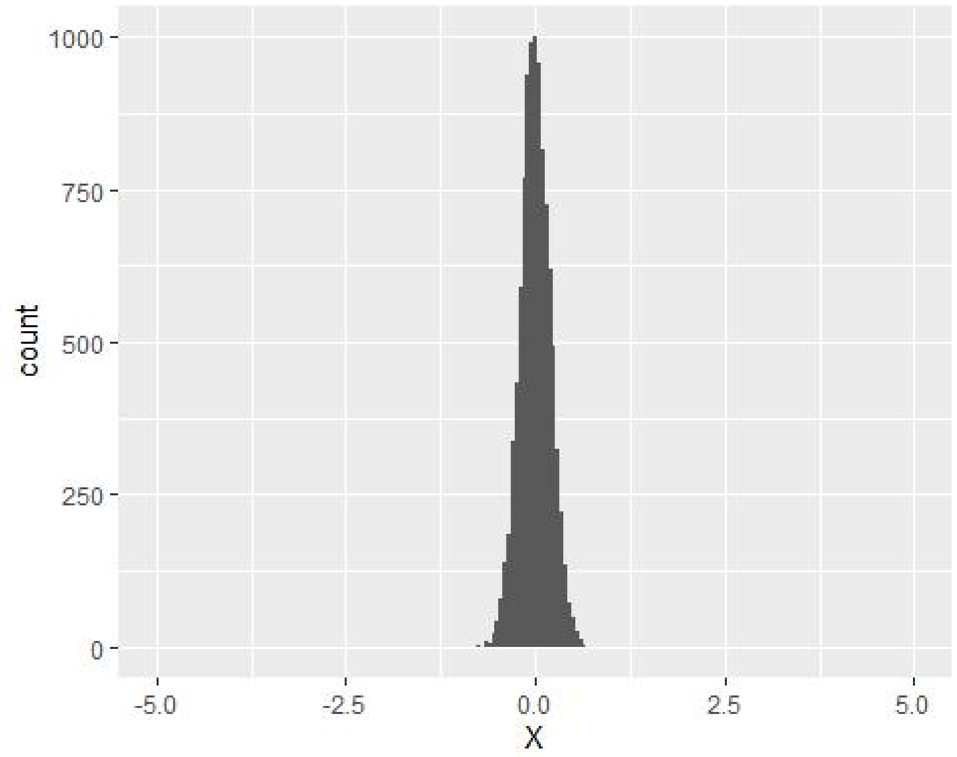
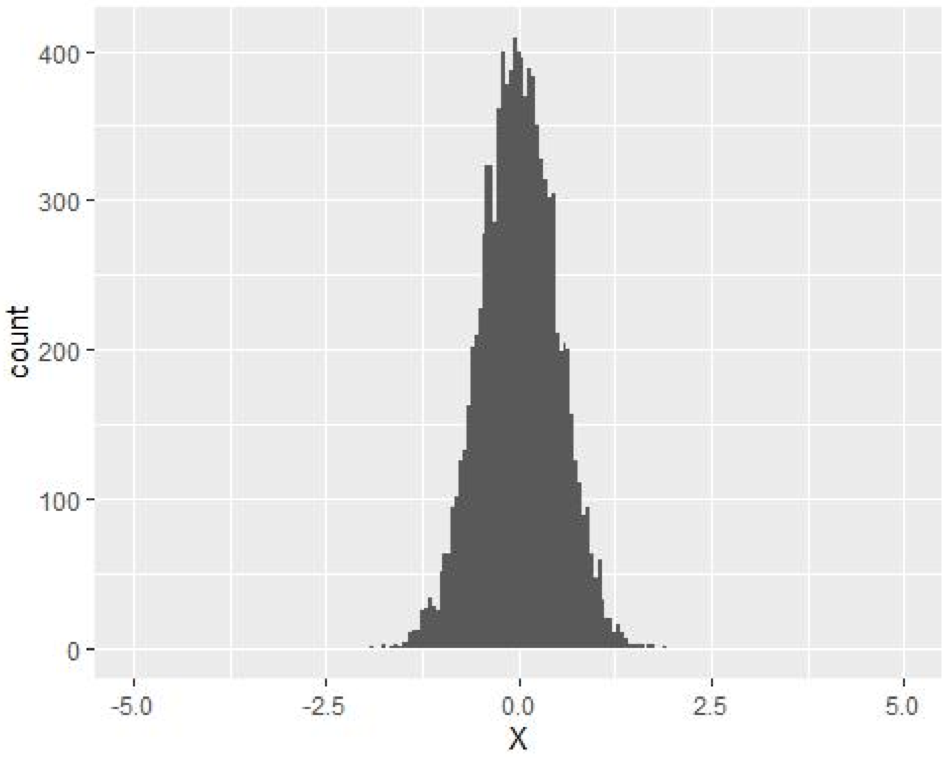
```
setwd("C:\\Users\\DELL\\Desktop\\CSC587\\datamining-main\\Rscripts")
data.file1 <- file.path('data', 'Su_raw_matrix.txt')
data.file2 <- file.path('data', 'diabetes_train.csv')
data.file3 <- file.path('data', 'titanic.csv')

#1. Use ”Su_raw_matrix.txt” for the following questions (30 points).
# (a)Use read.delim function to read Su_raw_matrix.txt into a variable c
alled su.
su <- read.delim(data.file1, header = TRUE)
# (b) Use mean and sd functions to find mean and standard deviation of L
iver_2.CEL column
mean(su[["Liver_2.CEL"]])

## [1] 241.8246

sd(su[["Liver_2.CEL"]])

## [1] 1133.352

# (c) Use colMeans and colSums functions to get the average and total va
lues of each column.
colMeans(su)

##      Brain_1.CEL      Brain_2.CEL Fetal_brain_1.CEL Fetal_brain_2.C
EL
##         204.9763         315.0924          198.3439          267.6551
## Fetal_liver_1.CEL Fetal_liver_2.CEL      Liver_1.CEL       Liver_2.C
EL
##         209.8722         399.1482          160.8558          241.8246

colSums(su)

##      Brain_1.CEL      Brain_2.CEL Fetal_brain_1.CEL Fetal_brain_2.C
EL
##          2588031          3978357           2504290           3379413
## Fetal_liver_1.CEL Fetal_liver_2.CEL      Liver_1.CEL       Liver_2.C
EL
##          2649846          5039645           2030966           3053278

#2. Use rnorm(n, mean = 0, sd = 1) function in R to generate 10000 numbe
rs for the following (mean, sigma) pairs
#and plot histogram for each, meaning you need to change the function pa
rameter accordingly.
#Then comment on how these histograms are different from each other and
state the reason. (20 points)
#(a) mean=0, sigma=0.2
sigma1 <- data.frame(X = rnorm(10000, mean = 0, sd = 0.2))
#(b) mean=0, sigma=0.5
sigma2 <- data.frame(X = rnorm(10000, mean = 0, sd = 0.5))
#Answer the sigma 0.5 is lower/shorter and wider.
#Please save your figures as image from RStudio.
#(Hint: to see the difference in plots you may need to set the xlim para
meter in plot function to c(-5,5))
# Start visualizing data using the ggplot2 package.
sigma1ggpot = ggplot(sigma1, aes(x = X)) + geom_histogram(binwidth = 0.0
5) + xlim(c(-5, 5))
sigma2ggpot = ggplot(sigma2, aes(x = X)) + geom_histogram(binwidth = 0.0
5) + xlim(c(-5, 5))

show(sigma1ggpot)

## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



```
show(sigma2ggpot)

## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```

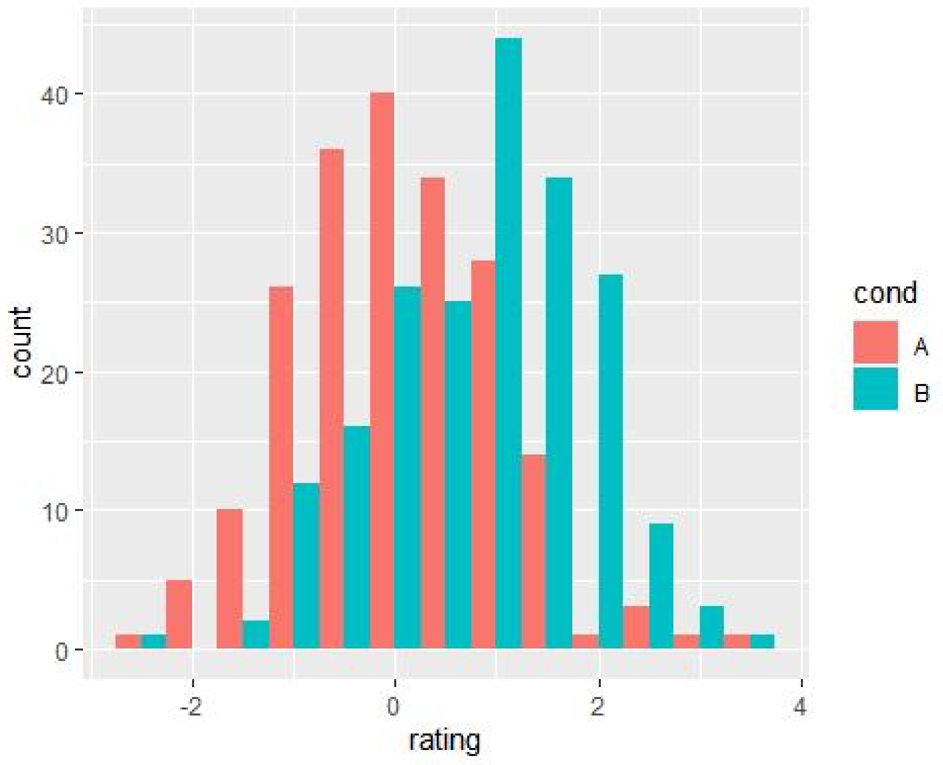


```
print("The sigma 0.5 is lower/shorter and wider. The reason for these di
fferences is that the standard deviation (sigma) controls the spread of
the distribution. A smaller sigma results in a narrower distribution, wh
ile a larger sigma leads to a wider distribution.")
```

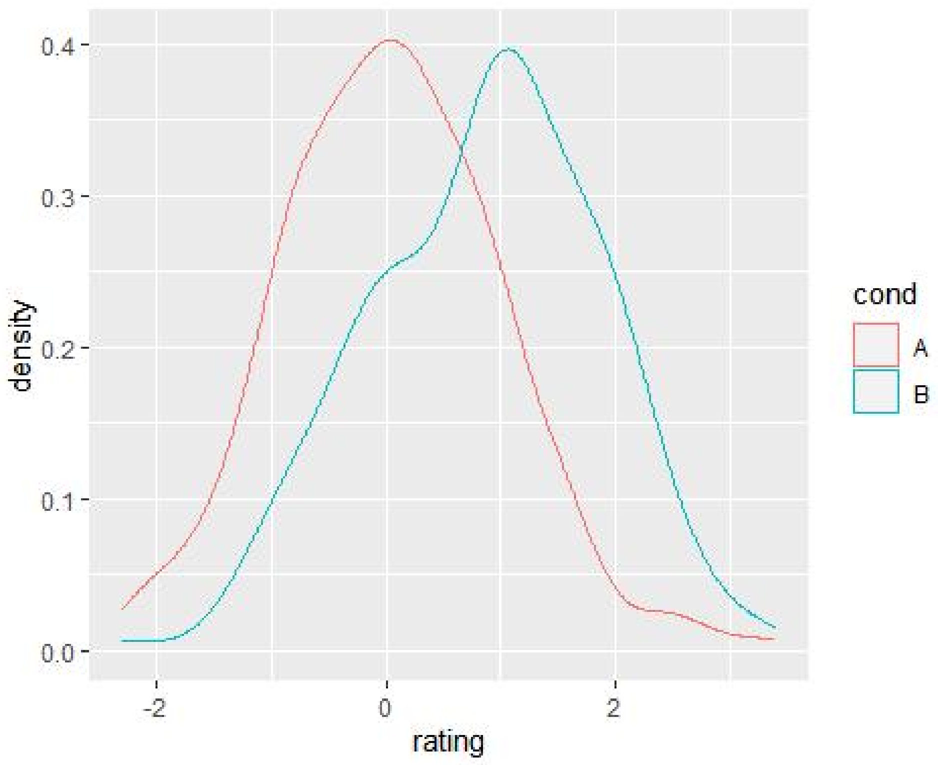
```
## [1] "The sigma 0.5 is lower/shorter and wider. The reason for these d
ifferences is that the standard deviation (sigma) controls the spread of
 the distribution. A smaller sigma results in a narrower distribution, w
hile a larger sigma leads to a wider distribution."
```

```
#3. Perform the steps below with ”dat” dataframe which is just a sample
data for you to observe how each plot function
#( 3b through 3e ) works. Notice that you need to have ggplot2 library i
nstalled on your system. Please refer slides how
#to install and import a library. Installation is done only once, but yo
u need to import the library every time you need it
#by saying library(ggplot2). Then run the following commands for questio
ns from 3a through 3e and observe how
#the plots are generated first. (20 points)
#(a) dat <- data.frame(cond = factor(rep(c("A","B"), each=200)), rating
= c(rnorm(200),rnorm(200, mean=.8)))
dat <- data.frame(cond = factor(rep(c("A","B"), each=200)), rating = c(r
norm(200),rnorm(200, mean=.8)))
#(b) # Overlaid histograms
#ggplot(dat, aes(x=rating, fill=cond)) + geom_histogram(binwidth=.5, al
pha=.5, position="identity")
t1 = ggplot(dat, aes(x=rating, fill=cond)) + geom_histogram(binwidth=.5,
 alpha=.5, position="identity")
show(t1)
```

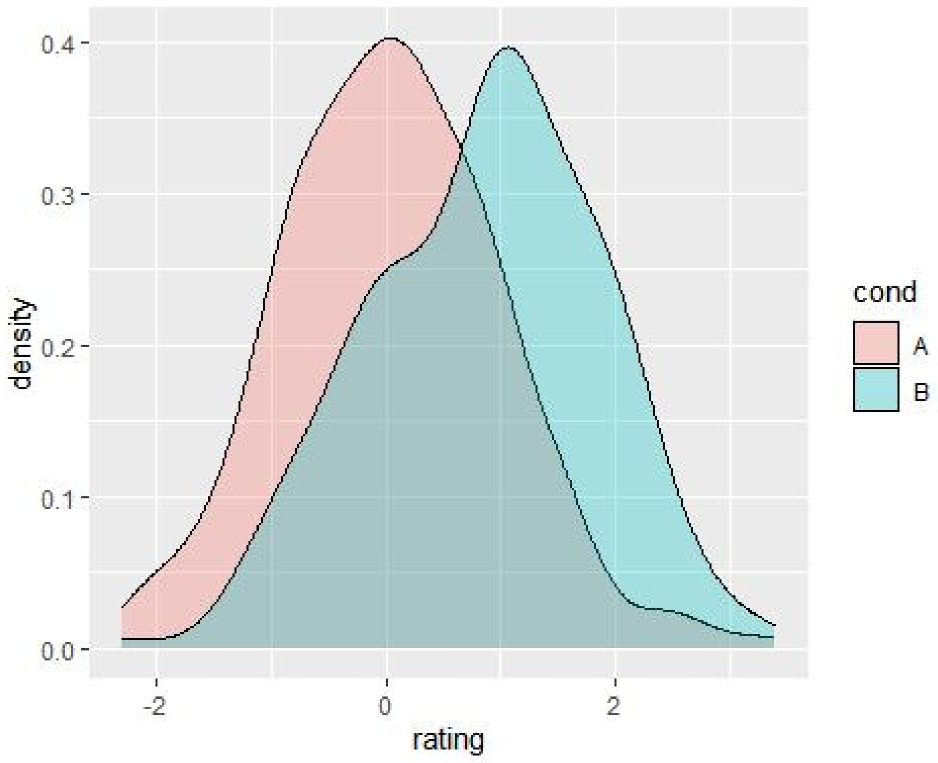
```
#(c) # Interleaved histograms
#ggplot(dat, aes(x=rating, fill=cond)) + geom_histogram(binwidth=.5, po
sition="dodge")
t2 = ggplot(dat, aes(x=rating, fill=cond)) + geom_histogram(binwidth=.5,
 position="dodge")
show(t2)
```



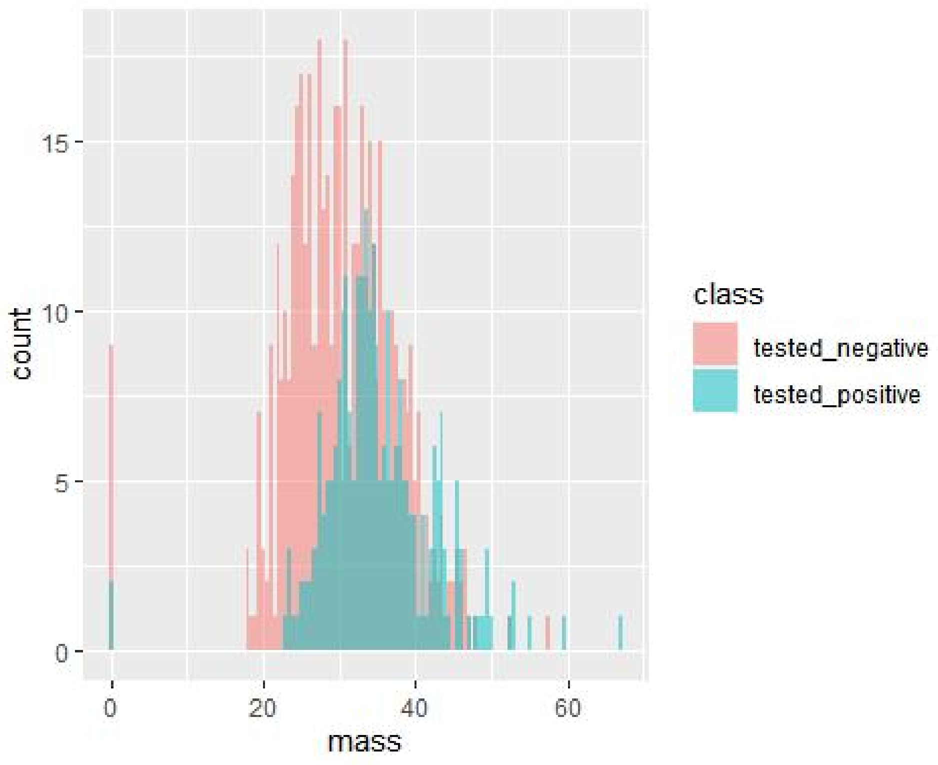
```
#(d) # Density plots
#ggplot(dat, aes(x=rating, colour=cond)) + geom_density()
t3 = ggplot(dat, aes(x=rating, colour=cond)) + geom_density()
show(t3)
```



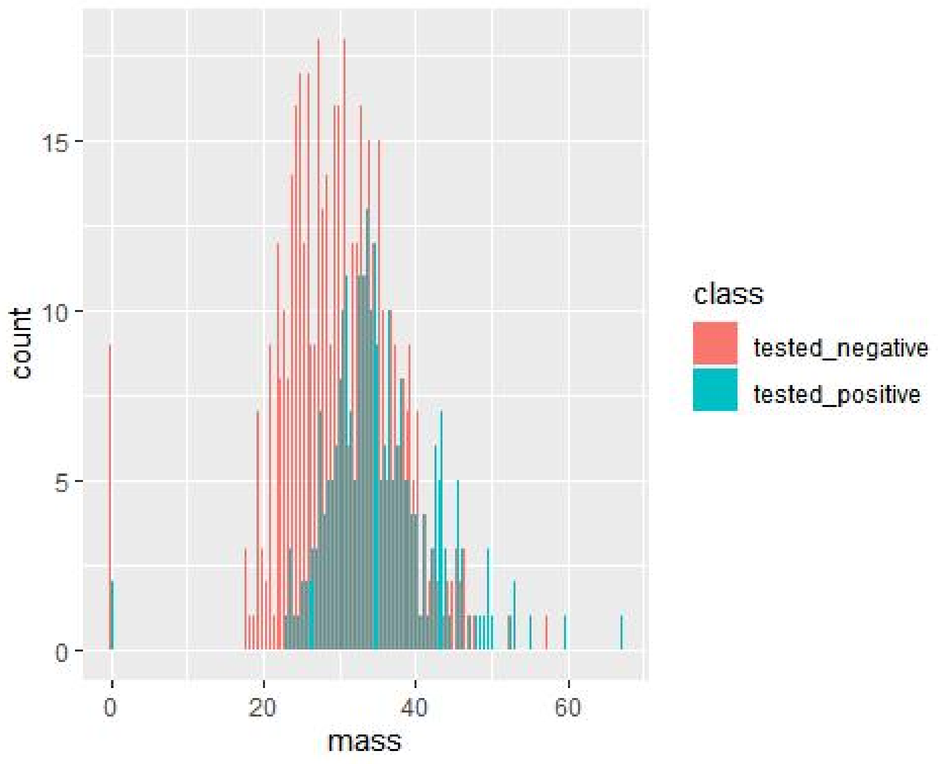
```
#(e) # Density plots with semitransparent fill
#ggplot(dat, aes(x=rating, fill=cond)) + geom_density(alpha=.3)
t4 = ggplot(dat, aes(x=rating, fill=cond)) + geom_density(alpha=.3)
show(t4)
```



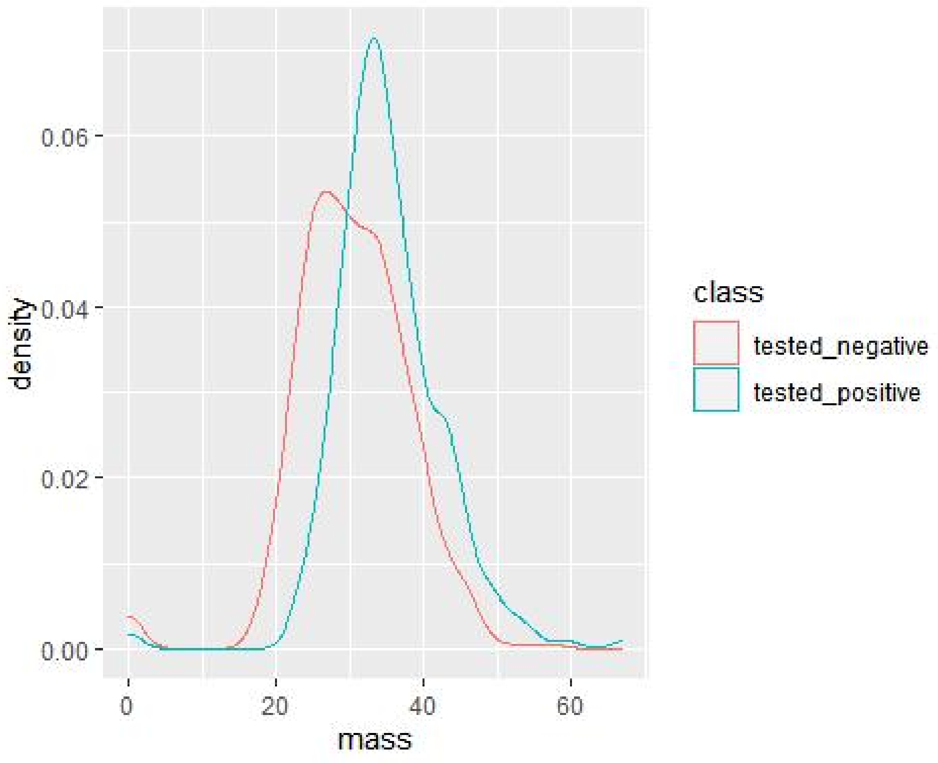
```
#(f) Read ”diabetes_train.csv” into a variable called diabetes and apply
 the same functions 3b through 3e for the
#mass attribute of diabetes and save the images. (Hint: instead of cond
above, use the class attribute to color
#your groups. When you have fill option, your plots should show same typ
e of chart for both groups in different
#colors on the same figure. Keep in mind that diabetes and dat are both
DataFrames)
diabetes <- read.csv(data.file2, header = TRUE, sep = ',')
p1 = ggplot(diabetes, aes(x=mass, fill=class)) + geom_histogram(binwidt
h=.5, alpha=.5, position="identity")
p2 = ggplot(diabetes, aes(x=mass, fill=class)) + geom_histogram(binwidt
h=.5, position="dodge")
p3 = ggplot(diabetes, aes(x=mass, colour=class)) + geom_density()
p4 = ggplot(diabetes, aes(x=mass, fill=class)) + geom_density(alpha=.3)

show(p1)
```



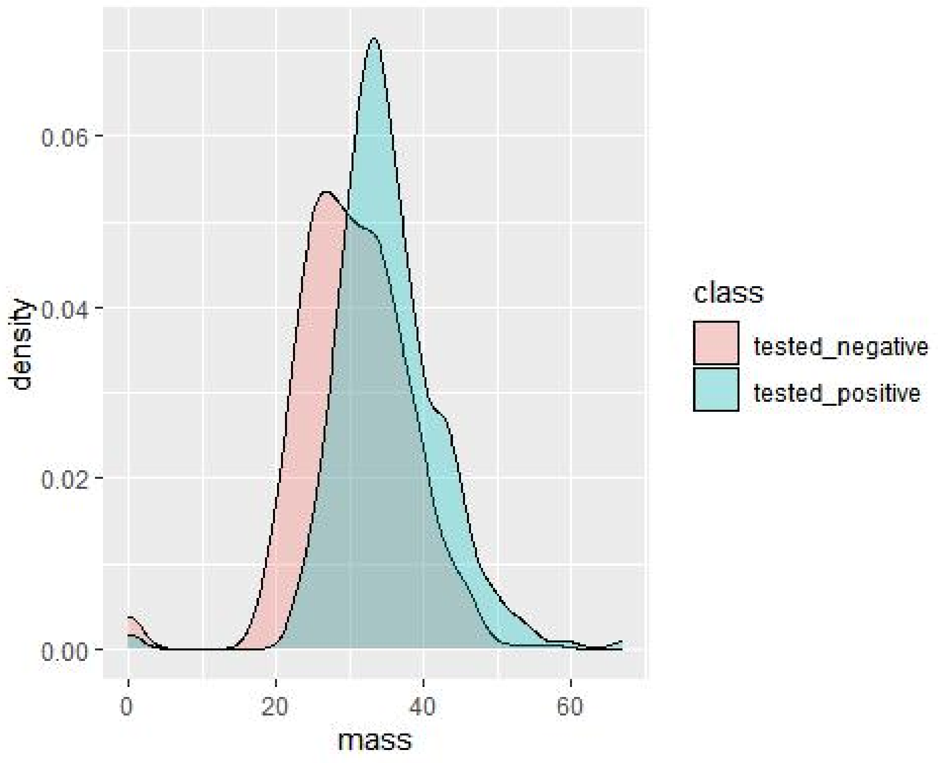
```r
show(p2)
```



```r
show(p3)
```


density
0.06
0.04
0.02
0.00
0
20
40
60
mass
class
tested_negative
tested_positive

```
show(p4)
```


density
0.06
0.04
0.02
0.00
0
20
40
60
mass
class
tested_negative
tested_positive

```r
#4. Read the titanic.csv file from DATA folder to a variable named passe
ngers and perform the following steps and
#explain the operation very briefly (20 points):
#passengers <- read.csv(data.file3, header = TRUE, sep = ',')
#(a) passengers \%>\% drop_na() \%>\% summary()
#passengers %>% drop_na() %>% summary()
print("This code drops/removes rows where any column contains a missing
value(NA) from passengers. Then make the summary of the passengers, typi
cally including measures like Min, 1st Qu., Median, Mean, 3rd Qu., and M
ax.")
```

```
## [1] "This code drops/removes rows where any column contains a missing
 value(NA) from passengers. Then make the summary of the passengers, typ
ically including measures like Min, 1st Qu., Median, Mean, 3rd Qu., and
Max."
```

```r
#(b) passengers \%>\% filter(Sex == "male")
#passengers %>% filter(Sex=="male")
print("This code filters the passengers data to include only rows where
the 'Sex' column is equal to 'male'.")
```

```
## [1] "This code filters the passengers data to include only rows where
 the 'Sex' column is equal to 'male'."
```

```r
#(c) passengers \%>\% arrange(desc(Fare))
#passengers %>% arrange(desc(Fare))
print("This code arranges the passengers data in descending order based
on the 'Fare' column.")
```

```
## [1] "This code arranges the passengers data in descending order based
 on the 'Fare' column."
```

```r
#(d) passengers \%>\% mutate(FamSize = Parch + SibSp)
#passengers %>% mutate(FamSize = Parch + SibSp)
print("This code adds a new column 'FamSize' with the value of column 'P
arch' plus the value of column 'SibSp' to the passengers data.")
```

```
## [1] "This code adds a new column 'FamSize' with the value of column '
Parch' plus the value of column 'SibSp' to the passengers data."
```

```r
#(e) passengers \%>\% group_by(Sex) \%>\% summarise(meanFare = mean(Far
e), numSurv = sum(Survived))
#passengers %>% group_by(Sex) %>% summarise(meanFare = mean(Fare), numSu
rv = sum(Survived))
print("This code groups the passengers data by the 'Sex' column. It then
 calculates the mean of 'Fare' and the sum of 'Survived' for each group
('Sex').")
```

```
## [1] "This code groups the passengers data by the 'Sex' column. It the
n calculates the mean of 'Fare' and the sum of 'Survived' for each group
 ('Sex')."
```

```r
#5. By using quantile(), calculate 10th,30th,50th,60th percentiles of sk
in attribute of diabetes data. (10 points)
skin <- with(diabetes, skin)
quantiles_skin <- quantile(skin, c(0.1, 0.3, 0.5, 0.6))
print(quantiles_skin)
```

```
## 10% 30% 50% 60%
##   0  10  23  27
```