Universität Bielefeld

(Fakultät für Wirtschaftswissenschaften)

Abschlussarbeit 2

(im Studiengang Wirtschaftswissenschaften)

Modul: Computergestützte Methoden

vorgelegt von *Kevin Wilm Schliewe und Timon Gerner*

Matrikel-Nr.: *4249212 und 2886213*

Geprüft durch *Lennart Oelschläger und J.-Prof. Dr. Timo Adam*

Bielefeld, im Februar 2024

1 Einleitung

Im Rahmen der Veranstaltung computergestuetzte Methoden haben wir uns mit den Thematiken Programmieren und Visualisieren in R auseinandergesetzt.

Um unser in der Vorlesung erlangtes Wissen um RStudio zu vertiefen, haben wir aufgetragen bekommen, uns mit Funktionen und Wahrscheinlichkeiten zu beschaeftigen. Konkret ging es um Wichtelgeschenke und die Moeglichkeit der Zuordnung des eigenen Geschenkes.

In Aufgabe 3.3 wurde gefordert, dass wir unseren Code kommentieren und die Ein- und Ausgaben unmissverstaendlich beschrieben werden. Dies fuehrte dazu, dass wir unseren Code entsprechend angepasst haben. Wir gehen deshalb in diesem Dokument nicht weiter auf die Bearbeitung von 3.3 ein, da diese bereits in den Screenshots einsehbar ist.

In Aufgabe 3.5 und Aufgabe 4 beschaeftigen wir uns mit einem Datensatz aus den USA: 01. BikeshareDatensatz Quelle 1 und erstellen einige Grafiken zur besseren Visualisierung der Zusammenhaenge.

Wir haben unser ganzes Projekt und den zugehoerigen Code auf GitHub. Der Link dazu ist hier: 02. GitHub Quelle 2.

Sehr hilfreich zur Bearbeitung der Aufgaben war die Dokumentation von R. 03. R Dokumentation Quelle 3

Dieses Dokument kann nur so gut aussehen, da wir erneut die Vorlage von Chloé Goupy verwendet haben, ein großes Dankeschön geht erneut an sie.
04. LaTeX-Dokument Quelle 4

Inhaltsverzeichnis

1

Programmieren in R

1 Aufgabe 3.1: Das spezifische Wichtel Unglueck

Diese Quellen haben bei der Bearbeitung sehr geholfen und lieferten stets Orientierung.
05. Oelschläger: R Skript 1 Quelle 5
06. Oelschläger: R Skript 2 Quelle 6
07. Oelschläger: R Skript 3 Quelle 7

Wie Wahrscheinlich ist es eigentlich, dass unter zehn Personen, eine dabei ist, die ihr eigenes Wichtelgeschenk zieht? Um ein genaueres Ergebnis zu bekommen, simulieren wir dieses Szenario im nachfolgenden Code eintausend mal.

```r
# Berechnung: 1 Person unter 10 bekommt eigenes Wichtelgeschenk
# Rueckgabe: Prozentangabe Wahrscheinlichkeit (ungefaehr 63%)
#
# Initialisiere:
anzahl_personen <- 10 # Anzahl der Personen - Integer
iterationen <- 1000 # Anzahl der Iterationen - Integer
zaehler <- 0 # Initialisiere Zähler für Person-zieht-eigenes-Geschenk-Fälle - Integer

for (i in 1:iterationen) { # For-Schleife für Iterationen - hier 1000
  # Zufällige Zuordnung von Geschenken zu Personen
  zuordnung <- sample(1:anzahl_personen, size = anzahl_personen, replace = FALSE)
  # beim ziehen wird nicht zurückgelegt - replace=FALSE

  # Überprüfen, ob mindestens eine Person ihr eigenes Geschenk gezogen hat
  if (any(zuordnung == 1:anzahl_personen)) {
    zaehler <- zaehler + 1
  }
}

p <- zaehler / iterationen # Wahrscheinlichkeit berechnen
prozent <- p * 100 # Wahrscheinlichkeit in Prozent umrechnen

# Ausgabe anzeigen
cat("Die Wahrscheinlichkeit, dass mindestens eine Person ihr eigenes Geschenk zieht, beträgt:", prozent, "%\n")
```

Fig. 1.1 Code von A3.1

```
>
> # Ausgabe anzeigen
> cat("Die Wahrscheinlichkeit, dass mindestens eine Person ihr eigenes Geschenk zieht, beträgt:", prozent, "%\n")
Die Wahrscheinlichkeit, dass mindestens eine Person ihr eigenes Geschenk zieht, beträgt: 64.3 %
```

Fig. 1.2 Ausgabe von 3.1

Die Simulation gibt aus, dass in etwa vierundsechzig Prozent der Faelle mindestens eine Person unter zehn dabei ist, die ihr eigenes Wichtelgeschenk zurueckbekommt.

2 Aufgabe 3.2: Das unspezifische Wichtel Unglueck

Hier loesen wir uns vom Spezialfall bei dem unter zehn Personen mindestens eine ihr eigenes Geschenk zurueckbekommt. Stattdessen wollen wir unter frei waehlbaren Parametern die Wahrscheinlichkeit des Ungluecks bestimmen.

```r
# Funktion zur Berechnung des Wichtel Ungluecks von n Personen die mitmachen
# und k Personen die ihr eigenes Geschenk zurueckbekommen
#
# Argumente:
# n (numeric): Eine Zahl zwischen 0 und 1000, Anzahl Teilnehmer
# k (numeric): Eine Zahl <= n, Anzahl Teilnehmer die ihr eigenes Geschenk ziehen
# iterationen (numeric): Eine Zahl zwischen 1 und 1000, Anzahl wie oft wir unser
# Experiment wiederholen
#
#
wichtel_unglueck <- function(n = NULL, k=NULL, iterationen=1000) {
  # Initialisiere Zähler für Geschenk-wird-selbst-gezogen-Fälle
  zaehler <- 0

  #Exceptions/Sonderfälle

  # n wird nicht übergeben
  if(is.null(n)) {
    stop("Das Argument für n wurde nicht übergeben, Wichtel_unglueck wird nun mit n=100 ausgeführt.")
  }

  # k wird nicht übergeben
  if(is.null(k)) {
    stop("Das Argument für n wurde nicht übergeben, Wichtel_unglueck wird nun mit k=10 ausgeführt.")
  }
  # falls n = k = 1 übergeben wird
  if(k==n& n==1) {
    return(1)
  }
  # falls k<0 übergeben wird
  if(k<0) {
    stop("Für k wurde weniger übergeben, bitte übergeben Sie mindestens k=1.")
  }
  # falls n = 0 übergeben wird
  if(n<=0) {
    stop("Für n wurde 0 oder weniger übergeben, bitte übergeben Sie mindestens n=1.")
  }
```

Fig. 1.3 Code von A3.2 Teil 1

```r
# falls iterationen = 0 übergeben wird
if(iterationen <= 0) {
  stop("Für das Argument iterationen wurde 0 oder weniger übergeben")
}

# falls k, n oder iterationen Buchstaben übergeben werden
if(is.character(n)) {
  stop("Der Funktion wurde für das Argument n mindestens 1 Buchstabe übergeben, bitte geben Sie ausschließlich Zahlen ein.")
}

if(is.character(k)) {
  stop("Der Funktion wurde für das Argument k mindestens 1 Buchstabe übergeben, bitte geben Sie ausschließlich Zahlen ein.")
}

if(is.character(iterationen)) {
  stop("Der Funktion wurde für das Argument iterationen mindestens 1 Buchstabe übergeben, bitte geben Sie ausschließlich Zahlen ein.")
}
# falls k>n übergeben wird
if(k>n) {
  stop("k darf nicht größer als n sein.")
}

# For-Schleife für die angegebene Anzahl von Iterationen
for (i in 1:iterationen) {
  # Zufällige Zuordnung von Geschenken zu Personen
  zuordnung <- sample(1:n, size = n, replace = FALSE)

  # Überprüfen, ob genau k oder mehr als k Personen ihr eigenes Geschenk gezogen haben
  if (sum(zuordnung == 1:n) >= k) {
    zaehler <- zaehler + 1
  }
}
p <- zaehler / iterationen   # Wahrscheinlichkeit ausrechnen

prozent <- p * 100 # Wahrscheinlichkeit in Prozent umrechnen

# Ausgabe anzeigen
cat("Die Wahrscheinlichkeit, dass mindestens", k, "Person(en) unter", n, "Person(en) ihr eigenes Geschenk ziehen, beträgt:", prozent, "%\n")

return(p) # Rückgabe der Wahrscheinlichkeit
```

Fig. 1.4 Code von A3.2 Teil 2

Wir haben hier beispielhaft die Faelle n=100, k=2, n=200, k=1 und n=500,k=1 ausgefuehrt.

```
> wichtel_unglueck(n=5, k=1)
Die Wahrscheinlichkeit, dass mindestens 1 Person(en) unter 5 Person(en) ihr eigenes Geschenk ziehen, beträgt: 63.5 %
[1] 0.635
> wichtel_unglueck(n=200, k=1)
Die Wahrscheinlichkeit, dass mindestens 1 Person(en) unter 200 Person(en) ihr eigenes Geschenk ziehen, beträgt: 61.8 %
[1] 0.618
> wichtel_unglueck(n=500, k=1)
Die Wahrscheinlichkeit, dass mindestens 1 Person(en) unter 500 Person(en) ihr eigenes Geschenk ziehen, beträgt: 63.4 %
[1] 0.634
> wichtel_unglueck(n=100, k=2)
Die Wahrscheinlichkeit, dass mindestens 2 Person(en) unter 100 Person(en) ihr eigenes Geschenk ziehen, beträgt: 26 %
[1] 0.26
```

Fig. 1.5 Code von A3.2 Teil 3

Die in Aufgabe 3.1 berechnete Wahrscheinlichkeit gilt weiterhin, auch unabhaengig davon wie gross wir n waehlen. Es bleibt dabei: ab fuenf Personen ist die Wahrscheinlichekeit groesser als dreiundsechzig Prozent, dass mindestens eine Person ihr eigenes Wichtelgeschenk zieht. Dieser Befund wurde ebenfalls 2004 von Armin Himmelrath im Spiegel veroeffentlicht, Stephan Kipp lieferte hierbei die Ergebnisse 08. Spiegelbericht - Beunruhigender Befund Quelle 8.

Fuer die "if-Bedingung" fuer n und k = 0 war diese Quelle sehr hilfreich: 09. Artikel zu Argumenten Quelle 9

3 Aufgabe 3.4: Testfaelle

Wir wollen das unsere Funktion bei Eingabe fuer alle Art von uebergebenen Argumenten funktioniert, um dies zu ueberpruefen erzeugen wir verschiedene Testfaelle.
Die library "testthat" liefert hierbei die Funktionen "test_that()" und "expect_error()" bzw. "expect_no_error()". Diese helfen uns in Kombination mit den in 3.2 erstellten "if-Bedingungen". Nun koennen wir die Funktionen aus "testthat" nutzen, um erwartete Ausgaben mit den tatsaechlichen zu ueberpruefen. Stimmen diese ueberein, wird uns angezeigt, dass unser Test erfolgreich ablaeuft und in der Konsole wird "Test passed" uebergeben.

```
install.packages(testthat)
library(testthat)
# A3.4 4 Testfälle
# Testfall für n=10, k=1, die Funktion wichtel_unglueck wird wie beabsichtigt genutzt und gibt eine W'keit von etwa 62% an
test_that("Die Funktion funktioniert so wie sie sollte.", {
  expect_no_error(wichtel_unglueck(n=10, k=1))
})
# Testfall für k=3, n=2, dies ist per Defintion verboten, wir suchen k unter n Personen die ihr eigenes Geschenk erhalten
test_that("k darf nicht größer als n sein, dies testen wir hier durch k=3, n=2", {
  expect_error(wichtel_unglueck(n=2, k=3), "k darf nicht größer als n sein.")
})
# Testfall für Buchstabeneingabe bei k
test_that("Die Funktion gibt einen Fehler wieder, da wir für k ein Zeichen angeben.", {
  expect_error(wichtel_unglueck(n=2, k="t", iterationen=150),
               "Der Funktion wurde für das Argument k mindestens 1 Buchstabe übergeben, bitte geben Sie ausschließlich Zahlen ein.")
})
# Achtung, 3.2 bzw. wichtel_unglueck wurde umgeschrieben, strings sind grundsätzlich größer als zahlen,deshalb testen wir erst auf strings und dann auf k>n

test_that("Die Funktion wichtel_unglueck bekommt keinen Wert für n übergeben", {
  expect_error(wichtel_unglueck(k=5), "Das Argument für n wurde nicht übergeben, Wichtel_unglueck wird nun mit n=100 ausgeführt.")
})
```

Fig. 1.6 Code von A3.4

Beispielhaft testen wir hier vier Szenarien. Im ersten Fall testen wir, ob die Funktion mit gueltigen Eingaben unsere gesuchte Wahrscheinlichkeit berechnet.

In den drei folgenden Testfaellen erwarten wir den Abrruch der Funktion, da wir bewusst moegliche, aber fuer die Funktion ungueltige Argumente uebergeben. Weiterhin soll auch eine Nachricht uebergeben werden, die uns aufgeklaert was falsch gelaufen ist.

Im zweiten Fall ueberpruefen wir, dass wir ein k uebergeben, dass groesser als n ist. Dies ist allerdings per Definition der Aufgabe verboten, da wir k Personen unter n suchen, k somit also

<= n sein muss.

Im dritten Fall geben wir fuer k einen Buchstaben ein, die Funktion erwartet positive ganze Zahlen fuer alle Argumente.

Im vierten Fall uebergeben wir keinen Wert fuer n. Ohne Initialisierung von n und k kann die Funktion nicht ausgefuehrt werden.

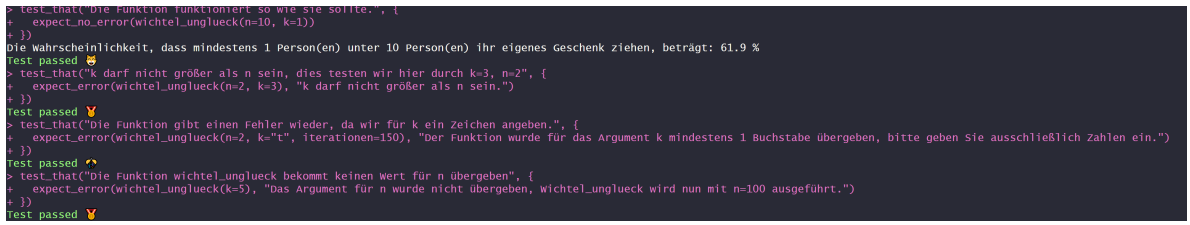

Fig. 1.7 Ausgabe von A3.4

genutzte Quellen:

10. Video zur testthat Bibliothek Quelle 10
11. Artikel zur testthat Bibliothek Quelle 11

4 Aufgabe 3.5: Vorbereitung der Daten

Nun wollen wir die aus Abschlussbericht 1 bekannten Daten einlesen und aufbereiten. Bevor wir die Daten in R einlesen entfernen wir per "suchen und ersetzen" in Excel saemtliche Anfuehrungszeichen.

Wir haben uns dafuer entschieden den gesamten Datensatz von "not available"-Werten(NAs) und unplausiblen Werten zu bereinigen, wie zum Beispiel negative Windgeschwindkeiten, bevor wir nach unserer gruppenspezifischen Station gefiltert haben. Dies ermoeglicht in der Zukunft den Vergleich zwischen Stationen ohne weitere Anpassung am Datensatz.
Im Folgenden werden wir die Visualisierung anhand der Station "18th St & Wyoming Ave NW" vornehmen und haben deshalb unseren "data.frame" umbenannt zu "daten_81".

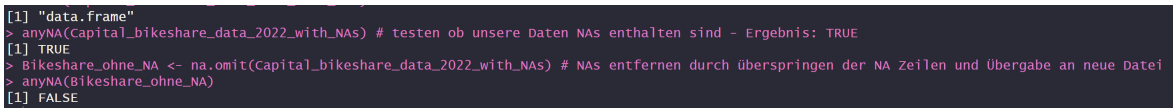

Fig. 1.8 Code von A3.5 - ohne NAs

Die Idee zur Nutzung des Befehls "na.omit" lieferte chatgpt:

12. Idee fuer Nutzung na.omit Quelle 12

```
> #Daten auf Plausiblität der Werte überprüfen
> # Station wird nicht auf Datenanomalien geprüft, da dies nicht sinnvoll erscheint
> range(Bikeshare_ohne_NA$wind_speed) # negative Windgeschwindigkeiten gefunden -> Datenanomalie - oder ist das dann Rückenwind :)
[1]  2.24 19.69
> range(Bikeshare_ohne_NA$count) # keine weiteren Datenanomalien gefunden für alle anderen Spalten
[1]   1 458
> range(Bikeshare_ohne_NA$date)
[1] "2022-01-01" "2022-11-30"
> range(Bikeshare_ohne_NA$precipitation)
[1] 0.00 4.05
> range(Bikeshare_ohne_NA$mean_temperature)
[1] 22 86
> range(Bikeshare_ohne_NA$max_temperature)
[1] 27 99
> range(Bikeshare_ohne_NA$min_temperature)
[1] 16 79
> range(Bikeshare_ohne_NA$snowfall)
[1] 0.0 6.9
> range(Bikeshare_ohne_NA$snow_depth)
[1] 0.0 7.1
>
> #Unplausible Daten entfernen
> Bikeshare_ohne_NA<- Bikeshare_ohne_NA[Bikeshare_ohne_NA$wind_speed >= 0, ] # negative Windgeschwindkeiten werden entfernt
> range(Bikeshare_ohne_NA$wind_speed) # Kontrolle Ergebnis: hat geklappt
[1]  2.24 19.69
>
```

Fig. 1.9 Code von A3.5 - Datenanomalien und Eliminierung dieser

```
> daten_81 <- Bikeshare_ohne_NA %>%
+   filter(station == "18th St & Wyoming Ave NW")
> range(daten_81$station)
[1] "18th St & Wyoming Ave NW" "18th St & Wyoming Ave NW"
```

Fig. 1.10 Code von A3.5 - Daten gefiltert nach "18th St & Wyoming Ave NW"

```
### Import der Daten -- kompletter Datensatz ####
#Originaler Datensatz, in Excel nur die Anführungszeichen entfernt

Capital_bikeshare_data_2022_with_NAs <- read.csv(file = "C:/Users/TGerner/Desktop/Capital_bikeshare_data_ohneAnfuherungszeichen.csv",
                                                 header = TRUE,
                                                 sep = ",",
                                                 dec = ".")
View(Capital_bikeshare_data_2022_with_NAs)

#Kontrolle ob Data.frame
class(Capital_bikeshare_data_2022_with_NAs) ## ist Data.Frame

#Entfernen NA´s des ganzen Datensatzes
anyNA(Capital_bikeshare_data_2022_with_NAs) # testen ob unsere Daten NAs enthalten sind - Ergebnis: TRUE
Bikeshare_ohne_NA <- na.omit(Capital_bikeshare_data_2022_with_NAs) # NAs entfernen durch überspringen der NA Zeilen und Übergabe an neue Datei
anyNA(Bikeshare_ohne_NA) # testen ob immer noch NAs enthalten sind - Ergebnis: FALSE
View(Bikeshare_ohne_NA) # anschauen des neuen Datensatzes ohne NAs

#Daten auf Plausiblität der Werte überprüfen
# Station wird nicht auf Datenanomalien geprüft, da dies nicht sinnvoll erscheint
range(Bikeshare_ohne_NA$wind_speed) # negative Windgeschwindigkeiten gefunden -> Datenanomalie - oder ist das dann Rückenwind :)
range(Bikeshare_ohne_NA$count) # keine weiteren Datenanomalien gefunden für alle anderen Spalten
range(Bikeshare_ohne_NA$date)
range(Bikeshare_ohne_NA$precipitation)
range(Bikeshare_ohne_NA$mean_temperature)
range(Bikeshare_ohne_NA$max_temperature)
range(Bikeshare_ohne_NA$min_temperature)
range(Bikeshare_ohne_NA$snowfall)
range(Bikeshare_ohne_NA$snow_depth)

#Unplausible Daten entfernen
Bikeshare_ohne_NA<- Bikeshare_ohne_NA[Bikeshare_ohne_NA$wind_speed >= 0, ] # negative Windgeschwindkeiten werden entfernt
range(Bikeshare_ohne_NA$wind_speed) # Kontrolle Ergebnis: hat geklappt

# hier kommt stationsfilter für unsere Gruppe: 81 - 18th St & Wyoming Ave NW
daten_81 <- Bikeshare_ohne_NA %>%
  filter(station == "18th St & Wyoming Ave NW")

View(daten_81) # Überprüfung des Datensatzes
```

Fig. 1.11 Code von A3.5

2

Visualisieren in R

1 Aufgabe 4.1: Verschiedene Zusammenhaenge in Bezug auf die Fahrradausleihe

Folgende R Skripte von Timo Adam haben massgeblich zur Erfuellung der Aufgaben beigetragen:
13. Adam: Skript 1 Quelle 13
14. Adam: Skript 2 Quelle 14

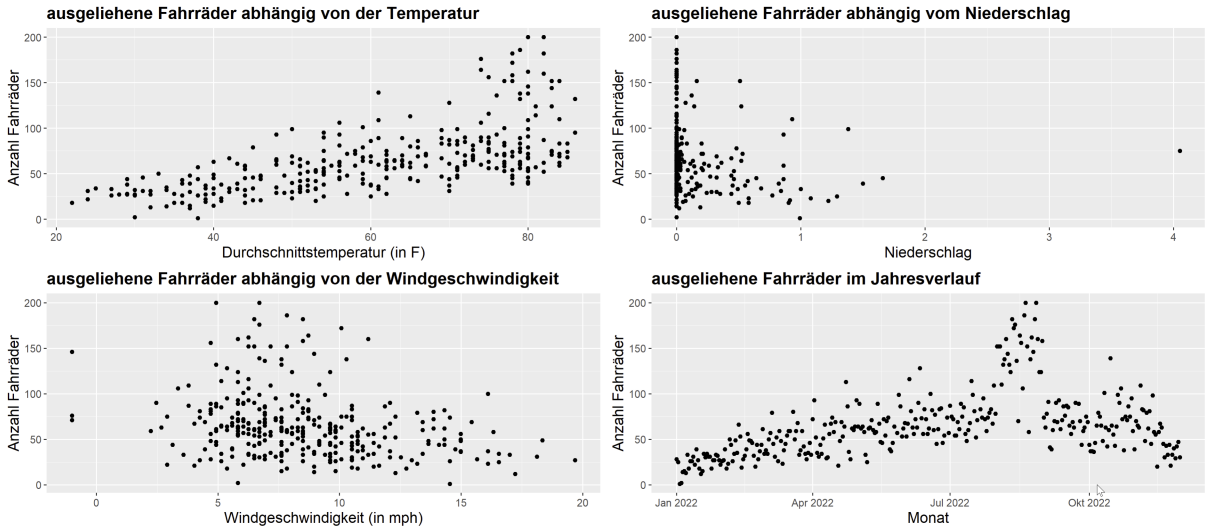

Fig. 2.1 Anzahl ausgeliehener Fahrraeder in Zusammenhang gestellt mit: Durch -schittstemperatur, Niederschlag, Windgeschwindigkeit und Zeitpunkt im Jahr

Wie in den Grafiken zu sehen ist, bestehen unterschiedliche Zusammenhaenge. Zum Beispiel wirken sich hoehere Temperaturen positiv auf die Anzahl ausgeliehener Fahrraeder aus.
Dies sieht man ebenfalls im Jahresverlauf, hier ist zu erkennen, dass im Sommer deutlich mehr Fahrraeder ausgeliehen werden als im Winter.
Hingegen wirken sich hoehere Windgeschwindkeiten negativ auf die Anzahl ausgeliehener Fahrraeder aus.
Wenig Regen wirkt sich positiv auf die Anzahl ausgeliehener Fahrraeder aus. Allerdings gibt es eine Ausnahme, bei der trotz hohen Niederschlagsmenge fuenfundsiebzig Fahrraeder ausgeliehen wurden.

2 Aufgabe 4.2: Vergleich Fahhradausleihe in Bezug zu Temperatur und Niederschlagsmenge

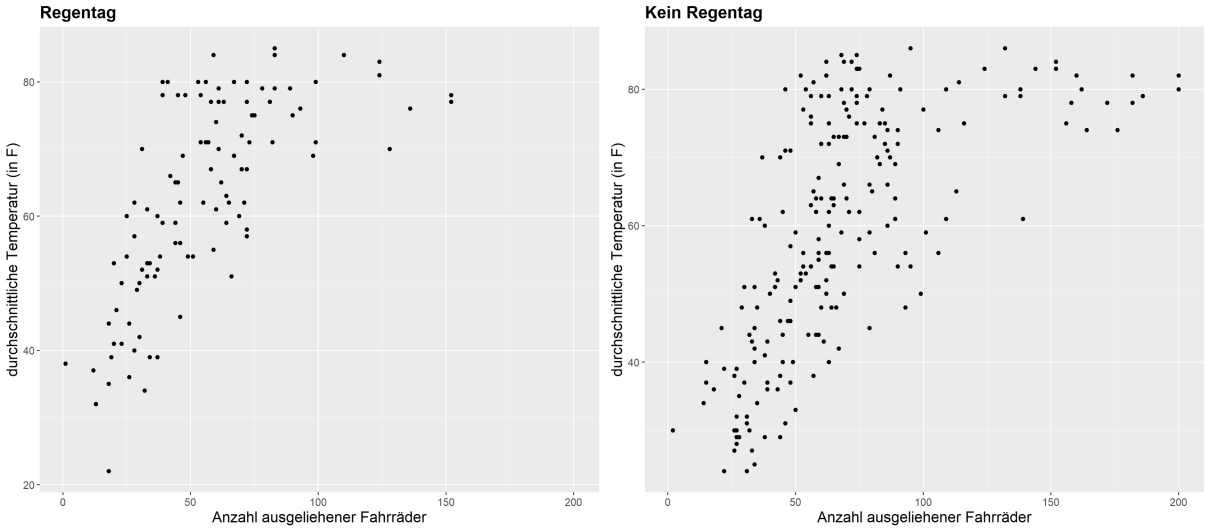

Fig. 2.2 Graphischer Zusammenhang zwischen ausgeliehenen Fahrraedern an Tagen mit und ohne Regen

Weiterhin ist der positive Zusammenhang zwischen Temperatur und der Anzahl ausgeliehener Fahrraeder zu erkennen, dieser wird nur geringfuegig durch den Niederschlag beeinflusst.

Es faellt auf, dass die Hoechstwerte auftreten, wenn es nicht regnet und die Temperatur im Durchschnitt 80 Grad Fahrenheit erreicht. Diese liegen bei 200 ausgeliehenen Fahrraedern.

Es liegen mehr Datenpunkte in der Grafik ohne Regen vor. Dies ist darauf zurueckzufuehren, dass es im Testgebiet weniger Tage mit Regen gab als ohne.

chatgpt lieferte hier die Idee fuer die Nutzung von coord_cartesian, dies aendert die x-Achse zum besseren Vergleich.
15. Idee fuer Nutzung coordcartesian x-Achsenskalierung Quelle 15

3 Aufgabe 4.3: Verteilungsfunktionen

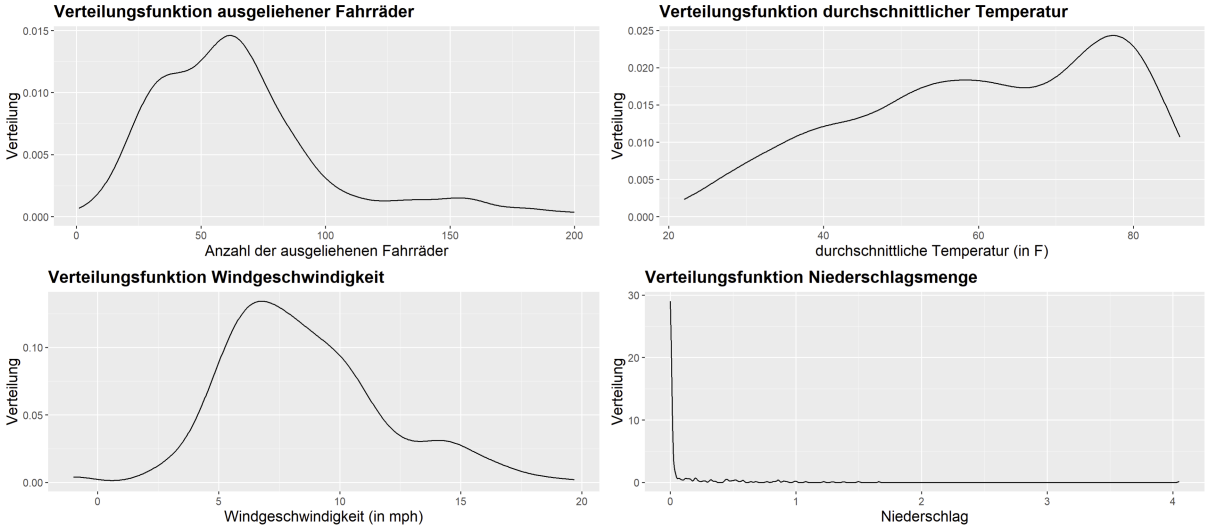

Fig. 2.3 Verteilungsfunktionen: Fahrraeder, durchschnittliche Temperatur, Windgeschwin -digkeit und Niederschlag

In der Verteilungsfunktion der ausgeliehenen Fahrraeder sieht man, dass ein Grossteil der Ausleihen zwischen null und einhundert stattfindet, das Maximum liegt bei etwa sechzig Ausleihen pro Tag. Mehr als einhundert Ausleihen werden selten erreicht. Am seltensten ist der Hoechstwert mit zweihundert.

In der Grafik mit der durchschnittlichen Temperatur sieht man einen Anstieg der Ausleihen bis achtzig Grad Fahrenheit, ab dann nehmen die Ausleihen pro Tag wieder ab. Etwa bei achtzig liegt auch der Hoechstwert der Ausleihen pro Tag.

Die Verteilungsfunktion der Windgeschwindigkeiten zeigt, dass die Ausleihen negativ beeinflusst werden, sobald die Windgeschwindigkeit ungefaehr sechs Meilen pro Stunde ueberschreitet.

Aus der Verteilungsfunktion der Niederschlagsmenge erkennt man, dass die meisten Ausleihen an Tagen stattfinden, die niederschlagsfrei sind.

4 Aufgabe 4.4: Verteilungsfunktion in Bezug zu Jahreszeiten

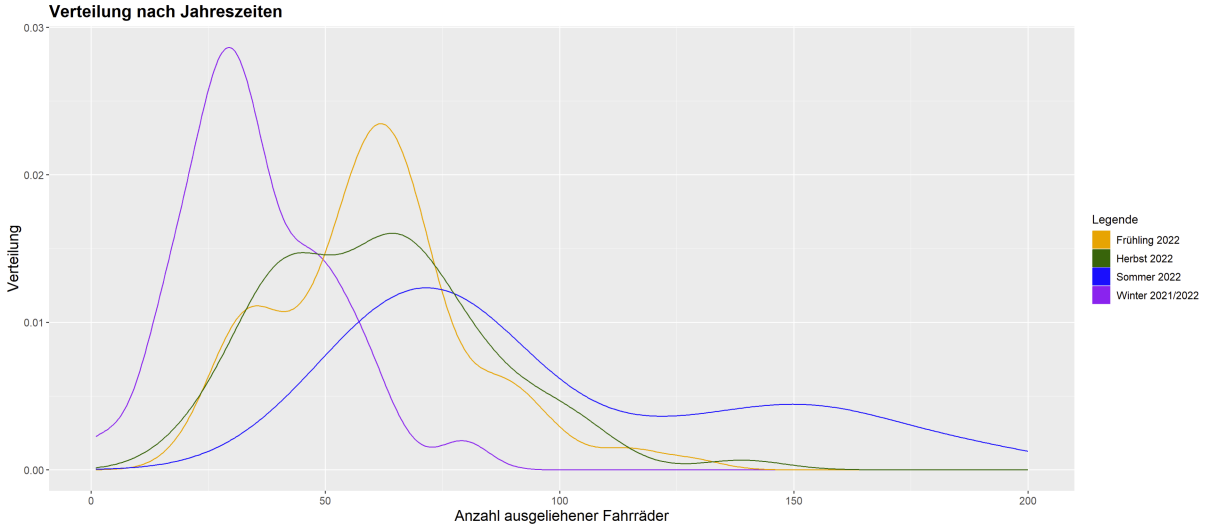

Fig. 2.4 Verteilungsfunktion ausgeliehener Fahrraeder in verschiedenen Jahreszeiten

Um die Verteilung nach Jahreszeiten grafisch darzustellen haben wir den Datensatz gemaess der Stichtage der 16. Jahreszeiten Quelle 16 aufgeteilt.

Deutlich zu erkennen, ist die hoehere Auslastung im Sommer, im Vergleich zum Winter. Fruehling und Herbst sind dazwischen und aehnlich in Bezug auf die Auslastung. Betrachten wir die Hoechstwerte, so erkennen wir, dass im Sommer bis zu zweihunert Ausleihen am Tag stattfinden, waehrend es im Fruehling und Herbst etwa einhundertvierzig sind und im Winter nicht mehr als einhundert.

Das wirkt sich am Beispiel der Verteilungsfunktion fuer den Winter derart aus, dass sich die meisten Ausleihen pro Tag in der Groessenordnung dreissig befinden. Im Sommer hingegen liegt diese Zahl bei siebzig, im Fruehling sechzig und im Herbst bei fuenfzig.

5 Aufgabe 4.5: 3D Streudiagramm

Wir werden im weiteren drei verschiedene Perspektiven anschauen, um die drei Dimensionen besser zu veranschaulichen.

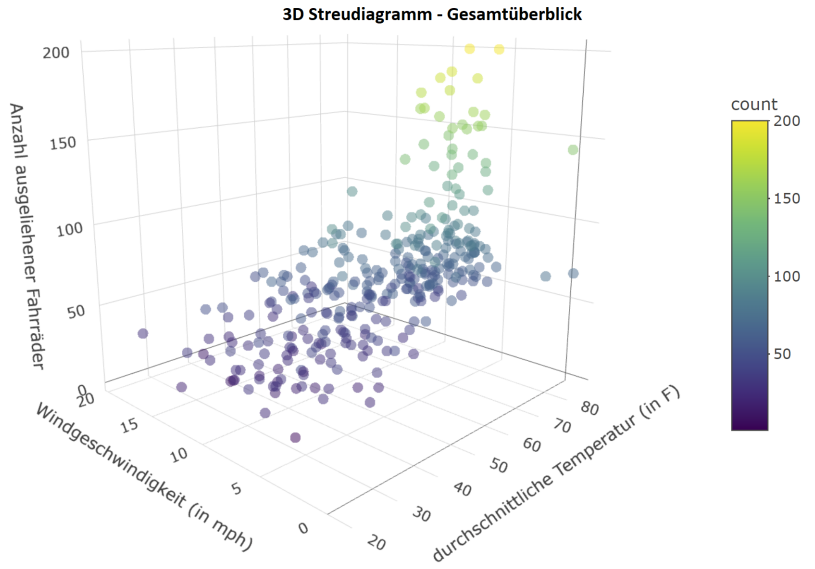

Fig. 2.5 Streudiagramm Perspektive 1: Gesamtueberblick

Wir sehen eine farblich gekennzeichnete Grafik, welche gleichzitig auf drei Achsen den Zusammenhang zwischen der Anzahl ausgeliehener Fahrraeder, der Windgeschwindigkeit und der durchschnittlichen Temperatur aufzeigt.

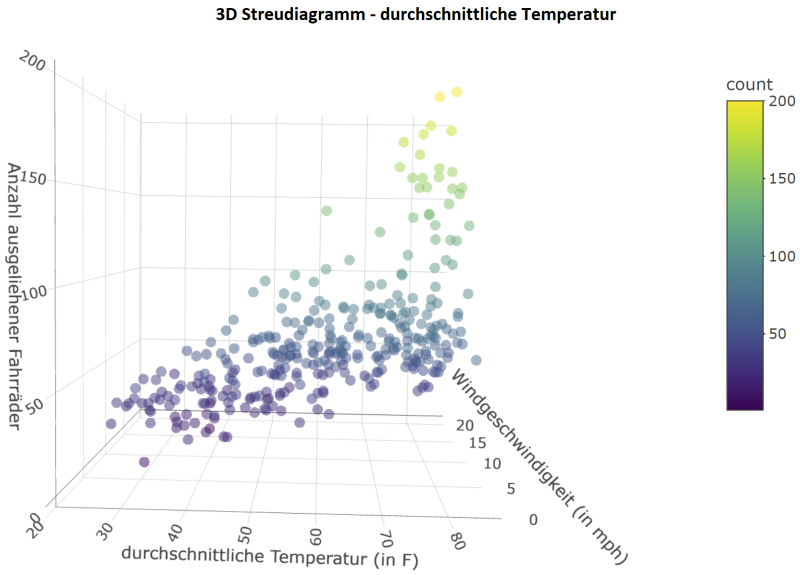

Fig. 2.6 Streudiagramm Perspektive 2: Durchschnittstemperatur im Fokus

Man sieht den positiven Zusammenhang zwischen steigender Durchschnittstemperatur und Anzahl ausgeliehener Fahrraeder.

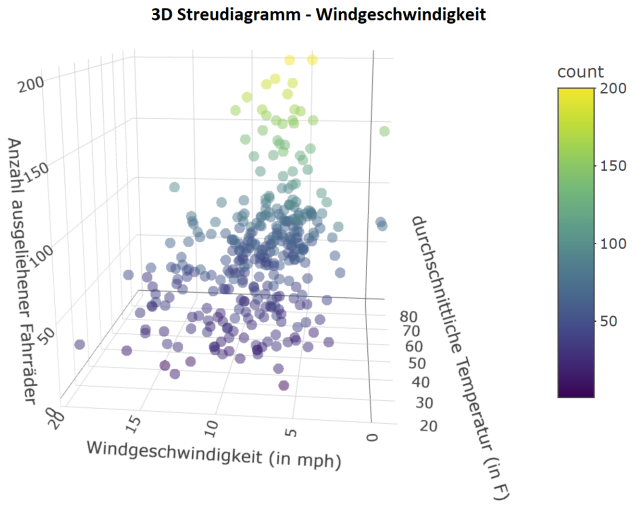

Fig. 2.7 Streudiagramm Perspektive 3: Windgeschwindigkeit im Fokus

Hier sieht man den Zusammenhang zwischen Windgeschwindigkeit und Anzahl ausgeliehener Fahrraeder. Umso niedriger die Windgeschwindigkeit desto groesser die Anzahl ausgeliehener Fahrraeder.

Quellenverzeichnis

01. BikeshareDatensatz, https://capitalbikeshare.com/system-data zuletzt besucht am 04.02.2024 (Quelle 1). *test 1*. test 2.
02. GitHub, https://github.com/TGerner/CoMet_Abgabe2 zuletzt besucht am 04.02.2024 (Quelle 2). *Autor: Kevin Wilm Schliewe und Timon Gerner*. Universitaet Bielefeld.
03. R Dokumentation, https://cran.r-project.org/doc/manuals/r-release/fullrefman.pdf zuletzt besucht am 04.02.2024 (Quelle 3). *R: A Language and Environment for Statistical Computing*. The R Core Team.
04. LaTeX-Dokument, LaTeX-Template von Chloé Goupy https://www.overleaf.com/latex/templates/thesis-layout-upariscite/qvmbkpnxvtdn zuletzt besucht am 30.11.2023 (Quelle 4). *LaTeX Vorlage*.
05. Oelschläger: R Skript 1, https://moodle.uni-bielefeld.de/pluginfile.php/235721/mod_resource/content/1/comet_ws_2324_programmieren_in_R_V01.R zuletzt besucht am 04.02.2024 (Quelle 5). *Idee fuer Wichtelfunktion*. Universitaet Bielefeld - Lennart Oelschläger.
06. Oelschläger: R Skript 2, https://moodle.uni-bielefeld.de/pluginfile.php/238872/mod_resource/content/1/comet_ws_2324_programmieren_in_R_V02.R zuletzt besucht am 04.02.2024 (Quelle 6). *R Skript 1, einlesen der Daten und erste Schritte*. Universitaet Bielefeld - Lennart Oelschläger.
07. Oelschläger: R Skript 3, https://moodle.uni-bielefeld.de/pluginfile.php/244178/mod_resource/content/1/comet_ws_2324_programmieren_in_R_V03.R zuletzt besucht am 04.02.2024 (Quelle 7). *Dokumentation und Verwendung von testthat*. Universitaet Bielefeld - Lennart Oelschläger.
08. Spiegelbericht - Beunruhigender Befund, https://www.spiegel.de/lebenundlernen/uni/hoehere-mathematik-wie-gross-ist-die-chance-der-selbst-bewichtelung-a-333813.html zuletzt besucht am 04.02.2024 (Quelle 8). *Autor: Armin Himmelrath*. Spiegel.
09. Artikel zu Argumenten, https://stackoverflow.com/questions/7964830/test-if-an-argument-of-a-function-is-set-or-not-in-r zuletzt besucht am 04.02.2024 (Quelle 9). *Test if an argument of a function is set or not in R*. Stackoverflow - $R_u ser$.
10. Video zur testthat Bibliothek, https://www.youtube.com/watch?v=KbwYdRbmgbY zuletzt besucht am 04.02.2024 (Quelle 10). *How to set up Automated Tests for Your R Package using testthat*. StatistikinDD - Wolf Riepl.
11. Artikel zur testthat Bibliothek, https://stackoverflow.com/questions/36332845/how-to-test-for-a-message-and-an-error-message-simultaneously-in-r-testthat zuletzt besucht am 04.02.2024 (Quelle 11). *How to test for a message and an error message simultaneously in R testthat?* Stackoverflow - Steph Locke.
12. Idee fuer Nutzung na.omit, https://chat.openai.com/ zuletzt besucht am 04.02.2024 (Quelle 12). *na.omit*. OpenAI.

13. Adam: Skript 1, https://moodle.uni-bielefeld.de/pluginfile.php/263446/mod_resource/content/0/R-Skript.R zuletzt besucht am 04.02.2024 (Quelle 13). *Skript 1, 1.2 Einfuehrung in die ggplot2-Syntax, 1.3 Scatterplots und Zeitreihenplots, 1.4 Boxplots, Histogramme und Kerndichteschaetzer, 1.5 Themen und weitere Anpassungen.* Universitaet Bielefeld - J.-Prof. Dr. Timo Adam.
14. Adam: Skript 2, https://moodle.uni-bielefeld.de/pluginfile.php/271311/mod_resource/content/0/R-Skript.R zuletzt besucht am 04.02.2024 (Quelle 14). *Skript 2, 1.5 Anordnung von Grafiken, Scatterplots, Grafiken anordnen mit grid.arrange(<...>), 3D Scatterplots, Aenderungen an Grafiken, Datum umspeichern.* Universitaet Bielefeld - J.-Prof. Dr. Timo Adam.
15. Idee fuer Nutzung coordcartesian x-Achsenskalierung, https://chat.openai.com/ zuletzt besucht am 04.02.2024 (Quelle 15). *X-Achsenskalierung.* OpenAI.
16. Jahreszeiten, https://www.linker.ch/eigenlink/jahreszeiten_beginn.htm zuletzt besucht am 04.02.2024 (Quelle 16). *Jahreszeitenbeginn.* linker.ch - Peterhans-Software.

Abbildungsverzeichnis