INF391 Reconocimiento de Patrones en Minería de Datos
Tarea 1

Universidad Técnica Federico Santa María, Campus San Joaquín
Departamento de Informática

11 de mayo de 2015

Profesor Marcelo Mendoza

Juan Pablo Escalona
juan.escalonag@alumnos.usm.cl
201073515-k

Rafik Masad
rafik.masad@alumnos.usm.cl
201073519-2

Gianfranco Valentino
gianfranco.valentino@alumnos.usm.cl
2860574-9

Introducción

En el presente informe se analizan los resultados obtenidos comparando diferentes técnicas de clustering sobre el dataset iris incluido en la librería *sklearn*.

Análisis de los resultados obtenidos

Se analizaron una gran diversidad de algoritmos, pero hay ciertos parámetros que son comunes entre ellos. Haciendo el símil con la teoría, tenemos que *n_clusters* corresponde a K, para ciertos algoritmos que esperan dicho input. También existen diversas condiciones de termino, estas pueden considerarse como *tol*, o tolerancia, es decir, cuanto es el máximo numero de cambios de puntos clasificados entre iteraciones para terminar el algoritmo. Por ejemplo, 0.1 significa que si entre dos iteraciones consecutivas, el numero de puntos que cambiaron de cluster, es menor o igual que el 10 % del total de datos, se termina el algoritmo. Por otro lado, también se limita el número máximo de iteraciones, ya que si bien, los métodos convergen a máximos locales, dependiendo del dataset y las semillas, podría tomar demasiadas iteraciones, aun cuando la solución es aceptable.
Para definir una medida común de errores, se intenta ver los labels originales, o bien, los que vienen con el dataset. Luego se comparan los grupos formados, con los esperados y se cuentan los puntos clasificados incorrectamente.
Ahora bien, lo último a tener en cuenta, es que los algoritmos que dependen de semillas, tienen resultados muy distintos dependiendo de estas, ya que al converger a máximos locales, el punto inicial es determinante, es por ello, que se realizan multiples ejecuciones, y se mantiene el mejor resultado.

1.1. K-means

Algoritmo: K-means

Parámetros utilizados:

- n_clusters: 3
- tol: 0.1
- max_iter: 300
- n_jobs: 1

Resultados

- Errores: 14
- Accuracy: 90.7 %
- Tiempo de ejecución: 0.0123851 [s]

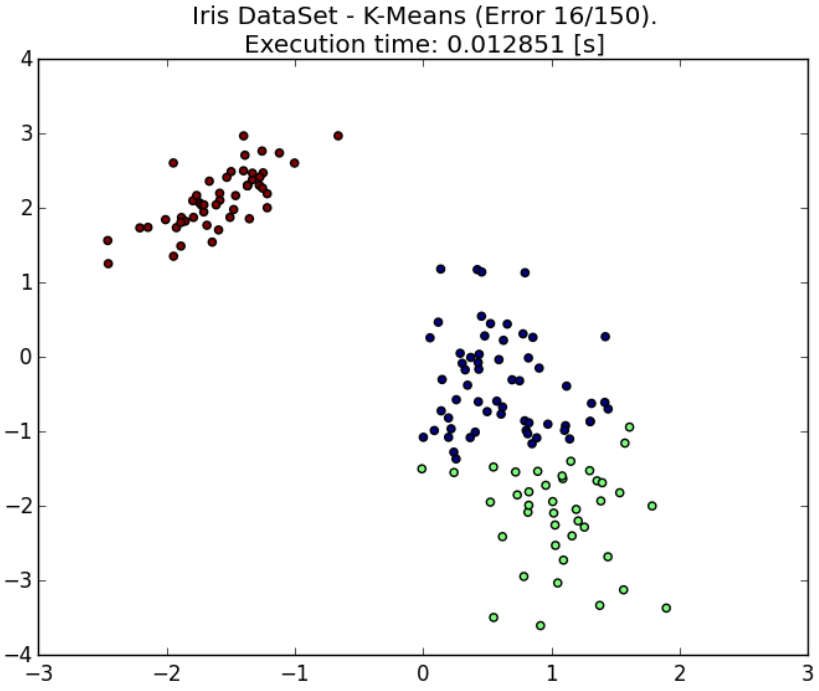

Figura 1: K-means++

1.2. Minibatch K-means

Algoritmo: Minibatch K-means

Parámetros utilizados:

- n_clusters: 3
- reassignment_ratio: 0.01
- max_iter: 100
- batch_size: 5
- init: 'k-means++'
- n_init: 3
- tol: 0.5

Resultados

- Errores: 15
- Accuracy: 90 %
- Tiempo de ejecución: 0.016015 [s]

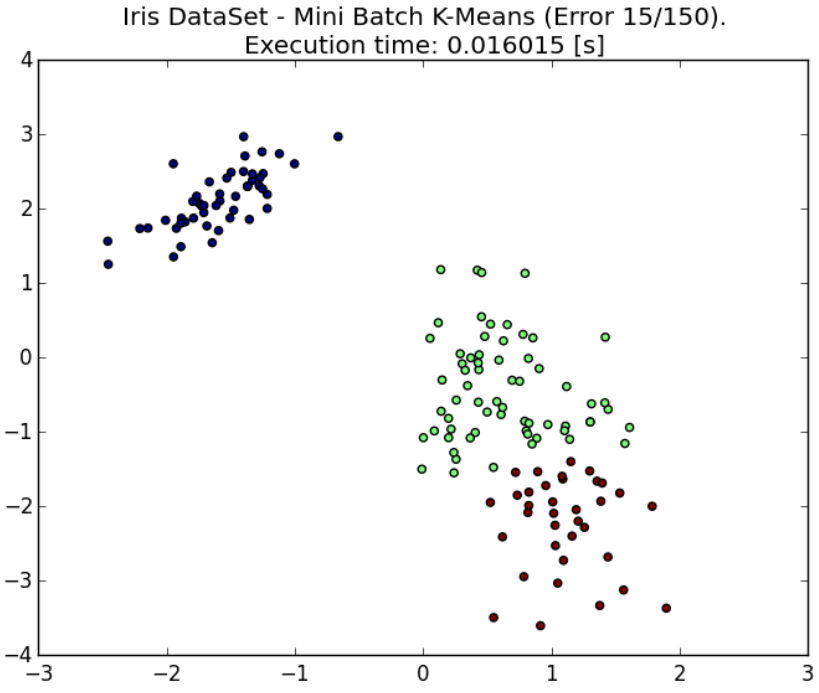

Figura 2: Minibatch k-means

En general k-means, al converger a un máximo local, es extremadamente dependiente de las semillas, explicando así las diferencias en la calidad de los resultados por ejecución.

1.3. HAC

Algoritmo: HAC

Parámetros utilizados:

- n_clusters: 3
- affinity: euclidean
- n_components: None
- linkage: average

Resultados

- Errores: 16
- Accuracy: 89.3 %
- Tiempo de ejecución: 0.002657 [s]

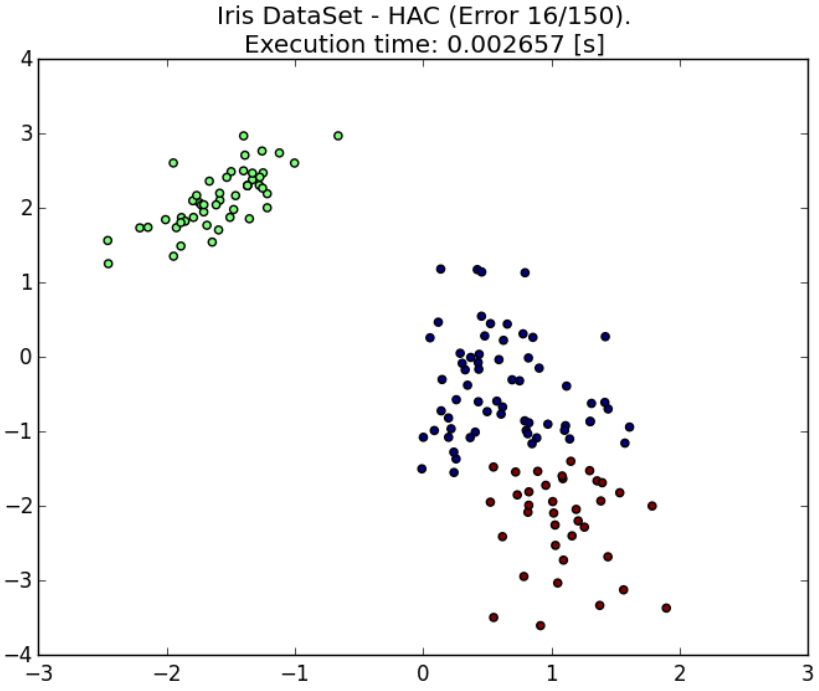

Figura 3: HAC

1.4. Ward

Algoritmo: Ward

Parámetros utilizados:

- n_clusters: 3

Resultados

- Errores: 16
- Accuracy: 89.3 %
- Tiempo de ejecución: 0.011104 [s]

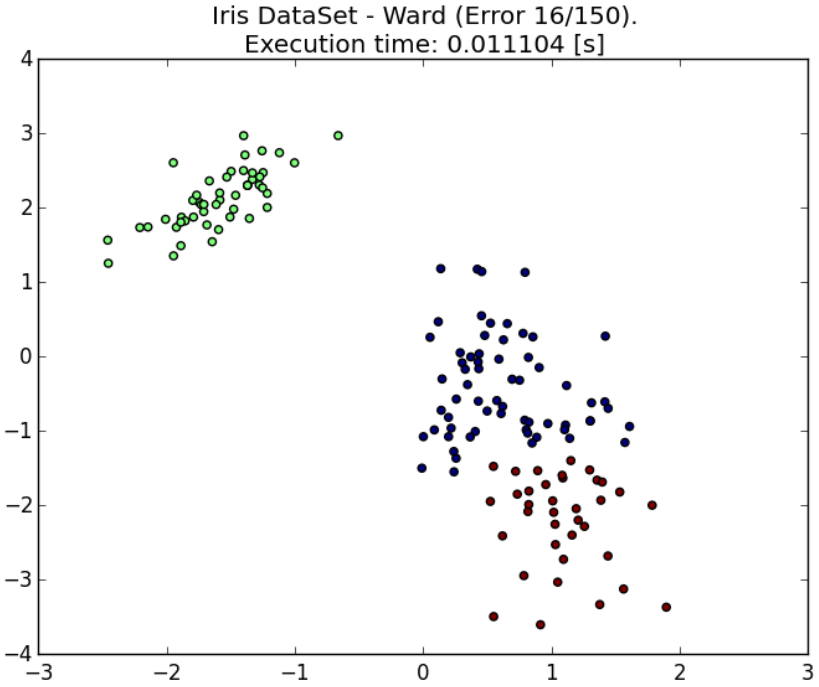

Figura 4: Ward

1.5. DBScan

Algoritmo: DBScan

Parámetros utilizados:

- min_samples: 14
- eps: 0.5

Resultados

- Errores: 22
- Accuracy: 85.3 %
- Tiempo de ejecución: 0.031344 [s]

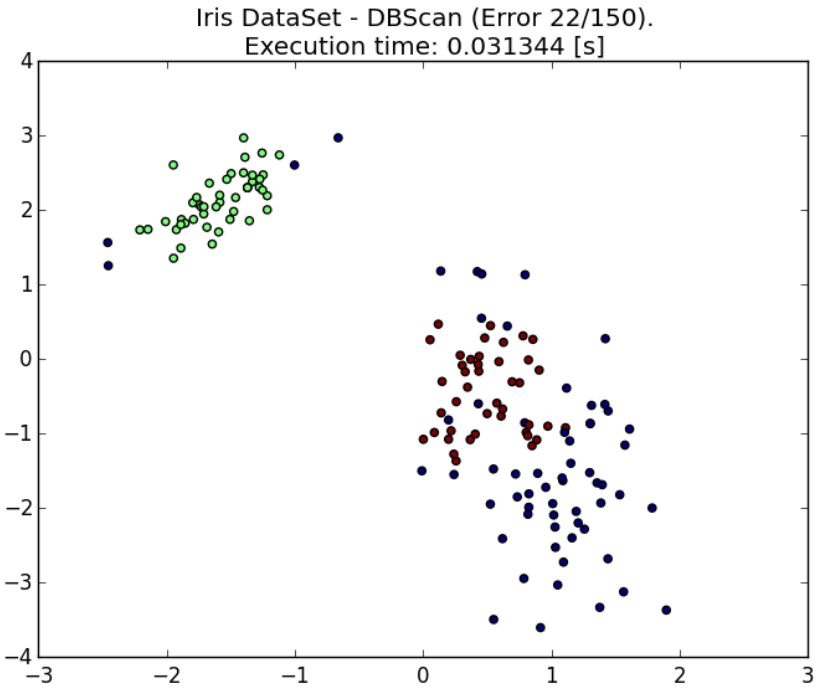

Figura 5: DBScan

1.6. C-Means

Algoritmo: C-Means

Parámetros utilizados:

- c: 3
- m: 0.01
- error: 0.3
- maxiter: 20
- seed: None

Resultados

- Errores: 10-50[1]
- Accuracy: 60-90 %
- Tiempo de ejecución: 0.017491 [s]

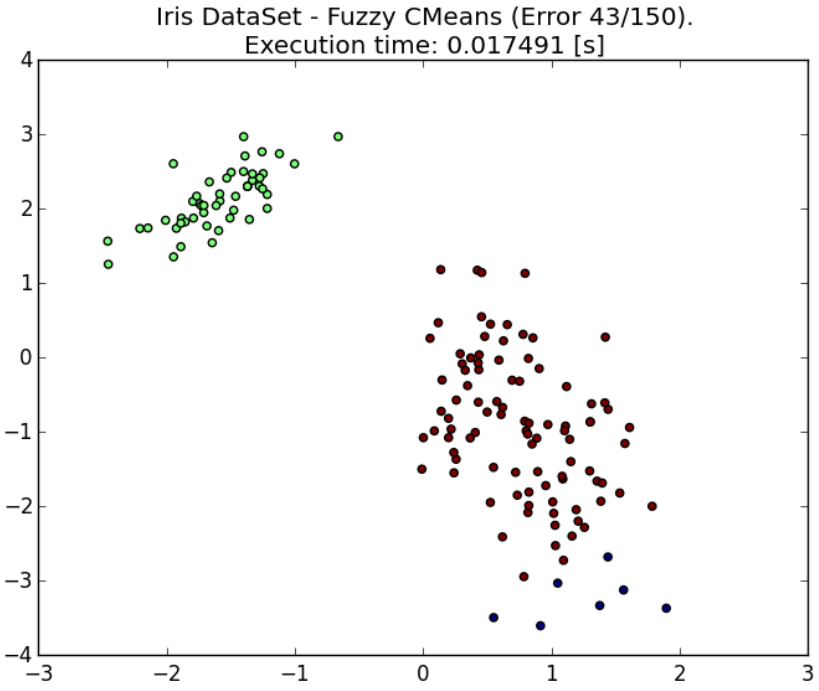

Figura 6: C-Means

[1] Muy variable

1.7. Mean shift

Algoritmo: Mean shift

Parámetros utilizados:

- bandwidth: 0.9
- bin_seeding: True
- min_bin_freq: 1
- cluster_all: False

Resultados

- Errores: 33
- Accuracy: 78 %
- Tiempo de ejecución: 0.080781 [s]

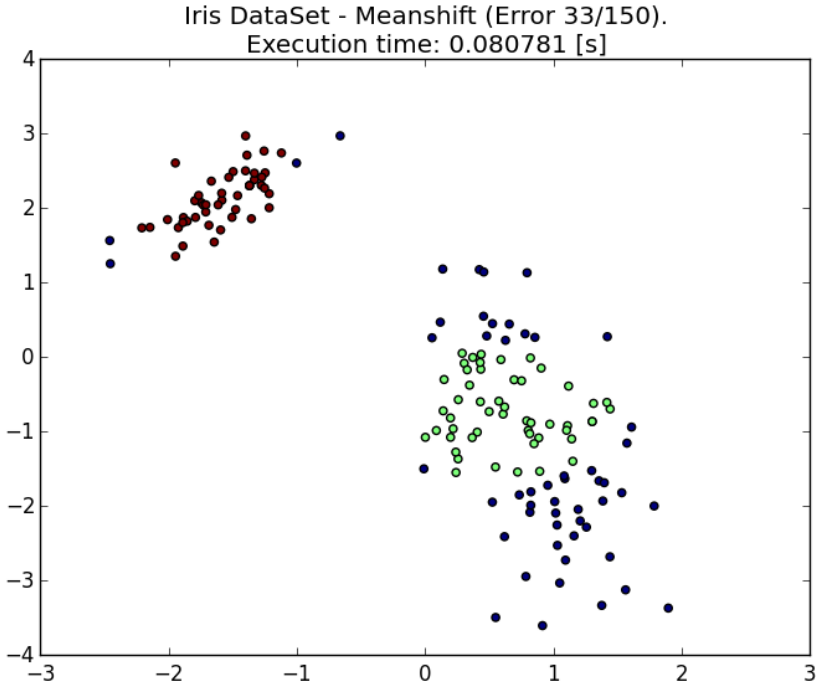

Figura 7: Mean shift

1.8. Spectral Clustering

Algoritmo: Spectral Clustering

Parámetros utilizados:

- n_clusters: 3
- n_components: 3
- eigen_solver: 'arpack'
- assign_labels: 'discretize'
- n_init: 1
- weight: 9.5

Resultados

- Errores: 6
- Accuracy: 96 %
- Tiempo de ejecución: 0.064943 [s]

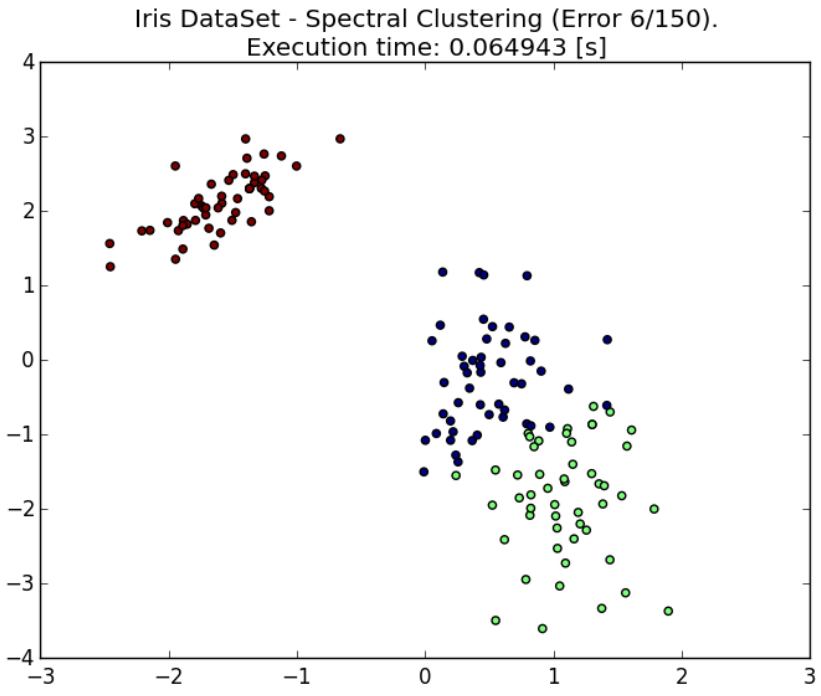

Figura 8: Spectral Clustering

2 Tabla comparativa

En la siguiente tabla se resumen los errores y tiempos de cada algoritmo

Algoritmo	Errores	Tiempo [s]
Spectral Clustering	6	0.064943
k-means	14	0.012385
Minibatch k-means	15	0.016015
HAC	16	0.002657
Ward	16	0.011104
DBScan	22	0.031344
Mean shift	33	0.080781
C-Means	10-50	0.017491

Conclusiones

En conclusion, podemos observar que el mejor resultado fue Spectral Clustering, y que los algoritmos que utilizan semillas, son muy dependientes de estas, por lo que se aplican técnicas como ejecuciones simultáneas con distintas semillas. Además, hay que recalcar el trade-off entre calidad y tiempo, si bien Spectral Clustering, tuvo el mejor resultado, comparando con el resto de los algoritmos, en general tiende a demorarse entre 3 a 6 veces más aproximadamente. Y debemos tener en cuenta, que este dataset es de juguete, y que el comportamiento no necesariamente es el mismo al aumentar el número de K, y el número de datos, por lo que la técnica a utilizar debe considerar el comportamiento de los algoritmos frente a las distintas entradas, por lo que no habría un algoritmo único a utilizar frente a todos los casos.