Solar Energy Forecast

Jakub Tłuczek

4 02 2022

Abstract

Following is the forecast of the total energy produced by a solar panel installation in eastern Poland. Data is collected monthly, with several exceptions whenever an inverter got broken and it wasn't possible to retrieve energy produced last month. Additionaly, data about amount of energy bought and sold each month is also included. Aim of this task is to predict the production of energy in the following year. This task has been done in order to test newly acquired knowledge in a real life use case.

Setup

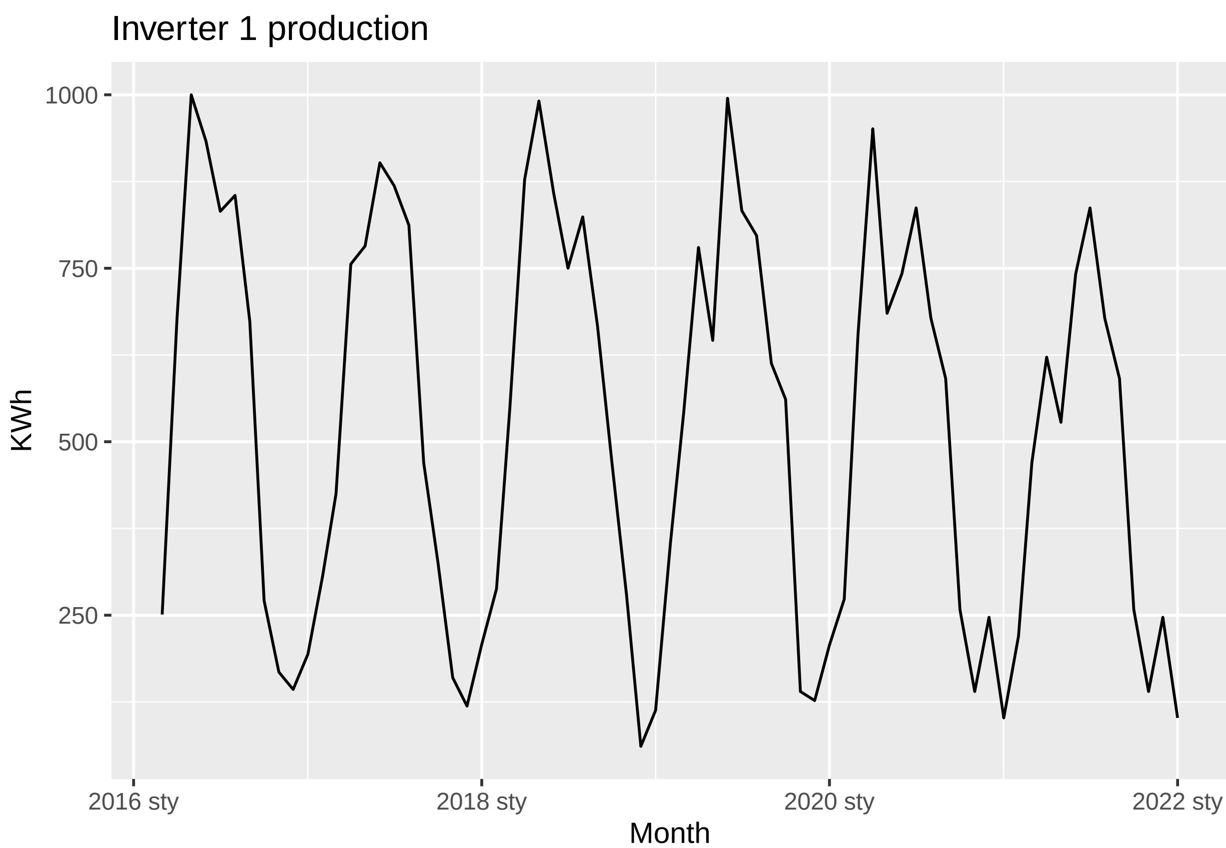
Firstly, a dataset has to be loaded into environment. As raw data is being put in on a monthly basis in simple csv file, it is read in a following way:

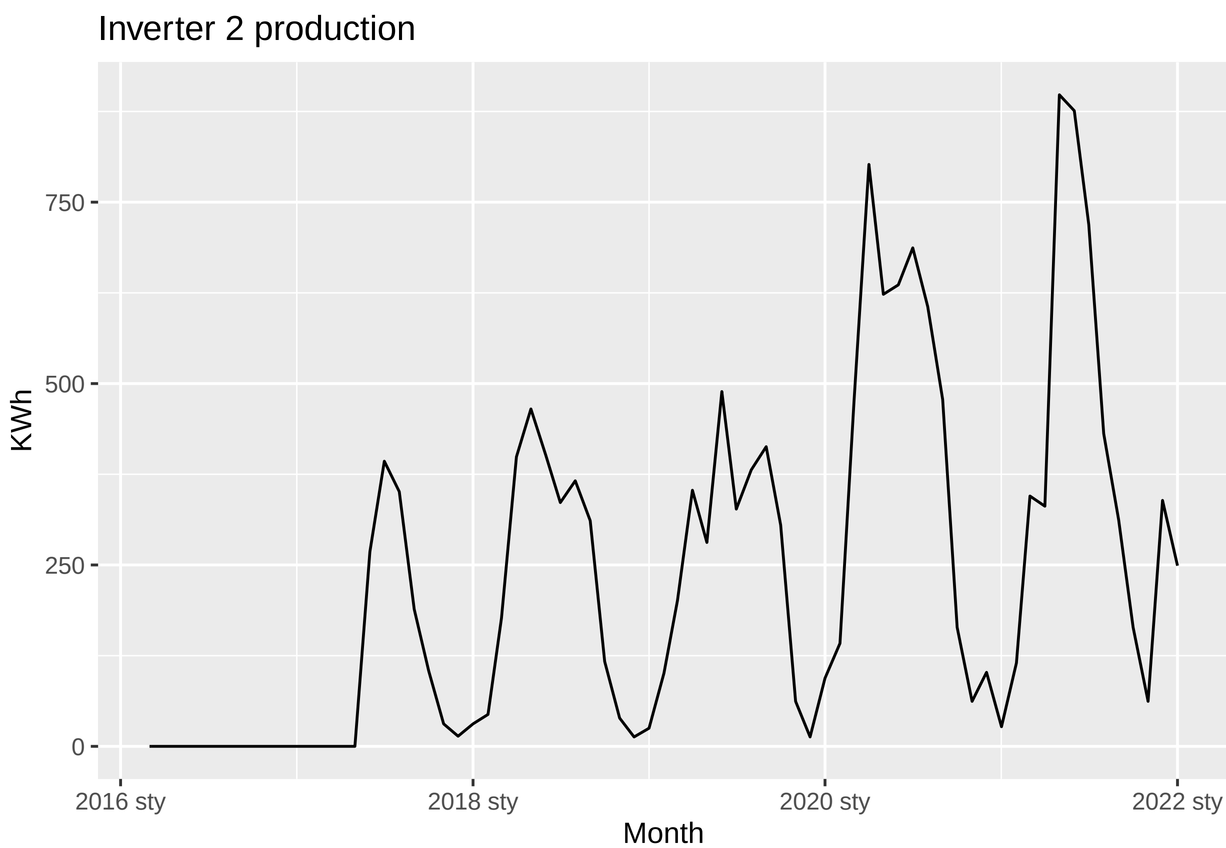
```
energia <- read.table("energia.csv", sep=";", header=T)
energia <- energia %>%
  mutate(month = yearmonth(Data)) %>%
  select(-Data)%>%
  as_tsibble(index = month)
```

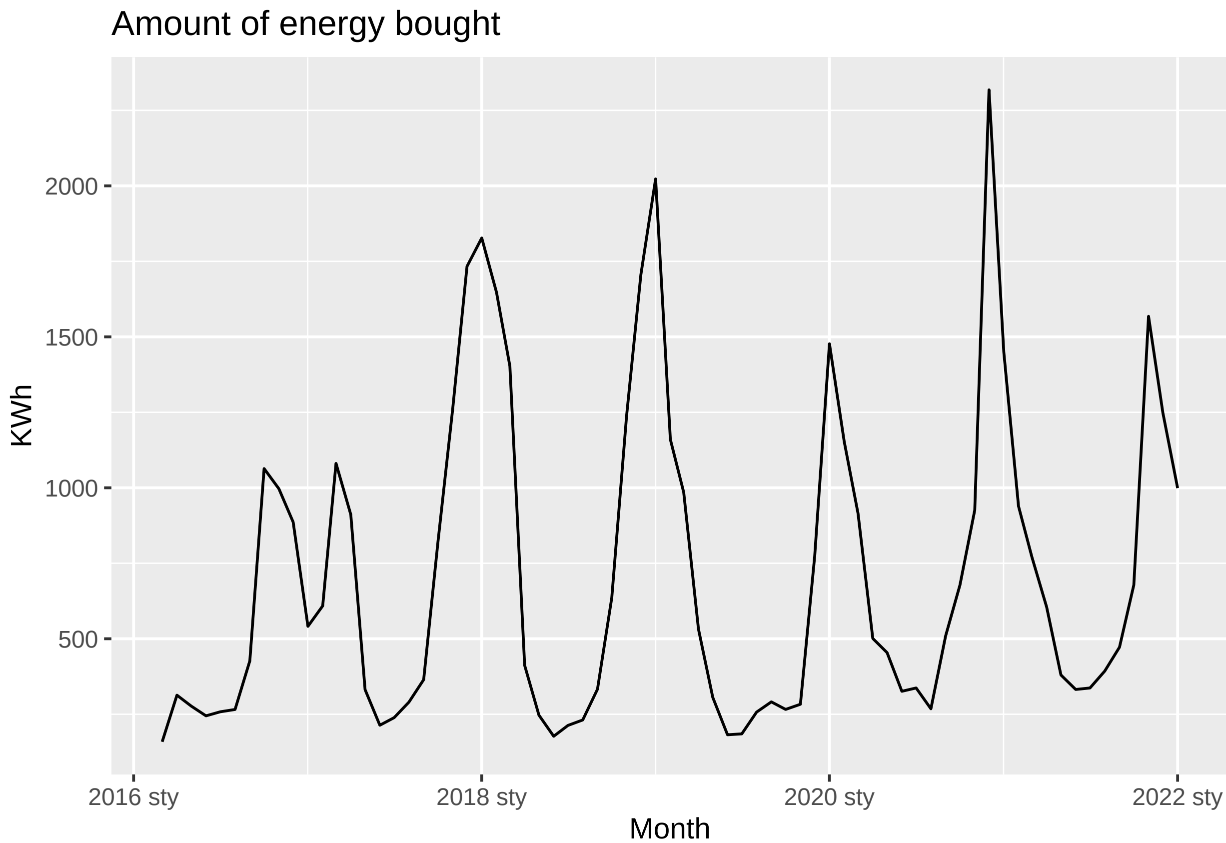
Aside from loading the table, `Data` which is in a `%y-%m-%d` format is being replaced by `month` variable, which is then set to an index of a resulting `tsibble`.

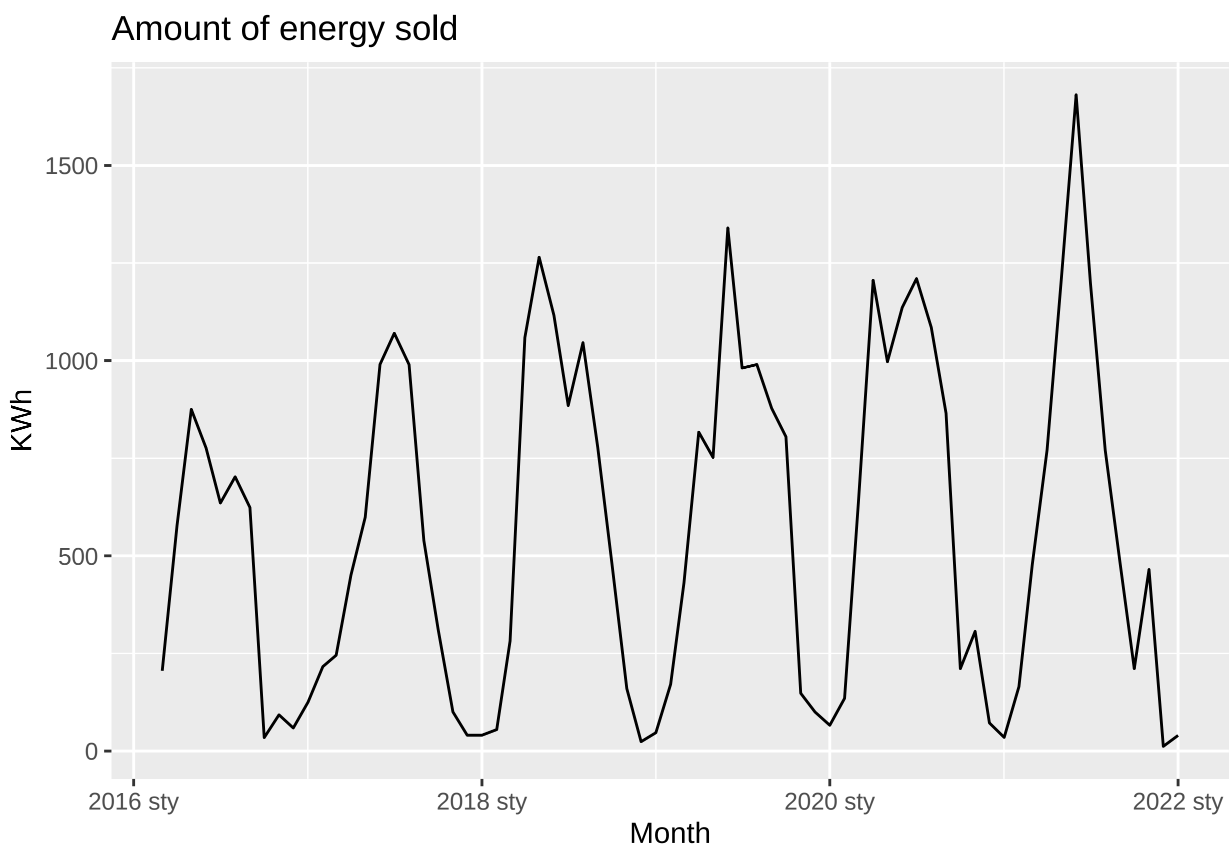
Preprocessing

Dataset includes several missing values, which have to be imputed in order to proceed. Since there has been no significant weather anomalies regarding suntime in eastern Poland, a seasonal split has been used:


Inverter 1 production
1000
750
500
250
KWh
2016 sty
2018 sty
2020 sty
2022 sty
Month


Inverter 2 production
KWh
750
500
250
0
2016 sty
2018 sty
2020 sty
2022 sty
Month

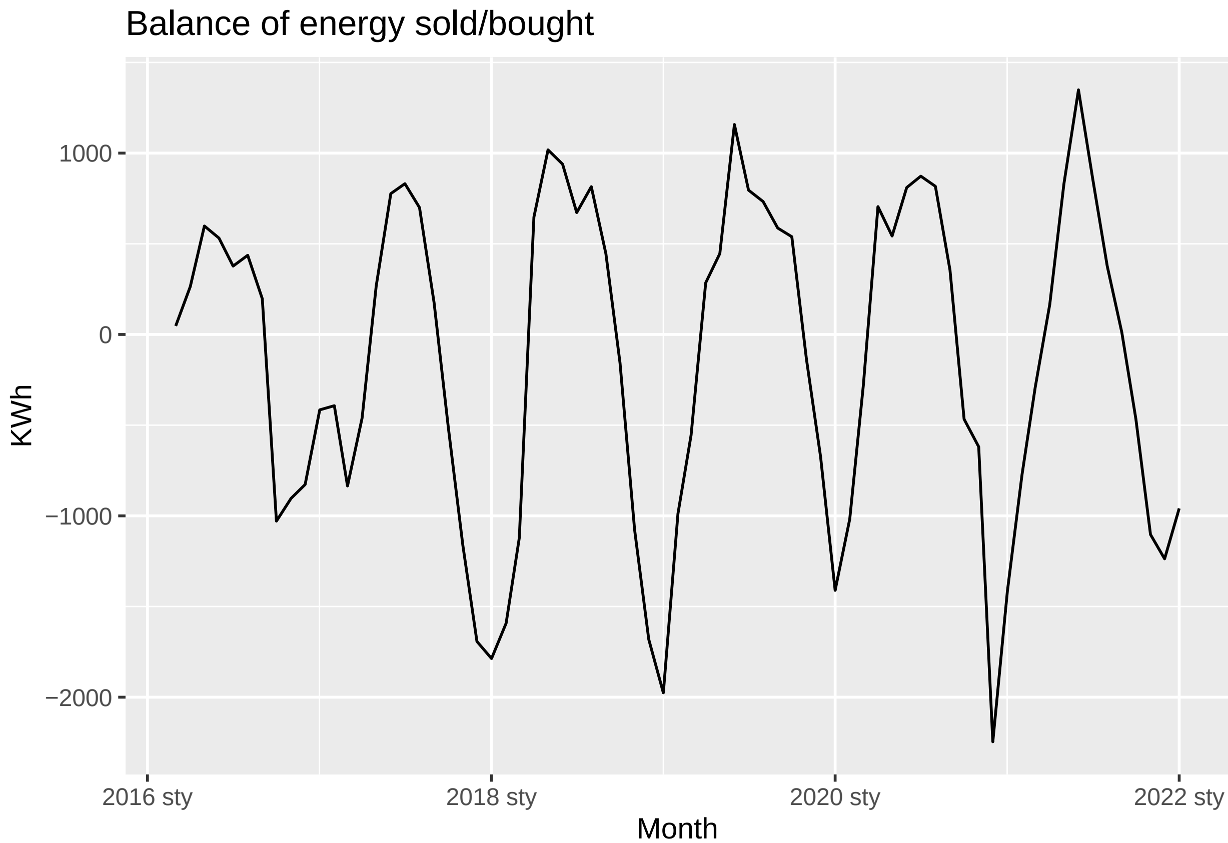

Amount of energy bought
KWh
2000
1500
1000
500
2016 sty
2018 sty
2020 sty
2022 sty
Month



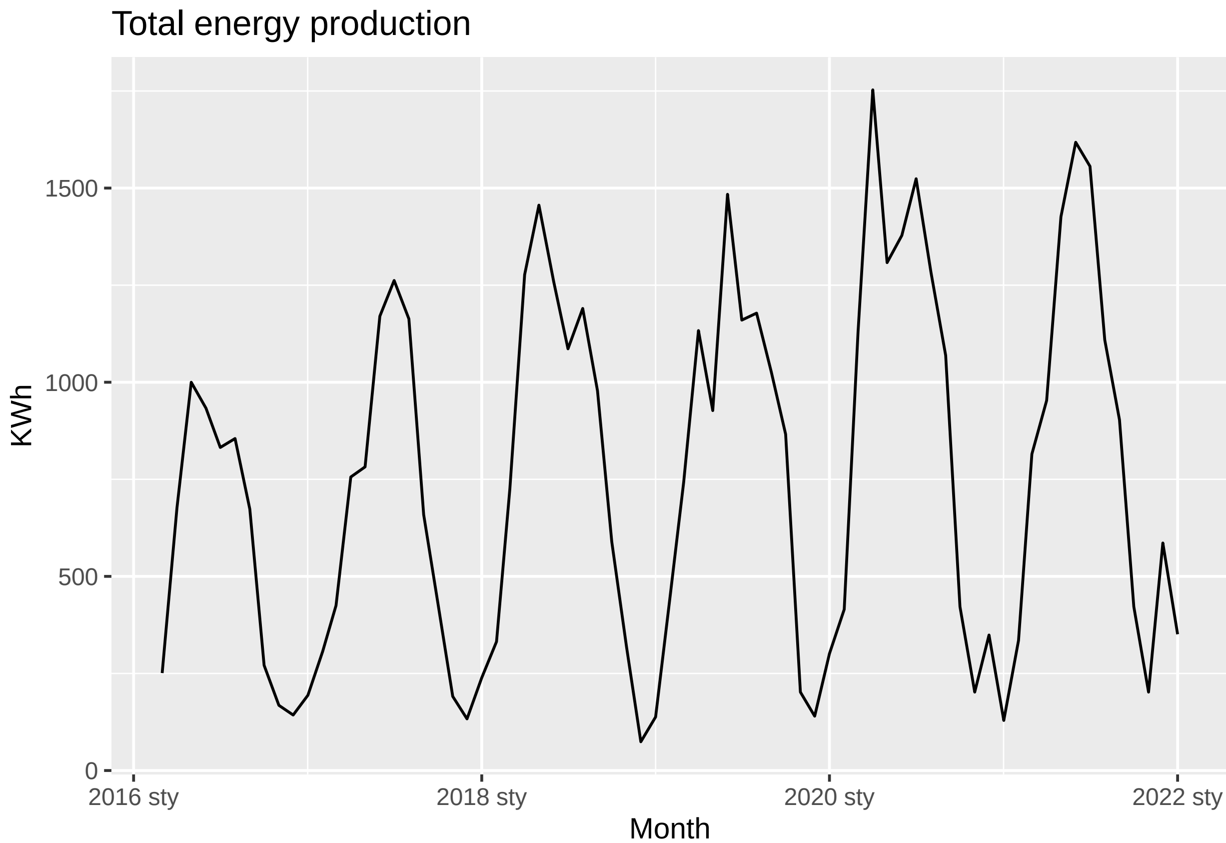
In all variables a strong seasonal trend can be noticed, where production and energy sold peaks during summer, while the amount of energy bought is noticeably higher during winter.

In the next step two calculated variables were added, namely balance of energy sold/bought, and monthly sum of energy produced.

```
energia %>%
  mutate(ProdukcjaRazem = Produkcja1 + Produkcja2,
         Bilans = Sprzedaz - Zakup) -> energia
```

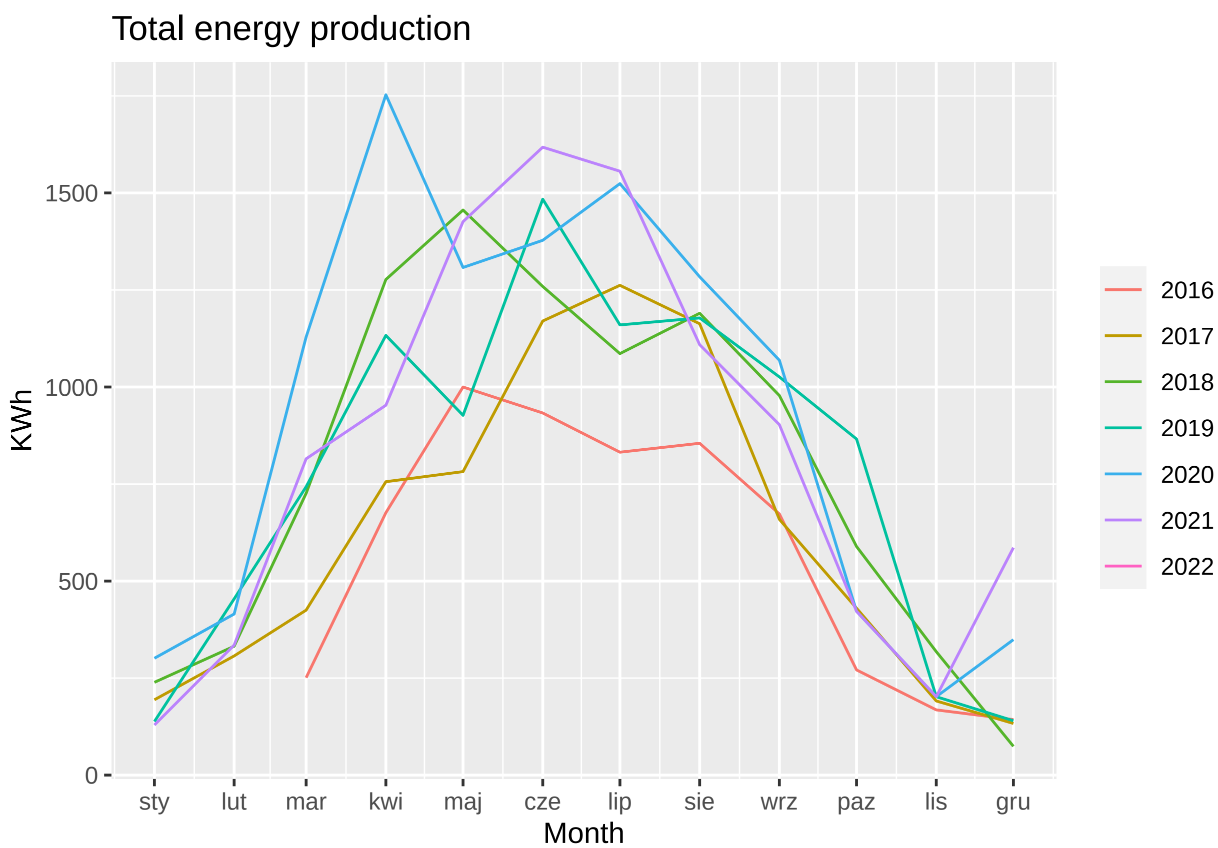


Balance of energy sold/bought
1000
0
−1000
−2000
KWh
2016 sty
2018 sty
2020 sty
2022 sty
Month

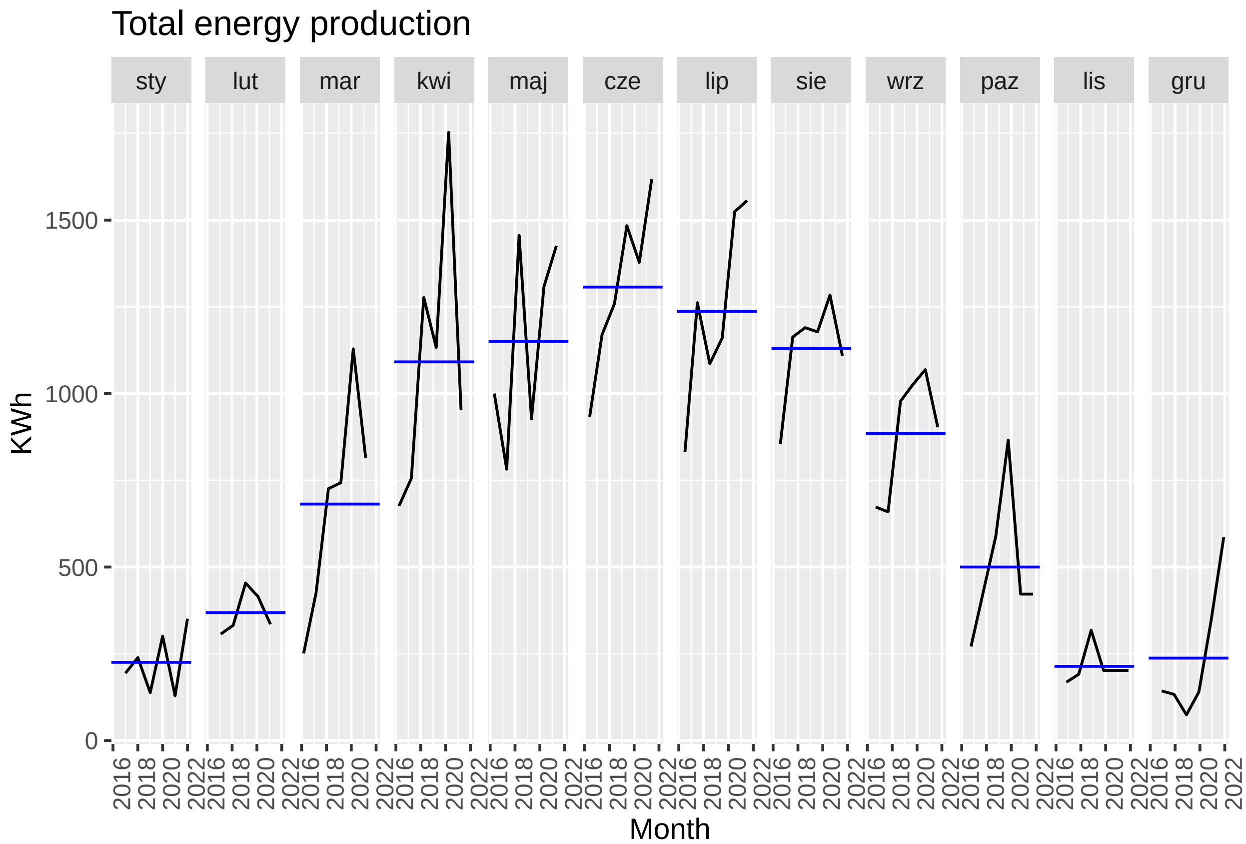


Exploration

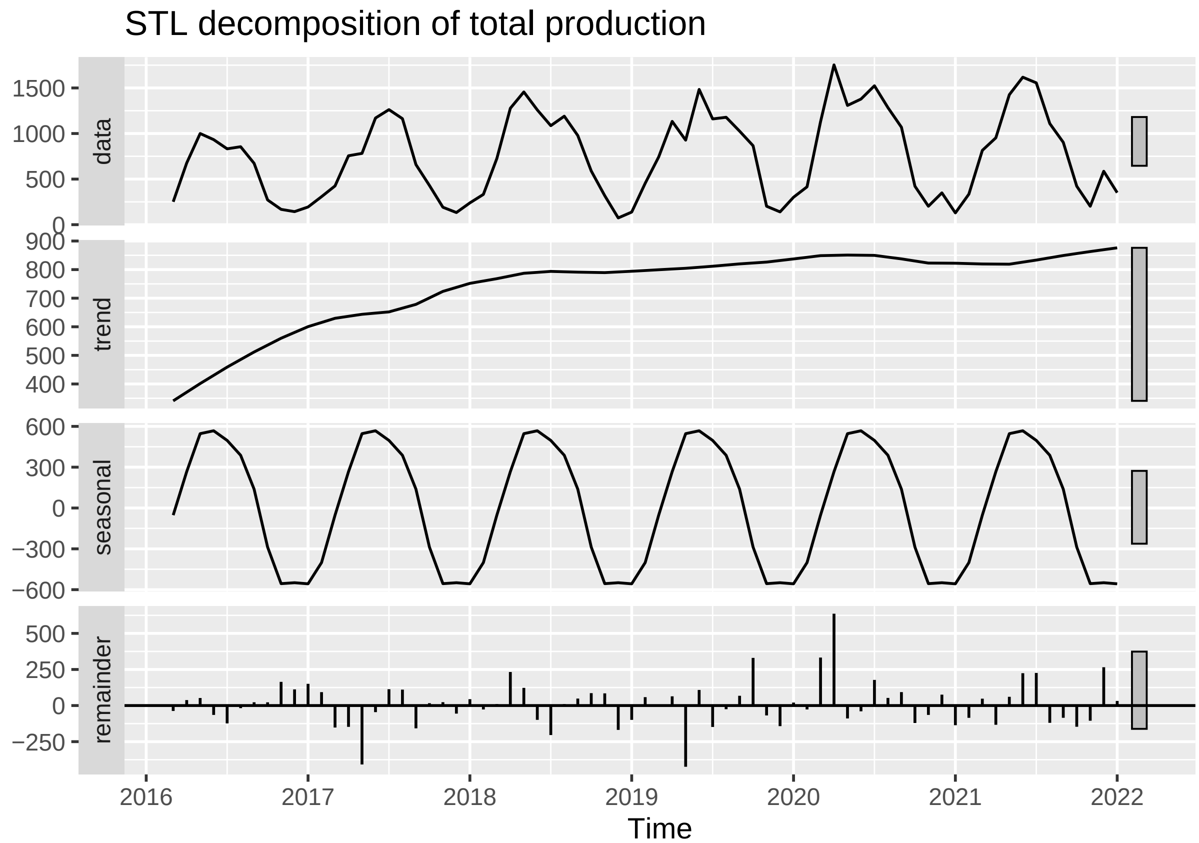
To make sure that data is indeed seasonal, which would be a valuable information in the following steps, following seasonal plots were examined:



Total energy production
KWh
0
500
1000
1500
sty
lut
mar
kwi
maj
cze
lip
sie
wrz
paz
lis
gru
Month
2016
2017
2018
2019
2020
2021
2022



As shown in the figures above, strong seasonal trend in energy production occuring in summer is clearly seen, and frankly expected - summer in northern hemisphere, and thus the most suntime, occurs in the middle of the year. Slight growing trend can also be noticed in second figure, which can be further confirmed by STL decomposition:



There is a slightly growing trend indeed, and remainders are in tolerable range - bigger values in March and April 2020 usually raise some questions about the correlation with the pandemic outburst and first lockdowns, yet in this case it just means that these two months were unusually sunny.

Forecasts

With preprocessing and exploration done, the next step is to forecast data for upcoming months - using ARIMA, ETS and NNAR methods. In order to NNAR to work with external regressor, values of transaction balance were predicted independently, as they are not yet known.

ARIMA

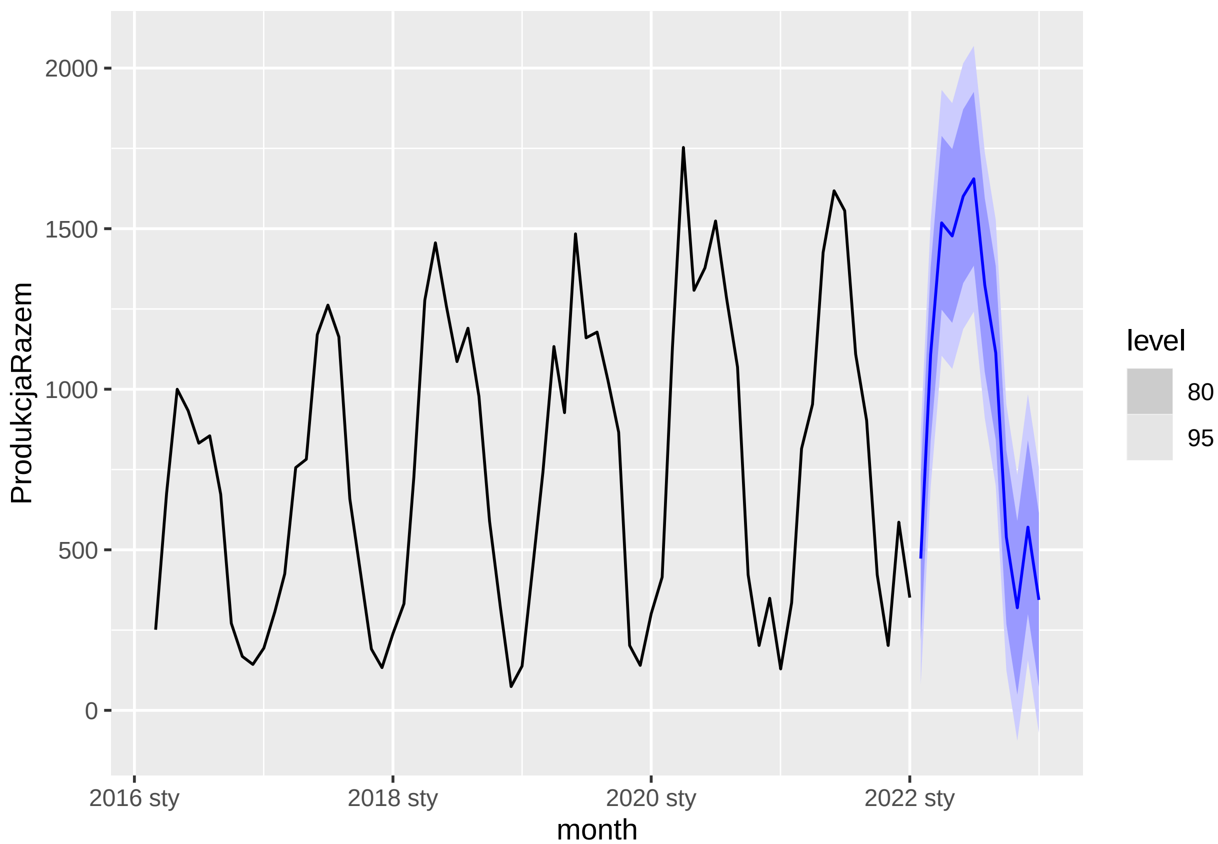
To find out which parameters would be good enough for non-seasonal and seasonal parts of ARIMA model, options `stepwise` and `approximation` were turned off - the best model turned out to be `ARIMA(0,0,1)(1,1,0)` with drift.

```
energia %>%
  model(auto = ARIMA(ProdukcjaRazem, stepwise = F, approximation = F)) -> fit_ari

forecast(fit_ari, h=12) %>%
  filter(.model=='auto') %>%
  hilo() %>%
  select(-ProdukcjaRazem,-.model)
```

```
## # A tsibble: 12 x 4 [1M]
##       month .mean                      `80%`                    `95%`
##       <mth> <dbl>                     <hilo>                   <hilo>
##  1 2022 lut  473. [ 216.19788,  729.1559]80 [  80.42608,  864.9277]95
##  2 2022 mar 1108. [ 837.51988, 1378.6831]80 [ 694.28263, 1521.9203]95
##  3 2022 kwi 1518. [1247.61602, 1788.7792]80 [1104.37877, 1932.0165]95
##  4 2022 maj 1477. [1206.65665, 1747.8199]80 [1063.41940, 1891.0571]95
##  5 2022 cze 1601. [1330.35268, 1871.5159]80 [1187.11543, 2014.7531]95
##  6 2022 lip 1655. [1384.80535, 1925.9686]80 [1241.56810, 2069.2058]95
##  7 2022 sie 1324. [1053.69815, 1594.8614]80 [ 910.46090, 1738.0986]95
##  8 2022 wrz 1113. [ 842.65933, 1383.8225]80 [ 699.42208, 1527.0598]95
##  9 2022 paź  539. [ 268.72114,  809.8844]80 [ 125.48390,  953.1216]95
## 10 2022 lis  319. [  48.72114,  589.8844]80 [ -94.51610,  733.1216]95
## 11 2022 gru  571. [ 300.03229,  841.1955]80 [ 156.79504,  984.4327]95
## 12 2023 sty  344. [  73.43032,  614.5935]80 [ -69.80693,  757.8308]95
```

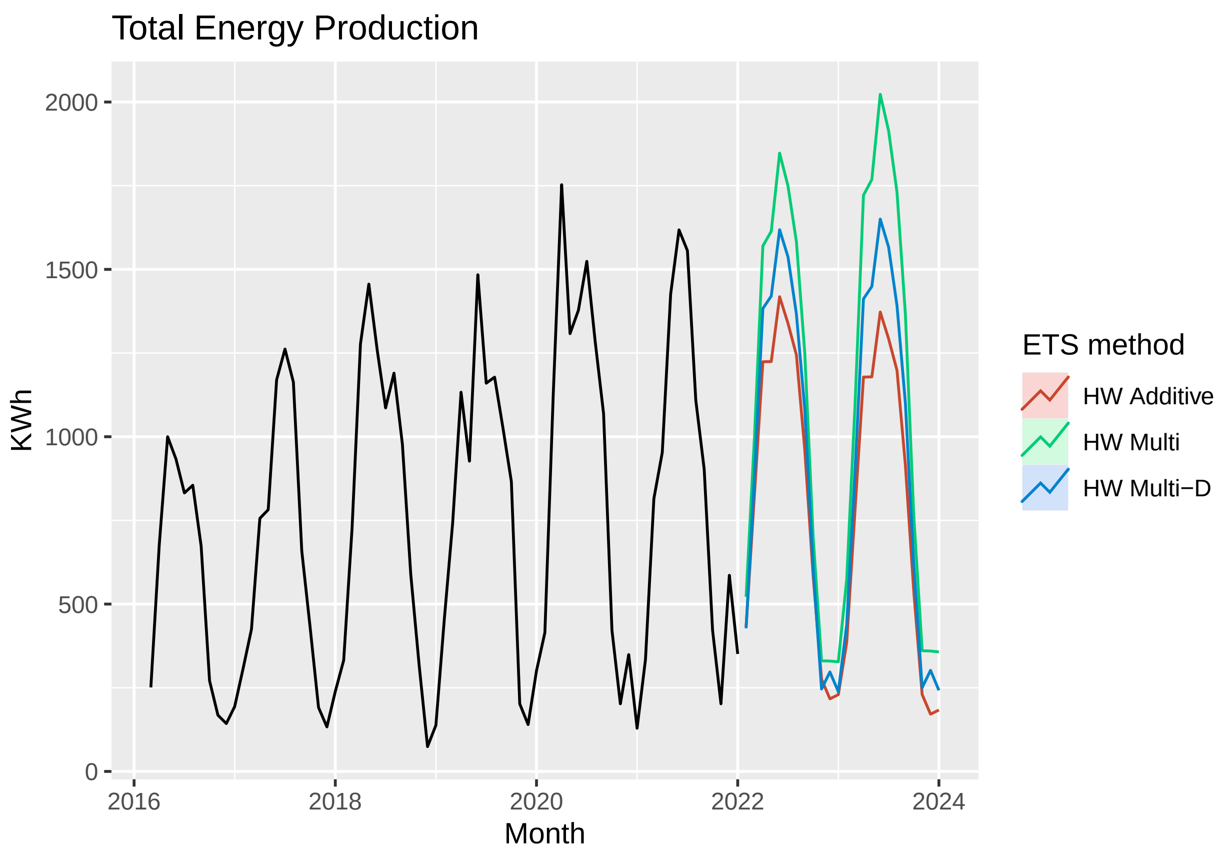
Corresponding plot:



ETS

Since data is seasonal, appropriate ETS method to deal with is Holt-Winters method. To select the one that doesn't under- or overestimate, three methods have been checked - additive, multiplicative, and multiplicative with damping:

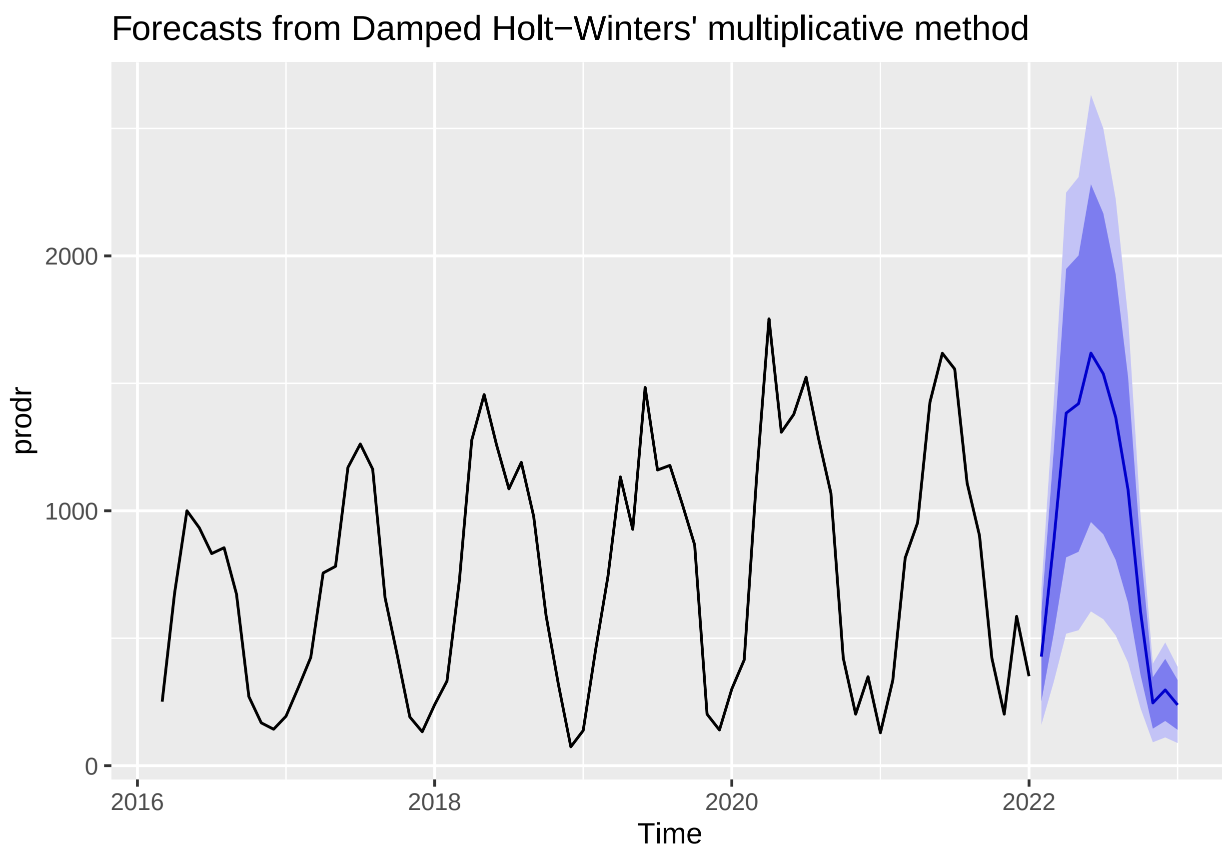
```
prodr <- ts(energia$ProdukcjaRazem, start = c(2016,3), frequency = 12)
ets_add <- hw(prodr, seasonal = "additive")
ets_mul <- hw(prodr, seasonal = "multiplicative")
ets_dmp <- hw(prodr, damped = T, seasonal = "multiplicative")
```



The most stable method of these three seems to be multiplicative with damping, and it is going to be considered from here on. It is not too optimistic as mulitplicative method without damping, nor it is decaying too much with time, contrary to an observed trend.

```
ets_dmp %>% forecast(h=12)
```

```
##          Point Forecast    Lo 80     Hi 80     Lo 95     Hi 95
## Feb 2022       427.7523 252.8246  602.6799 160.22349  695.2810
## Mar 2022       879.4450 519.7066 1239.1833 329.27267 1429.6173
## Apr 2022      1383.0841 817.1849 1948.9832 517.61603 2248.5521
## May 2022      1420.2641 839.0014 2001.5268 531.29947 2309.2287
## Jun 2022      1618.5201 955.9454 2281.0949 605.19947 2631.8408
## Jul 2022      1537.1846 907.7411 2166.6280 574.53386 2499.8353
## Aug 2022      1366.5036 806.8028 1926.2044 510.51512 2222.4921
## Sep 2022      1081.5649 638.4538 1524.6760 403.88492 1759.2449
## Oct 2022       605.2379 357.2094  853.2665 225.91105  984.5648
## Nov 2022       246.1042 145.2230  346.9855  91.81967  400.3888
## Dec 2022       297.1411 175.3066  418.9757 110.81123  483.4710
## Jan 2023       238.3364 140.5868  336.0860  88.84128  387.8316
```



Though the forecast itself looks very similar to the one produced with ARIMA, the prediction intervals are much larger.

NNAR

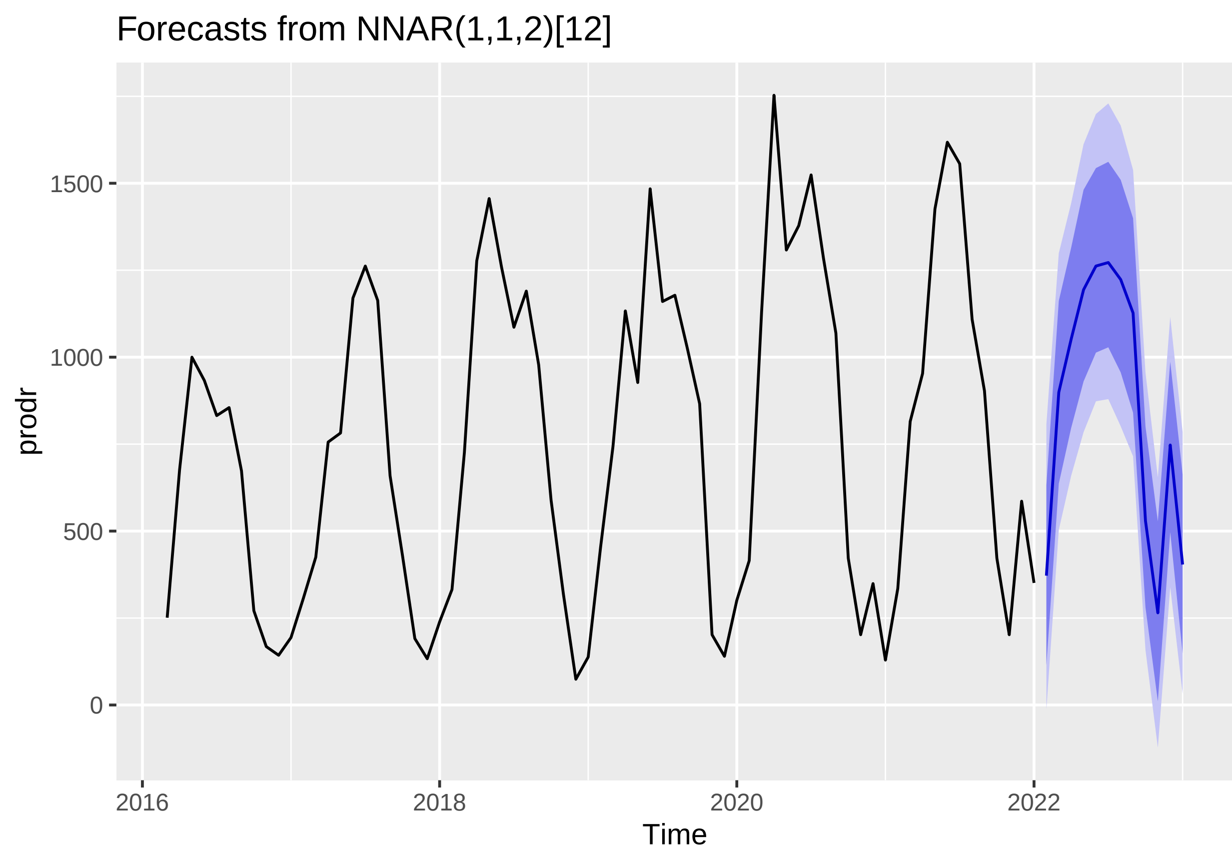
Last method used in the forecast is Neural Network Autoregression Model, which is not expected to outperform ARIMA - neural networks tend to have hard time working with trending data, and although an external regressor can help greatly with explaining the data, to forecast we should have new values for this regressor. To bypass that, future values of Balance, which was chosen to be a regressor, have been independently forecasted. Each of 20 networks produced each time is then averaged to produce final model. Forecast without an external regressor:

```
fitnn <- nnetar(prodr)
nnetforecast <- forecast(fitnn, h=12, PI = T)
nnetforecast
```

```
##          Point Forecast       Lo 80     Hi 80      Lo 95     Hi 95
## Feb 2022       371.8701  114.11035  632.5593  -12.77514  807.3017
## Mar 2022       898.8732  636.52832 1161.5786  503.53073 1298.7243
## Apr 2022      1051.5747  796.52165 1314.6475  659.15142 1442.6020
## May 2022      1193.8423  930.15822 1481.0351  784.34182 1612.0757
## Jun 2022      1261.9611 1013.11257 1543.5319  873.12663 1699.0019
## Jul 2022      1272.0796 1028.39895 1561.5970  879.36119 1729.4642
## Aug 2022      1223.1092  956.16481 1509.7547  801.32726 1665.9169
## Sep 2022      1126.8563  840.33777 1399.3825  714.88179 1538.1738
```

```
## Oct 2022       529.3970  280.85178  800.6322  158.32955  955.6715
## Nov 2022       264.9068   10.82373  527.9841 -122.49742  654.5677
## Dec 2022       747.4756  497.31582  987.0724  338.95282 1115.4308
## Jan 2023       403.8503  148.39725  662.6891   32.92804  784.6282
```

```
autoplot(nnetforecast)
```



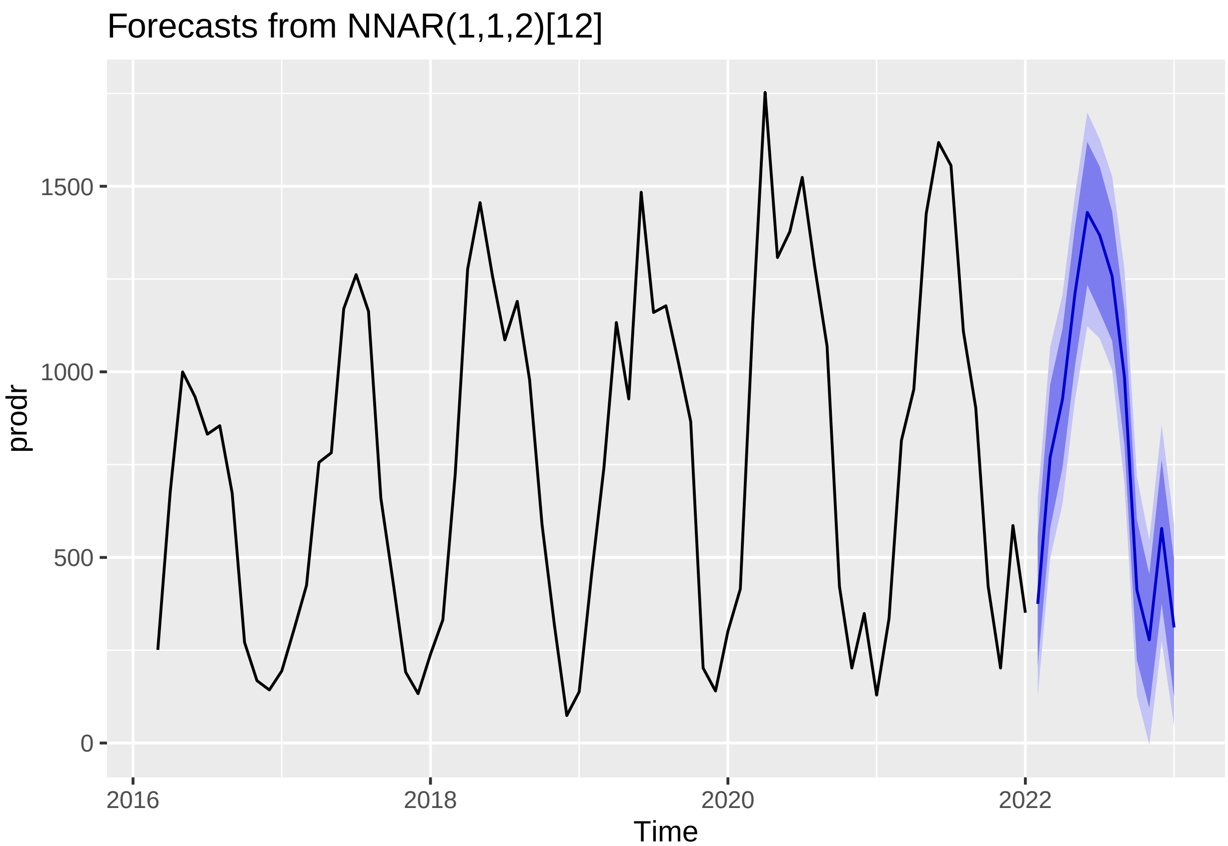
Prediction intervals are again very wide, as it was the case with ARIMA model. Finally, a model using external regressor of Balance of energy sold/bought :

```
fitnn2 <- nnetar(prodr, xreg = energia$Bilans)
balance_forecast <- c(-964.29,-588.84,213.32,638.36,923.74,789.57,698.11,361.37,-363.04,-827.66,-1507.6
nnetforecast2 <- forecast(fitnn2, PI=T, xreg = balance_forecast)
nnetforecast2
```

```
##          Point Forecast      Lo 80     Hi 80       Lo 95     Hi 95
## Feb 2022       374.8774  207.63858  557.6605  126.736424  658.5783
## Mar 2022       769.7026  577.87719  964.9936  497.640846 1065.9684
## Apr 2022       926.5775  742.89763 1115.6408  644.260919 1207.8935
## May 2022      1210.0008 1015.44456 1388.5649  924.653397 1476.1315
## Jun 2022      1429.7332 1234.61038 1619.7360 1122.853339 1699.2682
## Jul 2022      1367.8774 1161.31228 1552.8398 1089.878440 1627.1877
## Aug 2022      1257.9495 1083.68384 1431.7199 1005.260296 1526.9803
## Sep 2022       984.9729  800.27195 1165.5115  695.114639 1275.1594
## Oct 2022       412.2716  223.35015  602.4178  127.907964  717.8826
```

```
## Nov 2022       277.8476   94.44903  454.2928   -4.571555  546.9441
## Dec 2022       578.2546  375.57301  762.7909  274.521135  856.3878
## Jan 2023       311.3551  123.67115  492.6132   46.990851  588.5550
```

```
autoplot(nnetforecast2)
```



It's interesting to see how one variable can narrow the prediction intervals, and how much accuracy one can gain by knowing it beforehand.