CHAPTER 5

Ms. Tina Maru
Assistant Professor
Artificial Intelligence and Data Science Department

TOPICS

Exploratory Data Analysis:

Visualization before analysis,

Dirty Data,

Visualizing single variable,

Examining Multiple variable,

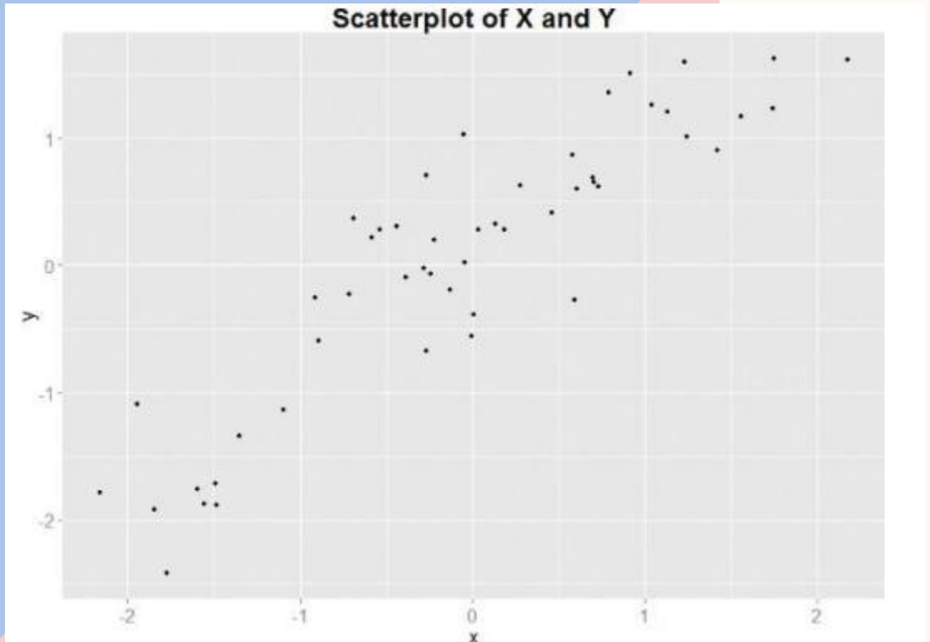
Data Exploration versus presentation.

EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations.

EDA

- A useful way to detect patterns and anomalies in the data is through the exploratory data analysis with visualization.
- Visualization gives a succinct, holistic view of the data that may be difficult to grasp from the numbers and summaries alone.
- Variables x and y of the data frame data can instead be visualized in a scatterplot, which easily depicts the relationship between two variables.

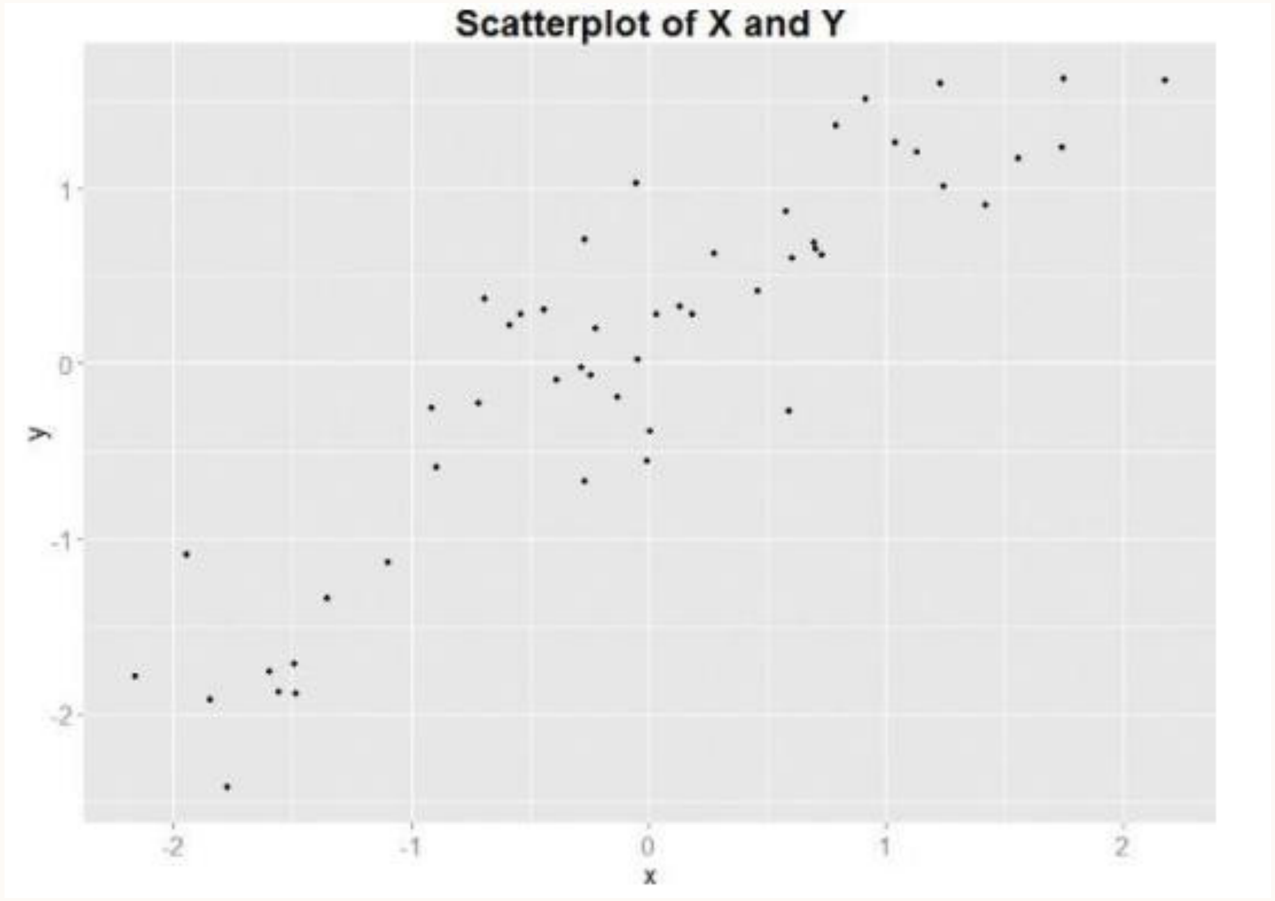


- An important facet of the initial data exploration, visualization assesses data cleanliness and suggests potentially important relationships in the data prior to the model planning and building phases.

SCATTERPLOT CODE

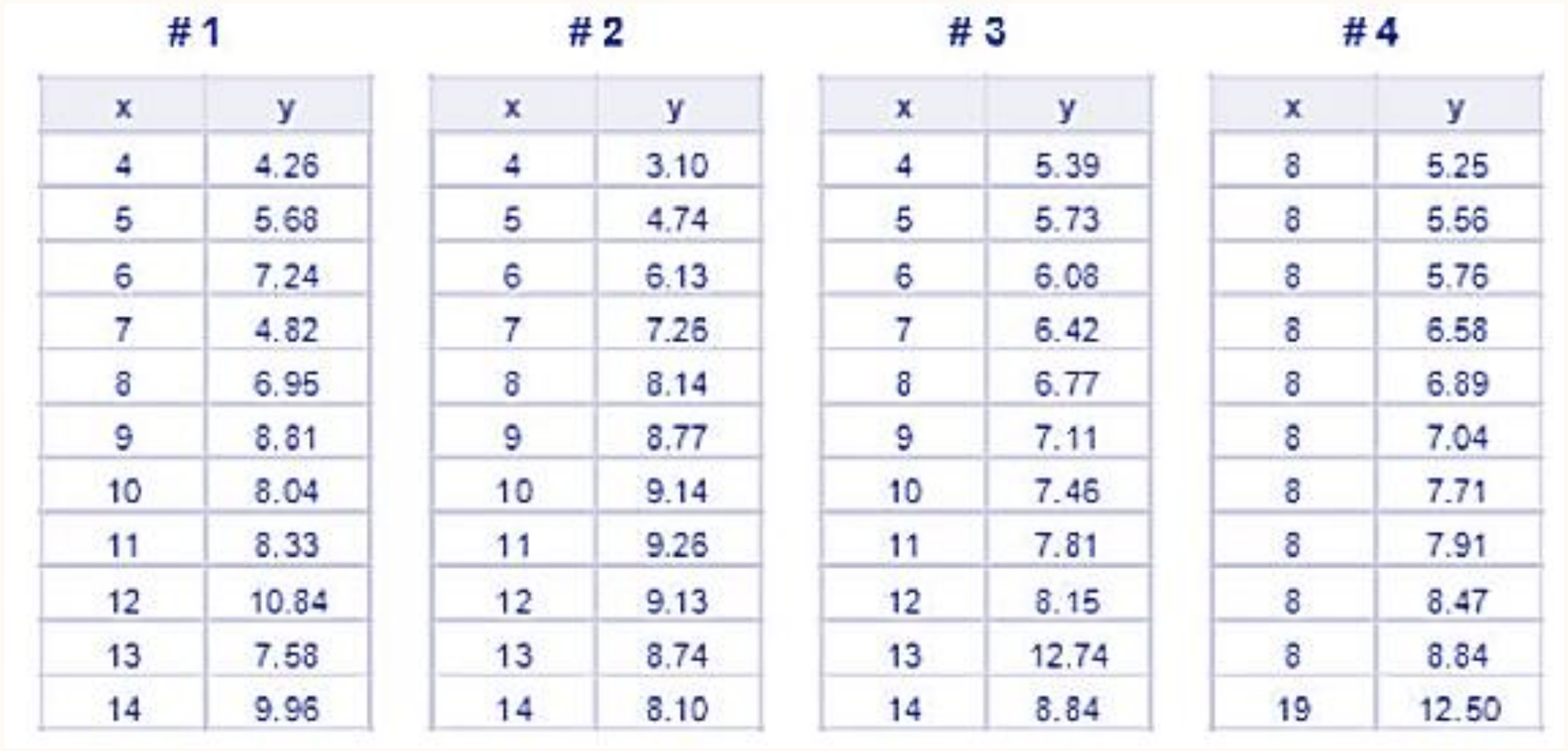
- The code to generate data as well as Figure is shown next.

```
x <- rnorm(50)
y <- x + rnorm(50, mean=0, sd=0.5)
data <- as.data.frame(cbind(x, y))
summary(data)
library(ggplot2)
ggplot(data, aes(x=x, y=y)) +
geom_point(size=2) +
ggtitle("Scatterplot of X and Y") +
theme(axis.text=element_text(size=12),
axis.title = element_text(size=14),
plot.title = element_text(size=20, face="bold"))
```



VISUALIZATION BEFORE ANALYSIS

- Consider Anscombe’s quartet. Anscombe’s quartet consists of four datasets, as shown in Figure.
- It was constructed by statistician Francis Anscombe in 1973 to demonstrate the importance of graphs in statistical analyses.



1

x	y
4	4.26
5	5.68
6	7.24
7	4.82
8	6.95
9	8.81
10	8.04
11	8.33
12	10.84
13	7.58
14	9.96

2

x	y
4	3.10
5	4.74
6	6.13
7	7.26
8	8.14
9	8.77
10	9.14
11	9.26
12	9.13
13	8.74
14	8.10

3

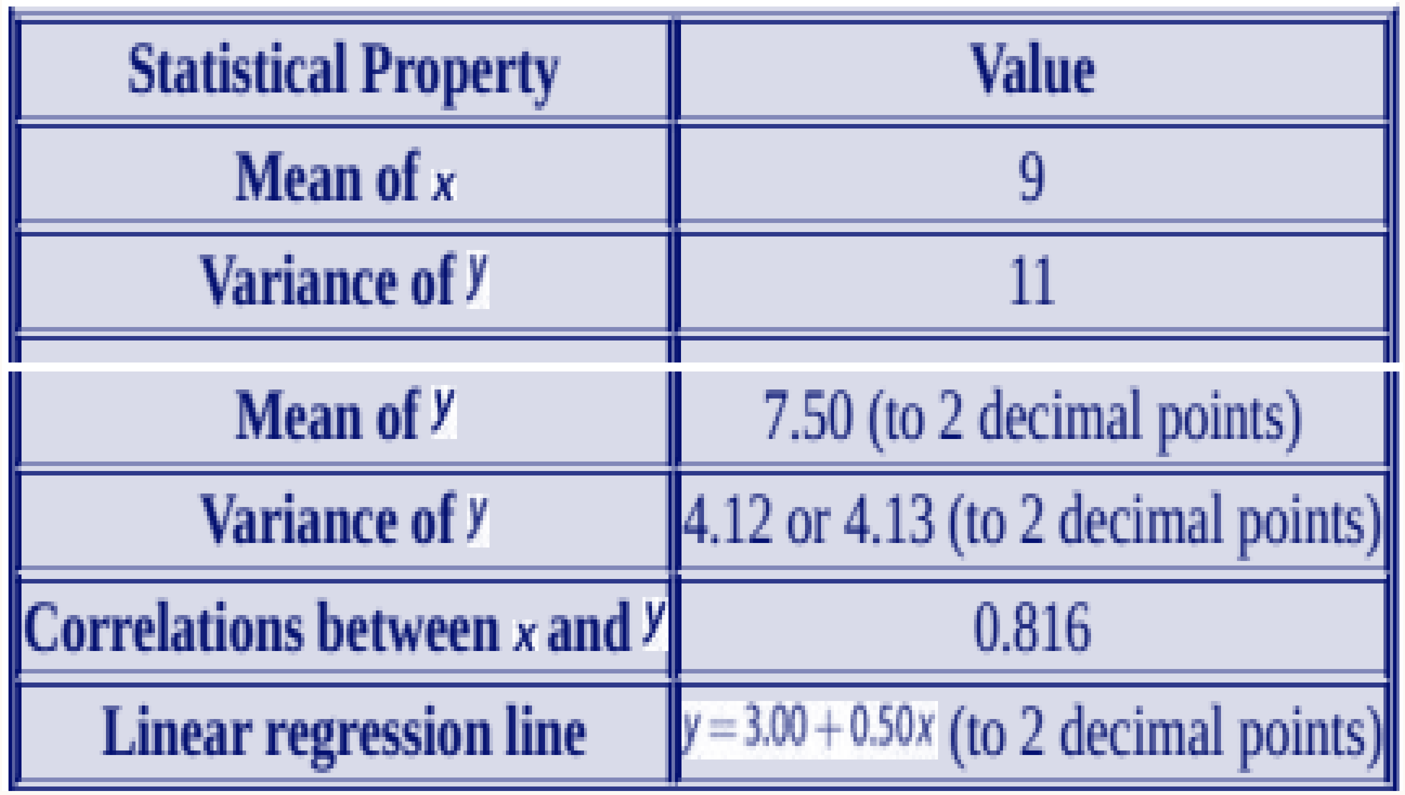
x	y
4	5.39
5	5.73
6	6.08
7	6.42
8	6.77
9	7.11
10	7.46
11	7.81
12	8.15
13	12.74
14	8.84

4

x	y
8	5.25
8	5.56
8	5.76
8	6.58
8	6.89
8	7.04
8	7.71
8	7.91
8	8.47
8	8.84
19	12.50

VISUALIZATION BEFORE ANALYSIS

- The four datasets in Anscombe’s quartet have nearly identical statistical properties, as shown.

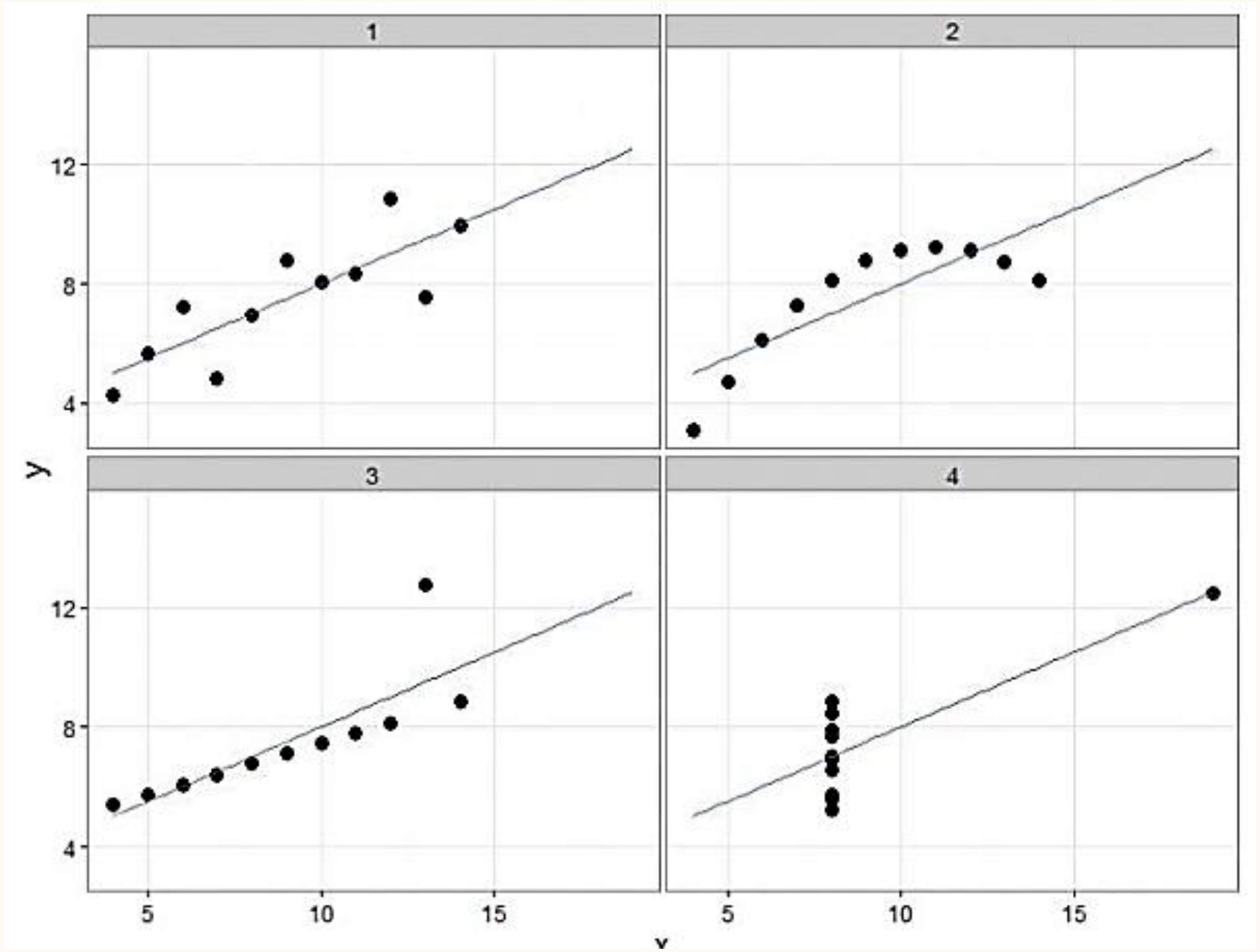


Statistical Property	Value
Mean of x	9
Variance of y	11
Mean of y	7.50 (to 2 decimal points)
Variance of y	4.12 or 4.13 (to 2 decimal points)
Correlations between x and y	0.816
Linear regression line	$y = 3.00 + 0.50x$ (to 2 decimal points)

- Based on the nearly identical statistical properties across each dataset, one might conclude that these four datasets are quite similar.

VISUALIZATION BEFORE ANALYSIS

- However, the scatterplots will tell a different story. Each dataset is plotted as a scatterplot, and the fitted lines are the result of applying linear regression models.



VISUALIZATION BEFORE ANALYSIS

- The code to generate scatterplot will require the R package ggplot2, which can be installed simply by running the command

install.packages(“ggplot2”).

- The anscombe dataset for the plot is included in the standard R distribution.
- Enter data() for a list of datasets included in the R base distribution.
- Enter data(DatasetName) to make a dataset available in the current workspace.
- Variable levels is created using the gl() function, which generates factors of four levels (1, 2, 3, and 4), each repeating 11 times. Variable mydata is created using the with(data, expression) function, which evaluates an expression in an environment constructed from data.
- In this example, the data is the anscombe dataset, which includes eight attributes: x1, x2, x3, x4, y1, y2, y3, and y4.
- The expression part in the code creates a data frame from the anscombe dataset, and it only includes three attributes: x, y, and the group each data point belongs to (mygroup).

VISUALIZATION BEFORE ANALYSIS

```
install.packages("ggplot2") # not required if package has been installed
data(anscombe) # load the anscombe dataset into the current workspace
anscombe
x1 x2 x3 x4 y1 y2 y3 y4
1 10 10 10 8 8.04 9.14 7.46 6.58
2 8 8 8 8 6.95 8.14 6.77 5.76
3 13 13 13 8 7.58 8.74 12.74 7.71
4 9 9 9 8 8.81 8.77 7.11 8.84
5 11 11 11 8 8.33 9.26 7.81 8.47
6 14 14 14 8 9.96 8.10 8.84 7.04
7 6 6 6 8 7.24 6.13 6.08 5.25
8 4 4 4 19 4.26 3.10 5.39 12.50
9 12 12 12 8 10.84 9.13 8.15 5.56
10 7 7 7 8 4.82 7.26 6.42 7.91
11 5 5 5 8 5.68 4.74 5.73 6.89
```

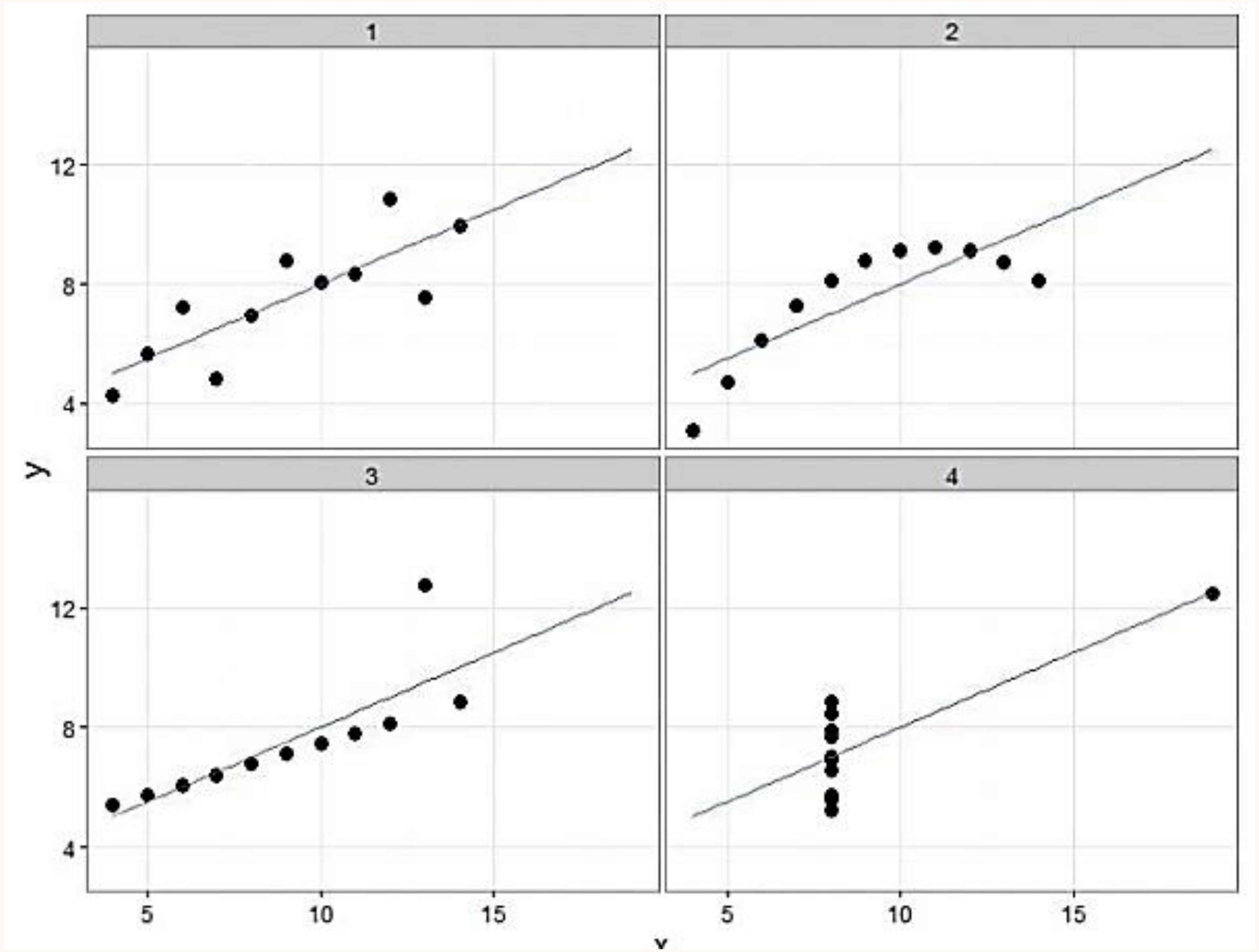
```
nrow(anscombe) # number of rows
[1] 11
# generates levels to indicate which group each data point
belongs to
levels <- gl(4, nrow(anscombe))
levels
[1] 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3
[34] 4 4 4 4 4 4 4 4 4 4 4
Levels: 1 2 3 4
# Group anscombe into a data frame
mydata <- with(anscombe, data.frame(x=c(x1,x2,x3,x4),
y=c(y1,y2,y3,y4),
mygroup=levels))
```

VISUALIZATION BEFORE ANALYSIS

```
mydata
x y mygroup
1 10 8.04 1
2 8 6.95 1
3 13 7.58 1
4 9 8.81 1
…4
1 19 12.50 4
42 8 5.56 4
43 8 7.91 4
44 8 6.89 4
```

```
# Make scatterplots using the ggplot2 package
library(ggplot2)
theme_set(theme_bw()) # set plot color theme
# create the four plots of Figure 3.7
ggplot(mydata, aes(x,y)) +
geom_point(size=4) +
geom_smooth(method="lm", fill=NA, fullrange=TRUE) +
facet_wrap(~mygroup)
```

VISUALIZATION BEFORE ANALYSIS

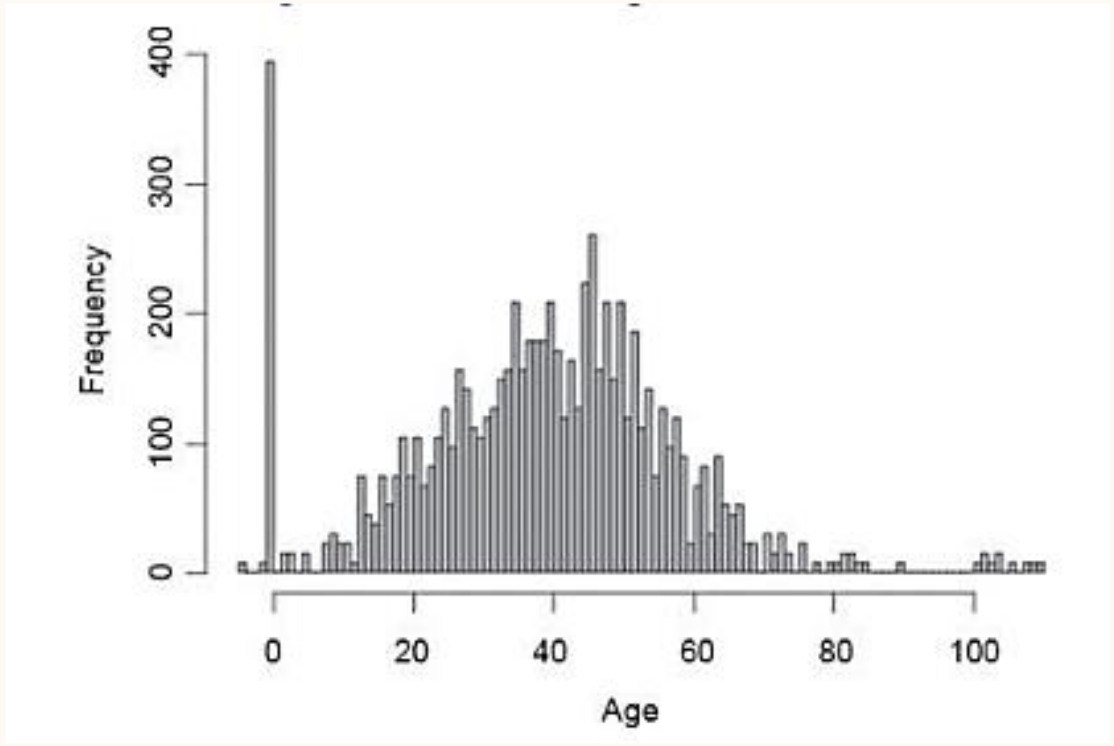


DIRTY DATA

- Dirty data can be detected in the data exploration phase with visualizations.
- In general, analysts should look for anomalies, verify the data with domain knowledge, and decide the most appropriate approach to clean the data.
- Consider a scenario in which a bank is conducting data analyses of its account holders to gauge customer retention. Figure shows the age distribution of the account holders.

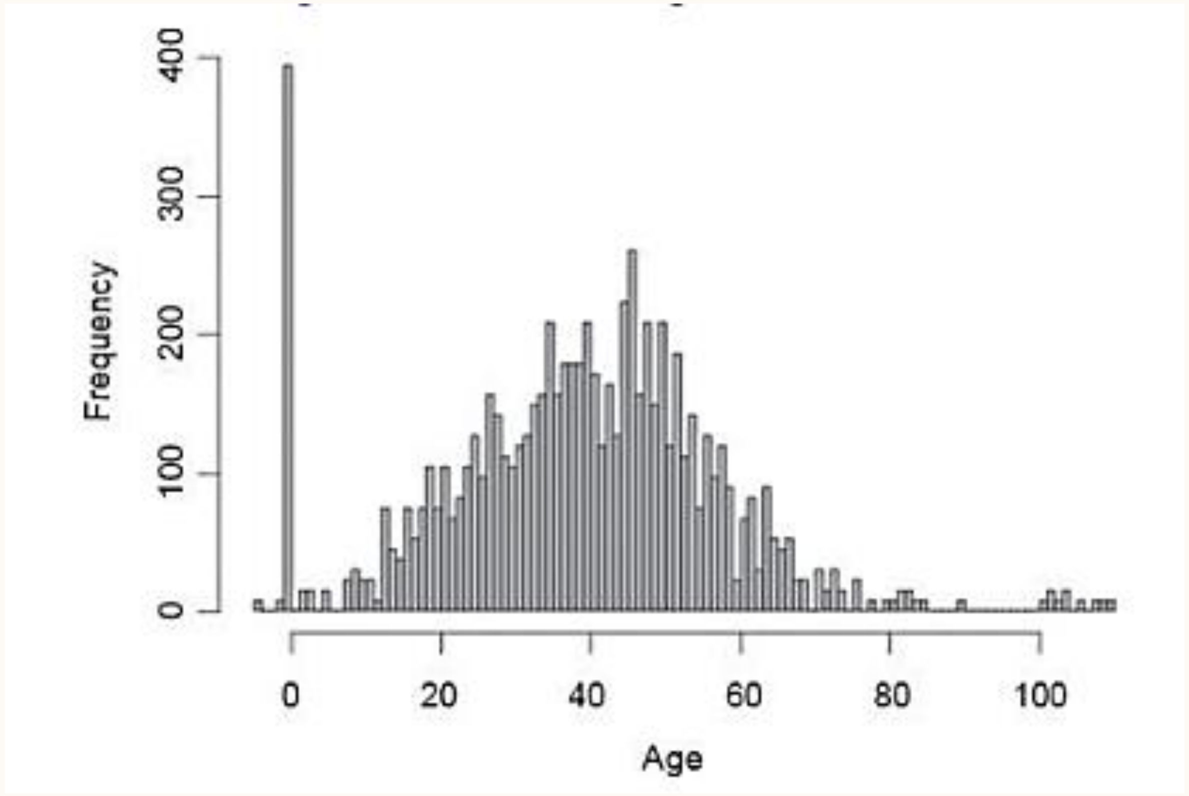
- If the age data is in a vector called age, the graph can be created with the following R script:

```
hist(age, breaks=100, main="Age Distribution of
Account Holders",
xlab="Age", ylab="Frequency", col="gray")
```



DIRTY DATA

- The figure shows that the median age of the account holders is around 40.
- A few accounts with account holder age less than 10 are unusual but plausible. These could be custodial accounts or college savings accounts set up by the parents of young children. These accounts should be retained for future analyses.
- However, the left side of the graph shows a huge spike of customers who are zero years old or have negative ages. This is likely to be evidence of **missing data**.
- The null age values could have been replaced by 0 or negative values during the data input. Such an occurrence may be caused by entering age in a text box that only allows numbers and does not accept empty values.
- Or by transferring data among several systems that have different definitions for null values (such as NULL, NA, 0, –1, or –2).



DIRTY DATA

- **Data cleansing** needs to be performed over the accounts with abnormal age values. Analysts should take a closer look at the records to decide if the missing data should be eliminated or if an appropriate age value can be determined using other available information for each of the accounts.
- In R, the is.na() function provides tests for missing values. The following example creates a vector x where the fourth value is not available (NA). The is.na() function returns TRUE at each NA value and FALSE otherwise.

```
x <- c(1, 2, 3, NA, 4)
is.na(x)
[1] FALSE FALSE FALSE TRUE FALSE
```

- Some arithmetic functions, such as mean(), applied to data containing missing values can yield an NA result. To prevent this, set the na.rm parameter to TRUE to remove the missing value during the function's execution.

```
mean(x)                 [1] NA
mean(x, na.rm=TRUE)     [1] 2.5
```

- The na.exclude() function returns the object with incomplete cases removed.

DIRTY DATA

- The na.exclude() function returns the object with incomplete cases removed.

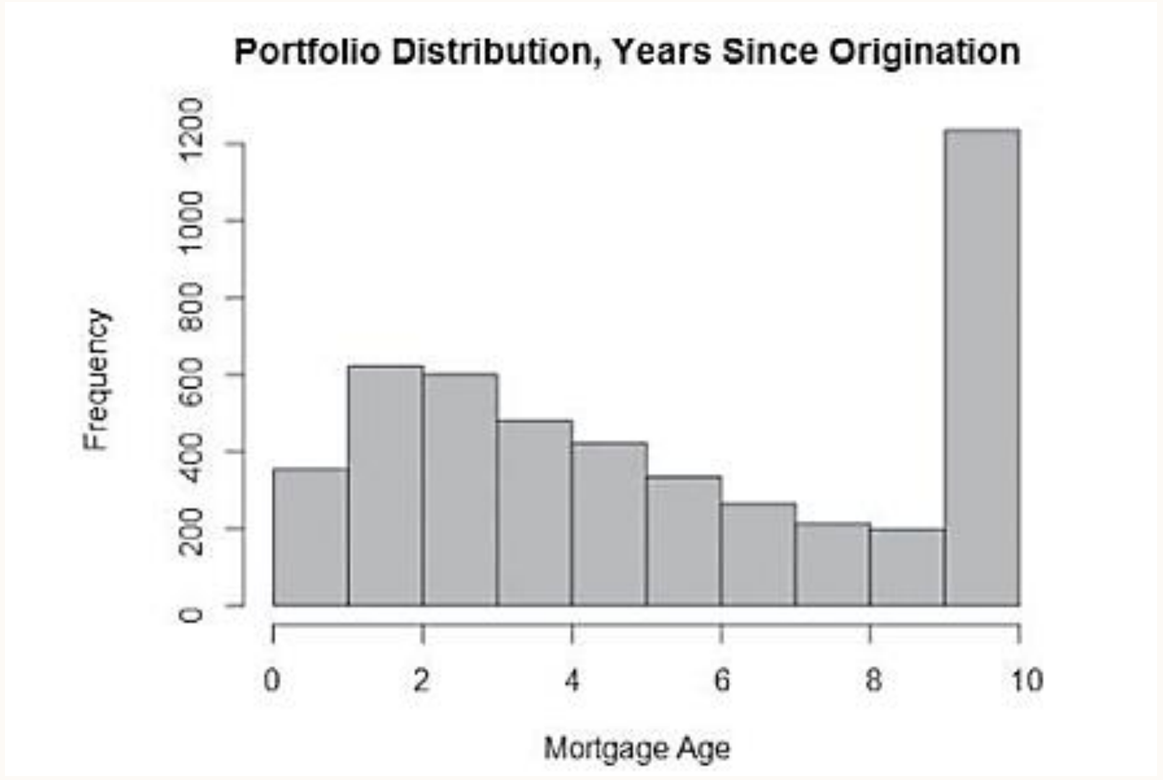
```
DF <- data.frame(x = c(1, 2, 3), y = c(10, 20, NA))
DF
x y
1 1 10
2 2 20
3 3 NA
DF1 <- na.exclude(DF)
DF1
x y
1 1 10
2 2 20
```

DIRTY DATA

- Account holders older than 100 may be due to bad data caused by typos.
- Another possibility is that these accounts may have been passed down to the heirs of the original account holders without being updated. In this case, one needs to further examine the data and conduct data cleansing if necessary.
- The dirty data could be simply removed or filtered out with an age threshold for future analyses.
- If removing records is not an option, the analysts can look for patterns within the data and develop a set of heuristics to attack the problem of dirty data.
- For example, wrong age values could be replaced with **approximation based on the nearest neighbor**—the record that is the most similar to the record in question based on analyzing the differences in all the other variables besides age.

DIRTY DATA

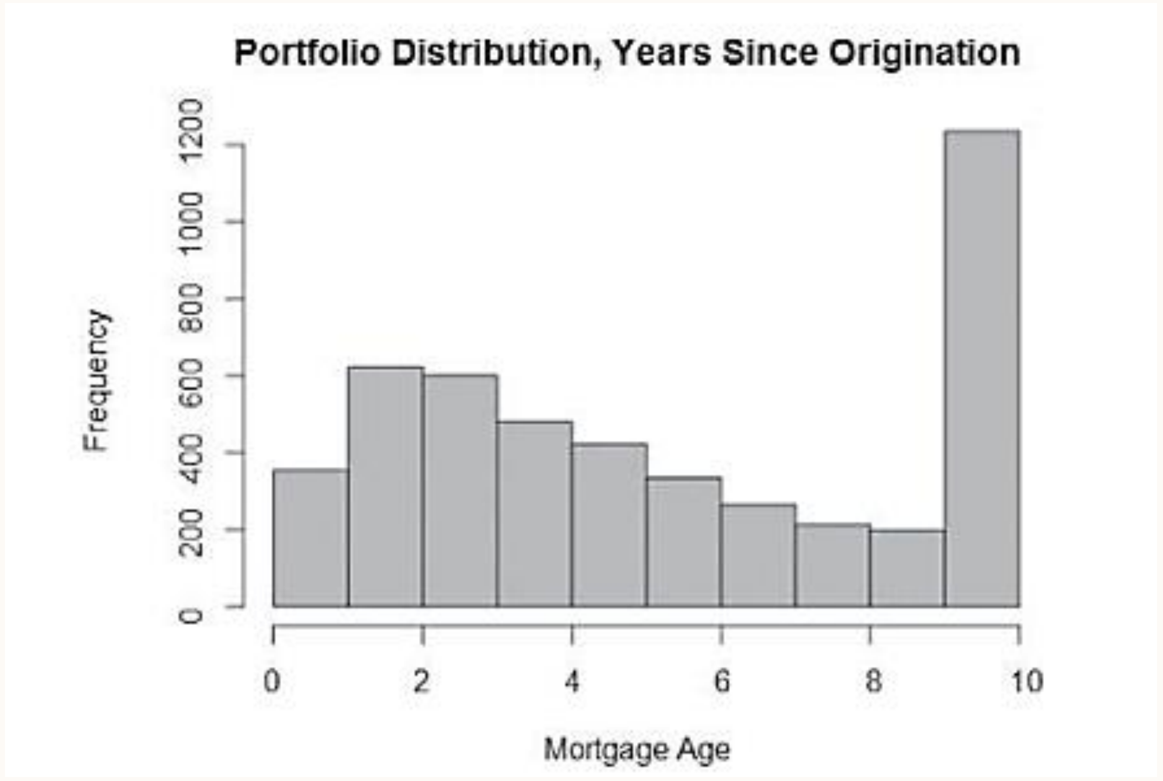
- Figure presents another example of dirty data.
- The distribution shown here corresponds to the age of mortgages in a bank's home loan portfolio. The mortgage age is calculated by subtracting the origination date of the loan from the current date. The vertical axis corresponds to the number of mortgages at each mortgage age.



DIRTY DATA

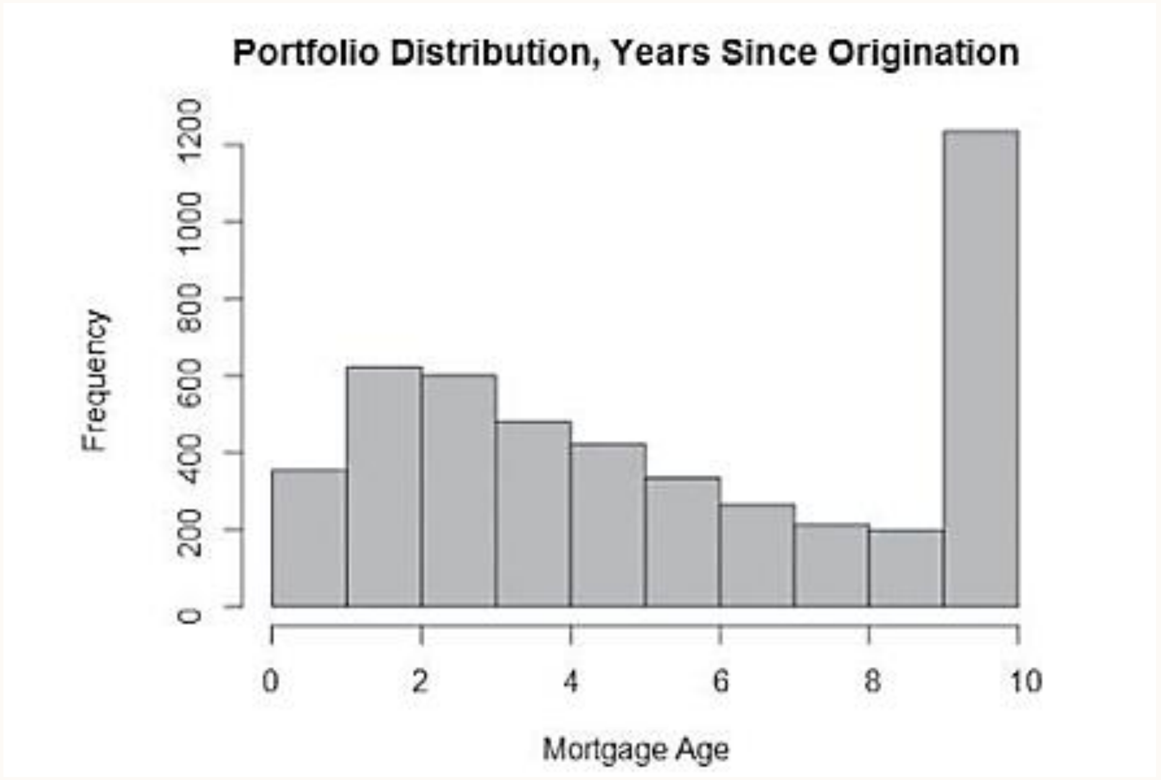
- If the data is in a vector called mortgage, Figure can be produced by the following R script.

hist(mortgage, breaks=10, xlab="Mortgage Age", col="gray",
main="Portfolio Distribution, Years Since Origination")



DIRTY DATA

- Figure shows that the loans are no more than 10 years old, and these 10-year-old loans have a disproportionate frequency compared to the rest of the population.
- One possible explanation is that the 10-year-old loans do not only include loans originated 10 years ago, but also those originated earlier than that. In other words, the 10 in the x-axis actually means ≥ 10. This sometimes happens when data is ported from one system to another or because the data provider decided, for some reason, not to distinguish loans that are more than 10 years old.
- Analysts need to study the data further and decide the most appropriate way to perform data cleansing.

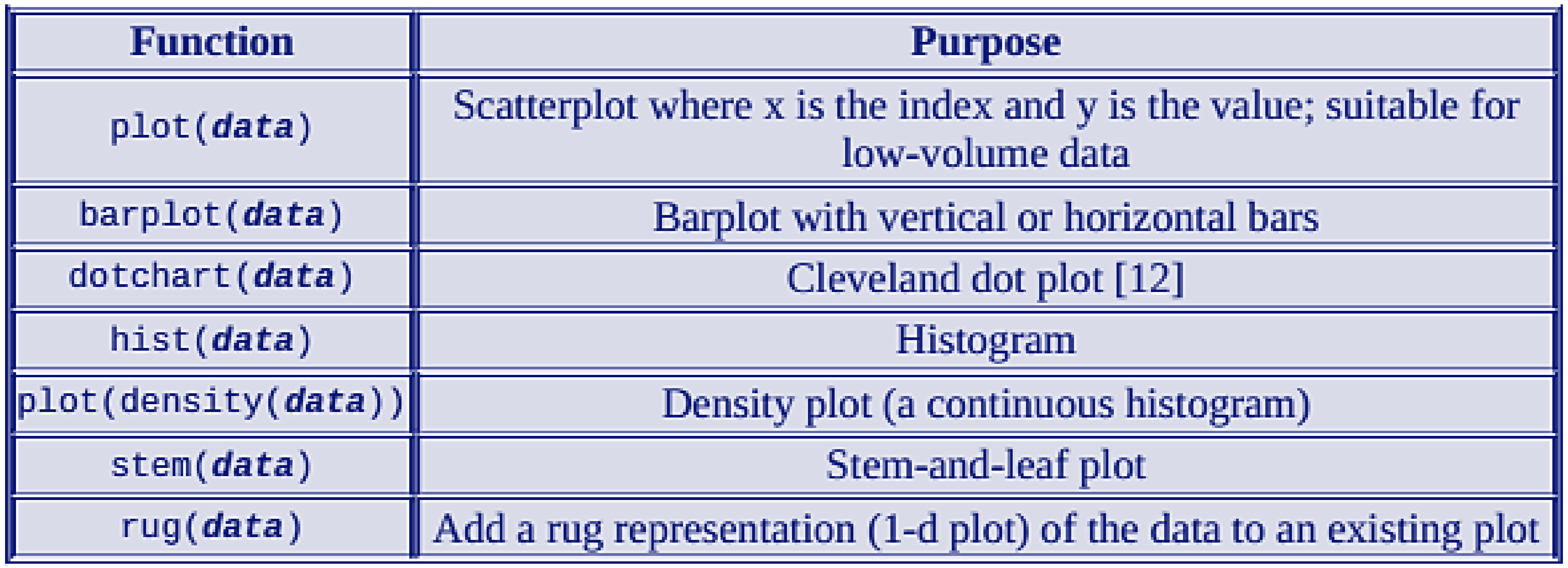


DIRTY DATA

- Data analysts should perform sanity checks against domain knowledge and decide if the dirty data needs to be eliminated.
- Consider the task to find out the probability of mortgage loan default. If the past observations suggest that most defaults occur before about the 4th year and 10-year-old mortgages rarely default, it may be safe to eliminate the dirty data and assume that the defaulted loans are less than 10 years old. For other analyses, it may become necessary to track down the source and find out the true origination dates.
- Dirty data can occur due to acts of omission.

VISUALIZING A SINGLE VARIABLE

- R has many functions available to examine a single variable. Some of these functions are listed in table.

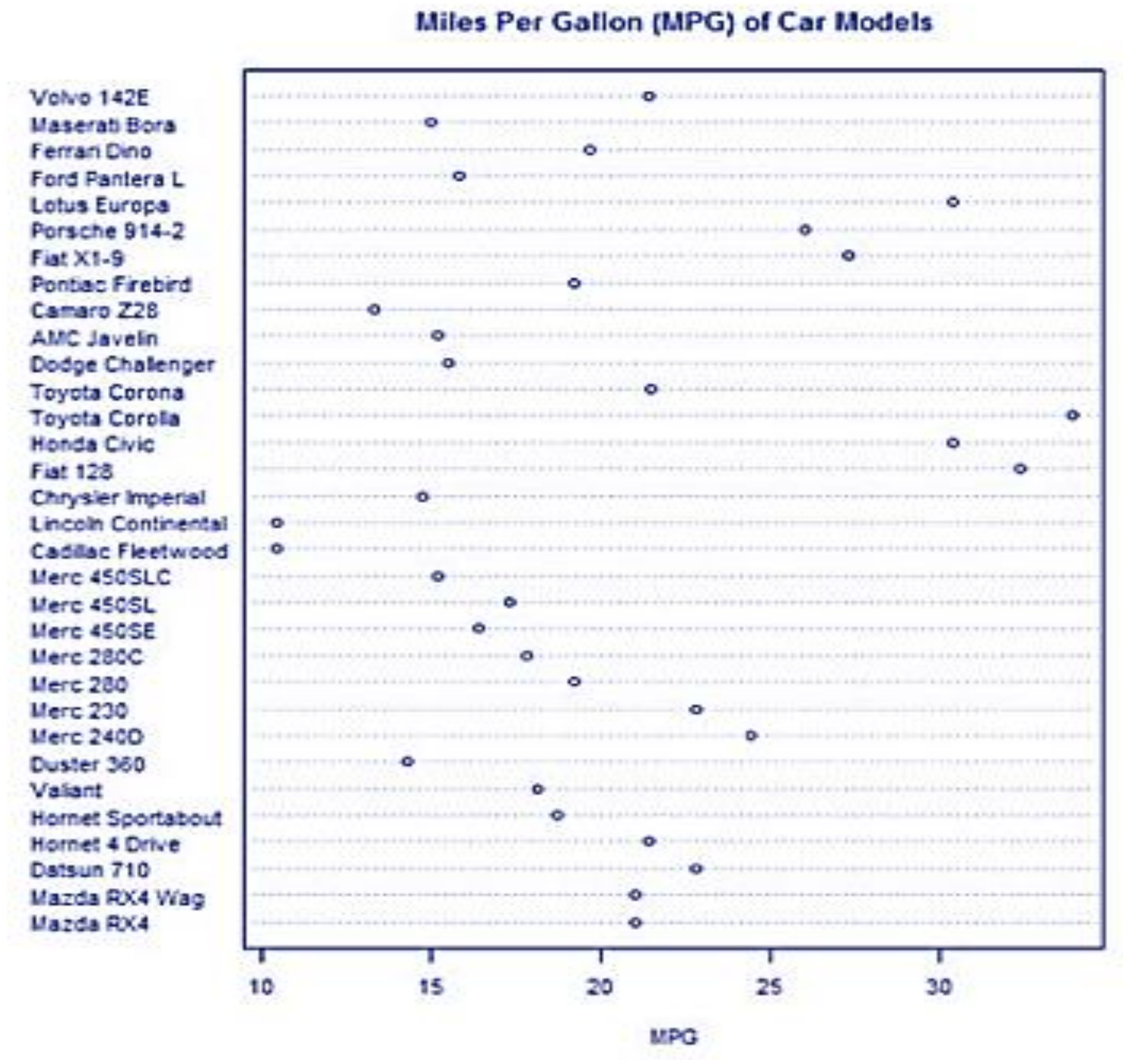


Function	Purpose
`plot(data)`	Scatterplot where x is the index and y is the value; suitable for low-volume data
`barplot(data)`	Barplot with vertical or horizontal bars
`dotchart(data)`	Cleveland dot plot [12]
`hist(data)`	Histogram
`plot(density(data))`	Density plot (a continuous histogram)
`stem(data)`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

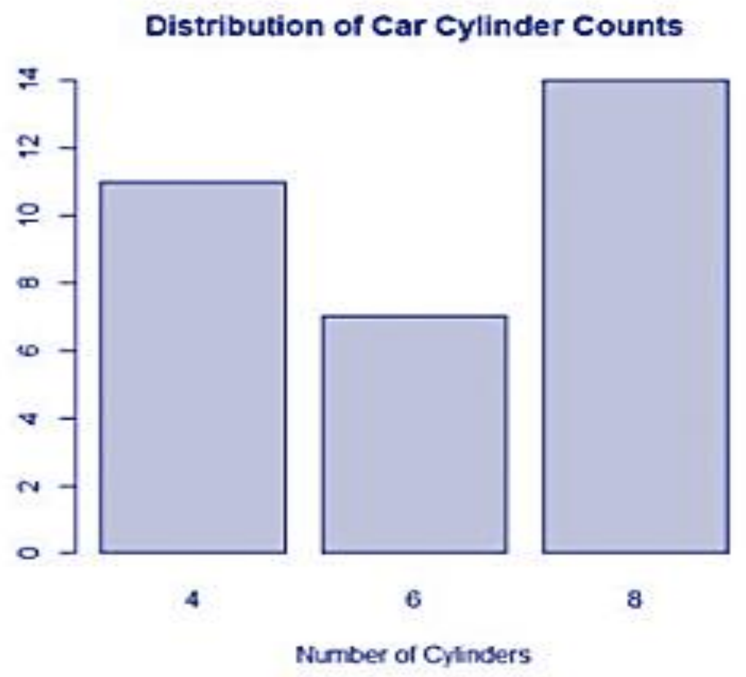
VISUALIZING A SINGLE VARIABLE

- **Dotchart and barplot** portray continuous values with labels from a discrete variable.
- A dotchart can be created in R with the function dotchart(x, label=…), where x is a numeric vector and label is a vector of categorical labels for x.
- A barplot can be created with the barplot(height) function, where height represents a vector or matrix.
- Figure shows (a) a dotchart and (b) a barplot based on the mtcars dataset, which includes the fuel consumption and 10 aspects of automobile design and performance of 32 automobiles. This dataset comes with the standard R distribution.

VISUALIZING A SINGLE VARIABLE



(a)



(b)

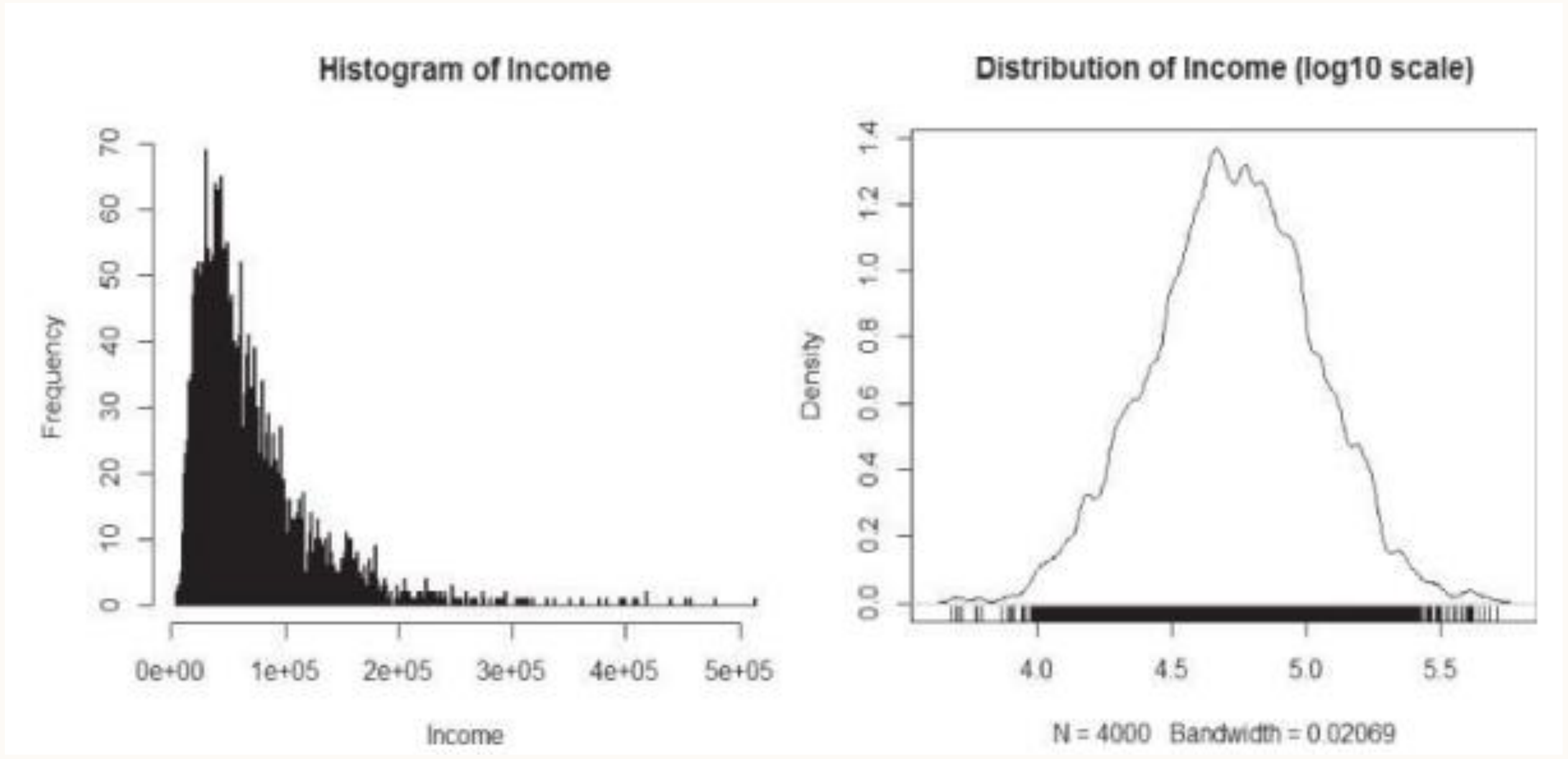
VISUALIZING A SINGLE VARIABLE

- The plots in Figure can be produced with the following R code.

```
data(mtcars)
dotchart(mtcars$mpg,labels=row.names(mtcars),cex=.7,
main="Miles Per Gallon (MPG) of Car Models",
xlab="MPG")
barplot(table(mtcars$cyl), main="Distribution of Car Cylinder Counts",
xlab="Number of Cylinders")
```

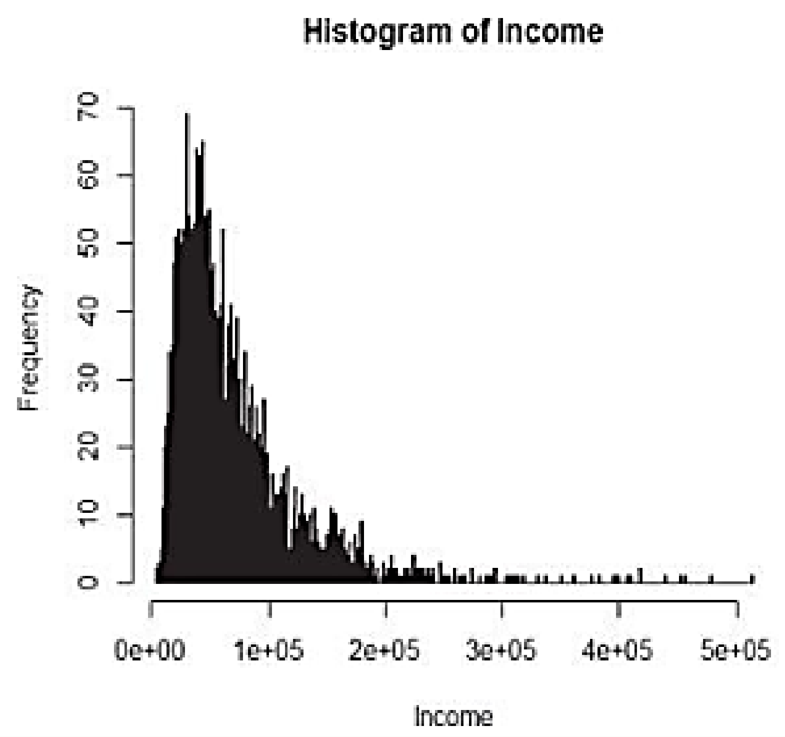
VISUALIZING A SINGLE VARIABLE

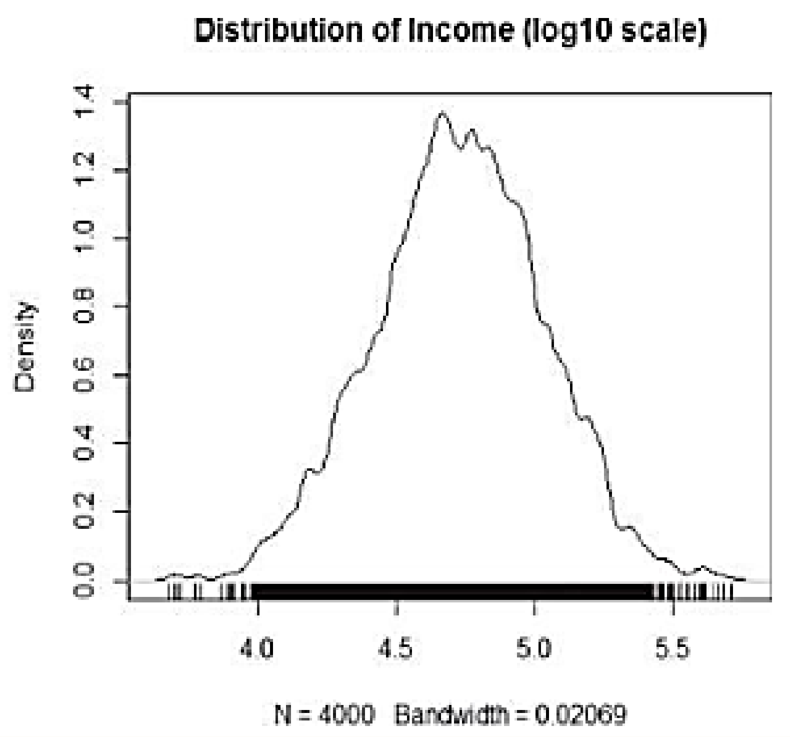
- **Histogram and Density Plot**
- Figure includes a histogram of household income.
- The histogram shows a clear concentration of low household incomes on the left and the long tail of the higher incomes on the right.



VISUALIZING A SINGLE VARIABLE

- **Histogram and Density Plot**
- Figure shows a density plot of the logarithm of household income values, which emphasizes the distribution. The income distribution is concentrated in the center portion of the graph. The rug() function creates a one-dimensional density plot on the bottom of the graph to emphasize the distribution of the observation.





VISUALIZING A SINGLE VARIABLE

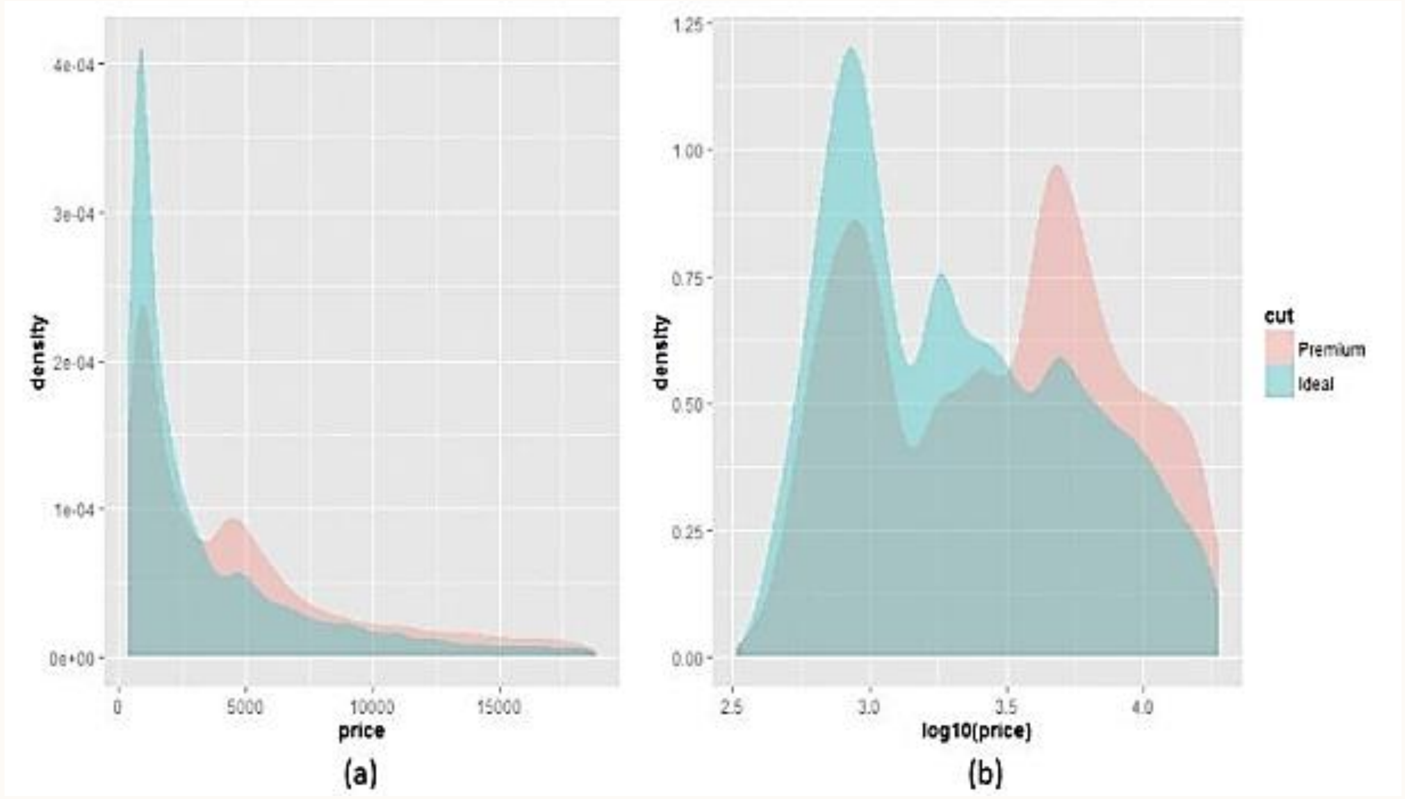
```
# randomly generate 4000 observations from the log normal distribution
income <- rlnorm(4000, meanlog = 4, sdlog = 0.7)
summary(income)
Min. 1st Qu. Median Mean 3rd Qu. Max.
4.301 33.720 54.970 70.320 88.800 659.800
income <- 1000*income
summary(income)
Min. 1st Qu. Median Mean 3rd Qu. Max.
4301 33720 54970 70320 88800 659800
# plot the histogram
hist(income, breaks=500, xlab="Income", main="Histogram of Income")
# density plot
plot(density(log10(income), adjust=0.5),
main="Distribution of Income (log10 scale)")
# add rug to the density plot
rug(log10(income))
```

VISUALIZING A SINGLE VARIABLE

- In the data preparation phase of the Data Analytics Lifecycle, the data range and distribution can be obtained.
- If the data is skewed, viewing the logarithm of the data (if it's all positive) can help detect structures that might otherwise be overlooked in a graph with a regular, nonlogarithmic scale.
- Examining if the data is unimodal or multimodal will give an idea of how many distinct populations with different behavior patterns might be mixed into the overall population.
- Many modeling techniques assume that the data follows a normal distribution.
- Therefore, it is important to know if the available dataset can match that assumption before applying any of those modeling techniques.

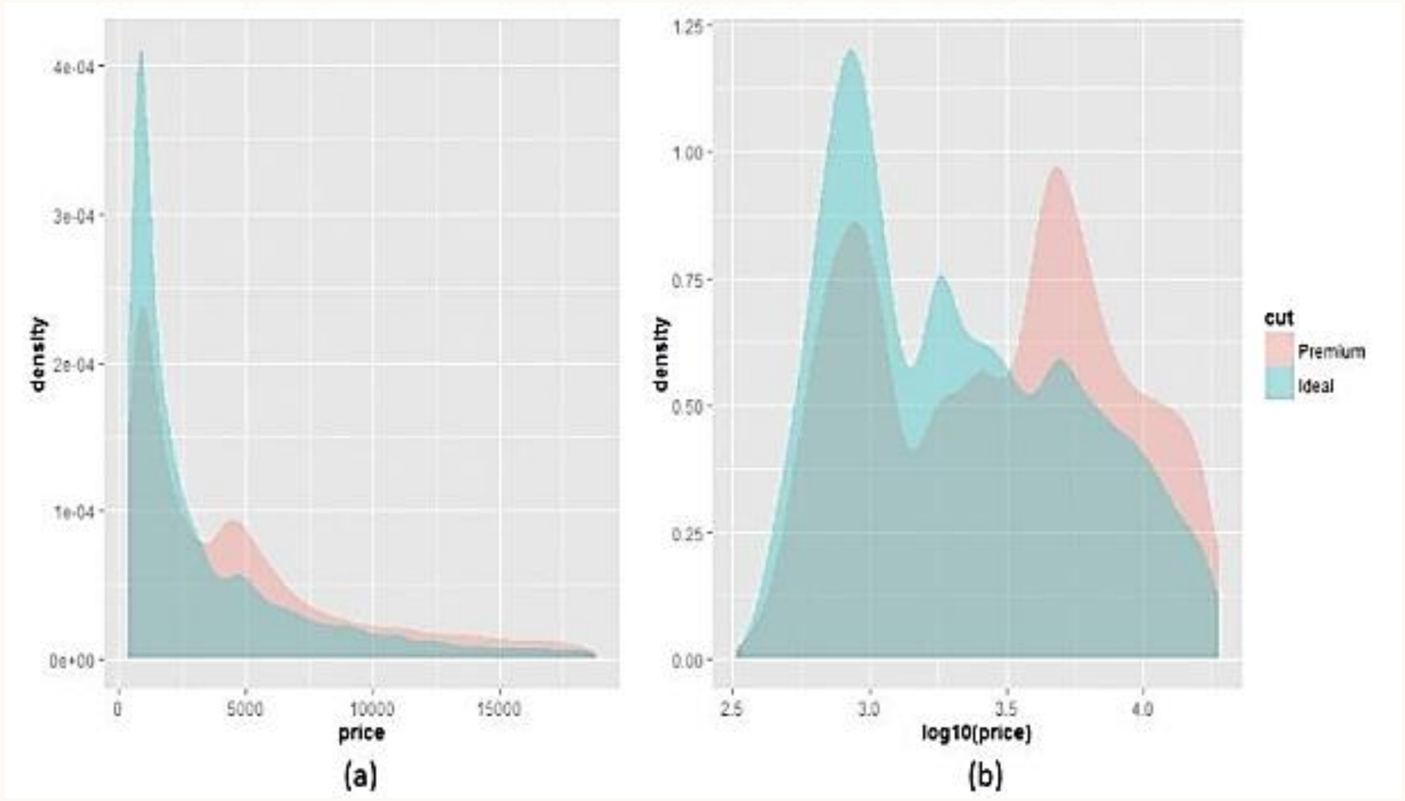
VISUALIZING A SINGLE VARIABLE

- Consider a density plot of diamond prices (in USD).
- Figure contains two density plots for premium and ideal cuts of diamonds.
- The group of premium cuts is shown in red, and the group of ideal cuts is shown in blue. The range of diamond prices is wide—in this case ranging from around \$300 to almost \$20,000. Extreme values are typical of monetary data such as income, customer value, tax liabilities, and bank account sizes.



VISUALIZING A SINGLE VARIABLE

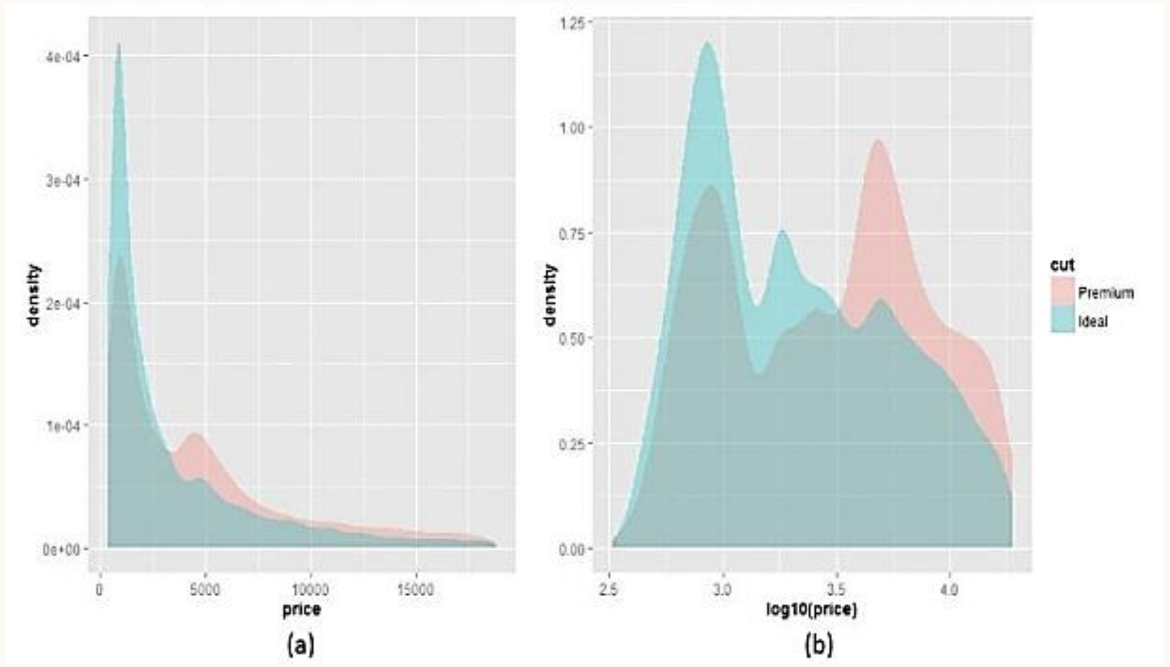
- Figure (b) shows more detail of the diamond prices than Figure (a) by taking the logarithm. The two humps in the premium cut represent two distinct groups of diamond prices: One group centers around (where the price is about $794), and the other centers around (where the price is about $5,012). The ideal cut contains three humps, centering around 2.9, 3.3, and 3.7 respectively.



VISUALIZING A SINGLE VARIABLE

- The R script to generate the plots in Figure 3.12 is shown next. The diamonds dataset comes with the ggplot2 package.

```
library("ggplot2")
data(diamonds) # load the diamonds dataset from ggplot2
# Only keep the premium and ideal cuts of diamonds
niceDiamonds <- diamonds[diamonds$cut=="Premium" |
diamonds$cut=="Ideal",]
summary(niceDiamonds$cut)
Fair Good Very Good Premium Ideal
0 0 0 13791 21551
# plot density plot of diamond prices
ggplot(niceDiamonds, aes(x=price, fill=cut)) +
geom_density(alpha = .3, color=NA)
# plot density plot of the log10 of diamond prices
ggplot(niceDiamonds, aes(x=log10(price), fill=cut)) +
geom_density(alpha = .3, color=NA)
```



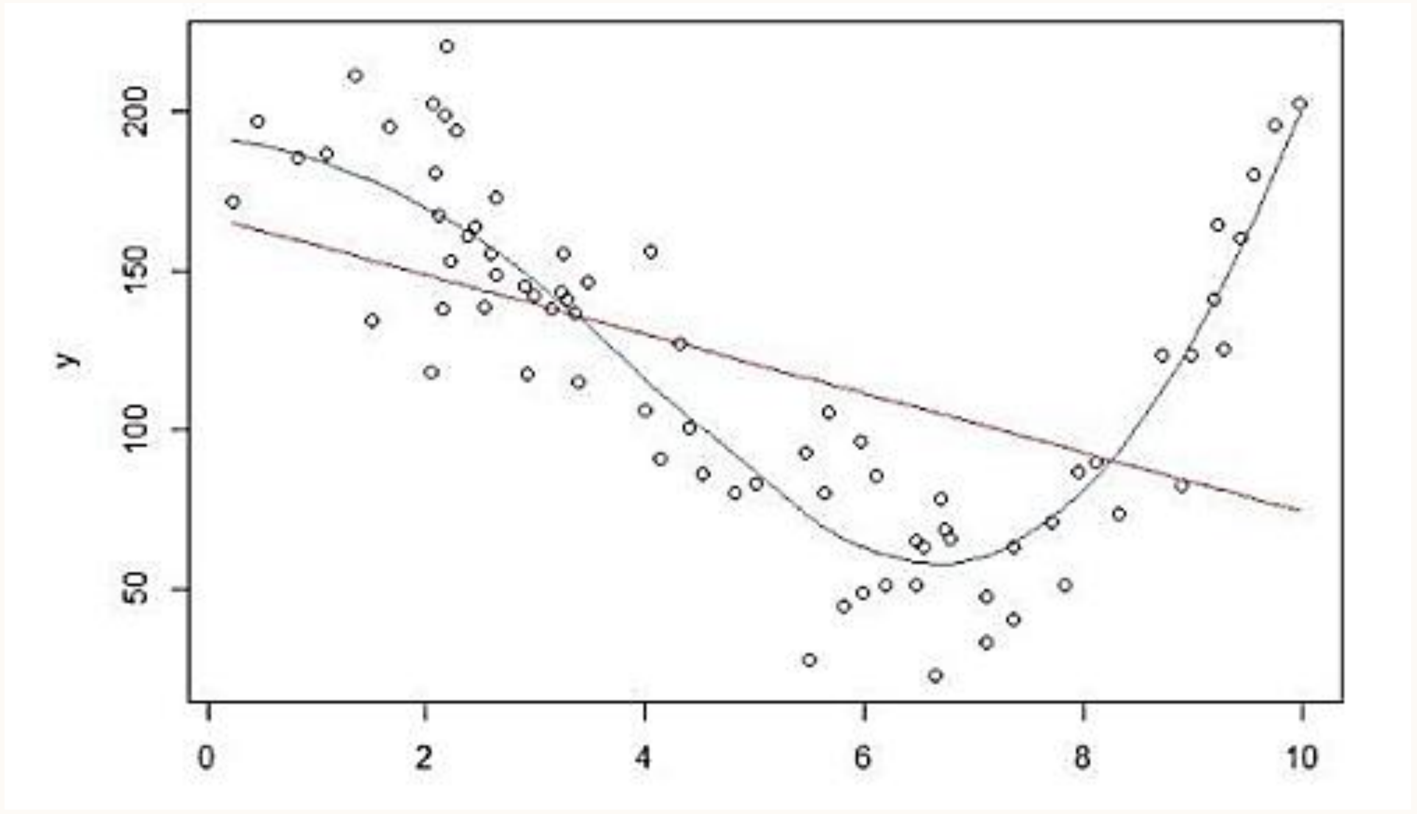
As an alternative to ggplot2, the lattice package provides a function called densityplot() for making simple density plots.

EXAMINING MULTIPLE VARIABLES

- A scatterplot is a simple and widely used visualization for finding the relationship among multiple variables.
- A scatterplot can represent data with up to five variables using x-axis, y-axis, size, color, and shape. But usually only two to four variables are portrayed in a scatterplot to minimize confusion.
- When examining a scatterplot, one needs to pay close attention to the possible relationship between the variables.
- If the functional relationship between the variables is somewhat pronounced, the data may roughly lie along a straight line, a parabola, or an exponential curve.
- If variable y is related exponentially to x, then the plot of x versus log(y) is approximately linear.
- If the plot looks more like a cluster without a pattern, the corresponding variables may have a weak relationship.

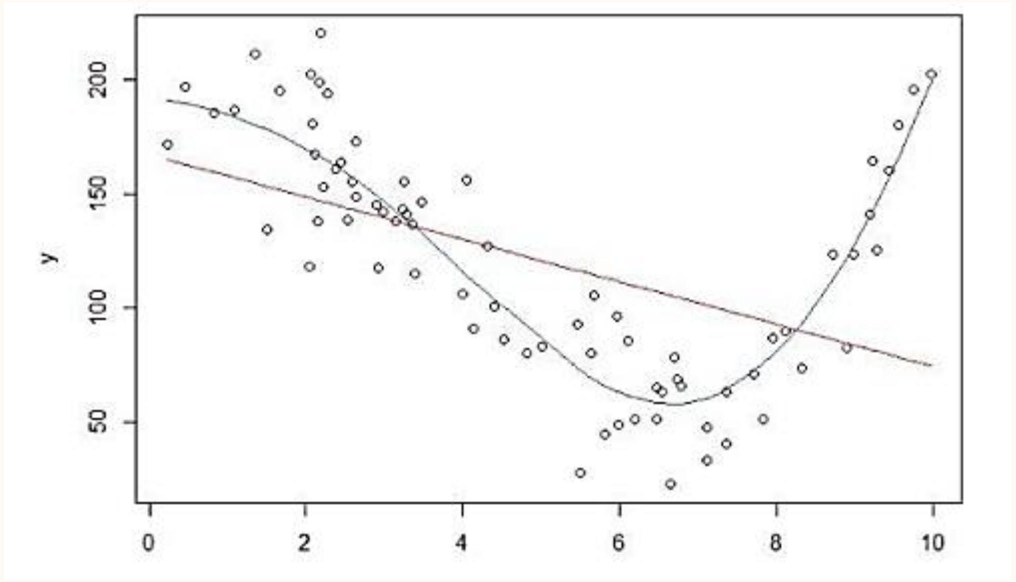
EXAMINING MULTIPLE VARIABLES

- The **scatterplot** in Figure portrays the relationship of two variables: x and y. The red line shown on the graph is the fitted line from the linear regression.
- Figure shows that the regression line does not fit the data well.
- This is a case in which linear regression cannot model the relationship between the variables. Alternative methods such as the loess() function can be used to fit a nonlinear line to the data.
- The blue curve shown on the graph represents the LOESS curve, which fits the data better than linear regression.



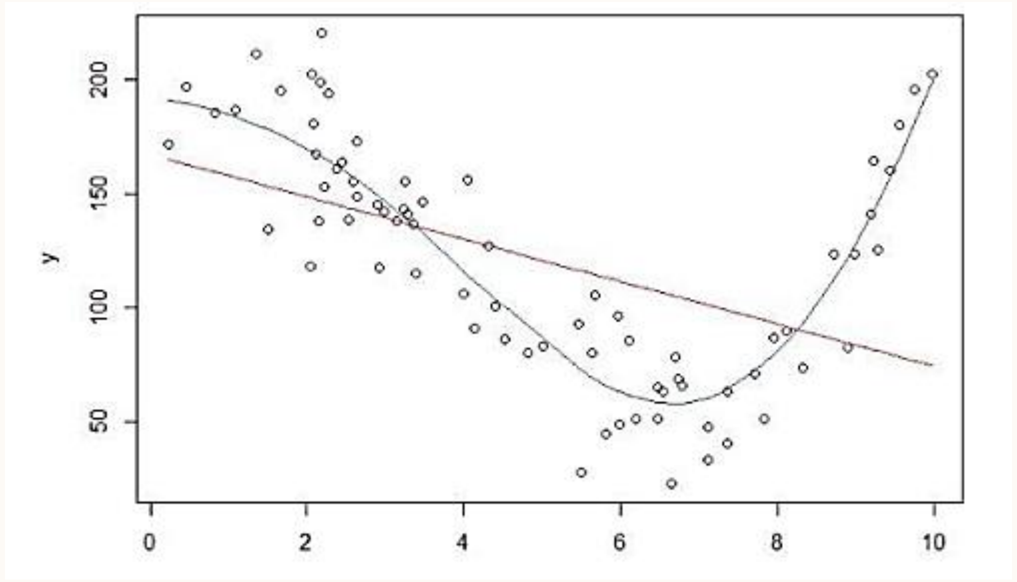
EXAMINING MULTIPLE VARIABLES

- The R code to produce Figure is as follows.
- The runif(75,0,10) generates 75 numbers between 0 to 10 with random deviates, and the numbers conform to the uniform distribution.
- The rnorm(75,0,20) generates 75 numbers that conform to the normal distribution, with the mean equal to 0 and the standard deviation equal to 20.
- The points() function is a generic function that draws a sequence of points at the specified coordinates.
- Parameter type="l" tells the function to draw a solid line.
- The col parameter sets the color of the line, where 2 represents the red color and 4 represents the blue color.



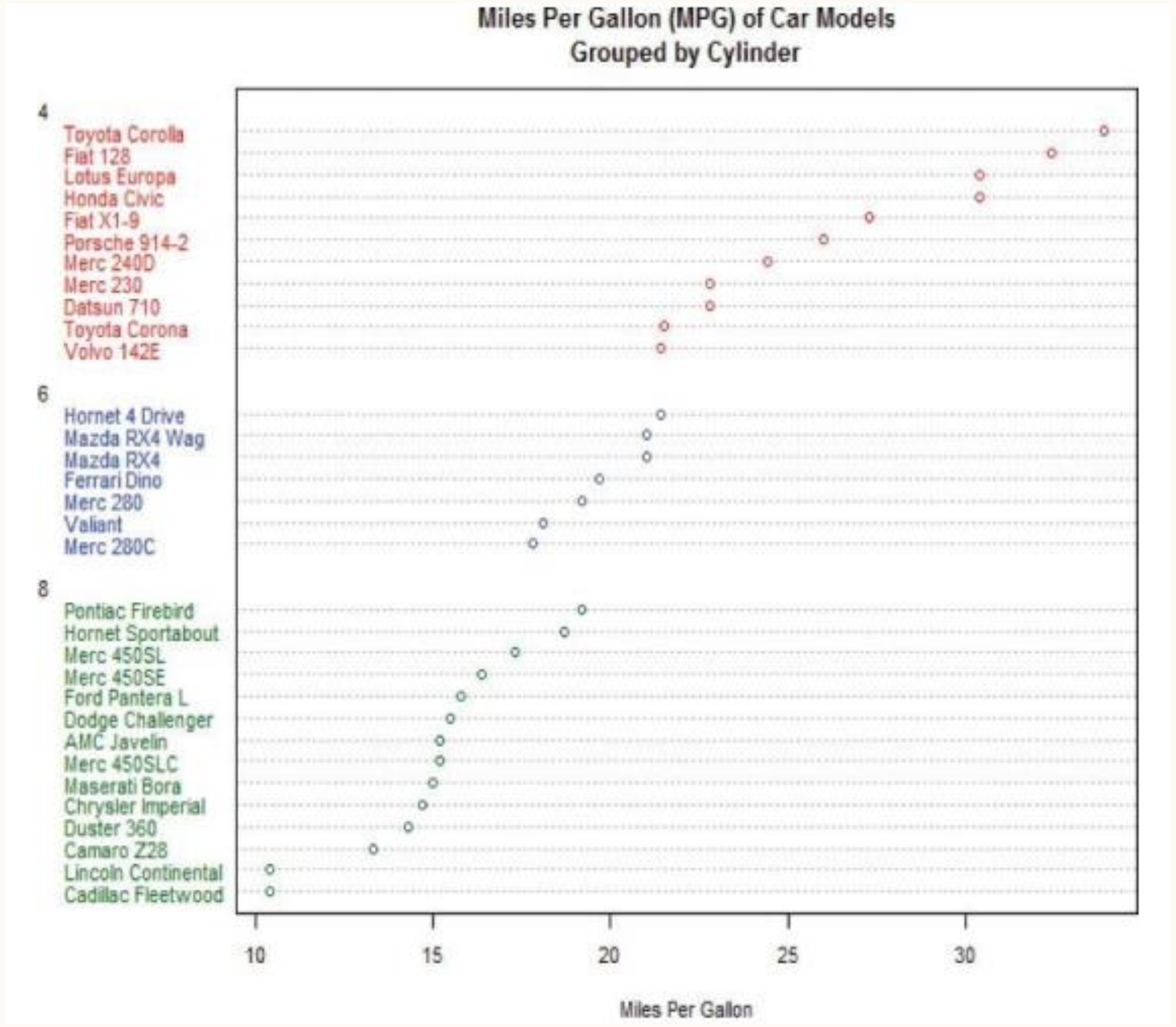
EXAMINING MULTIPLE VARIABLES

```
# 75 numbers between 0 and 10 of uniform distribution
x <- runif(75, 0, 10)
x <- sort(x)
y <- 200 + x^3 - 10 * x^2 + x + rnorm(75, 0, 20)
lr <- lm(y ~ x) # linear regression
poly <- loess(y ~ x) # LOESS
fit <- predict(poly) # fit a nonlinear line
plot(x,y)
# draw the fitted line for the linear regression
points(x, lr$coefficients[1] + lr$coefficients[2] * x,
type = “l”, col = 2)
# draw the fitted line with LOESS
points(x, fit, type = “l”, col = 4)
```

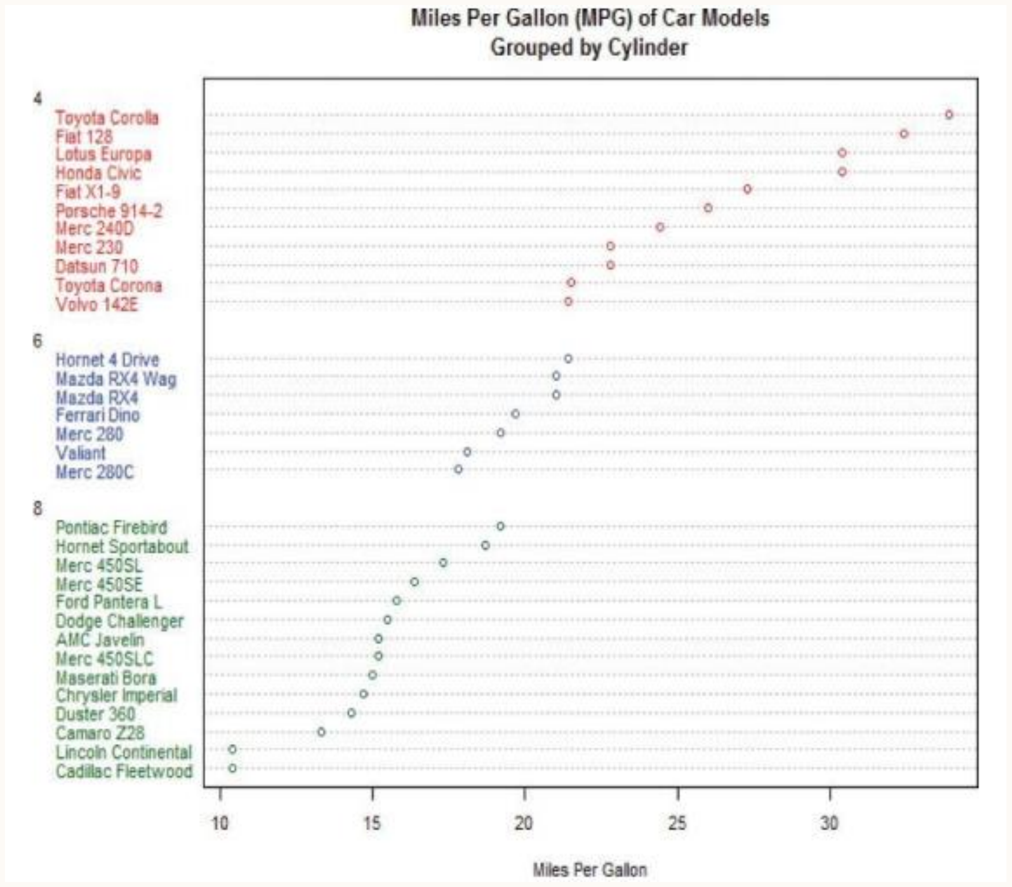


EXAMINING MULTIPLE VARIABLES

- **Dotchart and barplot** can visualize multiple variables. Both of them use color as an additional dimension for visualizing the data.
- For the same mtcars dataset, Figure shows a dotchart that groups vehicle cylinders at the y-axis and uses colors to distinguish different cylinders.
- The vehicles are sorted according to their MPG values.



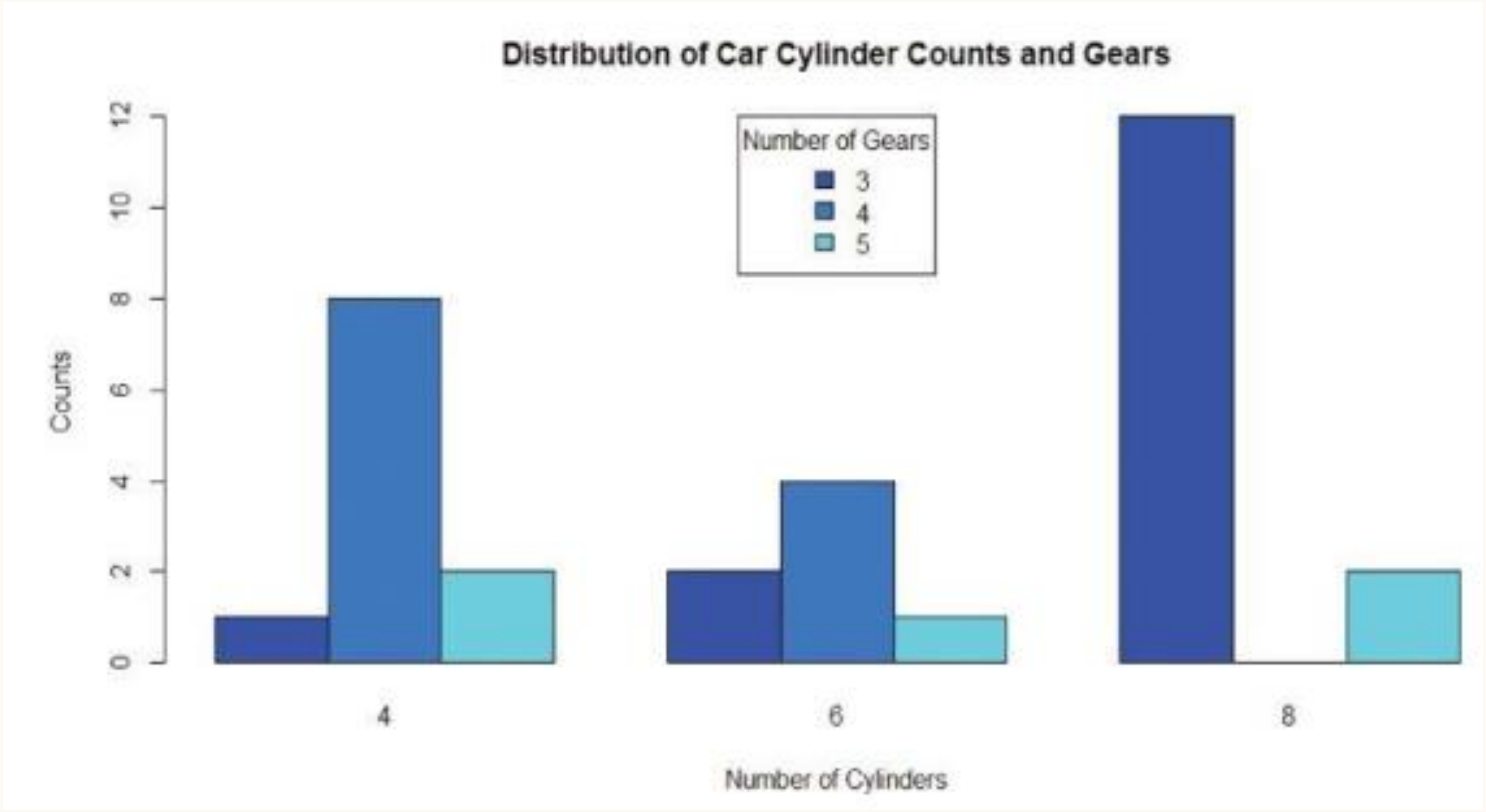
EXAMINING MULTIPLE VARIABLES



- The code to generate Figure
- # sort by mpg
- cars <- mtcars[order(mtcars$mpg),]
- # grouping variable must be a factor
- cars$cyl <- factor(cars$cyl)
- cars$color[cars$cyl==4] <- “red”
- cars$color[cars$cyl==6] <- “blue”
- cars$color[cars$cyl==8] <- “darkgreen”
- dotchart(cars$mpg, labels=row.names(cars), cex=.7, groups= cars$cyl,
- main=“Miles Per Gallon (MPG) of Car Models\nGrouped by Cylinder”,
- xlab=“Miles Per Gallon”, color=cars$color, gcolor=“black”)

EXAMINING MULTIPLE VARIABLES

- The barplot in Figure visualizes the distribution of car cylinder counts and number of gears.



EXAMINING MULTIPLE VARIABLES

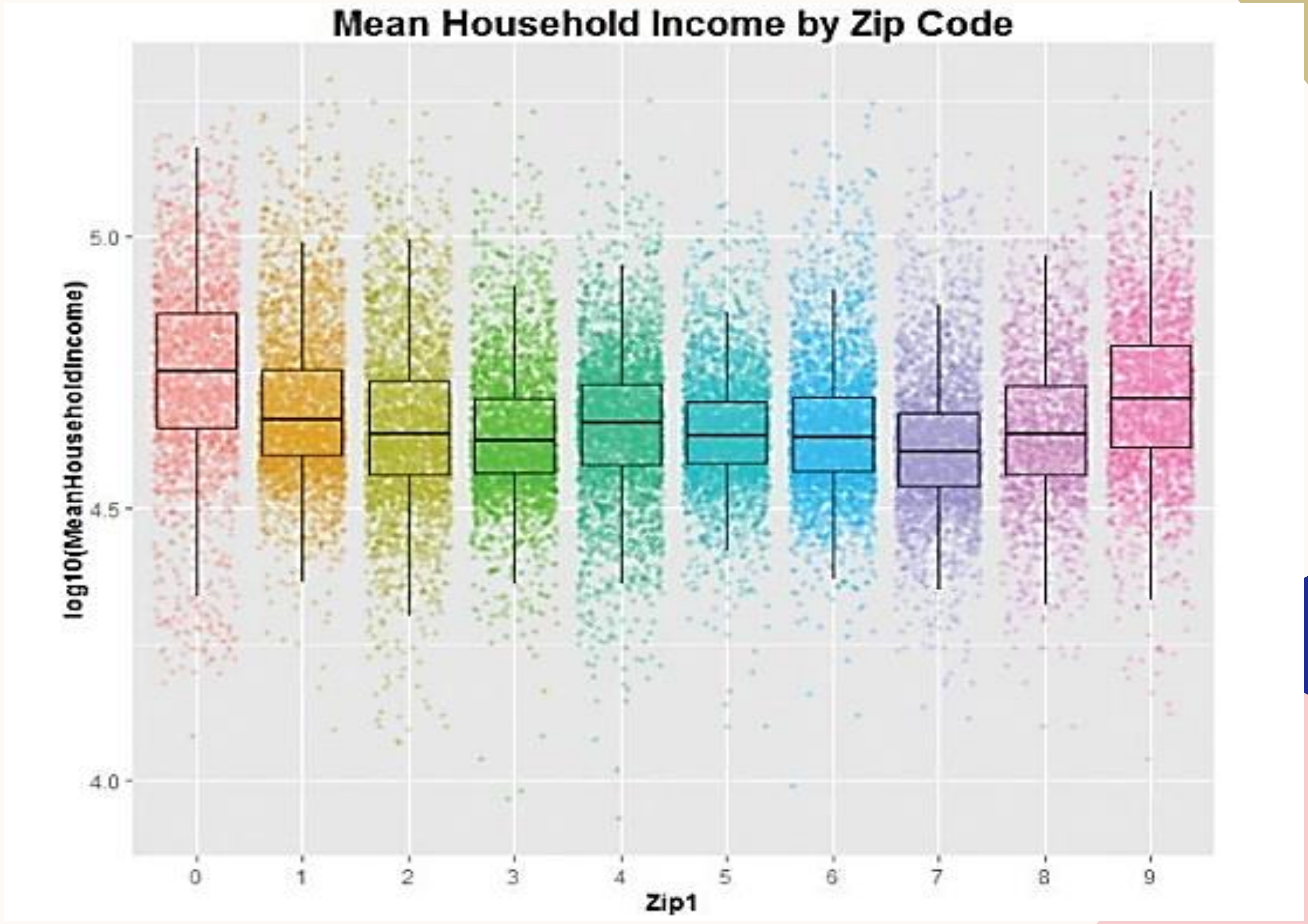
- The barplot in Figure visualizes the distribution of car cylinder counts and number of gears. The x-axis represents the number of cylinders, and the color represents the number of gears.
- The code to generate Figure

```
counts <- table(mtcars$gear, mtcars$cyl)
barplot(counts, main="Distribution of Car Cylinder Counts and Gears",
xlab="Number of Cylinders", ylab="Counts",
col=c("#0000FFFF","#0080FFFF","#00FFFFFF"),
legend = rownames(counts), beside=TRUE,
args.legend = list(x="top", title = "Number of Gears"))
```

EXAMINING MULTIPLE VARIABLES

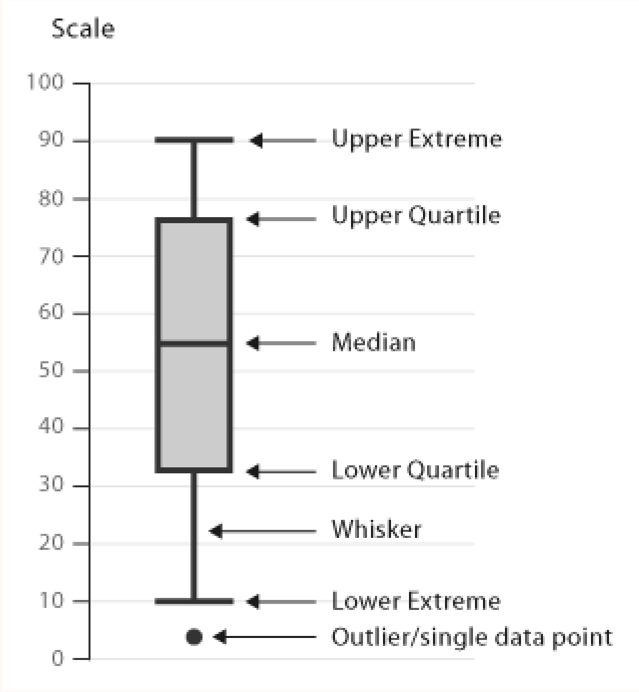
- **Box-and-whisker plots** show the distribution of a continuous variable for each value of a discrete variable.
- The box-and-whisker plot in Figure visualizes mean household incomes as a function of region in the United States.
- The first digit of the U.S. postal (“ZIP”) code corresponds to a geographical region in the United States.
- In Figure, each data point corresponds to the mean household income from a particular zip code.
- The horizontal axis represents the first digit of a zip code, ranging from 0 to 9, where 0 corresponds to the northeast region of the United States (such as Maine, Vermont, and Massachusetts), and 9 corresponds to the southwest region (such as California and Hawaii).
- The vertical axis represents the logarithm of mean household incomes. The logarithm is taken to better visualize the distribution of the mean household incomes.

EXAMINING MULTIPLE VARIABLES



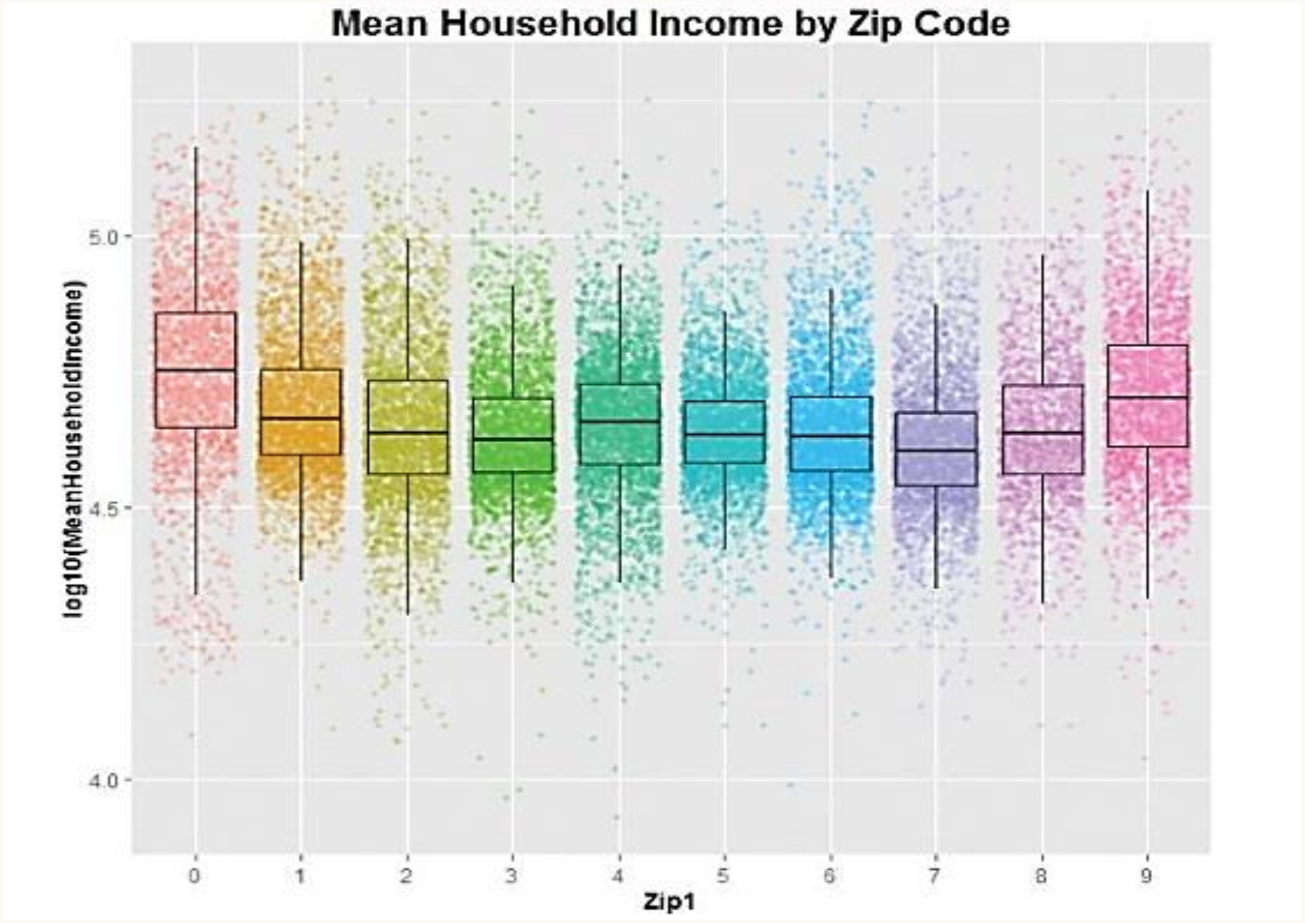
EXAMINING MULTIPLE VARIABLES

- In this figure, the scatterplot is displayed beneath the box-and-whisker plot, with some jittering for the overlap points so that each line of points widens into a strip.
- The “box” of the box-and-whisker shows the range that contains the central 50% of the data, and the line inside the box is the location of the median value.
- The upper and lower hinges of the boxes correspond to the first and third quartiles of the data. The upper whisker extends from the hinge to the highest value that is within 1.5 * IQR of the hinge. The lower whisker extends from the hinge to the lowest value within 1.5 * IQR of the hinge. IQR is the inter-quartile range.
- The points outside the whiskers can be considered possible outliers.



EXAMINING MULTIPLE VARIABLES

- The graph shows how household income varies by region. The highest median incomes are in region 0 and region 9.
- Region 0 is slightly higher, but the boxes for the two regions overlap enough that the difference between the two regions probably is not significant.
- The lowest household incomes tend to be in region 7, which includes states such as Louisiana, Arkansas, and Oklahoma.



EXAMINING MULTIPLE VARIABLES

- Assuming a data frame called DF contains two columns (MeanHouseholdIncome and Zip1), the following R script uses the ggplot2 library to plot a graph that is similar to Figure.

```
library(ggplot2)# plot the jittered scatterplot w/ boxplot
# color-code points with zip codes
# the outlier.size=0 prevents the boxplot from plotting the outlier
ggplot(data=DF, aes(x=as.factor(Zip1), y=log10(MeanHouseholdIncome))) +
geom_point(aes(color=factor(Zip1)), alpha=0.2, position="jitter") +
geom_boxplot(outlier.size=0, alpha=0.1) +
guides(colour=FALSE) +
ggtitle ("Mean Household Income by Zip Code")
```

EXAMINING MULTIPLE VARIABLES

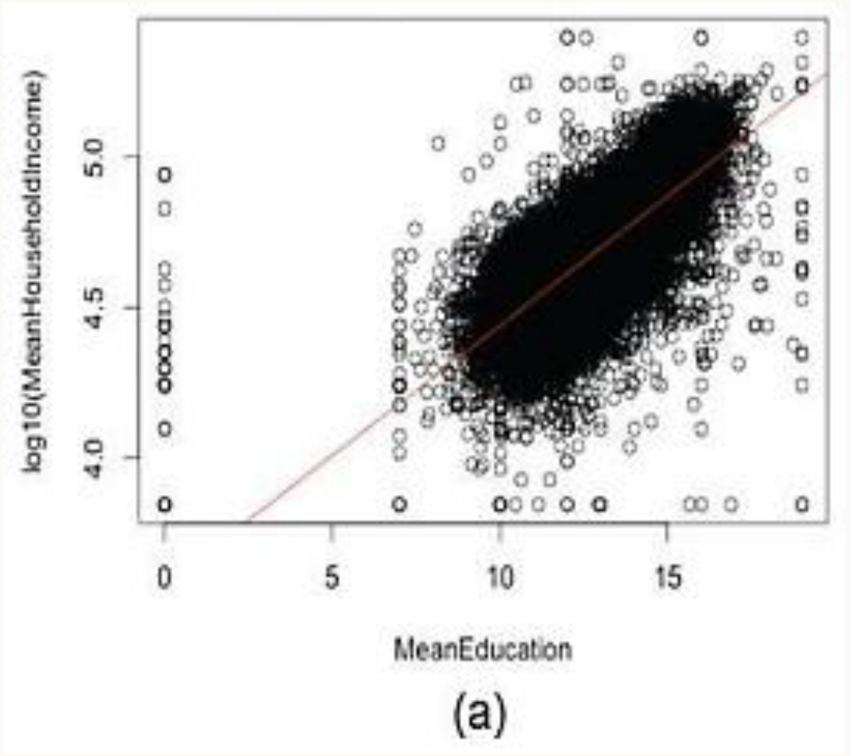
- Assuming a data frame called DF contains two columns (MeanHouseholdIncome and Zip1), the following R script uses the ggplot2 library to plot a graph that is similar to Figure.

```
library(ggplot2)# plot the jittered scatterplot w/ boxplot
# color-code points with zip codes
# the outlier.size=0 prevents the boxplot from plotting the outlier
ggplot(data=DF, aes(x=as.factor(Zip1), y=log10(MeanHouseholdIncome))) +
geom_point(aes(color=factor(Zip1)), alpha=0.2, position="jitter") +
geom_boxplot(outlier.size=0, alpha=0.1) +
guides(colour=FALSE) +
ggtitle ("Mean Household Income by Zip Code")
```

Alternatively, one can create a simple box-and-whisker plot with the boxplot() function provided by the R base package

EXAMINING MULTIPLE VARIABLES

- **Hexbinplot** for Large Datasets
- If there is too much data, the structure of the data may become difficult to see in a scatterplot. Consider a case to compare the logarithm of household income against the years of education, as shown in Figure.
- The cluster in the scatterplot on the left (a) suggests a somewhat linear relationship of the two variables. However, one cannot really see the structure of how the data is distributed inside the cluster.
- This is a Big Data type of problem. Millions or billions of data points would require different approaches for exploration, visualization, and analysis.

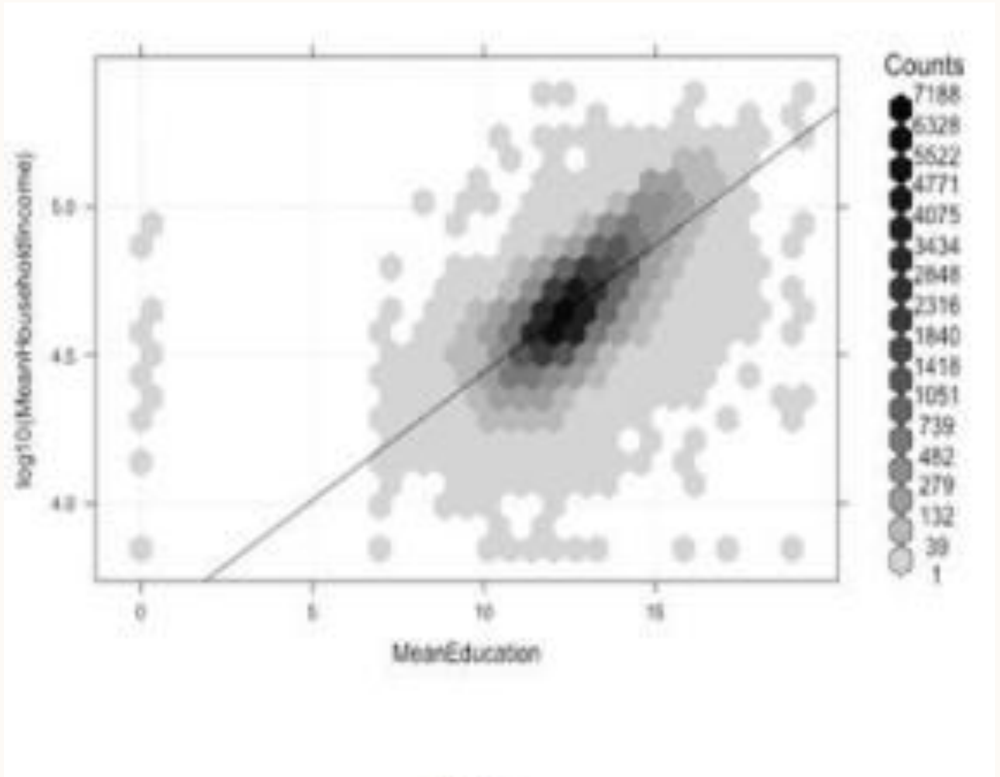


EXAMINING MULTIPLE VARIABLES

- Although color and transparency can be used in a scatterplot to address this issue, a hexbinplot is sometimes a better alternative.
- A hexbinplot combines the ideas of scatterplot and histogram.
- Similar to a scatterplot, a hexbinplot visualizes data in the xaxis and y-axis.
- Data is placed into hexbins, and the third dimension uses shading to represent the concentration of data in each hexbin.

EXAMINING MULTIPLE VARIABLES

- In Figure, the same data is plotted using a hexbinplot.
- The hexbinplot shows that the data is more densely clustered in a streak that runs through the center of the cluster, roughly along the regression line. The biggest concentration is around 12 years of education, extending to about 15 years.
- In Figure, note the outlier data at MeanEducation=0. These data points may correspond to some missing data that needs further cleansing.

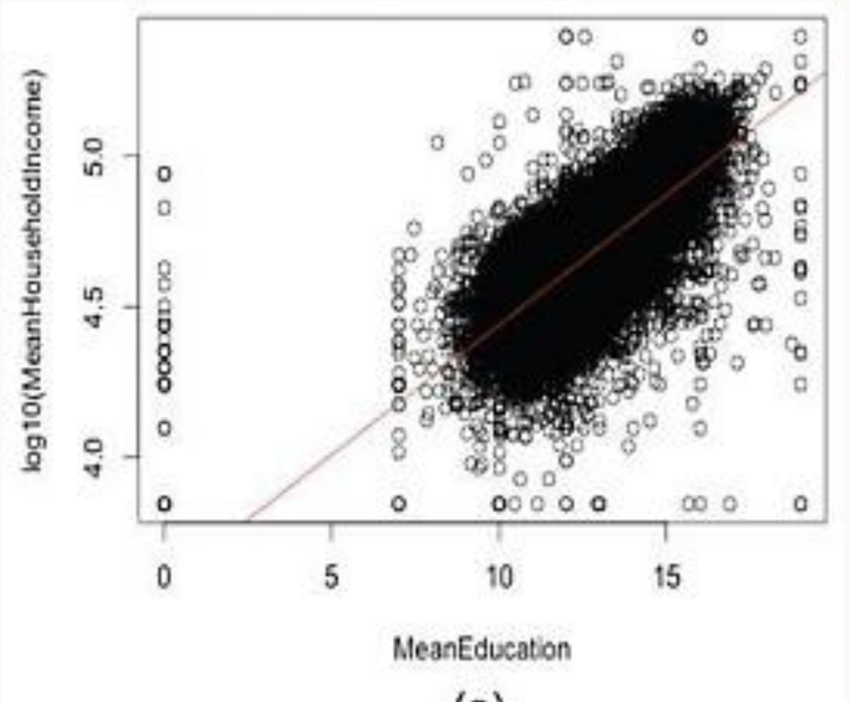


(b)

EXAMINING MULTIPLE VARIABLES

- Assuming the two variables MeanHouseholdIncome and MeanEducation are from a data frame named zcta, the scatterplot of Figure (a) is plotted by the following R code.

```
# plot the data points
plot(log10(MeanHouseholdIncome) ~
MeanEducation, data=zcta)
# add a straight fitted line of the linear regression
abline(lm(log10(MeanHouseholdIncome) ~
MeanEducation, data=zcta),
col='red')
```



(a)

EXAMINING MULTIPLE VARIABLES

Using the zcta data frame, the hexbinplot of Figure (b) is plotted by the following R code. Running the code requires the use of the hexbin package, which can be installed by running install .packages(“hexbin”).

library(hexbin)

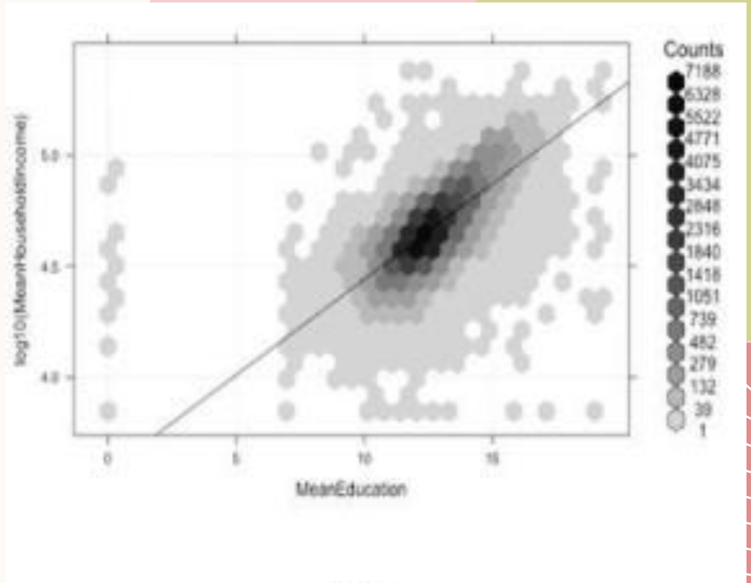
# “g” adds the grid, “r” adds the regression line

# sqrt transform on the count gives more dynamic range to the shading

# inv provides the inverse transformation function of trans

hexbinplot(log10(MeanHouseholdIncome) ~ MeanEducation,

data=zcta[illegible]s = sqrt, inv = function(x) x^2, type=c(“g”, “r”))

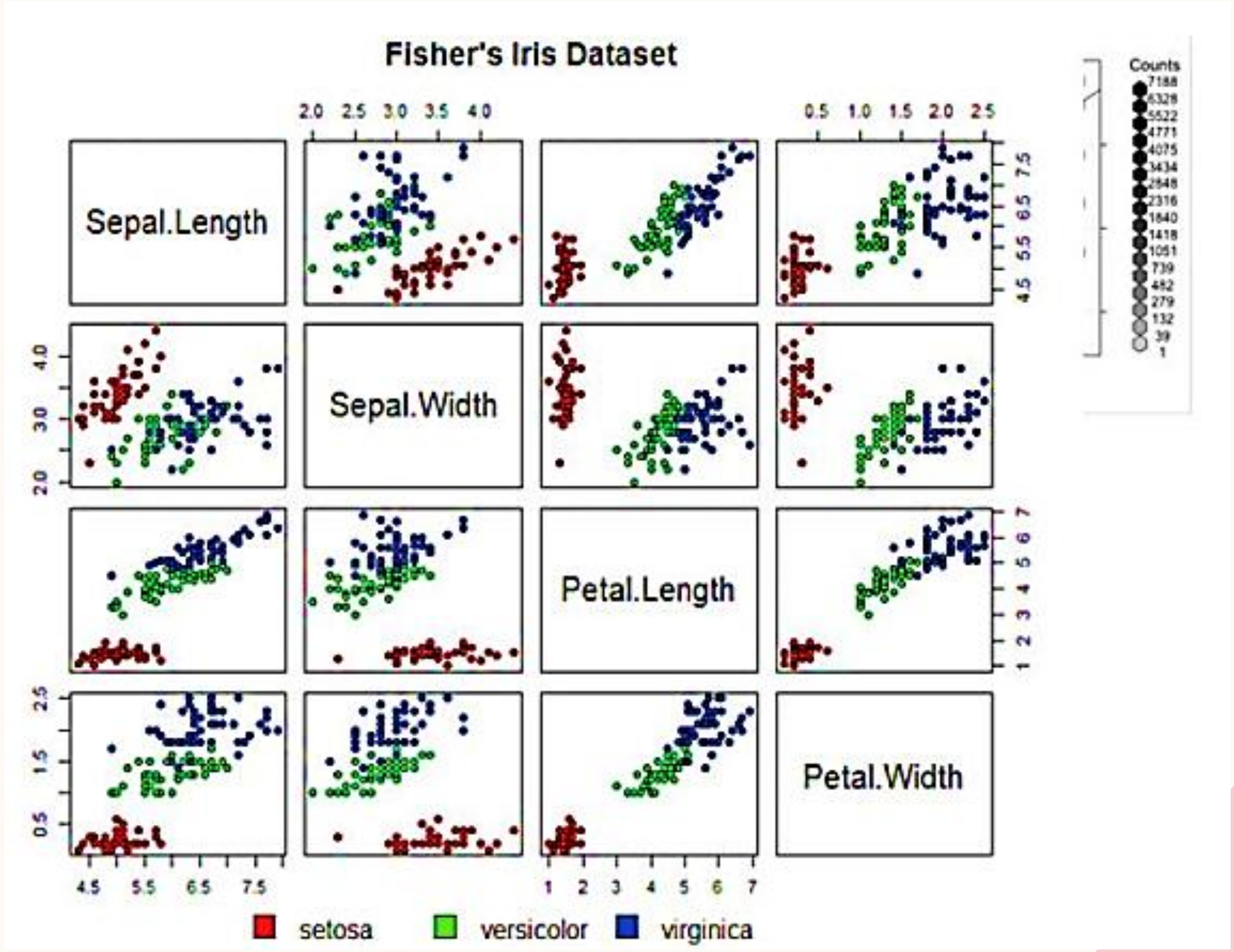


(b)

EXAMINING MULTIPLE VARIABLES

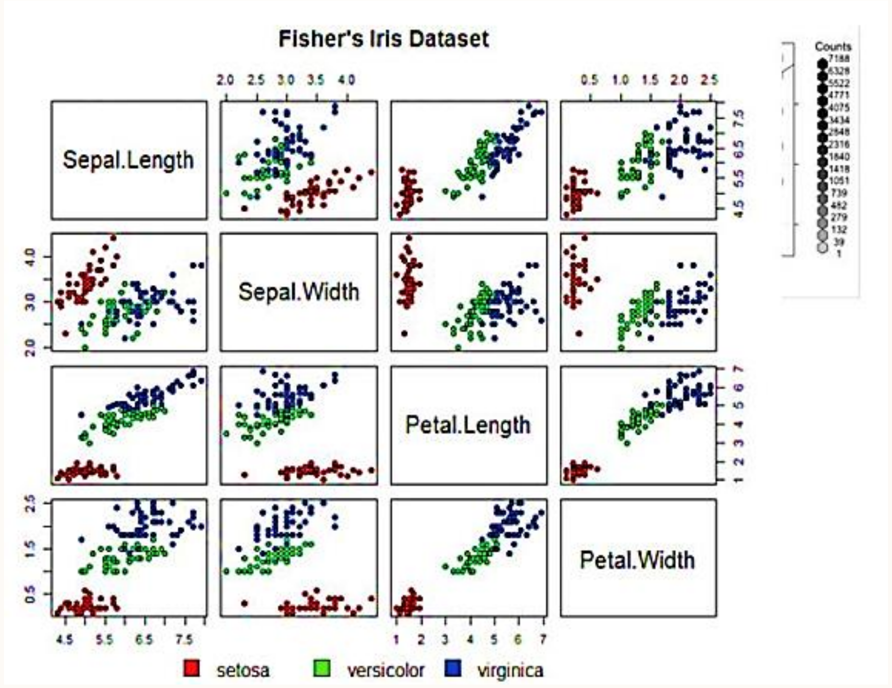
- A scatterplot matrix shows many scatterplots in a compact, side-by-side fashion.
- The scatterplot matrix, therefore, can visually represent multiple attributes of a dataset to explore their relationships, magnify differences, and disclose hidden patterns.
- Fisher's iris dataset includes the measurements in centimeters of the sepal length, sepal width, petal length, and petal width for 50 flowers from three species of iris.
- The three species are setosa, versicolor, and virginica. The iris dataset comes with the standard R distribution.
- In Figure, all the variables of Fisher's iris dataset (sepal length, sepal width, petal
- length, and petal width) are compared in a scatterplot matrix. The three different colors represent three species of iris flowers. The scatterplot matrix in Figure allows its viewers to compare the differences across the iris species for any pairs of attributes.

EXAMINING MULTIPLE VARIABLES



EXAMINING MULTIPLE VARIABLES

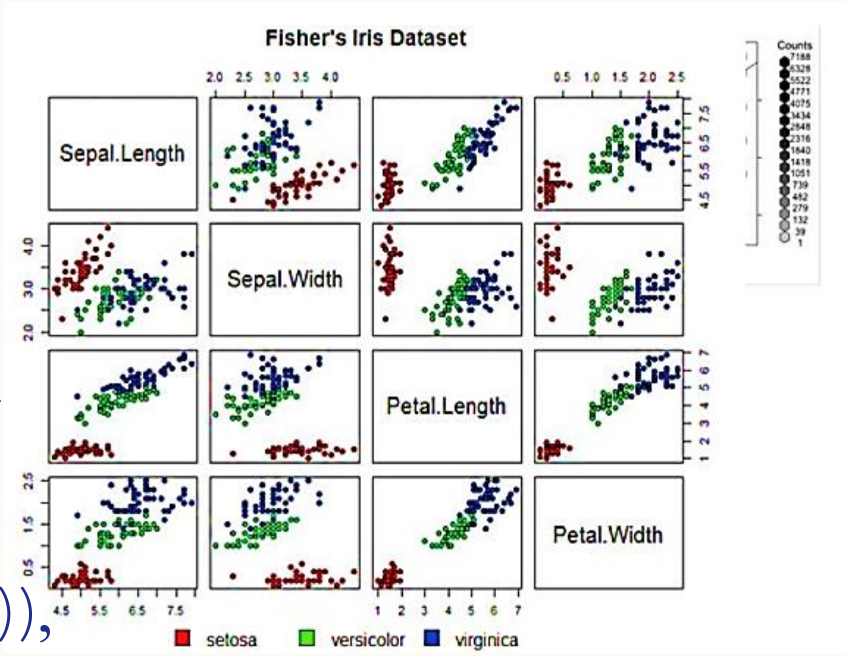
- Consider the scatterplot from the first row and third column of Figure, where sepal length is compared against petal length.
- The horizontal axis is the petal length, and the vertical axis is the sepal length.
- The scatterplot shows that versicolor and virginica share similar sepal and petal lengths, although the latter has longer petals.
- The petal lengths of all setosa are about the same, and the petal lengths are remarkably shorter than the other two species.
- The scatterplot shows that for versicolor and virginica, sepal length grows
  linearly with the petal length.



EXAMINING MULTIPLE VARIABLES

- The R code for generating the scatterplot matrix is provided next.

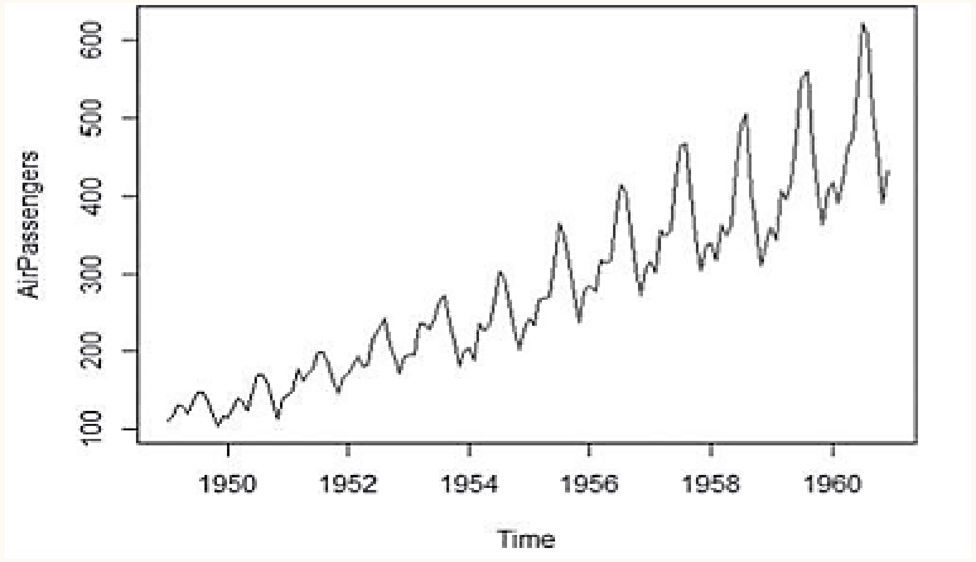
```
# define the colors
colors <- c(“red”, “green”, “blue”)
# draw the plot matrix
pairs(iris[1:4], main = “Fisher’s Iris Dataset”,
pch = 21, bg = colors[unclass(iris$Species)] )
# set graphical parameter to clip plotting to the figure region
par(xpd = TRUE)
# add legend
legend(0.2, 0.02, horiz = TRUE, as.vector(unique(iris$Species)),
fill = colors, bty = “n”)
```



The vector colors defines the color scheme for the plot. It could be changed to something
like colors <- c(“gray50”, “white”, “black”) to make the scatterplots grayscale.

ANALYZING A VARIABLE OVER TIME

- Visualizing a variable over time is the same as visualizing any pair of variables, but in this case the goal is to identify time-specific patterns.
- Figure plots the monthly total numbers of international airline passengers (in thousands) from January 1940 to December 1960. Enter plot(AirPassengers) in the R console to obtain a similar graph. The plot shows that, for each year, a large peak occurs mid-year around July and August, and a small peak happens around the end of the year, possibly due to the holidays. Such a phenomenon is referred to as a **seasonality effec**t.



DATA EXPLORATION VERSUS PRESENTATION

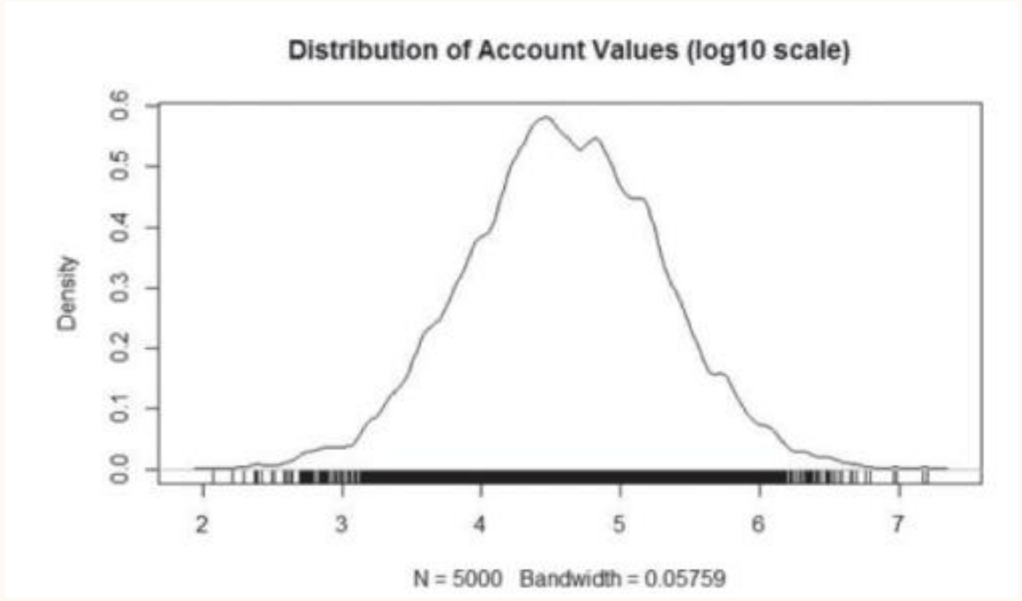
- The graphs which we learn before are more technical in nature and are better suited to technical audiences such as data scientists.

- Nontechnical stakeholders, however, generally prefer simple, clear graphics that focus on the message rather than the data.

DATA EXPLORATION VERSUS PRESENTATION

- Figure shows the density plot on the distribution of account values from a bank.
- The data has been converted to the log10 scale. The plot includes a rug on the bottom to show the distribution of the variable. This graph is more suitable for data scientists and business analysts because it provides information that can be relevant to the downstream analysis.

The graph shows that the transformed account values follow an approximate normal distribution, in the range from $100 to $10,000,000. The median account value is approximately $30,000 ( ), with the majority of the accounts between $1,000 ( ) and $1,000,000 ( )

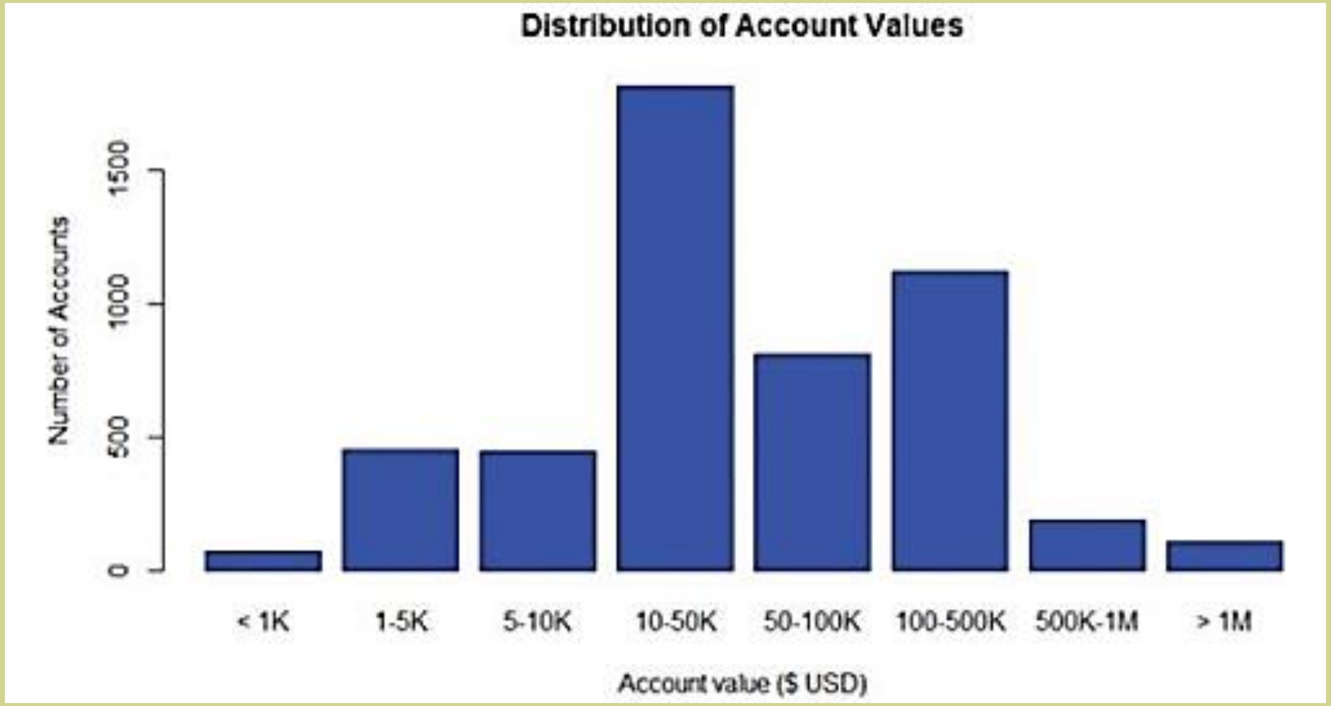


DATA EXPLORATION VERSUS PRESENTATION

- Density plots are fairly technical, and they contain so much information that they would be difficult to explain to less technical stakeholders.
- For example, it would be challenging to explain why the account values are in the log10 scale, and such information is not relevant to stakeholders.
- The same message can be conveyed by partitioning the data into log-like bins and presenting it as a histogram.

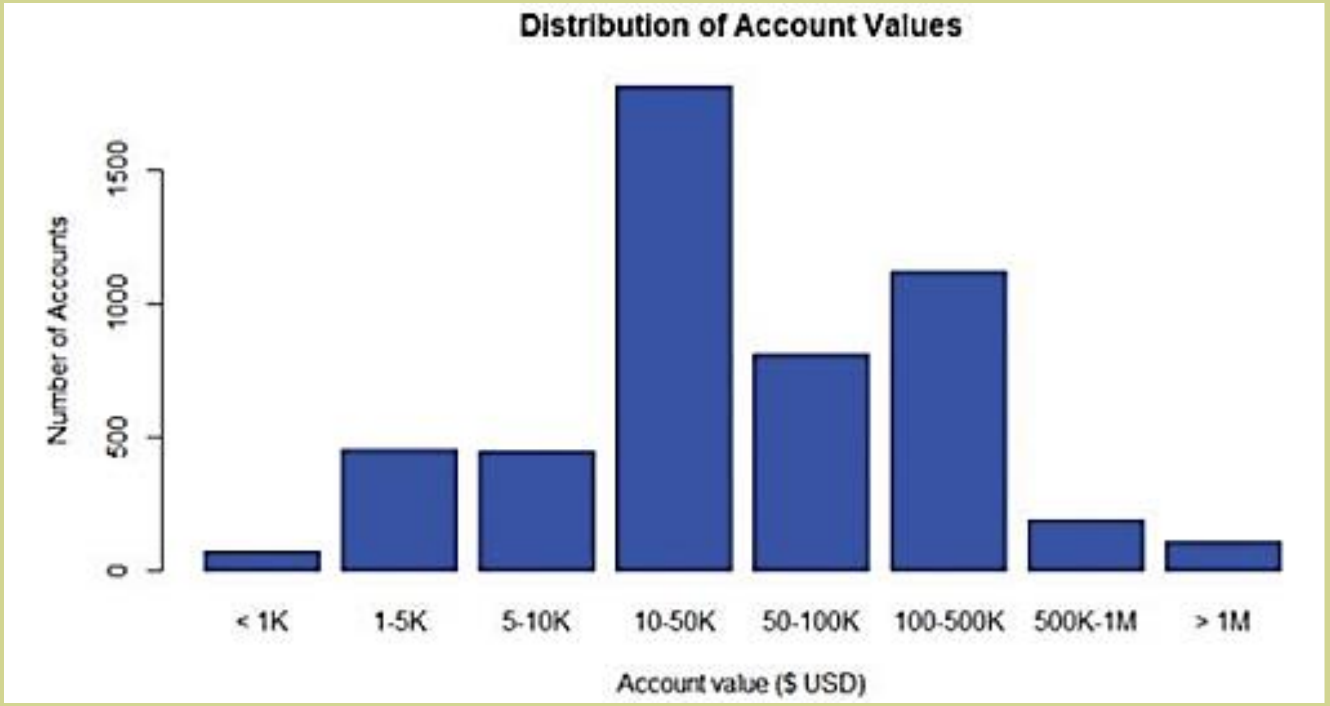
DATA EXPLORATION VERSUS PRESENTATION

- Figure shows, the bulk of the accounts are in the $1,000–1,000,000 range, with the peak concentration in the $10– 50K range, extending to $500K.
- This portrayal gives the stakeholders a better sense of the customer base than the density plot shown in Figure.



DATA EXPLORATION VERSUS PRESENTATION

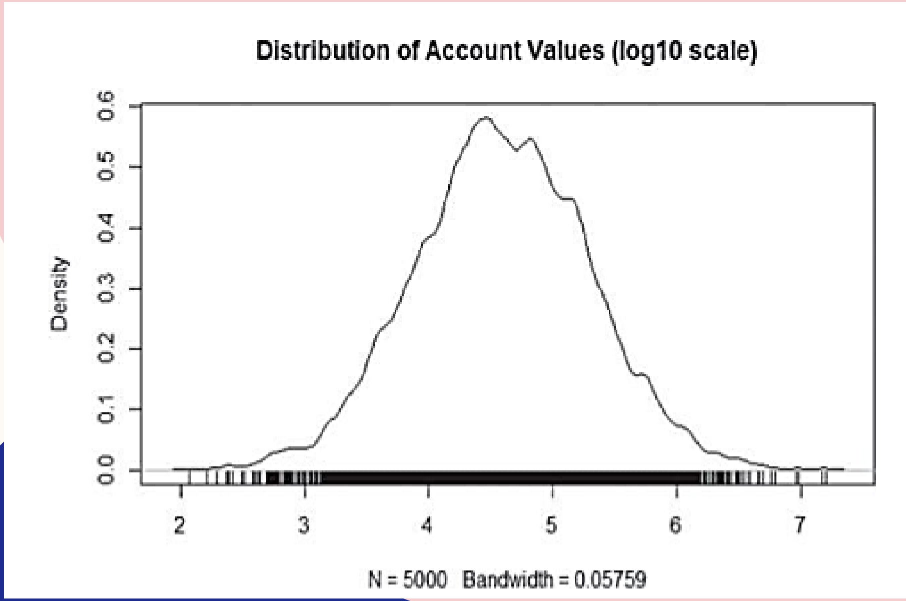
- Note that the bin sizes should be carefully chosen to avoid distortion of the data. In this example, the bins in Figure are chosen based on observations from the density plot in Figure. Without the density plot, the peak concentration might be just due to thesomewhat arbitrary appearing choices for the bin sizes.



DATA EXPLORATION VERSUS PRESENTATION

- Following is the R code to generate the plots in Figure on density plot.

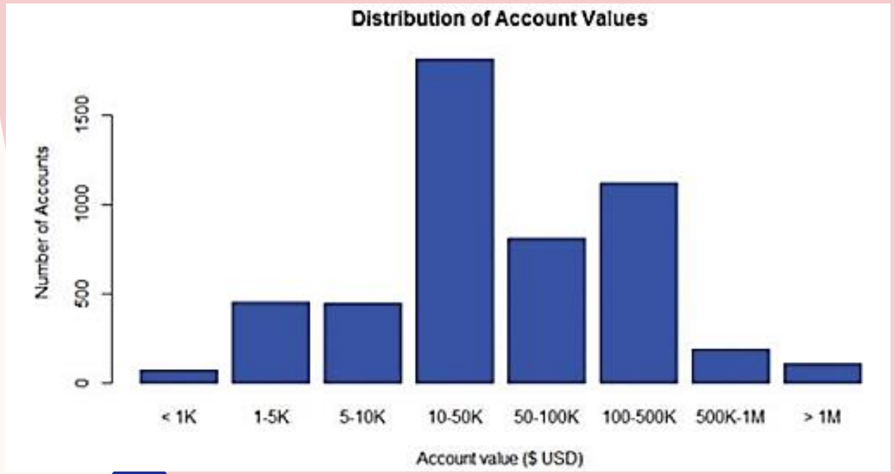
```
# Generate random log normal income data
income = rlnorm(5000, meanlog=log(40000), sdlog=log(5))
# Part I: Create the density plot
plot(density(log10(income), adjust=0.5),
main="Distribution of Account Values (log10 scale)")
# Add rug to the density plot
rug(log10(income))
# Plot the bins
plot(bins, main = "Distribution of Account Values",
xlab = "Account value ($ USD)",
ylab = "Number of Accounts", col="blue")
```



DATA EXPLORATION VERSUS PRESENTATION

- Following is the R code to generate the plots in Figure on histogram plot.

```
# Create “log-like bins”
breaks = c(0, 1000, 5000, 10000, 50000, 100000, 5e5, 1e6, 2e7)
# Create bins and label the data
bins = cut(income, breaks, include.lowest=T,
labels = c(“< 1K”, “1-5K”, “5-10K”, “10-50K”,
“50-100K”, “100-500K”, “500K-1M”, “> 1M”))
# Plot the bins
plot(bins, main = “Distribution of Account Values”,
xlab = “Account value ($ USD)”,
ylab = “Number of Accounts”, col=“blue”)
```


Distribution of Account Values
Number of Accounts
< 1K
1-5K
5-10K
10-50K
50-100K
100-500K
500K-1M
> 1M
Account value ($ USD)