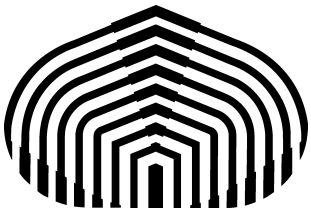

UNIVERSIDAD SIMÓN BOLÍVAR
INGENIERÍA DE LA COMPUTACIÓN
DEPT.COMPUTACIÓN Y TECNOLOGÍA DE LA INFORMACIÓN
CI5438: INTELIGENCIA ARTIFICIAL 2

Proyecto Clustering

PROFESOR:

Ivette Carolina Martinez

ALUMNOS:

Miguel C. Canedo R.	13-10214
José D. Bracuto D.	13-10173
Rafael A. Cisneros C.	13-11156

Índice

1. Resumen **3**

2. Detalles de implementación **4**
2.1. Ejecución 4
2.2. Algoritmo de k-means 4

3. Presentación y Discusión de Resultados **6**
3.1. Explorando los datos de Iris Data Set 6
3.1.1. Usando de 2 a 5 clusters 6
3.1.2. Diferencia entre clusters y etiquetas 10
3.2. Comprimiendo una imagen 11

4. Conclusiones **13**

5. Referencias **14**

Resumen

Para este trabajo se desarrollara el algoritmo de clustering para ser probado usando dos dataset.

Para la primera prueba se usara el iris dataset que contiene tres tipos de pétalos (iris-setosa, iris-virginica, iris-versicolor) en esta prueba se correrá el algoritmo con 2,3,4 y 5 clusters, dándose a evidenciar que la inicialización de los centros es muy importante puesto a que los tipos de datos que están relativamente cerca, serán agrupados de distintas formas en varios clusters, mientras que los que están bien diferenciados, en este caso las iris-setosas, siempre sarán separadas en clusters aislados de los demás.

Luego probaremos a leer una imagen y agrupando cada pixel en 2,4,8,16,32,64 y 128 clusters respectivamente. En esta podemos ver como las tonalidades se separan en claras y oscuras, luego se van separando entre si a medida que se aumenta el numero de clusters. Por ende se puede decir que siempre va a haber una separación de clusters diferenciable sin importar que color se este usando en la imagen y que obviamente al aumentar el numero de clusters, la imagen resultando va a tener mejor calidad.

Detalles de implementación

Para la realización del proyecto se uso Python como lenguaje de programación puesto a que ya se tiene experiencia con el mismo, además de poseer tres librerías que destacan al momento de manejar los datasets, estas son:

- **Pandas:** nos ayuda con la lectura y manejo de los datos para poder manipularlos a conveniencia y poder analizarlos.
- **Numpy:** librería que facilita el manejo de matrices, además de contar con varias funciones matemáticas para las operaciones entre arreglos y cálculos necesarios.
- **Matplotlib:** se usa para crear los gráficos de los datos.
- **Sklearn:** Usada para separar los datos.

Ejecución

Para ejecutar los ejercicios se dividieron en un script principal que contiene la definición del algoritmo de k-means, **kmenas.py**). Así mismo, se incluyen 2 scripts para la ejecución de los distintos ejercicios:

- **Ejercicio 2:** para ejecutarlo se tendrá que correr el siguiente comando:

```
$ python ejercicio2.py
```

- **Ejercicio 3:** para ejecutarlo se tendrá que correr el siguiente comando:

```
$ python ejercicio3.py
```

Algoritmo de k-means

Se hizo un script aparte donde definir el algoritmo de k-means, en este script se definen 4 funciones:

- **calcularDistancia:** esta función calcula al distancia euclidiana de los datos que recibe como parámetro.

```
def calcularDistancia(punto1, punto2):
    distancia = 0
    for i in range(len(punto1)):
        distancia += (punto1[i] - punto2[i])**2
```

```
5     return  sqrt(distancia)
```

- **calcularCluster:** De a acuerdo a un punto y los centros, devuelve el cluster al cual el punto tiene menos distancia de separación.
- **calcularCentro:**

  Se calcula el centre con respecto a los puntos dados, si el esta vacío se devuelve el mismo centro al que pertenece
- **Kmeans:**

  Esta función recibe como parámetros la data que se procesará y el número de clusters en que se separará.

  Primero se inician los centros aleatoriamente del conjunto de datos.

```
centros = [ np.ndarray.tolist(data.iloc[ randint(0,len(data)-1) ].
    values) for i in range(k) ]
```

  Luego se itera hasta que los centros no cambien.

  En estas iteraciones se calculan a que cluster pertenece cada dato.

```
for dato in range(len(data)):
    cluster = calcularCluster(data.iloc[dato], centros)
    clusters[cluster].append(data.iloc[dato])
```

  Y luego se recalculan los centros para ver si estos cambiaron o no.

```
nuevosCentros = [ calcularCentro(clusters[cluster], centros[cluster])
    for cluster in range(len(clusters)) ]
```

Presentación y Discusión de Resultados

Explorando los datos de Iris Data Set

Usando de 2 a 5 clusters

- **2 clusters** Se puede ver que con dos clusters, en cualquiera de las corridas, se logra clasificar un tipo de pétalo en un cluster mas una diferencia de 3 del tipo versicolor, aunque se trata de separar en dos conjuntos, el algoritmo en vez de separarlos equitativamente, separa las iris-setosas y tres versicolor como un conjunto y las demás iris-versicolor juntos con las virginicas como otro conjunto

Valores de la corrida 1

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	47	50
Cluster 2	50	3	0

Valores de la corrida 2

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	47	50
Cluster 2	50	3	0

Valores de la corrida 3

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	47	50
Cluster 2	50	3	0

Valores de la corrida 4

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	50	3	0
Cluster 2	0	47	50

Valores de la corrida 5

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	50	3	0
Cluster 2	0	47	50

- **3 clusters**

  En este caso podemos ver como las iris-setosas o se agrupan en un solo cluster o se separa mitad en un cluster y la otra mitad junto con 3 iris-versicolor las iris faltantes siempre se agrupan en un cluster junto con las virginicas

Valores de la corrida 1

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	46	50
Cluster 2	20	4	0
Cluster 3	30	0	0

Valores de la corrida 2

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	30	0	0
Cluster 2	20	4	0
Cluster 3	0	46	50

Valores de la corrida 3

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	2	36
Cluster 2	50	0	0
Cluster 3	0	48	14

Valores de la corrida 4

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	28	3	0
Cluster 2	0	47	50
Cluster 3	22	0	0

Valores de la corrida 5

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	48	14
Cluster 2	50	0	0
Cluster 3	0	2	36

- **4 clusters**

  A partir de 4 clusters podemos ver se deja de agrupar a las iris-setosas junto con alguna otra iris

Valores de la corrida 1

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	47	14
Cluster 2	23	0	0
Cluster 3	27	0	0
Cluster 4	0	3	36

Valores de la corrida 2

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	33	0	0
Cluster 2	0	3	36
Cluster 3	17	0	0
Cluster 4	0	47	14

Valores de la corrida 3

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	0	23
Cluster 2	50	0	0
Cluster 3	0	29	1
Cluster 4	0	21	26

Valores de la corrida 4

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	47	14
Cluster 2	0	3	36
Cluster 3	27	0	0
Cluster 4	23	0	0

Valores de la corrida 5

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	23	17
Cluster 2	0	0	32
Cluster 3	0	27	1
Cluster 4	50	0	0

- **5 clusters**

  Con 5 clusters vemos como una parte de las iris-virginicas se empieza a agrupar solo pero siguen agrupándose a igual cantidad con las iris-versicolor, por lo que podemos sospechar que tanto las iris-versicolor como las iris-virginicas son muy parecidas

Valores de la corrida 1

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	10	0	0
Cluster 2	0	47	14
Cluster 3	20	0	0
Cluster 4	0	3	36
Cluster 5	20	0	0

Valores de la corrida 2

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	2	27
Cluster 2	0	27	1
Cluster 3	0	21	0
Cluster 4	0	0	22
Cluster 5	50	0	0

Valores de la corrida 3

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	23	13
Cluster 2	0	27	1
Cluster 3	0	0	24
Cluster 4	50	0	0
Cluster 5	0	0	12

Valores de la corrida 4

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	17	0	0
Cluster 2	0	27	1
Cluster 3	0	0	32
Cluster 4	33	0	0
Cluster 5	0	23	17

Valores de la corrida 5

Cluster	Iris-setosa	Iris-versicolor	Iris-virginica
Cluster 1	0	21	26
Cluster 2	26	0	0
Cluster 3	0	29	1
Cluster 4	24	0	0
Cluster 5	0	0	23

Diferencia entre clusters y etiquetas

- **Para 2 clusters** Setosa y Virginica siempre se separan perfectamente en un único cluster Versicolor queda casi perfectamente separada en el cluster de virginica, mezclando unos poquitos con setosa. Con esto podemos deducir que tanto las iris-setosas como las iris-virginicas son muy distintas, tanto así que no existe posibilidad de juntar alguna de esas dos en el mismo cluster y por ende el algoritmo siempre las separara en clusters distintos sin importar la inicialización de los mismos.

- **Para 3 clusters**

Setosa siempre perfectamente separada de los otros tipos de iris, bien sea en uno o dos clusters, Versicolor y virginica mezclados siempre. Aquí se puede concluir que las iris-versicolor y las iris-virginicas son muy parecidas puesto a que no importa cuantas veces se corra el algoritmo, siempre va a haber clusters que contengan una combinación de iris-virginicas y versicolor, mientras que las iris-setosas sin se diferencian fácilmente de las demas puesto a que con 3 clusters ya no las agrupa con ningún otro tipo de pétalo.

Comprimiendo una imagen

En este caso podemos ver perfectamente como funciona el algoritmo de clustering, para esto se escogió una foto de gohan en la cual posee tonalidades de colores ligeros. Al correr el algoritmo con 2 clusters vemos como se diferencian los colores claros de los oscuros (se juntan en un cluster la camisa y el fondo, mientras que lo el cuerpo queda en el otro cluster).

La imagen en escénica sigue siendo la misma (solo se va delimitando cada objeto), hasta que llega a 16 clusters, en esta se puede ver como se van separando los colores oscuros entre si puesto a que ya se puede diferenciar la camisa del fondo.

Y a partir de 128 clusters vemos que la imagen ya separa los tonalidades claras entre si, es decir, cada detalle sin importar la tonalidad clara u oscura se van separando hasta poderse diferenciar cada una. Con esto podemos ver que al aumentar el numero de clusters aumenta la calidad de la imagen

Imagen comprimida con distintos clusters

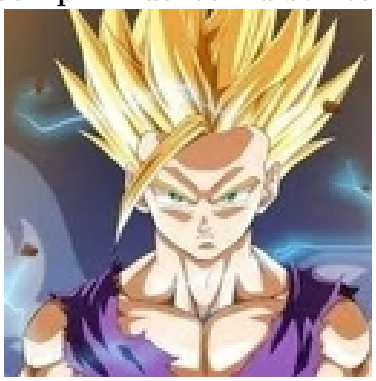

Imagen Original

2 Clusters

4 Clusters

8 Clusters

16 Clusters

32 Clusters

64 Clusters

128 Clusters

Conclusiones

Después de haber hecho las pruebas con el iris data set podemos decir que en el algoritmo de clustering es muy importante la inicialización de los centros puesto a que dependiendo de como se inicialicen, los datos que no tienen tanta diferencia entre si se van a ir agrupando en varios clusters combinandose entre si aunque sean de tipos distintos, como seria el caso de las iris-virginicas y las iris-versicolor. Mientras que si los datos suministrados son fácilmente diferenciables, no importa la forma en que se inicialicen los centros, esto se ve en el caso de las iris-setosas que sin importar la inicialización de los centros, siempre se agrupaba en clusters aislados.

Con el caso de las imágenes podemos concluir que mientras se trabaje con mas clusters, mas detallado serán las separaciones y que sin importar que tan diferente sean las tonalidades de la imagen, siempre habrá una separación de tonalidades calaras y oscuras que poco a poco se van separando entre si a medida de que se aumentan los clusters

Referencias

- **Tipos de Normalización:** http://es.coursera.org/lecture/data-analysis-with-python/data-normalization-in-python-pqNBS.
- **Documentación de la libreria Pandas:** http://pandas.pydata.org/pandas-docs/stable/reference.
- Laminas de las clases.