

CASO DE ESTUDIO: Agrupando Clientes



Acerca del Caso:

Una empresa retail peruana con presencia física y online desea personalizar sus campañas de marketing para aumentar las ventas y fidelización de clientes.

Actualmente, trata a todos los clientes por igual, lo que genera:

- Baja conversión en promociones.
- Desperdicio de recursos en públicos no relevantes.
- Falta de entendimiento sobre los patrones de compra.

Problema Central:

¿Cómo agrupar a los clientes en segmentos homogéneos basados en su comportamiento de compra, para diseñar estrategias de marketing efectivas?

Objetivos de la organización

- Diseñar campañas personalizadas según clúster.
- Incentivar a los de **alto ingreso y bajo gasto** con ofertas exclusivas.
- Ofrecer descuentos inteligentes a los de **alto gasto y bajo ingreso**.
- Potenciar la fidelización del grupo premium.

Solución

Aplicar modelo basado en Machine Learning para Agrupar Cliente

- Tipo: No supervisado
- Tarea: Agrupamiento o Cluster
- Nombre: k-Means

Desarrollo del Modelo

Variables necesarias para el estudio

CustomerID	Genero	Edad	IngresosAnuales (k\$)	ScoreGasto (1-100)
1	Male	19	15	39
2	Male	21	15	81
3	Female	20	16	6
4	Female	23	16	77
5	Female	31	17	40
6	Female	22	17	76
7	Female	35	18	6
8	Female	23	18	94
9	Male	64	19	3
10	Female	30	19	72
11	Male	67	19	14
12	Female	35	19	99
13	Female	58	20	15

Luego de carga Jupyter, codificar

a) Importando Librerias (asegurarse de haberlas registrado previamente)

```
## Importando librerias
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd # procesar datos
from sklearn.cluster import KMeans # Algoritmo de Agrupamiento
import seaborn as sns
import plotly as py
import plotly.graph_objs as go
import warnings
import os
```

b) Leyendo datos

```
## Leyendo Datos
df = pd.read_csv('c:/Otros/Grupo_Clientes.csv')
df.head()
```

	CustomerID	Genero	Edad	IngresosAnuales (k\$)	ScoreGasto (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6

c) Datos descriptivos

```
: df.describe()
```

	CustomerID	Edad	IngresosAnuales (k\$)	ScoreGasto (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000

Note los valores promedios

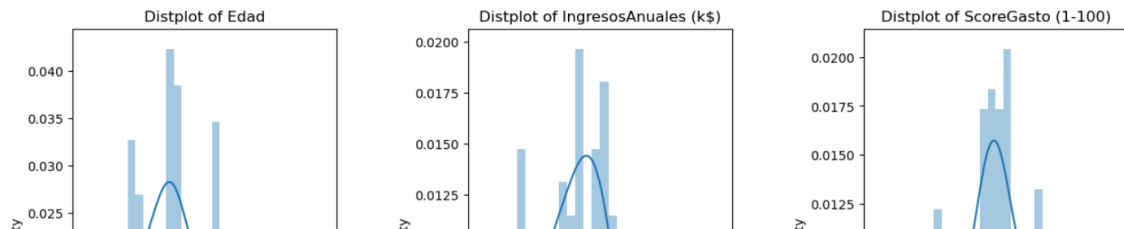
d) Calidad de data (Nulos)

```
# verificando calidad de data. Determinando si hay nulos
df.isnull().sum()
```

```
CustomerID      0
Genero          0
Edad            0
IngresosAnuales (k$)  0
ScoreGasto (1-100)  0
dtype: int64
```

e) Analizando datos por variable

```
## Analizando datos
plt.figure(1, figsize = (15 , 6))
n = 0
for x in ['Edad' , 'IngresosAnuales (k$)' , 'ScoreGasto (1-100)']:
    n += 1
    plt.subplot(1 , 3 , n)
    plt.subplots_adjust(hspace=0.5 , wspace = 0.5)
    sns.distplot(df[x] , bins = 20)
    plt.title('Distplot of {}'.format(x))
plt.show()
```

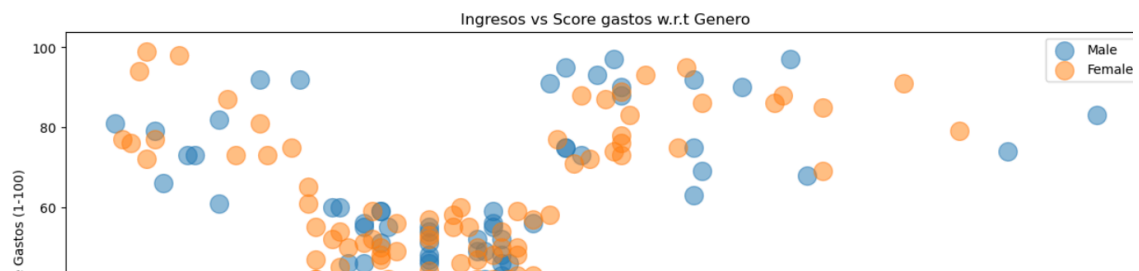


```
# Analizando el género
plt.figure(1, figsize = (15 , 5))
sns.countplot(y = 'Genero' , data = df)
plt.show()
```



f) Análisis de variables en conjunto

```
# Análisis de Variables. Note donde los puntos se entrecruzan
plt.figure(1, figsize = (15 , 6))
for gender in ['Male' , 'Female']:
    plt.scatter(x = 'IngresosAnuales (k$)', y = 'ScoreGasto (1-100)' ,
                data = df[df['Genero'] == gender] , s = 200 , alpha = 0.5 , label = gender)
plt.xlabel('Ingresos Anuales (k$)', plt.ylabel('Score de Gastos (1-100)')
plt.title('Ingresos vs Score gastos w.r.t Genero')
plt.legend()
plt.show()
```



g) Encontrando número de clusters

```

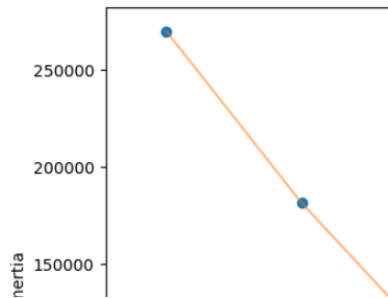
•[34]: 'Ingresos Anuales and Score de Gasto'''
X2 = df[['IngresosAnuales (k$)', 'ScoreGasto (1-100)']].iloc[:, :].values
inertia = []
for n in range(1, 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001, random_state= 111 , algorithm='elkan') )
    algorithm.fit(X2)
    inertia.append(algorithm.inertia_)

```

```

•[36]: # Note que a más cluster no hay variación. 5 puede ser una cantidad adecuada
plt.figure(1, figsize = (15,6))
plt.plot(np.arange(1, 11), inertia, 'o')
plt.plot(np.arange(1, 11), inertia, '-', alpha = 0.5)
plt.xlabel('Número de Clusters') , plt.ylabel('Inertia')
plt.show()

```



h) Aplicando clusters de acuerdo a análisis anterior

```

: # Se trabaja con 5 cluster y se aplica el Algoritmo de ML Kmeans
algorithm = (KMeans(n_clusters = 5 ,init='k-means++', n_init = 10 ,max_iter=300,
                    tol=0.0001, random_state= 111 , algorithm='elkan') )
algorithm.fit(X2)
labels2 = algorithm.labels_ # asigna grupos
centroids2 = algorithm.cluster_centers_

```

i) Datos para graficas clusters determinados por modelo

```

h = 0.02
x_min, x_max = X2[:, 0].min() - 1, X2[:, 0].max() + 1
y_min, y_max = X2[:, 1].min() - 1, X2[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z2 = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])

# Graficando grupos
plt.figure(1, figsize = (15, 7))
plt.clf()
Z2 = Z2.reshape(xx.shape)
plt.imshow(Z2, interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'IngresosAnuales (k$)', y = 'ScoreGasto (1-100)', data = df, c = labels2, s = 200 )
plt.scatter(x = centroids2[:, 0], y = centroids2[:, 1], s = 300, c = 'red', alpha = 0.5)
plt.ylabel('Score de Gasto (1-100)') , plt.xlabel('Ingresos Anuales(k$)')
plt.show()

```

j) Asignando modelo a cada cliente

```
[50]: df['Grupo'] = labels2
```

```
•[52]: ## Verifique el grupo creado y asignado a cada cliente.
df.head()
```

```
[52]:
```

	CustomerID	Genero	Edad	IngresosAnuales (k\$)	ScoreGasto (1-100)	Grupo
0	1	Male	19	15	39	3
1	2	Male	21	15	81	1
2	3	Female	20	16	6	3

CODIGO COMPLETO

```
## Importando librerias
```

```
import matplotlib.pyplot as plt
```

```
import numpy as np
```

```
import pandas as pd # procesar datos
```

```
from sklearn.cluster import KMeans # Algoritmo de Agrupamiento
```

```
import seaborn as sns
```

```
import plotly as py
```

Dr. Ricardo Mendoza Rivera

```

import plotly.graph_objs as go

import warnings

import os

warnings.filterwarnings("ignore")

py.offline.init_notebook_mode(connected = True)


## Leyendo Datos

df = pd.read_csv('c:/Otros/Grupo_Clientes.csv')

df.head()


df.describe()


# verificando calidad de data. Determinando si hay nulos

df.isnull().sum()


## Analizando datos

plt.figure(1 , figsize = (15 , 6))

n = 0

for x in ['Edad' , 'IngresosAnuales (k$)' , 'ScoreGasto (1-100)']:

    n += 1

    plt.subplot(1 , 3 , n)

    plt.subplots_adjust(hspace = 0.5 , wspace = 0.5)

    sns.distplot(df[x] , bins = 20)

```

```
plt.title('Distplot of {}'.format(x))  
plt.show()
```

```
# Analizando el género  
  
plt.figure(1, figsize = (15, 5))  
  
sns.countplot(y = 'Genero', data = df)  
  
plt.show()
```

```
# Análisis de Variables. Note donde los puntos se entrecruzan  
  
plt.figure(1, figsize = (15, 6))  
  
for gender in ['Male', 'Female']:  
  
    plt.scatter(x = 'IngresosAnuales (k$)', y = 'ScoreGasto (1-100)',  
                data = df[df['Genero'] == gender], s = 200, alpha = 0.5, label = gender)  
  
plt.xlabel('Ingresos Anuales (k$)', plt.ylabel('Score de Gastos (1-100)')  
  
plt.title('Ingresos vs Score gastos w.r.t Genero')  
  
plt.legend()  
  
plt.show()
```

"Ingresos Anuales and Score de Gasto"

```
X2 = df[['IngresosAnuales (k$)', 'ScoreGasto (1-100)']].iloc[:, :].values
```

```
inertia = []
```

```
for n in range(1, 11):
```



```

algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                    tol=0.0001, random_state= 111 , algorithm='elkan') )

algorithm.fit(X2)

inertia.append(algorithm.inertia_)

# Note que a más cluster no hay variación. 5 puede ser una cantidad adecuada

plt.figure(1 , figsize = (15 ,6))

plt.plot(np.arange(1 , 11) , inertia , 'o')

plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)

plt.xlabel('Número de Clusters') , plt.ylabel('Inertia')

plt.show()

# Se trabaja con 5 cluster y se aplica el Algoritmo de ML Kmeans

algorithm = (KMeans(n_clusters = 5 ,init='k-means++', n_init = 10 ,max_iter=300,
                    tol=0.0001, random_state= 111 , algorithm='elkan') )

algorithm.fit(X2)

labels2 = algorithm.labels_   # asigna grupos

centroids2 = algorithm.cluster_centers_

h = 0.02

x_min, x_max = X2[:, 0].min() - 1, X2[:, 0].max() + 1

y_min, y_max = X2[:, 1].min() - 1, X2[:, 1].max() + 1

xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))

```

```
Z2 = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
```

```
# Graficando grupos
```

```
plt.figure(1, figsize = (15, 7))
```

```
plt.clf()
```

```
Z2 = Z2.reshape(xx.shape)
```

```
plt.imshow(Z2, interpolation='nearest',
```

```
          extent=(xx.min(), xx.max(), yy.min(), yy.max()),
```

```
          cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')
```

```
plt.scatter( x = 'IngresosAnuales (k$)', y = 'ScoreGasto (1-100)', data = df, c = labels2 ,  
            s = 200 )
```

```
plt.scatter(x = centroids2[:, 0], y = centroids2[:, 1], s = 300, c = 'red', alpha = 0.5)
```

```
plt.ylabel('Score de Gasto (1-100)') , plt.xlabel('Ingresos Anuales(k$)')
```

```
plt.show()
```

```
df['Grupo'] = labels2
```

```
## Verifique el grupo creado y asignado a cada cliente.
```

```
df.head()
```

```
df["Grupo"] = 5
```

```
df.head()
```

Aplicar caso de Segmentación con de Algortimo no Supervisado k-means, realizando 3 análisis previos a la aplicación del algoritmo. Incluir un análisis adicional con Power BI para cada segmento, presentando conclusiones en la tabla siguiente

Segmento	Conclusión