2019 학년도 1학기
DATA MINING
HW2

과목명	데이터마이닝
담당교수명	송종우 교수님
제출일	2019.03.20
학번	182STG27
이름	임지연

Ⅰ. Description

KNN (K-Nearest Neighber) 방법은 지도학습 중 분류 문제에 사용하는 알고리즘이다. 즉 데이터가 기존 데이터의 그룹 중 어떤 그룹에 속하는지를 분류하는 문제를 말한다. K는 몇 번째로 가까운 데이터까지를 살펴본 것인지를 나타낸다. KNN 에서 가깝다는 개념은 유클리드 거리(Euclidean Distance)로 정의하는데, 유클리드 거리를 계산할 때 단위가 매우 중요하다. 따라서 알고리즘을 적용하기 전에 반드시 데이터에 표준화를 해 주어야 한다.

차원의 저주(Curse of Dimensionality) 란 데이터가 N개가 p차원 상의 점이라고 가정할 때, 차원의 수가 커질수록 데이터 양이 기하급수적으로 증가하게 된다는 것이다. 즉, 데이터의 수를 N개로 고정했을 때, p가 증가할 수록 공간의 크기가 커지고 데이터의 밀도가 점점 희박해지기 때문에 해당 공간을 표현하기 위해서는 데이터의 양이 더 많이 필요하다는 것이다. 이런 현상이 발생하면 training set에서의 학습을 느리게 할 뿐 아니라 예측이 불안정해지기 때문에 최적의 솔루션을 찾기도 어려워진다. 이론적으로는 데이터의 크기를 키워서 해결할 수 있지만 현실적으로 데이터가 무한정으로 주어지기 않기 때문에 차원의 저주 문제를 해결하기 위해서 PCA(Principle Component Analysis) 등 다양한 차원축소(Dimensionality Reduction) 방법을 사용할 수 있게 된다.

KNN 방법을 사용하면 차원이 커질 때 차원의 저주 문제가 발생할 수 있게 되고 따라서 효과적인 방법이 아닐 것이다. 따라서 알고리즘을 사용할 때 주의가 필요하다.

이번 과제는 총 3가지로 간단히 요약하자면, 1번에서는 p차원 상의 데이터가 N 개라고 가정할 때 원점과 데이터의 거리를 나타내는 공식이 주어져 있고 이를 유도해본다. 2번에서는 p차원 상의 unit sphere 에서 직접 데이터를 생성한 후 3번에서는 2번에서 생성한 데이터로부터 원점까지의 거리의 최소값들 중 중앙값을 계산해보고 1번 공식에 대입하여 구한 거리와 비교해 보는 것이다. p값과 N값이 변화할 때의 값을 여러 번 바꿔가며 계산해본다.

Ⅱ. Implementation

Question 1.

Consider N data points uniformly distributed in a p-dimensional unit ball centered at the origin. Consider a nearest neighbor estimate at the origin. The median distance from the origin to the closet data point is given by $d(p,N) = \left(1 - \frac{1}{2}^{\frac{1}{N}}\right)^{1/p}$, Derive the equation.

Hint | Order Statistic 에서 Minimum분포를 살펴보자. $X_1, X_2, \cdots, X_n \sim F$,

$$p\big(X_{(1)} < t\big) = 1 - P\big(X_{(1)} > t\big) = 1 - \prod_{i=1}^{n} P\big(X_{(i)} > t\big) = 1 - \big(1 - F(t)\big)^n,$$
$$\int_{-\infty}^{c} f(x)\,dx = \frac{1}{2},\ c = median$$

Sol) 차원이 p, 데이터 수가 N 일 때 unit sphere 에 대해서

Let r : Median , $\prod_{i=1}^{N} P(\|x_i\| > r) = \frac{1}{2} \Leftrightarrow \prod_{i=1}^{N}(1 - P(\|x_i\| < r)) = \frac{1}{2}$

$\Leftrightarrow \prod_{i=1}^{N}(1 - P(\|x_i\| < r)) = \frac{1}{2} \Leftrightarrow \prod_{i=1}^{N}(1 - r^p) = \frac{1}{2}$

$\Leftrightarrow (1 - r^p)^N = \frac{1}{2} \Leftrightarrow (1 - r^p)^N = \frac{1}{2}$

$\Leftrightarrow (1 - r^p) = \frac{1}{2}^{\frac{1}{N}} \Leftrightarrow r^p = 1 - \frac{1}{2}^{\frac{1}{N}}$

$\Leftrightarrow r = (1 - \frac{1}{2}^{\frac{1}{N}})^{\frac{1}{p}}$

Question 2.

P-dimensional unit sphere 에서 random point를 generate 하는 함수를 만들어라.

참고논문 | Annals y Math Stat 1972 vol 43 NO2 p640-646- George Morsatia,

"Choosing a point from the surface y sphere."

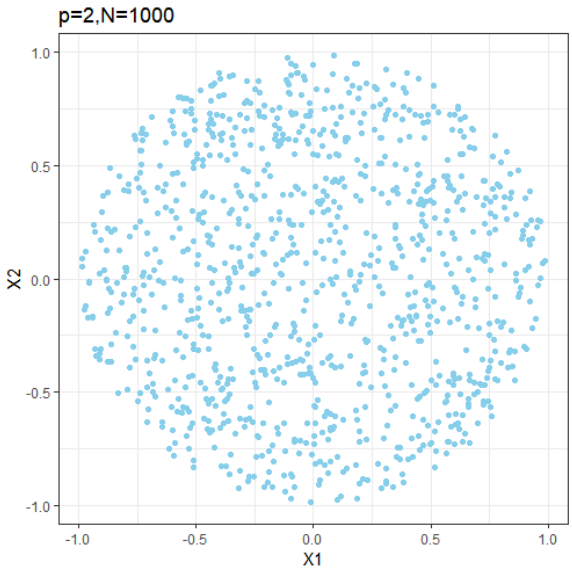
Hint | $\mathrm{X_i} \sim \text{iid unit sphere(p)},\ Let\ di = d(\mathrm{X_i}, 0),\ p(\mathrm{d_i} < \mathrm{r}) \propto r^p\ ,\ p(\mathrm{d_i} < \mathrm{r})$?

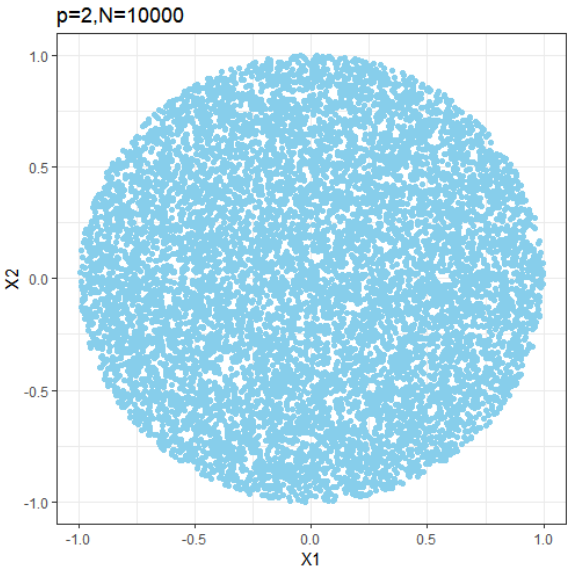
How to generate this distance ? -> Inverse CDF them이용

Sol) p차원 공간에서 크기가 1인 점을 생성하기 위하여 먼저 Standard Normal Distribution 에서 난수($\mathrm{X_1}, \mathrm{X_2}, \cdots, \mathrm{X_N}$) 를 생성한 후 단위벡터로 만들어주기 위해서 크기 S= $\sqrt{\mathrm{X_1^2} + \mathrm{X_2^2} + \cdots + \mathrm{X_N^2}}$ 로 각각의 모든 값을 나눠준다. 결과적으로 벡터들은 원점으로부터 길이가 1인 unit sphere 위에 위치하게 된다. $\left(\mathrm{X_1}/\sqrt{\mathrm{S}}, \mathrm{X_2}/\sqrt{\mathrm{S}}, \cdots, \mathrm{X_N}/\sqrt{\mathrm{S}}\right)$ 그 후 $F(r) = r^P = u \Leftrightarrow F^{-1}(u) = r = u^{\frac{1}{P}} = d$ 의 관계로부터 각각의 자료점에 d만큼을 곱해줘서 기존 단위벡터였던 점을 크기가 d인 점으로 만들 수 있다. 즉 자료점은 $\left(d * \mathrm{X_1}/\sqrt{S}, d * \mathrm{X_2}/\sqrt{\mathrm{S}}, \cdots, d * \mathrm{X_N}/\sqrt{\mathrm{S}}\right)$ 로 구성되어진다. 결과적으로 점은 unit sphere 내부에 위치하게 되며 uniform 하게 분포되어 있을 것이다.

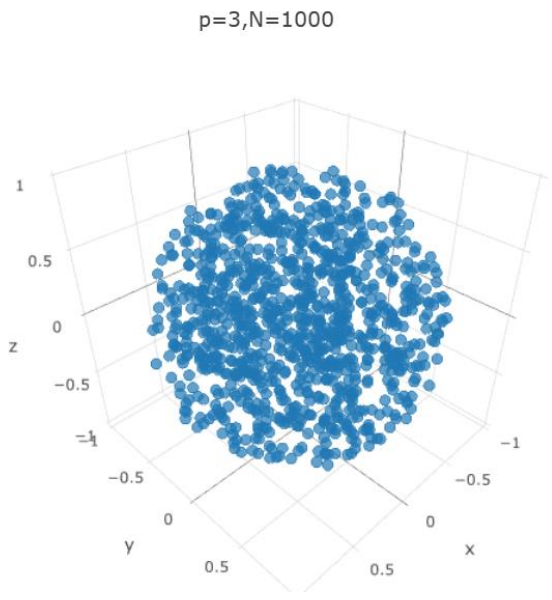
발생시킨 데이터를 p, N일 때를 생성해 그래프를 그려본 결과는 아래와 같다. p를 고정한 상태로 N (데이터 수)을 변화시켜봤을 때 값이 커질수록 해당 공간의 밀도가 높아지고, 더욱 많은 공간을 설명할 수 있게 된다. 반대로 N을 고정한 상태로 p를 변화시킨다면 차원이 커질수록 밀도가 감소한다는 것을 볼 수 있다.

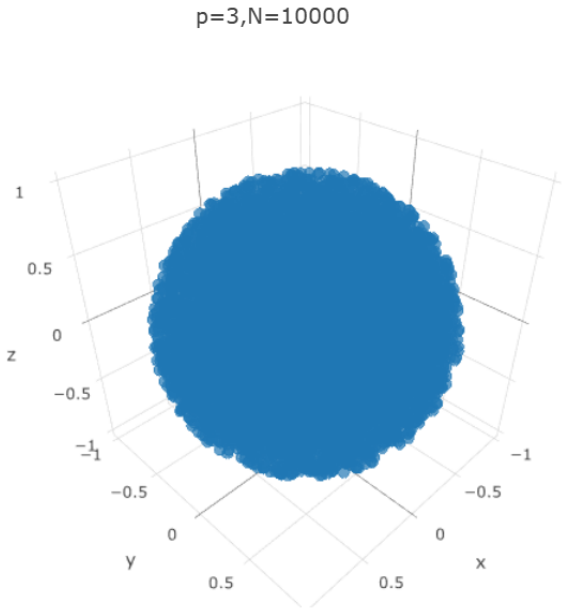
[2차원(p=2) 에서 발생한 데이터의 분포]





[3차원(p=3) 에서 발생한 데이터의 분포]





Question 3.

Compare the result from Question1 & Qeustion 2.

1) N = 100

	Equation	Simulation
P = 5	0.3697	0.3678
P = 10	0.6080	0.6063
P = 20	0.7798	0.7799

2) N = 1000

	Equation	Simulation
P = 5	0.2334	0.2319
P = 10	0.4831	0.4845
P = 20	0.6951	0.6970

3) N = 10000

	Equation	Simulation
P = 5	0.1473	0.1502
P = 10	0.3838	0.3831
P = 20	0.6195	0.6179

Simulation 수를 1000 으로 지정한 후 p(차원 수), N(데이터 수)를 변화시켜가면서 원점에서 데이터까지의 거리를 계산해 본 결과는 위와같다. 1번 문제의 공식으로부터 값을 대입해서 구할 수 있었고, 2번에서 난수를 생성하여 값을 구해 보았을 때와 비교해보았다. 위 표와 같이 값이 거의 같은 것을 알 수 있다.

또한 N 값이 고정되어 있을 때 차원이 증가할수록 원점과의 거리가 점점 멀어진다는 것을 알 수 있고, p값이 고정되어 있을 때 N이 커진다면 그만큼 거리가 더 가까워지고 있다는 것을 알 수 있다. 따라서 충분한 데이터의 수가 있다면 차원이 커지더라도 큰 문제가 발생하지 않을 것이다. 하지만 p가 클 때, N이 작은 경우는 심각한 문제가 될 수 있다.

Ⅲ. Discussion

이번 과제를 하며 지도학습 중 분류 문제에 사용하는 대표적인 기법인 KNN (K-Nearest Neighber) 방법과 KNN에서 발생할 수 있는 차원의 저주(Curse of Dimensionality) 문제에 대해서 이해하게 되었다.

원점으로부터의 거리의 median 값을 구하기 위해서 문제 1번에서는 공식을 유도해 보았고, 문제 2번에서는 데이터를 생성해보며 p(차원)가 커질 때 N(데이터수)가 작다면 원점으로부터의 거리가 멀어져서 차원의 저주 문제가 발생할 수 있다. 즉 점점 부피가 커지게 되면서 데이터가 unit sphere의 표면에 위치하게 될 것이다. 따라서 KNN 방법론을 사용하기 위해서는 차원의 수가 커질 때 주의가 필요하다.

Ⅳ. Appendix – R code

```
devtools::install_github("AckerDWM/gg3D")
library("gg3D")
library(ggplot2)
library(plotly)

### 1번 공식
    myequ = function(n,p){
      return( (1 -(1/2)^(1/n))^(1/p) )
    }

### 2번 함수 - My Function
    myftn = function(n,p){
      ### 1. 데이터수 n, 차원 p 에서 단위가 1 인 벡터 만들기
      mymat = matrix( rnorm(n*p,0,1), nrow = n, ncol = p )
      mymat = mymat / apply(mymat, 1, function(x) { sqrt(sum(x^2)) })
```

```
    ### 2. u ~ unif(0,1) 생성해서 위에서 생성한 단위벡터들의 값들에 u^(1/p) 곱해주기
    # (그럼 그 벡터들의 크기가 됨)
    mymat * (runif(n,0,1))^(1/p)
  }

### Graph
  # p = 2
  ggplot(data.frame( myftn(1000,2) ) , aes(x=X1, y=X2)) +
    geom_point(color = "sky blue") + theme_bw() + ggtitle("p=2,N=1000")
  ggplot(data.frame( myftn(10000,2) ) , aes(x=X1, y=X2)) +
    geom_point(color = "sky blue") + theme_bw() + ggtitle("p=2,N=10000")

  # p = 3
  mydata = data.frame( myftn(1000,3) )
  plot_ly(x=mydata$X1, y=mydata$X2, z=mydata$X3, type="scatter3d",
          mode="markers",colors = "sky blue", size = 3) %>% layout(title = "p=3,N=1000")
  mydata = data.frame( myftn(10000,3) )
  plot_ly(x=mydata$X1, y=mydata$X2, z=mydata$X3, type="scatter3d",
          mode="markers",colors = "sky blue", size = 3)%>% layout(title = "p=3,N=10000")

### 3번 - Compare  if n = 500, p = 3
  # equation
  c(myequ(100,5), myequ(100,10), myequ(100,20))

  # generate sample
  temp = c()
```

```
for ( i in 1: 1000) {
        temp[i] = min( apply(myftn(100,20),1,function(x){ sqrt(sum(x^2)) }) )
    }
    round(median(temp),4)
```