Progetto Intelligenza Artificiale 2018/2019 Sara Picchioni - 6157067
"Categorizzazione di documenti testuali"

Il codice permette di categorizzare i documenti testuali di due data sets (20 newsgroups e Reuters-21578) facendo uso degli algoritmi di Naive Bayes (Bernoulli) e Perceptron di Scikit-learn. Vi sono tre file python: due dedicati alla manipolazione dei data sets e uno utile per creare e visualizzare le learning curves.
Per l'utilizzo del codice è necessario scaricare i data sets, reperibili ai link: http://qwone.com/~jason/20Newsgroups/ (versione da 18828 documenti) e http://www.daviddlewis.com/resources/testcollections/reuters21578/ .
Tali dataset hanno bisogno di essere modificati per poter essere successivamente utilizzati per creare le curve di apprendimento: la classificazione viene applicata sul bag of words di tali documenti. Esso è la rappresentazione in forma vettoriale di interi del documento testuale, nel quale ad ogni indice del vettore corrisponde una parola presente nel documento e l'intero corrispondente della frequenza con cui appare tale parola all'interno del documento stesso. In seguito si spiega come avviene tale trasformazione.

"20news_groups" : inizialmente vi è un metodo che permette di convertire ogni parola del documento in lettere minuscole, per evitare che la differenza di carattere sia interpretata come differenza lessicale.
Il metodo principale del file è chiamato *scan_directory()*, attraverso il quale viene esplorata la directory nella quale si trova il data set per estrapolare il testo dai vari documenti. All'interno di tale metodo vengono usate le librerie **os** e **codecs** per la directory e la lettura dei documenti.
Successivamente si ha il metodo *create_training_data()* nel quale vengono creati i vettori X e y che dovranno essere passati come parametri al metodo per la creazione delle learning curves. Il vettore X presenta il bag of words del dataset (il modo in cui viene ottenuto sarà spiegato in seguito) e y è il vettore in cui sono presenti le etichette a cui i documenti devono essere assegnati.

"LearningCurves": contiene il codice ottenuto dal sito di Scikit-learn https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html . Grazie ad esso è infatti possibile creare e visualizzare i grafici delle curve dei due algoritmi. Sull'asse orizzontale si ha il numero di esempi, mentre sull'asse verticale il valore di precisione.
In seguito il metodo *Bern_Perc()* esegue i due algoritmi al bag of word dei dataset. Viene usato il metodo *ShuffleSplit()*, con parametri impostati a valori di default, per determinare permutazioni casuali della validazione incrociata (cross-validation). Essa è una tecnica statistica che suddivide il campione osservato in gruppi di egual numerosità ed ad ogni passo una parte del data set viene utilizzata come *validation* set, mentre la restante parte rappresenta il *training* set. In questo modo si allena il modello per ognuna delle parti, evitando problemi di overfitting e di campionamento asimmetrico. Tutto ciò al fine di verificare la bontà del modello di predizione utilizzato.
Vengono usate le librerie **matplotlib** e **numpy** per la grafica e **sklearn** per importare gli algoritmi di classificazione e il metodo per la valutazione delle curve.

"reuters": essendo i documenti all'interno di tale data set in formato HTLM sono necessari alcuni metodi per poter lavorare su tali documenti. Il codice di tali metodi è ripreso dal sito https://www.quantstart.com/articles/Supervised-Learning-for-Document-Classification-with-Scikit-Learn.
Vengono estrapolati i nomi delle categorie, con il metodo *get_topics_tags()*, selezionate le 10 categorie più frequenti, con il metodo *get_frequent_topic_list()*, e filtrati i documenti attraverso il metodo *topics_filter()*.
Per l'estrazione del bag of words si procede esattamente come con il precedente data set.
Infine è presente un ciclo for nel quale tutti i documenti vengono salvati, all'interno della lista, i documenti attraverso il parser creato.

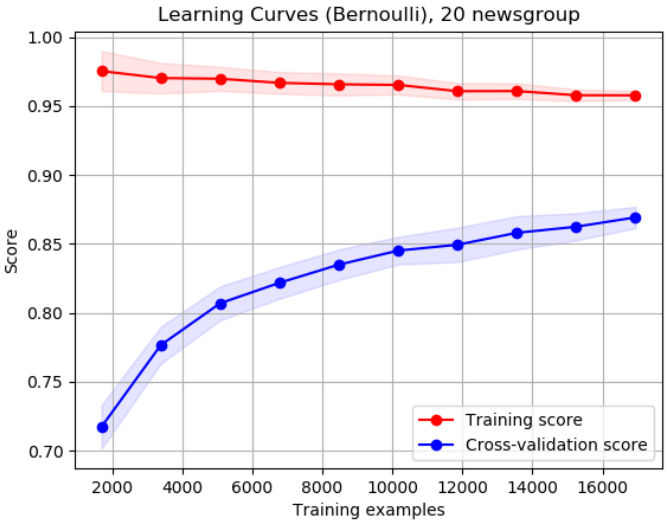
Estrazione bag-of-words
Per ottenere il bag-of-words viene utilizzato il metodo *CountVectorizer()*, presente nella libreria di **sklearn**, che converte una collezione di documenti testuali in una matrice che presenta la frequenza di ogni token presente nei documenti, e il metodo *fit_transform()*, anch'esso presente nella libreria di **sklearn**, che impara il "vocaboli" del dizionario e restituisce la matrice dei termini del documento.

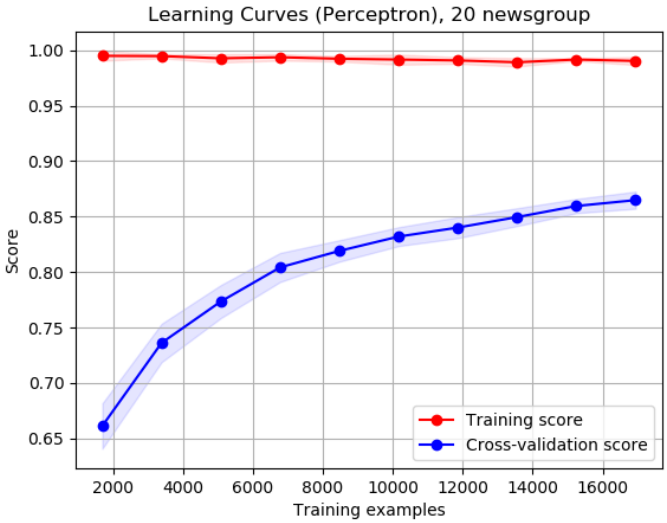
Confronto learning curves

Una curva di apprendimento mostra la relazione tra il punteggio del training set e il punteggio del validation set per uno stimatore con un numero variabile di campioni di allenamento. Questa visualizzazione viene in genere utilizzata per mostrare due cose: quanto lo stimatore beneficia dell'aumento dei dati e se lo stimatore è più sensibile all'errore a causa della varianza rispetto all'errore dovuto al bias.
Se i punteggi di addestramento e di convalidazione incrociata convergono insieme man mano che vengono aggiunti altri dati, il modello probabilmente non trarrà vantaggio da più dati. Se il punteggio di allenamento è molto maggiore del punteggio di convalidazione, allora il modello probabilmente richiede più esempi di addestramento per generalizzare in modo più efficace.
Le curve sono tracciate con i punteggi medi, tuttavia la variabilità durante la convalidazione incrociata viene mostrata con le aree ombreggiate che rappresentano una deviazione standard sopra e sotto la media per tutte le convalide incrociate. Se il modello soffre di errori dovuti a bias, allora ci sarà probabilmente più variabilità attorno alla curva del punteggio di allenamento. Se il modello soffre di errori dovuti alla varianza, allora ci sarà più variabilità attorno al punteggio convalidato.
Abbiamo quindi due punteggi da monitorare: uno per il validation set e uno per il training set. Se tracciamo l'evoluzione dei due punteggi mentre i training set cambiano, finiamo con due curve. Queste sono chiamate curve di apprendimento. In breve, una curva di apprendimento mostra come la precisione cambia al crescere della dimensione del set di allenamento.
Solitamente le curve di apprendimento vengono utilizzate per monitorare l'errore che commette il modello analizzato; per quanto riguarda la classificazione, però, si è più interessati a quanto sia buono il modello, invece che all'errore che compie, quindi si è soliti costruire le curve in base alla precisione che il modello raggiunge durante la generalizzazione. L'errore in questo caso è simmetrico alla precisione e viene calcolato come complemento ad 1 dello Score. Per cui è comunque possibile avere un'idea dell'errore di generalizzazione anche in questo tipo di curve di apprendimento.

A causa dell'utilizzo del metodo *ShuffleSplit()* per dividere i data set in training e test set le curve potrebbero presentare alcune variazioni rispetto alle seguenti.

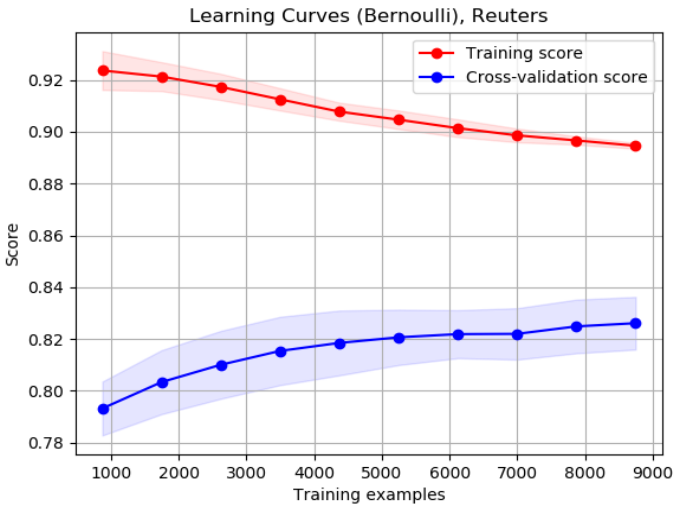
20 newsgroup

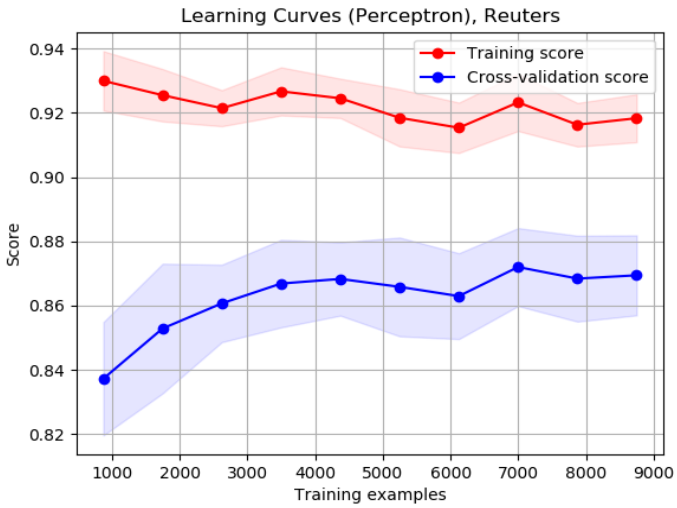




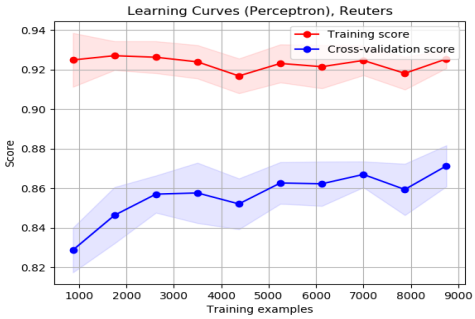
Il bag of word del data set su cui vengono applicati i due algoritmi contiene 18828 documenti
I due grafici sono molto simili: la precisione del training set è molto più alta rispetto a quella del validation set, raggiungendo quasi il massimo valore con l'uso del Perceptron. La precisione dei validation set presenta lo stesso andamento in entrambi, anche se con Bernoulli si raggiunge una precisione maggiore più velocemente rispetto al Perceptron. Da questo si può dedurre che aumentando ancora il numero di documenti questi possano raggiungere una precisione maggiore.
Andando più nel dettaglio, notiamo che inizialmente, con un insieme di 2000 documenti il Bernoulli ha una precisione maggiore. Lo 0,85 viene raggiunto dal Bernoulli con 10000 documenti, mentre il Perceptron solo raggiunti i 13000. Con l'intero data set Bernoulli presenta una precisione leggermente più elevata, superiore allo 0,85.
Si può notare un leggero errore di distorsione nella parte iniziale della curva del punteggio di apprendimento nel Bernoulli, mentre vi è un leggero errore di varianza in entrambe le curve di punteggio del validation set, più accentuata nella parte finale di quella di Bernoulli.
In generale i due algoritmi hanno un comportamento pressoché uguale su questo data set.

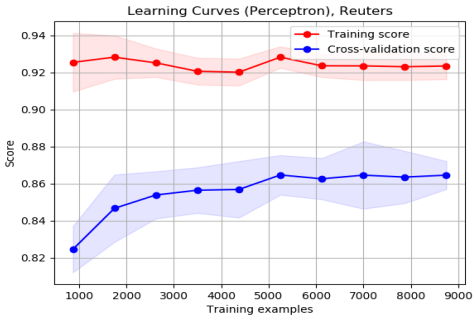
Reuters – 21578

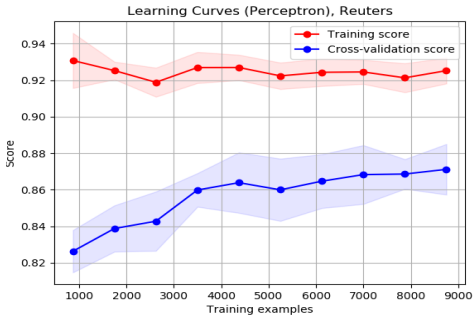




Il bag of word del data set su cui vengono applicati i due algoritmi contiene 9718 documenti.
In questo caso si hanno notevoli differenze tra i due grafici. La curva del punteggio del training set del Bernoulli ha un valore poco maggiore di 0,92 inizialmente e presenta un andamento decrescente all'aumentare del numero di esempi fino a raggiungere un valore minore di 0,90. In questo caso la curva del training set del Perceptron, inizialmente con una precisione di 0,93 in corrispondenza di un set da 1000 documenti, presenta una diminuzione di tale precisione con 6000 documenti, raggiungendo un valore leggermente superiore a 0,91. Con un set di grandezza massima si ha una precisione di 0,92.
Come già specificato, a causa della divisione casuale del data set, si hanno piccole variazioni nei grafici del Perceptron: a seguito di esecuzioni ripetute, alcune delle quali si possono vedere dalle figure sottostanti, si nota sempre che sia la curva del training set sia quella del validation set dei grafici ottenuti presentano un andamento irregolare, soprattutto a partire dalla seconda metà dal grafico.







Le curve del validation set hanno entrambe un andamento crescente: il valore della precisione iniziale è maggiore nel Percpetron e rimane tale anche con l'intero data set. Durante l'aumento però Perceptron presenta molta più variazione della precisione.
La curva del training set del Perceptron presenta errore di distorsione, che nel Bernoulli è poco accentuata. In entrambe le curve del validation set è invece molto evidenziato l'errore di varianza, soprattutto nel Perceptron.

In conclusione possiamo dedurre che il Perceptron ha in generale un modello più accurato applicato ai due data set, ma presenta un apprendimento più irregolare rispetto al Bernoulli.