In [1]:
```
import pandas as pd
import os
```

In [9]:
```
os.chdir("C:\\Users\\wichi\\OneDrive\\Documentos\\EBAC\\Data Science\\Actividades\\Clu
df_original = pd.read_excel("Amazon.xlsx")
df_original.head()
```

Out[9]:

	Cliente	Velocidad Entrega	Precio	Durabilidad	Imagen Producto	Valor Educativo	Servicio Retorno	Tamano Paquete	Calidad Producto	Numer Estrell
0	Adam	205	3	345	235	24	23	26	21	1
1	Anna	9	15	315	33	25	4	42	215	2
2	Bernard	17	26	285	3	43	27	41	26	3
3	Edward	135	5	355	295	18	23	39	195	1
4	Emilia	3	45	48	39	34	46	225	34	4

In [10]:
```
df_original.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Cliente            100 non-null    object
 1   Velocidad Entrega  100 non-null    int64 
 2   Precio             100 non-null    int64 
 3   Durabilidad        100 non-null    int64 
 4   Imagen Producto    100 non-null    int64 
 5   Valor Educativo    100 non-null    int64 
 6   Servicio Retorno   100 non-null    int64 
 7   Tamano Paquete     100 non-null    int64 
 8   Calidad Producto   100 non-null    int64 
 9   Numero Estrellas   100 non-null    int64 
dtypes: int64(9), object(1)
memory usage: 7.9+ KB
```

In [29]:
```
#Normalizar los datos
from sklearn.preprocessing import normalize
ar_normalize = normalize(df_original.iloc[:,1:])
```

In [21]:
```
columns = df_original.columns.tolist()
df_normalize = pd.DataFrame(ar_normalize, columns = columns[1:])
df_normalize
```

Out[21]:

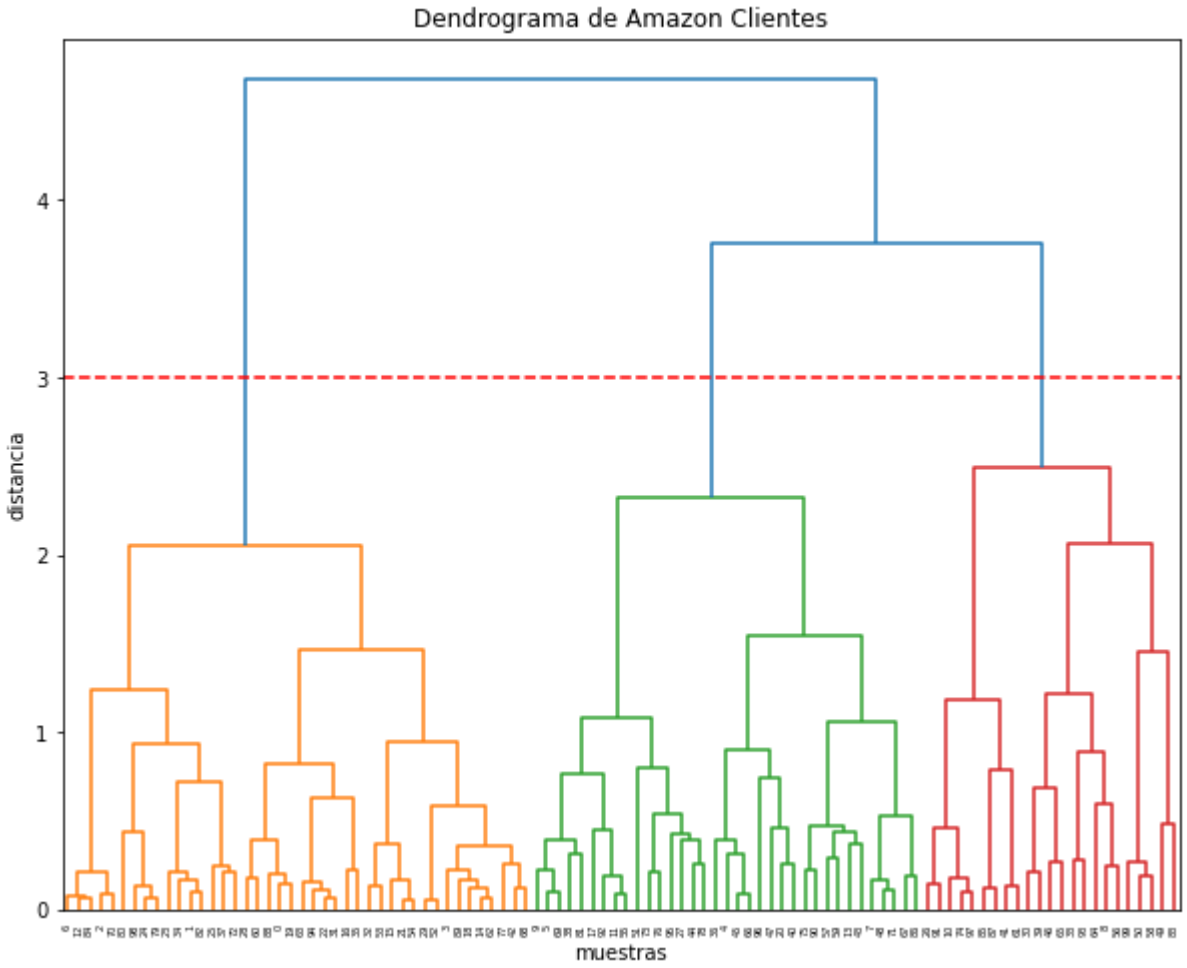
	Velocidad Entrega	Precio	Durabilidad	Imagen Producto	Valor Educativo	Servicio Retorno	Tamano Paquete	Calidad Producto	Numero Estrellas
0	0.438263	0.006414	0.737565	0.502399	0.051309	0.049171	0.055585	0.044895	0.036344
1	0.023235	0.038725	0.813234	0.085196	0.064542	0.010327	0.108431	0.555065	0.072287
2	0.057235	0.087535	0.959520	0.010100	0.144770	0.090902	0.138036	0.087535	0.111102
3	0.258856	0.009587	0.680696	0.565649	0.034514	0.044101	0.074781	0.373904	0.032597
4	0.011975	0.179625	0.191600	0.155675	0.135717	0.183617	0.898127	0.135717	0.171642
...	...	...	...	...	...	...	...	...	...
95	0.045932	0.122484	0.489936	0.382763	0.107174	0.321521	0.643041	0.260279	0.015311
96	0.640241	0.052479	0.096561	0.050380	0.069272	0.058776	0.745198	0.054578	0.094462
97	0.005043	0.070602	0.131118	0.126075	0.121032	0.136161	0.211805	0.932952	0.115989
98	0.357707	0.025386	0.773109	0.078465	0.060003	0.066926	0.096927	0.496175	0.062310
99	0.891358	0.064178	0.320889	0.178272	0.156879	0.021393	0.021393	0.156879	0.128356

100 rows × 9 columns

In [36]:

```
#Generar algoritmo de cluster jerarquicos
import matplotlib.pyplot as plt
import scipy.cluster.hierarchy as shc

plt.figure(figsize = (10,8))
plt.title('Dendrograma de Amazon Clientes')
plt.xlabel('muestras')
plt.ylabel('distancia')
dend = shc.dendrogram(shc.linkage(df_normalize, method ="ward"))
plt.axhline(y=3, color = "r", linestyle = "--") # Esta linea se coloca despues de graf
plt.show()
```



A partir de la visualización del dendrograma, se puede concluir que existen tres grupos o clusters en los que es posible categorizar a los clientes según sus calificaciones de productos. La linea punteada muestra el umbral de la distancia entre clusters para considerarse otro cluster.

In [34]:
```
# Imprimamos de una manera textual los cluster creados para corroborar la conclusión
Clusters = len(set(dend["color_list"]))-1
print(f'El numero de clusters mostrados en el dendograma son {Clusters}')
```

```
El numero de clusters mostrados en el dendograma son 3
```

In [40]:
```
#El siguiente paso es ahora si etiquetar los datos del data set, para eso tenemos que
from sklearn.cluster import AgglomerativeClustering
model_cluster = AgglomerativeClustering(n_clusters = 3, affinity = "euclidean", linkag
labels_data = model_cluster.fit_predict(df_normalize)
print("Estos son la categorizacion de los datos", labels_data)
```

```
Estos son la categorizacion de los datos [1 1 1 1 2 2 1 2 0 2 0 2 1 2 1 1 1 2 1 1 2 1
1 1 1 1 0 2 1 1 0 1 1 0 1 1 2
 1 2 0 2 0 1 2 2 2 0 2 2 0 0 2 1 1 1 2 0 2 0 2 1 0 1 1 0 0 2 2 1 2 1 2 1 2
 0 2 2 1 2 1 1 2 1 0 1 0 2 0 1 1 2 0 2 0 1 2 2 0 1 0]
```

Ahora ya con las etiquetas podemos integrarlo al Dataset normalizado

In [42]:
```
df_normalize["Cluster"] = labels_data
df_normalize
```

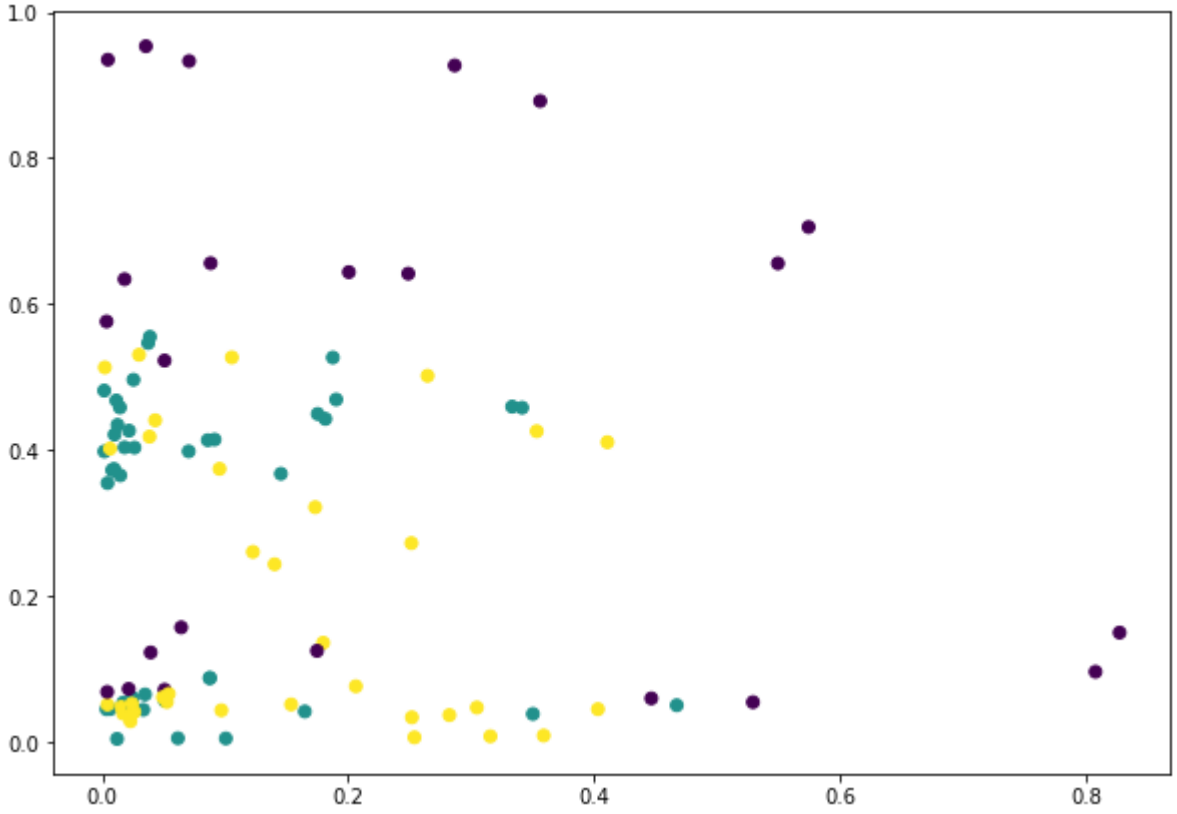
Out[42]:

	Velocidad Entrega	Precio	Durabilidad	Imagen Producto	Valor Educativo	Servicio Retorno	Tamano Paquete	Calidad Producto	Numero Estrellas	C
0	0.438263	0.006414	0.737565	0.502399	0.051309	0.049171	0.055585	0.044895	0.036344	
1	0.023235	0.038725	0.813234	0.085196	0.064542	0.010327	0.108431	0.555065	0.072287	
2	0.057235	0.087535	0.959520	0.010100	0.144770	0.090902	0.138036	0.087535	0.111102	
3	0.258856	0.009587	0.680696	0.565649	0.034514	0.044101	0.074781	0.373904	0.032597	
4	0.011975	0.179625	0.191600	0.155675	0.135717	0.183617	0.898127	0.135717	0.171642	
...	...	...	...	...	...	...	...	...	...	
95	0.045932	0.122484	0.489936	0.382763	0.107174	0.321521	0.643041	0.260279	0.015311	
96	0.640241	0.052479	0.096561	0.050380	0.069272	0.058776	0.745198	0.054578	0.094462	
97	0.005043	0.070602	0.131118	0.126075	0.121032	0.136161	0.211805	0.932952	0.115989	
98	0.357707	0.025386	0.773109	0.078465	0.060003	0.066926	0.096927	0.496175	0.062310	
99	0.891358	0.064178	0.320889	0.178272	0.156879	0.021393	0.021393	0.156879	0.128356	

100 rows × 10 columns

Podemos corroborar los cluster a traves de un grafico scatterplot, en este caso vamos a graficar solo la relacion que existe entre calidad de producto y precio

In [43]:
```
plt.figure(figsize = (10,7))
plt.scatter(df_normalize["Precio"],df_normalize["Calidad Producto"], c = df_normalize[
plt.show()
```



Al tener un data set con muchas dimensiones, puede se que la relacion entre las dimensiones que seleccionamos para graficar no siempre guarden una relacion es por eso que utilizaremos el metodo PCA para reducir las dimensiones del Dataset

In [47]:
```
#PCA
Valores_dataset = df_normalize.iloc[:,0:9].values
Valores_dataset
```

Out[47]:
```
array([[0.43826336, 0.00641361, 0.73756517, 0.50239946, 0.05130888,
        0.04917101, 0.05558462, 0.04489527, 0.03634379],
       [0.02323527, 0.03872544, 0.81323434, 0.08519598, 0.06454241,
        0.01032679, 0.10843125, 0.55506471, 0.0722875 ],
       [0.05723452, 0.08753514, 0.95951982, 0.01010021, 0.14476966,
        0.09090188, 0.13803618, 0.08753514, 0.11110229],
       [0.25885639, 0.00958727, 0.68069644, 0.56564916, 0.03451419,
        0.04410146, 0.07478074, 0.37390368, 0.03259673],
       [0.01197502, 0.17962533, 0.19160036, 0.15567529, 0.13571692,
        0.18361701, 0.89812667, 0.13571692, 0.17164198],
       [0.14494006, 0.251738  , 0.60264551, 0.03661644, 0.03966781,
        0.02898801, 0.73995715, 0.03356507, 0.00457705],
       [0.04755854, 0.02481315, 0.98218723, 0.06823617, 0.07237169,
        0.09304932, 0.07857498, 0.05996511, 0.06410064],
       [0.13323393, 0.04304481, 0.06354234, 0.52268696, 0.05739308,
        0.04509456, 0.70716471, 0.44069685, 0.05944283],
       [0.73804829, 0.0214705 , 0.12613916, 0.63069581, 0.09393342,
        0.00805144, 0.10198485, 0.07246292, 0.12882297],
       [0.00348086, 0.30457545, 0.56564012, 0.00522129, 0.06439595,
        0.0556938 , 0.75708755, 0.04699164, 0.06787681],
       [0.05320841, 0.03547227, 0.19509749, 0.10641681, 0.00886807,
        0.12415295, 0.12858698, 0.9533173 , 0.07537858],
       [0.30081705, 0.01696917, 0.70190646, 0.03548099, 0.00462795,
        0.03856629, 0.6402004 , 0.03856629, 0.04936485],
       [0.03421592, 0.01710796, 0.98981777, 0.0464359 , 0.05132388,
        0.03421592, 0.08065182, 0.05376788, 0.05865587],
       [0.37889397, 0.15360566, 0.08806725, 0.58370152, 0.05529804,
        0.07577879, 0.68610529, 0.05120189, 0.04710574],
       [0.33000964, 0.09127926, 0.69512668, 0.47043927, 0.00421289,
        0.0365117 , 0.04774608, 0.41426742, 0.05476756],
       [0.02878814, 0.00169342, 0.8213087 , 0.3979537 , 0.04572234,
        0.02878814, 0.04064208, 0.3979537 , 0.05757628],
       [0.03645394, 0.46706613, 0.64933583, 0.58098469, 0.08202137,
        0.06607277, 0.07062951, 0.05012417, 0.05240254],
       [0.41749646, 0.0153366 , 0.65606587, 0.36637445, 0.05793828,
        0.02556101, 0.50269982, 0.04771388, 0.04260168],
       [0.3782489 , 0.00999148, 0.69226685, 0.43534307, 0.04710269,
        0.05566682, 0.04853005, 0.42106953, 0.05566682],
       [0.36323349, 0.10046884, 0.76510885, 0.51780094, 0.00463702,
        0.04018754, 0.05255293, 0.00463702, 0.0061827 ],
       [0.386348  , 0.10536764, 0.10068463, 0.00468301, 0.04917156,
        0.04214705, 0.73757346, 0.52683818, 0.06087908],
       [0.03653752, 0.00429853, 0.89194526, 0.26865821, 0.02579119,
        0.03653752, 0.05588091, 0.35462884, 0.00429853],
       [0.02579636, 0.00343951, 0.78248953, 0.61051381, 0.0601915 ,
        0.05847175, 0.0722298 , 0.04471369, 0.00687903],
       [0.03042923, 0.19018267, 0.84948258, 0.06085845, 0.04817961,
        0.06339422, 0.09128768, 0.46911724, 0.05325115],
       [0.45206655, 0.01240967, 0.77117236, 0.04254744, 0.05850273,
        0.04609306, 0.03368339, 0.43433845, 0.06027554],
       [0.04106056, 0.18745039, 0.70517053, 0.05177201, 0.06069822,
        0.04998677, 0.41953184, 0.52664635, 0.06069822],
       [0.05696103, 0.35600642, 0.15664282, 0.11392205, 0.09018829,
        0.11866881, 0.17088308, 0.87814917, 0.0996818 ],
       [0.03874633, 0.09686582, 0.72276808, 0.45452426, 0.04768779,
        0.05811949, 0.49923156, 0.04321706, 0.05811949],
       [0.31413959, 0.02513117, 0.88856626, 0.31413959, 0.05564758,
        0.03051642, 0.04846725, 0.04846725, 0.06103283],
       [0.36927647, 0.3332495 , 0.53139785, 0.49537087, 0.0702526 ,
        0.00540405, 0.07565664, 0.4593439 , 0.05584181],
```

```
[0.04750309, 0.05066997, 0.00950062, 0.83922133, 0.09817306,
 0.00950062, 0.01266749, 0.52253404, 0.08867244],
[0.02461563, 0.03340693, 0.78242552, 0.60659956, 0.05802257,
 0.05626431, 0.07208864, 0.04395649, 0.06681386],
[0.04103525, 0.00157828, 0.73389965, 0.46559225, 0.05839632,
 0.03787869, 0.03630041, 0.48137504, 0.07102255],
[0.04861252, 0.52901858, 0.09150592, 0.81497458, 0.1000846 ,
 0.09722504, 0.12010152, 0.05433164, 0.09150592],
[0.02738021, 0.01140842, 0.87844842, 0.03878863, 0.03878863,
 0.02509853, 0.07073221, 0.46774526, 0.00456337],
[0.01909709, 0.35011336, 0.79571219, 0.47742731, 0.05304748,
 0.05092558, 0.08063217, 0.03819419, 0.05092558],
[0.03823337, 0.00424815, 0.0615982 , 0.0615982 , 0.07859081,
 0.0531019 , 0.98769534, 0.05097782, 0.0615982 ],
[0.00312254, 0.07025712, 0.71037754, 0.04215427, 0.03747046,
 0.040593  , 0.5698633 , 0.39812367, 0.04839935],
[0.        , 0.17312925, 0.56885325, 0.04451895, 0.01813735,
 0.0428701 , 0.73373825, 0.32152575, 0.0230839 ],
[0.0418933 , 0.00349111, 0.11171547, 0.78549943, 0.07331328,
 0.07680439, 0.15360878, 0.57603291, 0.04538441],
[0.21476784, 0.03843214, 0.08590714, 0.05199642, 0.05877857,
 0.05651785, 0.87037493, 0.41823211, 0.05651785],
[0.57760365, 0.08810903, 0.09398297, 0.07636116, 0.06657127,
 0.09006701, 0.44054515, 0.65592278, 0.08419307],
[0.37318997, 0.1751708 , 0.70829934, 0.34272549, 0.05483608,
 0.01980192, 0.04721996, 0.44935119, 0.05788253],
[0.07929518, 0.20616748, 0.13638772, 0.74537474, 0.09832603,
 0.07929518, 0.58678437, 0.07612338, 0.10466964],
[0.00176146, 0.022899  , 0.57247507, 0.32587042, 0.04227508,
 0.02994485, 0.74862124, 0.02818339, 0.04051362],
[0.07177865, 0.35889325, 0.13494386, 0.06603636, 0.1062324 ,
 0.04019604, 0.90441099, 0.00861344, 0.11197469],
[0.40592889, 0.24879512, 0.01309448, 0.58925161, 0.0680913 ,
 0.08380467, 0.04975902, 0.64162953, 0.01047558],
[0.03079523, 0.35323942, 0.05072156, 0.05072156, 0.06521343,
 0.04166414, 0.82422532, 0.42569879, 0.05072156],
[0.06073355, 0.00209426, 0.09214746, 0.47120858, 0.00628278,
 0.05026225, 0.70157722, 0.51309379, 0.08795893],
[0.21268415, 0.82710504, 0.03150876, 0.11815786, 0.29933325,
 0.11028067, 0.20480696, 0.14966663, 0.29933325],
[0.9228913 , 0.17460106, 0.20453267, 0.0149658 , 0.10476063,
 0.12471504, 0.12970364, 0.12471504, 0.12970364],
[0.13614992, 0.25135369, 0.42939589, 0.26182676, 0.37703053,
 0.26182676, 0.47128817, 0.27229983, 0.39797667],
[0.37464062, 0.34133923, 0.52449687, 0.49119548, 0.07159799,
 0.05661236, 0.07326305, 0.45789409, 0.05827743],
[0.03051268, 0.02615372, 0.73012479, 0.53397186, 0.05448692,
 0.0566664 , 0.10025594, 0.40320324, 0.03705111],
[0.03823712, 0.00804992, 0.87542873, 0.29180958, 0.03219968,
 0.04226208, 0.05634944, 0.37230877, 0.04829952],
[0.25536481, 0.02289478, 0.67803761, 0.06163978, 0.04931183,
 0.06340092, 0.67803761, 0.03698387, 0.05635637],
[0.56428699, 0.05067067, 0.08521885, 0.79460821, 0.10594776,
 0.00921285, 0.11055419, 0.07139958, 0.10825097],
[0.05482112, 0.25380148, 0.09745977, 0.55836326, 0.00812165,
 0.00609124, 0.78170857, 0.00609124, 0.01015206],
[0.94018329, 0.03935651, 0.16617193, 0.11806953, 0.13556131,
 0.10932364, 0.0962048 , 0.12244248, 0.13556131],
[0.20596919, 0.40298321, 0.00716415, 0.42089357, 0.0591042 ,
 0.0394028 , 0.77910087, 0.04477591, 0.06268628],
```

```
[0.26868493, 0.16467786, 0.85805831, 0.39002651, 0.04506973,
 0.05373699, 0.03293557, 0.04160283, 0.06760459],
[0.53817713, 0.20049736, 0.09708293, 0.06120446, 0.07597795,
 0.04854147, 0.47486217, 0.64370206, 0.09497243],
[0.3198163 , 0.08580437, 0.72543696, 0.42902186, 0.03900199,
 0.04212215, 0.05772294, 0.41342106, 0.04992254],
[0.04023907, 0.05096949, 0.73771626, 0.65723812, 0.09120856,
 0.06974772, 0.00804781, 0.0563347 , 0.0563347 ],
[0.22130333, 0.0040237 , 0.14485309, 0.94556878, 0.06437915,
 0.1287583 , 0.02011848, 0.06840285, 0.10059242],
[0.47875596, 0.01811509, 0.11645415, 0.58227076, 0.06728462,
 0.05952101, 0.08798758, 0.63402816, 0.00776361],
[0.05301222, 0.31554893, 0.11612201, 0.07825613, 0.08330492,
 0.09845127, 0.92140288, 0.00757317, 0.11107322],
[0.01460148, 0.41066668, 0.05840593, 0.48367409, 0.00547556,
 0.04562963, 0.64794077, 0.41066668, 0.05658074],
[0.45278576, 0.14523317, 0.72616585, 0.31609572, 0.05980189,
 0.03246388, 0.04100701, 0.36735449, 0.0734709 ],
[0.17544559, 0.28223856, 0.63312973, 0.03966596, 0.00457684,
 0.03508912, 0.69415429, 0.03661473, 0.05187087],
[0.05820916, 0.08731374, 0.95398343, 0.10024911, 0.1455229 ,
 0.09378142, 0.13582137, 0.08731374, 0.11318448],
[0.07616433, 0.0299217 , 0.11152634, 0.42162396, 0.01088062,
 0.04352247, 0.72084096, 0.53043014, 0.01088062],
[0.0296482 , 0.01811835, 0.81532558, 0.03953094, 0.04776655,
 0.03129533, 0.40354498, 0.40354498, 0.05929641],
[0.03814285, 0.09535713, 0.66749994, 0.33008239, 0.04841208,
 0.03960989, 0.53546698, 0.37409337, 0.06601648],
[0.06840051, 0.00456003, 0.15048113, 0.15048113, 0.10944082,
 0.12312092, 0.1869614 , 0.934807  , 0.11856089],
[0.04225181, 0.02414389, 0.09456358, 0.49293779, 0.06438371,
 0.05432376, 0.85509617, 0.05231177, 0.06840769],
[0.0299383 , 0.0063028 , 0.6539156 , 0.48058857, 0.03466541,
 0.04096821, 0.41756056, 0.40180356, 0.04254391],
[0.27590777, 0.02173819, 0.81100164, 0.27590777, 0.04849288,
 0.02508252, 0.04347638, 0.42640292, 0.05350939],
[0.00737441, 0.14011373, 0.52358287, 0.3318483 , 0.02212322,
 0.04572132, 0.73006626, 0.24335542, 0.03539715],
[0.40430304, 0.01437522, 0.78165255, 0.04132876, 0.05570397,
 0.03773495, 0.06109468, 0.45821011, 0.073673  ],
[0.44633507, 0.01460733, 0.70602093, 0.0308377 , 0.05842932,
 0.03408377, 0.39764397, 0.36518324, 0.07141361],
[0.03994652, 0.05404529, 0.64619364, 0.0963416 , 0.00939918,
 0.10339098, 0.74018544, 0.06579426, 0.07519344],
[0.02133038, 0.03732816, 0.81322057, 0.0853215 , 0.06132483,
 0.10131928, 0.10931818, 0.54659088, 0.06932372],
[0.50194895, 0.8074831 , 0.16586139, 0.10911934, 0.01309432,
 0.10911934, 0.16149662, 0.09602502, 0.09602502],
[0.03014465, 0.03478229, 0.98549814, 0.00695646, 0.06492694,
 0.06492694, 0.07883985, 0.06492694, 0.08811513],
[0.44300067, 0.54932083, 0.12404019, 0.07442411, 0.09923215,
 0.07796812, 0.15948024, 0.65564099, 0.09923215],
[0.02188841, 0.26448495, 0.07660943, 0.53809007, 0.04924892,
 0.04924892, 0.61105144, 0.50160939, 0.06566523],
[0.34454946, 0.5742491 , 0.12141267, 0.07875416, 0.09187986,
 0.07547274, 0.11813124, 0.70550604, 0.06890989],
[0.29097875, 0.0120405 , 0.73246376, 0.61205876, 0.0040135 ,
 0.05016875, 0.008027  , 0.0040135 , 0.03812825],
[0.31199059, 0.18138988, 0.67477034, 0.45710249, 0.04933805,
 0.00580448, 0.0536914 , 0.4425913 , 0.06530036],
```

```
[0.02799536, 0.026129  , 0.07278793, 0.66255678, 0.00559907,
 0.07092157, 0.73721106, 0.04105986, 0.06345614],
[0.08089801, 0.28651378, 0.12808851, 0.07078576, 0.11123476,
 0.0471905 , 0.09775176, 0.92695635, 0.08089801],
[0.36452026, 0.04938662, 0.59969463, 0.091718  , 0.08466277,
 0.00940697, 0.69376437, 0.06114533, 0.06584882],
[0.31413757, 0.44640602, 0.08266778, 0.81014426, 0.07274765,
 0.08266778, 0.13557516, 0.0595208 , 0.06944094],
[0.00491935, 0.06149192, 0.82399169, 0.55342725, 0.05411289,
 0.05165321, 0.06149192, 0.00491935, 0.03935483],
[0.04593152, 0.12248406, 0.48993623, 0.38276268, 0.10717355,
 0.32152065, 0.6430413 , 0.26027862, 0.01531051],
[0.64024082, 0.05247876, 0.09656091, 0.05037961, 0.06927196,
 0.05877621, 0.74519833, 0.05457791, 0.09446176],
[0.00504299, 0.0706018 , 0.13111762, 0.12607464, 0.12103165,
 0.13616061, 0.21180539, 0.93295232, 0.11598867],
[0.35770734, 0.02538568, 0.7731094 , 0.07846483, 0.06000252,
 0.06692589, 0.09692715, 0.49617469, 0.06231031],
[0.89135765, 0.06417775, 0.32088875, 0.17827153, 0.15687895,
 0.02139258, 0.02139258, 0.15687895, 0.1283555 ]])
```

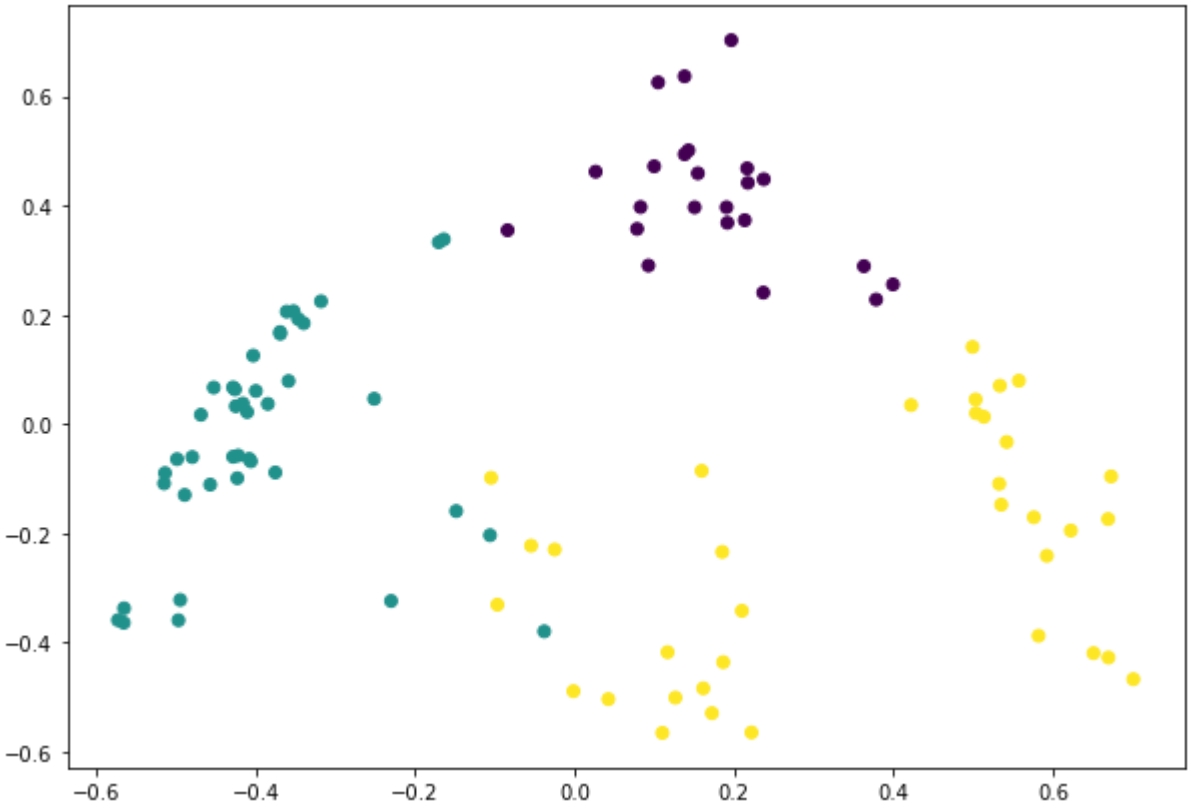
In [51]:
```
from sklearn import decomposition
pca = decomposition.PCA(n_components = 2)
pca.fit(Valores_dataset)
components_2= pca.transform(Valores_dataset)
data_frame_pca = pd.DataFrame(components_2, columns = ["PC1","PC2"])
data_frame_pca
```

Out[51]:

	PC1	PC2
0	-0.399932	0.060354
1	-0.408296	-0.064023
2	-0.497032	-0.359553
3	-0.339967	0.184345
4	0.582078	-0.387977
...	...	...
95	0.185277	-0.234391
96	0.535275	-0.147924
97	0.191641	0.368249
98	-0.384873	0.036584
99	-0.084130	0.354612

100 rows × 2 columns

In [53]:
```
plt.figure(figsize = (10,7))
plt.scatter(components_2[:,0],components_2[:,1], c = df_normalize["Cluster"])
plt.show()
```



Ahora si podemos ver claramente como se dividen los clusters, gracias a los componentes principales podemos representar los componentes principales del datases y con el cluster jerarquico podemos etiquetar los datos y obtener graficamente los clusters.

El siguiente paso es determinar que clientes se encuentran en el mismo cluster para de esta manera recomendarles los mismos productos.

In [55]:
```
#Agregamos los nombres al data set
df_original["Cluster"] = labels_data
df_original
```

Out[55]:

	Cliente	Velocidad Entrega	Precio	Durabilidad	Imagen Producto	Valor Educativo	Servicio Retorno	Tamano Paquete	Calidad Producto	Nume Estrel
0	Adam	205	3	345	235	24	23	26	21	
1	Anna	9	15	315	33	25	4	42	215	
2	Bernard	17	26	285	3	43	27	41	26	
3	Edward	135	5	355	295	18	23	39	195	
4	Emilia	3	45	48	39	34	46	225	34	
...	...	...	...	...	...	...	...	...	...	
95	Teofan	3	8	32	25	7	21	42	17	
96	Teofil	305	25	46	24	33	28	355	26	
97	Teofila	1	14	26	25	24	27	42	185	
98	Teon	155	11	335	34	26	29	42	215	
99	Teresa	125	9	45	25	22	3	3	22	

100 rows × 11 columns

In [82]:
```
#Segmentar Clientes por clusters
df_cluster1 = df_original.query("Cluster==0")
df_cluster2 = df_original.query("Cluster==1")
df_cluster3 = df_original.query("Cluster==2")
#Obtener los clientes unicos en cada cluster
clientes_1 = list(set(df_cluster1["Cliente"]))
clientes_2 = list(set(df_cluster2["Cliente"]))
clientes_3 = list(set(df_cluster3["Cliente"]))

clientes_diccionary = {f'clientes_{i}': cliente for i, cliente in enumerate([clientes_
```

In [83]:
```
choice = str(input("Escribe el nombre de un cliente: "))

for key, value in clientes_diccionary.items():
    if choice in value:
        print(f'El cliente pertenece al cluster {key}')
        print(f'y se le pueden recomendar los mismos productos que compraron: ')
        for cliente in clientes_diccionary[key] :
            print(cliente)
```

```
Escribe el nombre de un cliente: Xavier
El cliente pertenece al cluster clientes_0
y se le pueden recomendar los mismos productos que compraron:
Lesia
Xavier
Margaret
Marisol
Isadore
Leonid
Marianna
Tamara
Josephine
Maksym
Sandra
Teresa
Savina
Lawrence
Michaelina
Florent
Martha
Martin
Teofila
Sylvan
Herman
John
Sophia
```