Teil 3

Exploratory Data Analysis

Rebecca Karwen

Contents

Aufgabenstellung

Die ISDL-BikeShare AG bietet seit 2019 einen Fahrradverleih an. Hier können Nutzer über eine mobile App ein Fahrrad an einem bestimmten Ort ausleihen und an einem anderen Ort zurückgeben. Zunächst werden die Daten geladen.

1 Überblick über die Daten

Attributnamen sowie die zugehörigen Datentypen

Zunächst schauen wir uns die generelle Struktur des Datensets und die Attribute mit ihrem jeweiligem Datentyp an.

Anzahl an Attributen: 15
Anzahl an Beobachtungen im Datenset: 17379

```
## # A tibble: 15 x 3
##    AttributeNames Datatype  Description
##    <chr>          <chr>     <chr>
##  1 id             numeric   id
##  2 date           Date      date
##  3 season         numeric   season
##  4 year           numeric   year
##  5 month          numeric   month
##  6 hour           numeric   hour
##  7 holiday        numeric   holiday
##  8 weekday        numeric   weekday
##  9 workingday     numeric   workingday
## 10 weatherCond    numeric   weatherCond
## 11 tempC          character tempC
## 12 perceivedTempC numeric   perceivedTempC
## 13 humidity       character humidity
## 14 windspeed      numeric   windspeed
## 15 rentals        numeric   rentals
```

Bestimmung des Skalenniveaus

id : Nominalskala

season: Nominalskala Nominalskala, da Jahreszeiten diskrete Einheiten darstellen, keinen quantitativen Wert und keine Reihenfolge haben.

date: Intervalskala year: Intervalskala month: Intervalskala Alle 3 Attribute gehören zur Intervalskala, da die Abstände zwischen den Werten gleich sind, es keinen Nullpunkt gibt und es nicht möglich ist, die Werte zu multiplizieren, teilen oder Verhältnisse zu berechnen.

weekday: Nominalskala Der Wochentag gehört zur Nominalskala, da er eine diskrete Einheit darstellt. Die Wochentage haben zudem keinen quantitativen Wert.

workingday: Nominalskala Ob ein Wochentag ein Werktag ist oder nicht stellt eine diskete Einheit da, hat aber kein Reihenfolge, deswegen die Nominalskala.

weatherCond: Ordinalskala Die Wetterbedingungen stellen diskrete geordnete Einheiten da.

tempC: Intervalskala perceivedTempC: Intervalskala Beide Temperaturen gehören zur Intervallskala, da sie konstante Abstände und keinen natürlichen Nullpunkt haben.

humidity: Verhältnisskala, da ein Nullpunkt existiert.

windspeed: Verhältnisskala Windgeschwindigkeit wird der Verhältnisskala zugeordnet, da sie einen Nullpunkt und eine Ordnung mit gleichen Abständen hat.

rentals: Verhältnisskala Die verliehenen Fahrräder gehören zur Verhältnisskala, da es einen Nullpunkt gibt und man damit Häufigkeit berechnen kann.

Data Cleaning

Bevor die Daten analysiert werden, bereiten wir sie noch für die weitere Verarbeitung vor. Dafür schauen wir zunächst wo es NA Werte gibt. In unserem Datenset sind davon keine vorhanden.

```
colSums(is.na(bike_sharing_data))
```

```
##             id           date         season           year          month
##              0              0              0              0              0
##           hour        holiday        weekday     workingday    weatherCond
##              0              0              0              0              0
##          tempC perceivedTempC       humidity      windspeed        rentals
##              0              0              0              0              0
```

Suche nach empty Strings oder unmöglichen Daten

```
## Überprüfen auf Feiertag höher als 1
no_holiday <- bike_sharing_data %>%
  filter(holiday > 1)

## Überprüfen auf leere Datumsfelder
observationsEmptyDate <- bike_sharing_data %>%
  filter(date == "") %>%
  count()
##keins gefunden

## Überprüfen auf leere Jahreszeit
observationsEmptySeason <- bike_sharing_data %>%
  filter(season == "") %>%
  count()

## Überprüfen auf leeres Jahr
observationsEmptyYear <- bike_sharing_data %>%
  filter(year == "") %>%
  count()

## Überprüfen auf Jahr größergleich 2
observationsHighYear <- bike_sharing_data %>%
  filter(year >= 2) %>%
  count()

## Überprüfen auf Monat höher als 12
observationsHighMonth <- bike_sharing_data %>%
  filter(month > 12) %>%
  count()

## Überprüfen auf  Weekday höher als 7
observationsWeekday <- bike_sharing_data %>%
  filter(weekday > 7) %>%
```

```
  count()


## Überprüfen auf  WeatherCond höher als 4
observationsWeatherCond <- bike_sharing_data %>%
  filter(year > 4) %>%
  count()

## Überprüfung auf zu hohe Werte an windspeed
observationsWindspeed  <- slice_max(bike_sharing_data, bike_sharing_data$windspeed, n = 10)
## Muss angepasst werden

## Überprüfen auf Duplicate
dups <- sum(duplicated(bike_sharing_data))
#keine wurden gefunden

## Überprüfung auf zu hohe Anzahl an vermieteten Fahrrädern
slice_max(bike_sharing_data, bike_sharing_data$rentals, n = 10)
```

```
## # A tibble: 10 x 15
##       id date       season  year month  hour holiday weekday workingday
##    <dbl> <date>      <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl>      <dbl>
##  1 14774 2020-09-12      3     1     9    18       0       3          1
##  2 14965 2020-09-20      3     1     9    17       0       4          1
##  3 14749 2020-09-11      3     1     9    17       0       2          1
##  4 14726 2020-09-10      3     1     9    18       0       1          1
##  5 15085 2020-09-25      4     1     9    17       0       2          1
##  6 15781 2020-10-24      4     1    10    17       0       3          1
##  7 10623 2020-03-23      2     1     3    17       0       5          1
##  8 15109 2020-09-26      4     1     9    17       0       3          1
##  9 15445 2020-10-10      4     1    10    17       0       3          1
## 10 15589 2020-10-16      4     1    10    17       0       2          1
## # ... with 6 more variables: weatherCond <dbl>, tempC <chr>,
## #   perceivedTempC <dbl>, humidity <chr>, windspeed <dbl>, rentals <dbl>
```

```
## Zwischen 977 und 943 wirkt für gutes Wetter angebracht
```

Sicherstellen, dass die Daten die richtigen Bezeichnungen haben

```
bike_sharing_data <- bike_sharing_data %>%
  mutate(humidity = as.numeric(humidity),
         perceivedTempC = perceivedTempC / 10000,
         windspeed = windspeed / 10000,
         tempC= as.numeric(tempC),
         month = as.factor(month))


bike_sharing_data$season <- factor(bike_sharing_data$season,
                           levels = c(1,2,3,4),
                           labels = c("Winter", "Frühling", "Sommer", "Herbst"))

bike_sharing_data$weatherCond <-ordered(bike_sharing_data$weatherCond,
                                        levels = c(1,2,3,4),
                                        labels = c("klar bis teilweise bewölkt", "Nebel oder bewölktes
```

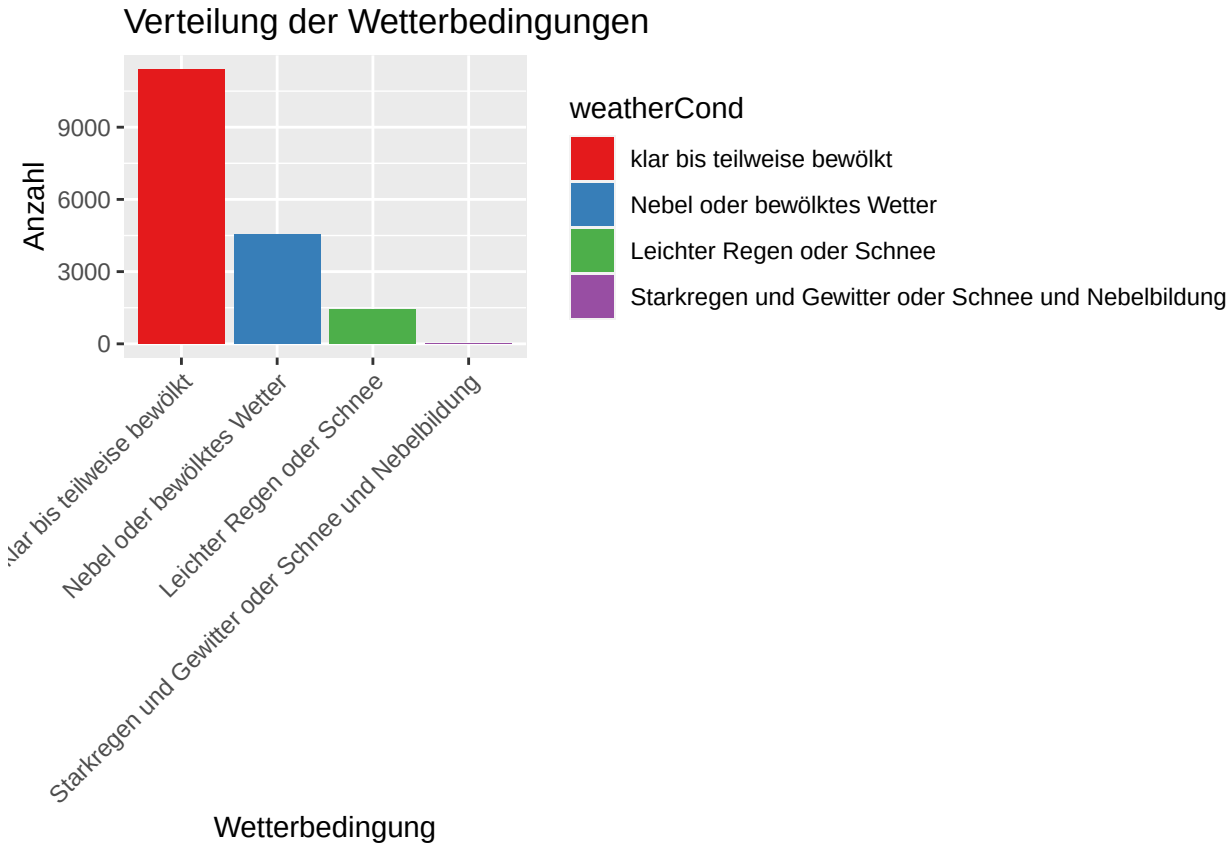
2 Verteilungen

2.1. Visualisierung der Verteilung der Wetterbedingungen

Um die Verteilung der Wetterbedingungen zu visualisieren, eignet sich ein Balkendiagramm sehr gut. Dieses zeigt auf der x Achse die verschiedenen Wetterbedingungen an und auf der y Achse wie häufig diese vorkommen.

```
bike_sharing_data_weatherCond <- bike_sharing_data %>%
  count(weatherCond)

ggplot(bike_sharing_data, aes(x =weatherCond, fill=weatherCond)) +
  geom_bar() +
  scale_fill_brewer(palette = "Set1") +
  labs(title = "Verteilung der Wetterbedingungen", x = "Wetterbedingung", y= "Anzahl")+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

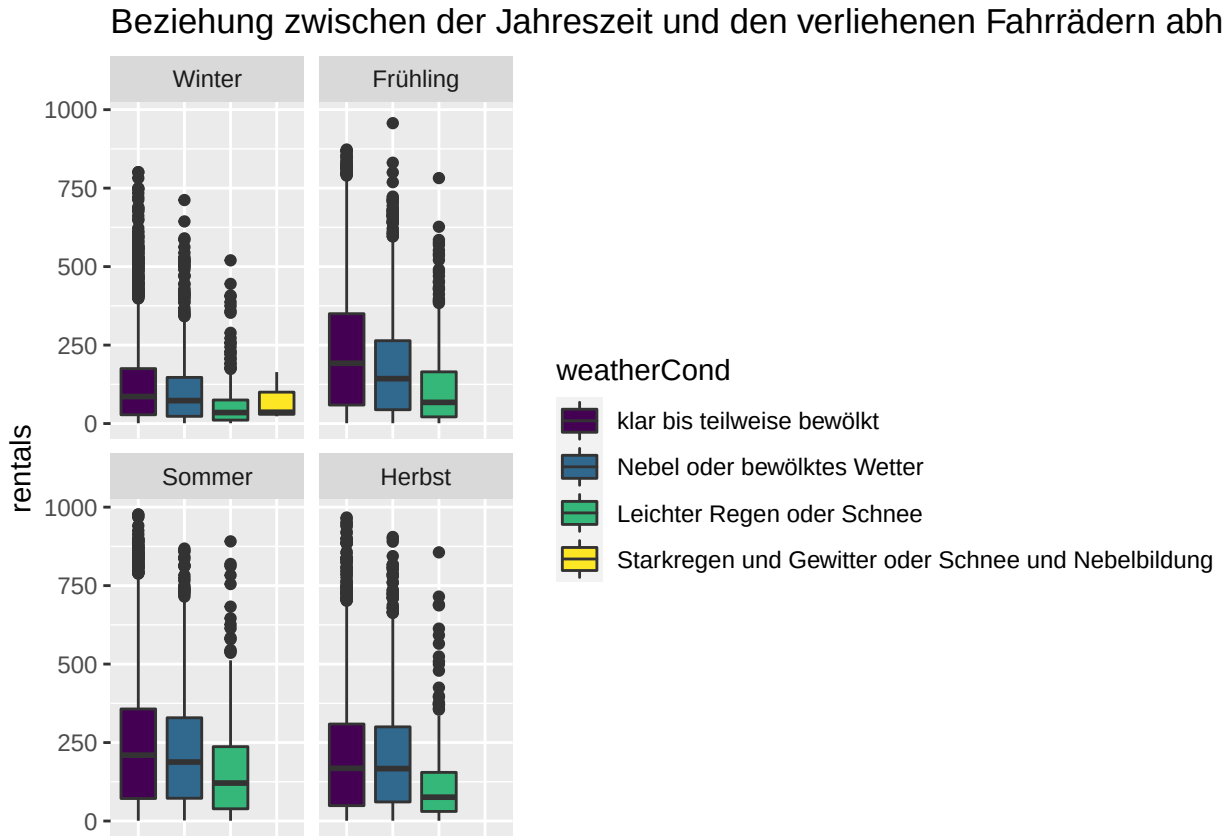


2.2. Visualisierung der Beziehung zwischen der Jahreszeit (season) und der Anzahl der verliehenen Fahrräder (rentals) abhängig von den Wetterbedingungen (weatherCond)

Um die Beziehung zwischen Jahreszeit und Anzahl der verliehenen Fahrräder darzustellen, eignet sich am besten ein Boxplot, bei dem die verliehenen Fahrräder mit jeweils einem Diagramm für die jeweilige Jahreszeit dargestellt werden.

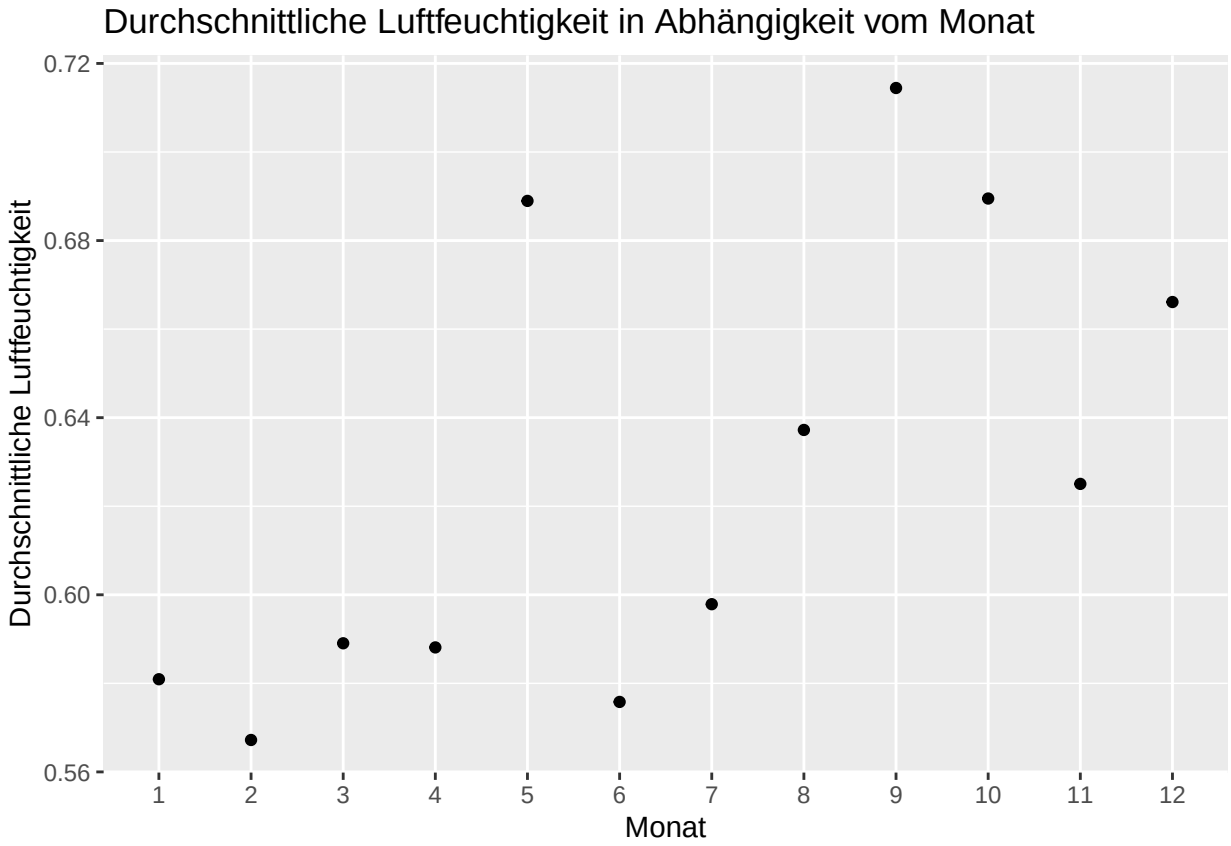
```
ggplot(bike_sharing_data, aes(x=weatherCond, y=rentals, fill= weatherCond)) +
  geom_boxplot() +
  facet_wrap(~season)+
  theme(axis.title.x=element_blank(),
```

```
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank()) +
  ggtitle("Beziehung zwischen der Jahreszeit und den verliehenen Fahrrädern abhängig von den Wetterbedi
```



2.3 Visualisierung der durchschnittliche Luftfeuchtigkeit (humidity) der gesamten Zeit in Abhängigkeit vom Monat (month)

```
bike_sharing_data %>%
  group_by(month) %>%
  summarize(meanHumidity = mean(humidity)) %>%
  ggplot(aes(x=month, y= meanHumidity)) +
  geom_point() +
  labs(title = "Durchschnittliche Luftfeuchtigkeit in Abhängigkeit vom Monat", x= "Monat", y= "Durchschn
```



Die durchschnittliche Luftfeuchtigkeit war am höchsten im Mai, September und Oktober.

3 Themenbereich T-Tests

3.1. Ist die durchschnittliche Anzahl an vermieteten Fahrrädern an nicht-Werktagen anders als an Werktagen? Stellen Sie eine ungerichtete Hypothese auf.

HO: Die durchschnittliche Anzahl an vermietet Fahrrädern ist an nicht-Werktagen genauso wie an Werktagen
H1: Die durchschnittliche Anzahl an vermieteten Fahrrädern ist an nicht-Werktagen anders als an Werktagen

Dafür wird zunächst ein Datenset mit den Werktagen und eins mit nicht-Werktagen erstellt.

```
bike_sharing_data %>%
  group_by(workingday) %>%
  summarize(meanRentals = mean(rentals),
            medianRentals = median(rentals),
            sd = sd(rentals)) %>%
  arrange(desc(meanRentals))
```

```
## # A tibble: 2 x 4
##   workingday meanRentals medianRentals    sd
##        <dbl>       <dbl>         <dbl> <dbl>
## 1          1        193.           151  185.
## 2          0        181.           119  173.
```

```
working_day <- bike_sharing_data %>%
  filter(workingday == 1)

not_working_day <- bike_sharing_data %>%
  filter(workingday == 0)
```

Vorbedingungen für den t-test

```
#1. Annahme, dass die Daten normalverteilt sind aufgrund des großen Datensets
#2. Vor dem T-Test Überprüfung der Homogenität der Varianz

leveneTest_data <- bike_sharing_data %>%
  mutate(WorkingDayYes = ifelse(workingday == 1,1,0))

leveneTest(leveneTest_data$rentals, leveneTest_data$WorkingDayYes)
```

```
## Warning in leveneTest.default(leveneTest_data$rentals,
## leveneTest_data$WorkingDayYes): leveneTest_data$WorkingDayYes coerced to factor.
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value  Pr(>F)
## group     1   2.739 0.09795 .
##       17377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Levene-Test for homogeneity of variance across groups
### H0 for Levene-Test: There is homogeneity of variance for both groups ("both groups have similar var
### H1 for Levene-Test: There is no homogeneity of variance for both groups ("both groups have differen
```

H0 wird akzeptiert, da der Levene-Test zeigt, dass Varianz der Homogenität besteht. Deshalb wird der normale t-test angewandt.

T-Test

```
t.test(working_day$rentals, not_working_day$rentals, conf.level = 0.95, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  working_day$rentals and not_working_day$rentals
## t = 3.994, df = 17377, p-value = 6.524e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   6.010212 17.594632
## sample estimates:
## mean of x mean of y
##  193.2078  181.4053
```

P ist kleiner als 0.05 , d.h. es gibt einen Unterschied der durchschnittlichen Anzahl an vermieteten Fahrrädern an nicht-Werktagen bzw. Werktagen. H0 wird also verworfen und H1 akzeptiert.

3.2. Ist die durchschnittliche Anzahl an vermieteten Fahrrädern im Jahr 2020 höher als im Jahr 2019? Stellen Sie eine gerichtete Hypothese auf.

H0: Die durchschnittliche Anzahl an vermieteten Fahrrädern war 2019 nicht höher als 2020 H1: Die durchschnittliche Anzahl an vermieteten Fahrrädern war 2019 höher als 2020

Dafür wird zunächst ein Datenset mit den Daten aus 2019 (Year 0) und ein Datenset mit den Daten aus 2020 (Year 1) erstellt.

```
bike_sharing_data %>%
  group_by(year) %>%
  summarize(meanRentals = mean(rentals),
            medianRentals = median(rentals),
            sd = sd(rentals)) %>%
  arrange(desc(meanRentals))
```

```
## # A tibble: 2 x 4
##    year meanRentals medianRentals    sd
##   <dbl>       <dbl>         <dbl> <dbl>
## 1     1        235.           191  209.
## 2     0        144.           109  134.
```

```
year_2020 <- bike_sharing_data %>%
  filter(year == 1)

year_2019 <- bike_sharing_data %>%
  filter(year == 0)
```

```
#1. Annahme, dass die Daten normalverteilt sind aufgrund des großen Datensets
#2. Vor dem T-Test Überprüfung der Homogenität der Varianz

leveneTest_data_2 <- bike_sharing_data %>%
  mutate(Year2020Yes = ifelse(year == 1,1,0))

leveneTest(leveneTest_data_2$rentals, leveneTest_data_2$Year2020Yes)
```

```
## Warning in leveneTest.default(leveneTest_data_2$rentals,
## leveneTest_data_2$Year2020Yes): leveneTest_data_2$Year2020Yes coerced to factor.
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1  1192.3 < 2.2e-16 ***
##       17377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Levene-Test for homogeneity of variance across groups
### H0 for Levene-Test: There is homogeneity of variance for both groups ("both groups have similar var
### H1 for Levene-Test: There is no homogeneity of variance for both groups ("both groups have differen
```

H0 wird verworfen, da der Levene-Test zeigt, dass keine Varianz der Homogenität besteht. Deshalb wird der Welch t-test angewandt.

T-Test

```
t.test(year_2019$rentals, year_2020$rentals, var.equal = TRUE,  conf.level = 0.95)
```

```
##
##  Two Sample t-test
##
## data:  year_2019$rentals and year_2020$rentals
## t = -34.108, df = 17377, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -96.09407 -85.64976
## sample estimates:
## mean of x mean of y
##  143.7944  234.6664
```

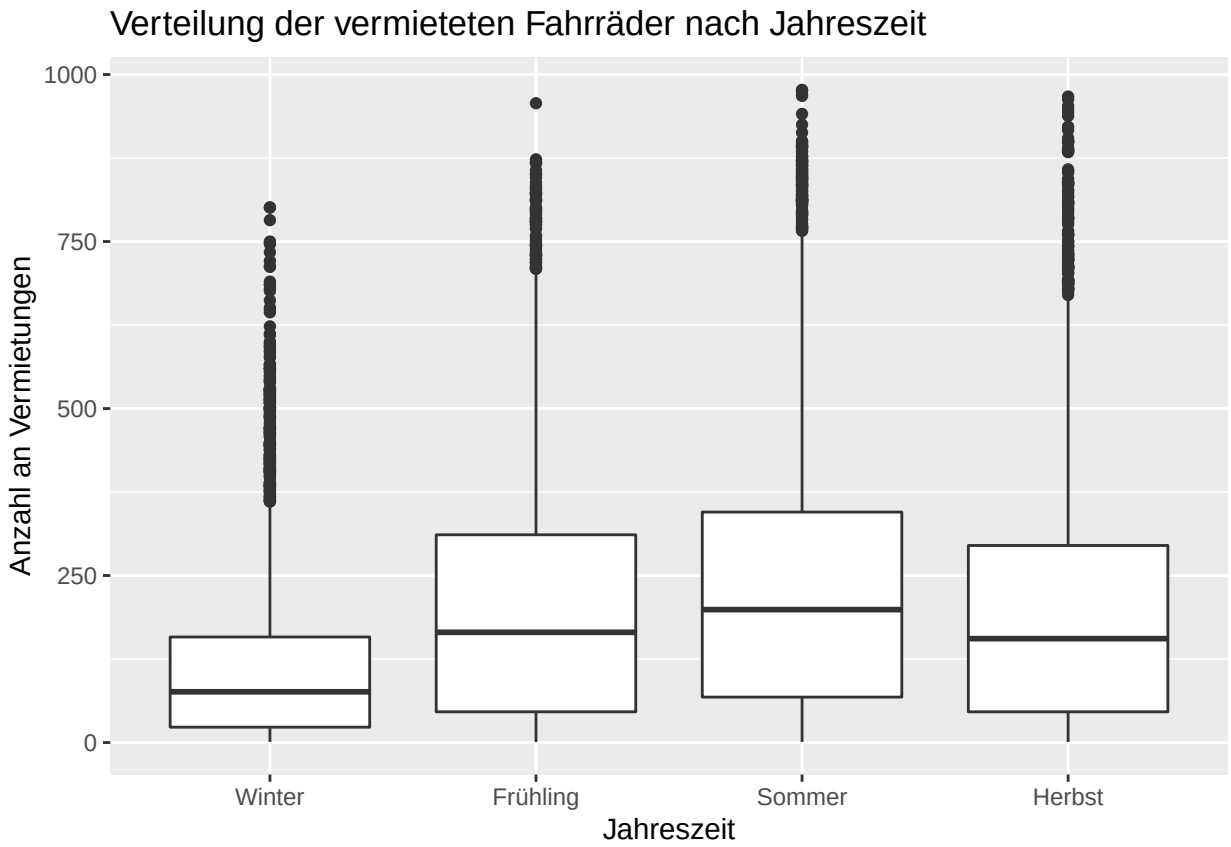
P ist kleiner als 0.05, deshalb wird H0 verworfen und H1 akzeptiert. Das bedeutet es wurden durchschnittlich mehr Fahrräder in 2020 als in 2019 vermietet.

4. ANOVA

Unterscheiden sich die Anzahl der vermieteten Fahrränder abhängig von der Jahreszeit? Wenn ja, wie genau? Interpretieren Sie Ihr Ergebnis.

Zunächst stellen werden die Daten graphisch dar, um einen besseren Überblick zu bekommen.

```
ggplot(bike_sharing_data, aes(x=season, y=rentals)) +
  geom_boxplot() +
  labs(title = "Verteilung der vermieteten Fahrräder nach Jahreszeit", x= "Jahreszeit", y = "Anzahl an V
```



Es scheint, dass mehr Fahrräder im Frühling und Sommer verliehen werden als im Winter und etwas mehr als im Herbst.

Danach werden die Hypothesen aufgestellt.

H0: Die Anzahl der vermieteten Fahrräder ist nicht abhängig von der Jahreszeit H1: Die Anzahl der vermieteten Fahrräder ist abhängig von der Jahreszeit

Zunächst erstellen wir für jede Jahreszeit einen Datensatz.

Dann führen wir den Levene Test durch der die Vorbedingung für die ANOVA ist.

```
## Levene-Test for homogeneity of variance across groups
### H0 for Levene-Test: There is homogeneity of variance for all three groups
### H1 for Levene-Test: There is no homogeneity of variance for both groups

leveneTest(rentals ~ season, bike_sharing_data)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     3   324.5 < 2.2e-16 ***
```

```
##       17375
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Da p < 0.05 ist, zeigt der Levene-Test, dass keine Homogene Varianz besteht und wir verwerfen deshalb H0. Deshalb wird nachhfolgend der Welch F-Test genutzt.

```
welch.test(rentals ~ season, data = bike_sharing_data)
```

```
##
##   Welch's Heteroscedastic F Test (alpha = 0.05)
## ------------------------------------------------------------
##   data : rentals and season
##
##   statistic  : 606.7254
##   num df     : 3
##   denom df   : 9462.113
##   p.value    : 0
##
##   Result     : Difference is statistically significant.
## ------------------------------------------------------------
```

Wie durch den Welch-F-Test bestimmt, gibt es einen statistisch signifikanten Unterschied zwischen den Gruppen. Die Nullhypothese wird abgelehnt und H1 akzeptiert.

Es wird sichtbar, dass der Signifikanzwert bei null liegt und somit ein statistisch signifikanter Unterschied in der Anzahl der Rentals in verschiedenen Jahreszeiten besteht. Um herauszufinden, in welcher Jahreszeit sich die Anzahl der verliehenen Fahrräder unterscheidet, wird der Bonferroni post hoc Test verwendet.

```
pairwise.t.test(bike_sharing_data$rentals, bike_sharing_data$season, p.adjust.method = "bonferroni")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  bike_sharing_data$rentals and bike_sharing_data$season
##
##          Winter Frühling Sommer
## Frühling <2e-16 -        -
## Sommer   <2e-16 6e-13    -
## Herbst   <2e-16 0.072    <2e-16
##
## P value adjustment method: bonferroni
```

Der Bonferroni post hoc Test zeigt, dass sich zwischen Winter, Frühling und Herbst die Anzahl an vermieteten Fahrrädern statistisch signifikant unterscheidet. Zwischen Sommer und Frühling auch, jedoch geringer. Zwischen Sommer und Herbst unterscheidet sich die Anzahl an vermieteten Fahrrädern wieder statistisch signifikant. Zwischen Frühling und Herbst gibt es keinen statistisch signifikanten Unterschied (p >= 0.072).

Interpretation

Im Frühling und Herbst werden gleich viele Fahrräder verliehen, jedoch mehr Fahrräder als im Winter und etwas mehr als im Sommer.