Data Exploration in R

Prof. Dr. Jan Kirenz

2020-11-22

Contents

Welcome

This book provides an introduction to data exploration in R. To use the code in this book, activate the following packages:

```r
library(tidyverse)
library(skimr)
library(gt)
```

To illustrate the different data exploration methods, we use the dataset `wage`, which contains wage and other data for a group of 3000 male workers in the Mid-Atlantic region (James et al. (2000)).

```r
library(tidyverse)

wage_df <- read_csv("https://raw.githubusercontent.com/kirenz/datasets/master/wage.csv")
```

The data frame includes 3000 observations on the following 11 variables:

- `X1`: An ID variable
- `year`: Year that wage information was recorded
- `age`: Age of worker
- `maritl`: A factor with levels: 1. Never Married 2. Married 3. Widowed 4. Divorced and 5. Separated indicating marital status
- `race`: A factor with levels: 1. White 2. Black 3. Asian and 4. Other indicating race
- `education`: A factor with levels: 1. < HS Grad 2. HS Grad 3. Some College 4. College Grad and 5. Advanced Degree indicating education level
- `region`: Region of the country (mid-atlantic only)
- `jobclass`: A factor with levels: 1. Industrial and 2. Information indicating type of job
- `health`: A factor with levels: 1. <=Good and 2. >=Very Good indicating health level of worker

- `health_ins`: A factor with levels: 1. Yes and 2. No indicating whether worker has health insurance
- `logwage`: Log of workers wage
- `wage`: Workers raw wage

Note that this book mainly covers the use of a collection of R packages called the tidyverse, an ecosystem of packages designed with common APIs and a shared philosophy. An R package is simply a bundle of functions, documentation, and data sets. There are about 25 packages in the tidyverse and they are especially designed for data science and share an underlying design philosophy, grammar, and data structures.

Chapter 1

Counts and Tables

You should use this method if the data is:

- Categorical

In this chapter you will learn how to do some simple data explorations for categorical variables using tables (also called tables) and simple counts.

1.1 Simple counts

Get an overview of the variable `maritl` and sort the values. We use the function gt() to print nice tables:

```
wage_df %>%
  count(maritl,
  sort = TRUE) %>%
  gt()
```

maritl	n
2. Married	2074
1. Never Married	648
4. Divorced	204
5. Separated	55
3. Widowed	19

Get an overview of the combined variables `maritl` and `education`:

```
wage_df %>%
  count(maritl, education) %>%
   gt()
```

maritl	education	n
1. Never Married	1. < HS Grad	62
1. Never Married	2. HS Grad	219
1. Never Married	3. Some College	164
1. Never Married	4. College Grad	143
1. Never Married	5. Advanced Degree	60
2. Married	1. < HS Grad	174
2. Married	2. HS Grad	651
2. Married	3. Some College	421
2. Married	4. College Grad	487
2. Married	5. Advanced Degree	341
3. Widowed	1. < HS Grad	2
3. Widowed	2. HS Grad	8
3. Widowed	3. Some College	2
3. Widowed	4. College Grad	5
3. Widowed	5. Advanced Degree	2
4. Divorced	1. < HS Grad	16
4. Divorced	2. HS Grad	73
4. Divorced	3. Some College	52
4. Divorced	4. College Grad	41
4. Divorced	5. Advanced Degree	22
5. Separated	1. < HS Grad	14
5. Separated	2. HS Grad	20
5. Separated	3. Some College	11
5. Separated	4. College Grad	9
5. Separated	5. Advanced Degree	1

Obtain the sum of a quantitative variable (`wage`) for the different levels of a categorical variable (`maritl`):

```
wage_df %>%
  count(maritl,
        wt = wage,
        name = "Wage_sum") %>%
  gt()
```

maritl	Wage_sum
1. Never Married	60092.052

2. Married	246516.180
3. Widowed	1891.234
4. Divorced	21044.489
5. Separated	5566.868

1.2 Total counts

Total counts are an useful way to represent the observations that fall into each combination of the levels of categorical variables. We create a contingency table of the two categorical variables `jobclass` and `race` and call the result `tab`:

```
tab <- table(wage_df$jobclass, wage_df$race)
```

1.3 Joint proportions

We can also view the percentage of each cell in relation to the total amount of all observations (here n = 3000). Therefore, you have to simply divide the numbers from our total counts with 3.000.

The following code generates tables of *joint* proportions:

```
# joint proportions
prop.table(tab)
```

```
##
##                   1. White    2. Black    3. Asian    4. Other
##   1. Industrial  0.441666667 0.037000000 0.028666667 0.007333333
##   2. Information 0.385000000 0.060666667 0.034666667 0.005000000
```

For example, around 44% of all people in the dataset are white industrial workers.

1.4 Conditional proportions: columns

You also may want to know the probability that workers have a certain jobclass, given that they have a particular ethnical background. This is a so called conditional probability. Conditional probability represents the chance that one event will occur given that a second event has already occurred.

The following code generates tables of *conditional* proportions:

```r
# conditional on columns
prop.table(tab, 2)
```

```
##
##                  1. White  2. Black  3. Asian  4. Other
##   1. Industrial  0.5342742 0.3788396 0.4526316 0.5945946
##   2. Information 0.4657258 0.6211604 0.5473684 0.4054054
```

We performed a columnwise evaluation and are now able to answer the following question:

- Approximately what proportion of all white workers are industrial workers?
- The answer is: around 53%.

1.5 Conditional proportions: rows

Now we want to obtain the probability that workers have a certain race, given their jobclass.

```r
# conditional on rows
prop.table(tab, 1)
```

```
##
##                   1. White   2. Black   3. Asian   4. Other
##   1. Industrial  0.85816062 0.07189119 0.05569948 0.01424870
##   2. Information 0.79326923 0.12500000 0.07142857 0.01030220
```

We performed a rowwise evaluation and are now able to answer the following question:

- Approximately what proportion of all industrial workers are white?
- The answer is: around 86%.

1.6 Chi-squared Test of Independence

Finally, let's test the hypothesis whether the variable `jobclass` is independent of the variable `race` at .05 significance level.

```r
chisq.test(tab)
```

```
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 29.331, df = 3, p-value = 1.908e-06
```

As the p-value is smaller than the .05 significance level, we reject the null hypothesis that the jobclass is independent of the race of the workers.

Chapter 2

Heatmap

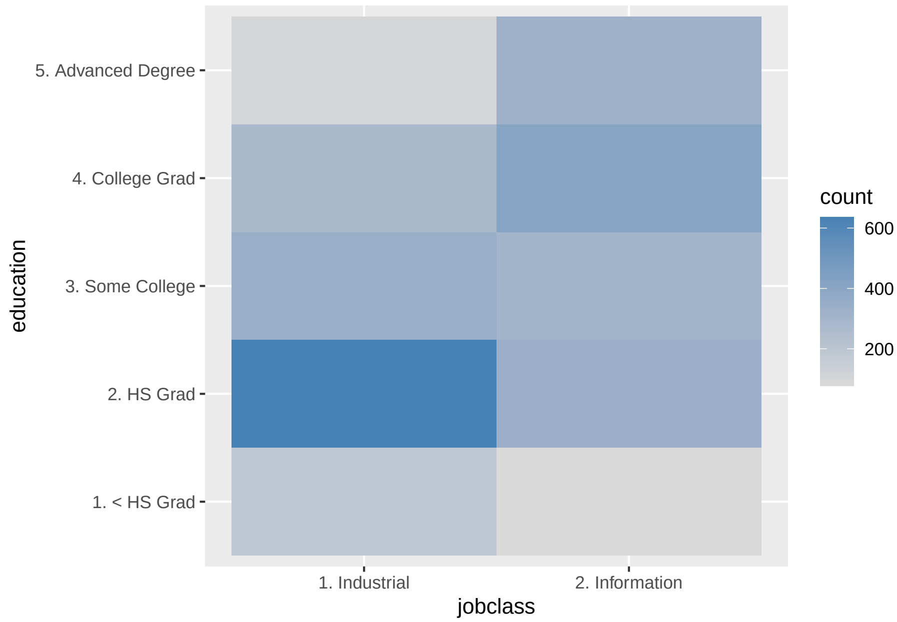
You should use this method if the data is:

- Categorical

In this chapter you will learn how to do some simple data explorations for categorical variables using heatmaps with the function geom_bin2d()

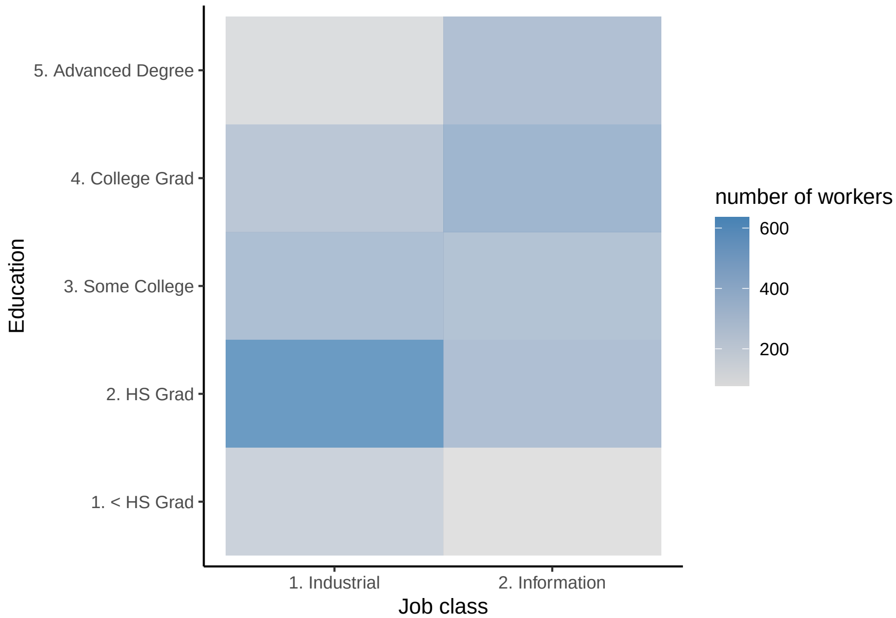
Basic plot:

```
wage_df %>%
  ggplot(aes(jobclass, education)) +
  geom_bin2d() +
  scale_fill_gradient(low = "gray85", high = "steelblue")
```



Plot with some adjustments:

```
wage_df %>%
  ggplot(aes(jobclass, education)) +
  geom_bin2d(binwidth = c(1, 1), alpha = 0.8) +
  theme_classic() +
  scale_fill_gradient(low = "gray85", high = "steelblue") +
  labs(fill = "number of\ workers", y = "Education", x = "Job class")
```


5. Advanced Degree
4. College Grad
3. Some College
2. HS Grad
1. < HS Grad
Education
1. Industrial
2. Information
Job class
number of workers
600
400
200

Chapter 3

Barplot

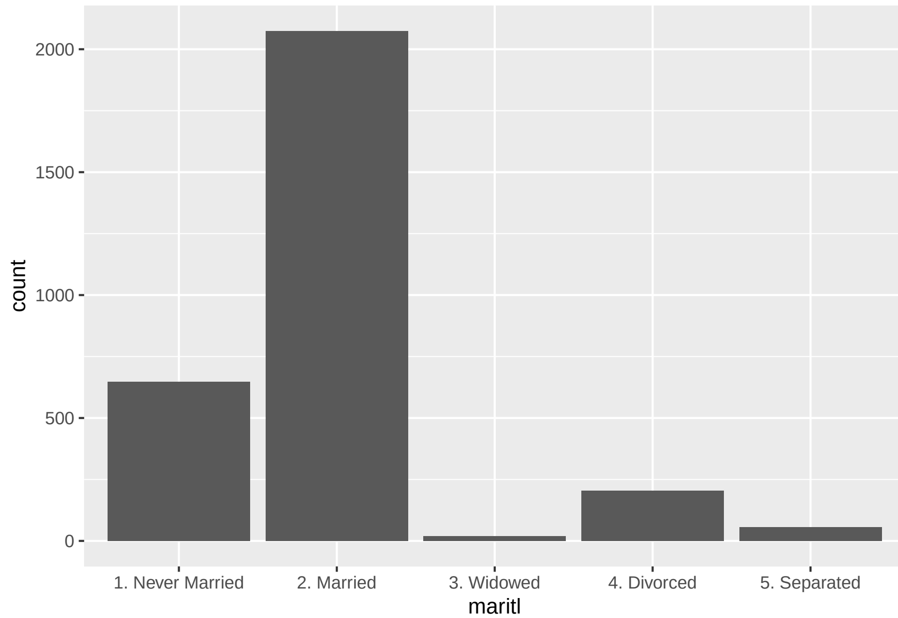
You should use this method if the data is:

- Categorical

In this chapter you will learn how to do some simple data explorations for categorical variables using barplots.

3.1 One variable

```
wage_df %>%
  ggplot(aes(x = maritl)) +
  geom_bar()
```

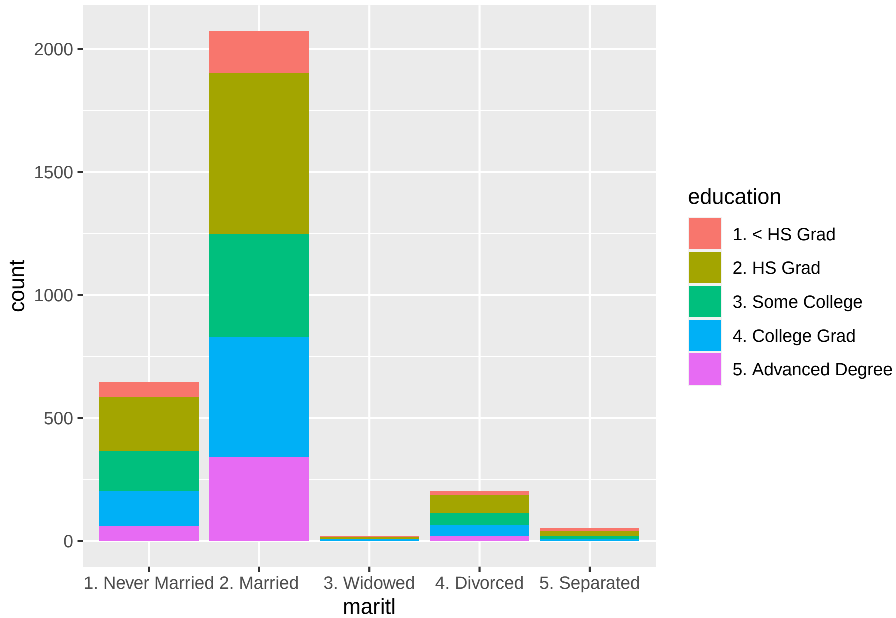


3.2 Two variables

3.2.1 Stacked barplot

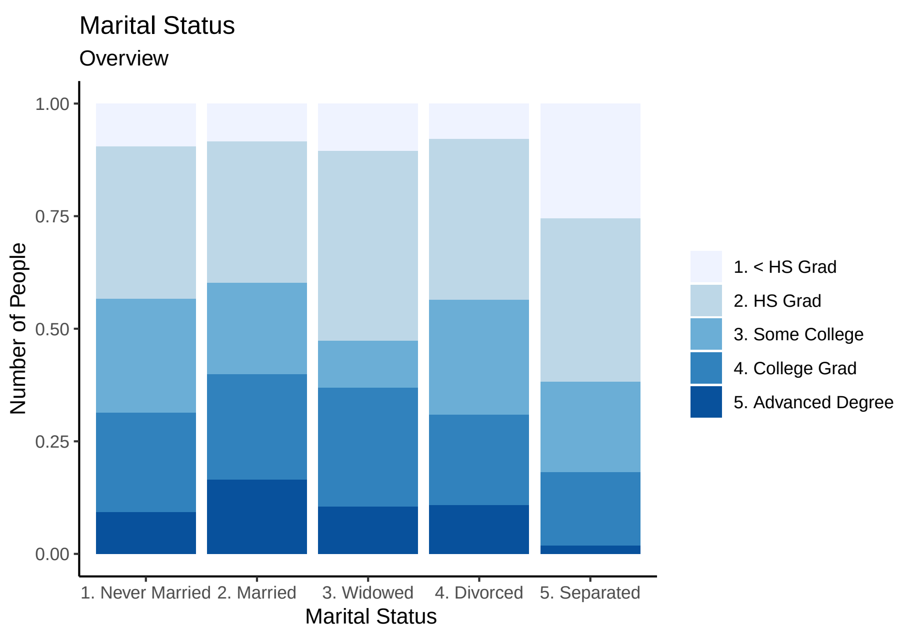
Absolute values:

```
wage_df %>%
  ggplot(aes(x = maritl, fill = education)) +
  geom_bar()
```



Relative values:

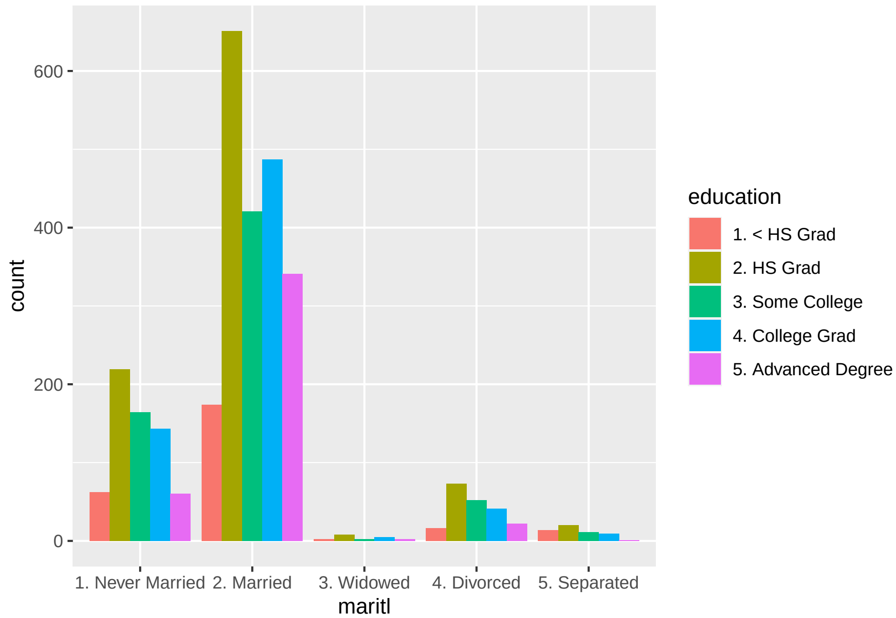
```r
wage_df %>%
  ggplot(aes(x = maritl, fill = education)) +
  geom_bar(position = "fill") +
  ggtitle("Marital Status", "Overview") +
  xlab(" Marital Status") +
  ylab("Number of People") +
  theme_classic() +
  scale_fill_brewer(palette = "Blues") +
  theme(legend.title = element_blank())
```



3.2.2 Side-by-side barplot

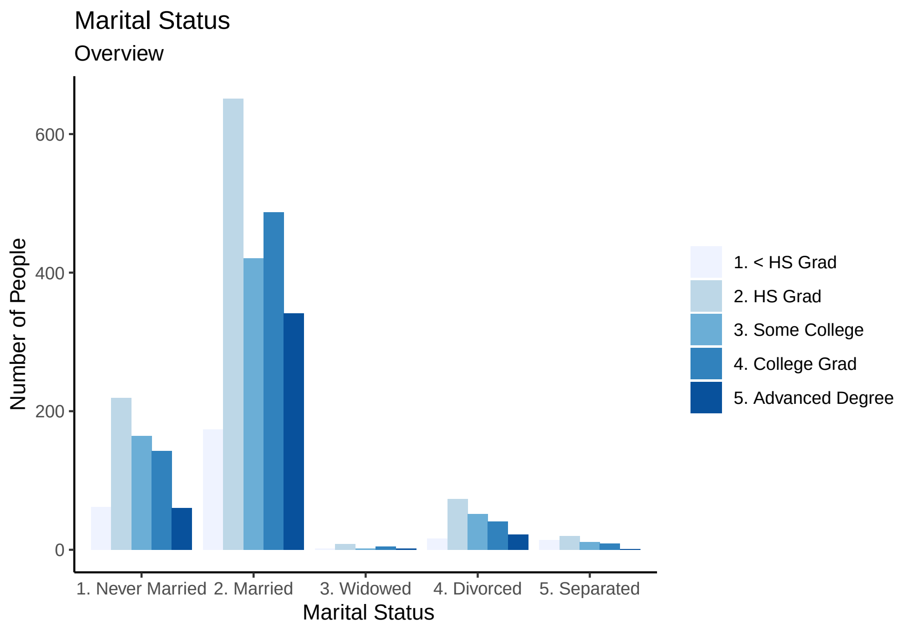
Basic plot:

```
wage_df %>%
  ggplot(aes(x = maritl, fill = education)) +
  geom_bar( position = "dodge")
```



Plot with some adjustments:

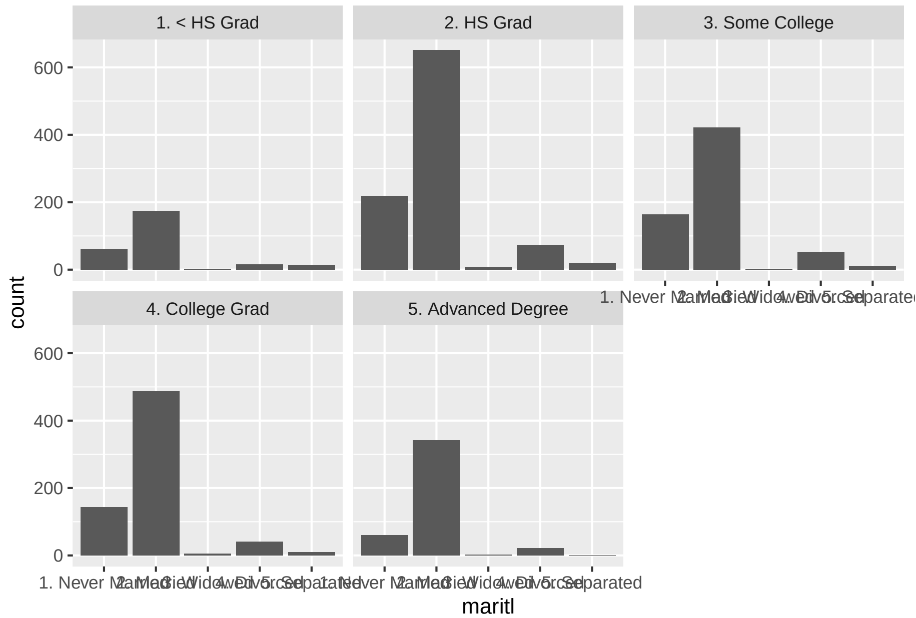
```
wage_df %>%
  ggplot(aes(x = maritl, fill = education)) +
  geom_bar( position = "dodge") +
  ggtitle("Marital Status", "Overview") +
  xlab(" Marital Status") +
  ylab("Number of People") +
  theme_classic() +
  scale_fill_brewer(palette = "Blues") +
  theme(legend.title = element_blank())
```



3.2.3 Faceted barplot

Basic plot:

```
wage_df %>%
  ggplot(aes(x = maritl)) +
  geom_bar(position = "dodge") +
  facet_wrap(~ education)
```

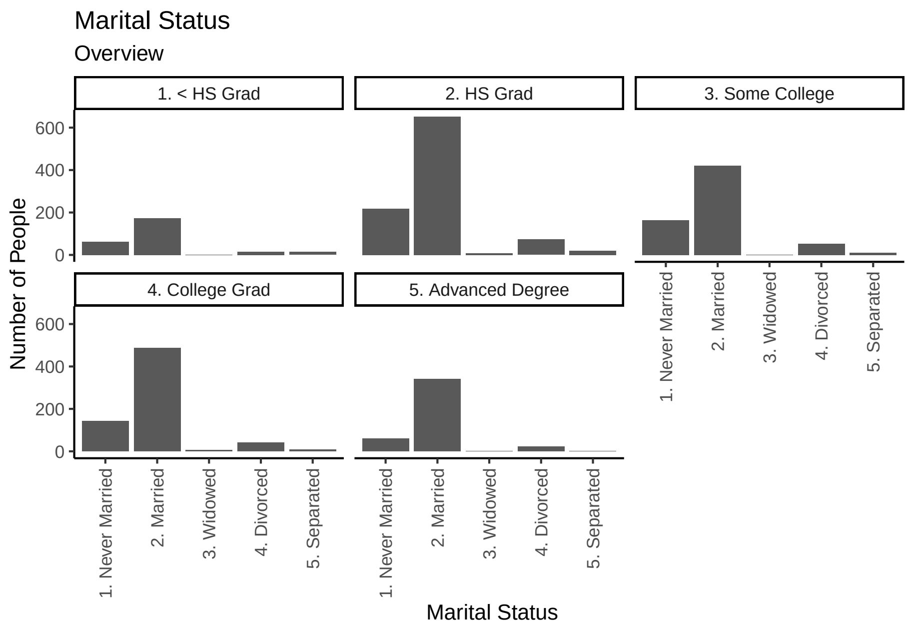


Plot with some adjustments to change our x labels using justifications):

Horizontal and vertical justification have the same parameterisation, either a string ("top", "middle", "bottom", "left", "center", "right") or a number between 0 and 1:"

- top = 1, middle = 0.5, bottom = 0
- left = 0, center = 0.5, right = 1

```
wage_df %>%
  ggplot(aes(x = maritl)) +
  geom_bar(position = "dodge") +
  facet_wrap(~ education) +
  ggtitle("Marital Status", "Overview") +
  xlab(" Marital Status") +
  ylab("Number of People") +
  theme_classic() +
  theme(legend.title = element_blank()) +
  theme(axis.text.x = element_text(angle = 90,
                                   vjust = 0.5,
                                   hjust=  1))
```


Marital Status
Overview
1. < HS Grad
2. HS Grad
3. Some College
4. College Grad
5. Advanced Degree
Number of People
0
200
400
600
1. Never Married
2. Married
3. Widowed
4. Divorced
5. Separated
Marital Status

Chapter 4

Histogram

You should use this method if the data is:

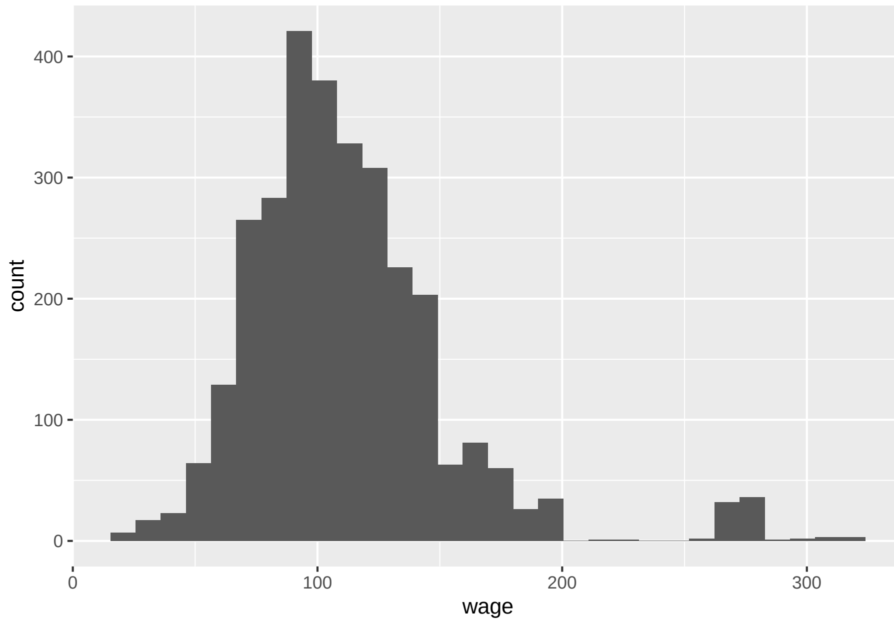
- Numeric

4.1 One variable

4.1.1 Basic Histogram

Basic plot:

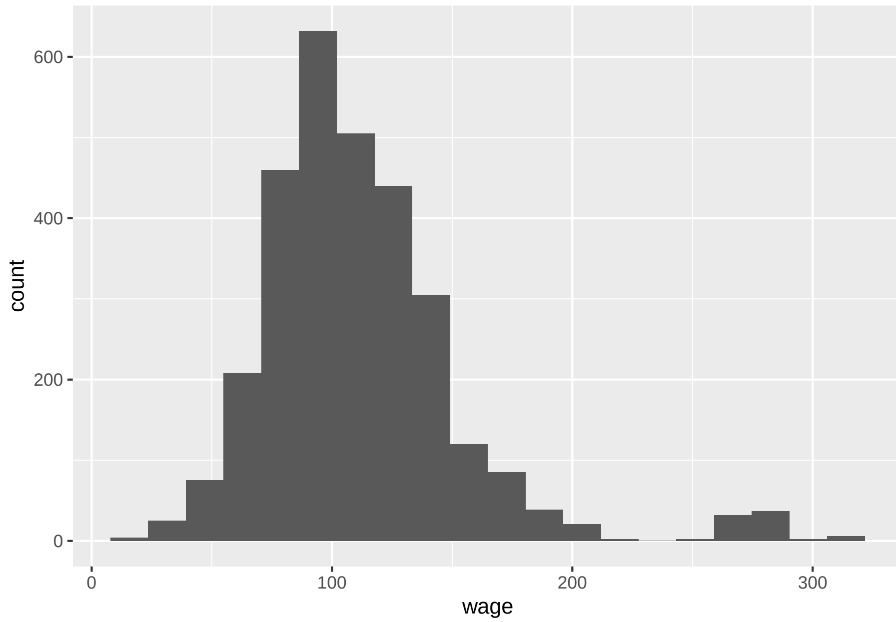
```
wage_df %>%
  ggplot(aes(x = wage)) +
  geom_histogram()
```



4.1.2 Bins

Adjust number of bins:

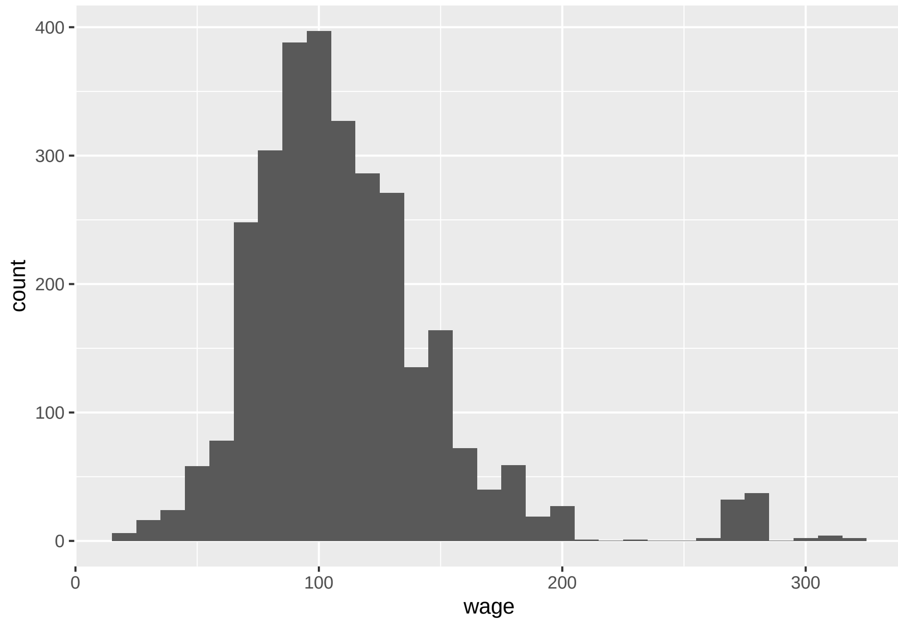
```
wage_df %>%
  ggplot(aes(x = wage)) +
  geom_histogram(bins = 20)
```



4.1.3 Binwidth

Instead of using bins, you can also change the binwidth:

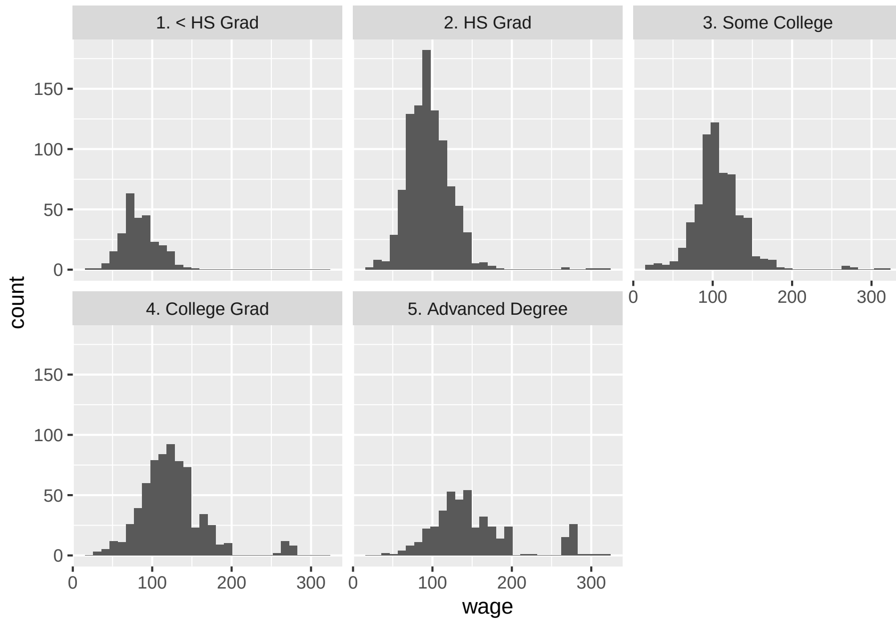
```
wage_df %>%
  ggplot(aes(x = wage)) +
  geom_histogram(binwidth = 10)
```



4.2 Two variables

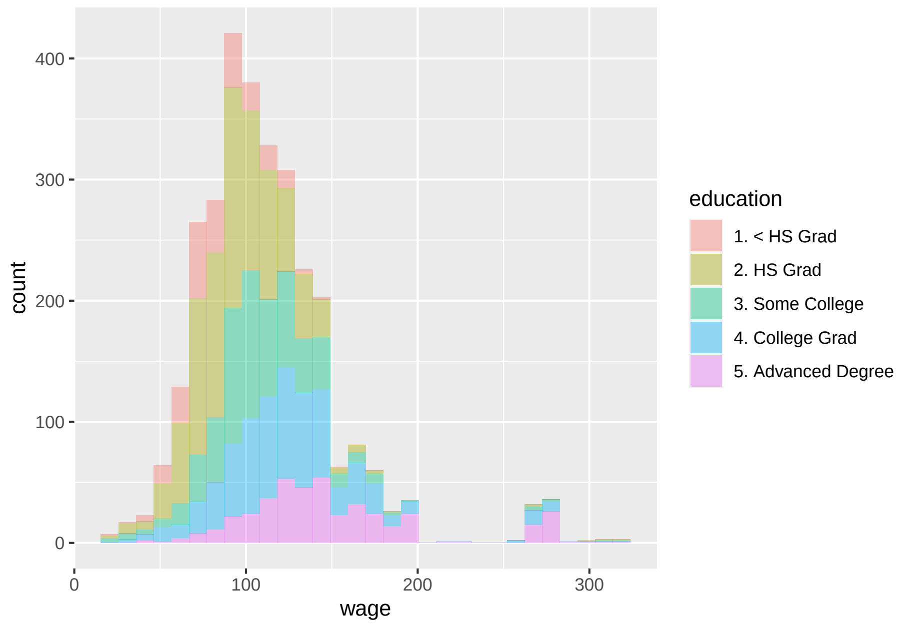
4.2.1 Faceted histogram

```
wage_df %>%
  ggplot(aes(x = wage)) +
  geom_histogram() +
  facet_wrap( ~ education)
```



4.2.2 Stacked histogram

```
wage_df %>%
  ggplot(aes(x = wage, fill = education)) +
  geom_histogram(alpha = 0.4)
```


400
300
200
100
0
count
0
100
200
300
wage
education
1. < HS Grad
2. HS Grad
3. Some College
4. College Grad
5. Advanced Degree

Chapter 5

Density plots

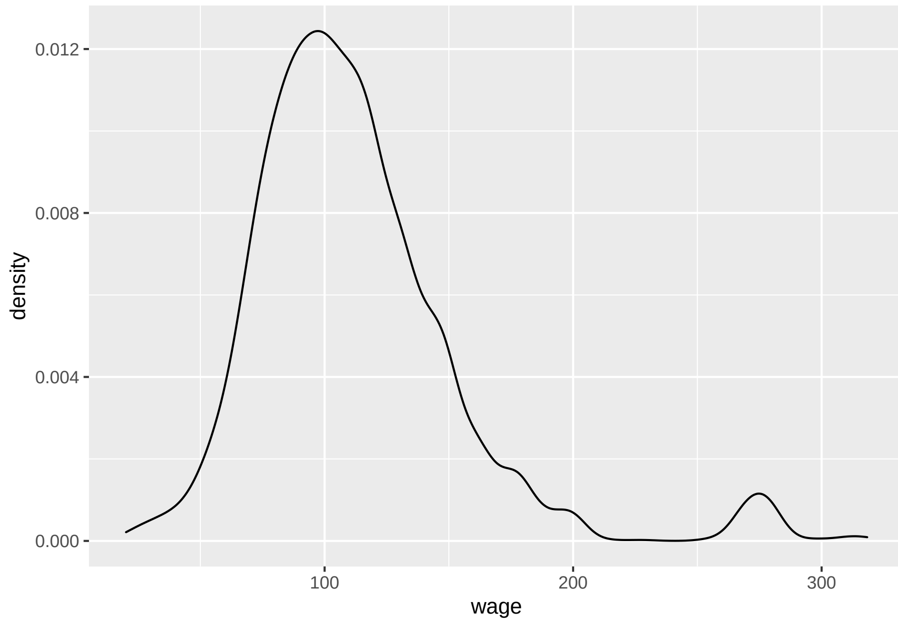
You should use this method if the data is:

- Numeric and continuous

In this chapter you will learn how to do some simple data explorations for numerical variables using density plots.

5.1 One variable

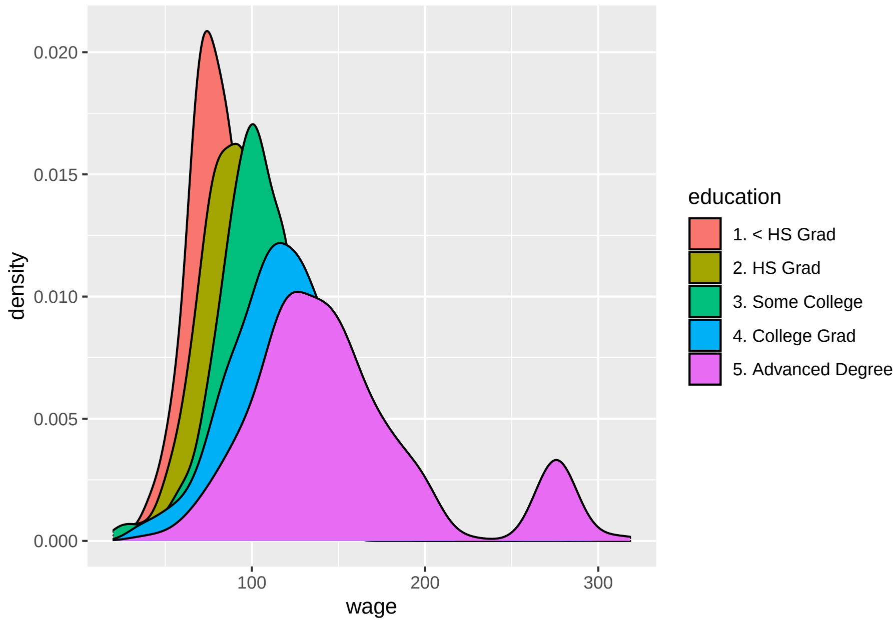
```
wage_df %>%
  ggplot(aes(x = wage)) +
  geom_density()
```



5.2 Two variables

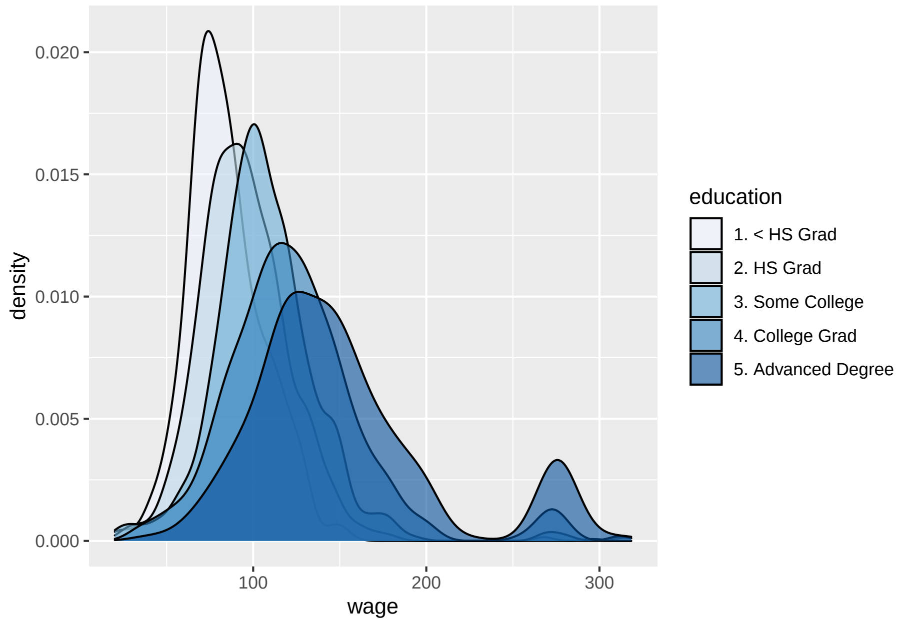
Combine your numeric variable with a categorical variable:

```
wage_df %>%
  ggplot(aes(x = wage, fill = education)) +
  geom_density()
```



Make some adjustments:

```
wage_df %>%
  ggplot(aes(x = wage, fill = education)) +
  geom_density(alpha = 0.6) +
  scale_fill_brewer(palette = "Blues")
```


0.020
0.015
0.010
0.005
0.000
density
100
200
300
wage
education
1. < HS Grad
2. HS Grad
3. Some College
4. College Grad
5. Advanced Degree

Chapter 6

Boxplot

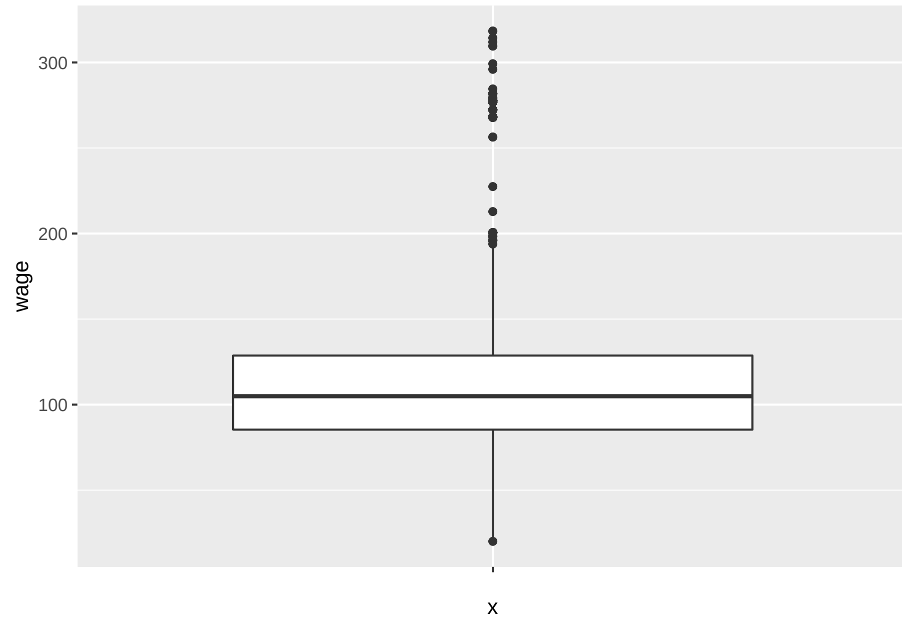
You should use this method if the data is:

- Categorical (at least ordinal) or
- Numerical

In this chapter you will learn how to do some simple data explorations for categorical (ordinal) and numerical variables using boxplots.

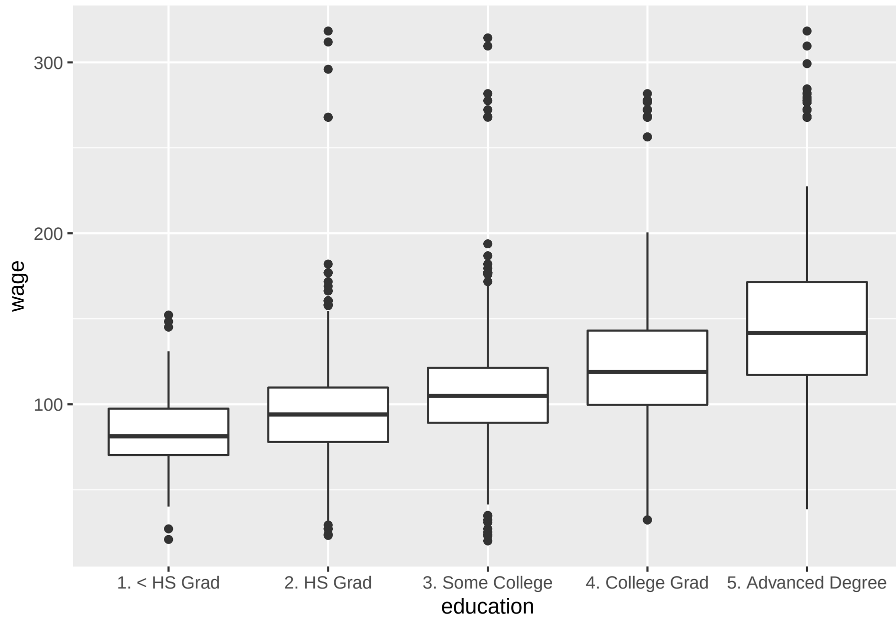
6.1 One variable

```
wage_df %>%
  ggplot(aes( x = "", y= wage)) +
  geom_boxplot()
```



6.2 Two variables

```
wage_df %>%
  ggplot(aes(x = education, y = wage)) +
  geom_boxplot()
```


300
200
100
wage
1. < HS Grad
2. HS Grad
3. Some College
4. College Grad
5. Advanced Degree
education

Chapter 7

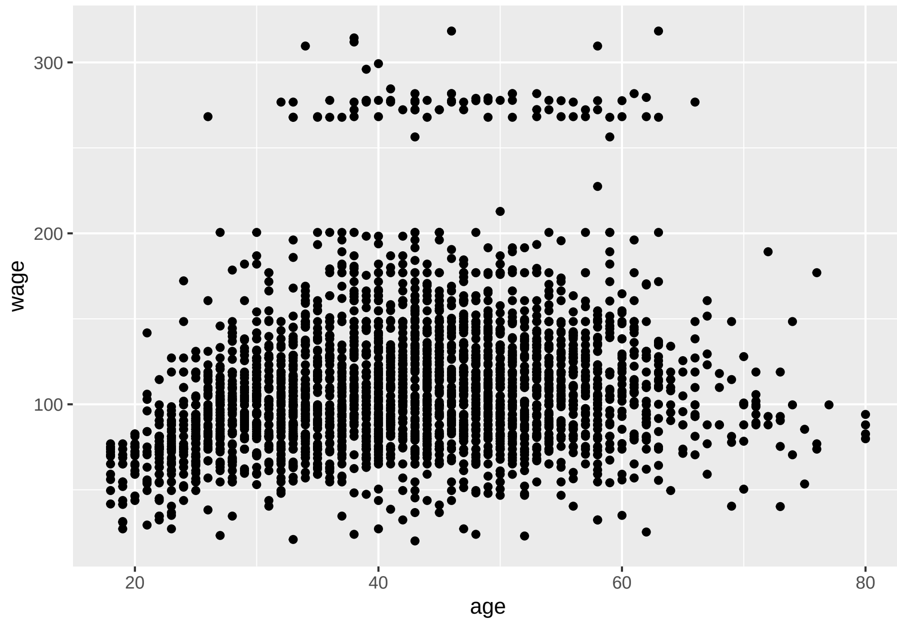
Scatterplot

You should use this method if the data is:

- Numerical

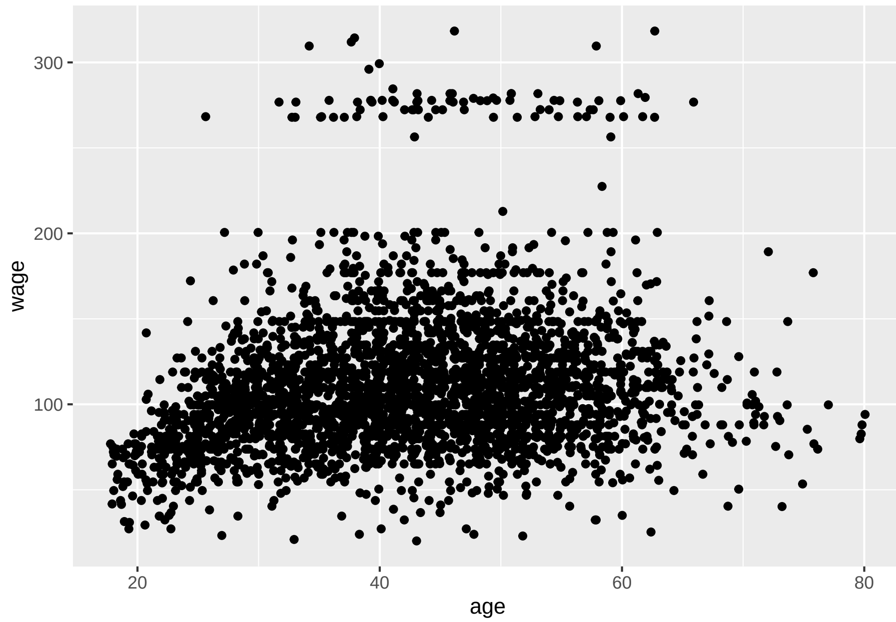
7.1 Two numeric variables

```
wage_df %>%
  ggplot(aes(x = age, y = wage)) +
  geom_point()
```



Use jitter

```
wage_df %>%
  ggplot(aes(x = age, y = wage)) +
  geom_jitter()
```



7.2 Two numeric, one categorical

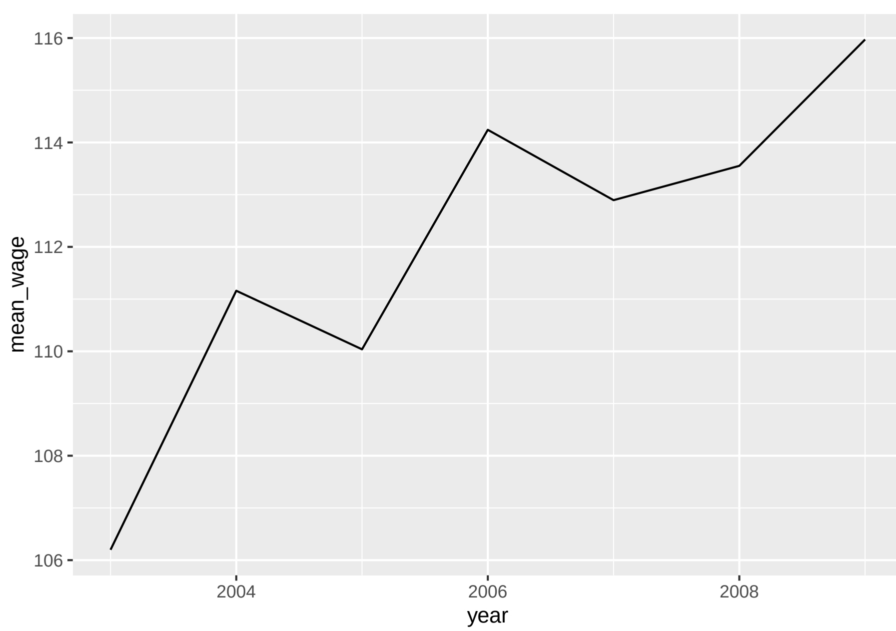
```
wage_df %>%
  ggplot(aes(x = age, y = wage, color = jobclass)) +
  geom_jitter()
```


wage
300
200
100
20
40
60
80
age
jobclass
1. Industrial
2. Information

Chapter 8

Line graph

```
wage_df %>%
  group_by(year) %>%
  mutate(mean_wage = mean(wage, na.rm = TRUE)) %>%
  ungroup() %>%
  ggplot(aes(x = year, y = mean_wage)) +
  geom_line()
```



Bibliography

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2000). *An introduction to Statistical Learning*, volume 7. New York: Springer.