Setup

```
#Liberias generales
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

#Liberias preprocesamiento
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

#Librerias modelos
from sklearn.ensemble import IsolationForest
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
```

```
data = pd.read_csv('https://raw.githubusercontent.com/Rwyld/Data-Science-Models/main/Modelos/KMeans/MktCampaignCSV%20-%20KMeans-PCA.csv')
data.head()
```

	ID	Income	Recency	MntWines	MntMeatProducts	MntFishProducts	MntSweetProc
0	5524	58138.0	58	635	546	172	
1	2174	46344.0	38	11	6	2	
2	4141	71613.0	26	426	127	111	
3	6182	26646.0	26	11	20	10	
4	5324	58293.0	94	173	118	46	

Analisis Exploratorio

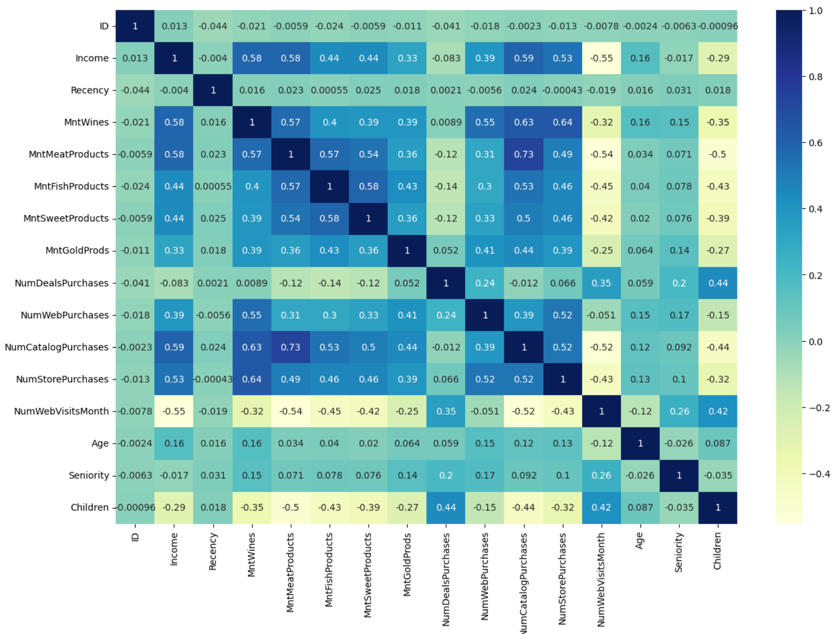
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2216 entries, 0 to 2215
Data columns (total 16 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   ID                   2216 non-null   int64  
 1   Income               2216 non-null   float64
 2   Recency              2216 non-null   int64  
 3   MntWines             2216 non-null   int64  
 4   MntMeatProducts      2216 non-null   int64  
 5   MntFishProducts      2216 non-null   int64  
 6   MntSweetProducts     2216 non-null   int64  
 7   MntGoldProds         2216 non-null   int64  
 8   NumDealsPurchases    2216 non-null   int64  
 9   NumWebPurchases      2216 non-null   int64  
 10  NumCatalogPurchases  2216 non-null   int64  
 11  NumStorePurchases    2216 non-null   int64  
 12  NumWebVisitsMonth    2216 non-null   int64  
 13  Age                  2216 non-null   int64  
 14  Seniority            2216 non-null   int64  
 15  Children             2216 non-null   int64  
dtypes: float64(1), int64(15)
memory usage: 277.1 KB
```

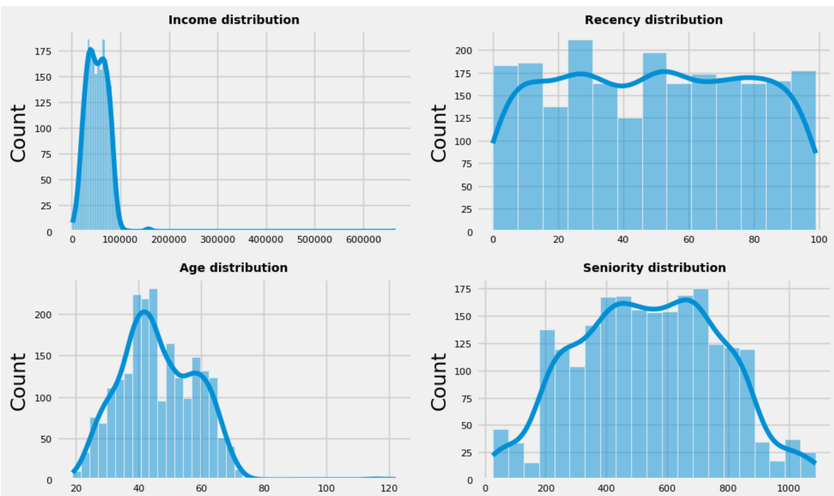
```
data.describe()
```

	ID	Income	Recency	MntWines	MntMeatProducts	MntFi
count	2216.000000	2216.000000	2216.000000	2216.000000	2216.000000	2
mean	5588.353339	52247.251354	49.012635	305.091606	166.995939	
std	3249.376275	25173.076661	28.948352	337.327920	224.283273	
min	0.000000	1730.000000	0.000000	0.000000	0.000000	
25%	2814.750000	35303.000000	24.000000	24.000000	16.000000	
50%	5458.500000	51381.500000	49.000000	174.500000	68.000000	
75%	8421.750000	68522.000000	74.000000	505.000000	232.250000	

```
corr = data.corr()

fig, ax = plt.subplots(1,1, figsize = (15,10))

ax = sns.heatmap(corr, annot=True, cmap="YlGnBu")
```



```
fig, ax = plt.subplots(2, 2, figsize = (10,6))

sns.histplot(data = data, x = data.Income, stat = 'count', kde = True, alpha = 0.5, ax = ax[0][0])
ax[0][0].set_title('Income distribution', fontsize = 10, fontweight = "bold")
ax[0][0].tick_params(labelsize = 8)
ax[0][0].set_xlabel("")

sns.histplot(data = data, x = data.Recency, stat = 'count', kde = True, alpha = 0.5, ax = ax[0][1])
ax[0][1].set_title('Recency distribution', fontsize = 10, fontweight = "bold")
ax[0][1].tick_params(labelsize = 8)
ax[0][1].set_xlabel("")

sns.histplot(data = data, x = data.Age, stat = 'count', kde = True, alpha = 0.5, ax = ax[1][0])
ax[1][0].set_title('Age distribution', fontsize = 10, fontweight = "bold")
ax[1][0].tick_params(labelsize = 8)
ax[1][0].set_xlabel("")

sns.histplot(data = data, x = data.Seniority, stat = 'count', kde = True, alpha = 0.5, ax = ax[1][1])
ax[1][1].set_title('Seniority distribution', fontsize = 10, fontweight = "bold")
ax[1][1].tick_params(labelsize = 8)
ax[1][1].set_xlabel("")

plt.tight_layout()
plt.show()
```



Detectando Anomalias

```
dataPurchase = data[['NumDealsPurchases',  'NumWebPurchases',  'NumCatalogPurchases',  'NumStorePurchases', 'Age', 'Income']]

pca = PCA(n_components=2)
checkData = pca.fit_transform(dataPurchase)

isofor = IsolationForest(n_estimators=100, max_samples='auto', contamination=0.05, random_state=42)
isofor.fit(checkData)

y_pred = isofor.predict(checkData)
anomalies = dataPurchase[y_pred == -1]

print("Anomalías detectadas")
display(anomalies.head(3))
```

Anomalías detectadas

	NumDealsPurchases	NumWebPurchases	NumCatalogPurchases	NumStorePurchases	Ag
9	1	1	0	0	6
16	2	4	1	6	6
20	15	0	28	0	3

```
print("\nNueva base de datos sin anomalias")
newData = data[y_pred == 1]
newData.head(3)
```

Nueva base de datos sin anomalias

	ID	Income	Recency	MntWines	MntMeatProducts	MntFishProducts	MntSweetProc
0	5524	58138.0	58	635	546	172	
1	2174	46344.0	38	11	6	2	
2	4141	71613.0	26	426	127	111	

```
newData.describe()
```

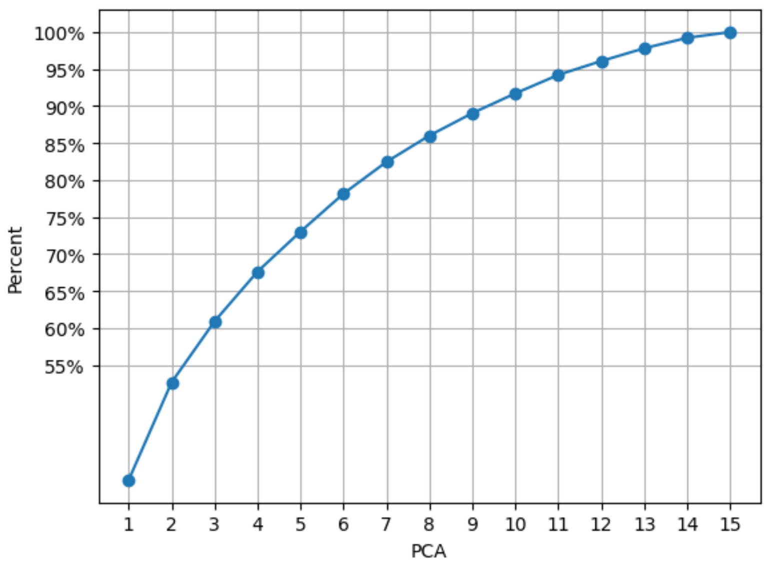
	ID	Income	Recency	MntWines	MntMeatProducts	MntFis
count	2106.000000	2106.000000	2106.00000	2106.000000	2106.000000	2
mean	5583.579772	51468.495252	49.15812	300.892213	158.948718	
std	3257.540072	19814.442832	29.01868	332.459088	208.359465	
min	0.000000	7500.000000	0.00000	0.000000	0.000000	
25%	2793.500000	35651.250000	24.00000	24.250000	16.000000	
50%	5501.500000	51075.000000	49.00000	173.500000	66.500000	
75%	8431.500000	67442.000000	74.00000	494.750000	220.500000	
max	11191.000000	102692.000000	99.00000	1493.000000	984.000000	2

Reduciendo data significativa

```
newData = newData.drop('ID', axis=1)
```

```
modelData = StandardScaler().fit_transform(newData)
pcaModel = PCA().fit(modelData)
```

```
plt.plot(
    range(1, len(pcaModel.components_) + 1),
    np.cumsum(pcaModel.explained_variance_ratio_),
    marker = "o"
)
plt.xticks(
    ticks = np.arange(newData.shape[1]) + 1,
)
plt.yticks(
    ticks  = np.linspace(0.55, 1, 10),
    labels = [f"{val:0.0%}" for val in np.linspace(0.55, 1, 10)]
);

plt.ylabel('Percent')
plt.xlabel('PCA')
plt.grid()
```



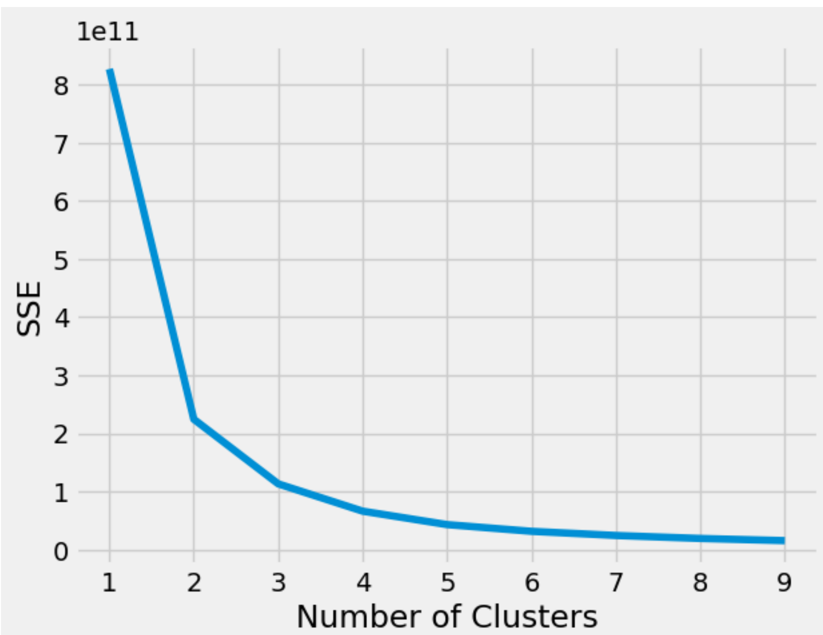

```
pca = PCA(n_components=10)
pcaData = pca.fit_transform(newData)

scaler = StandardScaler()
scalerData = scaler.fit_transform(pcaData)

unscalerData = scaler.inverse_transform(scalerData)
originalData = pca.inverse_transform(unscalerData)

newModelData = pd.DataFrame(originalData, columns = newData.columns)
newModelData.head(3)
```

	Income	Recency	MntWines	MntMeatProducts	MntFishProducts	MntSweetP
0	58137.999760	58.003654	635.009772	546.003264	171.991888	87
1	46343.999886	38.000726	10.999054	5.996289	1.989249	(
2	71613.000092	25.996878	425.999282	127.000776	111.007611	21

Modelando Clusters

```
kmeans_kwargs = {
    "init": "random",
    "n_init": 10,
    "max_iter": 300,
    "random_state": 1234,
}

# Una lista contiene los valores de SSE para cada k
sse = []
for k in range(1, 10):
    kmeans = KMeans(n_clusters=k, **kmeans_kwargs) # el operador (**) es de desempaquetado del diccionario de Python
    kmeans.fit(newModelData)
    sse.append(kmeans.inertia_)
```

```
plt.style.use("fivethirtyeight")
plt.plot(range(1, 10), sse)
plt.xticks(range(1, 10))
plt.xlabel("Number of Clusters")
plt.ylabel("SSE")
plt.show()
```




```
kmeans = KMeans(
    init="random",
    n_clusters=3,
    n_init=10,
    max_iter=300,
    random_state=1234
)
kmeans.fit(newModelData)

centroids = pd.DataFrame(kmeans.cluster_centers_, columns=newModelData.columns).sort_values(by = ['Income'], ascending = True).reset_index().
```

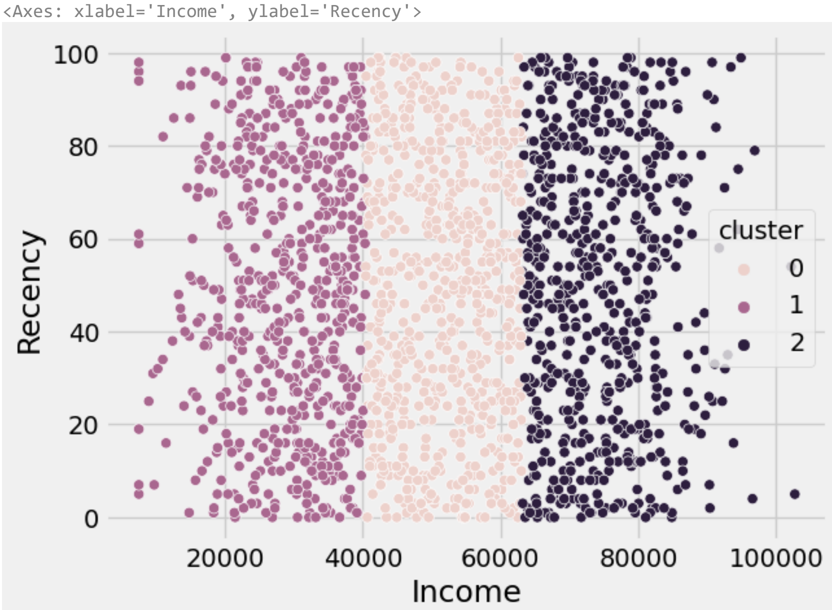
```
clusterData = pd.DataFrame(newModelData)
clusterData['cluster'] = kmeans.labels_
clusterData[clusterData['cluster'] == 1].head(3)
```

	Income	Recency	MntWines	MntMeatProducts	MntFishProducts	MntSweetPr
3	26646.000039	25.999822	10.999678	20.000800	10.002961	3.
7	33453.999906	31.998313	76.000075	55.996994	2.993640	0.
8	30350.999808	18.997455	14.001814	23.995691	2.989012	2.

Visualizacion Clusters

```
sns.scatterplot(x=clusterData.Income, y=clusterData.Recency, data=clusterData, hue = clusterData.cluster)
```

<Axes: xlabel='Income', ylabel='Recency'>




```
display(centroids)
```

	Income	Recency	MntWines	MntMeatProducts	MntFishProducts	MntSweetP
0	29112.412763	48.693443	31.273878	23.865221	8.855411	6
1	51642.767940	49.185319	268.328349	89.583649	23.674924	16
2	74562.830152	49.612926	616.486250	373.799789	79.750798	58

Interpretando

Los centroides nos indican el punto medio o el centro de cada cluster.

Por ejemplo, tomando en cuenta la variable Income:

- En el cluster 0 su centroide es de 29112
- En el cluster 1 su centroide es de 51642
- En el cluster 2 su centroide es de 74562

Y asi se observa para cada variable su respectivo centroide en su correspondiente Cluster. Por lo tanto, los centroides nos ayudan a predecir nuevos datos cercanos a los cluster mediante el calculo de la distancia euclidiana entre ellos.