Data Visualization

Fundamental Techniques in Data Science

Utrecht University

Kyle M. Lang

Department of Methodology & Statistics
Utrecht University

Outline

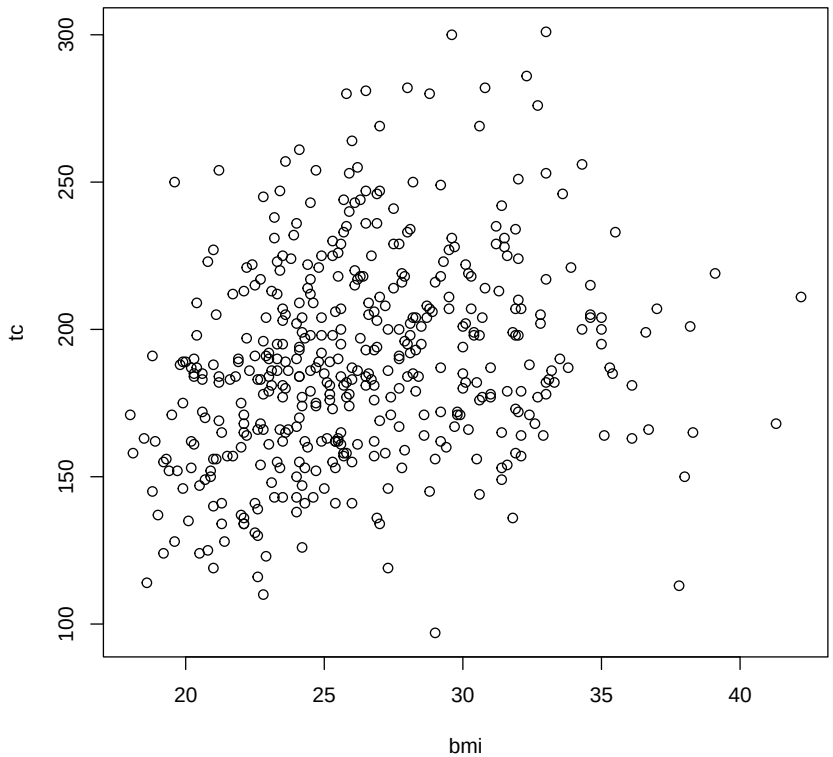
Base R Graphics

GGPlot

Base R Graphics: Scatterplots

We can create a basic scatterplot using the `plot()` function.

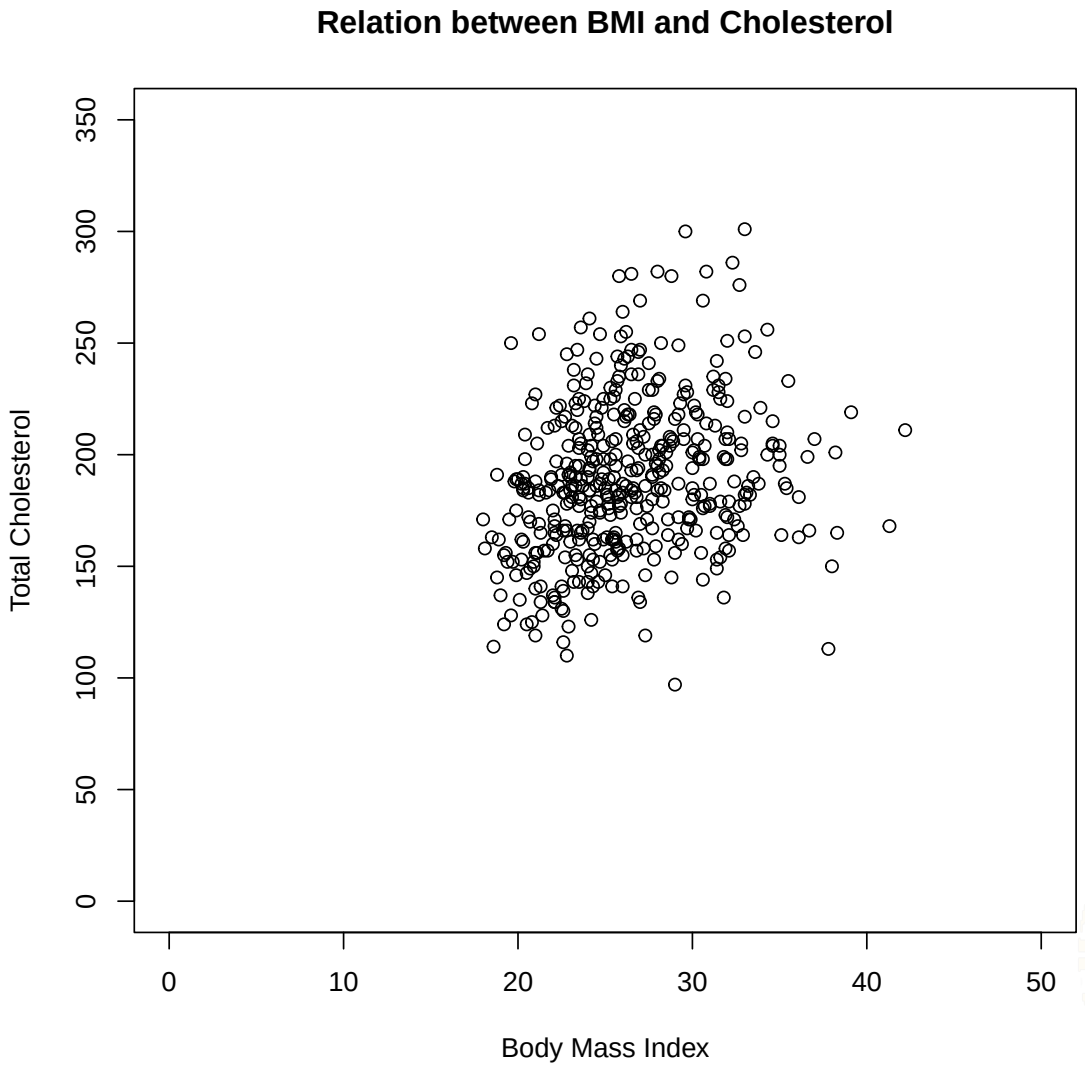
```
diabetes %$% plot(y = tc, x = bmi)
```



Base R Graphics: Scatterplots

```
diabetes %$% plot(y = tc,
                  x = bmi,
                  ylab = "Total Cholesterol",
                  xlab = "Body Mass Index",
                  main = "Relation between BMI and Cholesterol",
                  ylim = c(0, 350),
                  xlim = c(0, 50)
                  )
```

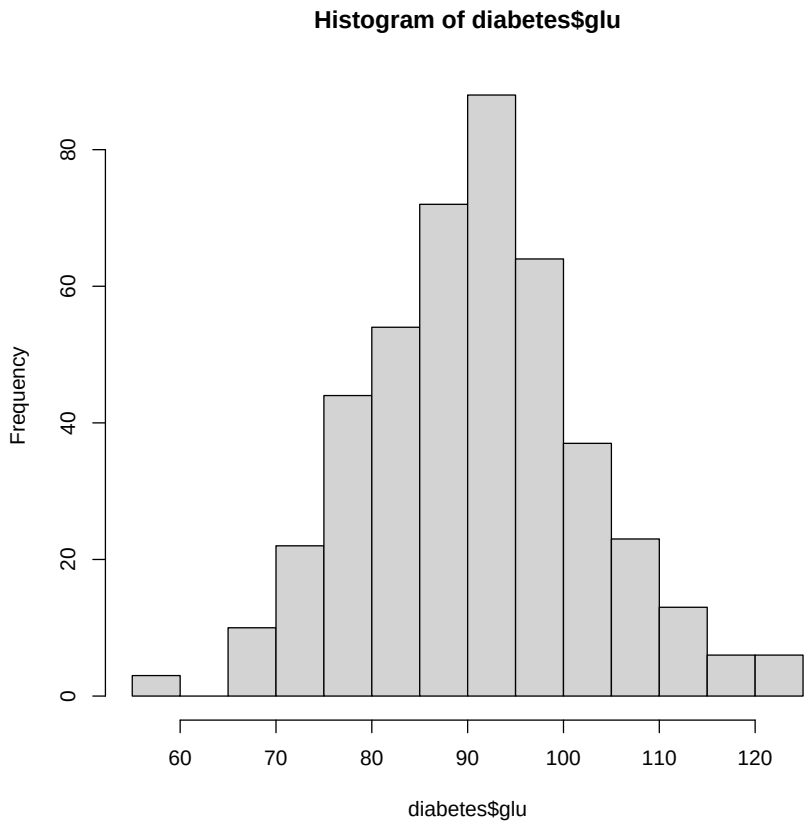
Base R Graphics: Scatterplots



Base R Graphics: Histograms

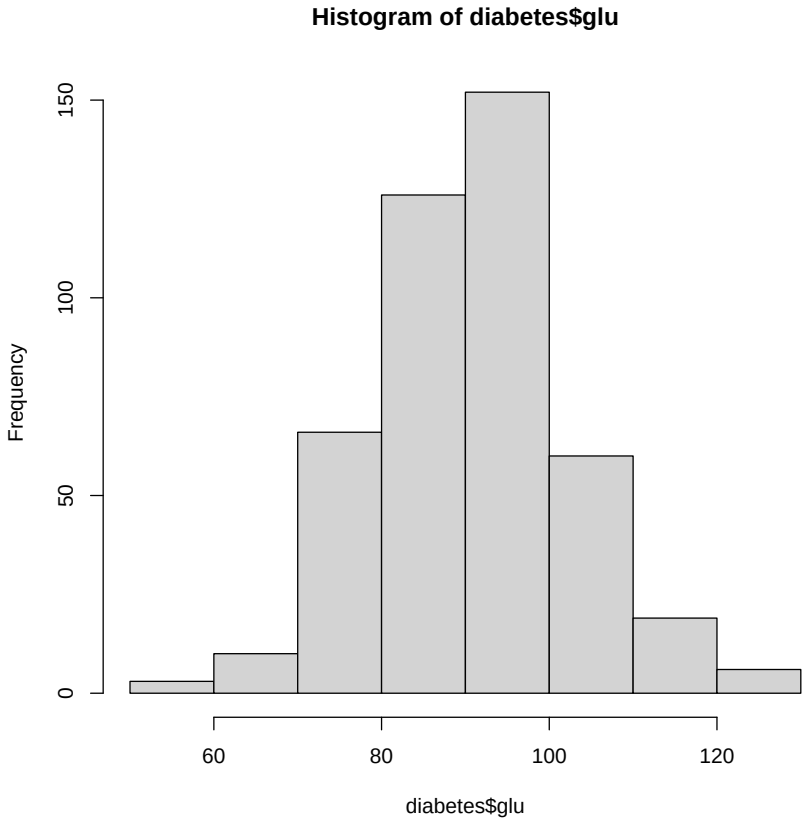
We can create a simple histogram with the `hist()` function.

```
hist(diabetes$glu)
```

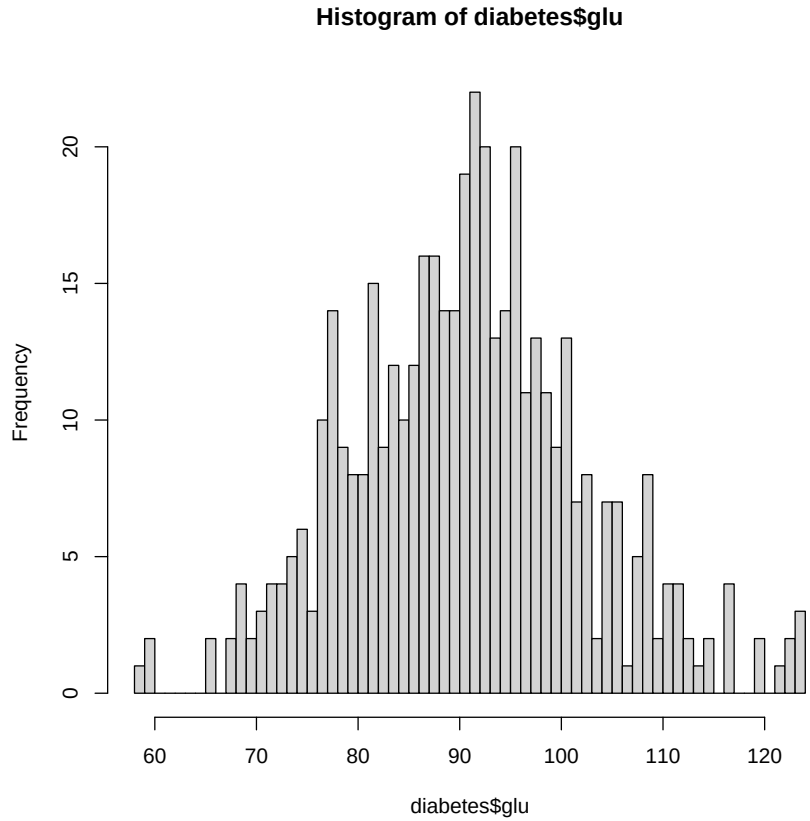


Base R Graphics: Histograms

```r
hist(diabetes$glu, breaks = 5)
```



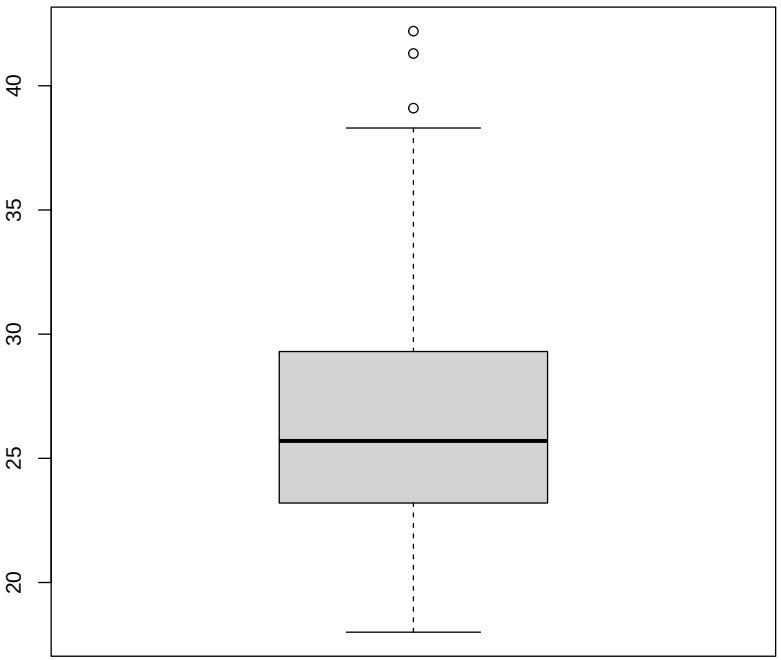

```r
hist(diabetes$glu, breaks = 50)
```



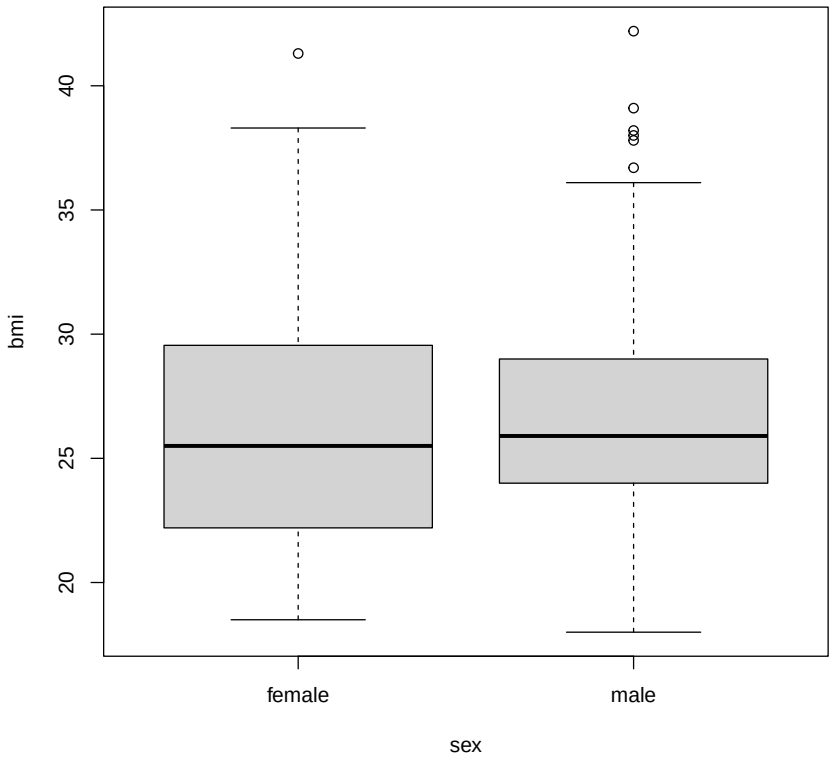
Base R Graphics: Boxplots

We can create simple boxplots via the `boxplot()` function.

```
boxplot(diabetes$bmi)
```



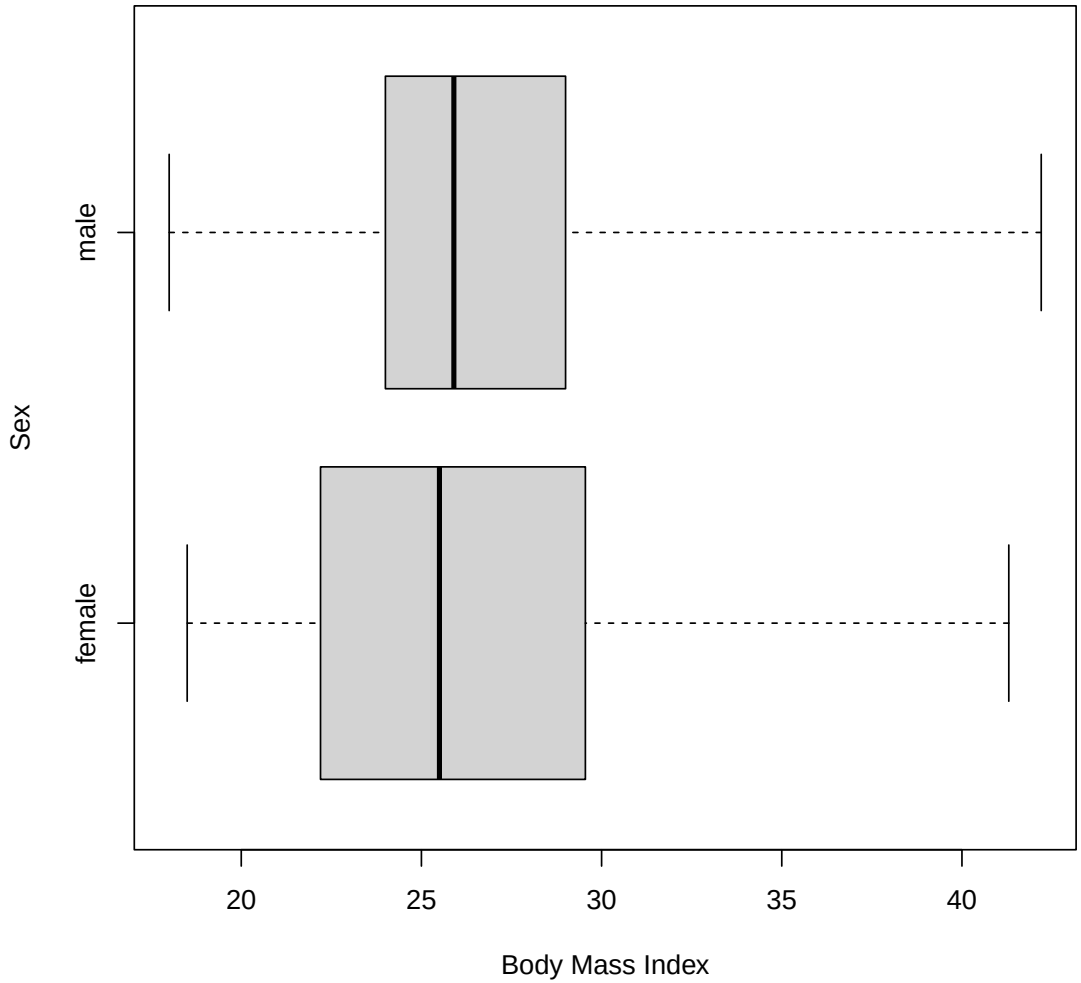

```
boxplot(bmi ~ sex, data = diabetes)
```



Base R Graphics: Boxplots

```
boxplot(bmi ~ sex,
        data = diabetes,
        horizontal = TRUE,
        range = 3,
        xlab = "Body Mass Index",
        ylab = "Sex")
```

Base R Graphics: Boxplots

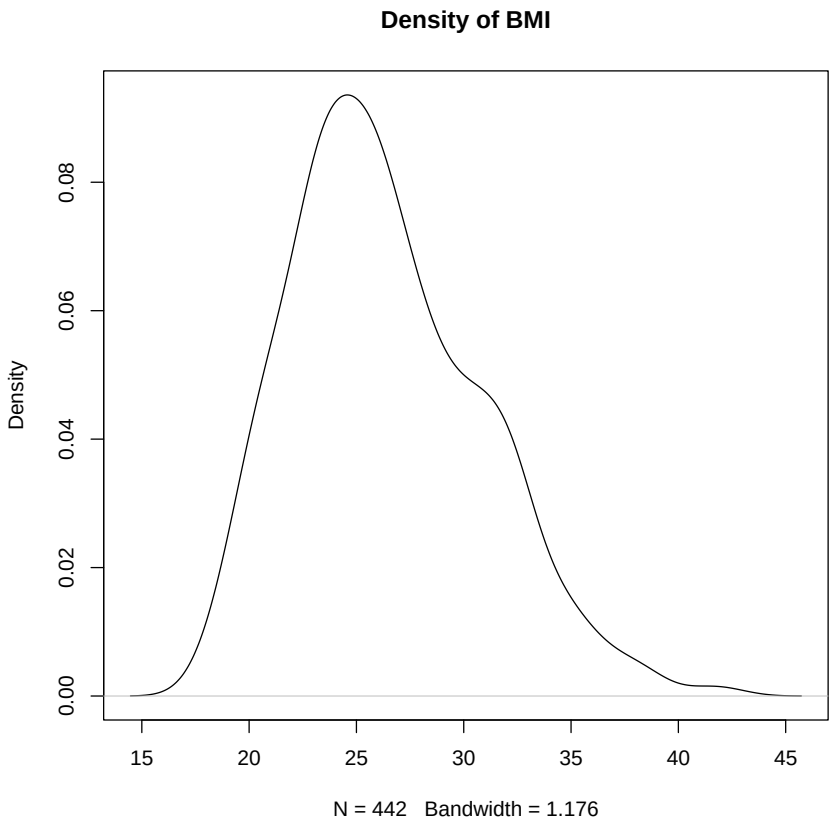


Base R Graphics: Kernel Density Plots

The `density()` function estimates the density of a variable.

- If we plot a density object, we get a kernel density plot.

```
density(diabetes$bmi) %>%
  plot(main = "Density of BMI")
```



GGPLOT

GGPlot

Base R graphics are fine for quick-and-dirty visualizations, but for publication quality graphics, you should probably use GGPlot.

- GGPlot uses the ”grammar of graphics” and ”tidy data” to build up a figure from modular components.

GGPlot: Basic Setup

We start by calling the `ggplot()` function to initialize the object that will define our plot.

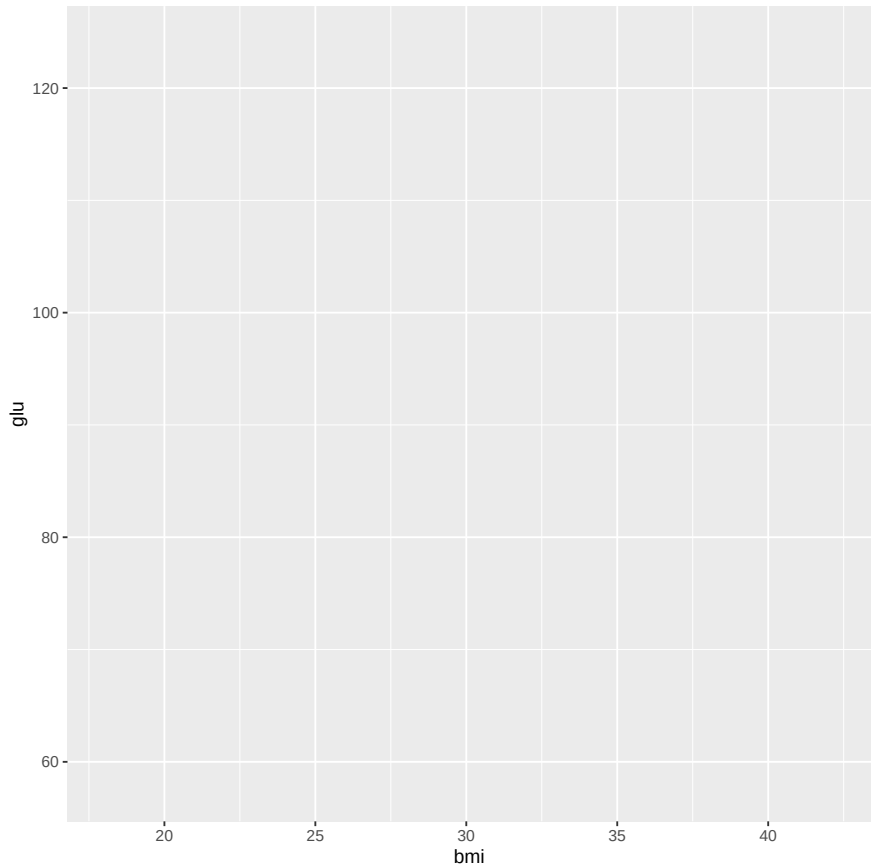
- We must specify a data source.
- We must also give some aesthetic via the `aes()` function.

```
library(ggplot2)
p1 <- ggplot(data = diabetes, mapping = aes(x = bmi, y = glu))
```

GGPlot: Basic Setup

At this point, our plot is pretty boring.

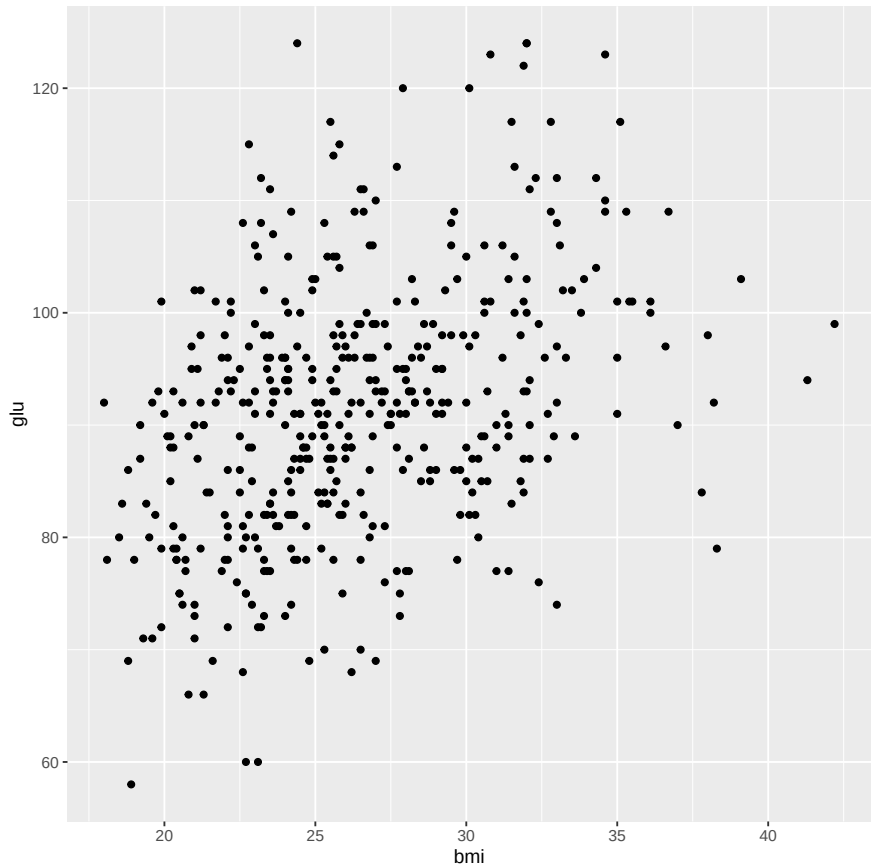
```
p1
```



GGPlot: Geometries

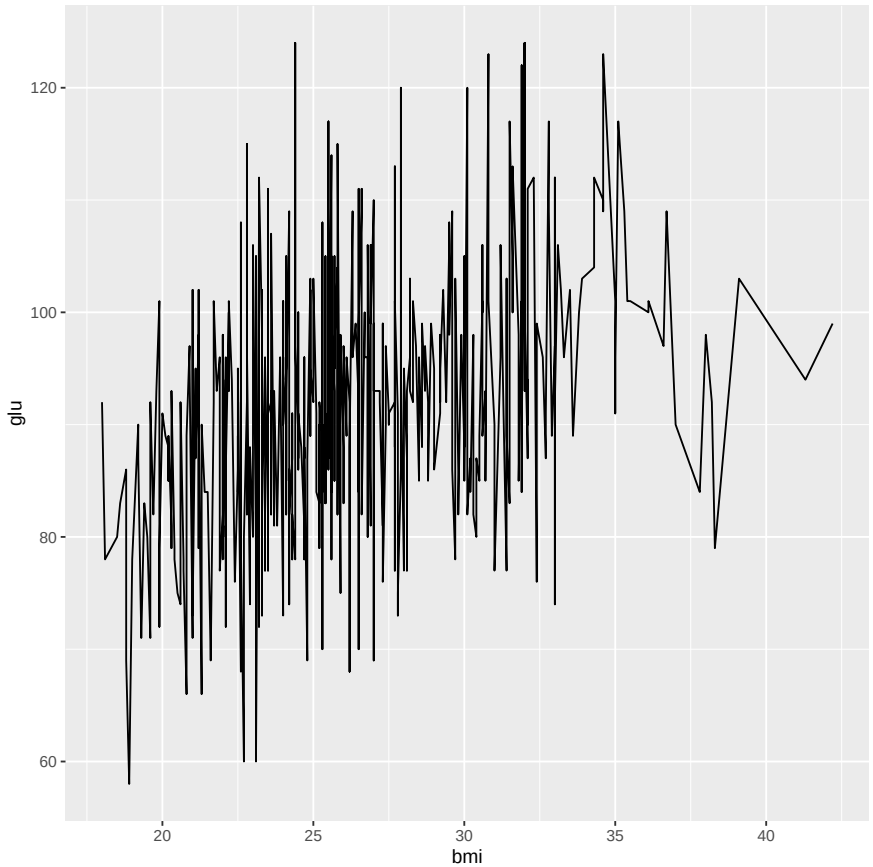
We need to define some geometry via a `geom_XXX()` function.

```
p1 + geom_point()
```



GGPlot: Geometries

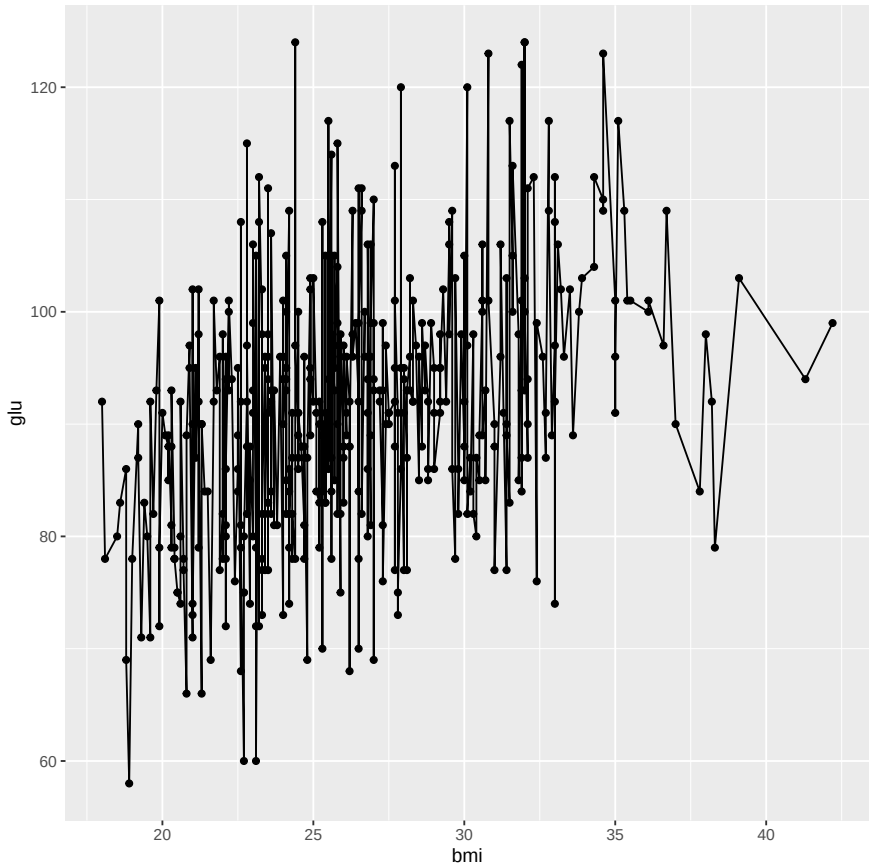
```
p1 + geom_line()
```



GGPlot: Geometries

We can also combine different geometries into a single figure

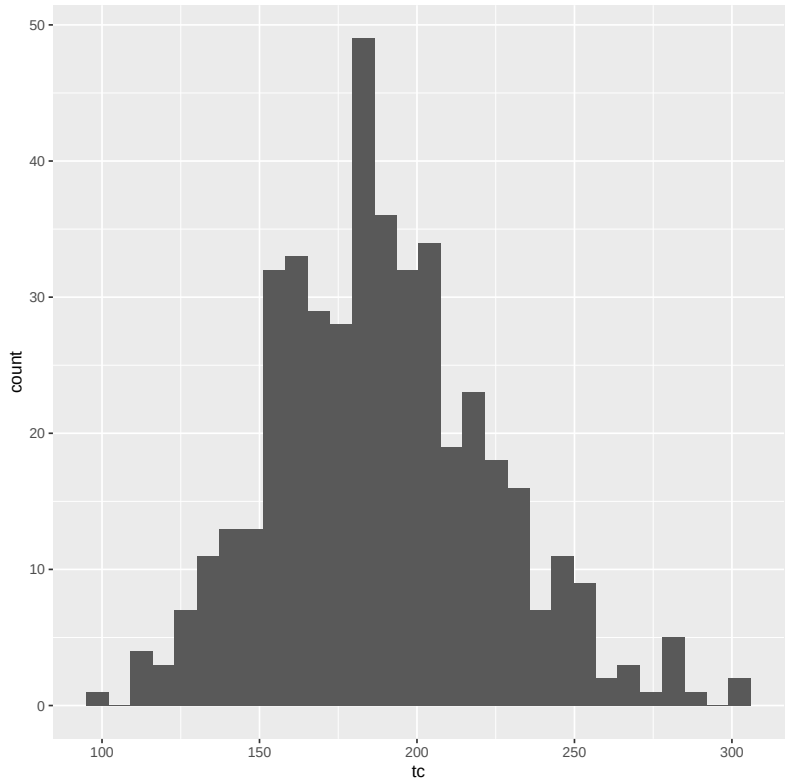
```
p1 + geom_point() + geom_line()
```



GGPlot: Geometries

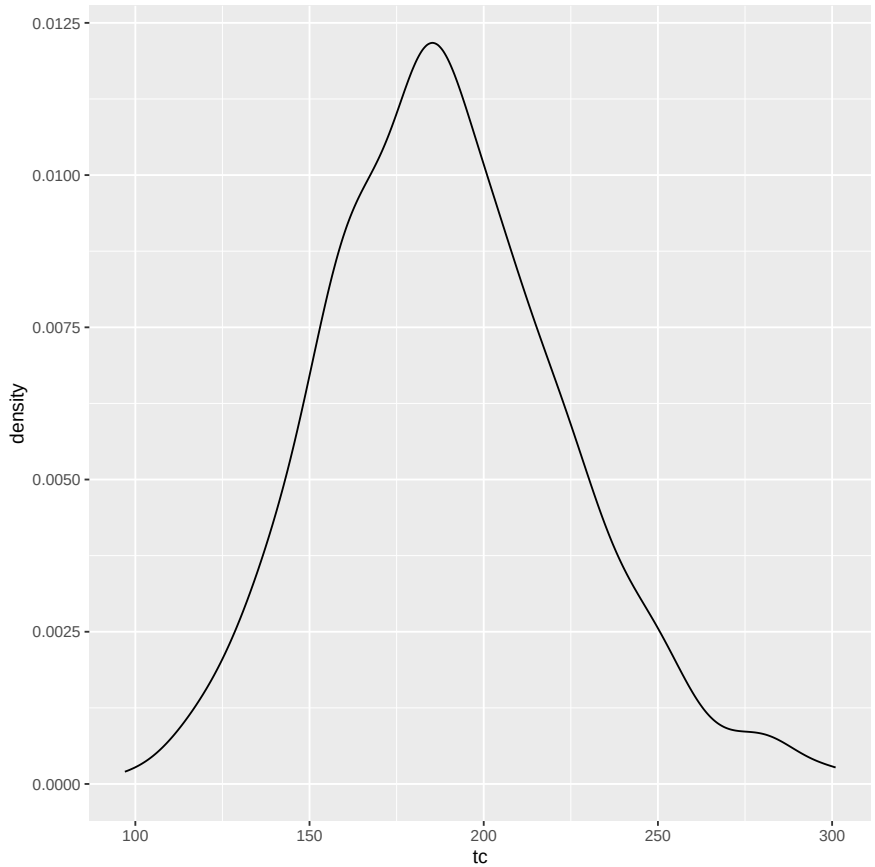
We can use different flavors of geometry for different types of data.

```
p2 <- ggplot(diabetes, aes(tc))
p2 + geom_histogram()
```



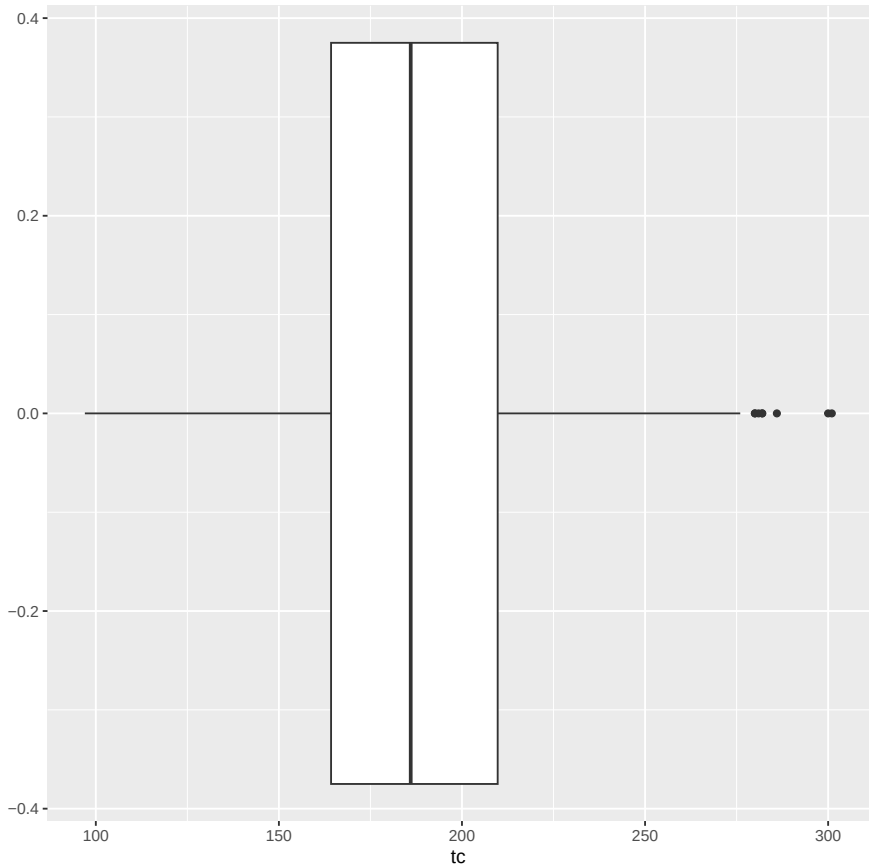
GGPlot: Geometries

```
p2 + geom_density()
```



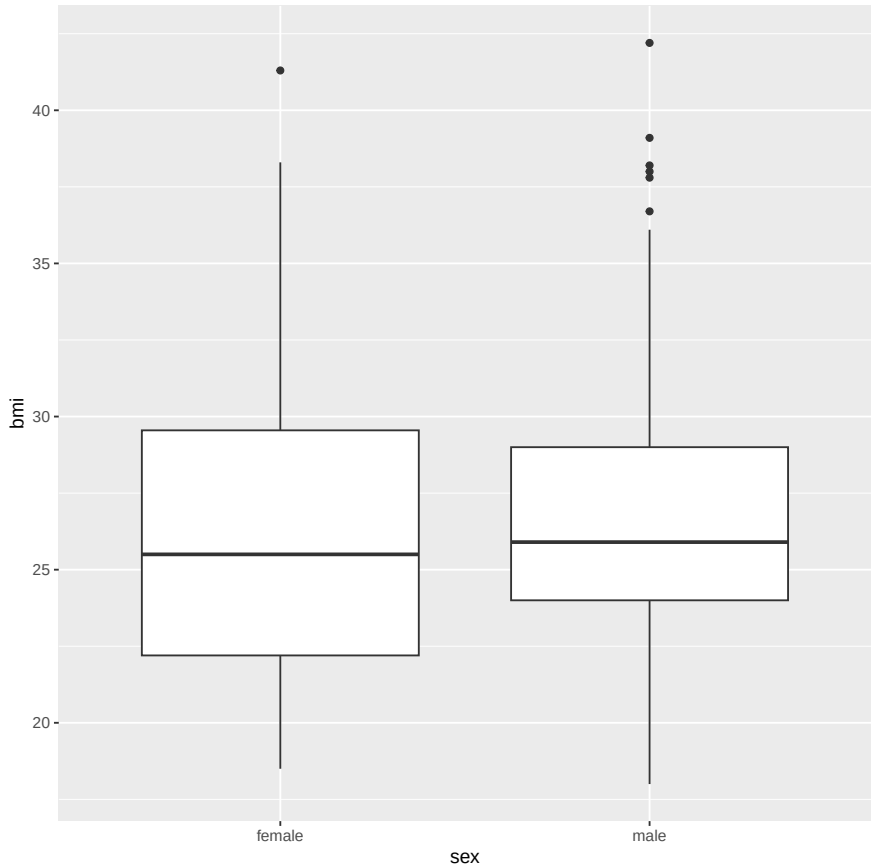
GGPlot: Geometries

```
p2 + geom_boxplot()
```



GGPlot: Geometries

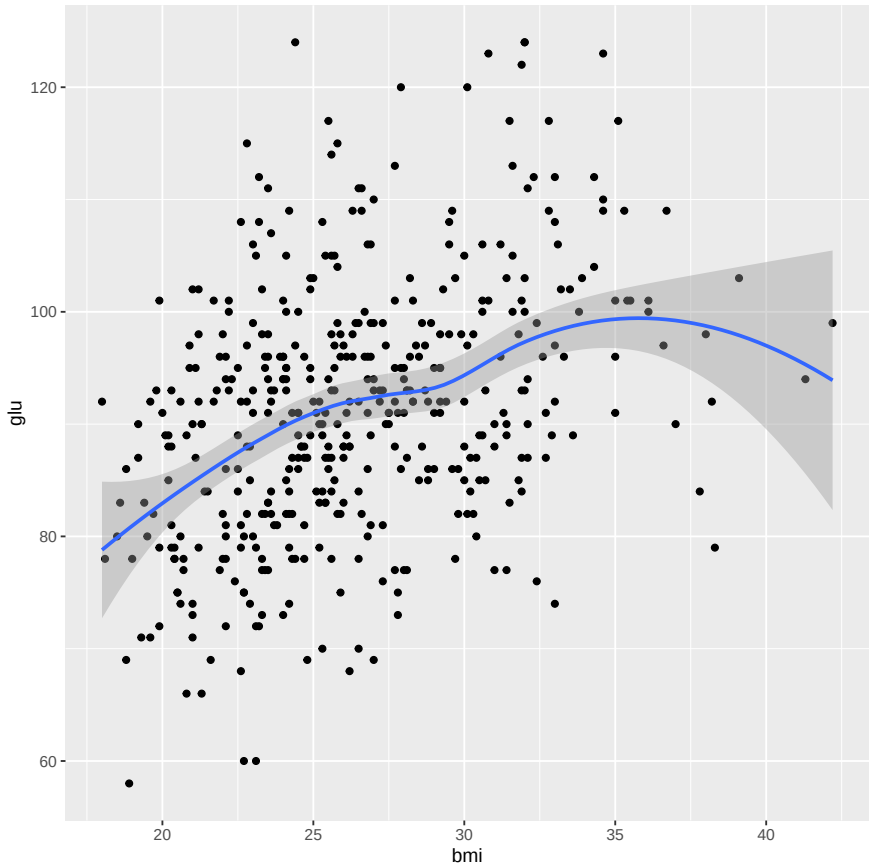
```
p3 <- ggplot(diabetes, aes(sex, bmi))
p3 + geom_boxplot()
```



GGPlot: Statistics

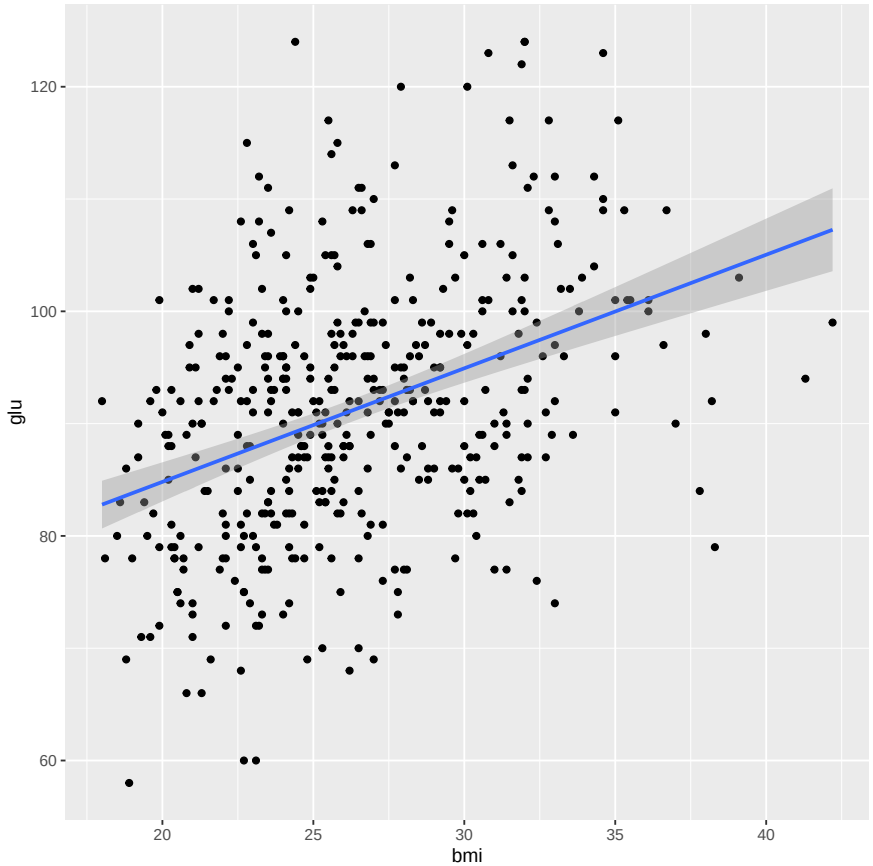
We can also add statistical summaries of the data.

```
p1 + geom_point() + geom_smooth()
```



GGPlot: Statistics

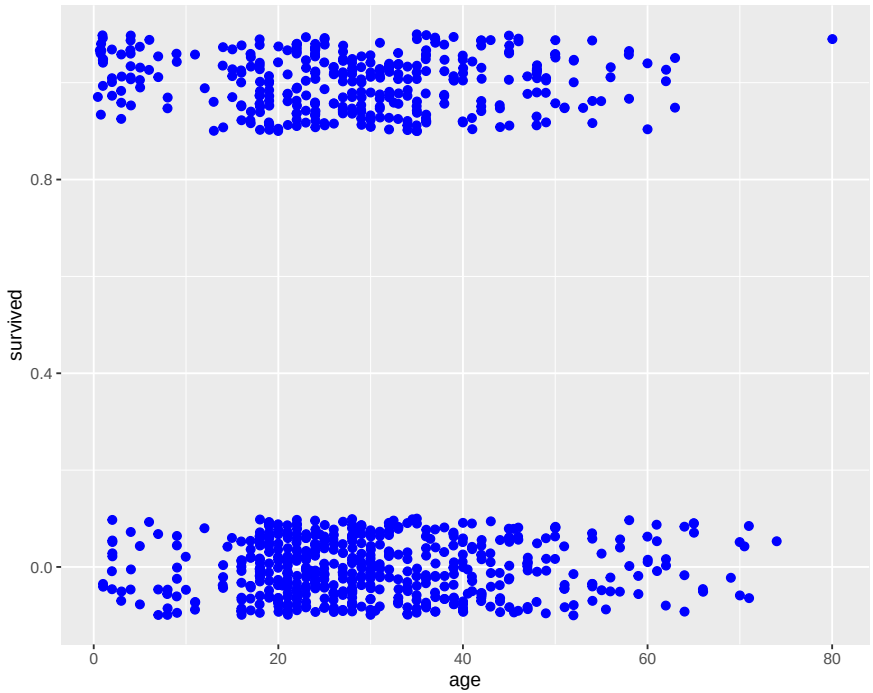
```
p1 + geom_point() + geom_smooth(method = "lm")
```



GGPlot: Styling

Changing style options outside of the `aes()` function applies the styling to the entire plot.

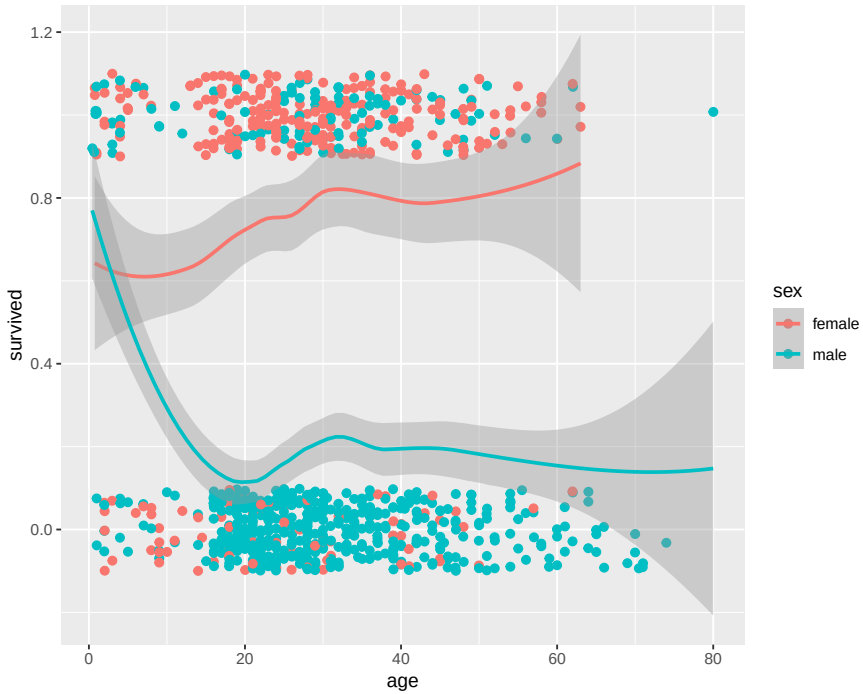
```
p5 <- ggplot(titanic, aes(age, survived))
p5 + geom_jitter(color = "blue", size = 2, height = 0.1)
```



GGPlot: Styling

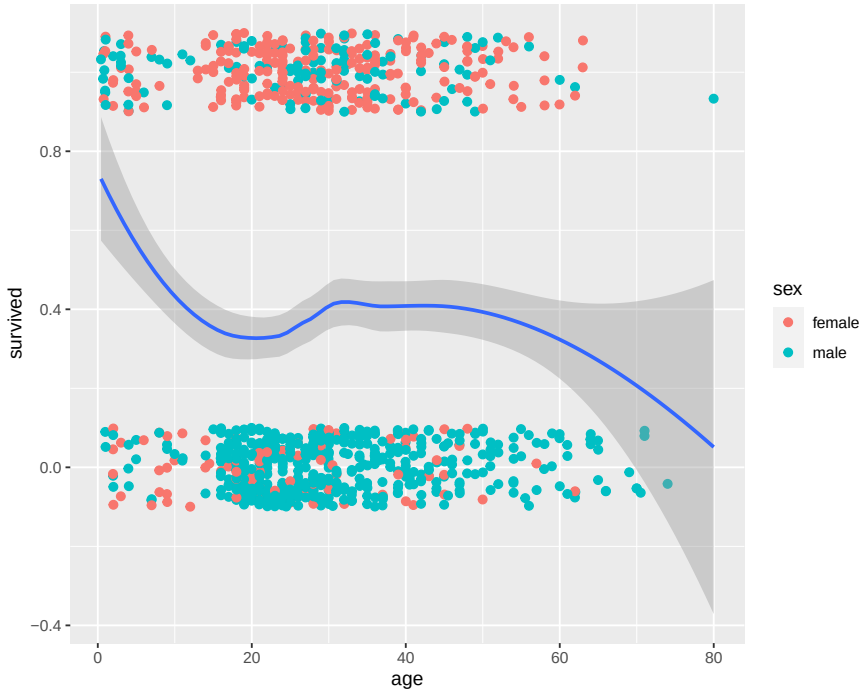
We can also apply styles as a function of variables by defining the style within the `aes()` function.

```
p6.1 <- ggplot(titanic, aes(age, survived, color = sex))
p6.1 + geom_jitter(size = 2, height = 0.1) + geom_smooth()
```



GGPlot: Styling

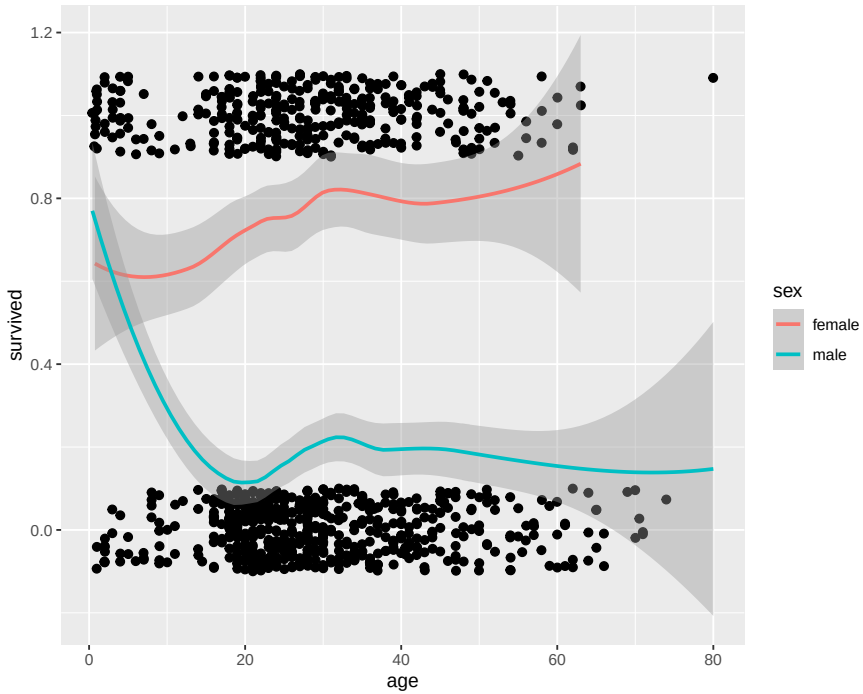
```
p6.2 <- ggplot(titanic, aes(age, survived))
p6.2 + geom_jitter(aes(color = sex), size = 2, height = 0.1) +
    geom_smooth()
```



geom_smooth inherits from ggplot, not from jitter, that is why only one line is plotted here

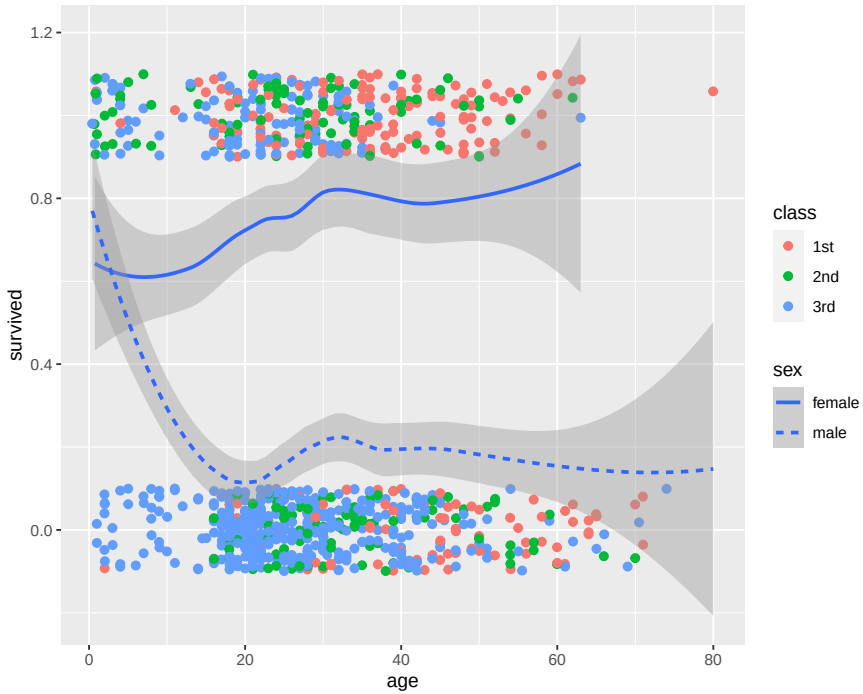
GGPlot: Styling

```
p6.2 + geom_jitter(size = 2, height = 0.1) +
    geom_smooth(aes(color = sex))
```



GGPlot: Styling

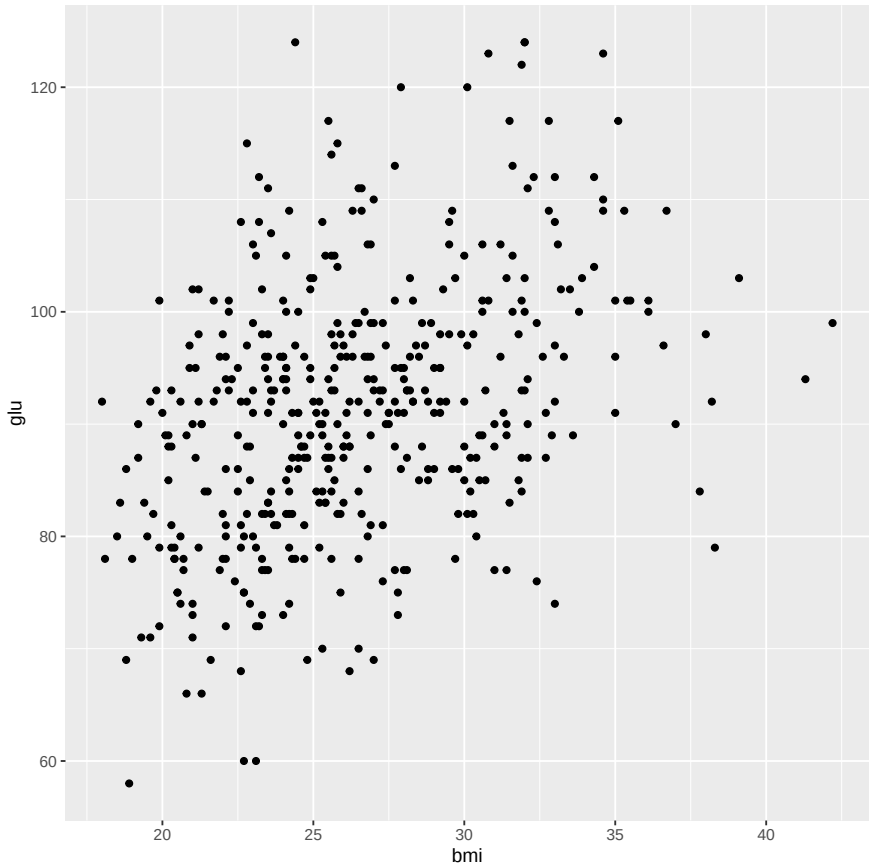
```
p6.2 + geom_jitter(aes(color = class), size = 2, height = 0.1) +
  geom_smooth(aes(linetype = sex))
```



GGPlot: Themes

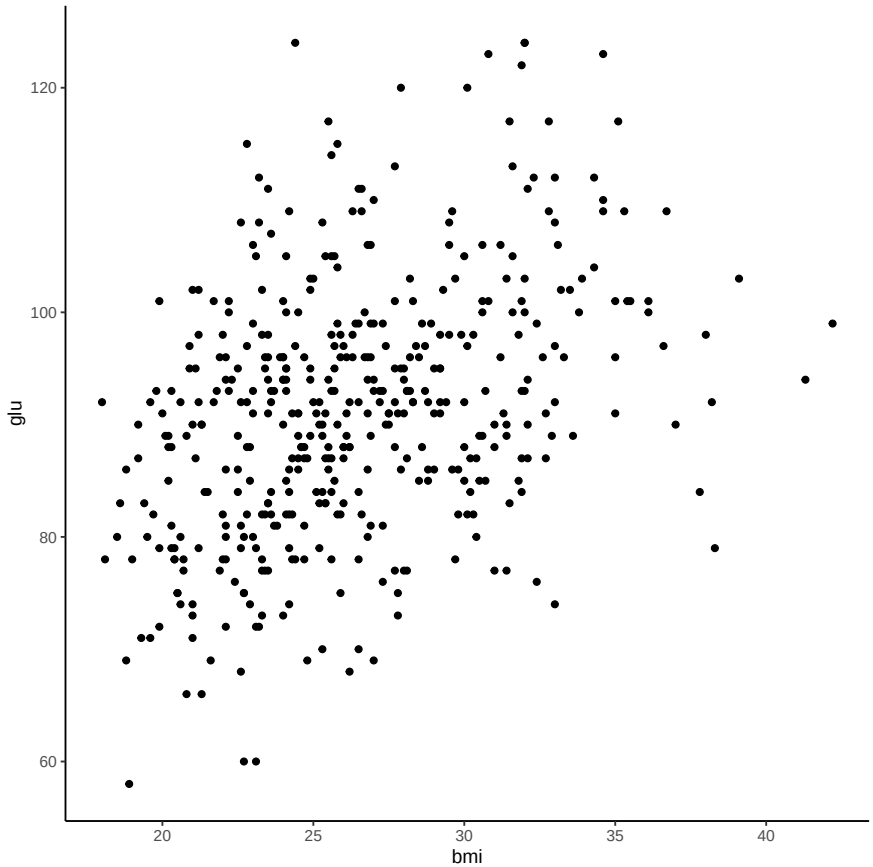
We can apply canned themes to adjust a plot's overall appearance.

```
(p1.1 <- p1 + geom_point())
```



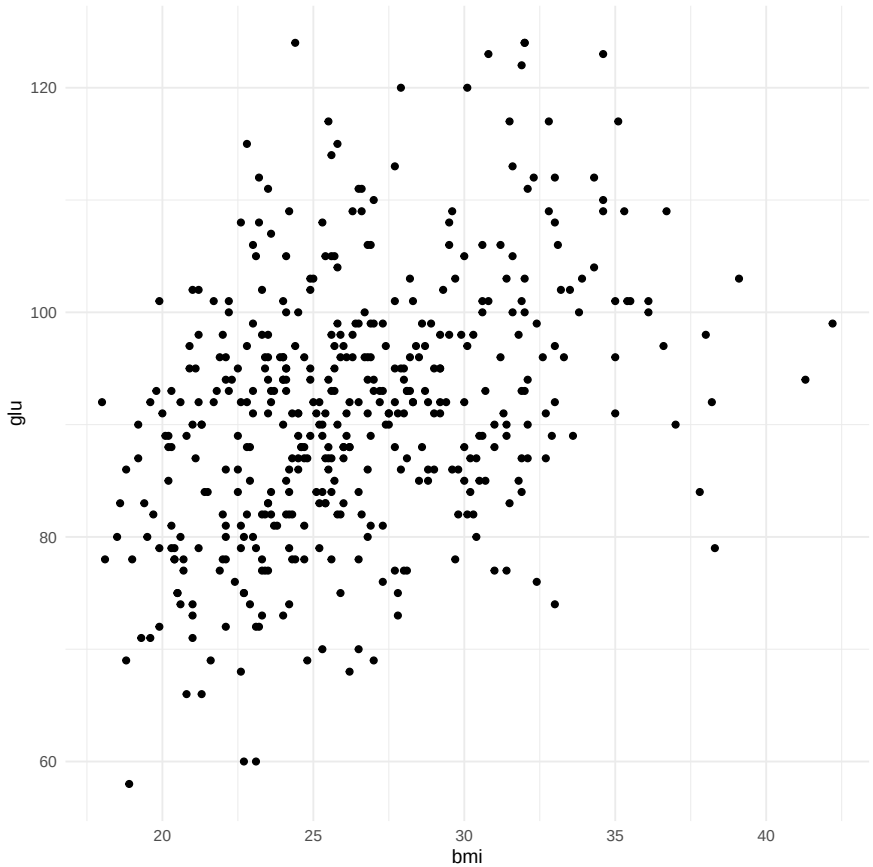
GGPlot: Themes

```
p1.1 + theme_classic()
```



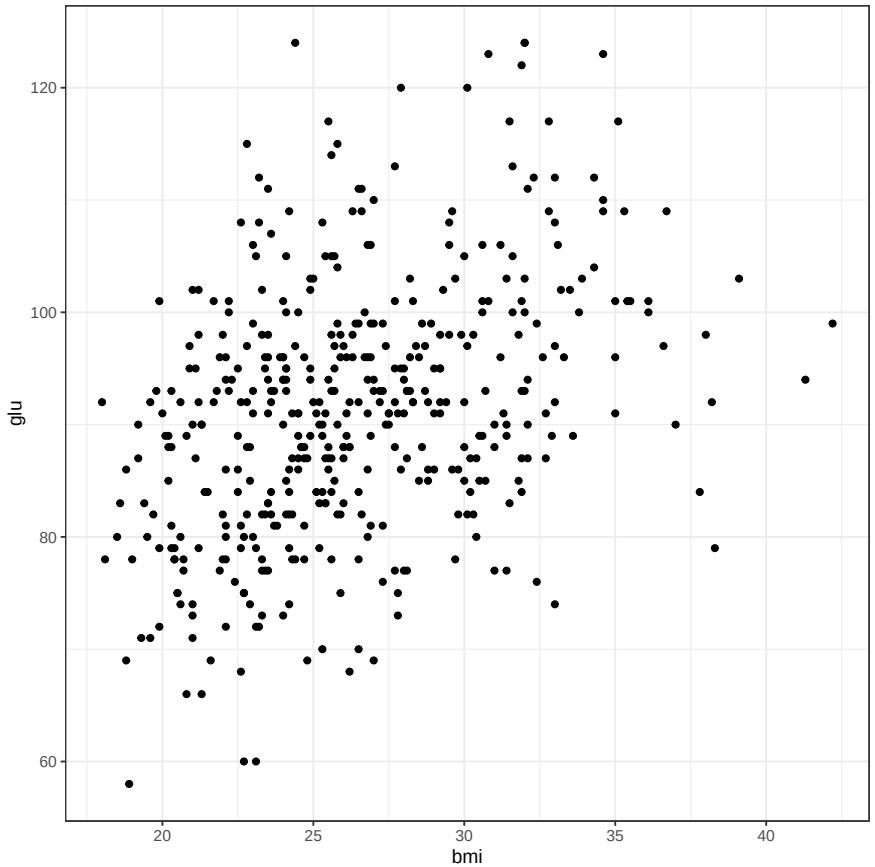
GGPlot: Themes

```
p1.1 + theme_minimal()
```



GGPlot: Themes

```
p1.1 + theme_bw()
```

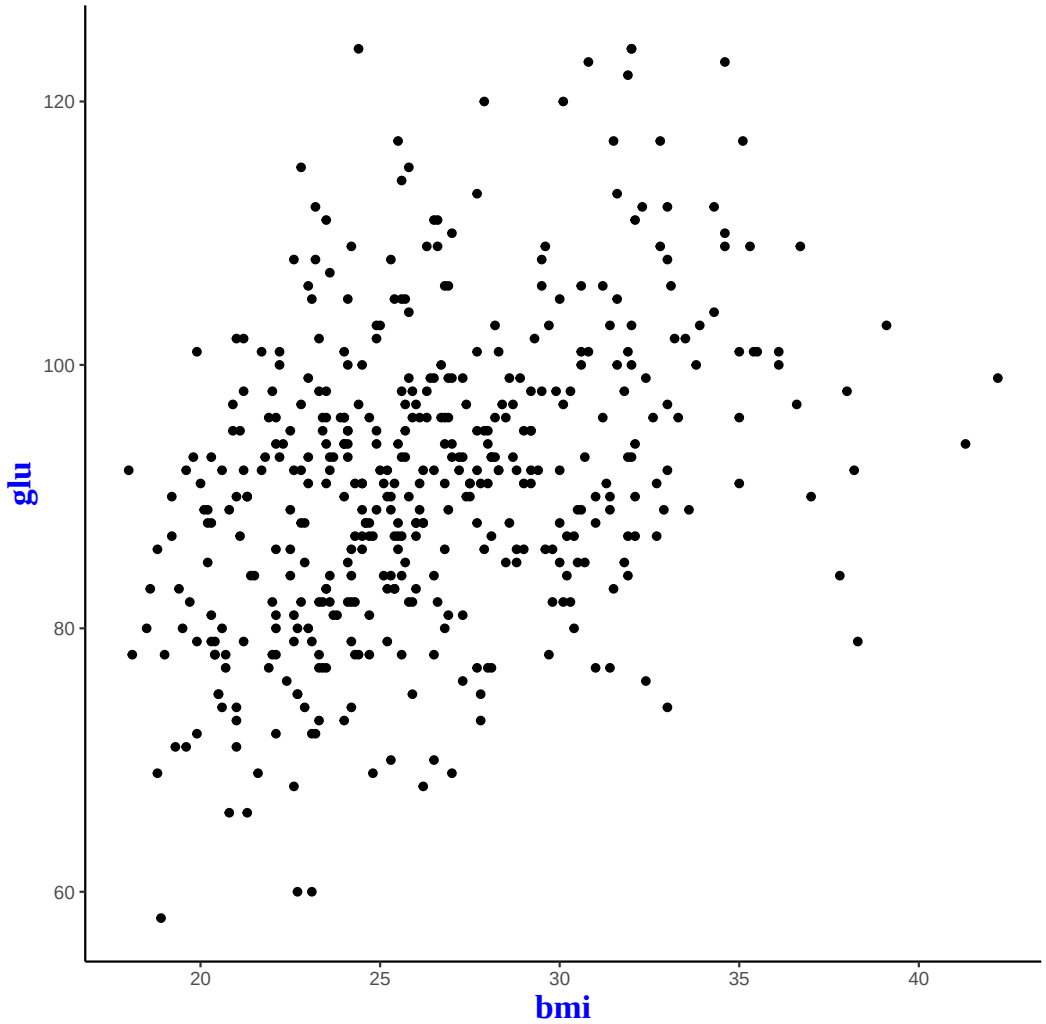


GGPlot: Themes

We can also modify individual theme elements.

```
p1.1 + theme_classic() +
    theme(axis.title = element_text(size = 16,
                                    family = "serif",
                                    face = "bold",
                                    color = "blue"),
          aspect.ratio = 1)
```

GGPlot: Themes

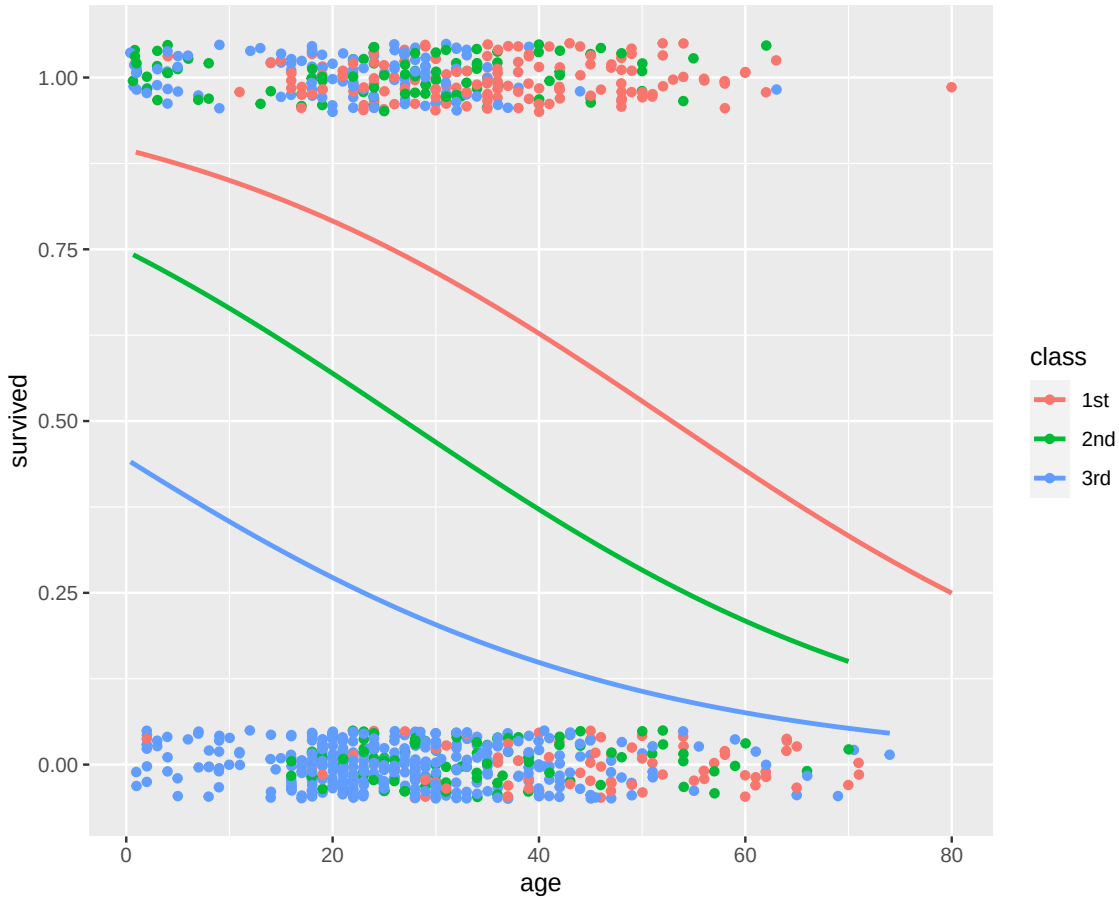


GGPlot: Facets

Faceting allow us to make arrays of conditional plots.

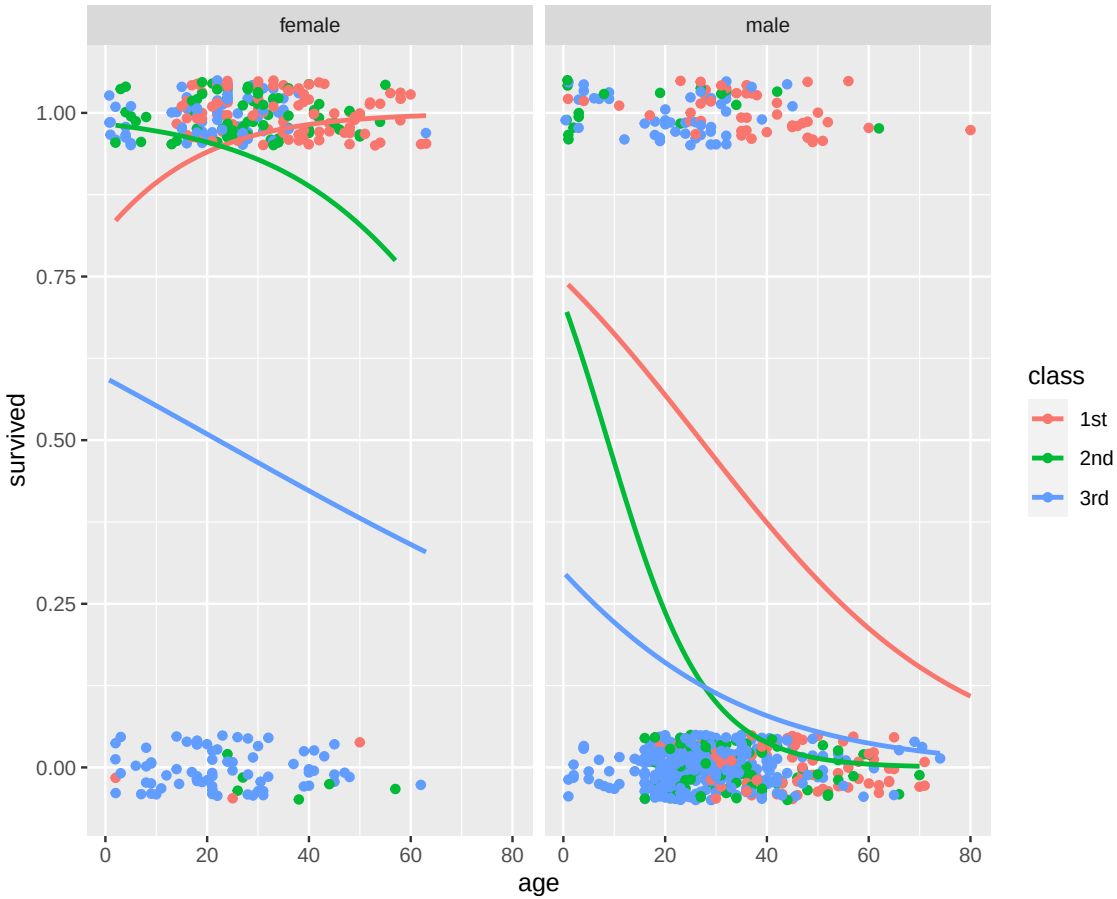
```
(p7 <- ggplot(titanic, aes(age, survived, color = class)) +
     geom_jitter(height = 0.05) +
     geom_smooth(method = "glm",
                 method.args = list(family = "binomial"),
                 se = FALSE)
)
```

GGPlot: Facets



GGPlot: Facets

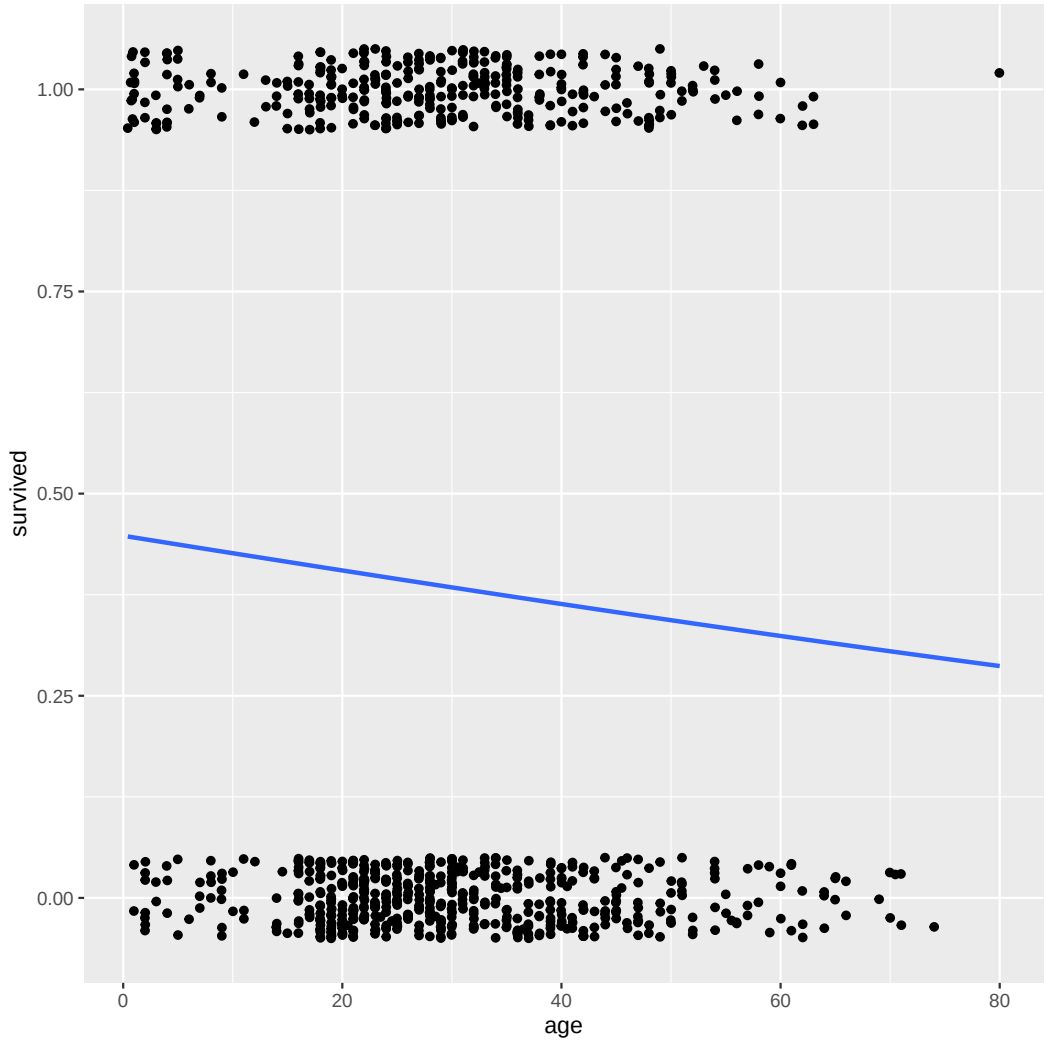
```
p7 + facet_wrap(vars(sex))
```



GGPlot: Facets

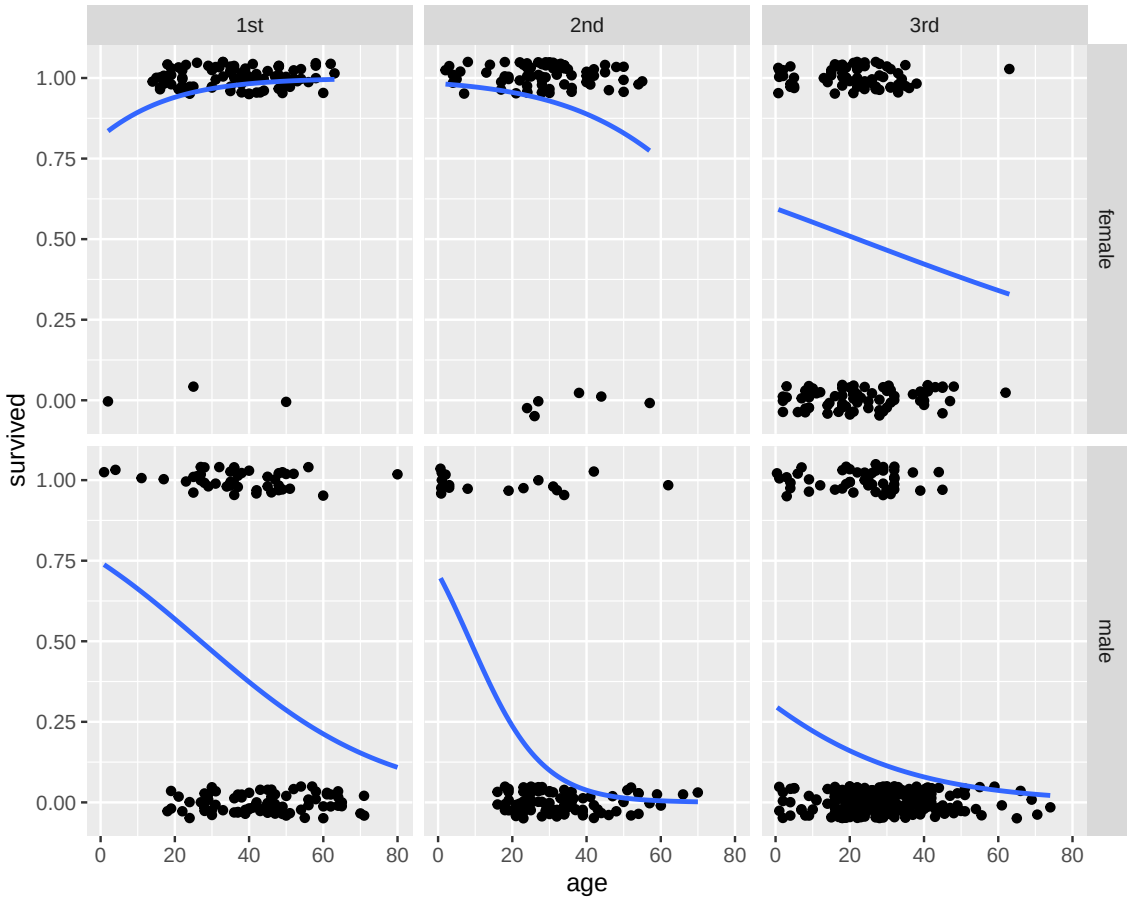
```
(p8 <- ggplot(titanic, aes(age, survived)) +
     geom_jitter(height = 0.05) +
     geom_smooth(method = "glm",
                 method.args = list(family = "binomial"),
                 se = FALSE)
)
```

GGPlot: Facets



GGPlot: Facets

```
p8 + facet_grid(vars(sex), vars(class))
```



GGPlot: Joining Multiple Figures

If we want to paste several different plots into a single figure (without faceting), we can use the utilities in the **gridExtra** package.

```
library(gridExtra)

grid.arrange(p1 + geom_point(),
             p3 + geom_boxplot(),
             p1 + geom_line(),
             p8 + facet_grid(vars(sex), vars(class)),
             ncol = 2)
```

GGPlot: Joining Multiple Figures