Practical 7: Data Stream Mining (cont) - solved

I. Evaluation of data stream clustering

Internal evaluation measures:

- “average.between” Average distance between clusters
- “average.within” Average distance within clusters
- “max.diameter” Maximum cluster diameter
- “entropy” entropy of the distribution of cluster memberships

External evaluation measures:

- “precision” and “recall”:
  - Precision=TP/(TP+FP)
  - Recall=TP/(TP+FN)
- “purity”: Average purity of clusters. The purity of each cluster is the proportion of the points of the majority true group assigned to it.
- “Euclidean”: Euclidean dissimilarity of the memberships

See the stream package for more measures

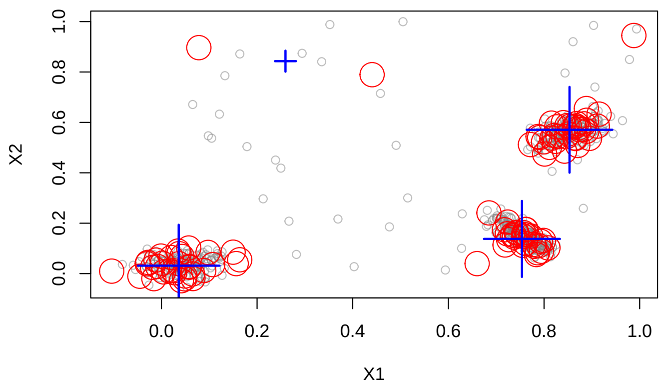
```
library("stream")
stream <- DSD_Gaussians(k = 3, d = 2, noise = .05)
```

1. Use Reservoir sampling to generate 100 data points and use K-means to generate 4 clusters.

```
Reservoir_Kmeans =
  DSC_TwoStage(micro = DSC_Sample(k = 100), macro = DSC_Kmeans(k = 4))
update(Reservoir_Kmeans, stream, n=500)
Reservoir_Kmeans
```

```
## Reservoir sampling + k-Means (weighted)
## Class: DSC_TwoStage, DSC_Macro, DSC
## Number of micro-clusters: 100
## Number of macro-clusters: 4
```

```r
plot(Reservoir_Kmeans, stream)
evaluate_static(Reservoir_Kmeans, stream
                , measure =
                  c("average.between", "precision", "recall"), n =500)
```




```
## Evaluation results for macro-clusters.
## Points were assigned to micro-clusters.
##
## average.between       precision          recall
##       0.7095732       0.9450260       0.9965974
## attr(,"type")
## [1] "macro"
## attr(,"assign")
## [1] "micro"
```

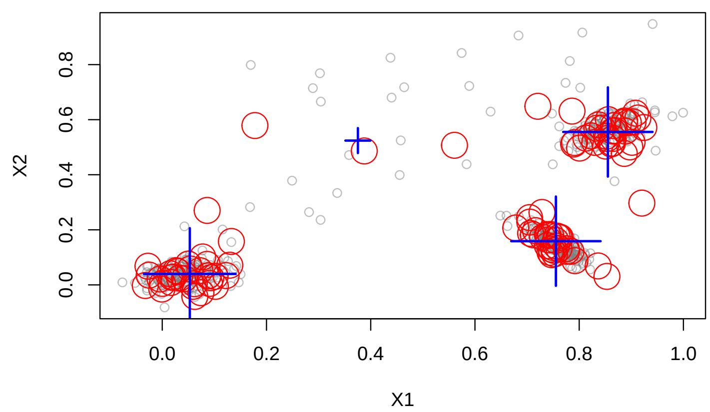
2. Use sliding window method rather than Reservoir sampling in the above example. Compare the precision and recall of the two methods.

   Hint: Window_Kmeans = DSC_TwoStage(micro = DSC_Window(horizon = 100), macro = DSC_Kmeans(k = 4)).

```r
Window_Kmeans = DSC_TwoStage(micro = DSC_Window(horizon = 100)
, macro = DSC_Kmeans(k = 4))
update(Window_Kmeans, stream, n=500)
Window_Kmeans
```

```
## Sliding window + k-Means (weighted)
## Class: DSC_TwoStage, DSC_Macro, DSC
## Number of micro-clusters: 100
## Number of macro-clusters: 4
```

```
plot(Window_Kmeans, stream)
```




```
evaluate_static(Window_Kmeans, stream
,measure = c("average.between"
,"precision","recall")
, n =500)
```

```
## Evaluation results for macro-clusters.
## Points were assigned to micro-clusters.
##
## average.between       precision          recall
##       0.7144800       0.9630804       0.9960112
## attr(,"type")
## [1] "macro"
## attr(,"assign")
## [1] "micro"
```

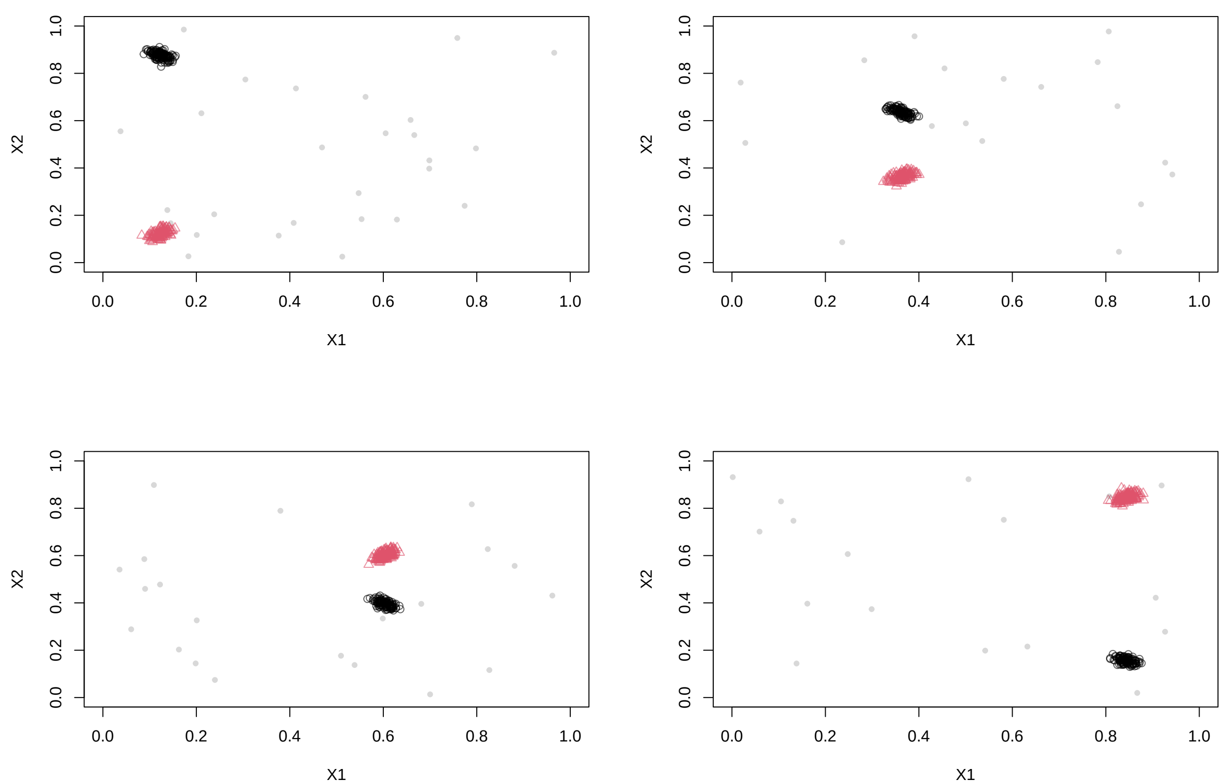
II. Concept Drift

Concept drift means the changes of the data generating process over time. It implies that the statistical properties of the data also change when time passes. A good data mining algorithm should be able to deal with concept drift. In the stream package, DSD_Benchmark(1) is an example data stream which contains concept drift. To show the concept drift we request four times 250 data points from the stream and plot them. To fast-forward in the stream we request 1400 points in between the plots and ignore them. The codes below will show 4 figures of the data at different time points.

```
stream <- DSD_Benchmark(1)
stream
```

```
## Benchmark 1: Two clusters moving diagonally from left to right, meeting
## in the center (d = 2, k = 2, 5% noise).Class: DSD_MG, DSD_R, DSD
## With 3 clusters in 2 dimensions. Time is 1
```

```
for(i in 1:4) {
plot(stream, 250, xlim = c(0, 1), ylim = c(0, 1))
tmp <- get_points(stream, n = 1400)
}
```



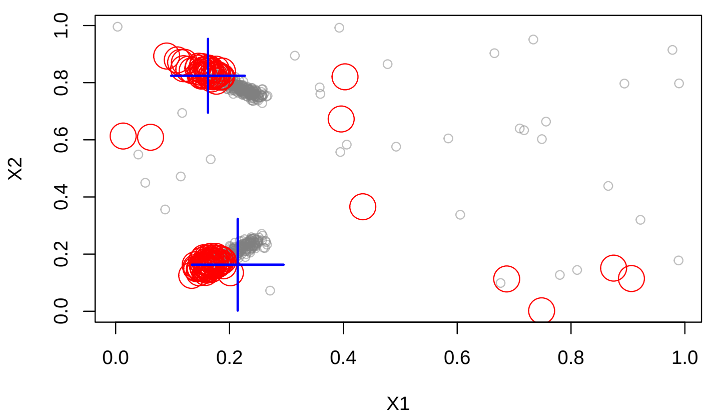
We can use animation package to demonstrate this:

```
reset_stream(stream)
animate_data(stream, n = 10000, horizon = 100
             , xlim = c(0, 1), ylim = c(0, 1))
library("animation")
animation::ani.options(interval = .1)
ani.replay()
```

III. Evaluation of data stream clustering with concept drift

1. Using Reservoir sampling and K-means

```
stream = DSD_Benchmark(1)
Reservoir_Kmeans= DSC_TwoStage(micro = DSC_Sample(k = 100, biased = TRUE)
                               , macro = DSC_Kmeans(k = 2))
update(Reservoir_Kmeans, stream, n=500)
plot(Reservoir_Kmeans, stream)
```




```
evaluate_stream(Reservoir_Kmeans, stream
                , measure = c( "precision", "recall"), n =5000, horizon=100)
```

```
##    points precision    recall
## 1       0 0.8166259 0.9876787
## 2     100 0.8506122 0.9928537
## 3     200 0.8781175 0.9954400
## 4     300 0.7689487 0.9792423
## 5     400 0.8482813 0.9930620
## 6     500 0.7690738 0.9782045
## 7     600 0.7250608 0.9659643
## 8     700 0.8335334 0.9904898
## 9     800 0.8164015 0.9881481
## 10    900 0.8341483 0.9932072
## 11   1000 0.8261734 0.9885769
## 12   1100 0.8000816 0.9849322
## 13   1200 0.8215686 0.9868111
## 14   1300 0.7381818 0.9712919
## 15   1400 0.4264864 0.9880656
```

```
## 16   1500 0.4397939 0.9912917
## 17   1600 0.4565357 0.9943899
## 18   1700 0.4175431 0.9910135
## 19   1800 0.4697097 0.9973202
## 20   1900 0.4096063 0.9949925
## 21   2000 0.4675325 0.9986790
## 22   2100 0.7979798 0.9909684
## 23   2200 0.4776610 0.9945280
## 24   2300 0.4411462 0.9967396
## 25   2400 0.9022526 0.9973321
## 26   2500 0.8169988 0.9877089
## 27   2600 0.8374083 0.9913169
## 28   2700 0.8020408 0.9859508
## 29   2800 0.8503457 0.9924063
## 30   2900 0.8865648 0.9959331
## 31   3000 0.9216327 0.9982317
## 32   3100 0.9391408 0.9991536
## 33   3200 0.8161141 0.9860270
## 34   3300 0.8316389 0.9912748
## 35   3400 0.8357664 0.9903892
## 36   3500 0.8497959 0.9923737
## 37   3600 0.8791517 0.9977293
## 38   3700 0.8605028 0.9943768
## 39   3800 0.8341483 0.9903241
## 40   3900 0.7984590 0.9859790
## 41   4000 0.7937574 0.9881948
## 42   4100 0.8671469 0.9944929
## 43   4200 0.8357664 0.9903892
## 44   4300 0.7845777 0.9816233
## 45   4400 0.9062626 0.9973321
## 46   4500 0.8479392 0.9929709
## 47   4600 0.7671068 0.9795606
## 48   4700 0.8519723 0.9961959
## 49   4800 0.8464646 0.9943047
## 50   4900 0.8731192 0.9953639
```

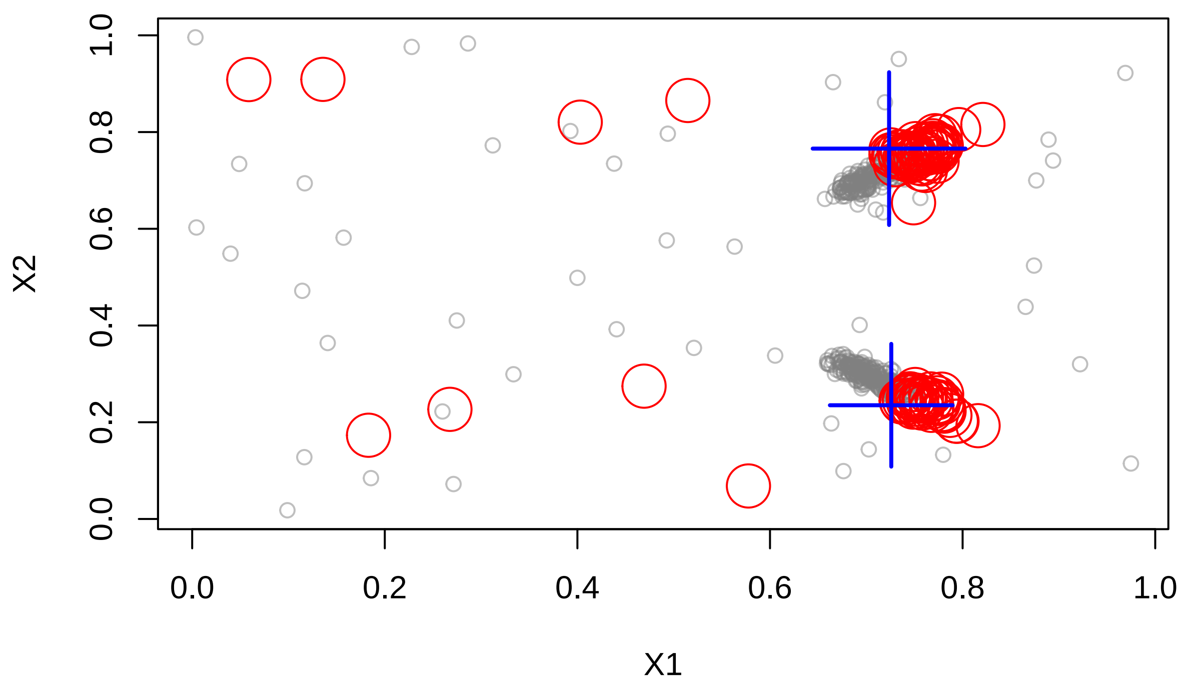
2. Evaluate the Sliding window + K-means clustering

```
#2. Sliding window + K-means clustering
Window_Kmeans = DSC_TwoStage(micro = DSC_Sample(k = 100, biased = TRUE)
, macro = DSC_Kmeans(k = 2))
update(Window_Kmeans, stream, n=500)
Window_Kmeans
```

```
## Reservoir sampling (biased) + k-Means (weighted)
```

```
## Class: DSC_TwoStage, DSC_Macro, DSC
## Number of micro-clusters: 100
## Number of macro-clusters: 2
```

```
plot(Window_Kmeans, stream)
```




```
evaluate_static(Window_Kmeans, stream
, measure = c("precision"
, "recall")
, n =5000, horizon=100)
```

```
## Evaluation results for macro-clusters.
## Points were assigned to micro-clusters.
##
## precision    recall
## 0.5207447 0.6257415
## attr(,"type")
## [1] "macro"
## attr(,"assign")
## [1] "micro"
```