Lösungen zu den Aufgaben

1. **Aufgabe**

   In dieser Aufgabe betrachten wir einige Probleme, die bei der Praxis der Datenanalyse auftauchen.

   Importieren Sie zunächst den Datensatz `diamonds`, der Teil des Tidyverse ist:

```
library(tidyverse)
data(diamonds)
```

Aufgaben

1. Zählen Sie die *fehlenden Werte* pro Spalte!
2. Welcher *Anteil von Werten fehlt insgesamt* (in der ganzen Tabelle)?
3. *Ersetzen* Sie fehlende Werte (numerischer Variablen) durch den Mittelwert!
4. Welche Variable zeichnet sich durch die höchste *Schiefe* aus?
5. Definieren wir "*Ausreißer*" als einen Wert, der mehr als 3SD-Einheiten vom Mittelwert entfernt ist. Wie viele Ausreißer gibt es für `price`?
6. Gibt es *Dubletten*? Wenn ja, wie viele?
7. Gibt es *Variablen*, die *konstant* oder fast konstant sind? Konstant ist hier definiert als (fast) keine Variabilität.

Lösung

Es gibt einige "R-Komfortfunktionen" für deskriptive Statistiken, die viele typische Kennwerte in einem Schritt ausgeben.

Ein Beispiel ist die Funktion `describe_distribution()` aus dem Paket `{{easystats}}`:

```
library(easystats)
describe_distribution(diamonds)
```

Variable	Mean	SD	IQR	Min	Max	Skewness	Kurtosis
carat	0.7979397	0.4740112	0.64	0.2	5.01	1.11664592	1.2566353
depth	61.7494049	1.4326213	1.50	43.0	79.00	-0.08229403	5.7394146
table	57.4571839	2.2344906	3.00	43.0	95.00	0.79689585	2.8018569
price	3932.7997219	3989.4397381	4374.75	326.0	18823.00	1.61839528	2.1776958
x	5.7311572	1.1217607	1.83	0.0	10.74	0.37867634	-0.6181607
y	5.7345260	1.1421347	1.82	0.0	58.90	2.43416672	91.2145572
z	3.5387338	0.7056988	1.13	0.0	31.80	1.52242256	47.0866193

Fehlende Werte pro Spalte

Es gibt keine fehlenden Werte.

Anteil fehlender Werte

Null

Fehlende Werte ersetzen

Erzeugen wir der Übung halber ein paar fehlende Werte, von R mit `NA` bezeichnet. Die können wir dann ersetzen.

Wenn der Preis größer ist als 4000, dann soll der Preis auf `NA` gesetzt werden, ansonsten bleibt der Preis, wie er ist.

```
diamonds2 <-
  diamonds %>%
  mutate(price = ifelse(price > 4000, yes = NA, no = price))
```

Jetzt ersetzen wir die fehlenden Werte durch den Mittelwert:

```
diamonds3 <-
  diamonds2 %>%
  mutate(price = replace_na(price, 3933))
```

Betrachten wir den Effekt des Ersetzens der fehlenden Werte:

```
diamonds3 %>%
  describe_distribution()
```

Variable	Mean	SD	IQR	Min	Max	Skewness	Kurtosis	
carat	0.7979397	0.4740112	0.64	0.2	5.01	1.11664592	1.2566353	5
depth	61.7494049	1.4326213	1.50	43.0	79.00	-0.08229403	5.7394146	5
table	57.4571839	2.2344906	3.00	43.0	95.00	0.79689585	2.8018569	5
price	2417.2545235	1383.7287911	2983.00	326.0	4000.00	-0.08494006	-1.7087434	5
x	5.7311572	1.1217607	1.83	0.0	10.74	0.37867634	-0.6181607	5
y	5.7345260	1.1421347	1.82	0.0	58.90	2.43416672	91.2145572	5
z	3.5387338	0.7056988	1.13	0.0	31.80	1.52242256	47.0866193	5

Oh nein! In diesem Fall hat unser Ersetzen (“Imputieren”) die zentralen Kennwerte der Verteilung (Lage und Streuung) massiv verändert. Das ging ins Auge! Ein Beispiel für eine Situation, in der Imputieren *nicht* funktioniert. Der Grund ist, dass *bestimmte* Werte (systematisch) fehlen, wir aber durch das Ersetzen mit dem Mittelwert davon ausgingen, dass die Daten *komplett zufällig* fehlten. Leider wissen wir nicht ohne Weiteres, ob Daten systematisch oder zufällig fehlen. Die

Moral von der Geschicht: Fehlende Werte können Probleme bereiten. Am besten man hat keine (Probleme und fehlende Werte) :-)

An anderer Stelle beschäftigen wir uns vielleicht ausführlicher mit dieser Frage. In der einschlägigen Literatur finden sich viele Erläuterungen.

Schiefe

Neben `y` hat auch der Preis eine massive Schiefe aufzuweisen.

Ausreißer

Der Mittelwert von `price` liegt ca. bei 4000, die SD auch.

```
diamonds %>%
  mutate(is_ausreisser = ifelse(price > 4000+3*4000, TRUE, FALSE)) %>%
  count(is_ausreisser)
```

```
## # A tibble: 2 × 2
##   is_ausreisser     n
##   <lgl>         <int>
## 1 FALSE         52798
## 2 TRUE           1142
```

Dubletten

```
diamonds %>%
  n_distinct()
```

```
## [1] 53794
```

Keine Dubletten.

Aber lassen wir nur die vier “C-Variablen” einfließen (`carat`, `cut`, `color`, `clarity`), dann finden sich plötzlich viele Dubletten:

```
diamonds %>%
  select(carat, cut, color, clarity) %>%
  n_distinct()
```

```
## [1] 13928
```

Quelle

Konstante Variablen

Unsere Tabelle oben zeigte, dass alle Variablen Streuung haben.