Basic Data Analytics Using R

Chapter 3 from "Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data"
1st Edition by EMC Education Services

Ka-Chun Wong, Department of Computer Science City University of Hong Kong
Charles Tappert Seidenberg, School of CSIS, Pace University

Data Analytics Using R

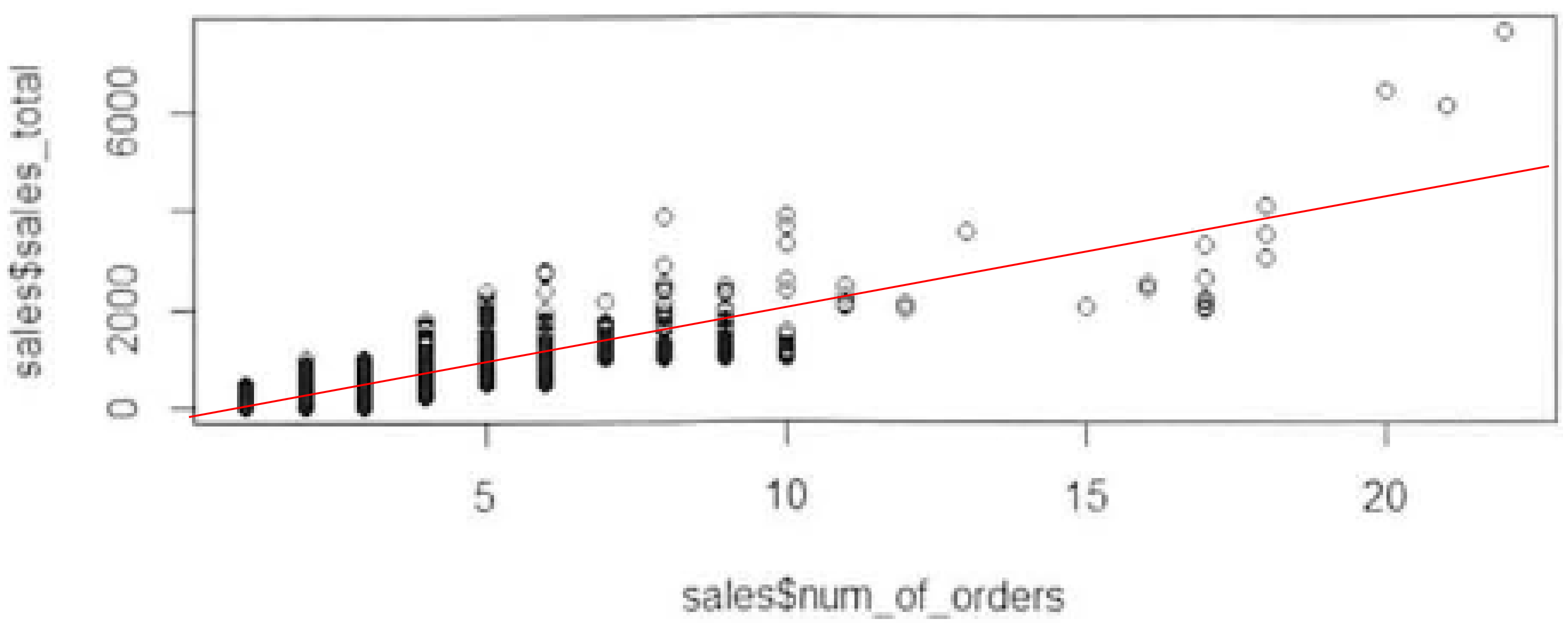
A. An overview of R
B. Using R to perform exploratory data analysis tasks using visualization

3.1 Introduction to R

- Generic R functions are functions that share the same name but behave differently depending on the type of arguments they receive (polymorphism)
- Some important functions used in chapter (most are generic)
  - head() displays first six records of a file
  - summary() generates descriptive statistics
  - plot() can generate a scatter plot of one variable against another
  - lm() applies a linear regression model between two variables
  - hist() generates a histogram
  - help() provides details of a function

3.1 Introduction to R
Example: number of orders vs sales

```
plot(sales$num_of_orders,sales$sales_total,
        main="Number of Orders vs. Sales")
```



abline(lm(formula = (sales$sales_total ~ sales$num_of_orders))

intercept = -154.1   slope = 166.2

3.1 Introduction to R

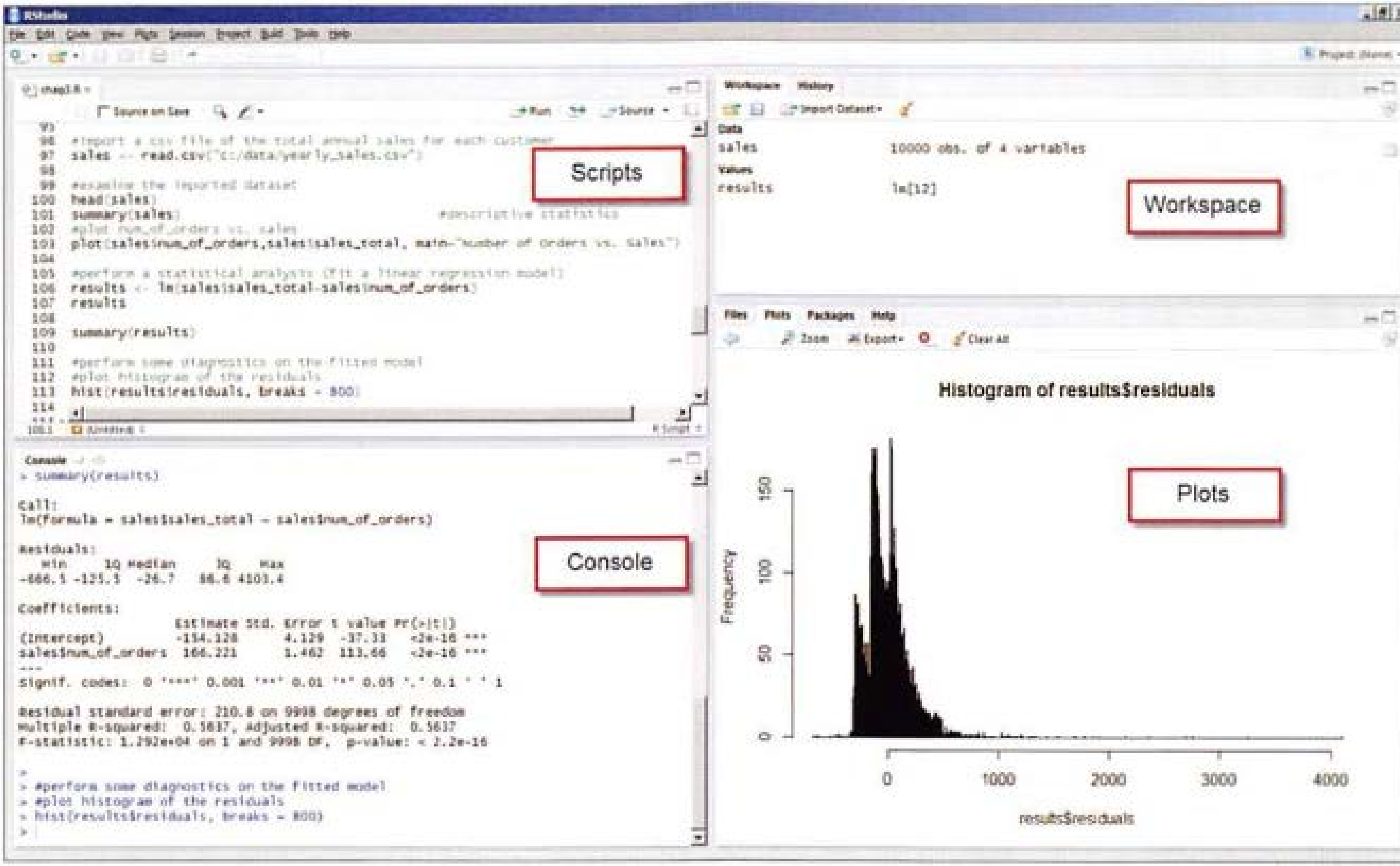
- 3.1.1 R Graphical User Interfaces
  - Getting R and RStudio
- 3.1.2 Data Import and Export
  - Necessary for project work
- 3.1.3 Attributes and Data Types
  - Vectors, matrices, data frames
- 3.1.4 Descriptive Statistics
  - summary(), mean(), median(), sd()

3.1.1 Getting R and RStudio

- Download R and install (32-bit and 64-bit)
  - https://www.r-project.org/

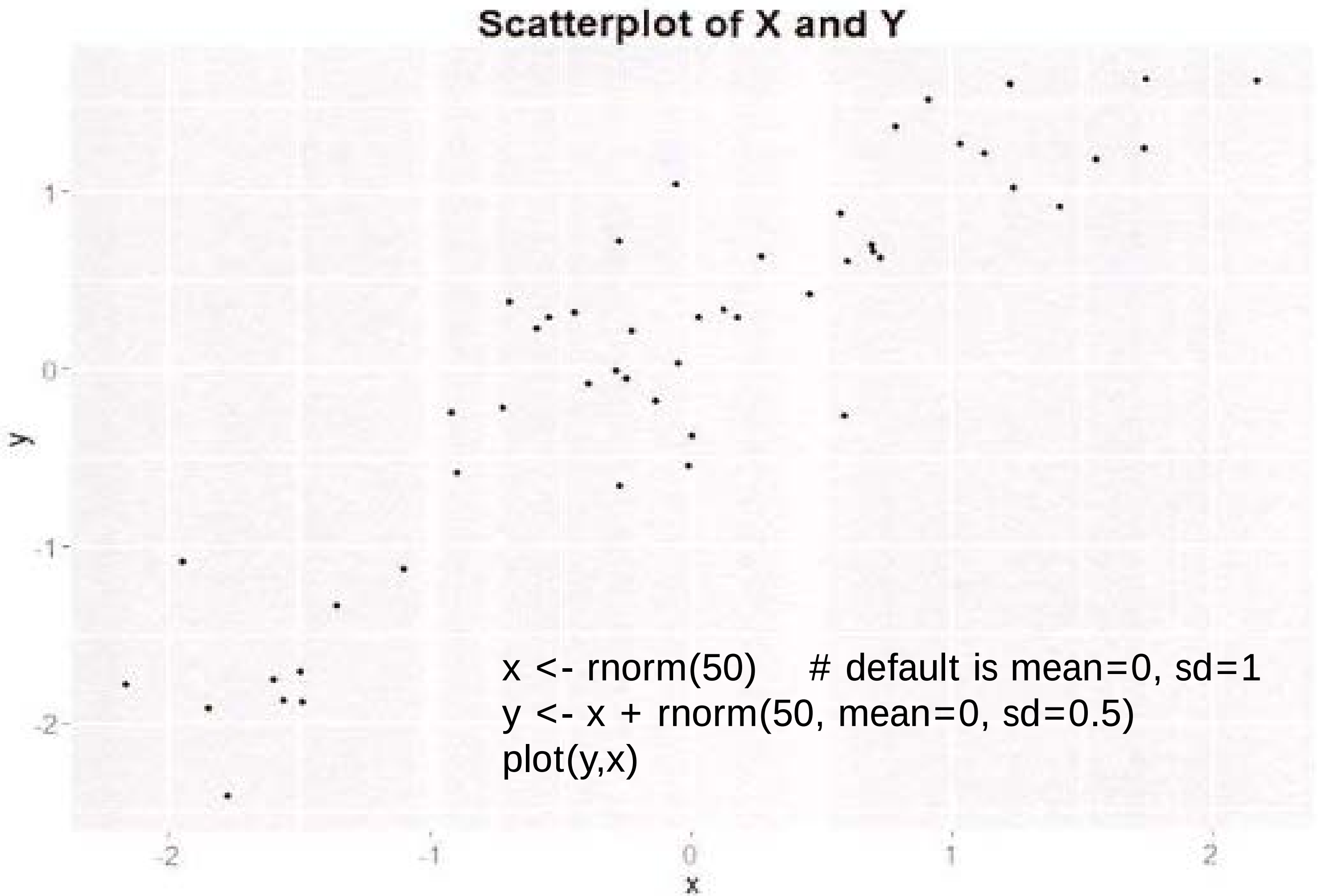
- Download RStudio and install
  - https://www.rstudio.com/

3.1.1 RStudio GUI



3.2 Exploratory Data Analysis

Scatterplots show possible relationships




```
x <- rnorm(50)     # default is mean=0, sd=1
y <- x + rnorm(50, mean=0, sd=0.5)
plot(y,x)
```

3.2 Exploratory Data Analysis

- 3.2.1 Visualization before Analysis
- 3.2.2 Dirty Data
- 3.2.3 Visualizing a Single Variable
- 3.2.4 Examining Multiple Variables
- 3.2.5 Data Exploration versus Presentation

3.2.1 Visualization before Analysis

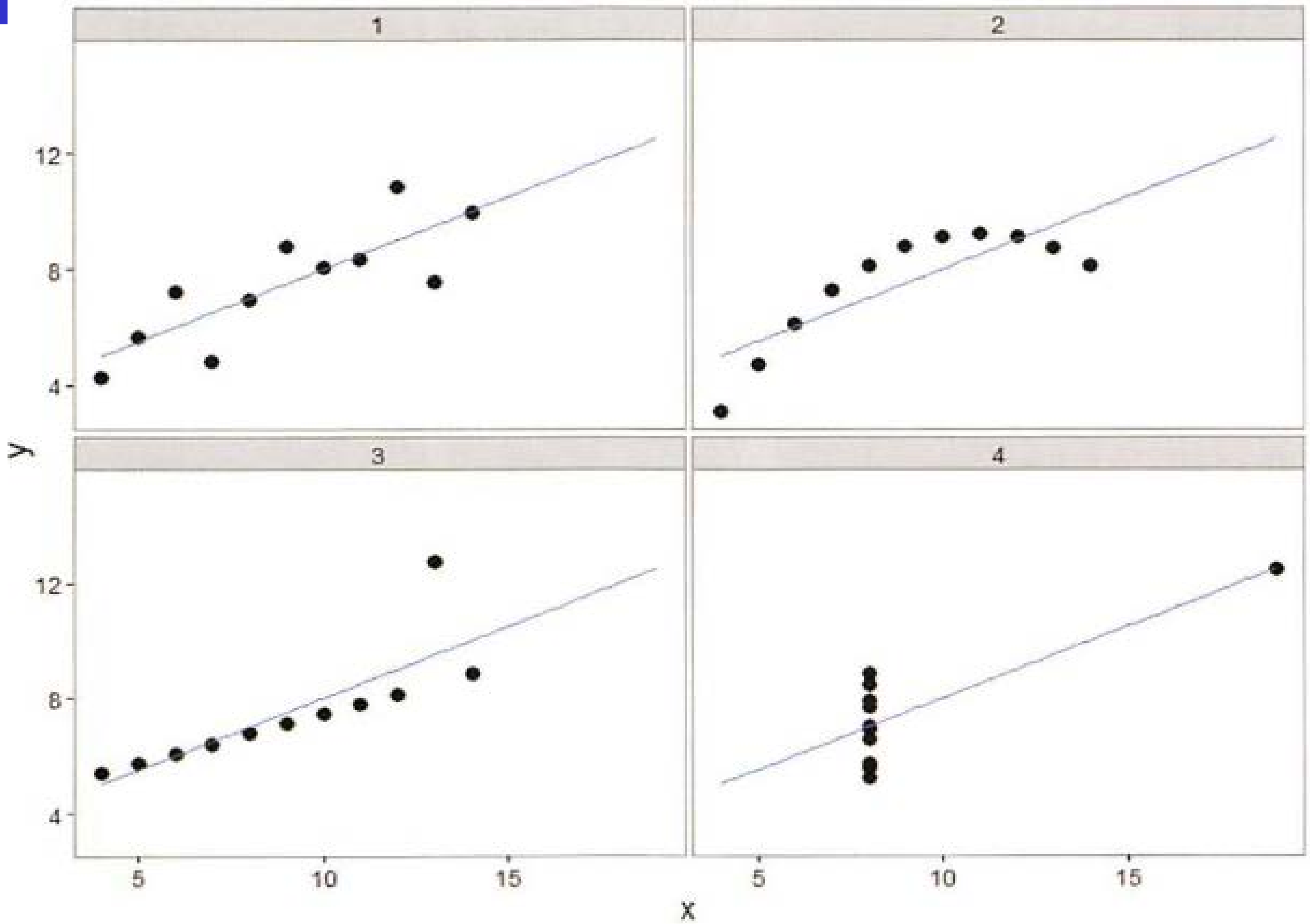
Anscombe's quartet

4 datasets with the same statistics

Statistical Property	Value
Mean of x	9
Variance of x	11
Mean of y	7.50 (to 2 decimal points)
Variance of y	4.12 or 4.13 (to 2 decimal points)
Correlations between x and y	0.816
Linear regression line	$y = 3.00 + 0.50x$ (to 2 decimal points)

3.2.1 Visualization before Analysis

Anscombe's quartet – visualized



3.2.1 Visualization before Analysis

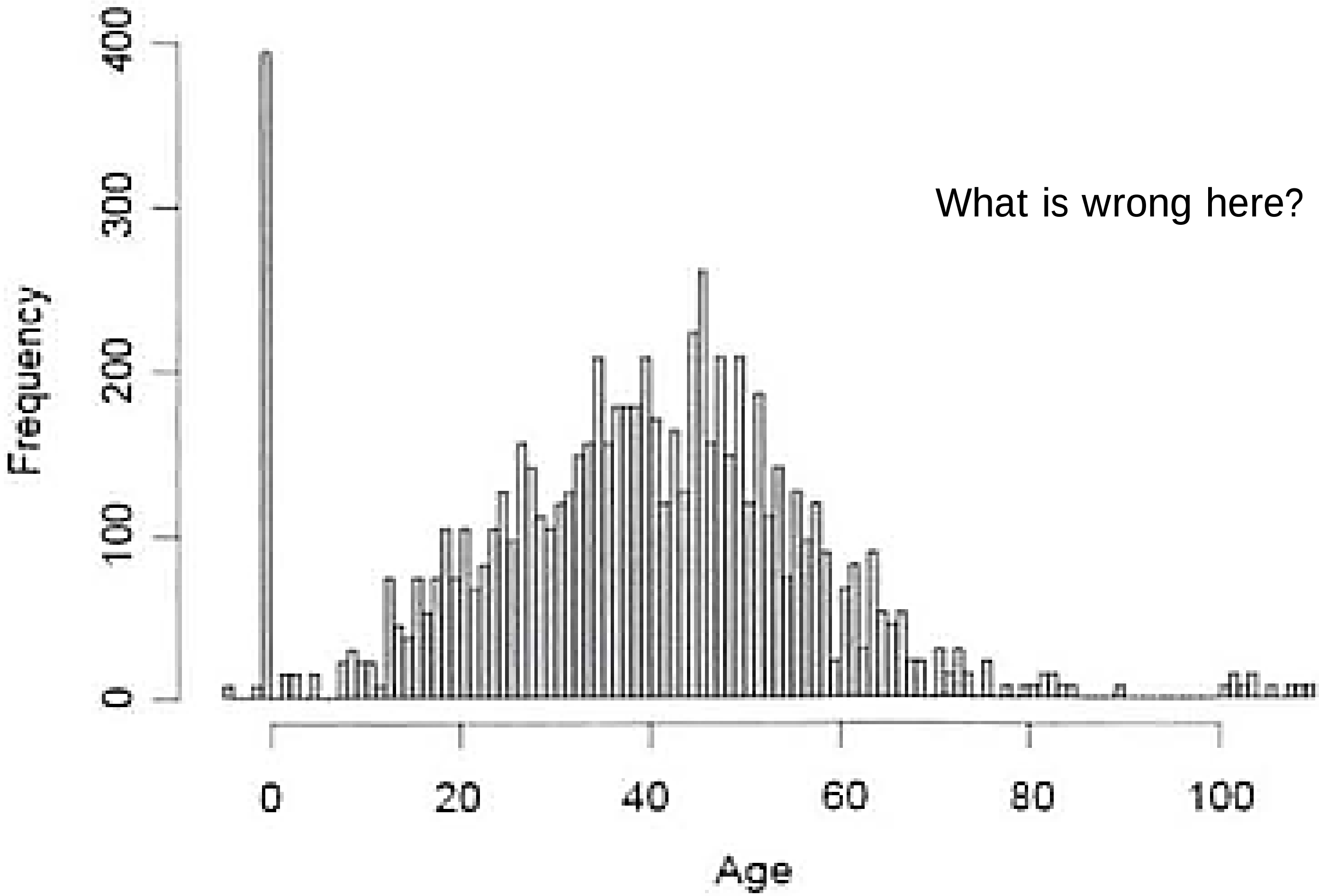
Anscombe’s quartet – Rstudio exercise

- Enter and plot Anscombe’s dataset #3
- and obtain the linear regression line

```
x <- 4:14
x
y <- c(5.39,5.73,6.08,6.42,6.77,7.11,7.46,7.81,8.15,12.74,8.84)
y
summary(x)
var(x)
summary(y)
var(y)
plot(y~x)
lm(y~x)
```

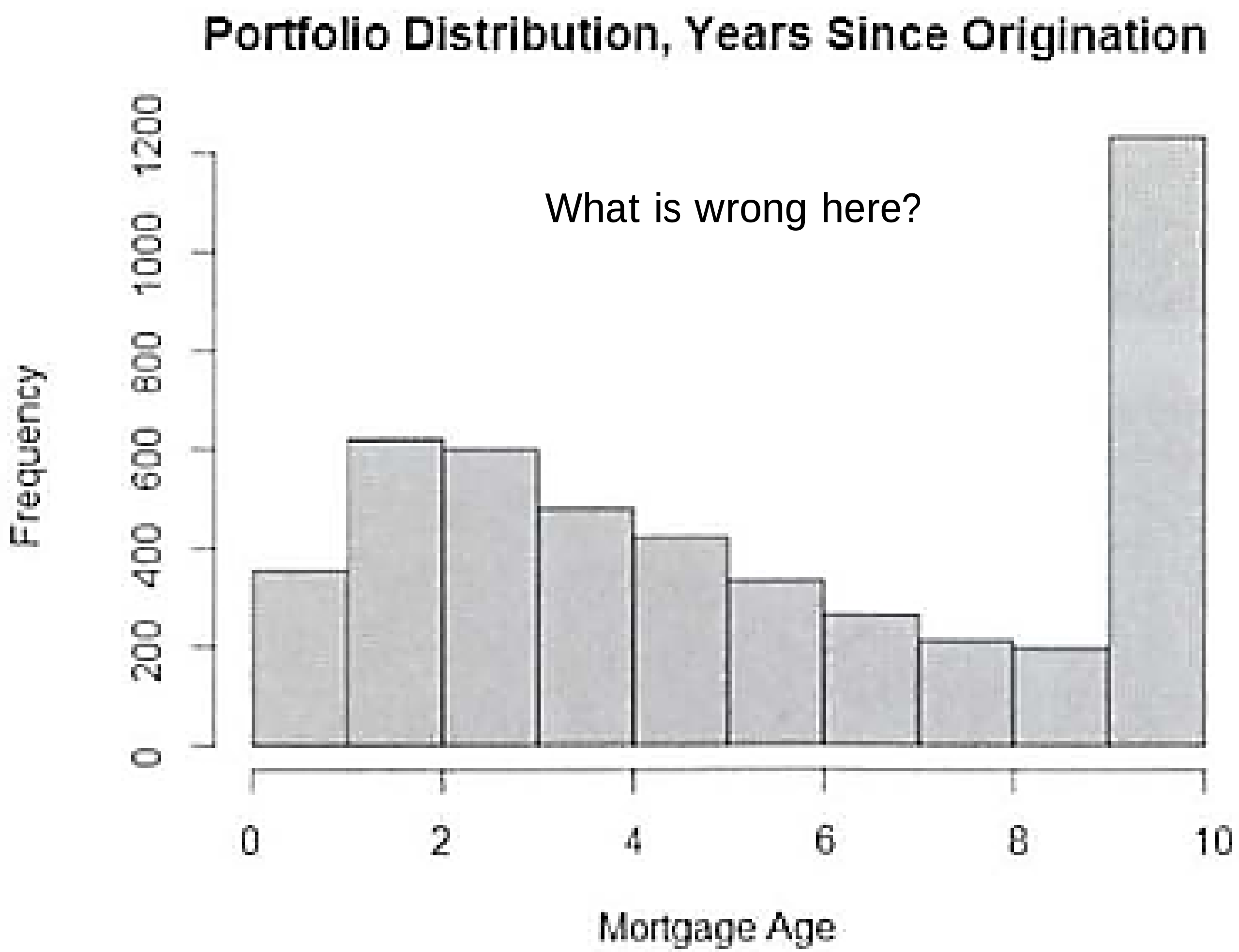
3.2.2 Dirty Data

Age Distribution of bank account holders



What is wrong here?

3.2.2 Dirty Data
Age of Mortgage



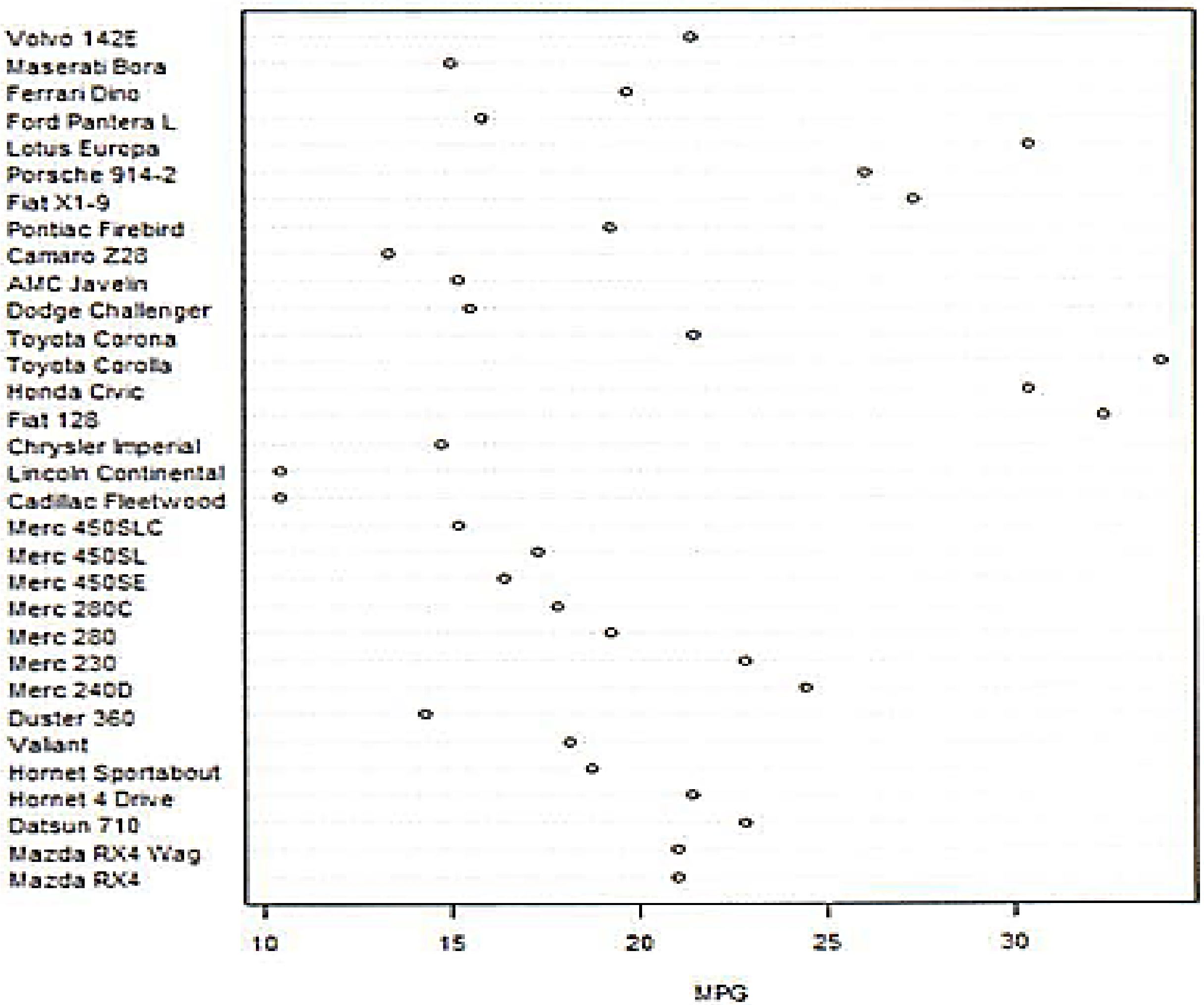
3.2.3 Visualizing a Single Variable

Example Visualization Functions

Function	Purpose
`plot(data)`	Scatterplot where x is the index and y is the value; suitable for low-volume data
`barplot(data)`	Barplot with vertical or horizontal bars
`dotchart(data)`	Cleveland dot plot [12]
`hist(data)`	Histogram
`plot(density(data))`	Density plot (a continuous histogram)
`stem(data)`	Stem-and-leaf plot
`rug(data)`	Add a rug representation (1-d plot) of the data to an existing plot

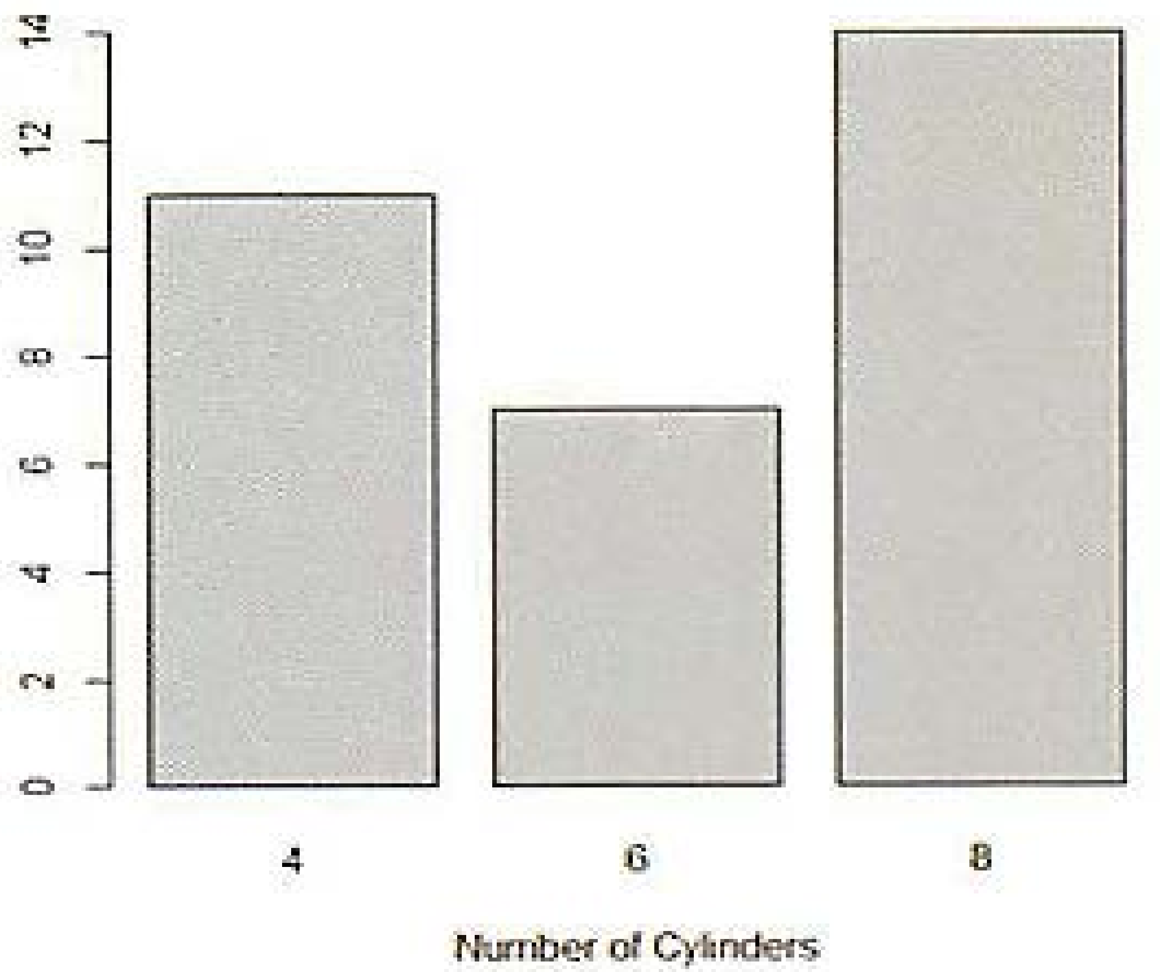
3.2.3 Visualizing a Single Variable

Dotchart – MPG of Car Models



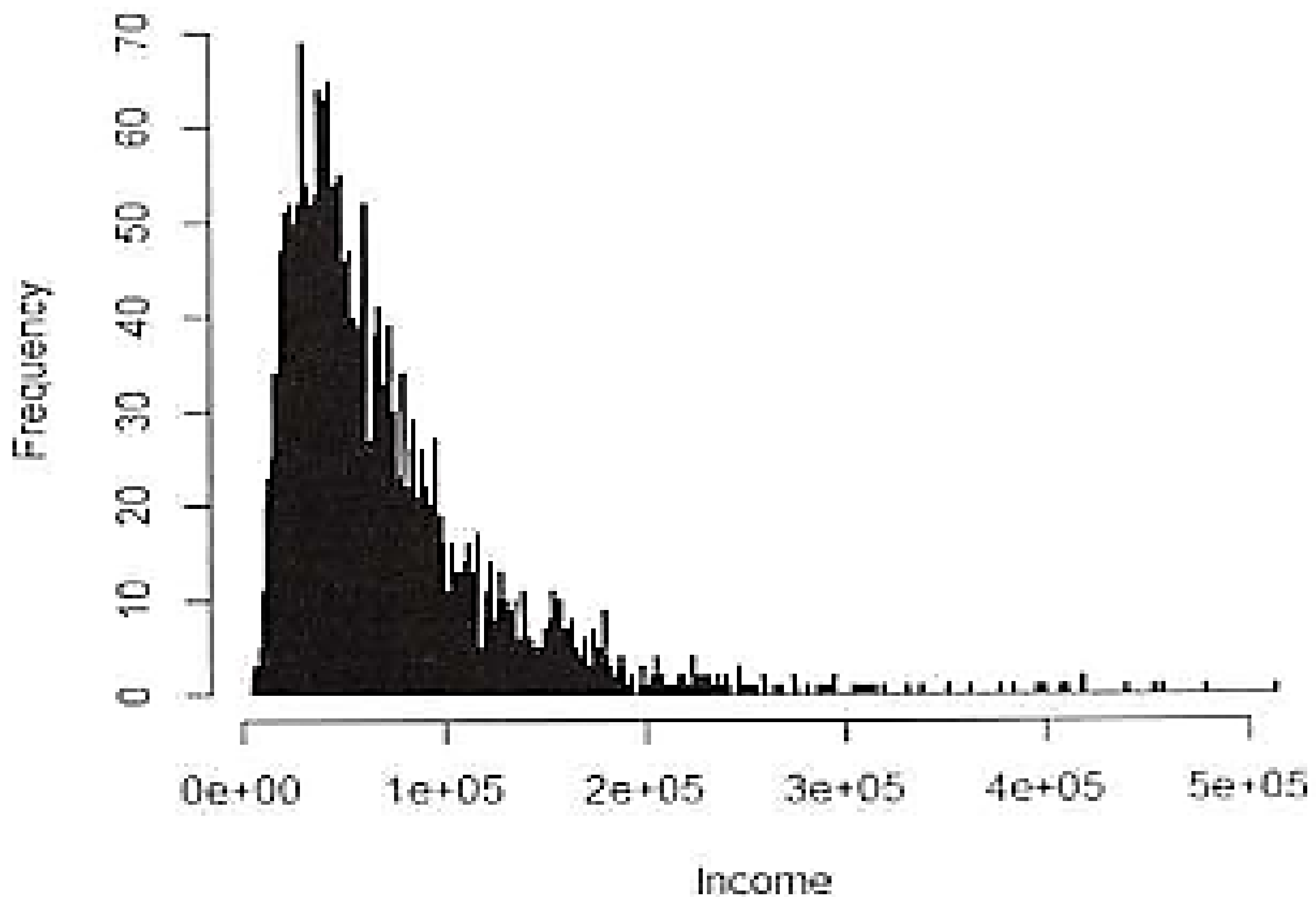
3.2.3 Visualizing a Single Variable

Barplot – Distribution of Car Cylinder Counts

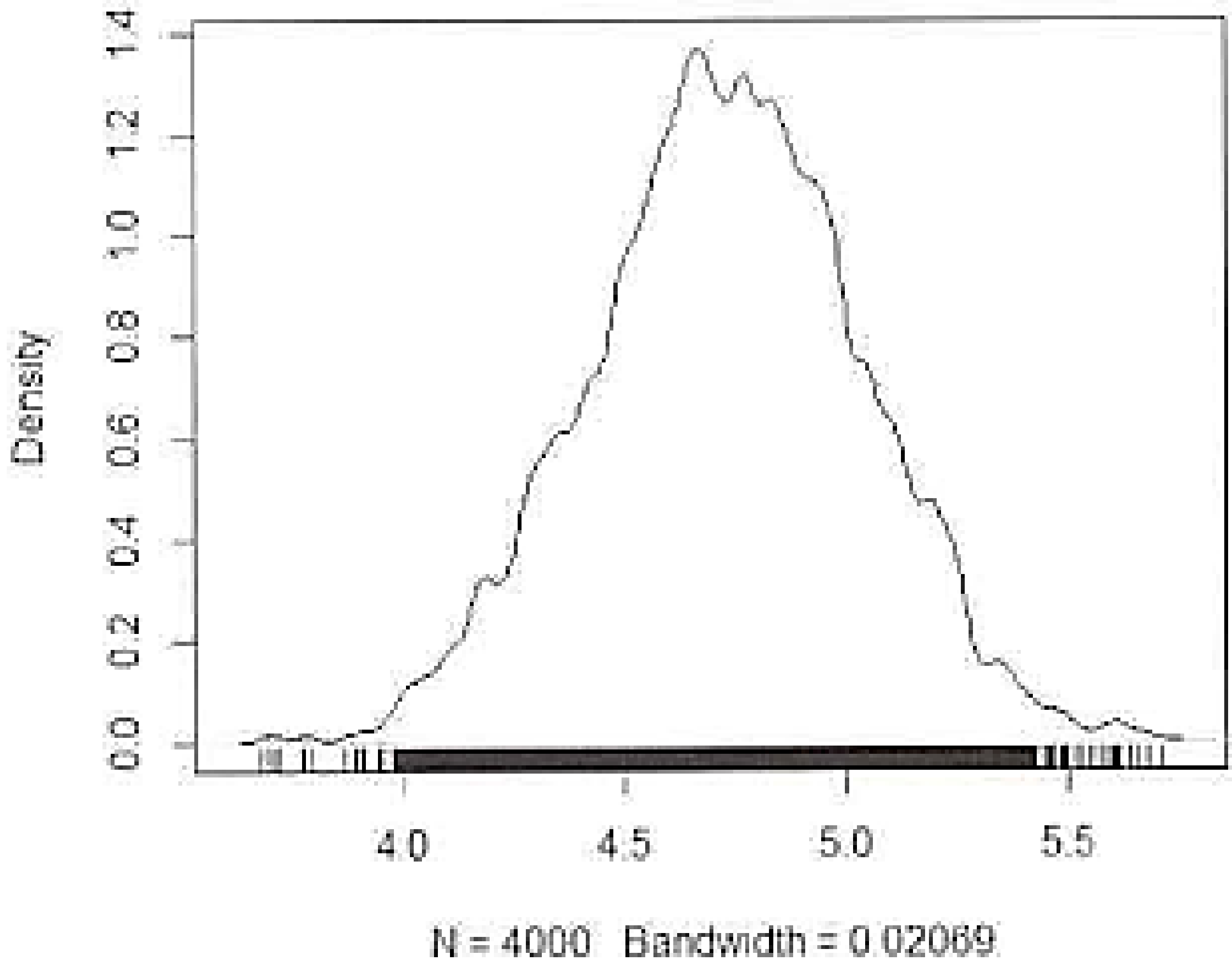


3.2.3 Visualizing a Single Variable

Histogram – Income



3.2.3 Visualizing a Single Variable
Density – Income (log10 scale)



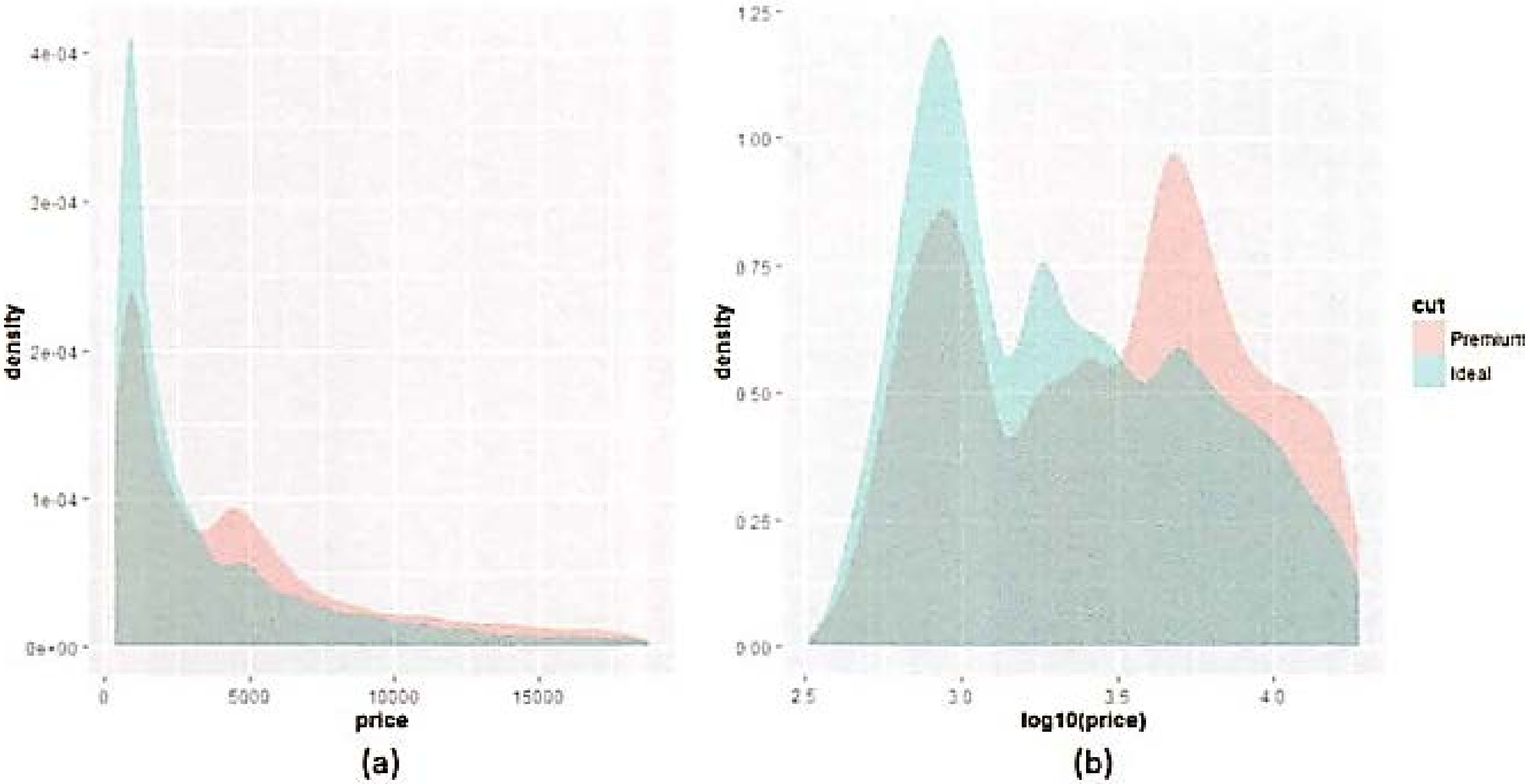
3.2.3 Visualizing a Single Variable

Density – Income (log10 scale)

- In this case, the log density plot emphasizes the log nature of the distribution
- The rug() function at the bottom creates a one-dimensional density plot to emphasize the distribution

3.2.3 Visualizing a Single Variable

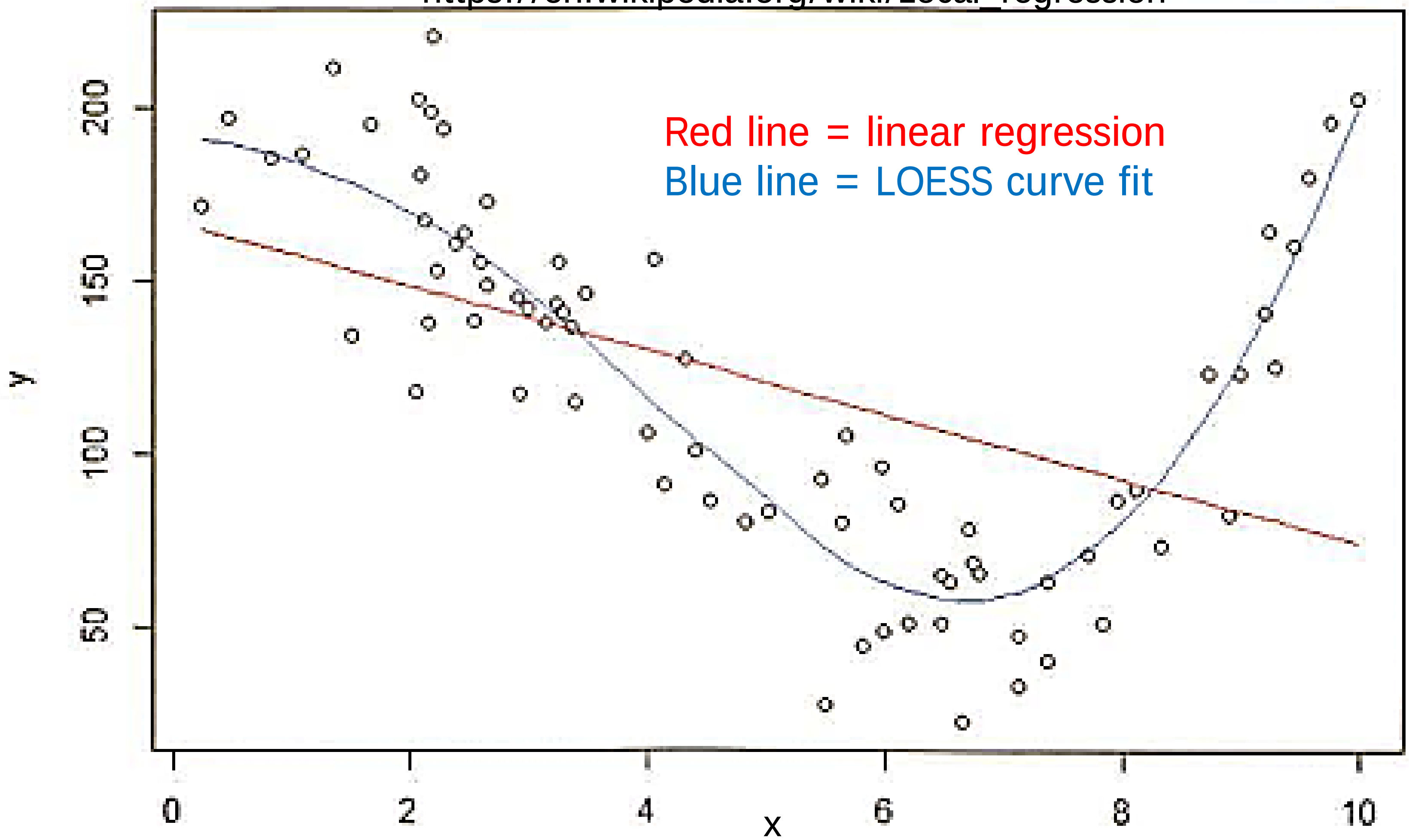
Density plots – Diamond prices, log of same



3.2.4 Examining Multiple Variables

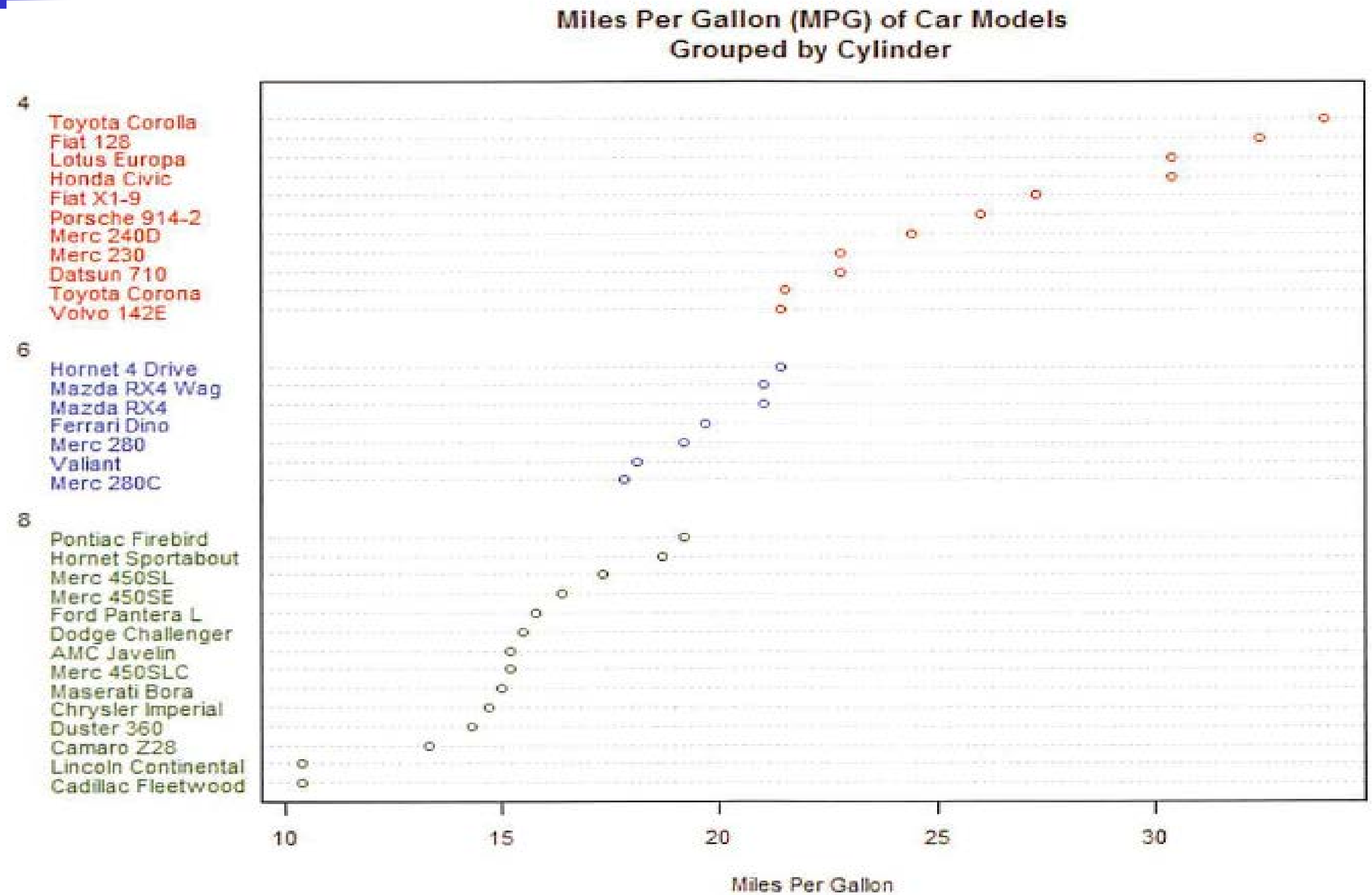
Examining two variables with regression

https://en.wikipedia.org/wiki/Local_regression



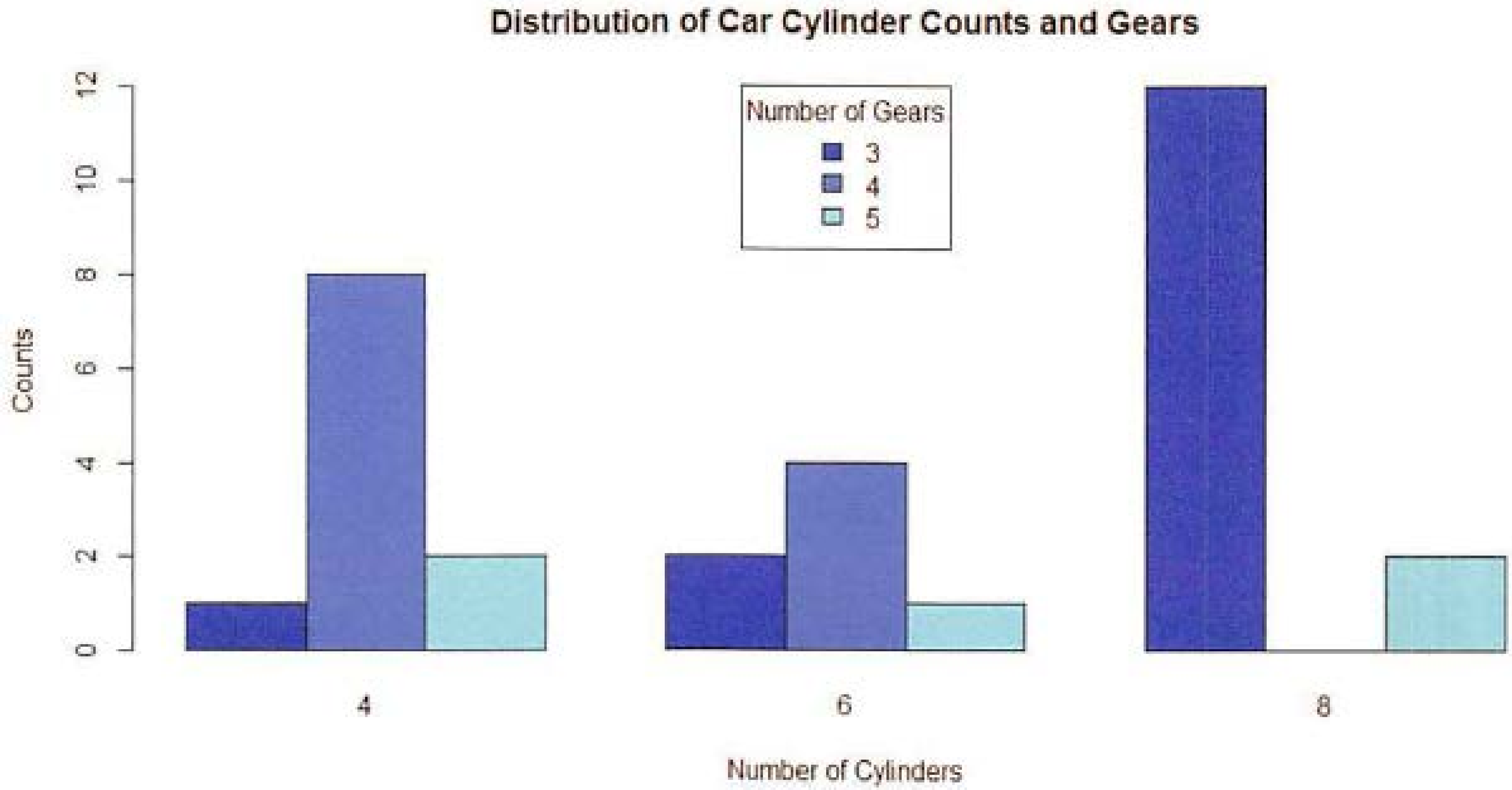
3.2.4 Examining Multiple Variables

Dotchart: MPG of car models grouped by cylinder



3.2.4 Examining Multiple Variables

Barplot: visualize multiple variables



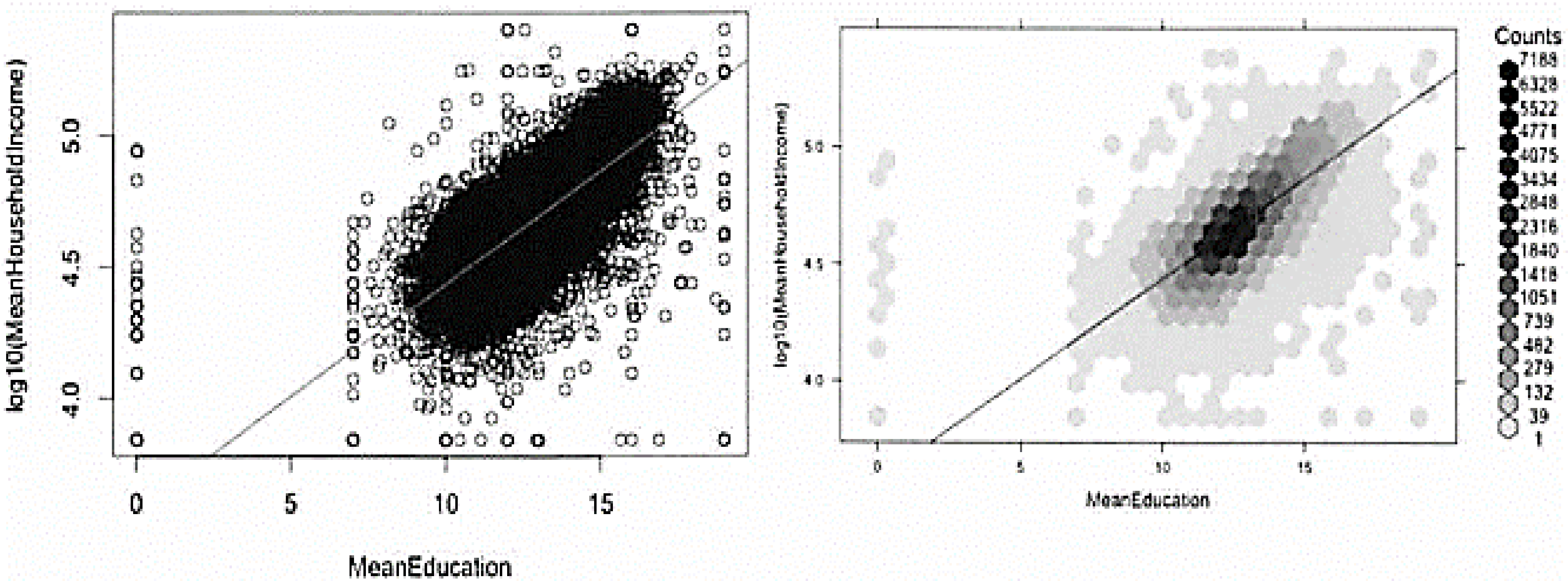
3.2.4 Examining Multiple Variables
Box-and-whisker plot: income versus region

Box contains central 50% of data
Line inside box is median value
Shows data quartiles

3.2.4 Examining Multiple Variables

Scatterplot (a) & Hexbinplot – income vs education

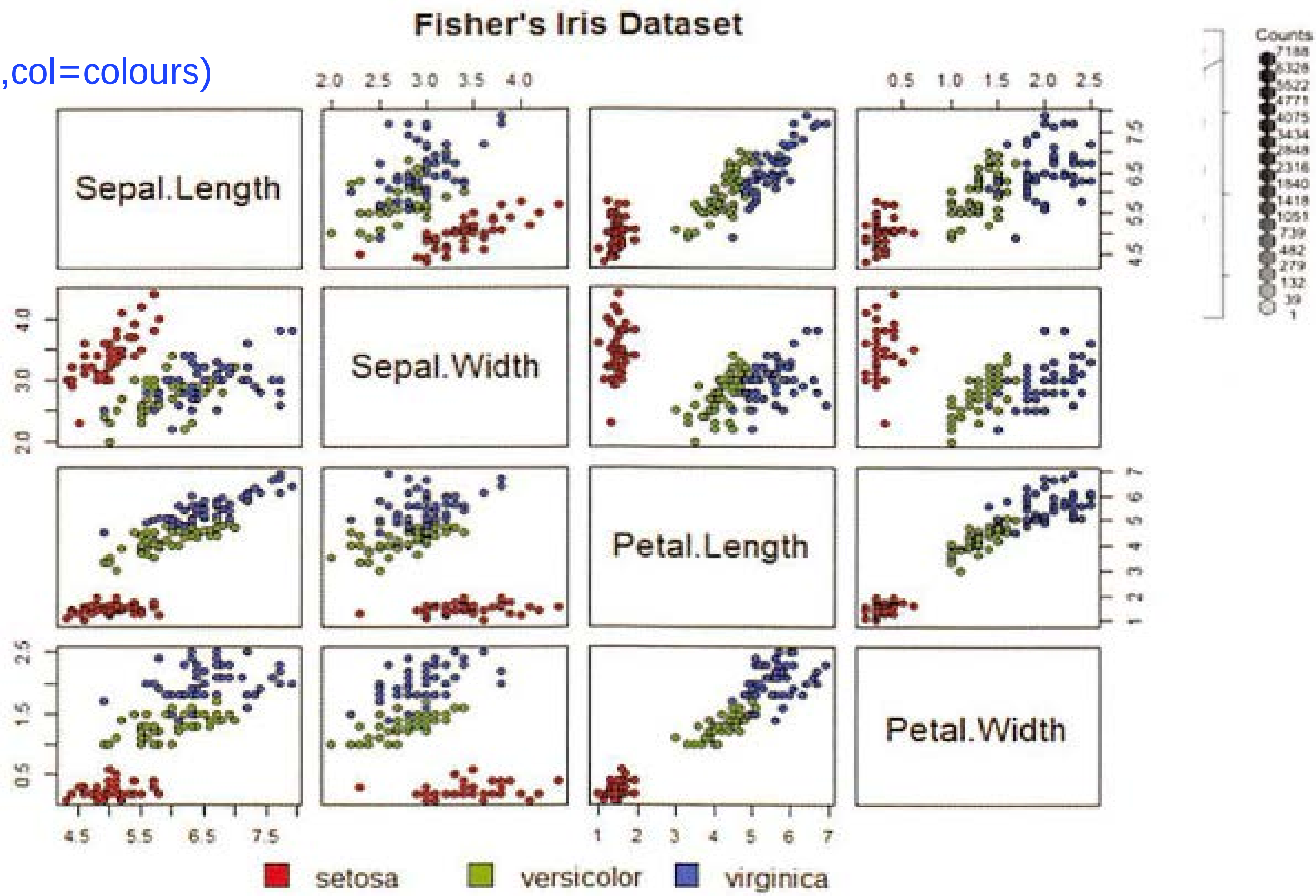
The hexbinplot combines the ideas of scatterplot and histogram
For high-volume big data hexbinplot may be better than scatterplot



(a) (b)

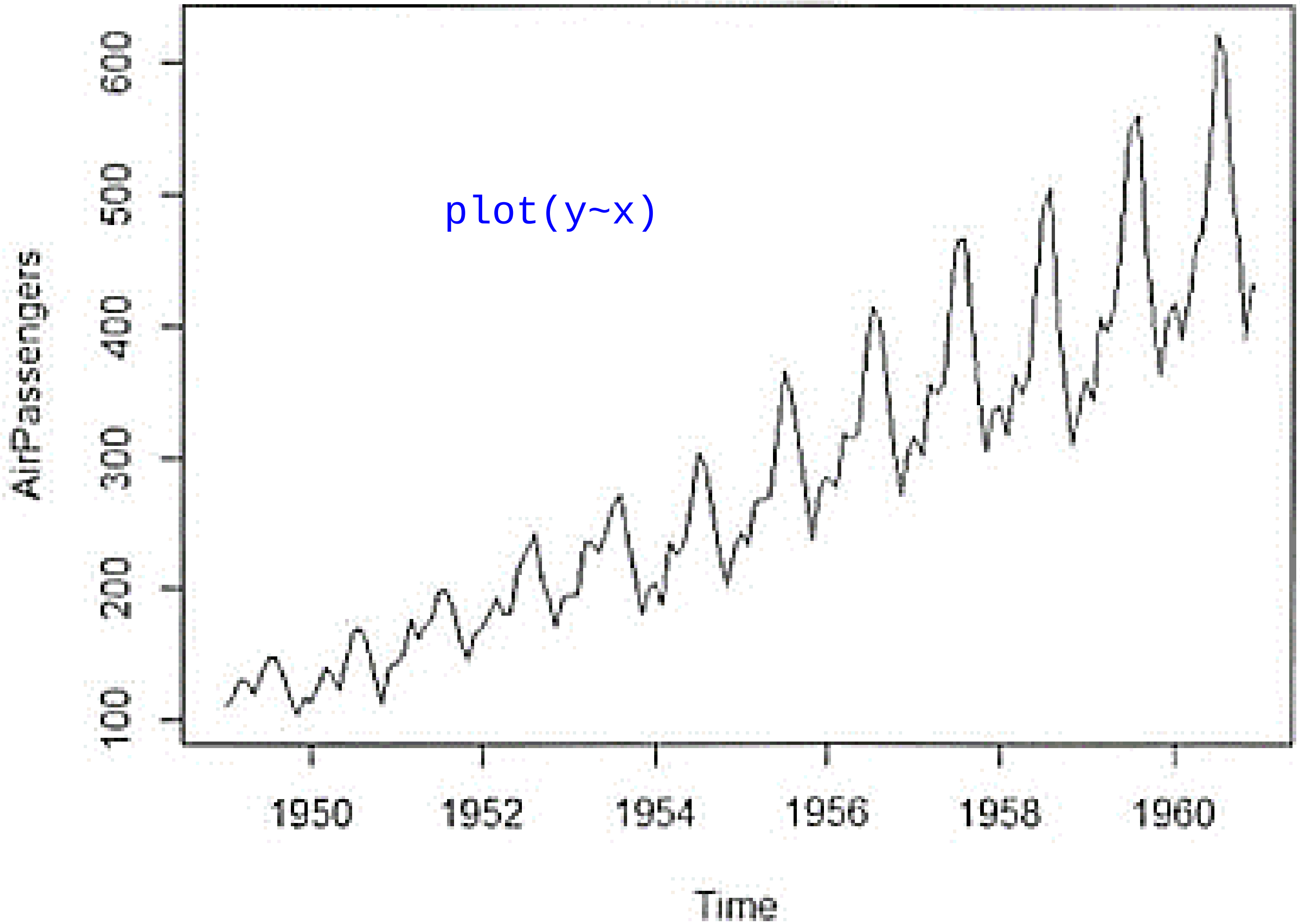
3.2.4 Examining Multiple Variables
Matrix of Scatterplots

pairs(data,col=colours)



3.2.4 Examining Multiple Variables

Variable over time – airline passenger counts

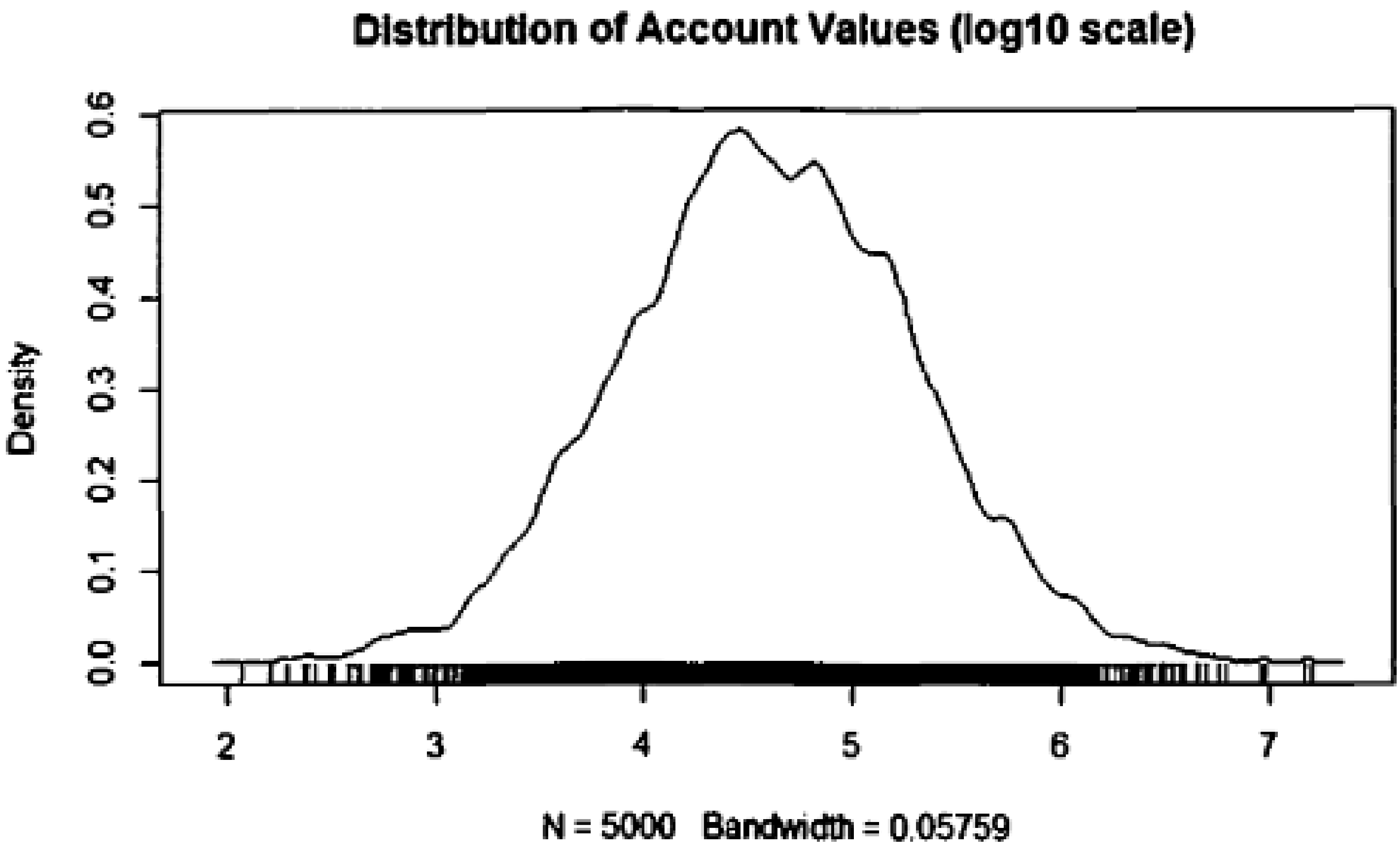


3.2.5 Exploration vs Presentation

- Data visualization for data exploration is different from presenting results to stakeholders
  - Data scientists prefer graphs that are technical in nature
  - Nontechnical stakeholders prefer simple and clear graphics that focus on the message rather than the data

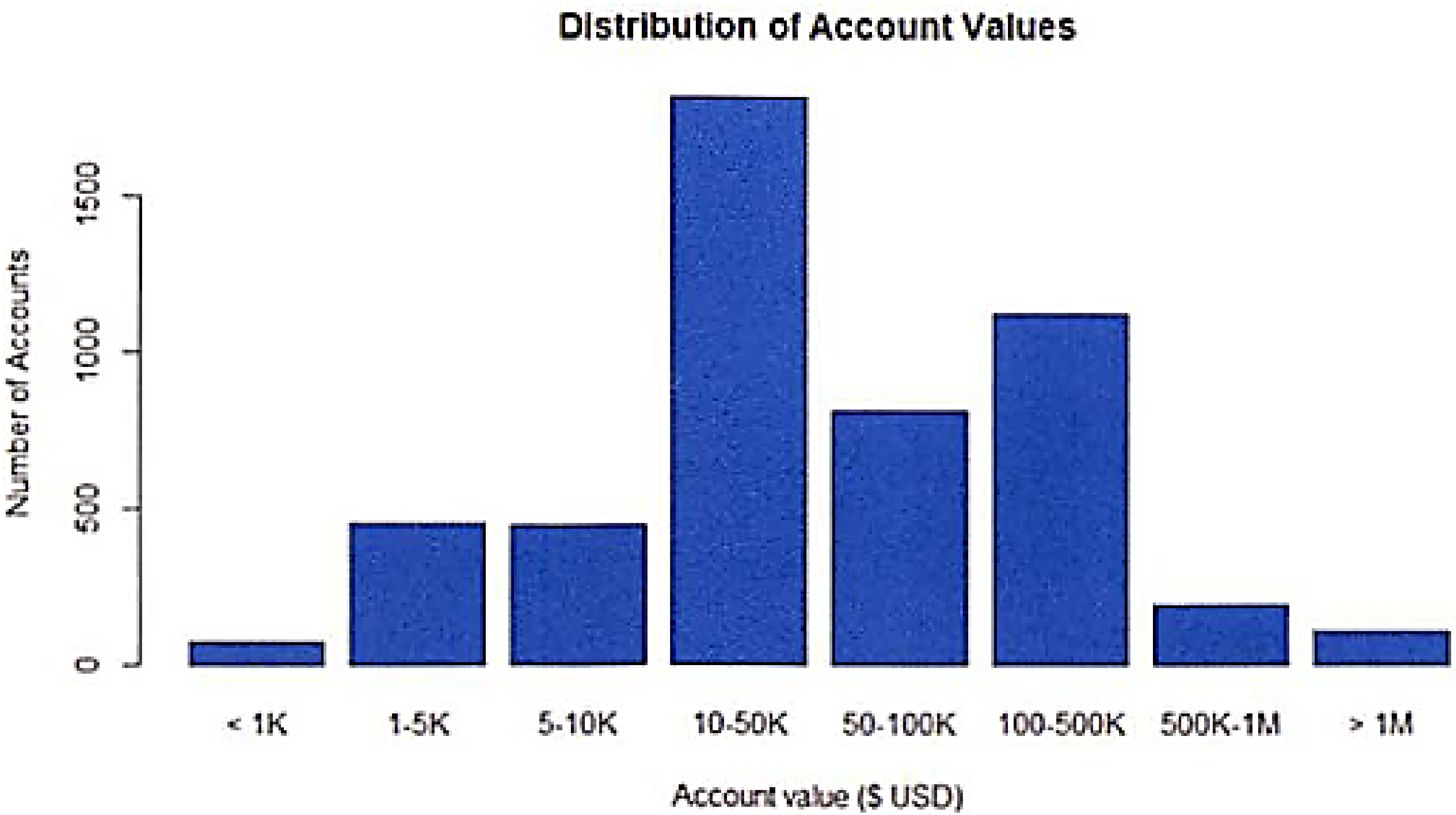
3.2.5 Exploration vs Presentation

Density plots better for data scientists



3.2.5 Exploration vs Presentation

Histograms better to show stakeholders



https://support.rstudio.com/hc/en-us/articles/201057987-Quick-list-of-useful-R-packages

Package Usage in R

Many useful R function come in packages, free libraries of code written by R's active user community. To install an R package, open an R session and type at the command line

```
install.packages("<the package's name>")
```

R will download the package from CRAN, so you'll need to be connected to the internet. Once you have a package installed, you can make its contents available to use in your current R session by running

```
library("<the package's name>")
```

There are thousands of helpful R packages for you to use, but navigating them all can be a challenge. To help you out, we've compiled this guide to some of the best. We've used each of these, and found them to be outstanding – we've even written some of them. But you don't have to take our word for it, these packages are also some of the top most downloaded R packages.

https://support.rstudio.com/hc/en-us/articles/201057987-Quick-list-of-useful-R-packages

To load data in R

DBI - The standard for communication between R and relational database management systems. Packages that connect R to databases depend on the DBI package.

odbc - Use any ODBC driver with the odbc package to connect R to your database.

RMySQL, RPostgresSQL, RSQLite - If you'd like to read in data from a database, these packages are a good place to start. Choose the package that fits your type of database.

XLConnect, xlsx - These packages help you read and write Micorsoft Excel files from R. You can also just export your spreadsheets from Excel as .csv's.

foreign - Want to read a SAS data set into R? Or an SPSS data set? Foreign provides functions that help you load data files from other programs into R.

haven - Enables R to read and write data from SAS, SPSS, and Stata.

R can handle plain text files – no package required. Just use the functions read.csv, read.table, and read.fwf. If you have even more exotic data, consult the CRAN guide to data import and export.

and more……

https://support.rstudio.com/hc/en-us/articles/201057987-Quick-list-of-useful-R-packages

To manipulate data in R

dplyr - Essential shortcuts for subsetting, summarizing, rearranging, and joining together data sets. dplyr is our go to package for fast data manipulation.
tidyr - Tools for changing the layout of your data sets. Use the gather and spread functions to convert your data into the tidy format, the layout R likes best.
stringr - Easy to learn tools for regular expressions and character strings.
lubridate - Tools that make working with dates and times easier.

and more……

https://support.rstudio.com/hc/en-us/articles/201057987-Quick-list-of-useful-R-packages

To visualize data in R

ggplot2 - R's famous package for making beautiful graphics. ggplot2 lets you use the grammar of graphics to build layered, customizable plots.
ggvis - Interactive, web based graphics built with the grammar of graphics.
rgl - Interactive 3D visualizations with R
htmlwidgets - A fast way to build interactive (javascript based) visualizations with R.
Packages that implement htmlwidgets include:

- leaflet (maps)
- dygraphs (time series)
- DT (tables)
- diagrammeR (diagrams)
- network3D (network graphs)
- threeJS (3D scatterplots and globes)

googleVis - Let's you use Google Chart tools to visualize data in R.

and more……

https://support.rstudio.com/hc/en-us/articles/201057987-Quick-list-of-useful-R-packages

To model data in R

car - car's Anova function is popular for making type II and type III Anova tables.
mgcv - Generalized Additive Models
lme4/nlme - Linear and Non-linear mixed effects models
randomForest - Random forest methods from machine learning
multcomp - Tools for multiple comparison testing
vcd - Visualization tools and tests for categorical data
glmnet - Lasso and elastic-net regression methods with cross validation
survival - Tools for survival analysis
caret - Tools for training regression and classification models

and more……

Data visualization with ggplot2 : : CHEAT SHEET

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and **geoms**—visual marks that represent data points.

To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.

Complete the template below to build a graph.

```
ggplot (data = <DATA> ) +
  <GEOM_FUNCTION>(mapping = aes( <MAPPINGS> ),
  stat = <STAT>, position = <POSITION> ) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION> +
  <SCALE_FUNCTION> +
  <THEME_FUNCTION>
```

required (ggplot, data, GEOM_FUNCTION, MAPPINGS)

Not required, sensible defaults supplied (stat, position, COORDINATE_FUNCTION, FACET_FUNCTION, SCALE_FUNCTION, THEME_FUNCTION)

ggplot(data = mpg, **aes**(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

last_plot() Returns the last plot.

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

Aes

Common aesthetic values.

color and **fill** - string ("red", "#RRGGBB")

linetype - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "dotdash", 5 = "longdash", 6 = "twodash")

lineend - string ("round", "butt", or "square")

linejoin - string ("round", "mitre", or "bevel")

size - integer (line width in mm)

shape - integer/shape name or a single character ("a")

0 1 2 3 4 5 6 7 8 9 10 11 12
13 14 15 16 17 18 19 20 21 22 23 24 25

Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

a + geom_blank() and **a + expand_limits()**
Ensure limits include values across all plots.

b + geom_curve(aes(yend = lat + 1, xend = long + 1), curvature = 1) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

a + geom_path(lineend = "butt", linejoin = "round", linemitre = 1)
x, y, alpha, color, group, linetype, size

a + geom_polygon(aes(alpha = 50)) - x, y, alpha, color, fill, group, subgroup, linetype, size

b + geom_rect(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1)) - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

a + geom_ribbon(aes(ymin = unemploy - 900, ymax = unemploy + 900)) - x, ymax, ymin, alpha, color, fill, group, linetype, size

LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

b + geom_abline(aes(intercept = 0, slope = 1))
b + geom_hline(aes(yintercept = lat))
b + geom_vline(aes(xintercept = long))

b + geom_segment(aes(yend = lat + 1, xend = long + 1))
b + geom_spoke(aes(angle = 1:1155, radius = 1))

ONE VARIABLE continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + geom_area(stat = "bin")
x, y, alpha, color, fill, linetype, size

c + geom_density(kernel = "gaussian")
x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot()
x, y, alpha, color, fill

c + geom_freqpoly()
x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5)
x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy))
x, y, alpha, color, fill, linetype, size, weight

discrete

d <- ggplot(mpg, aes(fl))

d + geom_bar()
x, alpha, color, fill, linetype, size, weight

TWO VARIABLES

both continuous

e <- ggplot(mpg, aes(cty, hwy))

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_point()
x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile()
x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl")
x, y, alpha, color, linetype, size

e + geom_smooth(method = lm)
x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

one discrete, one continuous

f <- ggplot(mpg, aes(class, hwy))

f + geom_col()
x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot()
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = "center")
x, y, alpha, color, fill, group

f + geom_violin(scale = "area")
x, y, alpha, color, fill, group, linetype, size, weight

both discrete

g <- ggplot(diamonds, aes(cut, color))

g + geom_count()
x, y, alpha, color, fill, shape, size, stroke

e + geom_jitter(height = 2, width = 2)
x, y, alpha, color, fill, shape, size

continuous bivariate distribution

h <- ggplot(diamonds, aes(carat, price))

h + geom_bin2d(binwidth = c(0.25, 500))
x, y, alpha, color, fill, linetype, size, weight

h + geom_density_2d()
x, y, alpha, color, group, linetype, size

h + geom_hex()
x, y, alpha, color, fill, size

continuous function

i <- ggplot(economics, aes(date, unemploy))

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv")
x, y, alpha, color, group, linetype, size

visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))

j + geom_crossbar(fatten = 2) - x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j + geom_errorbar() - x, ymax, ymin, alpha, color, group, linetype, size, width
Also **geom_errorbarh()**.

j + geom_linerange()
x, ymin, ymax, alpha, color, group, linetype, size

j + geom_pointrange() - x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

maps

data <- data.frame(murder = USArrests$Murder,
state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))

k + geom_map(aes(map_id = state), map = map)
+ expand_limits(x = map$long, y = map$lat)
map_id, alpha, color, fill, linetype, size

THREE VARIABLES

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2)); l <- ggplot(seals, aes(long, lat))

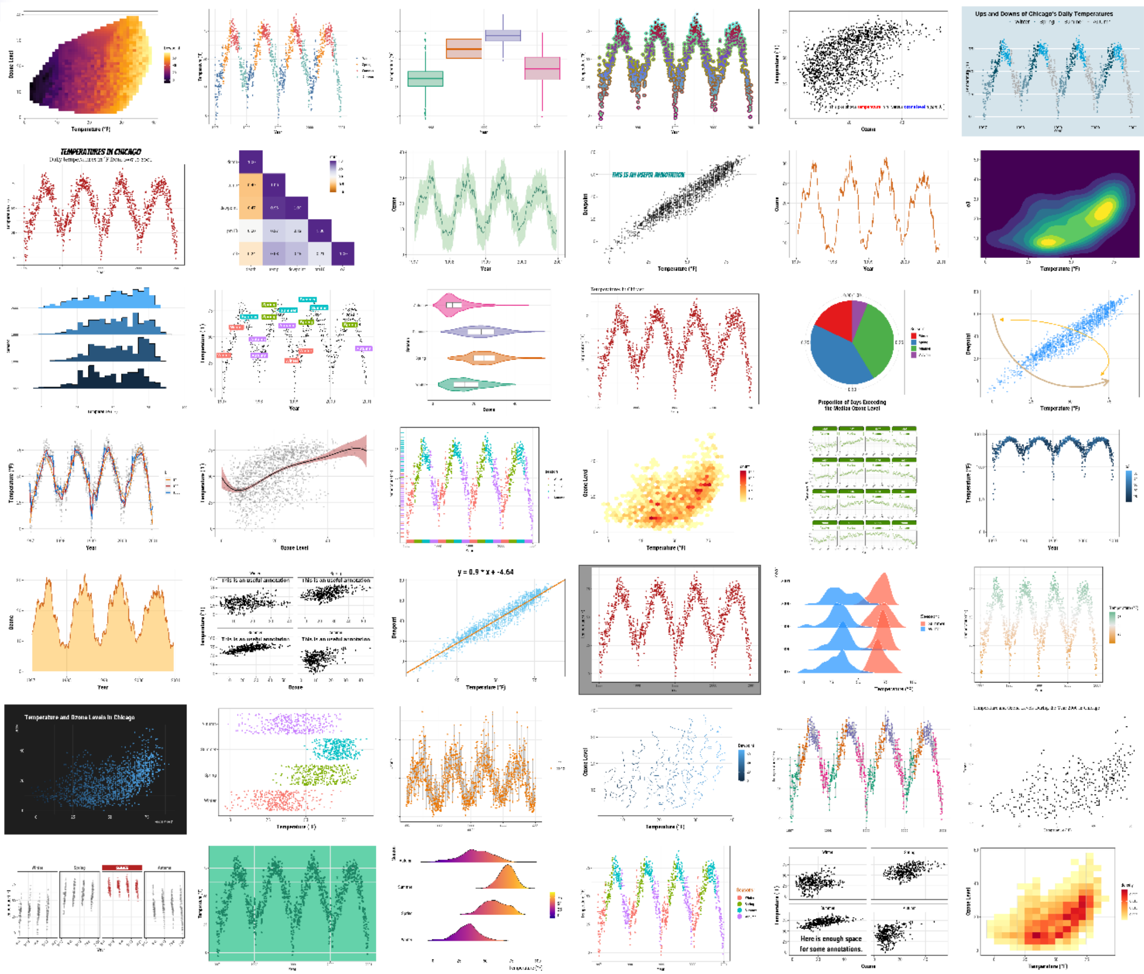
l + geom_contour(aes(z = z))
x, y, z, alpha, color, group, linetype, size, weight

l + geom_contour_filled(aes(fill = z))
x, y, alpha, color, fill, group, linetype, size, subgroup

l + geom_raster(aes(fill = z), hjust = 0.5, vjust = 0.5, interpolate = FALSE)
x, y, alpha, fill

l + geom_tile(aes(fill = z))
x, y, alpha, color, fill, linetype, size, width

ggplot2 examples



https://www.cedricscherer.com/2019/08/05/a-ggplot2-tutorial-for-beautiful-plotting-in-r/

Summary

- 3.1 Introduction to R and RStudio
- 3.2 Exploratory Data Analysis using R
  - 3.2.1 Visualization before Analysis
  - 3.2.2 Dirty Data
  - 3.2.3 Visualizing a Single Variable
  - 3.2.4 Examining Multiple Variables
  - 3.2.5 Data Exploration versus Presentation
- Exercises available in the textbook*

***Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data**
1st Edition by [EMC Education Services]

END

3.3 Statistical Methods for Evaluation

Statistics helps answer data analytics questions

- Model Building
  - What are the best input variables for the model?
  - Can the model predict the outcome given the input?
- Model Evaluation
  - Is the model accurate?
  - Does the model perform better than an obvious guess?
  - Does the model perform better than other models?
- Model Deployment
  - Is the prediction sound?
  - Does model have the desired effect (e.g., reducing cost)?

3.3 Statistical Methods for Evaluation Subsections

- 3.3.1 Hypothesis Testing
- 3.3.2 Difference of Means
- 3.3.3 Wilcoxon Rank-Sum Test
- 3.3.4 Type I and Type II Errors
- 3.3.5 Power and Sample Size
- 3.3.6 ANOVA (Analysis of Variance)

3.3.1 Hypothesis Testing

- Basic concept is to form an assertion and test it with data
- Common assumption is that there is no difference between samples (default assumption)
- Statisticians refer to this as the *null hypothesis* (H_0)
- The *alternative hypothesis* (H_A) is that there is a difference between samples

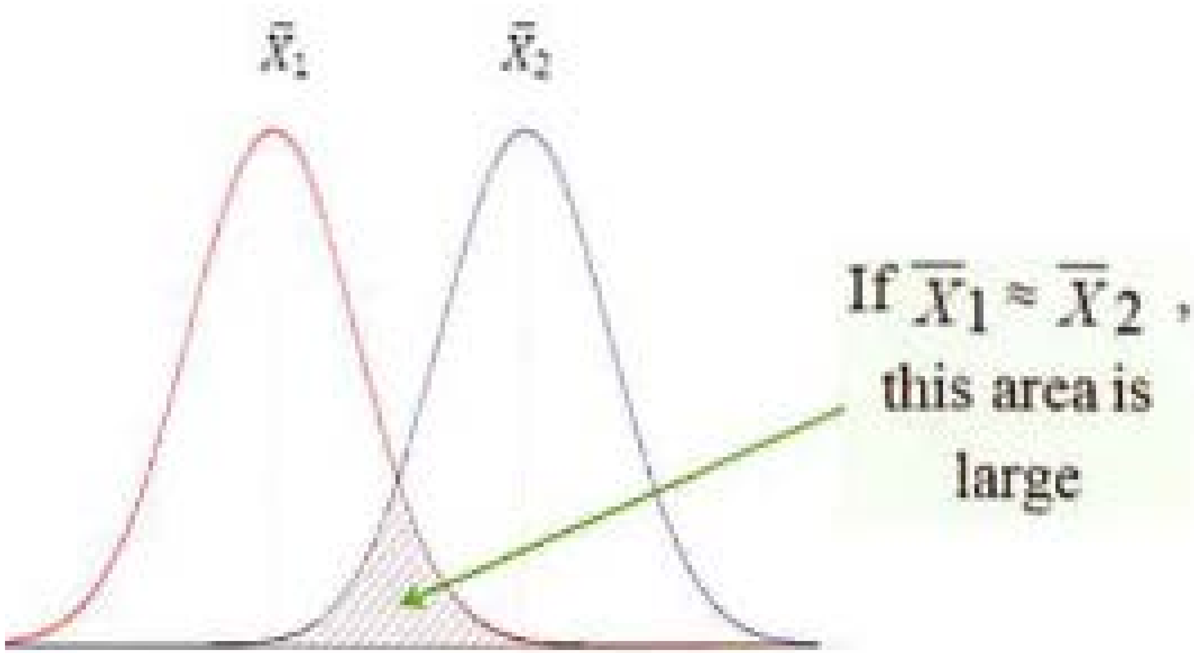
3.3.1 Hypothesis Testing

Example Null and Alternative Hypotheses

Application	Null Hypothesis	Alternative Hypothesis
Accuracy Forecast	Model X *does not predict* better than the existing model.	Model X *predicts* better than the existing model.
Recommendation Engine	Algorithm Y *does not produce* better recommendations than the current algorithm being used.	Algorithm Y *produces* better recommendations than the current algorithm being used.
Regression Modeling	This variable *does not affect* the outcome because its coefficient is *zero*.	This variable *affects* outcome because its coefficient is not *zero*.

3.3.2 Difference of Means

Two populations – same or different?



3.3.2 Difference of Means
Two Parametric Methods

- Student’s t-test
  - Assumes two normally distributed populations, and that they have equal variance
- Welch’s t-test
  - Assumes two normally distributed populations, and they don’t necessarily have equal variance

3.3.3 Wilcoxon Rank-Sum Test
A Nonparametric Method

- Makes no assumptions about the underlying probability distributions

3.3.4 Type I and Type II Errors

- An hypothesis test may result in two types of errors
  - Type I error – rejection of the null hypothesis when the null hypothesis is TRUE
  - Type II error – acceptance of the null hypothesis when the null hypothesis is FALSE

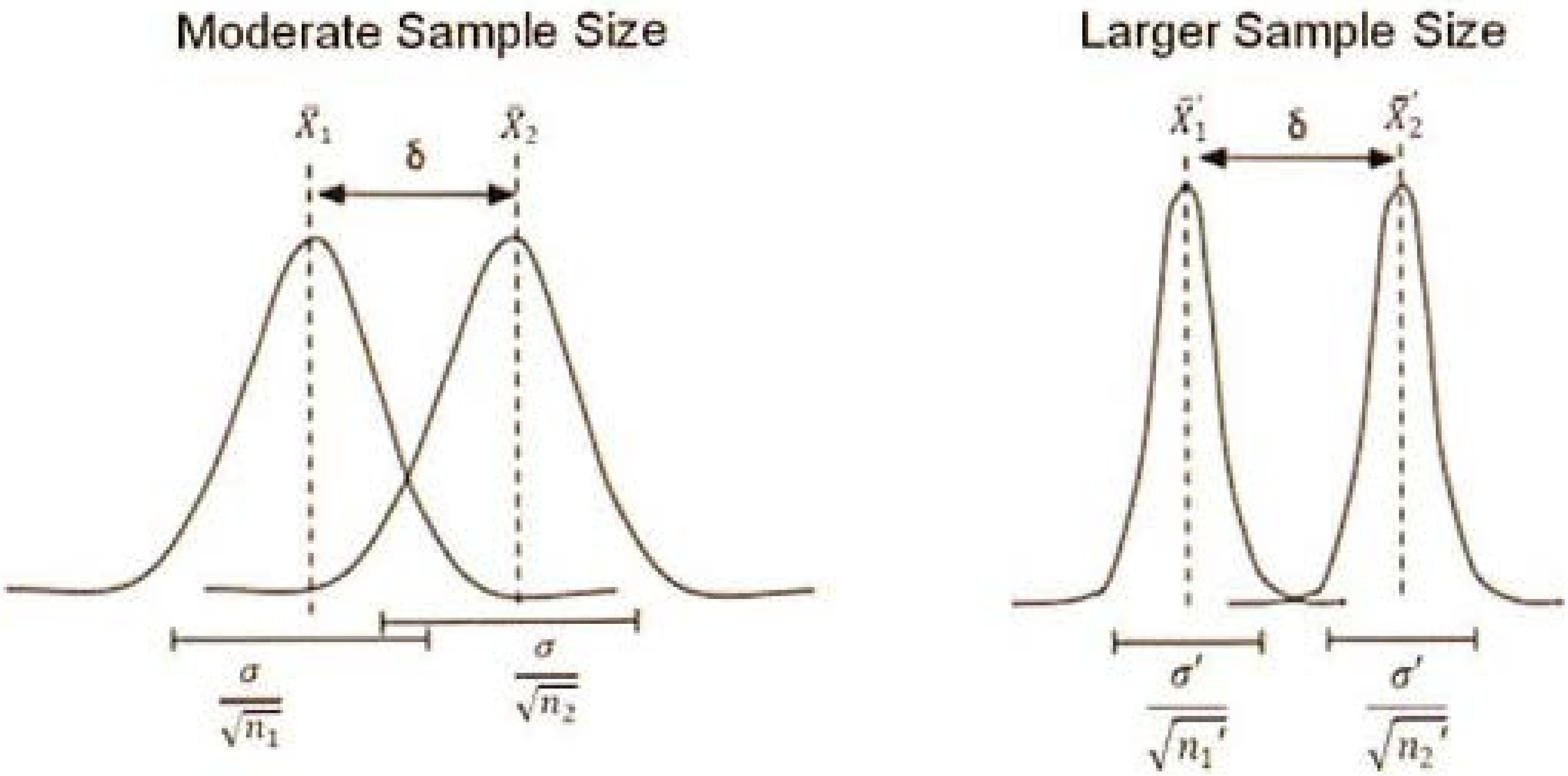
3.3.4 Type I and Type II Errors

	H_0 is true	H_0 is false
H_0 is accepted	Correct outcome	*Type II Error*
H_0 is rejected	*Type I error*	Correct outcome

3.3.5 Power and Sample Size

- The *power of a test* is the probability of correctly rejecting the null hypothesis
- The power of a test increases as the sample size increases
- *Effect size* δ = difference between the means
- It is important to consider an appropriate effect size for the problem at hand

3.3.5 Power and Sample Size



3.3.6 ANOVA (Analysis of Variance)

- A generalization of the hypothesis testing of the difference of two population means
- Good for analyzing more than two populations
- ANOVA tests if any of the population means differ from the other population means