**Image Pre-Processing**

In general, we prefer to use Deep learning and Convolution Neural Network for image data. In CNN we use data augmentation to pre-process the data. But as our data is very sensitive, improper use of data augmentation may lead to predicting a healthy person’s drawing look like Parkinson’s patient(drawing).

Therefore, applying computer vision to this problem is an appropriate choice.

### Importing The Necessary Libraries

The first step is usually importing the libraries that will be needed in the program.

The required libraries to be imported to  Python script are:

**OpenCV:**

[OpenCV](https://en.wikipedia.org/wiki/OpenCV) is a library of programming functions mainly aimed at real-time computer vision. Here, OpenCV is used for resizing. Rescaling, thresholding the image.

**Imutils:**

Imutils package has a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, and displaying Matplotlib images and video frames easier with OpenCV.

We will build\_montages  for visualization. Our paths import will help us to extract the file paths to each of the images in our dataset.

**sklearn.metrics**:

The module implements several loss, score, and utility functions to measure classification performance.

**sklearn.preprocessing**:

This package provides several common utility functions and transformer classes to change raw feature vectors into a representation that is more suitable for the downstream estimators.

**Sklearn.ensemble**

This package contains RandomForestClassifier and many more inbuilt algorithms.

**Scikit-image**

Scikit-image, or skimage,  is an open-source Python package designed for image preprocessing.

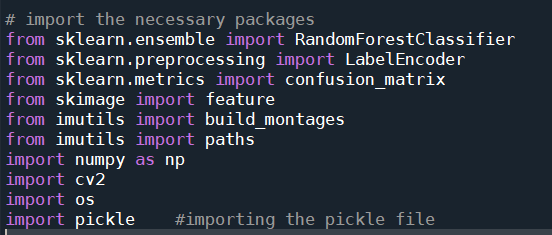
Histogram of Oriented Gradients (HOG) will come from the feature import of scikit-image.

**Pickle:**

Python pickle module is used for serializing and de-serializing python object structures. The process to converts any kind of python objects (list, dict, etc.) into byte streams.

Now we will open spyder from the start menu.

All the above modules can be imported into our program using the below code



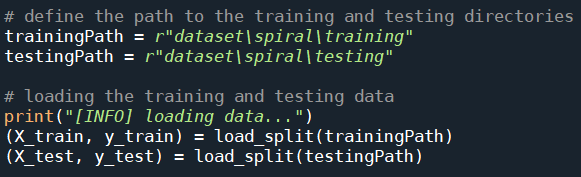
Once we have imported the packages we start loading the dataset.

**Loading Train Data And Test Data**

After importing the necessary libraries, the next step is to define the training path and testing path.

Our dataset contains both hand-drawn spiral and wave patterns. Here we are taking spiral patterns into consideration and training the model.

We split the data into train and test. Using the training dataset we train the model and the testing dataset is used to predict the results.

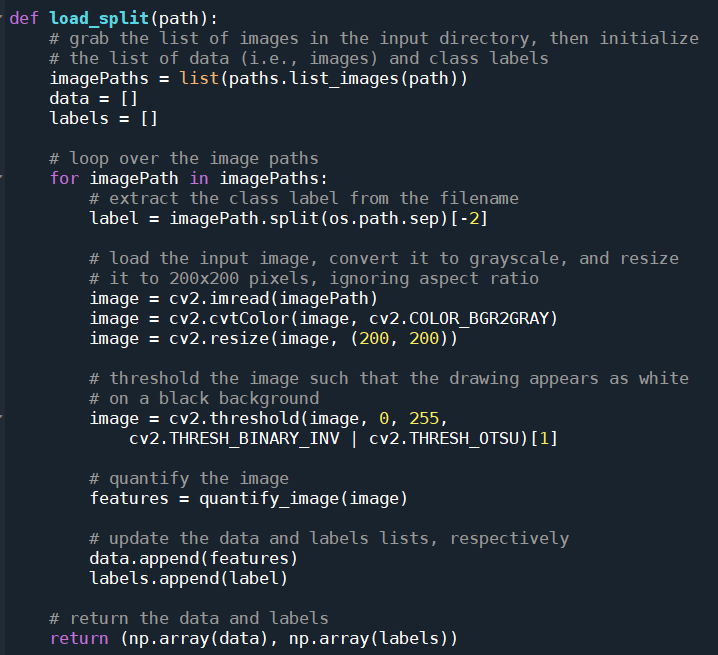


**Note:** Can try the same with wave pattern dataset

**load\_split()**

The load\_split  function accepts a dataset path and returns all feature data and associated class labels.

* From there we grab input image paths by making use of imutils .
* Initialising data and labels lists.
* loop over all image paths which we have grabbed in the previous step.
* Each label is extracted from the os.path.split() method in Python which is used to split the pathname into a pair head and tail. Here, the tail is the last pathname component and the head is everything leading up to that.
* Each image is loaded and preprocessed.
* Read the input image
  + **Imread()** a pre-defined method is used to read the input image
* **Convert image to grayscale**
  + cv2.cvtColor(frame, flag): is used for color conversion.
  + cv2.COLOR\_BGR2GRAY: The flag is used to convert the coloured image to grayscale.
  + we convert the image to grayscale to reduce the processing time
* **Resizing the image**
  + resize it to 200x200 pixels,
* **Threshold image**
  + The thresholding step segments the drawing from the input image, making the drawing appear as a white foreground on a black background.
  + Please refer to this [link](https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_thresholding/py_thresholding.html)for more information on image thresholding concept.
  + Features are extracted via our quantify\_image  function.
  + The features and label are appended to the data and labels lists respectively.
  + Finally, data and labels are converted to NumPy arrays and returned conveniently in a tuple.



**Quantifying Images**

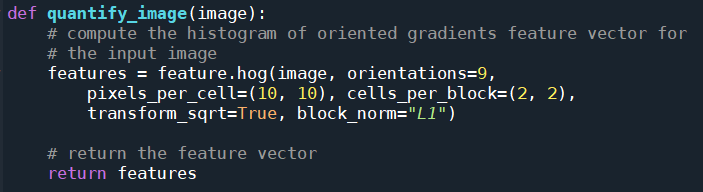
We will extract features from each input image with the quantify\_image  function.

HOG is a structural descriptor that will capture and quantify changes in local gradient in the input image. HOG will naturally be able to quantify how the directions of a both spirals and waves change.

It will be able to capture if these drawings have more of a “shake” to them, as we expect from a Parkinson’s patient.

The most important parameters for the HOG descriptor are the **orientations**,  **pixels\_per\_cell,** and the  **cells\_per\_block**. These three parameters (along with the size of the input image) effectively control the dimensionality of the resulting feature vector.

The resulting features are a 12,996-dim feature vector (list of numbers) quantifying the wave or spiral. We’ll train a Random Forest classifier on top of the features from all images in the dataset.

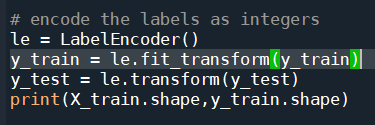


### Label Encoding

**Label Encoding** is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction. We apply  Encoding in order to convert the values into 0’s and 1’s.

As we have observed that we have labels in our dataset. We need to convert them into binary values by using Label encoding. 0:healthy,1:Parkinson

Create an object **le** and fit the **y\_train, y\_test**using **fit\_transform.**





Detailing 2 important approaches in Machine Learning to convert catego..

Detailing 2 important approaches in Machine Learning to convert categorical data into numerical data..

<https://towardsdatascience.com/categorical-encoding-using-label-encoding-and-one-hot-encoder-911ef77fb5bd>

### Model Building

There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms can be chosen according to the objective. As the dataset which we are using is a classification so you can use the following algorithms

* Random Forest Classification
* Decision Tree Classification

You will need to train the datasets to run smoothly and see an incremental improvement in the prediction rate.

**Training The Model**

* Once after splitting the data into train and test, the data should be fed to an algorithm to build a model.
* There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms that you can choose according to the objective that you might have it may be Classification algorithms are Regression algorithms.

1.Logistic Regression

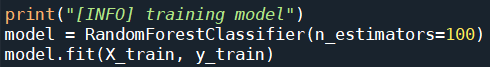
2.Decision Tree Classifier

3.Random Forest Classifier

4.KNN

**Random Forest classifier**

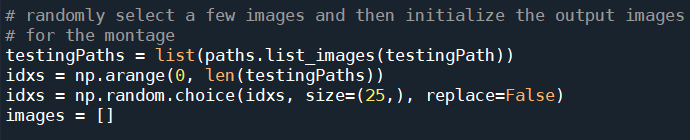
Initialize our **Random Forest classifier** and train the model using a number of estimators as 100



**Testing The Model**

After training the model, the model should be tested by using the test data which is been separated while splitting the data for checking the functionality of the model.

Here we are selecting 25 images from the test data and initialize the output images for montage



we’re going to create a montage so that we can share our work visually

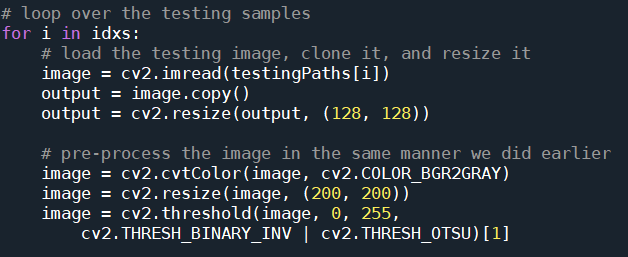
* First, we randomly sample images from our testing set

* Our images  list will hold each spiral image along with annotations added via OpenCV drawing functions .

* We proceed to loop over the random image indices.

* Inside the loop, each image is processed in the same manner as during training(convert to gray scale, resize, threshold) .

* From there we’ll automatically classify the image using our new HOG + Random Forest based classifier and add color-coded annotations

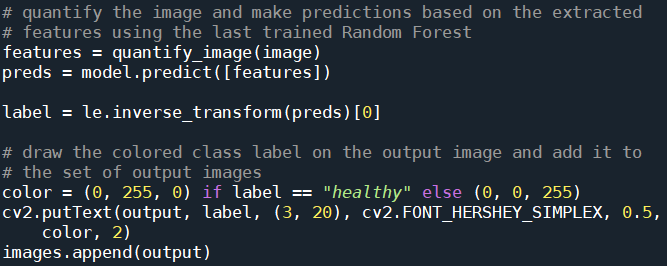


* Each image is quantified with HOG features.

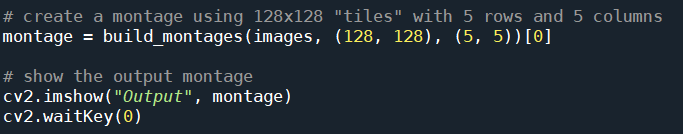
* Then the image is classified bypassing those features to model.predict .

* The class label is colored <strong>green</strong> for “healthy”  and <strong>red</strong> otherwise .The label  is drawn in the top left corner of the image using cv2.putText function .

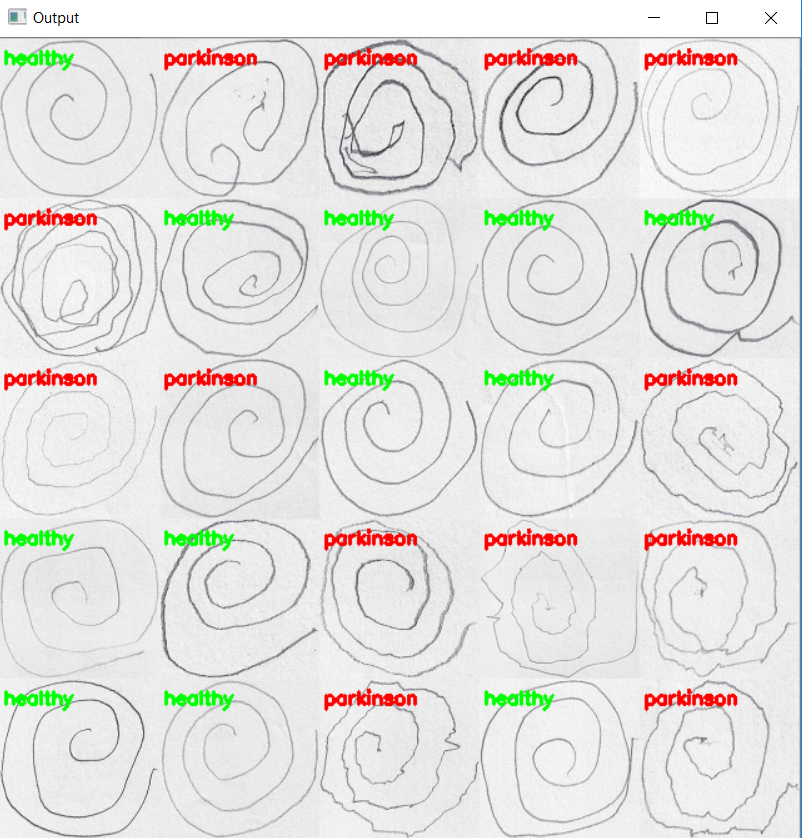
* Each output  image is then appended to an images  list so that we can develop a montage



* The montage is then displayed until a key is pressed



The **cv2.imshow()**function always takes two more functions to load and close the image. **cv2.waitKey()** function, you can provide any value to close the image and continue with further lines of code.



From the above montage, we can clearly differentiate between the geometric patterns drawn by healthy and Parkinson patient.

**Model Evaluation**

Evaluation is a process during the development of the model to check whether the model is the best fit for the given problem and corresponding data.

**Classification Evaluation Metrics:**

These model evaluation techniques are used to find out the accuracy of models built in the classification type of machine learning models. We have three types of evaluation methods.

* Accuracy\_score
* Confusion matrix
* Roc- Auc Curve

**Confusion Matrix**

It is a matrix representation of the results of any binary testing.

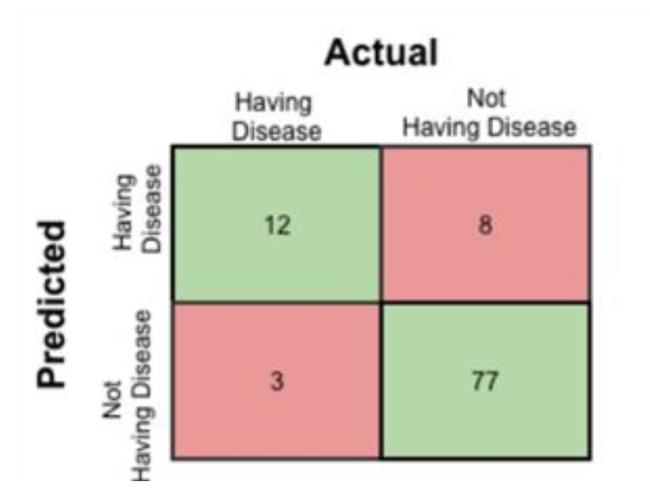
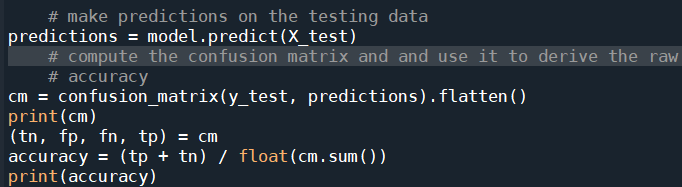


       Fig: Confusion Matrix of prediction of a disease

1. True Positive: 12 (You have predicted the positive case correctly!)
2. True Negative: 77 (You have predicted negative case correctly!)
3. False Positive: 8 (You have predicted these people as having disease, but in actual they do not have.)
4. False Negative: 3 (Wrong predictions )



We can use the **predict** method on the model and pass **X\_test** as a parameter to get the output as predictions.

The output of the confusion matrix and accuracy are as follows.



### Save The Model

After building the model we have to save the model.

Pickle in Python is primarily used in serializing and de-serializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions or transport data over the network. wb indicates write method and rd indicates the read method.

This is done by the below code

