Data Visualization

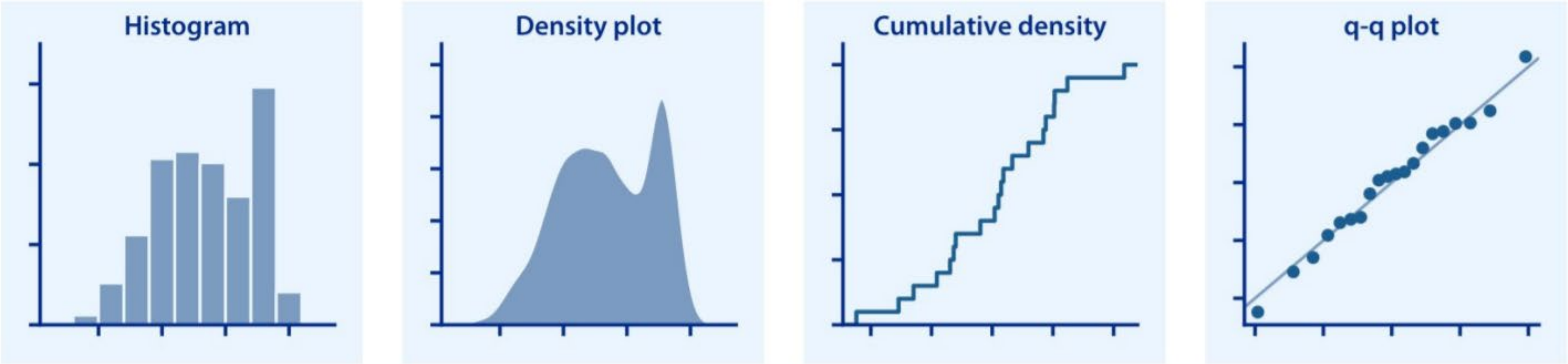
Prof. Dr. Javier Valdes
Javier.valdes@th-deg.de

IAI - Institut für Angewandte Informatik
Institute for Applied Informatics

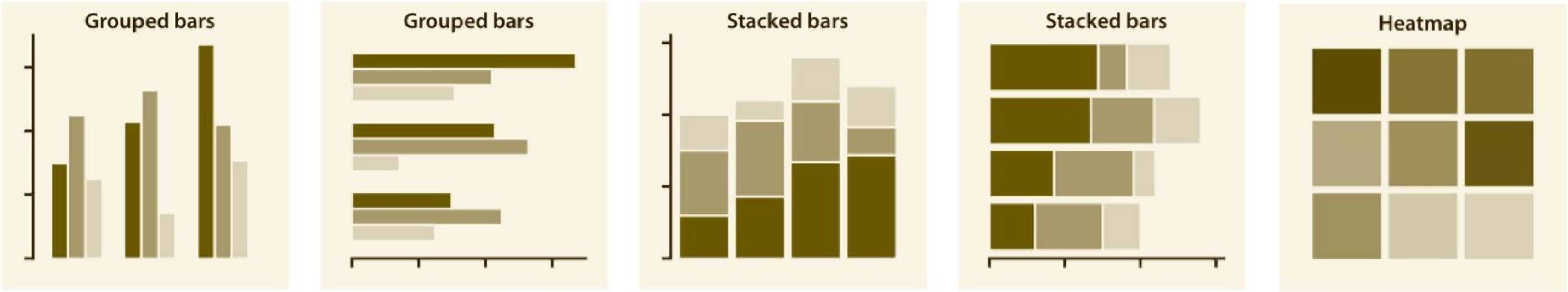
Technische Hochschule Deggendorf
Technologie Campus Freyung
Grafenauer Str. 22, D- 94078 Freyung, Germany

Tel.: +49 8551 91764 40
Fax: +49 8551 91764 69

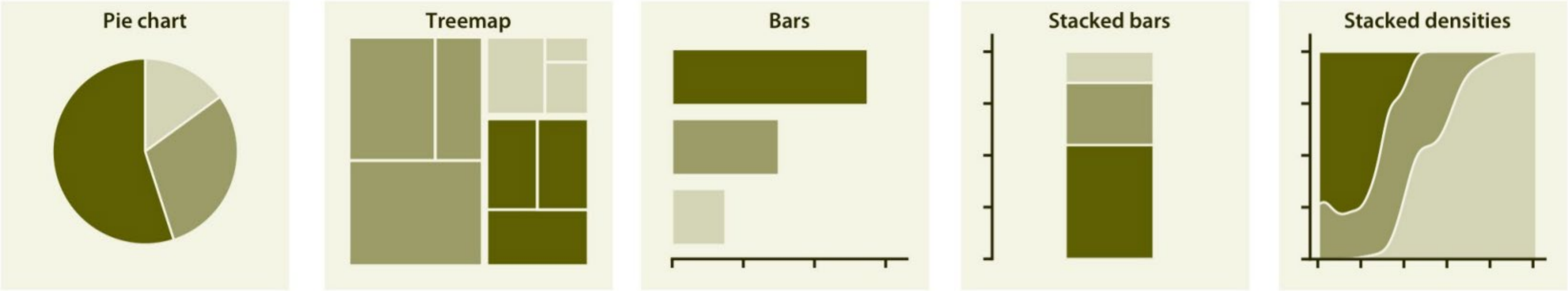
Distributions



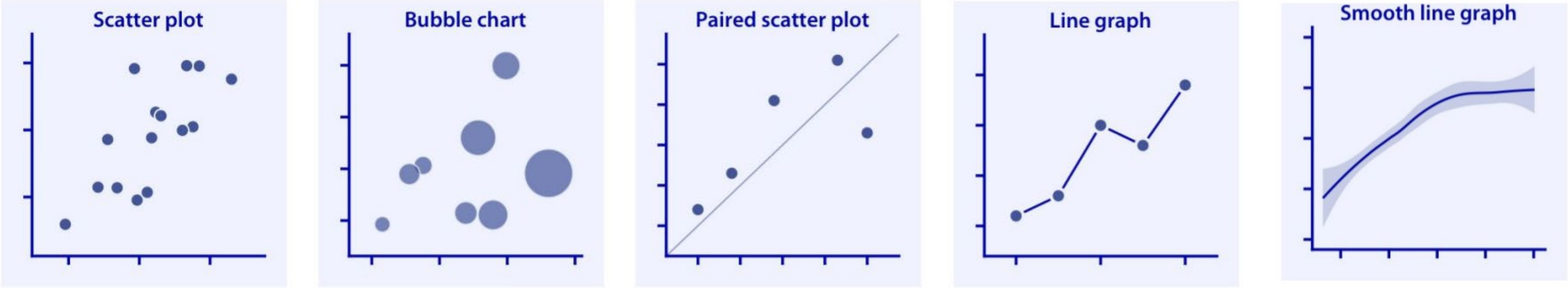
Comparisons



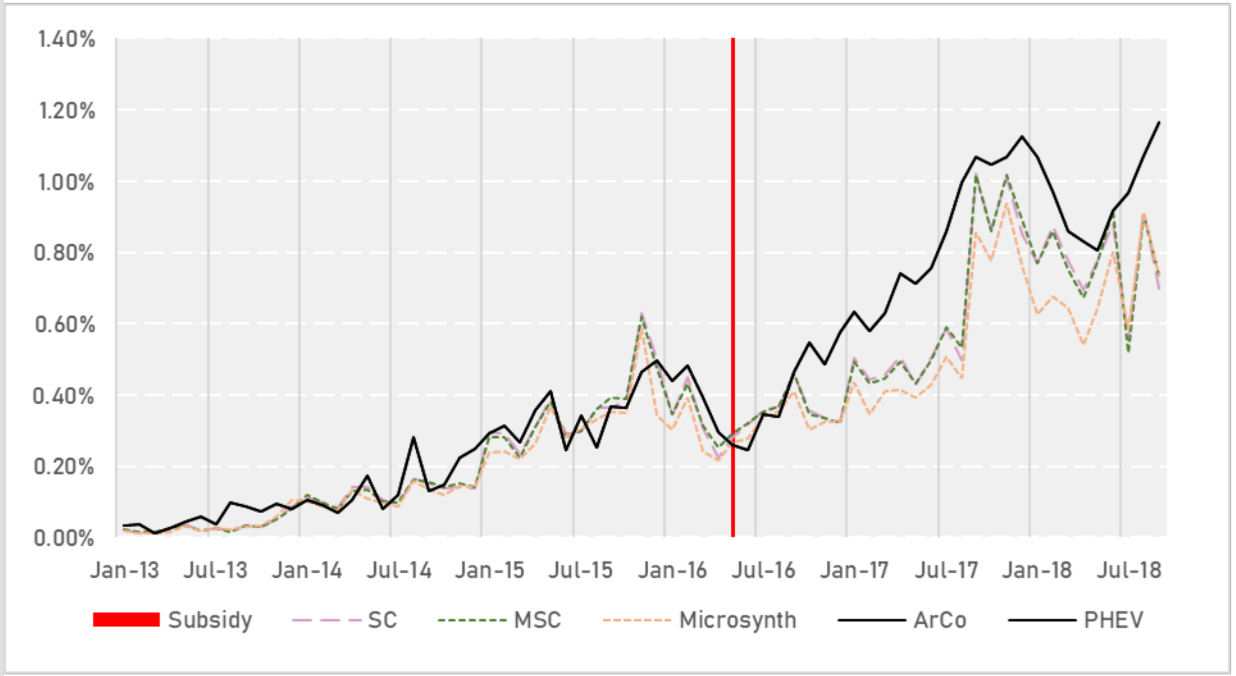
Proportions



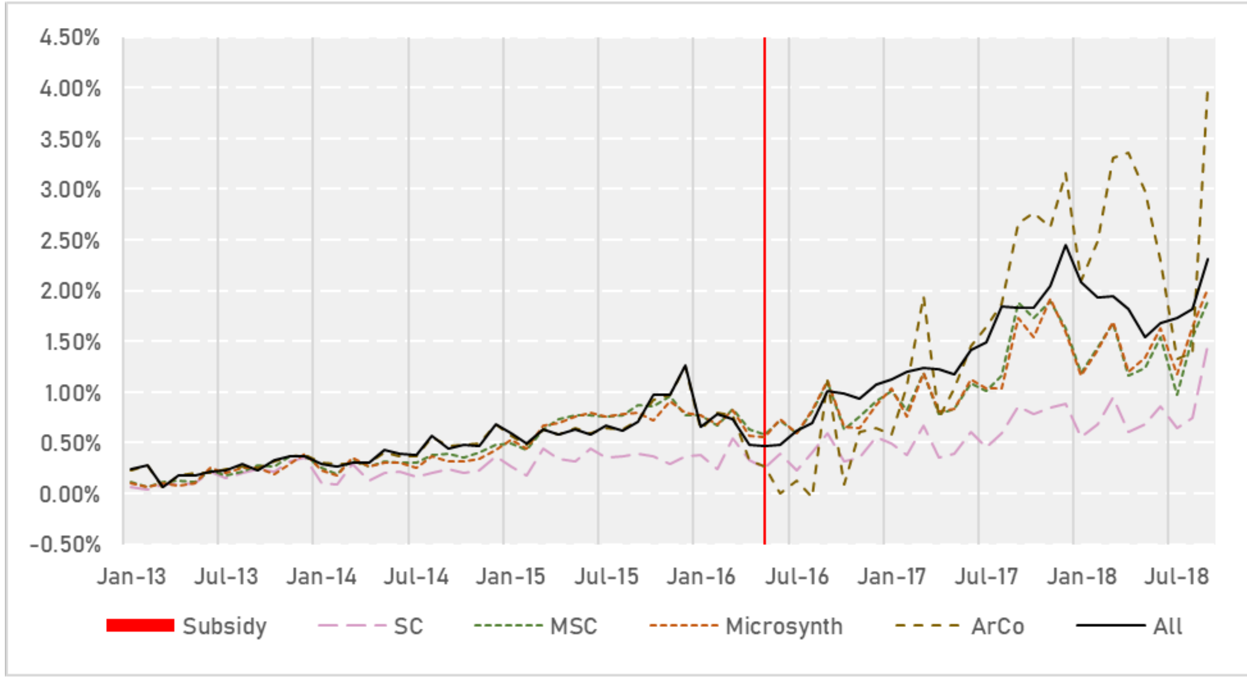
x-y relationships



Graphs



1.40%
1.20%
1.00%
0.80%
0.60%
0.40%
0.20%
0.00%
Jan-13 Jul-13 Jan-14 Jul-14 Jan-15 Jul-15 Jan-16 Jul-16 Jan-17 Jul-17 Jan-18 Jul-18
Subsidy
SC
MSC
Microsynth
ArCo
PHEV

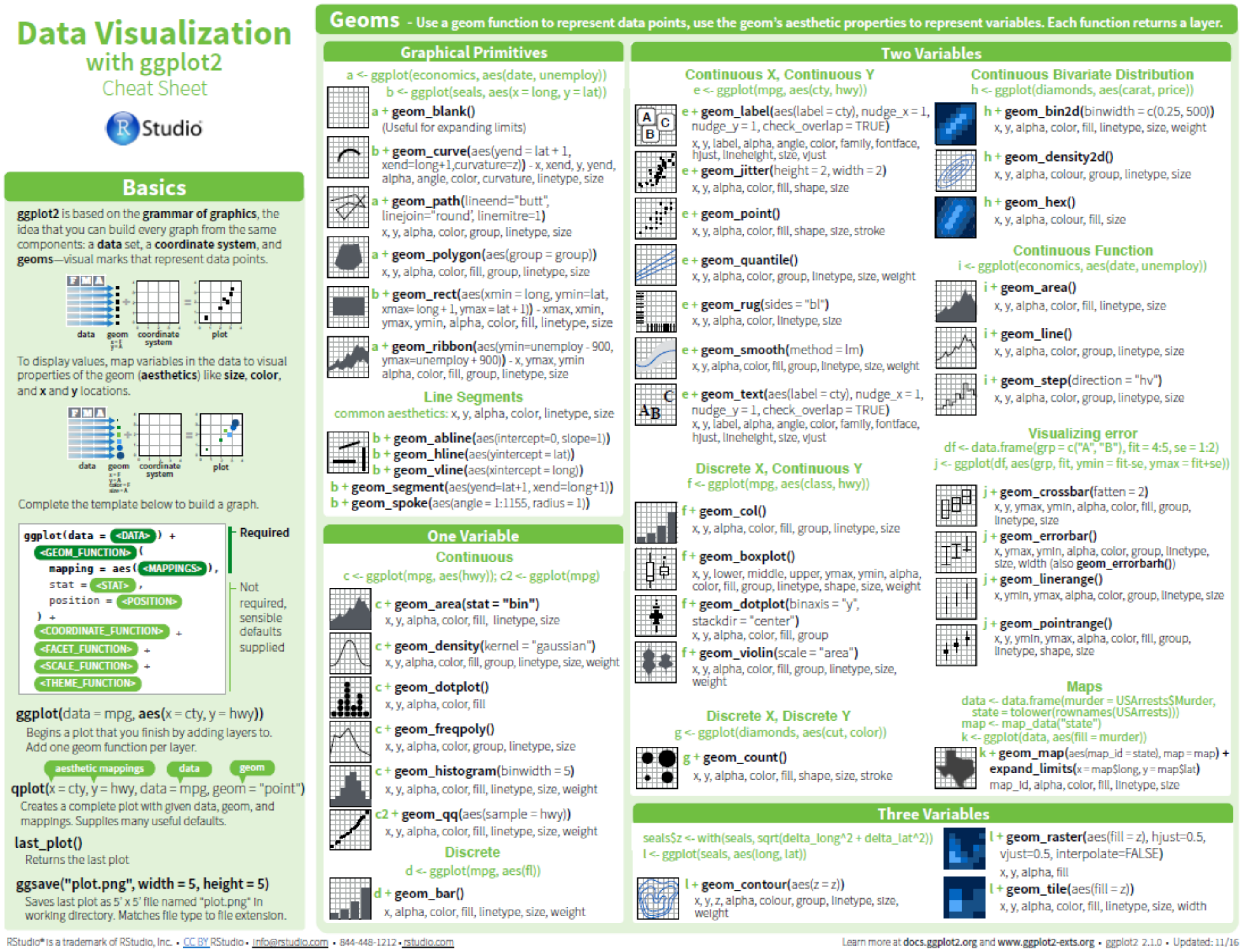


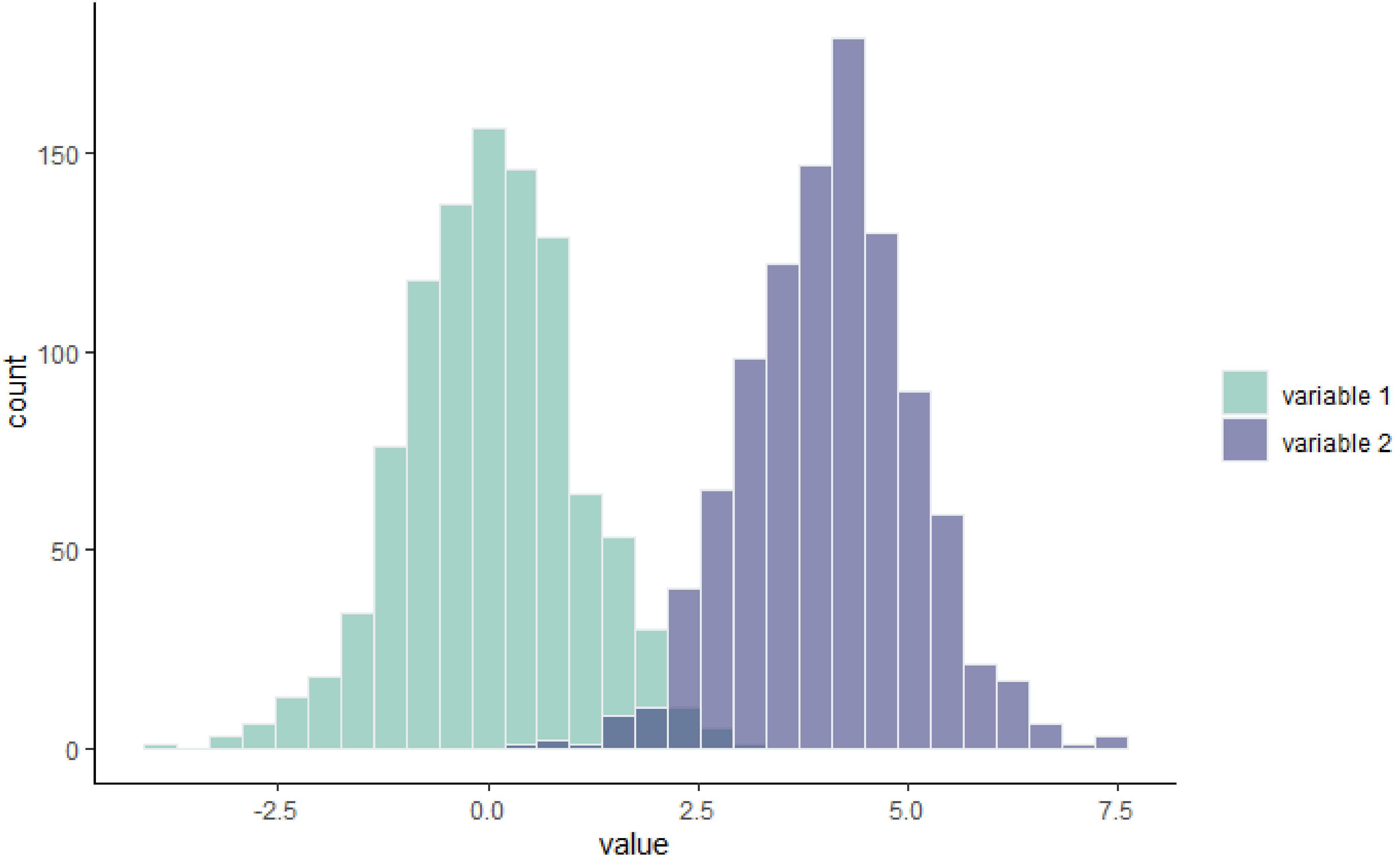
4.50%
4.00%
3.50%
3.00%
2.50%
2.00%
1.50%
1.00%
0.50%
0.00%
-0.50%
Jan-13 Jul-13 Jan-14 Jul-14 Jan-15 Jul-15 Jan-16 Jul-16 Jan-17 Jul-17 Jan-18 Jul-18
Subsidy
SC
MSC
Microsynth
ArCo
All

Plotting with ggplot2

The **concept** behind **ggplot2** divides plot into three (even more) different **fundamental** parts: Plot = data + Aesthetics + Geometry

- **data** is a data frame
- **Aesthetics** is used to indicate x and y variables. It can also be used to control the **color**, the **size** or the **shape** of points, the height of bars, etc…..
- **Geometry** defines the type of graphics (**histogram**, **box plot**, **line plot**, **density plot**, **dot plot**, ….)
- All the rest….

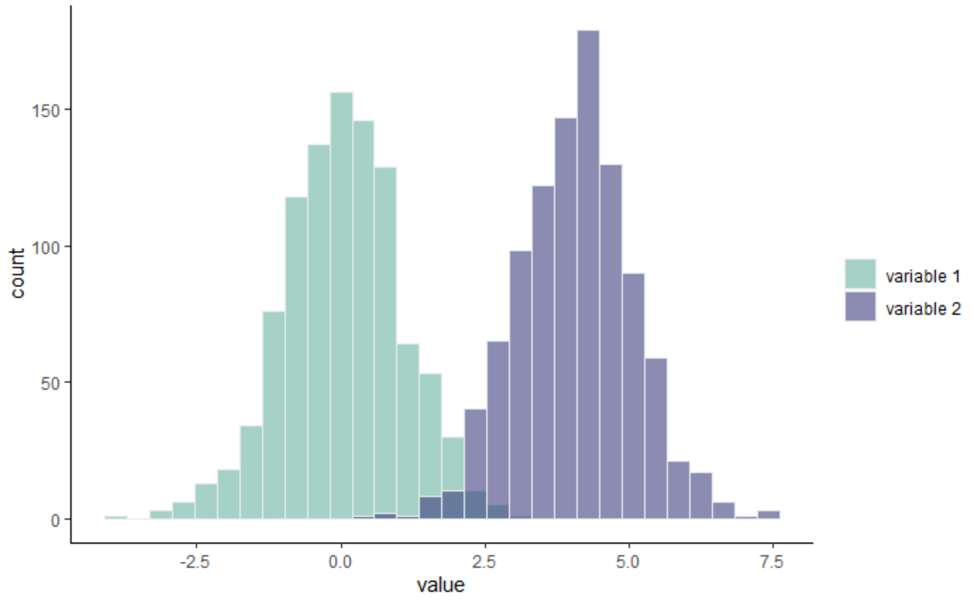




count
150
100
50
0
-2.5
0.0
2.5
5.0
7.5
value
variable 1
variable 2

Example

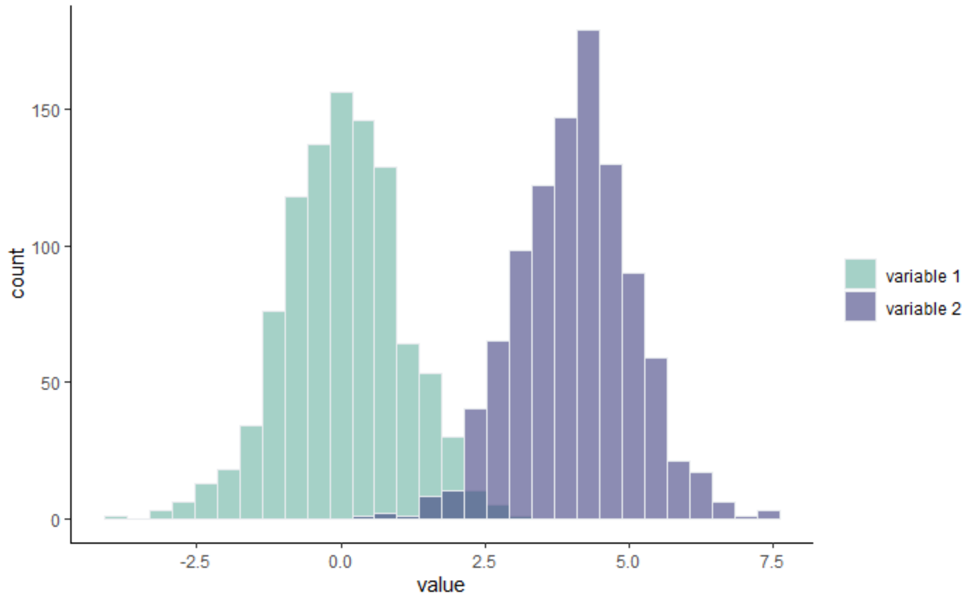
```r
data <- data.frame(
        type = c( rep("variable 1", 1000), rep("variable 2", 1000) ),
        value = c( rnorm(1000), rnorm(1000, mean=4) ) ) # Represent it

ggplot(data, aes(x=value, fill=type)) +
        geom_histogram( color="#e9ecef", alpha=0.6, position = 'identity') +
        scale_fill_manual(values=c("#69b3a2", "#404080")) + theme_classic()
        + labs(fill="")
```



1. Generate a dataframe with two variables and 1000 observations
2. Generate random values based on a normal distribution

Example

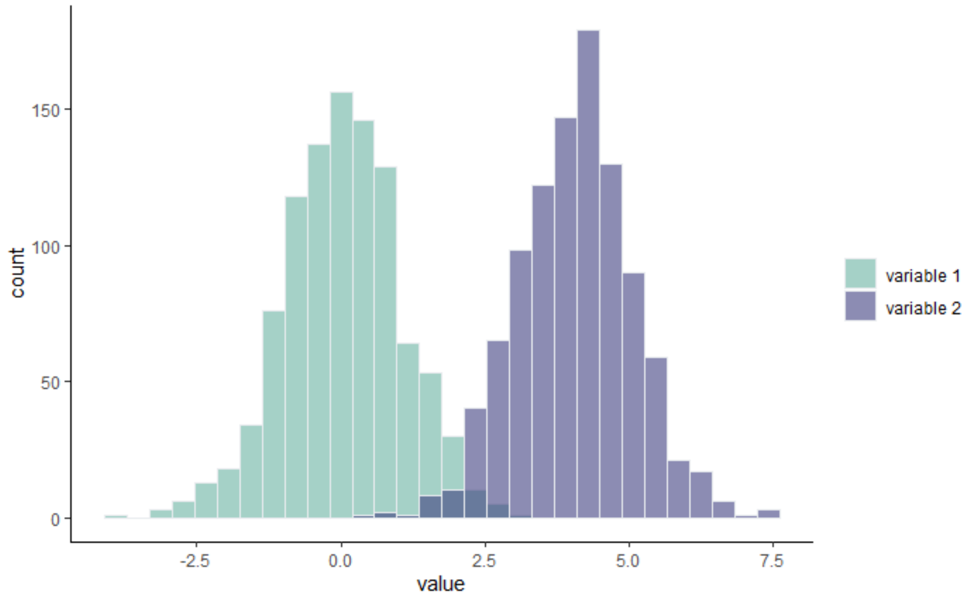
```
data <- data.frame(
        type = c( rep("variable 1", 1000), rep("variable 2", 1000) ),
        value = c( rnorm(1000), rnorm(1000, mean=4) ) ) # Represent it

ggplot(data, aes(x=value, fill=type)) +
        geom_histogram( color="#e9ecef", alpha=0.6, position = 'identity') +
        scale_fill_manual(values=c("#69b3a2", "#404080")) + theme_classic()
        + labs(fill="")
```



1. Generate a dataframe with two variables and 1000 observations
2. Generate random values based on a normal distribution
3. Select the data and variables we want to plot

Example

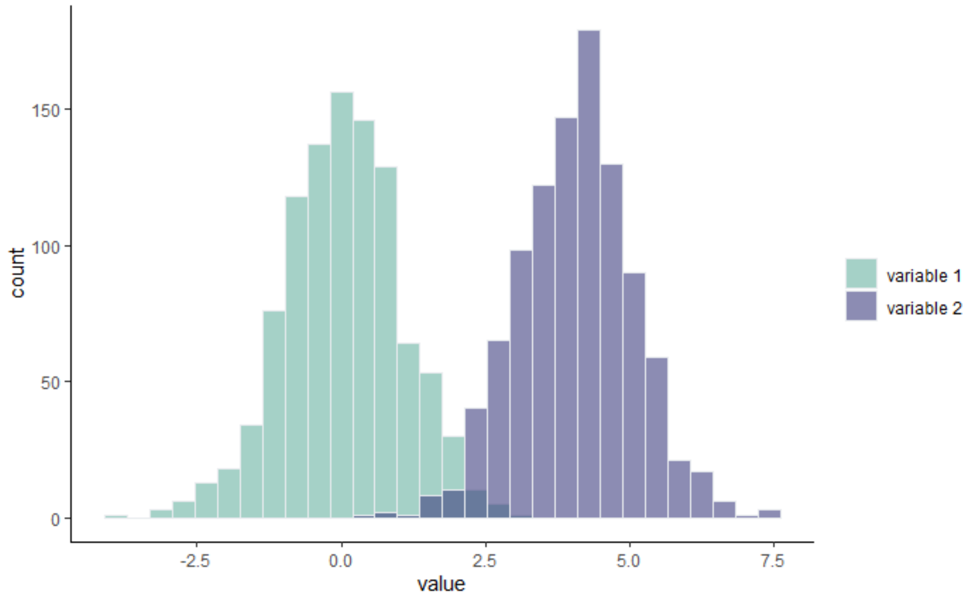
```
data <- data.frame(
        type = c( rep("variable 1", 1000), rep("variable 2", 1000) ),
        value = c( rnorm(1000), rnorm(1000, mean=4) ) ) # Represent it

ggplot(data, aes(x=value, fill=type)) +
        geom_histogram( color="#e9ecef", alpha=0.6, position = 'identity') +
        scale_fill_manual(values=c("#69b3a2", "#404080")) + theme_classic()
        + labs(fill="")
```



1. Generate a dataframe with two variables and 1000 observations
2. Generate random values based on a normal distribution
3. Select the data and variables we want to plot
4. Select the type of visualization, identify, color sets the lines, alpha for transparency and identify overlaps the bar

Example

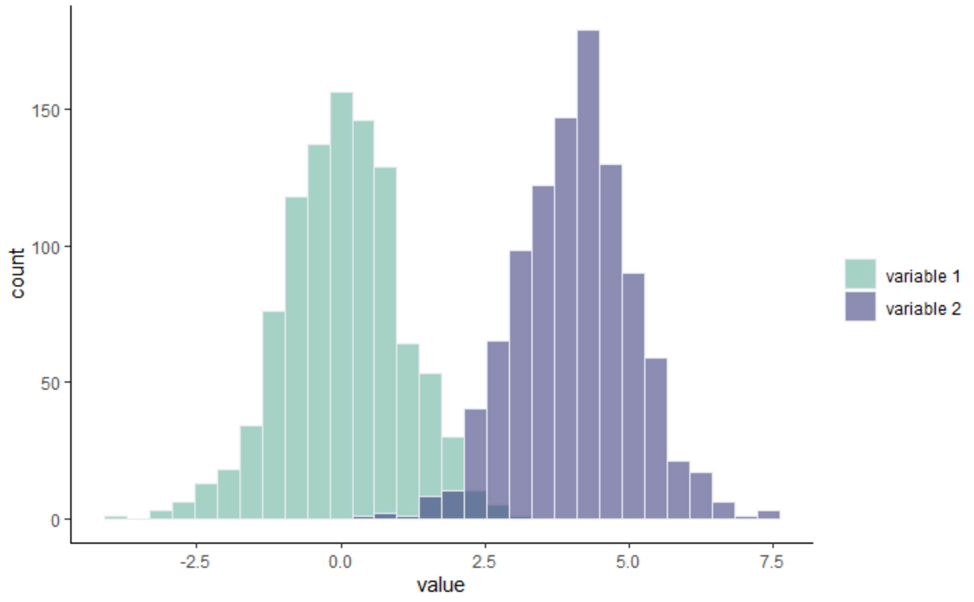
```
data <- data.frame(
        type = c( rep("variable 1", 1000), rep("variable 2", 1000) ),
        value = c( rnorm(1000), rnorm(1000, mean=4) ) ) # Represent it

ggplot(data, aes(x=value, fill=type)) +
        geom_histogram( color="#e9ecef", alpha=0.6, position = 'identity') +
        scale_fill_manual(values=c("#69b3a2", "#404080")) + theme_classic()
        + labs(fill="")
```



1. Generate a dataframe with two variables and 1000 observations
2. Generate random values based on a normal distribution
3. Select the data and variables we want to plot
4. Select the type of visualization, identify, color sets the lines, alpha for transparency and identify overlaps the bar
5. We fill each histogram with a different colour

Example

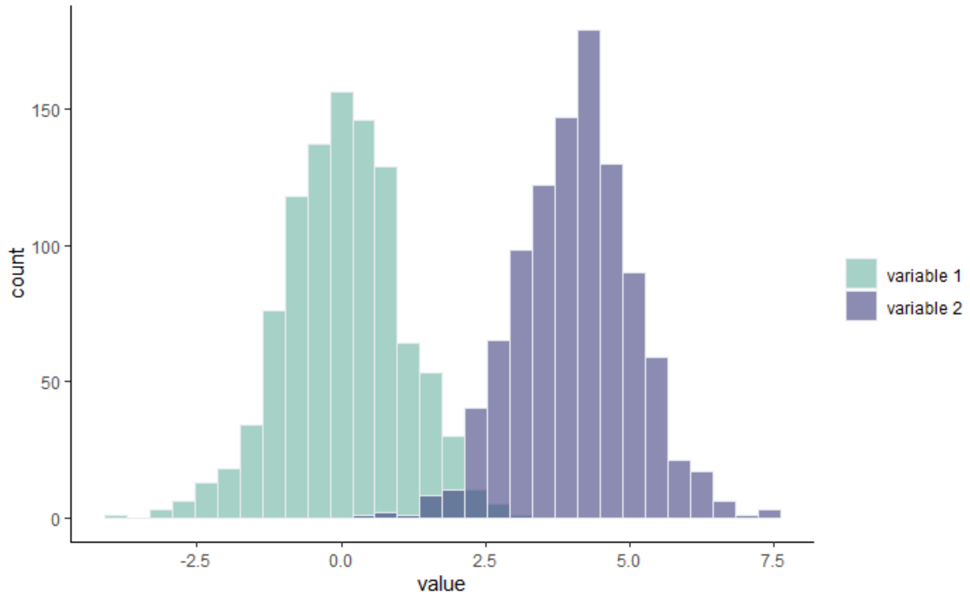
```r
data <- data.frame(
        type = c( rep("variable 1", 1000), rep("variable 2", 1000) ),
        value = c( rnorm(1000), rnorm(1000, mean=4) ) ) # Represent it

ggplot(data, aes(x=value, fill=type)) +
        geom_histogram( color="#e9ecef", alpha=0.6, position = 'identity') +
        scale_fill_manual(values=c("#69b3a2", "#404080")) + theme_classic()
        + labs(fill="")
```



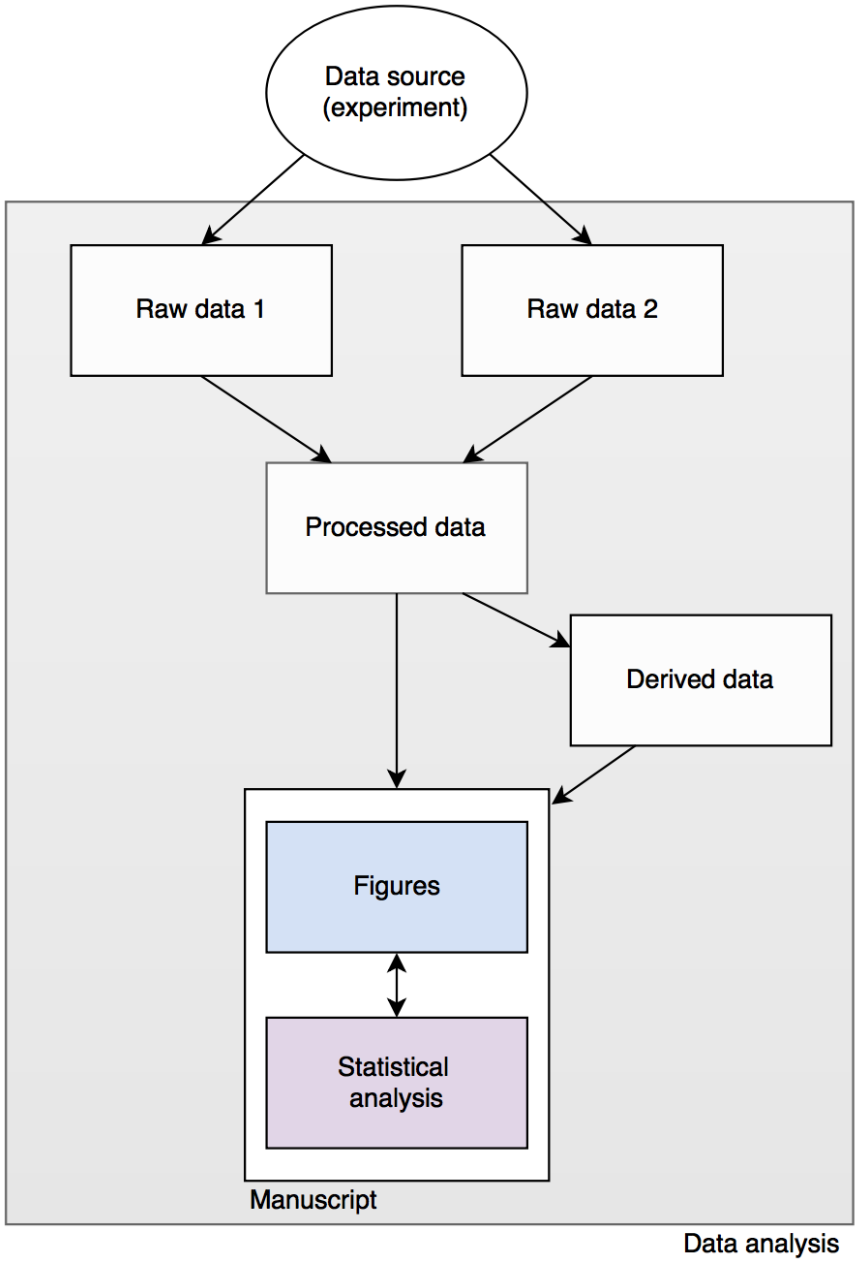
1. Generate a dataframe with two variables and 1000 observations
2. Generate random values based on a normal distribution
3. Select the data and variables we want to plot
4. Select the type of visualization, identify, color sets the lines, alpha for transparency and identify overlaps the bar
5. We fill each histogram with a different colour
6. Choose a preset theme

Example

```r
data <- data.frame(
        type = c( rep("variable 1", 1000), rep("variable 2", 1000) ),
        value = c( rnorm(1000), rnorm(1000, mean=4) ) ) # Represent it

ggplot(data, aes(x=value, fill=type)) +
        geom_histogram( color="#e9ecef", alpha=0.6, position = 'identity') +
        scale_fill_manual(values=c("#69b3a2", "#404080")) + theme_classic()
        + labs(fill="")
```



1. Generate a dataframe with two variables and 1000 observations
2. Generate random values based on a normal distribution
3. Select the data and variables we want to plot
4. Select the type of visualization, identify, color sets the lines, alpha for transparency and identify overlaps the bar
5. We fill each histogram with a different colour
6. Choose a preset theme
7. Names for labs added

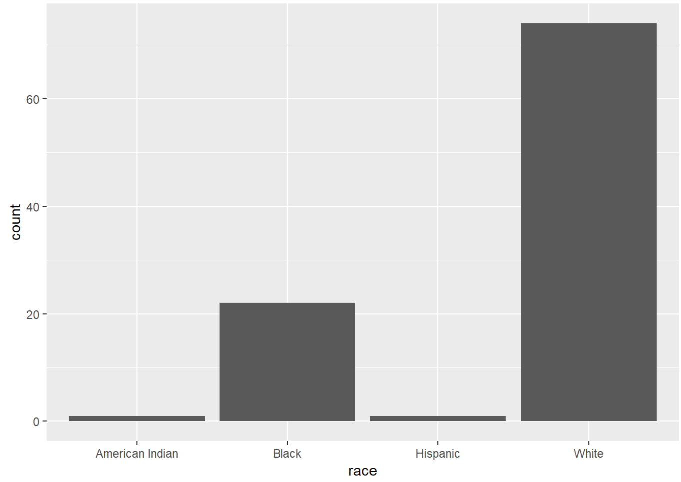

Data source
(experiment)
Raw data 1
Raw data 2
Processed data
Derived data
Figures
Statistical
analysis
Manuscript
Data analysis

Univariate Graphs

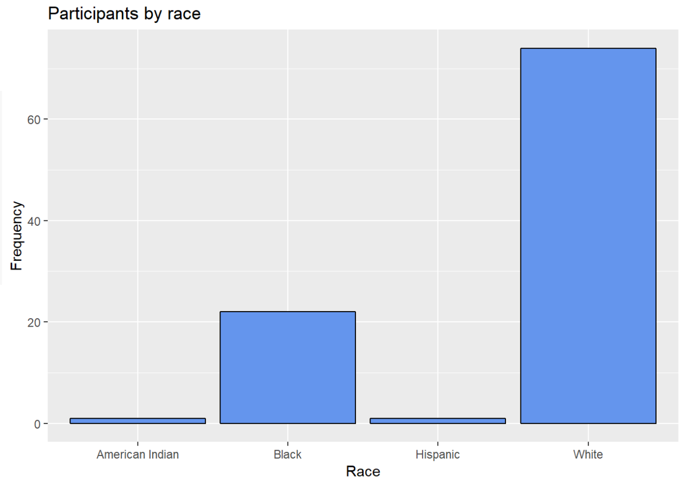
Univariate graphs plot the distribution of data from a single variable. The variable can be categorical (e.g., race, sex) or quantitative (e.g., age, weight).

- **Categorical**
- Quantitative

```r
library(ggplot2)

data(Marriage, package = "mosaicData") # plot the distribution of race
        ggplot(Marriage, aes(x = race)) +
        geom_bar()
```




```r
# plot the distribution of race with modified colors and labels
ggplot(Marriage, aes(x = race)) +
        geom_bar(fill = "cornflowerblue", color="black") +
        labs(x = "Race", y = "Frequency", title = "Participants by race")
```

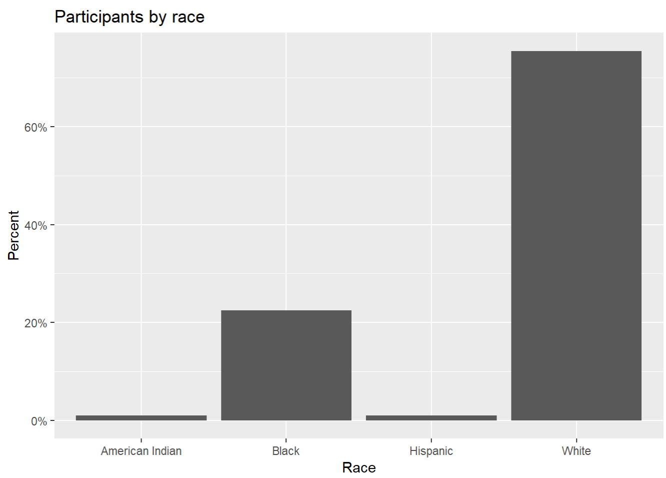


Univariate Graphs

Univariate graphs plot the distribution of data from a single variable. The variable can be categorical (e.g., race, sex) or quantitative (e.g., age, weight).

Percents

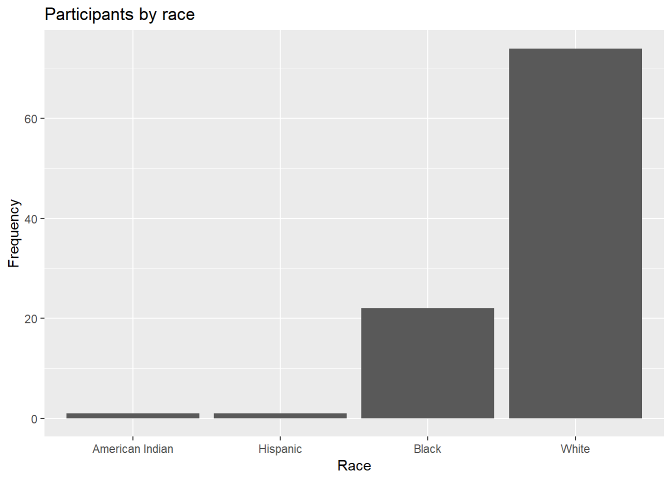
```
# plot the distribution as percentages
ggplot(Marriage, aes(x = race, y = ..count.. / sum(..count..))) +
        geom_bar() +
        labs(x = "Race", y = "Percent", title = "Participants by race") +
        scale_y_continuous(labels = scales::percent)
```



Sorting categories

```
# calculate number of participants in # each race category
library(dplyr)
plotdata <- Marriage %>% count(race)
```

```
# plot the bars in ascending order
ggplot(plotdata, aes(x = reorder(race, n), y = n)) +
        geom_bar(stat = "identity") +
        labs(x = "Race", y = "Frequency", title = "Participants by race")
```

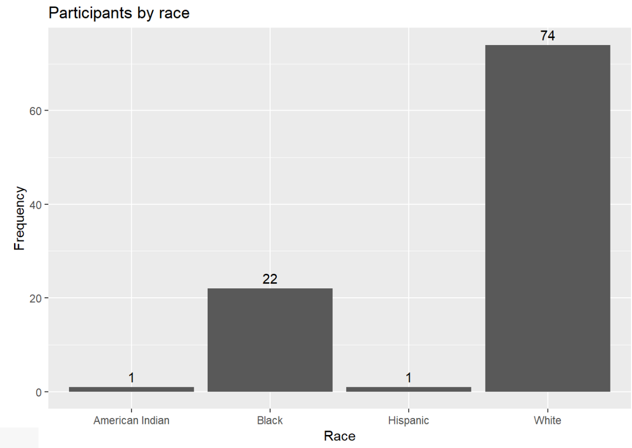


Univariate Graphs

Univariate graphs plot the distribution of data from a single variable. The variable can be categorical (e.g., race, sex) or quantitative (e.g., age, weight).

Labeling

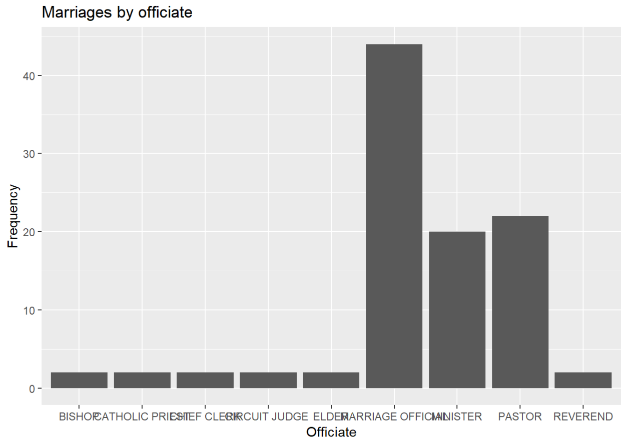
```r
# plot the bars with numeric labels
ggplot(plotdata, aes(x = race, y = n)) +
         geom_bar(stat = "identity") +
         geom_text(aes(label = n), vjust=-0.5) +
         labs(x = "Race", y = "Frequency", title = "Participants by race")
```



Overlapping labels

```r
# basic bar chart with overlapping labels
ggplot(Marriage, aes(x = officialTitle)) +
         geom_bar() +
         labs(x = "Officiate", y = "Frequency", title = "Marriages by officiate")
```

```r
# horizontal bar chart
ggplot(Marriage, aes(x = officialTitle)) +
         geom_bar() +
         labs(x = "", y = "Frequency", title = "Marriages by officiate") +
         coord_flip()
```

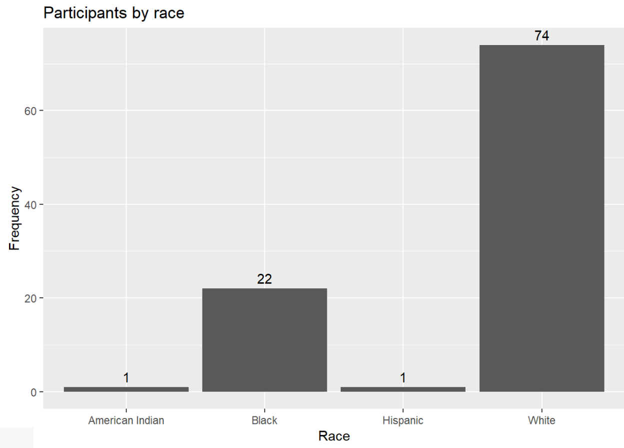


Univariate Graphs

Univariate graphs plot the distribution of data from a single variable. The variable can be categorical (e.g., race, sex) or quantitative (e.g., age, weight).

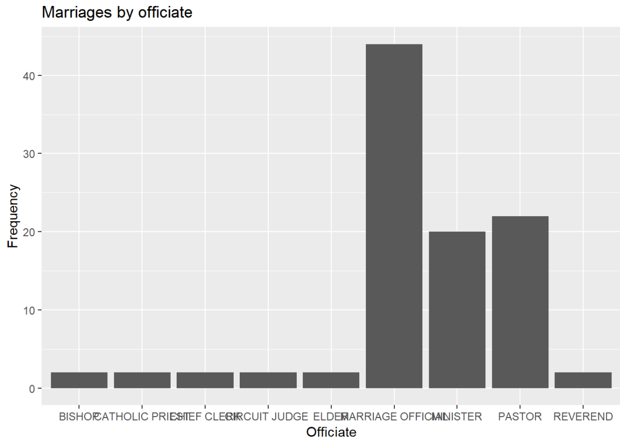
Labeling

```
# plot the bars with numeric labels
ggplot(plotdata, aes(x = race, y = n)) +
        geom_bar(stat = "identity") +
        geom_text(aes(label = n), vjust=-0.5) +
        labs(x = "Race", y = "Frequency", title = "Participants by race")
```



Overlapping labels

```
# basic bar chart with overlapping labels
ggplot(Marriage, aes(x = officialTitle)) +
        geom_bar() +
        labs(x = "Officiate", y = "Frequency", title = "Marriages by officiate")
```



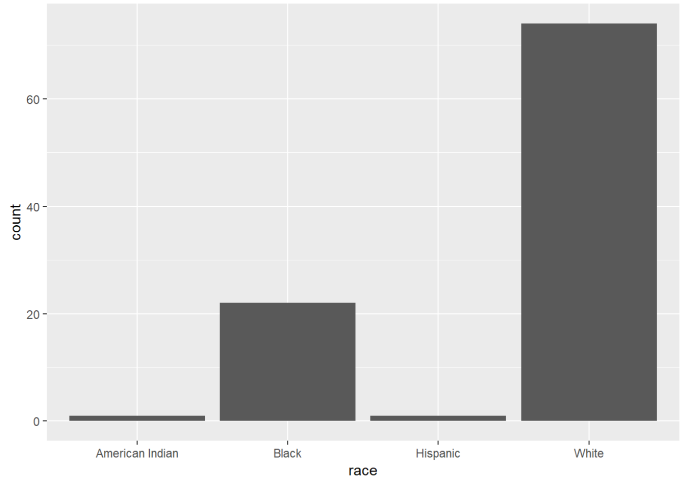

```
# horizontal bar chart
ggplot(Marriage, aes(x = officialTitle)) +
        geom_bar() +
        labs(x = "", y = "Frequency", title = "Marriages by officiate") +
        coord_flip()
```

Univariate Graphs

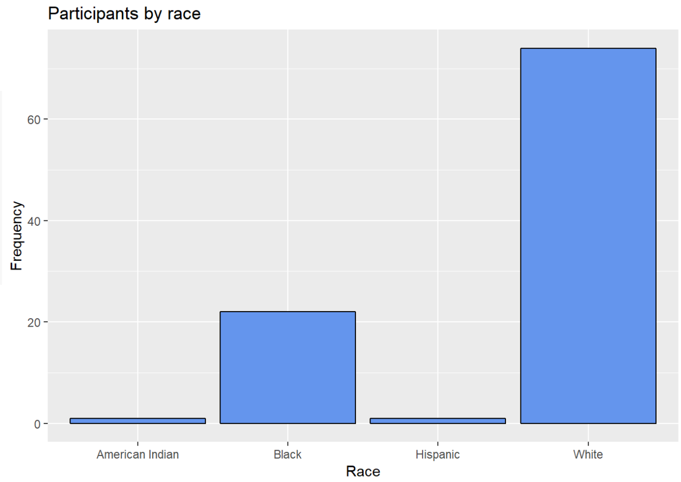
Univariate graphs plot the distribution of data from a single variable. The variable can be categorical (e.g., race, sex) or quantitative (e.g., age, weight).

- Categorical
- Quantitative

```
library(ggplot2)

data(Marriage, package = "mosaicData") # plot the distribution of race
        ggplot(Marriage, aes(x = race)) +
        geom_bar()
```




```
# plot the distribution of race with modified colors and labels
ggplot(Marriage, aes(x = race)) +
        geom_bar(fill = "cornflowerblue", color="black") +
        labs(x = "Race", y = "Frequency", title = "Participants by race")
```

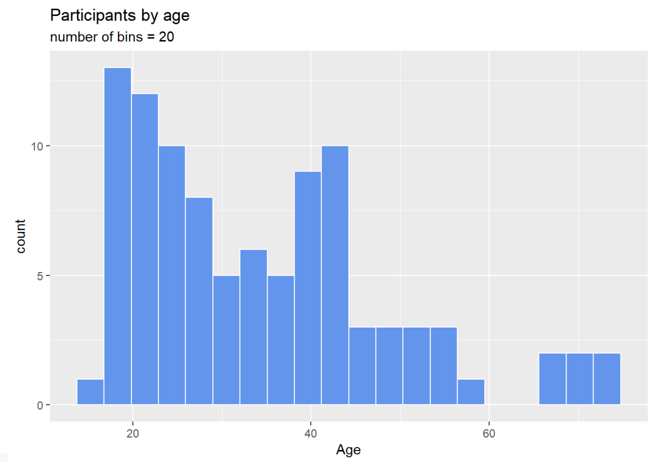


Univariate Graphs

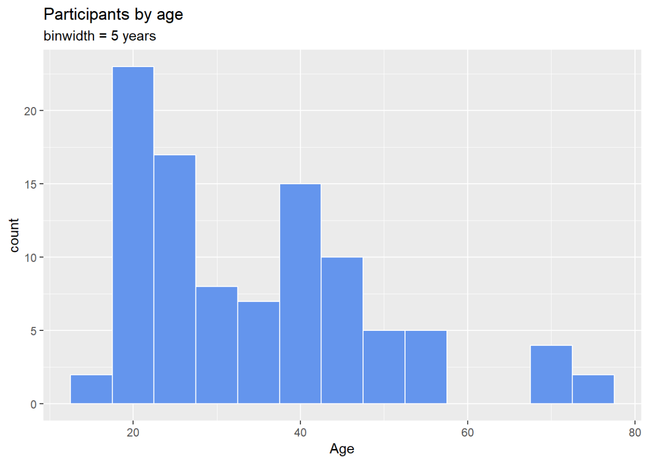
Univariate graphs plot the distribution of data from a single variable. The variable can be categorical (e.g., race, sex) or quantitative (e.g., age, weight).

- Categorical
- **Quantitative**

```
library(ggplot2) # plot the age distribution using a histogram
        ggplot(Marriage, aes(x = age)) +
        geom_histogram() +
        labs(title = "Participants by age", x = "Age")
```




```
# plot the histogram with a binwidth of 5
ggplot(Marriage, aes(x = age)) +
        geom_histogram(fill = "cornflowerblue", color = "white", binwidth = 5) +
        labs(title="Participants by age", subtitle = "binwidth = 5 years", x = "Age")
```



Univariate Graphs

Univariate graphs plot the distribution of data from a single variable. The variable can be categorical (e.g., race, sex) or quantitative (e.g., age, weight).

- Categorical
- **Quantitative**

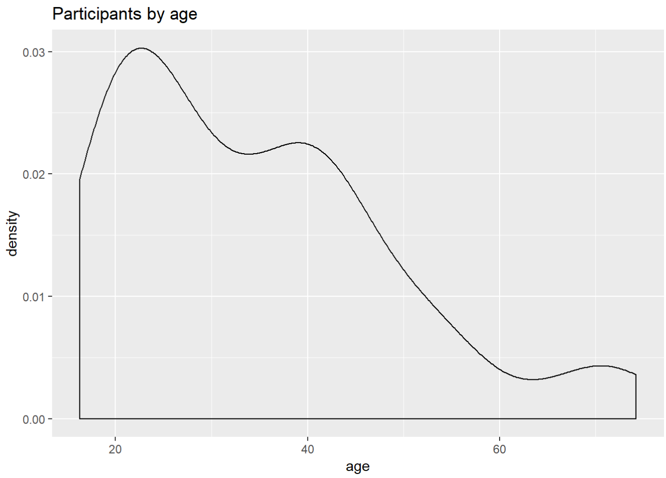
Computes and draws kernel density estimate, which is a smoothed version of the histogram. This is a useful alternative to the histogram for continuous data that comes from an underlying smooth distribution.

(What??) Basically, we are trying to draw a smoothed histogram, where the area under the curve equals one.

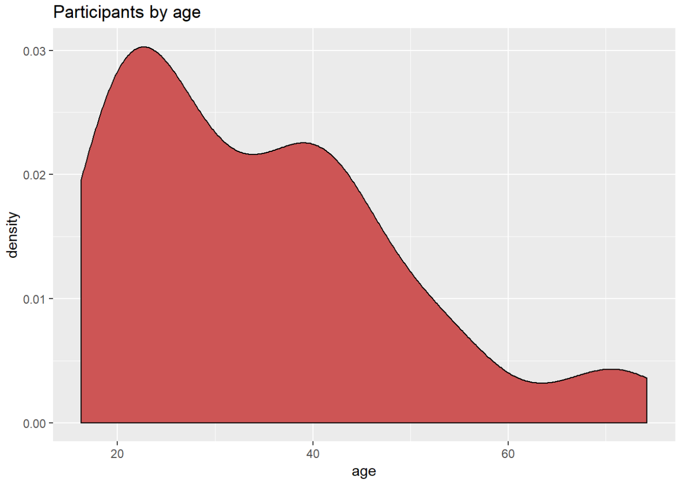
Univariate Graphs

Smoothing parameter

```
# Create a kernel density plot of age
ggplot(Marriage, aes(x = age)) +
        geom_density() +
        labs(title = "Participants by age")
```




```
# Create a kernel density plot of age
ggplot(Marriage, aes(x = age)) +
        geom_density(fill = "indianred3") +
        labs(title = "Participants by age")
```

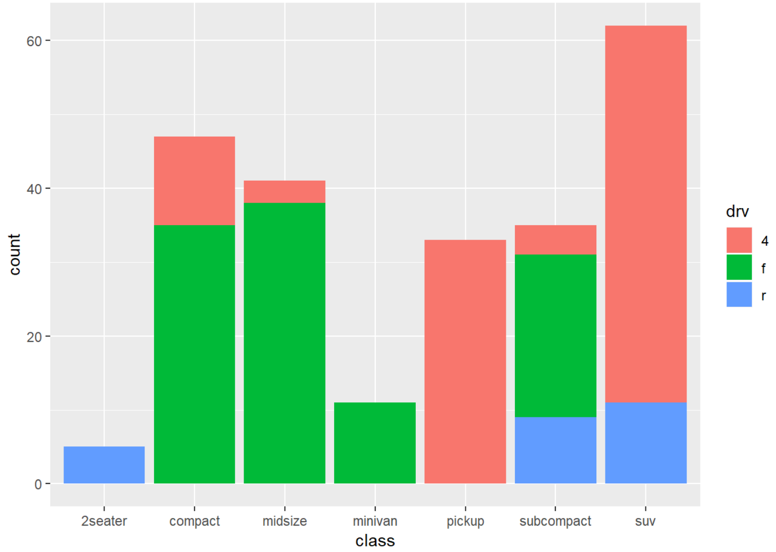


Bivariate Graphs

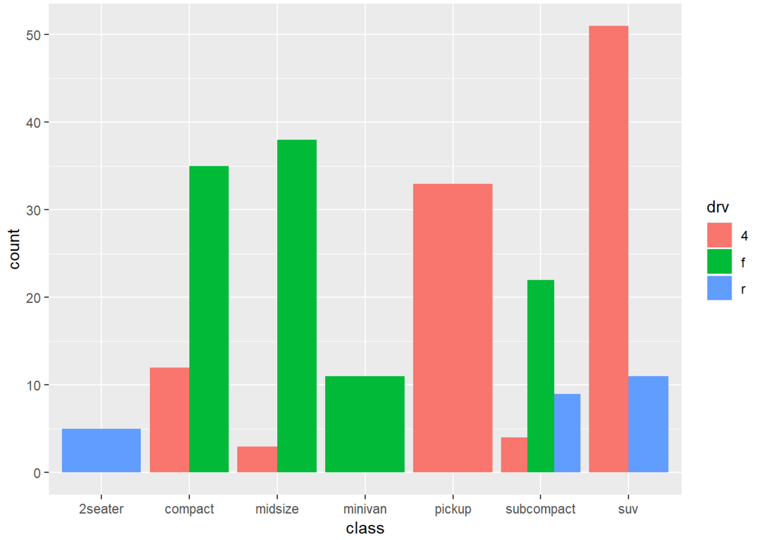
Categorical vs. Categorical

Stacked bar chart

```r
library(ggplot2)

# stacked bar chart

ggplot(mpg, aes(x = class, fill = drv)) +
        geom_bar(position = "stack")
```




```r
# grouped bar plot

ggplot(mpg, aes(x = class, fill = drv)) +
        geom_bar(position = "dodge")
```

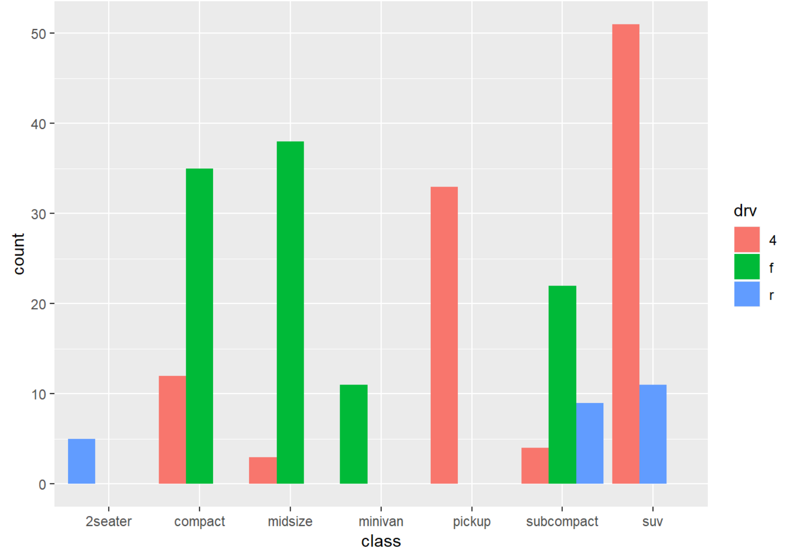


Bivariate Graphs

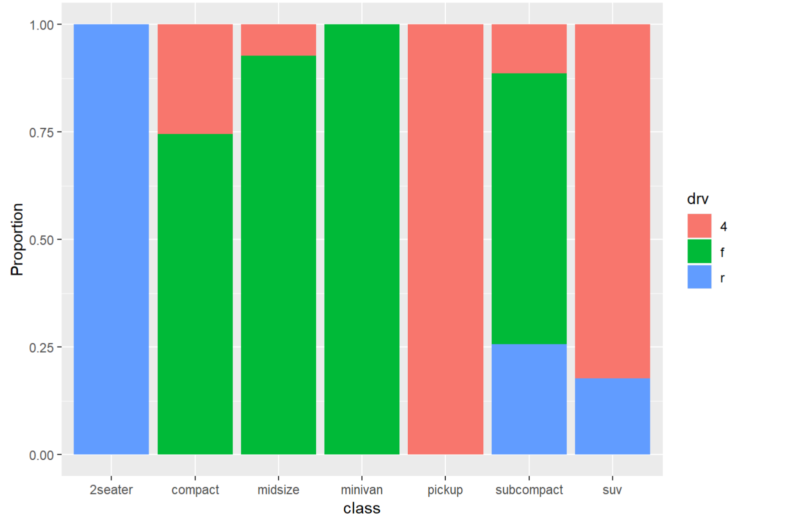
Categorical vs. Categorical

Stacked bar chart

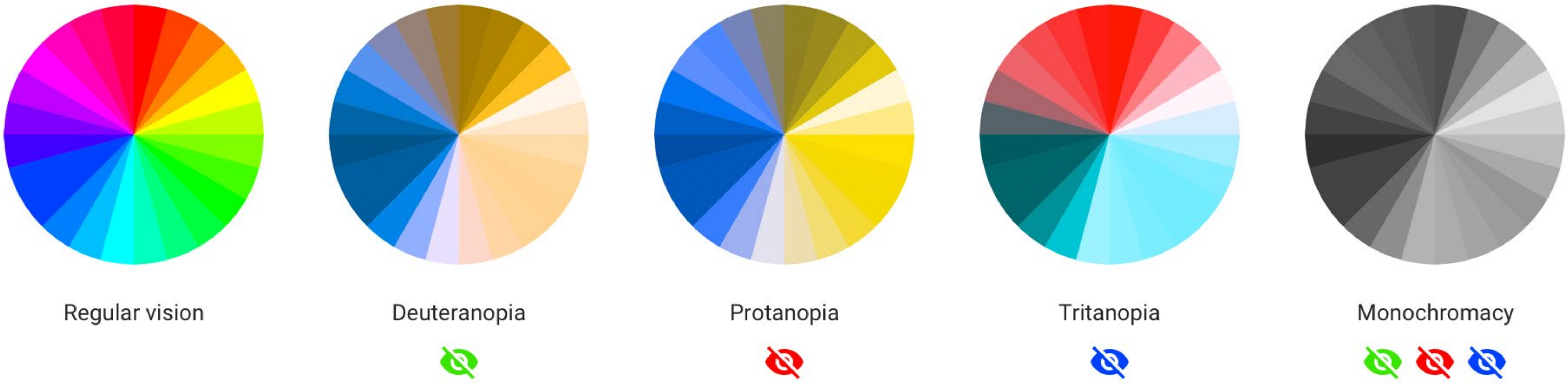
```
# grouped bar plot preserving zero count bars

ggplot(mpg, aes(x = class, fill = drv)) +
        geom_bar(position = position_dodge(preserve = "single"))
```




```
# bar plot, with each bar representing 100%

ggplot(mpg, aes(x = class, fill = drv)) +
        geom_bar(position = "fill") +
        labs(y = "Proportion")
```



Effective communication through visualization



Bivariate Graphs

Categorical vs. Categorical

Improving the color and labeling

You can use additional options to improve color and labeling.

- `factor` modifies the order of the categories for the class variable and both the order and the labels for the drive variable

- `scale_y_continuous` modifies the y-axis tick mark labels

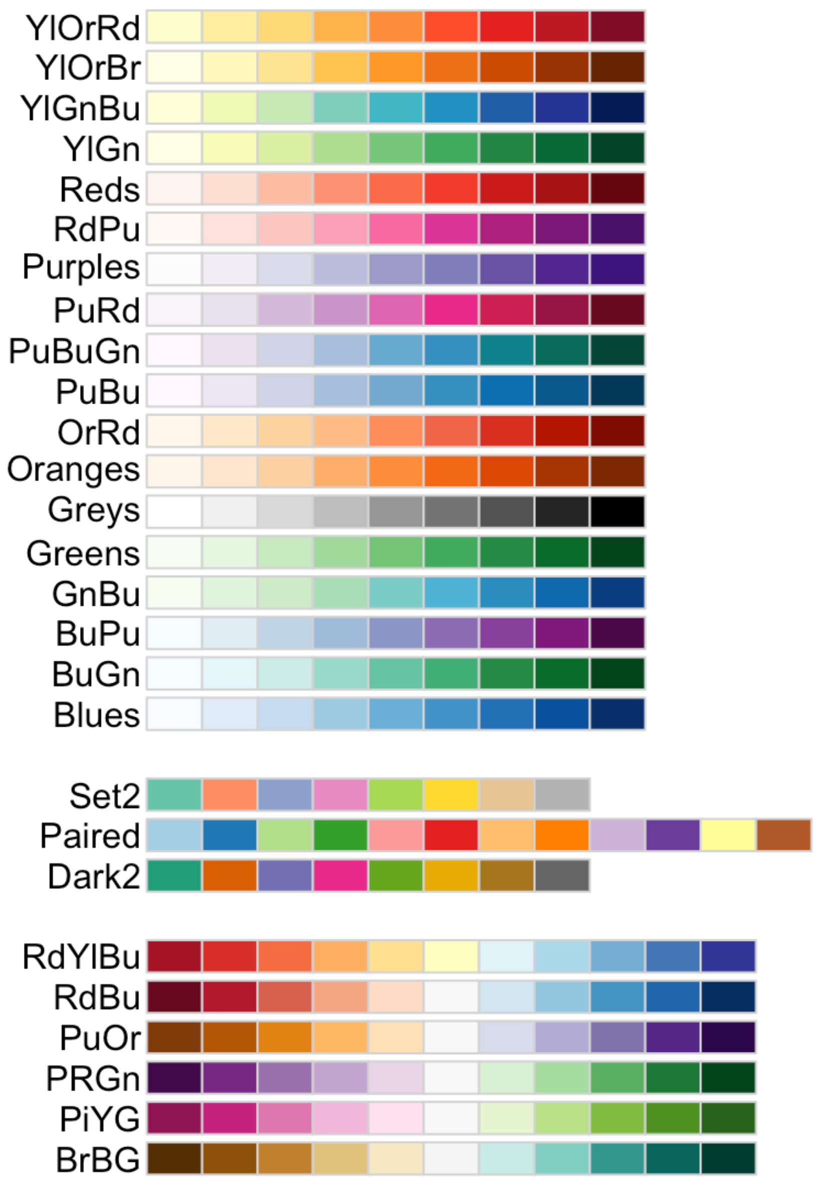
- `labs` provides a title and changed the labels for the x and y axes and the legend
- `scale_fill_brewer` changes the fill color scheme

- `theme_minimal` removes the grey background and changed the grid color

Bivariate Graphs

Categorical vs. Categorical

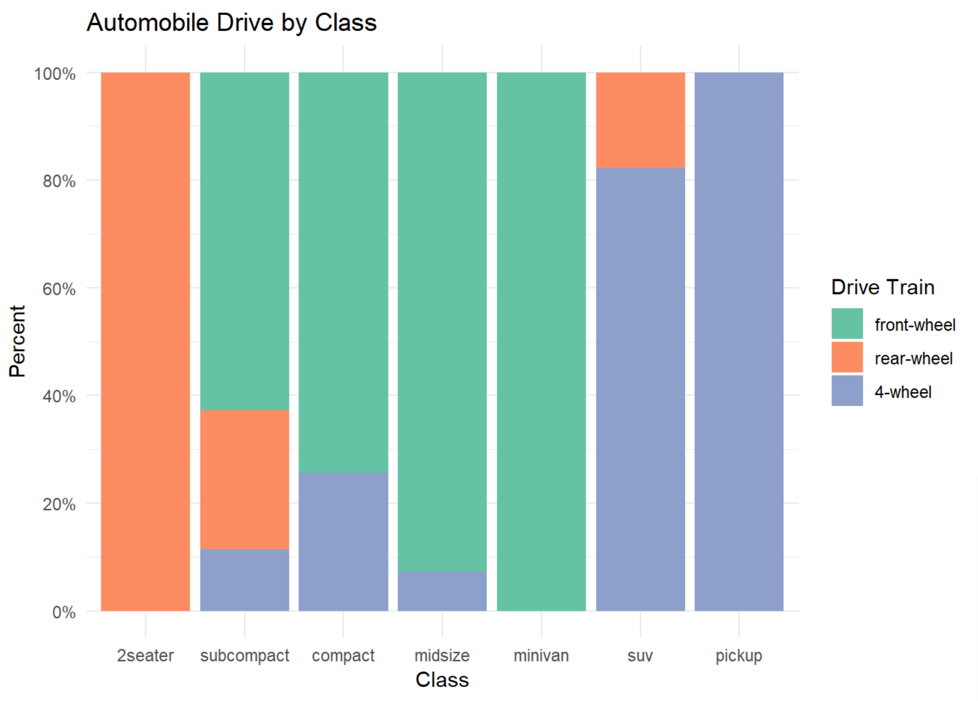
Improving the color and labeling



Bivariate Graphs

Categorical vs. Categorical

Improving the color and labeling



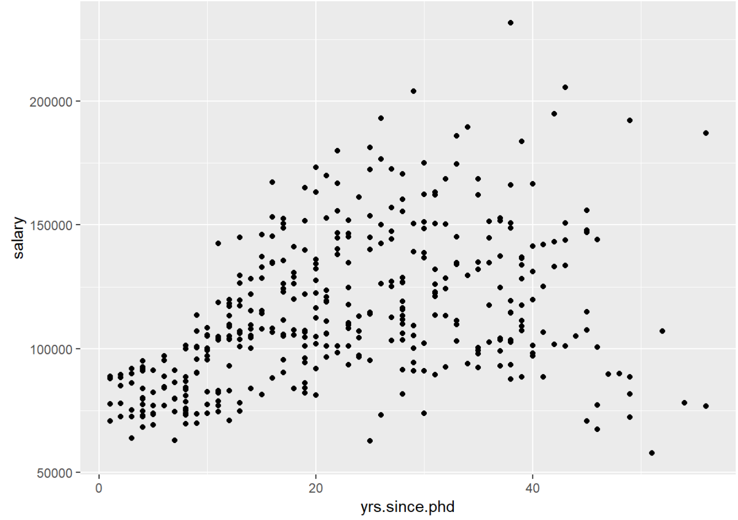

```r
# bar plot, with each bar representing 100%,
# reordered bars, and better labels and colors

library(scales)

ggplot(mpg,

aes(x = factor(class, levels = c("2seater", "subcompact", "compact", "midsize", "minivan", "suv", "pickup")),
fill = factor(drv, levels = c("f", "r", "4"), labels = c("front-wheel", "rear-wheel", "4-wheel")))) +
geom_bar(position = "fill") +
scale_y_continuous(breaks = seq(0, 1, .2), label = percent) +
scale_fill_brewer(palette = "Set2") +
labs(y = "Percent", fill = "Drive Train", x = "Class", title = "Automobile Drive by Class") +
theme_minimal()
```

Bivariate Graphs

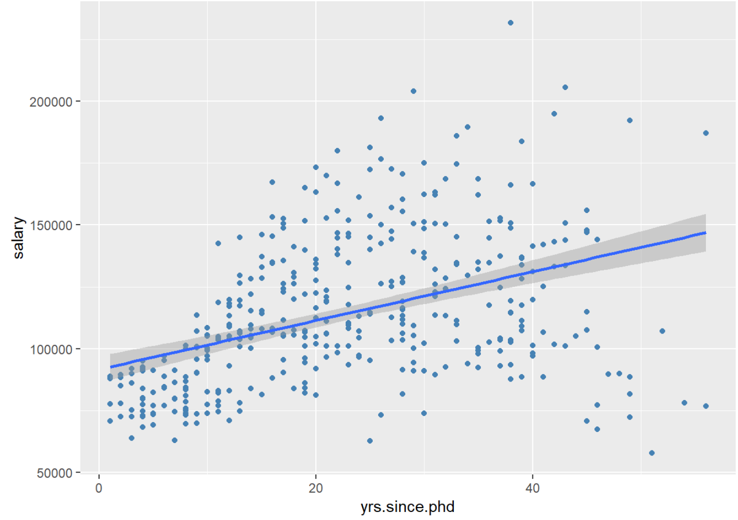
Quantitative vs. Quantitative

Scatterplot

```
data(Salaries, package="carData")

# simple scatterplot
ggplot(Salaries, aes(x = yrs.since.phd, y = salary)) +
geom_point()
```




```
# scatterplot with linear fit line

ggplot(Salaries, aes(x = yrs.since.phd, y = salary)) +
geom_point(color= "steelblue") +
geom_smooth(method = "lm")
```

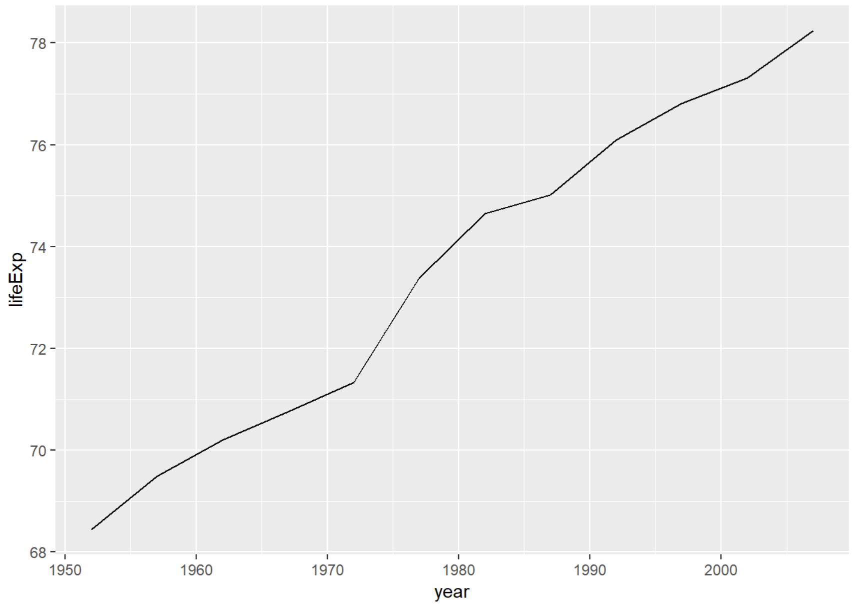


Bivariate Graphs

Quantitative vs. Quantitative

Line plot

```
data(gapminder, package="gapminder") # Select US cases

library(dplyr)

plotdata <- filter(gapminder, country == "United States")

# simple line plot

ggplot(plotdata, aes(x = year, y = lifeExp)) +
geom_line()
```

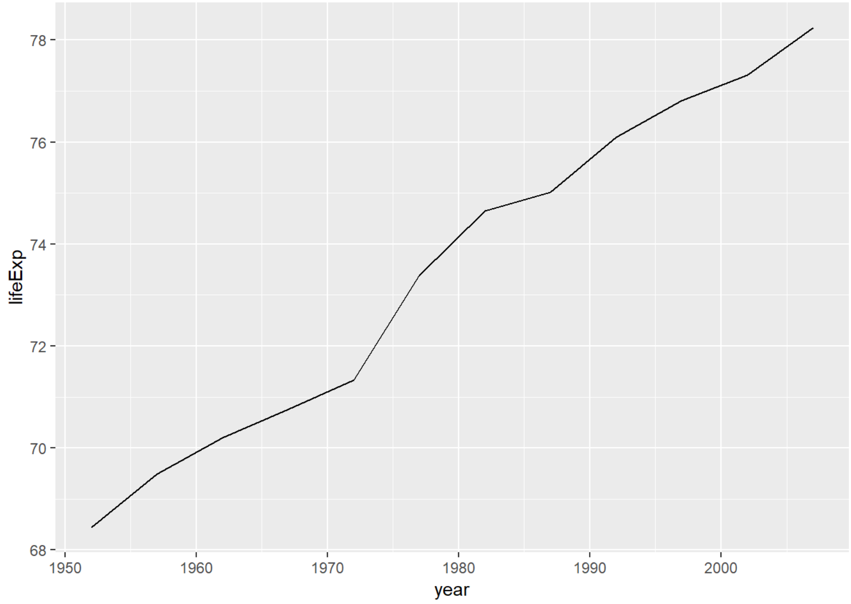


Bivariate Graphs

Categorical vs. Quantitative

Bar chart (on summary statistics)

```
data(gapminder, package="gapminder") # Select US cases

library(dplyr)

plotdata <- filter(gapminder, country == "United States")

# simple line plot

ggplot(plotdata, aes(x = year, y = lifeExp)) +
geom_line()
```

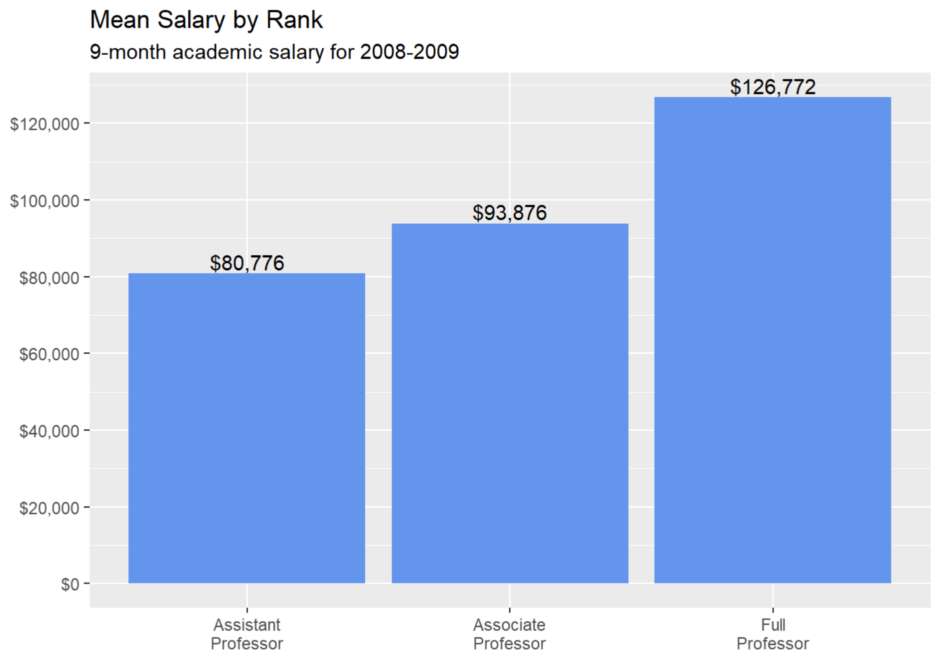


Bivariate Graphs

Categorical vs. Quantitative

Bar chart (on summary statistics)

```
# plot mean salaries
library(scales)

ggplot(plotdata, aes(x = factor(rank, labels =
c("Assistant\nProfessor", "Associate\nProfessor",
"Full\nProfessor")), y = mean_salary)) +
geom_bar(stat = "identity", fill = "cornflowerblue") +
geom_text(aes(label = dollar(mean_salary)), vjust = -0.25) +
scale_y_continuous(breaks = seq(0, 130000, 20000), label =
dollar) +
labs(title = "Mean Salary by Rank", subtitle = "9-month
academic salary for 2008-2009", x = "", y = "")
```

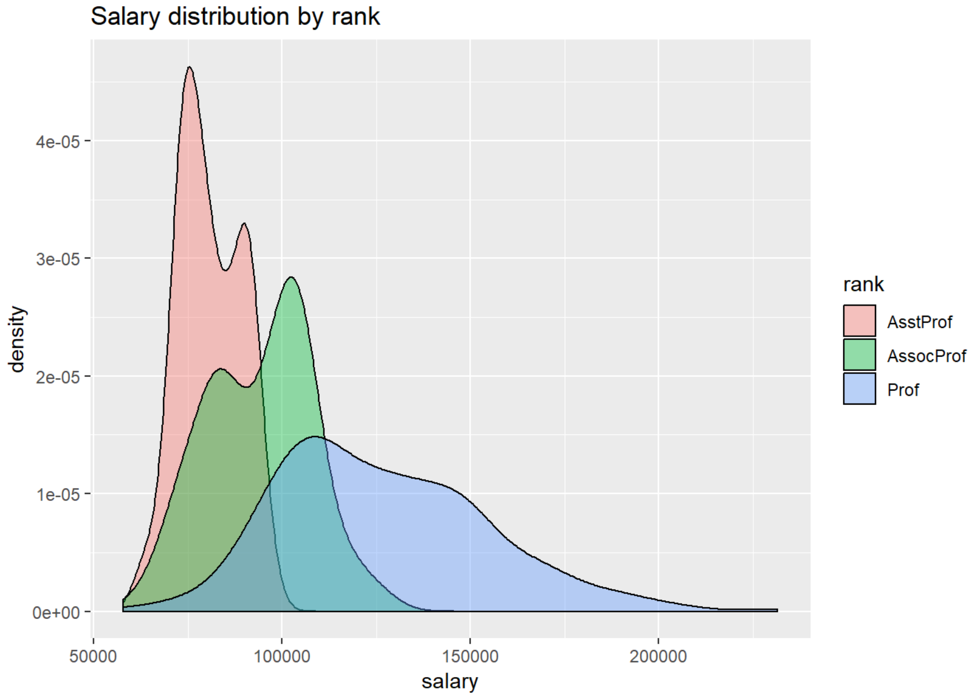


Bivariate Graphs

Categorical vs. Quantitative

Grouped kernel density plots

```
# plot the distribution of salaries # by rank using kernel
density plots

ggplot(Salaries, aes(x = salary, fill = rank)) +
geom_density(alpha = 0.4) +
labs(title = "Salary distribution by rank")
```

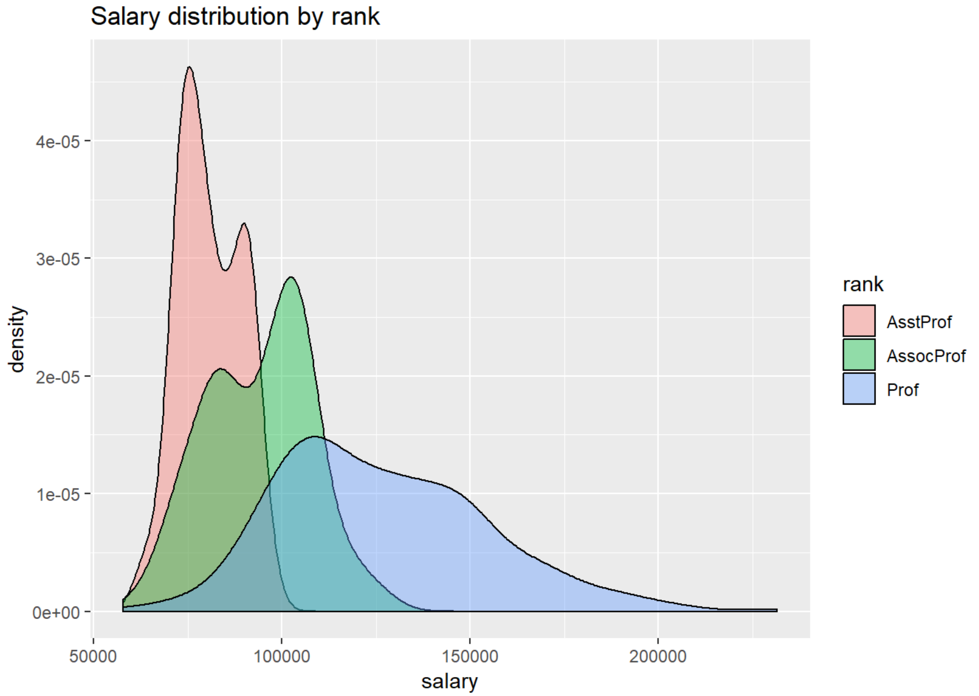


Bivariate Graphs

Categorical vs. Quantitative

Grouped kernel density plots

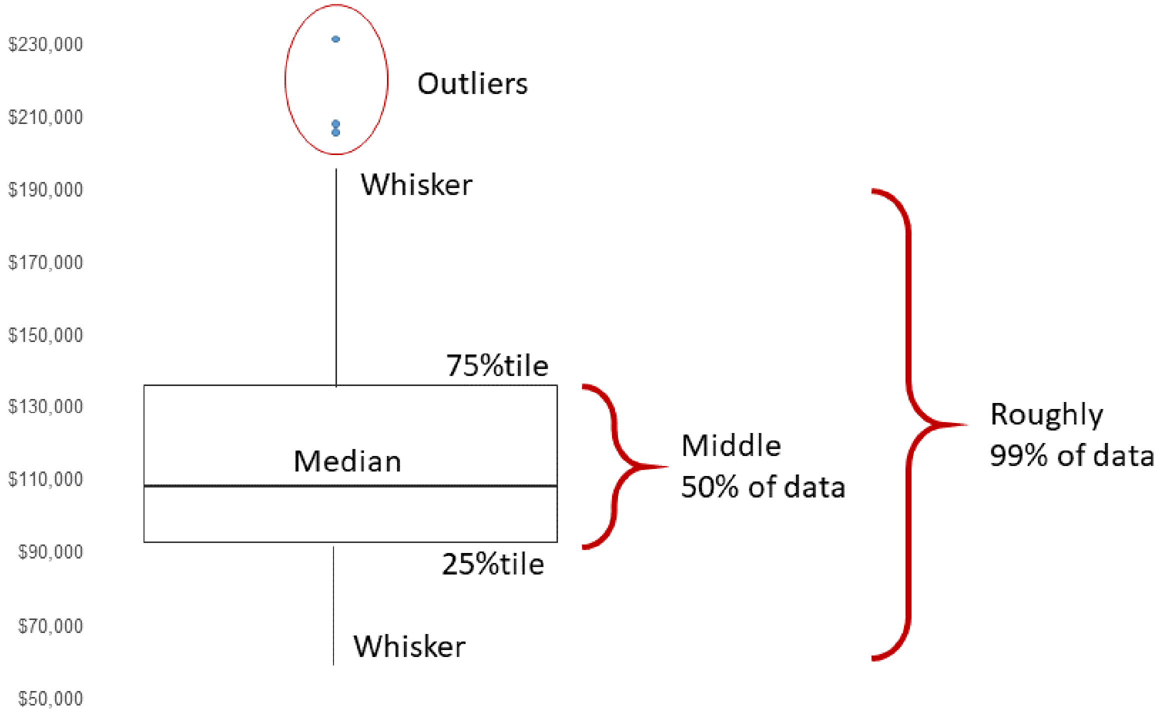
```
# plot the distribution of salaries # by rank using kernel
density plots

ggplot(Salaries, aes(x = salary, fill = rank)) +
geom_density(alpha = 0.4) +
labs(title = "Salary distribution by rank")
```



Bivariate Graphs

Categorical vs. Quantitative

Boxplots

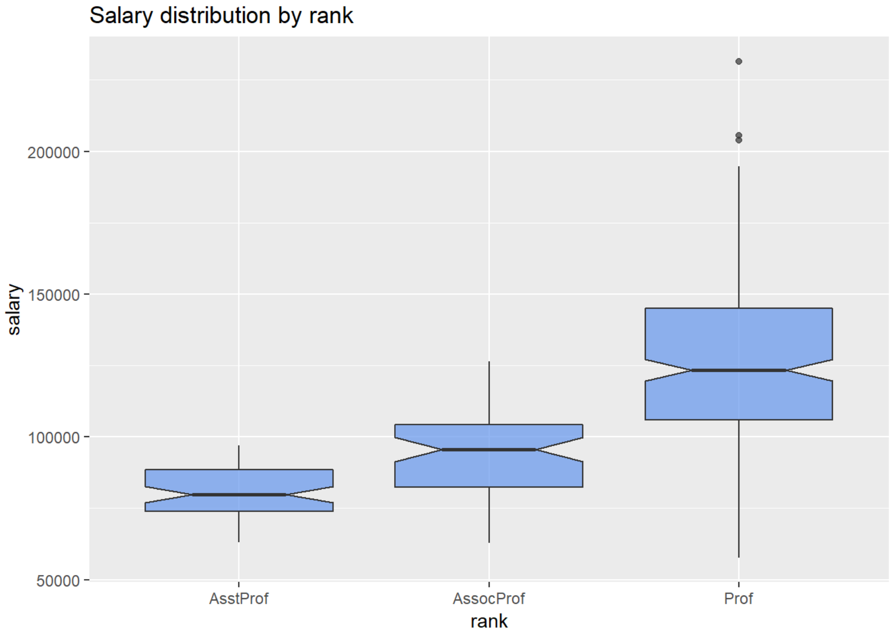


Bivariate Graphs

Categorical vs. Quantitative

Boxplots

```
# plot the distribution of salaries by rank using boxplots

ggplot(Salaries, aes(x = rank, y = salary)) +
geom_boxplot(notch = TRUE, fill = "cornflowerblue", alpha = .7) +
labs(title = "Salary distribution by rank")
```



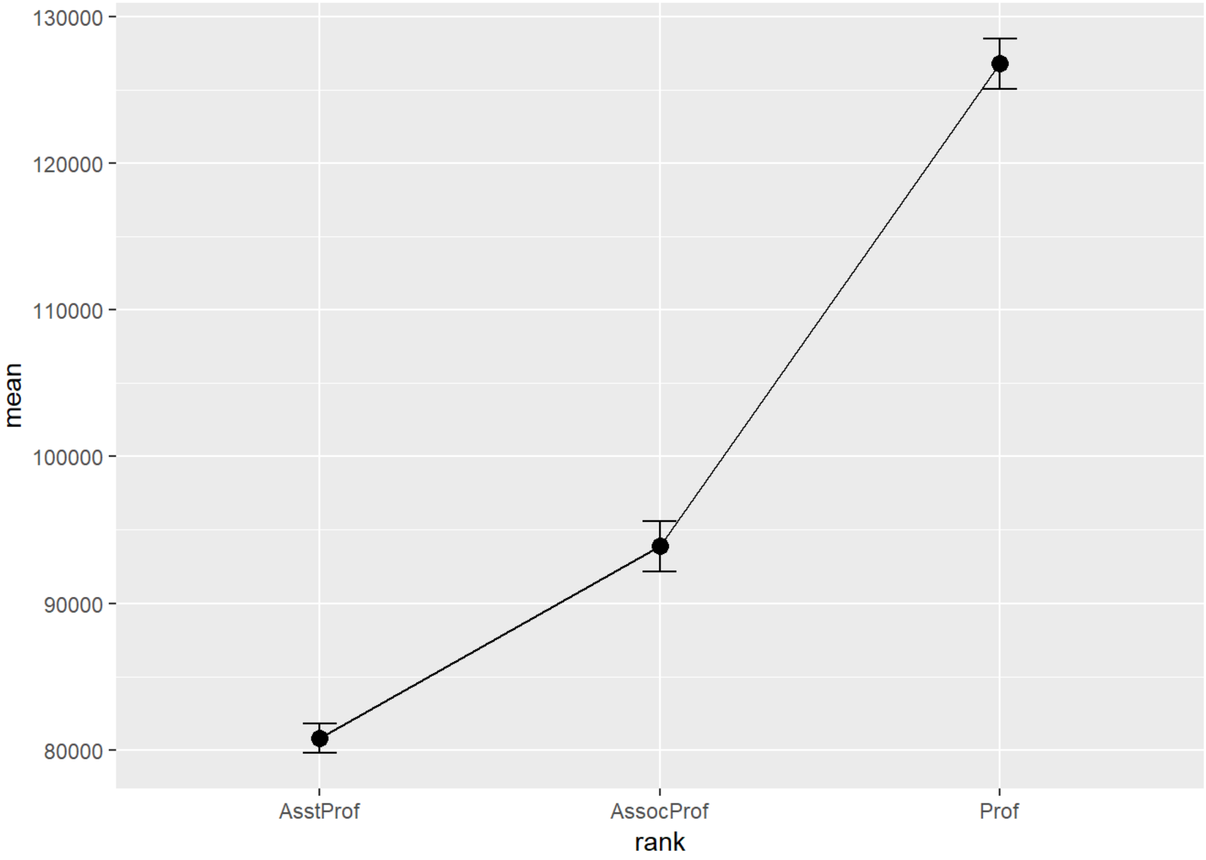
Bivariate Graphs

Categorical vs. Quantitative

Mean/SEM plots

```
# calculate means, standard deviations, # standard
errors, and 95% confidence # intervals by rank

library(dplyr)
plotdata <- Salaries %>% group_by(rank) %>%
summarize(n = n(), mean = mean(salary), sd =
sd(salary), se = sd / sqrt(n), ci = qt(0.975, df =
n - 1) * sd / sqrt(n))
```

```
# plot the means and standard errors
ggplot(plotdata, aes(x = rank, y = mean, group =
1)) + geom_point(size = 3) + geom_line() +
geom_errorbar(aes(ymin = mean - se, ymax = mean +
se), width = .1)
```

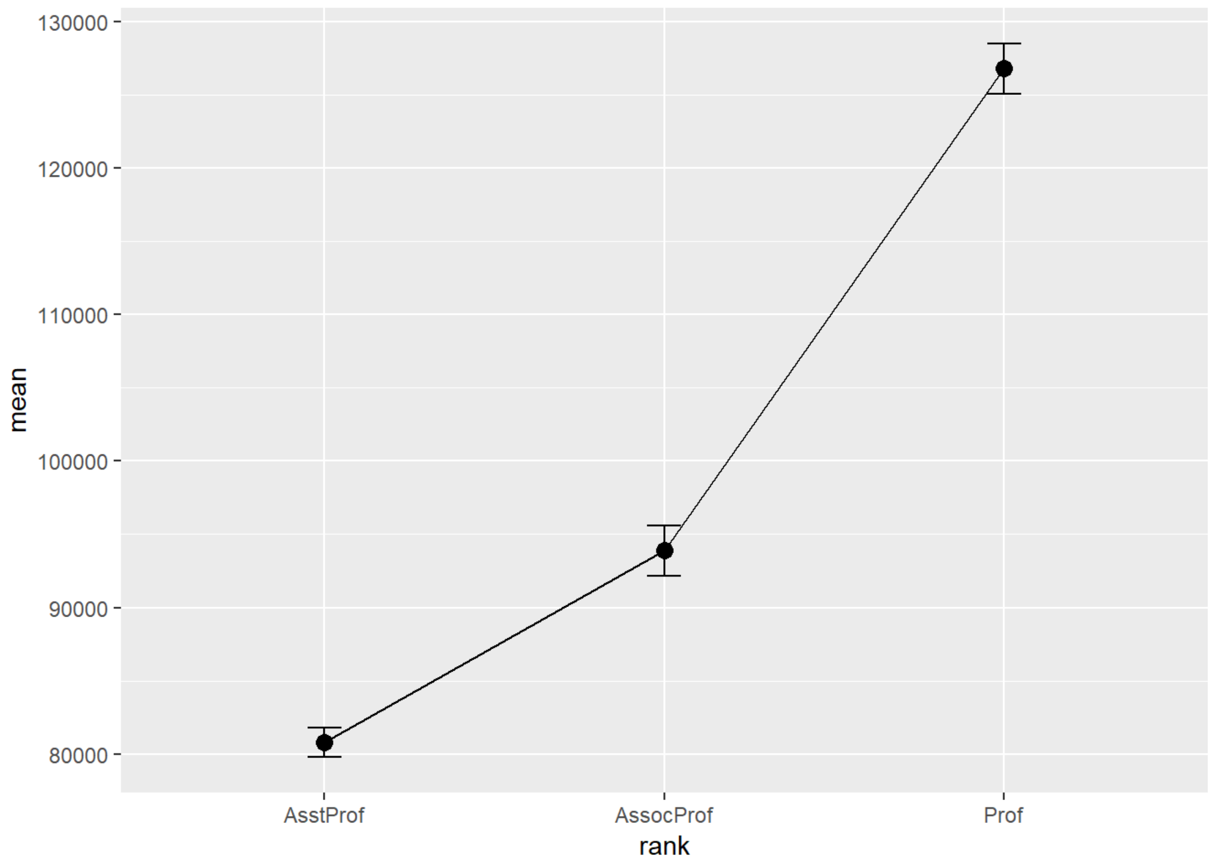


Bivariate Graphs

Categorical vs. Quantitative

Mean/SEM plots

```
# calculate means and standard errors by
rank and sex

plotdata <- Salaries %>% group_by(rank, sex)
%>% summarize(n = n(), mean = mean(salary),
sd = sd(salary), se = sd/sqrt(n))
```

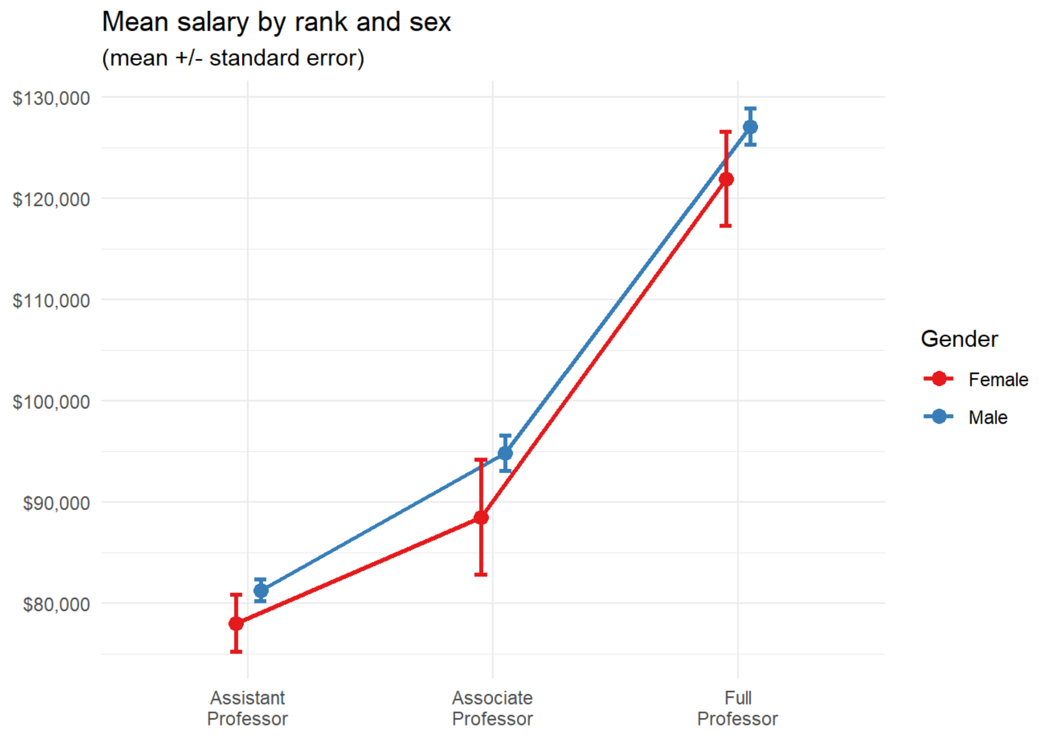


Bivariate Graphs

Categorical vs. Quantitative

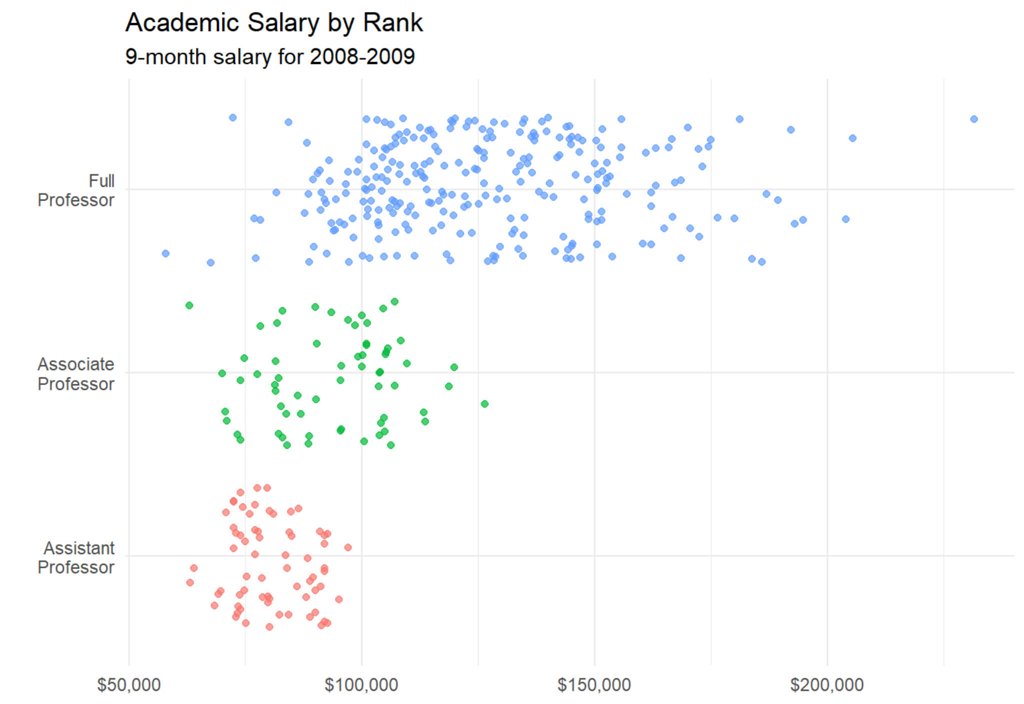
Mean/SEM plots

```r
# improved means/standard error plot

pd <- position_dodge(0.2)

ggplot(plotdata,
     aes(x = factor(rank, labels = c("Assistant\nProfessor",
"Associate\nProfessor", "Full\nProfessor")), y = mean, group=sex,
color=sex)) +
geom_point(position=pd, size = 3) + geom_line(position = pd, size = 1) +
geom_errorbar(aes(ymin = mean - se, ymax = mean + se), width = .1,
position = pd, size = 1) + scale_y_continuous(label = scales::dollar) +
scale_color_brewer(palette="Set1") + theme_minimal() +
labs(title = "Mean salary by rank and sex", subtitle = "(mean +/-
standard error)", x = "", y = "", color = "Gender")
```



Bivariate Graphs

```
# plot the distribution of salaries # by rank using jittering

library(scales)

ggplot(Salaries, aes(y = factor(rank, labels = c("Assistant\nProfessor", "Associate\nProfessor",
"Full\nProfessor")), x = salary, color = rank)) +
geom_jitter(alpha = 0.7, size = 1.5) + scale_x_continuous(label = dollar) +
labs(title = "Academic Salary by Rank", subtitle = "9-month salary for 2008-2009", x = "", y = "") +
theme_minimal() +
theme(legend.position = "none")
```



Statistical Models

Prof. Dr. Javier Valdes
Javier.valdes@th-deg.de

IAI - Institut für Angewandte Informatik
Institute for Applied Informatics

Technische Hochschule Deggendorf
Technologie Campus Freyung
Grafenauer Str. 22, D- 94078 Freyung, Germany

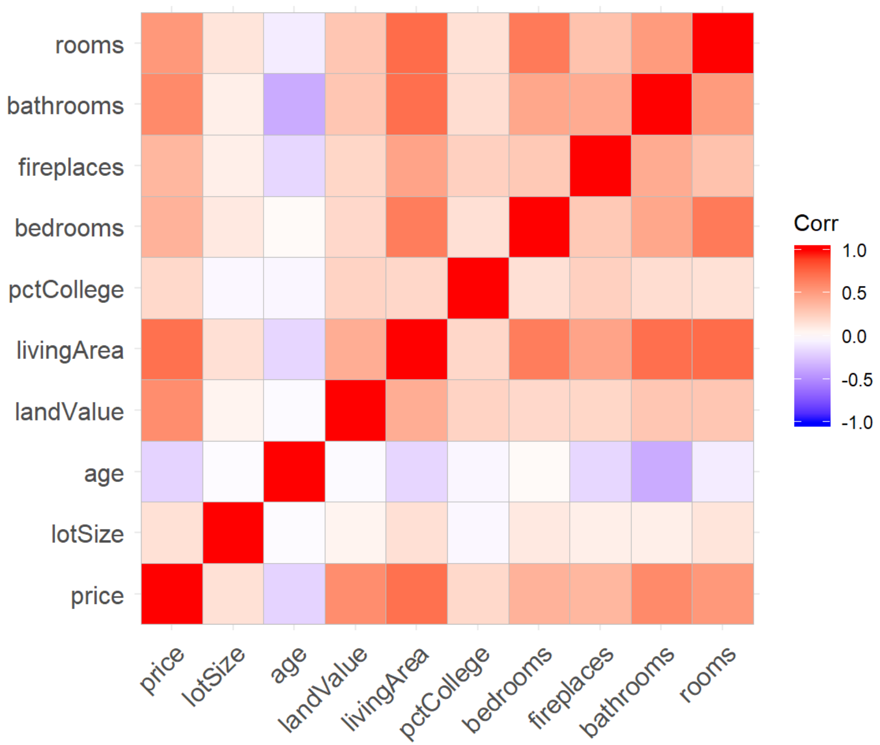
Tel.: +49 8551 91764 40
Fax: +49 8551 91764 69

Correlation plots

Correlation plots help you to visualize the pairwise relationships between a set of **quantitative variables** by displaying their correlations using color or shading.

In order to explore the relationships among the quantitative variables, we can calculate the Pearson Product-Moment correlation coefficients.

```
library(ggplot2)
library(ggcorrplot)
ggcorrplot(r)
```




```
data(SaratogaHouses, package="mosaicData")

# select numeric variables
df <- dplyr::select_if(SaratogaHouses, is.numeric)

# calulate the correlations
r <- cor(df, use="complete.obs") round(r,2)
```

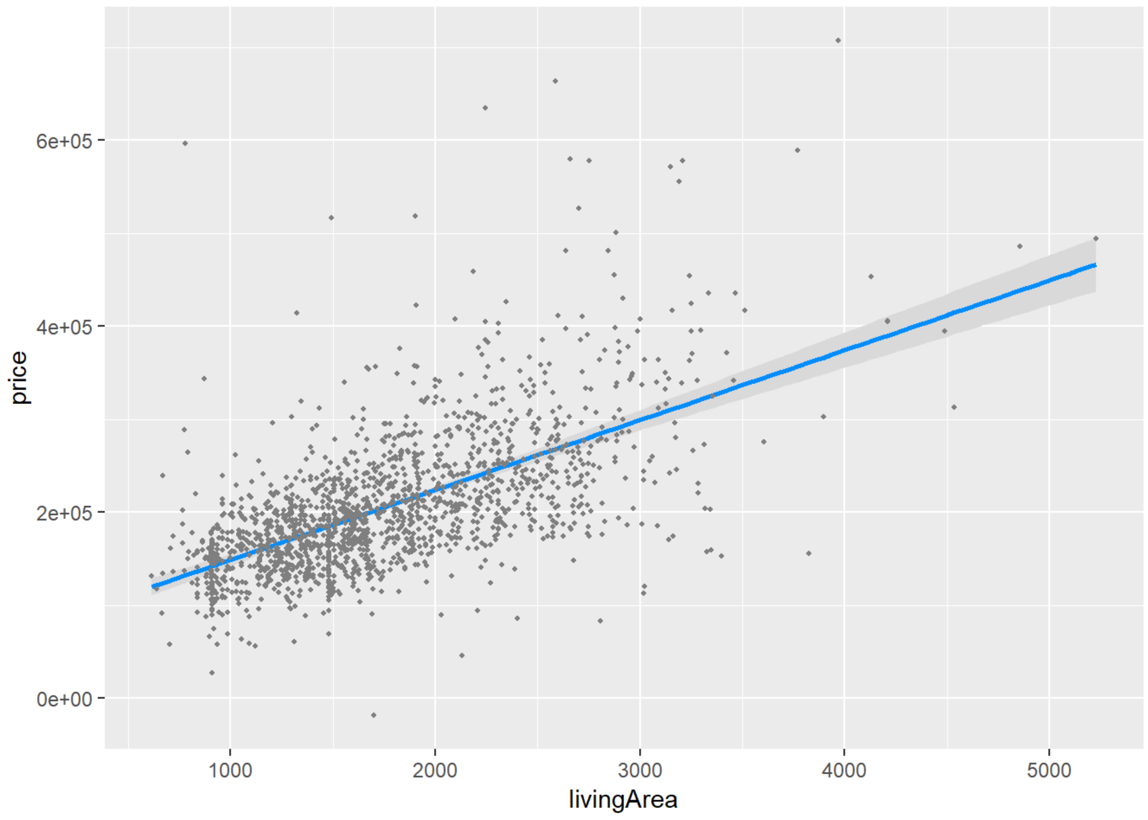
Linear Regression

Linear regression allows us to explore the relationship between a quantitative response variable and an explanatory variable while other variables are held constant.

Consider the prediction of home prices in the [Saratoga]() dataset from lot size (square feet), age (years), land value (1000s dollars), living area (square feet), number of bedrooms and bathrooms and whether the home is on the waterfront or not.

```
data(SaratogaHouses, package="mosaicData")

houses_lm <- lm(price ~ lotSize + age + landValue +
livingArea + bedrooms + bathrooms + waterfront, data =
SaratogaHouses)
```

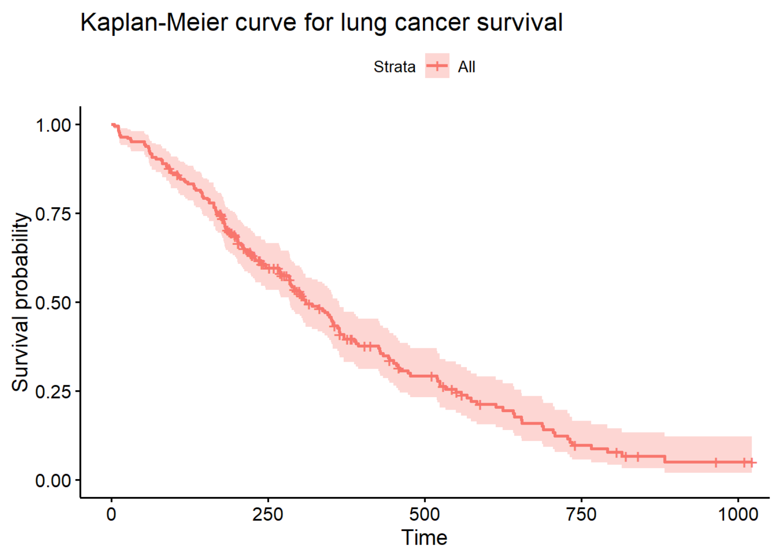
```
# conditional plot of price vs. living area

library(ggplot2)
library(visreg)
visreg(houses_lm, "livingArea", gg = TRUE)
```



Survival plots

In many research settings, the response variable is the time to an event. This is frequently true in healthcare research, where we are interested in time to recovery, time to death, or time to relapse.

If the event has not occurred for an observation (either because the study ended or the patient dropped out) the observation is said to be *censored*.




```
# plot survival curve

library(survival)
library(survminer)
data(lung)

sfit <- survfit(Surv(time, status) ~ 1, data=lung)
ggsurvplot(sfit, title="Kaplan-Meier curve for lung
cancer survival")
```

The outcome for each patient is measured by two variables

- *time* - survival time in days

- *status* - 1=censored, 2=dead