



PROCESO GESTIÓN DEL TALENTO HUMANO
FORMATO INFORME MENSUAL EJECUCIÓN CONTRACTUAL

Medellín, 30 de junio del 2022

Señor (a)
John Jairo Castro Maldonado
SUPERVISOR(A) CONTRATO No. CO1.PCCNTR.3367636
Instructor Grado 16
Centro de Servicios y Gestión Empresarial
Medellín

Asunto: Informe mensual de ejecución contractual de junio del año 2022.

Referencia: No 3367636 del año 2022

Jesus Arbey Benavides Guevara, identificado con la cédula de ciudadanía No. 1.085.260.713 de Pasto, en mi calidad de Contratista del SENA, en Centro de Servicios y Gestión Empresarial, en cumplimiento del Contrato de Prestación de Servicios de la referencia, a continuación presento el Informe de actividades realizadas en el mes objeto de cobro.

Valor y forma de Pago: Se fija como valor total para el contrato la suma de TREINTA Y CUATRO MILLONES TRESCIENTOS MIL PESOS MCTE (\$ 34.300.000) Incluido IVA. Esta suma será pagada por el SENA al contratista de la siguiente manera: Valor por mes: CUATRO MILLONES NOVECIENTOS MIL PESOS MCTE (\$ 4.900.000), y el equivalente por fracción de mes.

Plazo: Será hasta el treinta (30) de agosto (08) de 2022.

OBJETO: Prestar servicios profesionales para el desarrollo de las actividades de levantamiento de indicadores, bases de datos y el modelo matemático con base en analítica de datos y Big Data en el prototipo de software para modelo de Innovación en el desarrollo del proyecto de I+D+i SGPS 9748

Obligaciones Específicas:

No	Obligaciones	Acciones realizadas	Evidencias
1	Apoyar el desarrollo de aplicaciones o algoritmos usados en la actualidad para solucionar de forma eficiente y eficaz problemas o	Se implementó modelos de Kmeans y PCA para la determinación de conglomerados y en la clasificación de centros de formación en torno a la dimensión su productividad y gestión de	ANEXO 3. Modelos de clusterización y clasificación, supervisados y no supervisados para las dimensiones de productividad y procesos de centros de formación en la regional Antioquia. ANEXOS 4. Presentación de avances técnicos junio 2022. ANEXOS 5. Actas de reuniones



	retos identificados en Big Data y analítica de datos con aplicación en gestión de la innovación	<p>procesos con datos aleatorios.</p> <p>Se realizó mapas geoespaciales de Antioquia, donde se visualizó los centros de formación donde, posterior a la etapa de clusterización se realizó una clasificación de alta, media y baja productividad y gestión de manera respectiva, y que se visualiza como una alerta para cada centro.</p> <p>Se generan modelos predictivos usando modelos de machine learning supervisado: perceptrón; Random Forest, Support Vector Machine y Regresión logística. Posteriormente se realiza una etapa de validación de datos y medidas de prueba para medir el poder predictivo del modelo, determinando que el mejor modelo fue el generado por el Random Forest.</p>	<p>ANEXO 3. Modelos de clusterización y clasificación, supervisados y no supervisados para las dimensiones de productividad y procesos de centros de formación en la regional Antioquia.</p> <p>ANEXOS 4. Presentación de avances técnicos junio 2022.</p> <p>ANEXOS 5. Actas de reuniones</p> <p>ANEXO 3. Modelos de clusterización y clasificación, supervisados y no supervisados para las dimensiones de productividad y procesos de centros de formación en la regional Antioquia.</p> <p>ANEXOS 4. Presentación de avances técnicos junio 2022.</p> <p>ANEXOS 5. Actas de reuniones</p>
2	Revisar las herramientas tecnológicas o digitales de Big Data y analítica de datos necesarias para dar	Se implementó diferentes tipos de herramientas ofrecidas por la librería scikit-learn de python, de software libre para la implementación de	ANEXO 3. Modelos de clusterización y clasificación, supervisados y no supervisados para las dimensiones de productividad y procesos de centros de formación en la regional Antioquia.



	cumplimiento a los objetivos propuestos dentro del proyecto.	modelos supervisados como no supervisados.	
3	Revisar y actualizar indicadores para la gestión de la innovación según objetivos del proyecto	Se implementó, por medio de modelos de clusterización y caracterización (KMeans, Cluster Jerárquico y Métodos de Componentes Principales), dos dimensiones en términos de indicadores para la gestión de la innovación - proyecto SGPS 9748 prototipo innovación a lo largo del tiempo: procesos y productividad.	ANEXO 3. Modelos de clusterización y clasificación, supervisados y no supervisados para las dimensiones de productividad y procesos de centros de formación en la regional Antioquia. ANEXOS 4. Presentación de avances técnicos junio 2022. ANEXOS 5. Actas de reuniones
4	Realizar pruebas relacionadas con la compatibilidad de la tecnología y la aplicación de los sistemas de Big Data y analítica de datos para la evaluación y valoración de la Gestión de CTel para los Centros SENA Regional Antioquia	Se organizaron modelos de machine learning, desde la perspectiva geoespacial, para posterior visualización de los centros de formación. Se inició el acople entre los algoritmos generados desde python hacia el interfaz de programación de aplicaciones.	ANEXO 3. Modelos de clusterización y clasificación, supervisados y no supervisados para las dimensiones de productividad y procesos de centros de formación en la regional Antioquia. ANEXOS 5. Actas de reuniones
5	Apoyar la elaboración del manual del sistema y usuario y las actividades de transferencia	Gestión por realizar	Gestión por realizar
6	Presentar los informes de seguimiento del proyecto de acuerdo con las solicitudes del líder	Se elaboró el informe de pago mensual GTH-F-062 V09 Se elaboró diapositivas en presentación de informes	Informe en formato pdf. ANEXOS 4. Presentación de avances técnicos junio 2022.



	del área o supervisor	técnicos para el supervisor y coordinador del contrato del mes en curso.	
7	Divulgar en eventos académicos/empresariales los resultados del proyecto.	Se participó de la visita de la Universidad Digital de Antioquia desde la facultad de ingeniería y la vicerrectoría de académica al centro de servicios y gestión empresarial - laboratorio de inteligencia artificial.	ANEXO 1. Medios de divulgación: Visita Universidad Digital de Antioquia.
8	Redactar el informe de investigación y de resultados del proyecto.	Se elaboró un (1) informe de pago mensual GTH-F-062 V09. Se elaboró diapositivas en presentación de informes técnicos para el supervisor y coordinador del contrato del mes en curso.	Informe en formato pdf. ANEXOS 4. Presentación de avances técnicos junio 2022.
9	Redactar y someter a publicación artículo de investigación, libro de investigación o capítulo de libro de investigación relacionado con el proyecto, en medio reconocido por Minciencias.	Se mostró avances de la elaboración del artículo científico orientado en la visualización de mapas para actividades de ciencia tecnología e innovación.	ANEXOS 2. Avances de redacción de artículo científico orientado en la visualización de mapas para actividades de ciencia tecnología e innovación.
10	Apoyar actividades de Grupo de Investigación, Semilleros de investigación y SENNOVA que sean propias de su área de conocimiento y en relación con el objeto contractual	Se apoyó en la orientación de la visita por parte de la Universidad Digital de Antioquia, y en las posibles colaboraciones entre la facultad de ingeniería de esta universidad y el Centro De Servicios y Gestión Empresarial en temas de ciencia de datos.	ANEXO 1. Medios de divulgación: Visita Universidad Digital de Antioquia.



11	Apoyar la evaluación técnica, administrativa y presupuestal de los proyectos de investigación aplicada, desarrollo tecnológico e innovación que se formulen por el SENA para el desarrollo de actividades misionales, acorde con las indicaciones dadas por la Dirección General.	Gestión por realizar	Gestión por realizar
12	Desarrollar sus actuaciones en el marco de la normatividad vigente y presentar los informes mensuales de acuerdo con el objeto del presente contrato.	Se elaboró un (1) informe de pago mensual GTH-F-062 V09.	Informe en formato pdf.

A continuación relaciono los desplazamientos que realicé previo a la presentación de este informe. Una vez finalizado cada desplazamiento presenté al ordenador del gasto el informe en el Formato Informe Legalización Desplazamiento Contratista GTH-F-087, en el que se describieron las actividades desarrolladas y los resultados de cada desplazamiento. Cada informe cuenta con el visto bueno del Supervisor.

Se lista a continuación el soporte de la legalización de los desplazamientos realizados, los cuales forman parte integral del presente informe de ejecución contractual.

ITEM	No DE LA ORDEN DE VIAJE	LUGAR DE DESPLAZAMIENTO	FECHAS DE DESPLAZAMIENTOS	
			INICIO	FINALIZACIÓN
1.		No he realizado ningún desplazamiento		



Nota 1: Por cada desplazamiento que haya realizado el contratista, adjuntará el respectivo informe que la soporte. En caso de haber realizado el desplazamiento en fecha posterior a la presentación del informe de ejecución contractual, deberá reportarlo en el siguiente informe de ejecución contractual.

Para el trámite de la cuenta me permito adjuntar: Documentos electrónicos enunciados como evidencias del cumplimiento de las obligaciones contractuales y los desplazamientos realizados y el No.59345436 de la planilla, miplanilla correspondiente al mes de mayo del año 2022. (Decreto Ley 2106 de 2019 – “Decreto Ley Antitrámites”)

Declaración bajo la gravedad de juramento.

Me permito declarar **bajo la gravedad de juramento** que durante el periodo objeto de cobro SI___NO_X_ he suscrito otro u otros contratos con entidad(es) del Estado el (los) cual(es) se encuentra(n) en ejecución; la anterior declaración de acuerdo con lo establecido en el numeral 5º de la Directiva Presidencial No. 001 de 2022, a fin de identificar potenciales conflictos de interés y hacer la gestión del riesgos para la toma de decisiones dentro del principio de la debida diligencia frente a la ejecución de los contratos de prestación de servicios profesionales o de apoyo a la gestión.

Nota: En el evento de señalar **SI**, el análisis efectuado por el supervisor y ordenador de pago deberá quedar documentado y publicado en el SECOP II en los documentos relacionados en el archivo Gestión Contractual GC_ del periodo objeto de cobro.

Evidencias en 283 folios:

- ANEXOS.
- <https://drive.google.com/drive/folders/1j61SX1uD8L8vP3UX7BeYt0NUtED3ZxwB?usp=sharing>

Cordialmente,

JESUS ARBEY BENAVIDES GUEVARA
Contratista
C.C. No. 1.085.260.713 DE PASTO (NARIÑO)



Recibí a satisfacción:

JOHN JAIRO CASTRO MALDONADO
SUPERVISOR(A) CONTRATO No. CO1.PCCNTR.3367636 DE 2022
INSTRUCTOR GRADO 16
C.C. No. 88.265.552



ANEXOS



ANEXO 1. Medios de divulgación: Visita Universidad Digital de Antioquia

Visita de la Universidad Digital de Antioquia desde la facultad de ingeniería y la vicerrectoría de académica al centro de servicios y gestión empresarial - laboratorio de inteligencia artificial.





ANEXOS 2. Avances de redacción de artículo científico orientado en la visualización de mapas para actividades de ciencia tecnología e innovación.

Visualización geoespacial mediante analítica de datos usando Python para la medición de indicadores de Ciencia, Tecnología e Innovación en el SENA Regional Antioquia

Geospatial visualization through data analytics using Python for the measurement of indicators of Science, Technology and Innovation in the SENA Regional Antioquia

Resumen

Las actividades de Ciencia, Tecnología e Innovación - ACTI se relacionan con la realización de acciones que involucran la investigación, el desarrollo experimental, apoyo a la formación y capacitación científica y tecnológica, actividades de innovación, servicios científicos y tecnológicos, administración y otras actividades de apoyo en Ciencia Tecnología e Innovación - CTel, las cuales son realizadas por entidades públicas y privadas para aportar a la generación y apropiación del conocimiento. En ese sentido, debido al aumento exponencial que ha tenido la generación de productos de nuevo conocimiento a nivel educativo y en pro de dar respuesta de esta forma a una necesidad evidente de aportar nuevas tecnologías y conocimientos al sector productivo de la región, el SENA a través de su Sistema de Investigación SENNOVA potencializa tanto los recursos humanos como tecnológicos con los que cuenta la entidad para aportar al crecimiento económico y social de la región. Teniendo en cuenta lo anterior, en la regional Antioquia del SENA se está trabajando en el desarrollo de la visualización de un modelo de analítica de datos para la medición de indicadores KPIs, donde se identifique de manera gráfica las diferentes categorías de ACTI que se vienen trabajando en los últimos años en la entidad, dentro de las que se encuentran los procesos de gestión y productividad de proyectos por centro en un periodo de tiempo, con el propósito de facilitar la medición y análisis de los resultados obtenidos desde las perspectiva tanto de la estadística descriptiva como de la ciencia de datos, con técnicas tales como clusterización por K-means y Análisis de Componentes Principales (ACP), que permitieron agrupar los indicadores KPIs encontrados. Se utilizó Folium de Python como herramienta de visualización que permite generar comparaciones de manera gráfica entre los diferentes centros SENA de la Regional Antioquia en un periodo de tiempo determinado. Se observó que mediante estas técnicas se presenta mayor facilidad de interpretación, comparación y análisis de resultados estadísticos y de analítica de datos para medidores



de CTel frente a herramientas de visualización de la información usadas convencionalmente.

Palabras clave: Visualización de datos, Ciencia de datos, Folium, CTel, Técnicas de visualización, Herramientas de visualización, Geoespacial, Python, indicadores.

Abstract

Keywords: Data visualization, Data science, Folium, Visualization Techniques, Visualization Tools, Geospatial

Keywords: Clustering, Folium, CTel, KPIs, K-means, PCA, ACTI, SENA

Introducción

Imagen de visualización sugerida:

En la plataforma actual utilizada, con el ingeniero John Ferla se puede incorporar el mapa, la estadística descriptiva y el respectivo indicador.



Referencia:

https://s3.ca-central-1.amazonaws.com/pstorage-ryerson-5010877717/31132882/LiuSiyuan.pdf?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIA3OGA3B5WHYJSWH2T/20220602/ca-central-1/s3/aws4_request&X-Amz-Date=20220602T054935Z&X-Amz-Expires=10&X-Amz-SignedHeaders=host&X-Amz-Signature=7829fe837ccdc9b6cb13eca856ab8bfc3e4f0312fd395d952466b39e7212af18



ANEXO 3. Modelos de clusterización y clasificación, supervisados y no supervisados para las dimensiones de productividad y procesos de centros de formación en la regional Antioquia.


```

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import folium # plotting library
from folium import plugins
from folium.plugins import MarkerCluster

from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import sklearn.metrics as sm
from sklearn.decomposition import PCA
from sklearn.cluster import AgglomerativeClustering
from sklearn.cluster import DBSCAN
from io import IncrementalNewlineDecoder
import scipy.cluster.hierarchy as sch

#Cluster
from yellowbrick.cluster import SilhouetteVisualizer
from yellowbrick.cluster import KElbowVisualizer
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs

#Graficas
import matplotlib.pyplot as plt
from matplotlib import pyplot as plot
import matplotlib.cm as cm
import matplotlib.colors as colors
from plotly import graph_objects as go
from plotly.subplots import make_subplots
import plotly.express as px
import seaborn as sns
import branca
from sklearn.metrics import plot_confusion_matrix

from google.colab import drive
drive.mount('/content/drive')

```

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour



```

from google.colab import files
path1 = "/content/drive/MyDrive/Proyecto SGPS 9748/TPProceso.xlsx"
path2 = "/content/drive/MyDrive/Proyecto SGPS 9748/Datos.xlsx"
df = pd.read_excel(path1)
df_centros = pd.read_excel(path2, sheet_name= 'Centros_Formacion')
df

```

	Centro_Año	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
0	9101-2010	1	1	2	0	4.652924e+08
1	9127-2010	2	2	4	1	3.920052e+08
2	9201-2010	1	3	3	0	4.556627e+08
3	9202-2010	3	8	5	0	4.071852e+08
4	9203-2010	2	6	4	1	2.906175e+08
...	...	...	...	...	...	...
203	9501-2022	2	4	4	2	3.754077e+08
204	9502-2022	1	3	2	0	3.977569e+08
205	9503-2022	1	0	0	1	5.068536e+08
206	9504-2022	4	7	7	1	7.614273e+08
207	9549-2022	2	5	2	1	3.294535e+08

208 rows × 6 columns

df.columns

```
Index(['Centro_Año', 'Q_Proyectos', 'QNA_5007', 'QNA_5009', 'QNA_5010',
      'TV_Proyectos'],
      dtype='object')
```

```
df.columns = [ 'Centro_Año', 'Cantidad_Proyectos', 'Cantidad_Profesional', 'Cantidad_Maestria',
               'Cantidad_Doctorado', 'Total_Valor_Proyectos']
#df_2010_orig = df[df['Fecha']<2011]
#df_2010_orig = df_2010_orig.drop(['Fecha'],axis = 1)
df
```

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_
0	9101-2010	1	1	2	
1	9127-2010	2	2	4	
2	9201-2010	1	3	3	

```
df.describe()
```

	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado
count	208.000000	208.000000	208.000000	208.000000
mean	2.403846	3.552885	4.302885	1.182692
std	1.116579	2.319231	2.576293	0.909047
min	1.000000	0.000000	0.000000	0.000000
25%	2.000000	2.000000	2.000000	1.000000
50%	2.000000	3.000000	4.000000	1.000000
75%	3.000000	5.000000	6.000000	2.000000
max	6.000000	11.000000	16.000000	4.000000

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
df[['Cantidad_Proyectos', 'Cantidad_Profesional', 'Cantidad_Maestria', 'Cantidad_Doctorado', 'Tot
df
```

Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_
------------	--------------------	----------------------	-------------------	-----------

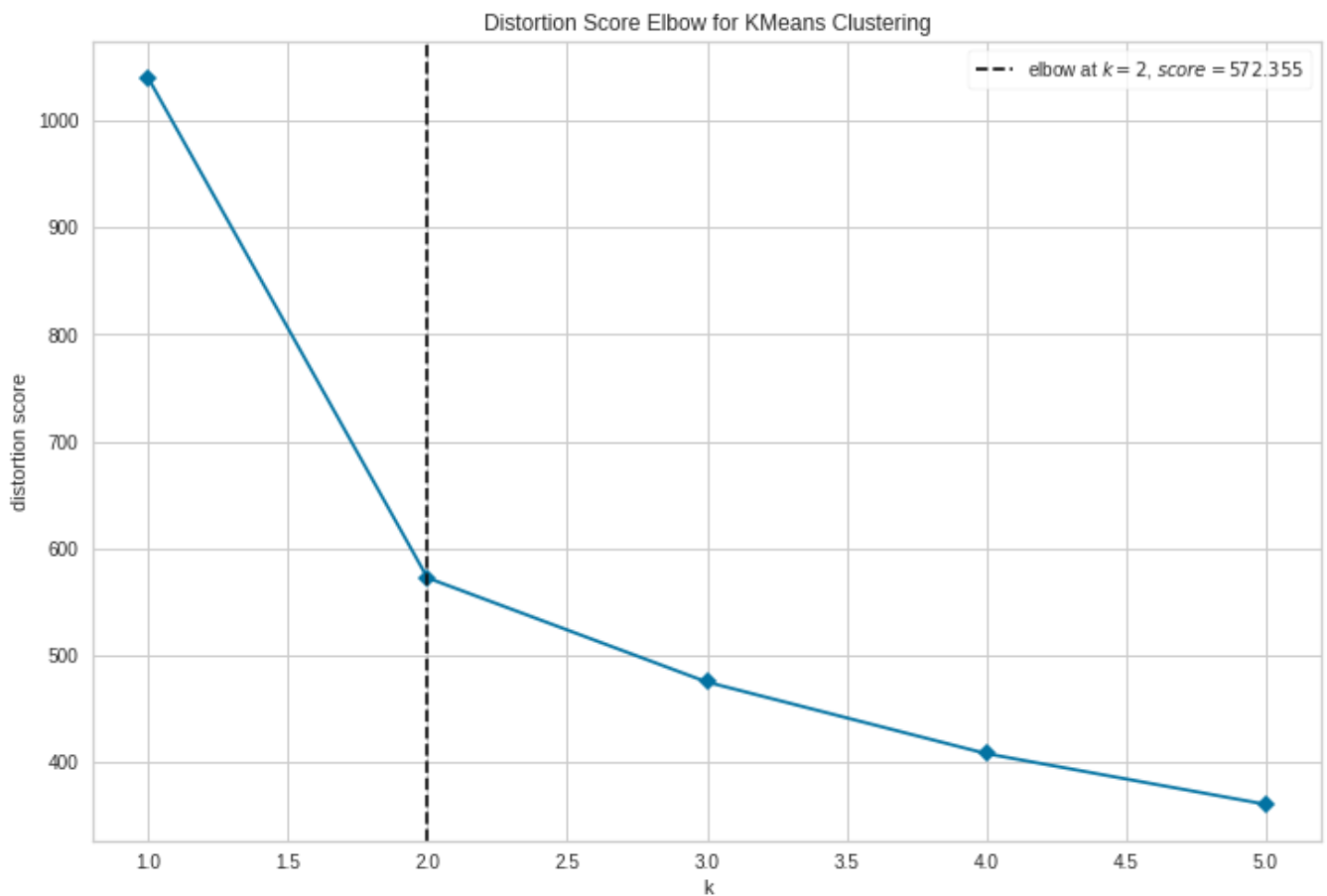
```
#Búsqueda de la cantidad óptima de clusters
##Calculando qué tan similares son los individuos dentro de los Custers
y = df[['Centro_Año']]
x = df.drop(['Centro_Año'], axis= 1)
wcss = []
for i in range (1,6):
    kmeans = KMeans(n_clusters=i, max_iter=300)
    kmeans.fit(x) #Aplico K-means a la base de datos
    wcss.append(kmeans.inertia_)
```

203	9501-2022	-0.362554	0.193251	-0.117850
-----	-----------	-----------	----------	-----------

distortion: suma media de las distancias al cuadrado de los centroides (Error medio cuadrático)

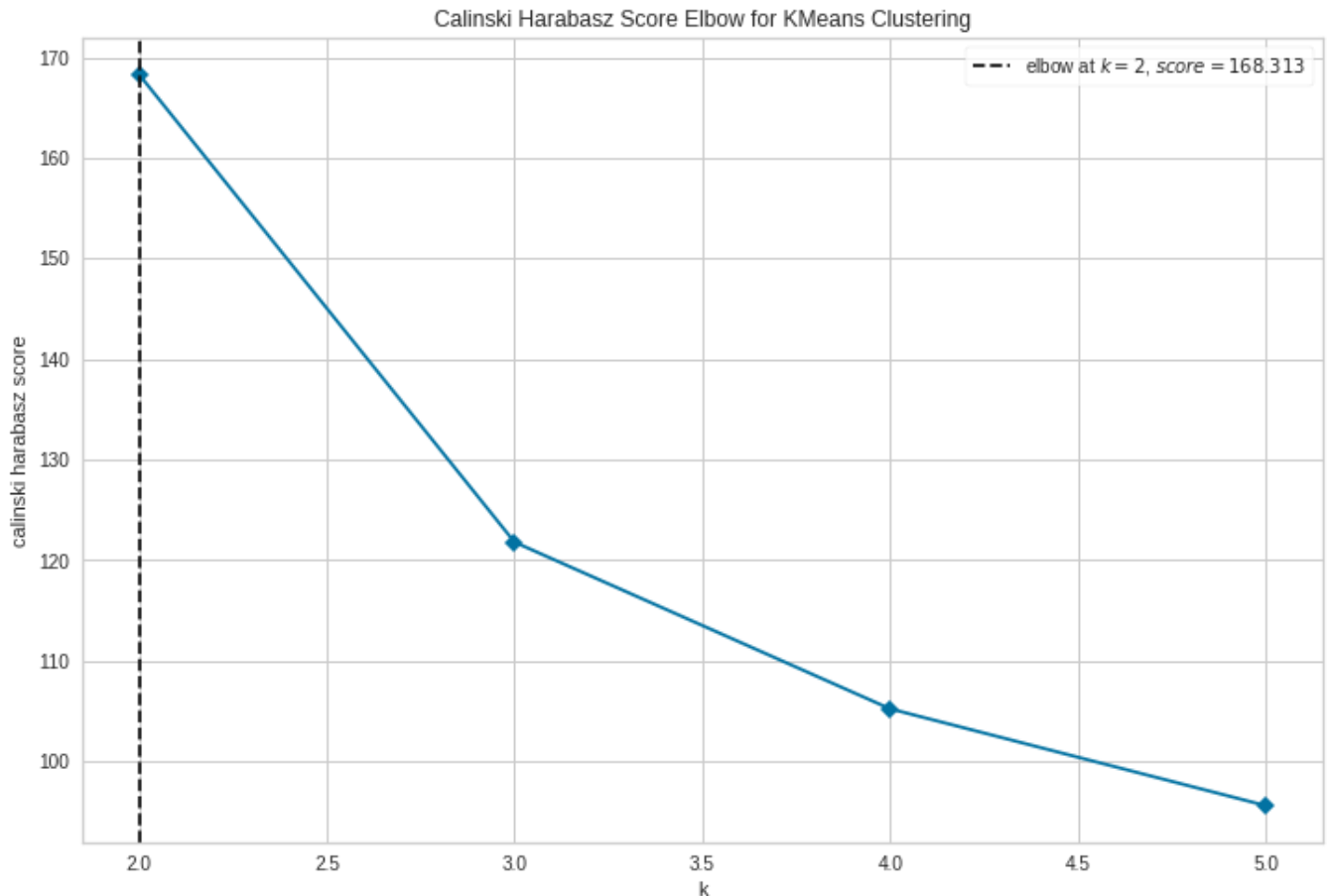
203	9501-2022	-0.362554	0.193251	-0.117850
-----	-----------	-----------	----------	-----------

```
#km_transformed = mms.transform(km)
plt.figure(figsize=(12,8))
model = KMeans(random_state=42)
visualizer = KElbowVisualizer(model, k=(1,6), timings=False)
visualizer.fit(x)
visualizer.show();
```



calinski_harabasz: relación de dentro entre a la dispersión del grupo

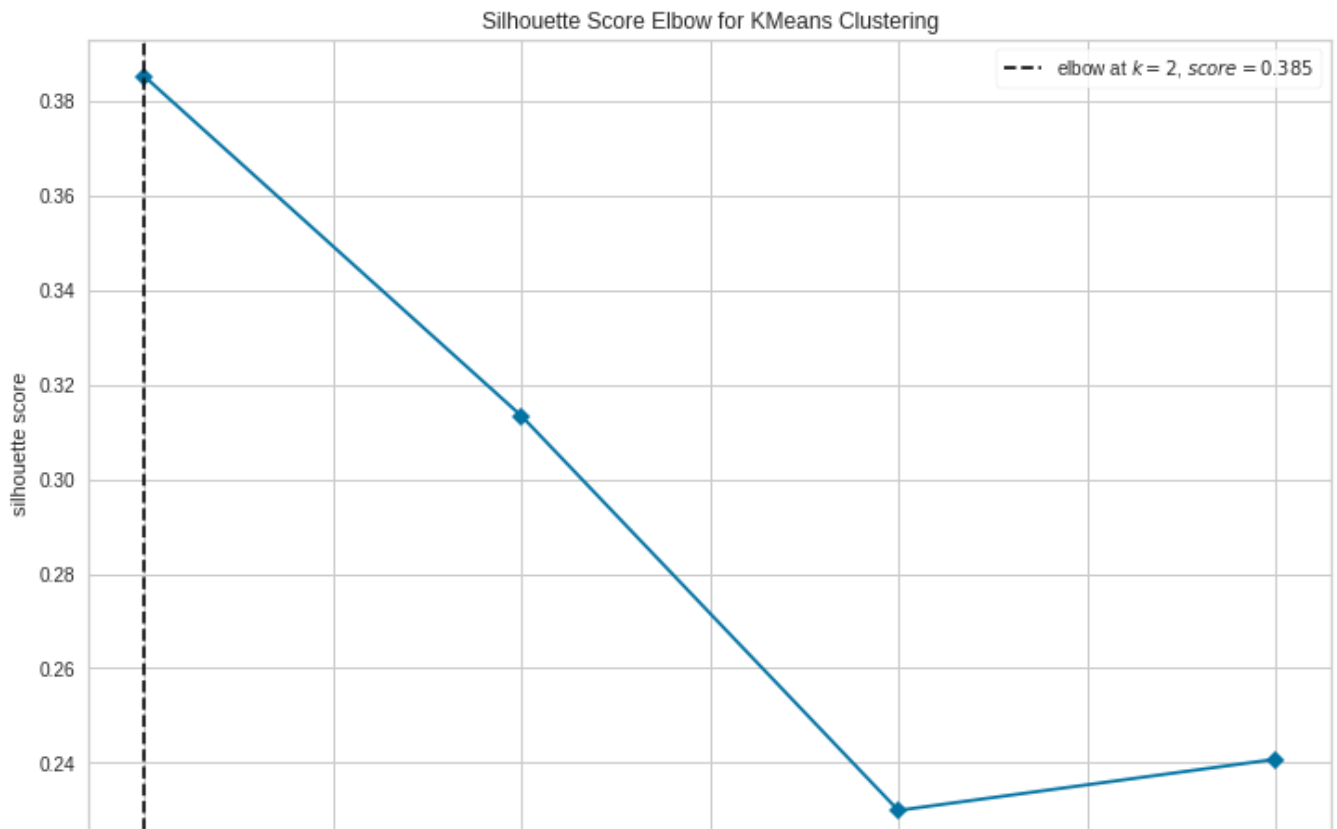
```
plt.figure(figsize=(12,8))
model = KMeans()
visualizer = KElbowVisualizer( model, k=(2,6), metric='calinski_harabasz', timings=False)
visualizer.fit(x)          # Fit the data to the visualizer
visualizer.show()          # Finalize and render the figure
```



<matplotlib.axes._subplots.AxesSubplot at 0x7fa06ace9690>

silhouette : relación media de la distancia intra-cluster y del cluster más cercano

```
plt.figure(figsize=(12,8))
model = KMeans()
visualizer = KElbowVisualizer( model, k=(2,6), metric='silhouette', timings=False)
visualizer.fit(x)          # Fit the data to the visualizer
visualizer.show()          # Finalize and render the figure
```



```
#Aplicando el método de k-means a la base de datos
clustering=KMeans(n_clusters=3, max_iter=300,init = 'random') #Crea el modelo
clustering.fit(x) #Aplica el modelo a la base de datos
clustering.cluster_centers_

array([[ 1.72023372,  1.4726154 ,  1.0494234 ,  1.60698425,  1.55894956],
       [-0.73086338, -0.5714416 , -0.58010313, -0.36167866, -0.6177202 ],
       [ 0.64401777,  0.45520125,  0.63085578,  0.03245148,  0.50453825]])
```

Se obtienen dos centroides con coordenadas en 5 dimensiones(cantidad de variables), por lo que no se genera visualización. Se debe reducir dimensionalidad como PCA.

```
#Agregando la clasificación en el archivo original
df["KMeans_Clusters"]=clustering.labels_
df
```

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_
0	9101-2010	-1.260308	-1.103402	-0.896032	
1	9127-2010	-0.362554	-0.671184	-0.117850	
2	9201-2010	-1.260308	-0.238966	-0.506941	
3	9202-2010	0.535199	1.922122	0.271241	
4	9203-2010	-0.362554	1.057686	-0.117850	
...	...	...	...	...	...

```
df.to_excel('TPProceso_Cluster_Kmeans.xlsx',index =False)
```

```
...
```

▼ Validación de clusters

```
...
```

Indices externos

208 rows × 7 columns

```
#Calcular agrupación de índices de validación externa con KMeans
labels_true = df["Centro_Año"]
rotulos_km = clustering.labels_
h = sm.homogeneity_score(labels_true, rotulos_km)
print("Homogeneidad: %0.3f" % h)
c = sm.completeness_score(labels_true, rotulos_km)
print("Compleitud: %0.3f" % c)
v = sm.v_measure_score(labels_true, rotulos_km)
print("Medida V: %0.3f" % v )
ari = sm.adjusted_rand_score(labels_true, rotulos_km)
print("Índice Rand corregido: %0.3f" % ari)
ami = sm.adjusted_mutual_info_score(labels_true, rotulos_km, average_method='arithmetic')
print("Adjusted Mutual Information: %0.3f" % ami )
fm = sm.fowlkes_mallows_score(labels_true, rotulos_km)
print("Índice Fowlkes-Mallows: %0.3f" % fm )
```

```
Homogeneidad: 0.177
Compleitud: 1.000
Medida V: 0.300
Índice Rand corregido: 0.000
Adjusted Mutual Information: 0.000
Índice Fowlkes-Mallows: 0.000
```

Índices internos

```
from sklearn.metrics import silhouette_score
#silhouette_avg = silhouette_score(x, clustering.labels_)
s= sm.silhouette_score(x, clustering.labels_, metric='euclidean')
```



```
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
-1, -1, -1, -1]]
```

```
#Contar número de clusters
#labels = db.labels_
#n_clusters_ = len(set(labels)) - (1 if -1 in labels else 0)
#n_noise_ = list(labels).count(-1)
#print('Número estimado de clusters: %d' % n_clusters_)
#print('Número estimado de datos ruidosos: %d' % n_noise_)
#print("Adjusted Rand Index: %0.3f"
#      % metrics.adjusted_rand_score(labels_true, labels))
#print("Adjusted Mutual Information: %0.3f"
#      % metrics.adjusted_mutual_info_score(labels_true, labels))
#print("Silhouette Coefficient: %0.3f"
#      % metrics.silhouette_score(X, labels))
```

▼ PCA

```
#Visualización de los Clusters que se crearon
#Aplicaremos Análisis de componentes principales (PCA) para darnos una idea de cómo se crearon
pca=PCA(n_components=2)
pca_productividad=pca.fit_transform(x)
pca_productividad_2=pd.DataFrame(data=pca_productividad,columns=["Componente_1", "Componente_2"])
pca_productividad_3=pd.concat([pca_productividad_2,df[["KMeans_Clusters"]]],axis=1)
pca_productividad_3=pd.concat([pca_productividad_3,df[["Centro_Año"]]],axis=1)
pca_productividad_3
```

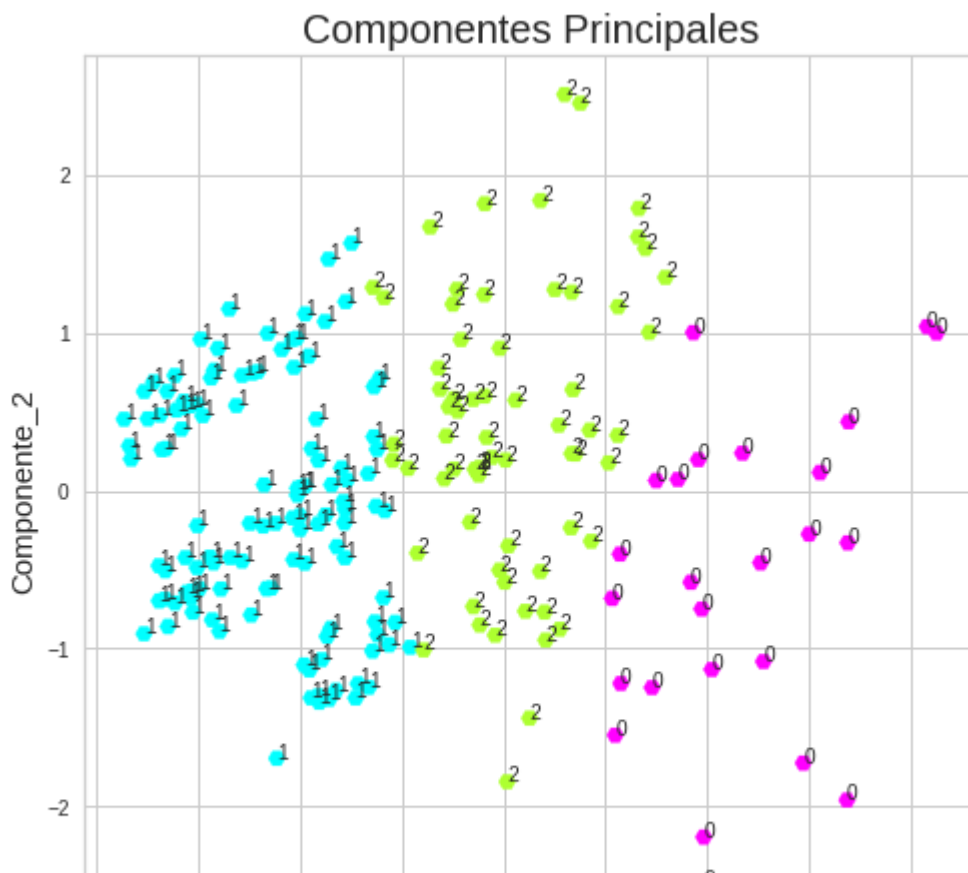
	Componente_1	Componente_2	KMeans_Clusters	Centro_Año
0	-2.071034	0.534751	1	9101-2010
1	-0.944058	0.028965	1	9127-2010
2	-1.562372	0.727969	1	9201-2010
3	0.503960	1.179333	2	9202-2010

```
pca_productividad_3 = pca_productividad_3.dropna()
pca_productividad_3['KMeans_Clusters'] = pca_productividad_3['KMeans_Clusters'].astype(int)
pca_productividad_3
```

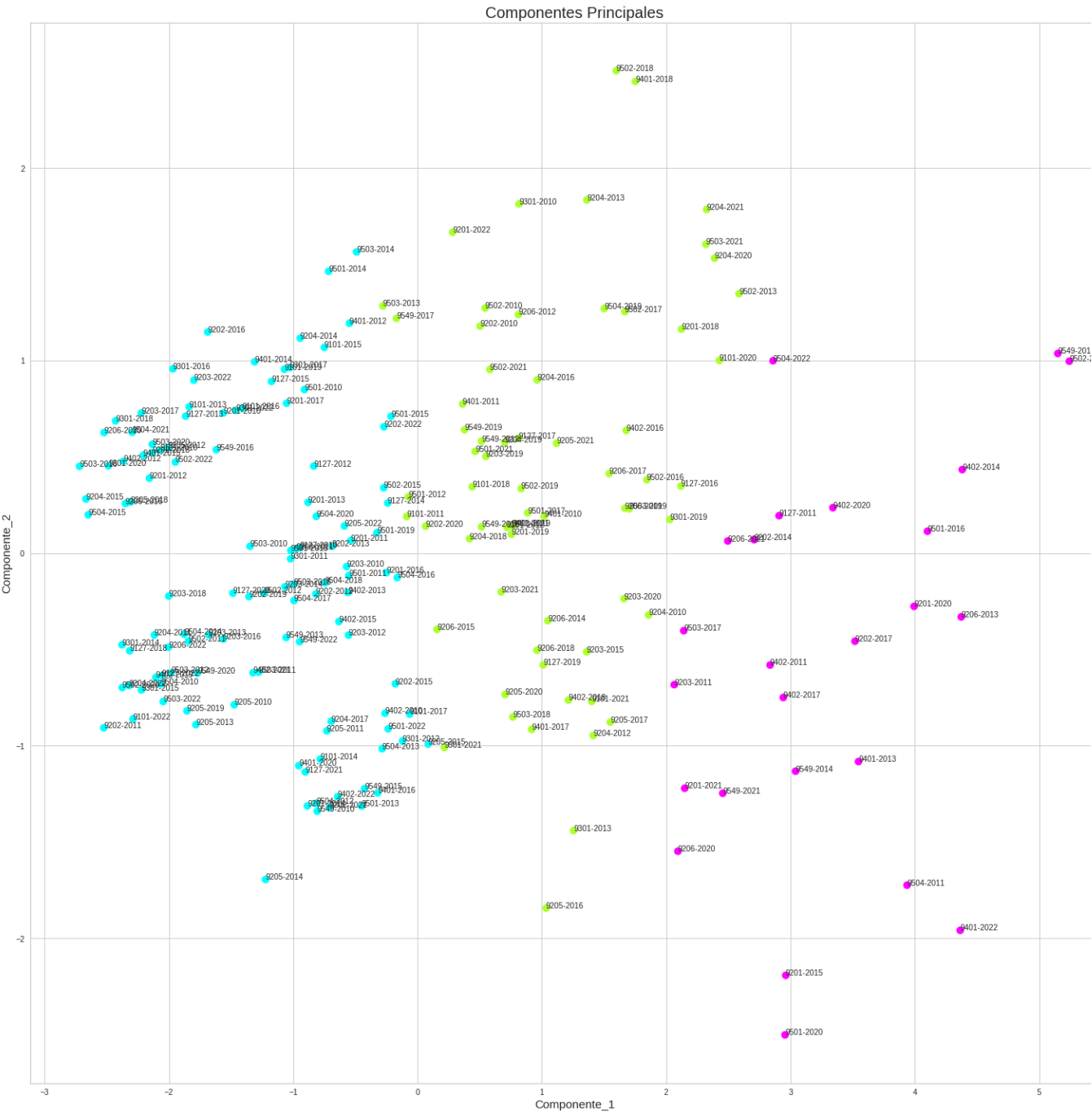
	Componente_1	Componente_2	KMeans_Clusters	Centro_Año
0	-2.071034	0.534751	1	9101-2010
1	-0.944058	0.028965	1	9127-2010
2	-1.562372	0.727969	1	9201-2010
3	0.503960	1.179333	2	9202-2010
4	-0.570482	-0.069628	1	9203-2010
...	...	...	...	...
203	-0.235608	-0.912225	1	9501-2022
204	-1.947535	0.473094	1	9502-2022
205	-2.045086	-0.770374	1	9503-2022
206	2.860241	0.999405	0	9504-2022
207	-0.948307	-0.459835	1	9549-2022

208 rows × 4 columns

```
fig=plt.figure(figsize=(8,8))
ax=fig.add_subplot(1,1,1)
annotations = df['KMeans_Clusters']
ax.set_xlabel("Componente_1",fontsize=15)
ax.set_ylabel("Componente_2", fontsize=15)
ax.set_title("Componentes Principales", fontsize=20)
color_theme=np.array(["MAGENTA","CYAN",'GREENYELLOW','mediumpurple'])
ax.scatter(x=pca_productividad_3['Componente_1'], y=pca_productividad_3['Componente_2'],c=col
for i, label in enumerate(annotations):
    plt.annotate(label, (pca_productividad_3['Componente_1'][i], pca_productividad_3['Compone
plt.show()
```



```
#revisar tamaño de etiqueta de punto
fig=plt.figure(figsize=(25,25))
ax=fig.add_subplot(1,1,1)
annotations = df['Centro_Año']
ax.set_xlabel("Componente_1",fontsize=15)
ax.set_ylabel("Componente_2", fontsize=15)
ax.set_title("Componentes Principales", fontsize=20)
color_theme=np.array(["MAGENTA","CYAN",'GREENYELLOW','mediumpurple'])
ax.scatter(x=pca_productividad_3['Componente_1'], y=pca_productividad_3['Componente_2'],c=col
for i, label in enumerate(annotations):
    plt.annotate(label, (pca_productividad_3['Componente_1'][i], pca_productividad_3['Compone
plt.show()
```



df

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_
0	9101-2010	-1.260308	-1.103402	-0.896032	
1	9127-2010	-0.362554	-0.671184	-0.117850	
2	9201-2010	-1.260308	-0.238966	-0.506941	
3	9202-2010	0.535199	1.922122	0.271241	
4	9203-2010	-0.362554	1.057686	-0.117850	
...	...	...	...	...	
203	9501-2022	-0.362554	0.193251	-0.117850	
204	9502-2022	-1.260308	-0.238966	-0.896032	
205	9503-2022	-1.260308	-1.535619	-1.674214	
206	9504-2022	-1.260308	-1.480004	-1.040422	

```
#df_2010 = df_2010.merge(df_centros,on='Codigo_Centro',how='inner')
#df_2010 = df_2010[['Codigo_Centro','Nombre_Centro','Cantidad_Proyectos','Cantidad_Profesional',
                    'Cantidad_Maestria','Cantidad_']
208 rows × 7 columns

#df_2010.head()
```

```
pip install mlxtend
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: mlxtend in /usr/local/lib/python3.7/dist-packages (0.20.0)
Requirement already satisfied: matplotlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (3.3.0)
Requirement already satisfied: joblib>=0.13.2 in /usr/local/lib/python3.7/dist-packages (1.1.0)
Requirement already satisfied: scikit-learn>=1.0.2 in /usr/local/lib/python3.7/dist-packages (1.0.2)
Requirement already satisfied: pandas>=0.24.2 in /usr/local/lib/python3.7/dist-packages (1.3.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (57.0.0)
Requirement already satisfied: numpy>=1.16.2 in /usr/local/lib/python3.7/dist-packages (1.21.0)
Requirement already satisfied: scipy>=1.2.1 in /usr/local/lib/python3.7/dist-packages (1.7.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (0.10)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (1.3.1)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (2.8.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (3.0.7)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (4.1.1)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (2022.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (1.16.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (2.2.0)
```

```
pip install mlxtend --upgrade --no-deps
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: mlxtend in /usr/local/lib/python3.7/dist-packages (0.20.0)
```

```
from mlxtend.plotting import plot_pca_correlation_graph
```

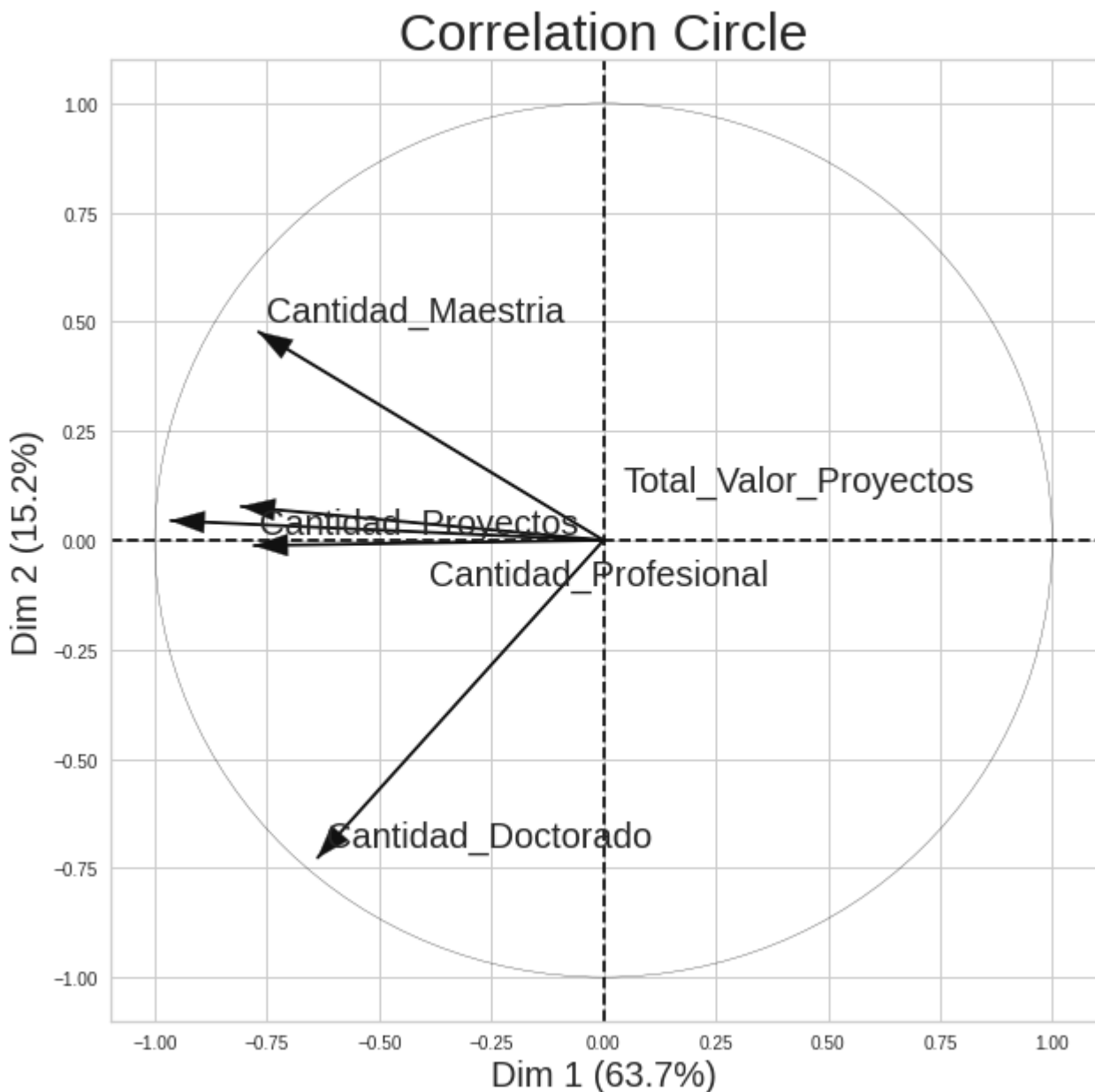
```
df.columns
```

```
Index(['Centro_Año', 'Cantidad_Proyectos', 'Cantidad_Profesional',  
      'Cantidad_Maestria', 'Cantidad_Doctorado', 'Total_Valor_Proyectos',  
      'KMeans_Clusters'],  
      dtype='object')
```

```
x1= df.drop(['Centro_Año', 'KMeans_Clusters'],axis = 1)
```

```
features = ['Cantidad_Proyectos',  
           'Cantidad_Profesional', 'Cantidad_Maestria', 'Cantidad_Doctorado',  
           'Total_Valor_Proyectos']
```

```
figure, correlation_matrix = plot_pca_correlation_graph(x1, features,  
                                                         dimensions=(1, 2),  
                                                         figure_axis_size=10)
```



correlation_matrix

	Dim 1	Dim 2
Cantidad_Proyectos	-0.964588	0.044963
Cantidad_Profesional	-0.779158	-0.012219
Cantidad_Maestria	-0.768787	0.476811
Cantidad_Doctorado	-0.637097	-0.726372
Total_Valor_Proyectos	-0.807754	0.077194

x1

	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	T
0	-1.260308	-1.103402	-0.896032	-1.304164	
1	-0.362554	-0.671184	-0.117850	-0.201456	
2	-1.260308	-0.238966	-0.506941	-1.304164	
3	0.535199	1.922122	0.271241	-1.304164	
4	-0.362554	1.057686	-0.117850	-0.201456	
...	...	...	...	...	
203	-0.362554	0.193251	-0.117850	0.901251	
204	-1.260308	-0.238966	-0.896032	-1.304164	
205	-1.260308	-1.535619	-1.674214	-0.201456	
206	1.432953	1.489904	1.049423	-0.201456	
207	-0.362554	0.625469	-0.896032	-0.201456	

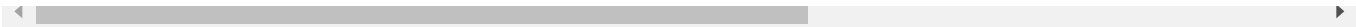
208 rows × 5 columns

```
gestion= df['KMeans_Clusters'].replace({1: 'Baja Gestión',0: 'Media Gestión', 2: 'Alta Gestión'})
df['Gestión'] = pd.concat([gestion], axis=1)
df_k_clas = df
df_k_clas
```

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_
0	9101-2010	-1.260308	-1.103402	-0.896032	
1	9127-2010	-0.362554	-0.671184	-0.117850	
2	9201-2010	-1.260308	-0.238966	-0.506941	
3	9202-2010	0.535199	1.922122	0.271241	
4	9203-2010	-0.362554	1.057686	-0.117850	
...	...	...	...	...	
203	9501-2022	-0.362554	0.193251	-0.117850	
204	9502-2022	-1.260308	-0.238966	-0.896032	
205	9503-2022	-1.260308	-1.535619	-1.674214	

```
df_k_clas.to_excel('TPProceso_Cluster_Kmeans.xlsx',index =False)
```

▼ DENDOGRAMA



```
df = df.drop(['Gestión', 'KMeans_Clusters'], axis=1)
df
```


	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_
0	9101-2010	-1.260308	-1.103402	-0.896032	
1	9127-2010	-0.362554	-0.671184	-0.117850	
-	-	-	-	-	-

```
from scipy.cluster.hierarchy import linkage,dendrogram
```

```
df_el = df.drop(['Centro_Año'],axis =1)
```

```
labels = df['Centro_Año'].to_numpy().tolist()
```

```
...      ...      ...      ...      ...
```

```
x.columns
```

```
Index(['Cantidad_Proyectos', 'Cantidad_Profesional', 'Cantidad_Maestria',
       'Cantidad_Doctorado', 'Total_Valor_Proyectos'],
      dtype='object')
```

```
206    9504-2022    1.432953    1.489904    1.049423
```

```
features = ['Cantidad_Proyectos', 'Cantidad_Profesional', 'Cantidad_Maestria',
            'Cantidad_Doctorado', 'Total_Valor_Proyectos']
```

```
X = x[features]
```

```
plt.figure(figsize=(20,10))
```

```
plt.title('Dendrograma de agrupamiento jerárquico para datos de procesos',fontsize=20)
```

```
plt.xlabel('sample index', fontsize=20)
```

```
plt.ylabel('distance',fontsize=20)
```

```
Z = linkage(X, 'complete','euclidean')
```

```
dn1 = sch.dendrogram(
```

```
    Z,
```

```
    labels=labels,
```

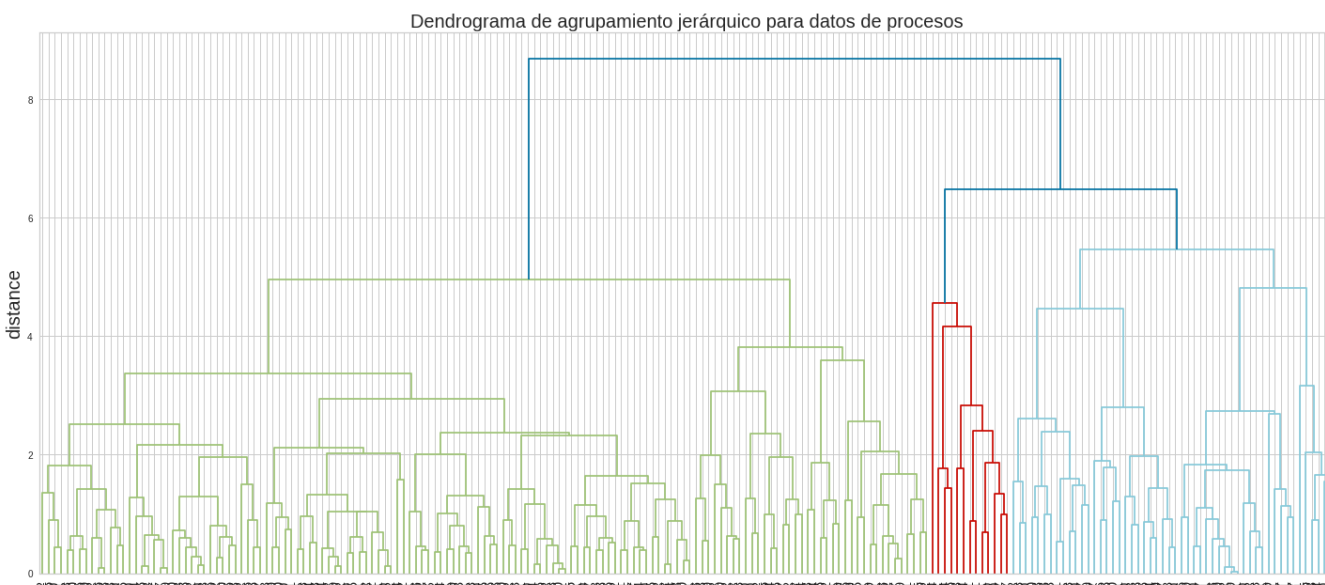
```
    leaf_rotation=90., # rotates the x axis labels
```

```
    leaf_font_size=15., # font size for the x axis labels
```

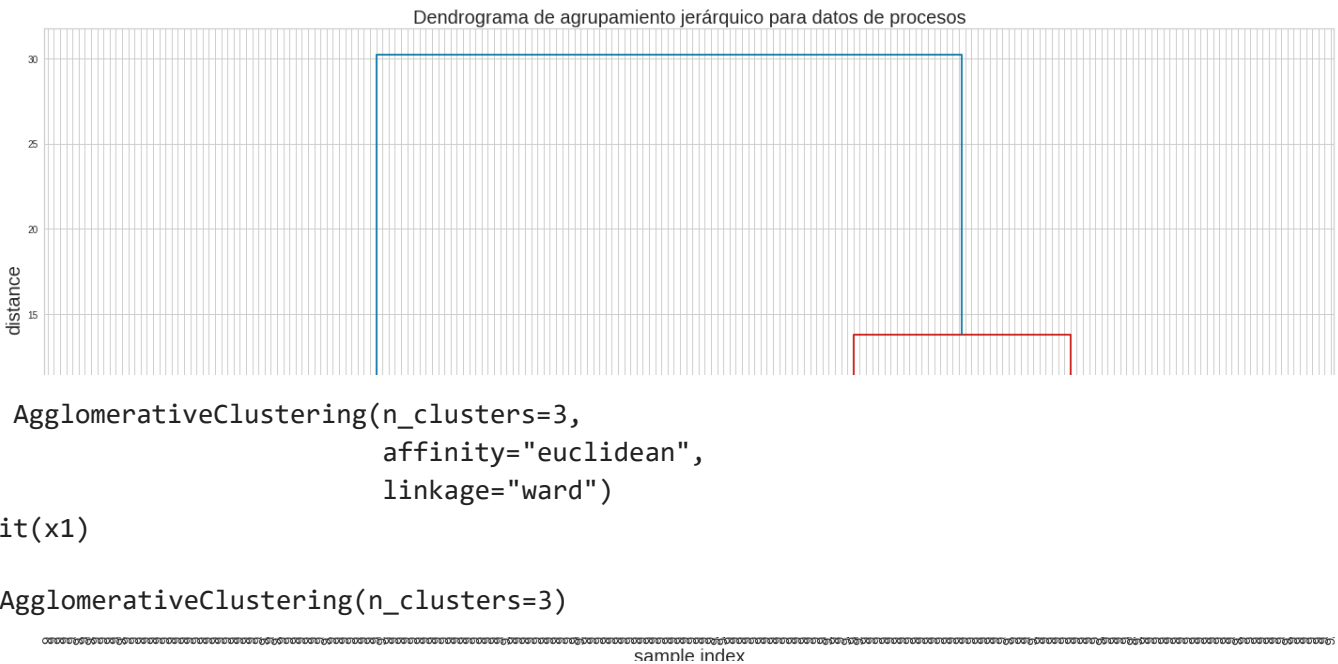
```
)
```

```
plt.tight_layout()
```

```
plt.show()
```



```
plt.figure(figsize=(20,10))
plt.title('Dendrograma de agrupamiento jerárquico para datos de procesos',fontsize=20)
plt.xlabel('sample index', fontsize=20)
plt.ylabel('distance',fontsize=20)
Z = linkage(X, 'ward','euclidean')
dn2 = sch.dendrogram(
    Z,
    labels=labels,
    leaf_rotation=90., # rotates the x axis labels
    leaf_font_size=15., # font size for the x axis labels
)
plt.tight_layout()
plt.show()
```



```

hac.fit(X)
categorias = hac.labels_
categorias

array([0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 2, 0, 0, 2, 0,
       0, 0, 0, 1, 2, 0, 0, 0, 2, 2, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 2, 1, 2, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 2, 0, 0, 0, 1, 0, 0, 2, 0, 2, 0, 0, 2, 0, 0, 2, 0, 1, 0, 1, 1,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 2, 1, 0,
       0, 0, 0, 1, 0, 2, 0, 0, 2, 1, 0, 1, 2, 1, 1, 2, 0, 0, 1, 0, 1, 0,
       0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1,
       0, 0, 1, 1, 1, 0, 1, 0, 2, 0, 1, 1, 1, 1, 1, 0, 0, 2, 2, 0, 0, 0, 0,
       1, 0, 2, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 2, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 2, 0, 0, 0, 0, 1, 0])

#Agregando la clasificación en el archivo original
df["Hac_Clusters"]=categorias
df

```

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_
0	9101-2010	-1.260308	-1.103402	-0.896032	
1	9127-2010	-0.362554	-0.671184	-0.117850	
2	9201-2010	-1.260308	-0.238966	-0.506941	
3	9202-2010	0.535199	1.922122	0.271241	

▼ Validación

índices de validación externa

204	9502-2022	-1.260308	-0.238966	-0.896032
-----	-----------	-----------	-----------	-----------

```
#Calcular agrupación de índices de validación externa con KMeans
labels_true = df["Centro_Año"]
rotulos_km = hac.labels_
h = sm.homogeneity_score(labels_true, rotulos_km)
print("Homogeneidad: %0.3f" % h)
c = sm.completeness_score(labels_true, rotulos_km)
print("Compleitud: %0.3f" % c)
v = sm.v_measure_score(labels_true, rotulos_km)
print("Medida V: %0.3f" % v)
ari = sm.adjusted_rand_score(labels_true, rotulos_km)
print("Índice Rand corregido: %0.3f" % ari)
ami = sm.adjusted_mutual_info_score(labels_true, rotulos_km, average_method='arithmetic')
print("Adjusted Mutual Information: %0.3f" % ami)
fm = sm.fowlkes_mallows_score(labels_true, rotulos_km)
print("Índice Fowlkes-Mallows: %0.3f" % fm)
```

```
Homogeneidad: 0.173
Compleitud: 1.000
Medida V: 0.294
Índice Rand corregido: 0.000
Adjusted Mutual Information: -0.000
Índice Fowlkes-Mallows: 0.000
```

índices de validación interna

```
from sklearn.metrics import silhouette_score
#silhouette_avg = silhouette_score(x, clustering.labels_)
rotulos_hac = hac.labels_
s = sm.silhouette_score(x, rotulos_hac, metric='euclidean')
print("Coeficiente de Silhueta: %0.3f" % s)
ch = sm.calinski_harabasz_score(x, rotulos_hac)
print("Calinski_harabasz: %0.3f" % ch)
dbs = sm.davies_bouldin_score(x, rotulos_hac)
print("Davies Bouldin: %0.3f" % dbs)
```

Coeficiente de Silhueta: 0.310
 Calinski_harabasz: 115.798
 Davies Bouldin: 1.333

x.head()

	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	Tot
0	-1.260308	-1.103402	-0.896032	-1.304164	
1	-0.362554	-0.671184	-0.117850	-0.201456	
2	-1.260308	-0.238966	-0.506941	-1.304164	
3	0.535199	1.922122	0.271241	-1.304164	
4	-0.362554	1.057686	-0.117850	-0.201456	

```
#Visualización de los Clusters que se crearon
#Aplicaremos Análisis de componetes principales (PCA) para darnos una idea de cómo se crearon
pca=PCA(n_components=2)
pca_productividad=pca.fit_transform(x)
pca_productividad_2=pd.DataFrame(data=pca_productividad,columns=["Componente_1", "Componente_2"])
pca_productividad_3=pd.concat([pca_productividad_2,df[["Hac_Clusters"]]],axis=1)
pca_productividad_3=pd.concat([pca_productividad_3,df[["Centro_Año"]]],axis=1)
pca_productividad_3
```

	Componente_1	Componente_2	Hac_Clusters	Centro_Año
0	-2.071034	0.534751	0	9101-2010
1	-0.944058	0.028965	0	9127-2010
2	-1.562372	0.727969	0	9201-2010
3	0.503960	1.179333	1	9202-2010
4	-0.570482	-0.069628	0	9203-2010
...	...	...	...	...
203	-0.235608	-0.912225	0	9501-2022
204	-1.947535	0.473094	0	9502-2022
205	-2.045086	-0.770374	0	9503-2022
206	2.860241	0.999405	1	9504-2022
207	-0.948307	-0.459835	0	9549-2022

208 rows × 4 columns

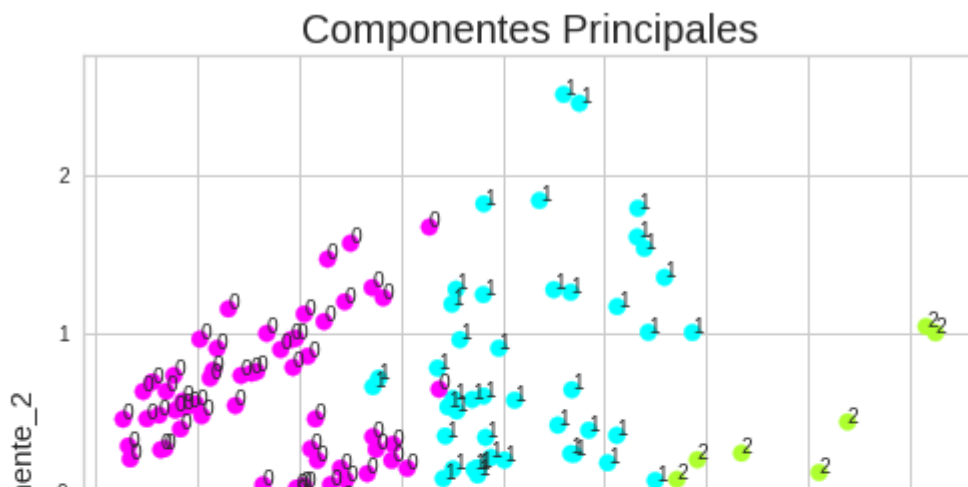
```
pca_productividad_3 = pca_productividad_3.dropna()
```

```
pca_productividad_3['Hac_Clusters'] = pca_productividad_3['Hac_Clusters'].astype(int)
pca_productividad_3
```

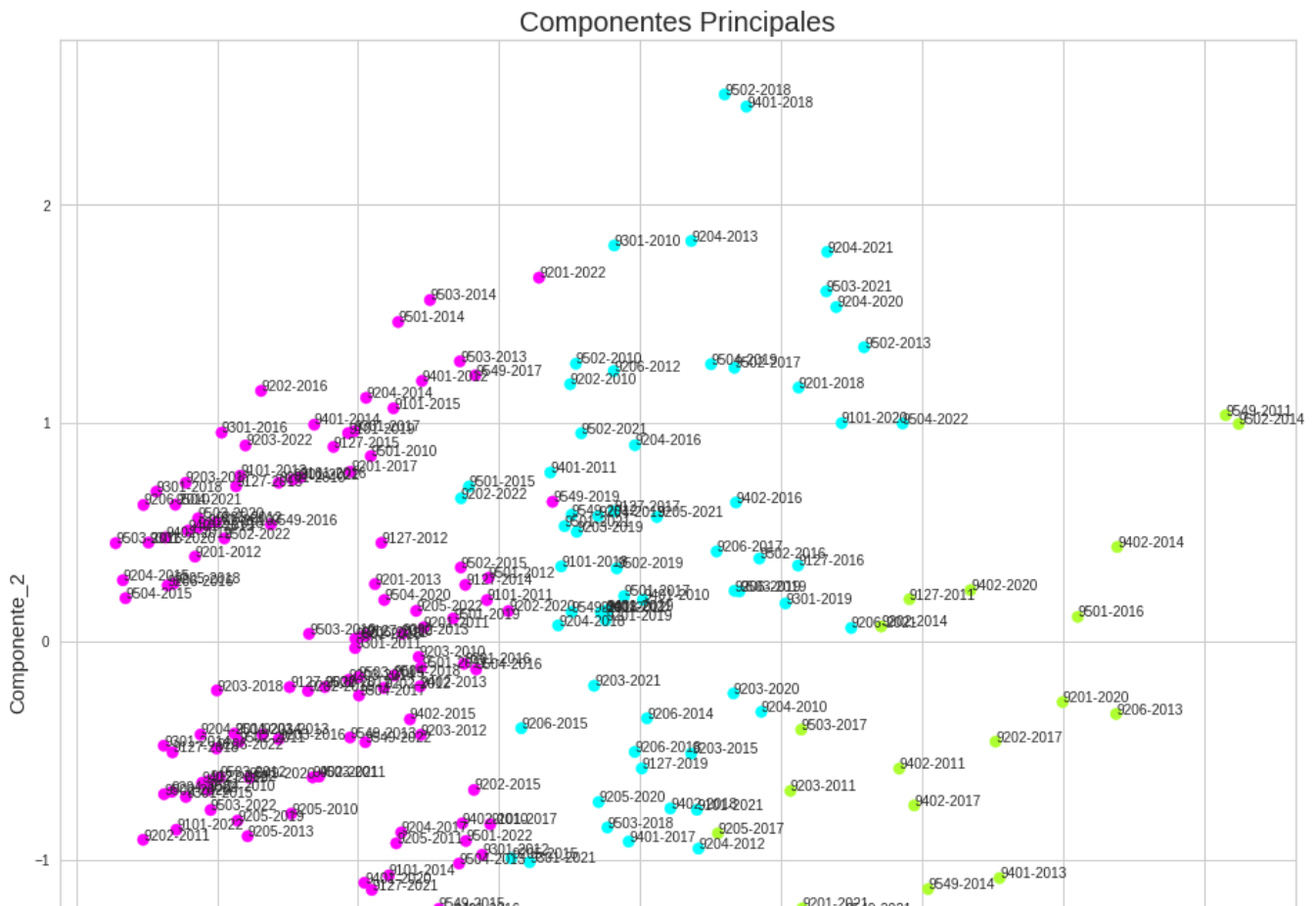
	Componente_1	Componente_2	Hac_Clusters	Centro_Año
0	-2.071034	0.534751	0	9101-2010
1	-0.944058	0.028965	0	9127-2010
2	-1.562372	0.727969	0	9201-2010
3	0.503960	1.179333	1	9202-2010
4	-0.570482	-0.069628	0	9203-2010
...	...	...	...	...
203	-0.235608	-0.912225	0	9501-2022
204	-1.947535	0.473094	0	9502-2022
205	-2.045086	-0.770374	0	9503-2022
206	2.860241	0.999405	1	9504-2022
207	-0.948307	-0.459835	0	9549-2022

208 rows × 4 columns

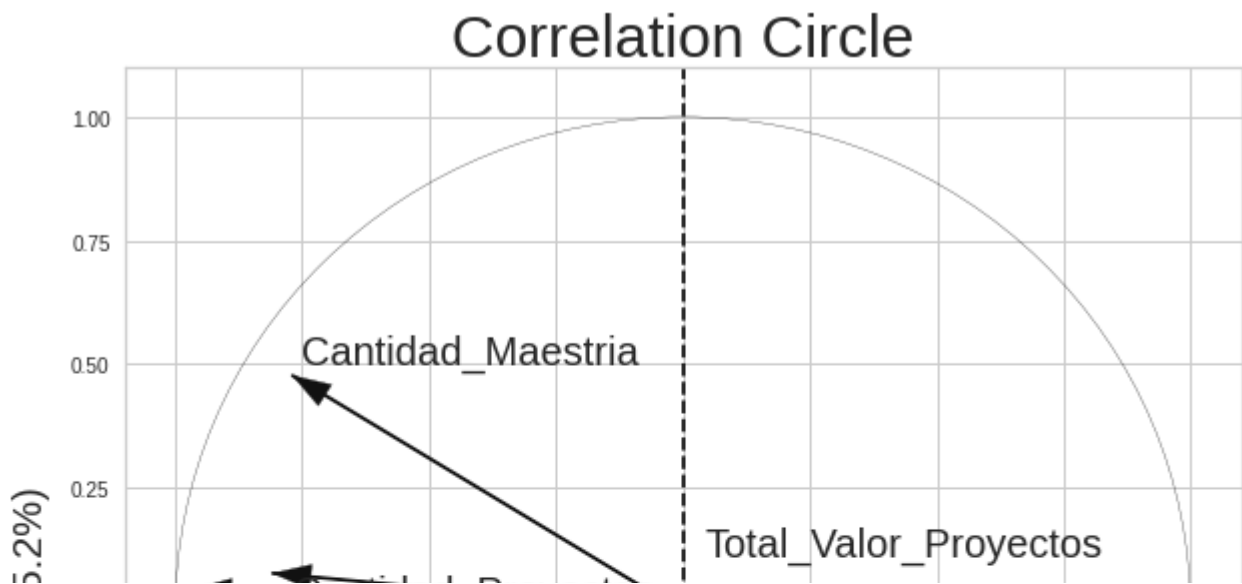
```
fig=plt.figure(figsize=(8,8))
ax=fig.add_subplot(1,1,1)
annotations = df['Hac_Clusters']
ax.set_xlabel("Componente_1",fontsize=15)
ax.set_ylabel("Componente_2", fontsize=15)
ax.set_title("Componentes Principales", fontsize=20)
color_theme=np.array(["MAGENTA","CYAN",'GREENYELLOW','mediumpurple'])
ax.scatter(x=pca_productividad_3['Componente_1'], y=pca_productividad_3['Componente_2'],c=col
for i, label in enumerate(annotations):
    plt.annotate(label, (pca_productividad_3['Componente_1'][i], pca_productividad_3['Compone
plt.show()
```



```
fig=plt.figure(figsize=(16,16))
ax=fig.add_subplot(1,1,1)
annotations = pca_productividad_3['Centro_Año']
ax.set_xlabel("Componente_1",fontsize=15)
ax.set_ylabel("Componente_2", fontsize=15)
ax.set_title("Componentes Principales", fontsize=20)
color_theme=np.array(["MAGENTA","CYAN",'GREENYELLOW','mediumpurple'])
ax.scatter(x=pca_productividad_3['Componente_1'], y=pca_productividad_3['Componente_2'],c=col
for i, label in enumerate(annotations):
    plt.annotate(label, (pca_productividad_3['Componente_1'][i], pca_productividad_3['Compone
plt.show()
```



```
x1= df.drop(['Centro_Año', 'Hac_Clusters'],axis = 1)
features = ['Cantidad_Proyectos',
            'Cantidad_Profesional', 'Cantidad_Maestria', 'Cantidad_Doctorado',
            'Total_valor_Proyectos']
figure, correlation_matrix = plot_pca_correlation_graph(x1, features,
                                                         dimensions=(1, 2),
                                                         figure_axis_size=10)
```

correlation_matrix

	Dim 1	Dim 2
Cantidad_Proyectos	-0.964588	0.044963
Cantidad_Profesional	-0.779158	-0.012219
Cantidad_Maestria	-0.768787	0.476811
Cantidad_Doctorado	-0.637097	-0.726372
Total_valor_proyectos	-0.807754	0.077194

```
gestion= df['Hac_Clusters'].replace({2: 'Baja Gestión',1: 'Media Gestión', 0: 'Alta Gestión'})
df['Gestión'] = pd.concat([gestion], axis=1)
df_hac_clas = df
df_hac_clas
```

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_
0	9101-2010	-1.260308	-1.103402	-0.896032	
1	9127-2010	-0.362554	-0.671184	-0.117850	

```
df_hac_clas.to_excel('TPProceso_Cluster_jerárquico.xlsx',index =False)
```

```
df = df_k_clas
centros_años = df["Centro_Año"].str.split("-",expand=True)
centros_años.columns = ['Codigo_Centro', 'Año']
df = pd.concat([df, centros_años], axis=1)
df.head()
```

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Do
0	9101-2010	-1.260308	-1.103402	-0.896032	-1
1	9127-2010	-0.362554	-0.671184	-0.117850	-0
2	9201-2010	-1.260308	-0.238966	-0.506941	-1
3	9202-2010	0.535199	1.922122	0.271241	-1
4	9203-2010	-0.362554	1.057686	-0.117850	-0

```
df.columns
```

```
Index(['Centro_Año', 'Cantidad_Proyectos', 'Cantidad_Profesional',
      'Cantidad_Maestria', 'Cantidad_Doctorado', 'Total_valor_Proyectos',
      'KMeans_Clusters', 'Gestión', 'Codigo_Centro', 'Año'],
      dtype='object')
```

```
df_map = df[['Codigo_Centro', 'Año', 'KMeans_Clusters', 'Gestión']]
df_map
```

	Codigo_Centro	Año	KMeans_Clusters	Gestión
0	9101	2010	1	Baja Gestión
1	9127	2010	1	Baja Gestión
2	9201	2010	1	Baja Gestión
3	9202	2010	2	Alta Gestión
4	9203	2010	1	Baja Gestión
...	...	...	...	...
203	9501	2022	1	Baja Gestión
204	9502	2022	1	Baja Gestión

```
df_map['Codigo_Centro'] = df_map['Codigo_Centro'].astype(int)
df_map['Año'] = df_map['Año'].astype(int)
```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>
""Entry point for launching an IPython kernel.

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>



```
df_map = df_map.merge(df_centros, on='Codigo_Centro', how='inner')
df_map.head()
```

	Codigo_Centro	Año	KMeans_Clusters	Gestión	Nombre
0	9101	2010	1	Baja Gestión	Centro_de_los_Recursos_Naturales_Renova
1	9101	2011	2	Alta Gestión	Centro_de_los_Recursos_Naturales_Renova
2	9101	2012	2	Alta Gestión	Centro_de_los_Recursos_Naturales_Renova
3	9101	2013	1	Baja Gestión	Centro_de_los_Recursos_Naturales_Renova
4	9101	2014	1	Baja Gestión	Centro_de_los_Recursos_Naturales_Renova

```
df_map = df_map.drop(['Codigo_Regional','Codigo_Municipio','Codigo_Departamento'],axis= 1)
```

df_map

	Codigo_Centro	Año	KMeans_Clusters	Gestión	Noml
0	9101	2010	1	Baja Gestión	Centro_de_los_Recursos_Naturales_Reno
1	9101	2011	2	Alta Gestión	Centro_de_los_Recursos_Naturales_Reno
2	9101	2012	2	Alta Gestión	Centro_de_los_Recursos_Naturales_Reno
3	9101	2013	1	Baja Gestión	Centro_de_los_Recursos_Naturales_Reno
4	9101	2014	1	Baja Gestión	Centro_de_los_Recursos_Naturales_Reno
...	...	...	...	...	
203	9549	2018	2	Alta Gestión	Complejo_Tecnológico_Turístico_y_Agi
204	9549	2019	2	Alta Gestión	Complejo_Tecnológico_Turístico_y_Agi

df_map.dtypes

```
Codigo_Centro      int64
Año                 int64
KMeans_Clusters     int32
Gestión             object
Nombre_Centro       object
latitud             float64
longitud            float64
dtype: object
```

▼ Mapa para 2010

```
df_map_2010 = df_map[df_map['Año']<2011]
df_map_2010 = df_map_2010.drop(['Año'],axis = 1)
```

```
latitude = 6.51033052
longitude = -75.48477721
```

```
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=4)
```

```
# set color scheme for the clusters
x = np.arange(2)
ys = [i + x + (i*x)**2 for i in range(4)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lng, cluster, cent, gest in zip(df_map_2010['latitud'],
                                         df_map_2010['longitud'],
                                         df_map_2010['KMeans_Clusters'],
                                         df_map_2010['Nombre_Centro'],
                                         df_map_2010['Gestión']) :

    label = folium.Popup(str(cent) + ' - Cluster ' + str(cluster), parse_html=True)
    folium.vector_layers.CircleMarker(
        [lat, lng],
        radius=10,
        #popup=label,
        tooltip = str(cent) + ' , ' + str(gest),
        color=rainbow[cluster-2],
        fill=True,
        fill_color=rainbow[cluster-2],
        fill_opacity=0.9).add_to(map_clusters)

map_clusters
```



▼ Mapa para 2014

```
df_map_2014 = df_map[df_map['Año']<2015]
df_map_2014 = df_map_2014.drop(['Año'],axis = 1)

latitude = 6.51033052
longitude = -75.48477721

map_clusters = folium.Map(location=[latitude, longitude], zoom_start=4)

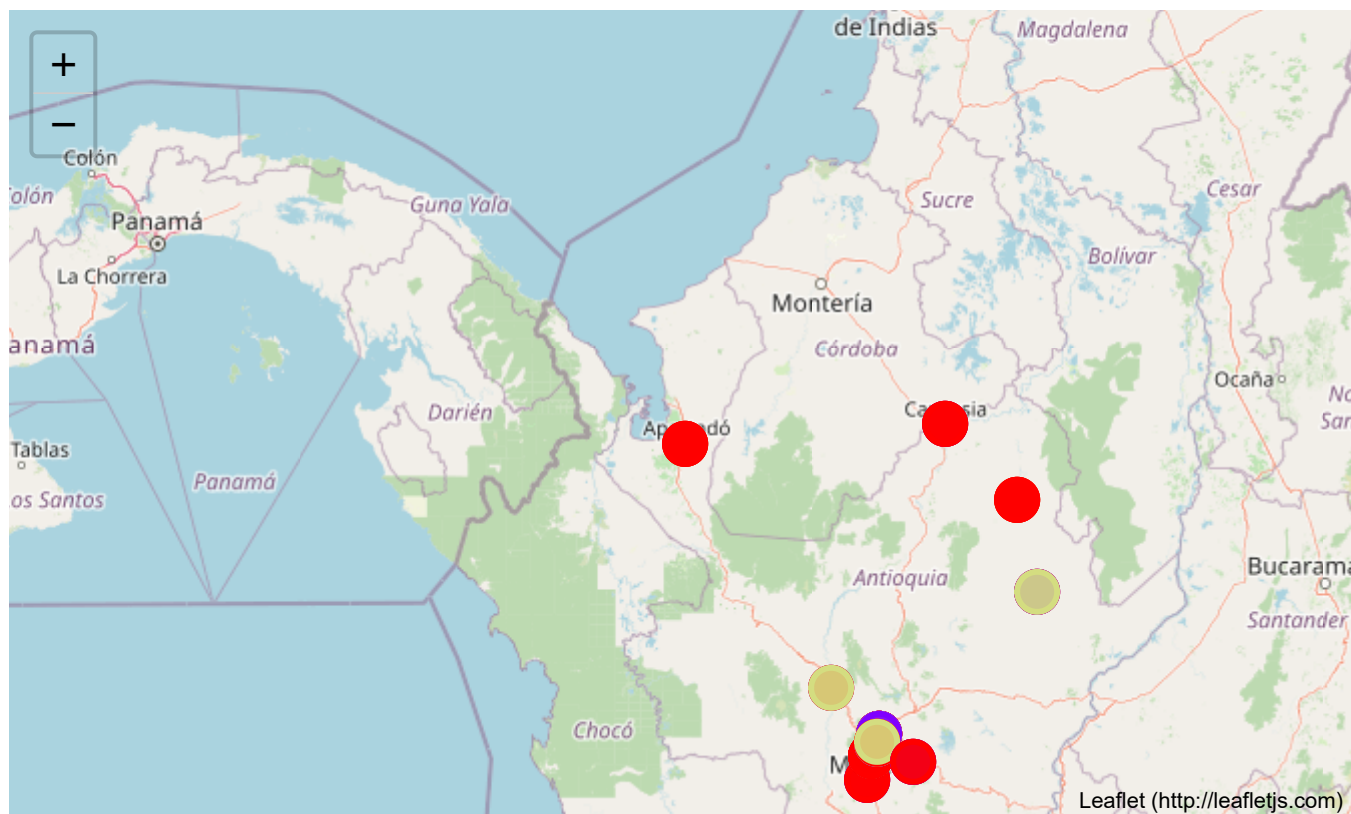
# set color scheme for the clusters
x = np.arange(2)
ys = [i + x + (i*x)**2 for i in range(4)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lng, cluster, cent, gest in zip(df_map_2014['latitud'],
                                         df_map_2014['longitud'],
                                         df_map_2014['KMeans_Clusters'],
                                         df_map_2014['Nombre_Centro'],
                                         df_map_2014['Gestión']) :

    label = folium.Popup(str(cent) + '- Cluster ' + str(cluster), parse_html=True)
    folium.vector_layers.CircleMarker(
        [lat, lng],
        radius=10,
        #popup=label,
        tooltip = str(cent) + ' , ' + str(gest),
        color=rainbow[cluster-2],
        fill=True,
```

```
fill_color=rainbow[cluster-2],
fill_opacity=0.9).add_to(map_clusters)
```

map_clusters



▼ Mapa para 2018

```
df_map_2018 = df_map[df_map['Año']<2019]
df_map_2018 = df_map_2018.drop(['Año'],axis = 1)
```

```
latitude = 6.51033052
longitude = -75.48477721
```

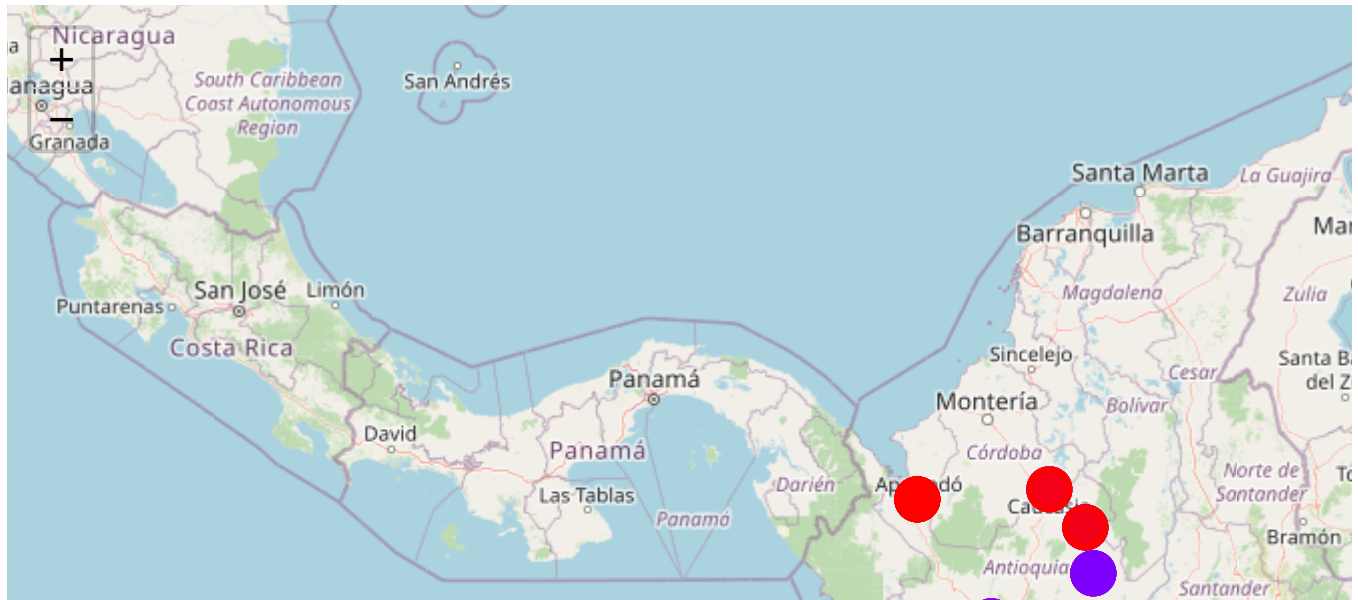
```
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=4)

# set color scheme for the clusters
x = np.arange(2)
ys = [i + x + (i*x)**2 for i in range(4)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lng, cluster, cent, gest in zip(df_map_2018['latitud'],
                                          df_map_2018['longitud'],
                                          df_map_2018['KMeans_Clusters'],
                                          df_map_2018['Nombre_Centro'],
                                          df_map_2018['Gestión']) :

    label = folium.Popup(str(cent) + '- Cluster ' + str(cluster), parse_html=True)
    folium.vector_layers.CircleMarker(
        [lat, lng],
        radius=10,
        #popup=label,
        tooltip = str(cent) + ' , ' + str(gest),
        color=rainbow[cluster-2],
        fill=True,
        fill_color=rainbow[cluster-2],
        fill_opacity=0.9).add_to(map_clusters)

map_clusters
```

▼ Mapa para 2022

```
df_map_2022 = df_map[df_map['Año']<2023]
df_map_2022 = df_map_2022.drop(['Año'],axis = 1)

latitude = 6.51033052
longitude = -75.48477721

map_clusters = folium.Map(location=[latitude, longitude], zoom_start=4)

# set color scheme for the clusters
x = np.arange(2)
ys = [i + x + (i*x)**2 for i in range(4)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lng, cluster, cent, gest in zip(df_map_2022['latitud'],
                                         df_map_2022['longitud'],
                                         df_map_2022['KMeans_Clusters'],
                                         df_map_2022['Nombre_Centro'],
                                         df_map_2022['Gestión']) :

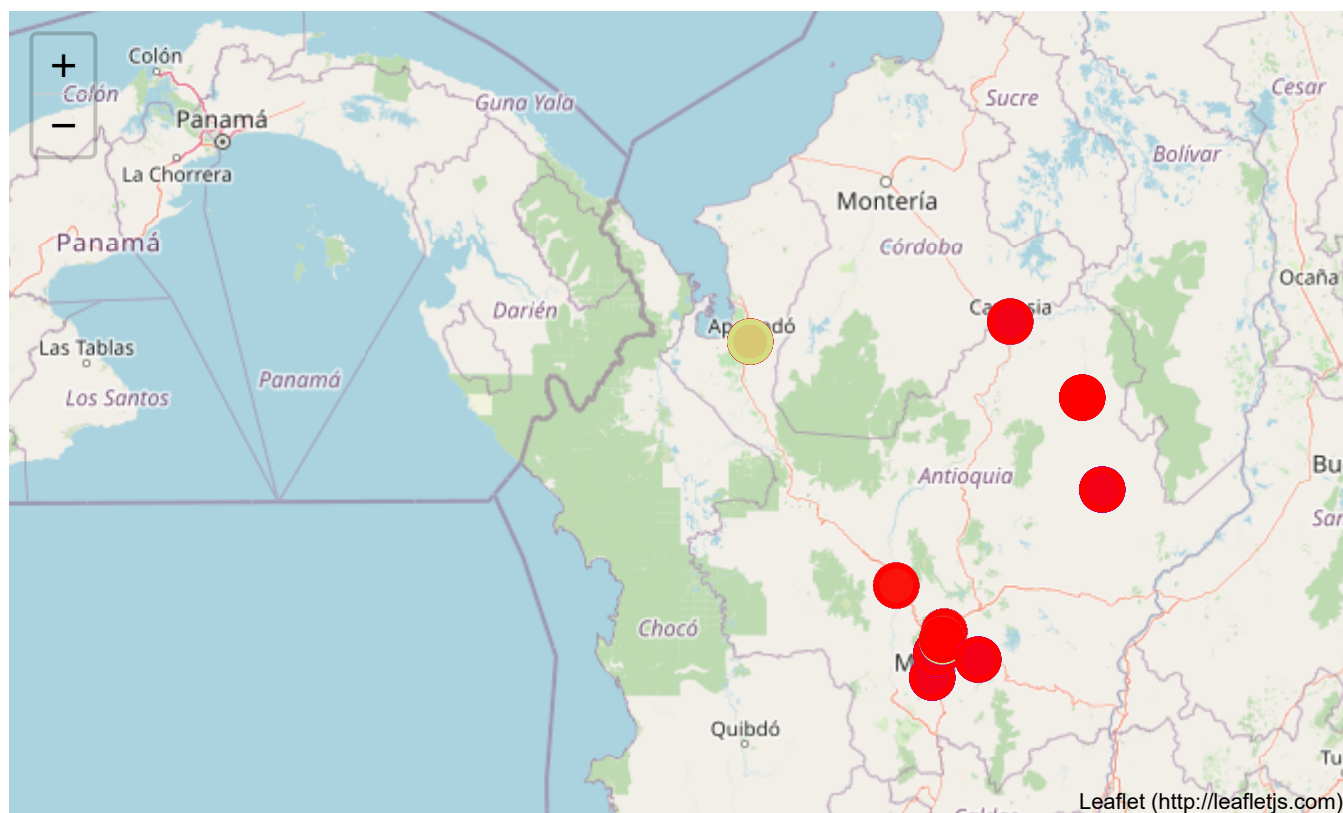
    label = folium.Popup(str(cent) + '- Cluster ' + str(cluster), parse_html=True)
    folium.vector_layers.CircleMarker(
        [lat, lng],
        radius=10,
        #popup=label,
        tooltip = str(cent) + ' , ' + str(gest),
```

```

color=rainbow[cluster-2],
fill=True,
fill_color=rainbow[cluster-2],
fill_opacity=0.9).add_to(map_clusters)

```

map_clusters



```
#from folium import plugins
```

```
#map_hooray = folium.Map(location=[51.5074, 0.1278],
#                             zoom_start = 13)
```

```
# Ensure you're handing it floats
```

```
#df_acc['Latitude'] = df_acc['Latitude'].astype(float)
```

```
#df_acc['Longitude'] = df_acc['Longitude'].astype(float)
```

```

# Filter the DF for rows, then columns, then remove NaNs
#heat_df = df_acc[df_acc['Speed_limit']=='40'] # Reducing data size so it runs faster
#heat_df = df_acc[df_acc['Year']=='2007'] # Reducing data size so it runs faster
#heat_df = heat_df[['Latitude', 'Longitude']]

# Create weight column, using date
#heat_df['Weight'] = df_acc['Date'].str[3:5]
#heat_df['Weight'] = heat_df['Weight'].astype(float)
#heat_df = heat_df.dropna(axis=0, subset=['Latitude','Longitude', 'Weight'])

# List comprehension to make out list of lists
#heat_data = [[[row['Latitude'],row['Longitude']] for index, row in heat_df[heat_df['Weight']]

# Plot it on the map
#hm = plugins.HeatMapWithTime(heat_data,auto_play=True,max_opacity=0.8)
#hm.add_to(map_hooray)
# Display the map
#map_hooray

#import folium
#import folium.plugins as plugins
#import pandas as pd
#import json
#import requests
#import time
#from folium.plugins import HeatMapWithTime
#import datetime

#cumulative = pd.DataFrame()

#mins = input('How many mins do you want to see? ')
#number_of_times = (int(mins)*60)/30

#startTime = time.time()
#for i in range(int(number_of_times)):
#    url = ("https://api.data.gov.sg/v1/transport/taxi-availability")
#    response = requests.get(url)
#    data = response.json()
#    df = pd.io.json.json_normalize(data['features'])
#    coordinateslist = df['geometry.coordinates'].tolist()
#    df1 = pd.DataFrame(coordinateslist)
#    result = df1.transpose()
#    result.columns = ['coordinates']
#    result['Timestamp'] = (df['properties.timestamp'])[0])
#    cumulative = cumulative.append(result)
#    time.sleep(30)
#endTime = time.time()

```

```

#elapsedTime = endTime - startTime
#print("Elapsed Time = %s" % elapsedTime, 'seconds')

#cumulative['coordinates'] = cumulative['coordinates'].astype(str)
#Latitude = []
#Longitude = []
#coordinates = []
#for i in cumulative['coordinates']:
#    i = i.split(", ")
#    lat = i[1][:-1]
#    long = i[0][1:]
#    Latitude.append(lat)
#    Longitude.append(long)
#cumulative['Latitude'] = Latitude
#cumulative['Longitude'] = Longitude

#lat_long_list = []
#for i in cumulative['Timestamp'].unique():
#    temp=[]
#    for index, instance in cumulative[cumulative['Timestamp'] == i].iterrows():
#        temp.append([instance['Latitude'],instance['Longitude']])
#    lat_long_list.append(temp)

#converting it to datetime format
#cumulative['Timestamp']= pd.to_datetime(cumulative['Timestamp'])
#creating a time index
#time_index = []
#for i in cumulative['Timestamp'].unique():
#    time_index.append(i)
#formatting the index
#date_strings = [d.strftime('%d/%m/%Y, %H:%M:%S') for d in time_index]

#Choosing the map type
#m = folium.Map(location=[1.352083,103.819839],zoom_start = 11, tiles="https://tiles.stadiama
#Plot it on the map
#HeatMapWithTime(lat_long_list,radius=5,auto_play=True,position='bottomright',name="cluster",
# Display the map
#m

```

▼ RED NEURONAL: PERCEPTRON

```
df.columns
```

```
Index(['Centro_Año', 'Cantidad_Proyectos', 'Cantidad_Profesional',
      'Cantidad_Maestria', 'Cantidad_Doctorado', 'Total_valor_Proyectos',

```

```
'KMeans_Clusters', 'Gestión', 'Codigo_Centro', 'Año'],
dtype='object')

df_proc_orig = pd.read_excel(path1)
df_proc_orig
df_prc = df[['Codigo_Centro', 'Año', 'Gestión']]

df_per = pd.concat([df_prc,df_proc_orig], axis=1).drop(['Centro_Año'],axis =1)
df_per
```

	Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyec
0	9101	2010	Baja Gestión	1	1	2	0	4.652924e-
1	9127	2010	Baja Gestión	2	2	4	1	3.920052e-
2	9201	2010	Baja Gestión	1	3	3	0	4.556627e-
3	9202	2010	Alta Gestión	3	8	5	0	4.071852e-
4	9203	2010	Baja Gestión	2	6	4	1	2.906175e-
...	...	...	...	...	...	...	...	...
203	9501	2022	Baja Gestión	2	4	4	2	3.754077e-
204	9502	2022	Baja Gestión	1	3	2	0	3.977569e-

```
proceso = df_per['Gestión'].replace({'Baja Gestión': 1, 'Media Gestión':2, 'Alta Gestión':3})
df_per['Gestión'] = pd.concat([proceso], axis=1)
df_per
```

	Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyec
0	9101	2010	1	1	1	2	0	4.652924e-
1	9127	2010	1	2	2	4	1	3.920052e-
2	9201	2010	1	1	3	3	0	4.556627e-
3	9202	2010	3	3	8	5	0	4.071852e-

```
#Clases
```

```
y = (df_per['Gestión']).astype(np.int8)
```

```
print(y)
```

```
0      1
1      1
2      1
3      3
4      1
```

```
..
203    1
204    1
205    1
206    2
207    1
```

```
Name: Gestión, Length: 208, dtype: int8
```

Dividimos los datos en un conjunto de aprendizaje y de prueba:

```
from sklearn.model_selection import train_test_split
datasets = train_test_split(df_per,
                             y,
                             test_size=0.3)
```

```
X_train, X_test, y_train, y_test = datasets
```

Ahora, creamos una instancia de Perceptron y ajustamos los datos de entrenamiento:

```
from sklearn.linear_model import Perceptron
p = Perceptron(random_state=42,
                max_iter=100,
                tol=0.001)
p.fit(X_train, y_train)

Perceptron(max_iter=100, random_state=42)
```

Recordando un poco sobre la matriz de confusión

		Actual Values	
		Yes	No
Predicted Values	Yes	True Positive	False Positive
	No	False Negative	True Negative

```

from sklearn.metrics import classification_report
y_perc_pred = p.predict(X_test)
print(classification_report(y_test, y_perc_pred))

```

	precision	recall	f1-score	support
1	0.56	1.00	0.71	35
2	0.00	0.00	0.00	7
3	0.00	0.00	0.00	21
accuracy			0.56	63
macro avg	0.19	0.33	0.24	63
weighted avg	0.31	0.56	0.40	63

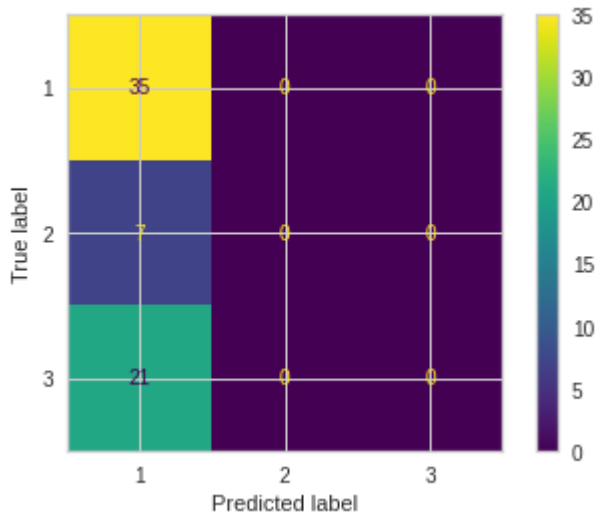
```

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
    _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
    _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
    _warn_prf(average, modifier, msg_start, len(result))

```

```
plot_confusion_matrix(p, X_test, y_test)
plt.show()
```

/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
warnings.warn(msg, category=FutureWarning)



```
path3 = '/content/drive/MyDrive/Proyecto Sennova borrador/sample_procesos.xlsx'
df_sample = pd.read_excel(path3)
df_sample
```

	Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyect
0	9101.0	2010.0	1.0	1.0	1.0	2.0	0.0	4.652924e+
1	9127.0	2010.0	1.0	2.0	2.0	4.0	1.0	3.920052e+
2	9201.0	2010.0	1.0	1.0	3.0	3.0	0.0	4.556627e+
3	9202.0	2010.0	1.0	3.0	8.0	5.0	0.0	4.071852e+
4	9203.0	2010.0	1.0	2.0	6.0	4.0	1.0	2.906175e+
5	9501.0	2022.0	1.0	2.0	4.0	4.0	2.0	3.754077e+
6	9502.0	2022.0	1.0	1.0	3.0	2.0	0.0	3.977569e+
7	9503.0	2022.0	1.0	1.0	0.0	0.0	1.0	5.068536e+
8	9504.0	2022.0	1.0	4.0	7.0	7.0	1.0	7.614273e+
9	9549.0	2022.0	1.0	2.0	5.0	2.0	1.0	3.294535e+

```
y_percep_perd = p.predict(df_sample)
y_percep_perd_s= pd.Series(y_percep_perd)
df_percep = df_sample[['Codigo_Centro','Año','Gestión']]
df_percep['Gestión'] = y_percep_perd_s.replace({1: 'Baja Gestión',2: 'Media Gestión', 3: 'Alt
df_percep
```


/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>
after removing the cwd from sys.path.

	Codigo_Centro	Año	Gestión
0	9101.0	2010.0	Baja Gestión
1	9127.0	2010.0	Baja Gestión
2	9201.0	2010.0	Baja Gestión
3	9202.0	2010.0	Baja Gestión
4	9203.0	2010.0	Baja Gestión
5	9501.0	2022.0	Baja Gestión
6	9502.0	2022.0	Baja Gestión
7	9503.0	2022.0	Baja Gestión
8	9504.0	2022.0	Baja Gestión
9	9549.0	2022.0	Baja Gestión

▼ Regresión Logística

```
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.model_selection import train_test_split
datasets = train_test_split(df_per,
                            y,
                            test_size=0.3)
```

```
X_train, X_test, y_train, y_test = datasets
```

```
# Instanciando a reg. logistica
reglog = LogisticRegression()
```

```
# Aplicando el modelo
reglog.fit(X_train, y_train)
```

```
LogisticRegression()
```

```
LogisticRegression()
```

```
LogisticRegression()
```

```
# Predicción
y_reglog_pred = reglog.predict(X_test)

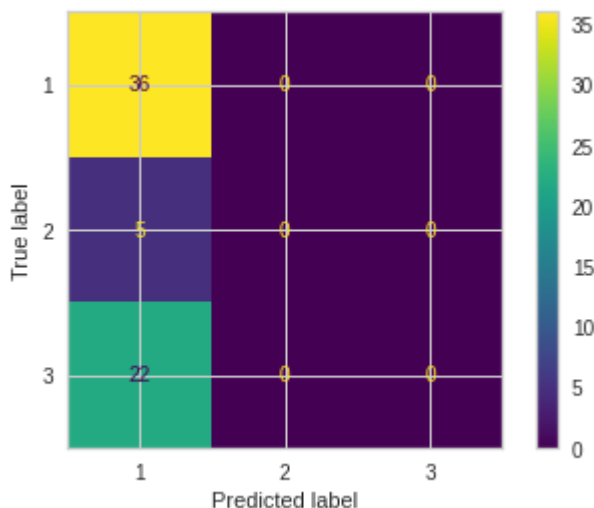
print(classification_report(y_test,y_reglog_pred))
```

	precision	recall	f1-score	support
1	0.57	1.00	0.73	36
2	0.00	0.00	0.00	5
3	0.00	0.00	0.00	22
accuracy			0.57	63
macro avg	0.19	0.33	0.24	63
weighted avg	0.33	0.57	0.42	63

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
  _warn_prf(average, modifier, msg_start, len(result))
```

```
# Matriz de confusión
plot_confusion_matrix(reglog, X_test, y_test)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning:
  warnings.warn(msg, category=FutureWarning)
```



Test con valores de muestra

```
df_sample.drop(['Gestión'], axis = 1)
```

	Codigo_Centro	Año	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
0	9101.0	2010.0	1.0	1.0	2.0	0.0	4.652924e+14
1	9127.0	2010.0	2.0	2.0	4.0	1.0	3.920052e+14
2	9201.0	2010.0	1.0	3.0	3.0	0.0	4.556627e+14
3	9202.0	2010.0	3.0	8.0	5.0	0.0	4.071852e+14
4	9203.0	2010.0	2.0	6.0	4.0	1.0	2.906175e+14
5	9501.0	2022.0	2.0	4.0	4.0	2.0	3.754077e+14
6	9502.0	2022.0	1.0	3.0	2.0	0.0	3.977569e+14
7	9503.0	2022.0	1.0	0.0	0.0	1.0	5.068536e+14
8	9504.0	2022.0	4.0	7.0	7.0	1.0	7.614273e+14
9	9549.0	2022.0	2.0	5.0	2.0	1.0	3.294535e+14

```
y_reglog_pred = reglog.predict(df_sample)
y_reglog_pred = pd.Series(y_reglog_pred)
```

```
df_logist = df_sample[['Codigo_Centro', 'Año', 'Gestión']]
df_logist['Productividad'] = y_reglog_pred.replace({1: 'Baja Gestión', 2: 'Media Gestión', 3:
df_logist
```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>

"""

	Codigo_Centro	Año	Gestión	Productividad
0	9101.0	2010.0	1.0	Baja Gestión
1	9127.0	2010.0	1.0	Baja Gestión
2	9201.0	2010.0	1.0	Baja Gestión
3	9202.0	2010.0	1.0	Baja Gestión
4	9203.0	2010.0	1.0	Baja Gestión
5	9501.0	2022.0	1.0	Baja Gestión
6	9502.0	2022.0	1.0	Baja Gestión
7	9503.0	2022.0	1.0	Baja Gestión
8	9504.0	2022.0	1.0	Baja Gestión
9	9549.0	2022.0	1.0	Baja Gestión

▼ Random Forest

```
from sklearn.ensemble import RandomForestClassifier
```

```
rf = RandomForestClassifier()
```

```
rf.fit(X_train, y_train)
```

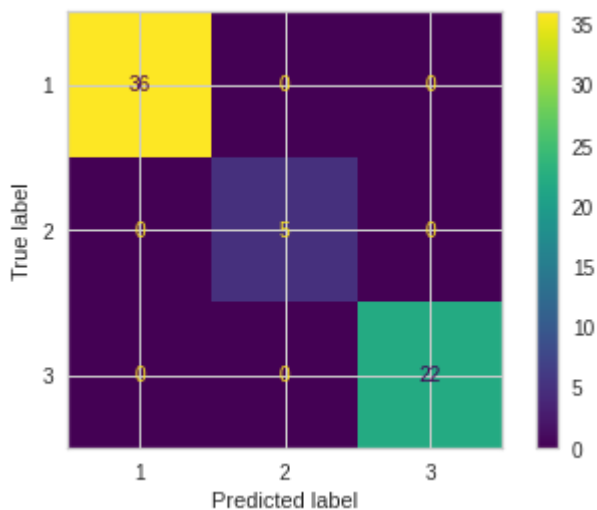
```
y_dt_pred = rf.predict(X_test)
print(classification_report(y_test,y_dt_pred))
```

	precision	recall	f1-score	support
1	1.00	1.00	1.00	36
2	1.00	1.00	1.00	5
3	1.00	1.00	1.00	22
accuracy			1.00	63
macro avg	1.00	1.00	1.00	63
weighted avg	1.00	1.00	1.00	63

```
# Matriz de confusión
```

```
plot_confusion_matrix(rf, X_test, y_test)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
warnings.warn(msg, category=FutureWarning)
```



Test con valores de muestra

```
df_sample
```

	Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyect
0	9101.0	2010.0	1.0	1.0	1.0	2.0	0.0	4.652924e+
1	9127.0	2010.0	1.0	2.0	2.0	4.0	1.0	3.920052e+
2	9201.0	2010.0	1.0	1.0	3.0	3.0	0.0	4.556627e+
3	9202.0	2010.0	1.0	3.0	8.0	5.0	0.0	4.071852e+
4	9203.0	2010.0	1.0	2.0	6.0	4.0	1.0	2.906175e+
5	9501.0	2022.0	1.0	2.0	4.0	4.0	2.0	3.754077e+
6	9502.0	2022.0	1.0	1.0	3.0	2.0	0.0	3.977569e+
7	9503.0	2022.0	1.0	1.0	0.0	0.0	1.0	5.068536e+
8	9504.0	2022.0	1.0	4.0	7.0	7.0	1.0	7.614273e+
9	9549.0	2022.0	1.0	2.0	5.0	2.0	1.0	3.294535e+

```
y_dt_pred_ram = rf.predict(df_sample)
y_dt_pred_ram = pd.Series(y_dt_pred_ram)
```

```
df_rd_forest = df_sample[['Codigo_Centro', 'Año', 'Gestión']]
df_rd_forest['Productividad'] = y_dt_pred_ram.replace({1: 'Baja Gestión', 2: 'Media Gestión',
df_rd_forest
```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:

▼ SVM

```

"""
#Escalamos el valor de costo total de proyectos
from sklearn.svm import SVC
scaler = StandardScaler()
X_train[['TV_Proyectos']] = scaler.fit_transform(X_train[['TV_Proyectos']])
X_train

```

	Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyec
14	9504	2010	1	1	2	1	1	-1.043
124	9502	2017	3	4	4	9	1	0.160
186	9402	2021	1	1	4	1	1	-0.257
109	9503	2016	1	1	0	2	0	-1.153
59	9501	2013	1	2	4	2	2	-0.632
...	...	...	...	...	...	...	...	...
80	9101	2015	1	2	6	5	0	-1.477
200	9301	2022	1	2	2	3	0	-0.663
74	9402	2014	2	5	7	8	2	3.256
207	9549	2022	1	2	5	2	1	-1.265
160	9101	2020	3	4	5	7	1	2.115

145 rows × 8 columns

```

y_svm = SVC(C = 100, kernel = 'linear', random_state=123)
y_svm.fit(X_train, y_train)

```

```
SVC(C=100, kernel='linear', random_state=123)
```

```

y_svm_pred = y_svm.predict(X_test)
print(classification_report(y_test,y_svm_pred))

```

	precision	recall	f1-score	support
1	0.00	0.00	0.00	36
2	0.08	1.00	0.15	5
3	0.00	0.00	0.00	22
accuracy			0.08	63

macro avg	0.03	0.33	0.05	63
weighted avg	0.01	0.08	0.01	63

```

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefined
_warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefined
_warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefined
_warn_prf(average, modifier, msg_start, len(result))

```

Test con valores de muestra

```

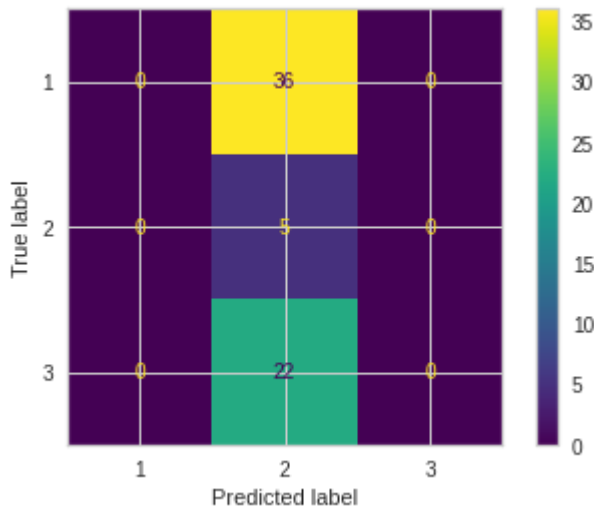
# Matriz de confusión
plot_confusion_matrix(y_svm, X_test, y_test, include_values =True)
plt.show()

```

```

/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
warnings.warn(msg, category=FutureWarning)

```



```

y_dt_pred_svm = pd.Series(y_svm.predict(df_sample))
df_SVM = df_sample[['Codigo_Centro', 'Año', 'Gestión']]
df_SVM['Productividad'] = y_dt_pred_svm.replace({1: 'Baja Gestión', 2: 'Media Gestión', 3: 'Alta Gestión'})
df_SVM

```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>
This is separate from the ipykernel package so we can avoid doing imports until

	Codigo_Centro	Año	Gestión	Productividad
0	9101.0	2010.0	1.0	Media Gestión
1	9127.0	2010.0	1.0	Media Gestión
2	9201.0	2010.0	1.0	Media Gestión
3	9202.0	2010.0	1.0	Media Gestión

▼ Reporte de parametros de validación de modelos

6 9502.0 2022.0 1.0 Media Gestión

```
from sklearn.metrics import accuracy_score, confusion_matrix, f1_score, precision_score, recall_score
```

```
p.predict(X_test)
```

```
reglog.predict(X_test)
```

```
rf.predict(X_test)
```

```
y_svm.predict(X_test)
```

```
y_test
```

Reporte de parámetros de validación

```
# perceptrón
```

```
acc_perc = accuracy_score(y_test, p.predict(X_test))
```

```
f1_perc = f1_score(y_test, p.predict(X_test), average='weighted')
```

```
#auc_perc = roc_auc_score(y_test, p.predict(X_test))
```

```
#recall_perc = recall_score(y_test, p.predict(X_test))
```

```
# Regresión logística
```

```
acc_log = accuracy_score(y_test, reglog.predict(X_test))
```

```
f1_log = f1_score(y_test, reglog.predict(X_test), average='weighted')
```

```
#Arboles de decisión
```

```
acc_rf = accuracy_score(y_test, rf.predict(X_test))
```

```
f1_rf = f1_score(y_test, rf.predict(X_test), average='weighted')
```

```
#Máquinas de soporte vectorial
```

```
acc_svm = accuracy_score(y_test, y_svm.predict(X_test))
```

```
f1_svm = f1_score(y_test, y_svm.predict(X_test), average='weighted')
```



```
Parametro = ['Accuracy', 'F1-Score']
Percep= [acc_perc,f1_perc]
r_log = [acc_log,f1_log]
rfor = [acc_rf,f1_rf]
svm_t = [acc_svm,f1_svm]

df_valid = pd.DataFrame()
df_valid['Parámetro'] = Parametro
df_valid['Perceptrón'] = Percep
df_valid['Regresión logística'] = r_log
df_valid['Random Forest'] = rfor
df_valid['Máquina de soporte vectorial'] = svm_t
```

df_valid

	Parámetro	Perceptrón	Regresión logística	Random Forest	Máquina de soporte vectorial
0	Accuracy	0.571429	0.571429	1.0	0.079365
1	F1-Score	0.415584	0.415584	1.0	0.011671

```
df_valid.to_excel('Validacion_Procesos.xlsx',index =False)
```

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

```
import folium # plotting library
from folium import plugins
from folium.plugins import MarkerCluster
```

```
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import sklearn.metrics as sm
from sklearn.decomposition import PCA
from sklearn.cluster import AgglomerativeClustering
from sklearn.cluster import DBSCAN
from io import IncrementalNewlineDecoder
import scipy.cluster.hierarchy as sch
```

```
#Cluster
from yellowbrick.cluster import SilhouetteVisualizer
from yellowbrick.cluster import KElbowVisualizer
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs
```

```
#Graficas
import matplotlib.pyplot as plt
from matplotlib import pyplot as plot
import matplotlib.cm as cm
import matplotlib.colors as colors
from plotly import graph_objects as go
from plotly.subplots import make_subplots
import plotly.express as px
import seaborn as sns
import branca
from sklearn.metrics import plot_confusion_matrix
```

```
from google.colab import drive
drive.mount('/content/drive')
```

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.moun



```
from google.colab import files
path = "/content/drive/MyDrive/Proyecto SGPS 9748/Datos.xlsx"
path2 = '/content/drive/MyDrive/Proyecto SGPS 9748/PCPA.xlsx'
```

```
#Dataframes
principal_proyectos= pd.read_excel(path,sheet_name= 'Principal_Proyectos')
principal_talento_humano = pd.read_excel(path,sheet_name= 'Principal_Talento_Humano')
```

```
principal_productos = pd.read_excel(path, sheet_name= 'Principal_Productos')
df_product_orig = pd.read_excel(path2)
df_centros = pd.read_excel(path, sheet_name= 'Centros_Formacion')
```

```
df_product_orig
```

	Centro_Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101-2010	1	1	0	0	0	1
1	9127-2010	0	0	0	0	1	0
2	9201-2010	0	0	0	0	2	0
3	9202-2010	2	1	0	0	2	0
4	9203-2010	1	1	0	1	1	1
...	...	...	...	...	...	...	...
203	9501-2022	1	1	1	2	3	1
204	9502-2022	2	1	3	1	3	1
205	9503-2022	3	2	4	1	1	2
206	9504-2022	1	2	2	4	1	4
207	9549-2022	5	3	1	2	2	4

208 rows × 7 columns

```
df_product_orig.columns
```

```
Index(['Centro_Año', 'P_8020', 'P_8025', 'P_8019', 'P_8039', 'P_8016',
      'P_8031'],
      dtype='object')
```

```
df_product_orig.columns = ['Centro_Año', 'Apropiación_Social', 'Articulo_Sometido_en_revista_
      'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_p
      'Informe_de_investigación', 'Libros_resultados_de_investigación', 'Productos_
df= df_product_orig
df
```

	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación
0	9101-2010	1	1	
1	9127-2010	0	0	
2	9201-2010	0	0	
3	9202-2010	2	1	
4	9203-2010	1	1	
...	...	...	...	
203	9501-2022	1	1	
204	9502-2022	2	1	
205	9503-2022	3	2	
206	9504-2022	1	2	

df.columns

```
Index(['Centro_Año', 'Apropiación_Social',
      'Articulo_Sometido_en_revista_Indexada',
      'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_producción',
      'Informe_de_investigación', 'Libros_resultados_de_investigación',
      'Productos_tecnológicos_certificados_o_validados'],
      dtype='object')
```

df.describe()

	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación_Pública_d
count	208.000000	208.000000	
mean	1.836538	1.778846	
std	1.376901	1.483935	
min	0.000000	0.000000	
25%	1.000000	1.000000	
50%	2.000000	1.000000	
75%	3.000000	3.000000	
max	7.000000	7.000000	

```
from sklearn.preprocessing import StandardScaler
```

```

scaler = StandardScaler()
df[['Apropiación_Social',
    'Articulo_Sometido_en_revista_Indexada',
    'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion
    'Informe_de_investigación', 'Libros_resultados_de_investigación',
    'Productos_tecnológicos_certificados_o_validados']] = scaler.fit_transform(df[['Apropi
    'Articulo_Sometido_en_revista_Indexada',
    'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion
    'Informe_de_investigación', 'Libros_resultados_de_investigación',
    'Productos_tecnológicos_certificados_o_validados']])
df

```

	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación
0	9101-2010	-0.609017		-0.526118
1	9127-2010	-1.337038		-1.201628
2	9201-2010	-1.337038		-1.201628
3	9202-2010	0.119003		-0.526118
4	9203-2010	-0.609017		-0.526118
...	...	...		...
203	9501-2022	-0.609017		-0.526118
204	9502-2022	0.119003		-0.526118
205	9503-2022	0.847024		0.149392
206	9504-2022	-0.609017		0.149392
207	9549-2022	2.303066		0.824901

208 rows × 7 columns

```

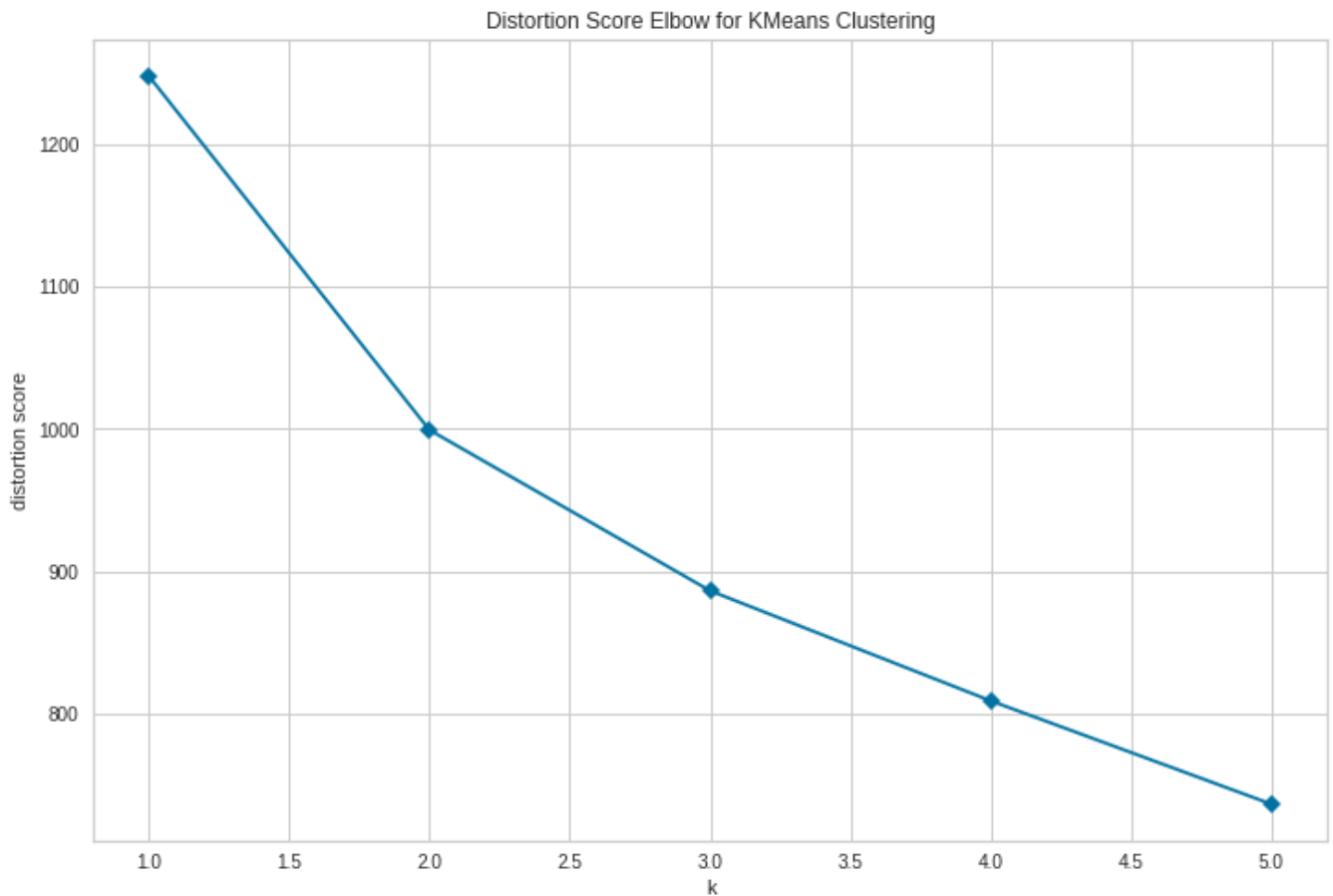
#Búsqueda de la cantidad óptima de clusters
##Calculando qué tan similares son los individuos dentro de los Custers
y = df[['Centro_Año']]
x = df.drop(['Centro_Año'], axis= 1)
wcss = []
for i in range (2,6):
    kmeans = KMeans(n_clusters=i, max_iter=300)
    kmeans.fit(x) #Aplico K-means a la base de datos
    wcss.append(kmeans.inertia_)

```

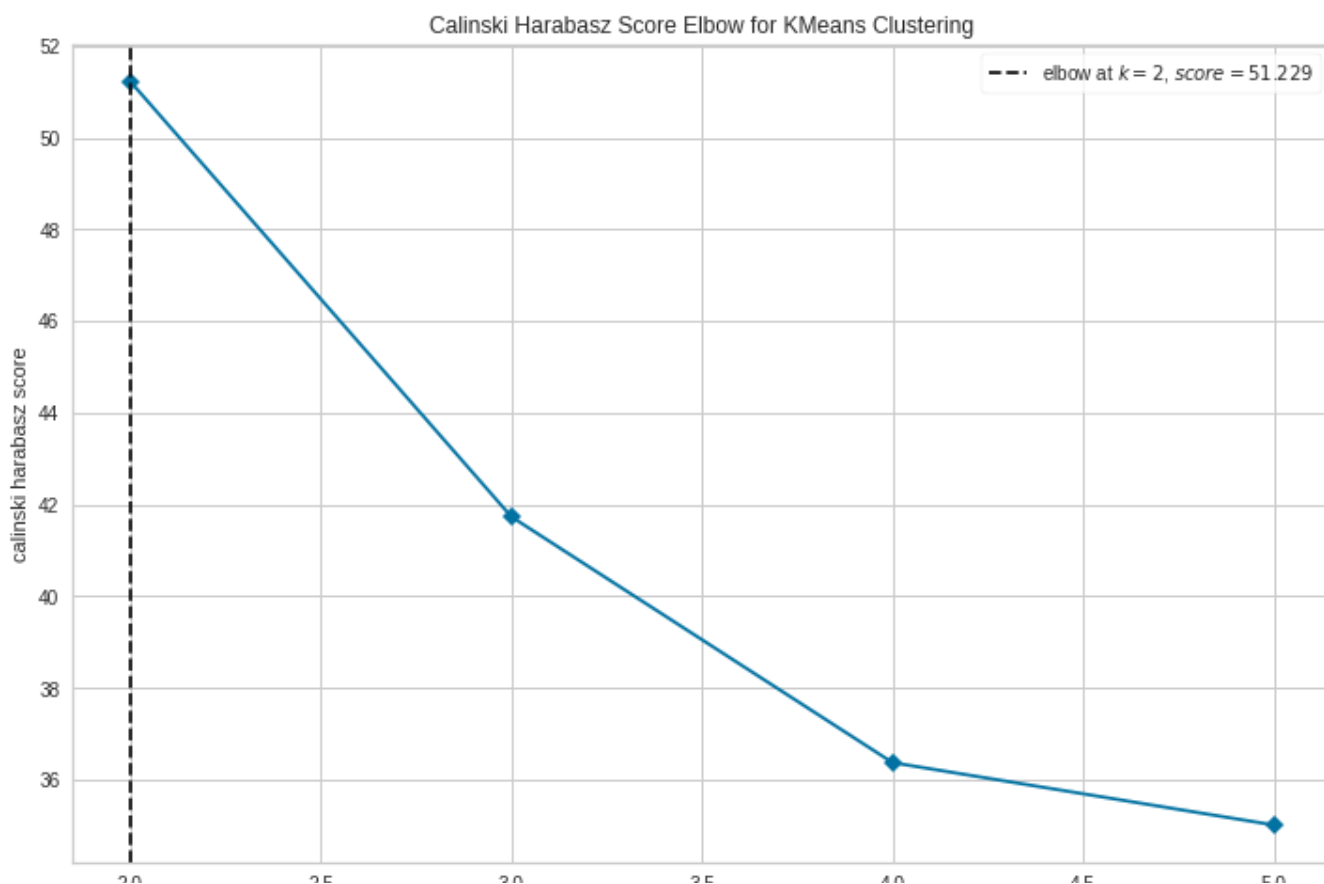
distortion: suma media de las distancias al cuadrado de los centroides (Error medio cuadrático)

```
#km_transformed = mms.transform(km)
plt.figure(figsize=(12,8))
model = KMeans(random_state=42)
visualizer = KElbowVisualizer(model, k=(1,6), timings=False)
visualizer.fit(x)
visualizer.show();
```

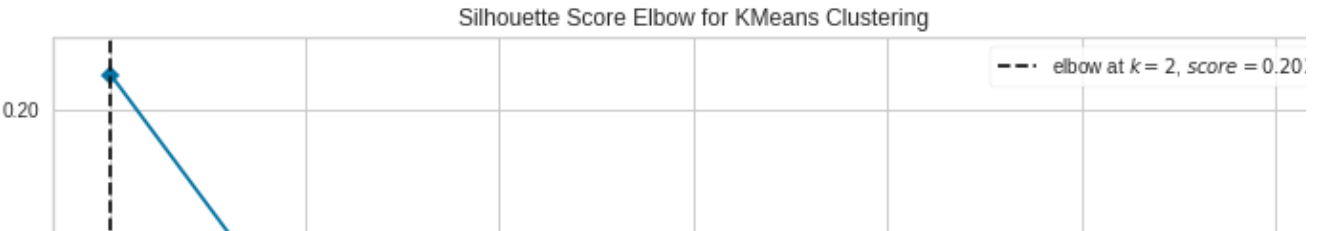
```
/usr/local/lib/python3.7/dist-packages/yellowbrick/utils/kneed.py:155: YellowbrickWarning:
warnings.warn(warning_message, YellowbrickWarning)
/usr/local/lib/python3.7/dist-packages/yellowbrick/cluster/elbow.py:361: YellowbrickWarning:
warnings.warn(warning_message, YellowbrickWarning)
```



```
plt.figure(figsize=(12,8))
model = KMeans()
visualizer = KElbowVisualizer( model, k=(2,6), metric='calinski_harabasz', timings=False)
visualizer.fit(x)          # Fit the data to the visualizer
visualizer.show()          # Finalize and render the figure
```



```
plt.figure(figsize=(12,8))
model = KMeans()
visualizer = KElbowVisualizer(model, k=(2,6), metric='silhouette', timings=False)
visualizer.fit(x)          # Fit the data to the visualizer
visualizer.show()          # Finalize and render the figure
```



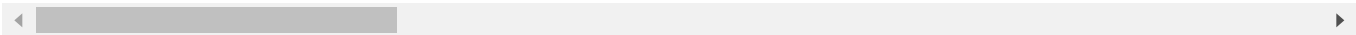
```
#Aplicando el método de k-means a la base de datos
clustering=KMeans(n_clusters=3, max_iter=300,init = 'random') #Crea el modelo
clustering.fit(x) #Aplica el modelo a la base de datos
clustering.cluster_centers_
```

```
array([[ -0.49031832, -0.62157072, -0.57310177, -0.57004169, -0.05491146,
        -0.33027296],
       [ 0.11900339,  0.24067671,  0.63618449,  0.85846574, -0.2914806 ,
        -0.18297187],
       [ 0.86435797,  0.93748642,  0.13446929, -0.26387211,  0.6338433 ,
        1.04583406]])
```

```
#Agregando la clasificación en el archivo original
df["KMeans_Clusters"]=clustering.labels_
df
```

	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación
0	9101-2010	-0.609017		-0.526118
1	9127-2010	-1.337038		-1.201628
2	9201-2010	-1.337038		-1.201628
3	9202-2010	0.119003		-0.526118
4	9203-2010	-0.609017		-0.526118
...	...	...		...
203	9501-2022	-0.609017		-0.526118
204	9502-2022	0.119003		-0.526118
205	9503-2022	0.847024		0.149392
206	9504-2022	-0.609017		0.149392
207	9549-2022	2.303066		0.824901

208 rows × 8 columns



▼ Validación de clusters

```
#Calcular agrupación de índices de validación externa con KMeans
labels_true = df["Centro_Año"]
rotulos_km = clustering.labels_
h = sm.homogeneity_score(labels_true, rotulos_km)
print("Homogeneidad: %0.3f" % h)
c = sm.completeness_score(labels_true, rotulos_km)
print("Compleitud: %0.3f" % c)
v = sm.v_measure_score(labels_true, rotulos_km)
print("Medida V: %0.3f" % v)
ari = sm.adjusted_rand_score(labels_true, rotulos_km)
print("Índice Rand corregido: %0.3f" % ari)
ami = sm.adjusted_mutual_info_score(labels_true, rotulos_km, average_method='arithmetic')
print("Adjusted Mutual Information: %0.3f" % ami)
fm = sm.fowlkes_mallows_score(labels_true, rotulos_km)
print("Índice Fowlkes-Mallows: %0.3f" % fm)
```

```
Homogeneidad: 0.197
Compleitud: 1.000
Medida V: 0.329
Índice Rand corregido: 0.000
Adjusted Mutual Information: 0.000
Índice Fowlkes-Mallows: 0.000
```

```
from sklearn.metrics import silhouette_score
#silhouette_avg = silhouette_score(x, clustering.labels_)
s = sm.silhouette_score(x, clustering.labels_, metric='euclidean')
print("Coeficiente de Silhueta: %0.3f" % s)
ch = sm.calinski_harabasz_score(x, clustering.labels_)
print("Calinski_harabasz: %0.3f" % ch)
dbs = sm.davies_bouldin_score(x, clustering.labels_)
print("Davies Bouldin: %0.3f" % dbs)
```

```
Coeficiente de Silhueta: 0.173
Calinski_harabasz: 41.791
Davies Bouldin: 1.893
```

▼ DBSCAN CLUSTERING

```
x.head()
```


	Componente_1	Componente_2	KMeans_Clusters	Centro_Año
0	-1.991854	0.502028	0	9101-2010
1	-2.773852	0.307726	0	9127-2010
2	-2.595027	0.595137	0	9201-2010
3	-1.549107	0.921996	0	9202-2010
4	-1.551687	0.336304	0	9203-2010
...	...	...	...	...
203	-0.654792	0.122801	0	9501-2022
204	-0.003204	0.018650	0	9502-2022
205	0.877874	-0.409651	1	9503-2022
206	0.935897	-0.788964	1	9504-2022
207	2.074298	1.292960	2	9549-2022

208 rows × 4 columns

```
pca_productividad_3 = pca_productividad_3.dropna()
pca_productividad_3['KMeans_Clusters'] = pca_productividad_3['KMeans_Clusters'].astype(int)
pca_productividad_3
```

	Componente_1	Componente_2	KMeans_Clusters	Centro_Año
0	-1.991854	0.502028	0	9101-2010
1	-2.773852	0.307726	0	9127-2010
2	-2.595027	0.595137	0	9201-2010
3	-1.549107	0.921996	0	9202-2010
4	-1.551687	0.336304	0	9203-2010
...	...	...	...	...
203	-0.654792	0.122801	0	9501-2022
204	-0.003204	0.018650	0	9502-2022
205	0.877874	-0.409651	1	9503-2022
206	0.935897	-0.788964	1	9504-2022
207	2.074298	1.292960	2	9549-2022

208 rows × 4 columns

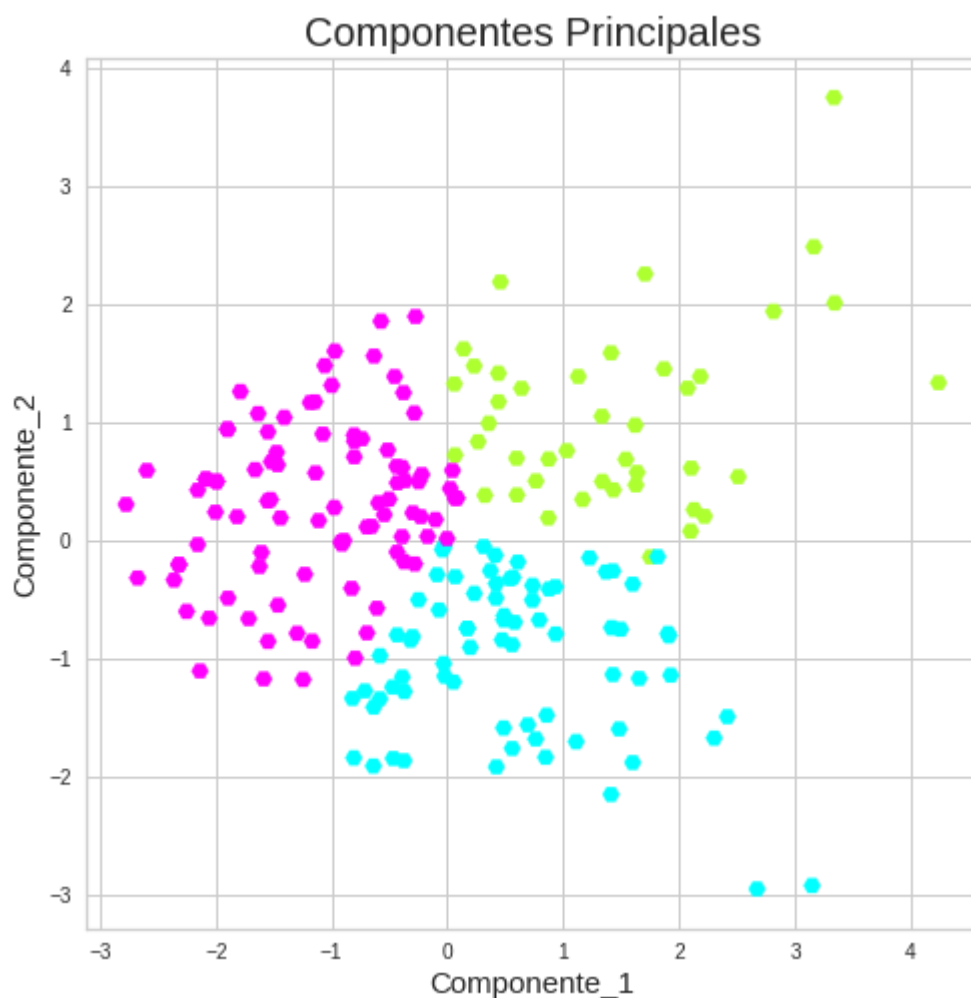
```
fig=plt.figure(figsize=(8,8))
```

```

ax=fig.add_subplot(1,1,1)
ax.set_xlabel("Componente_1",fontsize=15)
ax.set_ylabel("Componente_2", fontsize=15)
ax.set_title("Componentes Principales", fontsize=20)
color_theme=np.array(["MAGENTA","CYAN",'GREENYELLOW','mediumpurple'])
ax.scatter(x=pca_productividad_3['Componente_1'], y=pca_productividad_3['Componente_2'],c=col
#for i, label in enumerate(annotations):
#    plt.annotate(label, (pca_productividad_3['KMeans_Clusters'][i], pca_productividad_3['KMe
plt.show()

```

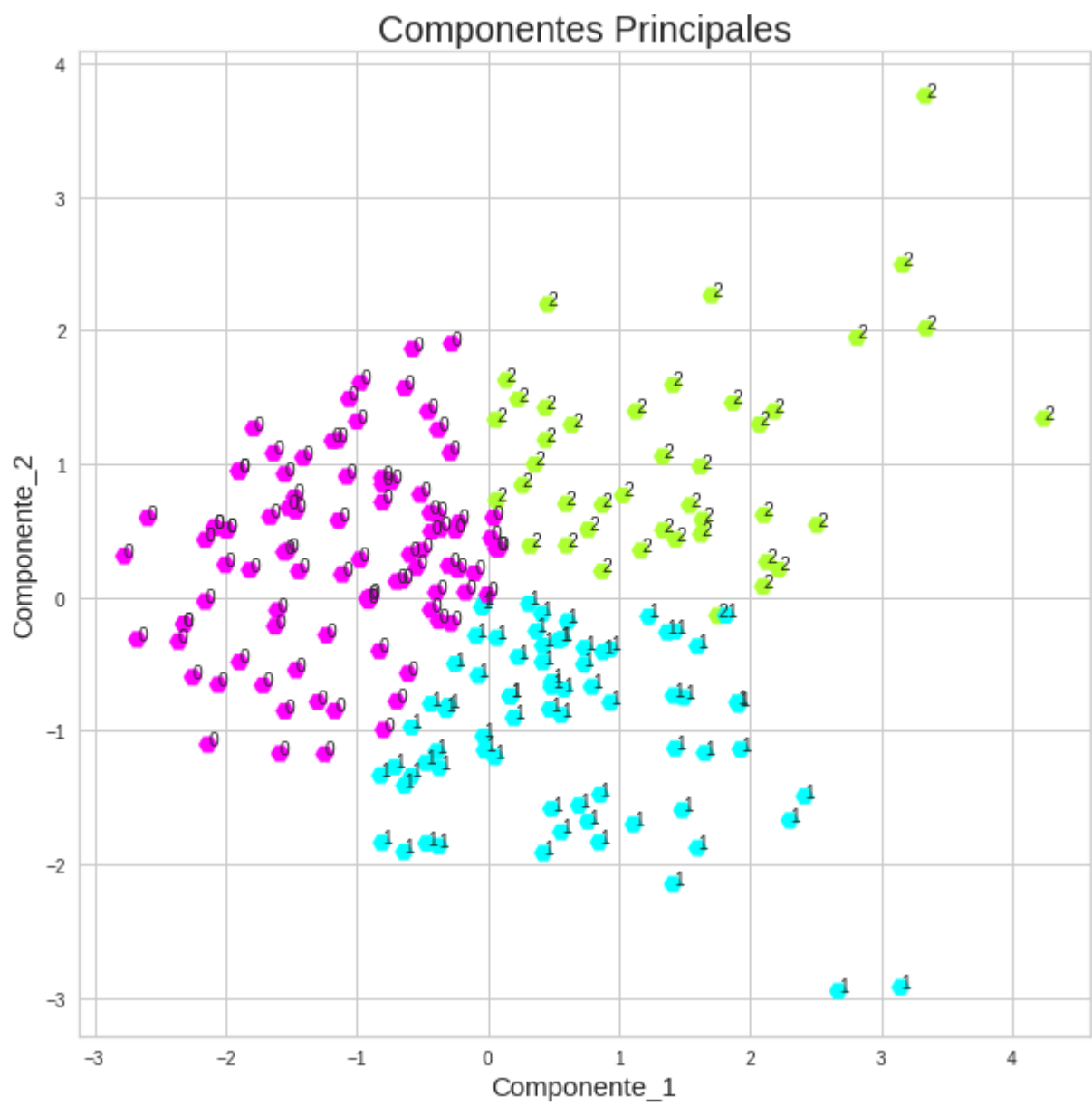
<matplotlib.collections.PathCollection at 0x7fa2af766890>



```

#revisar tamaño de etiqueta de punto
fig=plt.figure(figsize=(10,10))
ax=fig.add_subplot(1,1,1)
annotations = pca_productividad_3['KMeans_Clusters']
ax.set_xlabel("Componente_1",fontsize=15)
ax.set_ylabel("Componente_2", fontsize=15)
ax.set_title("Componentes Principales", fontsize=20)
color_theme=np.array(["MAGENTA","CYAN",'GREENYELLOW','mediumpurple'])
ax.scatter(x=pca_productividad_3['Componente_1'], y=pca_productividad_3['Componente_2'],c=col
for i, label in enumerate(annotations):
    plt.annotate(label, (pca_productividad_3['Componente_1'][i], pca_productividad_3['Compone
plt.show()

```



df

	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación
0	9101-2010	-0.609017		-0.526118
1	9127-2010	-1.337038		-1.201628
2	9201-2010	-1.337038		-1.201628
3	9202-2010	0.119003		-0.526118
4	9203-2010	-0.609017		-0.526118

```
pip install mlxtend
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: mlxtend in /usr/local/lib/python3.7/dist-packages (0.20.6)
Requirement already satisfied: pandas>=0.24.2 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.3.4)
Requirement already satisfied: scikit-learn>=1.0.2 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.0.2)
Requirement already satisfied: joblib>=0.13.2 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.1.0)
Requirement already satisfied: numpy>=1.16.2 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.21.0)
Requirement already satisfied: scipy>=1.2.1 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.7.3)
Requirement already satisfied: matplotlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (3.5.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from mlxtend) (59.0.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib>=3.0.0->mlxtend) (1.4.5)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib>=3.0.0->mlxtend) (2.8.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib>=3.0.0->mlxtend) (0.11.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib>=3.0.0->mlxtend) (3.0.9)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (from pandas>=0.24.2->mlxtend) (4.1.1)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.24.2->mlxtend) (2022.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.1->mlxtend) (1.16.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=1.0.2->mlxtend) (3.1.0)
```

```
pip install mlxtend --upgrade --no-deps
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: mlxtend in /usr/local/lib/python3.7/dist-packages (0.20.6)
```

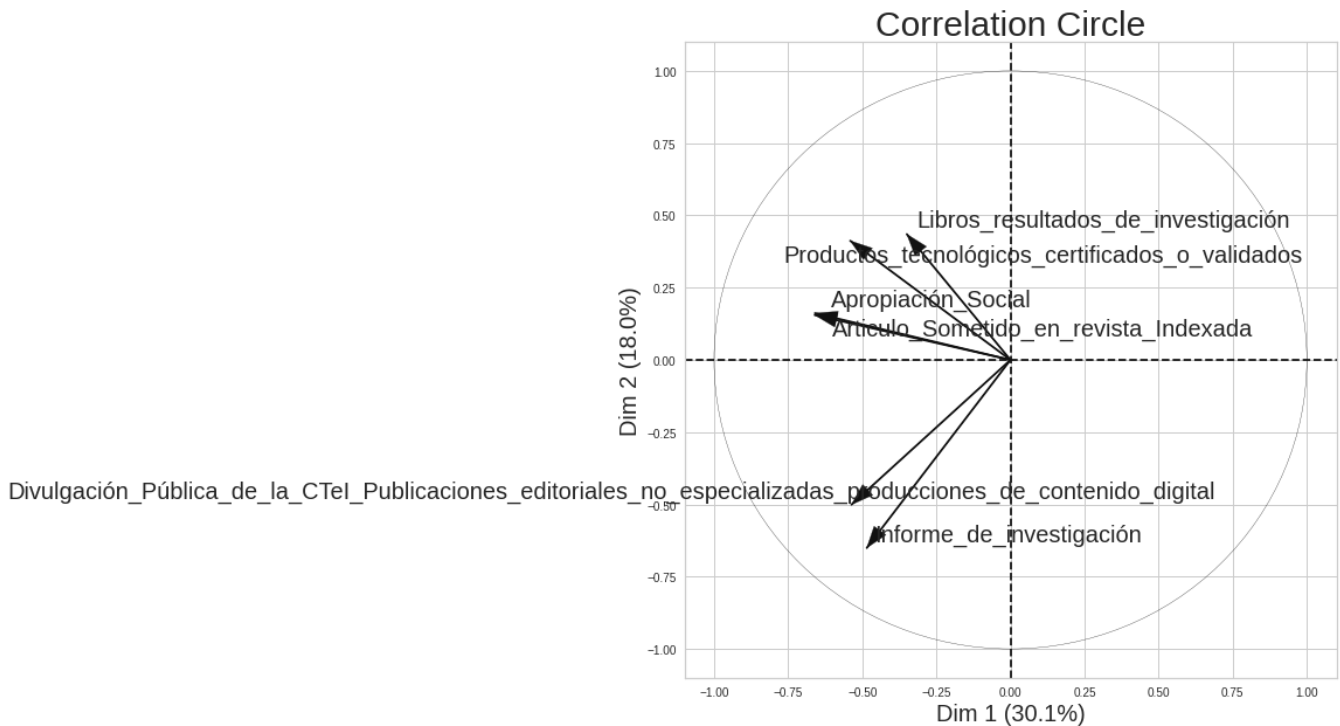
```
from mlxtend.plotting import plot_pca_correlation_graph
```

```
df.columns
```

```
Index(['Centro_Año', 'Apropiación_Social',
       'Articulo_Sometido_en_revista_Indexada',
       'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produ',
       'Informe_de_investigación', 'Libros_resultados_de_investigación',
       'Productos_tecnológicos_certificados_o_validados', 'KMeans_Clusters'],
      dtype='object')
```

```
x1= df.drop(['Centro_Año', 'KMeans_Clusters'],axis = 1)
features = ['Apropiación_Social',
            'Articulo_Sometido_en_revista_Indexada',
            'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion',
            'Informe_de_investigación', 'Libros_resultados_de_investigación',
            'Productos_tecnológicos_certificados_o_validados']
```

```
figure, correlation_matrix = plot_pca_correlation_graph(x1, features, dimensions=(1, 2), figu
```



Por tanto:

- **Cluster 2: Alta productividad**
- **Cluster 0: Media productividad**
- **Cluster 1: Baja Productividad**

correlation_matrix

Apropiación_Social

Articulo_Sometido_en_revista_Indexada

Divulgación_Pública_de_la_CTel_Publicaciones_editoriales_no_especializadas_producciones_de_c

Informe_de_investigación

Libros_resultados_de_investigación

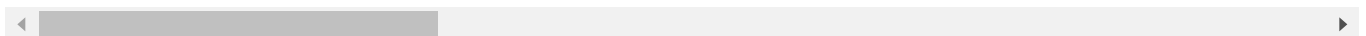
Productos_tecnológicos_certificados_o_validados



x1

	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación_Pública_de_
0	-0.609017		-0.526118
1	-1.337038		-1.201628
2	-1.337038		-1.201628
3	0.119003		-0.526118
4	-0.609017		-0.526118
...	...		...
203	-0.609017		-0.526118
204	0.119003		-0.526118
205	0.847024		0.149392
206	-0.609017		0.149392
207	2.303066		0.824901

208 rows × 6 columns



```

gestion= df['KMeans_Clusters'].replace({0: 'Baja Productividad',2: 'Media Productividad', 1:
df['Productividad'] = pd.concat([gestion], axis=1)
df_k_clas = df
df_k_clas

```


	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación
0	9101-2010	-0.609017		-0.526118
1	9127-2010	-1.337038		-1.201628
2	9201-2010	-1.337038		-1.201628
3	9202-2010	0.119003		-0.526118
4	9203-2010	-0.609017		-0.526118
...	...	...		...
203	9501-2022	-0.609017		-0.526118
204	9502-2022	0.119003		-0.526118
205	9503-2022	0.847024		0.149392
206	9504-2022	-0.609017		0.149392
207	9549-2022	2.303066		0.824901

```
df_k_clas.to_excel('Productividad_Cluster_Kmeans.xlsx', index = False)
```

▼ DENDOGRAMA

```
df = df.drop(['KMeans_Clusters', 'Productividad'], axis=1)
df
```

	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación
0	9101-2010	-0.609017		-0.526118
1	9127-2010	-1.337038		-1.201628
2	9201-2010	-1.337038		-1.201628
3	9202-2010	0.119003		-0.526118
4	9203-2010	-0.609017		-0.526118
...	...	...		...
203	9501-2022	-0.609017		-0.526118
204	9502-2022	0.119003		-0.526118

```
from scipy.cluster.hierarchy import linkage,dendrogram
```

```
df_el = df.drop(['Centro_Año'],axis =1)
```

```
labels = df['Centro_Año'].to_numpy().tolist()
```

207	9549-2022	2.303066	0.824901
------------	-----------	----------	----------

```
x.columns
```

```
Index(['Apropiación_Social', 'Articulo_Sometido_en_revista_Indexada',
      'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion',
      'Informe_de_investigación', 'Libros_resultados_de_investigación',
      'Productos_tecnológicos_certificados_o_validados'],
      dtype='object')
```



```
features = ['Apropiación_Social', 'Articulo_Sometido_en_revista_Indexada',
            'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion',
            'Informe_de_investigación', 'Libros_resultados_de_investigación',
            'Productos_tecnológicos_certificados_o_validados']
```

```
X = x[features]
```

```
plt.figure(figsize=(20,10))
```

```
plt.title('Dendrograma de agrupamiento jerárquico para datos de productividad',fontsize=20)
```

```
plt.xlabel('sample index', fontsize=20)
```

```
plt.ylabel('distance',fontsize=20)
```

```
Z = linkage(X, 'complete','euclidean')
```

```
dn1 = sch.dendrogram(
```

```
    Z,
```

```
    labels=labels,
```

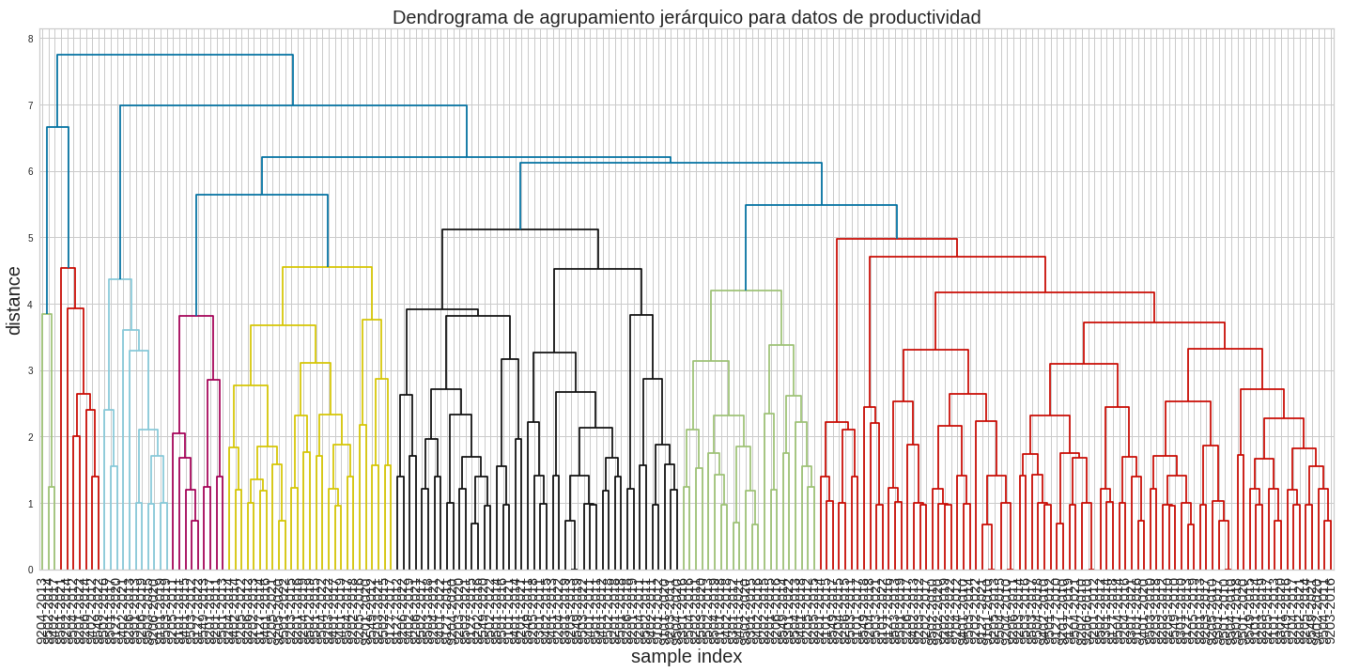
```
    leaf_rotation=90., # rotates the x axis labels
```

```
    leaf_font_size=15., # font size for the x axis labels
```

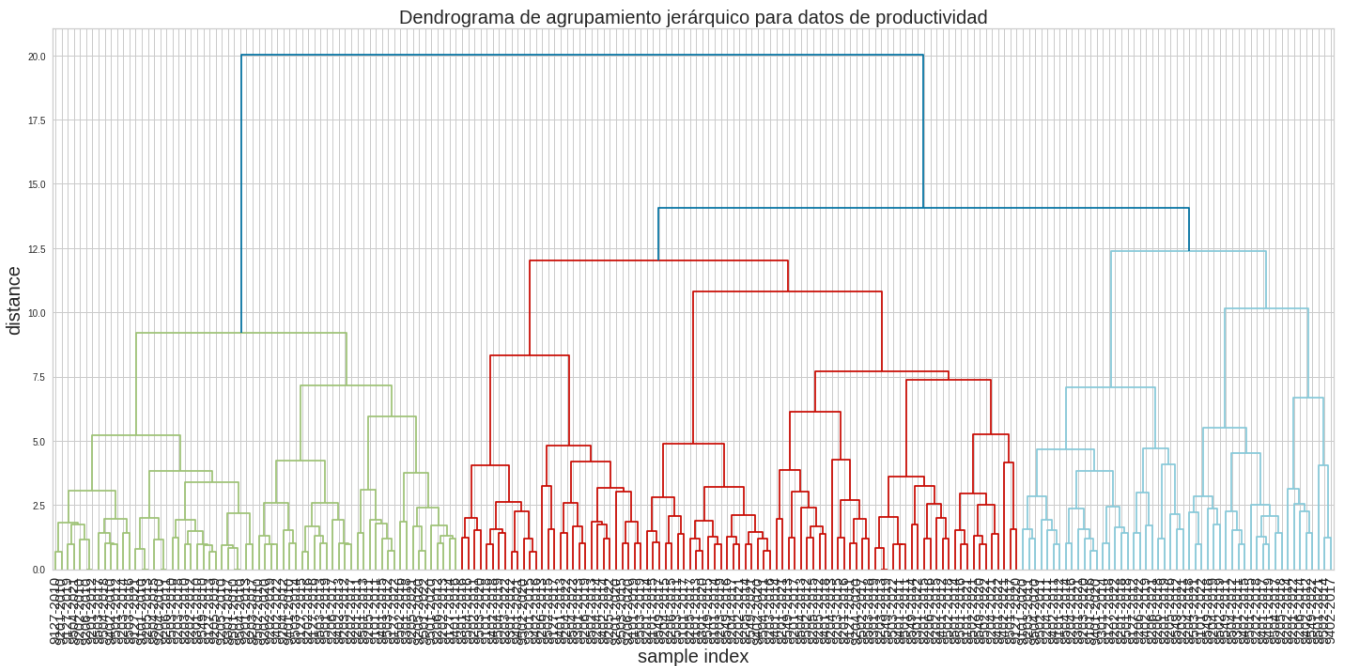
```
)
```

```
plt.tight_layout()
```

```
plt.show()
```



```
plt.figure(figsize=(20,10))
plt.title('Dendrograma de agrupamiento jerárquico para datos de productividad',fontsize=20)
plt.xlabel('sample index', fontsize=20)
plt.ylabel('distance',fontsize=20)
Z = linkage(X, 'ward','euclidean')
dn2 = sch.dendrogram(
    Z,
    labels=labels,
    leaf_rotation=90., # rotates the x axis labels
    leaf_font_size=15., # font size for the x axis labels
)
plt.tight_layout()
plt.show()
```



```

hac = AgglomerativeClustering(n_clusters=3,
                              affinity="euclidean",
                              linkage="ward")

```

```

hac.fit(x1)

```

```

AgglomerativeClustering(n_clusters=3)

```

```

hac.fit(X)
categorias = hac.labels_
categorias

```

```

array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 0, 0,
       1, 1, 0, 2, 0, 2, 0, 2, 2, 0, 0, 0, 1, 1, 1, 1, 2, 1, 2, 2, 0, 1,
       2, 1, 0, 0, 2, 1, 1, 1, 2, 2, 1, 2, 2, 2, 1, 1, 2, 1, 2, 2, 2, 1,
       2, 0, 1, 2, 2, 0, 0, 2, 2, 2, 0, 1, 0, 2, 2, 1, 2, 2, 2, 2, 2, 2,
       2, 1, 0, 0, 2, 2, 1, 2, 0, 1, 2, 1, 2, 0, 0, 2, 2, 1, 2, 2, 1, 1,
       2, 2, 2, 2, 2, 1, 1, 2, 2, 1, 1, 0, 0, 2, 2, 1, 0, 2, 2, 0, 0, 0,
       1, 0, 0, 0, 1, 2, 2, 0, 2, 2, 1, 0, 2, 1, 0, 1, 2, 0, 1, 2, 2, 0,
       0, 1, 0, 2, 2, 2, 0, 2, 2, 1, 0, 2, 1, 2, 2, 0, 2, 1, 2, 2, 0, 2,
       2, 1, 2, 2, 0, 2, 1, 0, 0, 2, 2, 0, 2, 0, 1, 2, 0, 1, 0, 0, 2, 2,
       2, 0, 2, 2, 1, 1, 2, 2, 2, 0])

```

```

#Agregando la clasificación en el archivo original
df["Hac_Clusters"]=categorias
df

```

	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación
0	9101-2010	-0.609017		-0.526118
1	9127-2010	-1.337038		-1.201628
2	9201-2010	-1.337038		-1.201628
3	9202-2010	0.119003		-0.526118
4	9203-2010	-0.609017		-0.526118
...	...	...		...
203	9501-2022	-0.609017		-0.526118
204	9502-2022	0.119003		-0.526118
205	9503-2022	0.847024		0.149392
206	9504-2022	-0.609017		0.149392
207	9549-2022	2.303066		0.824901

208 rows × 8 columns

▼ Validación

índices de validación externa

```
#Calcular agrupación de índices de validación externa con KMeans
labels_true = df["Centro_Año"]
rotulos_km = hac.labels_
h = sm.homogeneity_score(labels_true, rotulos_km)
print("Homogeneidad: %0.3f" % h)
c = sm.completeness_score(labels_true, rotulos_km)
print("Compleitud: %0.3f" % c)
v = sm.v_measure_score(labels_true, rotulos_km)
print("Medida V: %0.3f" % v)
ari = sm.adjusted_rand_score(labels_true, rotulos_km)
print("Índice Rand corregido: %0.3f" % ari)
ami = sm.adjusted_mutual_info_score(labels_true, rotulos_km, average_method='arithmetic')
print("Adjusted Mutual Information: %0.3f" % ami)
fm = sm.fowlkes_mallows_score(labels_true, rotulos_km)
print("Índice Fowlkes-Mallows: %0.3f" % fm)
```

```
Homogeneidad: 0.201
Compleitud: 1.000
Medida V: 0.334
Índice Rand corregido: 0.000
Adjusted Mutual Information: 0.000
Índice Fowlkes-Mallows: 0.000
```

índices de validación interna

```
from sklearn.metrics import silhouette_score
#silhouette_avg = silhouette_score(x, clustering.labels_)
rotulos_hac = hac.labels_
s= sm.silhouette_score(x, rotulos_hac, metric='euclidean')
print("Coeficiente de Silhueta: %0.3f" % s)
ch = sm.calinski_harabasz_score(x,rotulos_hac)
print("Calinski_harabasz: %0.3f" % ch)
dbs = sm.davies_bouldin_score(x, rotulos_hac)
print("Davies Bouldin: %0.3f" % dbs)
```

```
Coeficiente de Silhueta: 0.133
Calinski_harabasz: 32.426
Davies Bouldin: 2.295
```

```
x.head()
```

	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación_Pública_de_la
0	-0.609017		-0.526118
1	-1.337038		-1.201628
2	-1.337038		-1.201628
3	0.119003		-0.526118
4	-0.609017		-0.526118

PCA

```
#Visualización de los Clusters que se crearon
#Aplicaremos Análisis de componetes principales (PCA) para darnos una idea de cómo se crearon
pca=PCA(n_components=2)
pca_productividad=pca.fit_transform(x)
pca_productividad_2=pd.DataFrame(data=pca_productividad,columns=["Componente_1", "Componente_2"])
pca_productividad_3=pd.concat([pca_productividad_2,df[["Hac_Clusters"]]],axis=1)
pca_productividad_3=pd.concat([pca_productividad_3,df[["Centro_Año"]]],axis=1)
pca_productividad_3
```

	Componente_1	Componente_2	Hac_Clusters	Centro_Año
0	-1.991854	0.502028	1	9101-2010
1	-2.773852	0.307726	1	9127-2010
2	-2.595027	0.595137	1	9201-2010
3	-1.549107	0.921996	1	9202-2010
4	-1.551687	0.336304	1	9203-2010
...	...	...	...	...
203	-0.654792	0.122801	1	9501-2022
204	-0.003204	0.018650	2	9502-2022
205	0.877874	-0.409651	2	9503-2022
206	0.935897	-0.788964	2	9504-2022

```
pca_productividad_3 = pca_productividad_3.dropna()
pca_productividad_3['Hac_Clusters'] = pca_productividad_3['Hac_Clusters'].astype(int)
pca_productividad_3
```

	Componente_1	Componente_2	Hac_Clusters	Centro_Año
0	-1.991854	0.502028	1	9101-2010
1	-2.773852	0.307726	1	9127-2010
2	-2.595027	0.595137	1	9201-2010
3	-1.549107	0.921996	1	9202-2010
4	-1.551687	0.336304	1	9203-2010
...	...	...	...	...
203	-0.654792	0.122801	1	9501-2022
204	-0.003204	0.018650	2	9502-2022
205	0.877874	-0.409651	2	9503-2022
206	0.935897	-0.788964	2	9504-2022
207	2.074298	1.292960	0	9549-2022

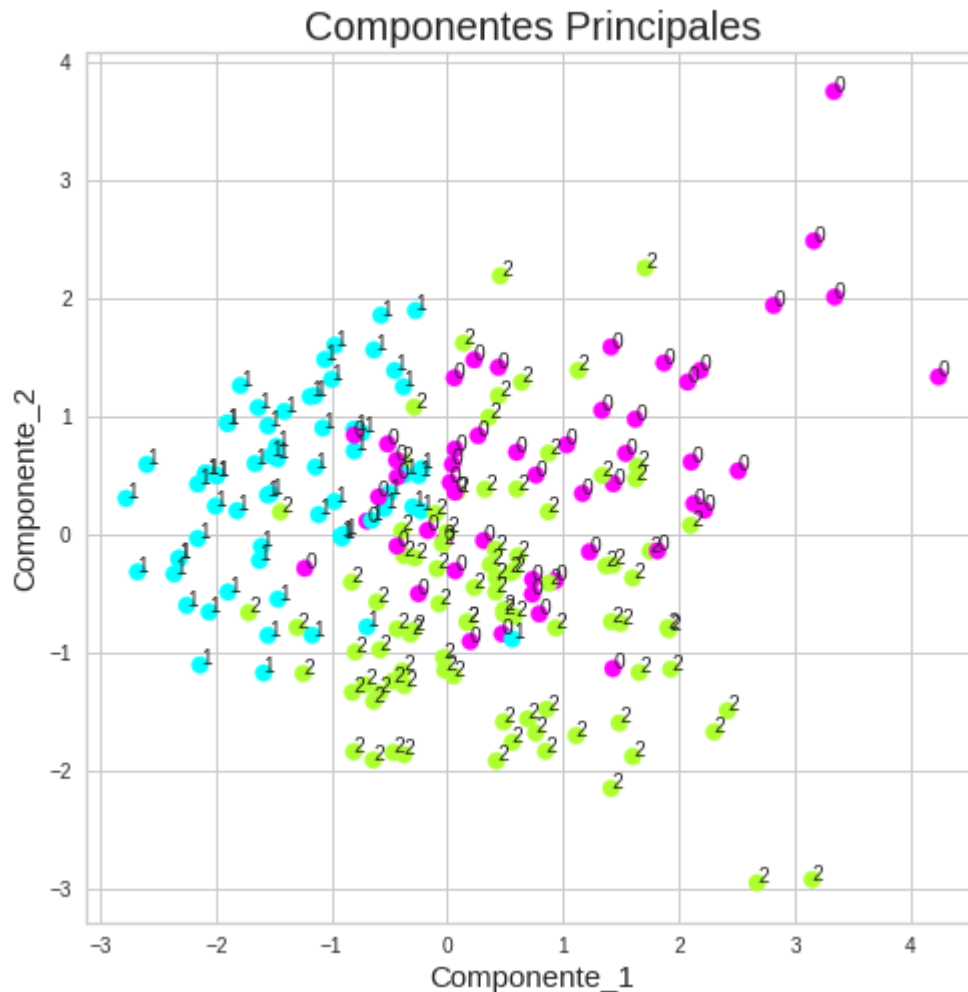
208 rows × 4 columns

```
fig=plt.figure(figsize=(8,8))
ax=fig.add_subplot(1,1,1)
annotations = df['Hac_Clusters']
ax.set_xlabel("Componente_1",fontsize=15)
ax.set_ylabel("Componente_2", fontsize=15)
```

```

ax.set_title("Componentes Principales", fontsize=20)
color_theme=np.array(["MAGENTA","CYAN",'GREENYELLOW','mediumpurple'])
ax.scatter(x=pca_productividad_3['Componente_1'], y=pca_productividad_3['Componente_2'],c=col
for i, label in enumerate(annotations):
    plt.annotate(label, (pca_productividad_3['Componente_1'][i], pca_productividad_3['Compone
plt.show()

```



```

fig=plt.figure(figsize=(25,25))
ax=fig.add_subplot(1,1,1)
annotations = pca_productividad_3['Hac_Clusters']
ax.set_xlabel("Componente_1",fontsize=15)
ax.set_ylabel("Componente_2", fontsize=15)
ax.set_title("Componentes Principales", fontsize=20)
color_theme=np.array(["MAGENTA","CYAN",'GREENYELLOW','mediumpurple'])
ax.scatter(x=pca_productividad_3['Componente_1'], y=pca_productividad_3['Componente_2'],c=col
for i, label in enumerate(annotations):
    plt.annotate(label, (pca_productividad_3['Componente_1'][i], pca_productividad_3['Compone
plt.show()

```




x1.columns

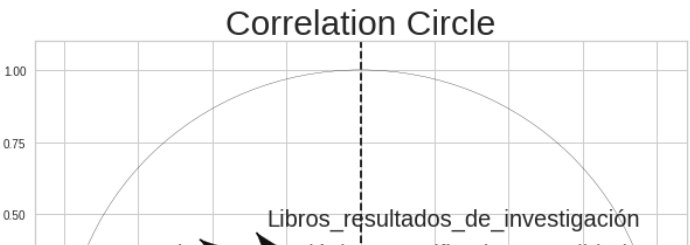
Index(['Apropiación_Social', 'Articulo_Sometido_en_revista_Indexada',

```
'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion',
'Informe_de_investigación', 'Libros_resultados_de_investigación',
'Productos_tecnológicos_certificados_o_validados'],
dtype='object')
```

df.columns

```
Index(['Centro_Año', 'Apropiación_Social',
'Articulo_Sometido_en_revista_Indexada',
'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion',
'Informe_de_investigación', 'Libros_resultados_de_investigación',
'Productos_tecnológicos_certificados_o_validados', 'Hac_Clusters'],
dtype='object')
```

```
x1= df.drop(['Centro_Año','Hac_Clusters'],axis = 1)
features = ['Apropiación_Social', 'Articulo_Sometido_en_revista_Indexada',
'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion',
'Informe_de_investigación', 'Libros_resultados_de_investigación',
'Productos_tecnológicos_certificados_o_validados']
figure, correlation_matrix = plot_pca_correlation_graph(x1, features,
                                                         dimensions=(1, 2),
                                                         figure_axis_size=10)
```



correlation_matrix

Apropiación_Social
Articulo_Sometido_en_revista_Indexada
Divulgación_Pública_de_la_CTel_Publicaciones_editoriales_no_especializadas_producciones_de_c
Informe_de_investigación
Libros_resultados_de_investigación
Productos_tecnológicos_certificados_o_validados

```
productividad= df['Hac_Clusters'].replace({0: 'Baja Productividad',2: 'Media Productividad',
df['Productividad'] = pd.concat([productividad], axis=1)
df_hac_clas = df
df_hac_clas
```

	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación
0	9101-2010	-0.609017		-0.526118
1	9127-2010	-1.337038		-1.201628
2	9201-2010	-1.337038		-1.201628
3	9202-2010	0.119003		-0.526118
4	9203-2010	-0.609017		-0.526118

```
df_hac_clas.to_excel('Productividad_Cluster_Dendograma.xlsx', index = False)
```

```
df = df_k_clas
centros_años = df["Centro_Año"].str.split("-", expand=True)
centros_años.columns = ['Codigo_Centro', 'Año']
df = pd.concat([df, centros_años], axis=1)
df.head()
```

	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación_P
0	9101-2010	-0.609017		-0.526118
1	9127-2010	-1.337038		-1.201628
2	9201-2010	-1.337038		-1.201628
3	9202-2010	0.119003		-0.526118
4	9203-2010	-0.609017		-0.526118

```
df.columns
```

```
Index(['Centro_Año', 'Apropiación_Social',
      'Articulo_Sometido_en_revista_Indexada',
      'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produccion',
      'Informe_de_investigación', 'Libros_resultados_de_investigación',
      'Productos_tecnológicos_certificados_o_validados', 'KMeans_Clusters',
      'Productividad', 'Codigo_Centro', 'Año'],
      dtype='object')
```

```
df_map = df[['Codigo_Centro', 'Año', 'KMeans_Clusters', 'Productividad']]
df_map
```

	Codigo_Centro	Año	KMeans_Clusters	Productividad
0	9101	2010	0	Baja Productividad
1	9127	2010	0	Baja Productividad
2	9201	2010	0	Baja Productividad
3	9202	2010	0	Baja Productividad
4	9203	2010	0	Baja Productividad
...	...	...	...	...
203	9501	2022	0	Baja Productividad
204	9502	2022	0	Baja Productividad
205	9503	2022	1	Alta Productividad
206	9504	2022	1	Alta Productividad
207	9549	2022	2	Media Productividad

208 rows × 4 columns

```
df_map['Codigo_Centro'] = df_map['Codigo_Centro'].astype(int)
df_map['Año'] = df_map['Año'].astype(int)
```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>
 """Entry point for launching an IPython kernel.

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>



```
df_map = df_map.merge(df_centros, on='Codigo_Centro', how='inner')
df_map.head()
```

	Codigo_Centro	Año	KMeans_Clusters	Productividad	
0	9101	2010	0	Baja Productividad	Centro_de_los_Recursos_Naturales_f
1	9101	2011	0	Baja Productividad	Centro_de_los_Recursos_Naturales_f
2	9101	2012	1	Alta Productividad	Centro_de_los_Recursos_Naturales_f
3	9101	2013	1	Alta Productividad	Centro_de_los_Recursos_Naturales_f

df_map = df_map.drop(['Codigo_Regional','Codigo_Municipio','Codigo_Departamento'],axis= 1)

Productividaa- - - - -

df_map

	Codigo_Centro	Año	KMeans_Clusters	Productividad	
0	9101	2010	0	Baja Productividad	Centro_de_los_Recursos_Naturales:
1	9101	2011	0	Baja Productividad	Centro_de_los_Recursos_Naturales:
2	9101	2012	1	Alta Productividad	Centro_de_los_Recursos_Naturales:
3	9101	2013	1	Alta Productividad	Centro_de_los_Recursos_Naturales:
4	9101	2014	1	Alta Productividad	Centro_de_los_Recursos_Naturales:
...	...	...	...	...	...
203	9549	2018	2	Media Productividad	Complejo_Tecnológico_Turístico
204	9549	2019	1	Alta Productividad	Complejo_Tecnológico_Turístico

◀ ▶

▼ Mapa para 2010

```
df_map_2010 = df_map[df_map['Año']<2011]
df_map_2010 = df_map_2010.drop(['Año'],axis = 1)

latitude = 6.51033052
longitude = -75.48477721
```

```
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=4)

# set color scheme for the clusters
x = np.arange(2)
ys = [i + x + (i*x)**2 for i in range(4)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lng, cluster, cent, gest in zip(df_map_2010['latitud'],
                                         df_map_2010['longitud'],
                                         df_map_2010['KMeans_Clusters'],
                                         df_map_2010['Nombre_Centro'],
                                         df_map_2010['Productividad']) :

    label = folium.Popup(str(cent) + '- Cluster ' + str(cluster), parse_html=True)
    folium.vector_layers.CircleMarker(
        [lat, lng],
        radius=10,
        #popup=label,
        tooltip = str(cent) + ' , ' + str(gest),
        color=rainbow[cluster-2],
        fill=True,
        fill_color=rainbow[cluster-2],
        fill_opacity=0.9).add_to(map_clusters)

map_clusters
```



▼ Mapa para 2014

```
df_map_2014 = df_map[df_map['Año']<2015]
df_map_2014 = df_map_2014.drop(['Año'],axis = 1)

latitude = 6.51033052
longitude = -75.48477721

map_clusters = folium.Map(location=[latitude, longitude], zoom_start=4)

# set color scheme for the clusters
x = np.arange(2)
ys = [i + x + (i*x)**2 for i in range(4)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

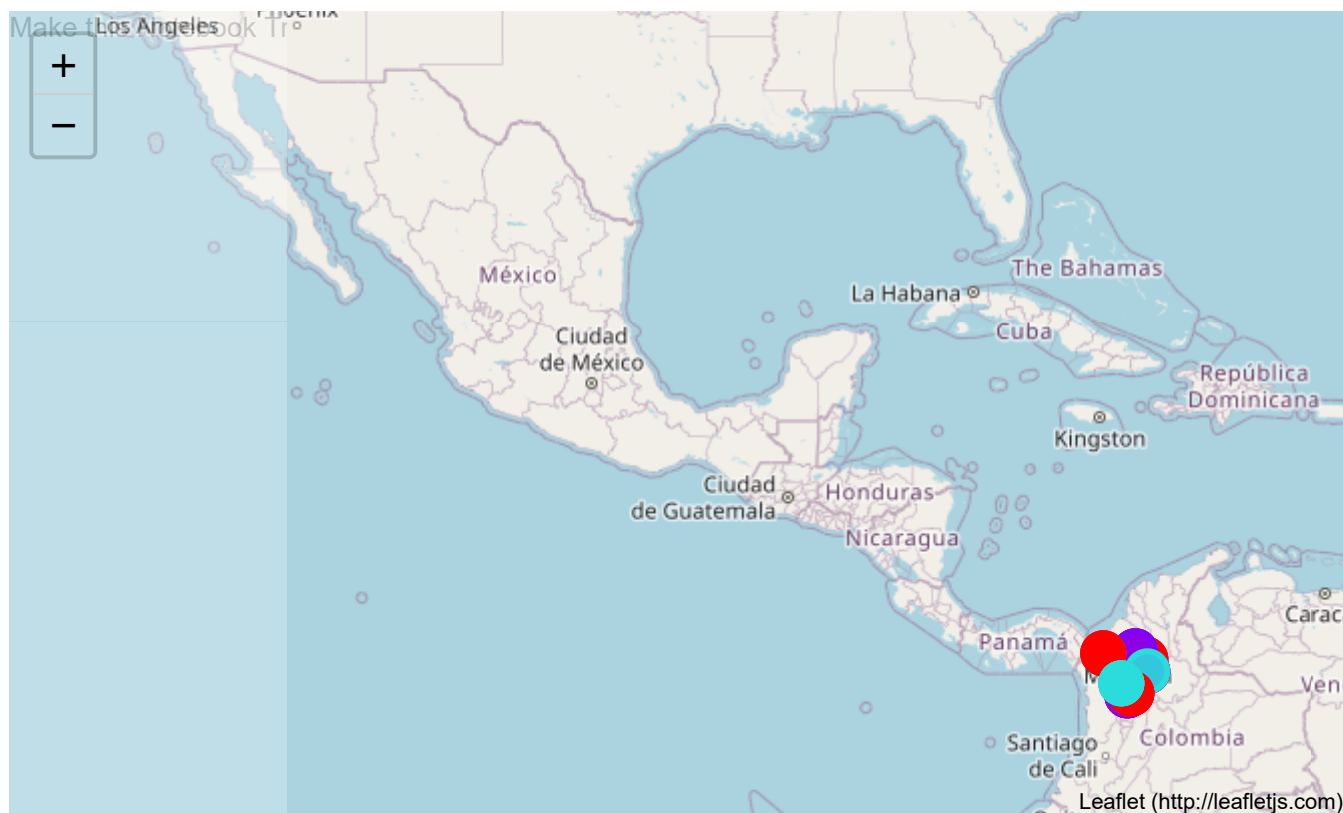
# add markers to the map
markers_colors = []
for lat, lng, cluster, cent, gest in zip(df_map_2014['latitud'],
                                         df_map_2014['longitud'],
                                         df_map_2014['KMeans_Clusters'],
                                         df_map_2014['Nombre_Centro'],
                                         df_map_2014['Productividad']) :

    label = folium.Popup(str(cent) + '- Cluster ' + str(cluster), parse_html=True)
    folium.vector_layers.CircleMarker(
        [lat, lng],
        radius=10,
        #popup=label,
        tooltip = str(cent) + ' , ' + str(gest),
        color=rainbow[cluster-1],
```



```
fill=True,
fill_color=rainbow[cluster-1],
fill_opacity=0.9).add_to(map_clusters)
```

map_clusters



▼ Mapa para 2018

```
df_map_2018 = df_map[df_map['Año']<2019]
df_map_2018 = df_map_2018.drop(['Año'],axis = 1)
```

```
latitude = 6.51033052
```

```
longitude = -75.48477721

map_clusters = folium.Map(location=[latitude, longitude], zoom_start=4)

# set color scheme for the clusters
x = np.arange(2)
ys = [i + x + (i*x)**2 for i in range(4)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lng, cluster, cent, gest in zip(df_map_2018['latitud'],
                                         df_map_2018['longitud'],
                                         df_map_2018['KMeans_Clusters'],
                                         df_map_2018['Nombre_Centro'],
                                         df_map_2018['Productividad']) :

    label = folium.Popup(str(cent) + ' - Cluster ' + str(cluster), parse_html=True)
    folium.vector_layers.CircleMarker(
        [lat, lng],
        radius=10,
        #popup=label,
        tooltip = str(cent) + ' , ' + str(gest),
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.9).add_to(map_clusters)

map_clusters
```



```
df_map_2022 = df_map[df_map['Año']<2023]
df_map_2022 = df_map_2022.drop(['Año'],axis = 1)
```

```
latitude = 6.51033052
longitude = -75.48477721
```

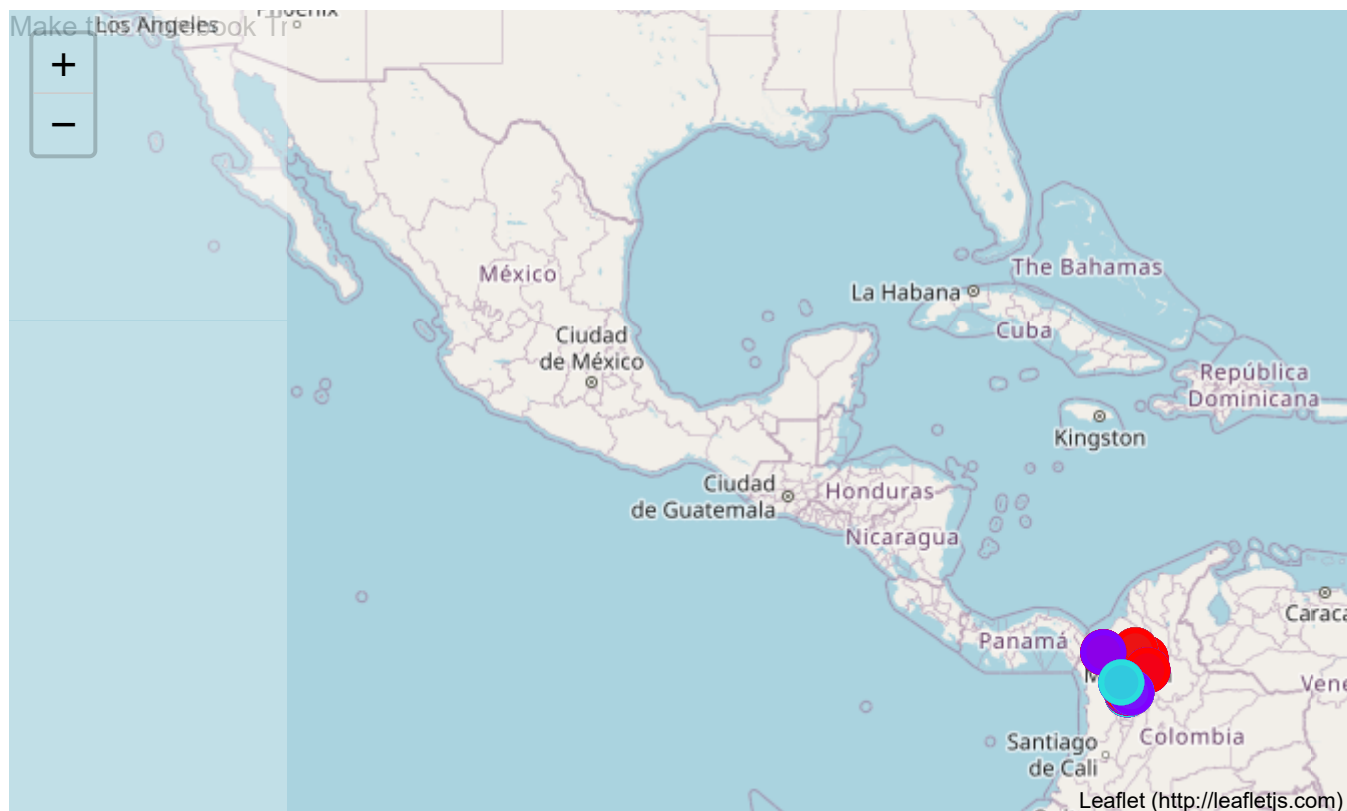
```
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=4)
```

```
# set color scheme for the clusters
x = np.arange(2)
ys = [i + x + (i*x)**2 for i in range(4)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]
```

```
# add markers to the map
markers_colors = []
for lat, lng, cluster, cent, gest in zip(df_map_2022['latitud'],
                                         df_map_2022['longitud'],
                                         df_map_2022['KMeans_Clusters'],
                                         df_map_2022['Nombre_Centro'],
                                         df_map_2022['Productividad']) :
```

```
label = folium.Popup(str(cent) + '- Cluster ' + str(cluster), parse_html=True)
folium.vector_layers.CircleMarker(
    [lat, lng],
    radius=10,
    #popup=label,
    tooltip = str(cent) + ' , ' + str(gest),
    color=rainbow[cluster-1],
    fill=True,
    fill_color=rainbow[cluster-1],
    fill_opacity=0.9).add_to(map_clusters)
```

map_clusters



▼ RED NEURONAL: PERCEPTRON

df.columns

```
Index(['Centro_Año', 'Apropiación_Social',  
      'Articulo_Sometido_en_revista_Indexada',  
      'Divulgación_Pública_de_la_CTeI_Publicaciones_editoriales_no_especializadas_produ',  
      'Informe_de_investigación', 'Libros_resultados_de_investigación',
```

```
'Productos_tecnológicos_certificados_o_validados', 'KMeans_Clusters',
'Productividad', 'Codigo_Centro', 'Año'],
dtype='object')
```

```
df_product_orig = pd.read_excel(path2)
df_product_orig.columns
df_prd = df[['Codigo_Centro', 'Año', 'Productividad']]
```

```
df_per = pd.concat([df_prd, df_product_orig], axis=1).drop(['Centro_Año'], axis=1)
df_per
```

	Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101	2010	Baja Productividad	1	1	0	0	0	1
1	9127	2010	Baja Productividad	0	0	0	0	1	0
2	9201	2010	Baja Productividad	0	0	0	0	2	0
3	9202	2010	Baja Productividad	2	1	0	0	2	0
4	9203	2010	Baja Productividad	1	1	0	1	1	1
...	...	...	...	...	...	...	...	...	...
203	9501	2022	Baja Productividad	1	1	1	2	3	1
204	9502	2022	Baja Productividad	2	1	3	1	3	1

```
Productividad = df_per['Productividad'].replace({'Baja Productividad': 1, 'Media Productividad': 0})
df_per['Productividad'] = pd.concat([Productividad], axis=1)
df_per
```

	Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101	2010	1	1	1	0	0	0	1
1	9127	2010	1	0	0	0	0	1	0
2	9201	2010	1	0	0	0	0	2	0
3	9202	2010	1	2	1	0	0	2	0
4	9203	2010	1	1	1	0	1	1	1

```
#Clases
```

```
y = (df_per['Productividad']).astype(np.int8)
print(y)
```

```
0      1
1      1
2      1
3      1
4      1
..
203    1
204    1
205    3
206    3
207    2
Name: Productividad, Length: 208, dtype: int8
```

Dividimos los datos en un conjunto de aprendizaje y de prueba:

```
from sklearn.model_selection import train_test_split
datasets = train_test_split(df_per,
                             y,
                             test_size=0.3)
```

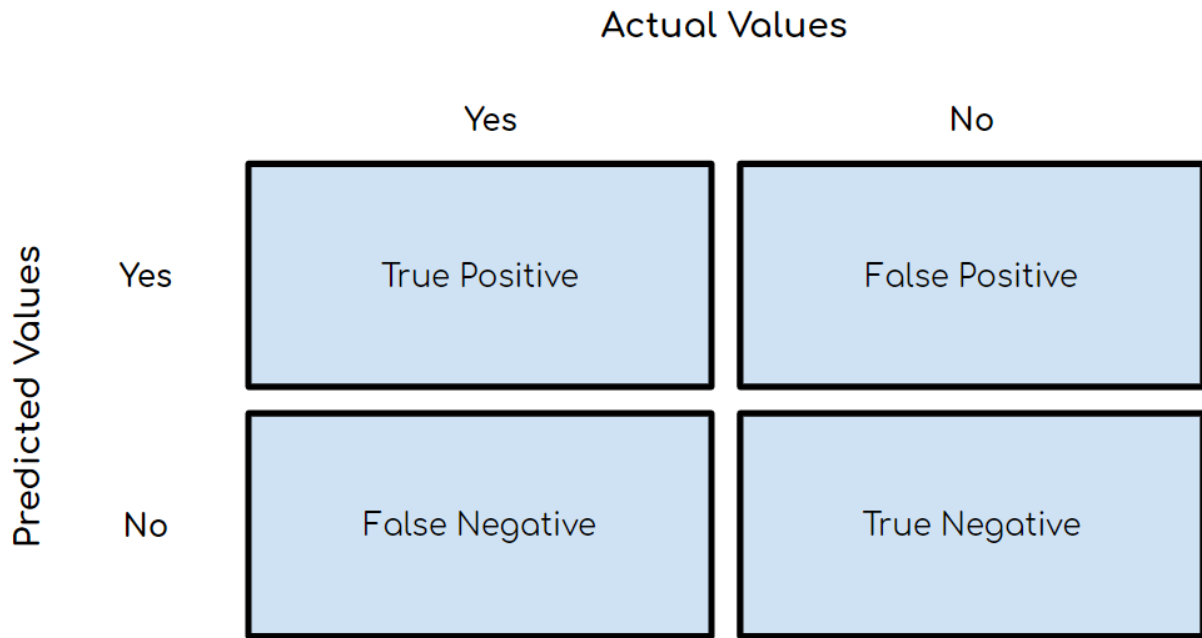
```
X_train, X_test, y_train, y_test = datasets
```

Ahora, creamos una instancia de Perceptron y ajustamos los datos de entrenamiento:

```
from sklearn.linear_model import Perceptron
p = Perceptron(random_state=42,
                max_iter=100,
                tol=0.001)
p.fit(X_train, y_train)

Perceptron(max_iter=100, random_state=42)
```

Recordando un poco sobre la matriz de confusión



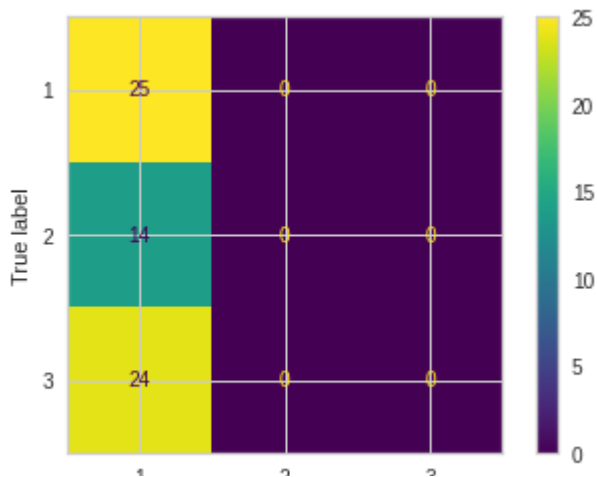
```
from sklearn.metrics import classification_report
y_perc_pred = p.predict(X_test)
print(classification_report(y_test, y_perc_pred))
```

	precision	recall	f1-score	support
1	0.40	1.00	0.57	25
2	0.00	0.00	0.00	14
3	0.00	0.00	0.00	24
accuracy			0.40	63
macro avg	0.13	0.33	0.19	63
weighted avg	0.16	0.40	0.23	63

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
_warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
_warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedWarning:
_warn_prf(average, modifier, msg_start, len(result))
```

```
plot_confusion_matrix(p, X_test, y_test)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
warnings.warn(msg, category=FutureWarning)
```



Test con valores de Prueba

```
path3 = '/content/drive/MyDrive/Proyecto Sennova borrador/sample.xlsx'
df_sample = pd.read_excel(path3)
df_sample.drop(['Productividad'], axis =1)
```

	Codigo_Centro	Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101.0	2010.0	10.0	5.0	5.0	5.0	5.0	5.0
1	9127.0	2010.0	5.0	5.0	5.0	5.0	5.0	5.0
2	9201.0	2010.0	0.0	0.0	0.0	0.0	0.0	1.0
3	9202.0	2010.0	0.0	0.0	0.0	0.0	0.0	0.0
4	9203.0	2010.0	0.0	1.0	0.0	1.0	2.0	0.0
5	9504.0	2022.0	12.0	12.0	12.0	12.0	12.0	12.0

```
y_percep_perd = p.predict(df_sample)
y_percep_perd_s= pd.Series(y_percep_perd)
df_percep = df_sample[['Codigo_Centro', 'Año', 'Productividad']]
df_percep['Productividad'] = y_percep_perd_s.replace({1: 'Baja Productividad', 2: 'Media Produ
df_percep
```



```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>
after removing the cwd from sys.path.

Codigo_Centro	Año	Productividad
---------------	-----	---------------

0101.0	2010.0	Baja Productividad
--------	--------	--------------------

```
#muestra_metricas(df_sample['Productividad'], p.predict(df_sample))
```

▼ Regresión Logística

Codigo_Centro	Año	Baja Productividad
---------------	-----	--------------------

```
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.model_selection import train_test_split
datasets = train_test_split(df_per,
                             y,
                             test_size=0.3)
```

```
X_train, X_test, y_train, y_test = datasets
```

```
# Instanciando a reg. logistica
reglog = LogisticRegression()
```

```
# Aplicando el modelo
reglog.fit(X_train, y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning:
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

Increase the number of iterations (max_iter) or scale the data as shown in:

<https://scikit-learn.org/stable/modules/preprocessing.html>

Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

```
extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
LogisticRegression()
```

```
LogisticRegression()
```

```
LogisticRegression()
```

```
# Predicción
```

```
y_reglog_pred = reglog.predict(X_test)
```

```
print(classification_report(y_test,y_reglog_pred))
```

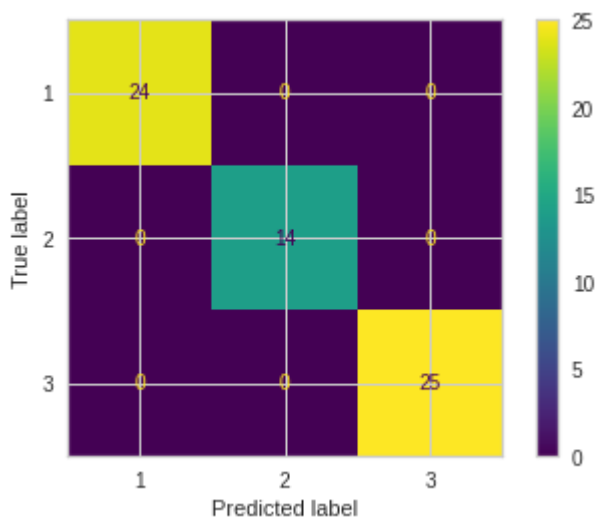
	precision	recall	f1-score	support
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	14
3	1.00	1.00	1.00	25
accuracy			1.00	63
macro avg	1.00	1.00	1.00	63
weighted avg	1.00	1.00	1.00	63

```
# Matriz de confusión
```

```
plot_confusion_matrix(reglog, X_test, y_test)
```

```
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
warnings.warn(msg, category=FutureWarning)
```



Test con valores de Prueba

```
df_sample.drop(['Productividad'], axis =1)
```

```

Codigo_Centro    Año  D 9020  D 9025  D 9010  D 9020  D 9010  D 9021
y_reglog_pred = reglog.predict(df_sample)
y_reglog_pred = pd.Series(y_reglog_pred)

df_logist = df_sample[['Codigo_Centro', 'Año', 'Productividad']]
df_logist['Productividad'] = y_reglog_pred.replace({1: 'Baja Productividad', 2: 'Media Product
df_logist

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user

```

	Codigo_Centro	Año	Productividad
0	9101.0	2010.0	Media Productividad
1	9127.0	2010.0	Media Productividad
2	9201.0	2010.0	Baja Productividad
3	9202.0	2010.0	Baja Productividad
4	9203.0	2010.0	Baja Productividad
5	9504.0	2022.0	Media Productividad

▼ Random Forest

```

from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()

rf.fit(X_train, y_train)

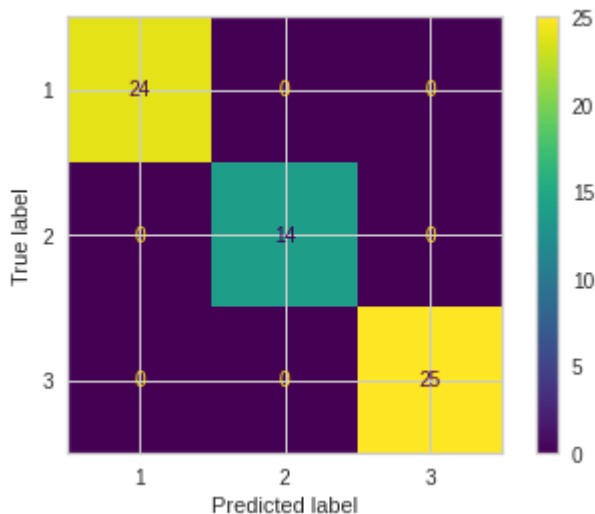
y_dt_pred = rf.predict(X_test)
print(classification_report(y_test, y_dt_pred))

```

	precision	recall	f1-score	support
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	14
3	1.00	1.00	1.00	25
accuracy			1.00	63
macro avg	1.00	1.00	1.00	63
weighted avg	1.00	1.00	1.00	63

```
# Matriz de confusión
plot_confusion_matrix(rf, X_test, y_test)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: f
warnings.warn(msg, category=FutureWarning)
```



Test con valores de Prueba

```
df_sample.drop(['Productividad'], axis =1)
```

	Codigo_Centro	Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101.0	2010.0	10.0	5.0	5.0	5.0	5.0	5.0
1	9127.0	2010.0	5.0	5.0	5.0	5.0	5.0	5.0
2	9201.0	2010.0	0.0	0.0	0.0	0.0	0.0	1.0
3	9202.0	2010.0	0.0	0.0	0.0	0.0	0.0	0.0
4	9203.0	2010.0	0.0	1.0	0.0	1.0	2.0	0.0
5	9504.0	2022.0	12.0	12.0	12.0	12.0	12.0	12.0

```
y_dt_pred_ram = rf.predict(df_sample)
y_dt_pred_ram = pd.Series(y_dt_pred_ram)
```

```
df_rd_forest = df_sample[['Codigo_Centro', 'Año', 'Productividad']]
df_rd_forest['Productividad'] = y_dt_pred_ram.replace({1: 'Baja Productividad', 2: 'Media Prod
df_rd_forest
```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>
"""

	Codigo_Centro	Año	Productividad
0	9101.0	2010.0	Baja Productividad
1	9127.0	2010.0	Baja Productividad
2	9201.0	2010.0	Baja Productividad
3	9202.0	2010.0	Baja Productividad
4	9203.0	2010.0	Baja Productividad

▼ SVM

```
from sklearn.svm import SVC
y_svm = SVC(C = 100, kernel = 'linear', random_state=123)
y_svm.fit(X_train, y_train)
```

```
SVC(C=100, kernel='linear', random_state=123)
```

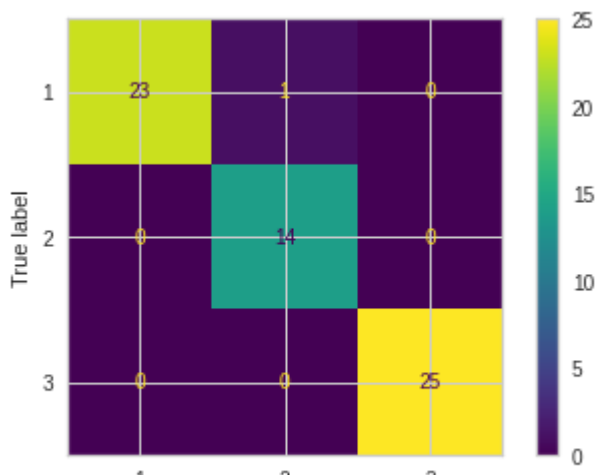
```
y_svm_pred = y_svm.predict(X_test)
print(classification_report(y_test,y_svm_pred))
```

	precision	recall	f1-score	support
1	1.00	0.96	0.98	24
2	0.93	1.00	0.97	14
3	1.00	1.00	1.00	25
accuracy			0.98	63
macro avg	0.98	0.99	0.98	63
weighted avg	0.99	0.98	0.98	63

Test con valores de muestra

```
# Matriz de confusión
plot_confusion_matrix(y_svm, X_test, y_test, include_values =True)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
warnings.warn(msg, category=FutureWarning)
```



```
df_sample.drop(['Productividad'], axis =1)
```

	Codigo_Centro	Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101.0	2010.0	10.0	5.0	5.0	5.0	5.0	5.0
1	9127.0	2010.0	5.0	5.0	5.0	5.0	5.0	5.0
2	9201.0	2010.0	0.0	0.0	0.0	0.0	0.0	1.0
3	9202.0	2010.0	0.0	0.0	0.0	0.0	0.0	0.0
4	9203.0	2010.0	0.0	1.0	0.0	1.0	2.0	0.0
5	9504.0	2022.0	12.0	12.0	12.0	12.0	12.0	12.0

```
y_dt_pred_svm = pd.Series(y_svm.predict(df_sample))
df_SVM = df_sample[['Codigo_Centro', 'Año', 'Productividad']]
df_SVM['Productividad'] = y_dt_pred_svm.replace({1: 'Baja Productividad', 2: 'Media Productivi
df_SVM
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
```

➤ Reporte de parametros de validación de modelos

```

        Codigo Centro      Año      Productividad

from sklearn.metrics import accuracy_score, confusion_matrix, f1_score, precision_score, recall_score

p.predict(X_test)

reglog.predict(X_test)

rf.predict(X_test)

y_svm.predict(X_test)

y_test

```

Reporte de parámetros de validación

```

# perceptrón
acc_perc = accuracy_score(y_test, p.predict(X_test))
f1_perc = f1_score(y_test, p.predict(X_test), average='weighted')
#auc_perc = roc_auc_score(y_test, p.predict(X_test))
#recall_perc = recall_score(y_test, p.predict(X_test))

# Regresión logística
acc_log = accuracy_score(y_test, reglog.predict(X_test))
f1_log = f1_score(y_test, reglog.predict(X_test), average='weighted')

#Arboles de decisión
acc_rf = accuracy_score(y_test, rf.predict(X_test))
f1_rf = f1_score(y_test, rf.predict(X_test), average='weighted')

#Máquinas de soporte vectorial
acc_svm = accuracy_score(y_test, y_svm.predict(X_test))
f1_svm = f1_score(y_test, y_svm.predict(X_test), average='weighted')

Parametro = ['Accuracy', 'F1-Score']
Percep= [acc_perc, f1_perc]
r_log = [acc_log, f1_log]
rfor = [acc_rf, f1_rf]
svm_t = [acc_svm, f1_svm]

df_valid = pd.DataFrame()
df_valid['Parámetro'] = Parametro
df_valid['Perceptrón'] = Percep

```

```
df_valid['Regresión logística'] = r_log
df_valid['Random Forest'] = rfor
df_valid['Máquina de soporte vectorial'] = svm_t
```

df_valid

	Parámetro	Perceptrón	Regresión logística	Random Forest	Máquina de soporte vectorial
0	Accuracy	0.380952	1.0	1.0	0.984127
1	F1-Score	0.210181	1.0	1.0	0.984232

```
df_valid.to_excel('Validacion_Productividad.xlsx',index =False)
```





ANEXOS 4. Presentación de avances técnicos junio 2022.



El empleo
es de todos

Mintrabajo

Proyecto SGPS 9748

Prototipo Innovación

Presentación de avances 12

Modelo de evaluación y valoración de la Gestión de CTel para los Centros SENA

Regional Antioquia aplicando analítica de datos

07 de junio del 2022



@SENAcomunica

www.sena.edu.co



Datos



@SENAcomunica

www.sena.edu.co

Dataframes identificados en base datos.

Principal_Productos	Principal_Proyectos	Principal_Talento_Humano	Principal_Equipos
Codigo_Producto Nombre_Producto Descripcion_producto Codigo_Regional Codigo_Centro Codigo_Rubro Codigo_Tipologia Codigo_Proyecto Nombre_Proyecto Codigo_Grupo_Investigacion Codigo_Semillero Codigo_persona Codigo_area_OCDE Fecha_registro	Codigo_Proyecto Nombre_Proyecto Valor_Proyecto Codigo_Rubro Fecha_inicio_Proyecto Fecha_Cierre_Proyecto Codigo_Regional Codigo_Centro Codigo_Estado_Proyecto Codigo_area_OCDE Serial_equipo Valor_Servicios_Personales Valor_Compra_Equipos Valor_Software Valor_Otros_Gastos	Codigo_Persona Codigo_Nivel_Academico Codigo_Tipo_Contrato Objeto_Contrato Codigo_Rubro Codigo_Area_OCDE Codigo_Estado_Contratacion Codigo_Regional Codigo_Centro Codigo_Proyecto Codigo_Producto	Serial_equipo Nombre_del_equipo Valor_equipo Fecha_de_compra Codigo_Proyecto Codigo_Regional Codigo_Centro Codigo_Rubro Codigo_Tipo_Equipo Codigo_area_OCDE Codigo_persona

Dataframes identificados en base datos.

Principal_Productos	Principal_Proyectos	Principal_Talento_Humano	Principal_Equipos
Codigo_Producto Nombre_Producto Descripcion_producto Codigo_Regional Codigo_Centro Codigo_Rubro Codigo_Tipologia Codigo_Proyecto Nombre_Proyecto Codigo_Grupo_Investigacion Codigo_Semillero Codigo_persona Codigo_area_OCDE Fecha_registro	Codigo_Proyecto Nombre_Proyecto Valor_Proyecto Codigo_Rubro Fecha_inicio_Proyecto Fecha_Cierre_Proyecto Codigo_Regional Codigo_Centro Codigo_Estado_Proyecto Codigo_area_OCDE Serial_equipo Valor_Servicios_Personales Valor_Compra_Equipos Valor_Software Valor_Otros_Gastos	Codigo_Persona Codigo_Nivel_Academico Codigo_Tipo_Contrato Objeto_Contrato Codigo_Rubro Codigo_Area_OCDE Codigo_Estado_Contratacion Codigo_Regional Codigo_Centro Codigo_Proyecto Codigo_Producto	Serial_equipo Nombre_del_equipo Valor_equipo Fecha_de_compra Codigo_Proyecto Codigo_Regional Codigo_Centro Codigo_Rubro Codigo_Tipo_Equipo Codigo_area_OCDE Codigo_persona



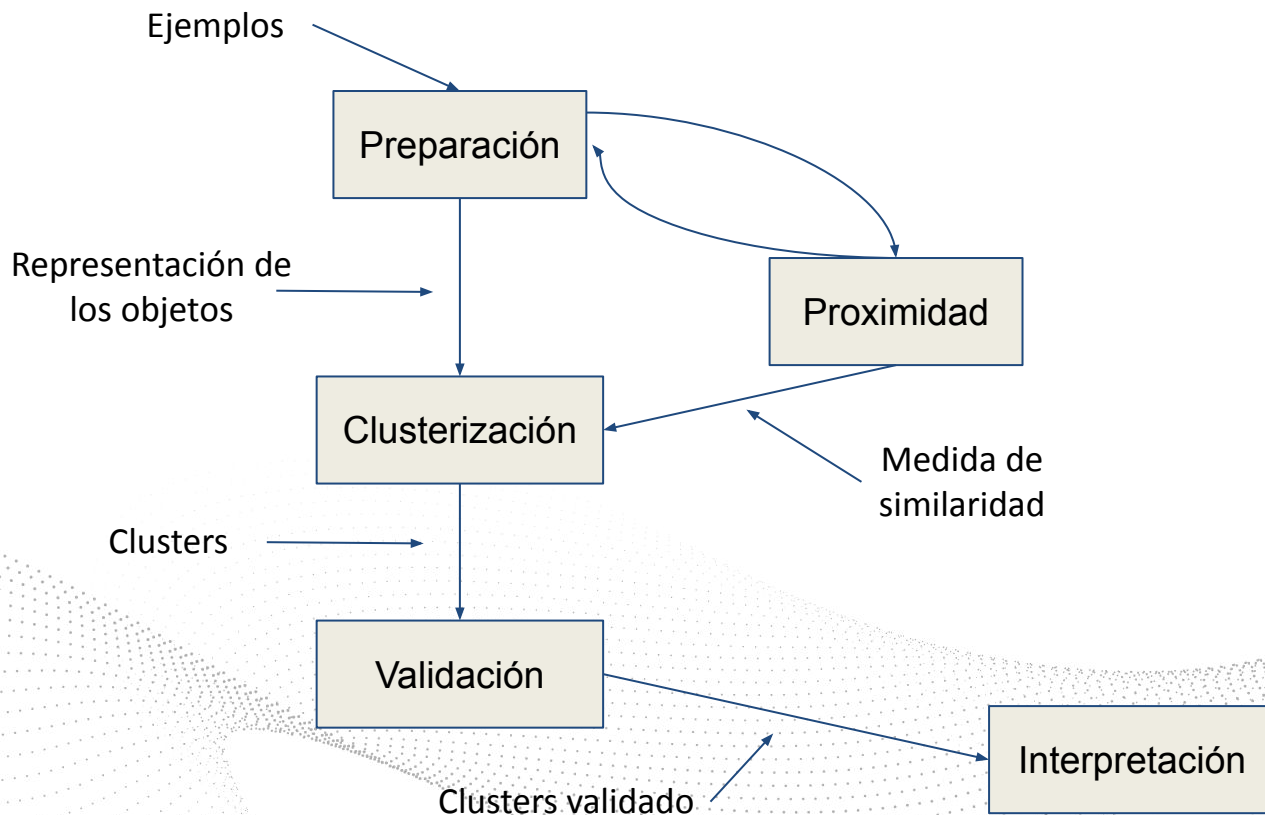
Etapas Clusterización - Clasificación



@SENAcomunica

www.sena.edu.co

Etapas del proceso de clusterización



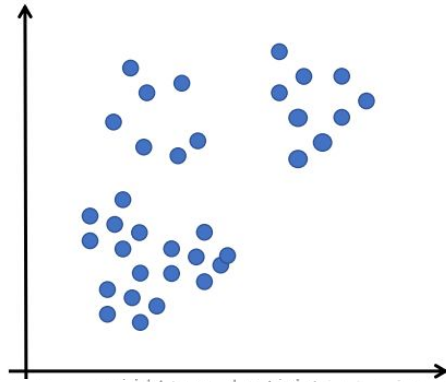
Etapas del proceso de clusterización

- **Preparación:** normalizaciones, conversiones de tipos, reducción del número de atributos
- **Proximidad:** definición (elección) de una medida de similitud apropiada al dominio de la aplicación, por ejemplo. distancia.
- **Clustering:** aplicación de algoritmos de clustering
- **Validación:** evaluación del resultado de la agrupación. Debe determinar si los conglomerados son significativos.
- **Interpretación:** etiquetado del clúster, definiendo su naturaleza, a través del análisis de sus objetos.

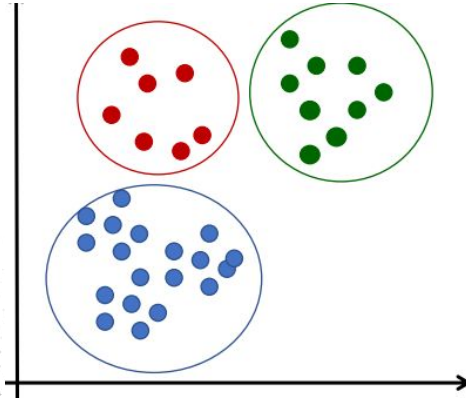
Agrupación de datos: tipos de algoritmos

Agrupación basado en particiones:

Conjunto original



Conjunto después de agrupación

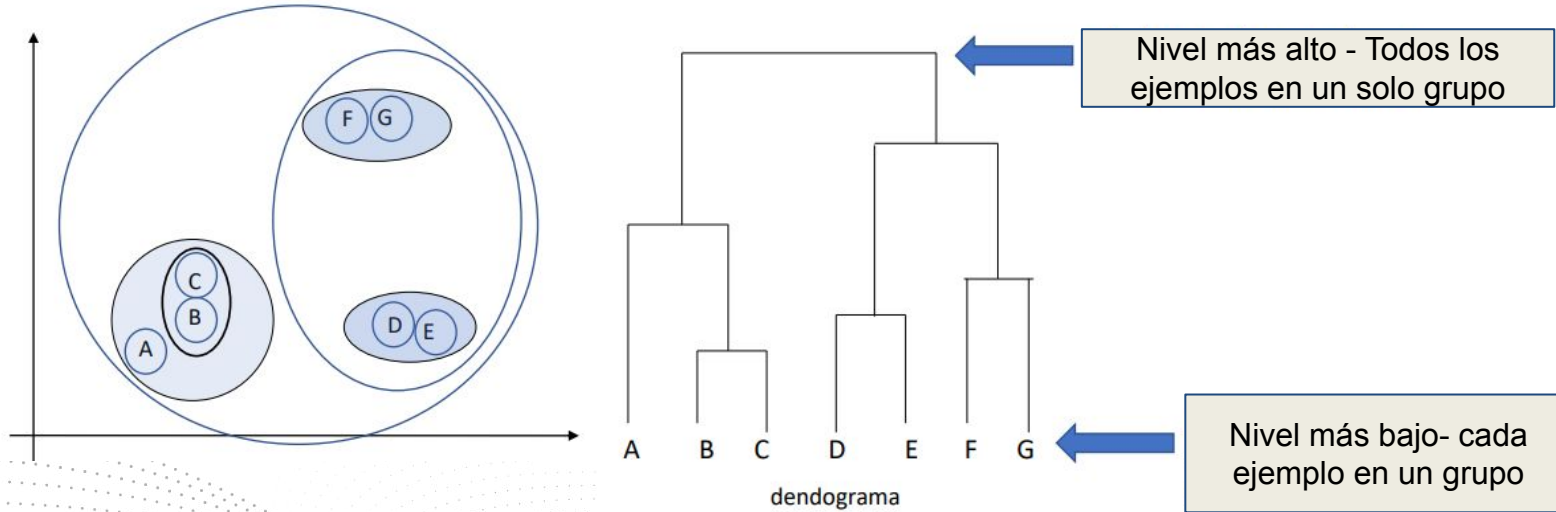


- **Ventaja:** más eficiente para grandes conjuntos de datos.
- **Inconveniente:** es necesario definir previamente el número de grupos deseado.

- Kmeans

Agrupación de datos: tipos de algoritmos

Agrupación jerárquica:



Ventaja: no es necesario establecer el número de grupos

Desventaja: Es más caro que el de particiones

Validación



Índices de validación externa: mide la calidad del agrupamiento conociendo información externa de antemano, es principalmente usada para escoger un algoritmo de clustering óptimo sobre un dataset específico.

- Homogeneidad
- Completitud
- Medida V
- Índice Rand corregido
- Adjusted Mutual Information
- Índice Fowlkes-Mallows

Valores entre 0 -1

0 : Aleatoriedad

1: Clusterización óptima

Índices de validación interno: miden el agrupamiento basado únicamente en la información de los datos. Evalúan que tanto la estructura del agrupamiento como la necesidad de información son buenas para el algoritmo y su resultado.

- Coeficiente de Silhueta
- Calinski_harabasz
- Davies Bouldin

Valores entre 0 -1

0 : Aleatoriedad

1: Clusterización óptima



Clusterización - Procesos



@SENAcomunica

www.sena.edu.co

Clusterización - Procesos 2010

1. Kmeans - Validación - PCA
2. Agrupamiento jerárquico - Validación -PCA

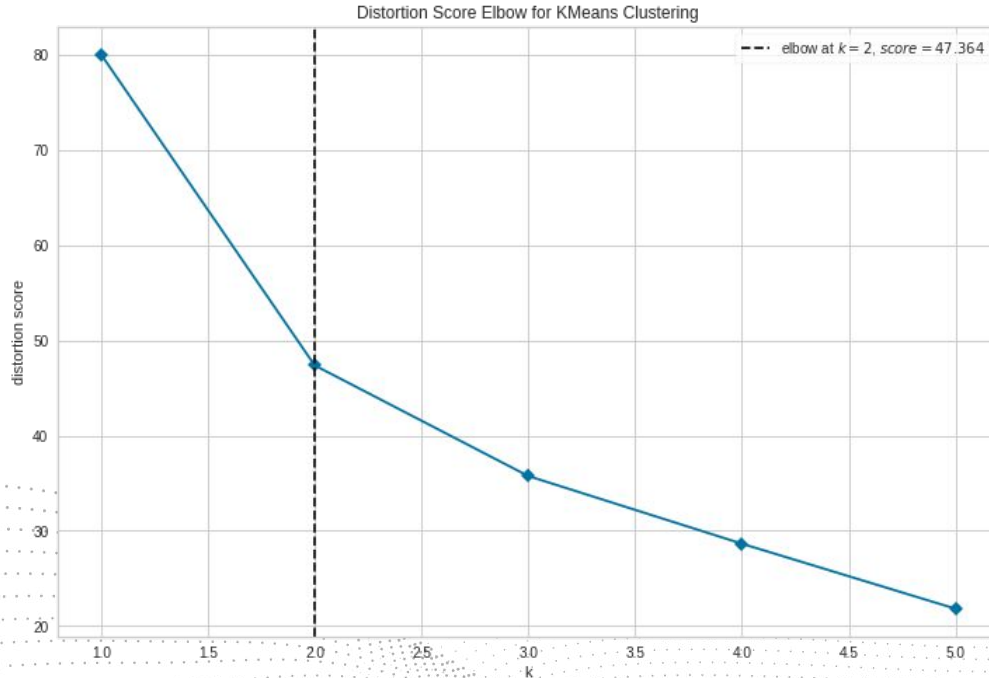
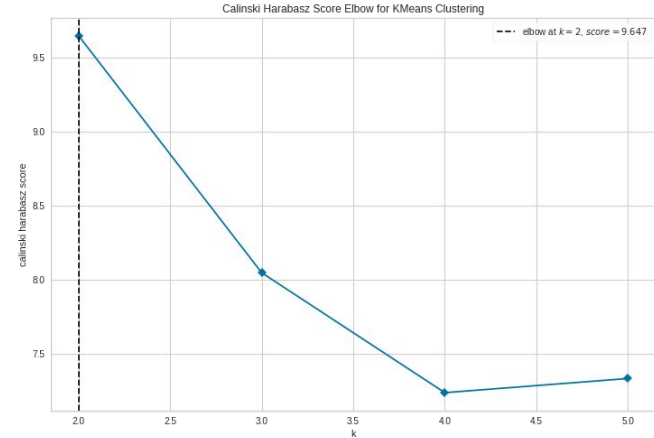
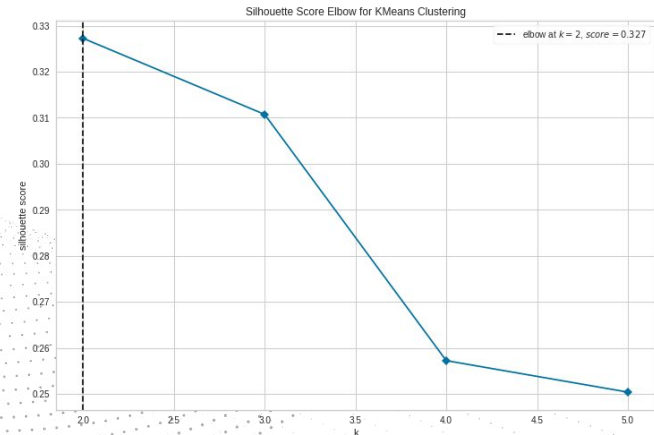
Datos procesos

	Codigo_Centro	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	Total_Valor_Proyectos
0	9101	1	3	1	0	4.010951e+08
1	9127	2	2	4	1	3.954825e+08
2	9201	1	0	2	0	5.069480e+08
3	9202	3	3	5	2	4.551236e+08
4	9203	2	3	2	2	4.610481e+08
5	9204	3	6	6	0	4.940284e+08
6	9205	1	3	0	1	4.156665e+08
7	9206	1	2	2	1	3.322490e+08

Normalización

	Codigo_Centro	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	Total_Valor_Proyectos
0	9101	-1.264911	-0.198228	-1.161049	-0.904534	-0.520118
1	9127	0.000000	-0.651320	0.198228	0.301511	-0.594640
2	9201	-1.264911	-1.557505	-0.707957	-0.904534	0.885360
3	9202	1.264911	-0.198228	0.651320	1.507557	0.197253
4	9203	0.000000	-0.198228	-0.707957	1.507557	0.275916
5	9204	1.264911	1.161049	1.104413	-0.904534	0.713818
6	9205	-1.264911	-0.198228	-1.614142	0.301511	-0.326644
7	9206	-1.264911	-0.651320	-0.707957	0.301511	-1.434231

Método del codo : Determinación de centroides



K-means

```
#Aplicando el método de k-means a la base de datos
clustering=KMeans(n_clusters=2, max_iter=300,init = 'random') #Crea el modelo
clustering.fit(x) #Aplica el modelo a la base de datos
clustering.cluster_centers_
```

```
array([[ -0.57495957, -0.48655943, -0.33724489, -0.2466911 , -0.43040562],
       [ 1.26491106,  1.07043074,  0.74193877,  0.54272042,  0.94689236]])
```

Se obtienen dos centroides con coordenadas en 5 dimensiones(cantidad de variables), por lo que no se genera visualización. Se debe reducir la dimensionalidad como PCA.

Centroides $k = 2$

	Codigo_Centro	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	Total_Valores_Proyectos	KMeans_Clusters
0	9101	-1.264911	-0.198228	-1.161049	-0.904534	-0.520118	0
1	9127	0.000000	-0.651320	0.198228	0.301511	-0.594640	0
2	9201	-1.264911	-1.557505	-0.707957	-0.904534	0.885360	0
3	9202	1.264911	-0.198228	0.651320	1.507557	0.197253	1
4	9203	0.000000	-0.198228	-0.707957	1.507557	0.275916	0
5	9204	1.264911	1.161049	1.104413	-0.904534	0.713818	1
6	9205	-1.264911	-0.198228	-1.614142	0.301511	-0.326644	0
7	9206	-1.264911	-0.651320	-0.707957	0.301511	-1.434231	0

Tabla 2. Cantidad de proyectos y cantidad de productos totales por centro de formación con cluster.

Validación del agrupamiento por KMeans

Externos

Homogeneidad: 0.224

Compleitud: 1.000

Medida V: 0.366

Índice Rand corregido: 0.000

Adjusted Mutual Information: -0.000

Índice Fowlkes-Mallows: 0.000

Internos

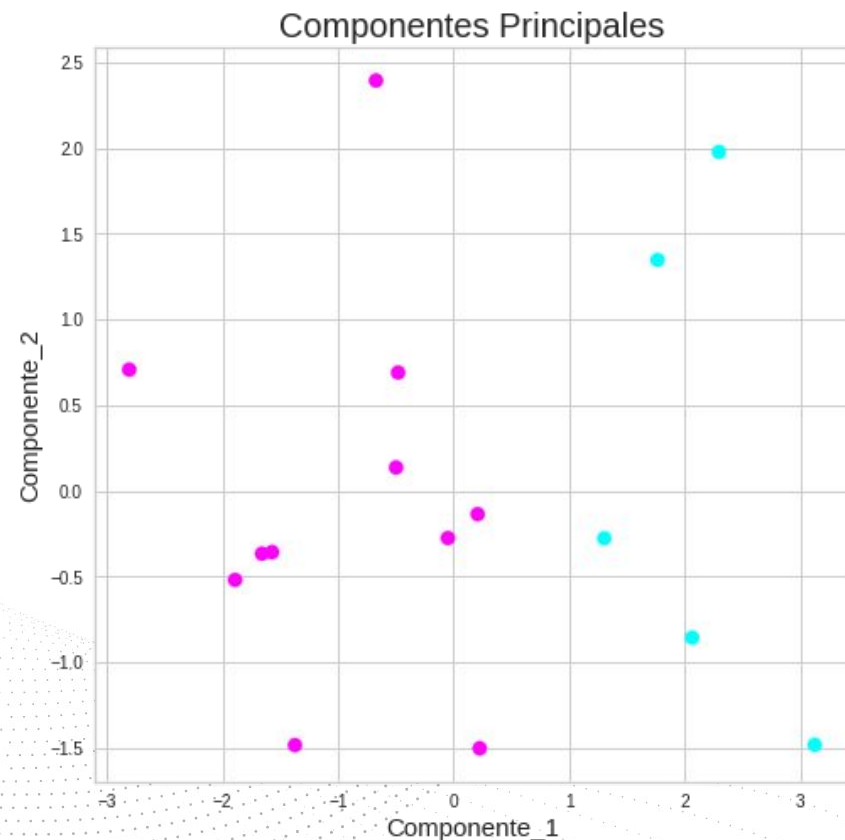
Coeficiente de Silhueta: 0.327

Calinski_harabasz: 9.647

Davies Bouldin: 1.084

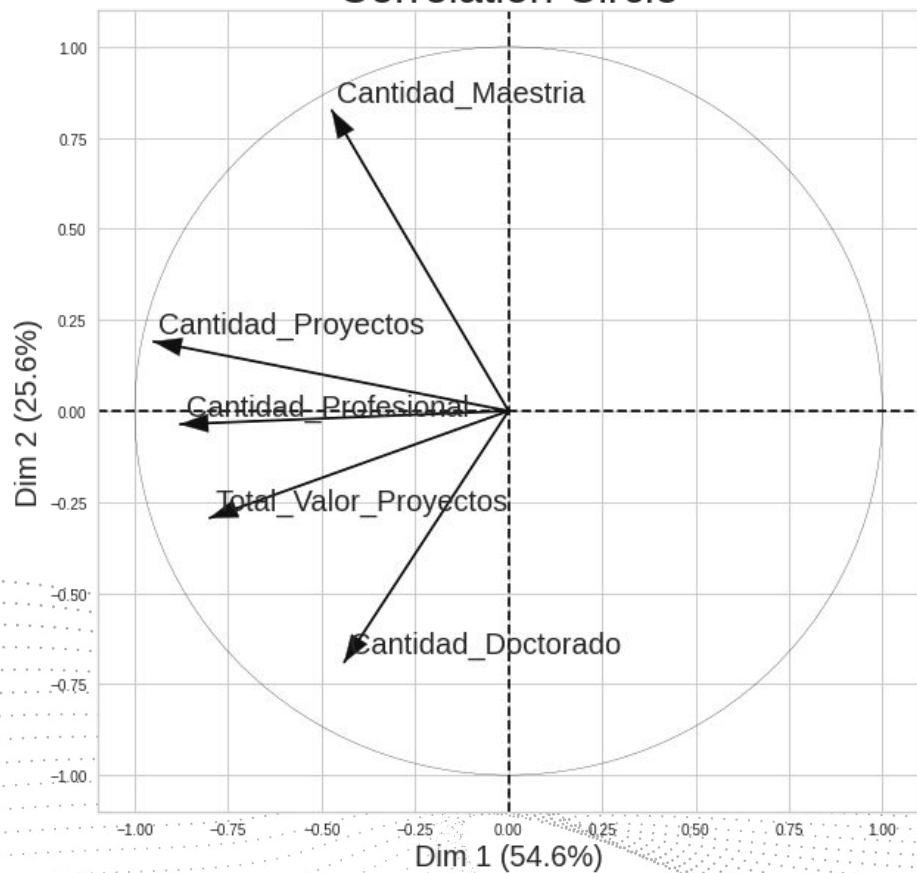


	Componente_1	Componente_2	KMeans_Clusters	Codigo_Centro
0	-1.656895	-0.366053	0	9101
1	-0.497273	0.136328	0	9127
2	-1.569897	-0.357092	0	9201
3	1.304405	-0.276405	1	9202
4	0.226627	-1.500502	0	9203
5	1.765279	1.346207	1	9204
6	-1.372057	-1.481934	0	9205
7	-1.889894	-0.518802	0	9206
8	2.296226	1.975542	1	9301
9	3.124895	-1.480215	1	9401
10	-0.048741	-0.274975	0	9402
11	0.208797	-0.135405	0	9501
12	2.065037	-0.855325	1	9502
13	-0.479540	0.689833	0	9503
14	-2.806286	0.706799	0	9504
15	-0.670683	2.391997	0	9549

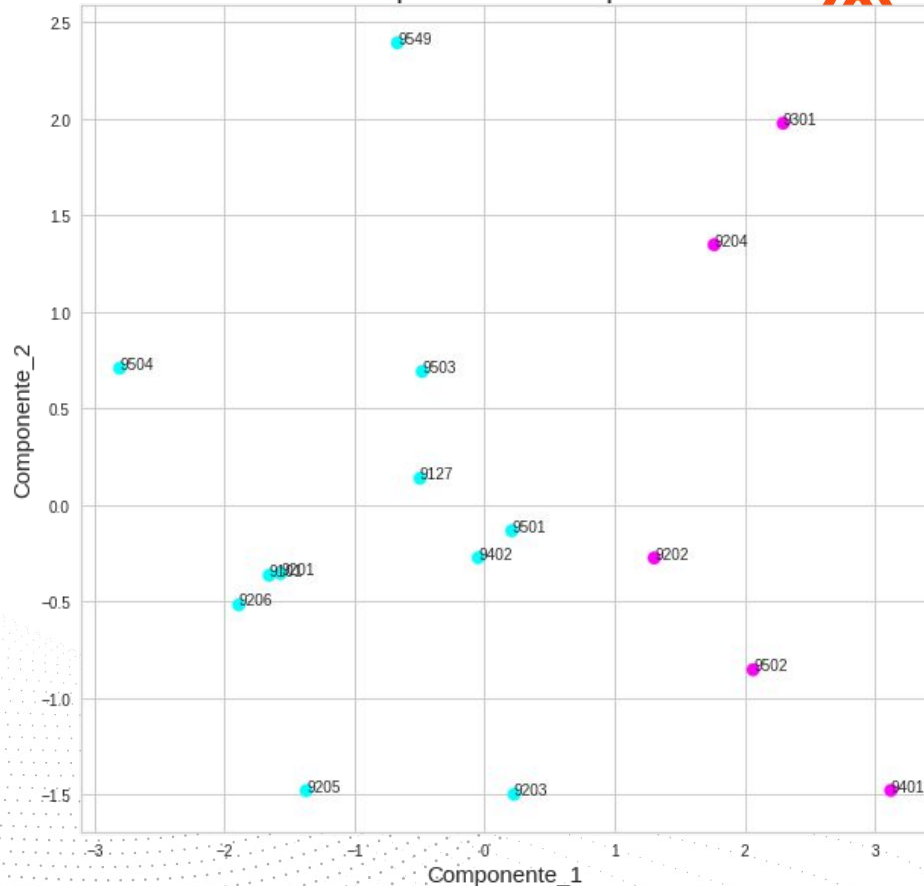




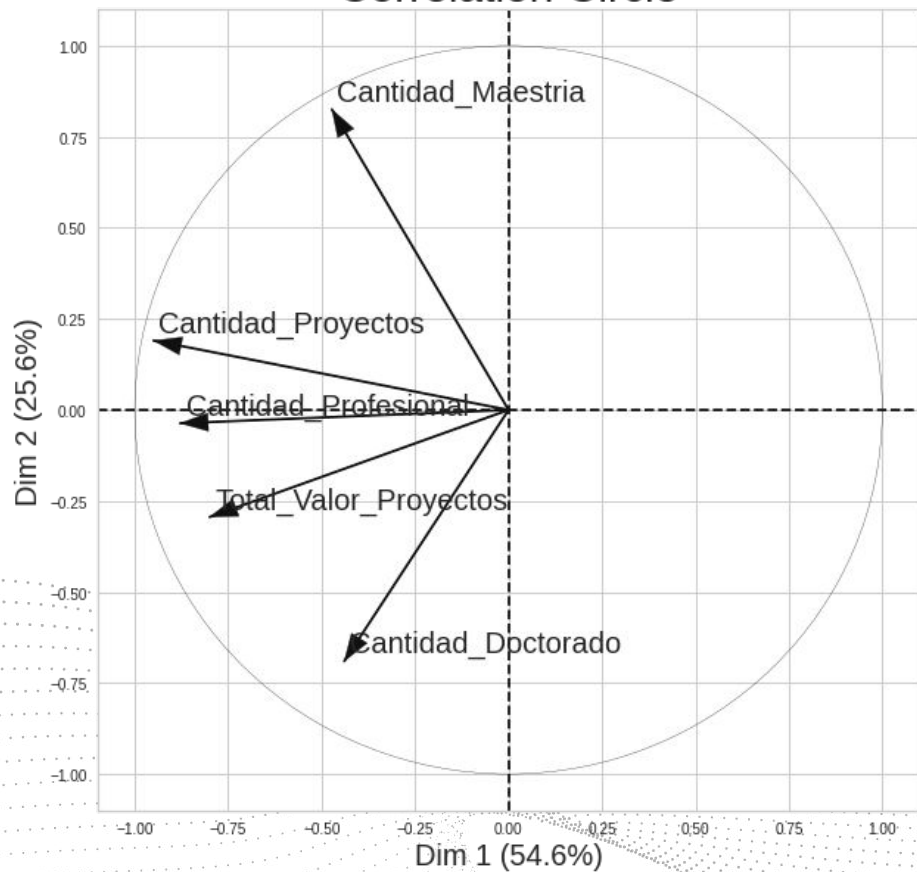
Correlation Circle



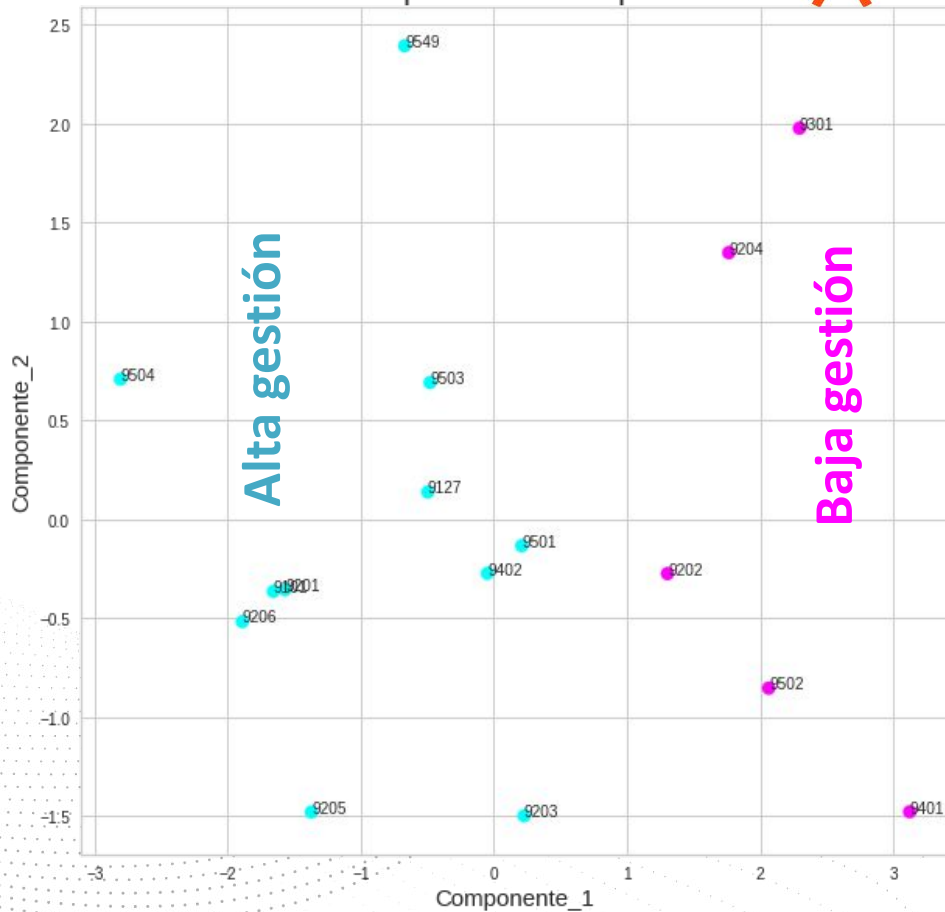
Componentes Principales



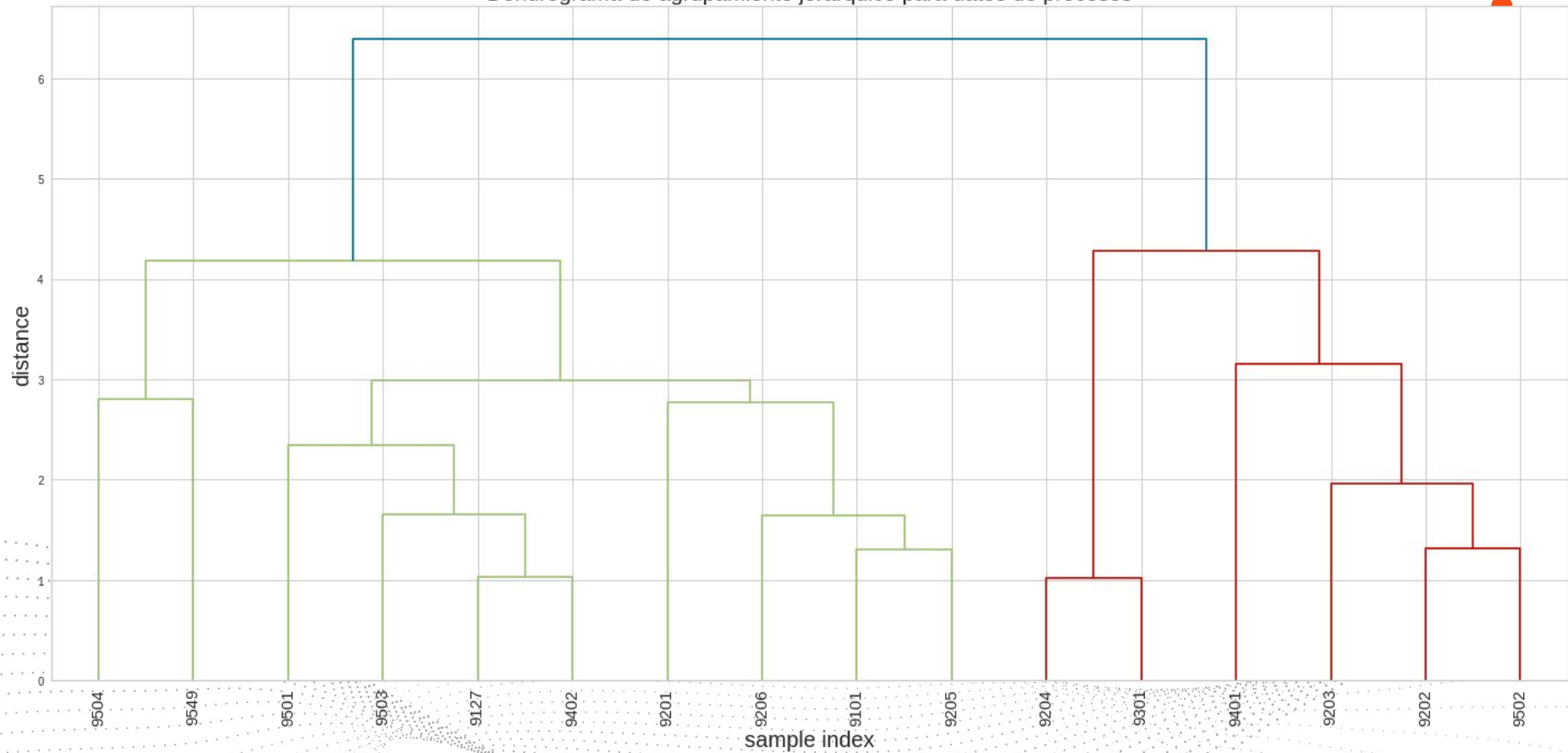
Correlation Circle



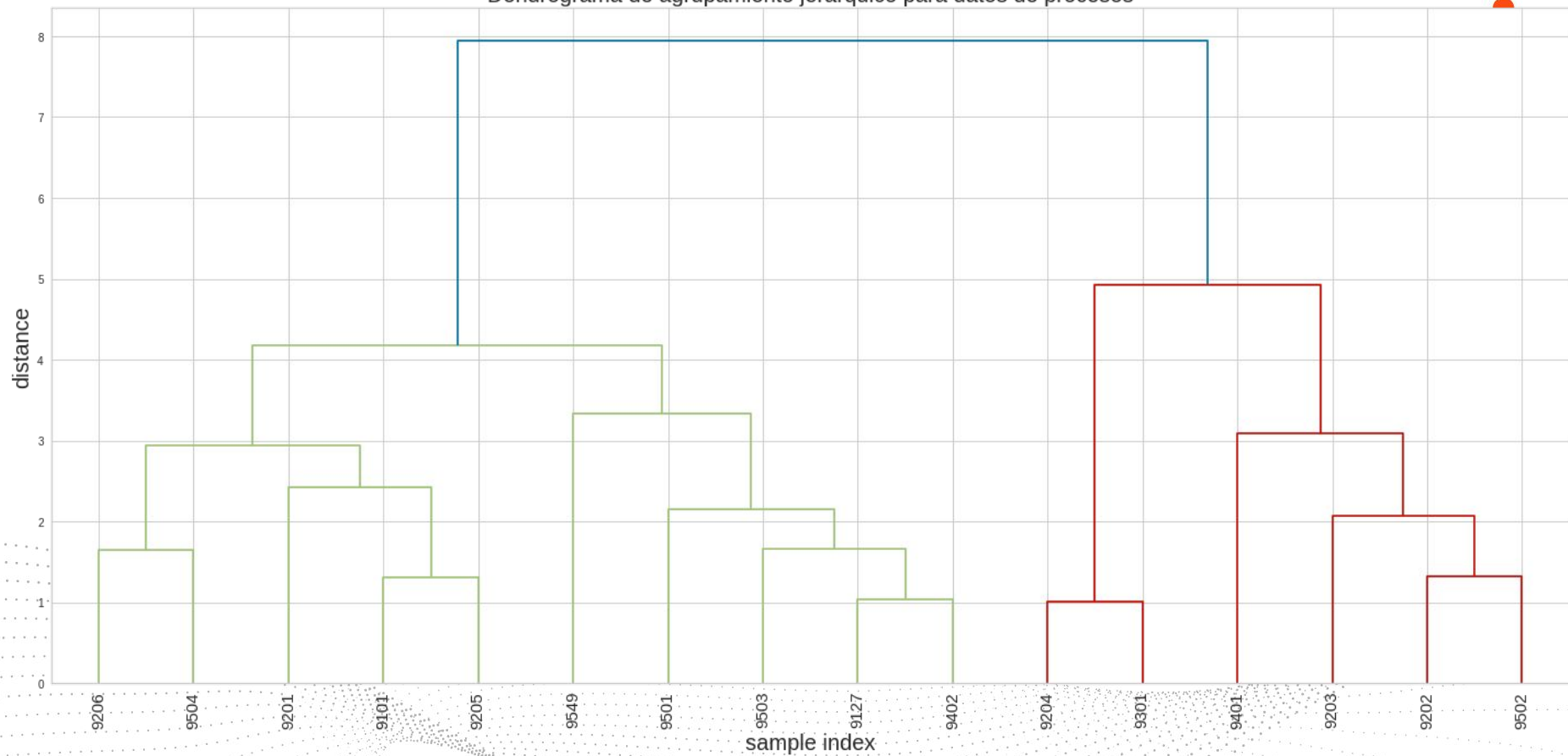
Componentes Principales



Dendrograma de agrupamiento jerárquico para datos de procesos



Dendrograma de agrupamiento jerárquico para datos de procesos



Validación del agrupamiento por Jerárquico

Externos

Homogeneidad: 0.239

Complejidad: 1.000

Medida V: 0.385

Índice Rand corregido: 0.000

Adjusted Mutual Information: 0.000

Índice Fowlkes-Mallows: 0.000

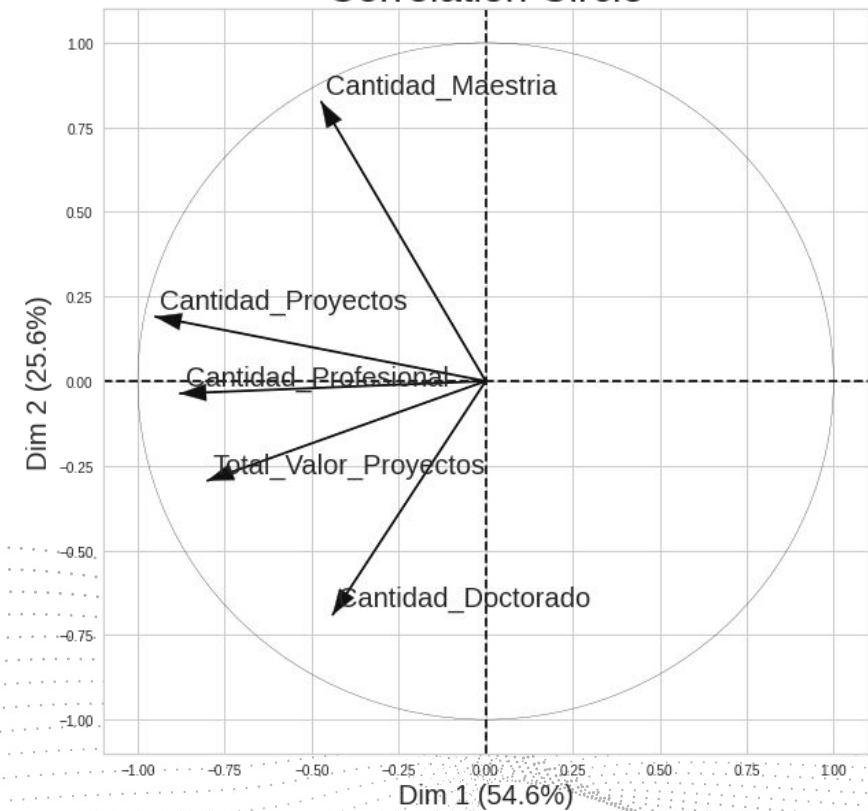
Internos

Coeficiente de Silhueta: 0.298

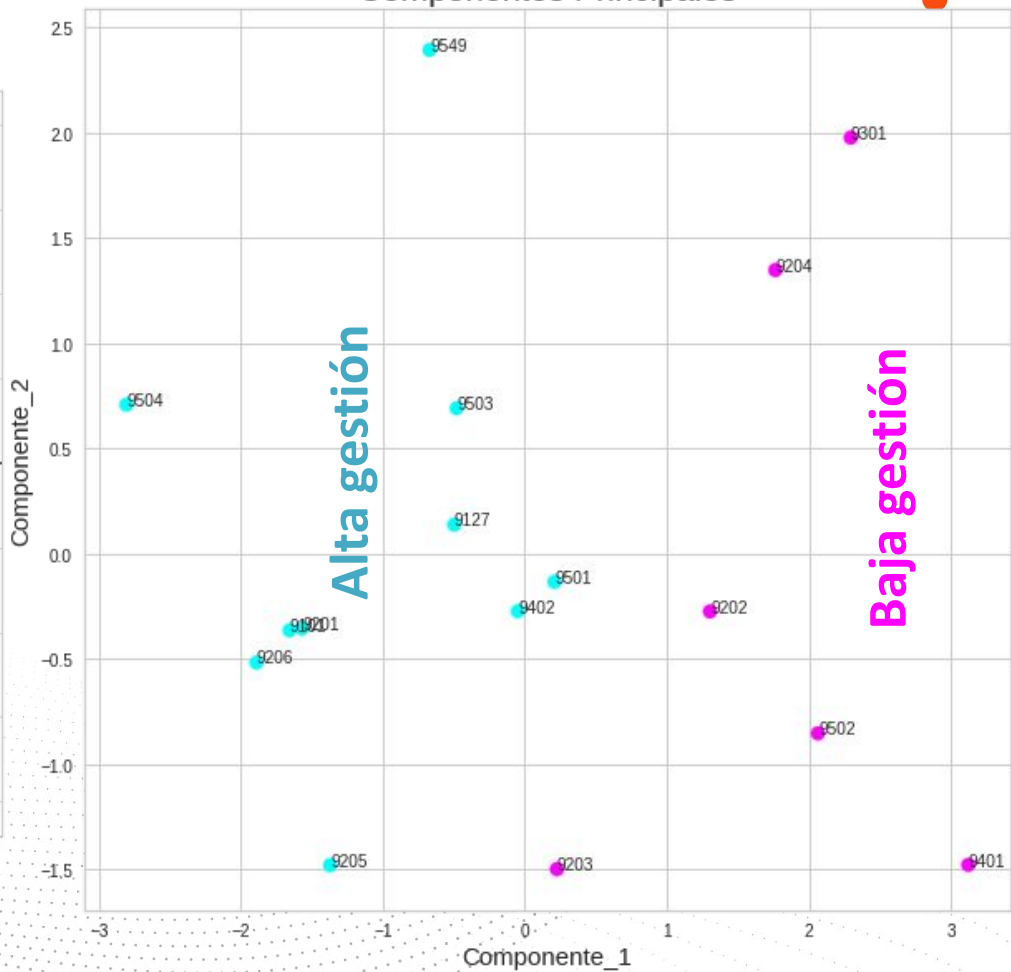
Calinski_harabasz: 9.140

Davies Bouldin: 1.169

Correlation Circle

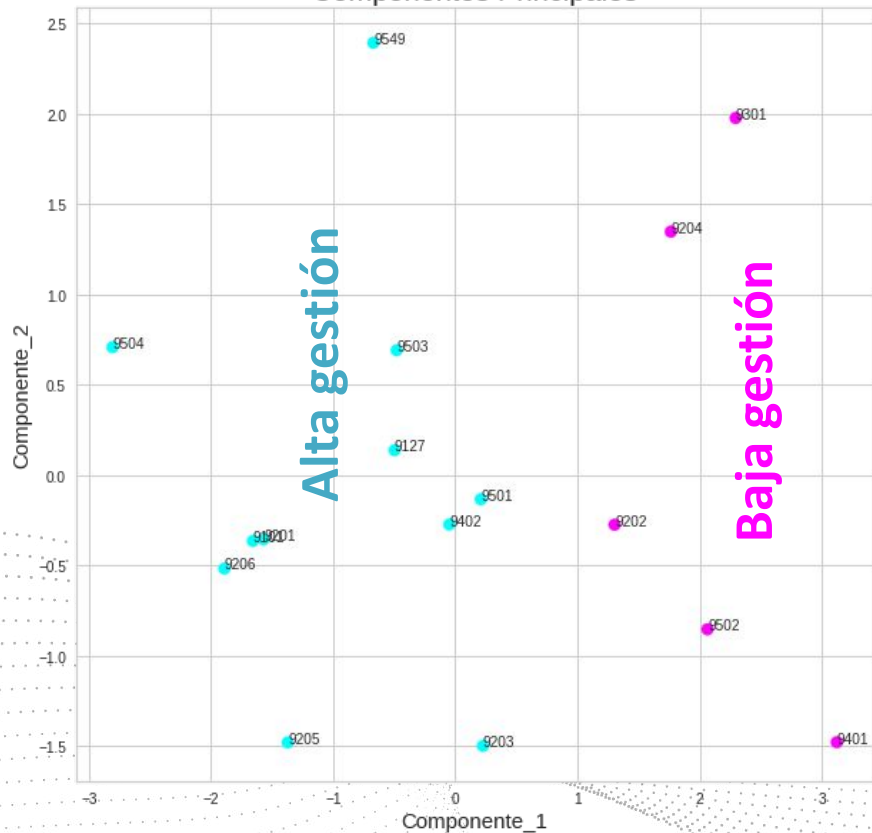


Componentes Principales



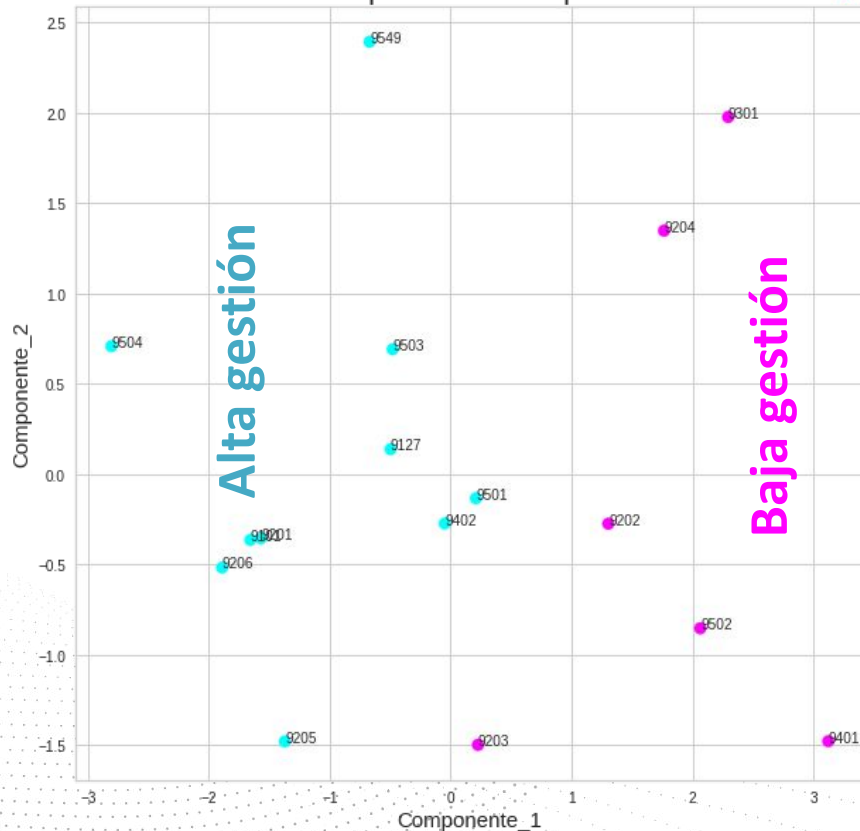
Kmeans

Componentes Principales



Jerárquico

Componentes Principales

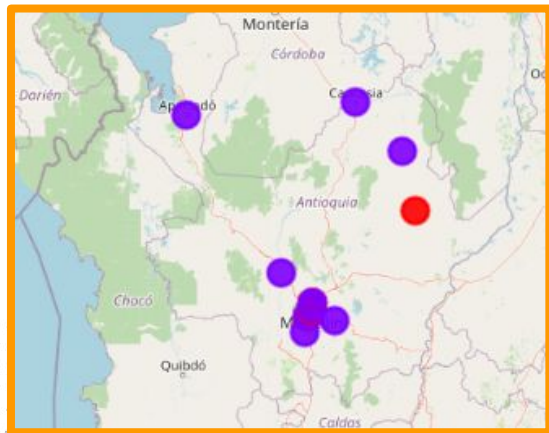
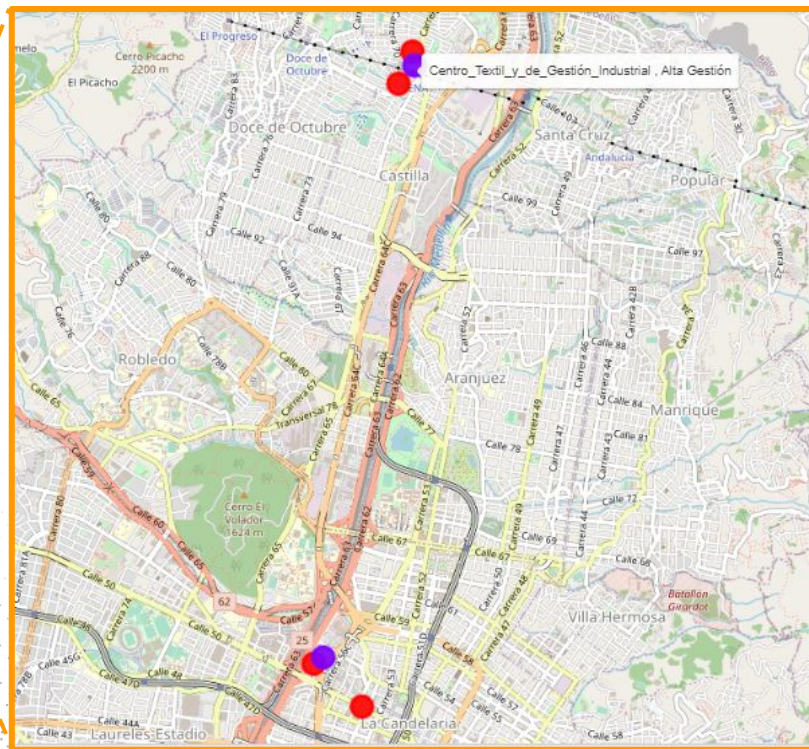


Codigo_Centro	Nombre_Centro	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	Total_valor_Proyectos	KMeans_Clusters	Gestión
9101	Centro_de_los_Recurso...s_Naturales_Renovables_La...	-1.264911	-0.198228	-1.161049	-0.904534	-0.520118	1	Alta Gestión
9127	Centro_de_Formación_Minero_Ambiental	0.000000	-0.651320	0.198228	0.301511	-0.594640	1	Alta Gestión
9201	Centro_del_Diseño_y_Manufactura_del_Cuero	-1.264911	-1.557505	-0.707957	-0.904534	0.885360	1	Alta Gestión
9202	Centro_de_Formación_en_Diseño_Confección_y_Moda	1.264911	-0.198228	0.651320	1.507557	0.197253	0	Baja Gestión
9203	Centro_para_el_Desarrollo_del_Hábitat_y_la_Con...	0.000000	-0.198228	-0.707957	1.507557	0.275916	0	Baja Gestión

Tabla de visualización en mapas

	Nombre_Centro	latitud	longitud	KMeans_Clusters	Gestión
0	Centro_de_los_Recurso...s_Naturales_Renovables_La...	6.053157	-75.628486	1	Alta Gestión
1	Centro_de_Formación_Minero_Ambiental	7.584361	-74.800660	1	Alta Gestión
2	Centro_del_Diseño_y_Manufactura_del_Cuero	6.180794	-75.605884	1	Alta Gestión
3	Centro_de_Formación_en_Diseño_Confección_y_Moda	6.181273	-75.605935	0	Baja Gestión
4	Centro_para_el_Desarrollo_del_Hábitat_y_la_Con...	6.303487	-75.567591	0	Baja Gestión
5	Centro_de_Tecnología_de_la_Manufactura_Avanzada	6.300949	-75.568664	0	Baja Gestión
6	Centro_Tecnológico_del_Mobiliario	6.180750	-75.606430	1	Alta Gestión
7	Centro_Textil_y_de_Gestión_Industrial	6.302387	-75.567365	1	Alta Gestión
8	Centro_de_Comercio	6.254794	-75.575466	0	Baja Gestión
9	Centro_de_Servicios_de_Salud	6.251389	-75.571571	0	Baja Gestión
10	Centro_de_Servicios_y_Gestión_Empresarial	6.255308	-75.574671	1	Alta Gestión
11	Complejo_Tecnológico_para_la_Gestión_Agroempre...	7.996707	-75.196365	1	Alta Gestión
12	Complejo_Tecnológico_Minero_Agroempresarial_	7.080799	-74.699840	0	Baja Gestión
13	Centro_de_la_Innovación_la_Agroindustria_y_la_...	6.153989	-75.372317	1	Alta Gestión
14	Complejo_Tecnológico_Agroindustrial_Pecuario_y...	7.890489	-76.631035	1	Alta Gestión
15	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128	1	Alta Gestión

Mapas de centros de formación en alta y baja gestión



-  Alta Gestión
-  Baja Gestión

Clusterización - Procesos 2010-2022

1. Kmeans - Validación - PCA
2. Agrupamiento jerárquico - Validación -PCA

Preprocesamiento

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	Total_Valor_Proyectos
0	9101-2010	-1.260308	-1.103402	-0.896032	-1.304164	-0.126344
1	9127-2010	-0.362554	-0.671184	-0.117850	-0.201456	-0.735003
2	9201-2010	-1.260308	-0.238966	-0.506941	-1.304164	-0.206320
3	9202-2010	0.535199	1.922122	0.271241	-1.304164	-0.608931
4	9203-2010	-0.362554	1.057686	-0.117850	-0.201456	-1.577040
...	...	...	...	...	...	...
203	9501-2022	-0.362554	0.193251	-0.117850	0.901251	-0.872848
204	9502-2022	-1.260308	-0.238966	-0.896032	-1.304164	-0.687234
205	9503-2022	-1.260308	-1.535619	-1.674214	-0.201456	0.218826
206	9504-2022	1.432953	1.489904	1.049423	-0.201456	2.333091
207	9549-2022	-0.362554	0.625469	-0.896032	-0.201456	-1.254502

208 rows x 6 columns

Tiempo: 13 años

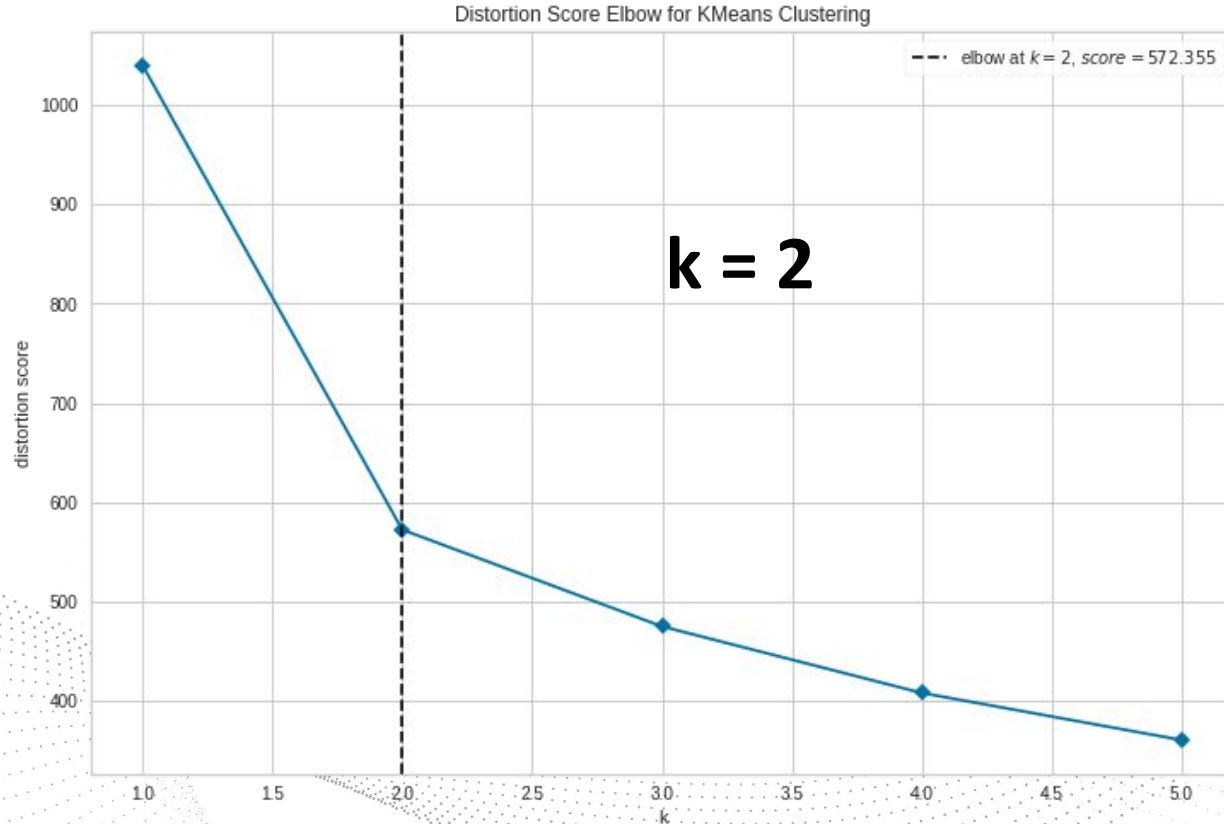
Centros de formación: 16

Columnas : 6

Filas: 208

KMeans - PCA

Método del codo : Determinación de centroides



Validación del agrupamiento por KMeans

Externos

Homogeneidad: 0.126

Compleitud: 1.000

Medida V: 0.224

Índice Rand corregido: 0.000

Adjusted Mutual Information: -0.000

Índice Fowlkes-Mallows: 0.000

Internos

Coeficiente de Silhueta: 0.385

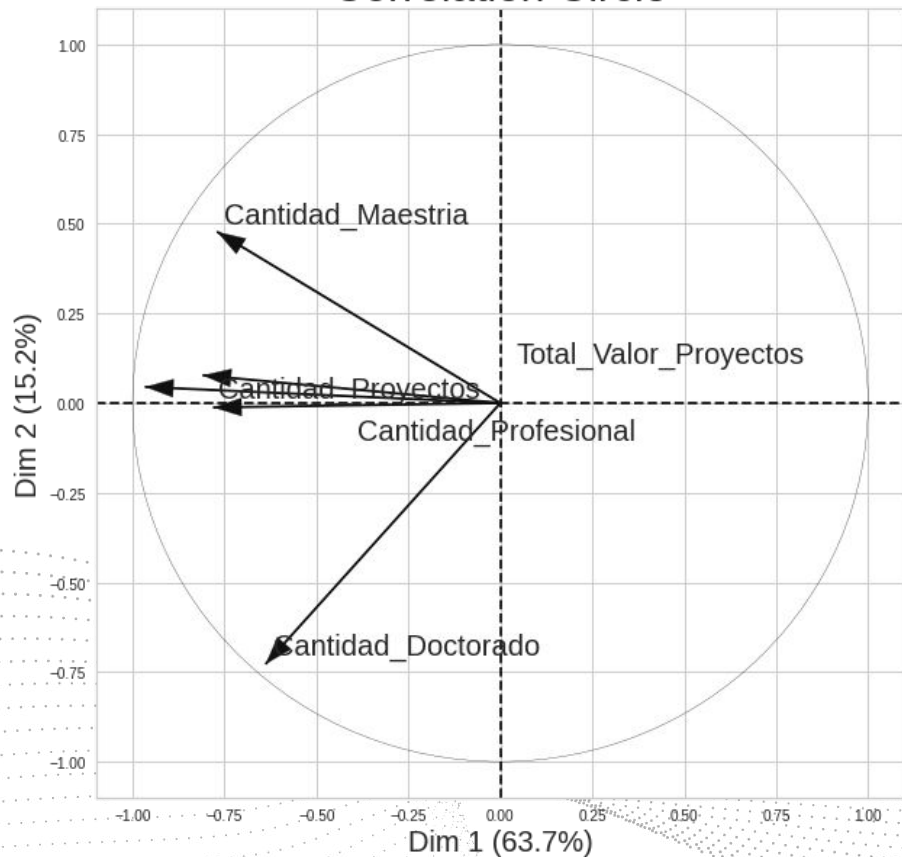
Calinski_harabasz: 168.313

Davies Bouldin: 1.050

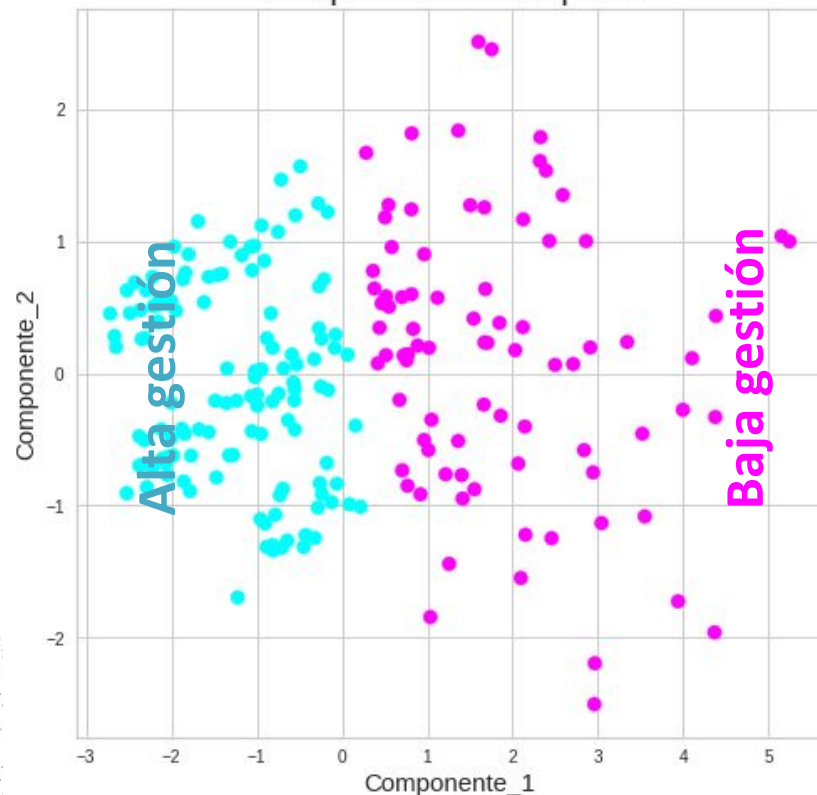
PCA CENTROS DE FORMACIÓN EN EL TIEMPO - KMeans



Correlation Circle



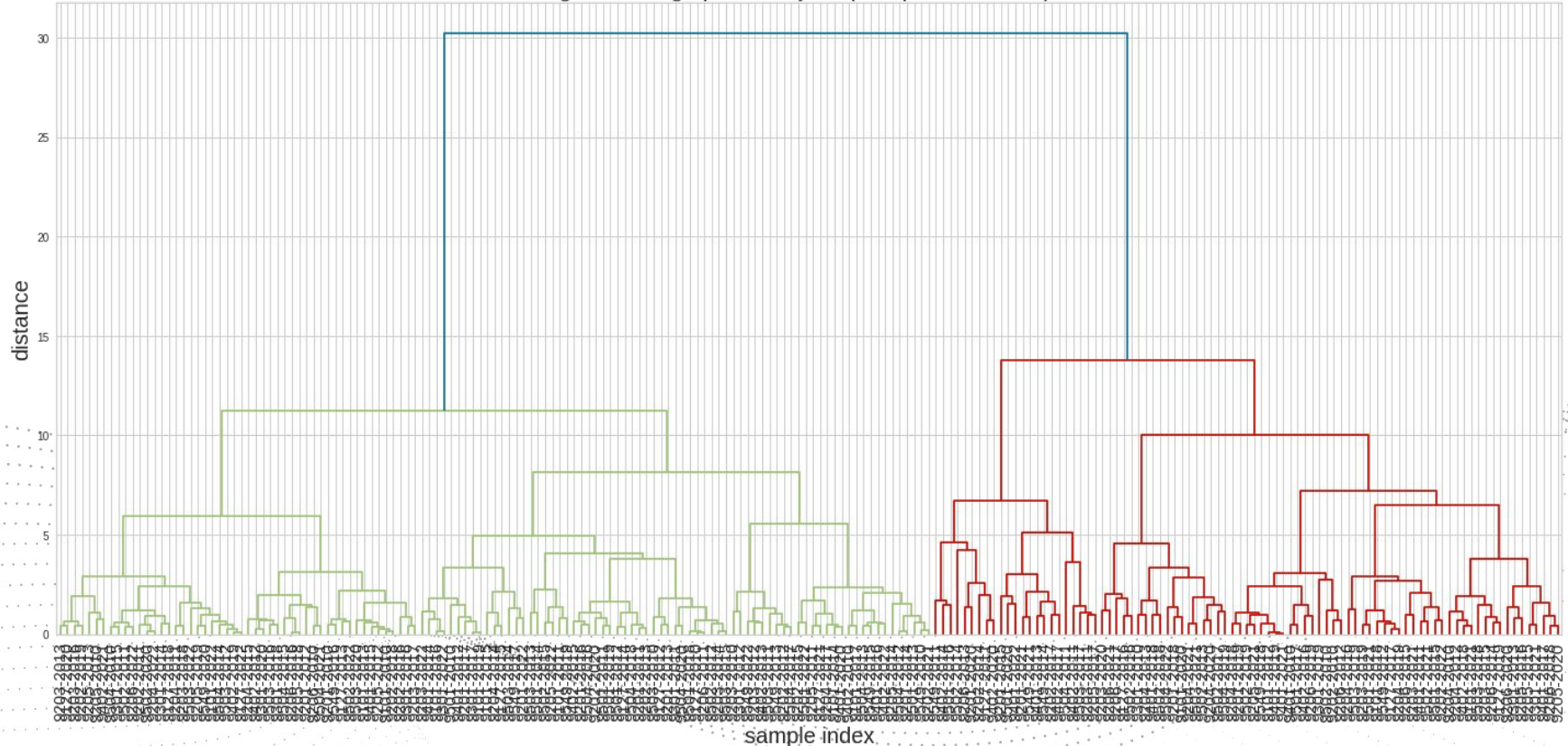
Componentes Principales



Jerárquico - PCA

Determinación de clusters - Dendrograma

Dendrograma de agrupamiento jerárquico para datos de procesos



Validación del agrupamiento por KMeans

Externos

Homogeneidad: 0.127

Compleitud: 1.000

Medida V: 0.226

Índice Rand corregido: 0.000

Adjusted Mutual Information: -0.000

Índice Fowlkes-Mallows: 0.000

Internos

Coeficiente de Silhueta: 0.372

Calinski_harabasz: 161.438

Davies Bouldin: 1.070

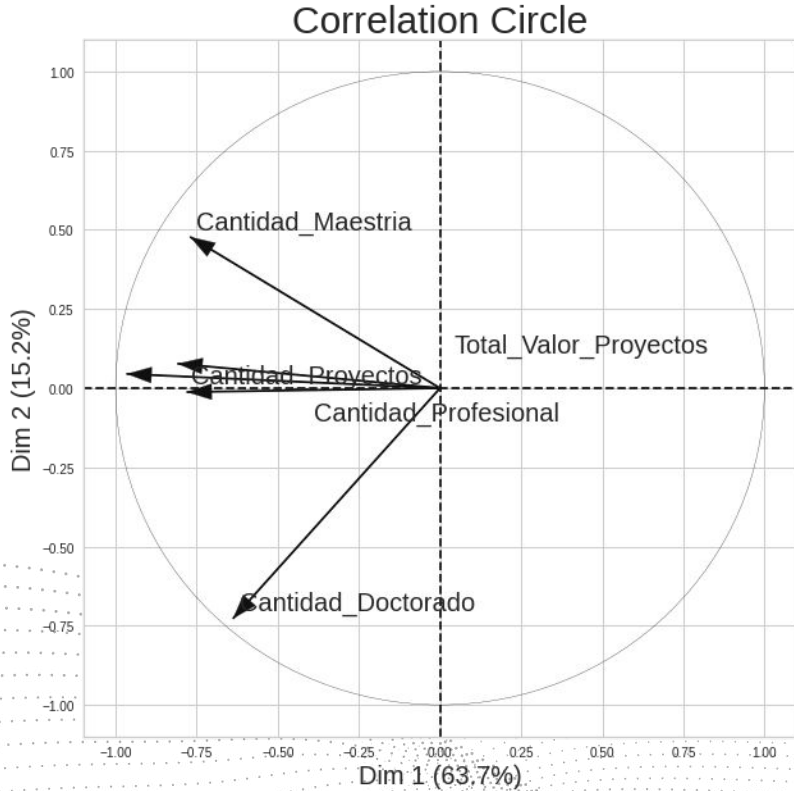
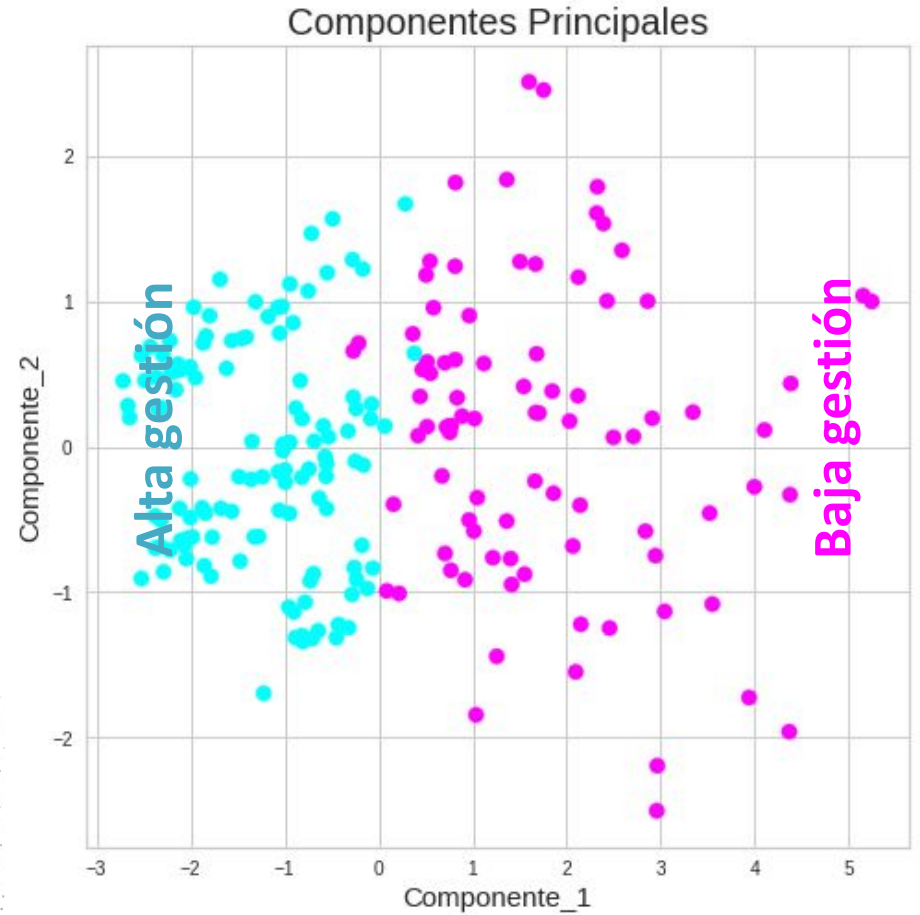


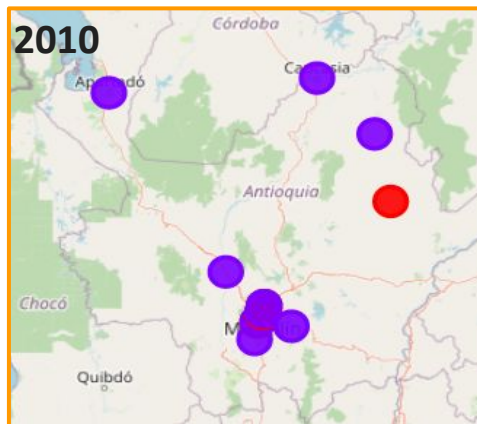
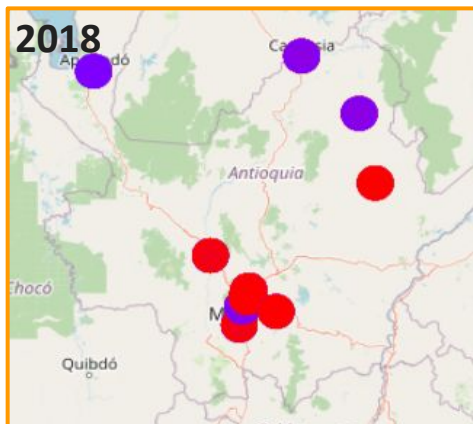
Tabla de datos final para visualización en mapas- Jerárquico-PCA



	Codigo_Centro	Año	KMeans_Clusters	Gestión	Nombre_Centro	latitud	longitud
0	9101	2010	1	Alta Gestión	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
1	9101	2011	1	Alta Gestión	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
2	9101	2012	0	Baja Gestión	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
3	9101	2013	1	Alta Gestión	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
4	9101	2014	1	Alta Gestión	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
...	...	...	...	...	...	...	...
203	9549	2018	0	Baja Gestión	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
204	9549	2019	1	Alta Gestión	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
205	9549	2020	1	Alta Gestión	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
206	9549	2021	0	Baja Gestión	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
207	9549	2022	1	Alta Gestión	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128

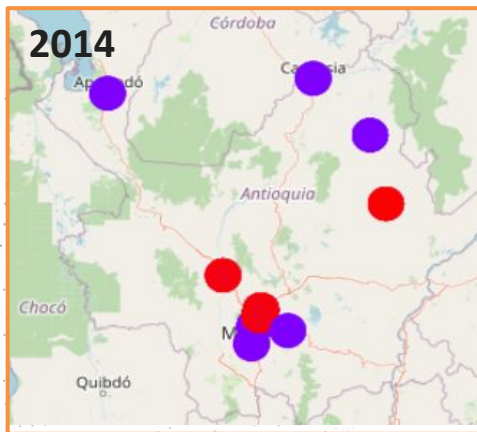
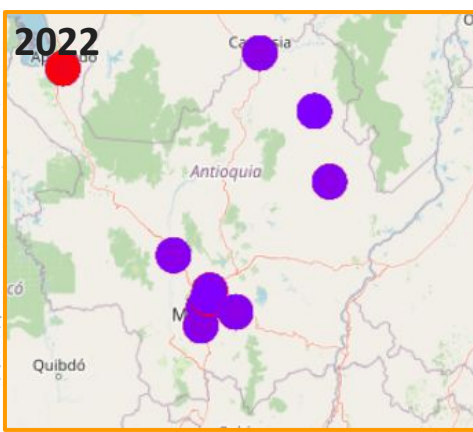
208 rows × 7 columns

Mapas de centros de formación en alta y baja gestión a lo largo del tiempo - Cluster Jerárquico - PCA



● Alta Gestión

● Baja Gestión





Clusterización - Productividad



@SENAcomunica

www.sena.edu.co

Clusterización - Productividad 2010-2022

1. Kmeans - Validación - PCA
2. Agrupamiento jerárquico - Validación -PCA

Preprocesamiento

	Centro_Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101-2010	1	1	0	0	0	1
1	9127-2010	0	0	0	0	1	0
2	9201-2010	0	0	0	0	2	0
3	9202-2010	2	1	0	0	2	0
4	9203-2010	1	1	0	1	1	1
...	...	...	...	...	...	...	...
203	9501-2022	1	1	1	2	3	1
204	9502-2022	2	1	3	1	3	1
205	9503-2022	3	2	4	1	1	2
206	9504-2022	1	2	2	4	1	4
207	9549-2022	5	3	1	2	2	4

208 rows × 7 columns

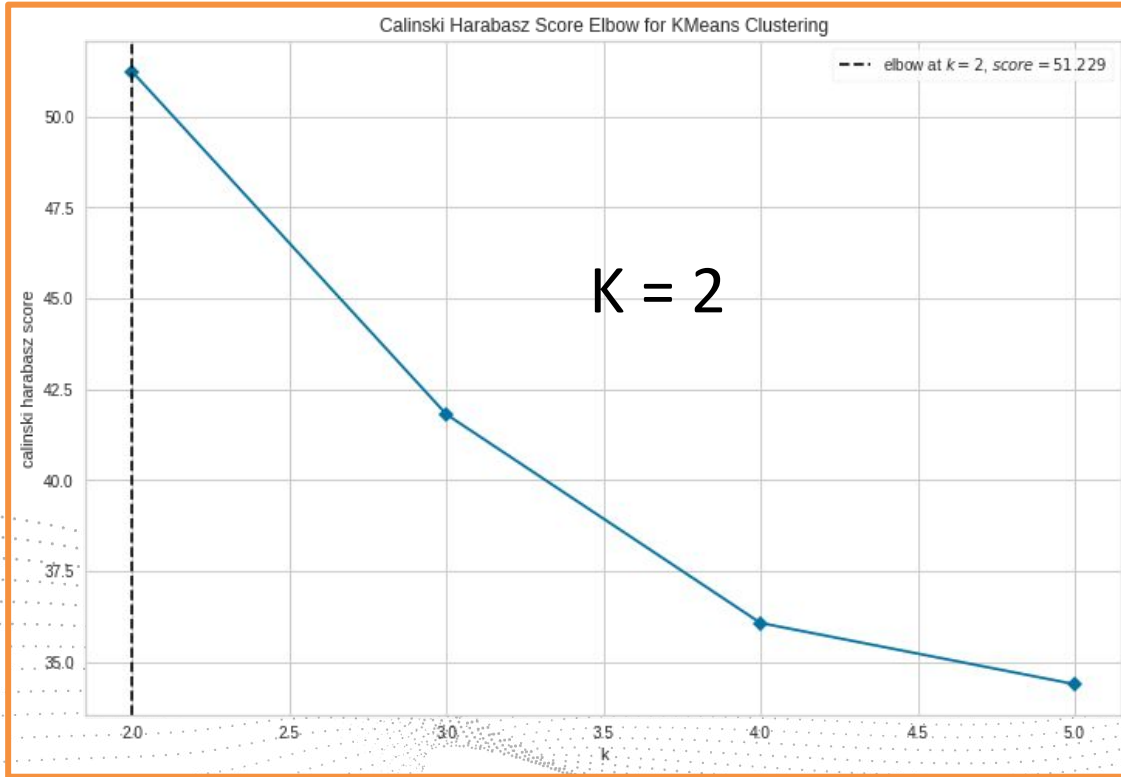
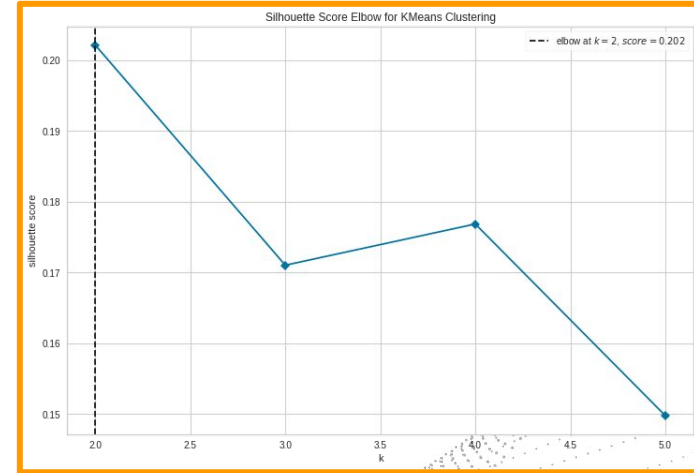
	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación_Pública_de_la_
0	9101-2010	-0.609017	-0.526118	
1	9127-2010	-1.337038	-1.201628	
2	9201-2010	-1.337038	-1.201628	
3	9202-2010	0.119003	-0.526118	
4	9203-2010	-0.609017	-0.526118	
...	...	...	...	
203	9501-2022	-0.609017	-0.526118	
204	9502-2022	0.119003	-0.526118	
205	9503-2022	0.847024	0.149392	
206	9504-2022	-0.609017	0.149392	
207	9549-2022	2.303066	0.824901	

208 rows × 7 columns

Tiempo: 13 años
Centros de formación :16
Columnas : 6
Filas: 208

KMeans - PCA

Método del codo : Determinación de centroides



Validación del agrupamiento por KMeans

Externos

Homogeneidad: 0.126

Compleitud: 1.000

Medida V: 0.223

Índice Rand corregido: 0.000

Adjusted Mutual Information: -0.000

Índice Fowlkes-Mallows: 0.000

Internos

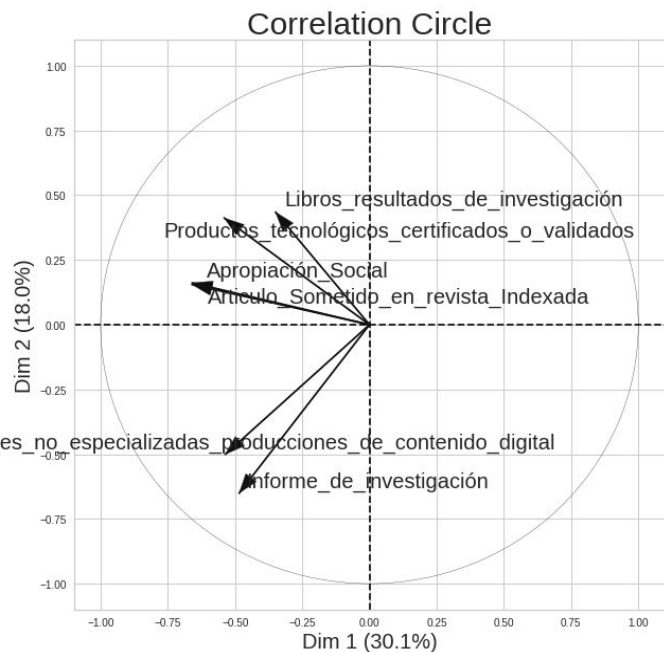
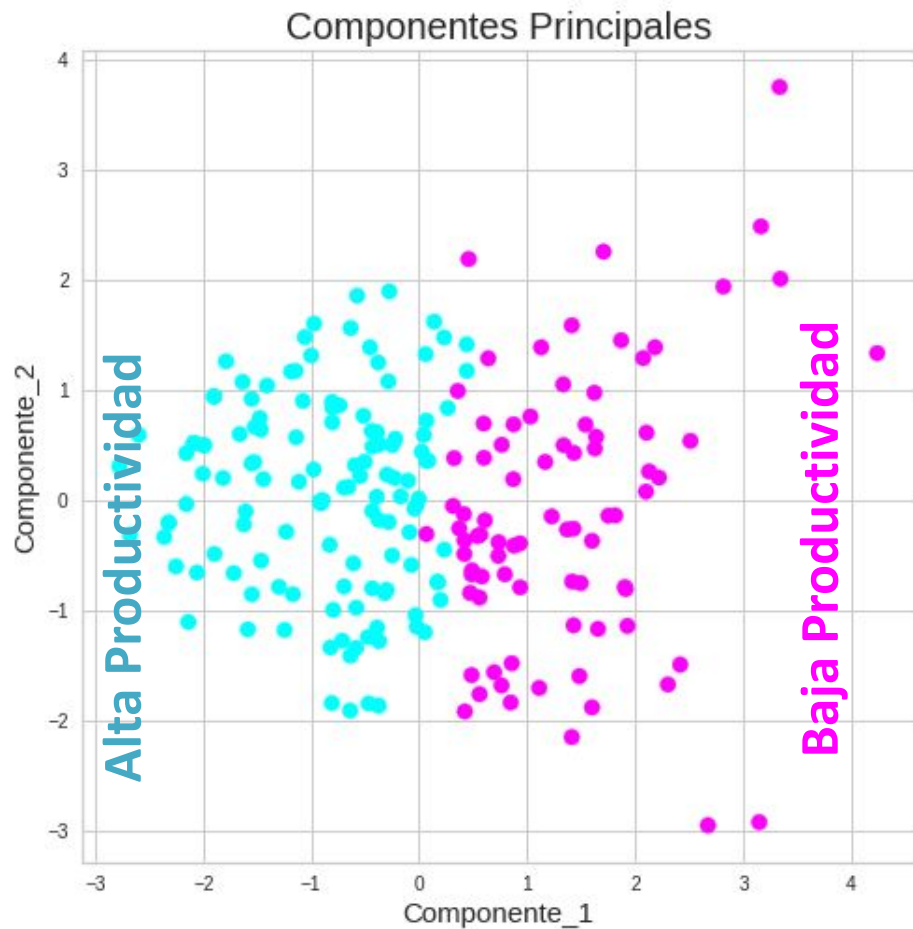
Coeficiente de Silhueta: 0.202

Calinski_harabasz: 51.229

Davies Bouldin: 1.916

PCA CENTROS DE FORMACIÓN EN EL TIEMPO - KMeans



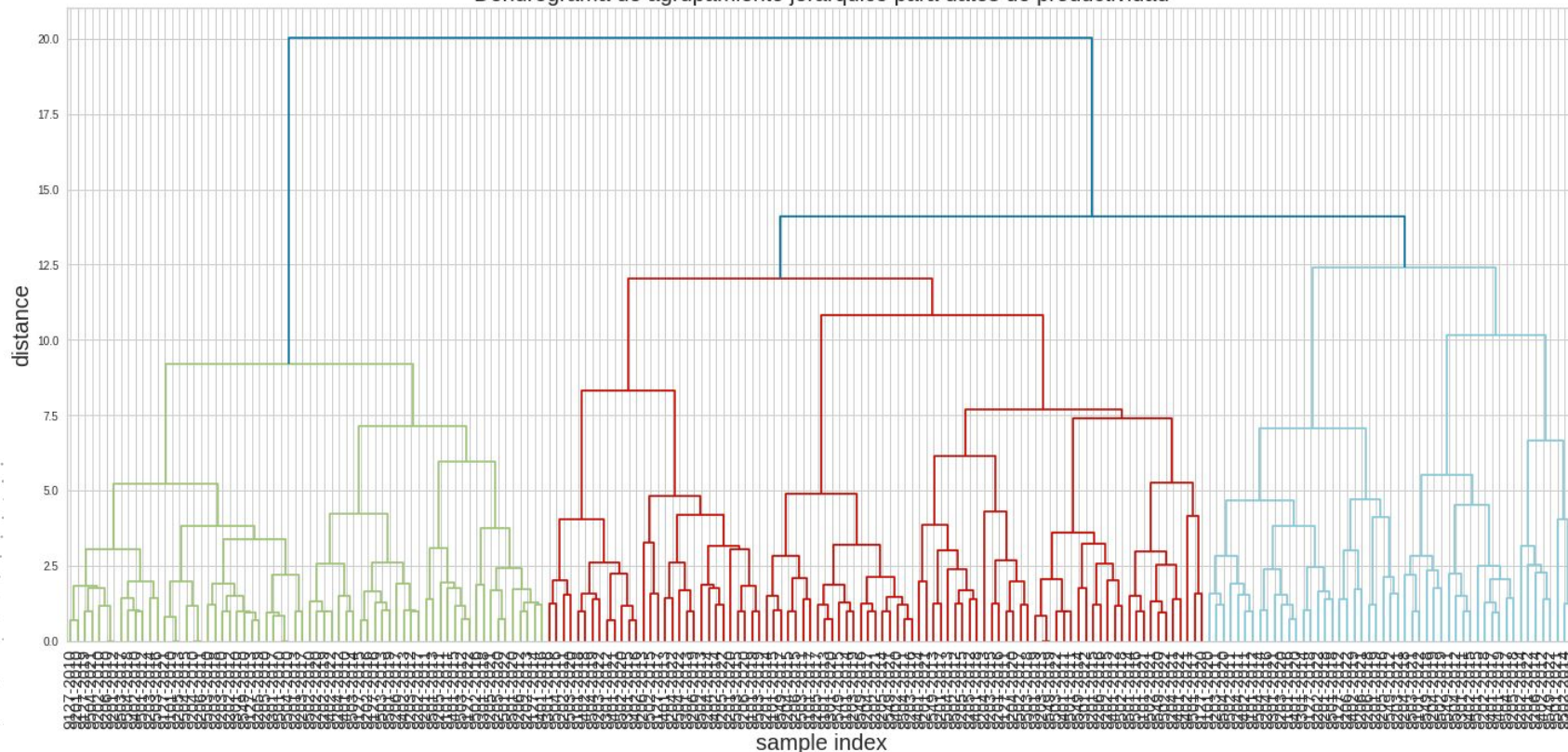


Jerárquico - PCA

Determinación de clusters -Dendrograma



Dendrograma de agrupamiento jerárquico para datos de productividad



Validación del agrupamiento por Jerárquico

Externos

Homogeneidad: 0.117

Complejidad: 1.000

Medida V: 0.210

Índice Rand corregido: 0.000

Adjusted Mutual Information: 0.000

Índice Fowlkes-Mallows: 0.000

Internos

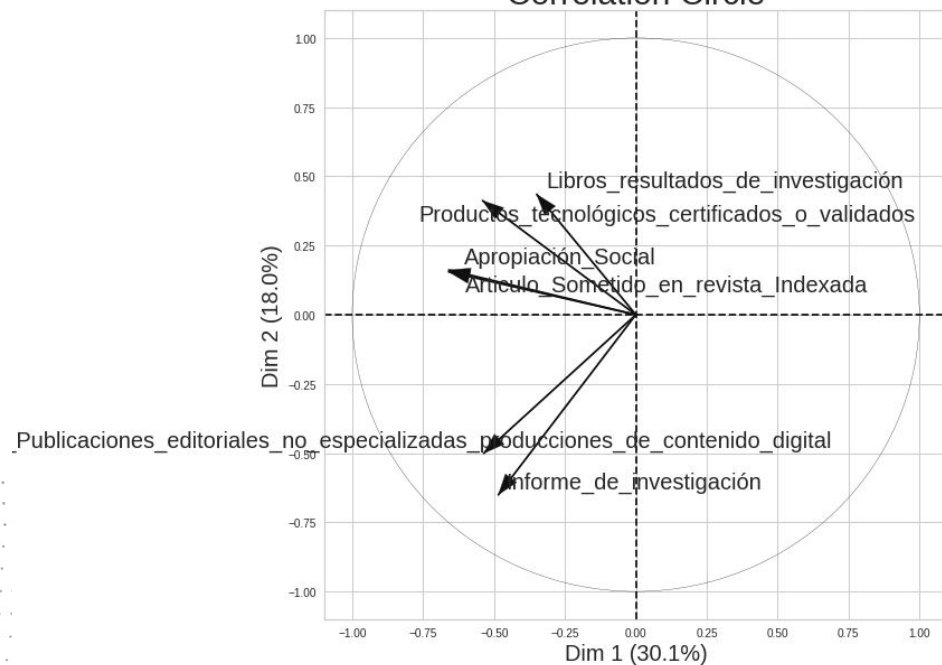
Coeficiente de Silhueta: 0.133

Calinski_harabasz: 39.485

Davies Bouldin: 1.874



Correlation Circle



Componentes Principales

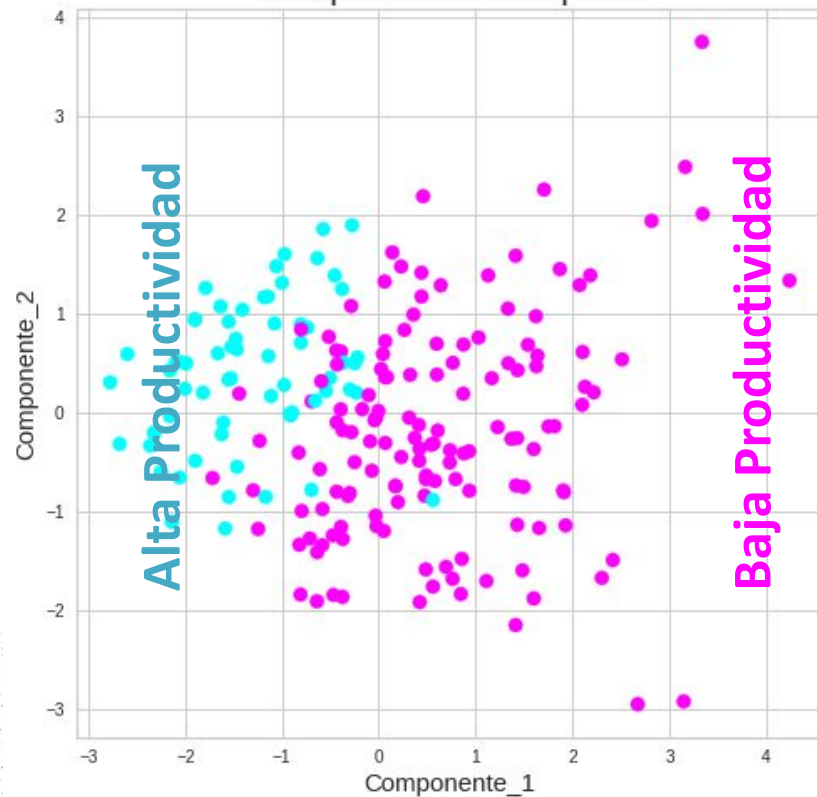


Tabla de datos final para visualización en mapas

Jerárquico-PCA

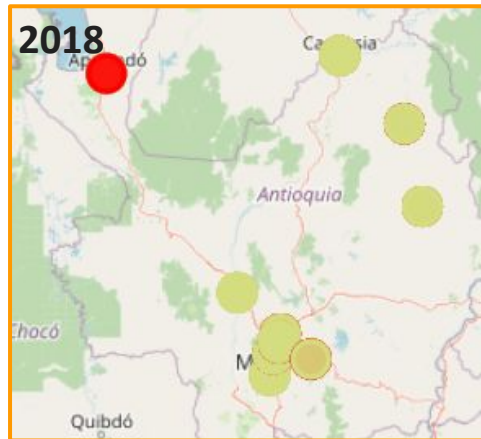
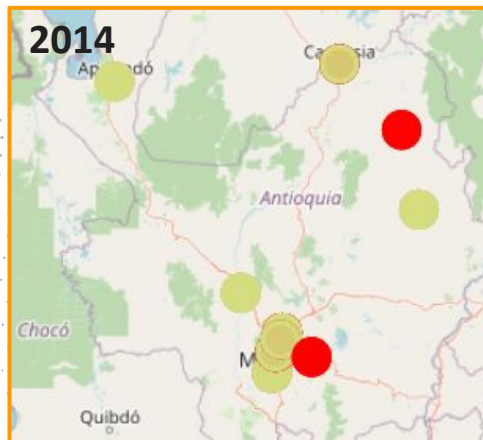
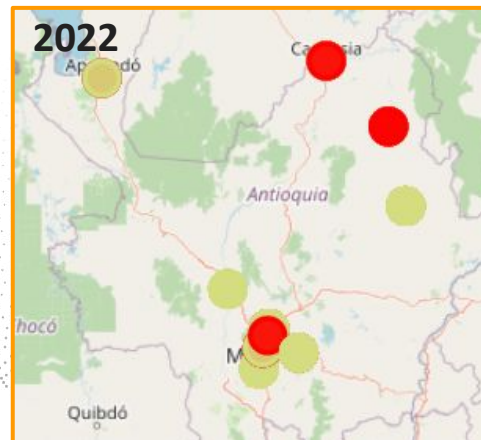


	Codigo_Centro	Año	KMeans_Clusters	Productividad	Nombre_Centro	latitud	longitud
0	9101	2010	1	Baja Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
1	9101	2011	1	Baja Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
2	9101	2012	0	Alta Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
3	9101	2013	0	Alta Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
4	9101	2014	0	Alta Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
...	...	...	...	...	...	...	...
203	9549	2018	0	Alta Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
204	9549	2019	0	Alta Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
205	9549	2020	0	Alta Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
206	9549	2021	0	Alta Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
207	9549	2022	0	Alta Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128

208 rows × 7 columns

Mapas de centros de formación en alta y baja gestión a lo largo del tiempo -

Cluster Jerárquico - PCA



 Alta Productividad

 Baja Productividad



MÉTODOS DE CLASIFICACIÓN

Machine Learning Supervisado



@SENAcomunica

www.sena.edu.co



PRODUCTIVIDAD



@SENAcomunica

www.sena.edu.co



Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
9101	2010	Baja Productividad	1	1	0	0	0	1
9127	2010	Baja Productividad	0	0	0	0	1	0
9201	2010	Baja Productividad	0	0	0	0	2	0
9202	2010	Baja Productividad	2	1	0	0	2	0
9203	2010	Baja Productividad	1	1	0	1	1	1

1: Alta Productividad
-1: Baja Productividad

	Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101	2010	-1	1	1	0	0	0	1
1	9127	2010	-1	0	0	0	0	1	0
2	9201	2010	-1	0	0	0	0	2	0
3	9202	2010	-1	2	1	0	0	2	0
4	9203	2010	-1	1	1	0	1	1	1
...	...	...	...	...	...	...	...	...	...
203	9501	2022	-1	1	1	1	2	3	1
204	9502	2022	1	2	1	3	1	3	1
205	9503	2022	1	3	2	4	1	1	2

Matriz de confusión y parámetros de validación



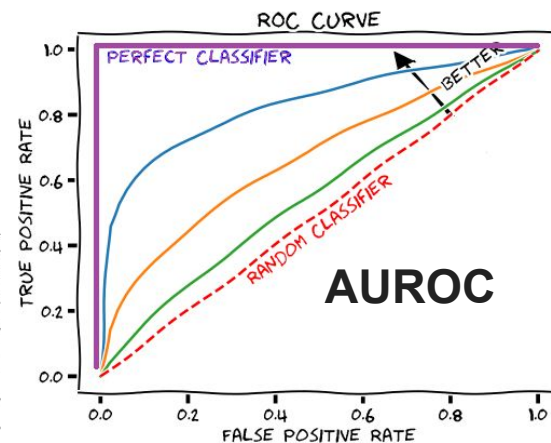
		Actual Values	
		Yes	No
Predicted Values	Yes	True Positive	False Positive
	No	False Negative	True Negative

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

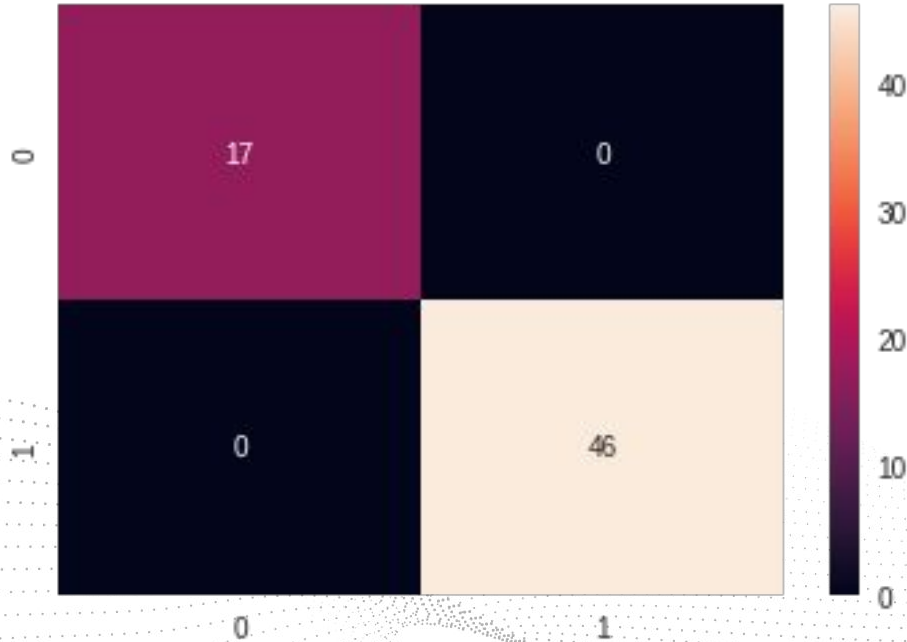
$$Recall = \frac{TP}{TP + FN}$$

$$F1-score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$



Regresión Logística

(y_test, y_reglog_pred)



Accuracy: 0.9841

AUROC: 0.9773

F1-Score: 0.984

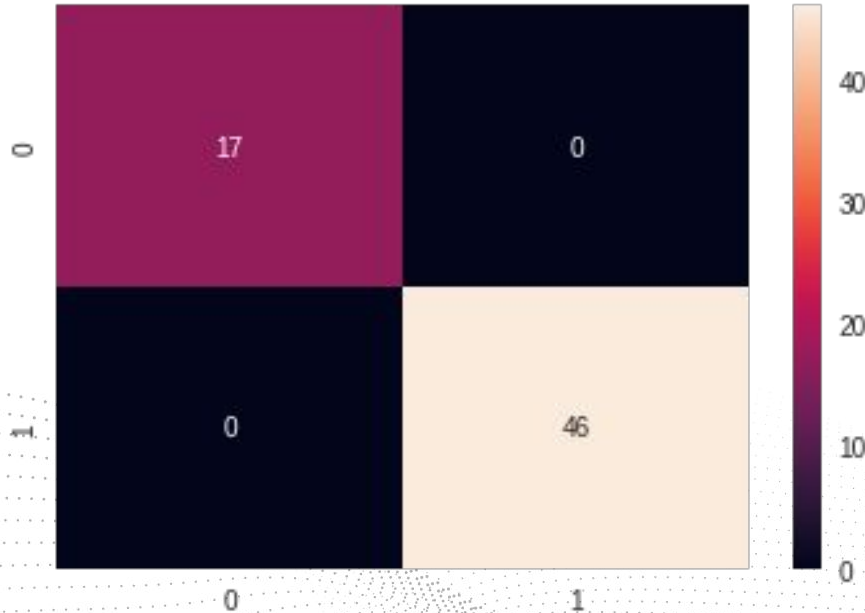
Regresión Logística

df_sample

	Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101.0	2010.0	1.0	10.0	5.0	5.0	5.0	5.0	5.0
1	9127.0	2010.0	1.0	5.0	5.0	5.0	5.0	5.0	5.0
2	9201.0	2010.0	-1.0	0.0	0.0	0.0	0.0	0.0	1.0
3	9202.0	2010.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0
4	9203.0	2010.0	-1.0	0.0	1.0	0.0	1.0	2.0	0.0
5	9504.0	2022.0	1.0	12.0	12.0	12.0	12.0	12.0	12.0

Random Forest

(y_test, y_rf_pred)



Accuracy: 1.0

AUROC: 1.0

F1-Score: 1.0

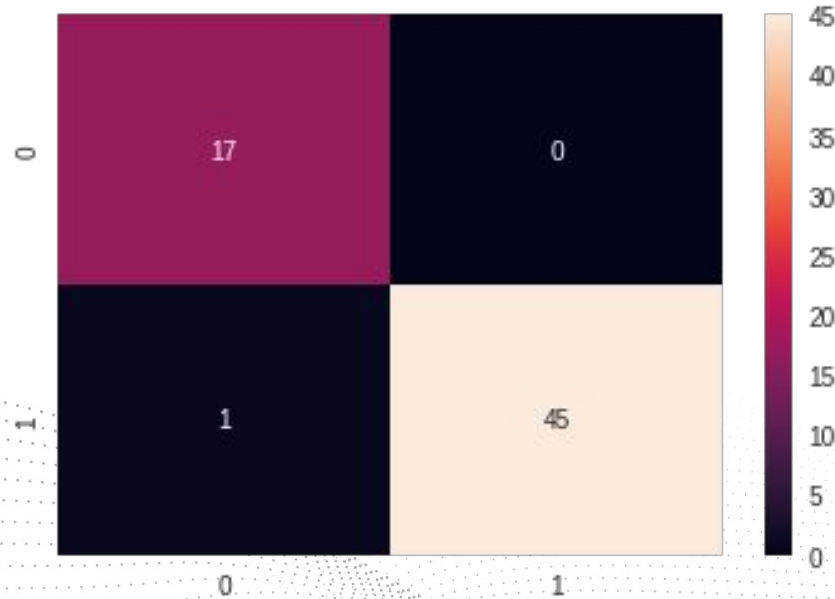
Random Forest

df_sample

	Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101.0	2010.0	1.0	10.0	5.0	5.0	5.0	5.0	5.0
1	9127.0	2010.0	1.0	5.0	5.0	5.0	5.0	5.0	5.0
2	9201.0	2010.0	-1.0	0.0	0.0	0.0	0.0	0.0	1.0
3	9202.0	2010.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0
4	9203.0	2010.0	-1.0	0.0	1.0	0.0	1.0	2.0	0.0
5	9504.0	2022.0	1.0	12.0	12.0	12.0	12.0	12.0	12.0

Support vector machine(SVM)

(y_test, y_svm_pred)



Accuracy: 0.9841

AUROC: 0.9891

F1-Score: 0.9842

Support vector machine(SVM)

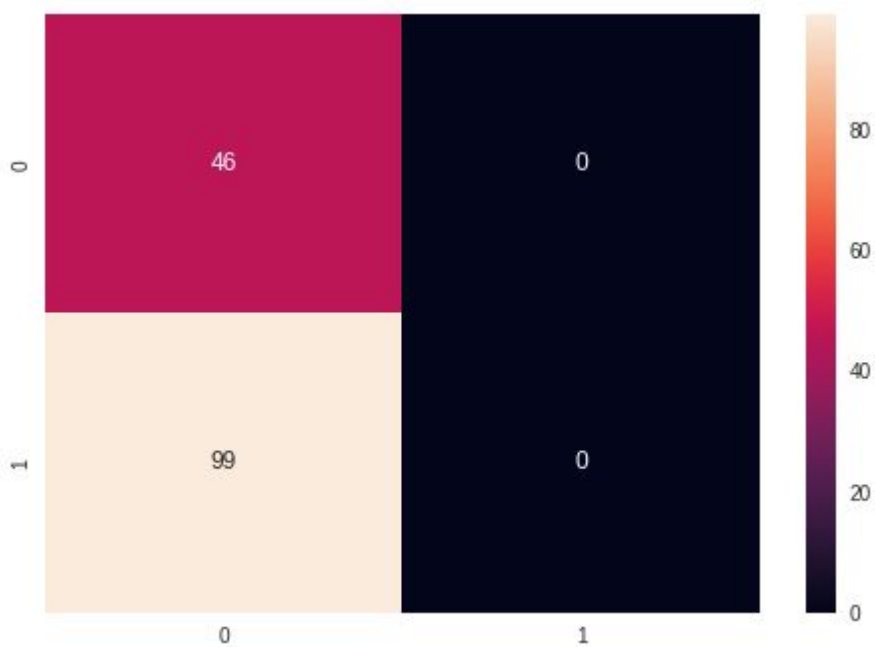
df_sample

	Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101.0	2010.0	1.0	10.0	5.0	5.0	5.0	5.0	5.0
1	9127.0	2010.0	1.0	5.0	5.0	5.0	5.0	5.0	5.0
2	9201.0	2010.0	-1.0	0.0	0.0	0.0	0.0	0.0	1.0
3	9202.0	2010.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0
4	9203.0	2010.0	-1.0	0.0	1.0	0.0	1.0	2.0	0.0
5	9504.0	2022.0	1.0	12.0	12.0	12.0	12.0	12.0	12.0

(y_test, y_svm_pred)

Red neuronal: Perceptrón

```
df_sample['Productividad'], p.predict(df_sample)
```



Accuracy: 0.3172

AUROC: 0.5

F1-Score: 0.1528

Red neuronal: Perceptrón

df_sample

	Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101.0	2010.0	1.0	10.0	5.0	5.0	5.0	5.0	5.0
1	9127.0	2010.0	1.0	5.0	5.0	5.0	5.0	5.0	5.0
2	9201.0	2010.0	-1.0	0.0	0.0	0.0	0.0	0.0	1.0
3	9202.0	2010.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0
4	9203.0	2010.0	-1.0	0.0	1.0	0.0	1.0	2.0	0.0
5	9504.0	2022.0	1.0	12.0	12.0	12.0	12.0	12.0	12.0

```
p.predict(df_sample)
```

¡Todos con alta productividad!

```
array([1, 1, 1, 1, 1, 1])
```



PROCESOS



@SENAcomunica

www.sena.edu.co



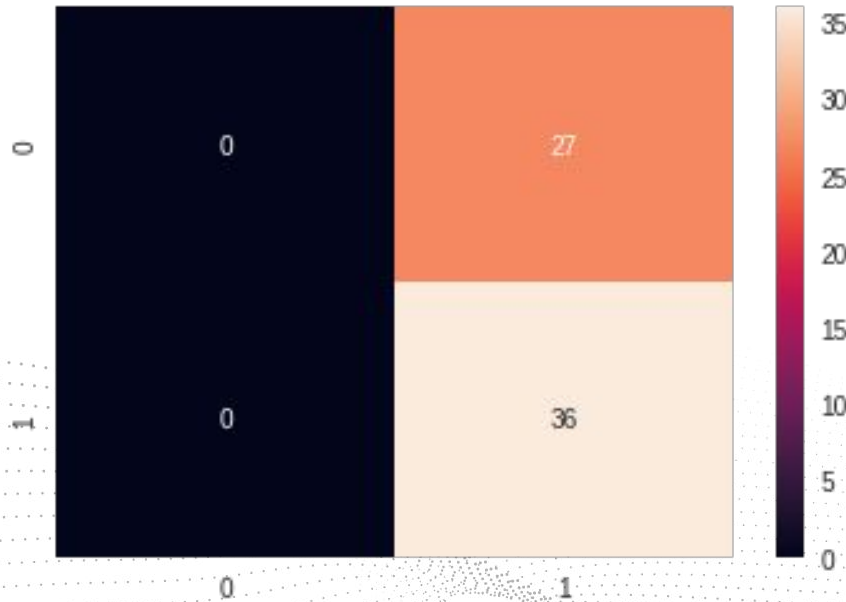
Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101	2010	Alta Gestión	1	1	2	0	4.652924e+08
9127	2010	Alta Gestión	2	2	4	1	3.920052e+08
9201	2010	Alta Gestión	1	3	3	0	4.556627e+08
9202	2010	Baja Gestión	3	8	5	0	4.071852e+08
9203	2010	Alta Gestión	2	6	4	1	2.906175e+08

1: Alta Gestión
-1: Baja Gestión

Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101	2010	1	1	1	2	0	4.652924e+08
9127	2010	1	2	2	4	1	3.920052e+08
9201	2010	1	1	3	3	0	4.556627e+08
9202	2010	-1	3	8	5	0	4.071852e+08
9203	2010	1	2	6	4	1	2.906175e+08

Regresión Logística

(y_test, y_reglog_pred)



Accuracy: 0.5714

AUROC: 0.5

F1-Score: 0.4155

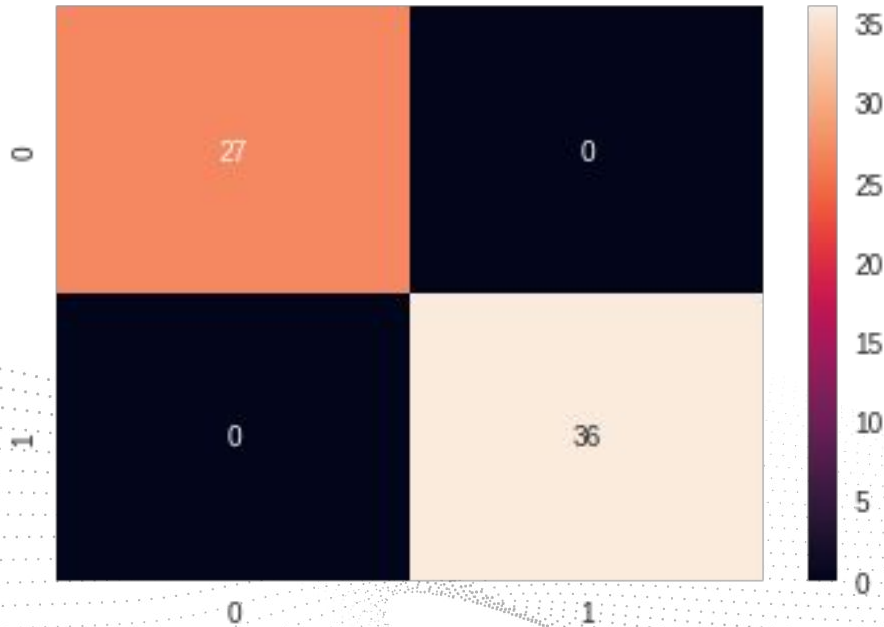
Regresión Logística

df_sample

	Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
0	9101	2010	1	1	1	2	0	465292400000000
1	9127	2010	1	2	2	4	1	392005200000000
2	9201	2010	1	1	3	3	0	455662700000000
3	9202	2010	-1	3	8	5	0	407185200000000
4	9203	2010	1	2	6	4	1	290617500000000

Random Forest

(y_test, y_dt_pred)



Accuracy: 1.0

AUROC: 1.0

F1-Score: 1.0

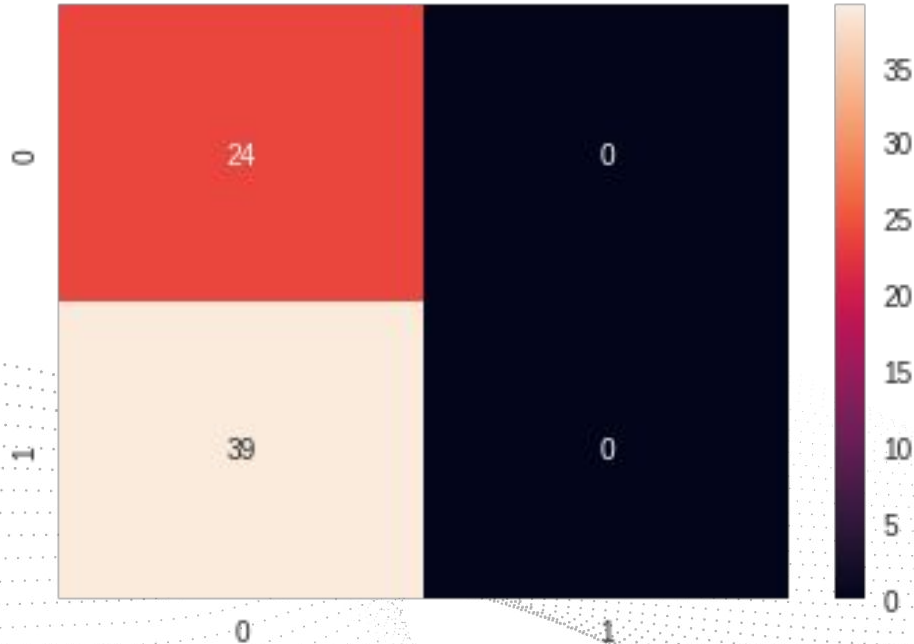
Random Forest

df_sample

	Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
0	9101	2010	1	1	1	2	0	4652924000000000
1	9127	2010	1	2	2	4	1	3920052000000000
2	9201	2010	1	1	3	3	0	4556627000000000
3	9202	2010	-1	3	8	5	0	4071852000000000
4	9203	2010	1	2	6	4	1	2906175000000000

Red neuronal: Perceptrón

(y_test, y_perc_pred)



Accuracy: 0.3809

AUROC: 0.5

F1-Score: 0.2101

Red neuronal: Perceptrón

df_sample

	Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
0	9101	2010	1	1	1	2	0	4652924000000000
1	9127	2010	1	2	2	4	1	3920052000000000
2	9201	2010	1	1	3	3	0	4556627000000000
3	9202	2010	-1	3	8	5	0	4071852000000000
4	9203	2010	1	2	6	4	1	2906175000000000



Contenido

1. Base de datos procesos
2. Base de datos productividad
3. Modelo de analítica de datos para medición de gestión por centros
 - Clústering método k-means para medir nivel de gestión
 - Clustering K-means y PCA para medir nivel de gestión
 - Representación en el plano factorial de variables de interés



@SENAcomunica

www.sena.edu.co



Contenido

3. Modelo de analítica de datos para medición de gestión por centros

- Dendograma de centros para medición de gestión
- Clustering jerárquico para medir nivel de gestión

4. Modelo de analítica de datos para medición de productividad por centros

- Clustering método k-means para medir nivel de productividad





Contenido

4. Modelo de analítica de datos para medición de productividad por centros

- Clustering K-means y PCA para medir nivel de productividad
- Representación en el plano factorial de variables de interés
- Dendograma de centros para medición de productividad
- Clustering jerárquico para medir nivel de productividad



1. Base de datos Procesos

A continuación se especifica el código en R statistics de como generar la tabla de procesos

Generación de la tabla de procesos

La tabla de procesos "TPProceso", contiene la información de las fechas en las que se registraron productos desde el 2010 hasta el 2022, además del Código_Centro, Q_Proyectos: cantidad de proyectos, QNA_5007: Profesional, QNA_5009: Profesional_con_maestría, QNA_5010: Profesional_con_doctorado, y TV_Proyectos: Total valor proyecto. Esta tabla se genera a partir de la Tabla_Principal_Proyectos, Principal_Talento_Humano y Nivel_Academico

```
Anio <- c(2010:2022) # Año
#
# Tabla_Proceso por Año, Centro, y Proyectos
TPProceso <- as.data.frame(matrix(data=NA, ncol = 7, nrow = 0))
names(TPProceso) <- c("Fecha",
                      "Codigo_Centro",
                      "Q_Proyectos",
                      "QNA_5007",
                      "QNA_5009",
                      "QNA_5010",
                      "TV_Proyectos")

paste("QNA", Nivel_Academico$Codigo_Nivel_Academico[c(7,9,10)], sep = "-"),
# Qty_Nivel_Academico (2) // Profesional, Maestría,
Doctorado
"TV_Proyectos" # Principal_Proyectos
TP <- TPProceso

for(y in 1:length(2010:2022)) # Por año
{
  PyY <- subset(Principal_Proyectos,
                year(Principal_Proyectos$Fecha_inicio_Proyecto) ==
Anio[y],
                select =
c(colnames(Principal_Proyectos[c(1:dim(Principal_Proyectos)[2])]))
# table(PyY$Codigo_Centro) # verificación
for(c in 1:dim(Centros_Formacion)[1]) # Por Centro
{
  PyC <- subset(PyY, PyY$Codigo_Centro ==
Centros_Formacion$Codigo_Centro[c],
                select = c(colnames(PyY[c(1:dim(PyY)[2])]))
  QPy <- dim(PyC)[1] # Qty Proyectos
  P <- TP
  P[1,] <- NA
  P[1,1] <- Anio[y] # Fecha_inicio_Proyecto
  P[1,2] <- Centros_Formacion$Codigo_Centro[c] # Codigo_Centro
```

```
P[3,] <- QPy # Qty Proyectos
TH0 <- as.data.frame(matrix(data=NA, ncol = 11, nrow = 0))
names(TH0) <- c("Codigo_Persona", "Codigo_Nivel_Academico",
               "Codigo_Tipo_Contrato", "Objeto_Contrato",
               "Codigo_Rubro", "Codigo_Area_OCDE", "Codigo_Estado_Contratacion",
               "Codigo_Regional", "Codigo_Centro", "Codigo_Proyecto", "Codigo_Producto")
}
TH <- TH0
for(p in 1:dim(PyC)[1])
{
  TH0 <- subset(Principal_Talento_Humano,
                Principal_Talento_Humano$Codigo_Proyecto ==
PyC$Codigo_Proyecto[p],
                select =
c(colnames(Principal_Talento_Humano[c(1:dim(Principal_Talento_Humano)[2])]))
)})
TH <- rbind(TH, TH0)
}
# table(TH$Codigo_Nivel_Academico) # verificación
QTHPPy <- subset(TH, TH$Codigo_Nivel_Academico == 5007, # Profesional
                select = c(colnames(TH[c(1:dim(TH)[2])]))
QTHMPy <- subset(TH, TH$Codigo_Nivel_Academico == 5009, # Maestría
                select = c(colnames(TH[c(1:dim(TH)[2])]))
QTHDPy <- subset(TH, TH$Codigo_Nivel_Academico == 5010, # Doctorado
                select = c(colnames(TH[c(1:dim(TH)[2])]))
P[4,] <- dim(QTHPPy)[1] # Qty Profesional (5007)
P[5,] <- dim(QTHMPy)[1] # Qty Maestría (5009)
P[6,] <- dim(QTHDPy)[1] # Qty Doctorado (5010)
P[7,] <- sum(PyC$Valor_Proyecto)
TPProceso <- rbind(TPProceso, P)
}
TPProceso[c(1:10),] # 10 filas, todas las columnas
```

Fech	Codigo_Centr	Q_Proyecto	QNA_500	QNA_500	QNA_501	TV_Proyecto
a	o	s	7	9	0	s
2010	9101	4	7	7	0	732675272
2010	9127	2	4	3	0	409080660
2010	9201	1	0	4	0	484823440

1. Base de datos Procesos

A continuación se evidencian las primeras 10 filas de la tabla de procesos

Fecha	Codigo_Centro	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
2010	9101	4	7	7	0	732675272
2010	9127	2	4	3	0	409080660
2010	9201	1	0	4	0	484823440
2010	9202	1	2	3	0	383844564
2010	9203	5	7	10	3	664259270
2010	9204	2	4	2	2	309492055
2010	9205	1	2	0	1	400557314
2010	9206	4	4	8	1	696281600

Fecha	Codigo_Centro	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
2010	9301	2	1	4	0	371920762
2010	9401	1	0	2	1	445157258

```

#View(TProceso)
#write_xlsx(TProceso,"TProceso.xlsx") #Exporta a excel la tabla
# -----

```

1. Base de datos Procesos etiquetada

A continuación se evidencian las primeras 10 filas de la tabla de procesos

Generación de la tabla de procesos con etiqueta

Después de ser generada la tabla de procesos, se procede a generar una columna denominada Centro_Año La cual es una etiqueta que diferencia los centros en cada uno de los 13 años de análisis.

```
# -----
TPProceso <- read_xlsx("TPProceso.xlsx")
TPProceso <- as.data.frame(matrix(data = NA, ncol = 6, nrow = 200))
names(TPProceso) <- c("Centro_Año", names(TPProceso[c(-1, -2)]))
TPProceso$Centro_Año <-
c(paste(TPProceso$Codigo_Centro, TPProceso$Fecha, sep="-"))
TPProceso[, c(2:6)] <- TPProceso[, c(3:7)]
TPProceso[c(1:10), ] # 10 filas, todas las columnas
```

Centro_Año	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101-2010	1	3	1	0	401095127
9127-2010	2	2	4	1	395482518
9201-2010	1	0	2	0	506948025
9202-2010	3	3	5	2	455123589
9203-2010	2	3	2	2	461048052
9204-2010	3	6	6	0	494028420
9205-2010	1	3	0	1	415666541
9206-2010	1	2	2	1	332249046
9301-2010	3	7	8	0	498824485
9401-2010	3	8	3	2	591271866

```
#View(TPProceso) #Visualización de tabla TPProceso en R
#write_xlsx(TPProceso, "TPProceso.xlsx") #Exporta a excel la tabla
# -----
```

2. Base de datos de productividad

A continuación se especifica parte del código en R statistics de como generar la tabla de productividad

Generación de la tabla de cantidad de productos de CTel proyectos y productos y centros en 13 años

La tabla "TPyPdPCY" almacena la información del total de proyectos y productos de CTel, por centro y por año, los campos que almacena la tabla son: Código_Centro, Cantidad_Proyectos, Fecha_inicio_Proyecto, Cantidad_Productos, Fecha_Registro, P_8020: cantidad de productos de Apropiación Social, P_8025: cantidad de productos de Artículo Sometido en revista Indexada, P_8019: cantidad de productos de Divulgación Pública de la CTel, Publicaciones editoriales no especializadas producciones de contenido digital, P_8039: cantidad de productos de Informe de investigación, P_8016: cantidad de Libros resultados de investigación, P_8031: cantidad de Productos tecnológicos certificados o validados.

Para ello primero debemos evidenciar como se creo la tabla de Cantidad de Proyectos y Cantidad de Productos por Centro y por Año. Para ello se requiere crear la tabla cantidad de proyectos y productos por centro y año: "CPyPdCA"

```
# =====
# 536: el peor año de la historia.
# Nineteen Eighty-Four 1984 (Novela) - George Orwell
set.seed(536) # semilla
# =====

# Base de datos: Datos.xlsx
# =====
data <- read_xlsx("Datos.xlsx")
Tipologia_Productos <- read_xlsx("Datos.xlsx", 1)
Semilleros <- read_xlsx("Datos.xlsx", 2)
Rubro <- read_xlsx("Datos.xlsx", 3)
Rol_Sennova <- read_xlsx("Datos.xlsx", 4)
Tipo_Contratacion <- read_xlsx("Datos.xlsx", 5)
Grupo_investigacion <- read_xlsx("Datos.xlsx", 6)
Area_conocimiento_OCDE <- read_xlsx("Datos.xlsx", 7)
Principal_Talento_Humano <- read_xlsx("Datos.xlsx", 8)
Principal_Proyectos <- read_xlsx("Datos.xlsx", 9)
Estado_Proyecto <- read_xlsx("Datos.xlsx", 10)
Nivel_Academico <- read_xlsx("Datos.xlsx", 11)
Municipio <- read_xlsx("Datos.xlsx", 12)
Centros_Formacion <- read_xlsx("Datos.xlsx", 13)
Departamento <- read_xlsx("Datos.xlsx", 14)
Regionales <- read_xlsx("Datos.xlsx", 15)
Principal_Productos <- read_xlsx("Datos.xlsx", 16)
Tipo_equipo <- read_xlsx("Datos.xlsx", 17)
Rol_Proyectos <- read_xlsx("Datos.xlsx", 18)
Estado_Contratacion <- read_xlsx("Datos.xlsx", 19)
Principal_Equipos <- read_xlsx("Datos.xlsx", 20)
```

Tabla

```
Tabla_productividad <- TPyPdPCY
Tabla_productividad[c(1:16),] # 16 filas, todas las columnas
```

Codigo_Centr	Cantidad_Proyectos	Fecha_inicio_Proyecto	Cantidad_Productos	Fecha_Registro	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
o	os	o	os	o	0	5	9	6	1	1
9101	1	2010	3	2010	1	1	0	0	0	1
9127	2	2010	1	2010	0	0	0	0	1	0
9201	1	2010	2	2010	0	0	0	0	2	0
9202	3	2010	5	2010	2	1	0	0	2	0
9203	2	2010	5	2010	1	1	0	1	1	1
9204	3	2010	4	2010	0	1	0	0	1	2
9205	1	2010	1	2010	0	0	1	0	0	0
9206	1	2010	3	2010	1	1	0	0	1	0
9301	3	2010	3	2010	0	0	0	2	1	0
9401	3	2010	7	2010	2	0	0	0	1	4
9402	2	2010	4	2010	0	0	0	1	3	0
9501	2	2010	2	2010	1	0	1	0	0	0
9502	3	2010	6	2010	2	1	0	0	1	2
9503	2	2010	2	2010	0	1	0	1	0	0
9504	1	2010	2	2010	1	0	1	0	0	0
9549	2	2010	3	2010	1	0	0	1	1	0

```
# Tabla: Total de proyectos y productos por centros y por año
# View(TPyPdPCY)
# write_xlsx(TPyPdPCY, "TPyPdPCY.xlsx")
# =====
```

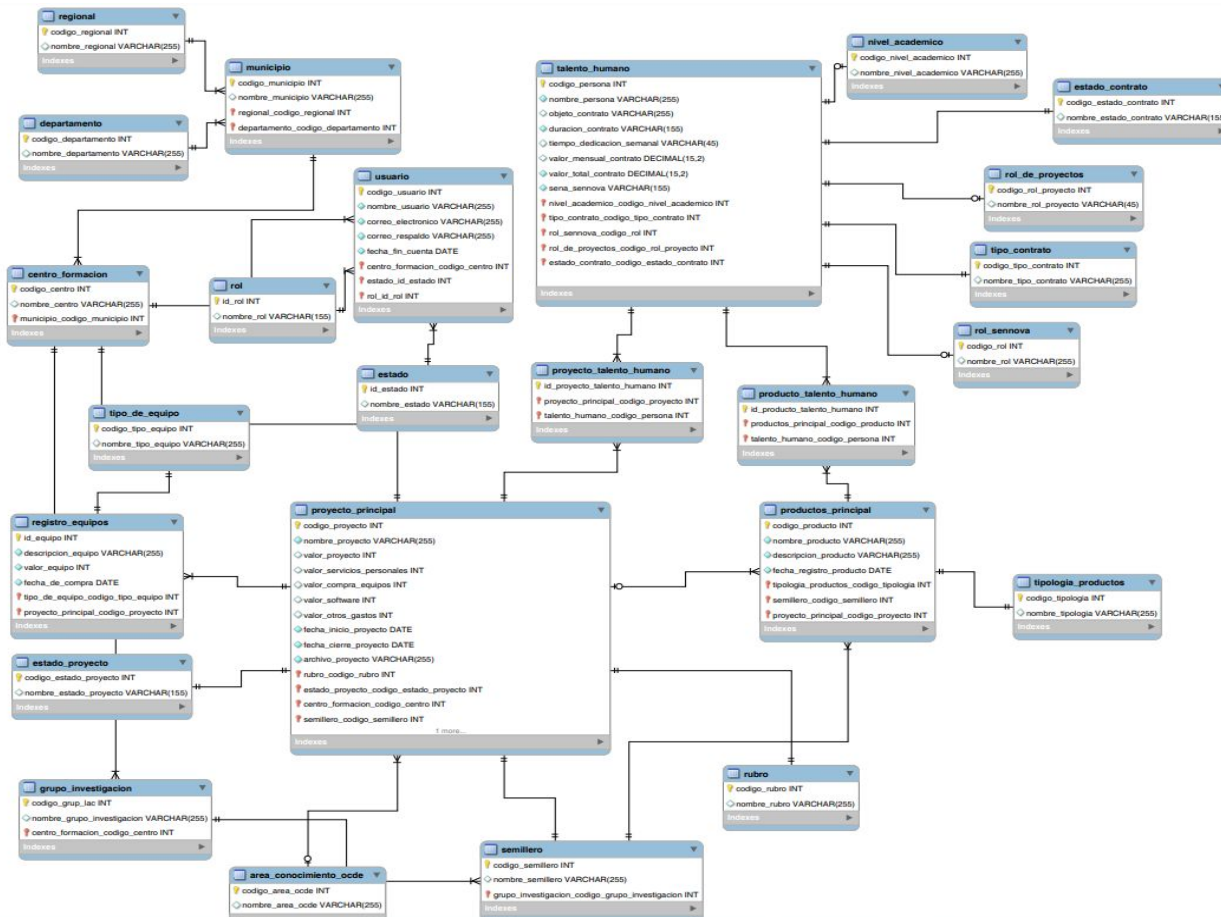
Indicadores y correspondencia de variables de entrada y salida del modelo



Nro	DIMENSIÓN (Proceso de entradas y salidas)	CATEGORIA	INDICADOR (Variables)	INDICADOR FINAL
1	Insumos para actividades de Ciencia, Tecnología e Innovación (ACTI) (Entradas en el modelo)	RECURSOS ECONÓMICOS	GASTO/INVERSIÓN en ACTI	Suma de los diferentes rubros ejecutados.
2		RECURSOS FÍSICOS Y TECNOLÓGICOS	NÚMERO DE LABORATORIOS PARA ACTI	AMBIENTES DE APRENDIZAJE (Laboratorios dotados con recursos de SENNOVA)
3			INVERSIÓN DE EQUIPOS DE PARA ACTI	Suma del rubro de equipos y maquinaria.
4		TALENTO HUMANO	NÚMERO DE LICENCIAS (Campo de conocimientos según la OCDE) (Por programas de educación)	Software de aplicaciones especializadas Software de programación
5			NÚMERO DE PERSONAS DEDICADAS A ACTI	
6			NÚMERO DE UNIDADES ORGANIZATIVAS (GRUPOS DE INVESTIGACIÓN - SEMILLEROS)	
7			NÚMERO DE PROGRAMAS IMPACTADOS DE EDUCACIÓN SUPERIOR DERIVADOS (PROGRAMAS DEL SENA POR ACTI)	
8		RECURSOS ESTRUCTURALES Y RELACIONALES	NÚMERO DE CONVENIOS DE COLABORACIÓN PARA ACTI	

9	Proceso (Planta del modelo)	PROYECTOS	NÚMERO DE PROYECTOS
10		TALENTO HUMANO	PROPORCIÓN DE GASTO EN ACTI POR PERSONAS DEDICADAS A ACTI
11		RECURSOS ECONÓMICOS	PORCENTAJE DE GASTO EN ACTI en relación con el PRESUPUESTO TOTAL
12			Porcentaje de gasto en INVESTIGACIÓN APLICADA en relación con el gasto total por ACTI
13		RELACIONES CON EL ENTORNO	EVENTOS DE MOVILIDAD PARA ACTI
14	Resultado (Salidas del modelo)	FORMACIÓN CIENTÍFICA Y TECNOLÓGICA	Formación de aprendices en semilleros de investigación
15		RELACIONES CON EL ENTORNO	Convenios con otras instituciones
16		PRODUCTOS RESULTADOS DE LAS ACTIVIDADES DE CTI	NÚMERO DE PRODUCTOS DE I+D (Productos de I+D por tipología Colciencias)
17		TRANSFERENCIA DE CONOCIMIENTO	NÚMERO DE EVENTOS DE CTI ORGANIZADOS POR LA UNIVERSIDAD (Número de eventos de CTI por clasificación ACTI)
19		FORMACIÓN CIENTÍFICA Y TECNOLÓGICA	Número de aprendices con formación científica
20		TALENTO HUMANO	PROPORCIÓN DE PRODUCTOS DE CTI POR PERSONA DEDICADAS A ACTI, EN RELACIÓN CON EL TOTAL DE PERSONAL
21			PROPORCIÓN DE PRODUCTOS DE CTI POR UNIDAD ORGANIZATIVA
22		RELACIONES CON EL ENTORNO	PROPORCIÓN DE EVENTOS DE MOVILIDAD PARA ACTI POR PERSONAS DEDICADAS A ACTI

Modelo relacional





3. Modelo de analítica de datos para medición de gestión por centros



@SENAcomunica

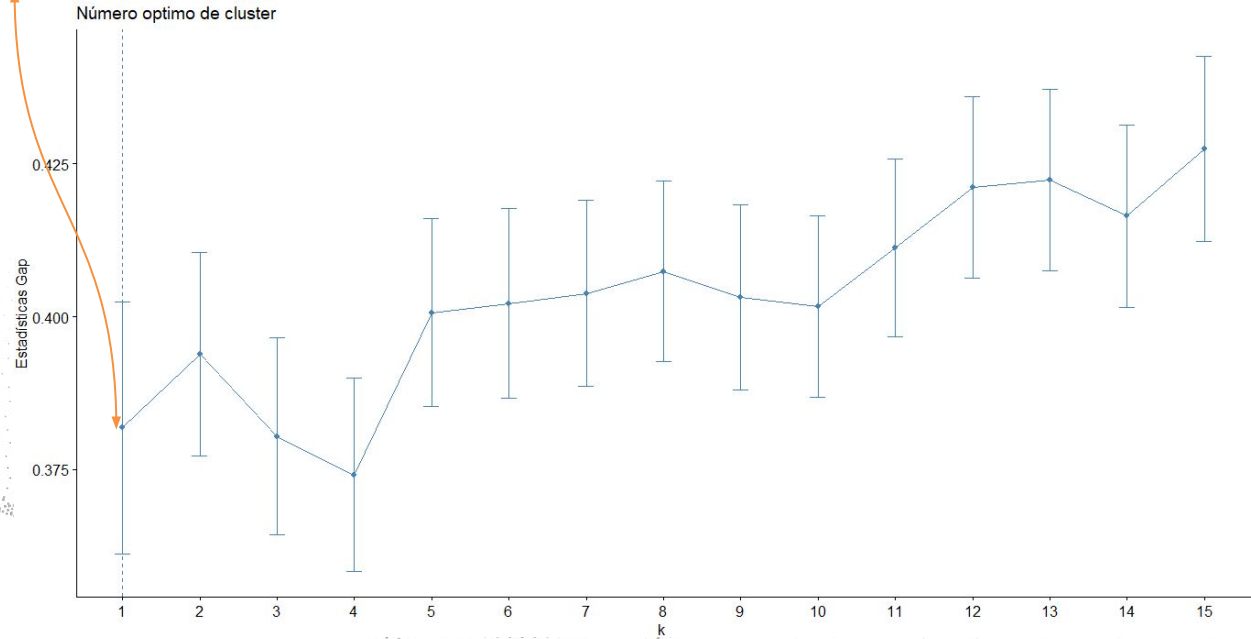
www.sena.edu.co

Clústering método k-means para medir nivel de gestión



Calculamos el k óptimo por Bootstrapping

```
#=====
#Calculo del óptimo método de Bootstrapping
#=====
fviz_nbclust(datos_tp, kmeans, method = "gap_stat", k.max = 15,
             diss = get_dist(datos_tp, method = "euclidean"), nstart = 100)+
  labs(title= "Número optimo de cluster") +
  xlab("k ") +
  ylab("Estadísticas Gap")
#=====
```

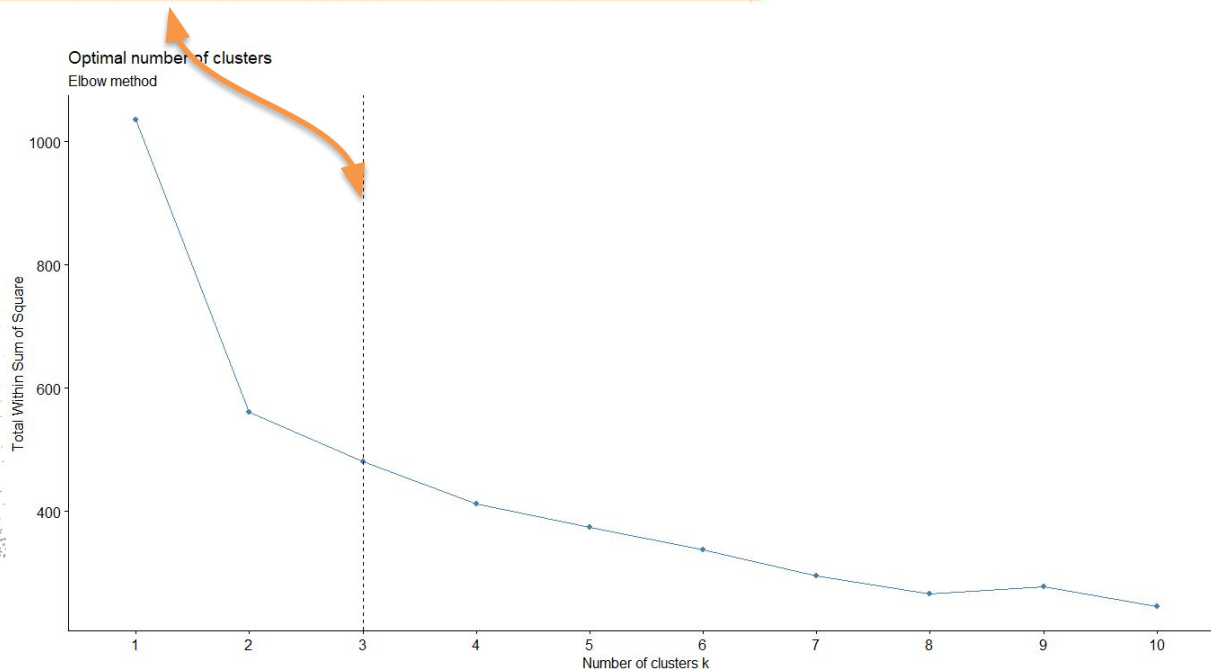


Clustering método k-means para medir nivel de gestión



Calculamos el k óptimo por Método de Elbow

```
#=====
#Método de Elbow
#=====
fviz_nbclust(datos_tp, kmeans, method = "wss") +
  geom_vline(xintercept = 3, linetype = 2) +
  labs(subtitle = "Elbow method")
#=====
```

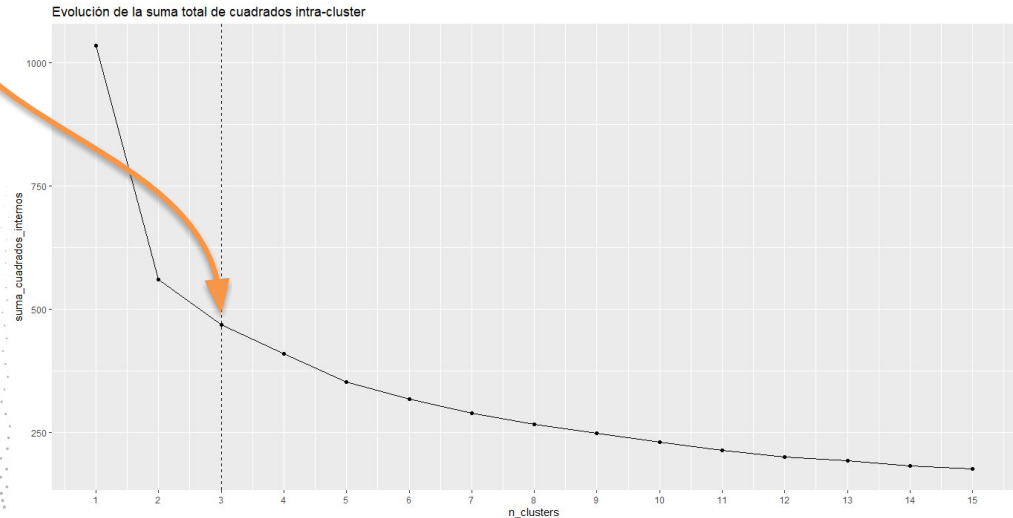


Clústering método k-means para medir nivel de gestión



Calculamos el número óptimo de cluster

```
#=====
data.frame(n_clusters = 1:15, suma_cuadrados_internos = total_withinss) %>%
  ggplot(aes(x = n_clusters, y = suma_cuadrados_internos)) +
  geom_line() +
  geom_point() +
  scale_x_continuous(breaks = 1:15) +
  labs(title = "Evolución de la suma total de cuadrados intra-cluster") +
  geom_vline(xintercept = 3, linetype = 2)
#=====
```

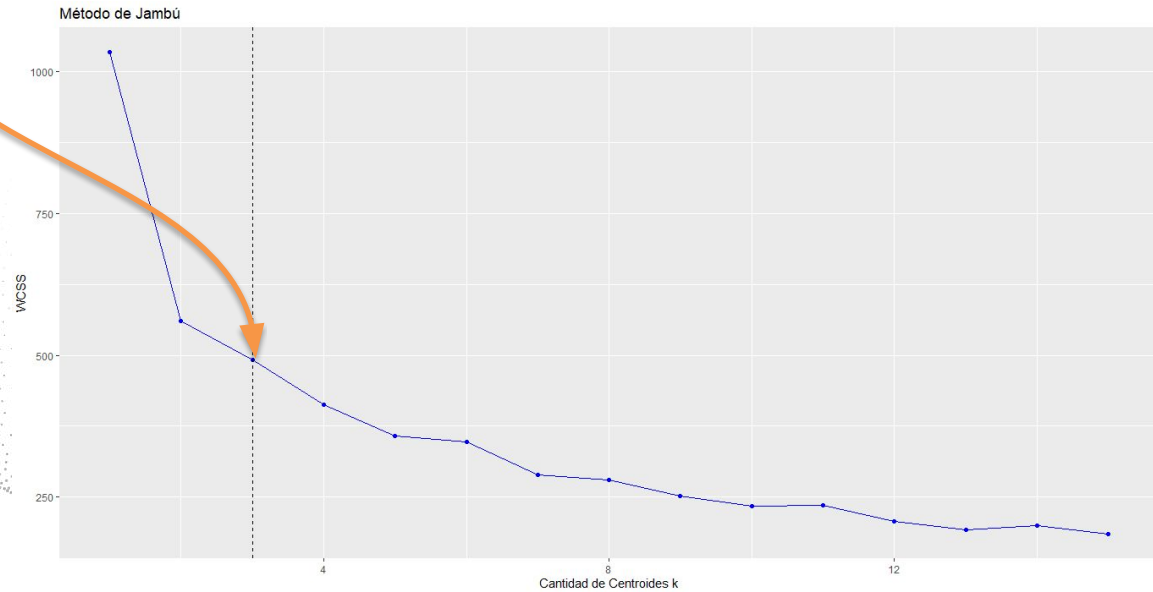


Clústering método k-means para medir nivel de gestión



Calculamos la cantidad de centroides por método de Jambú

```
#####  
set.seed(1984)  
wcscs <- vector()  
for(i in 1:15){  
  wcscs[i] <- sum(kmeans(datos_tp, i)$withinss)  
}  
  
ggplot() + geom_point(aes(x = 1:15, y = wcscs), color = 'blue') +  
  geom_line(aes(x = 1:15, y = wcscs), color = 'blue') +  
  ggtitle("Método de Jambú") +  
  geom_vline(xintercept = 3, linetype = 2) +  
  xlab('Cantidad de centroides k') +  
  ylab('wcscs')  
#####
```



Clústering método k-means para medir nivel de gestión



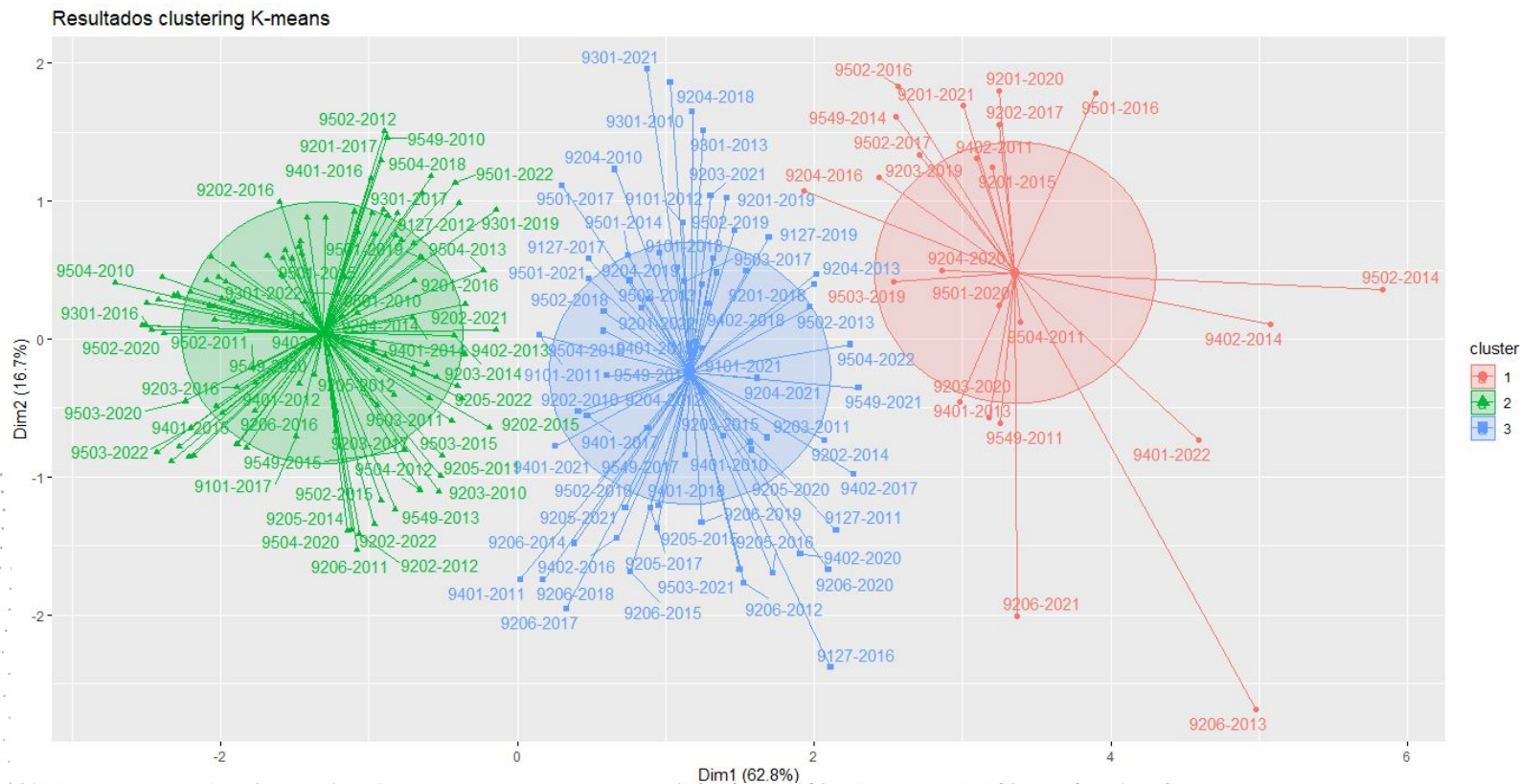
Aplicación del método K-means

```
#=====
set.seed(123) #Establecimiento de semilla
km_clusters <- kmeans(x = datos_tp, centers = 3, iter.max = 16 , nstart = 100)
#Tamaño de observaciones en cada cluster
km_clusters$size
# [1] 23 118 67
#=====
```

```
#=====
#Clustering mediante el método de k-means
#=====
ncluster <- c(1,2,3)
nColor <- c("red", "blue", "forestgreen")
fviz_cluster(object = km_clusters, data = datos_tp,
              show.clust.cent = TRUE,
              #ellipse.type = "convex", # Dibuja Poligono
              ellipse.type = "euclid", # Dibuja Elipse
              star.plot = TRUE,
              repel = TRUE, # Evita solapar etiquetas
              cex = 0.1) +
  labs(title = "Resultados clustering K-means")
#=====
```

Clústering método k-means para medir nivel de gestión

Clusters formados por el método K-means



Análisis de componentes principales



Mediante el análisis de componentes principales aplicado al método k-means es posible evidenciar la representatividad de las variables de interés que permiten medir el nivel de gestión por clúster, como se evidencia a continuación parte del código:

```
#=====
#Análisis de componentes principales para medir representatividad
#=====

#=====
set.seed(123)
pca_1 <- prcomp(datos_tp)
df.pca <- pca_1$x
view(df.pca)
kc <- kmeans(df.pca[,1:2], centers = 3, iter.max = 16 , nstart = 100)
kc$size
# [1] 118 68 22
view(kc$cluster)
#=====
```

	Centro_Año	PC1	PC2	Cluster
9101-2010	9101-2010	-2.03979861	0.13972792	1
9127-2010	9127-2010	-0.97175114	-0.03320817	1
9201-2010	9201-2010	-2.01358079	0.44805195	1
9202-2010	9202-2010	0.41178101	-0.52726177	2
9203-2010	9203-2010	-0.52687953	-1.10555031	1
9204-2010	9204-2010	0.65721124	1.23057457	2
9205-2010	9205-2010	-1.82643199	-0.79320951	1
9206-2010	9206-2010	-2.02877630	-0.48714911	1
9301-2010	9301-2010	1.18130269	1.64995790	2
9401-2010	9401-2010	1.57023473	-0.74914041	2
9402-2010	9402-2010	-0.69881613	-0.20823546	1
9501-2010	9501-2010	-0.69450397	0.42303928	1
9502-2010	9502-2010	0.87975476	-0.64635480	2

Showing 1 to 13 of 208 entries, 4 total columns

Aplicación de clustering mediante PCA



A continuación se evidencia la aplicación de clustering mediante el método de k-means y PCA

```
#=====
#Clustering mediante el método de k-means y pca
#=====

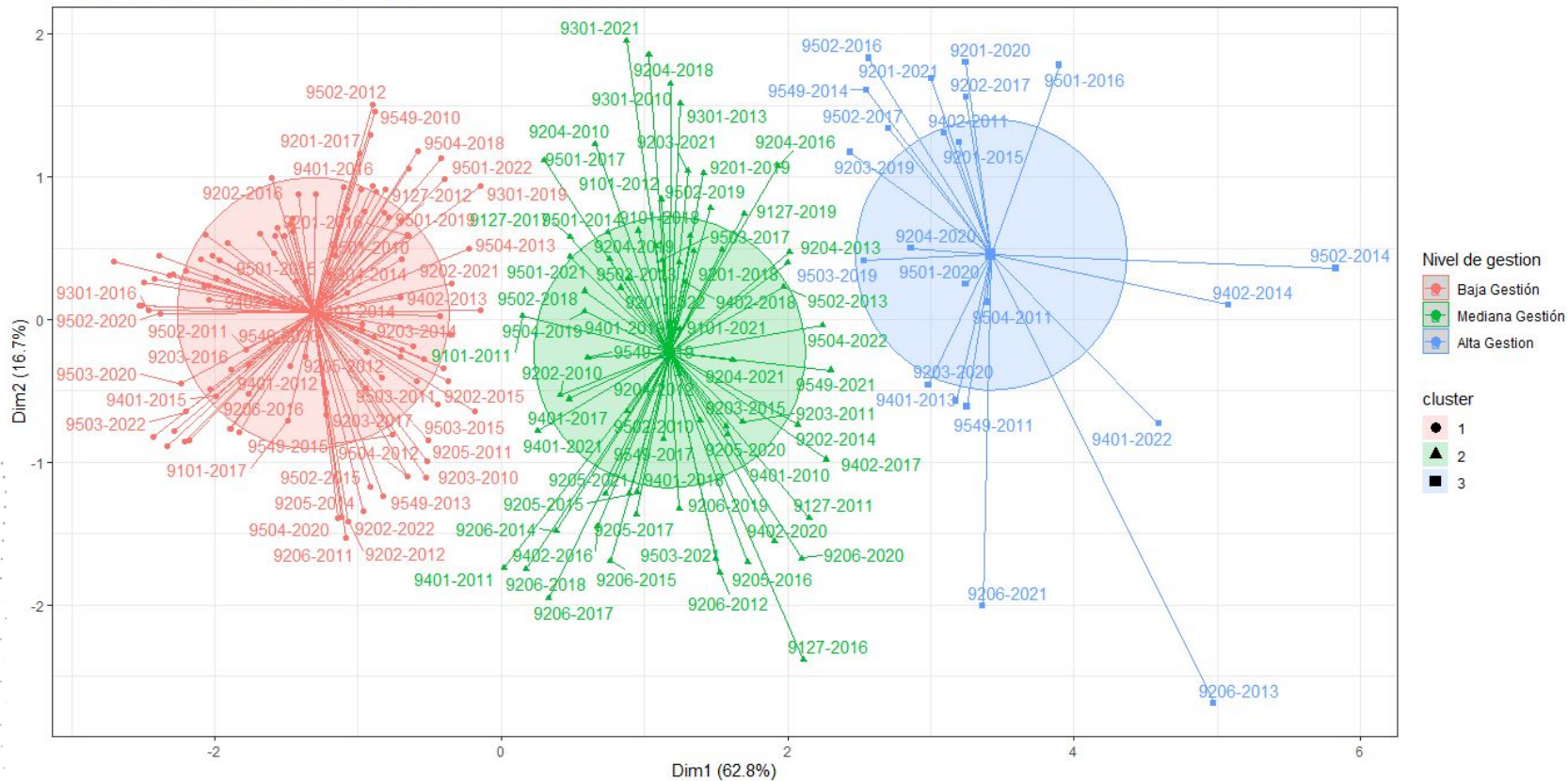
#=====
ncluster <- c(1,2,3)
ncolor <- c("red", "blue", "forestgreen")
fviz_cluster(object = kc, data = datos_tp,
              show.clust.cent = TRUE,
              #ellipse.type = "convex", # Dibuja Poligono
              ellipse.type = "euclid", # Dibuja Elipse
              star.plot = TRUE,
              repel = TRUE, # Evita solapar etiquetas
              cex = 0.1) +
  labs(title = "Resultados clustering k-means con pca") +
  scale_color_discrete("Nivel de gestion", labels = c("Baja Gestión", "Mediana Gestión", "Alta Gestion")) +
  theme_bw()
#=====
```

Aplicación de clustering mediante PCA



A continuación se evidencia la aplicación de clustering mediante el método de k-means y PCA

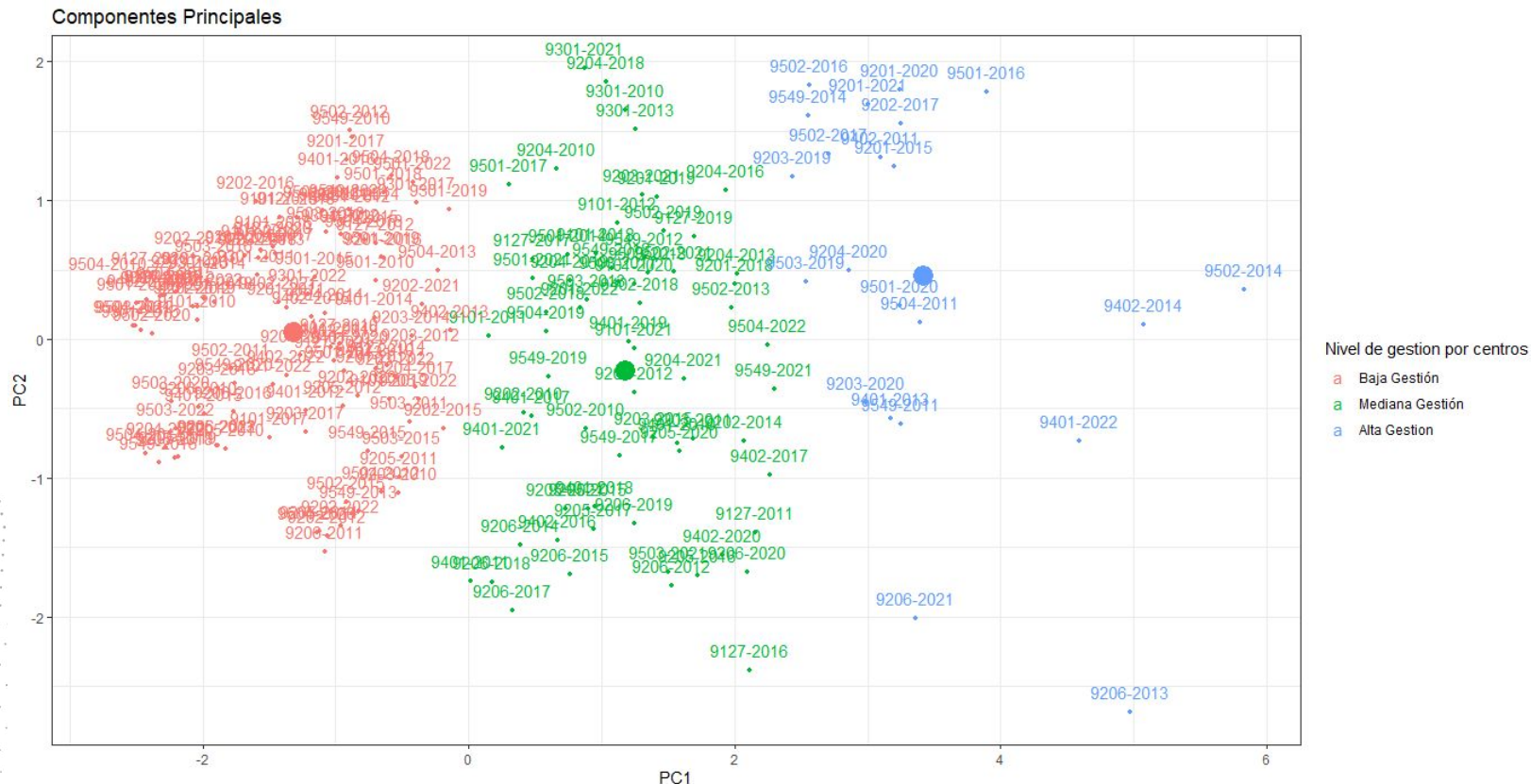
Resultados clustering K-means con pca



-  Baja Gestión
-  Mediana Gestión
-  Alta Gestión

● 1
 ▲ 2
 ■ 3

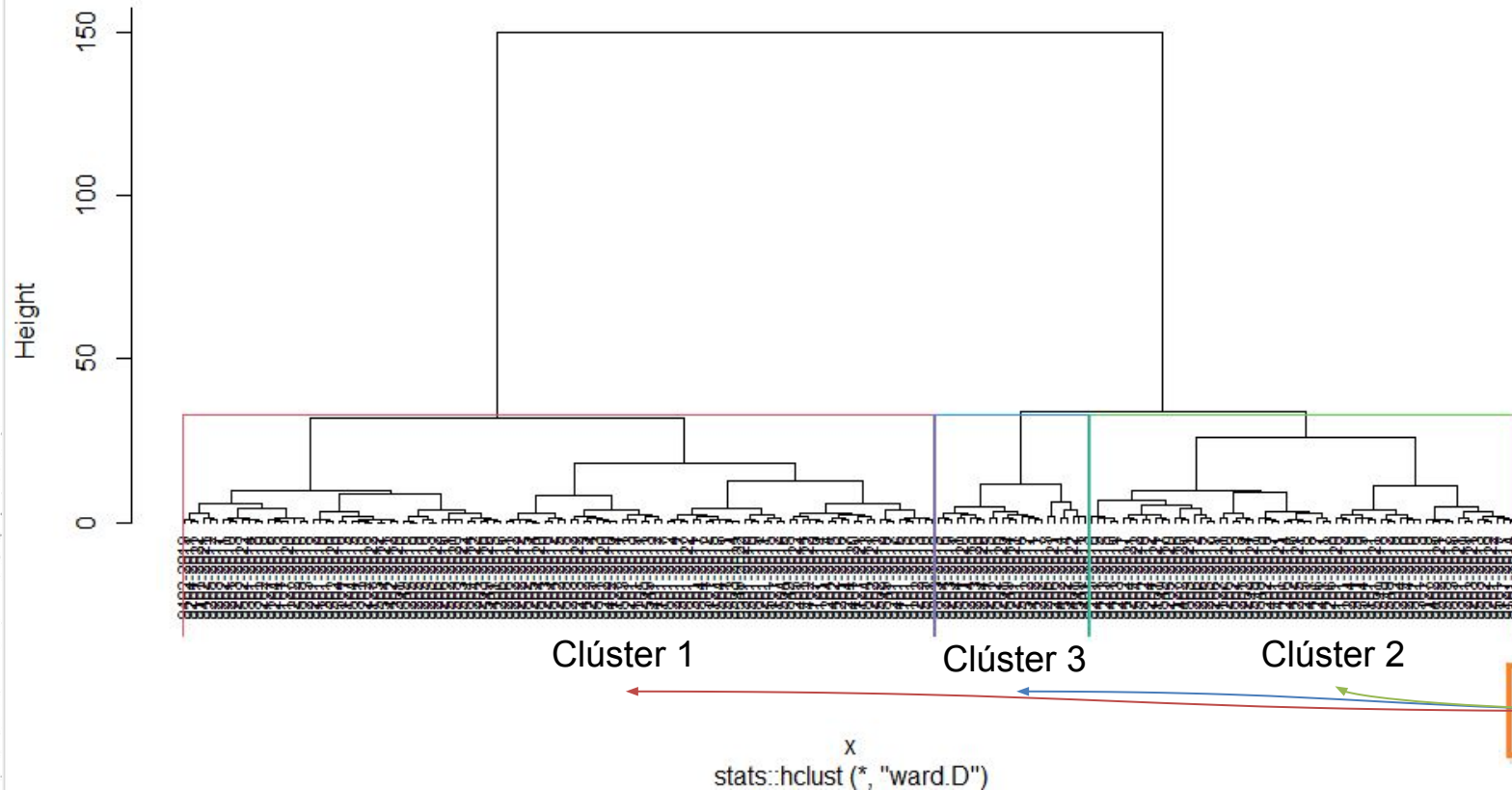
Gráfico de dos componentes principales para identificar comportamiento del nivel de gestión por centros



Implementación de clustering jerárquico



Dendograma para identificar centros con mayor gestión



```
> hc.cut$size  
[1] 117 67 24  
>
```

Visualización del clustering jerárquico

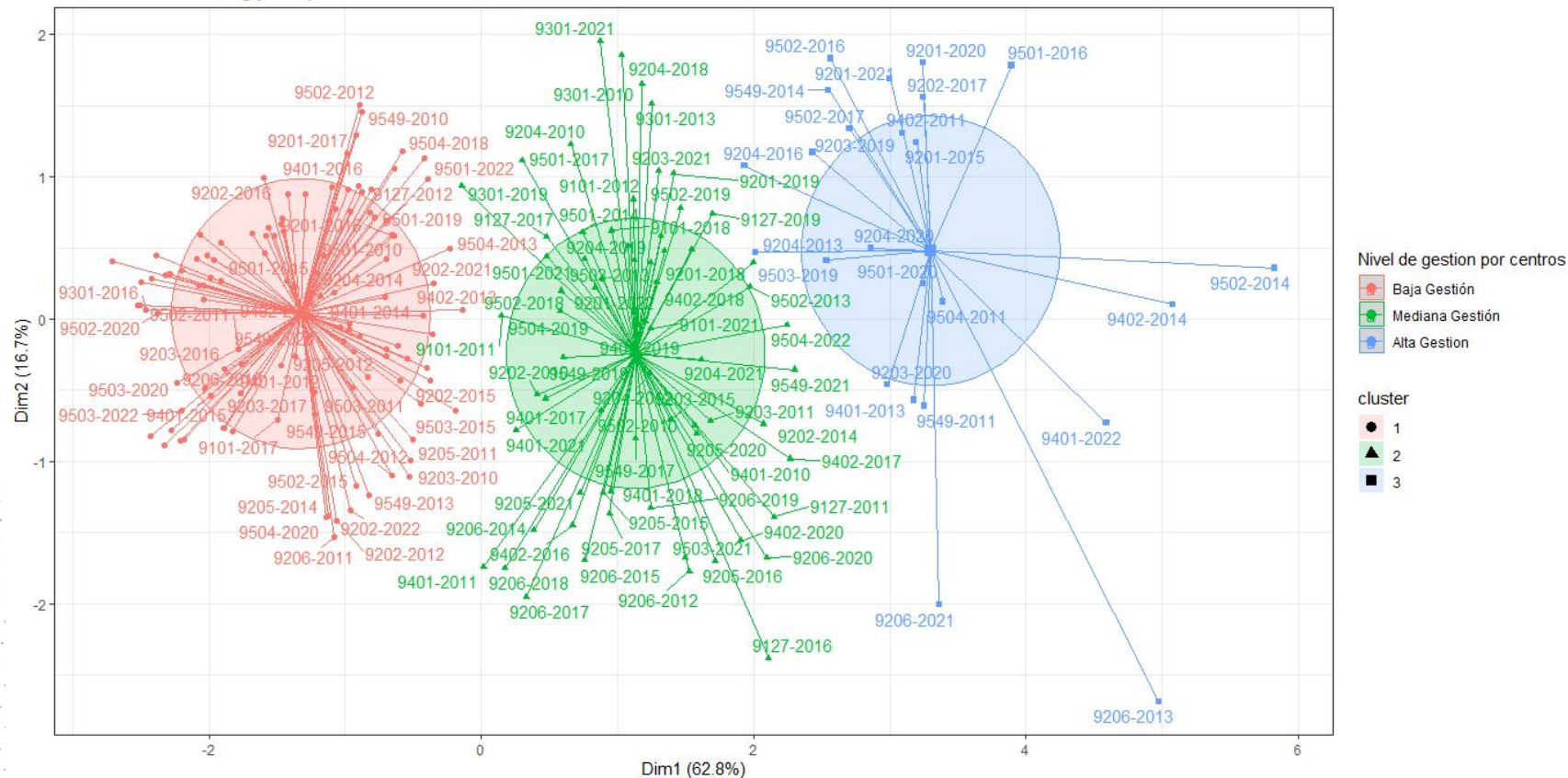


```
#=====
# visualización de los clústers después de implementar el dendograma
#=====
fviz_cluster(hc.cut, ellipse.type = "euclid", star.plot = TRUE,
  repel = TRUE, # Evita solapar etiquetas
  cex = 0.1) +
  labs(title = "Resultados clustering jerarquico") +
  scale_color_discrete("Nivel de gestion por centros",
    labels = c("Baja Gestión", "Mediana Gestión", "Alta Gestion")) +
  theme_bw()
#=====
```


Visualización del clustering jerárquico



Resultados clustering jerárquico





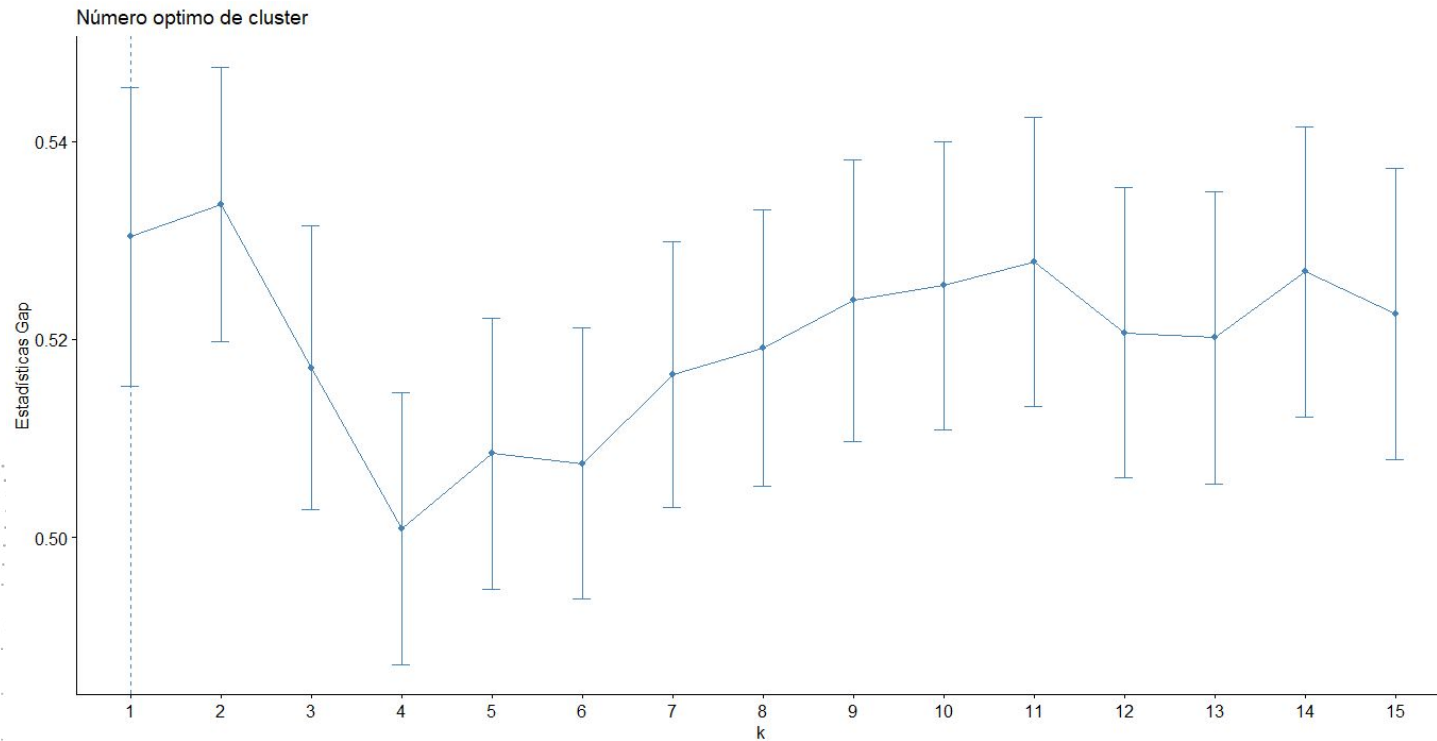
4. Modelo de analítica de datos para medición de productividad por centros



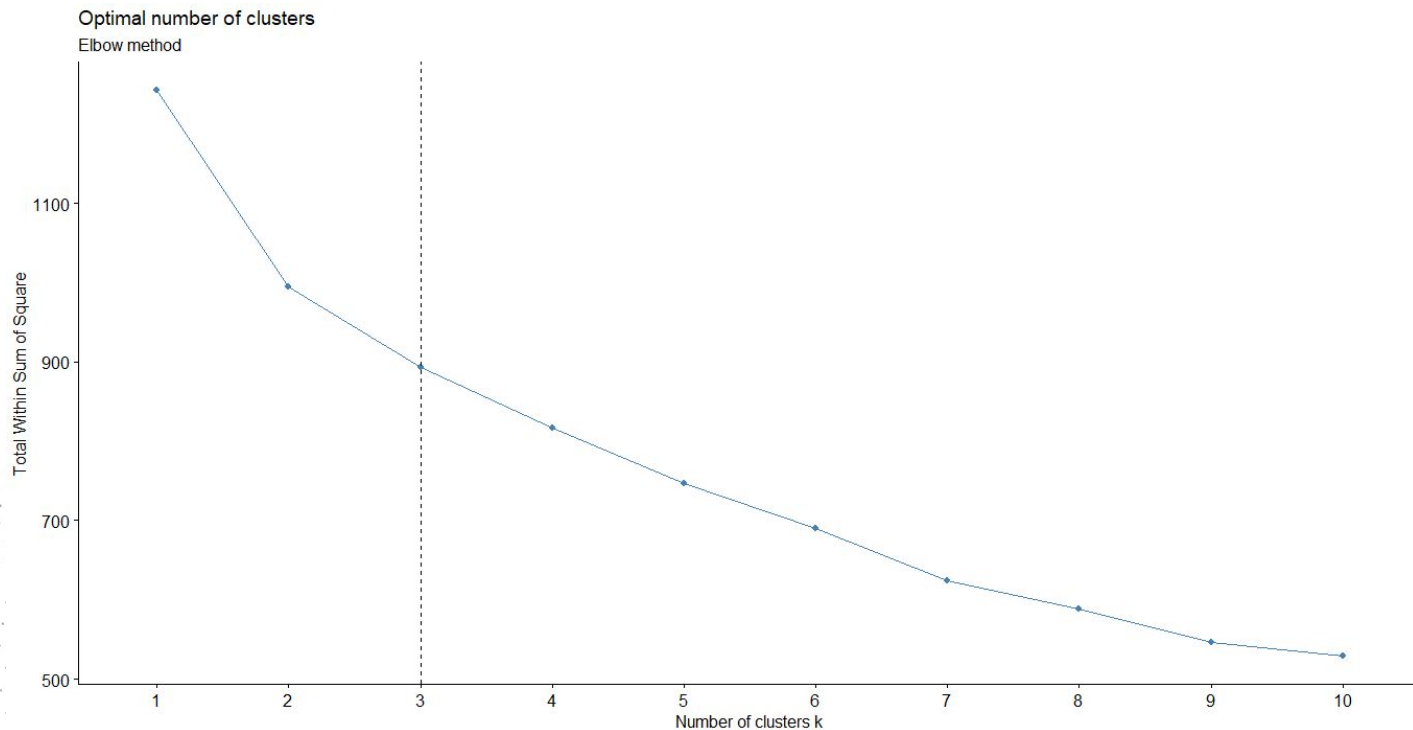
@SENAcomunica

www.sena.edu.co

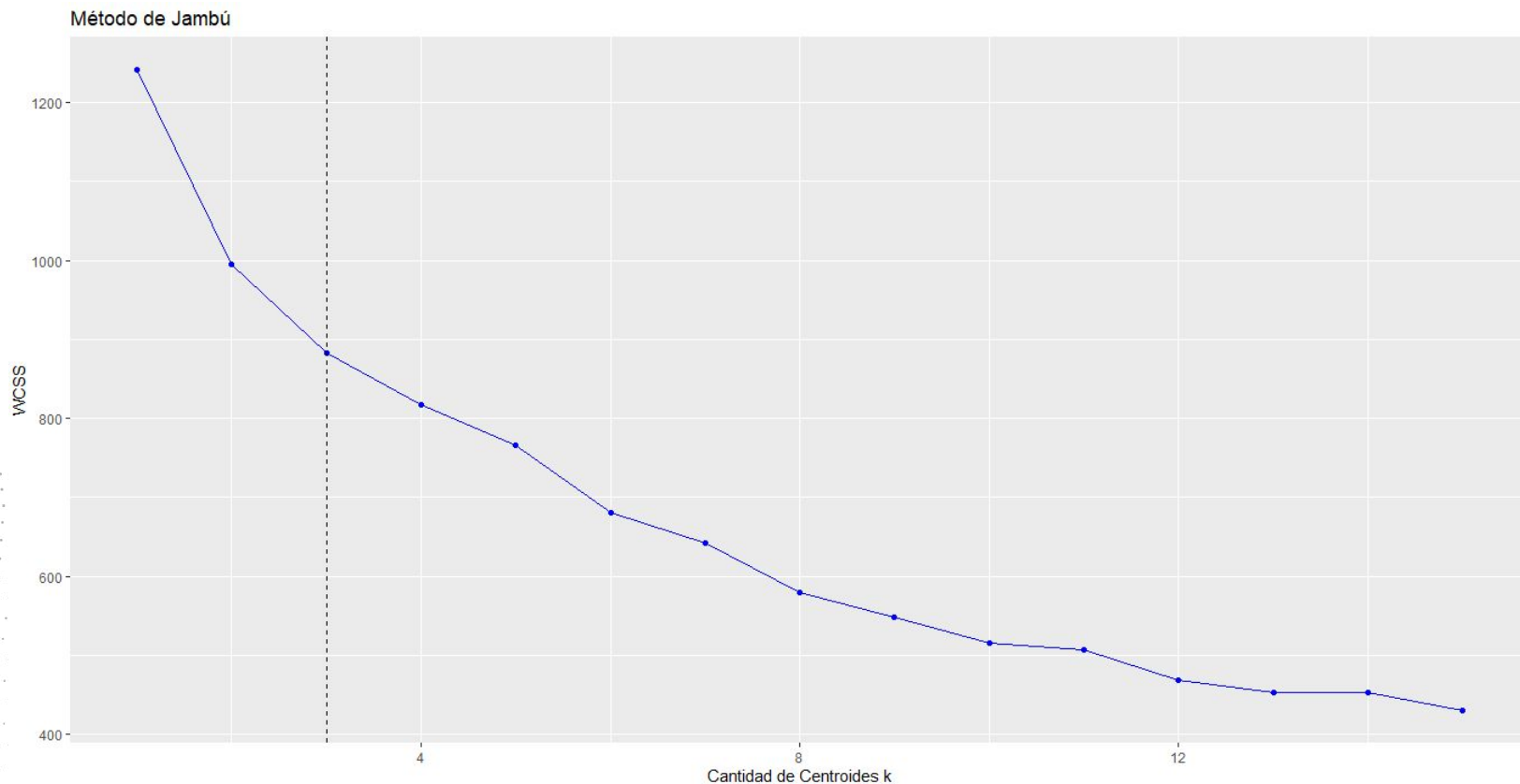
Cálculo del número óptimo de clúster por método de Bootstrapping



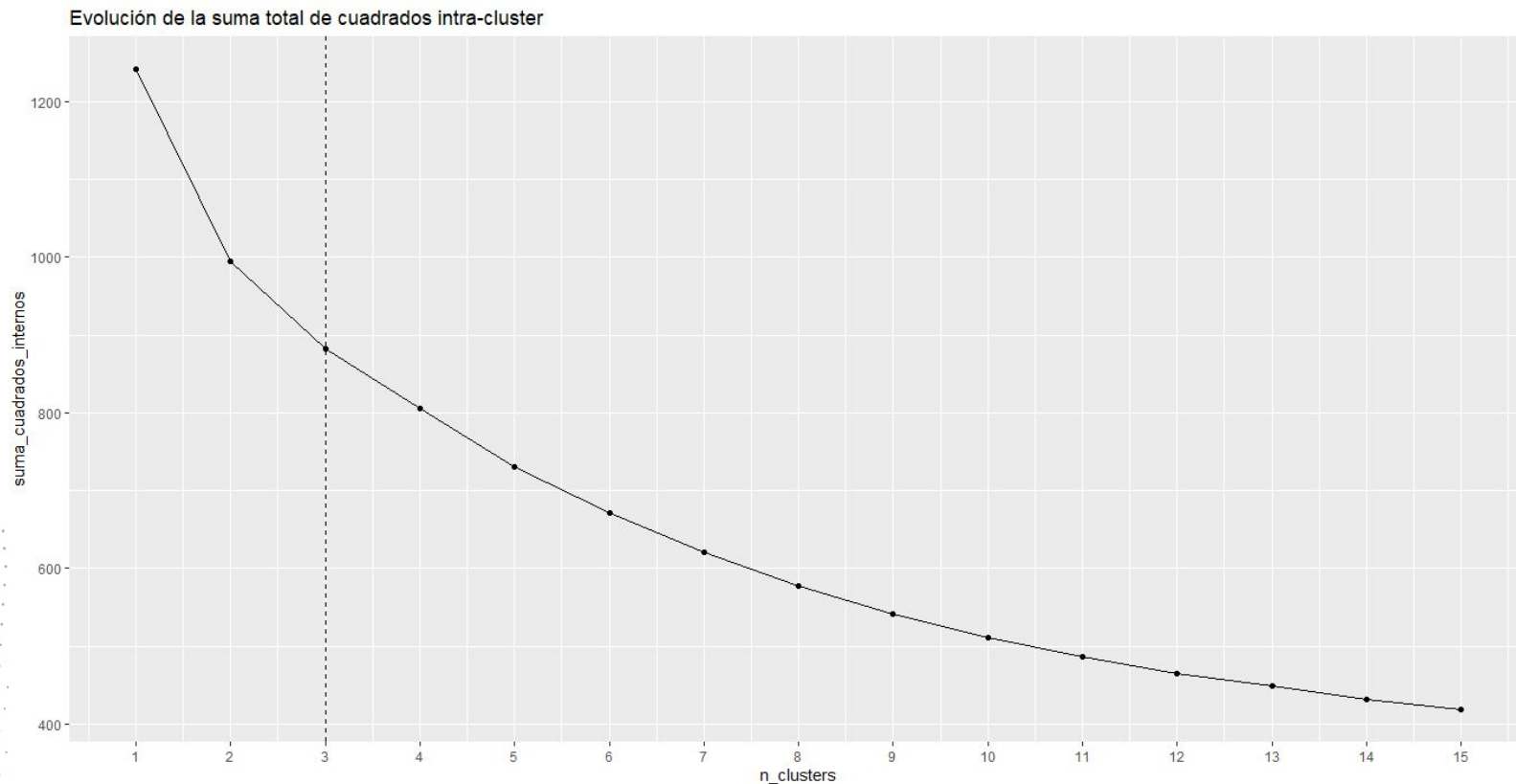
Cálculo del número óptimo de clúster por método de Elbow



Cálculo del número óptimo de clúster por método de Jambú



Estimación del número óptimo de clúster





Análisis de componentes principales para medir representatividad



```
#=====  
set.seed(123)  
pca_1 <- prcomp(datos_tppd)  
df.pca <- pca_1$x  
view(df.pca)  
kc <- kmeans(df.pca[,1:2], centers = 3, iter.max = 16 , nstart = 100)  
kc$size  
# [1] 75 42 91  
view(kc$cluster)  
#=====
```

Parte del código

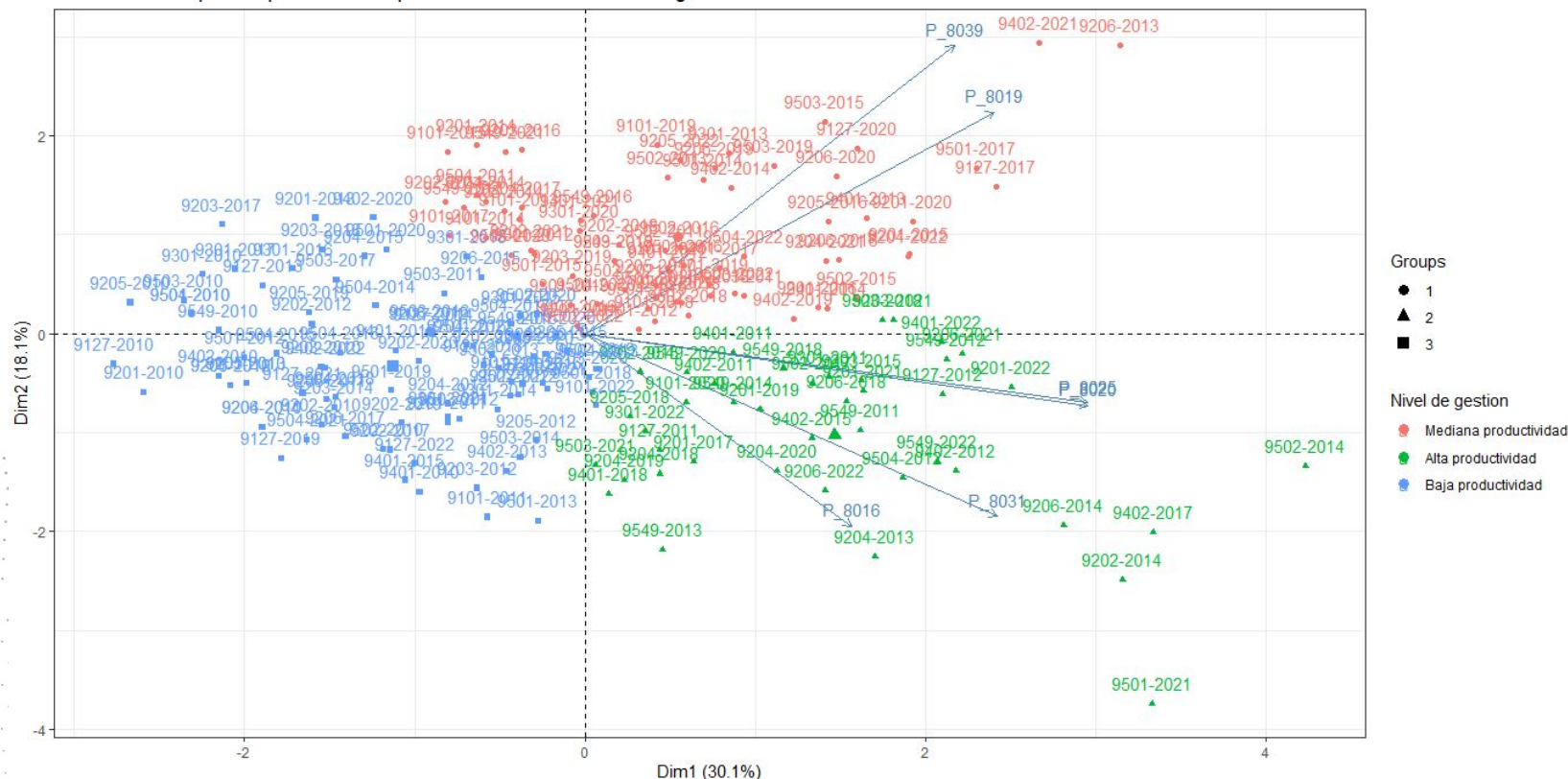
	Centro_Año	PC1	PC2	Cluster
9101-2010	9101-2010	-1.98705966	-0.500819403	3
9127-2010	9127-2010	-2.76717630	-0.306985411	3
9201-2010	9201-2010	-2.58878191	-0.593704928	3
9202-2010	9202-2010	-1.54537861	-0.919776807	3
9203-2010	9203-2010	-1.54795242	-0.335494793	3
9204-2010	9204-2010	-1.89355753	-0.942020551	3
9205-2010	9205-2010	-2.66833554	0.314118617	3
9206-2010	9206-2010	-2.08003491	-0.520232650	3
9301-2010	9301-2010	-2.24575061	0.597102844	3
9401-2010	9401-2010	-0.96917391	-1.601859772	3
9402-2010	9402-2010	-2.14967467	-0.428380318	3
9501-2010	9501-2010	-2.31207363	0.201293977	3
9502-2010	9502-2010	-1.18103371	-1.167669830	3

Showing 1 to 13 of 208 entries, 4 total columns

Plano factorial de pca para generar la gráfica de nivel de productividad por centros



Plano Factorial para la productividad por centros mediante clustering



Clustering mediante el método de k-means y pca



Resultados clustering K-means con pca

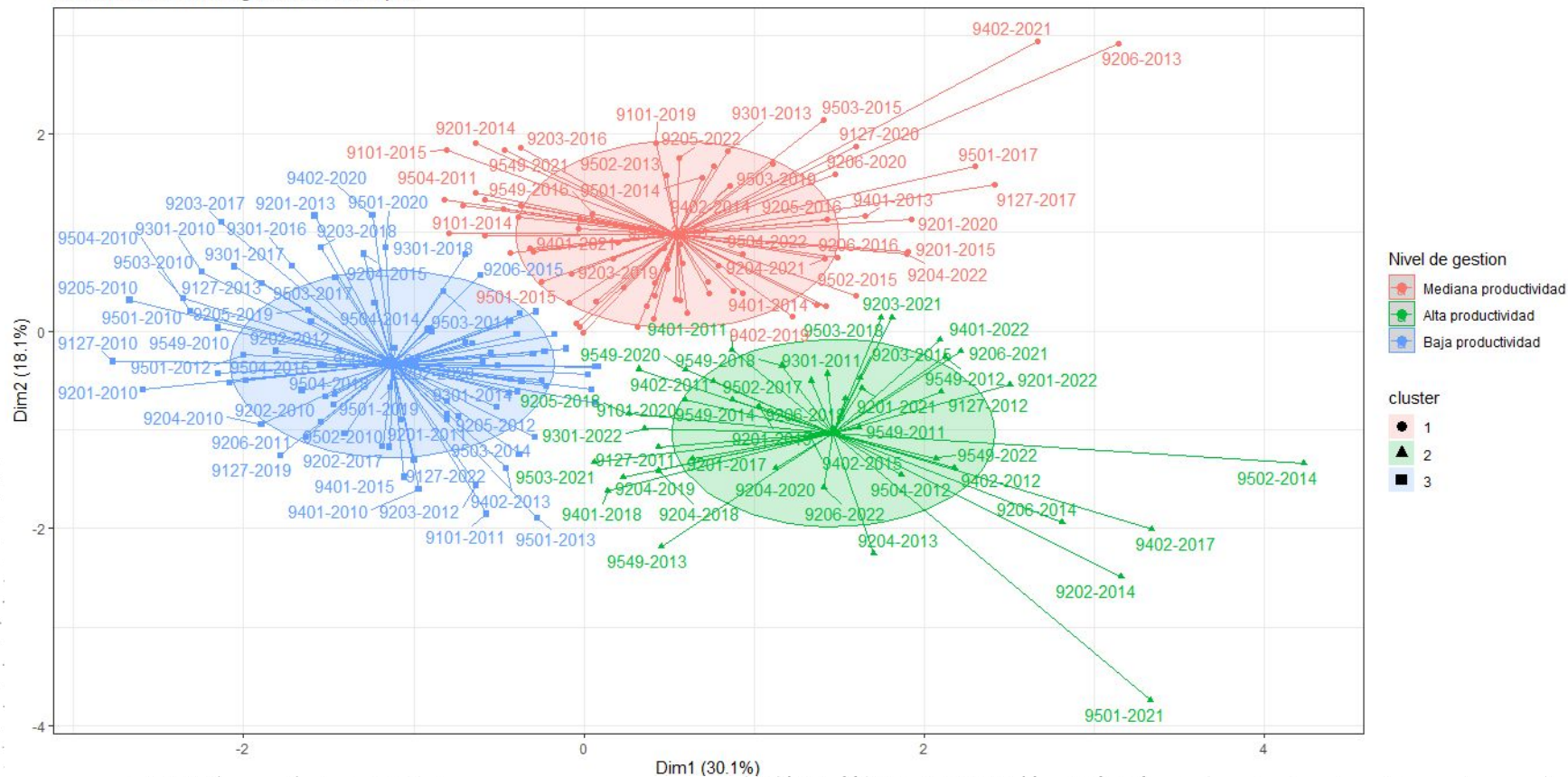
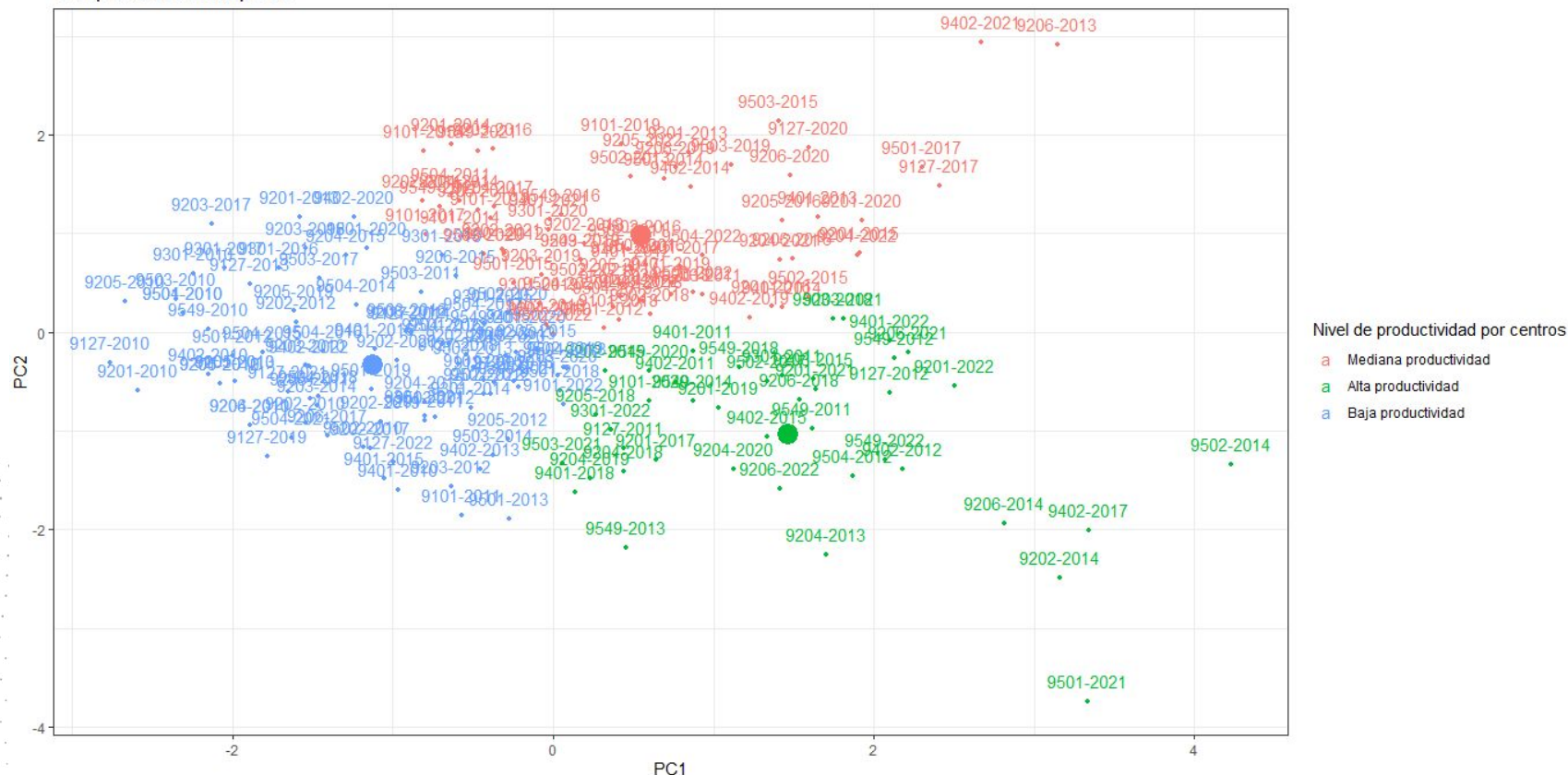


Gráfico de dos componentes principales para identificar comportamiento del nivel de productividad por centros



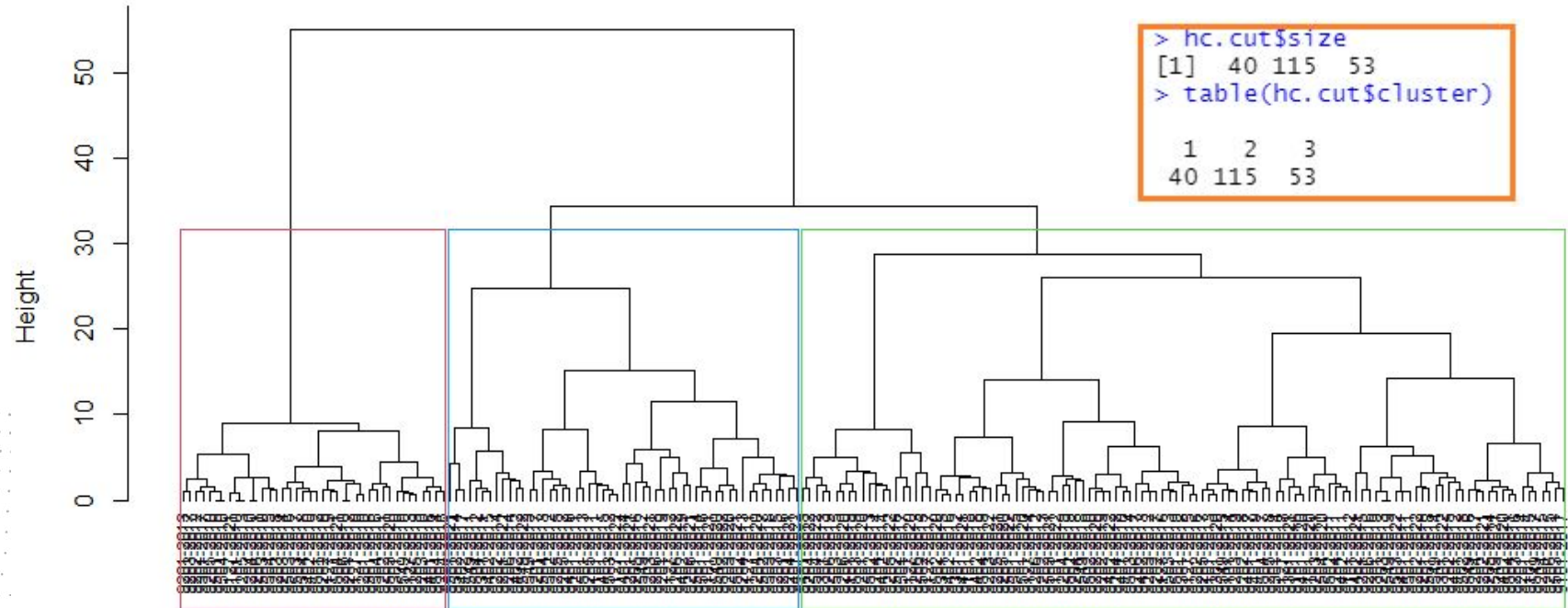
Componentes Principales



Implementación de clustering jerárquico



Dendrograma para identificar centros con mayor productividad





5. Actividades de apoyo SENNOVA



@SENAcomunica

www.sena.edu.co

Visita a la Universidad digital de Antioquia





6. Divulgación



@SENAcomunica

www.sena.edu.co

Actividades

- Proyecto Promoción de Ecosistemas de Innovación en América Latina - Componentes C3 OTC (Oficina de Transferencia de Conocimiento) y C4 (Gestión de Capital Documental).
- Cursos de transferencia de conocimientos dictados por la Universidad Politécnica de Valencia.
 - La función de transferencia de conocimiento en ecosistemas de innovación
 - Gestión del capital intelectual en SENNOVA.
- Elaboración de actas de los prototipos.

Pendiente

- Aportes para el artículo de visualización de la información
- Ajustar el plan de transferencia con el equipo de divulgadores



G R A C I A S

Línea de atención al ciudadano: 018000 910270
Línea de atención al empresario: 018000 910682



@SENAcomunica

www.sena.edu.co



El empleo
es de todos

Mintrabajo

Proyecto SGPS 9748

Prototipo Innovación

Presentación de avances 13

Modelo de evaluación y valoración de la Gestión de CTel para los Centros SENA

Regional Antioquia aplicando analítica de datos

14 de junio del 2022



@SENAcomunica

www.sena.edu.co



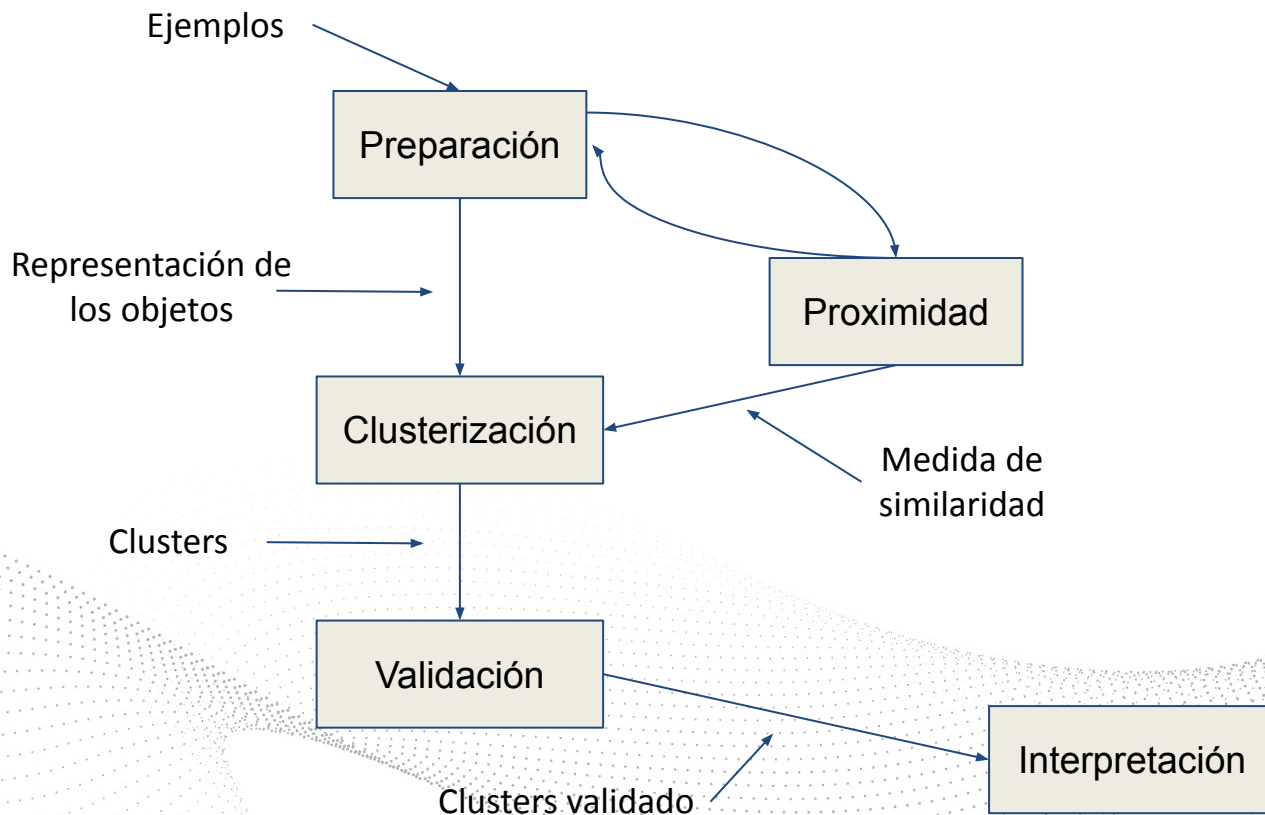
Etapas Clusterización - Clasificación



@SENAcomunica

www.sena.edu.co

Etapas del proceso de clusterización



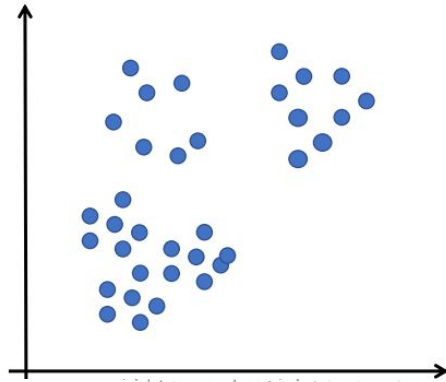
Etapas del proceso de clusterización

- **Preparación:** normalizaciones, conversiones de tipos, reducción del número de atributos
- **Proximidad:** definición (elección) de una medida de similitud apropiada al dominio de la aplicación, por ejemplo. distancia.
- **Clustering:** aplicación de algoritmos de clustering
- **Validación:** evaluación del resultado de la agrupación. Debe determinar si los conglomerados son significativos.
- **Interpretación:** etiquetado del clúster, definiendo su naturaleza, a través del análisis de sus objetos.

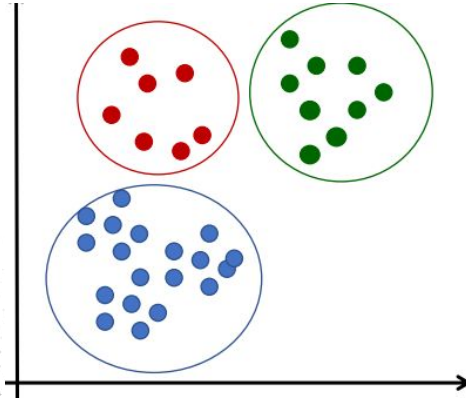
Agrupación de datos: tipos de algoritmos

Agrupación basado en particiones:

Conjunto original



Conjunto después de agrupación

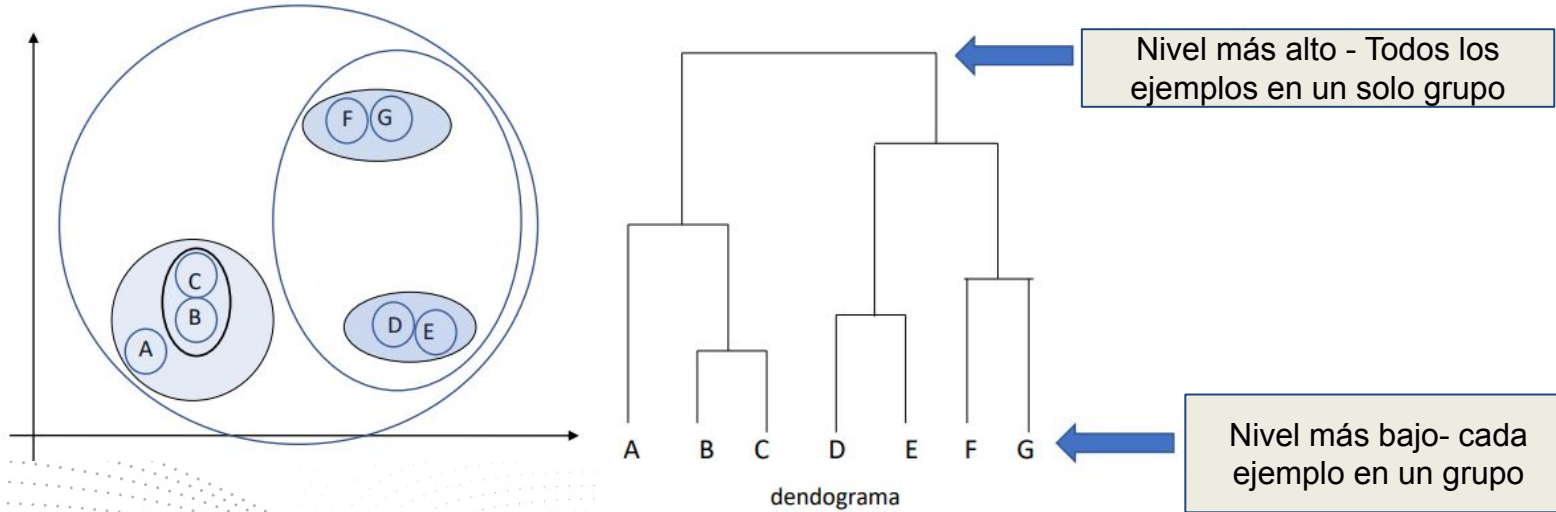


- **Ventaja:** más eficiente para grandes conjuntos de datos.
- **Inconveniente:** es necesario definir previamente el número de grupos deseado.

- Kmeans

Agrupación de datos: tipos de algoritmos

Agrupación jerárquica:



Ventaja: no es necesario establecer el número de grupos

Desventaja: Es más caro que el de particiones

Validación



Índices de validación externa: mide la calidad del agrupamiento conociendo información externa de antemano, es principalmente usada para escoger un algoritmo de clustering óptimo sobre un dataset específico.

- Homogeneidad
- Completitud
- Medida V
- Índice Rand corregido
- Adjusted Mutual Information
- Índice Fowlkes-Mallows

Valores entre 0 -1

0 : Aleatoriedad

1: Clusterización óptima

Índices de validación interno: miden el agrupamiento basado únicamente en la información de los datos. Evalúan que tanto la estructura del agrupamiento como la necesidad de información son buenas para el algoritmo y su resultado.

- Coeficiente de Silhueta
- Calinski_harabasz
- Davies Bouldin

Valores entre 0 -1

0 : Aleatoriedad

1: Clusterización óptima



Clusterización - Procesos



@SENAcomunica

www.sena.edu.co

Clusterización - Procesos 2010-2022

1. Kmeans - Validación - PCA
2. Agrupamiento jerárquico - Validación -PCA

Preprocesamiento

	Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	Total_Valor_Proyectos
0	9101-2010	-1.260308	-1.103402	-0.896032	-1.304164	-0.126344
1	9127-2010	-0.362554	-0.671184	-0.117850	-0.201456	-0.735003
2	9201-2010	-1.260308	-0.238966	-0.506941	-1.304164	-0.206320
3	9202-2010	0.535199	1.922122	0.271241	-1.304164	-0.608931
4	9203-2010	-0.362554	1.057686	-0.117850	-0.201456	-1.577040
...	...	...	...	...	...	...
203	9501-2022	-0.362554	0.193251	-0.117850	0.901251	-0.872848
204	9502-2022	-1.260308	-0.238966	-0.896032	-1.304164	-0.687234
205	9503-2022	-1.260308	-1.535619	-1.674214	-0.201456	0.218826
206	9504-2022	1.432953	1.489904	1.049423	-0.201456	2.333091
207	9549-2022	-0.362554	0.625469	-0.896032	-0.201456	-1.254502

208 rows × 6 columns

Tiempo: 13 años

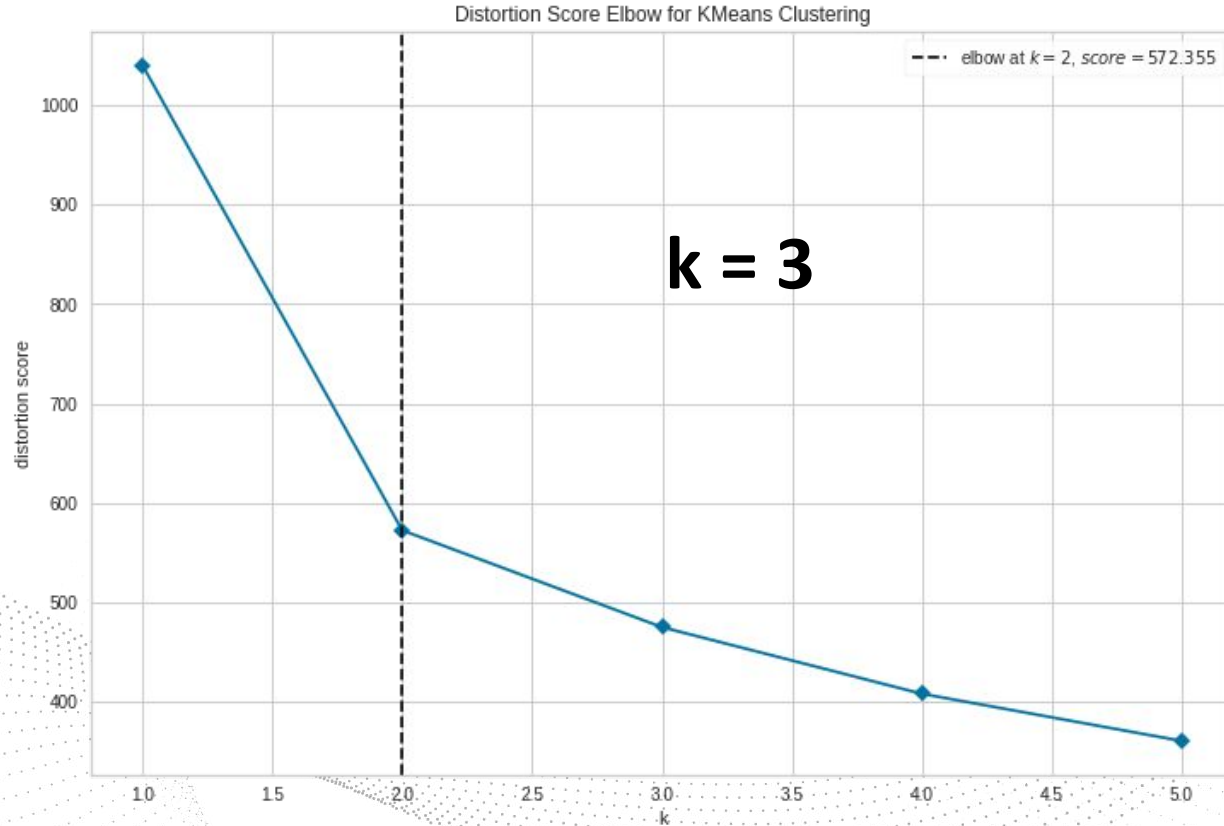
Centros de formación: 16

Columnas : 6

Filas: 208

KMeans - PCA

Método del codo : Determinación de centroides



Validación del agrupamiento por KMeans

Externos

Homogeneidad: 0.177

Compleitud: 1.000

Medida V: 0.300

Índice Rand corregido: 0.000

Adjusted Mutual Information: -0.000

Índice Fowlkes-Mallows: 0.000

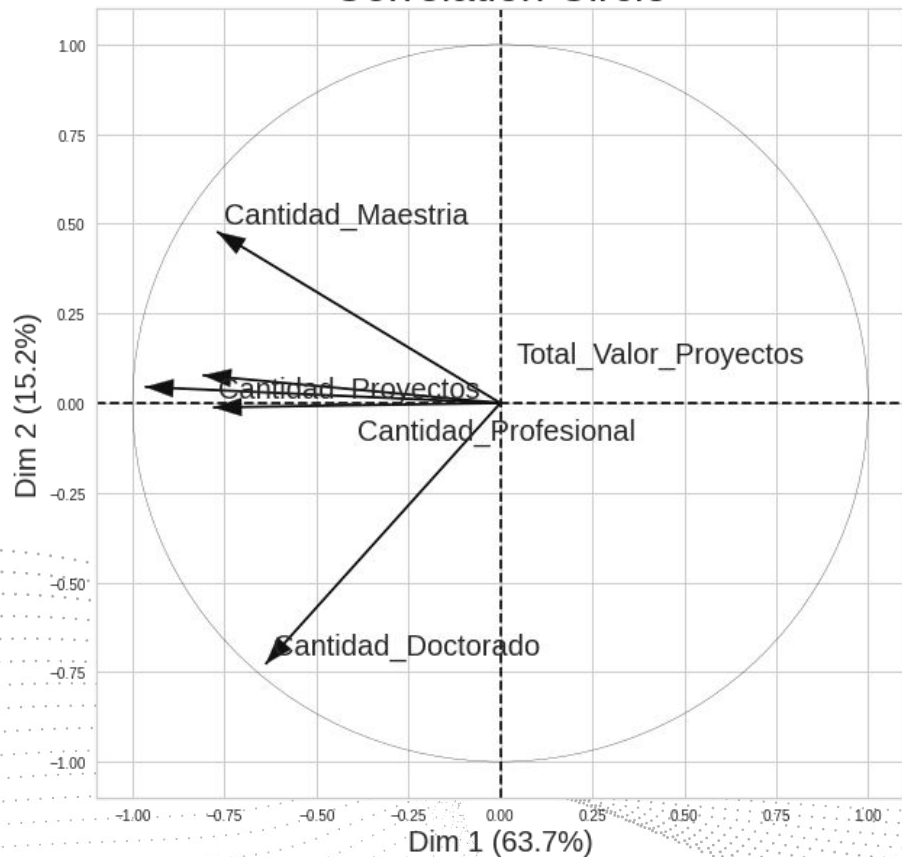
Internos

Coeficiente de Silhueta: 0.313

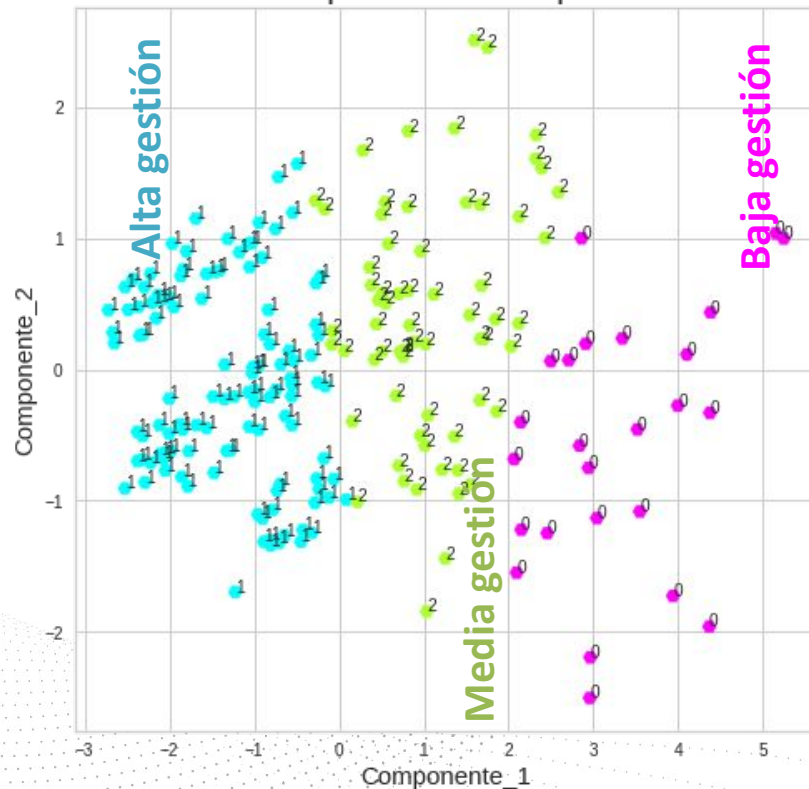
Calinski_harabasz: 122.043

Davies Bouldin: 1.291

Correlation Circle



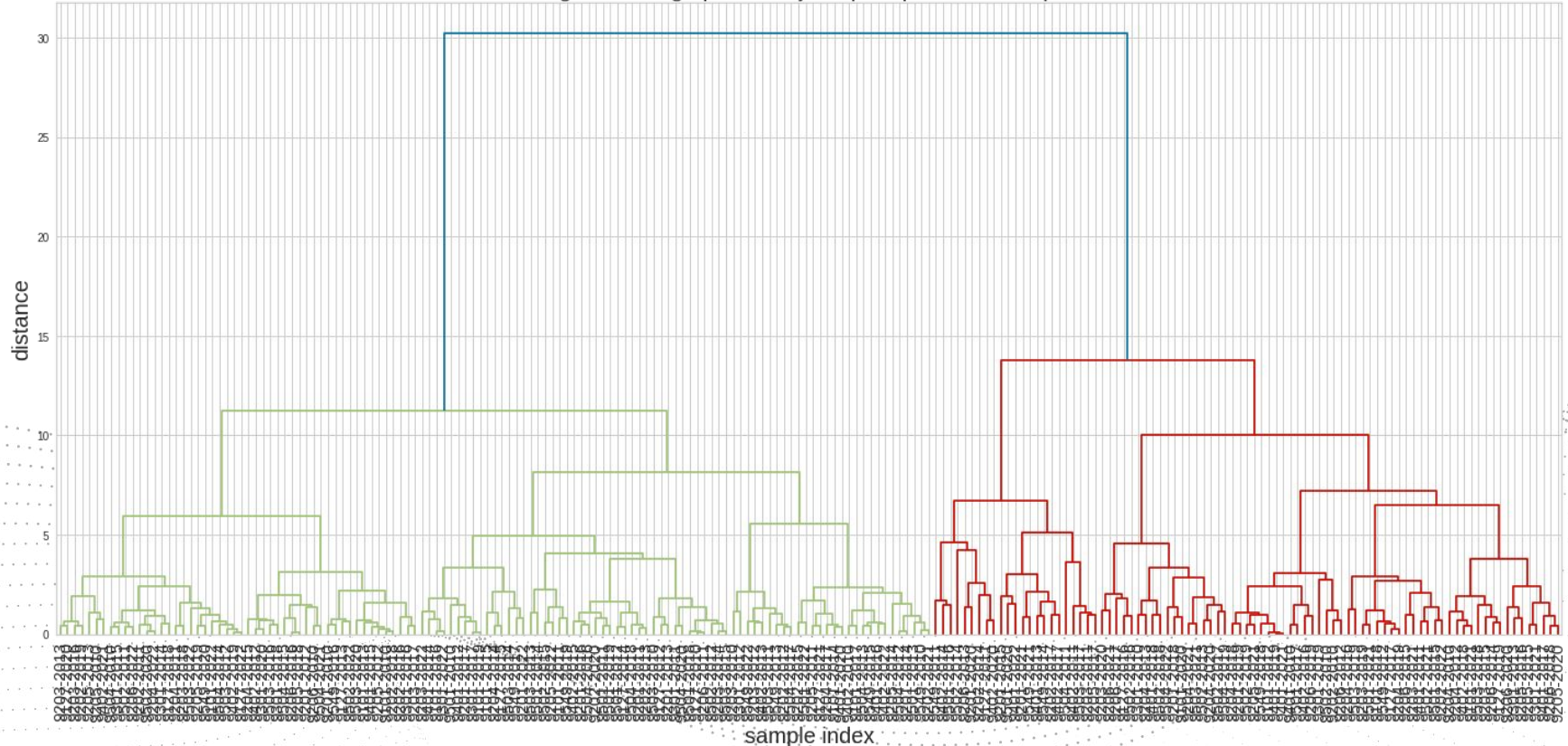
Componentes Principales



Jerárquico - PCA

Determinación de clusters - Dendrograma

Dendrograma de agrupamiento jerárquico para datos de procesos



Validación del agrupamiento por Jerárquico

Externos

Homogeneidad: 0.173

Complejidad: 1.000

Medida V: 0.294

Índice Rand corregido: 0.000

Adjusted Mutual Information: -0.000

Índice Fowlkes-Mallows: 0.000

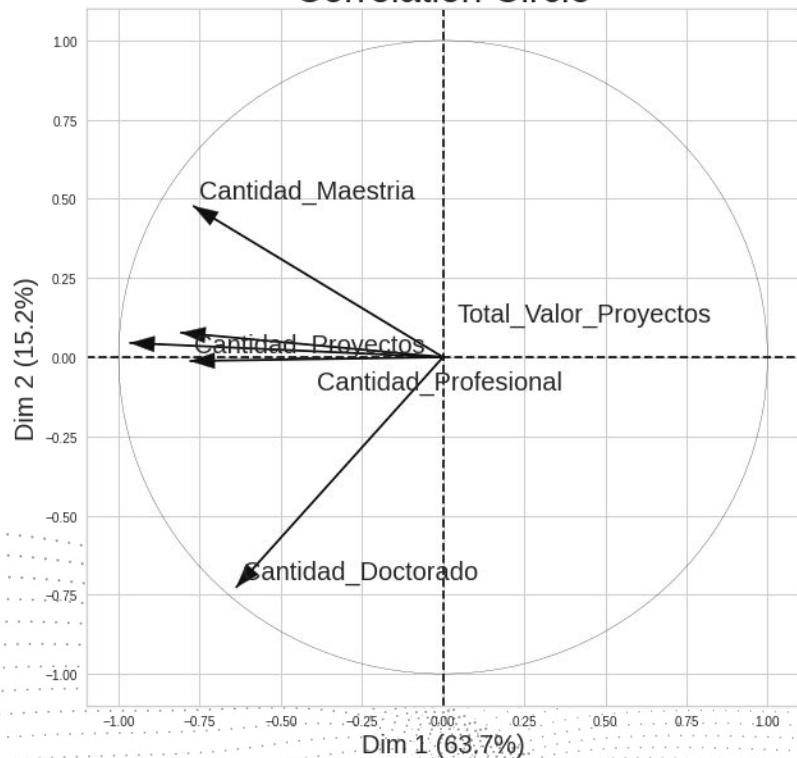
Internos

Coeficiente de Silhueta: 0.310

Calinski_harabasz: 115.798

Davies Bouldin: 1.333

Correlation Circle



Componentes Principales

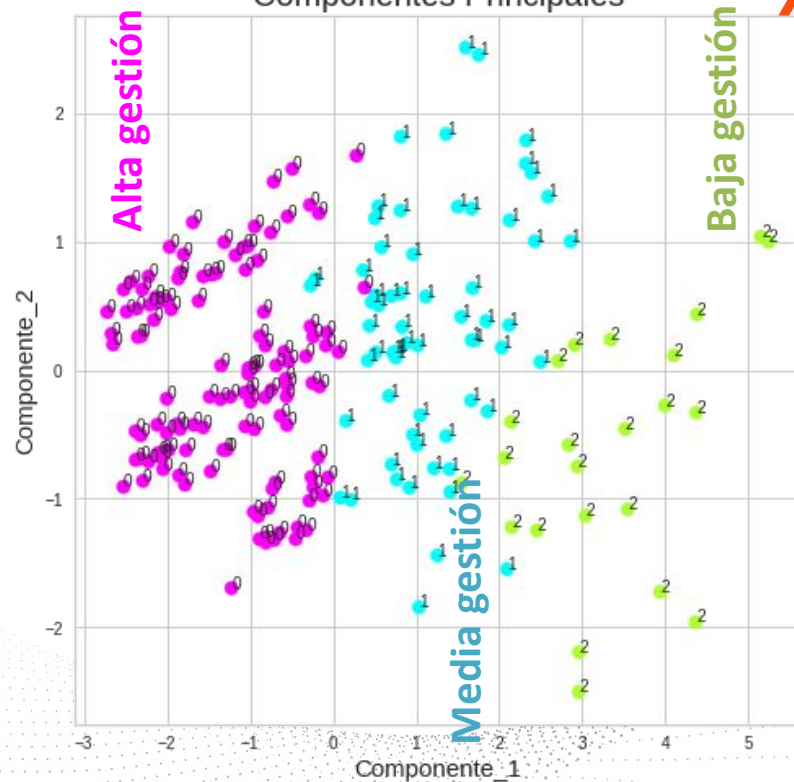
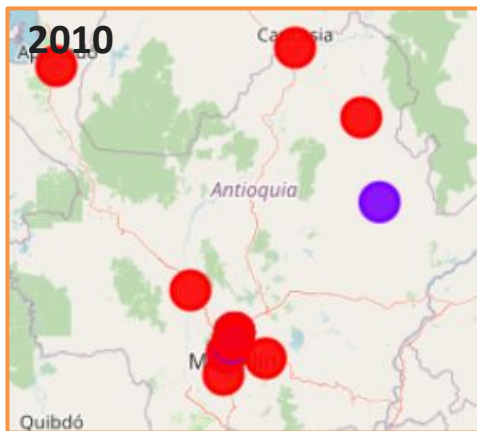
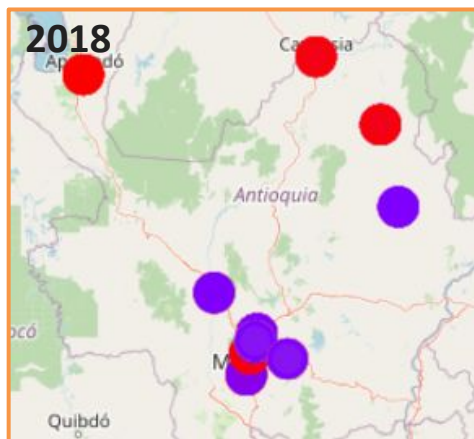
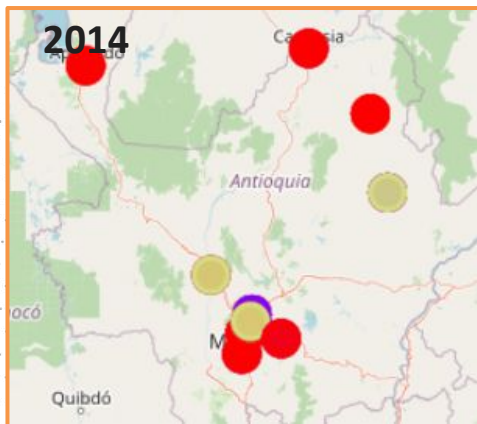
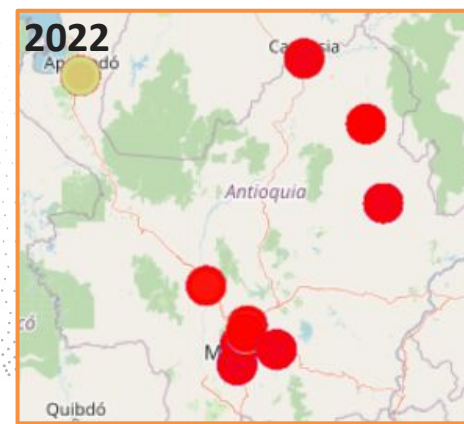


Tabla de datos final para visualización en mapas- Kmeans-PCA

Centro_Año	Cantidad_Proyectos	Cantidad_Profesional	Cantidad_Maestria	Cantidad_Doctorado	Total_Valor_Proyectos	Hac_Clusters	Gestión
9101-2010	-1.260308	-1.103402	-0.896032	-1.304164	-0.126344	0	Alta Gestión
9127-2010	-0.362554	-0.671184	-0.117850	-0.201456	-0.735003	0	Alta Gestión
9201-2010	-1.260308	-0.238966	-0.506941	-1.304164	-0.206320	0	Alta Gestión
9202-2010	0.535199	1.922122	0.271241	-1.304164	-0.608931	1	Media Gestión
9203-2010	-0.362554	1.057686	-0.117850	-0.201456	-1.577040	0	Alta Gestión
...	...	...	...	...	...	...	...
9501-2022	-0.362554	0.193251	-0.117850	0.901251	-0.872848	0	Alta Gestión
9502-2022	-1.260308	-0.238966	-0.896032	-1.304164	-0.687234	0	Alta Gestión
9503-2022	-1.260308	-1.535619	-1.674214	-0.201456	0.218826	0	Alta Gestión
9504-2022	1.432953	1.489904	1.049423	-0.201456	2.333091	1	Media Gestión

Mapas de centros de formación en alta y baja gestión a lo largo del tiempo - Cluster Kmeans- PCA



-  Alta Gestión
-  Media Gestión
-  Baja Gestión



Clusterización - Productividad



@SENAcomunica

www.sena.edu.co

Clusterización - Productividad 2010-2022

1. Kmeans - Validación - PCA
2. Agrupamiento jerárquico - Validación -PCA

Preprocesamiento



	Centro_Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
0	9101-2010	1	1	0	0	0	1
1	9127-2010	0	0	0	0	1	0
2	9201-2010	0	0	0	0	2	0
3	9202-2010	2	1	0	0	2	0
4	9203-2010	1	1	0	1	1	1
...	...	...	...	...	...	...	...
203	9501-2022	1	1	1	2	3	1
204	9502-2022	2	1	3	1	3	1
205	9503-2022	3	2	4	1	1	2
206	9504-2022	1	2	2	4	1	4
207	9549-2022	5	3	1	2	2	4

208 rows × 7 columns

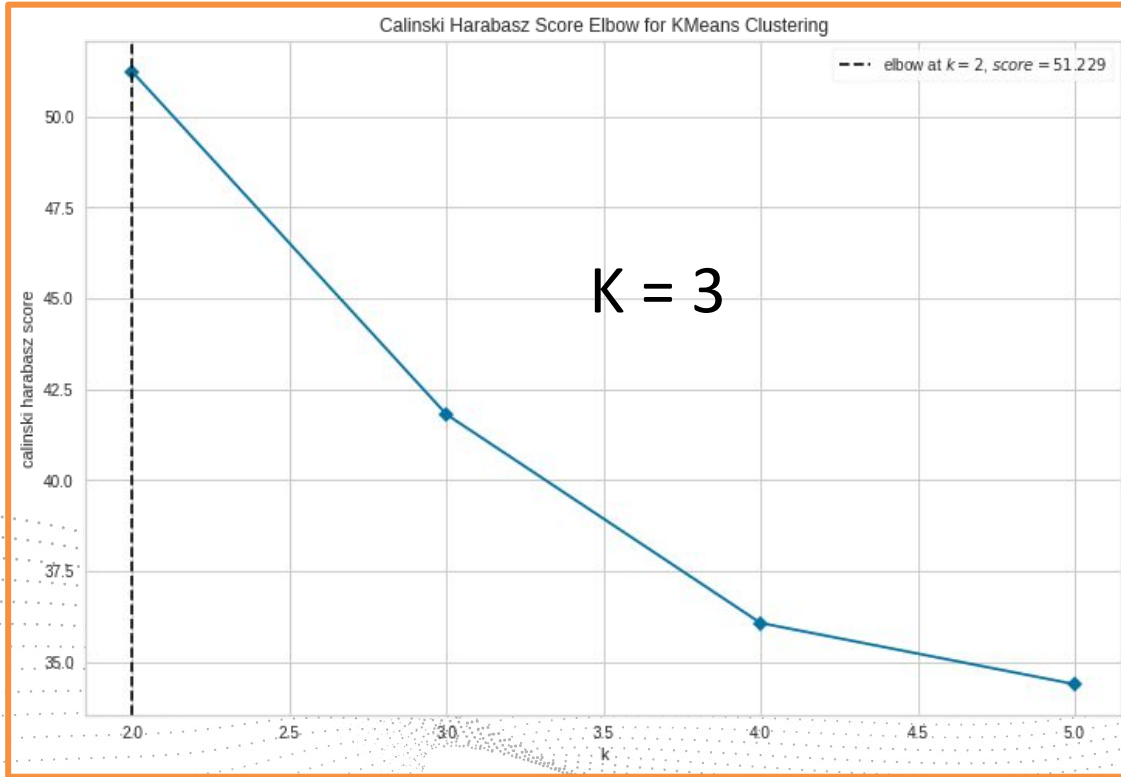
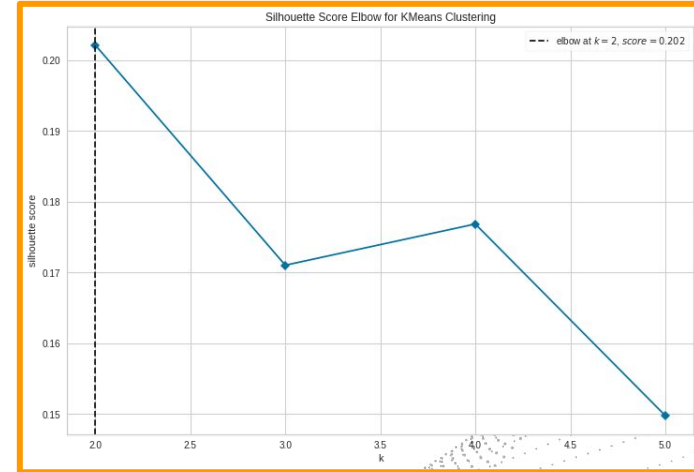
	Centro_Año	Apropiación_Social	Articulo_Sometido_en_revista_Indexada	Divulgación_Pública_de_la_
0	9101-2010	-0.609017	-0.526118	
1	9127-2010	-1.337038	-1.201628	
2	9201-2010	-1.337038	-1.201628	
3	9202-2010	0.119003	-0.526118	
4	9203-2010	-0.609017	-0.526118	
...	...	...	...	
203	9501-2022	-0.609017	-0.526118	
204	9502-2022	0.119003	-0.526118	
205	9503-2022	0.847024	0.149392	
206	9504-2022	-0.609017	0.149392	
207	9549-2022	2.303066	0.824901	

208 rows × 7 columns

Tiempo: 13 años
Centros de formación: 16
Columnas: 6
Filas: 208

KMeans - PCA

Método del codo : Determinación de centroides



Validación del agrupamiento por KMeans

Externos

Homogeneidad: 0.198

Compleitud: 1.000

Medida V: 0.330

Índice Rand corregido: 0.000

Adjusted Mutual Information: -0.000

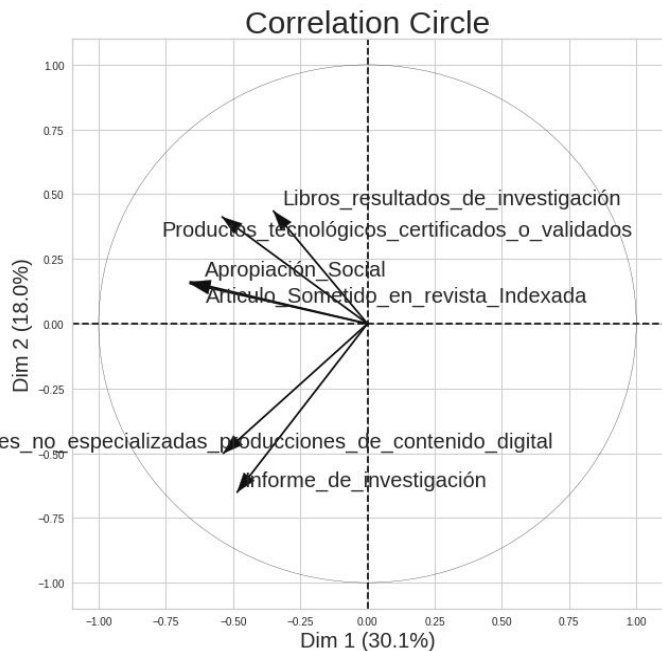
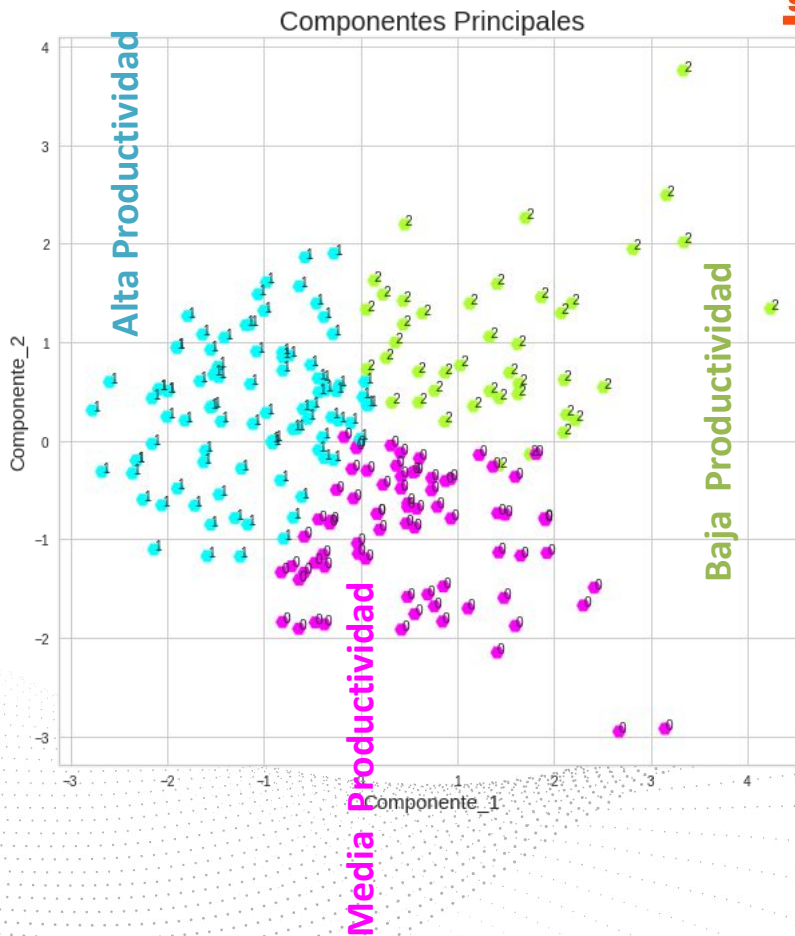
Índice Fowlkes-Mallows: 0.000

Internos

Coeficiente de Silhueta: 0.171

Calinski_harabasz: 41.734

Davies Bouldin: 1.902

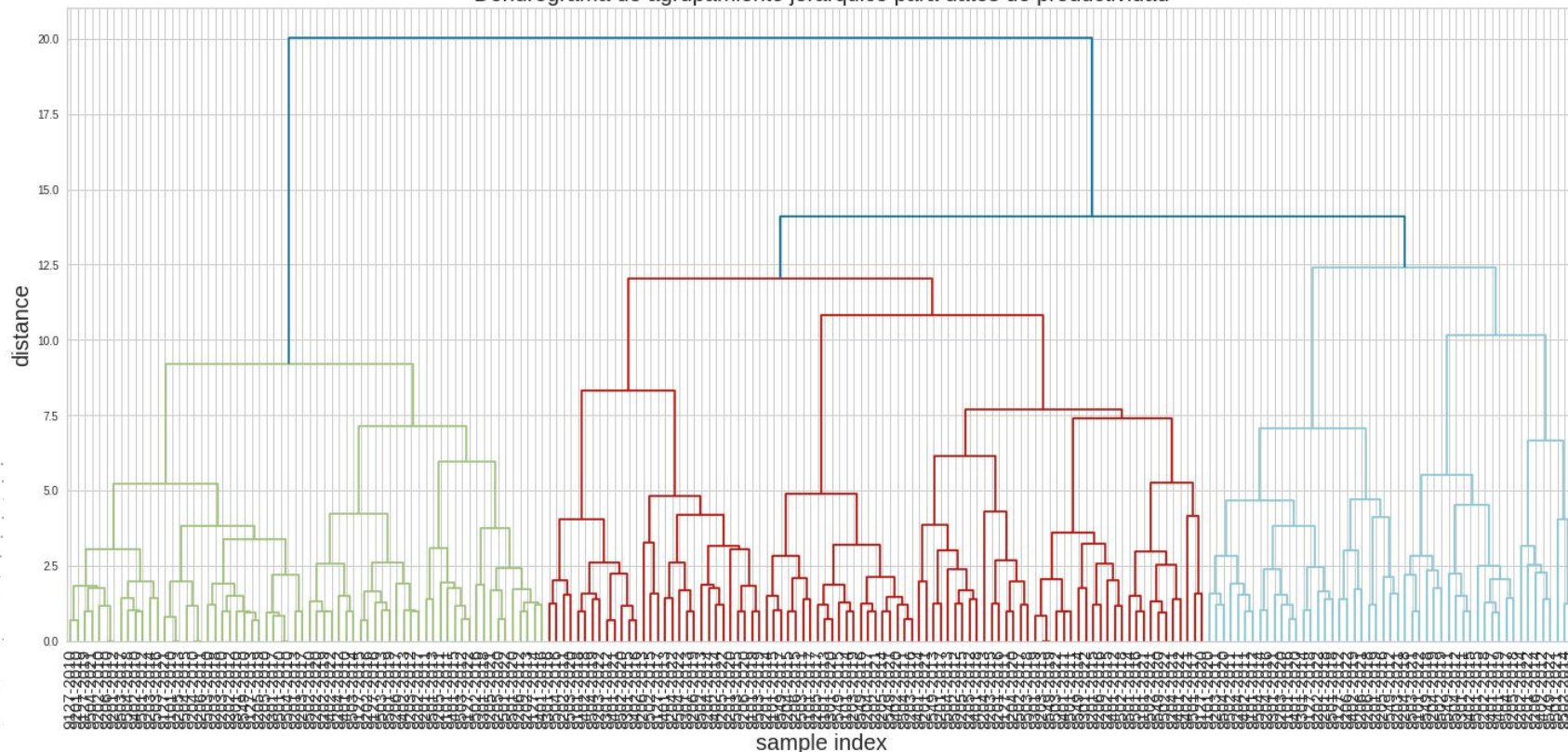


Jerárquico - PCA

Determinación de clusters -Dendrograma



Dendrograma de agrupamiento jerárquico para datos de productividad



Validación del agrupamiento por Jerárquico

Externos

Homogeneidad: 0.201

Complejidad: 1.000

Medida V: 0.334

Índice Rand corregido: 0.000

Adjusted Mutual Information: 0.000

Índice Fowlkes-Mallows: 0.000

Internos

Coeficiente de Silhueta: 0.133

Calinski_harabasz: 32.426

Davies Bouldin: 2.295

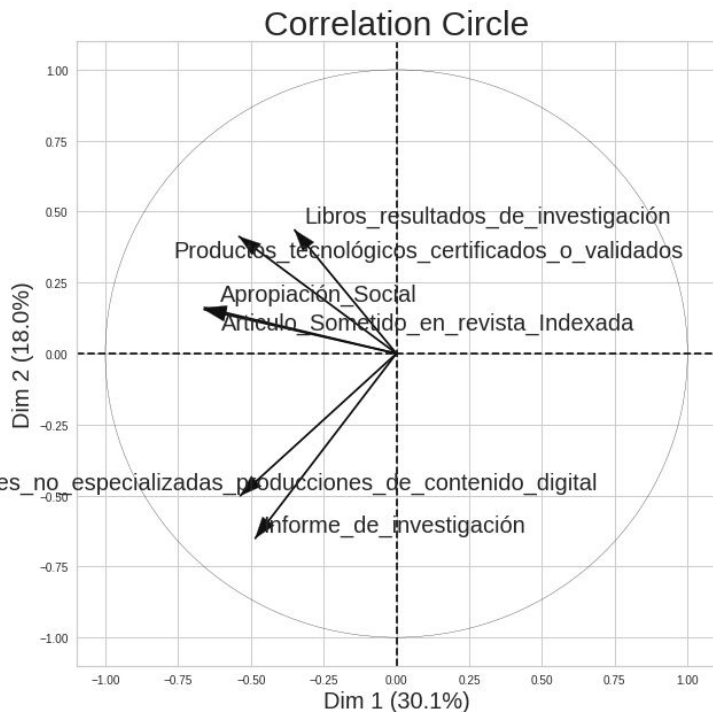
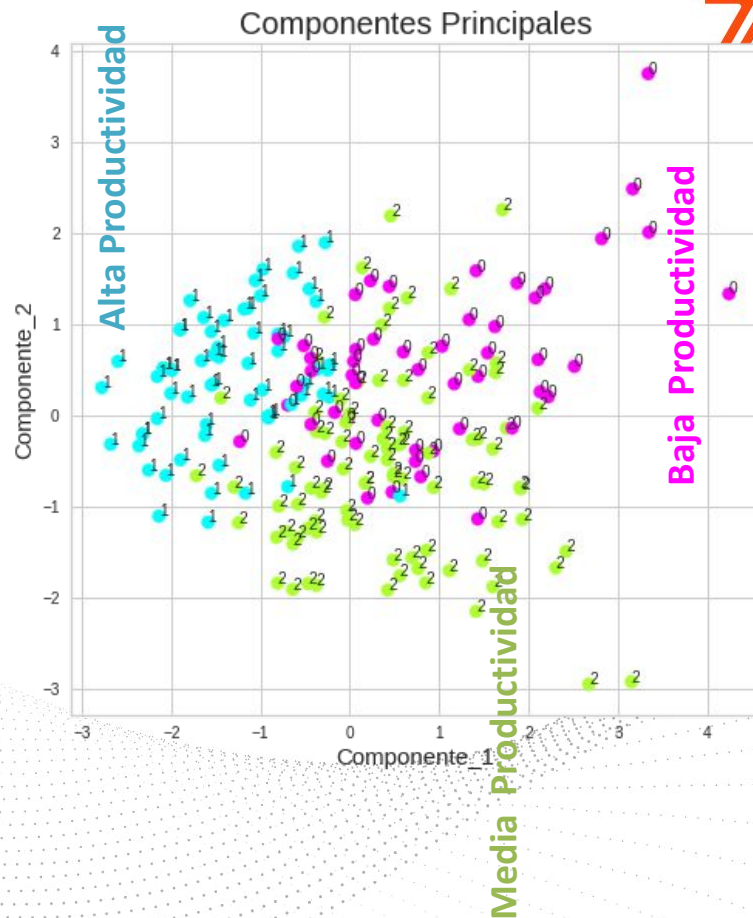


Tabla de datos final para visualización en mapas



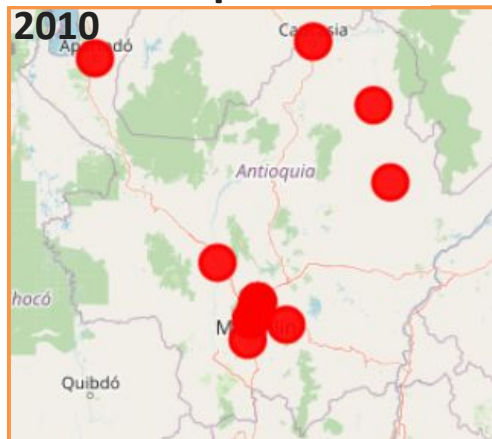
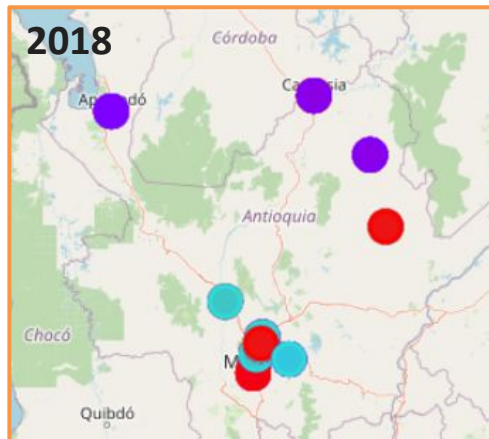
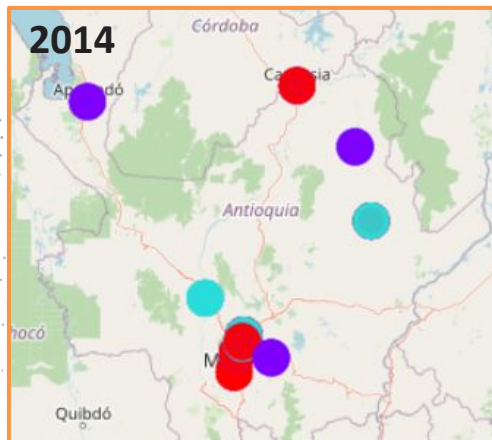
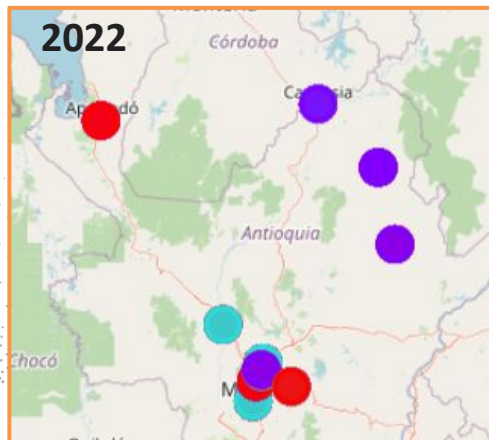
KMeans-PCA

	Codigo_Centro	Año	KMeans_Clusters	Productividad	Nombre_Centro	latitud	longitud
0	9101	2010	1	Alta Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
1	9101	2011	1	Alta Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
2	9101	2012	0	Baja Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
3	9101	2013	0	Baja Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
4	9101	2014	0	Baja Productividad	Centro_de_los_Recursos_Naturales_Renovables_La...	6.053157	-75.628486
...	...	...	...	...	...	...	...
203	9549	2018	2	Media Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
204	9549	2019	0	Baja Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
205	9549	2020	2	Media Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
206	9549	2021	0	Baja Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128
207	9549	2022	2	Media Productividad	Complejo_Tecnológico_Turístico_y_Agroindustria...	6.559073	-75.829128

208 rows × 7 columns

Mapas de centros de formación en alta y baja gestión a lo largo del tiempo -

Cluster Jerárquico - PCA



- Alta Productividad
- Media Productividad
- Baja Productividad



MÉTODOS DE CLASIFICACIÓN

Machine Learning Supervisado



@SENAcomunica

www.sena.edu.co



PRODUCTIVIDAD



@SENAcomunica

www.sena.edu.co



Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
9101	2010	Alta Productividad	1	1	0	0	0	1
9127	2010	Alta Productividad	0	0	0	0	1	0
9201	2010	Alta Productividad	0	0	0	0	2	0
9202	2010	Alta Productividad	2	1	0	0	2	0
9203	2010	Alta Productividad	1	1	0	1	1	1
...	...	...	...	...	...	...	...	...
9501	2022	Alta Productividad	1	1	1	2	3	1
9502	2022	Alta Productividad	2	1	3	1	3	1
9503	2022	Baja Productividad	3	2	4	1	1	2
9504	2022	Baja Productividad	1	2	2	4	1	4
9549	2022	Media Productividad	5	3	1	2	2	4

3: Alta Productividad
2: Media Productividad
1: Baja Productividad

Codigo_Centro	Año	Productividad	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
9101	2010	3	1	1	0	0	0	1
9127	2010	3	0	0	0	0	1	0
9201	2010	3	0	0	0	0	2	0
9202	2010	3	2	1	0	0	2	0
9203	2010	3	1	1	0	1	1	1
...	...	...	...	...	...	...	...	...
9501	2022	3	1	1	1	2	3	1
9502	2022	3	2	1	3	1	3	1
9503	2022	1	3	2	4	1	1	2
9504	2022	1	1	2	2	4	1	4
9549	2022	2	5	3	1	2	2	4

Matriz de confusión y parámetros de validación



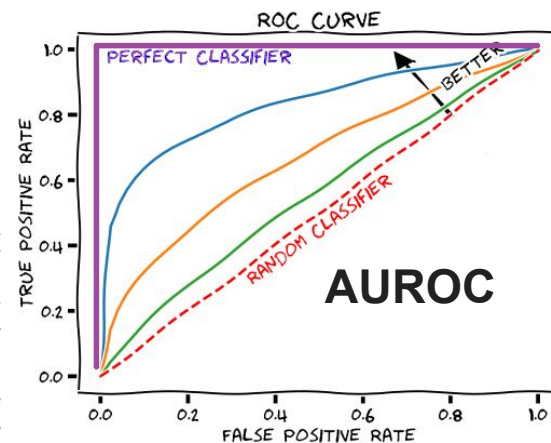
		Actual Values	
		Yes	No
Predicted Values	Yes	True Positive	False Positive
	No	False Negative	True Negative

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

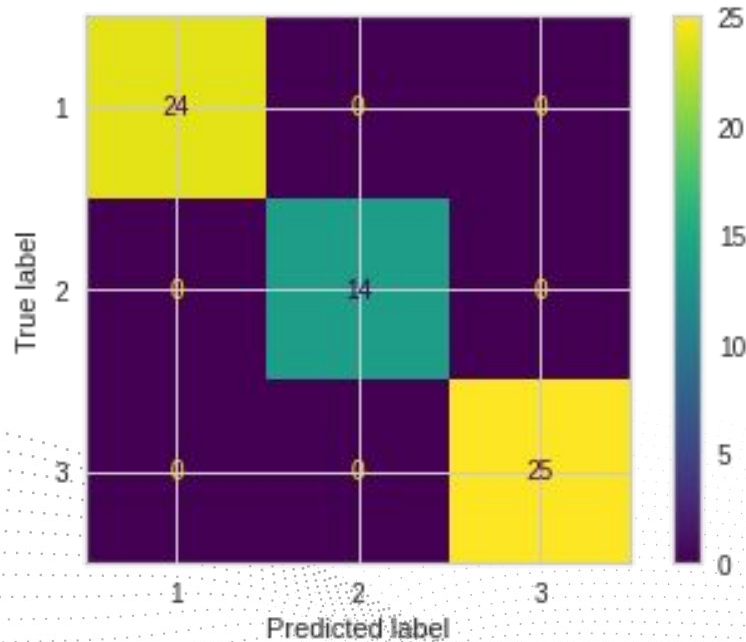
$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1-score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$



Regresión Logística



	precision	recall	f1-score	support
1	0.00	0.00	0.00	24
2	0.00	0.00	0.00	17
3	0.35	1.00	0.52	22
accuracy			0.35	63
macro avg	0.12	0.33	0.17	63
weighted avg	0.12	0.35	0.18	63

Regresión Logística

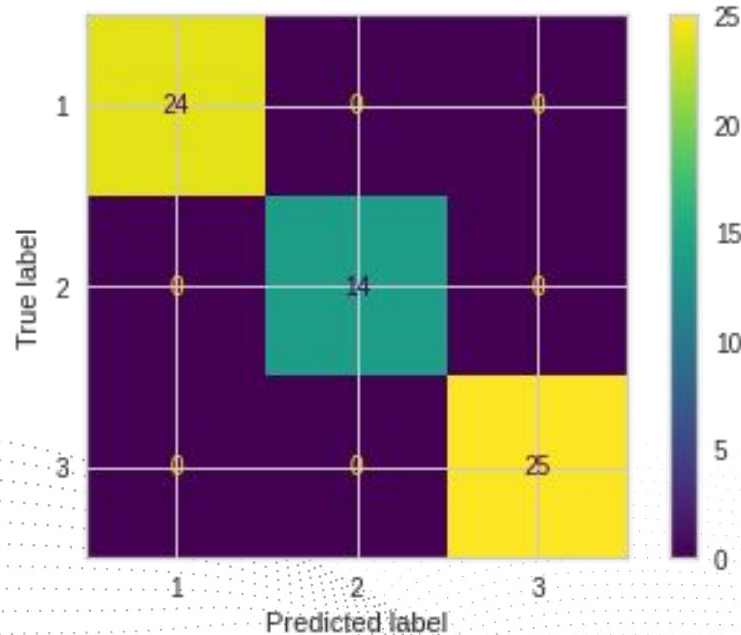
Datos de prueba

Codigo_Centro	Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
9101.0	2010.0	10.0	5.0	5.0	5.0	5.0	5.0
9127.0	2010.0	5.0	5.0	5.0	5.0	5.0	5.0
9201.0	2010.0	0.0	0.0	0.0	0.0	0.0	1.0
9202.0	2010.0	0.0	0.0	0.0	0.0	0.0	0.0
9203.0	2010.0	0.0	1.0	0.0	1.0	2.0	0.0
9504.0	2022.0	12.0	12.0	12.0	12.0	12.0	12.0

Predicción

	Codigo_Centro	Año	Productividad
0	9101.0	2010.0	Alta Productividad
1	9127.0	2010.0	Alta Productividad
2	9201.0	2010.0	Alta Productividad
3	9202.0	2010.0	Alta Productividad
4	9203.0	2010.0	Alta Productividad
5	9504.0	2022.0	Alta Productividad

Random Forest



	precision	recall	f1-score	support
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	14
3	1.00	1.00	1.00	25
accuracy			1.00	63
macro avg	1.00	1.00	1.00	63
weighted avg	1.00	1.00	1.00	63

Random Forest

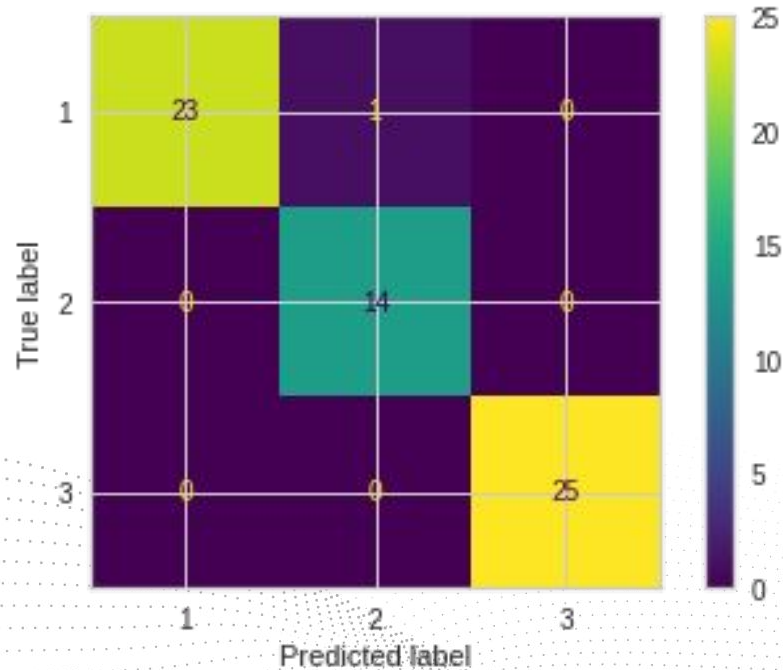
Datos de prueba

Codigo_Centro	Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
9101.0	2010.0	10.0	5.0	5.0	5.0	5.0	5.0
9127.0	2010.0	5.0	5.0	5.0	5.0	5.0	5.0
9201.0	2010.0	0.0	0.0	0.0	0.0	0.0	1.0
9202.0	2010.0	0.0	0.0	0.0	0.0	0.0	0.0
9203.0	2010.0	0.0	1.0	0.0	1.0	2.0	0.0
9504.0	2022.0	12.0	12.0	12.0	12.0	12.0	12.0

Predicción

	Codigo_Centro	Año	Productividad
0	9101.0	2010.0	Baja Productividad
1	9127.0	2010.0	Baja Productividad
2	9201.0	2010.0	Baja Productividad
3	9202.0	2010.0	Baja Productividad
4	9203.0	2010.0	Baja Productividad
5	9504.0	2022.0	Baja Productividad

Support vector machine(SVM)



	precision	recall	f1-score	support
1	1.00	0.96	0.98	24
2	0.93	1.00	0.97	14
3	1.00	1.00	1.00	25
accuracy			0.98	63
macro avg	0.98	0.99	0.98	63
weighted avg	0.99	0.98	0.98	63

Support vector machine(SVM)

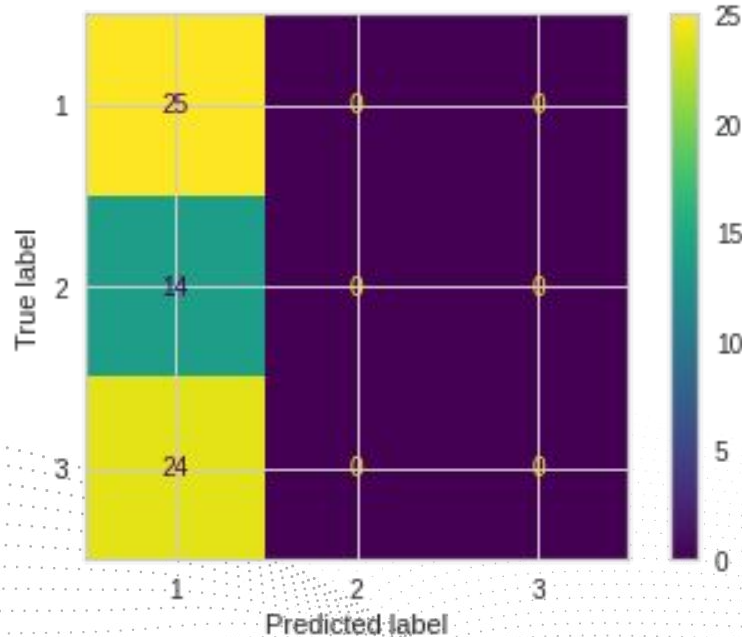
Datos de prueba

Codigo_Centro	Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
9101.0	2010.0	10.0	5.0	5.0	5.0	5.0	5.0
9127.0	2010.0	5.0	5.0	5.0	5.0	5.0	5.0
9201.0	2010.0	0.0	0.0	0.0	0.0	0.0	1.0
9202.0	2010.0	0.0	0.0	0.0	0.0	0.0	0.0
9203.0	2010.0	0.0	1.0	0.0	1.0	2.0	0.0
9504.0	2022.0	12.0	12.0	12.0	12.0	12.0	12.0

Predicción

	Codigo_Centro	Año	Productividad
0	9101.0	2010.0	Media Productividad
1	9127.0	2010.0	Media Productividad
2	9201.0	2010.0	Baja Productividad
3	9202.0	2010.0	Baja Productividad
4	9203.0	2010.0	Baja Productividad
5	9504.0	2022.0	Alta Productividad

Red neuronal: Perceptrón



	precision	recall	f1-score	support
1	0.56	1.00	0.71	35
2	0.00	0.00	0.00	7
3	0.00	0.00	0.00	21
accuracy			0.56	63
macro avg	0.19	0.33	0.24	63
weighted avg	0.31	0.56	0.40	63

Red neuronal: Perceptrón

Datos de prueba

Codigo_Centro	Año	P_8020	P_8025	P_8019	P_8039	P_8016	P_8031
9101.0	2010.0	10.0	5.0	5.0	5.0	5.0	5.0
9127.0	2010.0	5.0	5.0	5.0	5.0	5.0	5.0
9201.0	2010.0	0.0	0.0	0.0	0.0	0.0	1.0
9202.0	2010.0	0.0	0.0	0.0	0.0	0.0	0.0
9203.0	2010.0	0.0	1.0	0.0	1.0	2.0	0.0
9504.0	2022.0	12.0	12.0	12.0	12.0	12.0	12.0

Predicción

Codigo_Centro	Año	Productividad
9101.0	2010.0	Baja Productividad
9127.0	2010.0	Baja Productividad
9201.0	2010.0	Baja Productividad
9202.0	2010.0	Baja Productividad
9203.0	2010.0	Baja Productividad
9504.0	2022.0	Baja Productividad

Reporte de modelos - productividad

Parámetro	Perceptrón	Regresión logística	Random Forest	Máquina de soporte vectorial
Accuracy	0.380952	1.0	1.0	0.984127
F1-Score	0.210181	1.0	1.0	0.984232



PROCESOS



@SENAcomunica

www.sena.edu.co

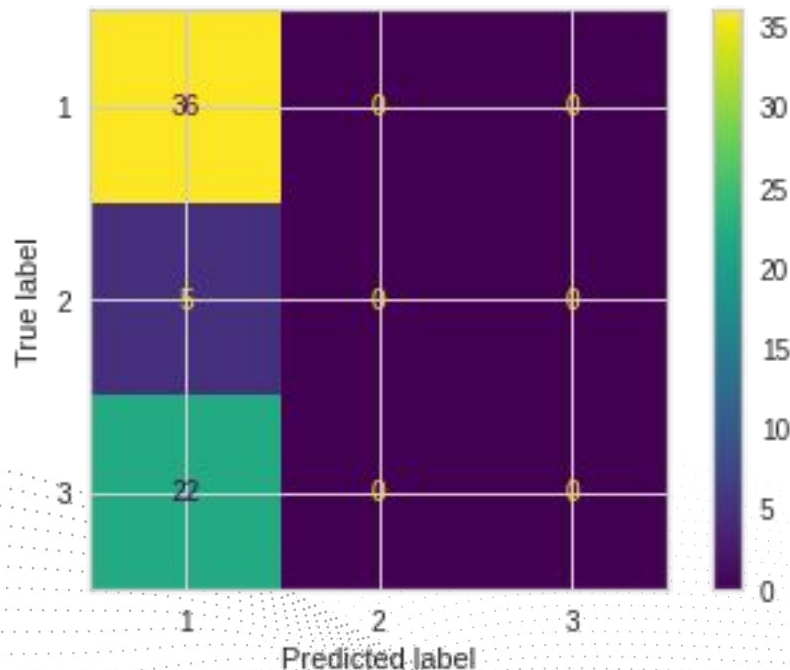


Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101	2010	Baja Gestión	1	1	2	0	4.652924e+08
9127	2010	Baja Gestión	2	2	4	1	3.920052e+08
9201	2010	Baja Gestión	1	3	3	0	4.556627e+08
9202	2010	Alta Gestión	3	8	5	0	4.071852e+08
9203	2010	Baja Gestión	2	6	4	1	2.906175e+08

3: Alta Gestión
2: Media Gestión
1: Baja Gestión

Codigo_Centro	Año	Gestión	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101	2010	1	1	1	2	0	4.652924e+08
9127	2010	1	2	2	4	1	3.920052e+08
9201	2010	1	1	3	3	0	4.556627e+08
9202	2010	3	3	8	5	0	4.071852e+08
9203	2010	1	2	6	4	1	2.906175e+08

Regresión Logística



	precision	recall	f1-score	support
1	0.57	1.00	0.73	36
2	0.00	0.00	0.00	5
3	0.00	0.00	0.00	22
accuracy			0.57	63
macro avg	0.19	0.33	0.24	63
weighted avg	0.33	0.57	0.42	63

Regresión Logística

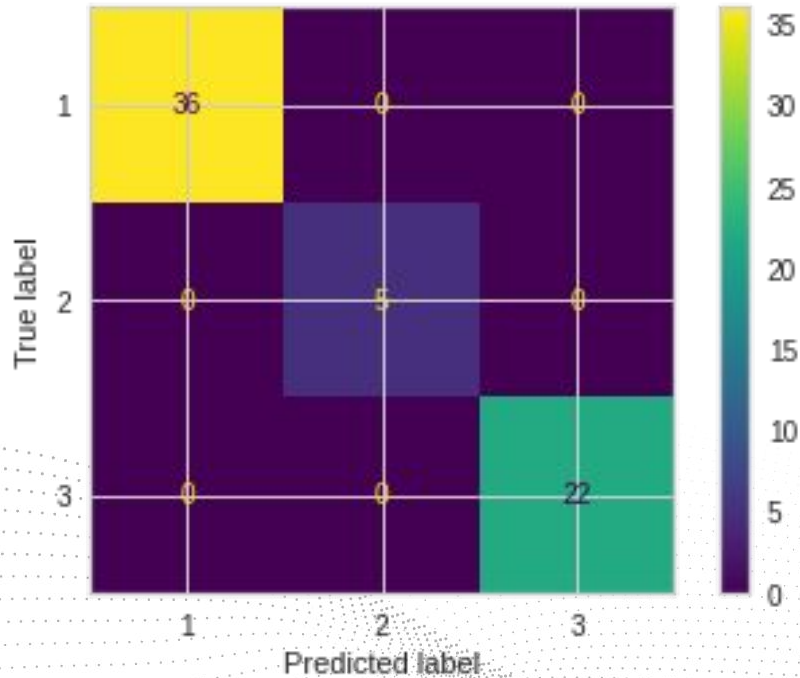
Datos de prueba

Codigo_Centro	Año	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101.0	2010.0	1.0	1.0	2.0	0.0	4.652924e+14
9127.0	2010.0	2.0	2.0	4.0	1.0	3.920052e+14
9201.0	2010.0	1.0	3.0	3.0	0.0	4.556627e+14
9202.0	2010.0	3.0	8.0	5.0	0.0	4.071852e+14
9203.0	2010.0	2.0	6.0	4.0	1.0	2.906175e+14
9501.0	2022.0	2.0	4.0	4.0	2.0	3.754077e+14
9502.0	2022.0	1.0	3.0	2.0	0.0	3.977569e+14
9503.0	2022.0	1.0	0.0	0.0	1.0	5.068536e+14
9504.0	2022.0	4.0	7.0	7.0	1.0	7.614273e+14
9549.0	2022.0	2.0	5.0	2.0	1.0	3.294535e+14

Predicción

Codigo_Centro	Año	Gestión	Productividad
9101.0	2010.0	1.0	Baja Gestión
9127.0	2010.0	1.0	Baja Gestión
9201.0	2010.0	1.0	Baja Gestión
9202.0	2010.0	1.0	Baja Gestión
9203.0	2010.0	1.0	Baja Gestión
9501.0	2022.0	1.0	Baja Gestión
9502.0	2022.0	1.0	Baja Gestión
9503.0	2022.0	1.0	Baja Gestión
9504.0	2022.0	1.0	Baja Gestión
9549.0	2022.0	1.0	Baja Gestión

Random Forest



	precision	recall	f1-score	support
1	1.00	1.00	1.00	36
2	1.00	1.00	1.00	5
3	1.00	1.00	1.00	22
accuracy			1.00	63
macro avg	1.00	1.00	1.00	63
weighted avg	1.00	1.00	1.00	63

Random Forest

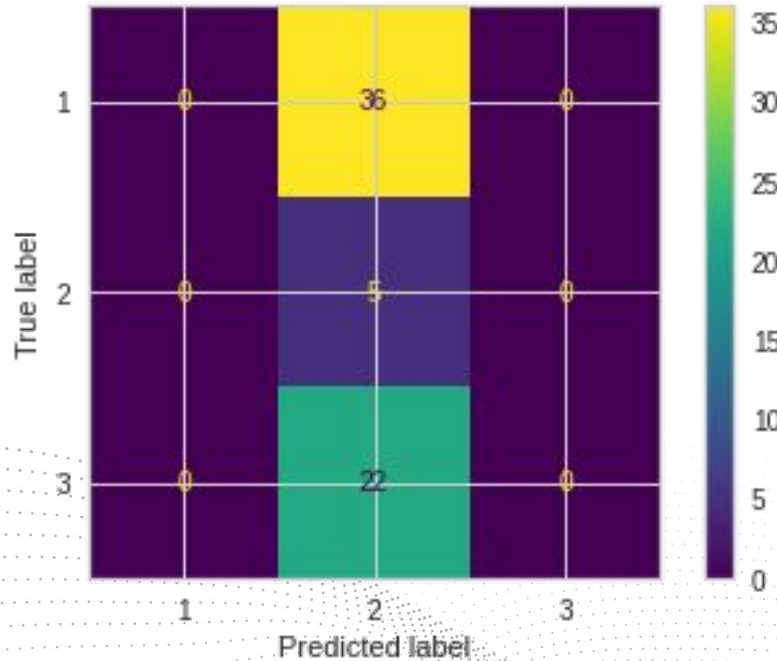
Datos de prueba

Codigo_Centro	Año	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101.0	2010.0	1.0	1.0	2.0	0.0	4.652924e+14
9127.0	2010.0	2.0	2.0	4.0	1.0	3.920052e+14
9201.0	2010.0	1.0	3.0	3.0	0.0	4.556627e+14
9202.0	2010.0	3.0	8.0	5.0	0.0	4.071852e+14
9203.0	2010.0	2.0	6.0	4.0	1.0	2.906175e+14
9501.0	2022.0	2.0	4.0	4.0	2.0	3.754077e+14
9502.0	2022.0	1.0	3.0	2.0	0.0	3.977569e+14
9503.0	2022.0	1.0	0.0	0.0	1.0	5.068536e+14
9504.0	2022.0	4.0	7.0	7.0	1.0	7.614273e+14
9549.0	2022.0	2.0	5.0	2.0	1.0	3.294535e+14

Predicción

Codigo_Centro	Año	Gestión	Productividad
9101.0	2010.0	1.0	Baja Gestión
9127.0	2010.0	1.0	Baja Gestión
9201.0	2010.0	1.0	Baja Gestión
9202.0	2010.0	1.0	Media Gestión
9203.0	2010.0	1.0	Baja Gestión
9501.0	2022.0	1.0	Baja Gestión
9502.0	2022.0	1.0	Baja Gestión
9503.0	2022.0	1.0	Baja Gestión
9504.0	2022.0	1.0	Media Gestión
9549.0	2022.0	1.0	Baja Gestión

Support vector machine(SVM)



	precision	recall	f1-score	support
1	0.00	0.00	0.00	36
2	0.08	1.00	0.15	5
3	0.00	0.00	0.00	22
accuracy			0.08	63
macro avg	0.03	0.33	0.05	63
weighted avg	0.01	0.08	0.01	63

Support vector machine(SVM)

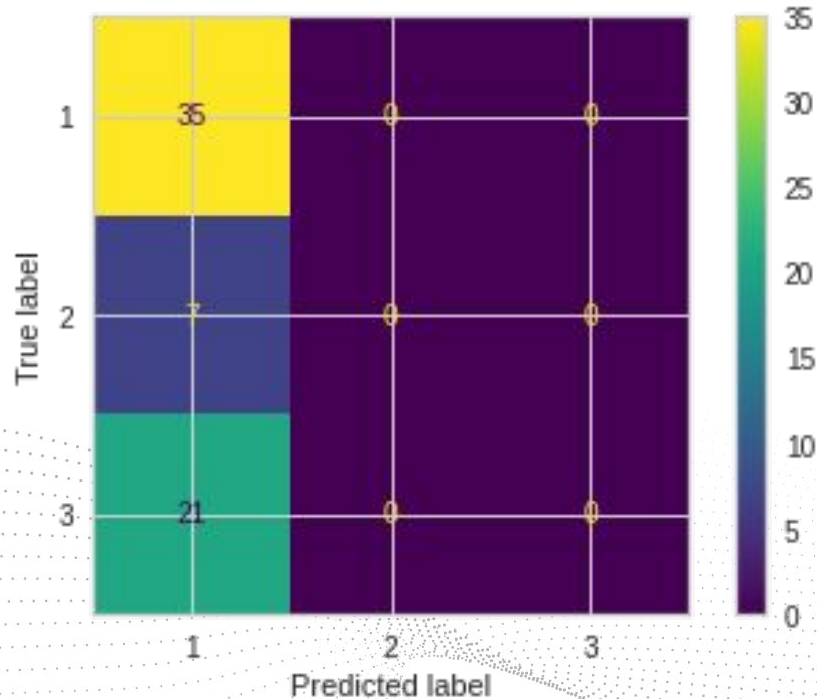
Datos de prueba

Codigo_Centro	Año	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101.0	2010.0	1.0	1.0	2.0	0.0	4.652924e+14
9127.0	2010.0	2.0	2.0	4.0	1.0	3.920052e+14
9201.0	2010.0	1.0	3.0	3.0	0.0	4.556627e+14
9202.0	2010.0	3.0	8.0	5.0	0.0	4.071852e+14
9203.0	2010.0	2.0	6.0	4.0	1.0	2.906175e+14
9501.0	2022.0	2.0	4.0	4.0	2.0	3.754077e+14
9502.0	2022.0	1.0	3.0	2.0	0.0	3.977569e+14
9503.0	2022.0	1.0	0.0	0.0	1.0	5.068536e+14
9504.0	2022.0	4.0	7.0	7.0	1.0	7.614273e+14
9549.0	2022.0	2.0	5.0	2.0	1.0	3.294535e+14

Predicción

Codigo_Centro	Año	Gestión	Productividad
9101.0	2010.0	1.0	Media Gestión
9127.0	2010.0	1.0	Media Gestión
9201.0	2010.0	1.0	Media Gestión
9202.0	2010.0	1.0	Media Gestión
9203.0	2010.0	1.0	Media Gestión
9501.0	2022.0	1.0	Media Gestión
9502.0	2022.0	1.0	Media Gestión
9503.0	2022.0	1.0	Media Gestión
9504.0	2022.0	1.0	Media Gestión
9549.0	2022.0	1.0	Media Gestión

Red neuronal: Perceptrón



	precision	recall	f1-score	support
1	0.56	1.00	0.71	35
2	0.00	0.00	0.00	7
3	0.00	0.00	0.00	21
accuracy			0.56	63
macro avg	0.19	0.33	0.24	63
weighted avg	0.31	0.56	0.40	63

Red neuronal: Perceptrón

Datos de prueba

Codigo_Centro	Año	Q_Proyectos	QNA_5007	QNA_5009	QNA_5010	TV_Proyectos
9101.0	2010.0	1.0	1.0	2.0	0.0	4.652924e+14
9127.0	2010.0	2.0	2.0	4.0	1.0	3.920052e+14
9201.0	2010.0	1.0	3.0	3.0	0.0	4.556627e+14
9202.0	2010.0	3.0	8.0	5.0	0.0	4.071852e+14
9203.0	2010.0	2.0	6.0	4.0	1.0	2.906175e+14
9501.0	2022.0	2.0	4.0	4.0	2.0	3.754077e+14
9502.0	2022.0	1.0	3.0	2.0	0.0	3.977569e+14
9503.0	2022.0	1.0	0.0	0.0	1.0	5.068536e+14
9504.0	2022.0	4.0	7.0	7.0	1.0	7.614273e+14
9549.0	2022.0	2.0	5.0	2.0	1.0	3.294535e+14

Predicción

Codigo_Centro	Año	Gestión
9101.0	2010.0	Baja Gestión
9127.0	2010.0	Baja Gestión
9201.0	2010.0	Baja Gestión
9202.0	2010.0	Baja Gestión
9203.0	2010.0	Baja Gestión
9501.0	2022.0	Baja Gestión
9502.0	2022.0	Baja Gestión
9503.0	2022.0	Baja Gestión
9504.0	2022.0	Baja Gestión
9549.0	2022.0	Baja Gestión

Reporte de modelos - procesos

Parámetro	Perceptrón	Regresión logística	Random Forest	Máquina de soporte vectorial
Accuracy	0.571429	0.571429	1.0	0.079365
F1-Score	0.415584	0.415584	1.0	0.011671



6. Divulgación



@SENAcomunica

www.sena.edu.co

Actividades

- Proyecto Promoción de Ecosistemas de Innovación en América Latina - Componentes C3 OTC (Oficina de Transferencia de Conocimiento) y C4 (Gestión de Capital Documental).
- Cursos de transferencia de conocimientos dictados por la Universidad Politécnica de Valencia.
 - La función de transferencia de conocimiento en ecosistemas de innovación
 - Gestión del capital intelectual en SENNOVA.
- Elaboración de actas de los prototipos.

Pendiente

- Aportes para el artículo de visualización de la información
- Reunión equipo de divulgadores para ajustar el plan de transferencia del proyecto 9748



G R A C I A S

Línea de atención al ciudadano: 018000 910270
Línea de atención al empresario: 018000 910682



@SENAcomunica

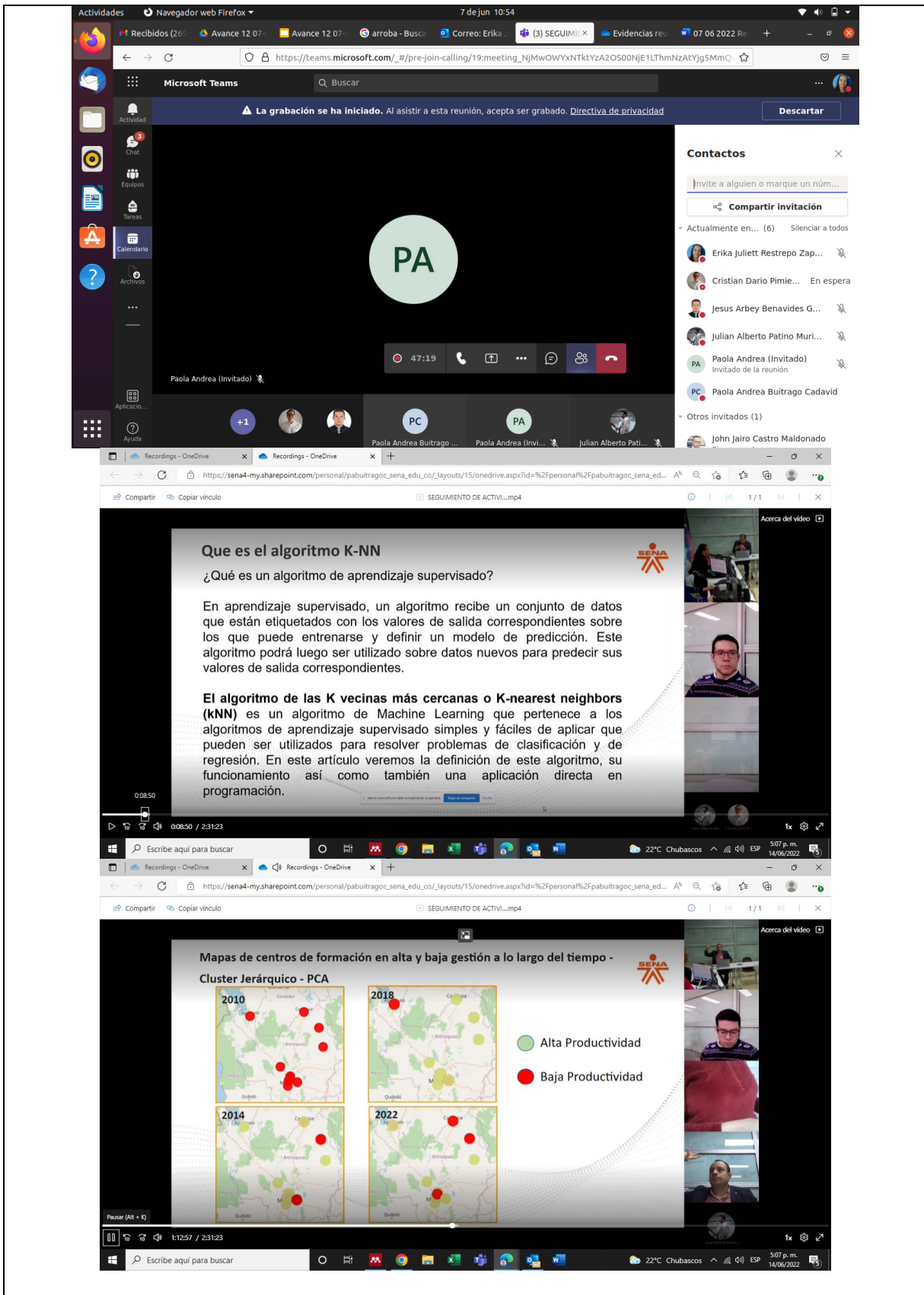
www.sena.edu.co



ANEXOS 5. Actas de reuniones.

ACTA No. 93		
NOMBRE DEL COMITÉ O DE LA REUNIÓN: Seguimiento Proyecto SGPS 9748 - Prototipo Innovación		
CIUDAD Y FECHA: Medellín, 07 de junio de 2022	HORA INICIO: 10:00 a.m.	HORA FIN: 1:00 p.m.
LUGAR Y/O ENLACE: Laboratorio de Inteligencia Artificial Piso 5	DIRECCIÓN GENERAL / REGIONAL / CENTRO Centro de Servicios y Gestión Empresarial	
AGENDA O PUNTOS PARA DESARROLLAR: 1. Avances de los modelos de analítica de datos. 2. Avances de actividades de divulgación		
OBJETIVO(S) DE LA REUNIÓN: Realizar seguimiento a la ejecución de las actividades del Proyecto SGPS 9748 Prototipo Innovación.		
DESARROLLO DE LA REUNIÓN		
<p>1. Avances de los modelos de analítica de datos.</p> <p>Para iniciar la investigadora Paola Buitrago inició presentando las generalidades de lo que es un modelo de aprendizaje supervisado y la importancia de realizar clusterización antes de aplicar un modelo K-Nearest Neighbors (KNN). Además, socializó el concepto de lo que comprende el algoritmo KNN, también presentó el proceso que llevó a cabo para implementar este algoritmo en R statistics, incluyendo la respectiva matriz de confusión con la explicación de las métricas asociadas tales como exactitud, precisión, sensibilidad y especificidad, también compartió la validación del modelo a través de la matriz de confusión, la evaluación de su rendimiento, el análisis y las conclusiones de los resultados obtenidos. Con respecto a los resultados anteriores se sugirió incluir la evaluación con todas las variables al tiempo, presentándolas todas en un cuadro resumen.</p> <p>Por su parte, el investigador Jesús Benavides presentó otros métodos de clasificación de Machine Learning supervisados presentando dos clústeres en la visualización de los mapas de centros de formación en alta y baja gestión a lo largo del tiempo, se acordó ajustar este proceso a tres clústeres porque se presenta una información muy parcializada, se considera mejor que se presente por ejemplo las categorías de alto, medio y bajo. Adicionalmente, presentó la matriz de confusión y los parámetros de validación. Los modelos supervisados que presentó fueron regresión logística, Random Forest, Support vector machine (SVM) y red neuronal: Perceptrón, para las bases de datos de productividad, procesos, con la explicación de las métricas asociadas tales como exactitud, precisión, sensibilidad y especificidad, también compartió la validación del modelo a través de la matriz de confusión, la evaluación de su rendimiento, el análisis y las conclusiones de los resultados obtenidos. Los resultados obtenidos con los modelos deben ser ajustados a los tres clústeres.</p>		
<p>2. Avances de actividades de divulgación</p> <p>La investigadora Erika Restrepo realizó las siguientes actividades:</p> <ul style="list-style-type: none">Participación en el Proyecto Promoción de Ecosistemas de Innovación en América Latina - Componentes C3 OTC (Oficina de Transferencia de Conocimiento) y C4 (Gestión de Capital Documental).Realización de los cursos de transferencia de conocimientos dictados por la Universidad Politécnica de Valencia:<ul style="list-style-type: none">La función de transferencia de conocimiento en ecosistemas de innovaciónGestión del capital intelectual en SENNOVA.		

<ul style="list-style-type: none"> • Elaboración de actas de los prototipos. <p>Tiene pendiente realizar las siguientes actividades:</p> <ul style="list-style-type: none"> • Aportes para el artículo de visualización de la información • Ajustar el plan de transferencia con el equipo de divulgadores. 		
CONCLUSIONES		
<p>Se continúa avanzando en la aplicación de los modelos de analítica de datos, se espera en los próximos días avanzar de una manera más rápida en estas actividades para poder integrarlas al aplicativo de recolección de información que se está ajustando.</p> <p>Se solicita a los investigadores tener en cuenta realizar los ajustes requeridos a los modelos. Además, tener en cuenta la tarea transversal de escribir los artículos.</p> <p>Se está gestionando la posibilidad de que el laboratorio de inteligencia tenga continuidad y se pueda continuar desarrollando proyectos utilizando las tecnologías que se tienen disponibles en el centro.</p>		
<p>De acuerdo con La Ley 1581 de 2012, Protección de Datos Personales se debe garantizar la seguridad y protección de los datos personales que se encuentran almacenados en este documento. El Servicio Nacional de Aprendizaje SENA solicita la siguiente clasificación de la información:</p> <p>La información de este documento se debe clasificar como:</p> <p>PÚBLICA ☐ PRIVADO ☐ SEMIPRIVADO ☐ SENSIBLE ☐</p> <p>Nota: Antes de contestar esta informacional por favor remitirse al instructivo</p>		
COMPROMISOS		
ACTIVIDAD	RESPONSABLE	FECHA
Continuar aplicando los modelos de analítica de datos.	Jesús Arbey Benavides Paola Andrea Buitrago	14 de junio de 2022
Continuar con las actividades de divulgación del prototipo.	Erika Restrepo	14 de junio de 2022
ASISTENTES: (Incorporar registro de asistencia)		
<p>Se anexa evidencia fotográfica de la reunión:</p> <p>Julián Alberto Patiño, Coordinador líder del proyecto y Cristian Pimienta, Dinamizador SENNOVA se conectaron de manera remota a la reunión.</p>		





REGISTRO DE ASISTENCIA Y APROBACIÓN DEL ACTA No. 93 DEL DÍA 09 DEL MES DE junio DEL AÑO 2022.

OBJETIVO(S) Reunión de seguimiento de ejecución de actividades proyecto 9748 Prototipo innovación										
No.	NOMBRES Y APELLIDOS	No. DOCUMENTO	PLANTA	CONTRATISTA	OTRO ¿CUAL?	DEPENDENCIA/ EMPRESA	CORREO ELECTRÓNICO	TELÉFONO/EXT. SENA	AUTORIZA GRABACIÓN	FIRMA O PARTICIPACIÓN VIRTUAL
1	John Jairo Castro H	88265552	X			SENA CES68	jcastro@sena.edu.co	42850	Si	[Firma]
2	Srika Juliet Rastrogo	139773982		X		SENA SENNOVA	erastrogo@sena.edu.co	3005017253	Si	[Firma]
3	Jesús Arbey Benavides	1085460713		X		SENA SENNOVA	jbenavides@sena.edu.co	3105595765	Si	Jesús Arbey
4	Paola Andrea Baetrago	43641529		X		SENA SENNOVA	Pbaetrago@sena.edu.co	3113671040	Si	Paola Baetrago

De acuerdo con La Ley 1581 de 2012, Protección de Datos Personales, el Servicio Nacional de Aprendizaje SENA, se compromete a garantizar la seguridad y protección de los datos personales que se encuentran almacenados en este documento, y les dará el tratamiento correspondiente en cumplimiento de lo establecido legalmente.