Introduction to dplyr and ggplot2

Karen Mazidi

Explore dplyr

load dplyr and data

The dplyr package was designed to be efficient with large data but we will demonstrate the basic features with a smaller data set from package mlbench.

```
library(dplyr)
library(mlbench)
data("PimaIndiansDiabetes2")
```

tbl

A tbl "tibble" is a data frame with enhanced features. Now when we type the name at the console we get a neater display of our data, one page at a time.

```
df <- tbl_df(PimaIndiansDiabetes2)
rm(PimaIndiansDiabetes2)
df
```

```
## # A tibble: 768 x 9
##    pregnant glucose pressure triceps insulin  mass pedigree   age diabetes
##  *    <dbl>   <dbl>    <dbl>   <dbl>   <dbl> <dbl>    <dbl> <dbl> <fct>
##  1     6.00   148       72.0    35.0    NA    33.6    0.627  50.0 pos
##  2     1.00    85.0     66.0    29.0    NA    26.6    0.351  31.0 neg
##  3     8.00   183       64.0    NA      NA    23.3    0.672  32.0 pos
##  4     1.00    89.0     66.0    23.0    94.0  28.1    0.167  21.0 neg
##  5     0      137       40.0    35.0   168    43.1    2.29   33.0 pos
##  6     5.00   116       74.0    NA      NA    25.6    0.201  30.0 neg
##  7     3.00    78.0     50.0    32.0    88.0  31.0    0.248  26.0 pos
##  8    10.0    115       NA      NA      NA    35.3    0.134  29.0 neg
##  9     2.00   197       70.0    45.0   543    30.5    0.158  53.0 pos
## 10     8.00   125       96.0    NA      NA    NA      0.232  54.0 pos
## # ... with 758 more rows
```

glimpse

The glimpse function is similar to str but can handle bigger data more efficiently.

```
glimpse(df)
```

```
## Observations: 768
## Variables: 9
## $ pregnant <dbl> 6, 1, 8, 1, 0, 5, 3, 10, 2, 8, 4, 10, 10, 1, 5, 7, 0,...
## $ glucose  <dbl> 148, 85, 183, 89, 137, 116, 78, 115, 197, 125, 110, 1...
## $ pressure <dbl> 72, 66, 64, 66, 40, 74, 50, NA, 70, 96, 92, 74, 80, 6...
## $ triceps  <dbl> 35, 29, NA, 23, 35, NA, 32, NA, 45, NA, NA, NA, NA, 2...
## $ insulin  <dbl> NA, NA, NA, 94, 168, NA, 88, NA, 543, NA, NA, NA, NA,...
## $ mass     <dbl> 33.6, 26.6, 23.3, 28.1, 43.1, 25.6, 31.0, 35.3, 30.5,...
```

```
## $ pedigree <dbl> 0.627, 0.351, 0.672, 0.167, 2.288, 0.201, 0.248, 0.13...
## $ age      <dbl> 50, 31, 32, 21, 33, 30, 26, 29, 53, 54, 30, 34, 57, 5...
## $ diabetes <fct> pos, neg, pos, neg, pos, neg, pos, neg, pos, pos, neg...
```

5 functions of dplyr

- select - remove columns
- mutate - create new columns from the data
- filter - remove rows
- arrange - rearrange rows
- summarize or summarise - summary statistics

These functions do not change the original data but return a new object. The functions assume that the data is already "tidy" – observations in rows, features in columns.

select – used to select columns

Select a couple of columns to print. Notice it doesn't run off the screen, you get a screen's worth at a time.

```
print(select(df, diabetes, pregnant))
```

```
## # A tibble: 768 x 2
##    diabetes pregnant
##  * <fct>       <dbl>
##  1 pos          6.00
##  2 neg          1.00
##  3 pos          8.00
##  4 neg          1.00
##  5 pos          0
##  6 neg          5.00
##  7 pos          3.00
##  8 neg         10.0
##  9 pos          2.00
## 10 pos          8.00
## # ... with 758 more rows
```

mutate – used to add columns

Add a column that is a binary factor indicating if glucose is above average for the population.

```
mutate(df, glucose_high = as.factor(ifelse(glucose>mean(glucose, na.rm=TRUE), 1, 0)))
```

```
## # A tibble: 768 x 10
##    pregnant glucose pressure triceps insulin  mass pedigree   age diabetes
##       <dbl>   <dbl>    <dbl>   <dbl>   <dbl> <dbl>    <dbl> <dbl> <fct>
##  1     6.00   148       72.0    35.0    NA    33.6    0.627  50.0 pos
##  2     1.00    85.0     66.0    29.0    NA    26.6    0.351  31.0 neg
##  3     8.00   183       64.0    NA      NA    23.3    0.672  32.0 pos
##  4     1.00    89.0     66.0    23.0    94.0  28.1    0.167  21.0 neg
##  5     0      137       40.0    35.0   168    43.1    2.29   33.0 pos
##  6     5.00   116       74.0    NA      NA    25.6    0.201  30.0 neg
##  7     3.00    78.0     50.0    32.0    88.0  31.0    0.248  26.0 pos
##  8    10.0    115       NA      NA      NA    35.3    0.134  29.0 neg
##  9     2.00   197       70.0    45.0   543    30.5    0.158  53.0 pos
```

```
## 10     8.00   125       96.0    NA      NA    NA      0.232  54.0 pos
## # ... with 758 more rows, and 1 more variable: glucose_high <fct>
```

filter – used to remove rows

We replace df with a df that filtered out rows with NAs in the glucose or mass columns.

```
df <- filter(df, !is.na(glucose), !is.na(mass))
glimpse(df)
```

```
## Observations: 752
## Variables: 9
## $ pregnant <dbl> 6, 1, 8, 1, 0, 5, 3, 10, 2, 4, 10, 10, 1, 5, 7, 0, 7,...
## $ glucose  <dbl> 148, 85, 183, 89, 137, 116, 78, 115, 197, 110, 168, 1...
## $ pressure <dbl> 72, 66, 64, 66, 40, 74, 50, NA, 70, 92, 74, 80, 60, 7...
## $ triceps  <dbl> 35, 29, NA, 23, 35, NA, 32, NA, 45, NA, NA, NA, 23, 1...
## $ insulin  <dbl> NA, NA, NA, 94, 168, NA, 88, NA, 543, NA, NA, NA, 846...
## $ mass     <dbl> 33.6, 26.6, 23.3, 28.1, 43.1, 25.6, 31.0, 35.3, 30.5,...
## $ pedigree <dbl> 0.627, 0.351, 0.672, 0.167, 2.288, 0.201, 0.248, 0.13...
## $ age      <dbl> 50, 31, 32, 21, 33, 30, 26, 29, 53, 30, 34, 57, 59, 5...
## $ diabetes <fct> pos, neg, pos, neg, pos, neg, pos, neg, pos, neg, pos...
```

arrange – arrange rows based on content

Arrange rows based on bmi as stored in the mass column.

```
arrange(df, mass)  # ascending order
```

```
## # A tibble: 752 x 9
##    pregnant glucose pressure triceps insulin  mass pedigree   age diabetes
##       <dbl>   <dbl>    <dbl>   <dbl>   <dbl> <dbl>    <dbl> <dbl> <fct>
##  1     1.00    83.0     68.0    NA      NA    18.2    0.624  27.0 neg
##  2     1.00    97.0     70.0    15.0    NA    18.2    0.147  21.0 neg
##  3     1.00    97.0     64.0    19.0    82.0  18.2    0.299  21.0 neg
##  4     0      104       76.0    NA      NA    18.4    0.582  27.0 neg
##  5     1.00    80.0     55.0    NA      NA    19.1    0.258  21.0 neg
##  6     3.00    99.0     80.0    11.0    64.0  19.3    0.284  30.0 neg
##  7     1.00   103       80.0    11.0    82.0  19.4    0.491  22.0 neg
##  8     1.00    92.0     62.0    25.0    41.0  19.5    0.482  25.0 neg
##  9     1.00   100       74.0    12.0    46.0  19.5    0.149  28.0 neg
## 10     1.00    95.0     66.0    13.0    38.0  19.6    0.334  25.0 neg
## # ... with 742 more rows
```

```
arrange(df, desc(mass)) # descending order
```

```
## # A tibble: 752 x 9
##    pregnant glucose pressure triceps insulin  mass pedigree   age diabetes
##       <dbl>   <dbl>    <dbl>   <dbl>   <dbl> <dbl>    <dbl> <dbl> <fct>
##  1     0      129      110      46.0   130    67.1    0.319  26.0 pos
##  2     0      180       78.0    63.0    14.0  59.4    2.42   25.0 pos
##  3     3.00   123      100      35.0   240    57.3    0.880  22.0 neg
##  4     1.00    88.0     30.0    42.0    99.0  55.0    0.496  26.0 pos
##  5     0      162       76.0    56.0   100    53.2    0.759  25.0 pos
##  6     5.00   115       98.0    NA      NA    52.9    0.209  28.0 pos
##  7    11.0    135       NA      NA      NA    52.3    0.578  40.0 pos
```

```
##  8     0      165       90.0    33.0   680    52.3    0.427  23.0 neg
##  9     7.00   152       88.0    44.0    NA    50.0    0.337  36.0 pos
## 10     1.00   122       90.0    51.0   220    49.7    0.325  31.0 pos
## # ... with 742 more rows
```

summarize - a more powerful summary

Get summary statistics on mass.

```
summarize(df, min=min(mass), max=max(mass), sd(mass))
```

```
## # A tibble: 1 x 3
##     min   max `sd(mass)`
##   <dbl> <dbl>      <dbl>
## 1  18.2  67.1       6.93
```

```
summarize(df, n_diabetic = sum(diabetes=="pos"), n_not_diabetic = sum(diabetes=="neg"))
```

```
## # A tibble: 1 x 2
##   n_diabetic n_not_diabetic
##        <int>          <int>
## 1        264            488
```

pipes - work similar to unix pipes

Pipes make code easier to read and let you make several commands in one neat group of lines instead of nesting functions in the typical R fashion. The pipe operator %>% comes from package magrittr but dplyr automatically loads it.

```
df %>%
  group_by(diabetes) %>%
  summarize(n_diabetic = n())
```

```
## # A tibble: 2 x 2
##   diabetes n_diabetic
##   <fct>         <int>
## 1 neg             488
## 2 pos             264
```

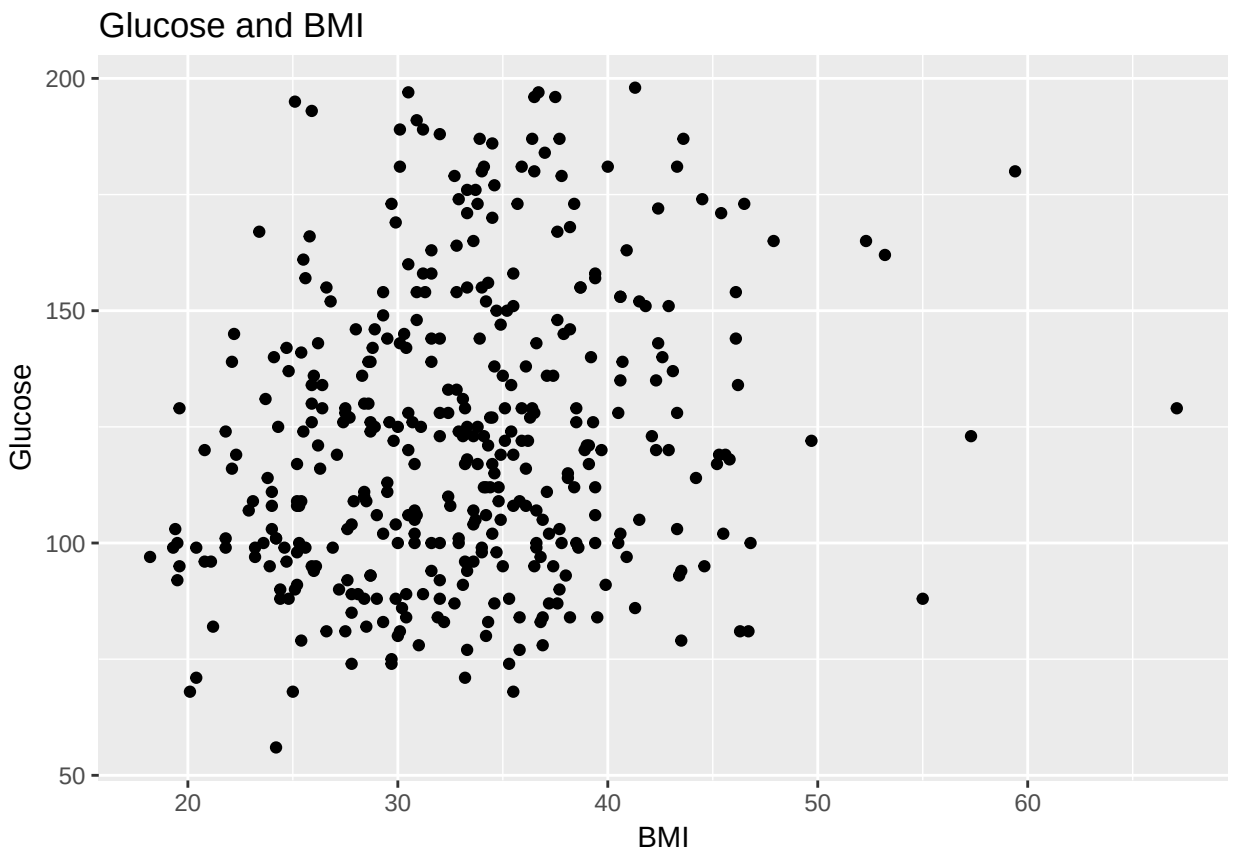
Explore ggplot2

Hadley Wickham developed ggplot2 in 2005, inspired by a grammar of graphics developed by Leland Wildinson in 1999. The ggplot2 functions are much more powerful than standard R graphs but also slower.

We have a short example below showing important components of building a ggplot. First we specify the data, then the aesthetics which are how the data is represented, followed by the geometry and finally labels.

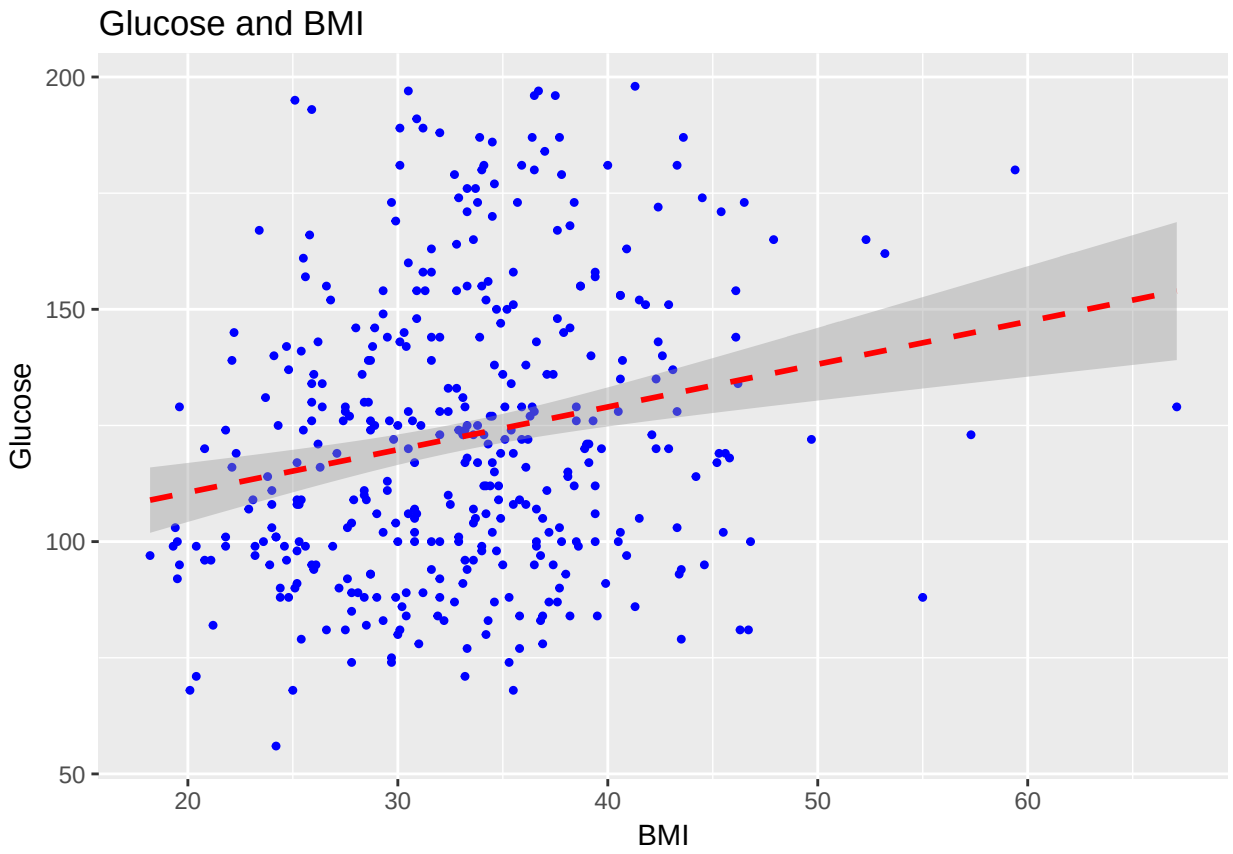
```
library(ggplot2)
# load data
library(mlbench)
data("PimaIndiansDiabetes2")
df <- PimaIndiansDiabetes2[complete.cases(PimaIndiansDiabetes2[]),]

ggplot(df, aes(x=mass, y=glucose)) +
```

```
  geom_point() +
  labs(title="Glucose and BMI", x="BMI", y="Glucose")
```



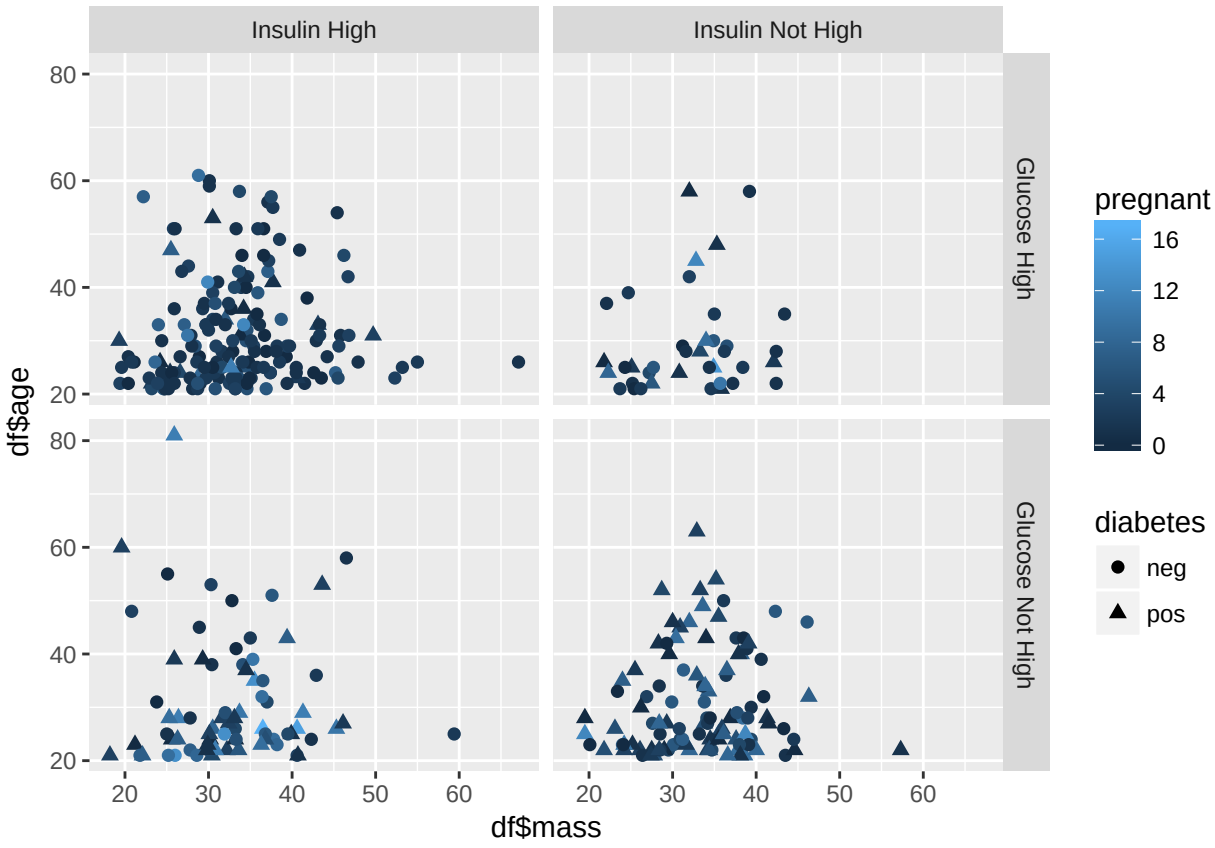
Next we add some color and a smoothing line which helps us see a trend in the data. By default the smoothing line has a shadow around it which specifies the 95

```
ggplot(df, aes(x=mass, y=glucose)) +
  geom_point(pch=20, color='blue', size=1.5) +
  geom_smooth(method='lm', color='red', linetype=2) +
  labs(title="Glucose and BMI", x="BMI", y="Glucose")
```



facet_grid

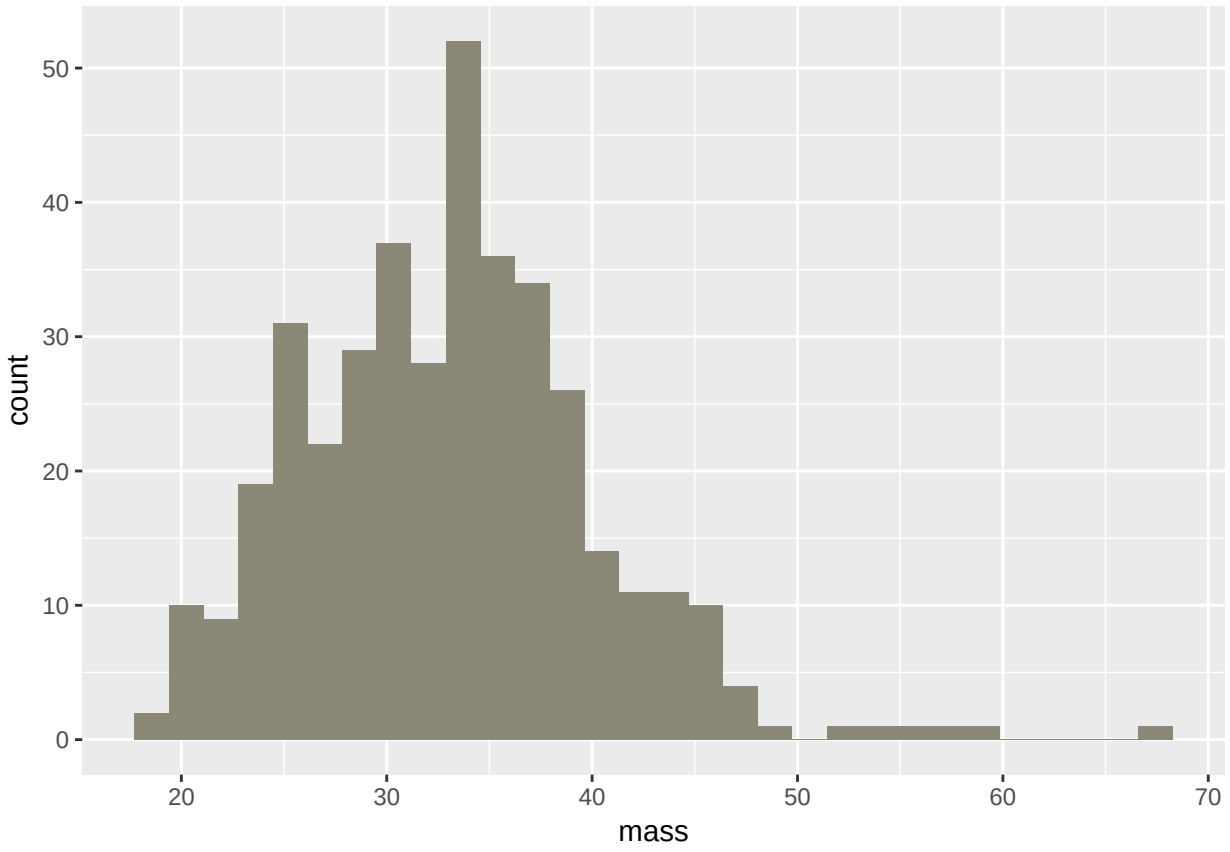
```
df$glucose_high <- factor(ifelse(df$glucose>mean(df$glucose, na.rm=TRUE), 1, 0),
                          levels=c(0,1), labels=c("Glucose High","Glucose Not High"))
df$insulin_high <- factor(ifelse(df$insulin>mean(df$insulin, na.rm=TRUE), 1, 0),
                          levels=c(0,1), labels=c("Insulin High","Insulin Not High"))

ggplot(df,
  aes(x=df$mass, y=df$age, shape=diabetes, col=pregnant)) +
  geom_point(size=2) +
  facet_grid(df$glucose_high~df$insulin_high)
```



histogram

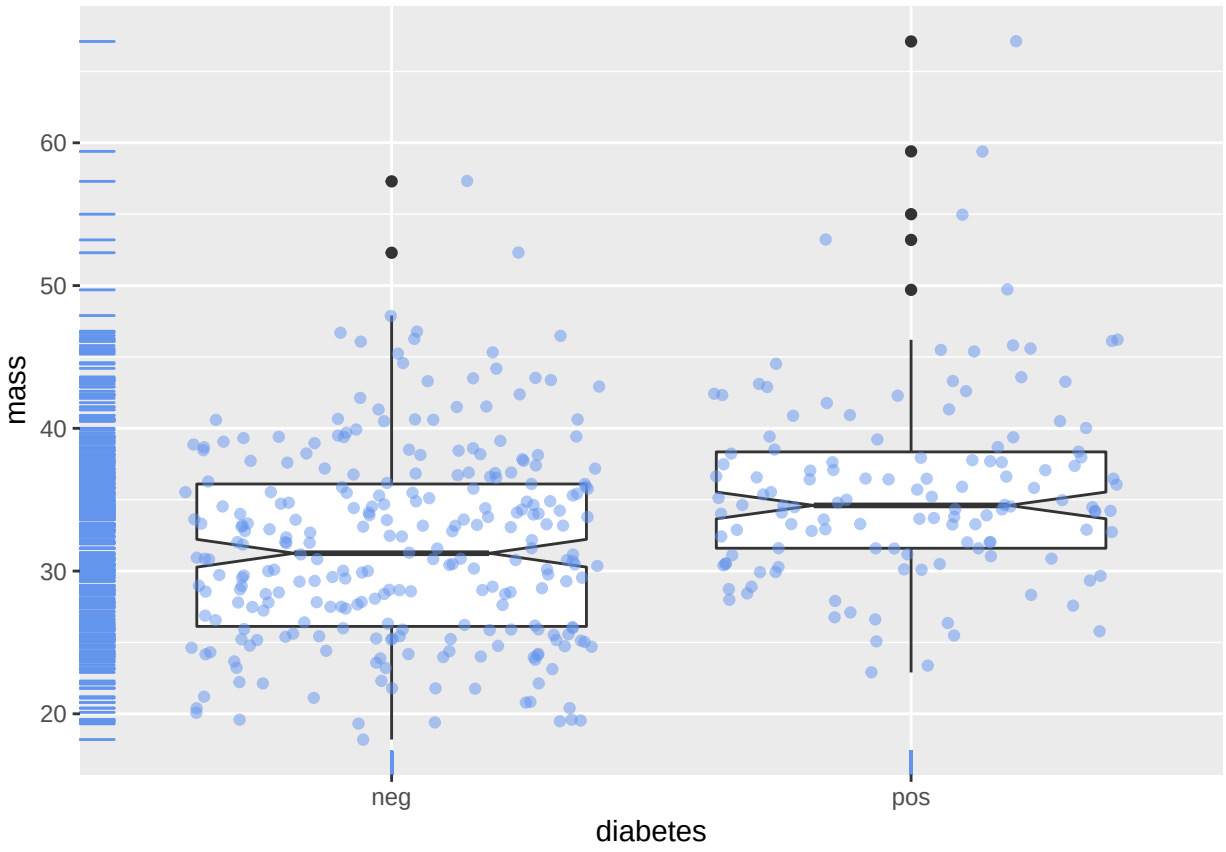
```
ggplot(df, aes(x=mass)) +
  geom_histogram(fill="cornsilk4")
```

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



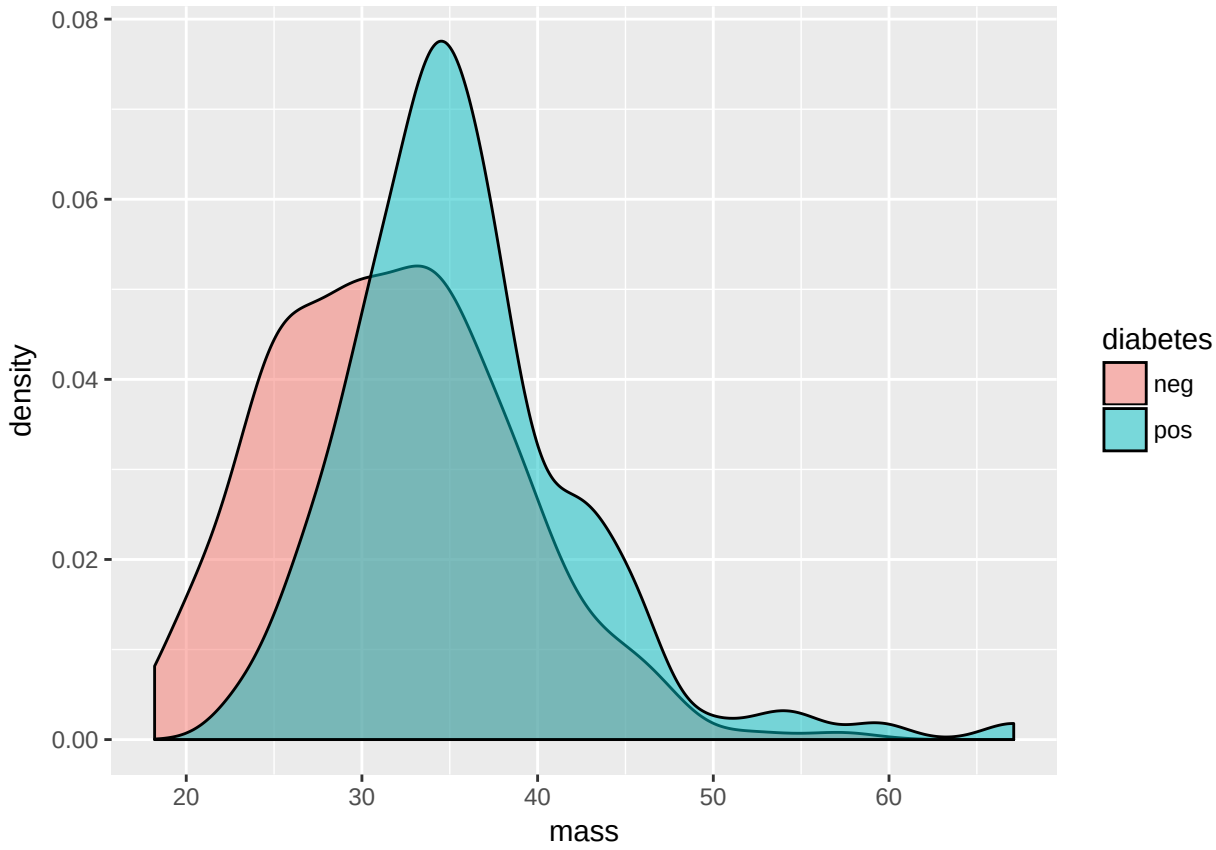
boxplot and rug

```
ggplot(df, aes(x=diabetes, y=mass)) +
  geom_boxplot(notch=TRUE) +
  geom_point(position="jitter", color="cornflowerblue", alpha=.5) +
  geom_rug(color="cornflowerblue")
```



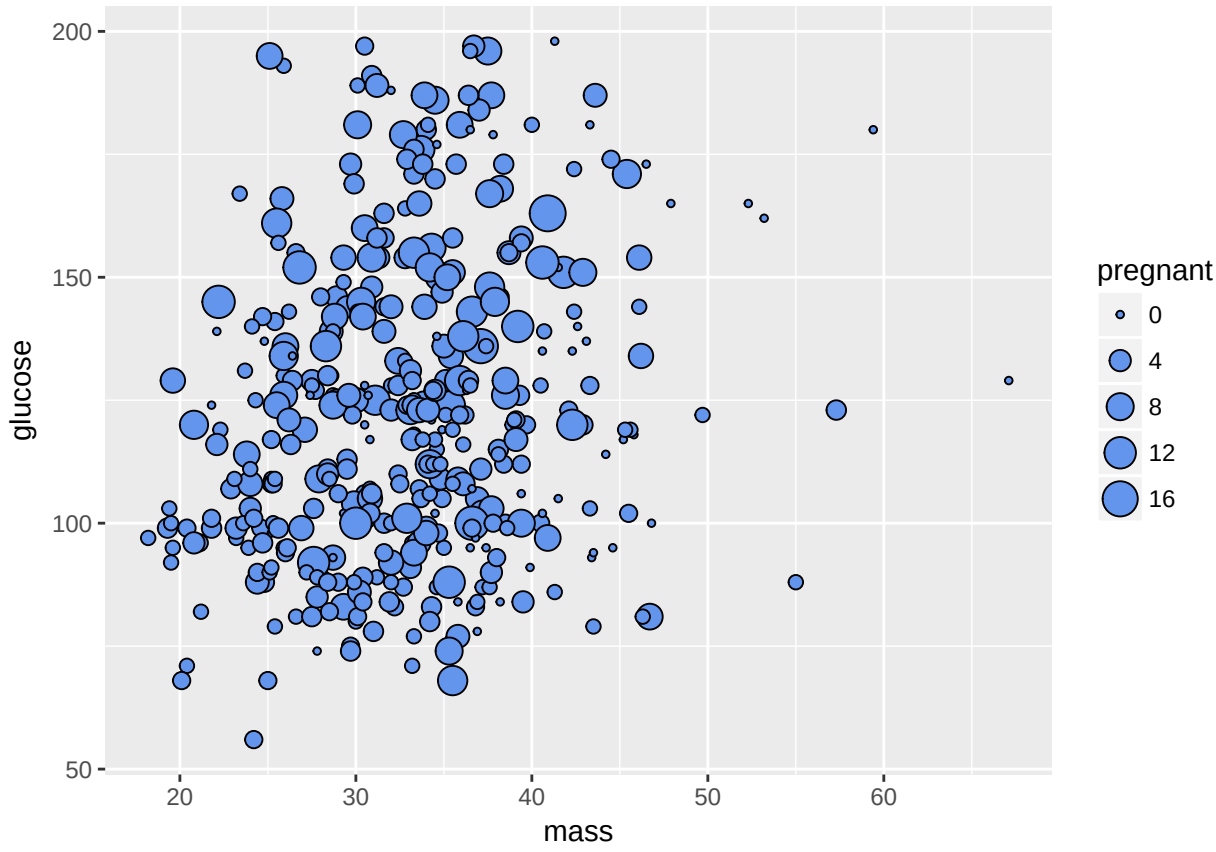
density plot

```
ggplot(df, aes(x=mass, fill=diabetes)) +
  geom_density(alpha=0.5)
```



bubble chart

```
ggplot(df,
       aes(x=mass, y=glucose, size=pregnant)) +
  geom_point(shape=21, fill="cornflowerblue")
```



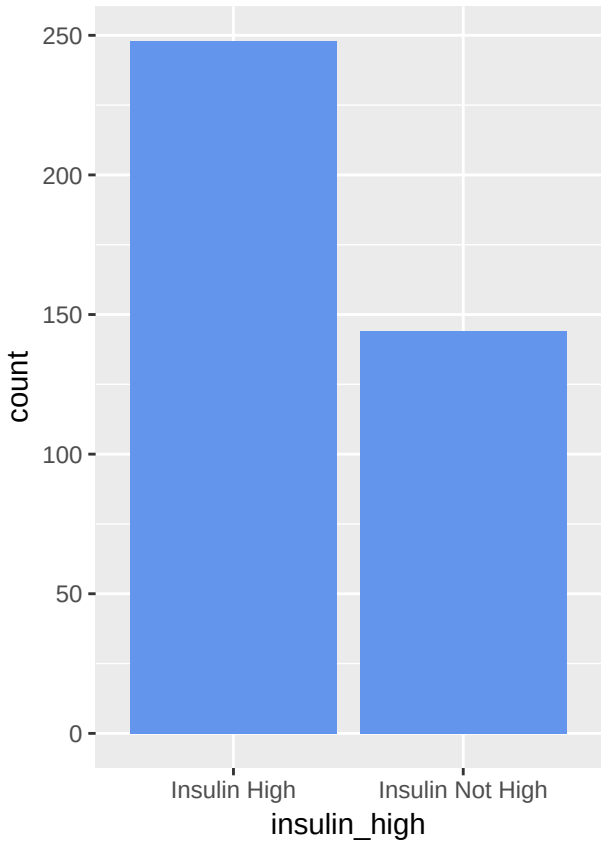
grid

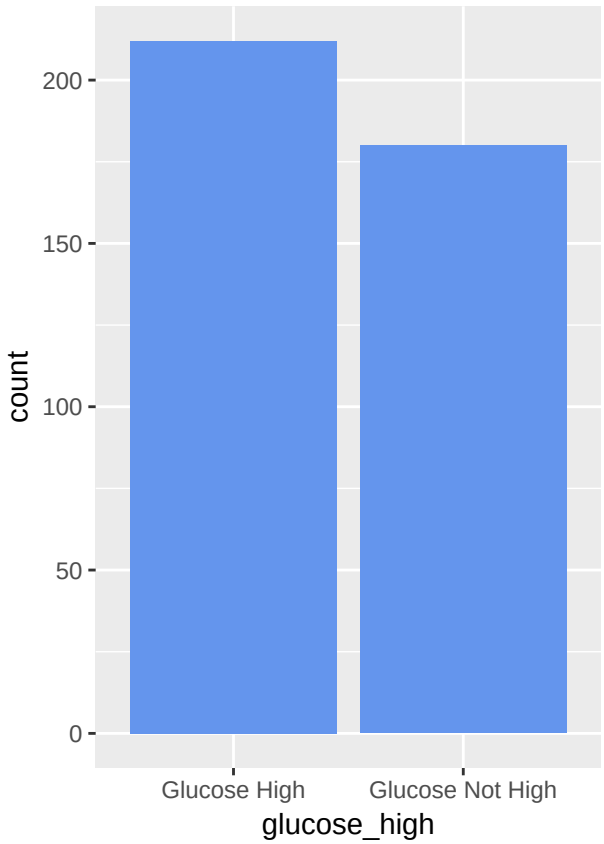
```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
p1 <- ggplot(df, aes(x=insulin_high)) + geom_bar(fill="cornflowerblue")
p2 <- ggplot(df, aes(x=glucose_high)) + geom_bar(fill="cornflowerblue")
grid.arrange(p1, p2, ncol=2)
```


250
200
150
100
50
0
count
Insulin High
Insulin Not High
insulin_high


200
150
100
50
0
count
Glucose High
Glucose Not High
glucose_high