Assignment 10

Prepared by:

- Josh Levine (jl2108)
- Harsh Patel (hkp49)
- Jaini Patel (jp1891)
- Yifan Liao (yl1463)
- Aayush Shah (avs93)

Sequential Probability Ratio Test

A sequential probability ratio test (SPRT) is a hypothesis test for sequential samples.

Sequential sampling works in a very non-traditional way; instead of a fixed sample size, you choose one item (or a few) at a time, and then test your hypothesis. You can either:

- Reject the null hypothesis (H_0) in favor of the alternate hypothesis (H_1) and stop,
- Keep the null hypothesis and stop,
- Fail to reach either conclusion and continue sampling.

If you fail to reach a conclusion, you repeat the sampling and then the hypothesis test. You keep on repeating this process until you have a sound conclusion, so you don't know the how big your sample will be until you're finished testing.

As in classical hypothesis testing, SPRT starts with a pair of hypotheses, say H_0 and H_1 for the null hypothesis and alternative hypothesis respectively. They must be specified as follows:

$H_0 : p = p_0$ $H_1 : p = p_1$

The next step is calculate the cumulative sum of the log-likelihood ratio, $\log \Lambda_i$, as new data arrive:

$S_i = S_{i-1} + \log \Lambda_i$

The stopping rule is a simple thresholding scheme:

$a < S_i < b$: continue monitoring (critical inequality)

$S_i \geq b$: Accept H_1

$S_i \leq a$: Accept H_0

where a and b ($0 < a < b < \infty$) are thresholds that depend on the desired type I and type II errors.

For this assignment,

S = cumulative sum of logs

$h_0 = hypothesis0 = p \leq p1$

$h_1 = hypothesis1 = p \geq p2$

alp1 = 0.01 ; alp2 = 0.01

We have our SPRT function that calculates our log sum S and then checks which hypothesis, h0 or h1, is correct depending on the following criteria:

$A < S < B$ - continue loop

$S \geq B$ - h_1 accepted

$S \leq A$ - h_0 accepted

where A and B are the thresholds we get based on alp1 and alp2 errors.

SPRT then returns the correct value to simulate. Then the simulate function (which calls SPRT) runs 100 iterations to choose either h_0 or h_1 depending on the higher count of accepted hypothesis and also calculates average steps to convergence.

Below is the code for SPRT function:

```r
# SPRT function


sprt <- function(alp1 = 0.01, alp2 = 0.01,
                 p1 = 0.45, p2 = 0.55, bernoulli_p = 0.3)
  {

  S = 0
  log_test = 0
  converge_count = 0

  # calculating stop threshold
  A = log(alp2/(1 - alp1))
  B = log((1 - alp2)/alp1)
  valid_hypothesis = -1

  while(TRUE){

    converge_count = converge_count + 1

    # generate bernoulli RV
    data_point = rbinom(1, 1, bernoulli_p)

    # log ratio
    log_test = (data_point*p2 + (1-data_point)*(1-p2))
    - (data_point*p1 + (1-data_point)*(1-p1))

    # cumulative sum
    S = S + log_test

    # check stop conditions
    if(S>=B){
      #Accept H1
      valid_hypothesis = 1
      break
    }
    if(S<=A){
      #Accept H0
      valid_hypothesis = 0
      break
    }
    #print(paste("data_point", data_point, "log_test", log_test, "S", S))
  }
  return(list(converge_count = converge_count, valid_hypothesis = valid_hypothesis))
}
```

The following code shows the simulation function:

```
#Simulation Function
simulate_sprt <- function(bernoulli_p = 0.3, iterations = 100){

  sum = 0
  h0 = 0
  h1 = 0

  # Averaging SPRT
  for(i in c(1:iterations)){
    # call the SPRT function that generates bernoulli RV with bernoulli_p
    result = sprt(bernoulli_p=bernoulli_p)

    # H0 is accepted
    if(result$valid_hypothesis == 0){
      h0 = h0 + 1
    }
    # H1 is true
    if(result$valid_hypothesis == 1){
      h1 = h1 + 1
    }
    sum = sum + result$converge_count
  }
  average = sum/iterations
  # print(paste("bernoulli_p=", bernoulli_p, ", avg_n=", average))
  return(list(average=average, h0=h0, h1=h1))
}
```

Below is the code to run the simulation function for a range of p values:

```
# Running the simulation function for a range of p values
test_list = seq(0.3, 0.68, by=0.02)

# data frame to store the result
img<-data.frame(bernoulli_p=numeric(),
                avg_steps_to_converge=numeric(), h0=numeric(),
                h1=numeric(), hypothesis_accepted=character())

#iterating over test_list
for(p in test_list){
  print(p)
  result = simulate_sprt(bernoulli_p = p)

  if(result$h0 < result$h1)
    accepted_h = "H1"
  if(result$h0 > result$h1 ){
    accepted_h = "H0"
  }
  if(result$h1 == result$h0){
    accepted_h = "H0/H1"
  }
```

```r
  new_data = data.frame(bernolli_p=p,
                        avg_steps_to_converge=result$average,
                        h0=result$h0, h1=result$h1,
                        accepted_hypothesis=accepted_h,
                        stringsAsFactors = FALSE)
  img<-rbind(img, new_data)
}
```

```
## [1] 0.3
## [1] 0.32
## [1] 0.34
## [1] 0.36
## [1] 0.38
## [1] 0.4
## [1] 0.42
## [1] 0.44
## [1] 0.46
## [1] 0.48
## [1] 0.5
## [1] 0.52
## [1] 0.54
## [1] 0.56
## [1] 0.58
## [1] 0.6
## [1] 0.62
## [1] 0.64
## [1] 0.66
## [1] 0.68
```

To show our results the code displays a table run on a sequence of Bernoulli from 0.3 to 0.68 with gap of 0.02

```r
#display img
img
```

```
##    bernolli_p avg_steps_to_converge h0  h1 accepted_hypothesis
## 1        0.30                  9.95  0 100                  H1
## 2        0.32                  9.98  0 100                  H1
## 3        0.34                  9.97  0 100                  H1
## 4        0.36                  9.96  0 100                  H1
## 5        0.38                  9.94  0 100                  H1
## 6        0.40                  9.95  0 100                  H1
## 7        0.42                  9.86  0 100                  H1
## 8        0.44                  9.82  0 100                  H1
## 9        0.46                  9.81  0 100                  H1
## 10       0.48                  9.79  0 100                  H1
## 11       0.50                  9.77  0 100                  H1
## 12       0.52                  9.69  0 100                  H1
## 13       0.54                  9.64  0 100                  H1
## 14       0.56                  9.59  0 100                  H1
## 15       0.58                  9.51  0 100                  H1
## 16       0.60                  9.50  0 100                  H1
## 17       0.62                  9.39  0 100                  H1
## 18       0.64                  9.41  0 100                  H1
## 19       0.66                  9.33  0 100                  H1
## 20       0.68                  9.30  0 100                  H1
```

Conclusion

`*   Why does it give the result you got?`

The results in general show that p values from 0.45 to 0.55 require the highest number of iterations to converge with the maximum at $p = 0.5$. This is because p is equidistant from both $p = 0.45$ and $p = 0.55$ so the log sum S will keep changing and going from one hypothesis to other until eventually it converges to either h_1 or h_0 based on the count. Both h_0 and h_1 are accepted almost equal amount of times at $p = 0.5$ so we cannot know for sure which will we see until the end.

As we go farther out from $p = 0.45$ and $p = 0.55$ we see that it takes less and less iterations to converge as S just goes to either h_0 or h_1 fairly soon. So, For $p < 0.5$ h_0 is accepted $p > 0.5$ h_1 is accepted and at $p = 0.5$ both of equal likelihood of getting accepted.

`* What do you think it would do for .54 ? Try it.`

Now for $p = 0.54$. From our above conclusions we should see that at $p = 0.54$ h_1 should be accepted because there is a higher probability for h_1 to accepted (h_0 is rejected) as it is closer to 0.55 than h_0. Also, $p = 0.54$ should have lower steps than $p = 0.5$ to converge. Both of these results can be verified based on our output table. Hence SPRT function works as expected.