Question & Answers

1. Estimate the power of the skewness test for normality against symmetric Beta(*α, α*) distributions and comment on the results.

|  |  |
| --- | --- |
| Codes & Explanation | Discussion (If applicable) |
| |  | | --- | | ```{r}  # Skewness Function  sk <- function(x) {  # Skewness Formula  xbar <- mean(x)  m3 <- mean((x - xbar)^3)  m2 <- mean((x - xbar)^2)  return( m3 / m2^1.5 )  }  ```  ```{r}  alpha <- 0.1 # To estimate power we need trustable alpha value  n <- 300 # Sample Size  m <- 2500 # Size for numeric table  ab <- 1:10 # Beta Distribution Trials  N <- length(ab) # Length of beta distribution trials  pwr <- numeric(N)  cv <- qnorm(1 - alpha/2, 0, sqrt(6 \* (n - 2)/((n + 1) \* (n +3)))) # coefficent of variation this will give us values that are taken from normal as a relative standard dispersion of the distribution    for (j in 1:N) {  a <- ab[j]  sktests <- numeric(m)  for (i in 1:m) {  x <- rbeta(n, a, a) # As skewness is best in same alpha and beta parameters, both are selected equal because mean is always equal to 0.5 from alpha / (alpha\*2)  sktests[i] <- as.integer(abs(sk(x)) >= cv)  }  pwr[j] <- mean(sktests)  }  pwr1 <- rev(pwr)    ``` | | Normality tests are concluded for to detect whether there are some non-normal behavior in a data set with sample size no less than 100 with certain exceptions on some of the tests that will go lower down to 50 samples(with extreme non normality), but preferred behavior is to apply the test to large data set. Otherwise, one might think of using non-parametric methods with no need of use of normality power.  Power of the scores of normality test is determined with the scores taken from the tests such as kolmogorov- smirnov or Shapiro-Wilk W test these tests ara some of the normality tests, and by these tests we can after visualization of the data with density plots, box plots or so to understand if the score is low to see, which part of the plot has discrepancies with normality. These power test is important for one other thing and with that thing it can also be altered. That being said that thing is the scale of the data (one can change it to squares- logs etc) in such a way that find a higher score and better scored normality test. |
| |  | | --- | | ```{r}  alpha <- 0.1  n <- 30  m <- 2500  df <- c(1:5, seq(10, 50, 10))  N <- length(df)  pwr <- numeric(N)  cv <- qnorm(1 - alpha/2, 0, sqrt(6 \* (n - 2)/((n + 1) \*(n + 3))))  for (j in 1:N) {  nu <- df[j]  sktests <- numeric(m)  for (i in 1:m) {  x <- rt(n, df = nu)  sktests[i] <- as.integer(abs(sk(x)) >= cv)  }  pwr[j] <- mean(sktests)  }    data.frame(df, pwr)  r <- rbind(df, pwr,pwr1)  r <- as.data.frame(r)  rownames(r) <- c('df','Heavy Tailed','Light Tailed')  r  ``` | | Power decreases as we increase parameters of beta distributions. (Order of light tailed is reversed. )  It is also important to mention that when degree of freedom is low, we achieve more power which can be resulted from formula of skewness test is positevely correlated with kurtosis. |

1. Conduct an MC study to estimate the coverage probabilities of the standard normal bootstrap confidence interval and the basic bootstrap confidence interval. For this purpose, sample from i) a normal population and also from ii) a *χ*2(5) distribution and check the empirical coverage rates for i) the sample mean and for ii) the sample skewness statistic.

2.A: Normal Population | Basic & Standard | Mean & Skewness  
2.B: χ2(5) Distribution | Basic & Standard | Mean & Skewness

|  |  |
| --- | --- |
| Code & Comments | Discussion (If applicable) |
| # We are defining our mean and skewness functions for our data  ```{r}  mn <- function(x,i){  x <- x[i] # We visit each object within the data  return(mean(x))  }  sk <- function(x,i) {  x <- x[i] # We visit each object within the data  # Skewness Formula  xbar <- mean(x)  m3 <- mean((x - xbar)^3)  m2 <- mean((x - xbar)^2)  return( m3 / m2^1.5 )  }  ``` |  |
| ```{r,warning=FALSE}  library(boot) # Package for bootstrap  n <- 100 # Sample Size  m <- 1000 # Replicate Numbers    reps <- replicate(m, expr = {  x <- rnorm(n, 0 , 1 ) # We are creating a sample from normal population  boot.obj <- boot(x, R = 50, statistic = mn ) # We create bootstrap samples of it 50 times in respect to `Mean`  boot.ci(boot.obj, type=c("basic","norm")) # We find Basic and Standard Confidence Intervals  } )    reps.skew <- replicate(m, expr = {  x <- rnorm(n, 0 , 1 ) # We are creating a sample from normal population  boot.obj <- boot(x, R = 50, statistic = sk ) # We create bootstrap samples of it 50 times in respect to `Skewness`  boot.ci(boot.obj, type=c("basic","norm")) # We find Basic and Standard Confidence Intervals  } )  ``` |  |

|  |  |
| --- | --- |
| ```{r}  # Empty Coverage Variables To Fill In  normal\_covarage <- 0 ;basic\_coverage <- 0;s.normal\_covarage <- 0 ;s.basic\_coverage <- 0  # As our mean and skewness values is 0, we compute the ones which are within our borders  for(i in 1:m){  if(0 >= reps[,i]$normal[2] & 0 <= reps[,i]$normal[3]){  normal\_covarage <- normal\_covarage + 1  }  if(0 >= reps[,i]$basic[4] & 0 <= reps[,i]$basic[5]){  basic\_coverage <- basic\_coverage + 1  }  if(0 >= reps.skew[,i]$normal[2] & 0 <= reps.skew[,i]$normal[3]){  s.normal\_covarage <- s.normal\_covarage + 1  }  if(0 >= reps.skew[,i]$basic[4] & 0 <= reps.skew[,i]$basic[5]){  s.basic\_coverage <- s.basic\_coverage + 1  }  }  # We define a empty matrix for our Normal Population which is going to be filled with Normal & Basic coverage rates of their confidence intervals  matrix <- matrix(0, nrow = 2, ncol = 2, dimnames = list(c(1:2), c("Normal","Basic")))  rownames(matrix) = c('Mean',"Skew")  matrix[1,1] <- (normal\_covarage / m)  matrix[1,2] <- (basic\_coverage / m)  matrix[2,1] <- (s.normal\_covarage / m)  matrix[2,2] <- (s.basic\_coverage / m)  as.data.frame(matrix)  ``` |  |

2.B: χ2(5) Distribution | Basic & Standard | Mean & Skewness

|  |  |
| --- | --- |
| Code & Explanation | Additional Explanation |
| ```{r,warning=FALSE}  m <- 1000  n <- 100  k <- 5  reps <- replicate(m, expr = {  x <- rchisq(n, df= k) # We are creating a sample from χ2(5) distribution  boot.obj <- boot(x, R = 50, statistic = mn ) # We create bootstrap samples of it 50 times in respect to `Mean`  boot.ci(boot.obj, type=c("basic","norm")) # We find Basic and Standard Confidence Intervals  } )  reps.skew <- replicate(m, expr = {  x <- rchisq(n, df= k ) # We are creating a sample from χ2(5) distribution  boot.obj <- boot(x, R = 50, statistic = sk ) # We create bootstrap samples of it 50 times in respect to `Skewness`  boot.ci(boot.obj, type=c("basic","norm") ) # We find Basic and Standard Confidence Intervals  } )  ``` |  |
| ```{r}  normal\_covarage <- 0 ;basic\_coverage <- 0;s.normal\_covarage <- 0 ;s.basic\_coverage <- 0  k <- 5  mu <- k # It is given by question  skew <- sqrt(8/k) # It is calculated by this formula    for(i in 1:m){  # Mean & Normal  if( mu >= reps[,i]$normal[2] & mu <= reps[,i]$normal[3]){  normal\_covarage <- normal\_covarage + 1  }  # Mean& Basic  if( mu >= reps[,i]$basic[4] & mu <= reps[,i]$basic[5]){  basic\_coverage <- basic\_coverage + 1  }  # Skewness & Normal  if( skew >= reps.skew[,i]$normal[2] & skew <= reps.skew[,i]$normal[3]){  s.normal\_covarage <- s.normal\_covarage + 1  }  # Skewness& Basic  if( skew >= reps.skew[,i]$basic[4] & skew <= reps.skew[,i]$basic[5]){  s.basic\_coverage <- s.basic\_coverage + 1  }  }  matrix <- matrix(0, nrow = 2, ncol = 2, dimnames = list(c(1:2), c("Normal","Basic")))  rownames(matrix) = c('Mean',"Skew")  matrix[1,1] <- (normal\_covarage / m)  matrix[1,2] <- (basic\_coverage / m)  matrix[2,1] <- (s.normal\_covarage / m)  matrix[2,2] <- (s.basic\_coverage / m)  as.data.frame(matrix)  ``` |  |

1. Construct a kernel density estimate for the simulated data from the normal location mixture 0.5*N*(0*,* 1) + 0.5*N*(3*,* 1). Plot the true density of the mixture over the density estimate for comparison; compare several choices of bandwidth. Which choice of smoothing parameter appears to be best? Discuss.

|  |  |
| --- | --- |
| Code & Explanation | Discussion |
| ```{r}  n <- 500  p <- 0.5  mu <- sample(c(0, 3), size = 1000, replace = TRUE, prob = c(p, 1 - p))    set.seed(196)  x <- rnorm(n, mu, 1)  S <- sd(x)  h <- rep(1, 8)  h[1] <- 1.06 \* sd(x) \* n^(-1/5)    for( i in 2:length(h)){  set.seed(196+i)  h[i] <- 0.9 \* min(c(S, IQR(x)/1.34)) \* n^(-1/5) + rnorm(1,0,sd=0.01)  }    par(mfrow = c(3, 3))  hist(x)  plot(density(x, bw = h[1],kernel = "gaussian"))  plot(density(x, bw = h[1],kernel = "biweight"))  plot(density(x, bw = h[2],kernel = "gaussian"))  plot(density(x, bw = h[2],kernel = "biweight"))  plot(density(x, bw = h[5],kernel = "gaussian"))  plot(density(x, bw = h[6],kernel = "gaussian"))  plot(density(x, bw = h[7],kernel = "gaussian"))  plot(density(x, bw = h[8],kernel = "gaussian"))  par(mfrow = c(1,1))  ``` | |  | | --- | | We construct a kernel density estimator which is in machine learning referred as kernel trick, with tuning the hyperparameter (bandwidth) which changes the smoothness of the fit to the real data without undersmoothing ==> which leads to getting too much detail (also can be thought as the bias-variance trade-off in machine learning) or oversmoothing ==> which leads to losing much of the important detail in the dataset. | | Here bandwidth approximately 0.5 for the bimodal gaussian mixture model gives relatively great smoothing, however when the values increase (bandwidth values) we ser a clear oversmoothing which makes the data fit as hill shaped and below values give the result of undersmoothing (or noise) in other words. | |

4- Plot a bivariate ASH density estimate of the geyser(MASS) data.

|  |  |
| --- | --- |
| Code & Explanation | Discussion |
| ```{r}  library(MASS)  geyser  # Data is not normally distributed  shapiro.test(geyser$waiting)    hist(geyser$waiting)  hist(geyser$duration)  ```    ```{r, warning= FALSE}  library(ash)  library(MASS)  attach(geyser)    # As Bivariate ASH Methods can be used for data with more than 2 dimensions, we are able to apply.  x <- as.matrix(geyser)  # It is not normally distributed and symmetric so Scott Rule cannot be applied.  shapiro.test(x)  # It seems biweight data as there are two peaks  hist(x, breaks = "Scott", prob = TRUE)    detach(geyser)  detach(package:MASS)  ```  ```{r}  nbin <- c(40, 40)  m <- c(5, 5)  b <- bin2(x, nbin = nbin)  est <- ash2(b, m = m, kopt = c(1, 0))    persp(x = est$x, y = est$y, z = est$z, shade = TRUE,  xlab = "X", ylab = "Y", zlab = "", main = "", theta = 30,  phi = 75, ltheta = 30, box = FALSE)  contour(x = est$x, y = est$y, z = est$z, main = "")  ``` | |  | | --- | | ASH, which is the abbreviation of the average shifted histogram is actually a early day kernel density estimator, or one can state it as the prometheus of the kernel density estimators (such as perceptrons being the early linear machine learning learners) ASH makes use as a simple kernel density estimator. | | It is basically used for computational simplicity of histograms which gives success compared with kernel density estimators, but also has comparable performance costs. | | ASH for our gayser data will generate the underlying distribution with delta\_bin\_width = h/m m is changing for sample size and h is the determined bin width will change smoothness of our density estimator, in that sense ASH is the estimator. | |