SURF ALGORITHM

S.U.R.F or Speeded Up Robust Features is a patented algorithm used mostly in computer vision tasks and tied to object detection purposes. The algorithm works as follow. Find features/key points that are likely to be found in different images of the same object

We will save the features of the Data in our local database and then we will the distance between the input image feature and the features of the data

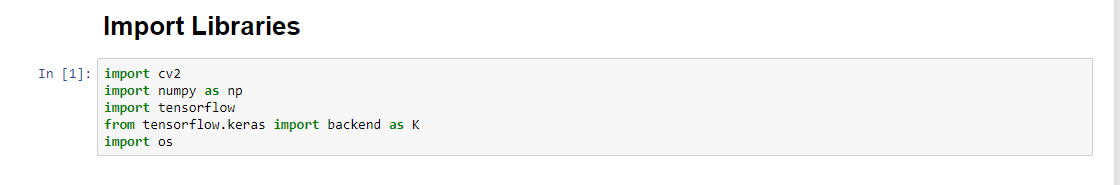
If the distance between the features Is low, then they will have most similar features

This technique is used for the unsupervised learning or for the semi-supervised learning. With this technique the data will be labeled by itself

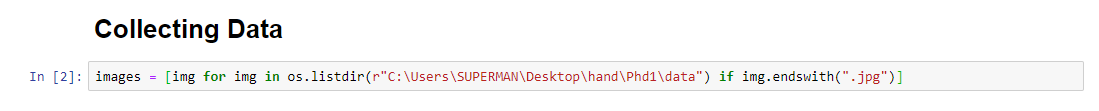
NumPy

Numpy is a numerical python library which deals with High dimension data, like more features, images, etc… This is very fastest python library for the N-D array

**Code Snippets**



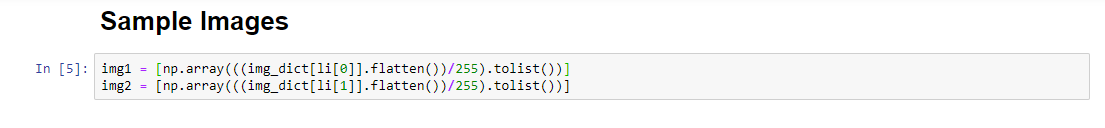
Cv2 is a library which help to read the images and to write or save the images. Here we are checking the image presence in the path of the list



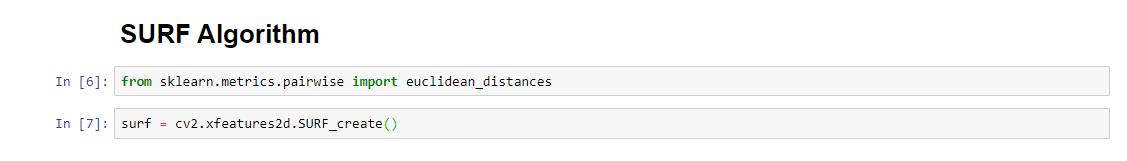
images is a list which is saving the path of the all the images, with the help of the os.listdir() which says the path and if statement checks it is image or not and every path will save in the list



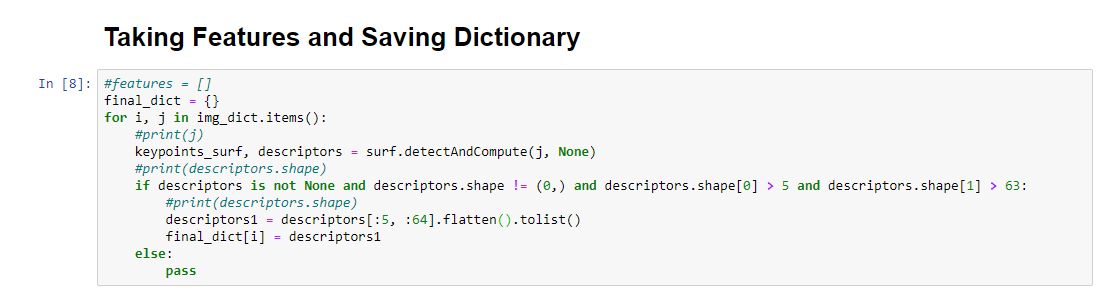
Here we are taking the data and saving into the dictionary with image and values as the NumPy array, and printing the first key and size if the image in the dictionary



Taking the image for the sample test and here we are flattening the image and converting into the list



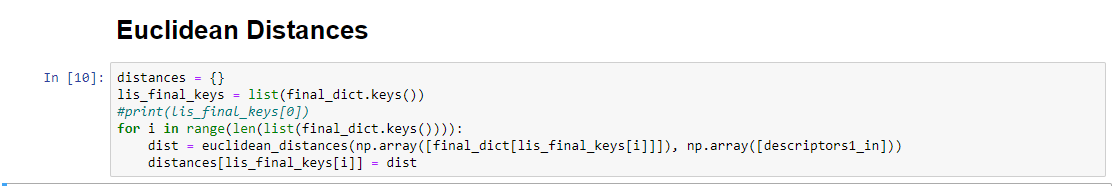
Importing Euclidean distance from Sci-Kit Learn and we are creating object the SURF With the help of the OpenCV. This is for the 2D images we will collect the features from this



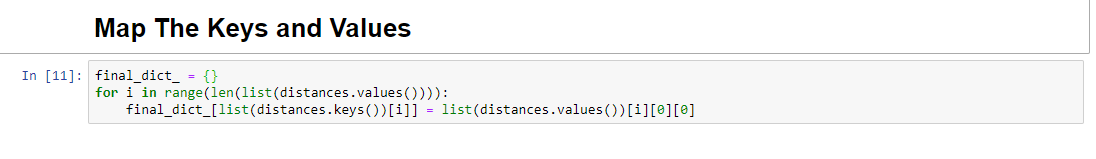
Sending images into the SURF Algorithm and then it will create key Points and Descriptors , then we will send in to the if statement where descriptors shouldn’t be None, shape shouldn’t be equal to (0, ), descriptors.shape[0] means row should not be great than 5, columns should be greater 63, then flat the descriptor and convert into the list save into the dictionary



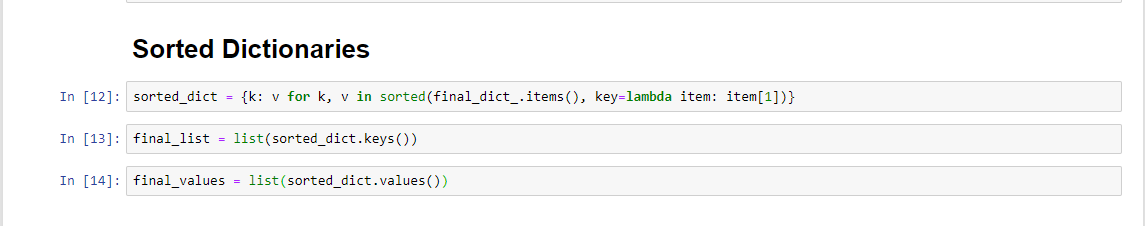
We are taking the input and reshaping it into the 48, 48 and will compute the features and key points, will send into the and then flat it and convert to list



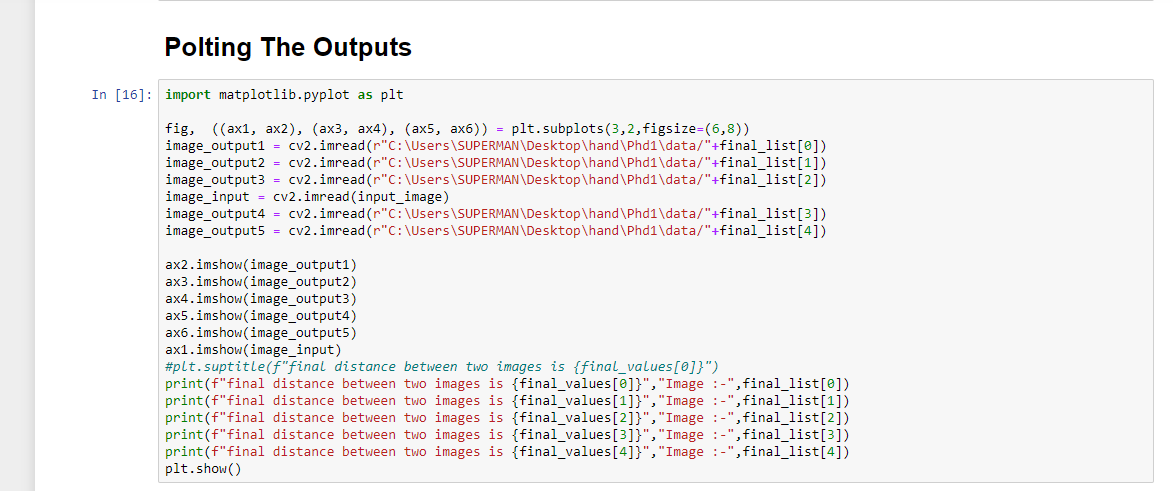
Finding the distances of the data base images and the input image we convert all the list images into the NumPy array and the will find the distances between the images



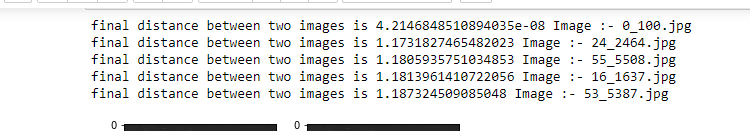
Mapping the all the keys and the values in the distances and appending into the final dictionaries



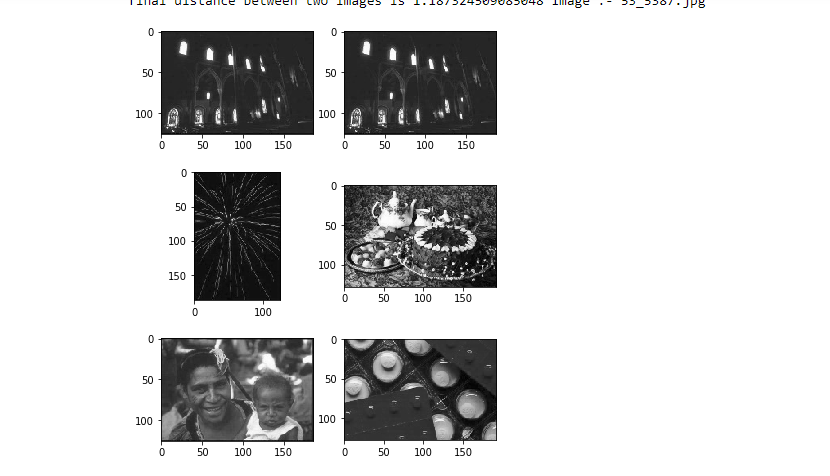
Here sorting the all the images according their distances in the final dict, it will help us to find witch image is nearer to the another image



Here we will plot the inputs and output images in the sorted dict we will first five images in the list and the one input ax1 contains input remaining all are out puts



Distance Between first five images and the input image features



All the Output images first one is input and next are output images