



Tecnológico de Monterrey

Escuela de Ingeniería y Ciencias

Campus LATAM

Nombre del trabajo:

**8.3 Fase 4a. Modelado de datos, regresión lineal, perfiles socioeconómicos de las regiones
de Jalisco**

Curso:

Análisis de Grandes Volúmenes de Datos

Alumnos:

Juan Carlos Alvarado Carricarte, A01793486

Bryan Rodolfo Alvarado Cruz, A01793670

Eduardo Gabriel Arévalo Aguilar, A01793897

Profesor:

Alberto De Obeso Orendain

Tutor:

Luis Angel Lozano Medina

Fecha de entrega:

05 de marzo

Resumen

En el presente trabajo se evaluaron las semejanzas existentes entre municipios de la región de Jalisco agrupándolos en clústeres. Se tomaron los datos demográficos de la tabla tasa crecimiento natural generados en el proceso de enriquecimiento de datos de la actividad pasada, la tabla contenía los campos cve municipio, desc municipio, nacimientos, muertes, población total y tasa crecimiento natural, pero se le adicionaron la tasa de mortalidad y tasa de natalidad para tener más características que nos permitieron obtener una mejor perspectiva.

Mediante el método de K-means, se hizo uso de tres variables: población total, tasa de mortalidad y tasa de natalidad. Se utilizó el método de Silhouette para determinar el número óptimo de clusters y se encontraron cuatro clusters. Se describió cada uno de los clústeres en términos de sus características, evaluando métricas como la mediana para cada una de las variables del dataset principal. Este análisis permitió una mejor comprensión de la estructura de las regiones de Jalisco en términos sociales como su población, facilitando la identificación de patrones y tendencias.

Palabras clave: K-means, clustering, aprendizaje no supervisado, análisis estadístico, segmentación de datos, Apache Spark, Databricks, datos demográficos, México, Jalisco, Población.

K-means, aprendizaje no supervisado

El análisis de datos mediante modelos de aprendizaje no supervisado, nos permite descubrir patrones de datos complejos. Una de las técnicas más comunes para el aprendizaje no supervisado es el clustering, que agrupa elementos de datos similares en grupos o clústeres para su posterior análisis. K-means es un algoritmo de clustering muy utilizado, este método divide los datos en k clusters de manera iterativa.

En este trabajo, utilizamos el método de k-means para analizar datos socioeconómicos de las regiones de Jalisco, México. Se seleccionaron variables relevantes relacionadas con la población y se agruparon en un solo vector, se escalaron las variables, y se usa el método de Siouhette para determinar el número óptimo de clusters. Luego, se genera el modelo de k-means y se describe cada uno de los grupos resultantes.

El objetivo final es proporcionar una segmentación clara y útil de las regiones de Jalisco, lo que permitirá una mejor comprensión de las características socioeconómicas de cada municipio y facilitará la toma de decisiones informadas en cuanto a políticas públicas y desarrollo económico.

Clustering con K-means

Para llevar a cabo el modelado de datos y el clustering utilizando K-means, se utiliza Apache Spark en conjunto con el lenguaje de programación Python sobre la plataforma de Databricks.

En primer lugar, se le agregan 2 características nuevas a nuestra tabla de crecimiento natural de la población (tasa de natalidad y de mortalidad):

cve_municipio	desc_municipio	nacimientos	muerdes	poblacion_total	tasa_crecimiento_natural	tasa_de_mortalidad	tasa_de_natalidad
19	Bolaños	291	43	7043	35.21226749964504	6.105352832599744	41.31762033224478
61	Mezquitic	770	115	22083	29.660825069057648	5.207625775483404	34.86845084454105
125	San Ignacio Cerro Gordo	522	139	18341	20.882176544354177	7.578648928629846	28.460825472984027
71	San Cristóbal de la Barranca	63	16	2924	16.073871409028726	5.471956224350205	21.545827633378934
106	Tuxcacuesco	122	35	5482	15.870120394016782	6.384531192995258	22.25465158701204
111	Valle de Guadalupe	176	71	6627	15.844273426889995	10.713746793420855	26.55802022031085
86	Taalua	471	138	21245	15.674276300305955	6.495646034361026	22.16992233466698

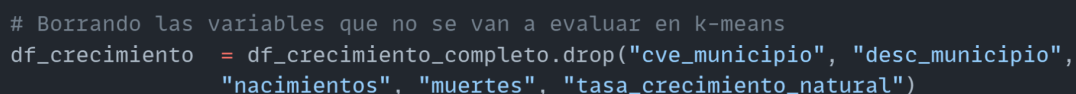
Posteriormente se carga la tabla de datos `tasa_crecimiento_natural_adicional` en formato CSV de Spark, se convierten los datos en un Dataframe de Spark, se seleccionan las variables numéricas necesarias para el modelo de clustering.

1. Justificación de las variables elegidas para el modelo de k-means.

Las variables seleccionadas para la aplicación del modelo K-means fueron la población total, la tasa de mortalidad y la tasa de natalidad. La elección de estas variables se basó en la idea de que la población total de una región es un factor importante a considerar al analizar su perfil económico. Por otro lado, la tasa de mortalidad y la tasa de natalidad también son variables relevantes, ya que proporcionan información sobre la salud y el bienestar de la población de una región.

Se decidió evitar la inclusión de la variable tasa de crecimiento natural, debido a que al hacer pruebas con K-means y PCA su inclusión generaba ruido y no aportaba información significativa en el modelo de clustering.

Las variables de tasa de natalidad, mortalidad y crecimiento natural tienen una unidad de x1000 (por cada mil habitantes), mientras que la de población total (cantidad de personas), era evidente que había una diferencia de magnitudes y medidas enormes, por lo que se optó a escalarlas luego de eliminar las variables que no se usarían y unificarlas en un solo vector



```
# Borrando las variables que no se van a evaluar en k-means
df_crecimiento = df_crecimiento_completo.drop("cve_municipio", "desc_municipio",
                                              "nacimientos", "muertes", "tasa_crecimiento_natural")
```

Agregando las variables a un único vector con Assembler:

```
# create a vector assembler and transform raw features into a single set of features
from pyspark.ml.feature import VectorAssembler

assembler = VectorAssembler(inputCols=[
    'poblacion_total',
    'tasa_de_natalidad',
    'tasa_de_mortalidad'], outputCol='features_assembler')

assembled_data = assembler.transform(df_crecimiento)

assembled_data.select('features_assembler').show(truncate=False)
```

snappify.com

Escalando los datos con Standard Scaler:

```
# Escalamos nuestros datos ya que tienen distintas magnitudes
from pyspark.ml.feature import StandardScaler

# creamos el objeto scaler
scaler = StandardScaler(inputCol="features_assembler", outputCol="features_scaler")

# ajustamos el modelo de escala
scalerModel = scaler.fit(assembled_data)

# transformamos los datos
scaled_data = scalerModel.transform(assembled_data)

scaled_data.show(truncate=False)
```

snappify.com

2. Código de determinación de la k y la generación del modelo.

Para determinar el número óptimo de clusters en el modelo de K-means, se utilizó el método de Silhouette. Este método consiste en calcular una medida de similitud para cada punto dentro de su propio cluster y en relación con los puntos en los clusters vecinos, lo que permite encontrar el valor de K que maximiza esta medida.

```

from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator

silhouette_scores=[]
evaluator = ClusteringEvaluator(featuresCol='features_scaler', \
metricName='silhouette', distanceMeasure='squaredEuclidean')

for K in range(2,11):

    KMeans_=KMeans(featuresCol='features_scaler', k=K)

    KMeans_fit=KMeans_.fit(scaled_data)

    KMeans_transform=KMeans_fit.transform(scaled_data)

    evaluation_score=evaluator.evaluate(KMeans_transform)

    silhouette_scores.append(evaluation_score)

```

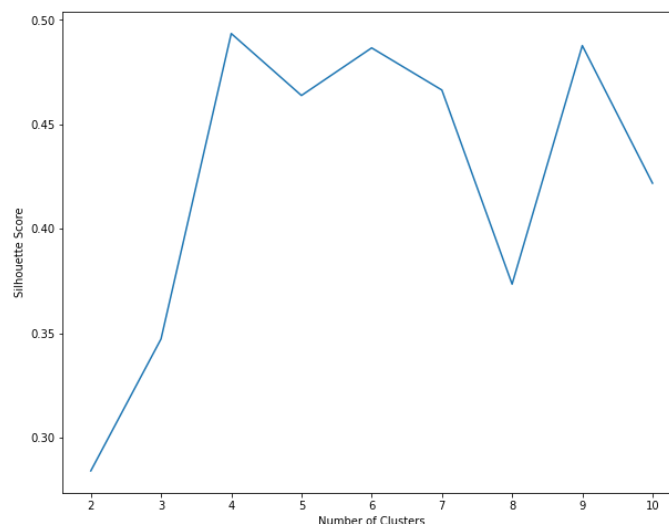
snappify.com

Graficamos para determinar el valor de K:

```

# plot
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1,1, figsize=(10,8))
ax.plot(range(2,11),silhouette_scores)
ax.set_xlabel('Number of Clusters')
ax.set_ylabel('Silhouette Score')

```



Finalmente, se generó el modelo de clustering utilizando K-means, con el número de clusters óptimo encontrado anteriormente que fue 4, ya que muestra el número más alto de Silhouette. Los resultados se interpretaron y describieron para cada uno de los grupos formados por el modelo.

```
# creamos el modelo con k = 3
KMeans_=KMeans(featuresCol='features_scaler', k=4)
KMeans_Model=KMeans_.fit(scaled_data)
KMeans_Assignments=KMeans_Model.transform(scaled_data)
```

snappify.com

Creamos el modelo con k = 3:

```
# creamos el modelo con k = 3
KMeans_=KMeans(featuresCol='features_scaler', k=4)
KMeans_Model=KMeans_.fit(scaled_data)
KMeans_Assignments=KMeans_Model.transform(scaled_data)
```

snappify.com

Usamos PCA para llevar de 3 dimensiones a 2 y ver nuestros resultados más fáciles

```
# in order to visualize the 3-dimensional data into 2, we will
use a dimensionality reduction technique
from pyspark.ml.feature import PCA as PCAml
pca = PCAml(k=2, inputCol="features_scaler", outputCol="pca")
pca_model = pca.fit(scaled_data)
pca_transformed = pca_model.transform(scaled_data)
```

Convertimos el resultado en un arreglo de Numpy, también extraemos el resultado de la predicción:

```
# extraemos los componentes principales
import numpy as np
x_pca = np.array(pca_transformed.rdd.map(lambda row: row.pca).collect())

x_pca

# -1 infiere las dimensiones
cluster_assignment = np.array(KMeans_Assignments.rdd.map(lambda row: row.prediction).collect()).reshape(-1,1)
cluster_assignment
```

snappify.com

Listo todo, tenemos para crear un gráfico de dispersión con Seaborn:

```
import seaborn as sns
import matplotlib.pyplot as plt

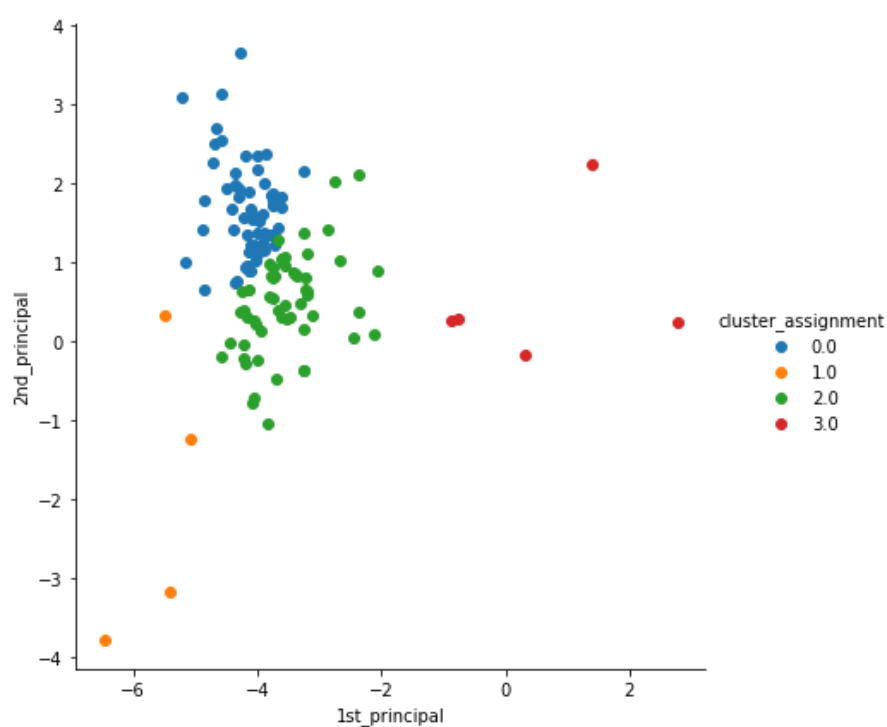
pca_data = np.hstack((x_pca, cluster_assignment))

pca_df = pd.DataFrame(data=pca_data, columns=("1st_principal", "2nd_principal",
                                             "cluster_assignment"))

sns.FacetGrid(pca_df, hue="cluster_assignment", height=6).map(plt.scatter, '1st_principal',
                                                             '2nd_principal').add_legend()

plt.show()
```

snappify.com



3. Descripción de los grupos, acompañada de una discusión de los resultados.

Gracias al uso de PCA logramos llevar nuestros resultados obtenidos del K-means a dos dimensiones y poder analizarlos de una manera más sencilla. Cada cluster agrupa municipios con características similares en cuanto a las variables que se han utilizado para la segmentación.

Anteriormente se había convertido el DF de Spark a un DF de Pandas, para poder usar `.describe()` y obtener un resumen estadístico de las variables y le agregamos al DF de Pandas, la columna de predicción que contiene los datos de los clústeres para etiquetar los municipios.

```
cluster_labels = np.array(cluster_assignment).astype(str).flatten()
pd_df_crecimiento_completo['cluster_label'] = np.core.defchararray.add('Cluster ', cluster_labels)
pd_df_crecimiento_completo.head()
```

cve_municipio	desc_municipio	nacimientos	muerteres	poblacion_total	tasa_crecimiento_natural	tasa_de_mortalidad	tasa_de_natalidad	cluster_label
19	BolaÑños	291	43	7043	35.212267	6.105353	41.317620	Cluster 1
61	Mezquitic	770	115	22083	29.660825	5.207626	34.868451	Cluster 1
125	San Ignacio Cerro Gordo	522	139	18341	20.882177	7.578649	28.460825	Cluster 1
71	San Crist�bal de la Barranca	63	16	2924	16.073871	5.471956	21.545828	Cluster 2
106	Tuxcacuesco	122	35	5482	15.870120	6.384531	22.254652	Cluster 2

Una vez realizado eso obtenemos las estadísticas usando `.describe()`:

```
# Generar estadísticas de cada cluster
cluster_stats = pd_df_crecimiento_completo.groupby('cluster_label').describe(percentiles=[])

# Imprimir la estadística de cada cluster
for cluster in cluster_stats.index:
    print(f"\nEstadísticas del Cluster {cluster}:")
    print(cluster_stats.loc[cluster])
```

```
Estadísticas del Cluster Cluster 0:
cve_municipio      count      58.000000
                   mean      62.206897
                   std       36.098189
                   min        4.000000
                   50%       61.000000
                   max      121.000000
nacimientos        count      58.000000
                   mean     273.517241
                   std     244.532511
                   min      36.000000
                   50%     191.500000
                   max    1204.000000
```

Los cuatro clústeres creados se pueden describir de la siguiente manera:

Cluster 0: Este cluster tiene 58 municipios. Los municipios en este cluster tienen una población promedio de 16,580 habitantes y una tasa de crecimiento natural promedio de 6.47, la tasa de mortalidad promedio es de 10.44 y la tasa de natalidad promedio es de 16.9, cada tasa por cada 1000 habitantes. Los municipios en este cluster tienen un número promedio de 273 nacimientos y 168 muertes.

Cluster 1: Este cluster tiene 4 municipios. Los municipios en este cluster tienen una población promedio de 13,523 habitantes y una tasa de crecimiento natural promedio de 25.40, la tasa de mortalidad promedio es de 7.40 y la tasa de natalidad promedio es de 32.80, cada tasa por cada 1000 habitantes. Los municipios en este cluster tienen un número promedio de 439 nacimientos y 92 muertes.

Cluster 2: Este cluster tiene 58 municipios. Los municipios en este cluster tienen una población promedio de 42,853 habitantes y una tasa de crecimiento natural promedio de 9.51, la tasa de mortalidad promedio es de 6.93 y la tasa de natalidad promedio es de 20.36, cada tasa por cada 1000 habitantes. Los municipios en este cluster tienen un número promedio de 706 nacimientos y 317 muertes.

Cluster 3: Este cluster tiene 5 municipios. Los municipios en este cluster tienen una población promedio de 969,382 habitantes y una tasa de crecimiento natural promedio de 5.95, la tasa de mortalidad promedio es de 6.95 y la tasa de natalidad promedio es de 12.90, cada tasa por cada 1000 habitantes. Los municipios en este cluster tienen un número promedio de 12,200 nacimientos y 7,181 muertes.

La siguiente tabla nos muestra un resumen para hacer una comparación sencilla entre clústeres:

Cluster	Municipios	Habitantes	Natalidad	Mortalidad	Crecimiento natural
0	58	16'580	16.9	10.44	6.47
1	4	13'523	32.8	7.4	25.4
2	58	42'853	20.36	6.93	9.51
3	5	962'382	12.9	6.95	5.95

** Natalidad, Mortalidad y Crecimiento Natural corresponde a un valor por cada 1000 habitantes.*

La tabla nos puede mostrar que el cluster 1 agrupa los 4 municipios más pequeños, y que esas poblaciones tienen la tasa de nacimientos más elevada y muy distante de los demás clústeres. El cluster 0 y 2 tienen la misma cantidad de municipios, pero una densidad poblacional mucho mayor en el cluster 2, a su vez este cluster tiene una tasa de natalidad un poco mayor. Y por último, el cluster 3 agrupa a las 5 ciudades más grandes con una densidad de población muchísimo más alta, también tiene la tasa de natalidad más baja, por lo que se asume que la población poco le interesa tener hijos, puede deberse a que el estilo de vida en esas ciudades es mucho más costoso, la tasa de crecimiento es casi igual a la de mortalidad, lo que puede representar un problema con la mano de obra a futuro.

En este trabajo se aplicó el algoritmo de clustering K-means para enriquecer el perfil socioeconómico de las regiones de Jalisco a partir de variables como la población total, tasa de mortalidad y tasa de natalidad. A través del método de Silhouette se determinó que el número óptimo de clústeres es 4.

El análisis de los resultados permitió identificar cuatro grupos con perfiles socioeconómicos diferenciados. El primer grupo se compone de 58 municipios con una población promedio de 16,580 habitantes y una tasa de crecimiento natural promedio de 6.47. El segundo grupo incluye 4 municipios con una población promedio de 13,523 habitantes y

una tasa de crecimiento natural promedio de 25.40. El tercer grupo está compuesto de 58 municipios con una población promedio de 42,853 habitantes y una tasa de crecimiento natural promedio de 9.51. Finalmente, el cuarto grupo está formado por 5 municipios con una población promedio de 969,382 habitantes y una tasa de crecimiento natural promedio de 5.95.

Los resultados obtenidos pueden ser utilizados para la toma de decisiones en políticas públicas y la asignación de recursos en función de las características socioeconómicas de cada región. Asimismo, estos hallazgos podrían ser útiles para la identificación de patrones y tendencias en el comportamiento demográfico y económico de los municipios de Jalisco.

En conclusión, el uso de técnicas de clustering como K-means puede ser una herramienta valiosa para el análisis de datos y la generación de perfiles socioeconómicos en regiones y territorios, lo que puede contribuir a la toma de decisiones informadas y basadas en evidencia.

Referencias

- Cosio, N. A. L. (2022, 5 enero). *Cómo seleccionar el mejor valor de K en K-means* - Nicolás Arriola Landa Cosio. Medium.
<https://medium.com/@nicolasarriola/c%C3%B3mo-seleccionar-el-mejor-valor-de-k-en-k-means-21121b604365>
- Apache Spark™ - Unified Engine for large-scale data analytics. (s. f.).
<https://spark.apache.org>
- Databricks. (2022, 26 abril). *Contact Us*.
<https://www.databricks.com/>
- Achari, S. (2015). *Hadoop Essentials: Delve Into the Key Concepts of Hadoop and Get a Thorough Understanding of the Hadoop Ecosystem*. Packt Publishing.

Yadav R. (2015) *Spark Cookbook: Over 60 recipes on Spark, covering Spark Core, Spark SQL, Spark Streaming, MLlib, and GraphX libraries*. Packt Publishing.