Tutorial 3

Lars Abt

Aufgabe 1a

Zuerst laden wir die Pakete, welche wir heute benutzen möchten. Diese sind heute nicht zwingend nötig, aber bleiben dem Stil meiner anderen Tutorien treu. Zudem stelle ich euch das Paket "pacman" vor:

```
#install.packages("pacman") # falls es noch nie installiert wurde
pacman::p_load(data.table, ggplot2)
    # so müsst ihr nicht extra library(pacman) laden

# Alternativ:
library(data.table)
library(ggplot2)
```

Das "pacman" Paket ist ein Paket-Manager, der es uns erlaubt, Pakete zu installieren und zu laden, ohne dass wir uns um die Installation kümmern müssen. Wenn das Paket nicht installiert ist, wird es automatisch installiert. Auch "pacman" muss nicht extra geladen werden, da ihr den relevanten Befehl 'p_load' immer mit dem Paketnamen aufrufen könnt, und somit mit einer einzigen Zeile schnell und effizient alle Pakete installieren und laden könnt, die ihr benötigt.

So, nun zur eigentlichen Übung. Wir lesen zunächst Daten zum Schlusskurs der Amazon-Aktie ein. Dafür lesen wir über diesen Github-Link mehr Daten ein als wir benötigen, und wir anschließend nach der Amazon-Aktie.

```
# stock_prices <- read.csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/mas
stock_prices <- fread('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/
# Darauf achten, dass der gesamte Link von R als eine einzige Zeile gelesen wird!

# Für bessere Übersicht der Zeilen:
# Unter dem Reiter "Code" die Option "Soft Wrap Long Lines" aktivieren!
amazon_prices <- stock_prices[stock_prices$stock_symbol == "AMZN", ]
```

Für welchen Tag liegt die erste Beobachtung vor? Und welchen Tag die letzte Beobachtung?

```
# View(amazon_prices)
min(amazon_prices$date)
```

```
[1] "2010-01-04"
```

```
max(amazon_prices$date)
```

```
[1] "2022-12-29"
```

```
nrow(amazon_prices)
```

```
[1] 3271
```

Die erste Beobachtung stammt vom 04. Januar 2010 (welche wir mit Hilfe von 'min()' identifizieren), während die letzte Beobachtung vom 29.12.2022 datiert (identifiziert mit Hilfe von 'max()'). Insgesamt haben wir 3.271 Beobachtungen zur Amazon Aktie (äquivalent nur Anzahl an Zeilen ermittelt via 'nrow()').

Aufgabe 1b

Wir berechnen nun die Tagesrenditen der Aktie, mit Hilfe des zur Verfügung gestellten Codes:

```
amazon_returns <- (amazon_prices$adj_close[2:3271] /
                     amazon_prices$adj_close[1:3270]) - 1
head(amazon_returns)
```

```
[1]  0.005899925 -0.018115673 -0.017013233  0.027076923 -0.024041342
[6] -0.022715064
```

Mit 'head()' sehen wir uns die ersten 6 Renditen an. Diese sind alle sehr nahe an 0, was nicht überraschend ist, da die Renditen in der Regel sehr klein sind. Wir sehen auch, dass die Renditen in Prozent angegeben sind, was bedeutet, dass wir sie mit 100 multiplizieren müssen, um sie in Prozent anzugeben.

Aufgabe 1c

Um eine Verteilung für die Tagesrenditen der Amazon-Aktie zu schätzen, könnte man eine Normalverteilung verwenden. Schätzen Sie die Parameter und der Normalverteilung und plotten Sie die geschätzte Verteilung.

Erinnerung: Der Maximum-Likelihood-Schätzer für die beiden Parameter ist durch das arithmetische Mittel bzw. die empirische Standardabweichung gegeben, welche beide als Funktion in R implementiert sind!

```
mu_hat <- mean(amazon_returns)
sigma_hat <- sd(amazon_returns)
x_seq <- seq(-0.15, 0.15, by = 0.001)

mu_hat
```

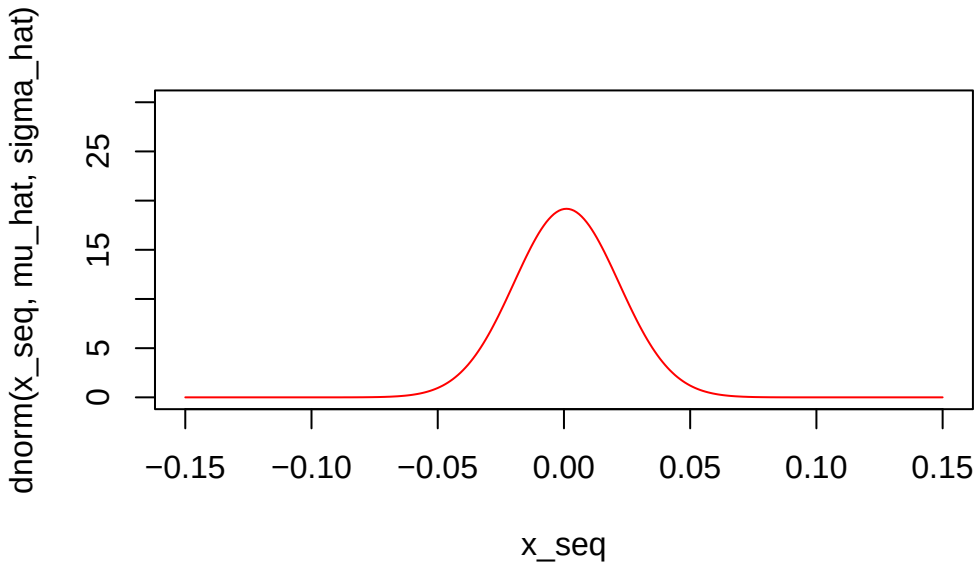
```
[1] 0.000990315
```

```
sigma_hat
```

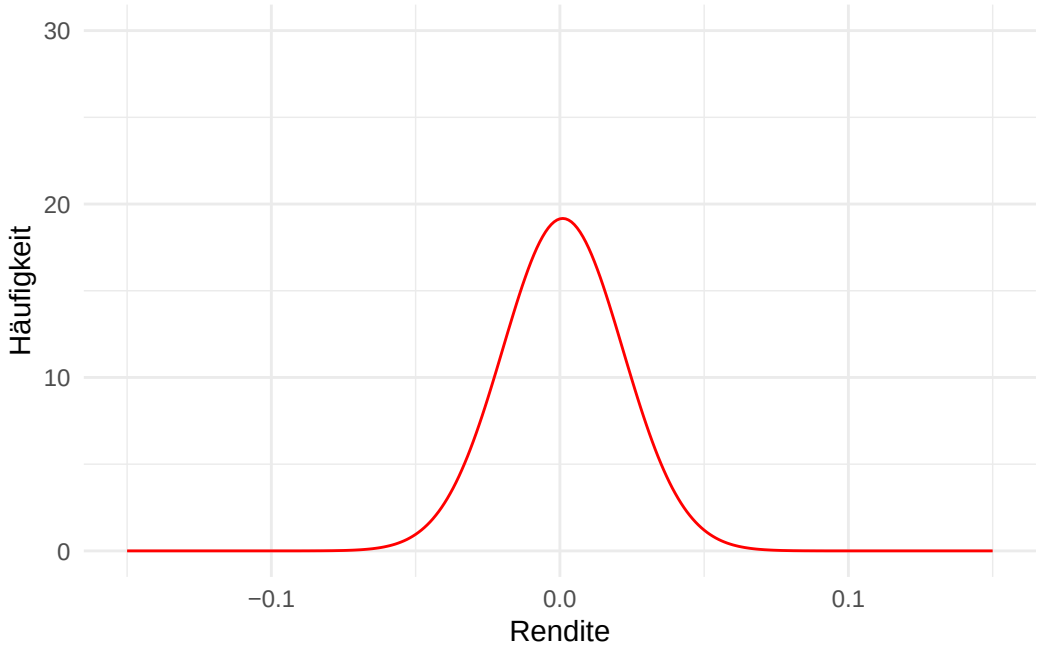
```
[1] 0.02081109
```

Unser Schätzer für liegt bei circa 0.001, was bedeutet, dass die Renditen im Durchschnitt um 0.1% steigen. Der Schätzer für liegt bei circa 0.021 und gibt die Volatilität der Renditen an. Dies bedeutet, dass die Renditen im Durchschnitt um 2.1% schwanken. Jetzt möchten wir das auch noch plotten, sowohl mit Standard-R-Befehlen, als auch mit Hilfe von ggplot2.

```
# Mit normalen R Befehlen:
plot(x_seq, dnorm(x_seq, mu_hat, sigma_hat), col = "red",
     type = "l", ylim = c(0, 30))
```




```
# Wir benutzen dnorm, welches die Dichte (density) der Normalverteilung berechnet,
  # mit unseren Schätzern für   und  .

# Mit ggplot2:
ggplot(data.table(x = x_seq, y = dnorm(x_seq, mu_hat, sigma_hat)),
       aes(x = x, y = y)) +
       geom_line(color = "red") +
       ylim(0, 30) +
       labs(x = "Rendite", y = "Häufigkeit") +
       theme_minimal()
```

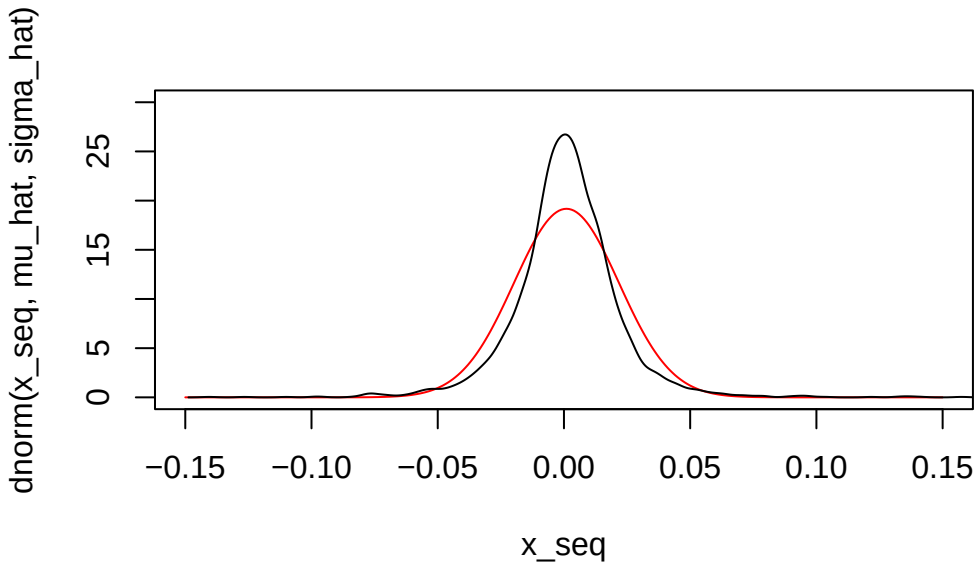


An sich sieht dies ganz gut aus. Wir haben eine Normalverteilung mit einem Maximum-Likelihood-Schätzer für die Parameter und geschätzt. Wir haben die Normalverteilung also so angepasst, dass sie die Daten bestmöglich beschreibt. Das ist aber nicht immer sinnvoll! Wir sollten uns auch die Daten anschauen, um zu sehen, ob eine Normalverteilung überhaupt sinnvoll ist, was wir in diesem Fall gar nicht erst getan haben. Dies entspricht nicht der guten wissenschaftlichen Praxis, und sollte für gewöhnlich so nicht umgesetzt werden!

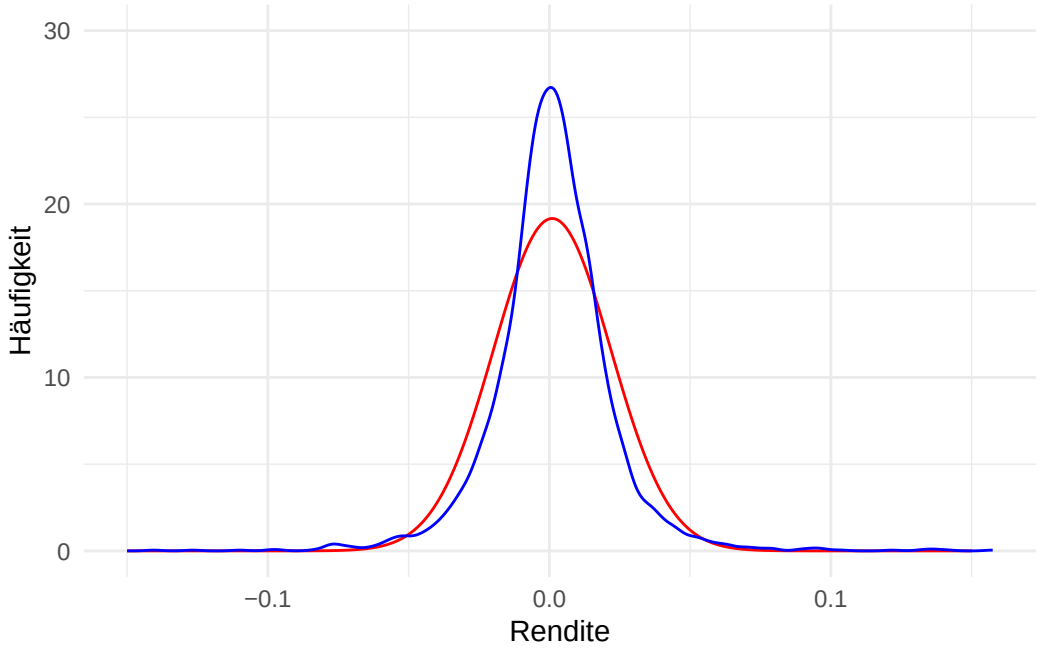
Aufgabe 1d

Jetzt möchten wir unsere Normalverteilung mit einer Kerndichteschätzung zu den Daten vergleichen, um mögliche Schwächen zu identifizieren:

```
# Mit normalen R Befehlen:
plot(x_seq, dnorm(x_seq, mu_hat, sigma_hat), col = "red",
     type = "l", ylim = c(0, 30)) # wie oben
lines(density(amazon_returns))
```




```
  # Wir benutzen density, welches die Kerndichteschätzung berechnet.

# Mit ggplot2:
ggplot() +
  geom_line(data = data.table(x = x_seq, y = dnorm(x_seq, mu_hat, sigma_hat)),
            aes(x = x, y = y), color = "red") + # wie oben
  geom_density(data = data.table(amazon_returns = amazon_returns),
            aes(x = amazon_returns), color = "blue") +
  ylim(0, 30) +
  labs(x = "Rendite", y = "Häufigkeit") +
  theme_minimal()
```



Der Gipfel der Normalverteilung (in rot) liegt unter dem des Kerndichteschätzers: Es kommen daher scheinbar viel mehr Beobachtungen sehr nah um die 0 herum vor, als die Normalverteilung erwarten würde. An den äußeren Rändern (insbesondere links) ist die Dichte des Kerndichteschätzers größer als die Dichte der Normalverteilung. Sehr negative returns scheinen also plausibler zu sein, als unter der Normalverteilung angenommen.

Aufgabe 1e

Stark negative Renditen von weniger als -5% werden von der Normalverteilung als weniger wahrscheinlich eingeschätzt (unterschätzt) als vom Kerndichteschätzer. Würde man in der Praxis auf die Normalverteilung vertrauen, würde man "zu selten" von negativen returns von $<$ -5% ausgehen. Diese kommen häufiger vor als von der Normalverteilung impliziert. Das heißt: Man würde das Risiko, dass die Aktie an einem Tag um mehr als -5% sinkt, unterschätzen, wobei genau ein solches Risiko mit einer Kennziffer wie dem value-at-risk quantifiziert werden soll. Dementsprechend sehen wir, dass eine Verteilungsannahme häufig Lücken und Schwächen aufweisen kann, welche der Kerndichteschätzer aufdecken kann, da er nicht an eine bestimmte Verteilung gebunden ist. Dies ist ein sehr wichtiger Punkt, den wir bei der Arbeit mit Verteilungen beachten müssen. Wir sollten uns immer auch die Daten anschauen, um zu sehen, ob unsere Verteilungsannahme sinnvoll ist. Wenn nicht, sollten wir entweder eine andere Verteilung verwenden oder einen Kerndichteschätzer verwenden, um die Daten besser zu beschreiben.

Aufgabe 2a

In dieser Aufgabe beschäftigen wir uns mit Daten zu Fahrrädern, die in Washington D.C. ausgeliehen wurden. Da diese Werte immer ganzzahlig sind, erscheint es sinnvoll, dass wir mit der Poisson-Verteilung arbeiten, und dementsprechend für eine Schätzung auf die Poisson-Regression zurückgreifen. Wir beginnen damit, dass wir die Daten einlesen:

```
# bikes <- read.csv("hhttps://tinyurl.com/5bkjupn2")
bikes <- fread("https://tinyurl.com/5bkjupn2")
```

Jetzt möchten wir eine Poisson-Regression anpassen:

```
bikes$weekend = as.factor(bikes$weekend)
# Hier nicht zwingend notwendig, da Weekend bereits 0/1 kodiert ist.
# Andernfalls nötig, da die Poisson-Regression nur mit Faktoren funktioniert.
# Bei der Interpretation unbedingt beachten,
# welche Stufe als Baseline verwendet wird.

head(bikes)
```

```
   temperature windspeed weekend count
         <int>     <num>  <fctr> <int>
1:          50     10.07       1    51
2:          46      6.49       0    81
3:          48      5.82       0    86
4:          51      6.49       0   121
5:          49      6.26       0   121
6:          44      8.05       0   104
```

```
dim(bikes)
```

```
[1] 365   4
```

Diese Befehle sind nicht zwingend notwendig, dennoch ist eine genauere Übersicht und ein besseres Verständnis immer hilfreich. Wir sehen, dass wir insgesamt 365 Zeilen und 4 Spalten haben. Es handelt sich also um einen Überblick für ein ganzes Jahr, welcher angibt, welche Temperatur herrschte, welche Windgeschwindigkeit vorlag, ob es ein Wochenendtag war, sowie die Anzahl an geliehenen Fahrrädern für den jeweiligen Tag. Dies möchten wir nun nutzen, um eine Poisson-Regression zu schätzen. Wir verwenden die Anzahl an geliehenen Fahrrädern als Zielvariable, und die Temperatur, Windgeschwindigkeit und ob es ein Wochenende war als erklärende Variablen.

```r
poisson <- glm(count ~ temperature + windspeed + weekend,
               family = poisson, data = bikes)
# family = poisson ist entscheidend, da sonst eine andere Regression geschätzt wird!

summary(poisson)
```

```
Call:
glm(formula = count ~ temperature + windspeed + weekend, family = poisson,
    data = bikes)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.3782223  0.0253155  172.95   <2e-16 ***
temperature  0.0162645  0.0003077   52.86   <2e-16 ***
windspeed   -0.0132829  0.0012019  -11.05   <2e-16 ***
weekend1    -0.3410637  0.0099013  -34.45   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 10853.3  on 364  degrees of freedom
Residual deviance:  5926.8  on 361  degrees of freedom
AIC: 8425.8

Number of Fisher Scoring iterations: 4
```

Wir schätzen unser Poisson-Regressionsmodell. Alle drei möglichen Effekte werden dabei als signifikant eingestuft, auf einem Signifikanzniveau von 0.001, wobei Temperatur einen positiven Einfluss hat, während die Windgeschwindigkeit und Wochenenden einen negativen Effekt aufweisen, wie die Vorzeichen zeigen. Zudem sehen wir, dass "weekend1" als Koeffizient angegeben wird, was bedeutet, dass "weekend = 0" die Baseline ist. Dies bedeutet, dass Wochentage der "Standard" sind, und der Effekt von Wochenenden (im Unterschied zu Wochentagen) hier geschätzt wird. Andersherum ist dies auch möglich, allerdings sieht der geschätzte Effekt dann logischerweise anders aus. Dies ist sehr wichtig für die Interpretation!

Aufgabe 2b

```r
exp(poisson$coef[2])
```

```
temperature
   1.016397
```

```r
exp(poisson$coef[3])
```

```
windspeed
0.9868049
```

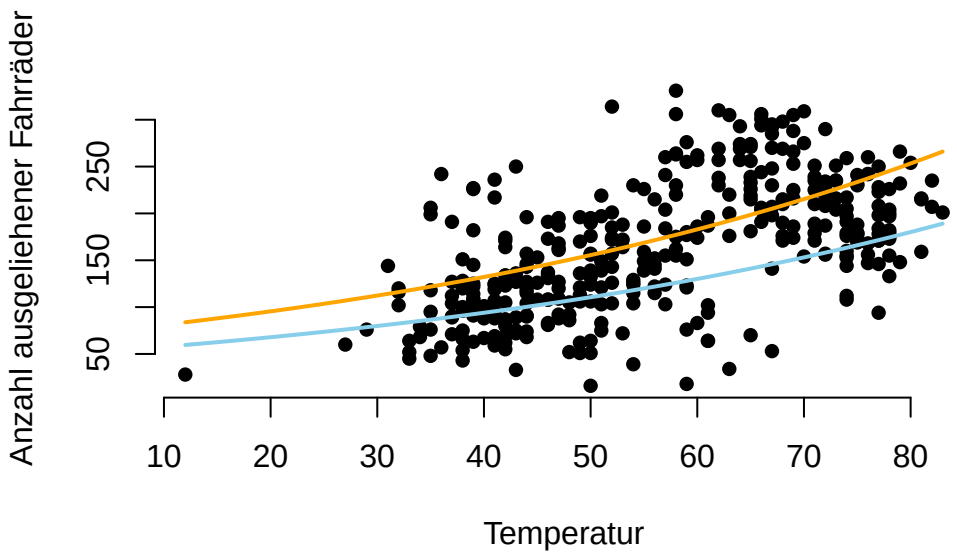
```r
# Diese Schreibweise ermöglicht es uns, direkt die Koeffizienten aus unserem
# summary anzusprechen. Natürlich ist es ebenso möglich, dass wir die Werte
# einfach kopieren. Ändern wir dann jedoch noch einmal unser Mödell, müssen wir
# hier die Zahlen nicht erneut anpassen. Dementsprechend ist diese Methode zu
# empfehlen. Es ist lediglich wichtig, dass wir die Koeffizienten in der
# richtigen Reihenfolge ansprechen. Dies ist in diesem Fall 2 und 3, da der
# erste Koeffizient der Intercept ist, und die anderen beiden die Temperatur
# und Windgeschwindigkeit beschreiben. Glücklicherweise zeigt unser Output auch
# die Namen der Koeffizieten an, sodass es leichter nachzuvollziehen ist.
```

Bei der Interpretation von Koeffizieten in einem Poisson-Modell müssen wir beachten, dass die Koeffizienten den Einfluss auf den Logarithmus der Anzahl an geliehenen Fahrrädern beschreiben. Um den Einfluss auf die Anzahl an geliehenen Fahrrädern zu erhalten, müssen wir also den Exponentialwert des Koeffizienten nehmen. Diese Werte, 1.0163 und 0.987, bedeuten, dass sich die Anzahl an geliehenen Fahrrädern um den Faktor 1.0163 erhöht, wenn die Temperatur um 1 Grad Fahrenheit steigt, und sich um den Faktor 0.987 verringert, wenn die Windgeschwindigkeit um 1 Meile/Stunde steigt. Dieser multiplikative Effekt ist sehr wichtig zu beachetn, da er uns sagt, dass die Anzahl an geliehenen Fahrrädern nicht linear von der Temperatur und der Windgeschwindigkeit abhängt. Dies ist ein sehr wichtiger Punkt, den wir bei der Interpretation von Poisson-Modellen beachten müssen, und eine sehr häufige Fehlerquelle!

Aufgabe 2c

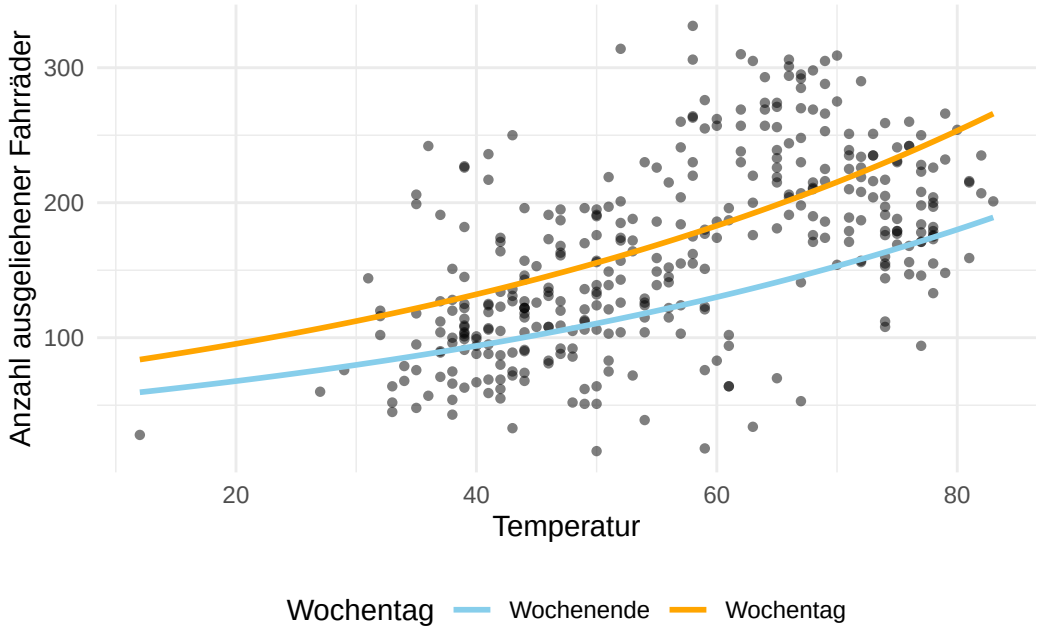
Nun möchten wir unsere Schätzung gerne plotten, und zwar über den Scatterplot all unserer Beobachtungen. Dabei möchten wir die Temperatur auf der x-Achse und die Anzahl an geliehenen Fahrrädern auf der y-Achse darstellen. Wir möchten zudem die Vorhersage für Wochentage und Wochenenden darstellen, um zu sehen, wie sich diese unterscheiden. Die Windgeschwindigkeit vernachlässigen wir hier, und nutzen einfachen das arithmetische Mittel, um eine Vorhersage zu erstellen.

```r
# Mittlere Windgeschwindigkeit:
wind <- mean(bikes$windspeed)

# Mit normalen R Befehlen:
plot(bikes$temperature, bikes$count, bty = "n", pch = 16,
     xlab = "Temperatur", ylab = "Anzahl ausgeliehener Fahrräder")
curve(exp(poisson$coef[1] + poisson$coef[2] * x + poisson$coef[3] * wind +
            poisson$coef[4] * 0), add = TRUE, lwd = 2, col = "orange") # Wochentage
curve(exp(poisson$coef[1] + poisson$coef[2] * x + poisson$coef[3] * wind +
            poisson$coef[4] * 1), add = TRUE, lwd = 2, col = "skyblue") # Wochenenden
```




```r
# Mit ggplot2 (etwas komplizierter in diesem Fall):
temp_range <- seq(min(bikes$temperature, na.rm = TRUE),
                  max(bikes$temperature, na.rm = TRUE),
                  length.out = 100)

# Daten für Wochentage
curve_data_weekday <- data.table(
  temperature = temp_range,
  count = exp(poisson$coef[1] + poisson$coef[2] * temp_range +
              poisson$coef[3] * wind + poisson$coef[4] * 0),
  weekday = "Wochentag"
```

```
)

# Daten für Wochenenden
curve_data_weekend <- data.table(
  temperature = temp_range,
  count = exp(poisson$coef[1] + poisson$coef[2] * temp_range +
              poisson$coef[3] * wind + poisson$coef[4] * 1),
  weekday = "Wochenende"
)

# Daten für die Vorhersagen verbinden
curve_data <- rbind(curve_data_weekday, curve_data_weekend)

# Erstellen des Plots
ggplot() +
  geom_point(data = bikes, aes(x = temperature, y = count),
             shape = 16, alpha = 0.5) +
  geom_line(data = curve_data, aes(x = temperature, y = count, color = weekday),
            linewidth = 1) +
  scale_color_manual(values = c("Wochentag" = "orange", "Wochenende" = "skyblue")) +
  labs(x = "Temperatur", y = "Anzahl ausgeliehener Fahrräder",
       color = "Wochentag") +
  theme_minimal() +
  theme(legend.position = "bottom")
```



Wir sehen, dass am Wochenende durchweg weniger Fahrräder ausgeliehen werden, als an Wochentagen. Dies ist nicht überraschend, da die meisten Menschen an Wochentagen zur Arbeit fahren und am Wochenende weniger Fahrräder ausleihen. Auch die Temperatur hat einen positiven Einfluss auf die Anzahl an ausgeliehenen Fahrrädern, was ebenfalls nicht überraschend ist, beides deckt sich mit den Ergebnissen unserer Regression. Außerdem zeigt dieser Plot sehr schön, dass der Effekt des Wochenendes/Wochentages kein linearer ist, also der Graph nicht einfach nur verschoben wird, sondern der Abstand mit zunehmender Anzahl ausgeliehener Fahrräder wächst, da es sich um einen multiplikativen Effekt handelt! In diesem Fall ist der Plot, welcher mit ggplot2 erstellt wurde, etwas komplizierter, da wir die Daten für die Vorhersagen in zwei separate data.tables aufteilen mussten, um sie dann zusammenzufügen. Dies ist nicht immer nötig, aber in diesem Fall war es notwendig, um den Plot zu erstellen. Zumindest sieht der fertige Plot dafür etwas besser aus.

Aufgabe 2d

Für kommenden Samstag sind eine Temperatur von 40 Grad Fahrenheit und eine Windgeschwindigkeit von 20 Meilen pro Stunde vorhergesagt. Dies möchten wir nutzen, um eine Schätzung und Vorhersage der ausgeliehenen Fahrräder für diesen Tag abzugeben. Dafür setzen wir die Werte in unsere Schätzung ein, und berechnen einen Wert für , den Parameter der Poissonverteilung.

```r
lambda_estimate <- exp(poisson$coef[1] + poisson$coef[2] * 40 +
                         poisson$coef[3] * 20 + poisson$coef[4] * 1)
lambda_estimate
```

```
(Intercept)
   83.26928
```

Wir erhalten eine Schätzung von 83.3. Dies ist unsere Schätzung für den Parameter der Poissonverteilung, die wir für die Anzahl an ausgeliehenen Fahrrädern verwenden möchten.

Aufgabe 2e

Wie wahrscheinlich ist es, gemäß unserer Schätzung, dass mehr/weniger als 75 (oder mehr/weniger als 100) Fahrräder an diesem Samstag ausgeliehen werden?

```r
ppois(q = 75, lambda = lambda_estimate)      # weniger als 75
```

```
[1] 0.1986084
```

```r
1 - ppois(q = 75, lambda = lambda_estimate)  # mehr als 75
```

```
[1] 0.8013916
```

```r
ppois(q = 100, lambda = lambda_estimate)     # weniger als 100
```

```
[1] 0.9675457
```

```r
1 - ppois(q = 100, lambda = lambda_estimate) # mehr als 100
```

```
[1] 0.03245435
```

Wir sehen also, dass die Wahrscheinlichkeit, dass weniger als 75 Fahrräder ausgeliehen werden, bei circa 19,9% liegt (die Wahrscheinlichkeit für mehr als 75 Fahrräder dann bei circa 80,1%), und die Wahrscheinlichkeit, dass mehr als 100 Fahrräder ausgeliehen werden, bei lediglich 3,2% liegt (die Wahrscheinlichkeit für weniger als 100 somit bei circa 96,8%). Dies ist auch realistisch, da der Parameter bei 83,3 liegt, und dieser für die Poisson-Verteilung sowohl den Erwartungswert als auch die Varianz beschreibt. Da 75 näher an 83,3 liegt, ist der zugehörige Wert logischerweise näher an 50%.