# hmrk 5 Beimnet Taye

2023-02-26

## P1

- 1.1: This is not reasonable to assume since taller kids are probably in a higher grade level due to underlying age differences and thus have a higher reading level. So without age information these two variables would seem dependent.
- 1.2: This is reasonable to assume due to the reasons listed above. Given a kid”s age you can probably infer their height and reading level. So given age you don”t also need that person”s reading level to infer their height.
- 1.3: This is reasonable to assume. I don”t see how age and zip code can be related. Zip codes probably contain a wide variety of ages thus given the probability of being in a certain age interval given a zip code is probably equal to probability of that age interval.
- 1.4: This is plausible since zip codes can be differential in terms of income. Higher income can mean better nutrition, which would shift the height distribution towards higher values, and better educational resources, which would shift the reading distribution to higher levels. Thus given the zip code you can probably infer the height and reading levels of an individual making them conditionally independent.

## P2

```r
# Logistic function
logistic <- function(x, l=1,e=0){

  1/(1+exp(-l*(x-e)))

}
#DGP
cancer_DGP <- function(n) {
  tibble(
    age = rlnorm(n, 3.584, 0.434),
    smoke = rbernoulli(n, p = case_when(
                              age %in% 25:64  ~ .145,
                              age %in% 0:10 ~ .00000000001,
                              TRUE ~ .08)
                       ),
    gene = runif(n,0,100),
    lung = rbernoulli(n, p = (logistic(gene,.5,50)+logistic(age,.1,60)+smoke)/9),
    skin = rbernoulli(n, p = (logistic(gene, .5,50)+logistic(age,.1,35))/7)
  )
}
```

- Age: I chose a logistic normal distribution to eliminate any negative ages while trying to create a right skewed distribution. Age is independent of gene since it is only a function of time and not any of your genes.

- Gene: I chose a uniform distribution over the interval [0,100] since I think there is an equal chance of acquiring a random mutation. This is independent of age since overall your genes don"t typically change with age.
- Smoke: I chose a Bernoulli distribution conditioned on age since it is a binary variable with how it was written. Age is an input with smoking more likely to occur given certain intervals of age (Interval prevalence data taken from CDC). I have smoking as conditionally independent with Lung and Skin given age.
- Lung: I chose a Bernoulli distribution conditioned on age and gene which are transformed using a logistic function to output probabilities that are weighted towards higher ages and higher gene scores respectively. Lung is also conditioned on smoking status with smokers contributing to a higher probability of lung cancer. Lung and skin are conditionally independent given age and gene.
- Skin: I chose a Bernoulli distribution conditioned on age and gene which are transformed using a logistic function to output probabilities that are weighted towards higher ages and higher gene scores respectively. I adjusted the age distribution logistic to scale up at a younger age than that of the lung variable since skin cancer can occur at an earlier age than lung cancer. Skin cancer also is of higher prevalence in general so I made sure it occurs more in this DGP than lung cancer. As stated above, lung and skin are conditionally independent given age and gene.

```r
# Testing DGP
test_prob <- cancer_DGP(10000) %>%
  filter(age > 60, gene > 50) %>%
  count(skin == TRUE) %>%
  mutate(p = n/sum(n))
test_prob
```

```
## # A tibble: 2 x 3
##   `skin == TRUE`      n     p
##   <lgl>           <int> <dbl>
## 1 FALSE             457 0.728
## 2 TRUE              171 0.272
```

```r
test_prob2 <- cancer_DGP(10000) %>%
  filter(age > 60, smoke == TRUE, gene > 50) %>%
  count(lung) %>%
  mutate(p = n/sum(n))
test_prob2
```

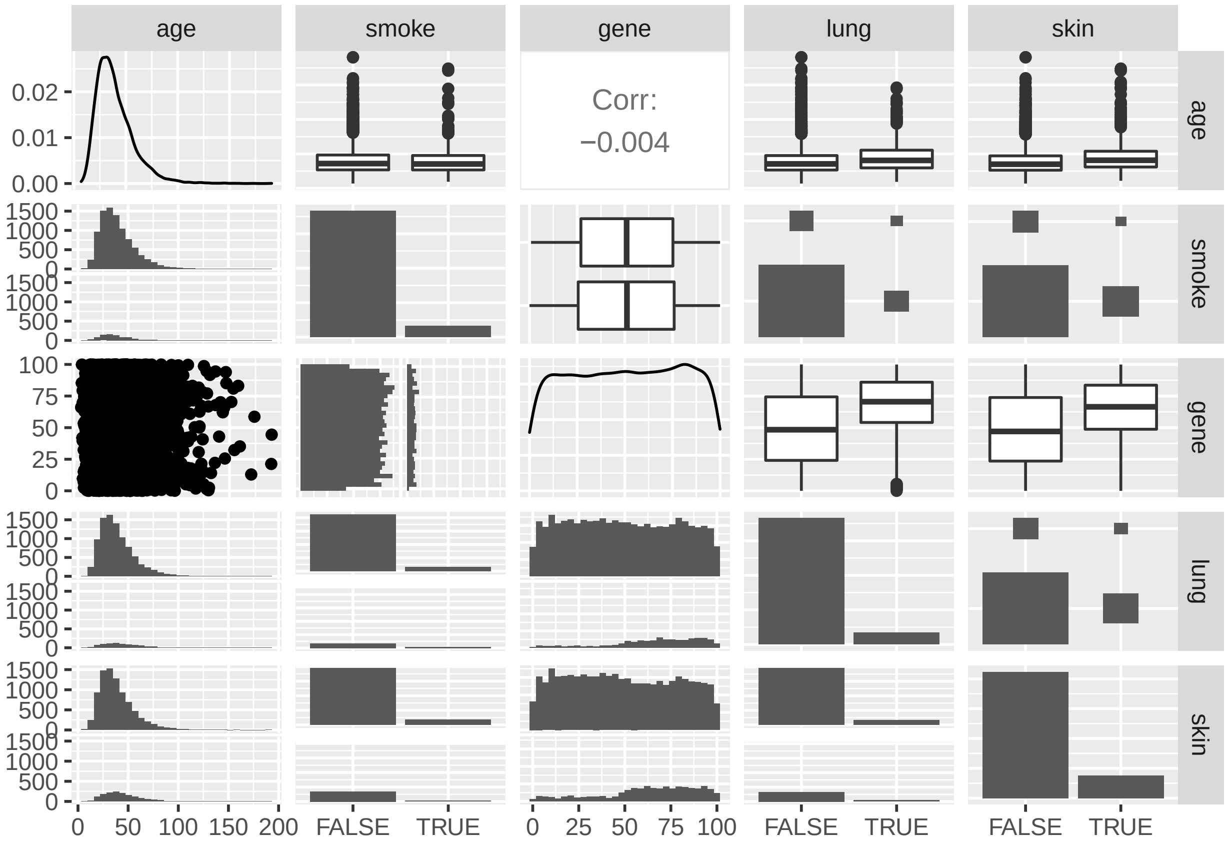
```
## # A tibble: 2 x 3
##   lung      n     p
##   <lgl> <int> <dbl>
## 1 FALSE    38 0.717
## 2 TRUE     15 0.283
```

```r
# Plots

plotsDGP <- cancer_DGP(10000) %>%
  ggpairs()
```

```r
plotsDGP
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



- One surprising thing is that the age distribution of smokers and non-smokers does not differ all that much. The same thing can be said with the age distribution between those with lung cancer and those without. I expected the distribution mass to shift right as I thought I had coded that smokers and people with Lung cancer should be of higher age. Upon closer inspection of the marginal distribution of age this could be due to there not being enough occurrences of older people.I could adjust the logistic normal distribution somehow to spread it out to increase the probability of older ages.

##P3

**Probability function**

```
prob <- function(df, ...) {
  final <- df %>%
  count(...) %>%
    mutate(p = n/sum(n))
    pull(final[2,3])
}
```

**A**

```r
DGP_A <- function(n){
  tibble(
  X = runif(n),
  W = rbernoulli(n),
  Y = rnorm(n,X+W,1+X*W)
)
}
### Independent: X and W

#P(X)
prob_ax <- prob(DGP_A(10000), X < .5)
prob_ax
```

```
## [1] 0.5068
```

```r
#P(X|W)
prob_axgw <- DGP_A(10000)%>%
  filter(W == 1) %>%
  count(X < .5) %>%
  mutate(p = n/sum(n))
  pull(prob_axgw[2,3])
```

```
## [1] 0.4922954
```

```r
# Probabilities are equal so X and W are independent

### conditionally independent: none

#P(Y|X)
prob_aygw <- DGP_A(10000)%>%
    filter(X < .5) %>%
    count(Y < .5) %>%
    mutate(p = n/sum(n))
  pull(prob_aygw[2,3])
```

```
## [1] 0.4337159
```

```r
#P(Y|X,W)
  prob_aygwx <- DGP_A(10000)%>%
    filter(W == 1,X< .5) %>%
    count(Y < .5) %>%
    mutate(p = n/sum(n))
  pull(prob_aygwx[2,3])
```

```
## [1] 0.2775842
```

```
# The probabilities are not equal so Y and W are not conditionally independent on X.
```

### B

```
DGP_B <- function(n){
  tibble(
    X = runif(n),
    W = rbernoulli(n,p= logistic(X)),
    Y = rnorm(n,X+W,1+X*W)
  )
}

### Independent: none

#P(W)
prob_bw <- prob(DGP_B(10000),W == 1)
prob_bw
```

```
## [1] 0.6245
```

```
#P(W|X)
prob_bwgx <- DGP_B(10000)%>%
  filter(X < .5) %>%
  count(W == 1) %>%
  mutate(p = n/sum(n))
pull(prob_bwgx[2,3])
```

```
## [1] 0.5480944
```

```
# The probabilities are not equal so X and W are not independent.

### conditionally independent: none

# P(Y|X)
prob_bygx <- DGP_B(10000)%>%
  filter(X < .5) %>%
  count(Y < 2) %>%
  mutate(p = n/sum(n))
pull(prob_bygx[2,3])
```

```
## [1] 0.8304739
```

```
# P(Y|X,W)
prob_bygxw <- DGP_B(10000)%>%
  filter(W == 1 & X < .5) %>%
  count(Y < 2) %>%
  mutate(p = n/sum(n))
pull(prob_bygxw[2,3])
```

```
## [1] 0.712953
```

```
# Probabilities are not equal so Y and W are not conditionally independent on X.
```

### C

```
DGP_C <- function(n){
  tibble(L = runif(n,-1,0),
         U = runif(n),
         M = runif(n,L,U),
         X = rnorm(n, L),
         Y = rnorm(n, U)
  )
}
### Independent: L and U

#P(L)
prob_cl <- prob(DGP_C(10000), L < -.5)
prob_cl
```

```
## [1] 0.5004
```

```
#P(L|U)
prob_clgu <- DGP_C(10000)%>%
  filter(U < .5) %>%
  count(L < -.5) %>%
  mutate(p = n/sum(n))
pull(prob_clgu[2,3])
```

```
## [1] 0.5056492
```

```
# The probabilities are the same so  L and U are independent.

### conditionally independent: M and X given L, M and Y given U

#P(M|L)
prob_cmgl <- DGP_C(10000) %>%
  filter(L < -.5) %>%
  count(M <.5) %>%
  mutate(p = n/sum(n))
pull(prob_cmgl[2,3])
```

```
## [1] 0.9239766
```

```
#P(M|L,X)
prob_cmglx <- DGP_C(10000) %>%
  filter(L < -.5, X < -.5) %>%
  count(M <.5) %>%
  mutate(p = n/sum(n))
pull(prob_cmglx[2,3])
```

```
## [1] 0.9309429
```

```
# The probabilities are the same proving M and X are conditionally independent on L.
```

### D

```
DGP_D <- function(n) {
  tibble(
    X = runif(n),
    W = rbernoulli(n, logistic(X)),
    Y = rnorm(n, W, 1+W)
  )
}
### Independent: none

#P(Y)
prob_dy <- prob(DGP_D(10000), Y< 1)
prob_dy
```

```
## [1] 0.6219
```

```
#P(Y|W)
prob_dygw <- DGP_D(10000) %>%
  filter(W == 1) %>%
  count(Y < 1) %>%
  mutate(p = n/sum(n))
pull(prob_dygw[2,3])
```

```
## [1] 0.495218
```

```
# Probabilities are not equal thus Y and W are not independent.

### Conditionally independent: none

#P(Y|X)
prob_dygx <- DGP_D(10000) %>%
  filter(X < .5) %>%
  count(Y < 1) %>%
  mutate(p = n/sum(n))
pull(prob_dygx[2,3])
```

```
## [1] 0.6351324
```

```
#P(Y|X,W)
prob_dygxw <- DGP_D(10000) %>%
  filter(X < .5, W == 1) %>%
  count(Y < 1) %>%
  mutate(p = n/sum(n))
pull(prob_dygxw[2,3])
```

```
## [1] 0.4899329
```

```
# Probabilities are not equal thus Y and W are not conditionally independent on X.
```

## P4

### Sampling functions

```
# Method A
sample_na = function(n) {
  sigma = matrix(c(
    2, 1,
    1, 1
  ), nrow=2)
  rmvnorm(
    n,
    mean=c(0,0),
    sigma=sigma
  ) %>%
    as_tibble() %>%
    set_names(c("X","Y"))
}
# Method B
sample_nb = function(n) {
  tibble(
    Y = rnorm(n, mean=0, sd=1),
    X = rnorm(n, mean=Y, sd=1)
  )
}
# Method C
sample_nc = function(n) {
  tibble(X = rnorm(n, mean=0, sd=sqrt(2)),
         Y = rnorm(n, mean=X/2, sd=1/sqrt(2))
  )
}
```

### Testing Joint Probabilites

```
# P(X>0,Y>0)
tibble(
Method_A=prob(sample_na(10000), X > 0 & Y > 0),
Method_B=prob(sample_nb(10000), X > 0 & Y > 0),
Method_C=prob(sample_nc(10000), X > 0 & Y > 0)
)
```

```
## Warning: The ‘x‘ argument of ‘as_tibble.matrix()‘ must have unique column names if ‘.name_repair‘ is
## Using compatibility ‘.name_repair‘.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was generated.
## # A tibble: 1 x 3
##   Method_A Method_B Method_C
##      <dbl>    <dbl>    <dbl>
## 1    0.378    0.376    0.381
```

```
# P(X<0,Y<0)
tibble(
Method_A = prob(sample_na(10000), X < 0 & Y < 0),
Method_B = prob(sample_nb(10000), X < 0 & Y < 0),
Method_C = prob(sample_nc(10000), X < 0 & Y < 0)
)
```

```
## # A tibble: 1 x 3
##   Method_A Method_B Method_C
##      <dbl>    <dbl>    <dbl>
## 1    0.370    0.372    0.378
```

```
# P(X<1,Y<1)
tibble(
Method_A = prob(sample_na(10000), X < 1 & Y < 1),
Method_B = prob(sample_nb(10000), X < 1 & Y < 1),
Method_C = prob(sample_nc(10000), X < 1 & Y < 1)
)
```

```
## # A tibble: 1 x 3
##   Method_A Method_B Method_C
##      <dbl>    <dbl>    <dbl>
## 1    0.714    0.705    0.707
```

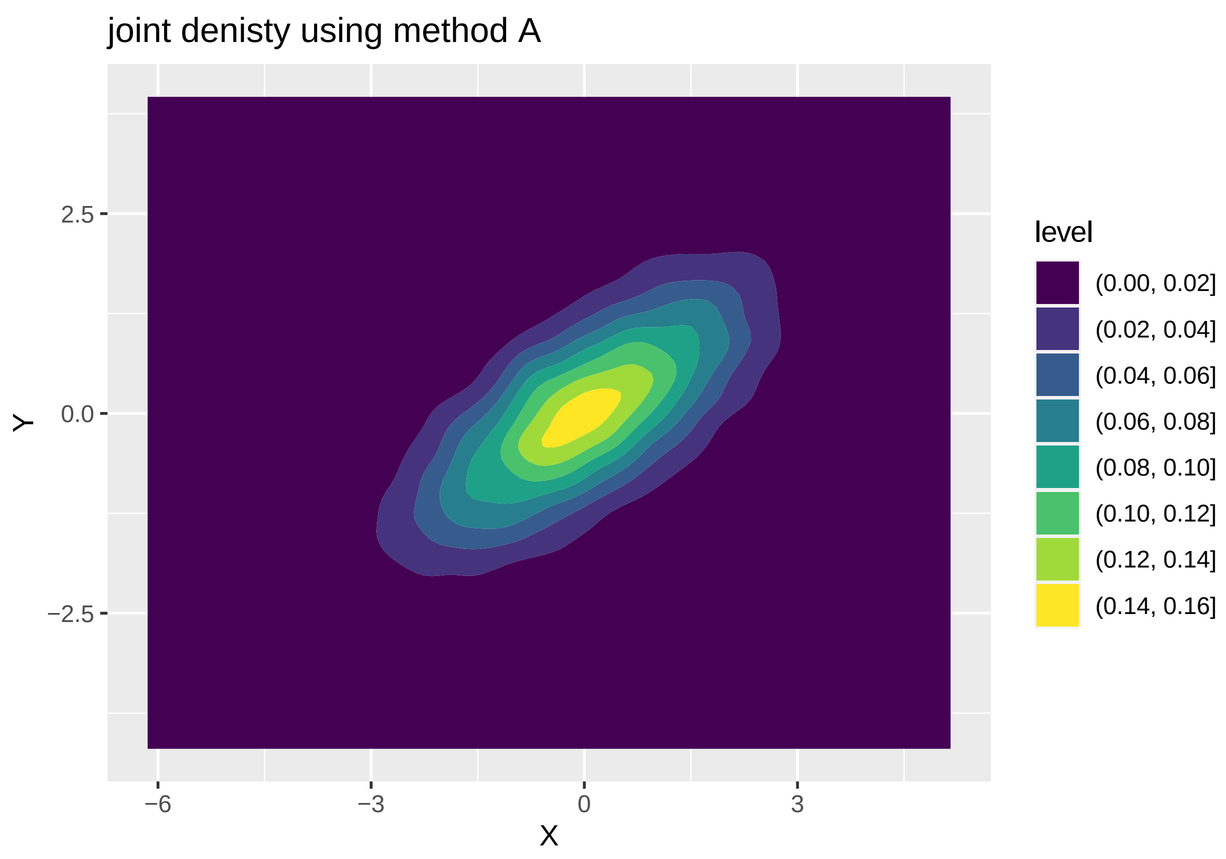
```
# P (X<1,Y<0)
tibble(
Method_A = prob(sample_na(10000), X < 1 & Y < 0),
Method_B = prob(sample_nb(10000), X < 1 & Y < 0),
Method_C = prob(sample_nc(10000), X < 1 & Y < 0)
)
```

```
## # A tibble: 1 x 3
##   Method_A Method_B Method_C
##      <dbl>    <dbl>    <dbl>
## 1    0.475    0.471    0.472
```

### Joint Density Plots

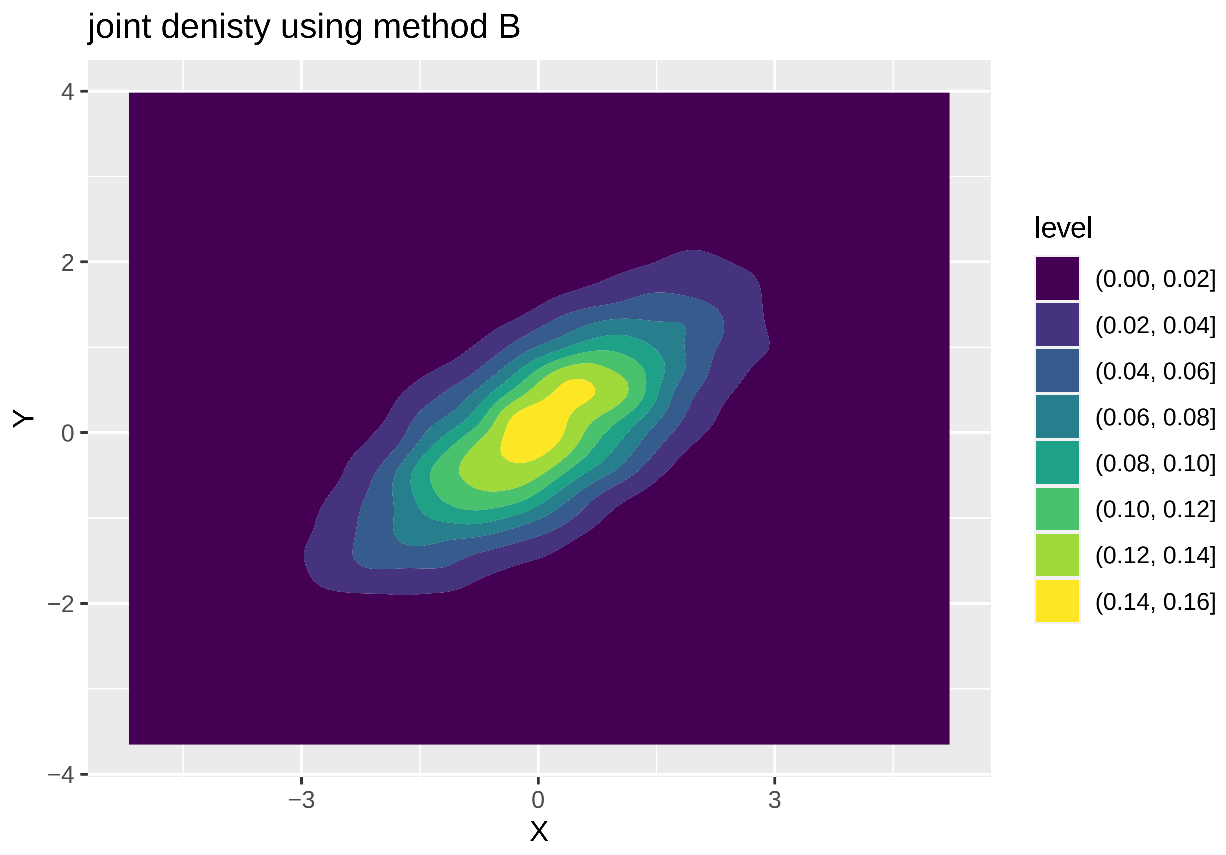
```
dena <- sample_na(10000) %>%
  ggplot(aes(x=X, y=Y)) +
  geom_density2d_filled() +
  labs(title = "joint denisty using method A")

dena
```



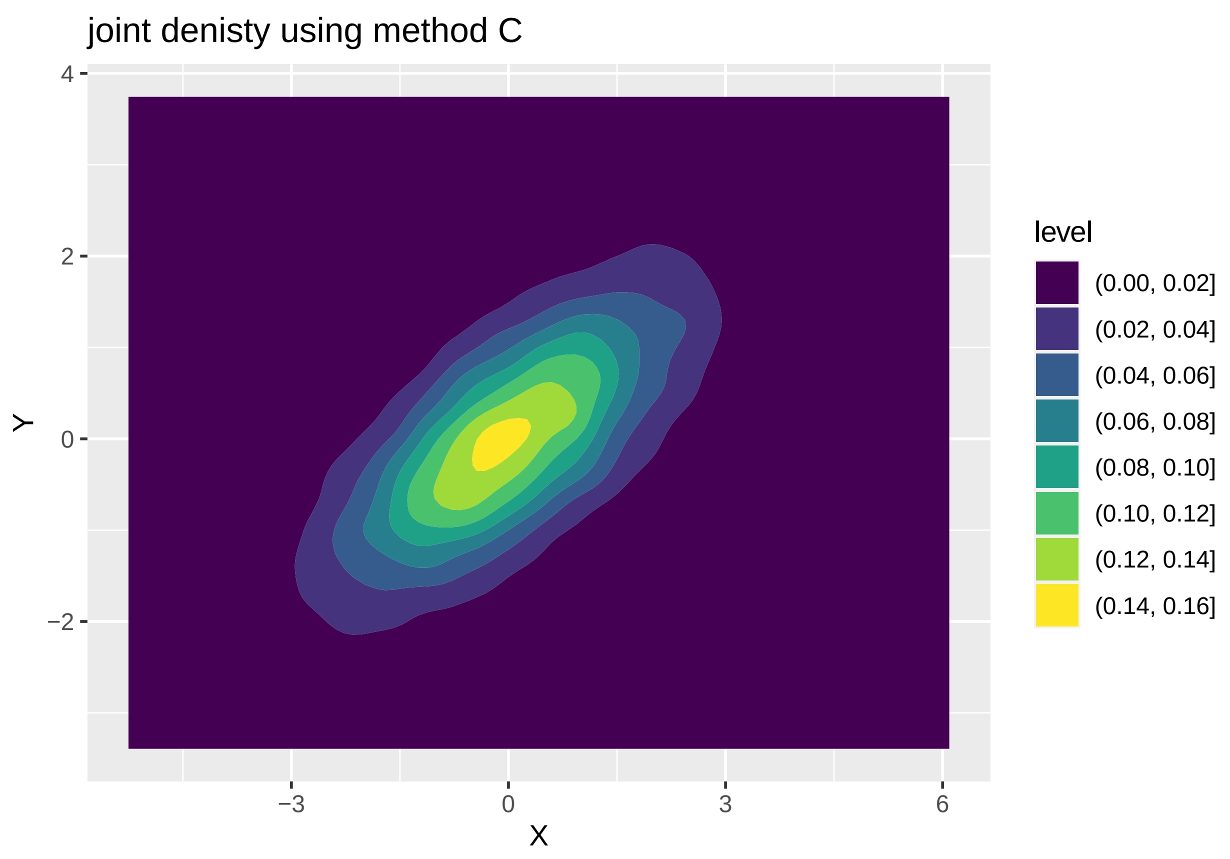
```r
denb <- sample_nb(10000) %>%
  ggplot(aes(x=X, y=Y)) +
  geom_density2d_filled() +
  labs(title = "joint denisty using method B")

denb
```



```
denc <- sample_nc(10000) %>%
  ggplot(aes(x=X, y=Y)) +
  geom_density2d_filled() +
  labs(title = "joint denisty using method C")

denc
```

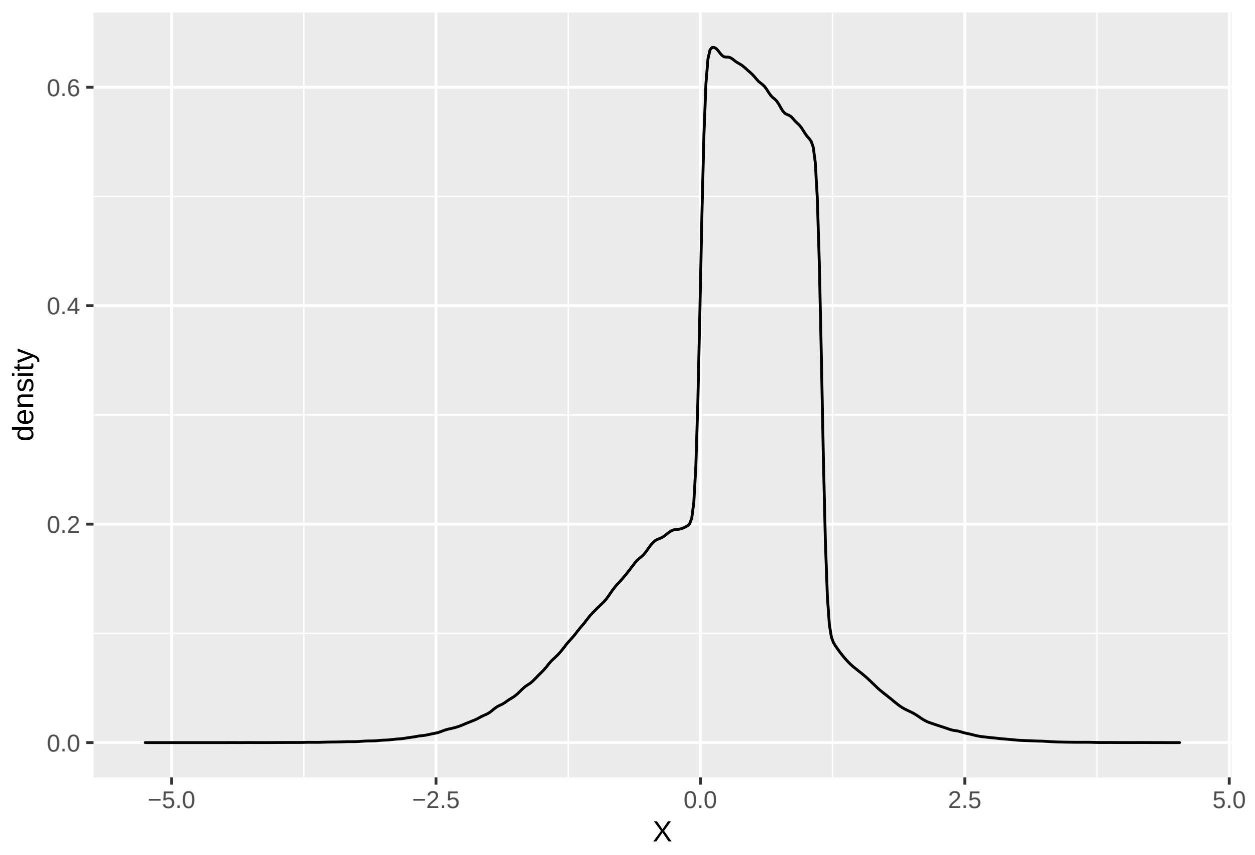


- Given that all three have the same joint density distribution when plotted and we can calculate the same probabilities of an event across all three of these sampling methods we can say that they each sample from the same joint distribution.

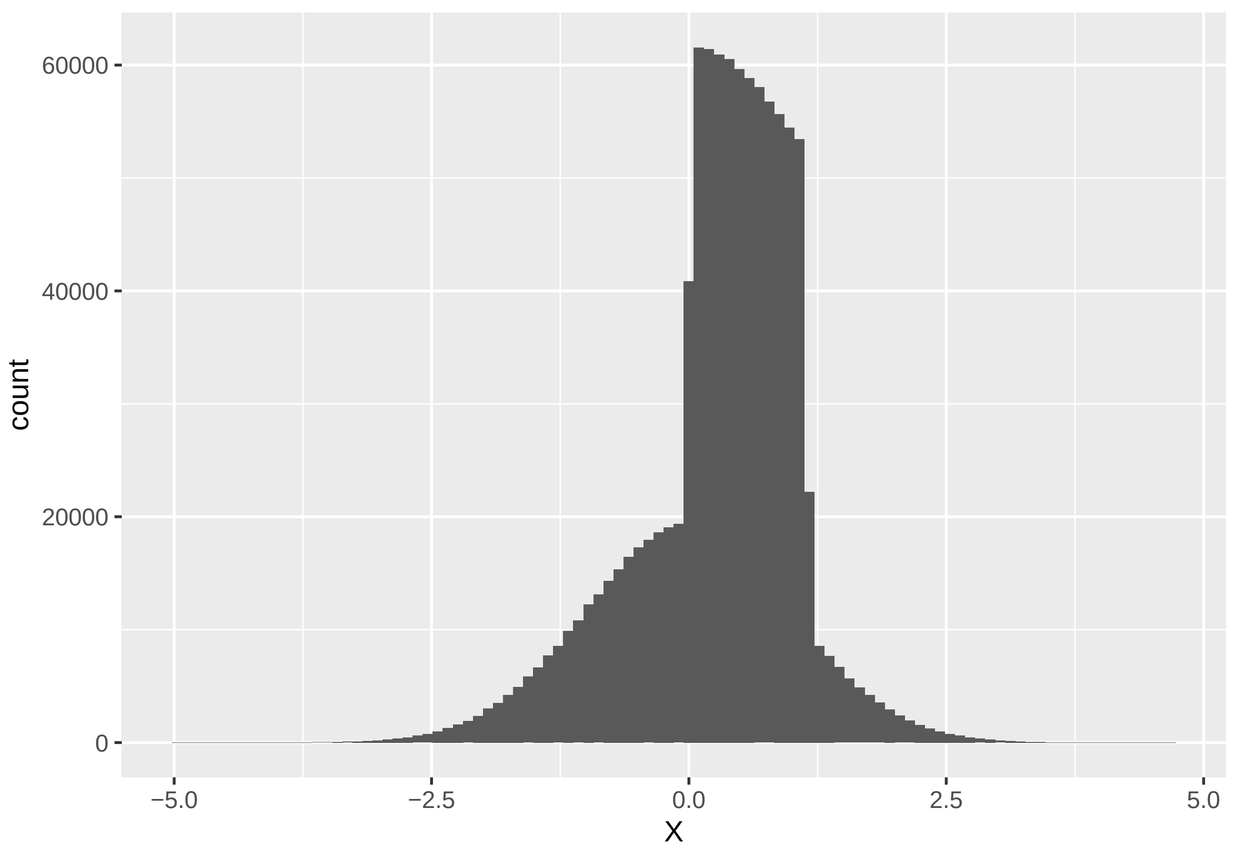
## P5

```
distrib <- function(n) {
  tibble(
    S = rbernoulli(n),
    N = rnorm(n),
    U = runif(n,0,1.15),
    X = ifelse(S,N,U)
  )
}

densityc <- distrib(1000000) %>%
  ggplot() +
  geom_density(aes(x=X))
densityc
```



```
histogramc <- distrib(1000000) %>%
  ggplot() +
  geom_histogram(aes(x=X), bins = 100)
histogramc
```


60000
40000
20000
0
count
−5.0
−2.5
0.0
2.5
5.0
X