Bee Classification Implementation

Assignment 4
Computer VIsion

Submitted by:
Shristi Maskay

Abstract:

Using a dataset of various bees to identify a bee as a honey bee or a bumble bee species using its appearance. So, for classification we have used 3 models LinearSVC, KNN, Random Forest and by using gridsearch on various models and based on F1 score KNN and Random Forest was found to give better results in prediction.

1. Load Data

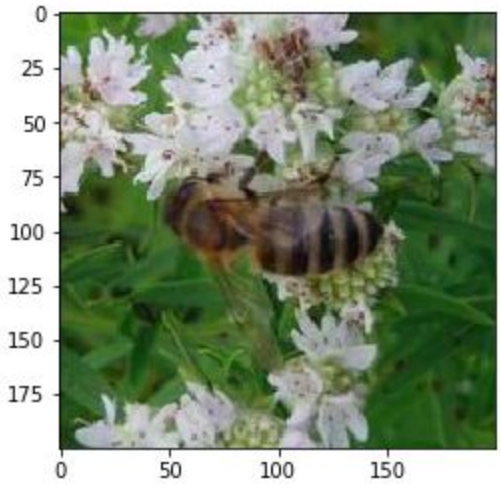
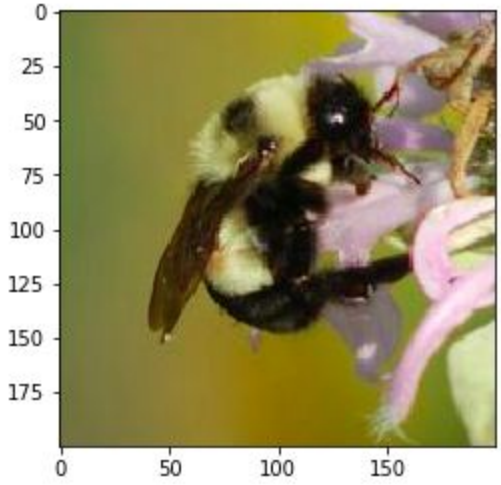
We will load our labels.csv file into a dataframe called labels which consists on ID and Genus of Bees, where the index is the image name (e.g. an index of 550 refers to an image named 550.jpg) and the genus column tells us the bee type.

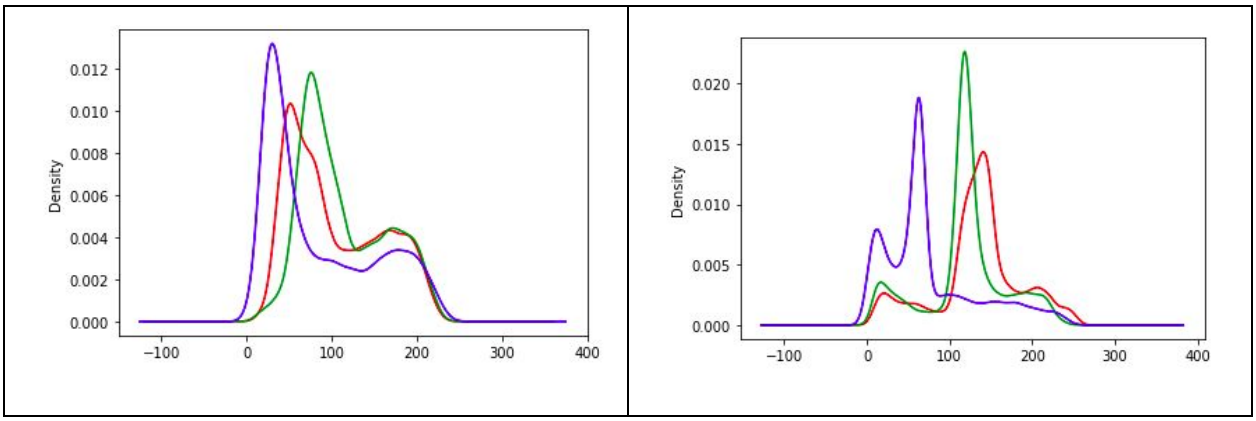
Genus = 1.0 (Bombus or bumble bee)

Genus = 0.0 (Apis or honey bee)

The function get_image helps to convert an index value from the dataframe into a file path where the image is located and returns the image as a numpy array.

2. Display Images of Each Bee

Honey Bee	Humble Bee
	

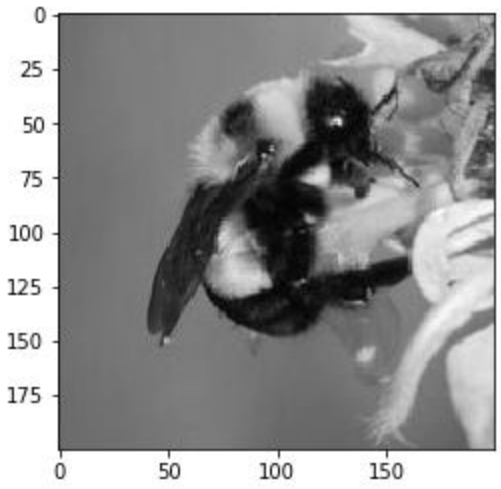


Most image formats have three color **"channels": red, green, and blue**. For each pixel in an image, there is a value for every channel.The way this is represented as data is as a three-dimensional matrix. The width of the matrix is the width of the image, the height of the matrix is the height of the image, and the depth of the matrix is the number of channels. So, as we saw, the height and width of our image are both 100 pixels. This means that the underlying data is a matrix with the dimensions 100x100x3. So using the Histogram of colors to get the image of bees but this might not be too accurate as bees are almost in same colors and are mostly found on top of colorful flowers so the color of flowers might be distracting for this scenario

3. Image manipulation rgb2grey

In most cases color information is quite useful in classification but in the case of bees, bees themselves are very similar in colors and are often on top of colourful flowers. So in this case the colour of the flowers may be distracting from separating honey bees from bumble bees, so it is required to convert the images to **black-and-white, or "grayscale."**

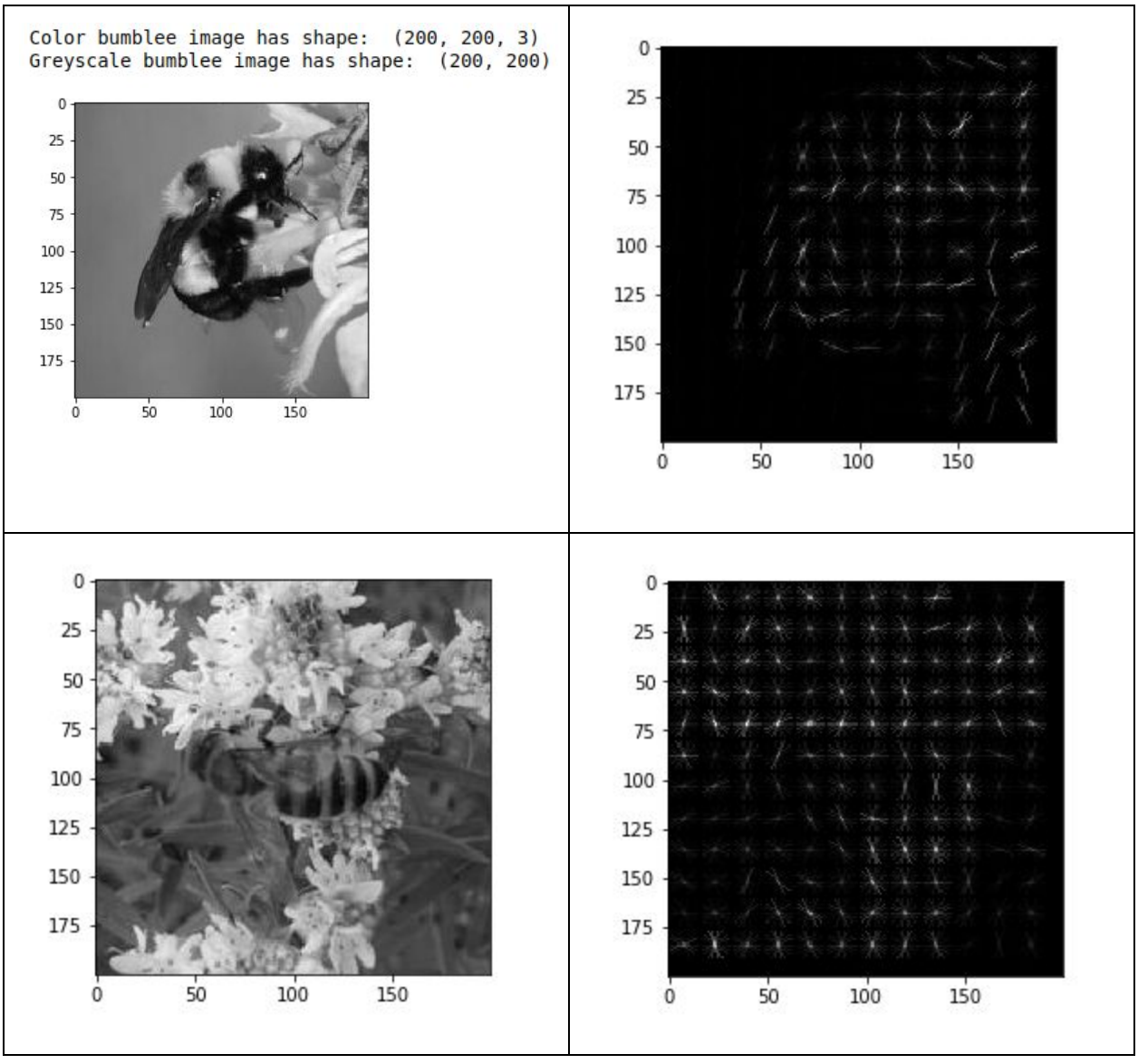
```
Color bumblee image has shape:  (200, 200, 3)
Greyscale bumblee image has shape:  (200, 200)
```



4. Histogram of oriented gradients feature extraction and image preprocessing

Library: from skimage.feature import hog
from skimage.color import rgb2grey

HOG is conversion of image to machine readable form. Like edge detection of an object, get the required object and remove unwanted colors from the image.The main concept of HOG is to get the shape of detecting objects from within an image using its edges using the intensity of gradients.



Here data is converted into a form where row = image and columns = features representing all the information for a given image in a single rowAlgorithms require data to be in a format where rows correspond to images and columns converting 3D to 1D array

Dimensionality of image is (400000,) and that of hog_features is(8100,)

All the images in the dataset needs to be preprocessed i.e converted to greyscale and converting 3D to 1D array.Created features for each image and append in the feature list and pass it to the model

5. Scale feature matrix

Library: from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

Features in a dataset might be in different scales so it needs to be transformed in such a manner that it has mean 0 and **standard** deviation 1. It is used in cases where data has negative values so StandardScale arranges the data in a **standard** normal distribution. Currently our dataset has 8100 features for 3969 images so PCA is used to reduce the number of features to 500 such that more information is contained within less number of features k/a components

Transforms the data in such a manner that it has mean as 0 and **standard** deviation as 1. In short, it standardizes the data. Standardization is useful for data which has negative values. It arranges the data in a **standard** normal distribution.

```
Feature matrix shape is:  (3969, 8100)
PCA matrix shape is:  (3969, 500)
```

6. Split into train and test sets

Library: Using sklearn's model selection library to split the dataset to training and testubg sets. In this case we are using 25% testset and 75 percent Trainset with randomstate: 27
from sklearn.model_selection import train_test_split

7. Train Model

i. Linear SVC

SVM are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. It can also be used for regression problems. We will look at the power of SVMs for classification.

Parameters used:

Param_grid: = {'C': [2,10,20,30,50,100,1000]}

Best Parameter :

0.6579375105746996 {'C': 100}

ii. KNN

K-Nearest Neighbours (k-NN) is a supervised machine learning algorithm i.e. it learns from a labelled training set by taking in the training data X along with it's labels y and learns to map the input X to it's desired output y. In this case have used parameters:

Parameters used:

```
parameters = [{
      'n_neighbors': [1, 3, 5, 10, 50, 100],
      'weights': ['uniform', 'distance'],
      'metric': ['euclidean', 'manhattan']
   }]
n_folds = 10
```

Best parameters:

0.884507224613602 {'metric': 'euclidean', 'n_neighbors': 100, 'weights': 'distance'}

iii. Random Forest

Random forests algorithms are used for classification and regression. The random forest is an ensemble learning method, composed of multiple decision trees. By averaging out the impact of several decision trees, random forests tend to improve prediction.In this case:

Parameters used:

```
hyper_param = {'max_depth': (5, 10),
               'min_samples_split': (2, 5, 10, 15, 100),
               'n_estimators':(50, 100)}
```

Best Parameters:

0.8842183489060248 {'max_depth': 10, 'min_samples_split': 2, 'n_estimators': 50}

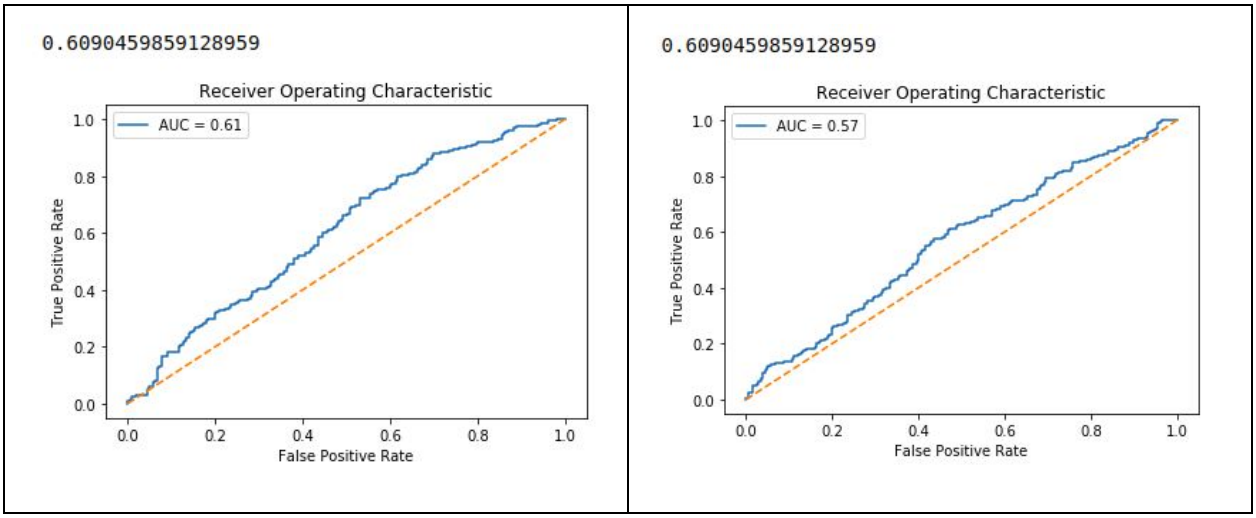
10. Score model (model comparison from Grid Search)

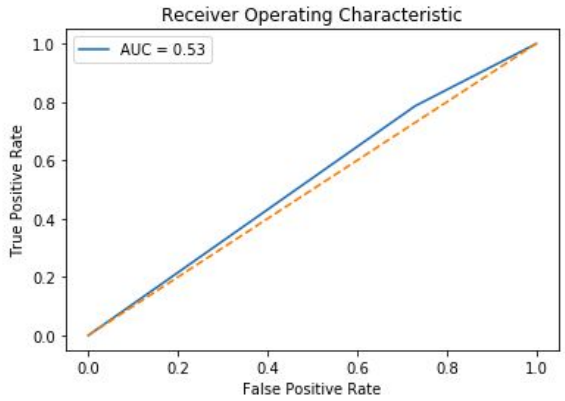
Model	Best parameter	F1 Score
LinearSVC	{'C': 30}	663306
KNN	{'n_neighbors': 100, 'weights': 'distance', ‘metric’:’euclidean;}	0.88434
Random Forest	{'max_depth': 10, 'min_samples_split': 10, 'n_estimators': 50}	0.88405

From the three models used and based on the F1 score used it is found that the KNN and Random forest accuracy is quite better than that on LinearSVC model around 88%
On prediction on test set the accuracy of the model obtained is :
Testset accuracy obtained:
LinearSVC = 68%
KNN = 79%
Random Forest = 0.79%
Accurace with KNN and Random Forest predicting the same output

11. ROC curve + AUC

ROC curve gives the trade-off between Sensitivity k/a TPR and Specificity (1-FPR). Classifier that gives curve closer to the top left corner indicates a better performance



0.5283954714754582  	Considering the above 3 ROC curves the 1st is obtained using KNN best fit model followed by Random forest and lastly Linear SVC. The first graph is more to the top left than 2nd one so KNN classification is found to be better abong the models used for bee classification.

CODE:
https://github.com/Kristiee/Bee_detection