Hackathon

Time Series Forecasting 2022

Università degli Studi di Milano

Mathias CARDARELLO

Gaspare MATTARELLA

Dataset

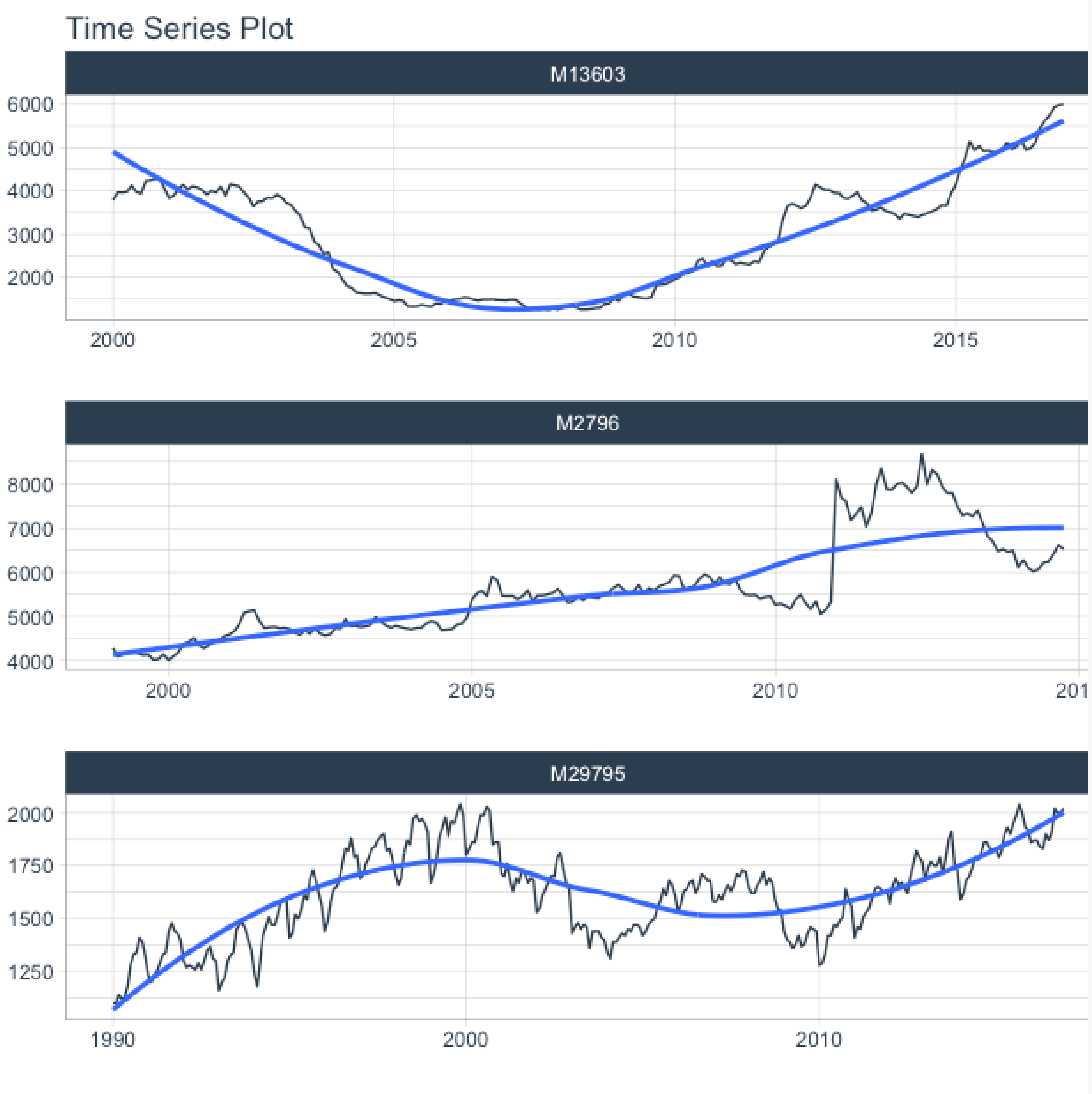
The data comes from the M4-Competition, the fourth of the Makridakis Competitions, a series of open competitions to evaluate and compare the accuracy of different time series forecasting methods. The exam dataset contains a subset of 120 time series:

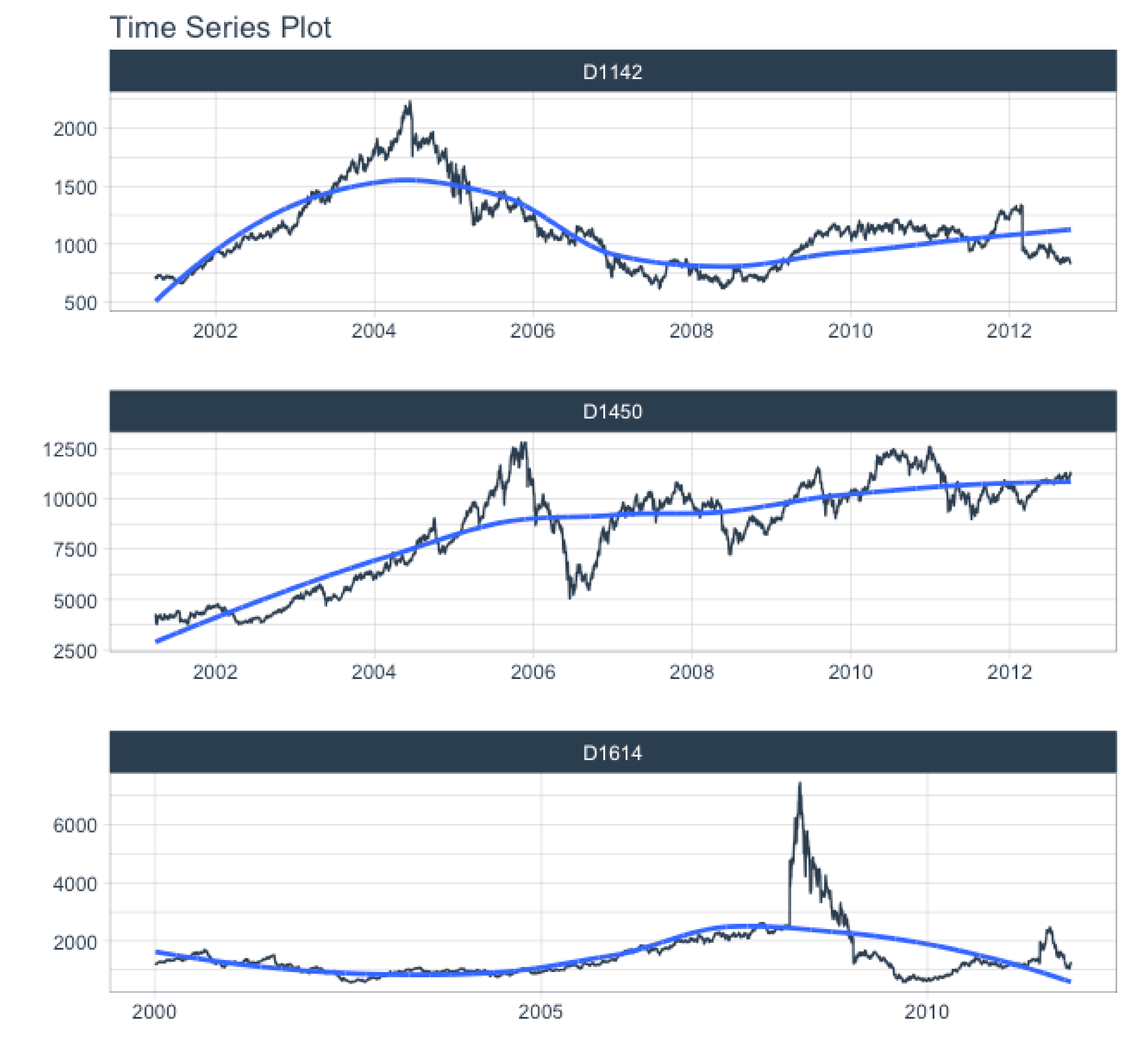
- 20 hourly time series
- 20 daily time series
- 20 weekly time series
- 20 monthly time series
- 20 quarterly time series
- 20 yearly time series

Overview

The Time Series present different shapes, ranges and trends

Sample of 3 Monthly TS

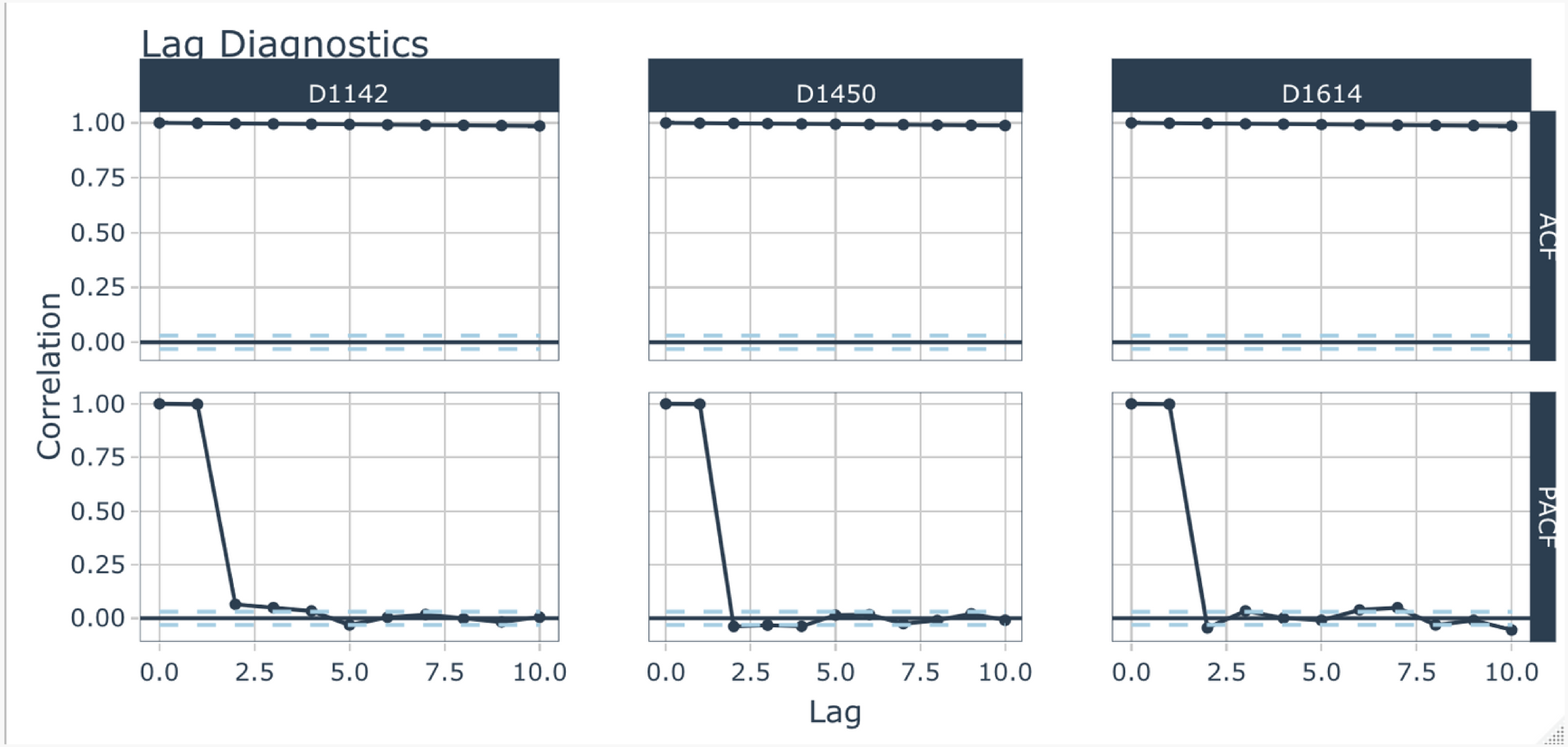




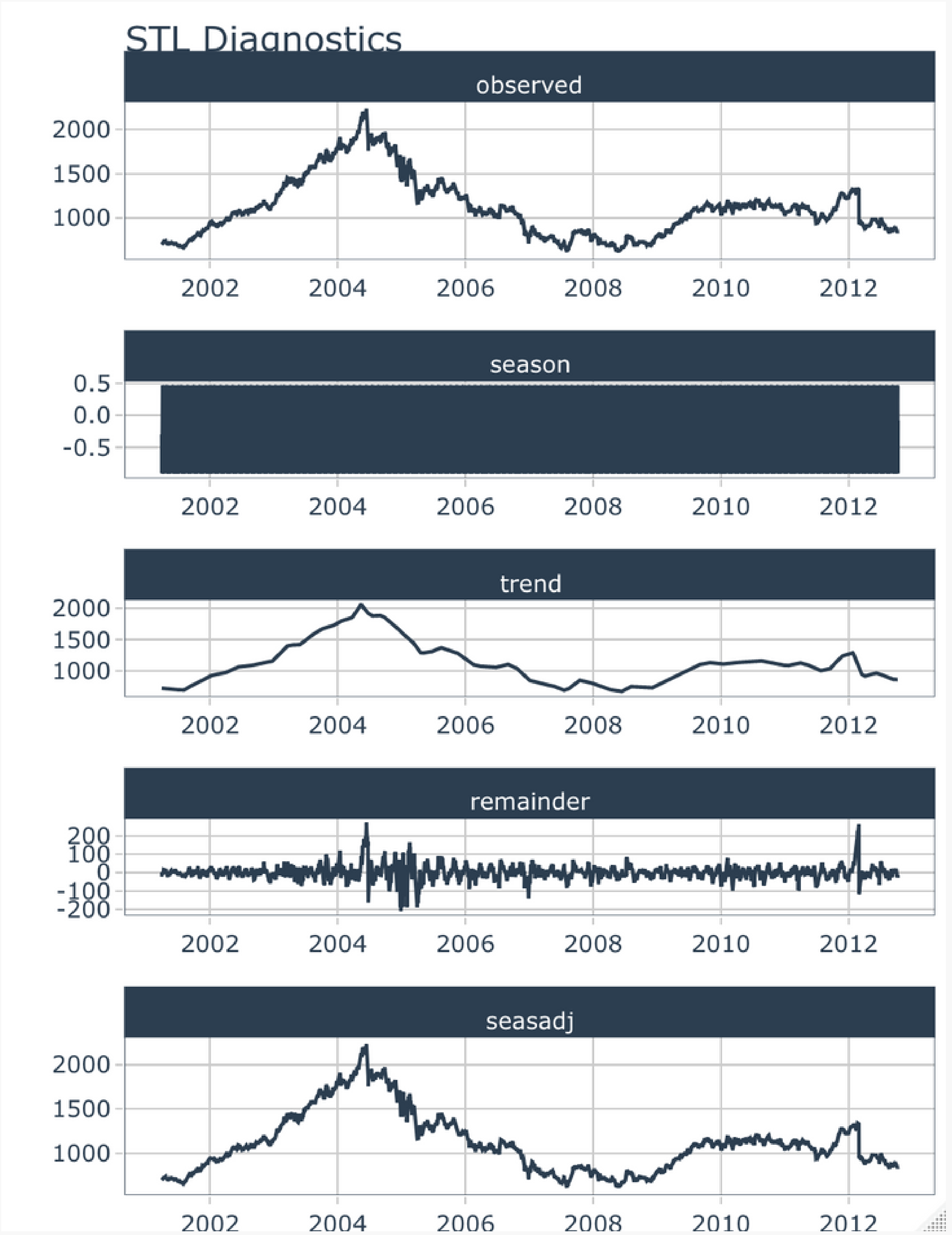
Sample of 3 Daily TS

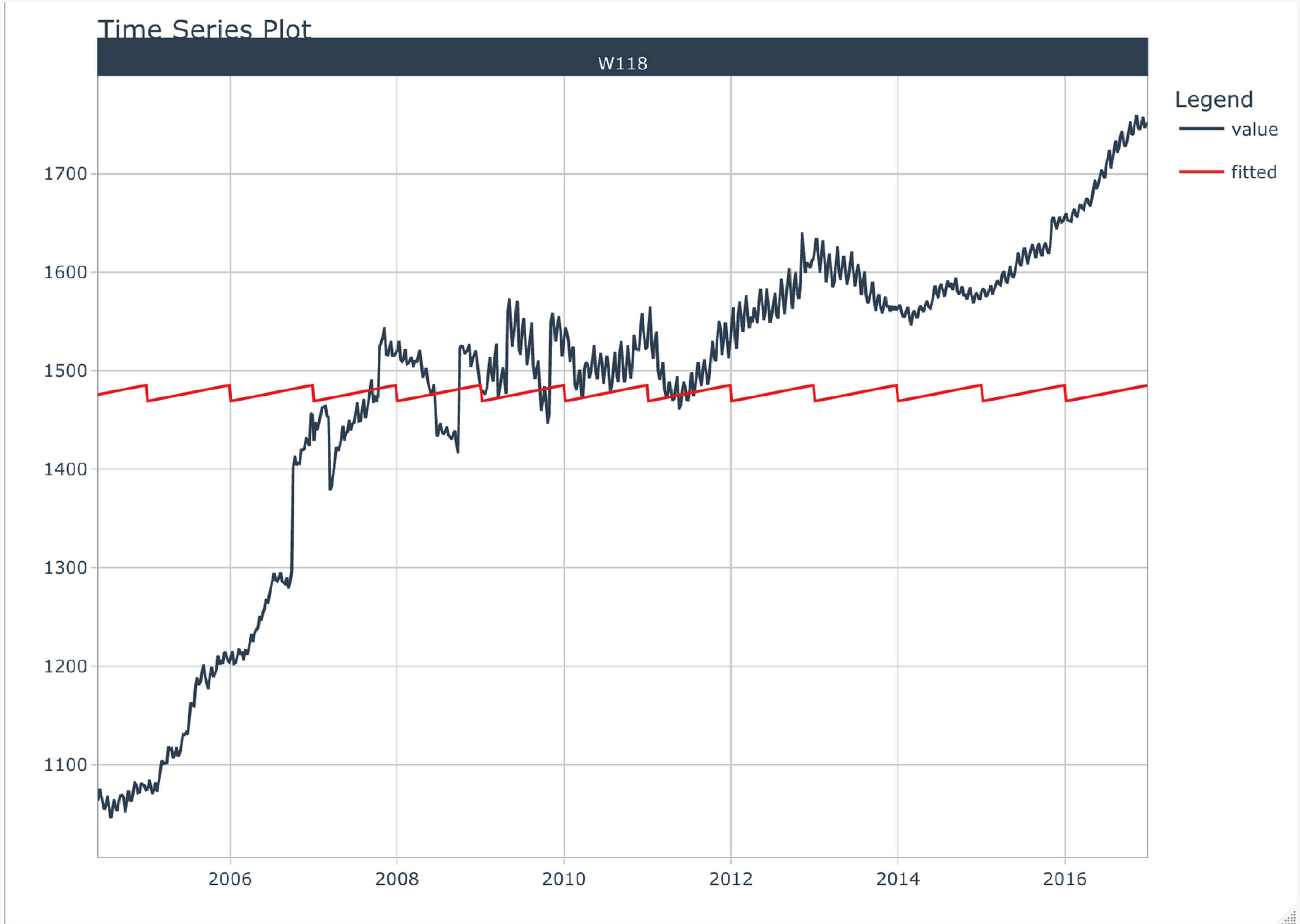
EDA

1. Checking the Autocorrelations

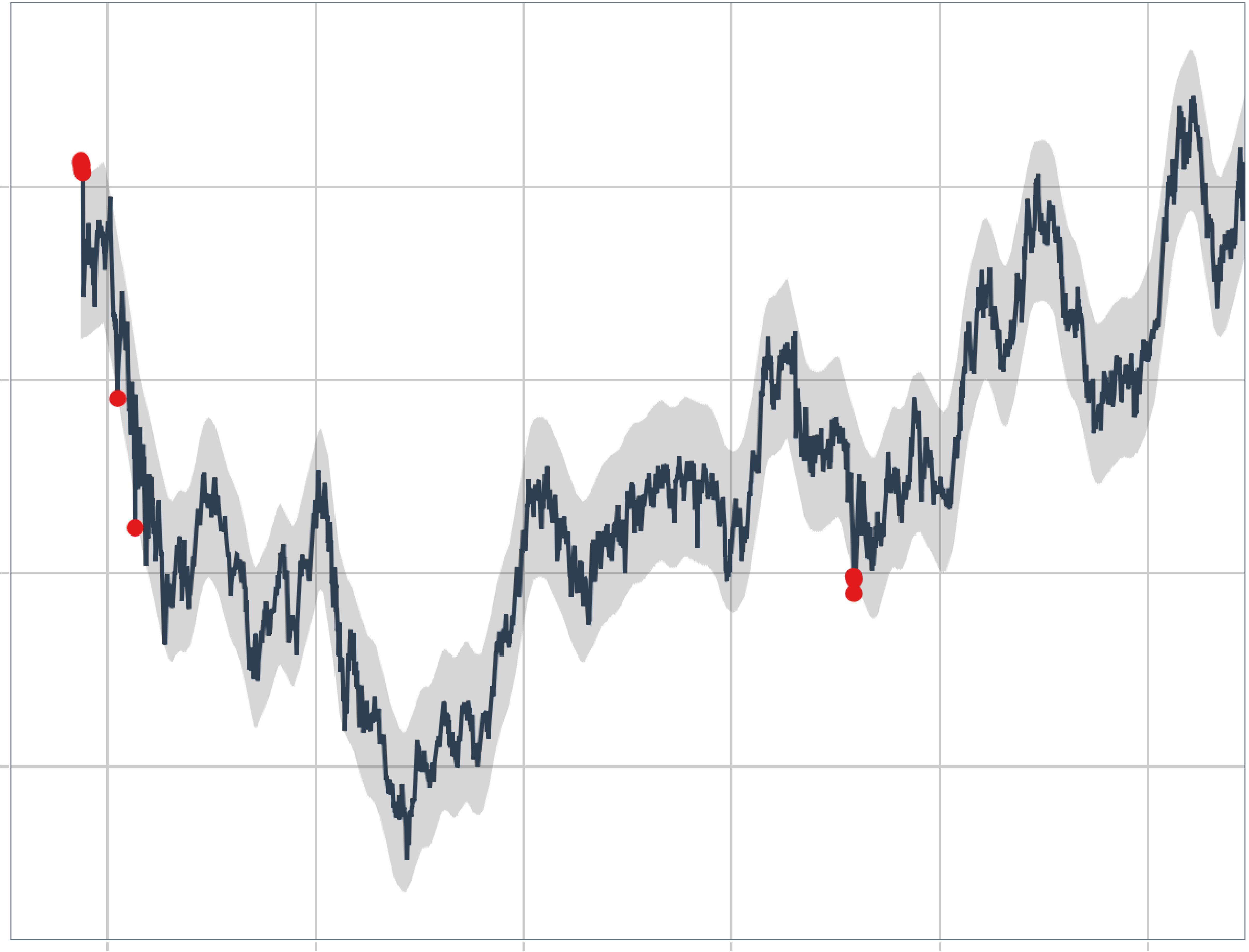


2. Checking Seasonal Components





3. Anomaly Detection



Preprocessing

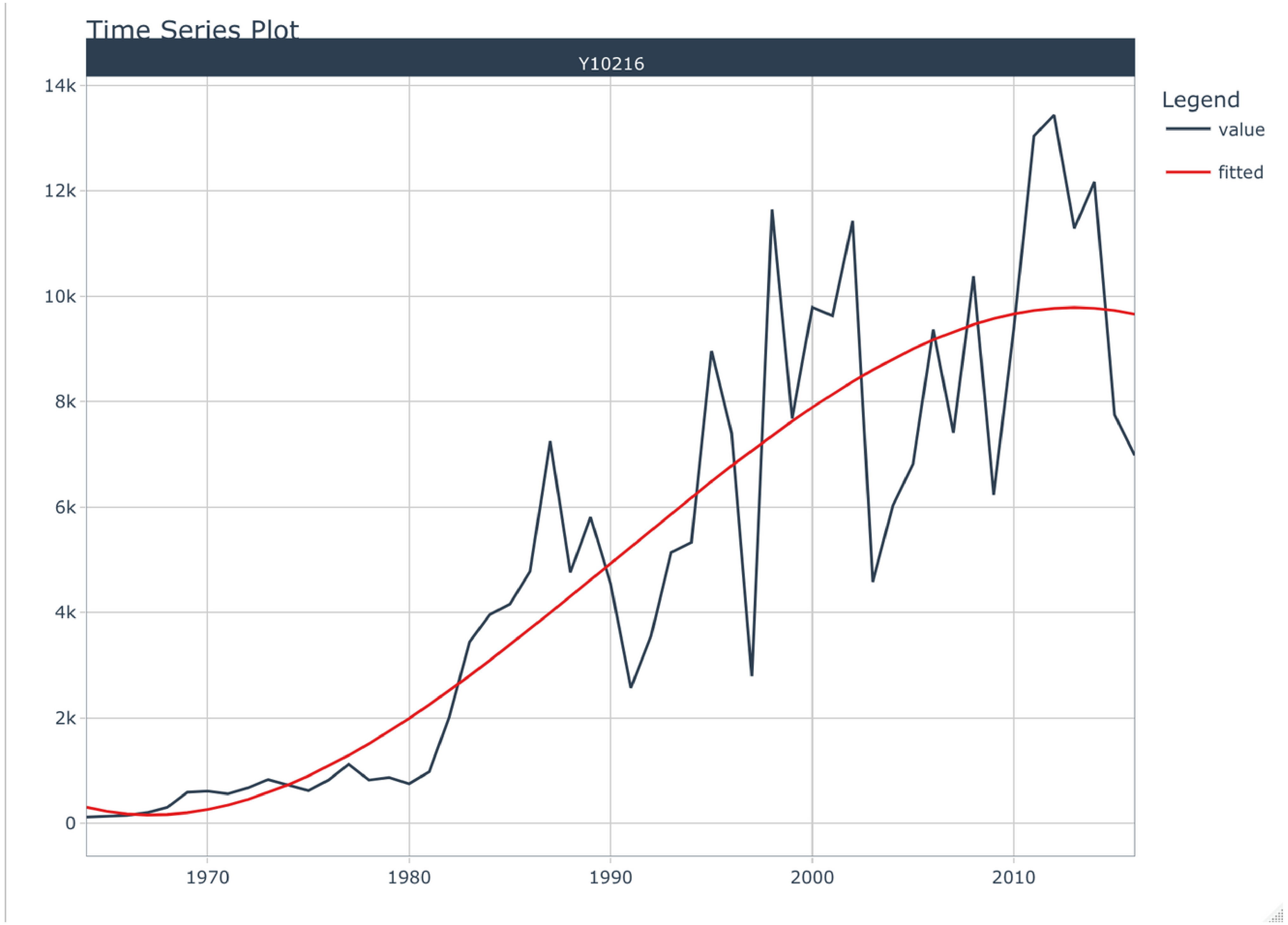
- Padding on all the series

Feature Engineering

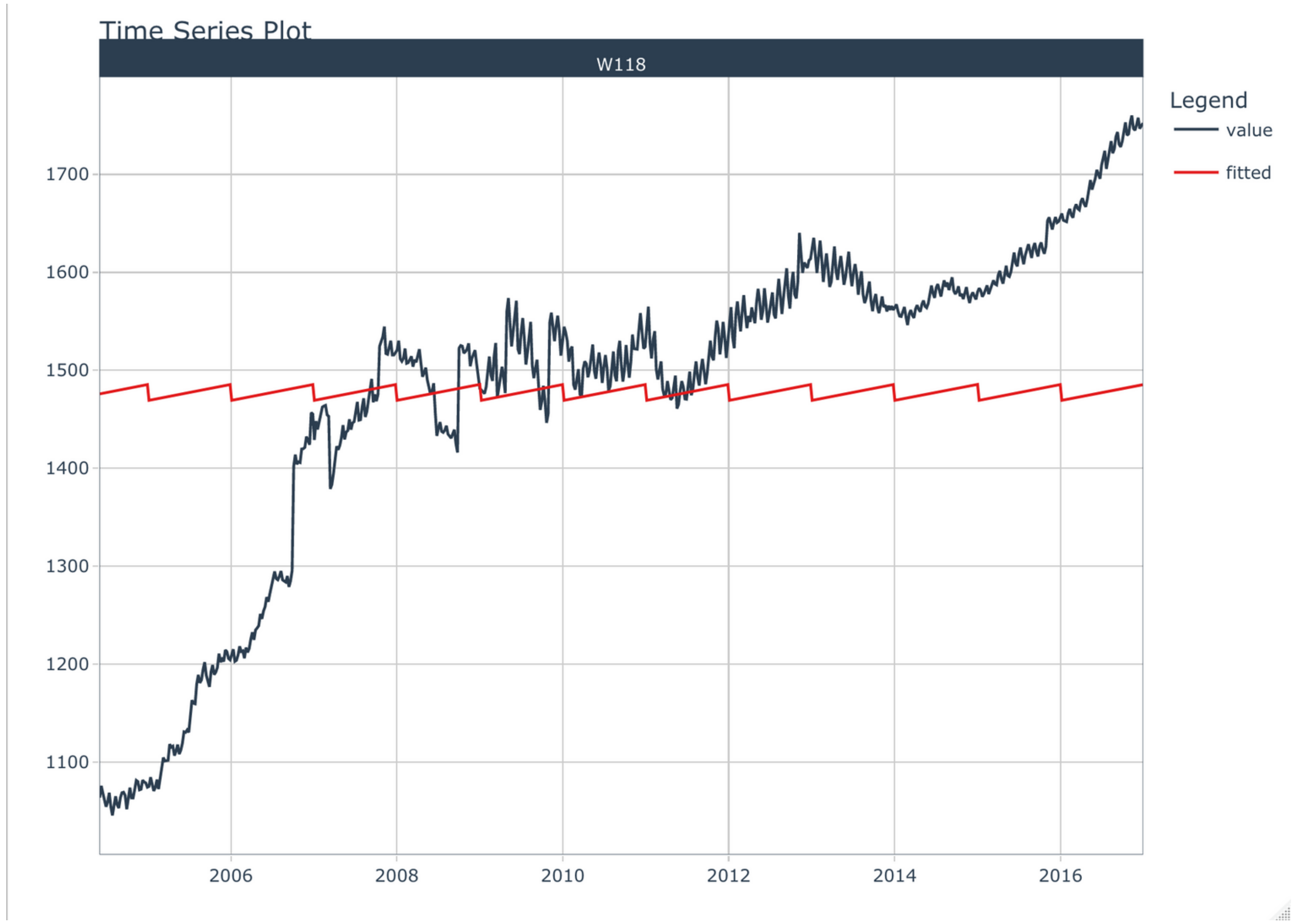
- TS Signatures

```
Groups: id [20]
$ id        <chr> "Y10216", "Y10216", "Y10216", "Y10216", "Y10216", "Y10216"…
$ date      <dttm> 1963-12-31 12:00:00, 1964-12-31 12:00:00, 1965-12-31 12:0…
$ value     <dbl> 120, 130, 150, 200, 300, 590, 610, 560, 670, 830, 720, 620…
$ type      <chr> "train", "train", "train", "train", "train", "train", "tra…
$ period    <chr> "Yearly", "Yearly", "Yearly", "Yearly", "Yearly", "Yearly"…
$ index.num <dbl> -189432000, -157809600, -126273600, -94737600, -63201600, …
$ year      <int> 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970, 1971, 1972…
$ half      <int> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2…
$ quarter   <int> 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4…
$ month     <int> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12…
$ month.lbl <ord> December, December, December, December, December, December…
$ day       <int> 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31…
$ wday      <int> 3, 5, 6, 7, 1, 3, 4, 5, 6, 1, 2, 3, 4, 6, 7, 1, 2, 4, 5, 6…
$ wday.lbl  <ord> Tuesday, Thursday, Friday, Saturday, Sunday, Tuesday, Wedn…
$ mday      <int> 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31…
$ qday      <int> 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92…
$ yday      <int> 365, 366, 365, 365, 365, 366, 365, 365, 365, 366, 365, 365…
$ mweek     <int> 5, 5, 5, 5, 6, 5, 5, 5, 5, 6, 6, 5, 5, 5, 5, 6, 6, 5, 5, 5…
$ week      <int> 53, 53, 53, 53, 53, 53, 53, 53, 53, 53, 53, 53, 53, 53, 53…
$ week2     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1…
$ week3     <int> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2…
$ week4     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1…
$ mday7     <int> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5…
```

We can already fit our TS on linear or non linear trends as in example



Or we can fit it on a their seasonal components



Feature Engineering

- Rolling Mean

```
Groups: id [20]
$ id           <chr> "D1017", "D1017", "D1017", "D1017", "D1017", "D1017", …
$ date         <dttm> 2001-09-28 12:00:00, 2001-09-29 12:00:00, 2001-09-30 …
$ value        <dbl> 2561.97, 2567.91, 2564.15, 2550.26, 2542.13, 2548.05, …
$ type         <chr> "train", "train", "train", "train", "train", "train", …
$ period       <chr> "Daily", "Daily", "Daily", "Daily", "Daily", "Daily", …
$ value_roll_7  <dbl> 2561.073, 2557.284, 2555.745, 2555.773, 2552.369, 2547…
$ value_roll_14 <dbl> 2553.569, 2551.678, 2531.139, 2502.455, 2483.890, 2472…
$ value_roll_30 <dbl> 2442.059, 2437.244, 2432.952, 2429.309, 2423.648, 2420…
$ value_roll_90 <dbl> 2368.128, 2365.619, 2363.115, 2361.461, 2359.592, 2356…
```

Feature Engineering

- Lag Features

```
$ value_lag1   <dbl> 1532.71, 1586.86, 1467.35, 1476.74,…
$ value_lag7   <dbl> 1485.23, 1495.11, 1517.30, 1509.71,…
$ value_lag14  <dbl> 1525.14, 1527.11, 1523.21, 1485.04,…
$ value_lag30  <dbl> 1531.70, 1534.56, 1551.40, 1572.63,…
$ value_lag90  <dbl> 1689.02, 1653.54, 1613.29, 1593.18,…
$ value_lag365 <dbl> 2561.97, 2567.91, 2564.15, 2550.26,…
```

Feature Engineering

Fourier Series Features

```
$ date_sin1_K1   <dbl> -2.073814e-12, 8.179562e-12, 3.881023e-12, -…
$ date_cos1_K1   <dbl> -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, …
$ date_sin1_K2   <dbl> 4.147629e-12, -1.635912e-11, -7.762047e-12, …
$ date_cos1_K2   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,…
$ date_sin7_K1   <dbl> 9.749279e-01, 7.818315e-01, 8.142876e-13, -7…
$ date_cos7_K1   <dbl> 0.2225209, -0.6234898, -1.0000000, -0.623489…
$ date_sin7_K2   <dbl> 4.338837e-01, -9.749279e-01, -1.628575e-12, …
$ date_cos7_K2   <dbl> -0.9009689, -0.2225209, 1.0000000, -0.222520…
…
```

Modelling Time!

Given our splits in train and test data, this is what we have to forecast

```
# Hourly: 48 hours to forecast
# Daily: 14 days to forecast
# Weekly: 13 weeks to forecast
# Monthly: 18 months to forecast
# Yearly: 6 years to forecast
# Quarterly: 8 quarters to forecast
```

Forecasting methods

Naive models: baseline

- Window - Mean
- Window - Weighted Mean

Iteratively forecast with nested modeling

- Prophet
- XGBoost
- Ensembles (mean and weighted mean)

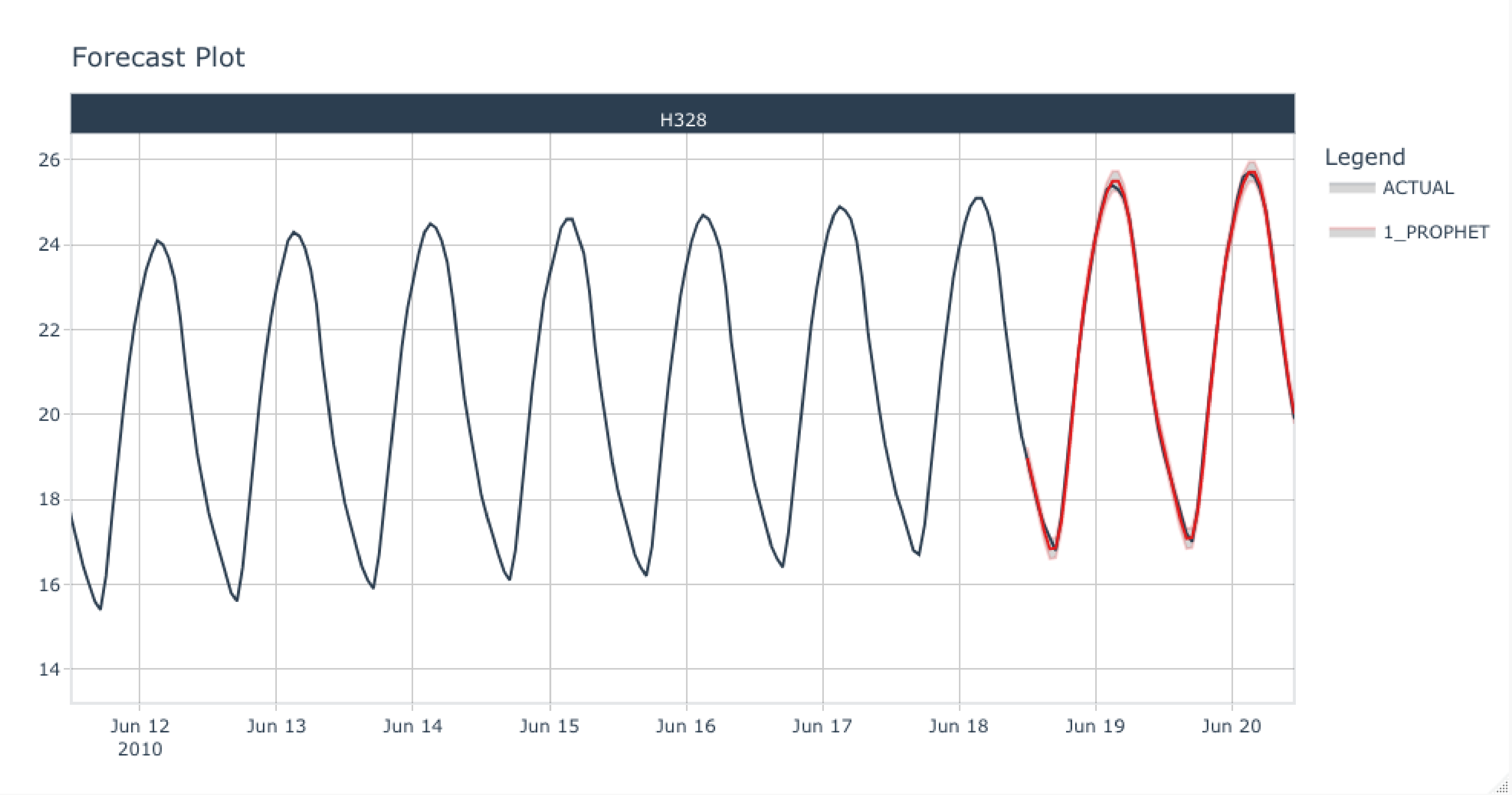
Libraries used

- library(tidymodels)
- library(modeltime)
- library(modeltime.ensemble)
- library(tidyverse)
- library(timetk)
- library(gt)

Hourly series

Search

id	.model_id	.model_desc	.type	mae	mape	mase	smape	rmse
▾ H118 (1)								
	1	PROPHET	Test	21.77	7.98	1.15	7.75	25.07
▾ H137 (1)								
	2	XGBOOST	Test	58.07	38.74	0.6	29.14	74.5
▾ H14 (1)								
	1	PROPHET	Test	14.88	8.5	2.52	8.97	16.71
▾ H153 (1)								
	2	XGBOOST	Test	73.99	14.13	0.68	14.19	103.75
▾ H179 (1)								
	1	PROPHET	Test	0.18	0.86	0.22	0.86	0.21
▾ H195 (1)								
	1	PROPHET	Test	0.19	0.93	0.34	0.93	0.22

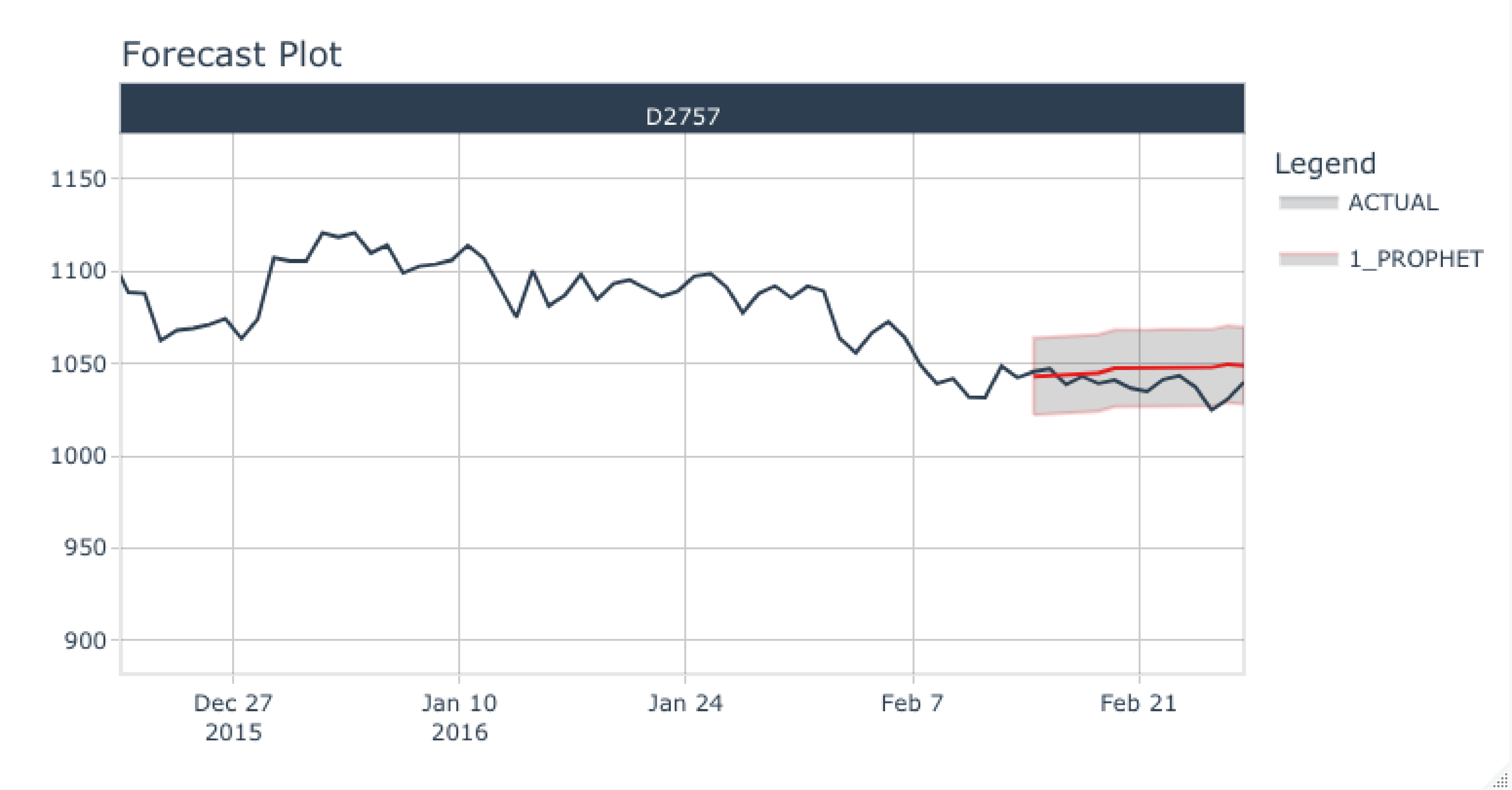


RMSE: 0.12
SMAPE: 0.49

Daily series

Search

id	.model_id	.model_desc	.type	mae	mape	mase	smape	rmse	rsq
D1017 (1)									
	2	XGBOOST	Test	159.9	6.26	8.27	6.49	170.08	0.81
D1142 (1)									
	2	XGBOOST	Test	12.66	1.46	1.64	1.45	14.16	0.01
D1450 (1)									
	2	XGBOOST	Test	150.76	1.35	2.74	1.36	168.66	0.31
D1614 (1)									
	2	XGBOOST	Test	268.32	23.71	7.77	21.08	274.15	0.56
D1627 (1)									
	2	XGBOOST	Test	175.32	3.34	2.14	3.36	196.66	0.2
D1790 (1)									
	1	PROPHET	Test	117.57	5.8	2.41	5.58	138.18	0.89
D1842 (1)									
	2	XGBOOST	Test	35.94	1.17	3.41	1.18	38.79	
D2013 (1)									

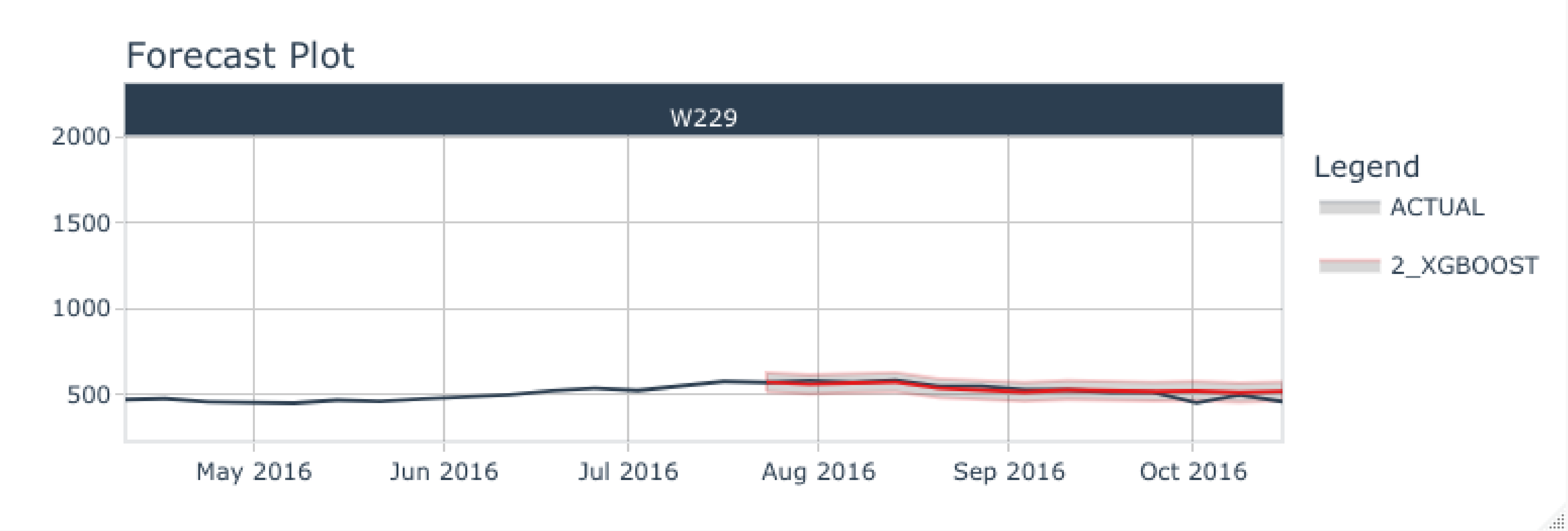


RMSE: 10.45
SMAPE: 0.82

Weekly series

Search

id	.model_id	.model_desc	.type	mae	mape	mase	smape	rmse	rsq
W118 (1)									
	2	XGBOOST	Test	51.12	2.96	5.68	3	52.06	0.02
W137 (1)									
	1	PROPHET	Test	109.25	1.57	3.12	1.58	117.51	0.42
W14 (1)									
	1	PROPHET	Test	69.32	3.01	1.11	3.07	92.46	0.68
W153 (1)									
	1	PROPHET	Test	88.17	2.93	0.79	2.97	110.36	0.09
W179 (1)									
	1	PROPHET	Test	31.44	1.16	4.36	1.16	33.9	0.19
W195 (1)									
	1	PROPHET	Test	76.93	7.17	2.13	6.76	104.63	0.76

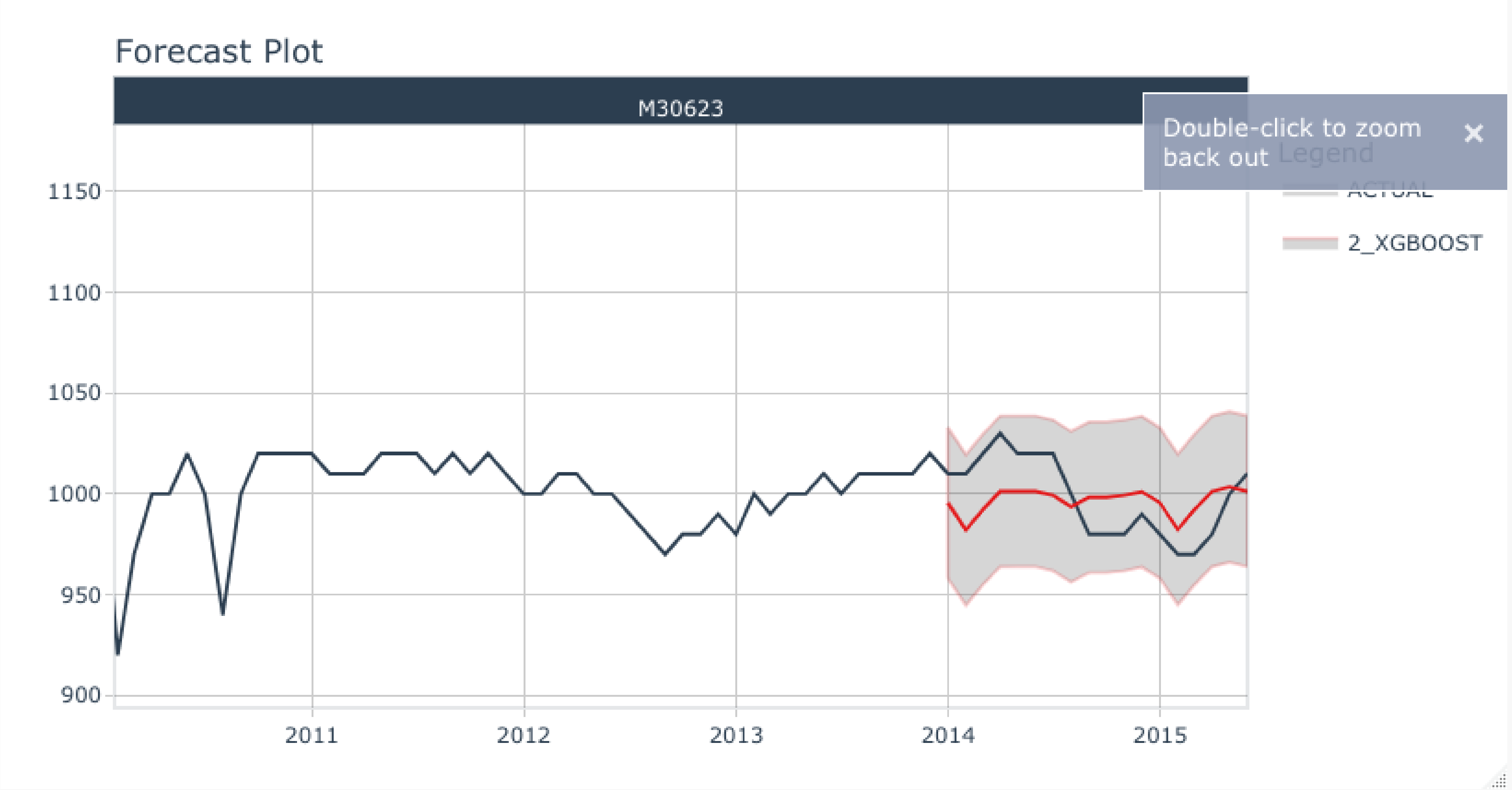


RMSE: 27.76
SMAPE: 3.57

Monthly series

Search

id	.model_id	.model_desc	.type	mae	mape	mase	smape	rmse	rsq
M13603 (1)									
	1	PROPHET	Test	589.65	16.17	4.47	14.75	647.53	0.85
M14250 (1)									
	1	PROPHET	Test	209.33	3.45	0.54	3.42	253.51	0.78
M15247 (1)									
	2	XGBOOST	Test	666.17	27.07	1.71	33.34	941.48	0.29
M24240 (1)									
	1	PROPHET	Test	529.75	6.72	0.78	6.52	615.44	0.63
M24608 (1)									
	2	XGBOOST	Test	318.54	8.58	1.19	8.43	389.56	0
M27235 (1)									
	2	XGBOOST	Test	40.66	3.65	4.11	3.57	50.93	0
M2796 (1)									
	2	XGBOOST	Test	422.88	5.24	2.05	5.47	577.08	0
M29066 (1)									

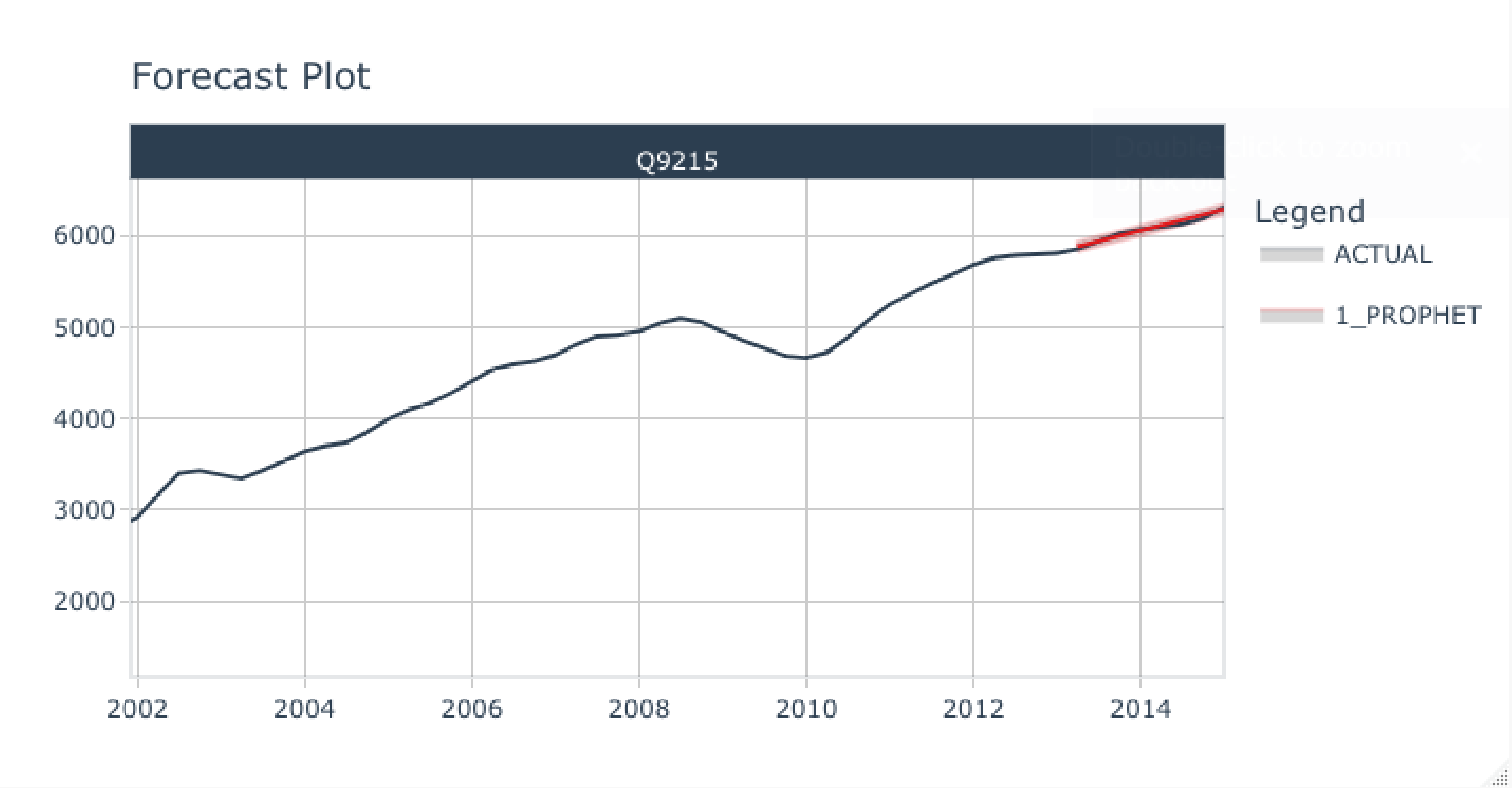


RMSE: 18.80
SMAPE: 1.75

Quarterly series

Search

id	.model_id	.model_desc	.type	mae	mape	mase	smape	rmse	rsq
▼ Q10211 (1)									
	2	XGBOOST	Test	76.19	8.47	1.52	8.02	86.15	0.43
▼ Q11644 (1)									
	1	PROPHET	Test	113.49	1.5	1.32	1.48	134.34	0.89
▼ Q12505 (1)									
	1	PROPHET	Test	153.95	9.52	0.74	9.36	190.23	0.23
▼ Q12642 (1)									
	1	PROPHET	Test	5032.91	33.51	1.96	40.95	5374.09	0.11
▼ Q13676 (1)									
	1	PROPHET	Test	92.91	4.85	2.98	4.72	96.09	0.94
▼ Q16137 (1)									

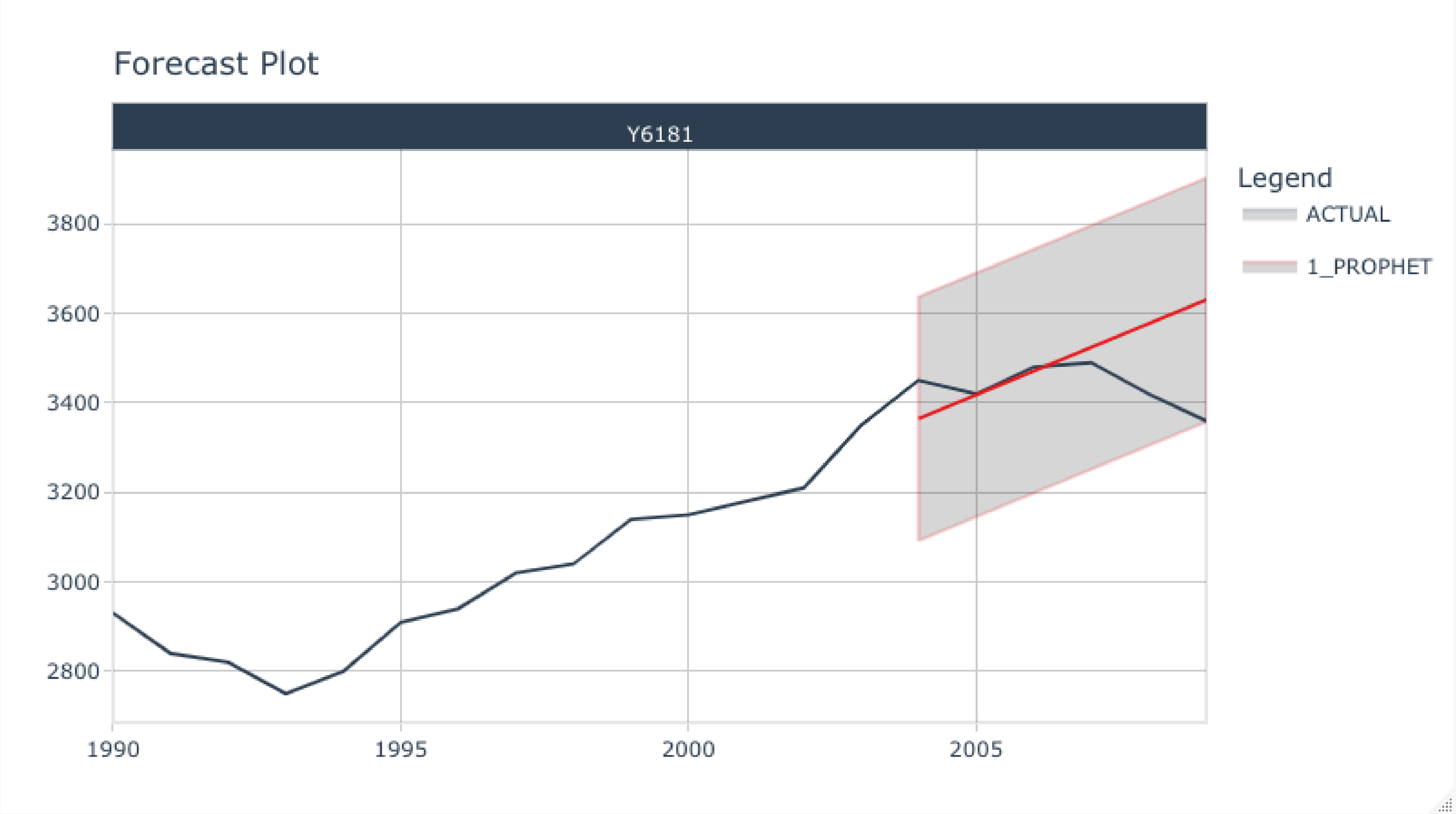


RMSE: 27.55
SMAPE: 0.40

Yearly series

Search

id	.model_id	.model_desc	.type	mae	mape	mase	smape	rmse	rsq
Y10216 (1)									
	1	PROPHET	Test	1493.77	20.91	0.5	18.04	1810.8	0.04
Y11735 (1)									
	1	PROPHET	Test	183.5	7.22	1.63	7.51	191.04	0.88
Y12596 (1)									
	1	PROPHET	Test	784.44	11.81	1.46	12.85	959.78	0.99
Y12733 (1)									
	2	XGBOOST	Test	550.15	8.22	3.48	8.6	569.82	
Y13764 (1)									
	1	PROPHET	Test	155.52	3.45	1.75	3.38	164.82	0.99
Y16231 (1)									
	1	PROPHET	Test	577.07	5.81	1.53	5.59	651.72	0.94
Y1849 (1)									
	1	PROPHET	Test	478.03	6.86	1.18	7.12	507.17	0.97
Y18950 (1)									



RMSE: 133.28
SMAPE: 2.67

Some comments

- Models show a better performance in the higher frequency series
- Total computation time:
  - Hourly: 1.383359 mins
  - Daily: 2.252042 mins
  - Weekly: 1.644516 mins
  - Monthly: 40.03017 secs
  - Quarterly: 37.62589 secs
  - Yearly: 1.092836 mins
- Some improvements could be done by fitting DL models such as Recurrent Neural Networks.
- In terms of computation time, Global models are less demanding but usually are also less accurate.

Thanks!

Time Series Forecasting 2022

Università degli Studi di Milano

Github repository
/mathicard/DSE-TS-forecasting

Sources
Iterative Forecasting with Nested Ensembles
Time Series Forecasting repository