DataMining

hw1

192stg11 우나영

2020

# ­­­­­1. Description

이번 시간에는 전반적인 statistical learning에 대해 배웠다. 예를 들면, response y와 input vector 가 주어졌을 때 response y를 예측하기 위한f를 추정한다고 가정해보자. 가장 이상적인 f는 일 것이다. 하지만 대개의 경우 에 관측치가 충분치 않아 을 계산하는 것은 불가능하다. 따라서 K-Nearest-Neighbor 기법을 활용하여 로 조건을 완화시켜 을 구할 수 있다. 하지만 K-Nearest-Neighbor 기법은 이고 이 클 때는 용이하지만 가 커지면 Curse of dimensionality가 발생한다. Curse of dimensionality란 데이터의 차원이 높아짐에 따라 데이터 사이의 거리가 멀어져 sparsity 현상이 나타나는 것을 말한다. 이러한 sparse sample에서 대부분의 데이터 point들이 sample의 가장자리에 위치한다. 예를 들어 N개의 데이터 point들이 원점을 중심으로 하는 p차원의 unit sphere 내부에 uniform 하게 분포한다고 가정하자. 이때 원점에서 가장 가까운 점 사이 거리의 중앙값을 구하는 공식은 아래와 같다.

일 때 이다. 즉, 가장 가까운 점과의 거리의 중앙값이 절반 보다 조금 더 멀리 위치해 있다. 따라서, 대부분의 점들이 sample의 가장자리에 위치함을 확인할 수 있다.

위와 같은 결과를 실제 시뮬레이션에서도 얻을 수 있는지 R을 이용해 확인해 볼 것이다. 이를 위해 첫 번째로, uniform하게 분포하는 N개의 임의의 데이터 point들을 원점을 중심으로 하는 p 차원의 unit sphere에서 생성시키는 “myrand” function을 만든다. 알고리즘에 따르면, (Box, G. E. P,; Muller, Mervin E., 1958) direction 관점에서, 평균이 0이고 표준편차가 1인Standard Normal분포를 따르는 을

위의 식 (1.2)에 대입한 값들이 원점을 중심으로 한 p차원 sphere의 surface위에 uniform하게 분포하게 된다. distance 관점에서, 원점으로부터 random point 까지의 거리 은 아래와 같은 분포를 따른다.

distance는 Inverse CDF Thm을 이용하여 로 구할 수 있다. 원점에서 거리가 random인 sample을 구하기 위해 Uniform(0,1)을 따르는 를 r에 대입시키면 distance는 로 구할 수 있다.

따라서 (1.3)은 원점을 중심으로 하는 p 차원의 sphere 내부에서 uniform하게 분포하게 된다. 다음으로는 위의 myrand function으로 N과 p를 바꿔가며 10,000번의 simulation을 통해 구한 원점에서 가장 가까운 점 사이의 거리 샘플의 median값과 (1.1)식에 대입한 값을 비교한다.

마지막 과제는 ISL(II)에서 Lab 2.3의 R코드를 실행해보는 것이다. Lab은 그래프 그리기, 통계량 계산, 그리고 데이터 로딩 및 추출 등 R에 대해 전반적으로 기초 문법 코드를 다룬다. Lab의 코드 실행 결과는 첨부하지 않고 Discussion파트에서 Lab을 통해 배운 내용을 정리할 것이다.

# 2. Implementation

Curse of dimensionality 현상의 예제에서 사용된 원점에서 가장 가까운 점 사이 거리의 중앙값을 구하는 공식인 식(1.1)을 증명해 볼 것이다. radius가 인 sphere의 부피는 이다. 또한 p차원의 unit sphere에 uniform하게 분포된 N개의 데이터 라고 할 때, 하나의 데이터 point 이 원점으로부터 거리가 r보다 작을 확률은 radius가 r인 p차원의 sphere의 부피를 p차원의 unit sphere의 부피로 나누는 것이다.

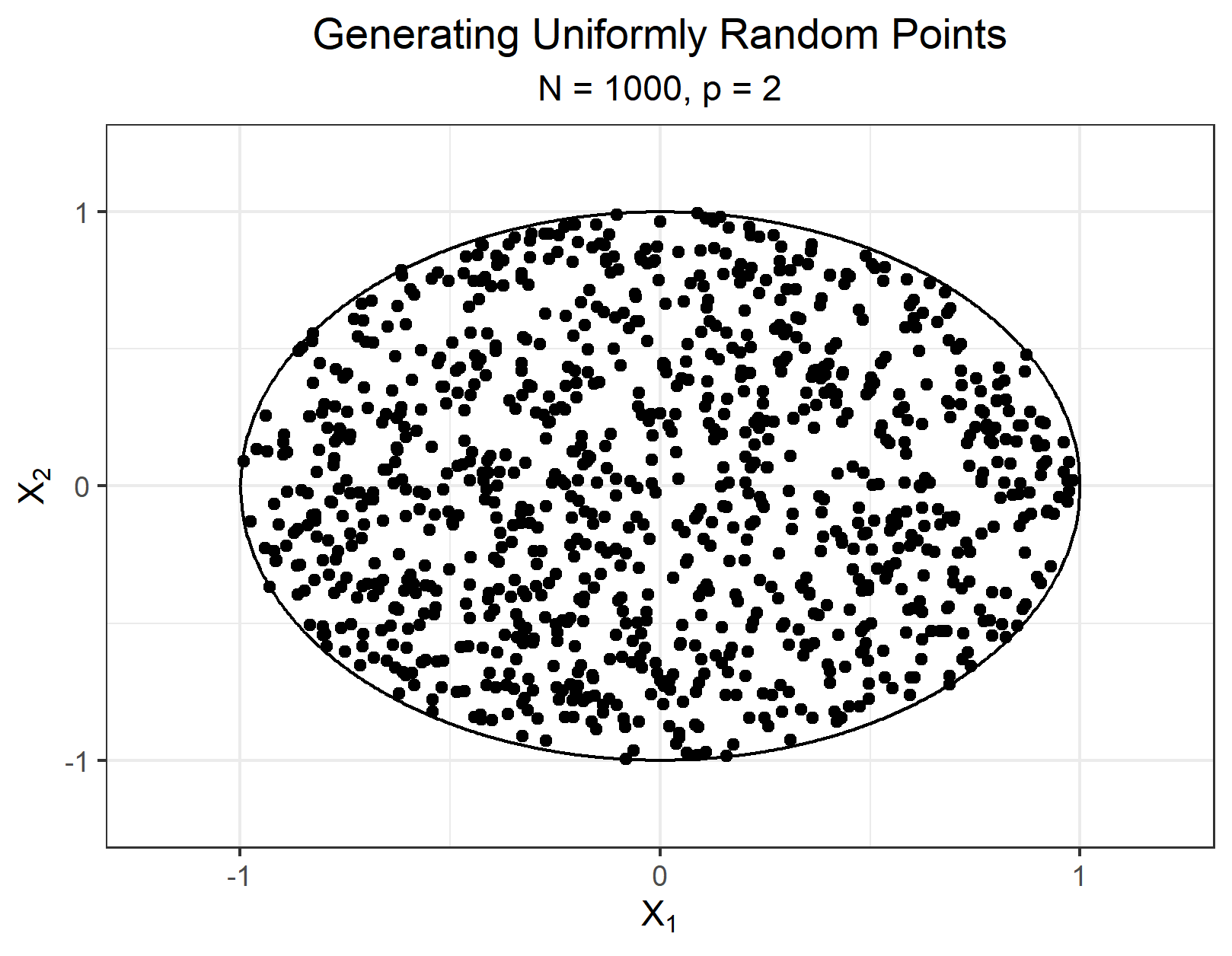
따라서 을 원점에서 가장 가까운 점이라고 할 때,

따라서 원점에서 가장 가까운 점 사이 거리의 중앙값은,

이므로 위의 식을 r에 대해서 derive하면,

다음으로 curse of dimensionality를 구현해 볼 것이다. 아래는 원점을 중심으로 하는 p 차원의 unit sphere에서 uniform하게 분포하는 N개의 임의의 데이터 point들을 생성하는 myrand function의 pseudo code이다.

|  |
| --- |
|  |
| 1. 생성  2. 각각 observation의 을 계산  3. Unif(0,1)을 따르는 를 생성  4. return |

텍스트, 하얀색이(가) 표시된 사진

자동 생성된 설명

위의 그림을 통해 myrand function이 생성한 random sample들이 unit sphere 내부에 위치하는 것으로 보아 p차원의 unit sphere 내부 데이터의 샘플링이 잘 이루어졌다고 판단할 수 있다. 좌측은 2차원의 sphere에서 생성시킨 1000개의 random sample이며 우측은 3차원의 sphere에서 생성시킨 1000개의 random sample이다.

다음은 myrand function을 사용하여 N과 p의 값을 바꿔가며 10,000의 simulation을 통해 구한 median 값과 (1.1)을 통해 구한 median값을 비교해 본 table이다.

1)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 5 | 0.3697 | 0.3704 | 0.0007 |
| 10 | 0.6080 | 0.6078 | 0.0002 |
| 15 | 0.7177 | 0.7181 | 0.0004 |
| 20 | 0.7798 | 0.7806 | 0.0008 |
| 30 | 0.8472 | 0.8466 | 0.0006 |

2)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 5 | 0.2681 | 0.2693 | 0.0012 |
| 10 | 0.5178 | 0.5171 | 0.0007 |
| 15 | 0.6448 | 0.6452 | 0.0004 |
| 20 | 0.7196 | 0.7196 | 0 |
| 30 | 0.8030 | 0.8028 | 0.0002 |

3)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 5 | 0.2334 | 0.2334 | 0 |
| 10 | 0.4831 | 0.4824 | 0.0007 |
| 15 | 0.6157 | 0.6165 | 0.0008 |
| 20 | 0.6951 | 0.6946 | 0.0005 |
| 30 | 0.7847 | 0.7850 | 0.0003 |

위의 표를 통해 simulation을 통해 구한 median 값과 식 (1.1)을 통해 구한 median 값의 error가 0.001 이내로 매우 작으므로 두 방법을 이용한 결과값이 비슷하다고 판단할 수 있다. Curse of dimensionality 현상에 의해 p가 커짐에 따라 원점과 가장 가까운 데이터 point 사이 거리의 median 값이 커진다. 반면에 p가 동일할 때 N이 클수록 median값이 줄어든다.

# 3. Discussion

본 과제를 통해 Statistical Learning을 할 때, K-Nearest-Neighbor에서 빈번히 발생하는 문제인 Curse of dimensionality에 대해 공부할 수 있었다. Curse of dimensionality는 데이터의 차원이 높아짐에 따라 데이터 sparsity가 발생하는 문제이다. 대표적인 현상이 바로 차원이 높아짐에 따라 데이터들이 sample의 가장자리에 모이는 것이다. 이러한 현상을 직관적으로 보여주는 지표가 원점을 중심으로 하는 p차원 sphere의 random sample 중 가장 원점과 가까운 데이터 point의 median distance이다. median distance를 구하는 식은 (1.1)이며 이를 증명해보았다. 또한 직접 myrand function을 이용해 p차원의 sphere에서 데이터를 샘플링하여 원점과 가장 가까운 데이터의 median distance를 계산하는 simulation을 했다. 이를 식(1.1)으로 계산한 값과 비교해본 결과 오차가 0.001로 굉장히 낮다는 것을 확인했다. simulation을 해본 결과 p가 커짐에 따라 median distance 값이 점점 커지는 반면 N이 커짐에 따라 median distance값이 점점 작아지는 것을 알 수 있었다.

마지막 과제인 Lab은 Basic command, Graphics, Indexing Data, Loading Data 그리고 Additional Graphical and Numerical Summarize로 총 4개의 주제로 이루어졌다. Basic command에서 vector생성과 병합 방법 그리고 변수에 값을 저장하는 방법과 삭제 방법을 배웠다. Graphics에서는 R의 가장 기본적인 그래프 그리는 함수인 plot함수와 그래프를 저장하는 방법을 배웠다. Indexing Data에서는 matrix를 생성하고 matrix의 값을 추출하는 방법을 배웠다. Loading Data에서는 간단한 데이터를 R에서 불러들여 변수의 명을 살피는 법과 데이터 사이즈를 확인하는 방법을 배웠다. 마지막으로 Additional Graphical and Numerical Summarize에서는 numeric 변수의 통계량 계산 방법과 변수들 간의 관계를 살펴볼 수 있는 간단한 plotting 기법을 배웠다. 이번 Lab을 통해 가장 흥미로웠던 것은 3가지로 첫번째, plot의 argument 중에서 horizontal을 사용하면 x축과 y축이 바뀐다는 것과 varwidth로 변수의 variance 크기에 따라 서로 다른 boxplot을 시각화 해준다는 것이 새로웠다. 두 번째로는, pairs를 이용하면 각각의 numerical 변수들을 매칭하여 scatter plot을 그려준다는 것이다. pairs를 이용하면 변수들의 상관관계를 시각적으로 살펴볼 수 있다는 장점이 있다. 마지막으로 가장 새로웠던 것은 identify() 기능이었다. 예를 들면, scatter plot에서 마우스로 정보가 궁금한 점을 클릭하고 identify() 함수를 끄면 그 점의 row number를 알려준다.

# 4. Appendix

## ------------- HW2 ------------- ##

## myrand 함수 만들기 ##

myrand <- function(n, p){

xx <- matrix(rnorm(n\*p), nrow = n, ncol = p)

S <- rowSums(xx^2)

u <- runif(n)

result <- (xx\*(u^(1/p)))/sqrt(S)

return(result)

}

## plot 그려보기 ##

# 2차원

library(ggplot2)

sample1 <- data.frame(myrand(1000, 2))

x <- seq(-1, 1, length=500)

y <- sqrt(1-x^2)

ggplot(sample1) + geom\_point(aes(X1, X2)) + geom\_path(aes(x=c(x, rev(x)), y=c(y, -y))) + theme\_bw() +

labs(title = "Generating Uniformly Random Points", subtitle = "N = 1000, p = 2", x = expression(X[1]), y = expression(X[2])) + theme(plot.title = element\_text(hjust = 0.5), plot.subtitle = element\_text(hjust = 0.5))+

scale\_x\_continuous(breaks = c(-1, 0, 1), limits = c(-1.2, 1.2)) + scale\_y\_continuous(breaks = c(-1, 0, 1), limits = c(-1.2, 1.2))

ggsave('2dim.png')

# 3차원

library(scatterplot3d)

sample2 <- data.frame(myrand(1000, 3))

png('3dim.png')

source('http://www.sthda.com/sthda/RDoc/functions/addgrids3d.r')

s3d <- scatterplot3d(sample2[,1:3], xlab = expression(X[1]), ylab = expression(X[2]), zlab = expression(X[3]),

main = "Generating Uniformly Random Points", sub = "N = 1000, p = 3", pch = "",grid = FALSE, box = FALSE)

addgrids3d(sample2[, 1:3], grid = c("xy", "xz", "yz"))

s3d$points3d(sample2[, 1:3], pch = 16)

dev.off()

## ------------- HW3 ------------- ##

d <- function(p, n){

(1-(1/2)^(1/n))^(1/p)

}

simulation <- function(n , p , n.sim = 10000){

sim.res <- c()

for(i in 1:n.sim){

x <- myrand(n = n, p = p)

dist1 <- sqrt(rowSums(x^2))

min1 <- min(dist1)

sim.res[i] <- min1

}

return(median(sim.res))

}

p <- c(5, 10, 15, 20, 30)

for(N in c(100, 500, 1000)) print(data.frame(p = p, dpn = d(p = p, n = N), sim = sapply(p, function(p) simulation(n = N, p = p, n.sim = 10000))))