local bakery data analysis

Yufenyuy

2025-07-20

Hier werden R-Packete für die Daten Manipulation bzw. Auswertung beladen

```
library(conflicted)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
```

```
library(DBI)
```

```
## Warning: Paket 'DBI' wurde unter R Version 4.4.3 erstellt
```

```
library(RPostgres)
```

```
## Warning: Paket 'RPostgres' wurde unter R Version 4.4.3 erstellt
```

```
library(ggplot2)
library(lubridate)
library(dplyr)
library(tidyr)
library(stringr)
library(fpp3)
```

```
## Warning: Paket 'fpp3' wurde unter R Version 4.4.3 erstellt
```

```
## Registered S3 method overwritten by 'tsibble':
##   method               from
##   as_tibble.grouped_df dplyr
## -- Attaching packages --------------------------------------------- fpp3 1.0.1 --
## v tsibble     1.1.5     v feasts      0.4.1
## v tsibbledata 0.4.1     v fable       0.4.1
```

```
## Warning: Paket 'tsibble' wurde unter R Version 4.4.2 erstellt
```

```
## Warning: Paket 'tsibbledata' wurde unter R Version 4.4.3 erstellt
```

```
## Warning: Paket 'feasts' wurde unter R Version 4.4.3 erstellt
```

```
## Warning: Paket 'fabletools' wurde unter R Version 4.4.3 erstellt
```

```
## Warning: Paket 'fable' wurde unter R Version 4.4.3 erstellt
```

Erstellte Umgebung Variablen werden für die Datenbank Verbindung eingelesen

```
con <- dbConnect(
  Postgres(),
  dbname = Sys.getenv("DBNAME"),
  host = Sys.getenv("HOST"),
  user = Sys.getenv("USER"),
  password = Sys.getenv("PASSWORD"),
  port = Sys.getenv("PORT")
)
```

Die gezielte Daten Tabelle liegt in einer spezifisches Schema in PostgreSQL. Die Tabellen enhalten in diesem Schema werden aufgelisted und geprüft, ob die gewünschte Tabelle drin liegt

```
schema_name <- "baker_yelp_dbt_prod"

# SQL query to list tables in the given schema
tables <- dbGetQuery(con, paste0("
  SELECT table_name
  FROM information_schema.tables
  WHERE table_schema = '", schema_name, "'
    AND table_type = 'BASE TABLE';
"))

print(tables)
```

```
##                         table_name
## 1                        pdtn_fuel
## 2                monthly_ts_issales
## 3           weekly_product_pdtn_ts
## 4                          sales_t1
## 5           monthly_ts_should_sales
## 6                products_weekly_ts
## 7                           pdtn_t1
## 8                           feeding
## 9                          sales_t2
## 10              business_reviews_t1
## 11       weekly_timeseries_issales
## 12                    pdtn_oldstuck
## 13              business_reviews_t2
## 14       weekly_ts_is_should_sales
## 15                    businesses_t1
## 16          weekly_ts_should_sales
## 17                    businesses_t2
## 18 daily_expected_production_dates
## 19          daily_product_pdtn_amt
## 20                    businesses_t3
## 21      monthly_ts_is_should_sales
## 22                           items
## 23                       datetable
```

Hier werden die Daten mittels SQL selektiert

```
product_ts <- dbGetQuery(con, "SELECT * FROM baker_yelp_dbt_prod.products_weekly_ts")
```

Datentypen und Werte einsehen

```
str(product_ts)
```

```
## 'data.frame':    425 obs. of  15 variables:
##  $ endofweek     : Date, format: "2023-07-09" "2023-07-02" ...
##  $ banana50_amt  : num  9 3 0 0 0 0 0 0 0 0 ...
##  $ square50_amt  : num  148 142 124 136 148 ...
##  $ local50_amt   : num  7.5 12.5 17.5 17.5 22.5 22.5 25 15 30 30 ...
##  $ banana100_amt : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ local100_amt  : num  190 185 177 201 190 ...
##  $ special100_amt: num  675 1732 1674 1752 1657 ...
##  $ special150_amt: num  1319 128 149 158 163 ...
##  $ local200_amt  : num  253 244 250 234 269 ...
##  $ special200_amt: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ local250_amt  : num  50.4 50.4 42 50.4 49 42 40.6 33.6 42 50.4 ...
##  $ local300_amt  : num  569 519 524 580 543 ...
##  $ special400_amt: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ special500_amt: num  706 624 623 697 630 ...
##  $ special800_amt: num  60.5 21.6 32.7 68.7 44.1 ...
```

Statistik der Daten einsehen, um die Zentrale Tendenz und die Streuung der Daten zu verstehen.

```
summary(product_ts)
```

```
##    endofweek           banana50_amt    square50_amt     local50_amt
##  Min.   :2015-05-24   Min.   :  0.0   Min.   :  0.00   Min.   : 0.000
##  1st Qu.:2017-06-04   1st Qu.:  0.0   1st Qu.:  0.00   1st Qu.: 0.000
##  Median :2019-06-16   Median : 48.0   Median :  0.00   Median : 0.000
##  Mean   :2019-06-16   Mean   :135.8   Mean   : 66.88   Mean   : 1.688
##  3rd Qu.:2021-06-27   3rd Qu.:276.0   3rd Qu.:135.00   3rd Qu.: 0.000
##  Max.   :2023-07-09   Max.   :542.4   Max.   :450.00   Max.   :40.000
##  banana100_amt      local100_amt   special100_amt   special150_amt
##  Min.   :  0.000   Min.   :  0.0   Min.   :   0.0   Min.   :   0.0
##  1st Qu.:  0.000   1st Qu.: 66.3   1st Qu.: 196.3   1st Qu.:   0.0
##  Median :  0.000   Median :170.0   Median : 973.5   Median :   0.0
##  Mean   :  7.916   Mean   :161.9   Mean   : 889.5   Mean   : 164.8
##  3rd Qu.:  0.000   3rd Qu.:238.0   3rd Qu.:1440.5   3rd Qu.: 301.0
##  Max.   :107.500   Max.   :528.7   Max.   :2352.9   Max.   :1318.6
##   local200_amt   special200_amt     local250_amt    local300_amt
##  Min.   :  0.0   Min.   :  0.000   Min.   :  0.0   Min.   :  0.0
##  1st Qu.:  0.0   1st Qu.:  0.000   1st Qu.: 15.4   1st Qu.: 72.1
##  Median : 93.2   Median :  0.000   Median : 56.0   Median :349.3
##  Mean   :113.5   Mean   :  8.179   Mean   :143.2   Mean   :319.5
##  3rd Qu.:231.6   3rd Qu.:  0.000   3rd Qu.:237.3   3rd Qu.:518.7
##  Max.   :351.6   Max.   :134.940   Max.   :607.6   Max.   :777.7
##  special400_amt   special500_amt   special800_amt
##  Min.   :  0.00   Min.   :   0.0   Min.   :  0.00
##  1st Qu.:  0.00   1st Qu.: 100.4   1st Qu.:  2.40
##  Median :  0.00   Median : 435.2   Median : 19.20
##  Mean   : 13.17   Mean   : 384.1   Mean   : 25.08
##  3rd Qu.:  0.00   3rd Qu.: 581.0   3rd Qu.: 38.40
##  Max.   :534.48   Max.   :1004.7   Max.   :110.40
```

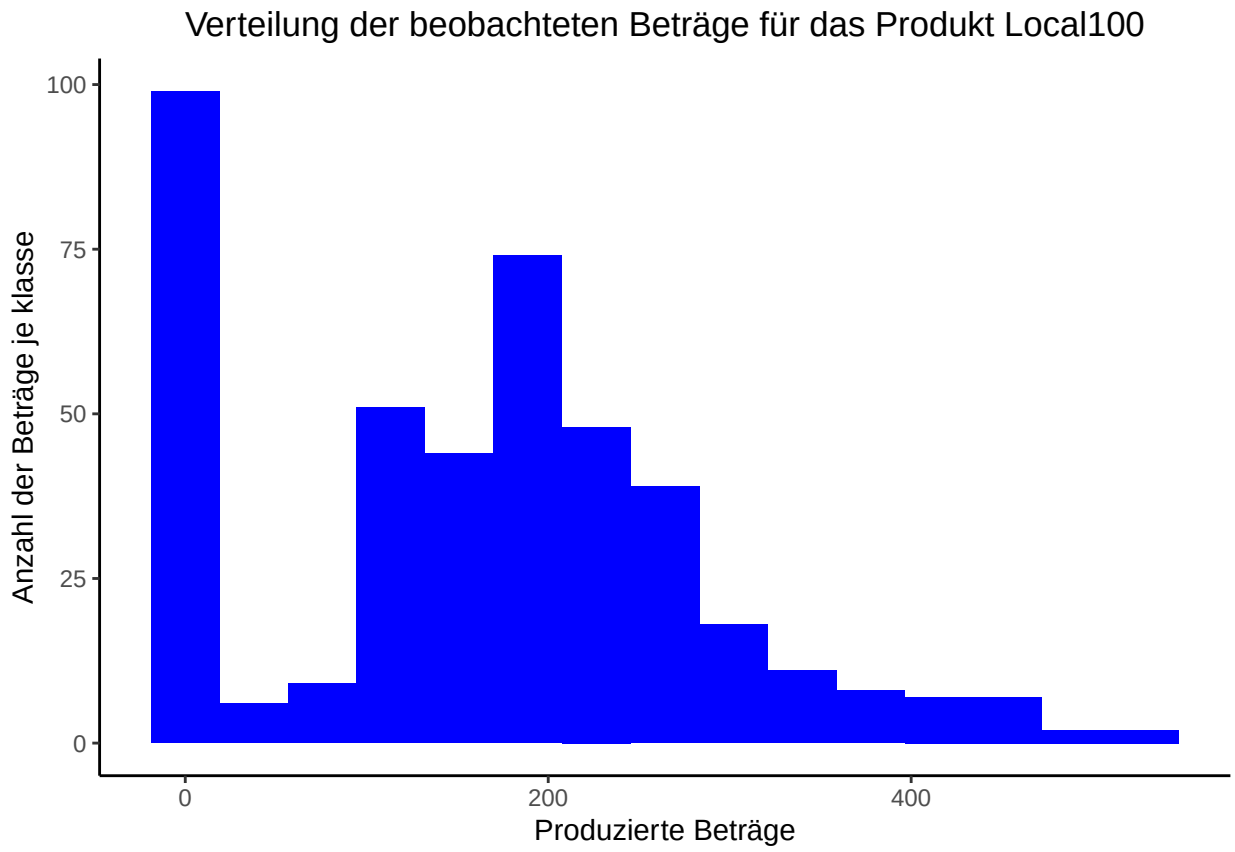
Beobachtungen:

- Die Daten wurden von mitte 2015 bis mitte 2023 gesammelt.

- Bei viele Produkte(spezial150_amt) liegt der mittleren Wert bei 0. D.H Bis zu 50% ihre Beträge liegen bei 0 Geld Einheiten(GE).
- Der minimale produzierte Betrag bei alle Produkte liegt bei 0 GE.
- Der höchste durchschnittliche produzierte Betrag über den ganzen Zeitraum ist bei dem Produkt special200_amt beobachtet und dasselbe Produkt hat der höchste produzierte Betrag unter alle Produkten.Allerding liegt sein Mittelwert bei 0 GE.
- Bei der Produkte local100_amt, local300_amt und special100_amt liegen der Mittelwert und der Durchschnitt nah beieinander.Diese weißt auf einer quasi Normal Verteilung hin spricht die Daten sind quasi gleichverteilt.

Ein Histogram für das Produkt Local100_amt

```
g1 <- product_ts %>%
  ggplot(aes(x = local100_amt)) +
  geom_histogram(fill = "blue", bins = 15) +
  xlab("Produzierte Beträge") +
  ylab("Anzahl der Beträge je klasse") +
  theme_classic() +
  ggtitle("Verteilung der beobachteten Beträge für das Produkt Local100") +
  theme(
    plot.title = element_text(hjust = 0.5)
  )
g1
```



Ein neue Datensatz wird hier erzeugt

```
product_ts_long <- product_ts %>% pivot_longer(cols = ends_with("amt"), names_to = "products", values_t
```

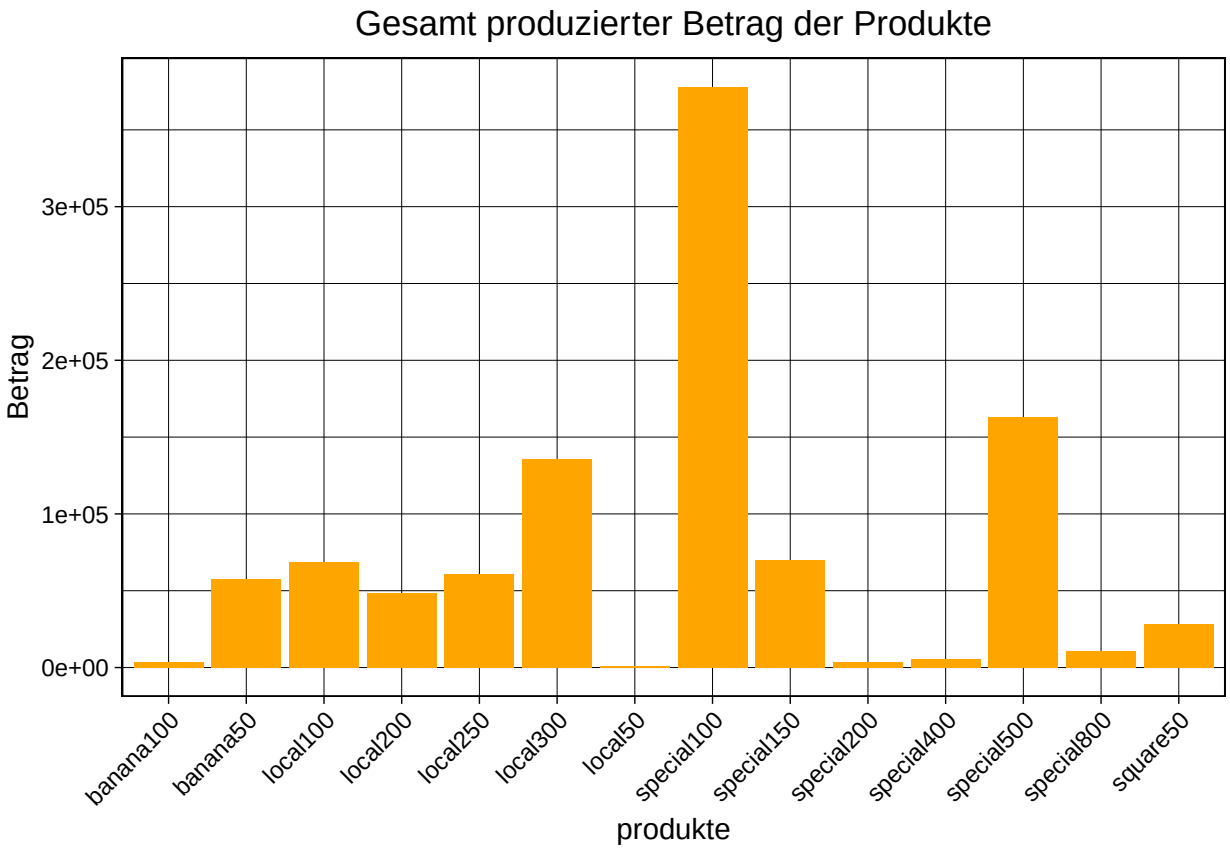
Statistik der Daten einsehen.

```
summary(product_ts_long)
```

```
##    endofweek            products         weekly_amount
##  Min.   :2015-05-24   Length:5950        Min.   :   0.0
##  1st Qu.:2017-06-04   Class :character   1st Qu.:   0.0
##  Median :2019-06-16   Mode  :character   Median :  13.8
##  Mean   :2019-06-16                      Mean   : 174.0
##  3rd Qu.:2021-06-27                      3rd Qu.: 229.9
##  Max.   :2023-07-09                      Max.   :2352.9
```

Die produzierte Menge der Produkte werden hier verglichen

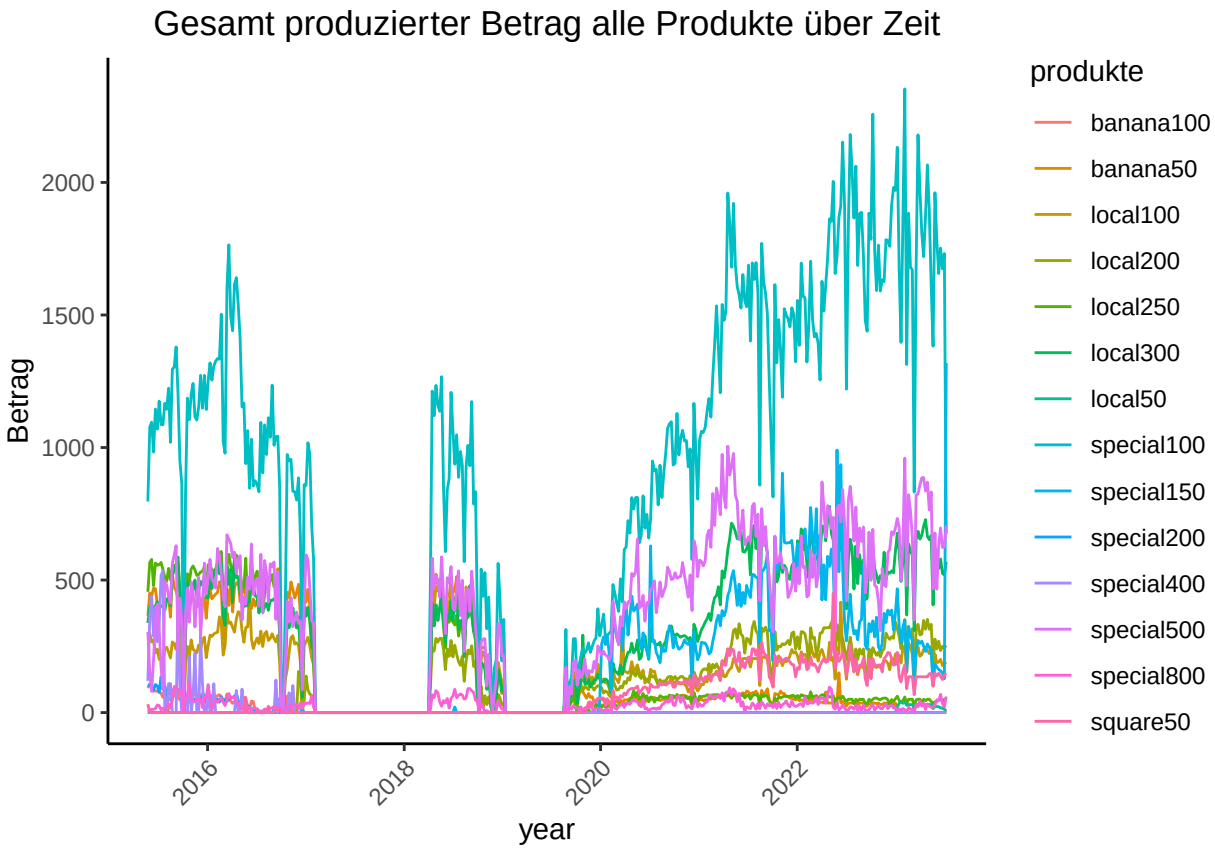
```
gplot_2 <- product_ts_long %>%
  mutate(
    produkte = str_sub(products, 1, str_length(products) - 4)
    ) %>%
  group_by(produkte) %>%
  summarise(total_amount = sum(weekly_amount)) %>%
  ggplot(aes(x = produkte, y = total_amount)) +
  geom_col(fill = "orange") +
  ylab("Betrag") +
  ggtitle("Gesamt produzierter Betrag der Produkte") +
  theme_linedraw() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
gplot_2
```



+Aus der obige Graphic stellt sich fest, dass Special100, Special500 und local300 die höchste produzierte Beträge erzielt hatten.

Zeitlich produzierte Menge für jedes Produkt

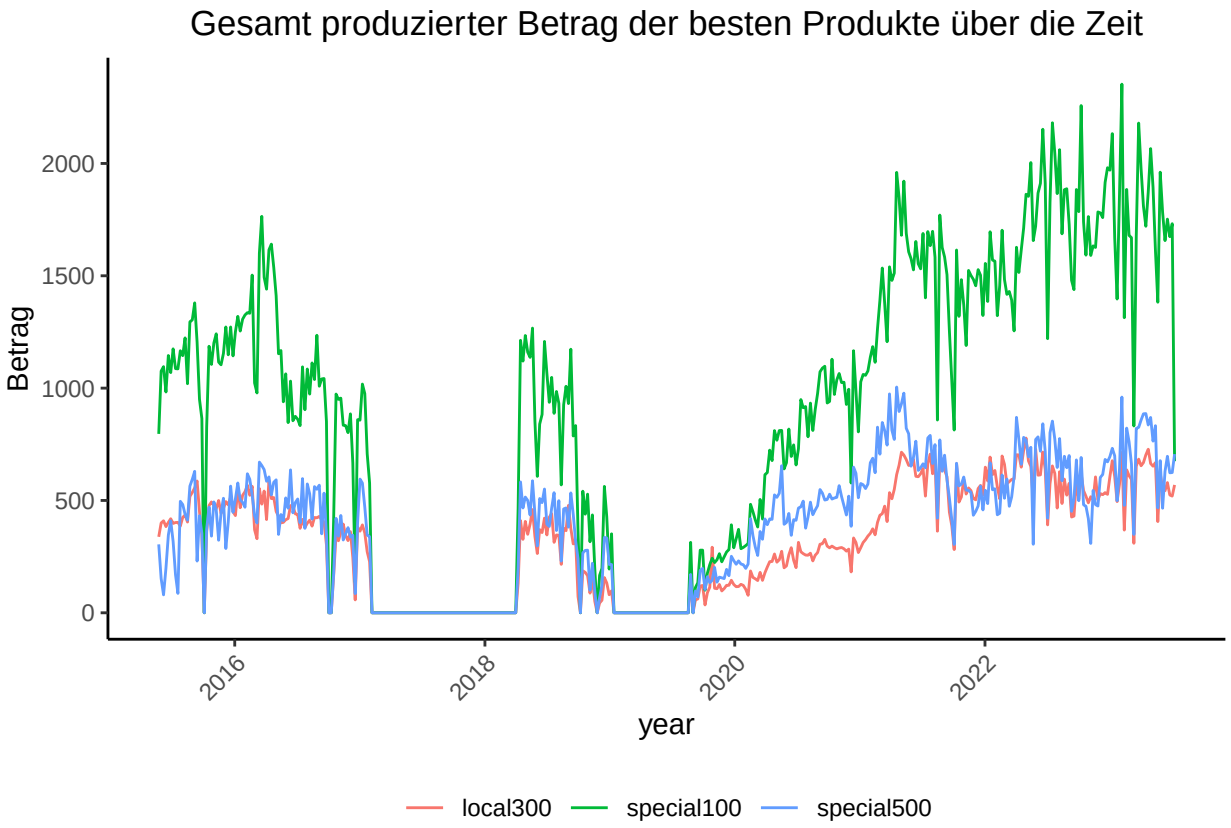
```
gplot_3 <- product_ts_long %>%
  mutate(
    produkte = str_sub(products, 1, str_length(products) - 4)
    ) %>%
  ggplot(aes(x = endofweek, y = weekly_amount, col = produkte )) +
  geom_line() +
  xlab("year") +
  ylab("Betrag") +
  ggtitle("Gesamt produzierter Betrag alle Produkte über Zeit") +
  theme_classic() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
gplot_3
```



Aus der obigen Graphik kann man sehen, dass produktionsdaten für alle Produkte zu derselben Zeiten gesammelt wurden. Es gibt Produkte, die nach 2020 in der Produktion gewachsen sind. Die steigende Produktionsmengen könnte von den Nachfrage am Markt oder die Qualität der Produkte bzw. wachsenden Kundenbasis gewesen sein. Bei viele anderen Produkte ist die Produktion nach wie vor 2020 niedrig geblieben sind.Diese könnte man als eine Niche betrachten.

Zeitlich produzierte Menge für die drei meist produzierte Produkte

```
gplot_4 <- product_ts_long %>%
  dplyr::filter(products %in% c("special100_amt","special500_amt", "local300_amt")) %>%
  mutate(
    produkte = str_sub(products, 1, str_length(products) - 4)
    ) %>%
  ggplot(aes(x = endofweek, y = weekly_amount, col = produkte )) +
  geom_line() +
  xlab("year") +
  ylab("Betrag") +
  ggtitle("Gesamt produzierter Betrag der besten Produkte über die Zeit") +
  theme_classic() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1),
    legend.position = "bottom",
    legend.title = element_blank()
  )
gplot_4
```



+Vor 2020 wurde bei diesen Produkten in verschiedene Zeitabständen beobachtet.

+Ab 2020 ist bei diesen Produkten eine steigende Trend zu sehen.

+Special100 ist mit Abstand das meist produzierte Produkt, während local300 and special500 nach 2023 gegen einander konkuriet haben.

Zeit Rheihe Analyse folgt.

Ziel der Zeitrheihe ist eine mindestens 6 monatige Forecast für das meist produzierte Produkt zu entwickeln.

Ein geeignetes Objekt für Zeitrheihe Analyse wird hier erzeugt. Diese ist für die meist produzierten Produkte

```
products <- product_ts_long %>%
  dplyr::filter(year(endofweek) %in% c(2020, 2021, 2022, 2023) & products %in% c("special100_amt","speci
  arrange(desc(endofweek)) %>%
  mutate(weekly = yearweek(endofweek))
products
```

```
## # A tibble: 552 x 4
##    endofweek  products       weekly_amount   weekly
##    <date>     <chr>                  <dbl>   <week>
##  1 2023-07-09 special100_amt          675. 2023 W27
##  2 2023-07-09 local300_amt            569. 2023 W27
##  3 2023-07-09 special500_amt          706. 2023 W27
##  4 2023-07-02 special100_amt         1732. 2023 W26
##  5 2023-07-02 local300_amt            519. 2023 W26
##  6 2023-07-02 special500_amt          624. 2023 W26
```

```
##  7 2023-06-25 special100_amt         1674. 2023 W25
##  8 2023-06-25 local300_amt            524. 2023 W25
##  9 2023-06-25 special500_amt          623. 2023 W25
## 10 2023-06-18 special100_amt         1752. 2023 W24
## # i 542 more rows
```

Die schlüssel sowie das Index der Zeitrheihen werden hier definiert. Der Schlüssel bestimmt eine Zeile eindeutigt.

```
local_bakery_production_ts <- products |> as_tsibble(index = weekly, key = c(endofweek, products))
```

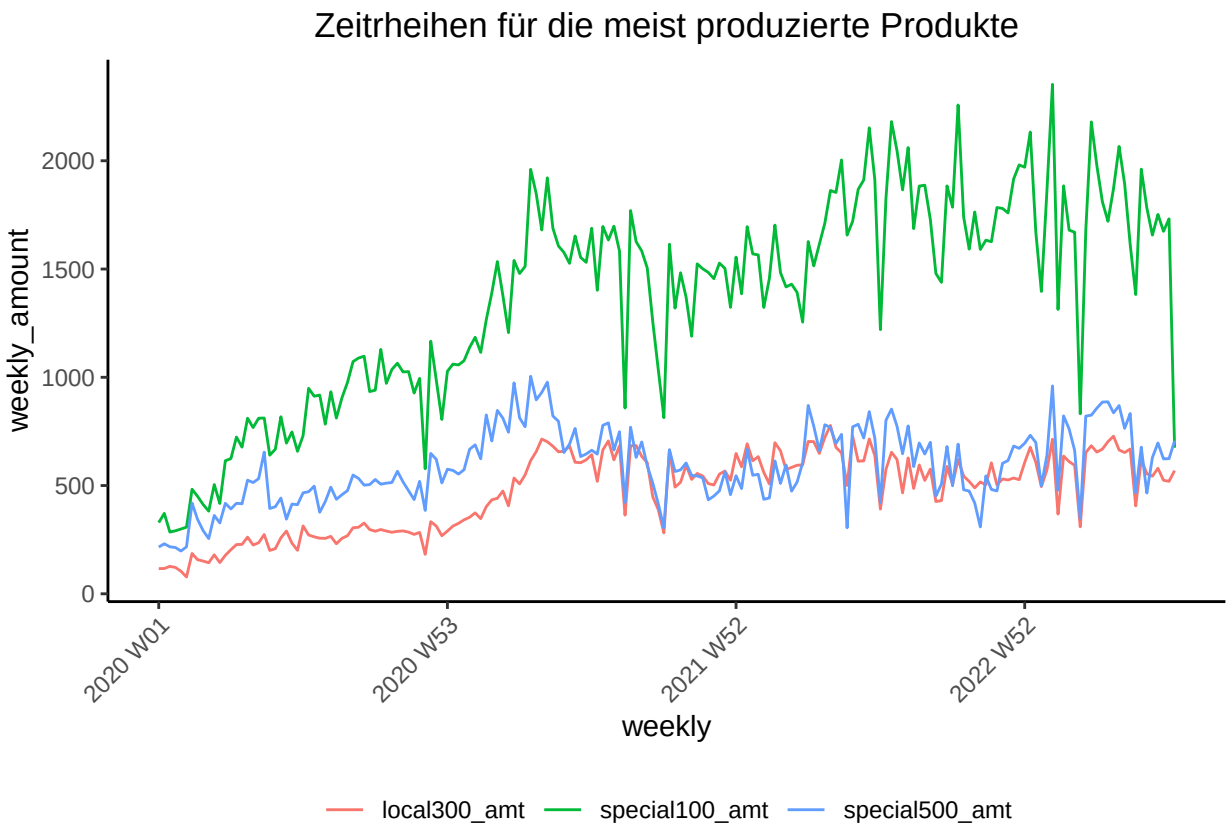
Der Kopt der Rheihen anschauen. Da sieht man, dass es sich um wöchenliche Zeitrheihen handelt.

```
head(local_bakery_production_ts,6)
```

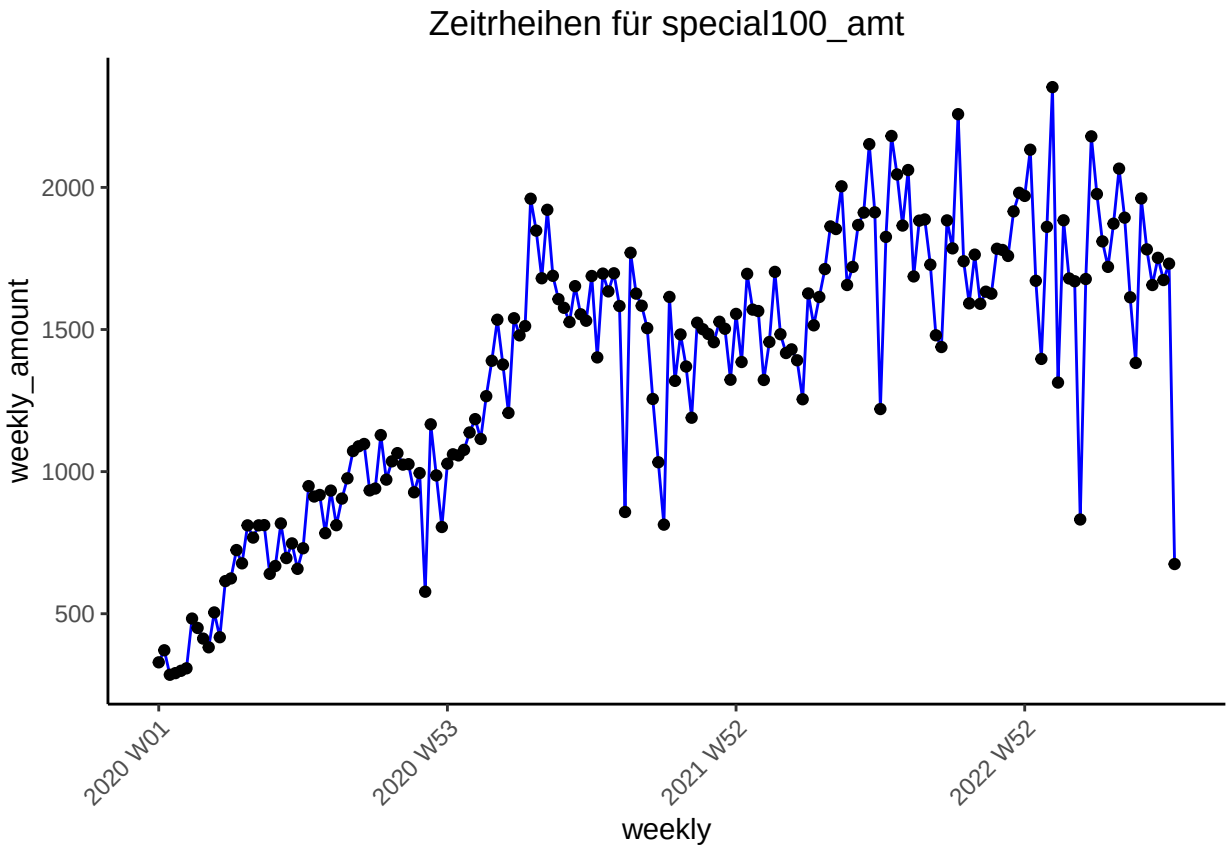
```
## # A tsibble: 6 x 4 [1W]
## # Key:       endofweek, products [6]
##   endofweek  products       weekly_amount   weekly
##   <date>     <chr>                  <dbl>   <week>
## 1 2020-01-05 local300_amt            116. 2020 W01
## 2 2020-01-05 special100_amt          329. 2020 W01
## 3 2020-01-05 special500_amt          216  2020 W01
## 4 2020-01-12 local300_amt            117. 2020 W02
## 5 2020-01-12 special100_amt          371. 2020 W02
## 6 2020-01-12 special500_amt          231. 2020 W02
```

Zeitrheihen für die meist produzierte Produkte graphisch darstellen.

```
local_bakery_production_ts %>%
  ggplot(aes(x = weekly)) +
  geom_line(aes(y = weekly_amount, color = products)) +
  labs(title = "Zeitrheihen für die meist produzierte Produkte") +
  theme_classic() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1),
    legend.position = "bottom",
    legend.title = element_blank()
  )
```




```
local_bakery_production_ts[local_bakery_production_ts$products == "special100_amt",] %>%
  ggplot(aes(x = weekly)) +
  geom_line(aes(y = weekly_amount), color = "blue") +
  geom_point(aes(y = weekly_amount), color = "black") +
  labs(title = "Zeitrheihen für special100_amt") +
  theme_classic() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```



Betrachte man die Graphik für das meist produzierte Produkt sprich special100_amt, dann kann man folgendes beobachten

- Die Zeitrheihe zeigt allgemein einen steigenden Trend von 2020 bis Mitte 2023, obwohl der Trend ab Ende 2022 gesunken ist.
- Die Variation ist über die ganze Zeit ziemlich konstant bzw. gleich.
- Es gibt keine Woche ohne Beobachtung.

Aus der Beobachtung kann man davon ausgehen, dass die Zeitrheihe **nicht Stationär** ist. D.h die durchschnittliche produzierte Menge dieses Produkts hatte sich von eine Woche auf die Andern geändert. Die Stationarität ist ein wichtiges Konzept in statistische Zeitrheihe Analyse.

—-Die Analyse ist noch nicht abgeschlossen—-