In [3] : %matplotlib inline

#1

import pandas as pd

import matplotlib.pyplot as plt #End of #1

#1

fruits = pd.read\_csv('D:\\As a Trainer\\Freelance Training\\ML-N

IVT-2\\Classification\\fruits.csv')

In [4] : print("Printing Table: ")

fruits.head()

Printing Table :

Out[4] :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **SI** | **Name** | **Category** | **Supplier** | **Brand** | **Price**  **Group** |
| 0 | 1 | Apple | A | A.Co. | Nutts | P1 |
| 1 | 2 | Apple | A | Fruits\_Lover | Fruits#1 | P2 |
| 2 | 3 | Lemon | B | Fruits\_Lover | Fruits#1 | P2 |
| 3 | 4 | Watermelon | B | A.Co. | Fruits#1 | P1 |
| 4 | 5 | Apple | B | A.Co. | Costa  Rica | P2 |

In [5]: # Print the list of fruits.txt upto row=4

print("Matrix Size:")

print(fruits.shape)

print("Coloumn Headings:")

print(fruits.columns)

Matrix Size:

(60, 9)

Coloumn Headings:

Index(['Sl', 'Name', 'Category', 'Supplier', 'Brand', 'Price

Group',

'Color Score', 'Purification', 'Customer Demand'],

dtype='object')

In [6]: # Returns the names of unique fruits

print(fruits['Name'].unique())

['Apple' 'Lemon' 'Watermelon' 'Grapes']

In [8]: # Returns the name of fruits with their number of instances

print(fruits.groupby('Name').size()) #End of #1

Name

Mango 5

apple 19

lemon 16

orange 19

dtype: int64

In [7]: #2

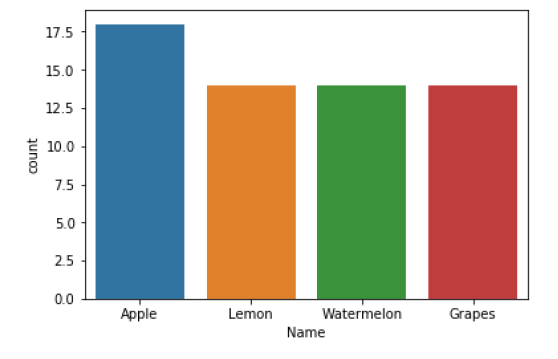
import seaborn as sns #End of #2

#2

# Plot a BarChart for Fruits and their counts

sns.countplot(fruits['Name'])

plt.show()



In [10]: #3

import pylab as pl #End of #3

#3

# Plot Histograms Different Colomns

fruits.drop('Sl' ,axis=1).hist(bins=40, figsize=(11,11))

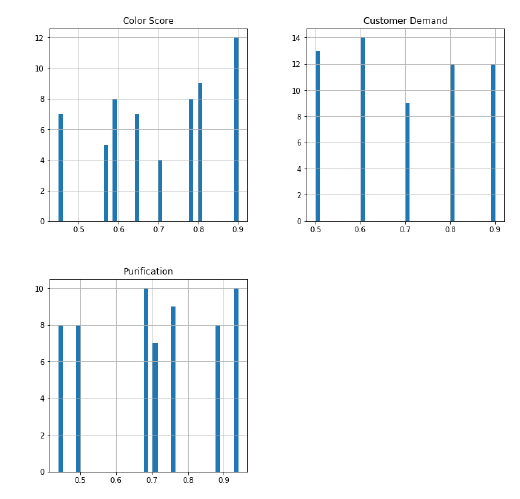
pl.suptitle("Histogram for each numeric input variable vs Other

s")

plt.savefig('fruits\_hist')

plt.show()





In [11]: fig=plt.figure()

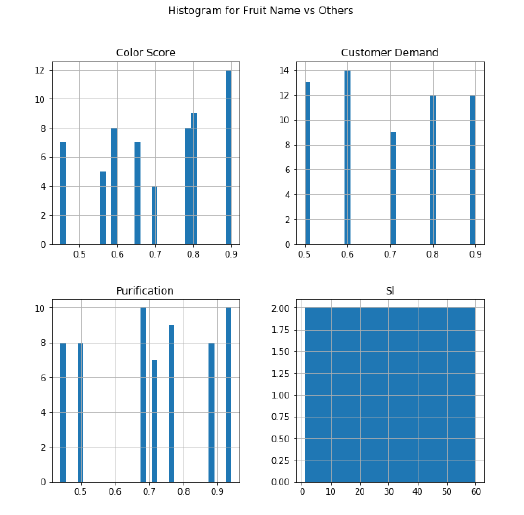
fruits.drop('Name' ,axis=1).hist(bins=30, figsize=(9,9))

pl.suptitle("Histogram for Fruit Name vs Others")

plt.savefig('fruits\_hist')

plt.show()

<Figure size 432x288 with 0 Axes>



In [28]: fig=plt.figure()

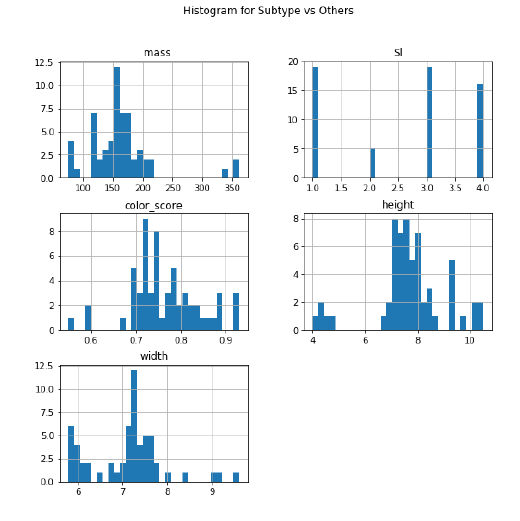
fruits.drop('Subtype' ,axis=1).hist(bins=30, figsize=(9,9))

pl.suptitle("Histogram for Subtype vs Others")

plt.savefig('fruits\_hist')

plt.show() #End of #3

<Figure size 432x288 with 0 Axes>



In [13]: #4

from pandas.tools.plotting import scatter\_matrix

from pandas.plotting import scatter\_matrix

from matplotlib import cm #End of #4

#4

cmap = cm.get\_cmap('gnuplot')

feature\_names =['Color Score','Purification','Customer Demand']

X = fruits[feature\_names]

y = fruits['Sl']

plt.figure()

scatter = pd.scatter\_matrix(X, c = y, marker = 'o', s=40, hist\_k

wds={'bins':15}, figsize=(13,13), cmap = 'viridis')

plt.suptitle('Scatter-matrix for each input variable using VIRID

IS COLORMAP')

plt.figure()

scatter = pd.scatter\_matrix(X, c = y, marker = 'o', s=40, hist\_k

wds={'bins':15}, figsize=(13,13), cmap = cmap)

plt.suptitle('Scatter-matrix for each input variable using GNUPL

OT COLORMAP')

plt.savefig('fruits\_scatter\_matrix') #End of #4

C:\Users\Subhadeep Chakrabort\Anaconda3\lib\site-packages\ipy

kernel\_launcher.py:14: FutureWarning: pandas.scatter\_matrix i

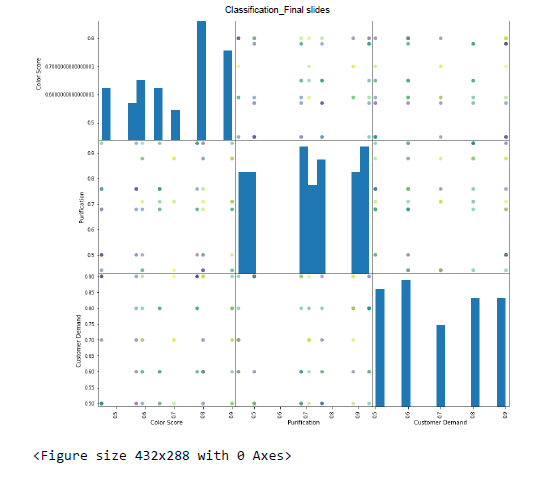
s deprecated, use pandas.plotting.scatter\_matrix instead

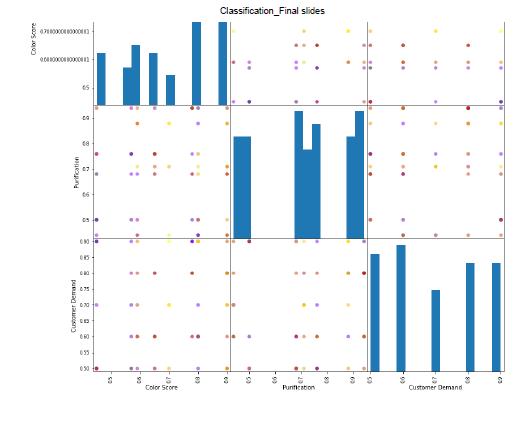
C:\Users\Subhadeep Chakrabort\Anaconda3\lib\site-packages\ipy

kernel\_launcher.py:18: FutureWarning: pandas.scatter\_matrix i

s deprecated, use pandas.plotting.scatter\_matrix instead

<Figure size 432x288 with 0 Axes>





In [106]: #5

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, random

\_state=0)

#X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, rando

m\_state=0)

from sklearn.preprocessing import MinMaxScaler #5

#5=> Training data test upon the previous algorithm & Simulation

scaler = MinMaxScaler()

#fit\_transform(X[, y]) => Fit to data, then transform it.

X\_train = scaler.fit\_transform(X\_train)

#transform(X) => Scaling features of X according to feature\_rang

e.

X\_test = scaler.transform(X\_test)

label=['Mass Training Data','Height Training Data']

print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Printing Training Dat

a: ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")

print(X\_train)

print("\n")

print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Printing Test Da

ta: ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")

print(X\_test)

print("\n")

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Printing Training Data: ~

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[[0.27857143 0.49230769 0.41176471 0.72972973]

[0.35 0.93846154 0.44117647 0.45945946]

[0. 0. 0. 0.7027027 ]

[0.27142857 0.50769231 0.52941176 0.37837838]

[0.31428571 0.46153846 0.41176471 0.67567568]

[0.2 0.72307692 0. 0.48648649]

[0.3 0.52307692 0.35294118 0.7027027 ]

[0.27857143 0.50769231 0.44117647 0.64864865]

[0.14285714 0.63076923 0.02941176 0.48648649]

[0.30714286 0.47692308 0.5 0.75675676]

[0.14285714 0.56923077 0.14705882 0.45945946]

[0.45714286 0.8 0.5 0.59459459]

[0.28571429 0.47692308 0.55882353 0.37837838]

[0.01428571 0.04615385 0. 0.59459459]

[0.47857143 0.61538462 0.58823529 0.72972973]

[0.28571429 0.53846154 0.52941176 0.32432432]

[0.19285714 0.64615385 0.05882353 0.43243243]

[0.3 0.53846154 0.5 0.83783784]

[0.14285714 0.53846154 0.05882353 0.45945946]

[0.36428571 0.58461538 0.38235294 1. ]

[0.31428571 0.56923077 0.44117647 0.40540541]

[1. 0.8 1. 0.54054054]

[0.32857143 0.55384615 0.5 0.48648649]

[0.15 0.63076923 0.08823529 0.40540541]

[0.27142857 0.69230769 0.20588235 0.45945946]

[0.15 0.61538462 0.02941176 0.45945946]

[0.29285714 0.58461538 0.41176471 0.59459459]

[0.37142857 0.43076923 0.64705882 0.10810811]

[0.27857143 0.47692308 0.35294118 0.89189189]

[0.39285714 0.8 0.41176471 0.45945946]

[0.95 0.83076923 0.94117647 0.54054054]

[0.01428571 0.04615385 0.02941176 0.7027027 ]

[0.33571429 0.6 0.52941176 0.89189189]

[0.3 0.55384615 0.38235294 0.56756757]

[0.28571429 0.52307692 0.47058824 0.78378378]

[0.30714286 0.49230769 0.47058824 0.81081081]

[0.34285714 0.46153846 0.47058824 0.91891892]

[0.24285714 0.52307692 0.29411765 0.54054054]

[0.14285714 0.69230769 0.08823529 0.43243243]

[0.03571429 0.10769231 0.11764706 0.67567568]

[0.41428571 0.50769231 0.76470588 0. ]

[0.15714286 0.67692308 0.05882353 0.51351351]

[0.42857143 0.87692308 0.44117647 0.45945946]

[0.44285714 1. 0.44117647 0.45945946]]

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Printing Test Data:

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[[1.02142857 0.8 1.11764706 0.51351351]

[0.26428571 0.6 0.38235294 0.54054054]

[0.42142857 0.96923077 0.41176471 0.40540541]

[0.22857143 0.47692308 0.26470588 0.45945946]

[0.34285714 0.55384615 0.38235294 1. ]

[0.35714286 0.49230769 0.47058824 0.13513514]

[0.23571429 0.58461538 0.52941176 0.54054054]

[0.5 0.95384615 0.44117647 0.43243243]

[0.27857143 0.53846154 0.38235294 0.62162162]

[0.22857143 0.47692308 0.44117647 0.86486486]

[0.02857143 0.09230769 0.05882353 0.64864865]

[0.32142857 0.50769231 0.32352941 1.02702703]

[0.29285714 0.53846154 0.38235294 0.64864865]

[0.37142857 0.64615385 0.52941176 0.64864865]

[0.40714286 0.63076923 0.5 0.51351351]]

In [104]: X1=X\_train.transpose()

plt.figure()

plt.plot(X1[0],label='Mass')

plt.plot(X1[1],label='Width')

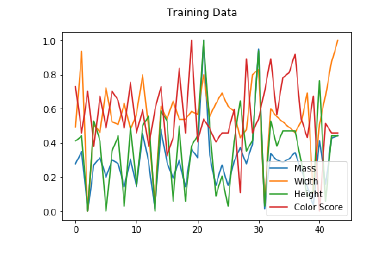
plt.plot(X1[2],label='Height')

plt.plot(X1[3],label='Color Score')

plt.legend(loc="lower right")

plt.suptitle('Training Data')

Out[104]: Text(0.5,0.98,'Training Data')



In [84]: X2=X\_test.transpose()

plt.plot(X2[0],label='Mass')

plt.plot(X2[1],label='Width')

plt.plot(X2[2],label='Height')

plt.plot(X2[3],label='Color Score')

plt.legend(loc="lower right")

plt.suptitle('Test Data')

Out[84]: Text(0.5,0.98,'Test Data')

