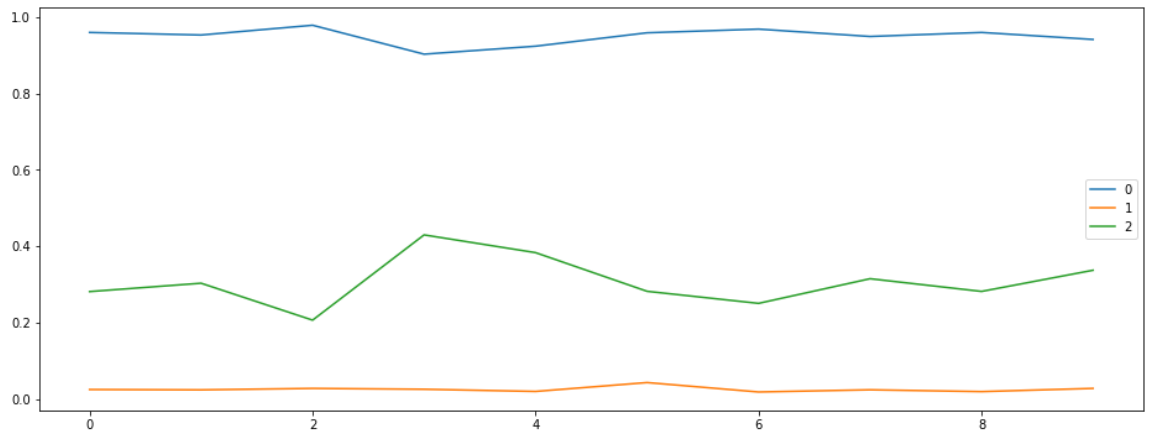
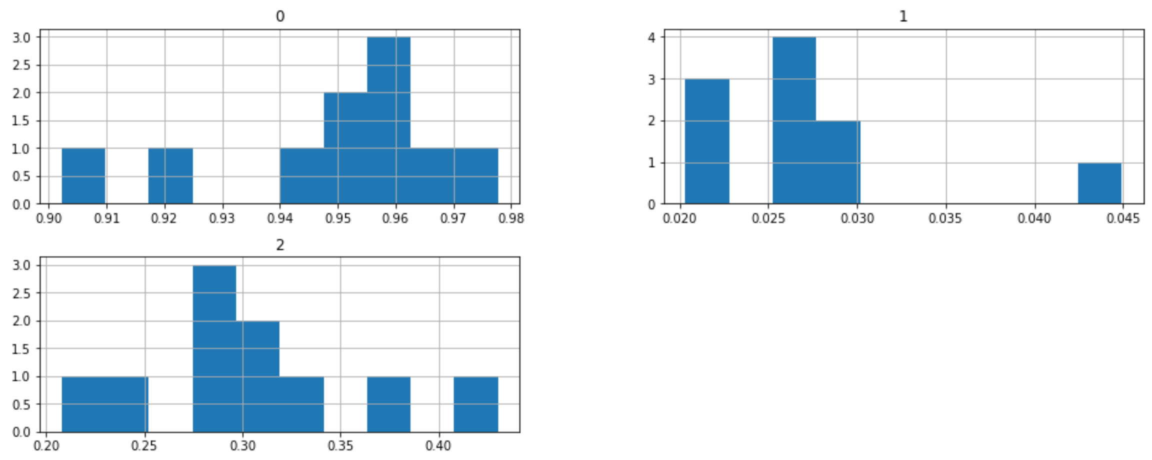
In [39]:

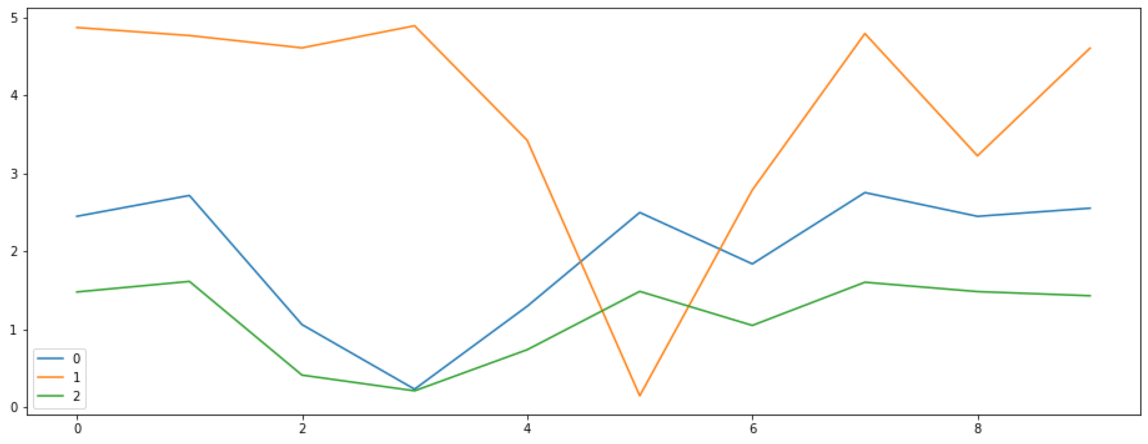
```
#Cristian Rony Yaguno Mamani
#laboratorio de inteligencia Artificial
#tema:Algoritmo deteccion de anomalias
import matplotlib.pyplot as plt # importando matplotlib
import seaborn as sns # importando seaborn
%matplotlib inline
import pandas as pd
import math
from sklearn import preprocessing
from numpy import mean
import numpy as np
from scipy import stats # importando scipy.stats
import pandas as pd # importando pandas
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = (16.0, 6.0)
#1. caracteristicas de anomalias
p=np.array([50.93,50.08,51.74,50.31,50.47,50.75,49.98,51.02,50.85,50.07])
y=np.array([1.42,1.36,1.57,1.53,1.19,2.38,1.05,1.4,1.13,1.58])
z=np.array([15,16,11,24,21,15,13,17,15,18])
X = np.array(list(zip(p,y,z))).reshape(len(p), 3)
dataframe = pd.DataFrame(X, index=['1', '2', '3', '4','5','6','7','8','9','10'],
                        columns=['peso', 'talla', 'edad'])
print(dataframe)
x_normalized = preprocessing.normalize(X, norm='l2')
#forma normalizad
print('Normalizada')
print(x_normalized)
data = pd.DataFrame(x_normalized)
data.plot()
#------------------------------------------------------------
#2. establecer  la media y desviacion estadar o varianza
#promedio por cada columna
n=10
i = 0
while i < n:
   i += 1
M1 = sum(x_normalized)/n
print ('Media')
print(M1)
M2=np.mean(x_normalized, axis=0)
print('Media con libreria')
print(M2)
#M1=mean(x_normalized.T)
#print(M1)
#----------------------------------------------------------------------------
#fórmula de desviación estándar y varianza
for dato in x_normalized:
  r = (x_normalized - M1)**2
  V = sum(r)/n #varianza
print ('Varianza σ2')
print (V)
V1=np.var(x_normalized,axis=0)
print('varianza con libreria')
print(V1)
D=np.sqrt(sum((x_normalized - M1)**2)/n)#desviacion estadar

print ('desviacion estandar σ')
print (D)
D1=np.std(x_normalized,axis=0)
```

```
D2=np.std(x_normalized)
print('Desviacion estadar con libreria')
print(D1)
print('Desviacion estadar  de todo')
print(D2)
#---------------------------------------------------------
#data = pd.DataFrame(np.random.randn(250))
data = pd.DataFrame(x_normalized)
data.hist()
#-----------------------------------------------------------
#3. Dando un nuevo ejemplo x,calcular p(x)
px=(1/(np.sqrt(2 * np.pi*D1)))*np.exp( - ( x_normalized- M1)**2 / (2 * D1**2) )
print('calcular p(x)')
print(px)
dat = pd.DataFrame(px)
dat.plot()

#4. Evaluar si: p(x)<E existen anomalias
error=0.1
#for n in px:
#    if px<error:
#       print('Existen anomalias')
```

```
     peso  talla  edad
1   50.93   1.42  15.0
2   50.08   1.36  16.0
3   51.74   1.57  11.0
4   50.31   1.53  24.0
5   50.47   1.19  21.0
6   50.75   2.38  15.0
7   49.98   1.05  13.0
8   51.02   1.40  17.0
9   50.85   1.13  15.0
10  50.07   1.58  18.0
Normalizada
[[0.95891756 0.02673597 0.28242221]
 [0.95224683 0.02585974 0.30423222]
 [0.97770811 0.0296676  0.20786218]
 [0.90222205 0.02743788 0.43039812]
 [0.92304755 0.02176395 0.38406972]
 [0.95802017 0.04492784 0.28315867]
 [0.96759801 0.02032769 0.25167615]
 [0.94839916 0.02602428 0.31600913]
 [0.95892214 0.02130938 0.28286789]
 [0.94062363 0.02968215 0.33815109]]
Media
[0.94877052 0.02737365 0.30808474]
Media con libreria
[0.94877052 0.02737365 0.30808474]
Varianza σ2
[4.39045792e-04 4.41154214e-05 3.68581560e-03]
varianza con libreria
[4.39045792e-04 4.41154214e-05 3.68581560e-03]
desviacion estandar σ
[0.02095342 0.00664194 0.06071092]
Desviacion estadar con libreria
[0.02095342 0.00664194 0.06071092]
Desviacion estadar  de todo
0.38740677943100704
calcular p(x)
[[2.45108608 4.87260156 1.48073644]
 [2.71835157 4.76958977 1.61585454]
 [1.06199295 4.61169386 0.4144846 ]
 [0.23368279 4.89488114 0.21275787]
 [1.29726088 3.42661661 0.73981337]
 [2.50015827 0.1489264  1.48823905]
 [1.84061759 2.78865352 1.05151525]
 [2.75558875 4.79512638 1.60537718]
 [2.45082647 3.22659856 1.48529829]
 [2.55538185 4.6081951  1.43225048]]
```




0
1
2
0.0 0.5 1.0 1.5 2.0 2.5 3.0
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98
0 1 2 3 4
0.020 0.025 0.030 0.035 0.040 0.045
0.20 0.25 0.30 0.35 0.40


0
1
2
0 1 2 3 4 5
0 2 4 6 8