Learning curves:
Naive Bayes e Random Forest

Niccoló Biondi

12 Febbraio 2018

1 Scopo del progetto

Il progetto si é basato sull'analisi di due algoritmi di classificazione, Naive Bayes, con il modello di Bernoulli, e Random Forest. Sono state esaminate le learning curves dei due classificatori per evidenziare le differenze nell'apprendimento di un database di immagini di articoli di moda, Fashion MNIST.

2 Fashion MNIST

Fashion MNIST é un database di 70000 immagini di articoli di moda, di cui 60000 costituiscono il training set e le restante 10000 sono il test set. Ciascun immagine é un immagine 28X28 in scala dei grigi ed é classificata in una delle 10 classi che costituiscono il dataset.

3 Descrizione del lavoro

mnist_reader.py Questo file viene utilizzato nel progetto per estrarre i file che rappresentano FashionMNIST.

plot_curve.py Questo file implementa la funzione che viene utilizzata per tracciare il grafico della curva di apprendimento. In particolare al suo interno viene richiamato il metodo learning_curve che implementa i metodi per l'apprendimento e per il test del classificatore in questione.

naive_bayes.py In questo file viene caricato il dataset d'immagini e viene implementato l'algoritmo Bernoulli Naive Bayes. In particolare viene utilizzata la funzione BernoulliNB() di sklearn e sono stati impostati come parametri alpha=1.0 che rappresenta il Laplace Smothing, binarize=8.0 che rappresenta la soglia per binarizzare le immagini. All'interno del file viene ultilizzato il metodo ShuffleSplits per impostare la cross validation e lasciare il test set di 10000 immagini. Il risultato finale sará la curva di apprendimento di tale algoritmo.

random_forest.py Questo file viene utilizzato per l'apprendimento del classificatore Random Forest Classifier. In particolare nel metodo RandomForestClassifier() di sklearn sono stati settati come valori per tale funzione n_estimator=30, che rappresenta il numero di alberi di decisione che costituiscono la foresta, criterion='gini' che rappresenta il metodo di split del singolo albero e, in un caso specifico, min_samples_leaf che rappresenta il numero minimo di esempi per foglia. Anche in questo caso é utilizzata la funzione ShuffleSplits per impostare la cross validation e lasciare il test set di 10000 immagini. Il risultato finale sará la curva di apprendimento di tale algoritmo.

Per quanto riguarda l'algoritmo Random Forest sono state analizzate due versioni: la prima in cui ciascun albero di decisione viene costruito fin quando ogni foglia contiene un solo elemento, l'altra in cui la costruzione viene fermata quando ciascuna foglia ne contiene 3. Questa differenziazione é stata fatta per poter confrontare anche l'effetto dell'operazione di pruning sugli alberi di decisione.

4 Conclusioni

Dall'analisi eseguita sull'intero database si ottengo i risultati che si attendevano. Il classificatore Naive Bayes ha un tasso di errore molto elevato sia sul training set sia sul test set. Questo problema deriva dall'ipotesi sbagliata che viene fatta di indipendenza tra tutti gli attributi. Il risultato ottenuto é riscontrabile in Figura 1.

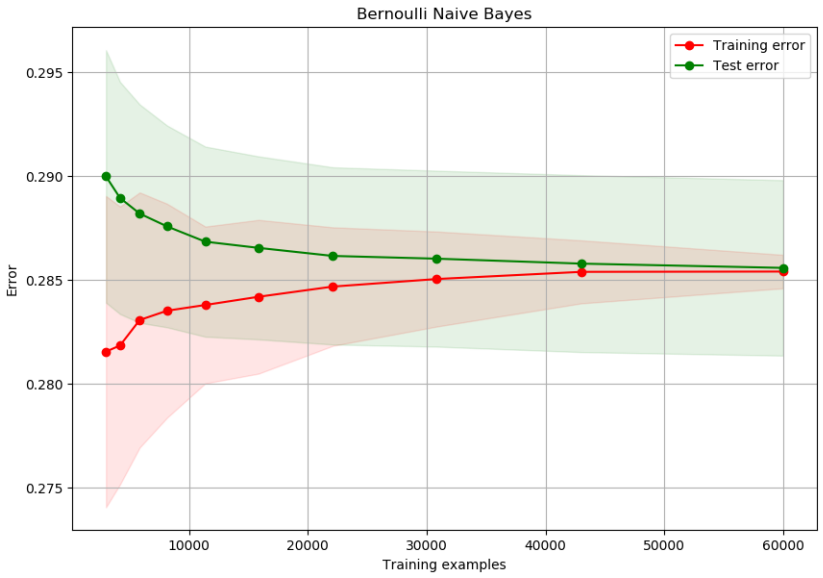

Figura 1: Learning curve di Bernoulli Naive Bayes

Per quanto riguarda il classificatore Random Forest sono stati esaminati due diversi scenari. Il primo riguarda l'apprendimento nel caso di alberi di decisione non potati, quindi un apprendimento piú lento ma che non porta alcun errore sul training set. In questo scenario si nota che non é presente il precedente problema di overfit e si raggiunge un livello molto basso di errore. Un difetto di tale soluzione é, senz'altro, il tempo di addestramento che é necessario per tracciare la learning curve. Il fatto di dover costruire ciascun albero ogni volta fin quando ogni foglia é costituita da soltanto un elemento aumenta notevolmente il tempo d'esecuzione. Questo comporta che per questo tipo di soluzione si é dovuto diminuire i numeri di split di cross validation da 50, utilizzati per Bernoulli Naive Bayes, a 10.

Il risultato é il grafico rappresentato in Figura 2.

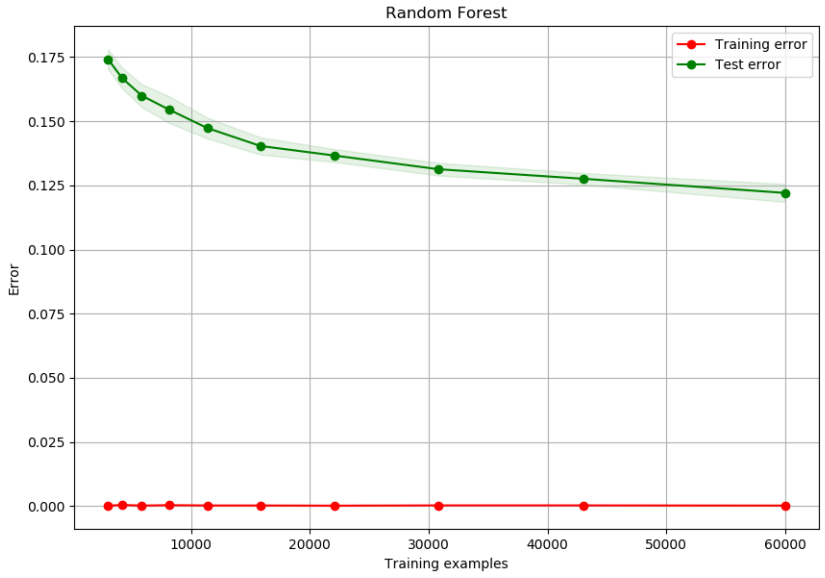

Figura 2: Learning curve di Random Forest senza pruning

L'ultimo grafico d'analisi, visibile in Figura 3 rappresenta il caso in cui gli alberi di decisione hanno subito l'operazione di pruning. In questo caso il tempo di esecuzione del programma diminuisce, ma con esso aumenta l'errore che il Random Forest commette sia sul training set sia che sul test set, anche se questi aumenti sono quasi trascurabili.

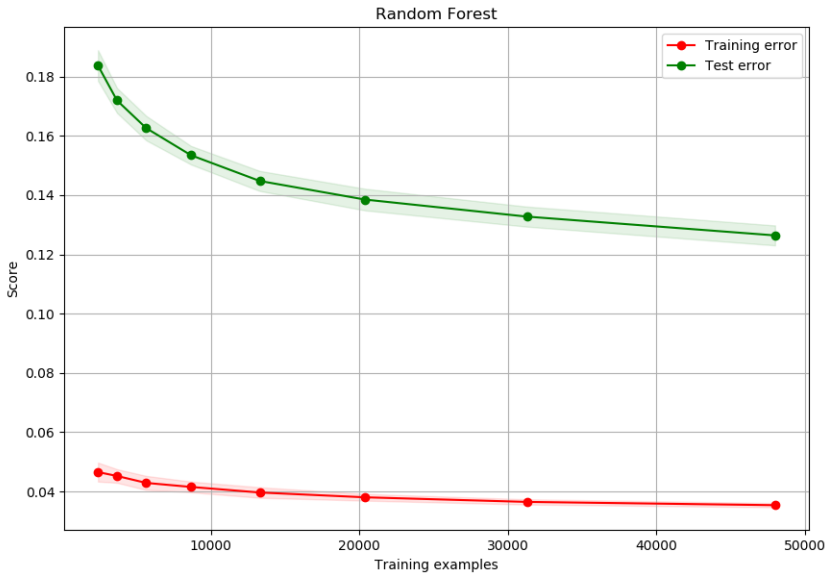

Figura 3: Learning curve di Random Forest con pruning

5 Riproduzione Risultati

Per poter riprodurre i risultati presentati in questo progetto, sará necessario lanciare il file corrispondente al classificatore che si desidera, naive_ bayes.py per Naive Bayes e random_ forest.py per Random Forest. Inoltre é necessario scaricare i file relativi al dataset Fashion MNIST e adattare nei due file dei classificatori sopra citati il path da dove poter estrarre correttamente le immagini. Per quanto riguarda la posssibilitá di poter scegliere tra le due implementazioni di Random Forest, quella con l'operazione di pruning o no, si deve soltanto adattare il commento per eseguire il classificatore che si desidera.