Raw Data to Feature Space

Assignment-1

I. INTRODUCTION

The aim of this assignment-01 is to extract features space from the given data (raw data). In this document we extract feature spaces by following particular tasks one by one. It Includes set-up of environment to extract features from a given raw data set.

II. SETTING UP ENVIRONMENT

A. Programming Environment

Programming language used in this assignment is Python, because it is easy to understand and has many great advantages over other programming languages. Even new developers will find Python code to be compact and legible, which is advantageous for machine and deep learning projects. There are numerous libraries and frameworks available: Many libraries and frameworks are included with the Python language, making development simple. This also helps you save a lot of time. Python is an open-source programming language that has a lot of resources and good documentation all over the world [1]. Anaconda environment is selected to implement tasks of this assignment, which has Spyder IDE for code development. Anaconda is a free opensource environment well build environment to develop projects. We can get access to over 7,000+ open-source packages, we can be install them with “conda install ’packagename’” command [2].

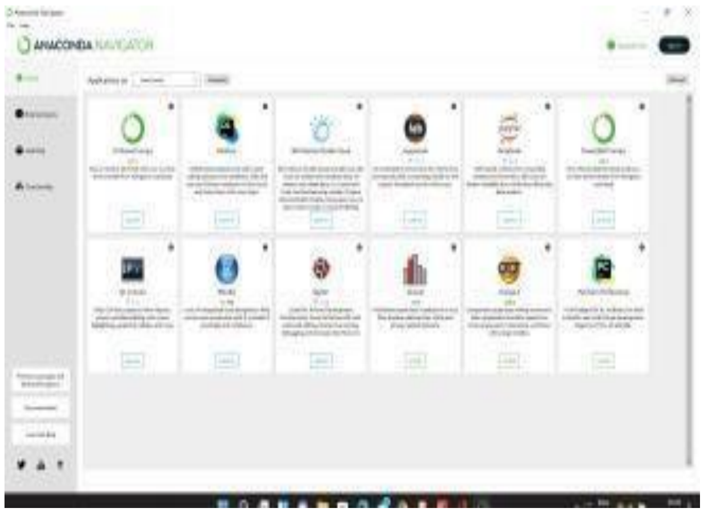
B. System Specifications

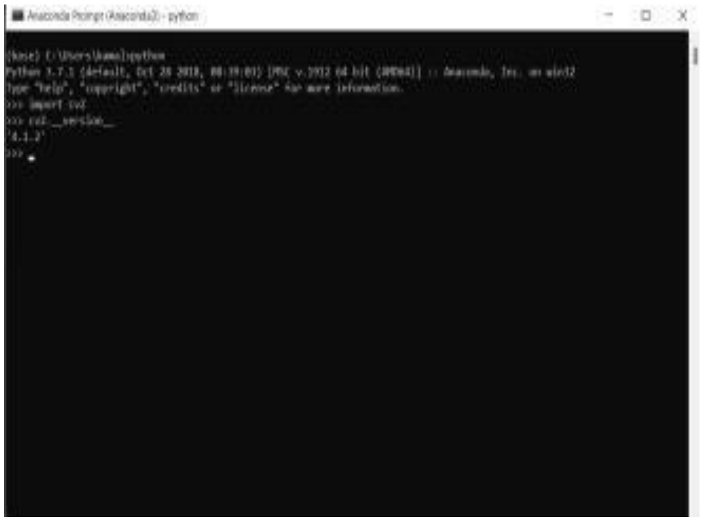
- The operating system used for this assignment is Windows11.
- System type is Windows- 64-bit operating system, x64 basedprocessor.
- 8 GB of installed RAM

C. Installations

Anaconda environment was downloaded from the official Anaconda website[3] and for proper installation steps for windows OS and support for troubleshooting [4]. Spyder IDE (Fig.2.) is integrated with the Anaconda environment which can be accessed from Anaconda by running “spyder” command, Computer vision tool used in this assignment is OpenCV, which was downloaded and installed by running “conda install -c conda-forge opencv” or check If library is already installed by checking version, open a anaconda prompt, then execute command ”python” then type these

two commands ”import cv2 and cv2. version ”. ”openCV version is shown in Fig.1 Right.





```
(base) C:\Users\kamal>python
Python 3.7.1 (default, Oct 28 2018, 08:39:03) [MSC v.1912 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'4.1.2'
>>>
```

Fig. 1. Left: Anaconda Navigator. Right: Spyder IDE

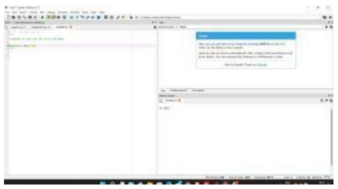
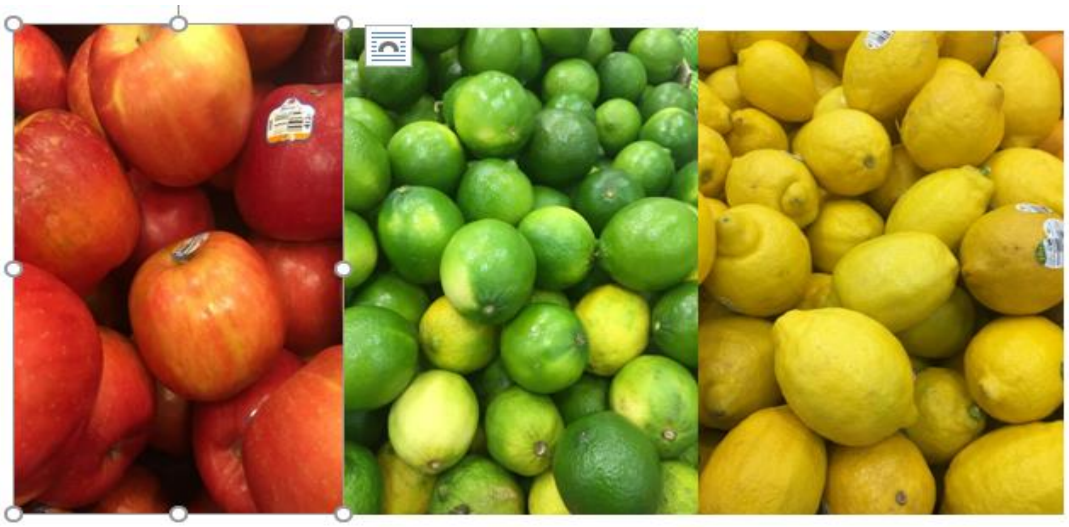


Fig. 2. Spyder environment

III. DOWNLOADING IMAGES TO GENERATE DATASETS

In this assignment we use 3 different fruits to extract features namely Apple, Lemon and Cantaloupe. Images that are used in this assignment are downloaded from VICOS[5]. The exact images used are in Fig.3 left, middle and right respectively.



_

Fig. 3.Left: Apple Raw Image. Middle: Lemon Raw Image. Right: Cantaloupe Raw Image

IV. READING AND DISPLAYING IMAGES

We use OpenCV function, imread() to read a images and assigned them to unique variables. First we are checking on RGB scales of each image and displaying result with help of matplotlib plyplots.

```
image0 = cv2 . imread ( ” . . / d a t a / i n p u t  d a t a / FIDS30  /
     o r a n g e s / 12 . j p g ” )
# t h ir d value  of    array  represents    Red scale  p
l t . imshow ( image0 [ : , : , 0 ] )
# t h ir d value  of    array  represents    Green scale  p
l t . imshow ( image0 [ : , : , 1 ] )
# t h i r d value      of    array  represents    Blue scale
p l t . imshow ( image0 [ : , : , 2 ] )  #same  for  remaining
images
```

Next before extracting features we have to convert the image to Gray scale we are not considering color parameter in this assignment. We use opencv - cvtColor function to convert the Fig.3.left image to gray scale image and displaying dimensions. Similarly, we do same for remaining two fruits.

```
image0Gray  = cv2 . c v t C o l o r ( image0 , cv2 .
     COLOR BGR2GRAY) height image0Gray , width _image0gray = image0Gray . shape
#same  for  remaining  images
```

The original apple image is converted to a apple’s Gray channel image (Fig.5) and it’s shape (480, 640).

Fig. 5. Gray Scale image of Fig.3.left.

V. IMAGE RESIZING TO REDUCE IMAGE DIMENSIONS

We can see there are different dimensions for each input image, this might cause difficulties while extracting features. So, we have to resize the image. Aspect ratio of gray scale image should not be reduced so, we take fixed height value for all images and based on that widths are calculated. We are going to take 81 features. For ease calculations we are taking height as 261 which can be divisible by 12, width can be calculated accordingly. Then passing the new height, width values to resize the gray scale image by OpenCV function.

#image r e s i z e function def i m r e s i z e (h e i g h t , width) : r a t i o =261/ h e i g h t
width = width * r a t i o temp= int (width) / 12 width = int (temp) *12 h e i g h t =261
return (h e i g h t , int (width)) *# r e s i z e function on image0* height image0Gray ,
width image0gray = i m r e s i z e (height image0Gray , width image0gray) *# r e s i z i ~~n~~g*
Gray image r e s i m a g e 0 = cv2 . r e s i z e
(image0Gray , d s i z e
=(width image0gray , h ei ght im ag e0G ra y)
, i n t e r p o l a t i o n =cv2 . INTE~~R~~CUBIC) *#same*
for remaining images

The code will take height and width of a Gray scale image and generates the a new image with resized Gray scale image Fig.6.

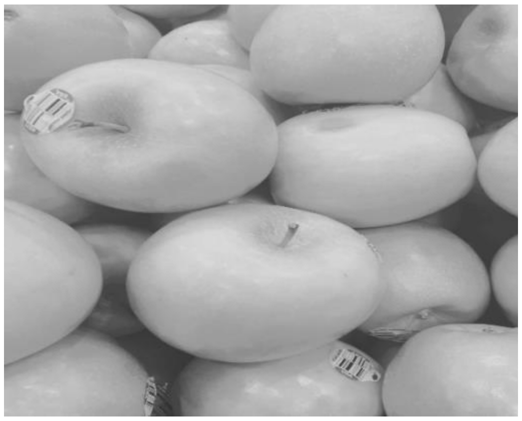


Fig. 6. Resized Gray scale image

VI. 9X9 BLOCK-FEATURE VECTORS

Now, we have to generate block/feature vectors. We are considering for 12x912pixel blocks to extract 81 features in this assignment. In order to do this we are dividing the images to non-overlapping blocks, which are taken with step 9. Each pixel in that block are considered to be a feature. Feature vector values are further used to classify or identify class. Also, here we add label column at end to identify the class. Apple - 0, Lemon - 1, Cantaloupe - 2.

# Create feature vectors using 9X9 matrix and label − 0 for apple , 1 for
lemon and 2 for cantaloupea

oo = round (((height _ image0Gray −9) *(width image0gray −12)) / 144)

f l a t o = np . z e r o s ((oo , 82) , np . u i n t 8) k = 0 for i in range (0 , height image0Gray −12 ,12) : for
j in range (0 , width image0g~~r~~ay −12 ,12) : crop tmp1 = r e s i m a g e 0 [i : i +12 , j : j +12] f l a t o [k ,
0 :144] = crop tmp1 . f l a t t e n
( )

```
            k = k + 1
f s p a c e i m a g e 0 =          pd . DataFrame ( f l a t o ) f s p a c e i m a g e 0 . t o _c s v ( ' . . / d a t a
/ c s v f i l e /  f s p a c e i m a g e 0 . csv ' , i n d e x = F a l s e )
#same  for  remaining  images
```

Fig. 7. Screenshot of the csv file that is generated by 12X12 block feature.

VII. 9X9 SLIDING WINDOW BLOCK-FEATURE VECTORS

A sliding window is a rectangular window that ”slides” across a image and has a fixed width and height. So here it will move one block at a time, this means repeated pixel values observed and every pixel is considered on better side. N number of feature vectors are generated based on nxn pixel blocks, here 81 features as we are takin 12x12 pixel blocks. And this feature vectors are used to classify or identify class of image.

```
# s l i d i n g window 12X12 matrix and labels − 0 for apple , 1 for lemon and 2 for cantaloupe  sw
fspaceimage0 = pd . DataFrame ( s w _ f l a t o ) sw fspaceimage0 . t o  c s v ( ' . . / d a t a / c s v f i l e
/ sw fspaceimage0 . csv ' , i n d e x = F a l s e )
#same for remaining images but lables are allocated accordingly
```

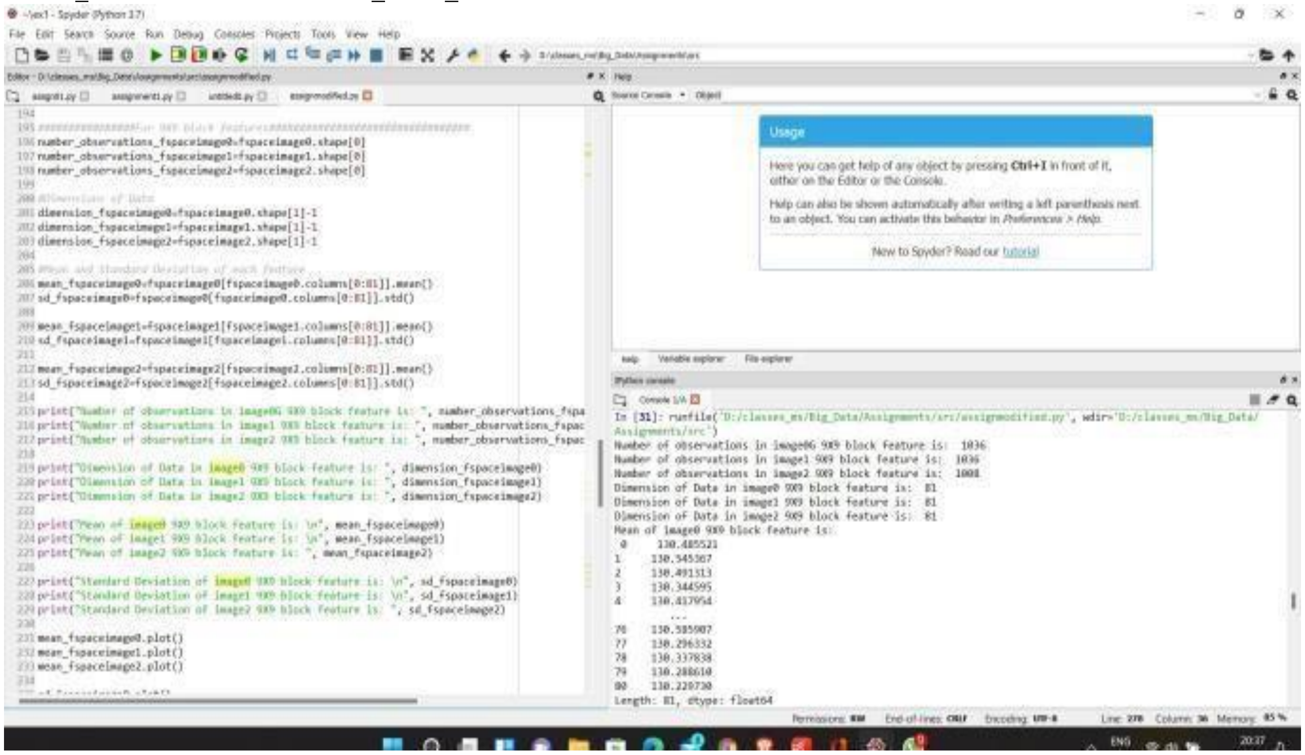
The code will take a Gray scale image with height and weight values. It generates an output of csv file (Fig.8). This csv file will contain value of each pixel of an image with 142 columns i.e total 141 features and 1 column (144) is to label the vegetable/fruit type.
dimension of the data, standard deviations, mean of each feature, from these datasets. Also, presenting visual representations, histogram, scatter plot of the data. Table.1 displays statistical information of each feature of an image.

Number of Observations in apple	1036
Number of Observations in lemon	1036
Number of Observations in cantaloupe	1008
Dimension of Data in apple	144
Dimension of Data in lemon	144
Dimension of Data in cantaloupe	144

TABLE I
STATISTICAL INFORMATION OF 12X12 BLOCK FEATURE

t a b l es i z e , 142) , np

. u i n t 11) k = 0 for i in range (0 , height image0Gray −12) : for j in range (0 , width image0gray −12) : sw crop tmp1 = r e s i m a g e 0 [i : i +12 , j : j +12]-s w f l a t o [k , 0 : 11 1] = sw crop tmp1 . f l a t t e n ( ) k = k + 1



```
##############For 9X9 block features####################################

number_observations_fspaceimage0=fspaceimage0.shape[0]
number_observations_fspaceimage1=fspaceimage1.shape[0]
number_observations_fspaceimage2=fspaceimage2.shape[0]

#Dimension of Data
dimension_fspaceimage0=fspaceimage0.shape[1]-1
dimension_fspaceimage1=fspaceimage1.shape[1]-1
dimension_fspaceimage2=fspaceimage2.shape[1]-1

#Mean and Standard Deviation of each feature
mean_fspaceimage0=fspaceimage0[fspaceimage0.columns[0:81]].mean()
sd_fspaceimage0=fspaceimage0[fspaceimage0.columns[0:81]].std()

mean_fspaceimage1=fspaceimage1[fspaceimage1.columns[0:81]].mean()
sd_fspaceimage1=fspaceimage1[fspaceimage1.columns[0:81]].std()

mean_fspaceimage2=fspaceimage2[fspaceimage2.columns[0:81]].mean()
sd_fspaceimage2=fspaceimage2[fspaceimage2.columns[0:81]].std()

print("Number of observations in image0G 9X9 block feature is: ", number_observations_fspa
print("Number of observations in image1 9X9 block feature is: ", number_observations_fspac
print("Number of observations in image2 9X9 block feature is: ", number_observations_fspac

print("Dimension of Data in image0 9X9 block feature is: ", dimension_fspaceimage0)
print("Dimension of Data in image1 9X9 block feature is: ", dimension_fspaceimage1)
print("Dimension of Data in image2 9X9 block feature is: ", dimension_fspaceimage2)

print("Mean of image0 9X9 block feature is: \n", mean_fspaceimage0)
print("Mean of image1 9X9 block feature is: \n", mean_fspaceimage1)
print("Mean of image2 9X9 block feature is: ", mean_fspaceimage2)

print("Standard Deviation of image0 9X9 block feature is: \n", sd_fspaceimage0)
print("Standard Deviation of image1 9X9 block feature is: \n", sd_fspaceimage1)
print("Standard Deviation of image2 9X9 block feature is: ", sd_fspaceimage2)

mean_fspaceimage0.plot()
mean_fspaceimage1.plot()
mean_fspaceimage2.plot()
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

```
In [31]: runfile('D:/classes_ms/Big_Data/Assignments/src/assignmodified.py', wdir='D:/classes_ms/Big_Data/
Assignments/src')
Number of observations in image0G 9X9 block feature is:  1036
Number of observations in image1 9X9 block feature is:  1036
Number of observations in image2 9X9 block feature is:  1008
Dimension of Data in image0 9X9 block feature is:  81
Dimension of Data in image1 9X9 block feature is:  81
Dimension of Data in image2 9X9 block feature is:  81
Mean of image0 9X9 block feature is:
 0     130.485521
1     130.545367
2     130.491313
3     130.344595
4     130.417954
         ...
76    130.585907
77    130.296332
78    130.337838
79    130.288610
80    130.220730
Length: 81, dtype: float64
```

Fig. 9. Statistical Information Code and Output.

Table.2 displays statistical information of Sliding window features of an image.

Histogram plot on a random feature of each image dataset. The outputs of those Histogram plots are shown in images Fig.11.

Histogram plot on a random Sliding window feature of each image dataset. The outputs of those Histogram plots are shown in images Fig.12.

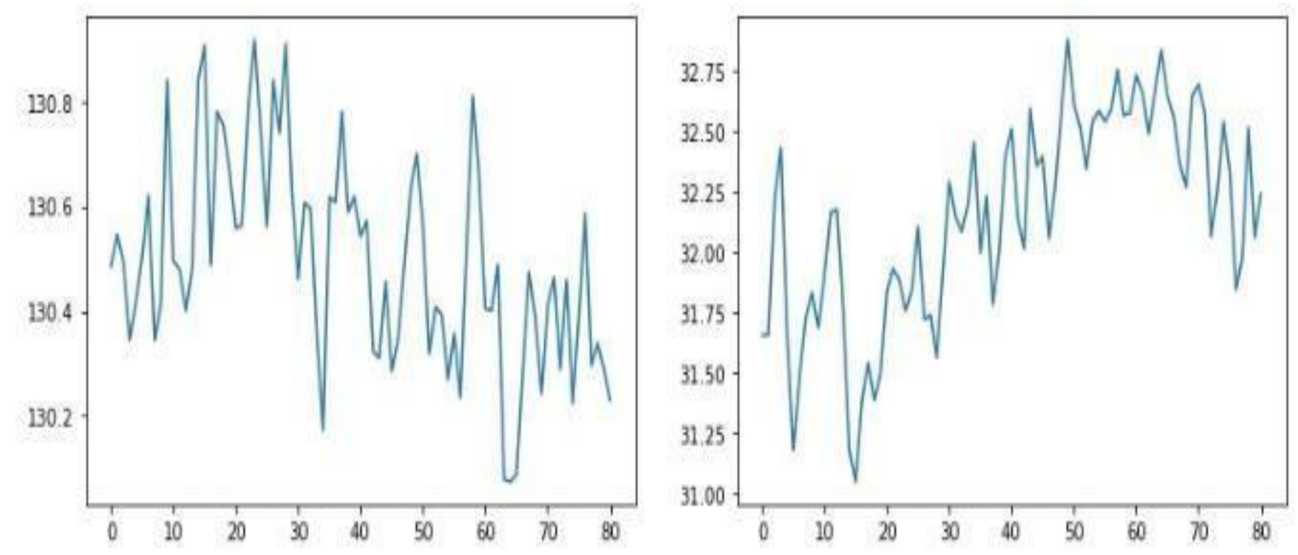


Fig. 10. Left: Mean distribution of each feature of apple image.
Right:Standard deviation distribution of each feature of apple image.

Number of Observations in Apple	83916
Number of Observations in Lemon	83916
Number of Observations in Cantaloupe	81648
Dimension of Data in Apple	144
Dimension of Data in Lemon	144
Dimension of Data in Cantaloupe	144

TABLE II
STATISTICAL INFORMATION OF SLIDING WINDOW FEATURE

A. Is the dataset imbalanced, inaccurate, or incomplete?

By observing statistical information of three datasets, Yes data is imbalanced. The observations vary from one to another datasets (Table.1) so, this is not a balanced data. By, using code below, we will get null values.

```
df =pd.readcsv('../data/csvfile/fspaceimage0.csv') df.
isna().sum().sum()
#same follows for other two datasets
```

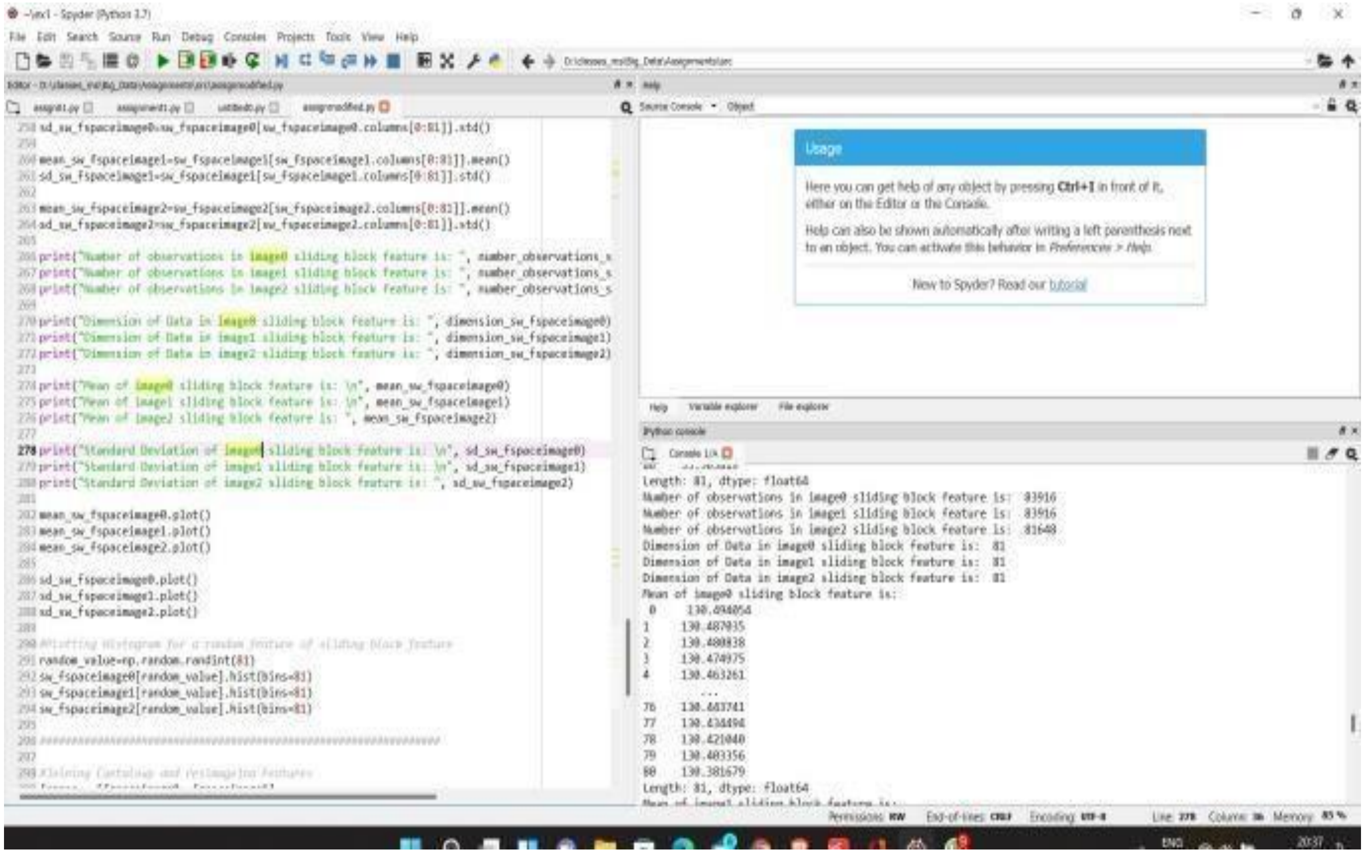
The output is zero. so, there are no null values means the data given is complete. Also, by visualizing the data we got continuous plots means no missing data. The above data is accurate, we have different feature vectors for each labeled class.

B. Is it a trivial data or possibly a big data?

It is trivial data, because data is not that huge and it can be manageable on single system or processor to run predictions or classifications. If we potentially add more features and more observations for single class and repeat on more classes it would be possibly a big data.

C. Does it have scalability problem? Are they high dimensional?

There is no scalability problem in this data because, only observations may change but, we restricted features to 144 for this assignment. Uncontrollable and continuous growth in the features create the scalability problem[6]. No, data is low dimensional. As, Number of observtions (n) is greater than number of features (p).



```
sd_sw_fspaceimage0=sw_fspaceimage0[sw_fspaceimage0.columns[0:81]].std()

mean_sw_fspaceimage1=sw_fspaceimage1[sw_fspaceimage1.columns[0:81]].mean()
sd_sw_fspaceimage1=sw_fspaceimage1[sw_fspaceimage1.columns[0:81]].std()

mean_sw_fspaceimage2=sw_fspaceimage2[sw_fspaceimage2.columns[0:81]].mean()
sd_sw_fspaceimage2=sw_fspaceimage2[sw_fspaceimage2.columns[0:81]].std()

print("Number of observations in image0 sliding block feature is: ", number_observations_s
print("Number of observations in image1 sliding block feature is: ", number_observations_s
print("Number of observations in image2 sliding block feature is: ", number_observations_s

print("Dimension of Data in image0 sliding block feature is: ", dimension_sw_fspaceimage0)
print("Dimension of Data in image1 sliding block feature is: ", dimension_sw_fspaceimage1)
print("Dimension of Data in image2 sliding block feature is: ", dimension_sw_fspaceimage2)

print("Mean of image0 sliding block feature is: \n", mean_sw_fspaceimage0)
print("Mean of image1 sliding block feature is: \n", mean_sw_fspaceimage1)
print("Mean of image2 sliding block feature is: ", mean_sw_fspaceimage2)

print("Standard Deviation of image0 sliding block feature is: \n", sd_sw_fspaceimage0)
print("Standard Deviation of image1 sliding block feature is: \n", sd_sw_fspaceimage1)
print("Standard Deviation of image2 sliding block feature is: ", sd_sw_fspaceimage2)

mean_sw_fspaceimage0.plot()
mean_sw_fspaceimage1.plot()
mean_sw_fspaceimage2.plot()

sd_sw_fspaceimage0.plot()
sd_sw_fspaceimage1.plot()
sd_sw_fspaceimage2.plot()

#Plotting Histogram for a random feature of sliding block feature
random_value=np.random.randint(81)
sw_fspaceimage0[random_value].hist(bins=81)
sw_fspaceimage1[random_value].hist(bins=81)
sw_fspaceimage2[random_value].hist(bins=81)

###################################################################

#Joining Catalog and fspaceimage features
```

Usage

Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.

New to Spyder? Read our tutorial

```
Length: 81, dtype: float64
Number of observations in image0 sliding block feature is:  83916
Number of observations in image1 sliding block feature is:  83916
Number of observations in image2 sliding block feature is:  81648
Dimension of Data in image0 sliding block feature is:  81
Dimension of Data in image1 sliding block feature is:  81
Dimension of Data in image2 sliding block feature is:  81
Mean of image0 sliding block feature is:
 0     130.494054
1     130.487035
2     130.480838
3     130.474975
4     130.463261
         ...
76    130.443741
77    130.434494
78    130.421040
79    130.403356
80    130.381679
Length: 81, dtype: float64
```

Fig. 11. Statistical Information of sliding window dataset’s Code and Output.

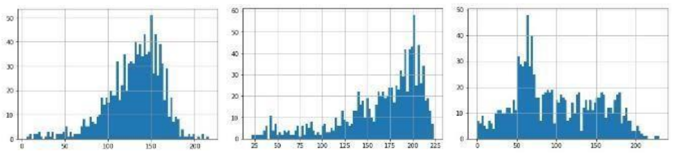


Fig. 12. Left: Histogram plot of Apple. Middle: Histogram plot of Lemon. Right: Histogram plot of Cantaloupe

D. Do you need to standardize?

We do not standardize the data as the all features are comparable and distribution is observed by plotting and it is not in gaussian distribution too. *E. Do you need to normalize?*

```
df = pd . r e a d c s v ( ' . . / d a t a / c s v f i l e / f s p a c e i m a g e 0 . csv ' ) df .
d e s c r i b e ( )
#same for remaining images
```

By executing the above code we can understand feature wise statistical data, even after reducing the image dimensions and generating feature vectors we can see that all values will be in between 0 - 261. And, standard deviations of each feature tells
the data across the features is in same range. So, no normalization is required.

F. How do they affect the data characteristics?

The given data is managable and there is no constant growth in features, this does not have any effect on data characteristics. If we normalize or standardize, if some feature vectors normalized and attain same values as other feature vector labeling different class, will lead to inaccurate data.

IX. CONSTRUCTION OF FEATURE SPACE

A. Merge feature vectors of image0 and image1

Based on 82th column, here features vectors of 2 different fruits are merged and randomly arranged.

Fig. 14. Left: Histogram plot of sw Apple. Middle: Histogram plot of sw Lemon. Right: Histogram plot of sw Cantaloupe

```
# Joining         Apple  and Lemon      Features
f ra me s =       [ fspaceimage0 , f s p a c e i m a
                  g e 1
] merge1          =        pd . c o n c a t ( fr a me s )
#Randomly         arranging       the       rows in de
x1       =        np . a r a n g e ( len ( merge1 ) )
random merge1          =          np . random . p e r  m u t a t i o n (  in de x1 ) # Resetting index value
random merge1=merge1 . sample ( f r a c =1) . r e s e t i n d e
x ( drop =True )  random_       merge1 . t o  c s v ( ' . . / d a t a / c s v f i l e /            Merged
      image0image1 . csv ' ,            i n d_e x = F a l s e )
```

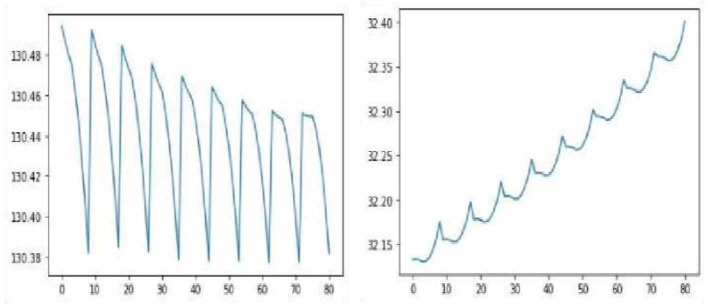
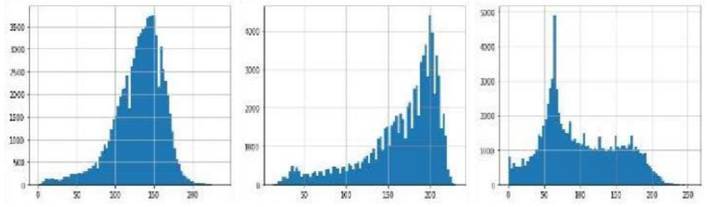
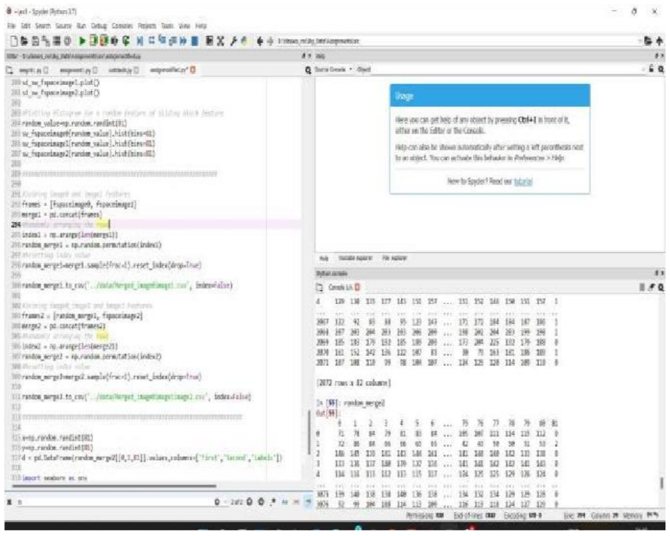


Fig. 13. Left: Mean distribution of each sw feature of Apple image. Right:Standard deviation distribution of each sw feature of Apple image.





```
sd_sw_fspaceimage1.plot()
sd_sw_fspaceimage2.plot()

#Plotting Histogram for a random feature of Sliding Block feature
random_value=np.random.randint(81)
sw_fspaceimage0[random_value].hist(bins=81)
sw_fspaceimage1[random_value].hist(bins=81)
sw_fspaceimage2[random_value].hist(bins=81)

##################################################################

#Joining Image0 and Image1 Features
frames = [fspaceimage0, fspaceimage1]
merge1 = pd.concat(frames)
#Randomly arranging the rows
index1 = np.arange(len(merge1))
random_merge1 = np.random.permutation(index1)
#Resetting index value
random_merge1=merge1.sample(frac=1).reset_index(drop=True)

random_merge1.to_csv('../data/Merged_image0image1.csv', index=False)

#Joining Image0, Image1 and Image2 Features
frames2 = [random_merge1, fspaceimage2]
merge2 = pd.concat(frames2)
#Randomly arranging the rows
index2 = np.arange(len(merge2))
random_merge2 = np.random.permutation(index2)
#Resetting index value
random_merge2=merge2.sample(frac=1).reset_index(drop=True)

random_merge2.to_csv('../data/Merged_image0image1image2.csv', index=False)

##################################################################

x=np.random.randint(81)
y=np.random.randint(81)
d = pd.DataFrame(random_merge2[[0,1,81]].values,columns=['First','Second','Labels'])

import seaborn as sns
```

```
[2072 rows x 82 columns]

In [99]: random_merge2
Out[99]:
```

Fig. 15. Code and Output for generating merged datasets.

Feature vectors of Apple and Feature space of Lemon are the inputs for this code. A data of randomly arranged lemon and apple is the output of this code.

B. Merge feature space of image0image1 with feature vector of image2

Here the feature space generated in step 8.1 is merged with feature vectors of cantaloupe, similar to step 8.1.

frames2 = [random merge1 , f s p a c e i m a g e 2] merge2 = pd . c o n c a t (frames2) *#Randomly arranging the rows* in de x2 = np . a r a n g e (len (merge2)) random merge2 = np . random . p e r m u t a t i o n (in de x2) *# Resetting index value* random merge2=merge2 . sample (f r a c =1) . r e s e t i n d e x (drop =True) random merge2 . t o c s v (' . . / d a t a / c s v f i l e / Merged image0image1image2 . csv ' , i n d e x = F a l s e)

Feature space of AppleLemon or image0image1 and feature vector of Cantaloupe are the inputs of this code. A feature space of all the 3 values arranged randomly will be the output of this code.

X. DISPLAY SUBSPACESL

A. 2D-Plot with 2 random features from Feature Space

Two random values are selected using np.random.randint(81). Scatter plot is plotted on Feature Space and labels are displayed.

x=np . random . r a n d i n t (144) y=np . random . r a n d i n t (144) d = pd . DataFrame (random merge2 [[0 , 1 , 144]] . v a l u e s , columns =[' F i r s t ' , ' Second ' , ' L a b e l s ']) import s e a b o r n as s n s s n s . s c a t t e r p l o t (d a t a =d , x= ' F i r s t ' , y= '

Second ' , hue= ' L a b e l s ')

2 random values X and Y are considered and final feature space of AppleLemonCantaloupe/image0image1image2 is given as input to the above code. A scattered plot differentiated with colour based on the labels. The output will be 2D scatter plot Fig.16. We can see that three fruits distributions on the graph, the

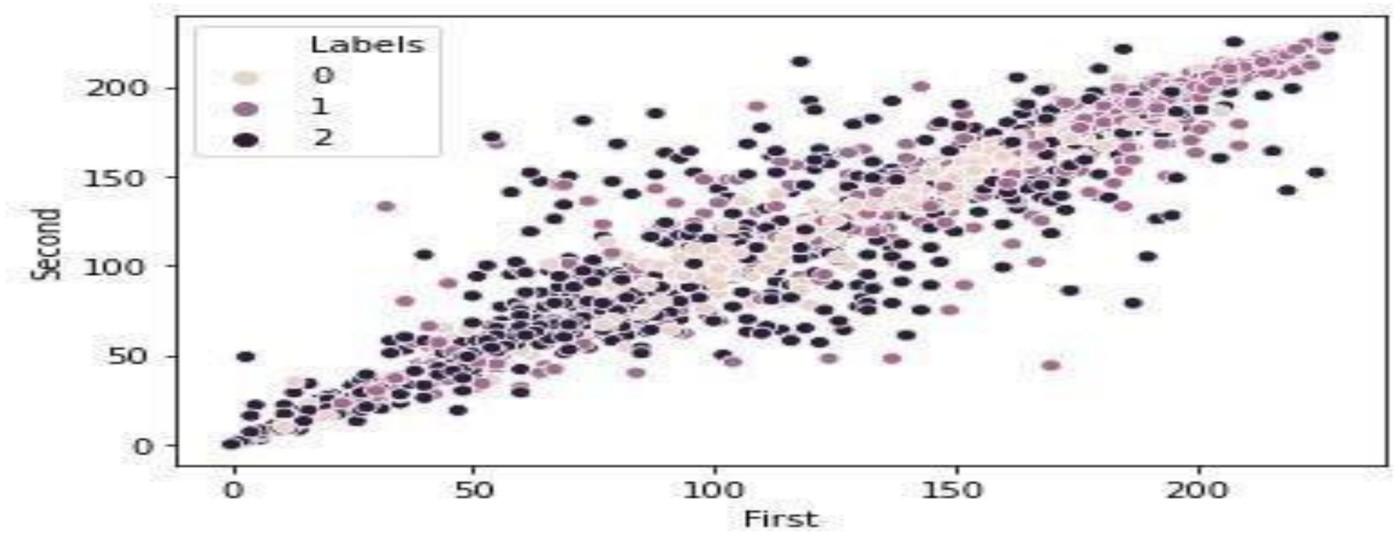


Fig. 16. 2D scatter plot on Feature space for two features.

distribution is slant across the graph. Majority of Apple data is present in middle, lemon data is to the right most part and in outlier's of distribution, cantaloupe were at left most part and in outlier parts of distribution. So, we can not exactly tell how they can be differentiated now as we are projecting in 2 - D space.

B. 3D-Plot with 3 features from Feature Space

3 different classes observations as axes are plotted on 3D plot. The datapoints on plot are based on label and associated colors.

```
import m a t p l o t l i b f i g = p l t . f i g u r e ( ) ax = p l t . axes ( p r o j e c t i o n = ' 3d ' ) bx =
randommerge2 [ 2 2 ] by = randommerge2 [ 7 ] bz = randommerge2 [ 1 7 ] ax . s c a t t e r 3 D ( bx

, by , bz , c=random merge2 [ 8 1 ] , cmap= m a t p l o t l i b . c o l o r s . L i s t e _ d C o l o r m a p (
[ ' r e d ' , ' b l u e ' , ' g r e e n ' ] )
      )
p l t . show ( )
```

The input is feature space of AppleLemonCantaloupe. The output will be 3D scatter plot Fig.17. Now, we see the red points Apple data is in middle, green

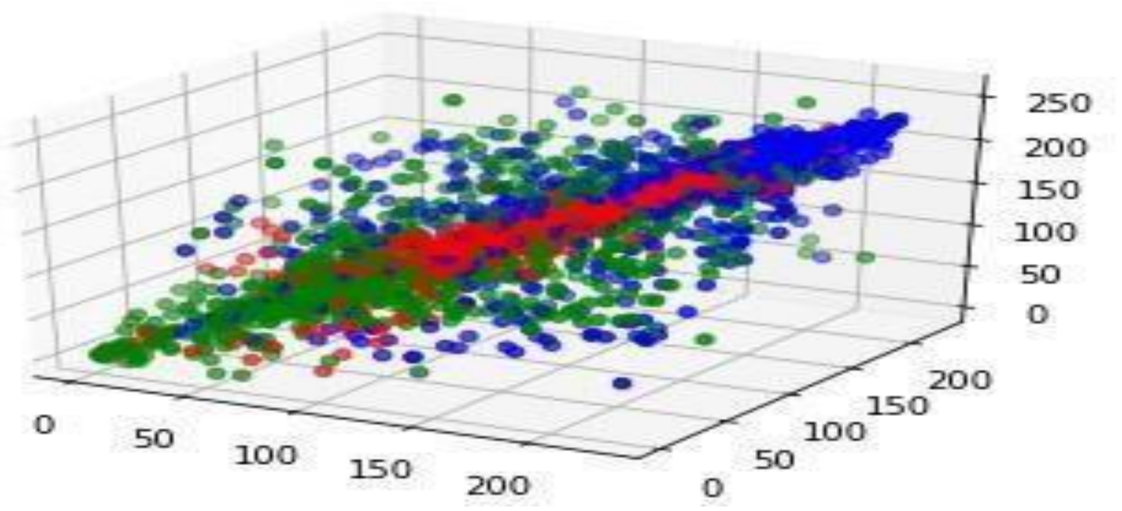


Fig. 17. 3D scatter plot plotted using 3 different variables from feature space.

points cantaloupe to the top and lemons were at the bottom. Now, we can potentially think for two planes classifying them.

XI. READ ANY NUMBER OF INPUTS

Modified code such that it can read any number of images from a folder that consists of many similar images, generate a feature spaces, and generate a spreadsheets for the feature spaces. import pandas as pd import cv2

```
import m a t p l o t l i b . p y p l o p l t
        as t
import numpy as np         import  os

def   i m r e s i z e ( h e i g h t , width ) : r a t i o =261/ h e i g h t width = width * r a t i
 o temp= int ( width ) / 9 width
 = int ( temp ) *12 h e i g h t =261 return
 ( h e i g h t , int ( width ) ) p a t h = ' . . / d a t a / i n p u t d a t a / FIDS30 / ' f e a t u r e n a m e = str ( input
( ' Name of F r u i t s : ␣
     ' ) )

c o u n t = int ( input ( ' E n t e r number␣of␣ f i l e s :
     ' ) )
for i in range ( 0 , c o u n t ) :
    i g = str ( input ( ' E n t e r F i l e name : ' ) ) imagepath = p a t h + f e a t u r e n a m e + ' / '+ i g image=
    cv2 .
    imread ( imagepath )   image=cv2 . c v t C o l o r ( image ,   cv2 . COLOR
       BGR2GRAY) raw height , raw width =
image . shape h e i g h t , width = i m r e s i z e ( raw  height
, raw width )

_
image=cv2 . r e s i z e ( image ,  d s i z e =( width , h e i g h t ) ,           i n t e r p o l a t i o n =cv2 .
                                                                 INTER CUBIC ) p l t . imshow ( image , cmap= p l t
. get cmap ( ' gray ' ) )  p
l t . a x i s ( ' o f f ' )
      r it m p = np . z e r o s ( ( h e i g h t , width )
      , np  . u i n t 8 ) r it h 1 = image .
 _ mean ( ) for k in range ( h e i g h t ) :  for l in range ( width ) :  i f
          ( image [ k ] [ l ]< r i  t h 1 ) : r i t m p [ k ] [ l ] = 0 else :
                 r i
t m p [ k ] [ l ] = 261  _-  p l t . imshow ( ri  tmp ,  cmap= p l t . get  _
cmap (
     ' gray ' ) ) r io o = round ( ( ( h e i g h t −12) *( width

     −12) ) _        / 142 ) r i f l a t o = np . z e r o s ( ( r i  o o , 142) , np . u  i n t 8 ) p = 0 for k in range
( 0 , h e i g h t −9 ,9) :      for       l          in range ( 0 , width −12 ,12)
:  r i c r o p t m p 1 = image [ k : k +12 , l : l +12]        r i f l a t o [ p , 0 : 144 ] =        r i c r o p t m p 1 . f l a t
t e n ( )

               _
               p = p + 1  _

                                                  _
r if s p a c e O = pd . DataFrame ( r i  f l a t o ) r i  f  s
p a c e O . t o c s v ( ' . . / d a t a / c s v f i l e /
 f s p a c e ' + f e a t u r e n a m e + str ( i ) + ' . csv '
   , i n d e x = F a l s e )
```
Fig. 18. Code and output for Multiple image inputs.

XII. BLOCK SIZE MODIFICATION EFFEVTS ON DIMENSIONALITY AND DOMAIN CLASSIFICATION

If we reduce the block size, the features are reduced and observations are increased. Vice-versa if we increase the block size. And, observations will determine the dimensionality of feature space. More observations means increase in dimensionality and results in complexity. As of features will help to classify more accurately, means more features with sufficient observations gives the better classification as a we will get more training data to increase overall accuracy. Here, block size modification will change these parameters and r i o t a b l e_s i z e = round ((h e i g h t −12) *(width −12))

```
     r i s w f l a t o = np . z e r o s ( (          r i o t a b l e_s i z e ,  142) , np . u i n t 11) p = 0 for k in
range ( 0 , h e i g h t −12) :     for       l          in range ( 0 , width −12) :       ri sw  crop  tmp1 =
image [ k : k
```

```
                    +12 , l : l +12] r i s w f l a t o [ p ,-0 : 11-1 ] =
                      ri sw  crop  tmp1 . f l a t t e n ( )
                p = p + 1
          r i_s w_f s p a c e C =    pd . DataFrame (        r i s w f l a t o )
        r i-s w f-s p a c e C . t o  c s v ( ' . . / d a t a / c s v f i l e /  sw fs-pace ' + f e a t u r e  n a m e + str ( i ) + ' .
            csv  ' , i n d e x = F a l s e )
```

Above code will take count of the number of images to be
read, file folder name and each image name will looping through initial count we declared. This generates .csv files with features for all the images we gave.

results effect on dimensionality and domain classificstion accordingly based on above statements. Given 12x12 block is considerably good feature space to run different classifications/regressions and more. REFERENCES

[1] https://www.section.io/engineeringeducation/why-python-is-goodformachinelearning/

[2] https://docs.anaconda.com/anaconda/

[3] https://www.anaconda.com/products/individua
l

[4] https://docs.anaconda.com/anaconda/install/w indows/

[5] https://www.vicos.si/Downloads/FIDS30

[6] 2.3 Big Data Scalability, Machine Learning
Models and Algorithms for Big Data
Classification – Shan Suthahara