Reporte Kmeans - KNN **Lenin Falconí**
Prof. Henry Paz Lab Date: 17/12/18

1. Dataset

Se dispone de un dataset de clientes de un Supermercado. La información disponible en el mismo es: género, edad, ingreso anual, y un score de la tienda. El dataset se puede descargar de:
https://www.kaggle.com/vjchoudhary7/customer-segmentation-tutorial-in-python#Mall_Customers.csv

2. Cluster = 2

En esta sección analizamos la clasificación de la data con KNN para un el grupo óptimo de elbow que es K = 2. Sin embargo, en *marketing* se recomienda grupos de **4** o **5**

2.1. Tramiento del Dataset y programas

El género se sustituye por números: $1 -> $ male $2 -> $ female. El resto de datos se mantiene igual. No fue necesario escalar o normalizar la data debido a que los datos se mentienen con valores homogéneos o similares y no existen datos con valores muy mayores o muy menores respecto a la mayoría.

El código desarrollado para este ejemplo se puede obtener de mi repositorio GIT: https://github.com/LeninGF/KmeansKnn

El script kmeans.py permite obtener la clusterización de la data. Genera las etiquetas, las mismas que se salvan en un archivo de csv denominado *label.csv*. Con estas etiquetas se conformará el dataset de entrenamiento. Este script también genera los datos artificiales para la etapa de prueba.

2.2. Etiquetado de Datos

Se utilizan como etiquetas los grupos generados por Kmeans utilizando dos centros –que fue lo que se concluyó del análisis de elbow realizado anteriormente–. Esto permite asignar a clase 0 y clase 1 cada ejemplo

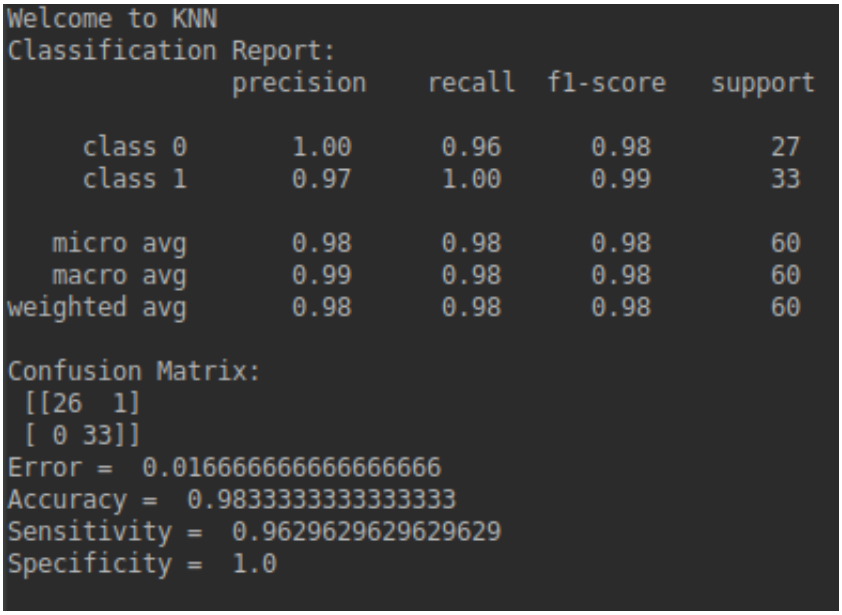

Figura 1: Resultados KNN

según Kmeans. El archivo que contiene esta data es el archivo textit-Mall_Customers_Class1.csv que es el archivo de entrenamiento con 200 ejemplos.

2.3. Entrenamiento KNN

El modelo de KNN se entrena con los 200 ejemplos según la etiqueta antes obtenida de Kmeans.

2.4. Testeo

Para evaluar el rendimiento de la clasificación de KNN se utilizó como métricas la matriz de confusión y otros valores como la precisión y el recall sobre un dataset de 60 datos generados artificialmente. Las etiquetas se obtuvieron utilizando la función de predicción del kmeans. Estos datos están en los archivos *xtest.csv, ytest.csv*

En la figura 1 se puede observar el resultado de clasificación de KNN en 2 clases '0' y '1'. El modelo obtenido con KNN tiene un accuracy de 98 % con un error del 1.6 %. Por tanto el modelo es capaz de predecir nuevas clases según los datos que ingresen. Esto también se confirma al disponer de una diagonal balanceada en la matriz de confusión.

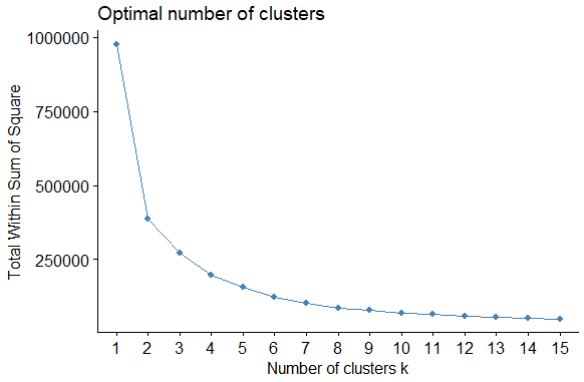

Figura 2: Método de Elbow

3. Cluster = 4

Analizando más de cerca la gráfica de elbow (figura 2) y el dendograma (figura 3) para la data utilizada en este ejemplo con 4 centros, se observa que una buena opción de clusterización existe con hasta 4 grupos. Porque la distancia indicada en elbow entre 4 y 5 ya no es significativa. En la figura 4 se observa la clusterización para 4 grupos según kmeans. Apesar de existir intersecciones entre algunas clases, se nota que aparece un grupo muy específico que no se intersecta con ninguno de los demás grupos.

3.1. Tratamiento del Dataset y programas

El tratamiento de los datos es similar a la etapa anterior. En este caso, las etiquetas generadas por el programa kmeans.py se encuentra en *labels4.csv*, el archivo de entrenamiento se nombra *Mall_Customers_Class4.csv* y los de prueba *xtest4.csv* y *ytest4.csv*. El código para este ejemplo se encuentra en la rama de **cluster4** del repositorio.

3.2. Etiquetado de Datos

El etiquetado de datos se realiza con números de 0 al 3. Recordemos que un Kmeans nunca puede tener variables dependientes. Si existen variables dependientes sencillamente el Kmeans no va a operar adecuadamente. Las cuatro clases que obtenemos pueden utilizarse para categorizar los clientes como VIP, AAA, AA, A que pueden ser de interés para el mall estimular la compra en dichos grupos o analizar nuevos clientes.

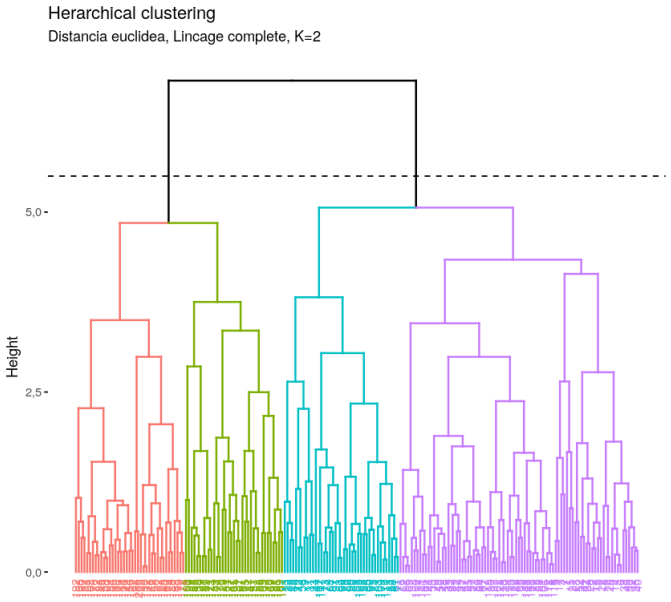

Figura 3: Dendograma k = 4

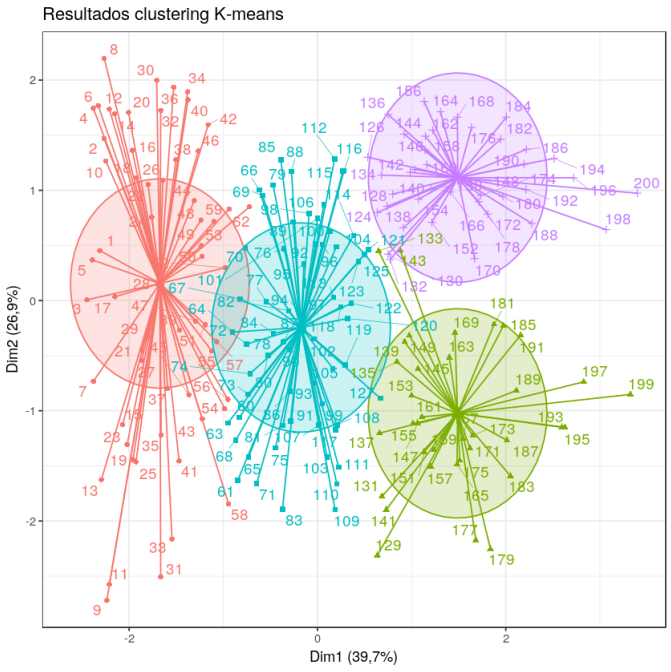

Figura 4: Kmeans 4

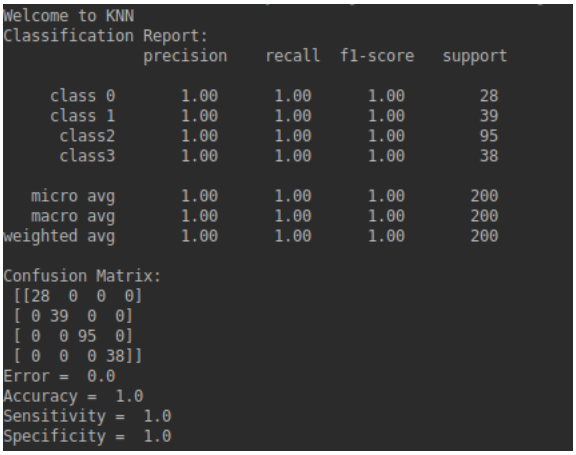

Figura 5: Training Error

3.3. Entrenamiento KNN

El modelo KNN se entrena con 200 ejemplos y con las etiquetas obtenidas. Los datos no fueron escalados y normalizados. Sin embargo, cuando existen datos que divergen mucho (es decir, observe los valores mínimos y máximos de la data) es recomendable escalar, normalizar, y finalmente si estas técnicas no reducen el ruido, utilizar logaritmos a la data. En este caso no fue necesario.

3.4. Entrenamiento y Testeo

En este caso se utiliza 60 datos sintéticos que equivale a un testeo con 20 % de datos. Lo que nos daría un 80 % de entrenamiento. En la figura 5 se puede observar el resultado de entrenamiento. En cambio en la figura 6 se puede observar el resultado de testeo.

De los resultados obtenidos se observa que contrariamente al caso con K = 2, en el caso con K = 4, la clasificación llega al 100 % de accuracy con error del 0 %. En el caso K = 2, se obtuvo 98 % en accuracy y un error de 0.016.

Al analizar los resultados de testeo en comparación con los de entrenamiento, se observa que el modelo no presenta overfitting. Sin embargo, el hecho de haber alcanzado con cluster 4 un error de 0 % en clasificación implica que hace falta más data o más variables para una consistencia del modelo y evitar una posible falta de generalización del aprendizaje.

```
Classification Report:
              precision    recall  f1-score   support

     class 0       1.00      1.00      1.00         5
     class 1       1.00      0.92      0.96        24
      class2       0.91      0.95      0.93        21
      class3       0.91      1.00      0.95        10

   micro avg       0.95      0.95      0.95        60
   macro avg       0.95      0.97      0.96        60
weighted avg       0.95      0.95      0.95        60

Confusion Matrix:
 [[ 5  0  0  0]
 [ 0 22  2  0]
 [ 0  0 20  1]
 [ 0  0  0 10]]
Error =  0.0
Accuracy =  1.0
Sensitivity =  1.0
Specificity =  1.0
```

Figura 6: Testing Error

4. Conclusiones

- La clasificación con KNN en 2 grupos confirma que los datos originales clusterizados con Kmeans se pueden separar en 2 grupos diferentes. Sin embargo hay datos distantes de los centroides.
- La clusterización con 4 grupos mejora los resultados de clasificación en el KNN reduciendo el error a 0 comparado coin la clusterización en 2 grupos.
- La clusterización en 4 grupos es más significativa en términos de marketing ya que estos pueden ser clientes vip, aaa, aa, a.
- Al comparar los resultados de training y testing se observa que no existe overfitting en el modelo. No obstante, el tener un error de 0 % en testing y training indica que necesitamos mas datos para mejorar el modelo.