Saving a ggplot. After the expenditure of so much effort, naturally you will want to save your ggplots so that you can incorporate them into your PA exam report. While `ggplot2` offers a way to save a graph for import into Microsoft Word (read Section 3.7 of *Data Visualization: A Practical Introduction* if you are interested), a much simpler solution that is allowed by the SOA is to just copy and paste. Simply right-click the ggplot created in R Markdown, select "`Copy Image`", and paste it on your report in Word. Simple and effective!

2.2 Data Exploration

Now that we have learned how to make some simple graphs by `ggplot2`, in this section we will apply our data visualization techniques to perform *exploratory data analysis* (EDA), which is often the warm-up part in a PA exam project. To give you some idea of how important EDA is, let's look at the very first task in the June 2019 PA exam:

> 1. *(5 points)* Explore the relationship of each variable to (the target variable)
>
> Use graphical displays and summary statistics to form preliminary conclusions regarding which variables are likely to have significant predictive power.

The first task in the Hospital Readmissions sample project is in a similar spirit:

> 1. *(6 points)* Perform univariate exploration of the four non-factor variables
>
> Use graphical displays and summary statistics to determine if any of these variables should be transformed and, if so, what transformation should be made. Do your recommended transformations, if any, and delete the original variables.

Data exploration is indispensable in any modeling exercise. Among its many purposes, it allows us to perform "sense checks" on the data (do the observations make sense?), identify potential data errors that may derail our analysis, appreciate the basic relationships between variables, determine the need for any transformations of variables, and, most importantly, decide on an appropriate predictive model that is likely to make practical sense. To accomplish EDA, we will draw upon a variety of numerical summary statistics and graphical tools that provide insights into the central tendency, dispersion, and other aspects of the variables of interest. After completing this section, you should be able to use R and `ggplot2` to generate useful summary statistics and plots for different types of variables that commonly arise in Exam PA.

2.2.1 Univariate Data Exploration

Let's begin with *univariate* data exploration—exploration that sheds light on the distribution of only one variable at a time. Such exploration usually takes two forms:

- *Statistical summaries:* To summarize the data by numerical statistics that capture different distributional properties of a variable. These summary statistics, such as the mean, variance, and frequencies, provide a broad overview of a variable's distribution and make it easy for us to compare different variables.

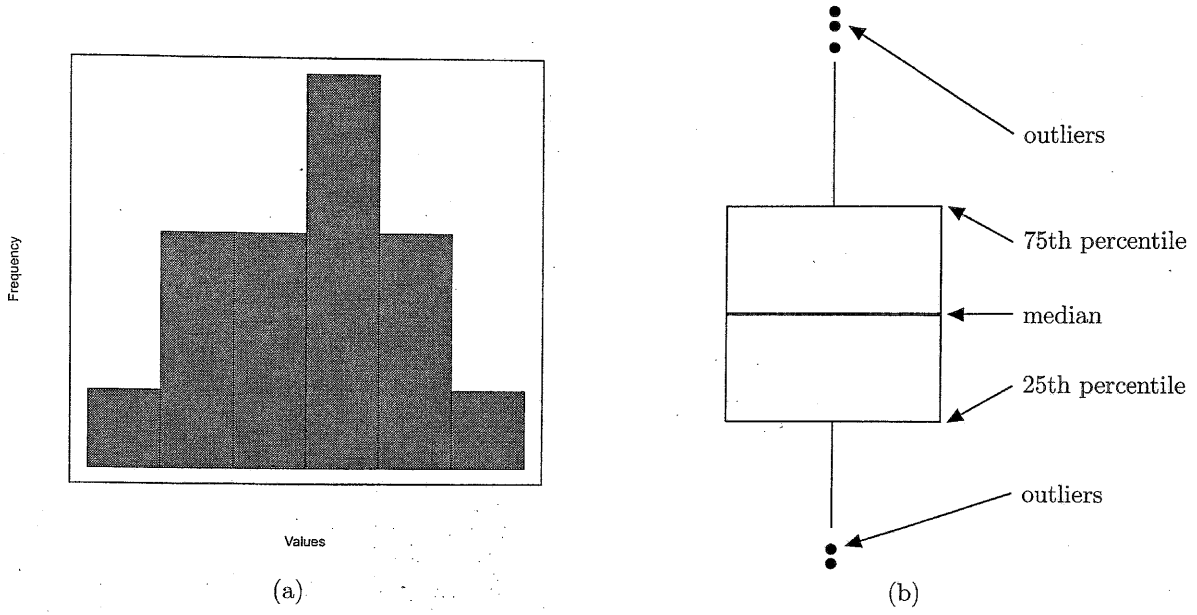


Figure 2.2.1: Prototypical histogram (left) and box plot (right).

- *Graphical summaries:* To visualize the distribution of a variable by graphical displays such as histograms, box plots, and bar charts. Graphical summaries allow us to get a quick glimpse at the overall distribution of a variable and are often more informative than looking at a table of numerical statistics. Furthermore, they may reveal outliers that cannot be easily detected by numerical statistics alone.

Very often, summary statistics and graphical representations are used in combination with one another, but the specific statistical and graphical tools will depend on whether the variables you are analyzing are numeric or categorical.

Numeric variables.

- *Statistical summaries:* The central tendency of a numeric variable, whether it be continuous or discrete, is often quantified by its mean or median, which can be readily produced in R by applying the `summary()` function to the variable of interest. Common measures of dispersion or spread include variance, standard deviation, and interquartile range (defined as the difference between the 75% quantile and the 25% quantile of a variable).

- *Graphical summaries—histograms and box plots:* To visualize the distribution of numeric variables, histograms and box plots are good graphical aids. *Histograms* divide the observations into several equally spaced bins (or buckets) and provide a visual summary of the count or relative frequency in each bin. Looking at a histogram, we can learn about the overall shape of the distribution of a numeric variable. Figure 2.2.1 (a) shows a prototypical histogram.

  Box plots, also known as box-and-whiskers plots, visualize the distribution of a numeric variable by placing its 25% quantile, the median, the 75% quantile in a "box," with the rest of the data points constituting the whiskers. The amount of spacing between different parts of a box plot reflects the degree of dispersion and skewness of the variable's distribution. "Outliers," defined as data points that are above or below 1.5 times the interquartile range from either

Variable	Description
`amt`	settled claim amount (continuous numeric variable)
`inj`	injury code, with seven levels: 1 (no injury), 2, 3, 4, 5, 6 (fatal), 9 (not recorded)
`legrep`	legal representation (0 = no, 1 = yes)
`op_time`	operational time (a standardized amount of time elapsed between the time when the injury was reported and the time when the claim was settled)

Table 2.2: Data dictionary for the personal injury insurance claims dataset.

edge of the box, are shown as large dotted points. See Figure 2.2.1 (b) for a prototypical box plot.

Although box plots do not directly show the actual shape of the variable's distribution, they offer a useful graphical summary of the key numeric statistics and allow for a visual comparison of the distributions of different numeric variables (in particular, the relative magnitude of the five summary statistics) or the distribution of the same numeric variable across different levels of another categorical variable.

Case study: Personal injury insurance claims. For the purposes of illustration, in this section we will look at the personal injury insurance dataset named `persinj.csv`.[iv] This dataset contains the information of $n = 22,036$ settled personal injury insurance claims. These claims were reported during the period from July 1989 to the end of 1999, with claims settled with zero payment not included. The variables in the dataset are described in Table 2.2. In Section 4.2, we will build a model to predict the size of personal injury insurance claims using other variables in the dataset. For now, we will perform data exploration of the variables in the dataset. The insights we gain here will go a long way towards constructing a good predictive model.

To get started, run CHUNK 1 to load the external CSV file into R as a data frame using the `read.csv()` function, which takes as an argument the name of the CSV file supplied as a character string.

```
# CHUNK 1
persinj <- read.csv("persinj.csv")
```

[iv] This file was downloaded online from `http://www.businessandeconomics.mq.edu.au/our_departments/Applied_Finance_and_Actuarial_Studies/research/books/GLMsforInsuranceData/data_sets`, which is part of the companion web page of the textbook *Generalized Linear Models for Insurance Data* (2008), by de Jong and Heller, and pre-processed to suit our purpose.

NOTE

- All data for Exam PA will be provided in CSV format, so `read.csv()` is the only function we have to learn for importing external datasets.
- Throughout this manual, make sure to save all the CSV files and Rmd files in the current working directory, which is where external files are read into R and results are saved. To ask RStudio to change the current working directory, choose `Session > Set Working Directory > Choose Directory...` .

Then run CHUNK 2 to apply the `summary()` function to the claim amount variable.

```
# CHUNK 2
summary(persinj$amt)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      10    6297   13854   38367   35123 4485797
```

When the `summary()` function is applied to a numeric variable, a five-number summary as well as its mean are produced. We can see that the mean of claim amount is way higher than its median, indicating that its distribution is highly skewed to the right. In fact, the largest claim amount, 4,485,797, is almost an astronomical figure.

In the following example, we illustrate how to calculate the summary statistics for the two groups of injuries classified by legal representation, which is a binary categorical variable. Doing so helps us understand the effect of legal representation on claim amount.

Example 2.2.1. (Based on Exercise 4.5.1 of PA Module 4: Calculating the summary statistics for two groups of injuries) Write R code to calculate the summary statistics for claim amount separately for injuries with legal representation and those without legal representation. Comment on the central tendency and dispersion of claim amount for these two groups of injuries.

Solution. To extract the two groups of injuries, we can use the method of logical subsetting introduced in Chapter 1 to split the dataset into two subsets, one corresponding to injuries without legal representation (`legrep = 0`) and one to injuries with legal representation (`legrep = 1`), then look at the summary statistics of the claim amount variable within the two subsets. This is done in CHUNK 3.

```
# CHUNK 3
persinj.0 <- persinj[persinj$legrep == 0, ]
persinj.1 <- persinj[persinj$legrep == 1, ]
summary(persinj.0$amt)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      10    4061   11164   32398   29641 2798362

summary(persinj.1$amt)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      20    7305   15309   41775   38761 4485797
```

The comparison is apparent: Claims with legal representation not only are larger on average (perhaps with legal advice the claimant is able to fight for larger settled claims), but also are more spread out. The relative variability is confirmed when the standard deviations of the two groups of claim amounts are computed by the `sd()` function.

```
sd(persinj.0$amt)

## [1] 77820.33

sd(persinj.1$amt)

## [1] 97541.38
```

□

Now we turn to visual representations. In `ggplot2`, a histogram is constructed by the `geom_histogram()` function, which only requires the `x` aesthetic (but not the `y` aesthetic). Run CHUNK 4 to make a histogram colored in blue for the claim amount variable (Figure 2.2.2). Note that:

- We have restricted the range of the horizontal axis to be between 0 and 100,000 to produce better visual effects. (Try to see how the histogram looks if the command `xlim(0, 100000)` is lifted.)
- Recall that the `fill` aesthetic is for coloring the *interior* of shapes. Had we used the `color` aesthetic and typed `color = "blue"` inside the `geom_histogram()` function, only the border lines of the vertical bars would be in blue and the interior would remain gray. (Try it!)

The histogram corroborates the earlier findings from the `summary()` function that the distribution of claim amount is heavily lopsided to the right with a pronounced tail. To correct for skewness, we can try to apply a *log transformation* to claim amount. The log transformation shrinks the variable values, pulls in extreme values while preserving their relative order and is a very commonly used tool for dealing with positive-valued skewed variables such as income. (A downside of the log transformation is that it cannot be directly applied when the variable of interest is non-positive.)

CHUNK 5 produces histograms for the *log* of claim amount for different choices of the `bins` parameter, which controls the number of bins in a histogram (see Figure 2.2.3). By default,

```
# CHUNK 4
library(ggplot2)
ggplot(persinj, aes(x = amt)) +
  geom_histogram(fill = "blue", alpha = 0.7) +
  xlim(0, 100000)
```

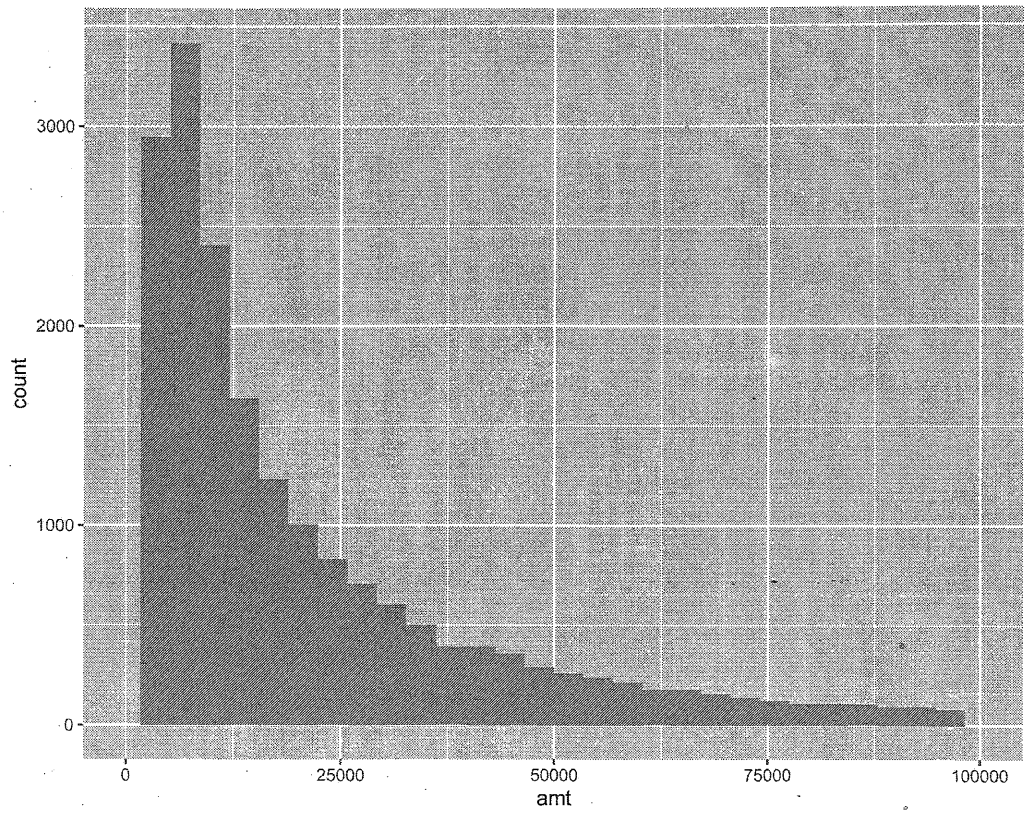


Figure 2.2.2: A histogram of claim amount in the personal injury insurance claims dataset.

`geom_histogram()` chooses a bin size based on a rule of thumb (which is 30 here). When plotting histograms, it is often a good idea to experiment with a few values of `bins` as different values of `bins` can reveal different patterns. In the current context, 20, 30, and 40 are all fine values for `bins` while 80 makes the function too jagged. Regardless of the value of `bins`, it is evident that the log transformation has effectively removed the skewness of claim amount and symmetrizes the resulting distribution. We will also see in the next subsection that the log transformation makes it much easier to discover relationships between variables.

Besides histograms, box plots are also convenient graphical aids to visualize the distribution of a numeric variable. They are constructed in `ggplot2` by the `geom_boxplot()` function, which takes the `y` aesthetic representing the numeric variable of interest (the `x` aesthetic is optional, but can be added to achieve splitting, as we will see later in this section). Use CHUNK 6 to draw a box plot for each of claim amount and the log of claim amount (Figure 2.2.4). While most of the raw claim amounts are so close that the 25% percentile, median, and 75% percentile all degenerate to the same line, the log transformation corrects for skewness and re-positions the data points for easy visual inspection. Still there are quite a number of "outliers," which are shown as large dotted points.

```
# CHUNK 5
p1 <- ggplot(persinj, aes(x = log(amt))) +
  geom_histogram(fill = "blue", alpha = 0.7) +
  ggtitle("Default value")
p2 <- ggplot(persinj, aes(x = log(amt))) +
  geom_histogram(bins = 20, fill = "blue", alpha = 0.7) +
  ggtitle("Bins = 20")
p3 <- ggplot(persinj, aes(x = log(amt))) +
  geom_histogram(bins = 40, fill = "blue", alpha = 0.7) +
  ggtitle("Bins = 40")
p4 <- ggplot(persinj, aes(x = log(amt))) +
  geom_histogram(bins = 80, fill = "blue", alpha = 0.7) +
  ggtitle("Bins = 80")
library(gridExtra)
grid.arrange(p1, p2, p3, p4, ncol = 2)
```

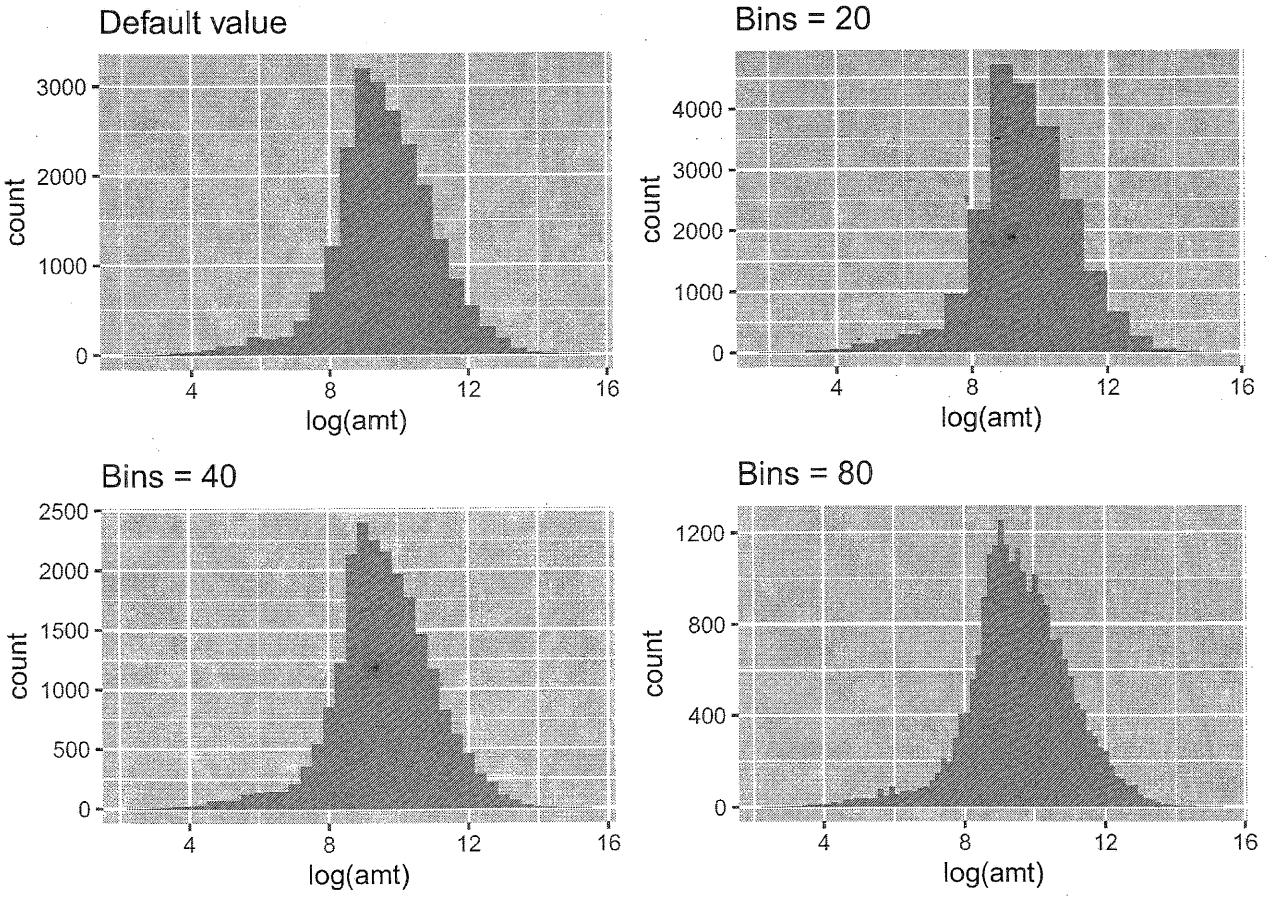


Figure 2.2.3: Four histograms of the log of claim amount with different values of `bins` in the personal injury insurance claims dataset.

```
# CHUNK 6
p1 <- ggplot(persinj, aes(y = amt)) +
  geom_boxplot()
p2 <- ggplot(persinj, aes(y = log(amt))) +
  geom_boxplot()
grid.arrange(p1, p2, ncol = 2)
```

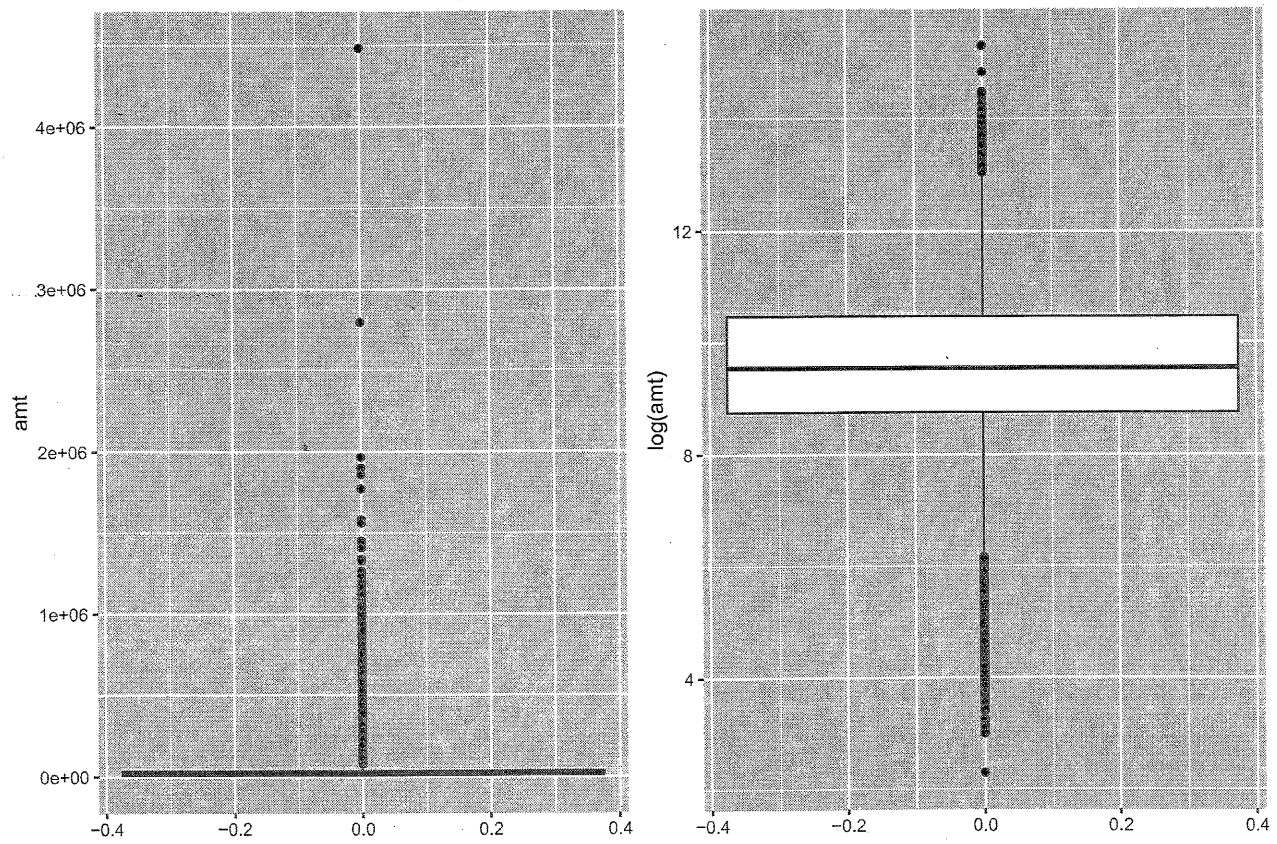


Figure 2.2.4: Box plots of claim amount and the log of claim amount in the personal injury insurance claims dataset.

Categorical variables.

- *Statistical summaries—Frequency tables:* Categorical variables, even when coded as numbers, do not always have a natural order, so statistical summaries like the mean and median may not make sense. To understand the distribution of a categorical variable, we can look at the relative frequency of each of its levels through a frequency table, constructed by the `table()` function in R.
- *Graphical summaries—bar charts:* When the number of levels of a categorical variable increases, a frequency table becomes more and more difficult to read. In most cases, the frequencies *per se* are not that important; what truly matters is their relative magnitude. In this regard, *bar charts* extract the information in a frequency table and present the numeric counts visually, highlighting the relative frequency of each level in the variable.

Case study revisited. The personal injury insurance claims dataset has two categorical variables, injury code (`inj`) and legal representation (`legrep`). Recall that `inj` has 7 levels while `legrep` is binary. Run CHUNK 7 to make two frequency tables for `inj`, one showing the raw counts and one showing the percentage counts of the seven levels of `inj`.

```
# CHUNK 7
table(persinj$inj)

##
##     1     2     3     4     5     6     9
## 15638  3376  1133   189   188   256  1256

table(persinj$inj)/nrow(persinj)

##
##           1           2           3           4           5           6
## 0.709656925 0.153203848 0.051415865 0.008576874 0.008531494 0.011617353
##           9
## 0.056997640
```

We can see that the predominant group of injuries is those of injury code 1, followed by codes 2, 9 and 3.

The numbers in a frequency table can be depicted in a bar chart created in `ggplot2` by the `geom_bar()` function, which takes the `x` aesthetic representing the categorical variable of interest. Run CHUNK 8 to produce two bar charts for injury code corresponding to the two frequency tables in CHUNK 7 (see Figure 2.2.5). (The command `..prop..` in the second plot in CHUNK 8, as its name indicates, computes proportions rather than raw counts and relies on the so-called stat function associated with a geom function. This concept is rarely used in Exam PA. If you are interested, read pages 80 and 81 of *Data Visualization: A Practical Introduction.*)

```
# CHUNK 8
# first convert inj and legrep to factors (original data type is integer)
persinj$inj <- as.factor(persinj$inj)
persinj$legrep <- as.factor(persinj$legrep)

p1 <- ggplot(persinj, aes(x = inj)) +
  geom_bar(fill = "blue")
p2 <- ggplot(persinj, aes(x = inj)) +
  geom_bar(fill = "blue", aes(y = ..prop.., group = 1))
grid.arrange(p1, p2, ncol = 2)
```

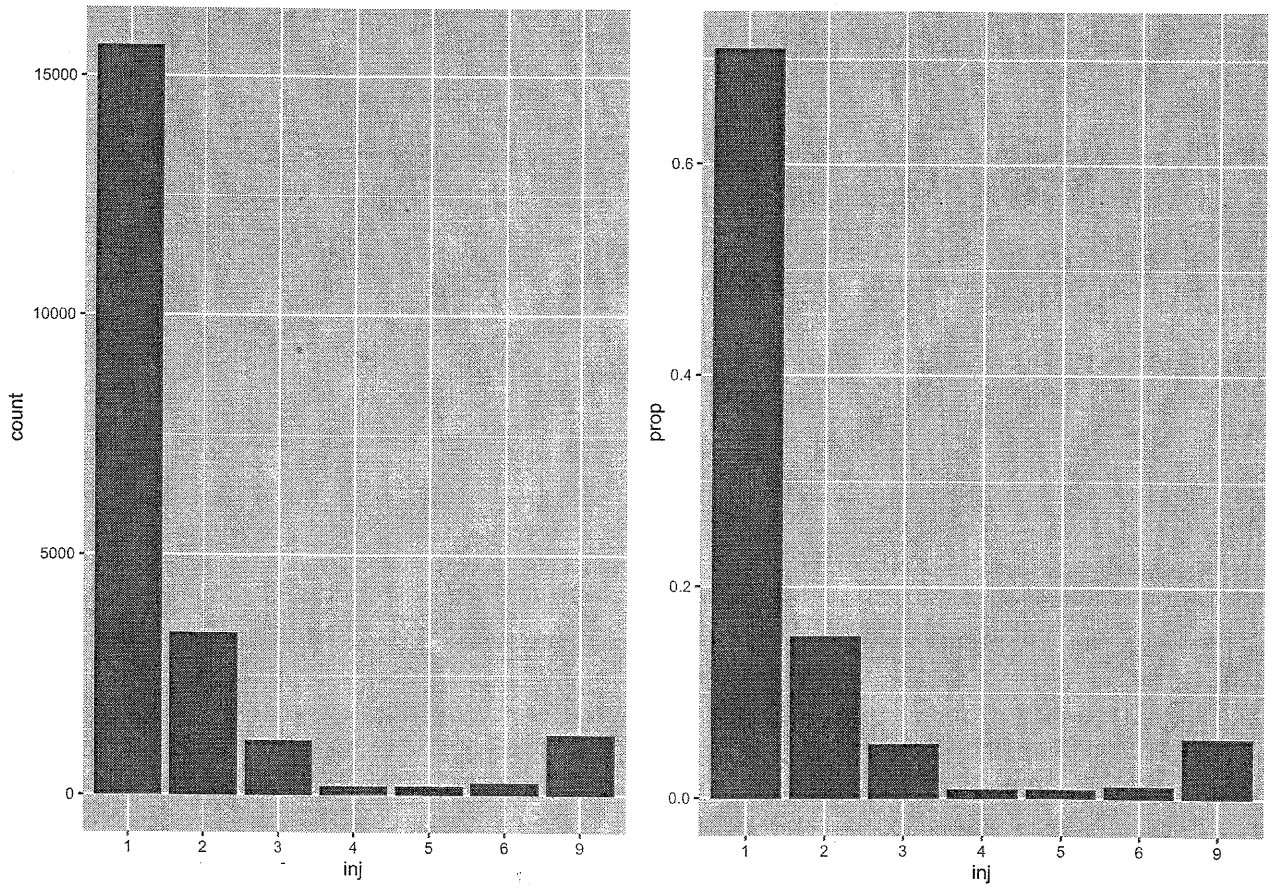


Figure 2.2.5: Bar charts of injury code in the personal injury insurance claims dataset.

2.2.2 Bivariate Data Exploration

Data exploration becomes even more intriguing and challenging when two or more variables are explicitly analyzed together. This has the important advantage of revealing relationships, patterns, and outliers which become apparent only when variables are considered in combination with one another. This subsection focuses on *bivariate data exploration*, where pairs of variables are investigated either numerically or graphically to identify potentially interesting relationships that can provide useful input for a predictive model. Of particular interest is the relationship between the target variable and each predictor variable in a given setting.

There are three types of bivariate combinations. The specific type will determine the most appropriate data exploration tools.

Numeric vs. numeric. The relationship between two numeric variables (one of which is usually the target variable) is typically visualized by a *scatterplot*, where values of the two variables are graphed on a two-dimensional plane, as we saw in Section 2.1. Such a plot often gives us a good sense of the nature of the relationship (e.g., increasing, decreasing, quadratic) between the two numeric variables and is one of the most commonly used graphical displays in data exploration.

Run CHUNK 9 to make two scatterplots, one for claim amount against operational time and one for the log of claim amount against operational time (see Figure 2.2.6). We have set `alpha` to a very small value due to the large number of overlapping observations. Both plots exhibit an increasing relationship, but the scatterplot for the log of claim amount displays a much more conspicuous upward sloping trend, indicating that the log of claim amount is approximately positively linear in operational time. This is a further manifestation of the merits of the log transformation in uncovering relationships that would otherwise be obscure.

Although a scatterplot itself is confined to depicting the relationship between only two numeric variables, the effect of a third, categorical variable can be incorporated and investigated by decorating the observations by color, shape, or other visual elements according to the levels assumed by this third variable. This way, we can visually inspect whether the relationship between the two numeric variables varies with the levels of the third, categorical variable. In statistical language, this phenomenon is known as *interaction* (see Section 3.2 for details) and is an important modeling issue to keep in mind when constructing an effective predictive model.

Now run CHUNK 10 to make a scatterplot for the log of claim amount against operational time, with the observations color-distinguished by legal representation (note that the `color` aesthetic is mapped to `legrep`); see Figure 2.2.7. The scatterplot shows that the two smoothed lines corresponding to the two levels of `legrep` have markedly different slopes and intercepts (keep in mind that we are on the log scale, so a small change in the intercept and slope can matter a lot on the original scale). In other words, the linear relationship between the log of claim amount and operational time depends materially on whether legal representation is present or not. We can also roughly tell the effect of legal representation on the (log of) claim amount: Injuries with legal representation (i.e., those with `legrep = 1`) tend to produce higher claim amounts, except when operational time is extraordinarily large (90 or higher), in which case legal representation does not seem to have a noticeable impact on claim amount. In Section 4.2, we will formally assess the extent of interaction and construct a model that properly takes the interaction effect into account.

```
# CHUNK 9
p1 <- ggplot(persinj, aes(x = op_time, y = amt)) +
  geom_point(alpha = 0.05) +
  geom_smooth(method = "lm", se = FALSE)
p2 <- ggplot(persinj, aes(x = op_time, y = log(amt))) +
  geom_point(alpha = 0.05) +
  geom_smooth(method = "lm", se = FALSE)
grid.arrange(p1, p2, ncol = 2)
```

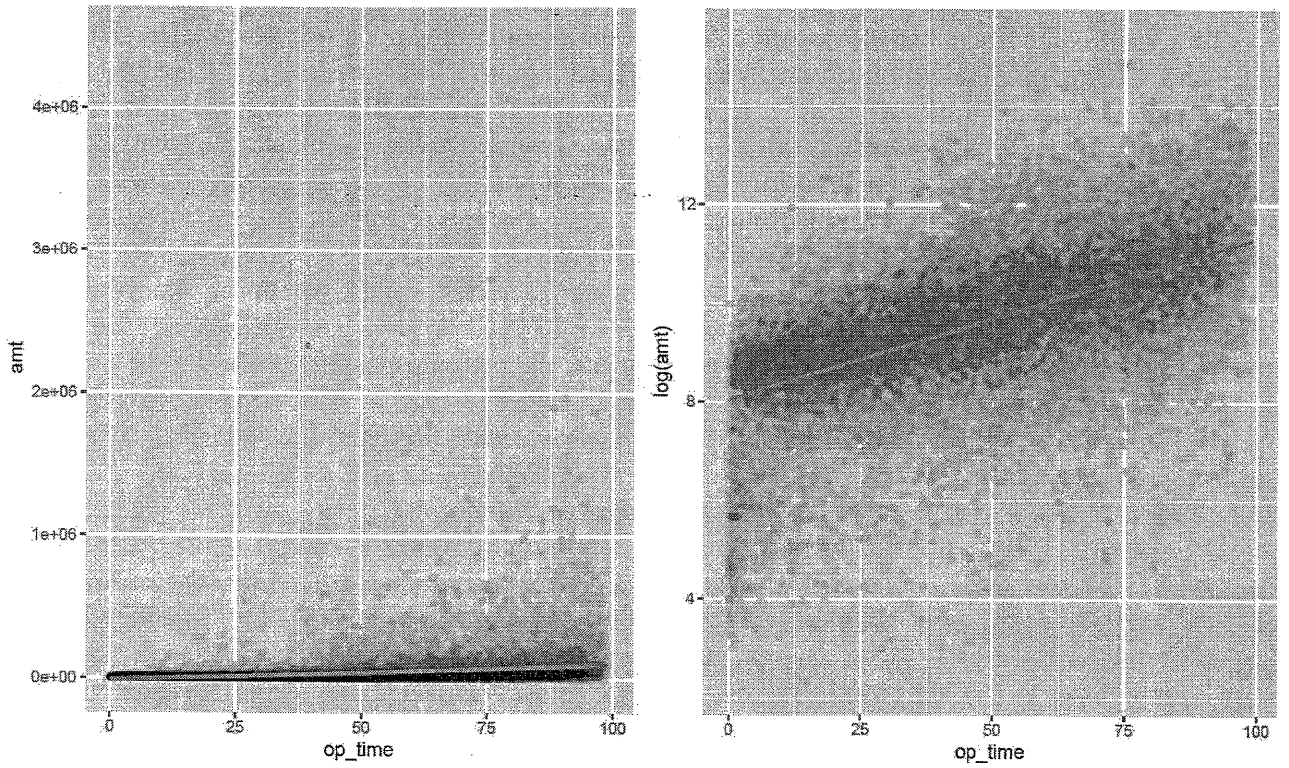


Figure 2.2.6: Scatterplots of claim amount (left) and the log of claim amount (right) against operational time in the personal injury insurance claims dataset.

```
# CHUNK 10
ggplot(persinj, aes(x = op_time, y = log(amt), color = legrep)) +
  geom_point(alpha = 0.25) +
  geom_smooth(method = "lm", se = FALSE)
```

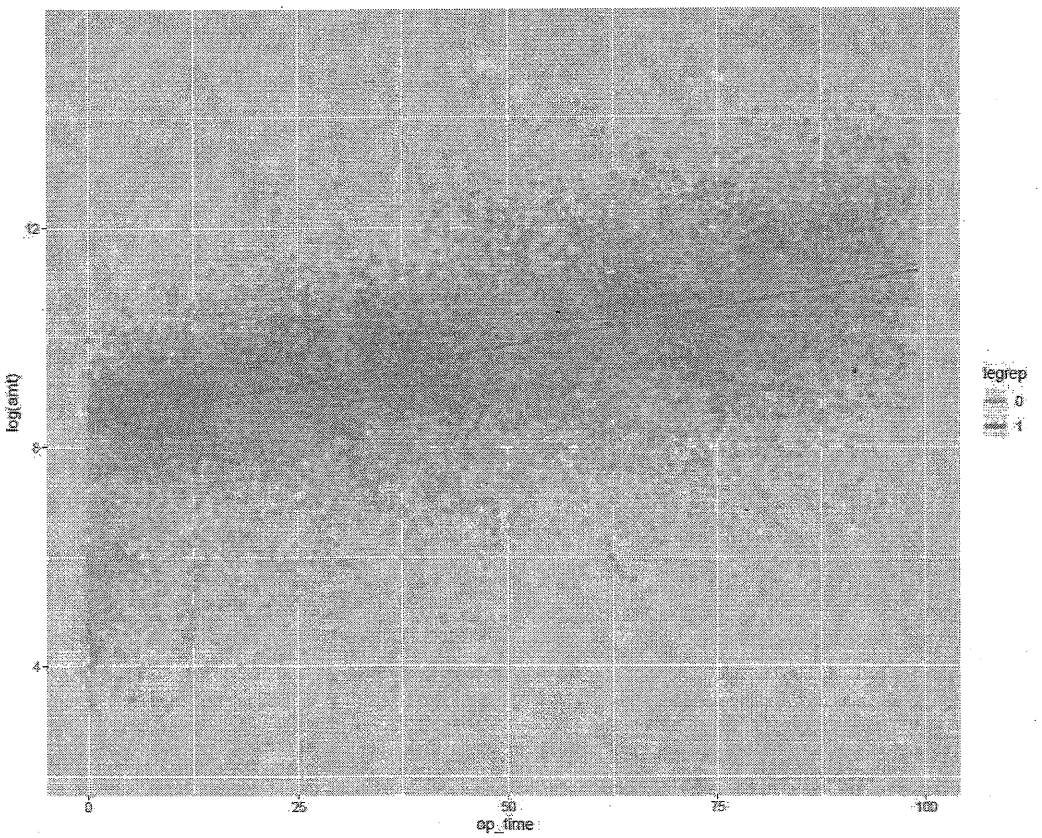


Figure 2.2.7: Scatterplot of the log of claim amount against operational time in the personal injury insurance claims dataset.

Numeric vs. categorical. To understand the interplay between a numeric variable and a categorical variable, it is best to make use of *split box plots*, where a series of box plots of the numeric variable indexed (or "split") by all possible levels of the categorical variable are made. In effect, we are looking at the conditional distribution of the numeric variable given different levels of the categorical variable.

Run CHUNK 11 to construct two split box plots for the log of claim amount, one split by injury code and one split by legal representation (Figure 2.2.8). The categorical variable that is used to split the numeric variable is simply entered into the x aesthetic and a collection of box plots of the numeric variable for each level of the categorical variable will be shown. The first split box plot makes it clear that the log of claim amount on average increases from injury code 1 to injury code 4, then decreases down the line to injury code 9. The split box plot by legal representation is in agreement with what we observed in Figure 2.2.7, showing that larger claim amounts are associated with the use of legal representation.

```
# CHUNK 11
p1 <- ggplot(persinj, aes(x = inj, y = log(amt), fill = inj)) +
  geom_boxplot()
p2 <- ggplot(persinj, aes(x = legrep, y = log(amt), fill = legrep)) +
  geom_boxplot()
grid.arrange(p1, p2, ncol = 2)
```

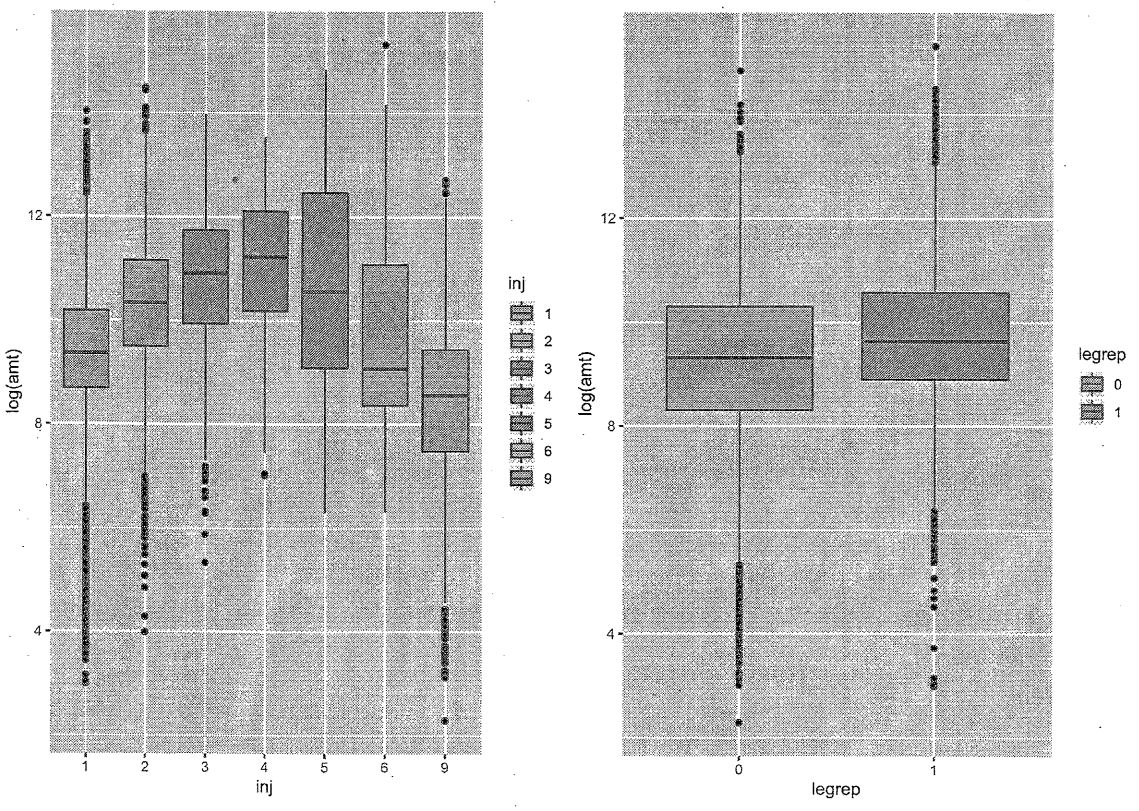


Figure 2.2.8: Two split box plots for the log of claim amount, one split by injury code (left) and one split by legal representation (right), in the personal injury insurance claims dataset.

In CHUNK 12, we split the log of claim amount by injury code (the x aesthetic), followed by

legal representation (the `fill` aesthetic), to view a three-way relationship (Figure 2.2.9). Now the effect of legal representation is even more pronounced: Regardless of the injury code, larger claim sizes tend to be injuries with legal representation, with the effect being most prominent for injuries of code 5 and code 9.

```
# CHUNK 12
ggplot(persinj, aes(x = inj, y = log(amt), fill = legrep)) +
  geom_boxplot()
```

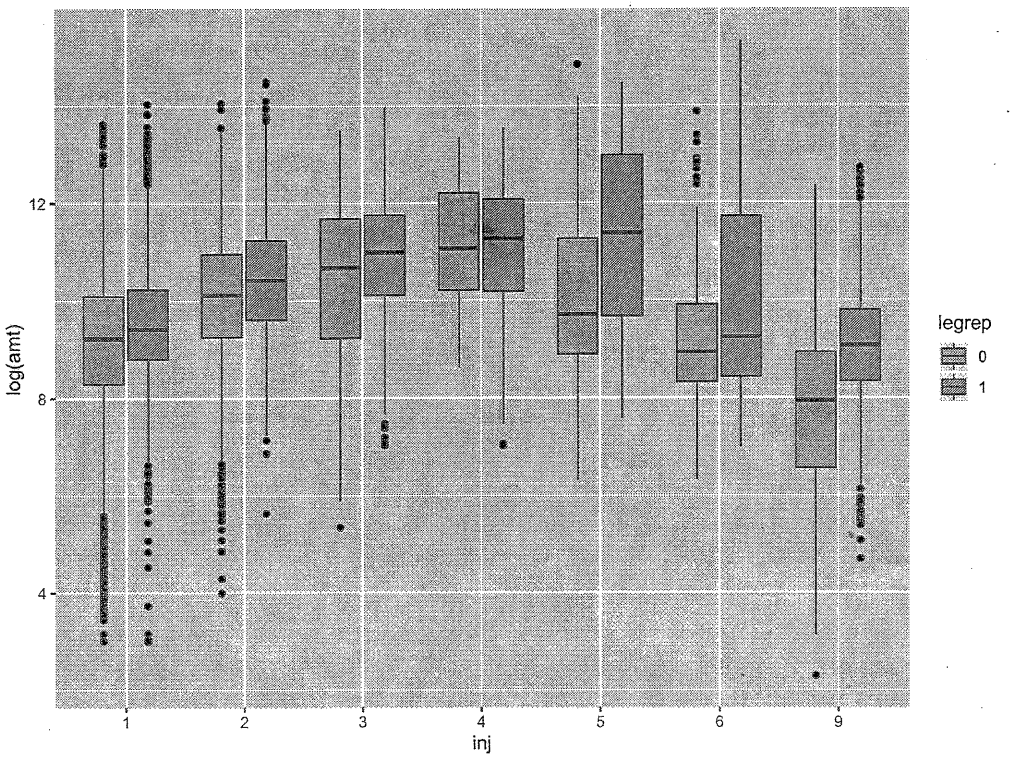


Figure 2.2.9: Box plots of the log of claim amount split by injury code and legal representation in the personal injury insurance claims dataset.

Although not as effective as split box plots, histograms can also be adapted to visualize the distribution of a numeric variable split by a categorical variable. They can either be histograms stacked on top of one another (using the `fill` aesthetic) to highlight the contribution of each categorical level to the overall distribution of the numeric variable, or faceted histograms placed side by side for comparison (using the `facet_wrap()` function). Run CHUNK 13 to produce both types of histograms split by legal representation (Figure 2.2.10).

```
# CHUNK 13
p1 <- ggplot(persinj, aes(x = log(amt), fill = legrep)) +
  geom_histogram(bins = 50)
p2 <- ggplot(persinj, aes(x = log(amt))) +
  geom_histogram(fill = "blue") +
  facet_wrap(~ legrep)
grid.arrange(p1, p2)
```

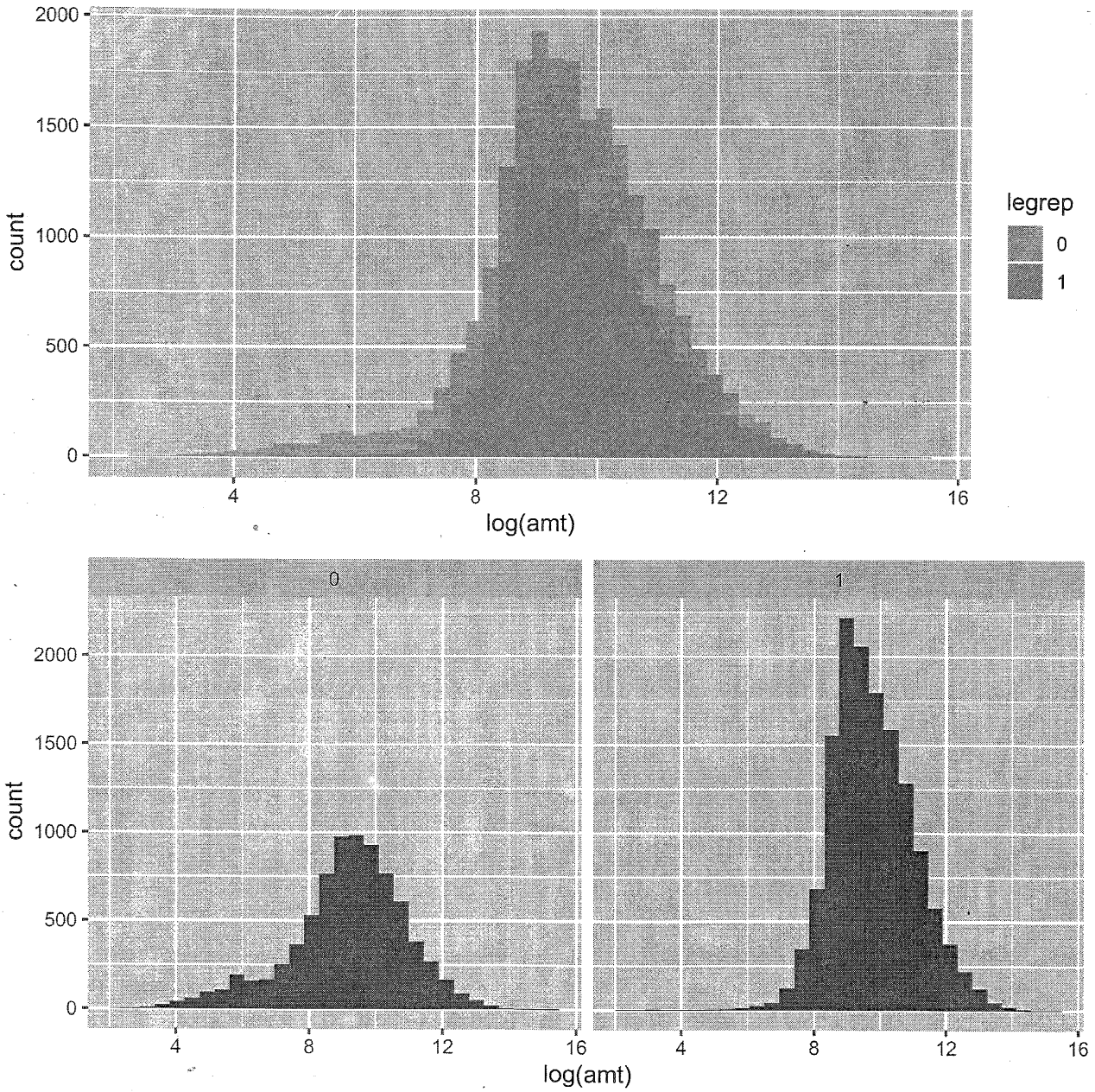


Figure 2.2.10: Stacked (above) and faceted (below) histograms of the log of claim amount in the personal injury insurance claims dataset.

Categorical vs. categorical. When examining a pair of categorical variables, it is often useful to construct a two-way frequency table showing the number of observations for every combination of the two variables' levels using the `table()` function. When two arguments are supplied to the `table()` function, the first argument will correspond to the rows of the two-way frequency table while the second argument will correspond to its columns. Run CHUNK 14 to make a two-way frequency table for injury code crossed with legal representation.

```
# CHUNK 14
table(persinj$legrep, persinj$inj)

##
##          1    2    3    4    5    6    9
##   0   5571 1152  374   56   85  121  649
##   1  10067 2224  759  133  103  135  607
```

To visualize the distribution of a categorical variable split by another categorical variable, *split bar charts* can be of use. Run CHUNK 15 to produce three bar charts for injury code split by legal representation (see Figure 2.2.11):

- The first bar chart has counts within each level of injury code color-distinguished by legal representation.
- The second bar chart has counts within each level of injury code separated according to legal representation and placed side by side (note the argument `position = "dodge"`).
- In the third bar chart, the relative proportions of injuries with and without legal representation for each level of injury code are shown (note the argument `position = "fill"`). This makes it easy to compare proportions across different injury codes, although we lose the ability to see the number of injuries in each code.

A cursory glance at the filled bar chart shows that there is a higher proportion of injuries with legal representation for codes 1 to 4 than for codes 5, 6, and 9.

```
# CHUNK 15
p1 <- ggplot(persinj, aes(x = inj, fill = legrep)) +
  geom_bar()
p2 <- ggplot(persinj, aes(x = inj, fill = legrep)) +
  geom_bar(position = "dodge")
p3 <- ggplot(persinj, aes(x = inj, fill = legrep)) +
  geom_bar(position = "fill") +
  ylab("Proportion")
grid.arrange(p1, p2, p3, ncol = 2)
```

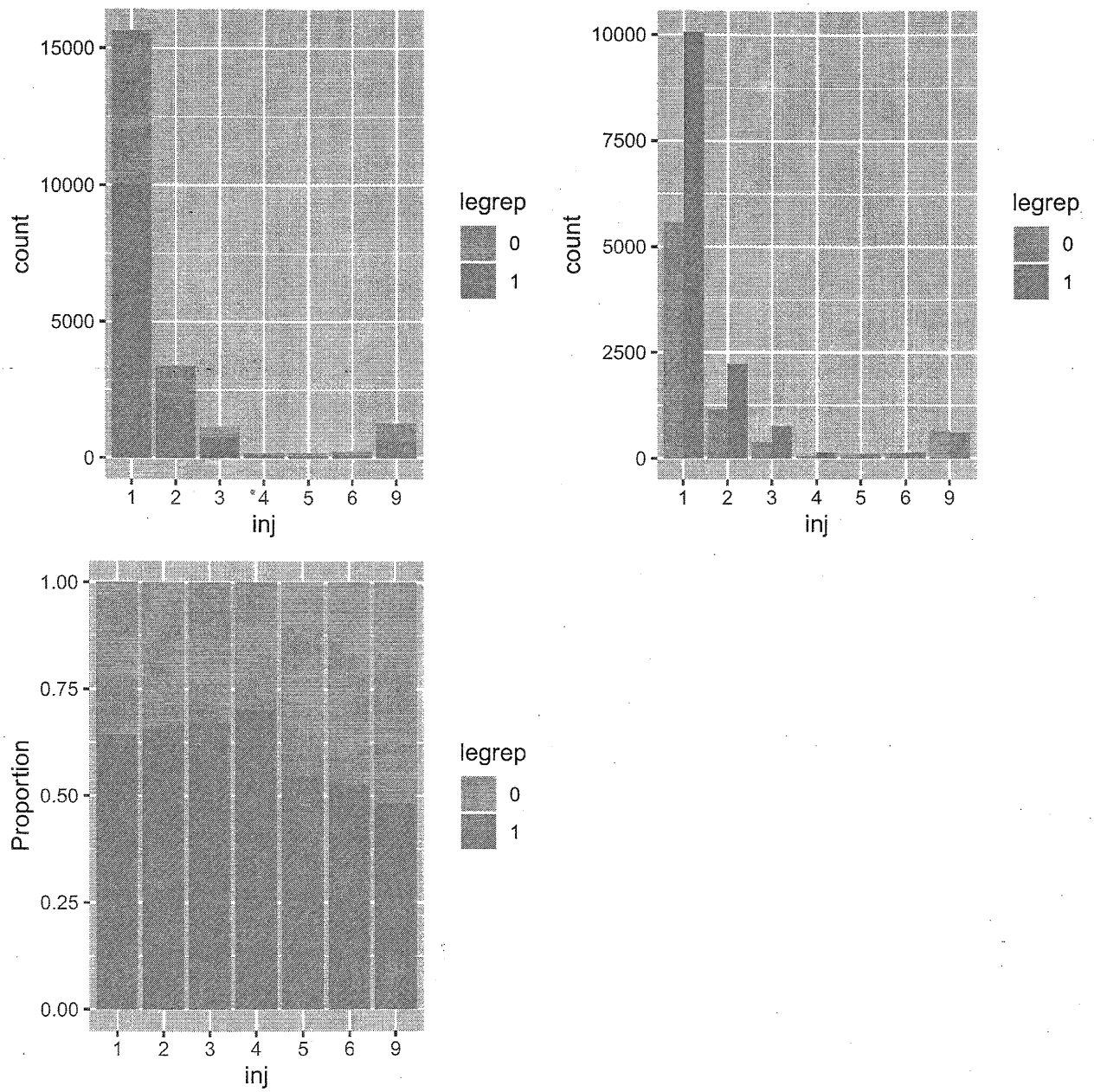


Figure 2.2.11: Stacked (top left), dodged (top right), and filled (bottom left) bar charts for injury code split by legal representation in the personal injury insurance claims dataset.

2.3 End-of-Chapter Practice Problems

Problem 2.3.1. (Small differences in code, large differences in output!) Consider the personal injury insurance dataset in Section 2.2 again and the following chunks of R commands (which look similar!):

```
persinj <- read.csv("persinj.csv")
persinj$inj <- as.factor(persinj$inj)
persinj$legrep <- as.factor(persinj$legrep)
library(ggplot2)

# CHUNK 1
ggplot(persinj, aes(x = inj, color = legrep)) + geom_bar()

# CHUNK 2
ggplot(persinj, aes(x = inj, fill = legrep)) + geom_bar()

# CHUNK 3
ggplot(persinj, aes(x = inj)) + geom_bar(fill = legrep)

# CHUNK 4
ggplot(persinj, aes(x = inj)) + geom_bar(aes(fill = legrep))
```

Make a guess of what each chunk of code does. Then run the code in R and see the output.

Problem 2.3.2. (Data exploration: Univariate and bivariate) The `ggplot2` package comes with a dataset named `diamonds` that contains the prices and other attributes of approximately 54,000 diamonds. To load the dataset, use the following commands:

```
library(ggplot2)
data(diamonds)
```

(a) Determine the number of observations and variables in the `diamonds` dataset.

With the aid of appropriate graphical displays and/or summary statistics, complete the following tasks.

(b) Perform univariate exploration of the price of diamonds (`price`) and the quality of the cut (`cut`). Determine if any of these two variables should be transformed and, if so, what transformation should be made. Do your recommended transformation(s), if any, and delete the original variable(s).

(c) Explore the relationship between the price of diamonds and the weight of diamonds (`carat`).

(d) Explore the relationship between the price of diamonds and the quality of the cut.

(e) Reconcile the apparent contradiction between what you get in parts (c) and (d).

Solutions

Solution to Problem 2.3.1. Although the four chunks of code look similar, they produce drastically different output.

- *CHUNK 1:* Here we are making a bar chart for injury code with the *boundary* (not the interior) of the vertical bars color-coded according to legal representation. As you can see, the colors are hardly perceptible.
- *CHUNK 2:* This is similar to CHUNK 1, except that this time it is the *interior* of the vertical bars that is color-coded according to legal representation. The output is the same as the left panel of Figure 2.2.11.
- *CHUNK 3:* This chunk of code does not work (try to run it in R and you will get an error!). The reason is that the `fill` argument (not `fill` aesthetic) of the `geom_bar()` function is mapped to a variable (`legrep` here) instead of a constant (e.g., `"blue"`). This is the opposite of the mistake discussed on page 70.
- *CHUNK 4:* This chunk of code generates the same output as CHUNK 2. Instead of putting the `fill` aesthetic in the `ggplot()` call, it is placed inside the `geom_bar()` function.

□

Solution to Problem 2.3.2. (a) The number of rows of the `diamonds` dataset can be obtained by the `nrow()` function:

```
nrow(diamonds)

## [1] 53940
```

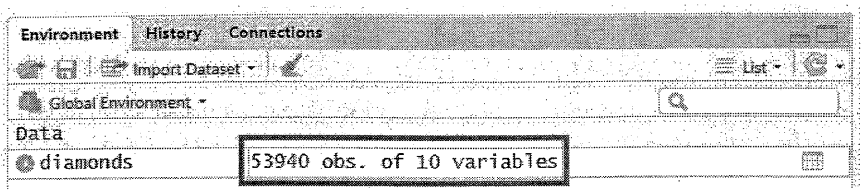
To get the number of variables, we can first extract the column names of the dataset via the `colnames()` function, then apply the `length()` function to calculate its length:

```
length(colnames(diamonds))

## [1] 10
```

More efficiently, we can apply the `dim()` function to `diamonds` to get both of its row and column dimensions.

```
dim(diamonds)

## [1] 53940    10
```

Remark. You can also see the number of observations and variables of the `diamonds` dataset directly from the environment pane:



However, it is desirable to know how to write code to extract these attributes of a dataset.

(b)

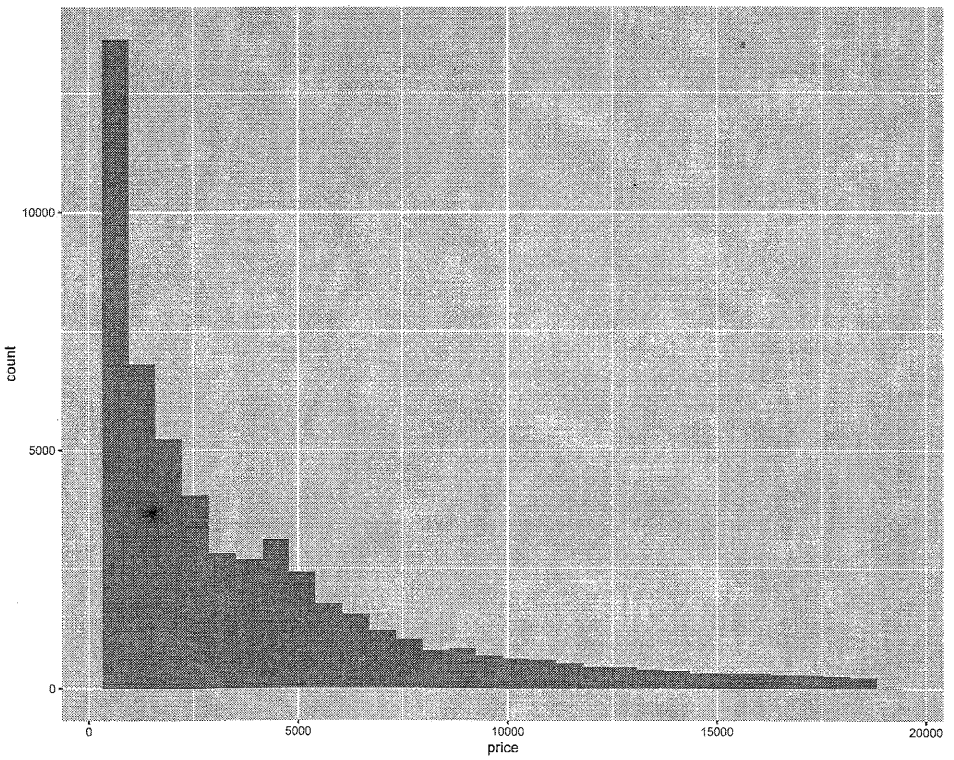
- The `price` variable is a (positive and technically continuous) numeric variable. Let's use the `summary()` function to learn about its numeric statistics.

```
summary(diamonds$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     326     950    2401    3933    5324   18823
```

The variable ranges from 326 to 18,823 and its mean is much higher than its median, an indication of its pronounced right skewness. This is confirmed by the histogram below.

```
ggplot(diamonds, aes(x = price)) +
  geom_histogram()
```

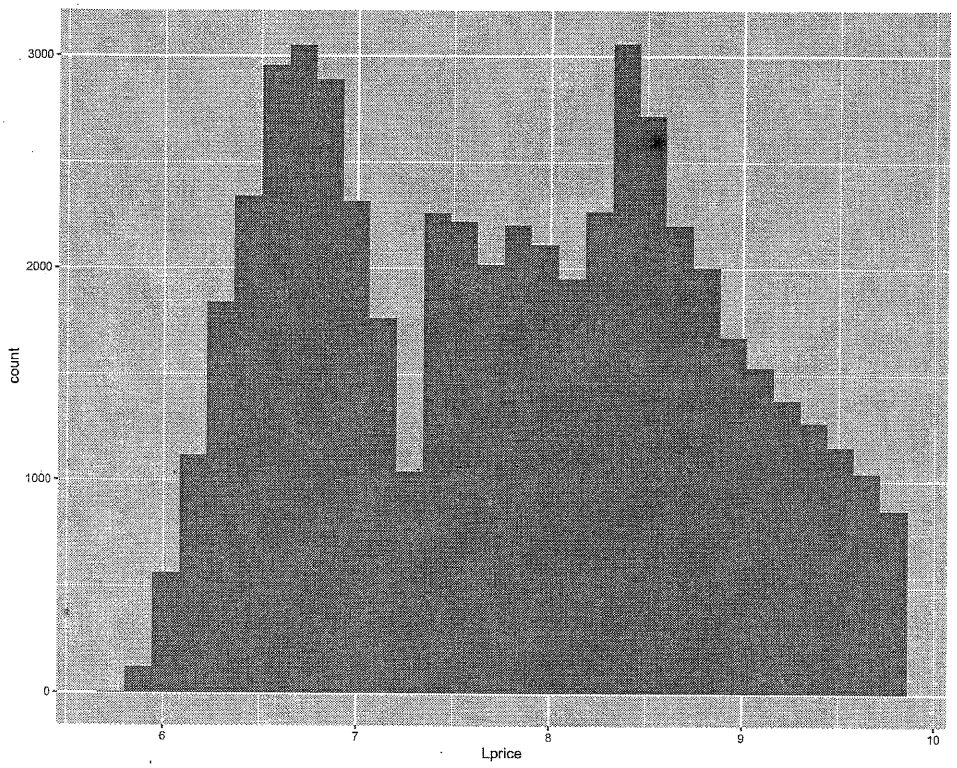


To deal with the right skewness of `price`, we can use a log transformation. The following commands create the log-transformed `price` and delete the original `price` variable (recall what you learned in Section 1.3).

```
diamonds$Lprice <- log(diamonds$price)
diamonds$price <- NULL
```

The resulting histogram shows that the distribution of the log-transformed `price` is closer to symmetric, although it is not bell-shaped.

```
ggplot(diamonds, aes(x = Lprice)) +
  geom_histogram()
```



Remark. For both histograms, you can experiment with different values of the `bins` parameter.

- The `cut` variable is a 5-level categorical variable; its levels are `"Fair"`, `"Good"`, `"Very Good"`, `"Premium"`, and `"Ideal"`.

```
levels(diamonds$cut)
```

```
## [1] "Fair"      "Good"      "Very Good" "Premium"   "Ideal"
```

The following table shows the counts and percentage for each level:

```
table(diamonds$cut)
```

```
##
##      Fair      Good Very Good   Premium     Ideal
##      1610      4906     12082     13791     21551
```

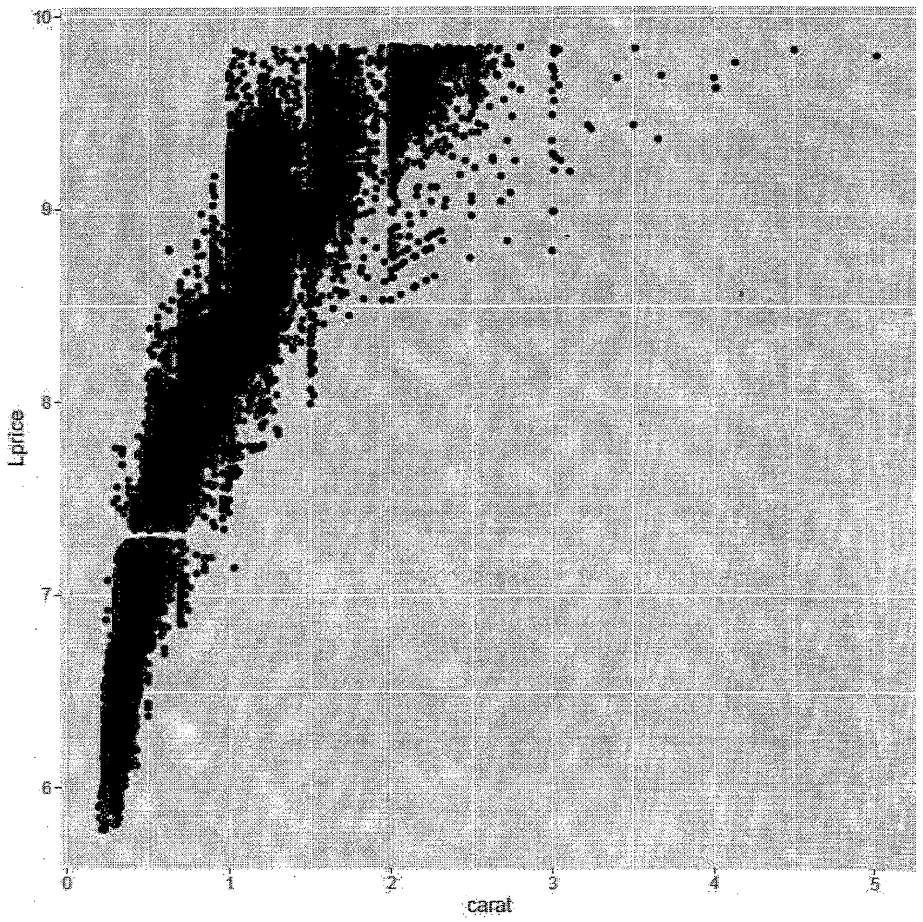
```
table(diamonds$cut)/nrow(diamonds)
```

```
##
##       Fair       Good  Very Good    Premium      Ideal
## 0.02984798 0.09095291 0.22398962 0.25567297 0.39953652
```

The number of observations increases from `"Fair"` to `"Ideal"`. About 40% of the diamonds are of `"Ideal"` quality.

Since `cut` is a categorical variable, numeric transformations such as the log transformation are not applicable and so we would prefer to leave it as it is.

(c) Because `Lprice` and `carat` are both numeric variables, a scatterplot for the two variables is appropriate for exploring their relationship. The scatterplot shows that the two variables are strongly positively related; the heavier the diamond, the more expensive it is, conforming to our intuition.

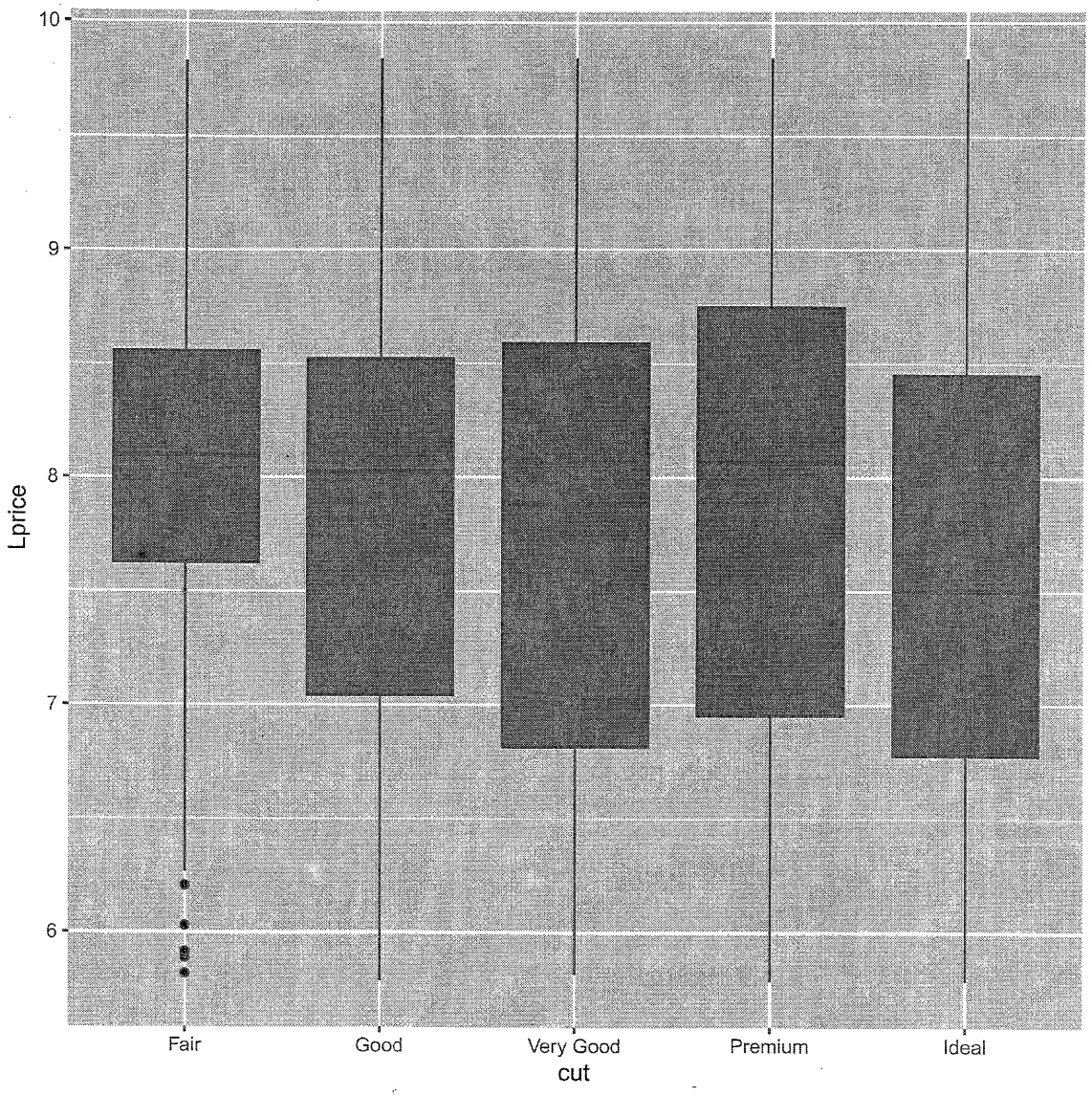


Remark. (i) You can add the `alpha` argument to the `geom_point()` function to reduce the amount of overlapping.

(ii) If you plot `Lprice` against the log of `carat`, the resulting relationship is very close to linear.

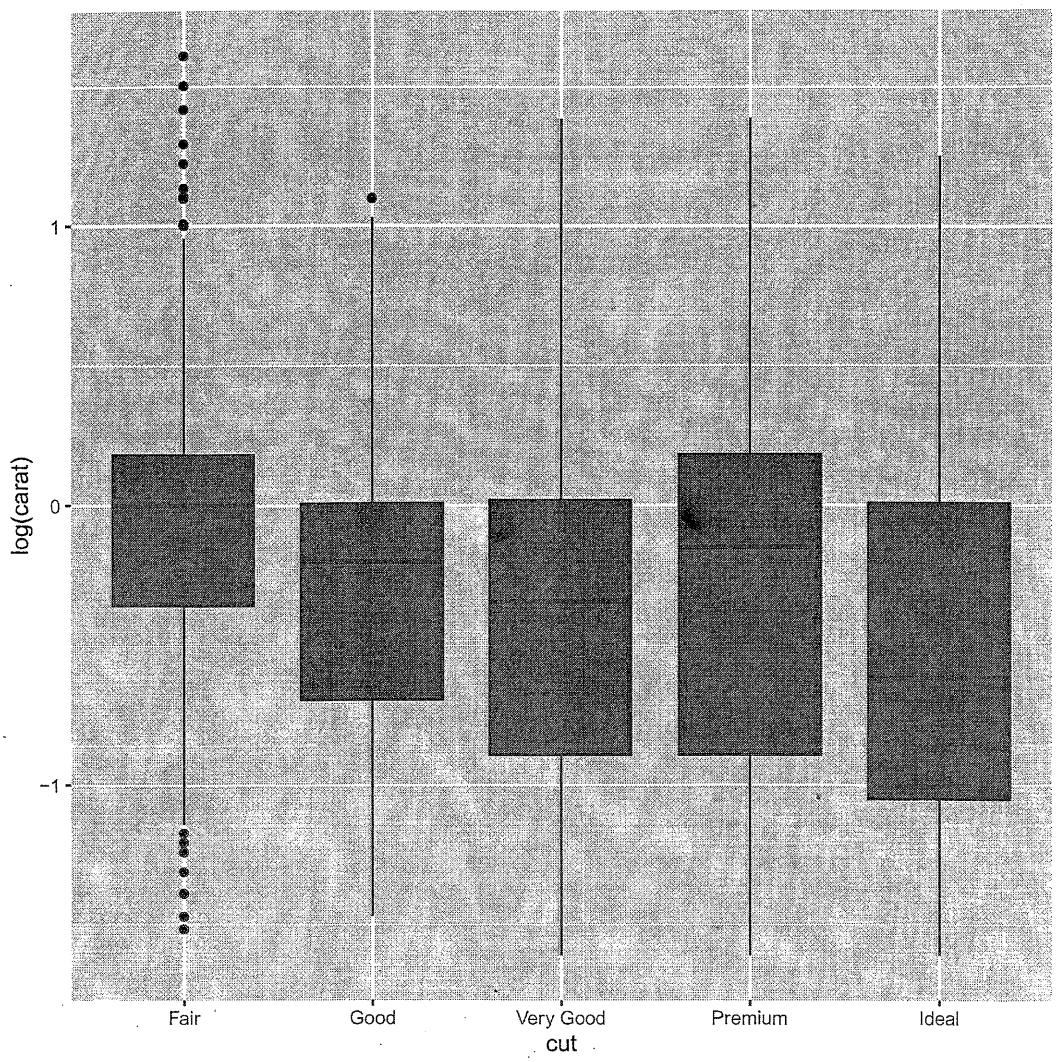
(d) As `cut` is a categorical variable, a split box plot for `Lprice` broken by the levels of `cut` is appropriate for exploring their relationship. The split box plot, however, suggests the counter-intuitive idea that diamonds of a higher quality tend to be cheaper (though only slightly). How can this be the case?

```
ggplot(diamonds, aes(x = cut, y = Lprice)) +
  geom_boxplot(fill = "red")
```



(e) To reconcile the contradiction between parts (c) and (d), one can look at the relationship between `carat` and `cut`. Again, as `carat` is numeric and `cut` is categorical, a split box plot for `carat` split by `cut` will serve our purpose.

```
ggplot(diamonds, aes(x = cut, y = log(carat))) +
  geom_boxplot(fill = "red")
```



The split box plot shows that the weight of a diamond tends to drop as the quality of the cut becomes higher (`"Premium"` is an exception). According to part (c), the `carat` is an important predictor of `Lprice`, so the findings in part (d) may be a result of the negative relationship between `carat` and `cut`—higher quality diamonds may be less pricey because they weigh less.

Remark. In Chapter 3, we will explore the apparent inconsistency between what you get in parts (c) and (d) using linear models.

□