Documentazione

Studente: Palmiotta Alessandro

Matricola: 678185

E-mail: alessandropalmiotta@gmail.com

Link Repository GitHub: https://github.com/P4lmiotta/Machine-Learning.git

1. INTRODUZIONE

Il task di classificazione dei generi floreali, a partire da una determinata immagine (nel caso del progetto, di tipo .JPG), è argomento su cui sono stati svolti numerosi studi in ambito di Machine Learning. Il task consiste nel confronto e nella valutazione di alcuni dei principali modelli di classificazione basati su apprendimento supervisionato. L'Apprendimento Supervisionato è una tecnica di Machine Learning che mira ad istruire un sistema informatico, in modo da consentirgli di elaborare autonomamente previsioni sui valori di output rispetto ad un dato input, sulla base di una serie di esempi ideali costituiti da coppie < Dati, Etichetta >, che vengono inizialmente forniti al modello.

In questo caso, si tratta di classificare 5 tipologie di fiori: Margherita, Dente di Leone, Girasole, Rosa e Tulipano.

I modelli di classificazione confrontati sono stati scelti fra i principali delle seguenti categorie di apprendimento:

- **Probabilistic Classifiers**: Naïve Bayes Classifier;
- **Ensemble Learning Models**: Random Forest Classifier & Extra Trees Classifier.

Al termine di tale confronto, è stato adottato il modello più prestante in termini di accuratezza, precisione e richiamo, il Random Forest Classifier.

2. DATASET

Il dataset contiene 4526 immagini di fiori, raggruppati in 5 cartelle (ognuna dedicata ad ogni tipologia, ovvero Margherita, Dente di Leone, Girasole, Rosa e Tulipano):

- 818 immagini per la tipologia 'Margherita';
- 1054 immagini per la tipologia 'Dente di Leone';
- 828 immagini per la tipologia 'Rosa';
- 804 immagini per la tipologia 'Girasole';
- 1022 immagini per la tipologia 'Tulipano'.

3. FEATURES

Per quanto riguarda le features, sono estratte 4 features per ogni esempio, ovvero:

- Color Histogram, che quantifica il colore del fiore in analisi, per poi essere riportato in un istogramma;
- Hu Moments, che quantifica la forma del fiore in analisi;
- Haralick Texture, che quantifica la texture del fiore in analisi;
- HOG Histogram, che conta le occorrenze dell'orientamento del gradiente in porzioni localizzate di un'immagine in analisi, per poi essere descritte in un istogramma.

Inoltre, viene effettuato un 'flip' orizzontale e verticale per ogni esempio, per poi calcolarne l'HOG, in modo da avere un miglioramento nel livello d'informazione della feature HOG Histogram.

Una volta estratte le features, quest'ultime verranno immagazzinate in file .h5 (grazie alla libreria h5py), ovvero 'data.h5' e 'labels.h5', presenti nella directory 'datasets_file'.

3.1. Normalizzazione delle features

Le features sono normalizzate tramite la classe MinMaxScaler, presente nella libreria sklearn, con un range di valori che va da 0 a 1.

4. MODELLO DI CLASSIFICAZIONE UTILIZZATI

Innanzitutto, il set di features è suddiviso in modo da avere il 75% del set dedicato al training set, il 25% del set dedicato al test set (tramite la classe 'train_test_split' della libreria sklearn).

Al fine di ottenere una predizione sui nuovi esempi, sono stati applicati modelli di classificazione basati su apprendimento supervisionato, derivati dalla libreria sklearn. L'idea di utilizzare più modelli ha avuto lo scopo di valutare l'accuratezza di ogni singolo modello in fase di test.

• **GaussianNB**: I classificatori basati sul modello Naïve Bayes, utilizzano il teorema di Bayes:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)}$$

dove P(y|x1,…,xn) è la probabilità a posteriori, P(y) è la probabilità a priori, P(x1,…,xn|y) è la verosimiglianza e P(x1,…,xn) è la funzione di partizione. Nell' utilizzo del classificatore GaussianNB, si presume che la probabilità delle feature sia gaussiana:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

i parametri σy e μy sono stimati usando la massima probabilità.

• **Random Forest**: È un modello d'insieme ottenuto dall'aggregazione tramite bagging di alberi di decisione. Esso è un meta-stimatore che si adatta ad una serie di alberi decisionali addestrati su

vari sotto-campioni del dataset e utilizza la media di ogni singolo output di ogni albero per migliorare l'accuratezza predittiva e il controllo dell'overfitting. Il Random Forest deve essere dotato di due matrici: una matrice X sparsa che contiene i campioni di addestramento e una matrice Y di dimensioni che contiene i valori target.

• **Extra Trees**: Tale modello è simile al precedente, la differenza risiede nella scelta degli alberi, la quale avviene in maniera puramente casuale.

4.1. Ottimizzazione degli iperparametri:

Al fine di rendere notevolmente alta l'accuratezza di ciascun classificatore utilizzato è stato seguito un procedimento di ottimizzazione degli iperparametri (in questo caso, per i modelli Random Forest ed Extra Trees).

Partiamo dal presupposto che ciascun modello di classificazione prevede la presenza di parametri opportunamente passati in fase di costruzione di un determinato modello, dunque ciascun valore associato a questi ultimi prendi il nome di iperparametro. Se non esplicitati, ai parametri verranno associati valori di default, che molto spesso non permettono al modello di esaltare la sua massima accuratezza. Dunque, qualsiasi parametro di un qualsiasi classificatore, può essere opportunamente settato sulla base di diversi approcci di ottimizzazione. La tecnica di ottimizzazione utilizzata in tale progetto è stata:

• **Exhaustive grid search**: tale metodo, fornito da GridSearchCV, genera in maniera esaustiva i possibili candidati (iperparametri) attraverso una griglia di valori specificata opportunamente dal parametro "param_grid", caratterizzato da un range di valori per ogni singolo parametro specificato. In maniera del tutto automatica, vengono valutate tutte le possibili combinazioni di assegnazioni degli iperparametri e viene mantenuta la combinazione migliore. Per quanto riguarda il parametro "cv", si è scelta la classe StratifiedKFold, ovvero una variazione del k-fold che restituisce fold stratificati (ogni set contiene approssimativamente la stessa percentuale di campioni di ciascuna classe target del set completo), con 'n_splits' pari a 5.

Al termine di tale processo, verranno mostrati quelli che sono gli iperparametri migliori per un determinato modello di classificazione.

5. VALUTAZIONE

Di seguito vengono riportati tutti i valori delle metriche utilizzate al fine di valutare la "bontà" di ciascuno dei classificatori:

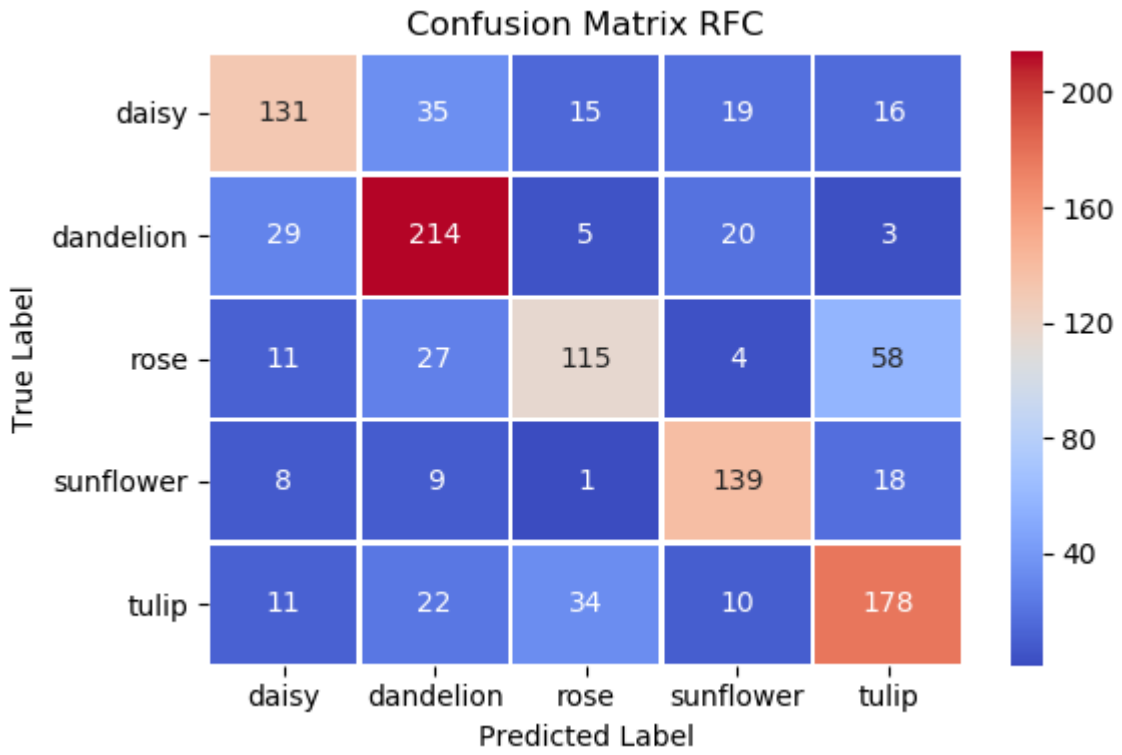
CLASSIFICATORI	Accuratezza	Precisione	Recall	F1-measure
GaussianNB	0.442	0.444	0.426	0.414
Extra Trees	0.609	0.642	0.599	0.595
Random Forest	0.689	0.691	0.687	0.685

6. CONCLUSIONE

Dopo aver confrontato i diversi classificatori, si è arrivati alla conclusione di utilizzare il RandomForestClassifier. Tale scelta si è basata sulle metriche sopra riportate. Esso, visto il valore più alto di accuratezza, riesce a predire meglio quale sarà la tipologia di un determinato fiore.

Di seguito vengono riportate:

- **Confusion matrix Random Forest Classifier:**

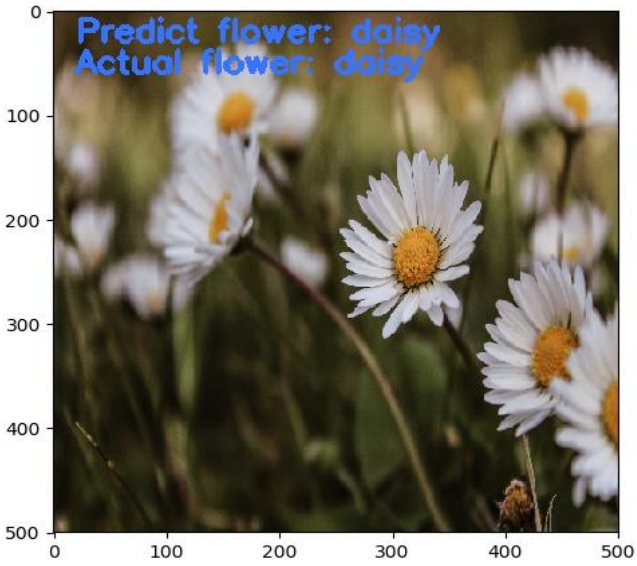


- **Metriche utilizzate Random Forest Classifier:**

TIPOLOGIA	Precision	Recall	F1-score	Support
Margherita	0.69	0.61	0.65	216
Dente di Leone	0.71	0.80	0.75	271
Rosa	0.67	0.52	0.59	215
Girasole	0.74	0.80	0.77	175
Tulipano	0.65	0.71	0.68	255

6.1. Esempi di Predizione

Infine, tramite la libreria matplotlib, verranno visualizzate predizioni su 25 immagini presenti nella directory del progetto 'test_images'. Ecco un esempio:



N.B. : Per ‘Actual Flower’ s’intende la tipologia del fiore, mentre per ‘Predict Flower’ s’intende la tipologia predetta del fiore predetta dal classificatore utilizzato (Random Forest Classifier).