non per importanza, sarebbe da approfondire e applicare la stilometria (studio dello stile linguistico), da cui si è preso spunto per la creazione delle feature aggiuntive che di fatti hanno dimostrato di essere di rilevante importanza per aumentare l'accuratezza dei classificatori.