Regressions- und Zeitreihenanalysen zu myPoints

Hans Rudolf Keller, Helmut Gehrer (Transferbericht Gruppe 3)

11. September 2020

Inhaltsverzeichnis

1 Zusammenfassung

Dieser Regressions- und Zeitreihenanalyse liegt ein gefiltertes Dataset von rund 1 Million Beobachtungen zugrunde.

Die Analysen führten zur Erkenntnis, dass

- Sportart, Dauer und Geschlecht Einfluss auf den Kalorienverbrauch haben.
- Montag der inaktivste Tag in Bezug auf sportliche Aktivitäten ist.

2 myPoints

2.1 Lancierung

Als Krankenversicherer hat sich Visana nicht nur dazu verpflichtet, ihre Kundinnen und Kunden vor den finanziellen Folgen von Krankheit und Unfall zu schützen, sondern sie präventiv bei einem gesunden, aktiven Lebensstil zu unterstützen. Zum einen ist moderate Bewegung gesund und tut unserem Körper gut. Wer sich zum anderen regelmässig bewegt, muss weniger oft zum Arzt und sorgt dafür, dass die Gesundheitskosten nicht noch stärker steigen. Denn Fakt ist: Bewegungsmangel verursacht in der Schweiz jedes Jahr 2,1 Millionen Erkrankungen, 2,4 Milliarden Franken direkte Behandlungskosten sowie mindestens 2900 vorzeitige Todesfälle (Quelle: Bundesamt für Sport, 2013). Mit dem Bonusporgramm myPoints setzt Visana gezielt Anreize, mehr Verantwortung für die eigene Gesundheit zu übernehmen. Sie belohnt ihre Versicherten dafür mit bis zu 120 Franken pro Jahr.

2.2 Voraussetzungungen

Die Teilnahme bei myPoints ist kostenlos. Voraussetzungen sind

- eine Visana-Zusatzversicherung,
- die Nutzung der Visana-App und des Online-Kundenportals myVisana
- Mindestalter von 12 Jahren
- Wohnsitz in der Schweiz

Optimal ist die Aktivierung einer Gesundheitsapp wie «Apple Health» (iOS) oder «Google Fit» (Android) auf dem Smartphone, mit neueren Releases der App ist aber auch direkte Synchronisation mit Garmin Connect, Fitbit oder Polar möglich.

2.3 Punkte sammeln

2.3.1 Tägliche Bewegung

Mit den genannten Apps zeichnen unsere Kundinnen und Kunden ihren täglichen Bewegungsumfang via Smartphone auf. Je mehr Schritte im Alltag gemacht respektive je mehr Kalorien verbrannt werden, desto schneller wächst das Punktekonto bei myPoints an. Egal ob man Treppen steigt, mit dem Fahrrad zur Arbeit fährt oder abends noch eine Runde Nordic Walken geht. All diese Aktivitäten sind Geld wert. Ab 5’000 Schritten oder 200 verbrannten Kalorien gibt es Punkte. Um eine regelmässige Bewegung zu fördern, gilt pro Woche ein Punktemaximum.

2.3.2 Kundentreue

Mit ihrem Bonusprogramm belohnt Visana aber nicht nur die Bewegungsmenschen unter ihren Versicherten. myPoints ist so gestaltet, dass grundsätzlich alle daran teilnehmen können. Also auch Menschen, die gerade verletzt oder nur eingeschränkt mobil sind. Sie sammeln Punkte, indem sie zum Beispiel Mehrjahresverträge oder eine Sachversicherung bei Visana abschliessen, das Online-Portal myVisana nutzen oder bei der Aktion «Kunden werben Kunden» mitmachen.

So können Versicherte genauso auf das Punkttotal kommen wie über Bewegungsaktivitäten.

2.4 Datenschutz

Für myPoints werden nebst der Anzahl Schritte und Kalorien keine weiteren Daten verwendet. Um das Bonusprogramm zu optimieren und noch kundenfreundlicher zu gestalten, können die anonymisierten Daten zusätzlich ausgewertet werden. Für die Ermittlung der Schritt und Kaloriendaten gelten die Bedingungen der oben genannten Fitness-Apps.

Im Rahmen dieser Analyse stehen nur anonymisierte Daten zur Verfügung.

Details zum neuen digitalen Bonusprogramm von Visana, inkl. FAQ, Erklärfilm und Teilnahmebedingungen, finden Sie auf www.visana.ch/mypoints.

3 Verfügbare Daten

Aus den verfügbaren Daten wurden nur diejenigen extrahiert, welche für die Aufgabenstellung ausgewählt worden sind. Informationen über Schritte bzw. Loyalität sind in den bereitgestellten Datasets nicht vorhanden.

3.1 members

Members enthält einen Teil der Stammdaten der Teilnehmer.

Insgesamt sind **33'666 Beobachtungen** von Teilnehmern mit 7 Attributen vorhanden.

Attribut	Format	Bedeutung
id	chr (UUID)	nicht sprechende, generierte Id des Partners
onboarded_at	Date	Datum des Beginns der Teilnahme an myPoints
state	Factor	Aktueller Status des Teilnehmers: "ACTIVE": Aktiver Teilnehmer; "INACTIVE": Familienoberhaupt eines anderen Teilnehmers. sammelt lediglich Loyalitätspunkte; "DELETED": gelöschte Teilnahme
os	Factor	Betriebssystem des Teilnehmers: "ANDROID", "IOS", "" (nur bei state: INACTIV möglich)
year_of_birth	int	Jahrgang des Teilnehmers ; bei inaktiven Teilnehmern sind nullwerte möglich.
gender	Factor	Geschlecht des Teilnehmers: "f" female "m" male
municipal_code	int	Gemeindenummer (gem. BFS) des Wohnorts des Teilnehmers

```
summary(members)
```

```
##       id             onboarded_at            state              os
##  Length:33666       Min.   :2019-05-10   ACTIVE  :31454          : 2160
##  Class :character   1st Qu.:2019-09-23   DELETED :   53   ANDROID:12620
##  Mode  :character   Median :2019-10-21   INACTIVE: 2159   IOS    :18886
##                     Mean   :2019-11-20
##                     3rd Qu.:2020-01-15
##                     Max.   :2020-08-17
##
##  year_of_birth  gender    municipal_code
##  Min.   :1900   f:17850   Min.   :   1
##  1st Qu.:1970   m:15816   1st Qu.: 371
##  Median :1981             Median : 955
##  Mean   :1979             Mean   :1957
##  3rd Qu.:1989             3rd Qu.:2939
##  Max.   :9999             Max.   :6810
##  NA's   :134              NA's   :50
```

3.2 activities

Activities enthält jede einzelne Aktivität.

Aufgrund der Eigenschaft der App auch nicht explizit aufgezeichnete Aktivitäten als “OTHER” hochzuladen, enthält das ursrüngliche Dataset einen grossen Anteil Daten, welche nicht sinnvoll ausgewertet werden können. Zudem werden manuelle Eingaben erlaubt. Auch sind die Messungen einiger Fitness-Gadgets nicht sonderlich zufriedenstellend. Daher wurden die Daten vorgängi gemäss folgenden Filtern bereinigt:

- Einheit = Kalorien
- Kalorien > 50 und <= 6’000
- Dauer >= 10 Minuten und <= 10 Stunden

Nach dieser Bereinigung enthält das Dataset noch immer **926’785 Beobachtungen** mit 11 Attributen.

Attribut	Format	Bedeutung
begin_at	POSIXct	Datum / Uhrzeit des Beginns der Aktivität
end_at	POSIXct	Datum / Uhrzeit des Endes der Aktivität
duration	num	Dauer der Aktivität in Minuten
category	Factor	Kategorisierung der Sportart nach folgenden Obergruppen: Ballsport, Cardio, Kampfsport, Kraft, Laufen, Radsport, Schiessen, Varia, Wassersport, Wintersport
type	Factor	Art der Aktivität. Die folgenden Werte sind möglich: AEROBICS, ARCHERY, BADMINTON, BASEBALL, BASKETBALL, BIATHLON, BIKING, BOWLING, BOXING, CALISTHENICS, CIRCUIT_TRAINING, CORE_TRAINING, CRICKET, CROSSFIT, CURLING, DANCING, DIVING, ELEVATOR, ELLIPTICAL, ERGOMETER, FENCING, FISHING, FLEXIBILITY, FUNCTIONAL_STRENGTH, GARDENING, GOLF, GYMNASTICS, HANDBALL, HIKING, HOCKEY, HORSEBACK_RIDING, HUNTING, ICE_SKATING, INTERVAL_TRAINING, JUMP_ROPE, KAYAKING, KETTLEBELL_TRAINING, KICK_SCOOTER, KICKBOXING, KITESURFING, LACROSSE, MARTIAL_ARTS, MIXED_METABOLIC_CARDIO, P90X, PADDLE_SPORTS, PARAGLIDING, PILATES, PLAY, POLO, RACQUETBALL, ROCK_CLIMBING, ROWING, RUNNING, SAILING, SCUBA_DIVING, SKATEBOARDING, SKATING, SKIING, SLEDDING, SNOW_SPORTS, SNOWBOARDING, SNOWSHOEING, SOFTBALL, SQUASH, STAIR_CLIMBING, STAIRS, STANDUP_PADDLEBOARDING, STRENGTH_TRAINING, SURFING, SWIMMING, TABLE_TENNIS, TRACK_AND_FIELD, TREADMILL, VOLLEYBALL, WAKEBOARDING, WALKING, WATER_FITNESS, WATER_POLO, WATER_SPORTS, WEIGHTLIFTING, WINDSURFING, YOGA, ZUMBA
calories	int	Verbrannte Kilo-Kalorien während der Aktivität.
upload_at	POSIXct	Datum / Uhrzeit des Synchronisation (Hintergrund: Die Daten werden nur synchronisiert, wenn die App im Vordergrund läuft. Aufgrund dieses Datums kann somit ermittelt werden, wie häufig ein Benutzer seine Daten synchronisiert.

Attribut	Format	Bedeutung
member_id	chr (UUID)	nicht sprechende, generierte Id des Partners
year_of_birth	int	Jahrgang des Teilnehmers
gender	Factor	Geschlecht des Teilnehmers: “f” female “m” male
os	Factor	Betriebssystem des Teilnehmers: “ANDROID”, “IOS”

```
summary(activities)
```

```
##     begin_at                       end_at                         duration
##  Min.   :2019-05-10 09:21:53   Min.   :2019-05-10 10:23:56   Min.   : 10.00
##  1st Qu.:2019-11-12 18:12:27   1st Qu.:2019-11-12 18:42:21   1st Qu.: 14.70
##  Median :2020-01-10 12:58:25   Median :2020-01-10 13:25:51   Median : 21.85
##  Mean   :2020-01-14 09:14:31   Mean   :2020-01-14 09:49:11   Mean   : 34.66
##  3rd Qu.:2020-03-09 17:39:11   3rd Qu.:2020-03-09 18:14:55   3rd Qu.: 40.58
##  Max.   :2020-08-17 14:25:09   Max.   :2020-08-17 15:00:00   Max.   :600.00
##
##         category                    type           calories
##  Laufen     :701766   WALKING          :615637   Min.   :  51.0
##  Radsport   :137289   BIKING           :137003   1st Qu.:  70.0
##  Kraft      : 35535   RUNNING          : 75780   Median : 109.0
##  Cardio     : 25398   STRENGTH_TRAINING: 22021   Mean   : 197.6
##  Varia      : 10639   ELLIPTICAL       : 11931   3rd Qu.: 217.0
##  Wassersport:  8203   HIKING           :  9733   Max.   :6000.0
##  (Other)    :  7955   (Other)          : 54680
##    upload_at                    member_id         year_of_birth  gender
##  Min.   :2019-05-10 10:26:46   Length:926785      Min.   :1931   f:433263
##  1st Qu.:2019-11-19 20:02:17   Class :character   1st Qu.:1967   m:493522
##  Median :2020-01-21 15:37:07   Mode  :character   Median :1979
##  Mean   :2020-01-21 22:22:27                      Mean   :1977
##  3rd Qu.:2020-03-18 21:41:19                      3rd Qu.:1988
##  Max.   :2020-08-17 15:26:18                      Max.   :2008
##
##        os
##         :     0
##  ANDROID:722876
##  IOS    :203909
##
##
##
##
```

4 Regressionsanalysen

```
# Achtung: Bei grosser Menge ergibt sich ein grosses PDF aufgrund vieler
# überdeckter Datenpunke im Scatterplot. Je nach Zweck der Analyse kann
# mit dieser Variable deren Umfang angepasst werden:
sample_size <- 1000 # für Druck
#sample_size <- 10000 # ursprüngliche Analyse
#sample_size <- as.integer(count(activities)) # Gesamtes Dataset
#
# Subset auswählen für Regression -------------------------------------
tabelle <- activities[1:sample_size, ] %>%
  mutate(age = 2020 - year_of_birth) %>%
  select(calories, duration, age, gender, category)
```

> *Aufgrund dieser Menge der Aktivitäten beschränken wir die Regressionsanalyse auf die ersten **1'000 Stichproben.***

4.1 Hypothese

Kalorienverbrauch ist **abhängig von**

- Sportart (Daten vorhanden)
- Trainingsdauer (Daten vorhanden)
- Intensität (Daten nicht vorhanden)
- Geschlecht (Daten vorhanden)
- Körpergewicht (Daten nicht vorhanden)

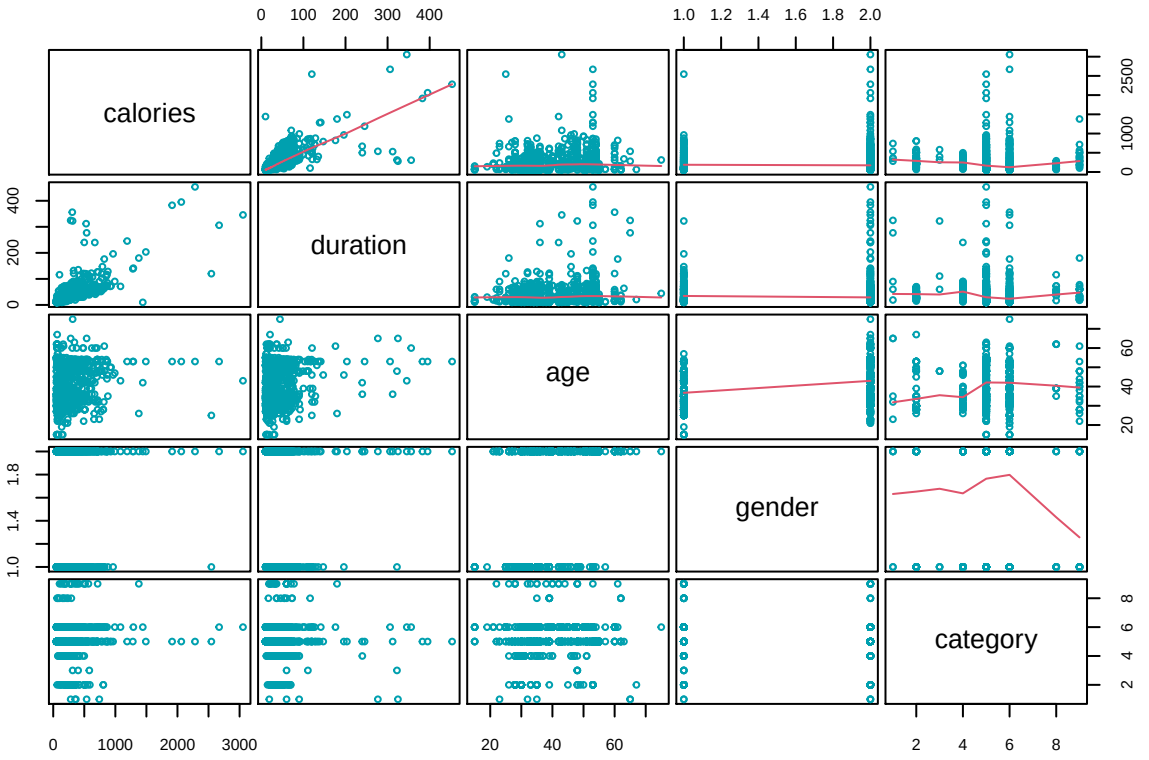
unabhängig von

- Alter (Daten vorhanden)

4.2 Scatterplot

Der folgenden Scatterplot zeigt Zusammenhänge aller relevaten Variablen untereinander auf:

```
# Variablen paarweise darstellen --------------------------------------
pairs(~calories + duration + age + gender + category,
      data = tabelle, cex = 0.6, gap = 0.3,
      cex.labels = 1.2, cex.axis = 0.7,
      upper.panel = panel.smooth,
      col=col_fill_default)
```



4.3 Regressionsmodell

```
# Regression berechnen ------------------------------------------------
# define intercept-only model: maximal restringiert
intercept_only_model <- lm(formula = calories ~ 1, data = tabelle)
# define total model - unrestringiert
total_model <- lm(formula = calories ~ ., data = tabelle)
#
# perform stepwise regression
model_forw <- step(intercept_only_model,
                   direction = "forward",
                   scope = formula(total_model))
```

```
## Start:  AIC=11210.88
## calories ~ 1
##
##             Df Sum of Sq      RSS   AIC
## + duration   1  40595421 33187512 10414
## + age        1   1502694 72280239 11192
## + category   7   1746184 72036749 11201
## <none>                   73782933 11211
## + gender     1     32551 73750382 11212
##
## Step:  AIC=10413.93
## calories ~ duration
##
##             Df Sum of Sq      RSS   AIC
```

```
## + category  7   2216809 30970703 10359
## + gender    1    132794 33054718 10412
## <none>                  33187512 10414
## + age       1     24994 33162518 10415
##
## Step:  AIC=10358.8
## calories ~ duration + category
##
##          Df Sum of Sq      RSS   AIC
## + gender  1     84775 30885928 10358
## <none>                30970703 10359
## + age     1     17240 30953463 10360
##
## Step:  AIC=10358.06
## calories ~ duration + category + gender
##
##        Df Sum of Sq      RSS   AIC
## <none>              30885928 10358
## + age   1    2270.4 30883657 10360
```

```
model_backw <- step(total_model,
                    direction = "backward",
                    scope = formula(total_model))
```

```
## Start:  AIC=10359.98
## calories ~ duration + age + gender + category
##
##            Df Sum of Sq      RSS   AIC
## - age       1      2270 30885928 10358
## <none>                  30883657 10360
## - gender    1     69805 30953463 10360
## - category  7   2168192 33051850 10414
## - duration  1  39261164 70144821 11178
##
## Step:  AIC=10358.06
## calories ~ duration + gender + category
##
##            Df Sum of Sq      RSS   AIC
## <none>                  30885928 10358
## - gender    1     84775 30970703 10359
## - category  7   2168790 33054718 10412
## - duration  1  41068226 71954154 11202
```

```
model_both <- step(intercept_only_model,
                   direction = "both",
                   scope = formula(total_model))
```

```
## Start:  AIC=11210.88
## calories ~ 1
```

```
##
##             Df Sum of Sq      RSS   AIC
## + duration   1  40595421 33187512 10414
## + age        1   1502694 72280239 11192
## + category   7   1746184 72036749 11201
## <none>                   73782933 11211
## + gender     1     32551 73750382 11212
##
## Step:  AIC=10413.93
## calories ~ duration
##
##             Df Sum of Sq      RSS   AIC
## + category   7   2216809 30970703 10359
## + gender     1    132794 33054718 10412
## <none>                   33187512 10414
## + age        1     24994 33162518 10415
## - duration   1  40595421 73782933 11211
##
## Step:  AIC=10358.8
## calories ~ duration + category
##
##             Df Sum of Sq      RSS   AIC
## + gender     1     84775 30885928 10358
## <none>                   30970703 10359
## + age        1     17240 30953463 10360
## - category   7   2216809 33187512 10414
## - duration   1  41066046 72036749 11201
##
## Step:  AIC=10358.06
## calories ~ duration + category + gender
##
##             Df Sum of Sq      RSS   AIC
## <none>                   30885928 10358
## - gender     1     84775 30970703 10359
## + age        1      2270 30883657 10360
## - category   7   2168790 33054718 10412
## - duration   1  41068226 71954154 11202
```

```r
#
# Alternative für Rückwärtselimination und Forward Selection
model_bothAIC <- step(total_model,
                      direction = "both",
                      data = tabelle, k = 2)
```

```
## Start:  AIC=10359.98
## calories ~ duration + age + gender + category
##
##             Df Sum of Sq      RSS   AIC
## - age        1      2270 30885928 10358
```

```
## <none>                  30883657 10360
## - gender    1     69805 30953463 10360
## - category  7   2168192 33051850 10414
## - duration  1  39261164 70144821 11178
##
## Step:  AIC=10358.06
## calories ~ duration + gender + category
##
##            Df Sum of Sq      RSS   AIC
## <none>                  30885928 10358
## - gender    1     84775 30970703 10359
## + age       1      2270 30883657 10360
## - category  7   2168790 33054718 10412
## - duration  1  41068226 71954154 11202
```

```
N <- NROW(total_model$resid)
model_bothBIC <- step(total_model,
                      direction = "both",
                      data = tabelle,
                      k = log(N))
```

```
## Start:  AIC=10413.97
## calories ~ duration + age + gender + category
##
##            Df Sum of Sq      RSS   AIC
## - age       1      2270 30885928 10407
## - gender    1     69805 30953463 10409
## <none>                  30883657 10414
## - category  7   2168192 33051850 10434
## - duration  1  39261164 70144821 11227
##
## Step:  AIC=10407.13
## calories ~ duration + gender + category
##
##            Df Sum of Sq      RSS   AIC
## - gender    1     84775 30970703 10403
## <none>                  30885928 10407
## + age       1      2270 30883657 10414
## - category  7   2168790 33054718 10427
## - duration  1  41068226 71954154 11246
##
## Step:  AIC=10402.97
## calories ~ duration + category
##
##            Df Sum of Sq      RSS   AIC
## <none>                  30970703 10403
## + gender    1     84775 30885928 10407
## + age       1     17240 30953463 10409
## - category  7   2216809 33187512 10424
```

```
## - duration  1  41066046 72036749 11240
```

4.4 Vorläufige Interpretation

Für die ersten 10'000 Zeilen gilt:

> Alle 3 Ansätze (forward, backward, both) ergeben dasselbe Ergebnis: Sportart, Trainingsdauer, Geschlecht haben einen Einfluss auf den Kalorienverbrauch; Alter hat keinen Einfluss auf den Kalorienverbrauch.

Dies ist mit den theoretischen Erwartungen kompatibel, d.h. das Modell ist plausibel.

Für alle n=926'785 Beobachtungen hat auch das Alter einen Einfluss!

4.5 Finales Regressionsmodell

```
final_model <- lm(formula = calories ~ duration + category + gender + age,
                  data = tabelle)
summary(final_model)
```

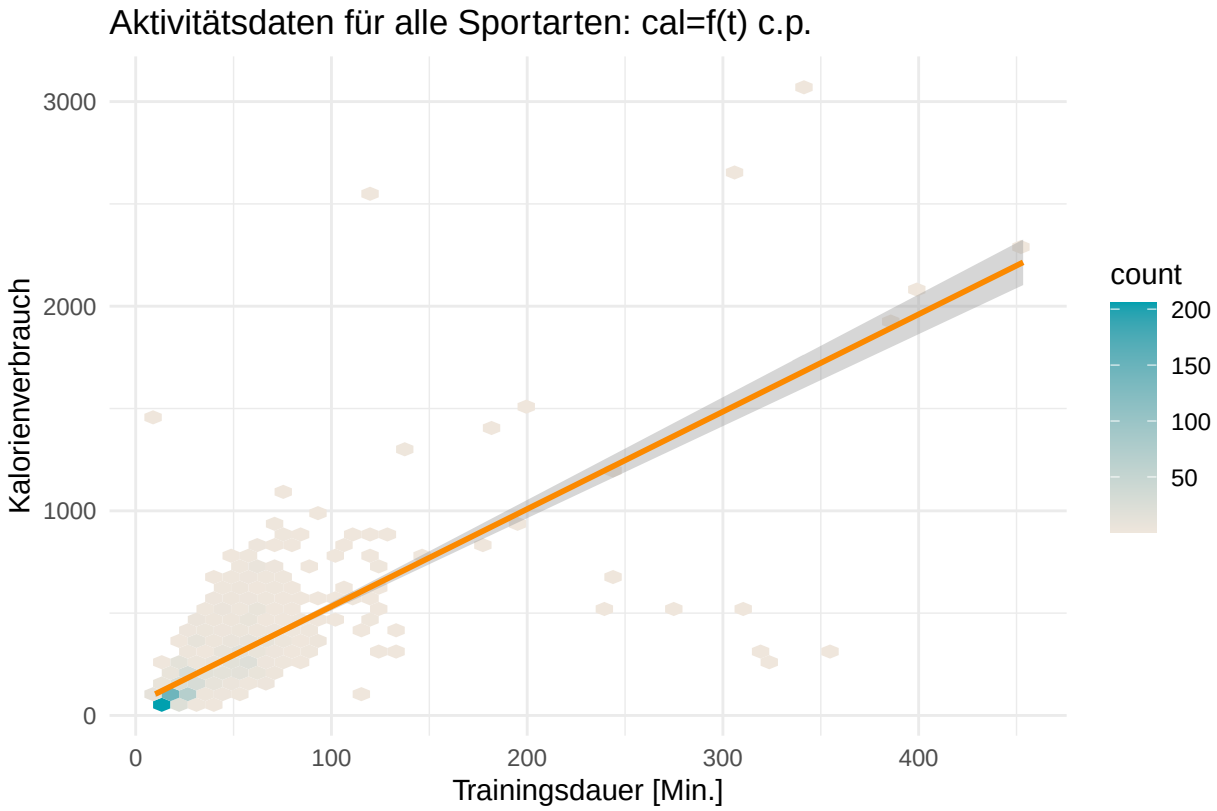
```
##
## Call:
## lm(formula = calories ~ duration + category + gender + age, data = tabelle)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1529.04   -62.94   -43.46    19.42  1903.48
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -333.8785    84.5780  -3.948 8.45e-05 ***
## duration               5.0042     0.1411  35.458  < 2e-16 ***
## categoryCardio       423.9414    83.5551   5.074 4.66e-07 ***
## categoryKampfsport   -65.1304   129.3321  -0.504 0.614661
## categoryKraft        296.6912    82.8979   3.579 0.000362 ***
## categoryLaufen       371.8093    81.0216   4.589 5.03e-06 ***
## categoryRadsport     359.4128    81.4058   4.415 1.12e-05 ***
## categoryVaria        193.6670    97.9050   1.978 0.048193 *
## categoryWassersport  456.6140    89.5358   5.100 4.07e-07 ***
## genderm               18.7035    12.5096   1.495 0.135201
## age                   0.1635     0.6065   0.270 0.787493
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 176.7 on 989 degrees of freedom
## Multiple R-squared:  0.5814, Adjusted R-squared:  0.5772
## F-statistic: 137.4 on 10 and 989 DF,  p-value: < 2.2e-16
```

```
# Das finale Modell hat p von ca. 0, R2 = 0.60
summary(final_model)$coeff
```

```
##                          Estimate  Std. Error    t value      Pr(>|t|)
## (Intercept)          -333.8785194  84.5780093 -3.9475807  8.453373e-05
## duration                5.0041808   0.1411294 35.4580923 2.269497e-178
## categoryCardio        423.9414367  83.5551274  5.0737932  4.656244e-07
## categoryKampfsport    -65.1303900 129.3321050 -0.5035903  6.146615e-01
## categoryKraft         296.6912153  82.8978668  3.5789970  3.616589e-04
## categoryLaufen        371.8093042  81.0216396  4.5890123  5.025758e-06
## categoryRadsport      359.4128006  81.4057763  4.4150774  1.120840e-05
## categoryVaria         193.6669593  97.9049540  1.9781119  4.819327e-02
## categoryWassersport   456.6139872  89.5358331  5.0997905  4.074272e-07
## genderm                18.7034917  12.5096465  1.4951255  1.352007e-01
## age                     0.1635319   0.6064799  0.2696411  7.874926e-01
```

4.6 Plot Kalorienverbrauch vs. Trainingsdauer

```
#
# Graph 1: Bins ----------------------------------------------------------
Bin_Graph <- tabelle %>%
  ggplot(aes(x = duration, y = calories)) +
  geom_hex(bins = 50) +
  geom_smooth(method = "lm", formula = y ~ x, col = col_line_alt) +
  scale_fill_gradient(low = col_background, high = col_fill_default) +
  labs(
    x = "Trainingsdauer [Min.]",
    y = "Kalorienverbrauch",
    title = "Aktivitätsdaten für alle Sportarten: cal=f(t) c.p.",
    caption = "Quelle: Visana 2020"
  ) +
  theme_minimal()
Bin_Graph
```



Quelle: Visana 2020

4.7 Test auf Normalverteilung

```
#
# Test auf Normalverteilung ------------------------------------------
# qqPlot(final_model, simulate=TRUE, id=list(method="y", n=2))
jarque.bera.test(final_model$resid)
```

```
##
##  Jarque Bera Test
##
## data:  final_model$resid
## X-squared = 38237, df = 2, p-value < 2.2e-16
```

Interpretation: Daten sind bei Extrema nicht normalverteilt

5 Zeitreihenanalysen

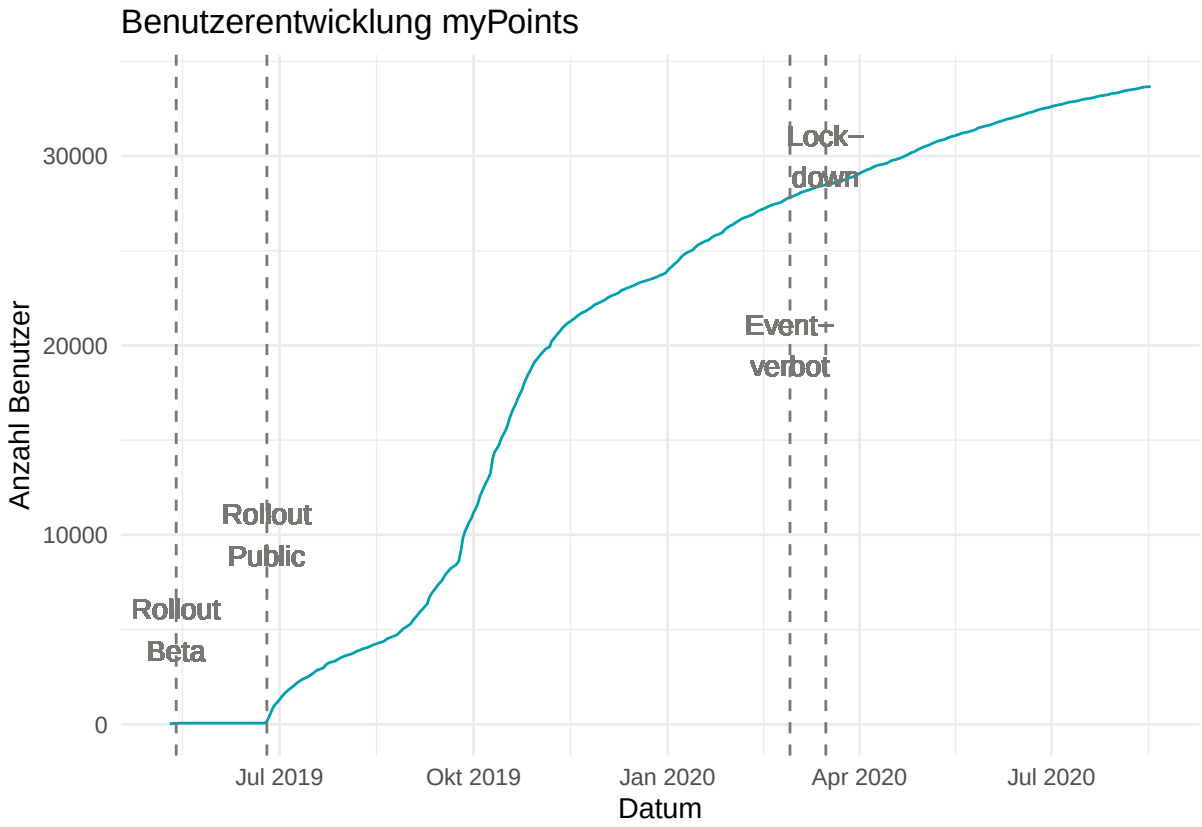
5.1 Onboarding von Teilnehmern

```
#
daily_onboardings <- members %>%
  group_by(onboarded_at) %>%
  summarise(count = n()) %>%
  complete(onboarded_at = seq.Date(min(onboarded_at),
                                   max(onboarded_at), by = "day"),
           fill = list(count = 0)) %>%
  mutate(cumulated = cumsum(count))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
# Kumulierter Benutzerzuwachs über alles
ggplot(daily_onboardings,
       aes(x = onboarded_at, y = cumulated)) +
  geom_line(color = col_line_default) +
  geom_vline(xintercept = milestones.myPoints.beta_rollout,
             color = col_notes,
             linetype = "dashed") +
  geom_text(
    aes(x = milestones.myPoints.beta_rollout, y = 5000),
    label = "Rollout\nBeta",
    color = col_notes
  ) +
  geom_vline(xintercept = milestones.myPoints.public_rollout,
             color = col_notes,
             linetype = "dashed") +
  geom_text(
    aes(x = milestones.myPoints.public_rollout, y = 10000),
    label = "Rollout\nPublic",
    color = col_notes
  ) +
  geom_vline(xintercept = milestones.covid19.major_events_forbidden,
             color = col_notes,
             linetype = "dashed") +
  geom_text(
    aes(x = milestones.covid19.major_events_forbidden, y = 20000),
    label = "Event-\nverbot",
    color = col_notes
  ) +
  geom_vline(xintercept = milestones.covid19.lockdown,
             color = col_notes,
             linetype = "dashed") +
  geom_text(aes(x = milestones.covid19.lockdown, y = 30000),
            label = "Lock-\ndown",
            color = col_notes) +
```

```
  scale_x_date() +
  theme_minimal() +
  labs(title = "Benutzerentwicklung myPoints",
       x = "Datum",
       y = "Anzahl Benutzer")
```



Ein erster Analyseversuch wurde mit dem Onboarding von Teilnehmern vorgenommen.

```
member_onboardings <- members %>%
  group_by(week = floor_date(onboarded_at, unit = "week")) %>%
  summarise(count = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
week_of_first_onboarding <- min(member_onboardings$week)
member_onboardings.ts <-    ts(member_onboardings$count,
  start = decimal_date(week_of_first_onboarding),
  frequency = 52
)
# member_onboardings.decomposed <- decompose(member_onboardings.ts)
# plot(member_onboardings.decomposed)
# member_onboardings.stl <-
#  stl(member_onboardings.ts, s.window="periodic")
```

Ein erstes Fazit nach dem Erstellen dieser Time-Series:

> Sowohl die Funktion decompose als auch stl benötigen für das Zerlegen der Daten ein **Minimum von 2 Perionden,** in unserem Beispiel also 2 Jahren. (Fehlermeldung “Zeitreihe

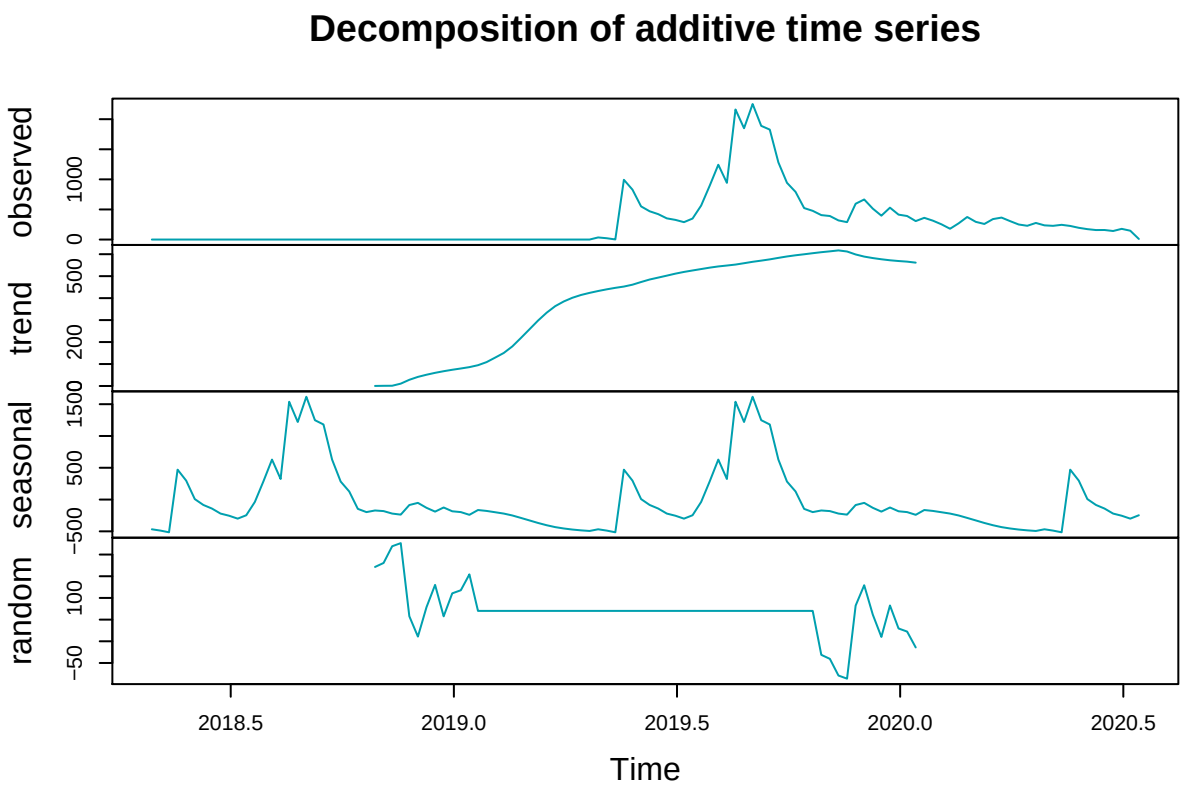
hat keine oder weniger als 2 Perioden")

Was machen wir mit unseren 18 Monaten? Hat es einen Einfluss, wenn die Daten entsprechend modifiziert werden?

Experiment:

Wie füllen die Daten entsprechend der Realität auf. Ein Jahr vor dem ersten Onboarding werden die Daten mit 0 jeweils Onbardings pro Woche ergänzt.

```
empty_year <- rep(0, 52)
onboarding_with_empty_prefix <- c(empty_year, member_onboardings$count)
week_of_first_empty_entry <-
  floor_date(week_of_first_onboarding - 365, "week")
onboardings_with_empty_prefix.ts <-
  ts(
    onboarding_with_empty_prefix,
    start = decimal_date(week_of_first_empty_entry),
    frequency = 52
  )
onboardings_with_empty_prefix.decomposed <-
  decompose(onboardings_with_empty_prefix.ts)
plot(onboardings_with_empty_prefix.decomposed, col=col_line_default)
```



5.1.1 Fazit

Effekt der Marketing-Aktion wird als saisonale Komponente ausgewiesen. Trend und Random sind vorne sowie hinten stark beschnitten, spiegeln somit die korrekten Orignaldaten.

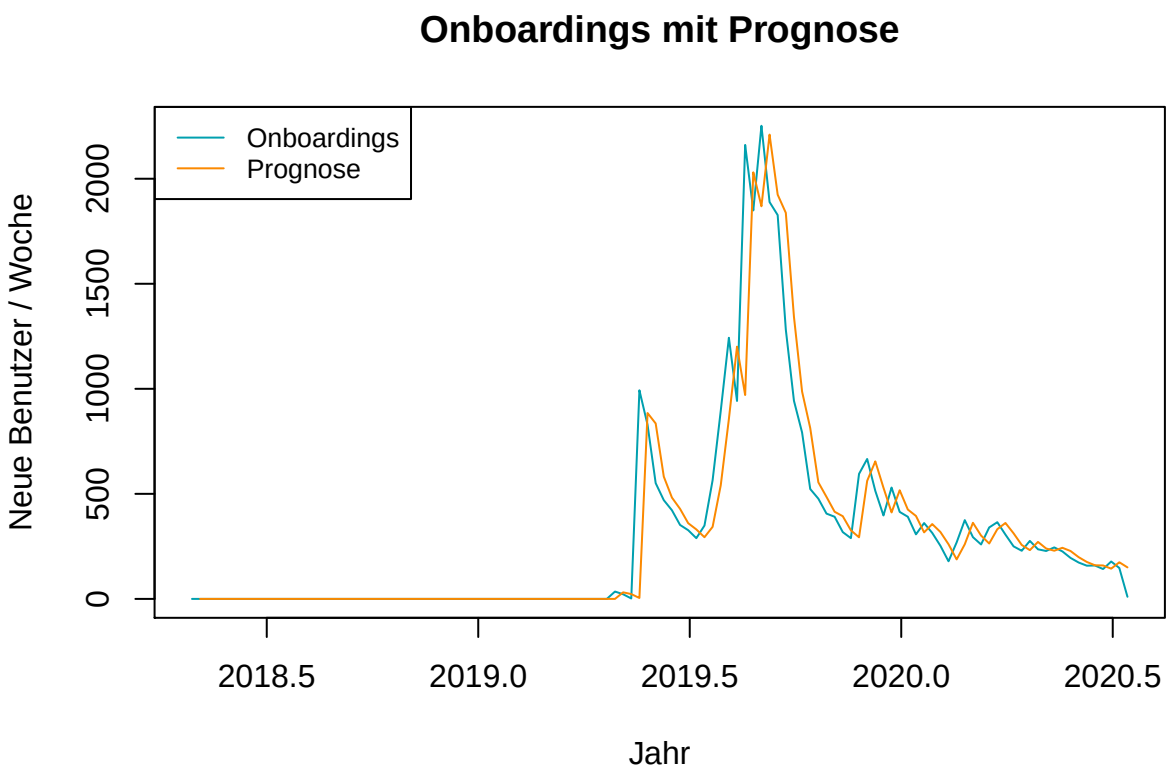
Aufgrund des kurzen Erhebungszeitraums der Daten ist keine sinnvolle Analyse von saiso-

nalen Daten möglich.

5.1.2 Prognose mit Holt-Verfahren

Den Daten ist kein klarer Trend zu entnehmen. Es ist zwar - im Rahmen der Marketing-Aktion im Herbst 2019 - ein klarer Anstieg der Onboardings zu verzeichnen, dieser flacht aber rasch wieder ab. Daher wird für die Glättung bzw. Prognose das Holt-Verfahren ohne beta und gamma gewählt. Da die Glättung aufgund dieser Konstellation den Originaldaten folgt, wird auf deren Darstellung verzichtet.

```
onboardings_with_empty_prefix.hw <-
  HoltWinters(onboardings_with_empty_prefix.ts,
              beta = FALSE,
              gamma = FALSE)
plot(
  onboardings_with_empty_prefix.ts,
  col = col_line_default,
  main = "Onboardings mit Prognose",
  xlab = "Jahr",
  ylab = "Neue Benutzer / Woche"
)
lines(onboardings_with_empty_prefix.hw$fitted[, 2], col=col_line_alt)
legend("topleft", legend=c("Onboardings", "Prognose"),
       col=c(col_line_default, col_line_alt), lty=1:1, cex=0.8)
```



5.2 Aktivitäten

Analyse 2 beschäftigt sich mehr mit Aktivitägsdaten. Insbesondere soll herausgefunden werden ob **wöchentliche** saisonale Tendenzen bestehen.

Da die Default-TS-Klasse die X-Achse inkorrekt aufbaut, wenn mit einer frequency von 7 gearbeitet wird, wird für diese Tests auf die Klasse msts aus der Library forecast zurückgegriffen.

```
first_activity_begin_date <-
  as.Date(trunc(min(activities$begin_at), "day"))
last_activity_begin_date <-
  as.Date(trunc(max(activities$begin_at), "day"))

# Daten aufbereiten; Dabei wird
# - der Beginntimestamp auf Datum gekürzt und
# - zur Gruppierung verwendet.
# - Mit complete werden allfällige fehlende Tage mit 0 ergänzt
# - zum Schluss noch alles nach Datum sortiert.
daily_activities <- activities %>%
  mutate(activity_date = as.Date(trunc(begin_at, "day"))) %>%
  group_by(activity_date) %>%
  summarise(count = n()) %>%
  complete(
    activity_date = seq.Date(first_activity_begin_date,
                             last_activity_begin_date,
                             by = "day"),
    fill = list(count = 0)) %>%
  arrange(activity_date)
```

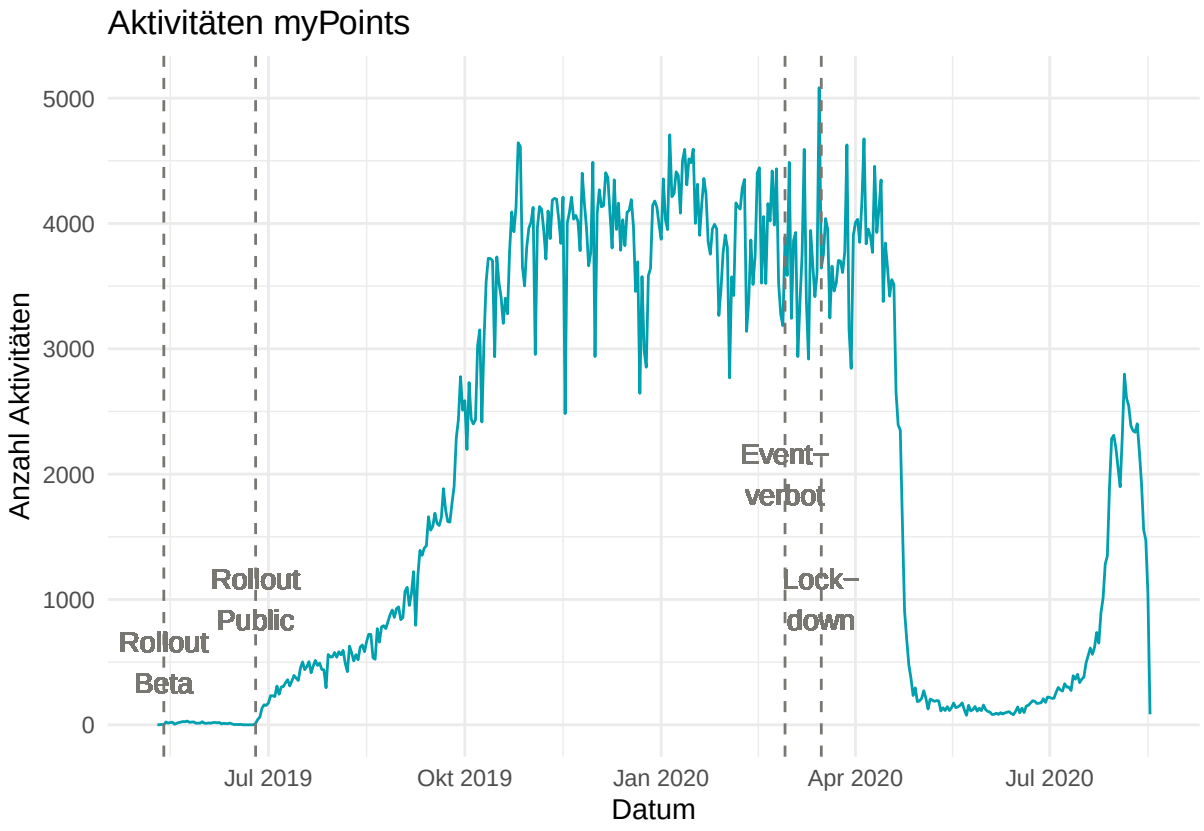
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

5.2.1 Aktivitäten nach Datum

Das erste Diagramm gibt die Aktivitäten nach Datum aus und ergänzt die wichtigsten Meilensteine, bzw. Events.

```
# Anzahl Aktivitäten über alles, inkl. evtl. relevanter Meilensteine
ggplot(daily_activities,
       aes(x = activity_date, y = count)) +
  geom_line(color = col_line_default) +
  geom_vline(xintercept = milestones.myPoints.beta_rollout,
             color = col_notes,
             linetype = "dashed") +
  geom_text(
    aes(x = milestones.myPoints.beta_rollout, y = 500),
    label = "Rollout\nBeta",
    color = col_notes
  ) +
  geom_vline(xintercept = milestones.myPoints.public_rollout,
             color = col_notes,
             linetype = "dashed") +
  geom_text(
    aes(x = milestones.myPoints.public_rollout, y = 1000),
    label = "Rollout\nPublic",
    color = col_notes
```

```
  ) +
  geom_vline(xintercept = milestones.covid19.major_events_forbidden,
             color = col_notes,
             linetype = "dashed") +
  geom_text(
    aes(x = milestones.covid19.major_events_forbidden, y = 2000),
    label = "Event-\nverbot",
    color = col_notes
  ) +
  geom_vline(xintercept = milestones.covid19.lockdown,
             color = col_notes,
             linetype = "dashed") +
  geom_text(aes(x = milestones.covid19.lockdown, y = 1000),
            label = "Lock-\ndown",
            color = col_notes) +
  scale_x_date() +
  theme_minimal() +
  labs(title = "Aktivitäten myPoints",
       x = "Datum",
       y = "Anzahl Aktivitäten")
```

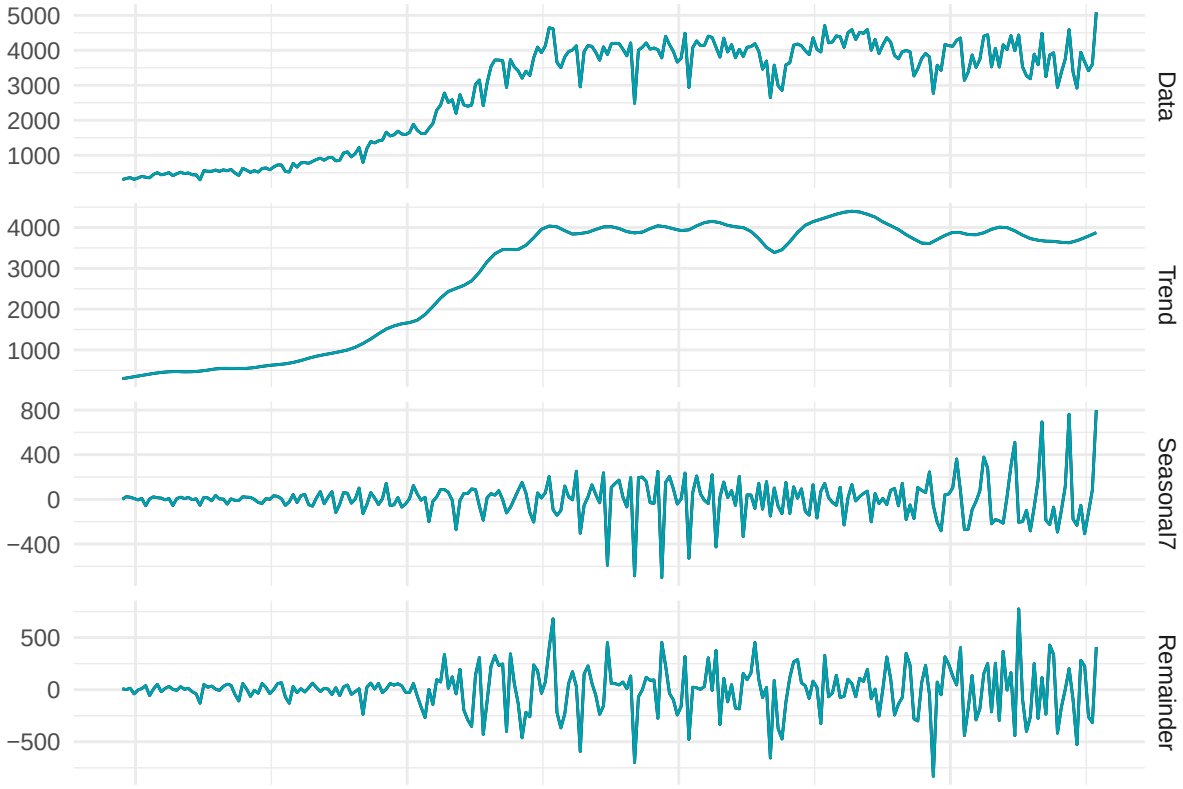



```
daily_activities.msts <- msts(
  daily_activities$count,
  seasonal.periods = c(7, 365.25),
  start = decimal_date(first_activity_begin_date)
)
```

Ein Einbruch der Aktivitäten nach dem Covid-19-Lockdown von Ende Mitte März ist in den Daten klar ersichtlich. Der grösste Einbruch erfolgt allerdings erst einige Tage nach diesem Lockdown und ist auf einen Fehler in der App zurück zu führen, aufgrund dessen sehr viele Aktivitäten unkategorisiert geliefert worden sind. Diese Daten wurden vorgängig aus den Daten gefiltert.

Für die Analyse der Wochendaten muss daher ein Zeitraum **vor** dem Lockdown verwendet werden. Die Periode zwischen dem Public Going-Live und dem Lockdwon wird somit für die weitere Analyse herangezogen:

```
daily_activities_extract <- daily_activities %>%
  filter (activity_date >= "2019-7-8") %>%
  filter(activity_date <= "2020-3-15")
daily_activities_extract.msts <-
  msts(
    daily_activities_extract$count,
    seasonal.periods = c(7),
    start = decimal_date(as.Date("2019-7-8"))
  )
autoplot(mstl(daily_activities_extract.msts, s.window = 7)) +
    geom_line(color = col_line_default) +
    theme_minimal() +
      theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank())
```

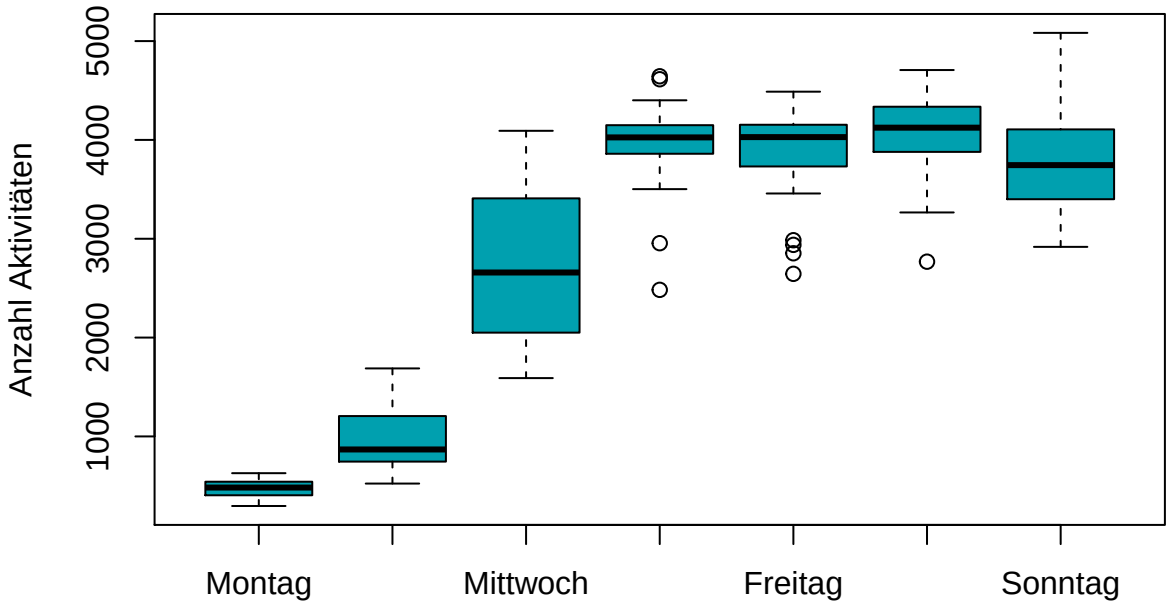


Ein wöchentlicher Trend (Seasonal7) scheint gegeben zu sein. Lässt sich dieser noch anders visualisieren, bzw. lässt sich herausfinden, wie die Verteilung unter den Wochentagen ist?

5.2.2 Gegencheck wöchentliche Tendenzen

Der folgende Boxplot der Aktivitäten / Wochentag zeigt die Tendenzen in Bezug auf die Wochentage auf:

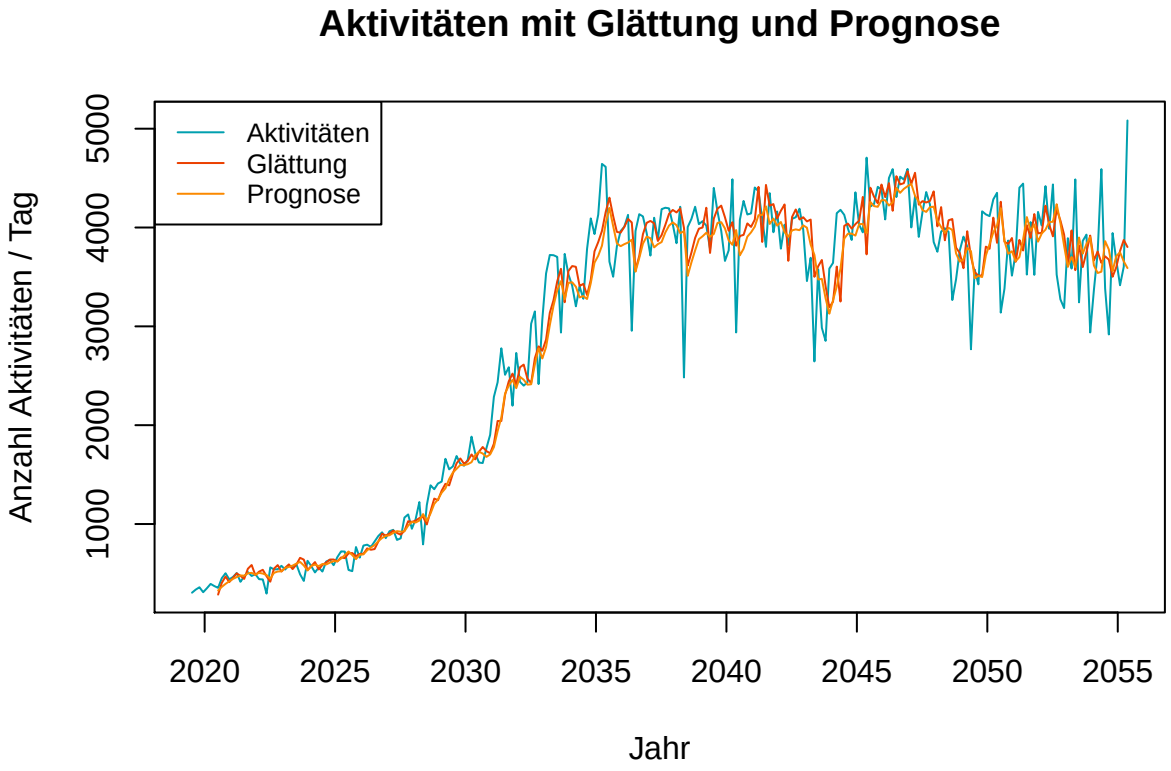
```
weekdays <- matrix(daily_activities_extract$count, ncol = 7) %>%
    as.data.frame() %>%
    rename(Montag = V1,
                  Dienstag = V2,
                  Mittwoch = V3,
                  Donnestag = V4,
                  Freitag = V5,
                  Samstag = V6,
                  Sonntag = V7)
boxplot(weekdays, col=col_line_default, ylab="Anzahl Aktivitäten")
```



5.2.3 Glättung und Prognose

Mit HoltWinters (inkl Beta (Trend) und Gamma (saisonales)) wird die Timeline geglättet und eine Kurzfrist-Prognose ausgegeben:

```
daily_activities_extract.hw = HoltWinters(daily_activities_extract.msts)
plot(daily_activities_extract.msts,
     col=col_line_default,
         main="Aktivitäten mit Glättung und Prognose",
         xlab="Jahr",
         ylab="Anzahl Aktivitäten / Tag")
lines(daily_activities_extract.hw$fitted[,1], col=col_line_alt2)
lines(daily_activities_extract.hw$fitted[,2], col=col_line_alt)
legend("topleft", legend=c("Aktivitäten", "Glättung", "Prognose"),
       col=c(col_line_default, col_line_alt2, col_line_alt), lty=1:1,
            cex=0.8)
```


Aktivitäten mit Glättung und Prognose
Aktivitäten
Glättung
Prognose
Anzahl Aktivitäten / Tag
1000
2000
3000
4000
5000
2020
2025
2030
2035
2040
2045
2050
2055
Jahr