Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación

Máquinas de aprendizaje

Reporte: Ejemplo con k-means

BUAP

Docente: Abraham Sánchez López

Alumno	Matrícula
Taisen Romero Bañuelos	202055209

Ejemplo con k-means

Bien, en la parte del ejercicio 2 me extenderé debido a que haré varias pruebas. Pero hablando de lo que se discutió en el PDF me quedo con que nstart es más importante que iter.max porque le da al algoritmo más opciones para buscar un mejor centroide, aunque de cierta forma también es lo que trata de hacer iter.max, yo me lo imagino como que nstart abre más nodos iniciales en la búsqueda de una solución óptima e iter.max lo que hace es buscar reacomodar un poco los centroides sugeridos por nstart (algo así como si hiciera una búsqueda en anchura de cada uno de los nodos iniciales, por así decirlo).

También, podemos hacer una "competencia" de los algoritmos de agrupación incluidos en mlr. En palabras simples, el algoritmo MacQueen fue el más rápido porque actualiza los centroides más rápido (siguiendo la analogía de los árboles de búsqueda supongo que sería equivalente a decir que no desarrolla al 100% la búsqueda de cada centroide). Por lo tanto, es una buena opción a considerar con datos grandes ya que la pequeña mejora que ofrecen los otros algoritmos no siempre compensa el costo computacional.

Ejercicio 2

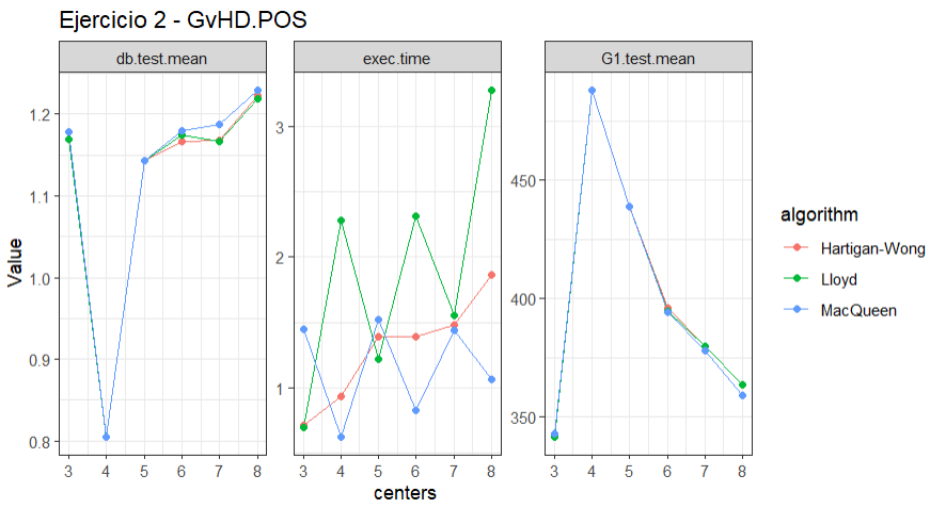
Como mencioné anteriormente, se argumenta que nstart es más importante que iter.max, por lo que decidí empezar mis pruebas aumentando sólo nstart (de 10 a 15) para ir comparando resultados.

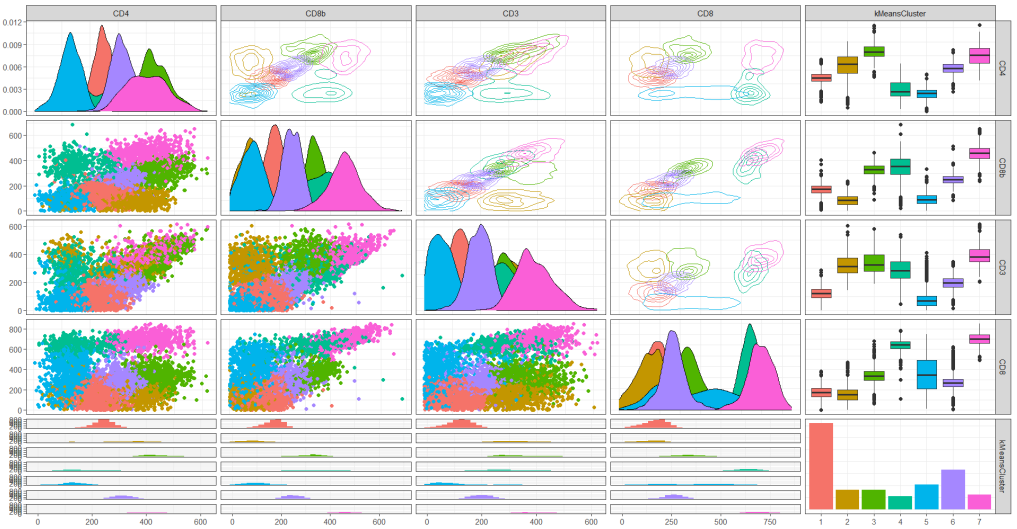
Como adelanto puedo decir que en los diagramas de distribución CD3 fue la faceta más estable. Sufrió cambios en los tres experimentos pero fue el que tuvo cambios menos radicales. Esto sugiere que los 3 modelos diferentes identifican de manera similar ese grupo de datos.

Nota: newCell tiene los mismos datos que se usaron para el ejercicio 1.

Modelo 1 - iter.max=100 y **nstart=10**:

```
[Tune] Result: centers=7; algorithm=Lloyd : db.test.mean=1.0442276,G1.test.mean=497.8034184
Hubo 48 avisos (use warnings() para verlos)
```



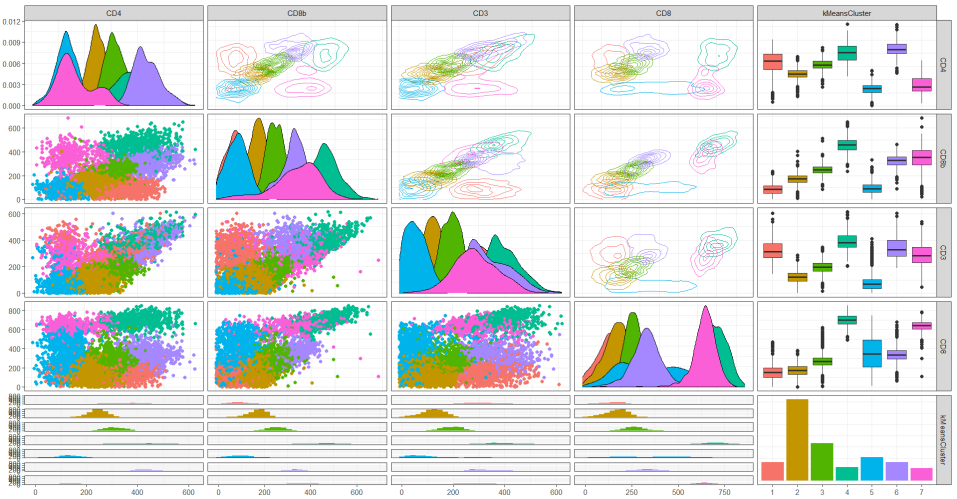


Sobre la predicción de la nueva célula

```
> suppressWarnings(predict(tunedKMeansModel_2, newdata = newCell))
Prediction: 1 observations
predict.type: response
threshold:
time: 0.03
  response
1        2
```

Modelo 2 - iter.max=100 y **nstart=15** (Omití uno de los ploteos porque los resultados eran exactamente iguales a los del ejemplo anterior):

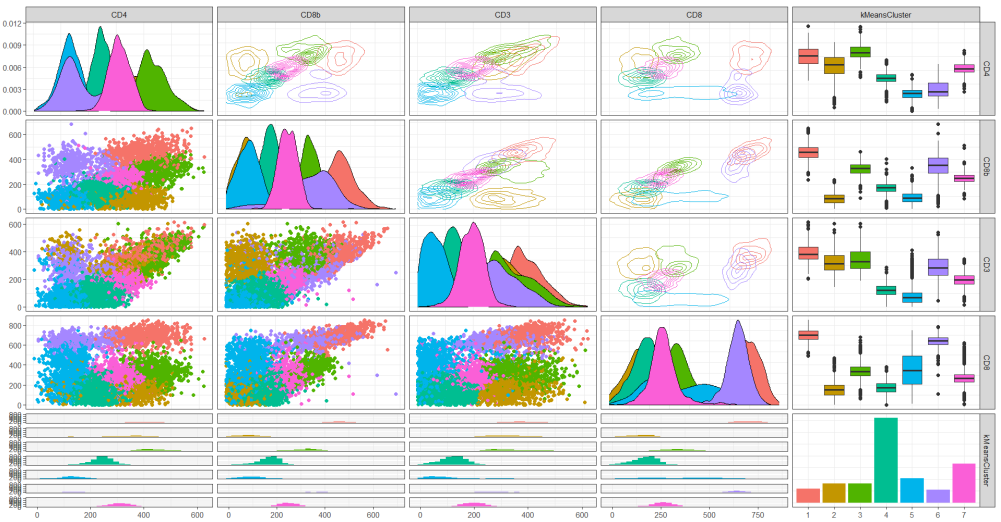
```
[Tune] Result: centers=7; algorithm=Lloyd : db.test.mean=1.0524140,G1.test.mean=493.4914750
Hubo 50 o más avisos (use warnings() para ver los primeros 50)
```




```
> suppressWarnings(predict(tunedKMeansModel_2, newdata = newCell))
Prediction: 1 observations
predict.type: response
threshold:
time: 0.02
  response
1        4
```

Modelo 3 - iter.max=200 y nstart=15:

```
[Tune] Result: centers=7; algorithm=Lloyd : db.test.mean=1.0517085,G1.test.mean=493.7393906
Hubo 14 avisos (use warnings() para verlos)
```




```
> suppressWarnings(predict(tunedKMeansModel_2, newdata = newCell))
Prediction: 1 observations
predict.type: response
threshold:
time: 0.03
  response
1        7
```

Empecemos hablando de las diferencias en las predicciones del modelo 2 y 3. El modelo 2 predijo que la nueva célula pertenecería al grupo 4, mientras que el tercer modelo predijo que pertenecería al grupo 7. Recordemos que el modelo 2 y 3 difieren en el número de iteraciones máximas (100 y 200, respectivamente), por lo que estos cambios podrían significar que el modelo 3 es más estable, pues, si vemos los diagramas de densidad podemos notar que los datos están distribuidos de una forma un poco más uniforme, a excepción de CD8 que muestra que los datos no están tan bien diferenciados como en el modelo 2. Otro detalle importante a considerar es que el modelo 2 tiende a asignar muchos datos al grupo 2 (véase el gráfico de barras de abajo a la derecha), lo que sugiere que el modelo está altamente sesgado.

Ahora, sobre el número de clústers. Las métricas db.test.mean y G1.test.mean fueron muy similares entre los modelos, esto me hace pensar que el número de iteraciones no cambió de forma considerable la calidad de los grupos en términos de separación/diferenciación. Incluso en tiempo de ejecución no es significativa la diferencia que percibí, por lo que si queremos encontrar el número de clusters adecuado creo que es preferible probar otras cosas. Sin embargo, el trabajo hecho no fue en vano, pues, gracias a los experimentos noté que el modelo 3 es bastante mejor que el 2, pues, no está sesgado en la predicción de nuevas células. Quizá sea mejor idea usar el método de los codos para buscar el número de clústers adecuados en lugar de hacer pruebas con los parámetros de la función makeLearner(). Supongo que eso será lo que veremos en las próximas actividades.