Testing AutoML packages

July 2018

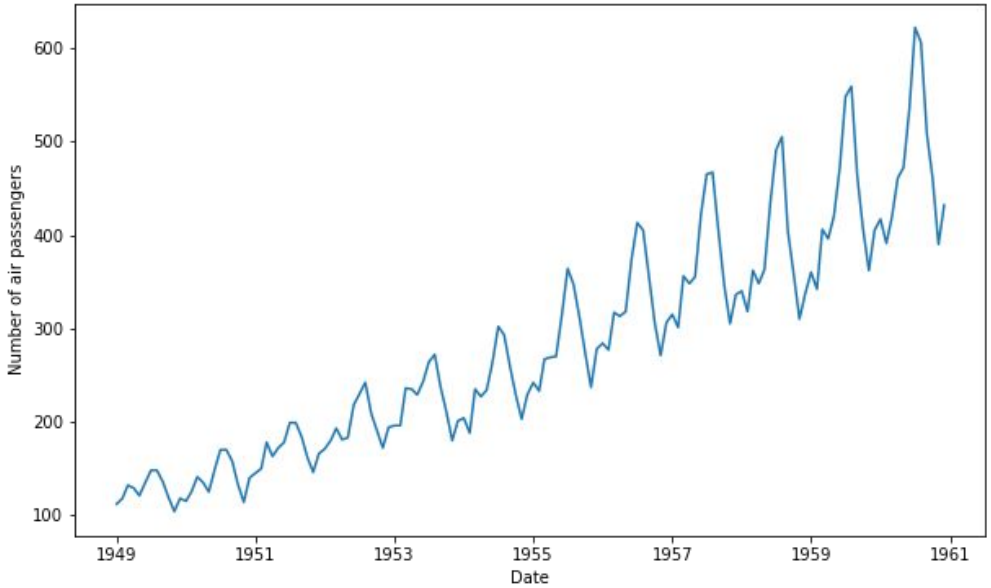
About experiments

1. 4 libraries: Prophet, Pyramid (SARIMAX), PyFlux, PyAF
2. Experiments on Real Airline passengers dataset
3. Experiments on synthetic datasets
    a. Adding lags to dataset
    b. Influence of exogenous features
    c. Different sizes for train data
    d. Adding Redundant features
    e. Adding noise
4. Pros and cons of each library

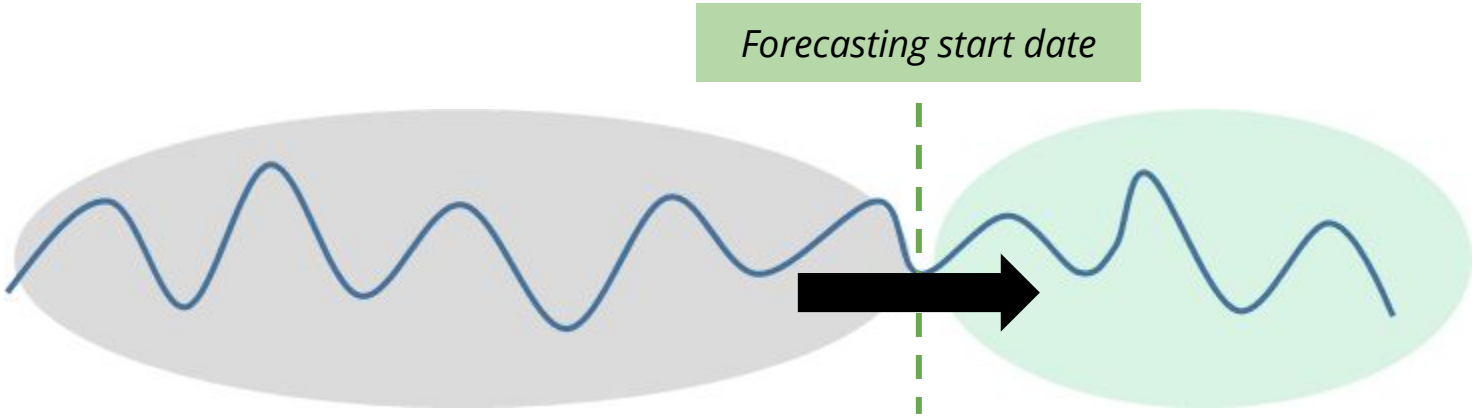
Airline passengers dataset

Air passengers dataset

— 1 time series
— 144 timestamps



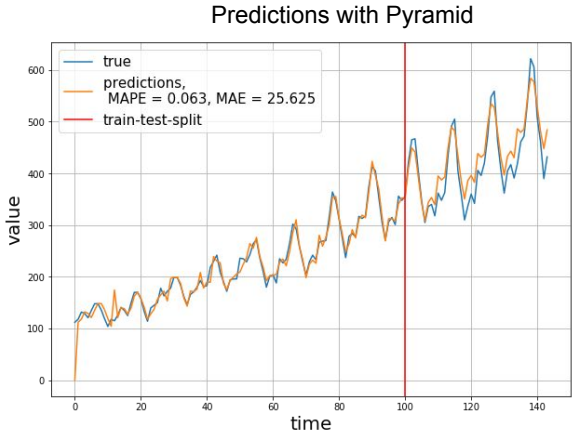
Autoregressive forecasting (without exogenous features)

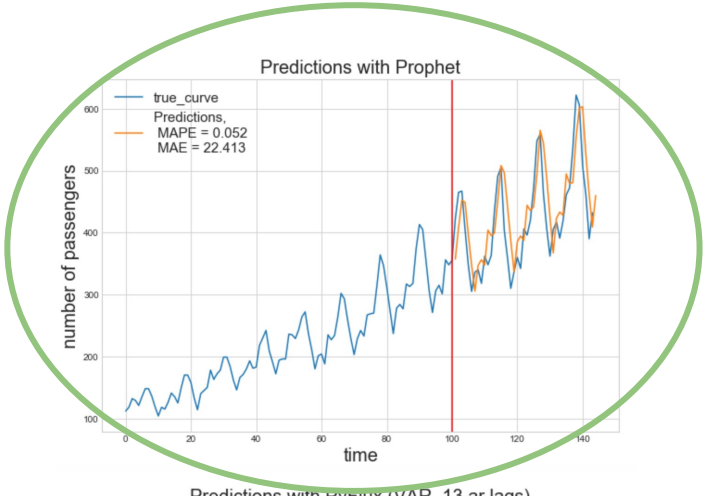


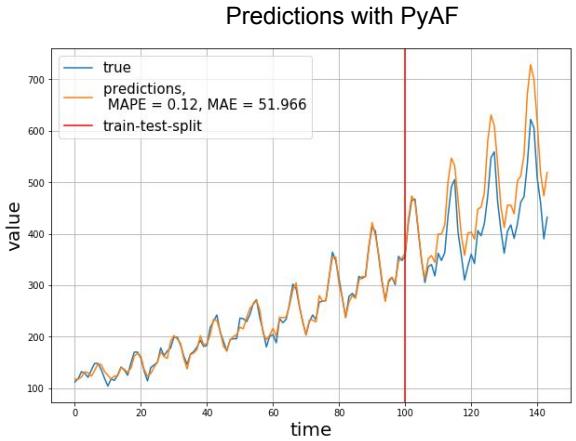
Airline passengers dataset

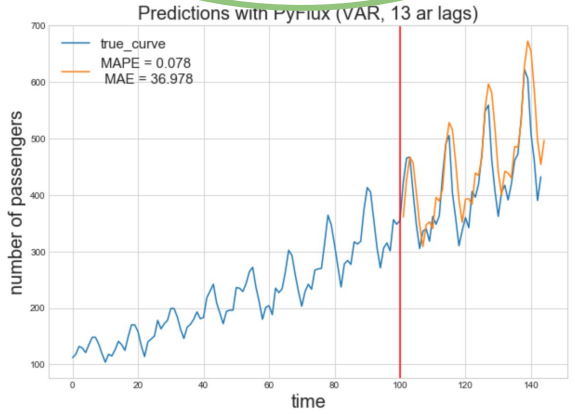
Air passengers dataset characteristics:

— 1 time series
— 144 timestamps









Synthetic datasets

Exogenous data:

$$Exog_t^{(1)} = t$$

$$Exog_t^{(2)} = tsin(t) + 200 + \xi_t$$

$$Exog_t^{(3)} = 100sin(tmodT) + 100 + 2t$$

$$Exog_t^{(4)} = 100cos(tmodT) + 100 + 2t$$

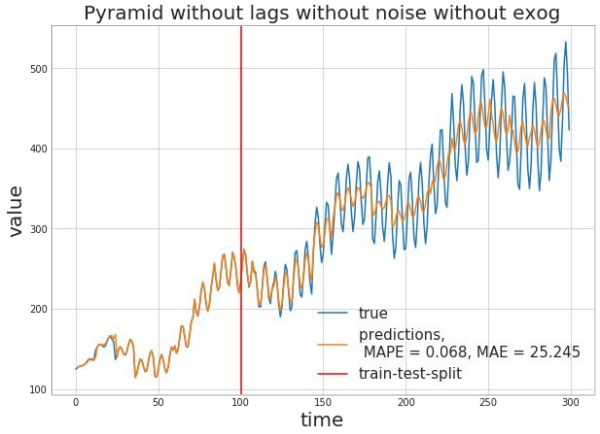
Time series 1:

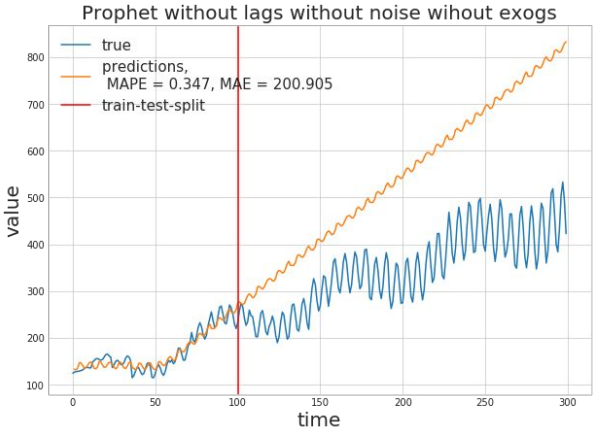
$$y_t = \frac{1}{4}Exog_t^{(1)} + \frac{1}{4}Exog_t^{(2)} + \frac{1}{4}Exog_t^{(3)} + \frac{1}{4}Exog_t^{(4)} + \xi_t$$

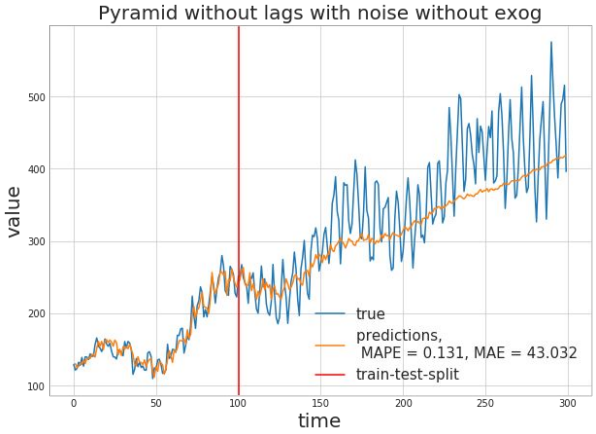
Time series 2:

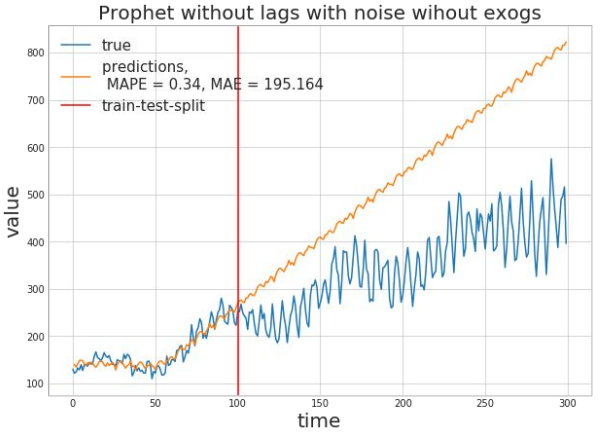
$$y_t = \frac{1}{4}Exog_t^{(1)} + \frac{1}{4}Exog_t^{(2)} + \frac{1}{4}Exog_t^{(3)} + \frac{1}{4}Exog_t^{(4)} + 0.3y_{t-1} + 0.2y_{t-2} + 0.1y_{t-3} + \xi_t$$

1-st dataset **without exogenous**

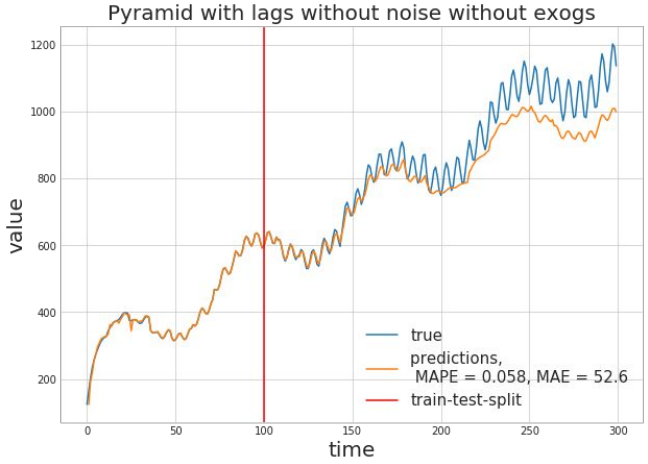


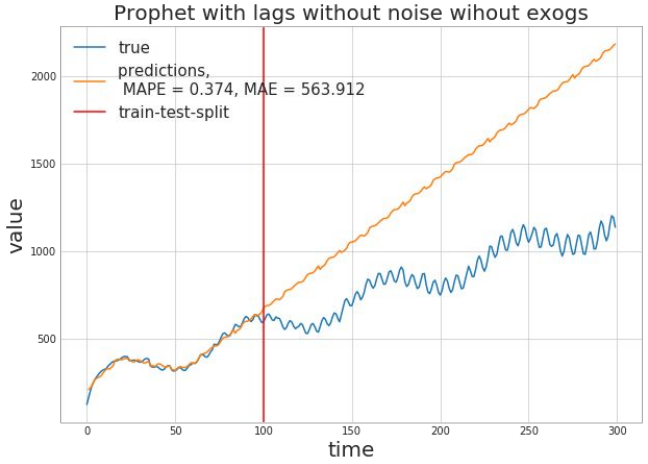


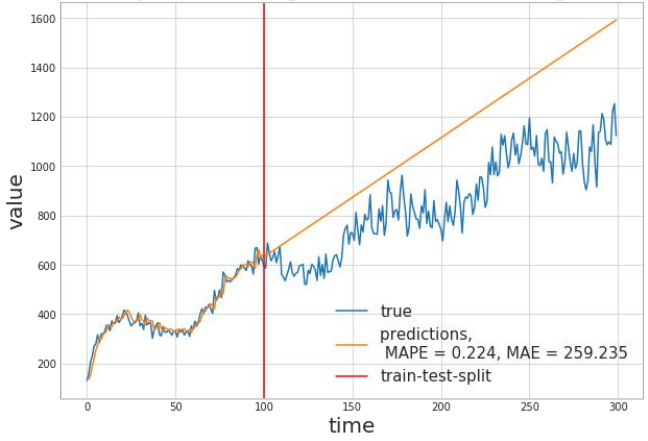


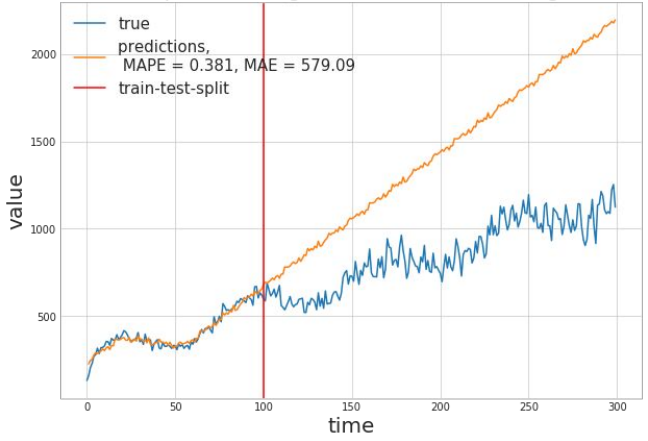


2nd dataset **without exogenous**

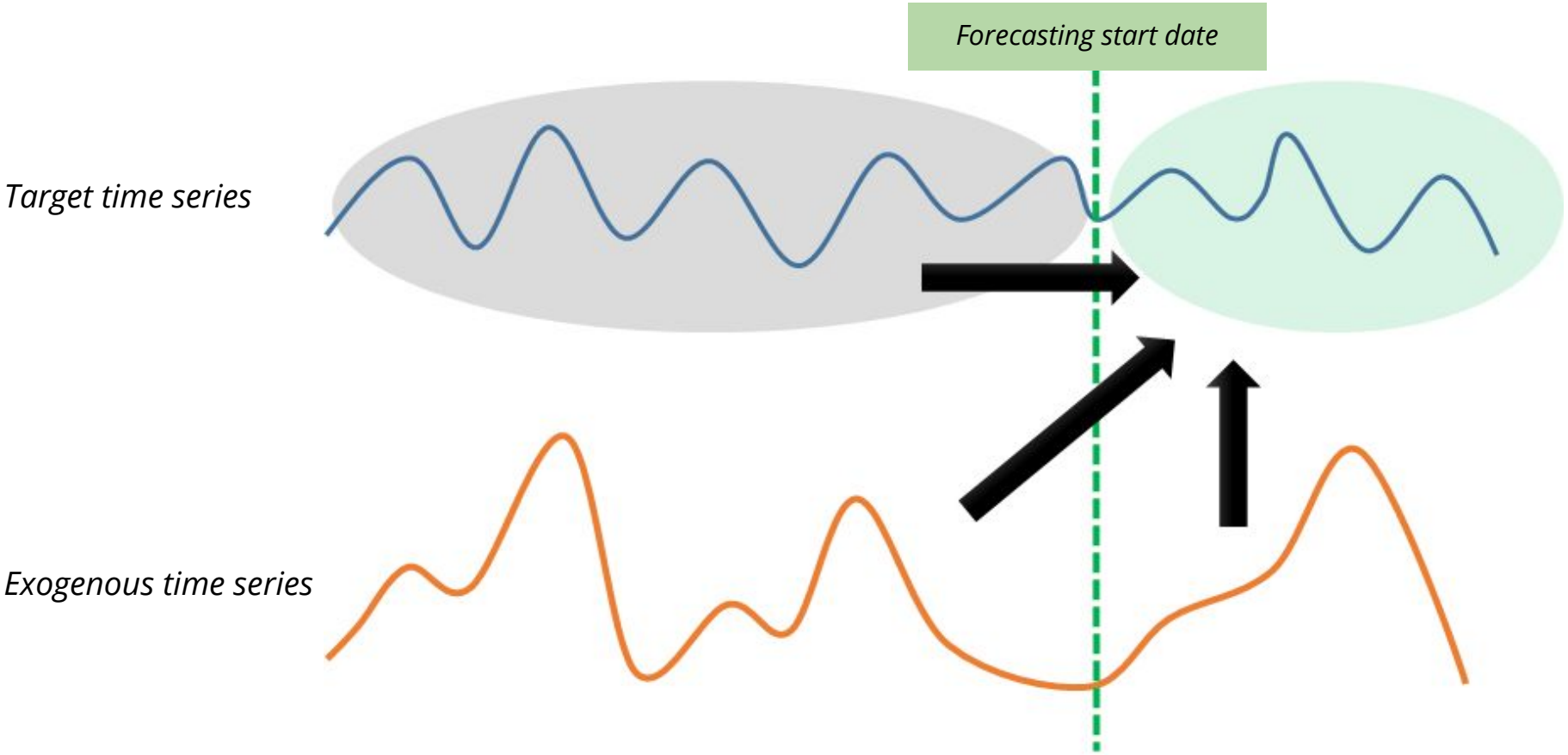




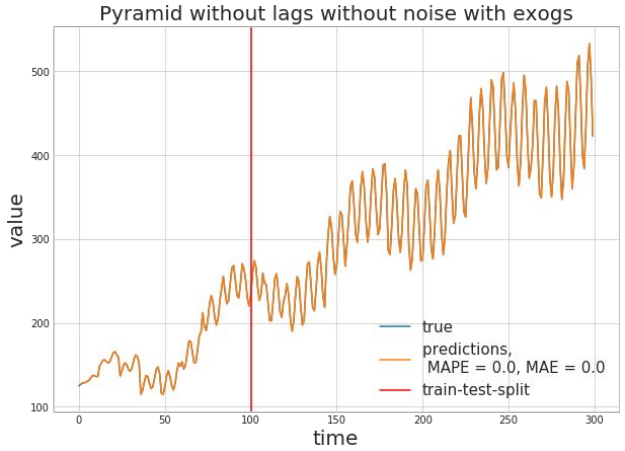


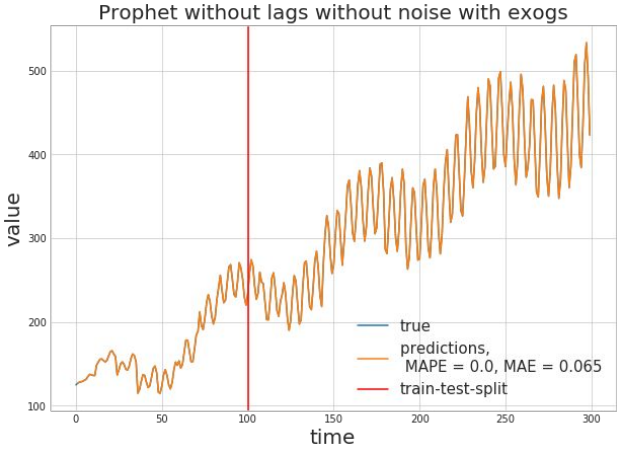


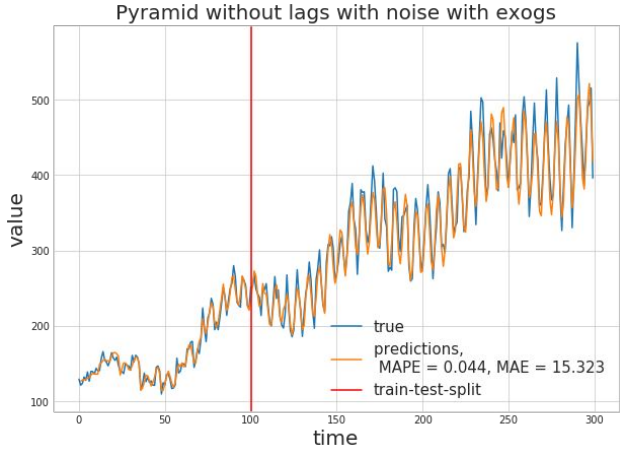
Forecasting using autoregressive and **exogenous features**

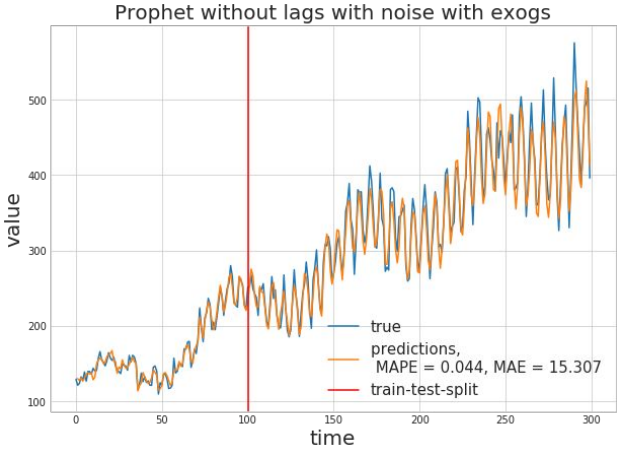


1-st dataset **with exogenous**

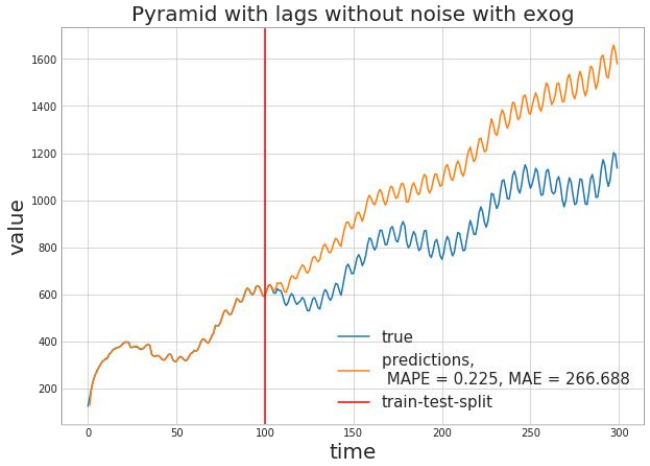


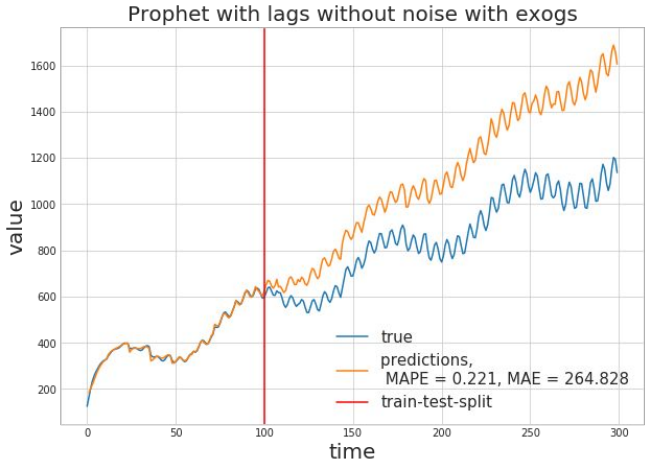


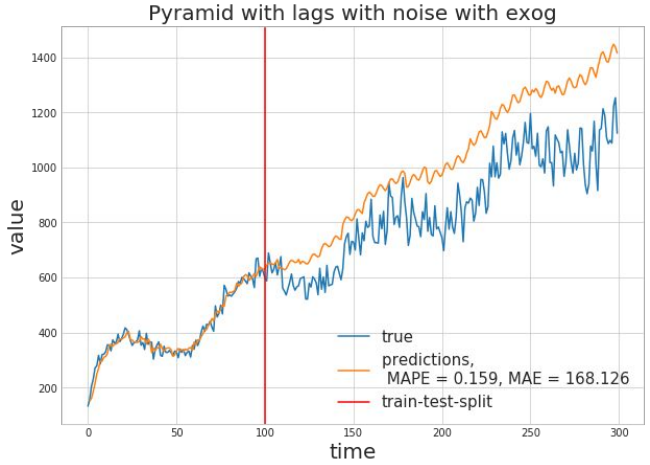


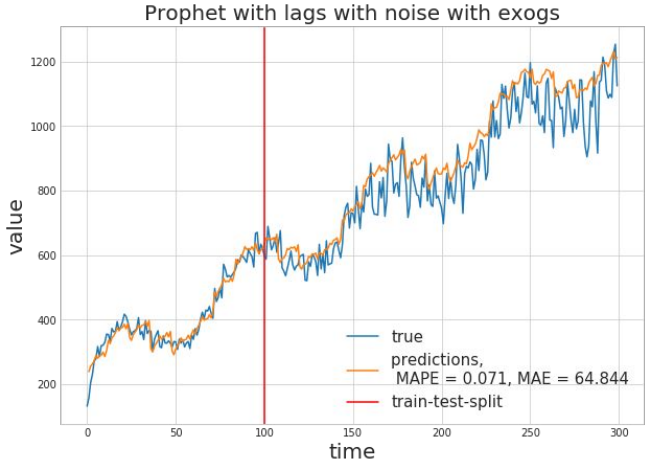


2nd dataset **with exogenous**







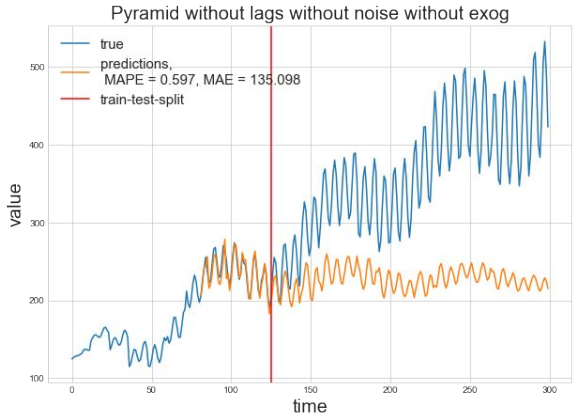


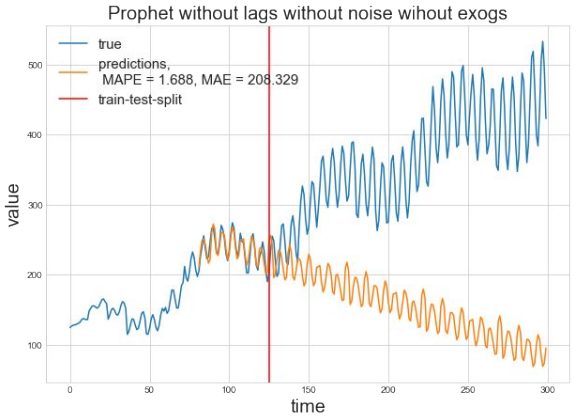
Forecasting on **small data**

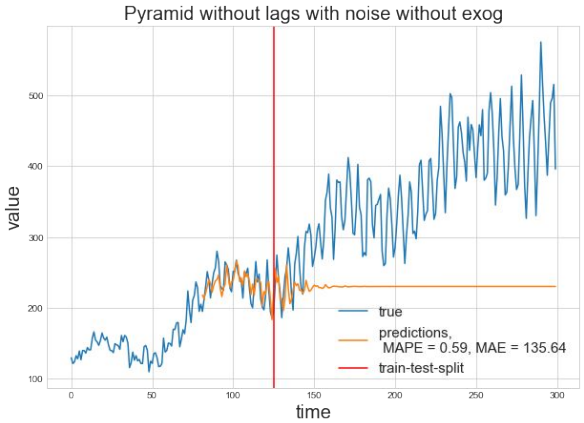
Influence of training size on Prophet and Sarimax

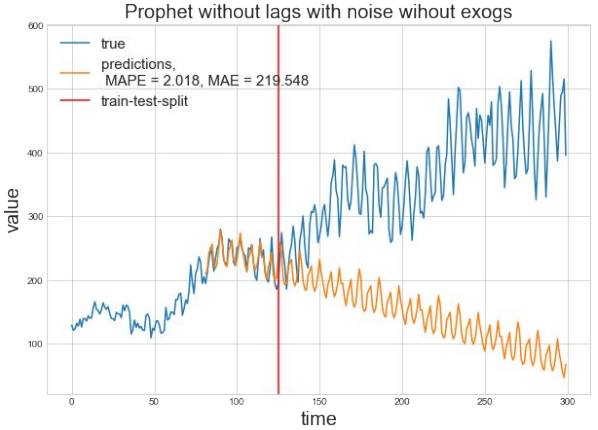
1. Start point = 80
2. Middle point = 125
3. End point = 175

1st dataset without exogenous on small data

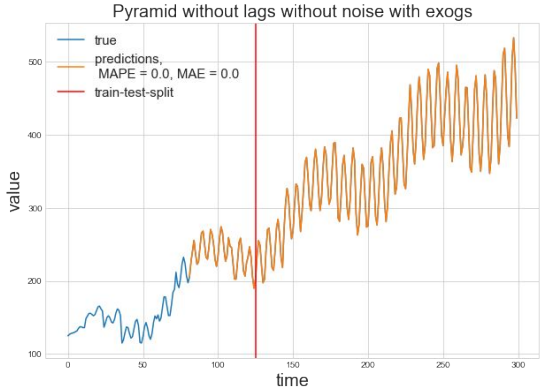


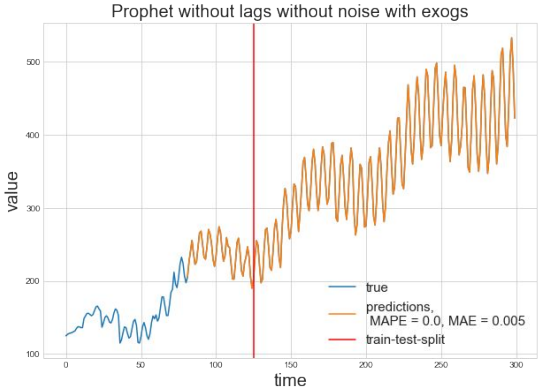


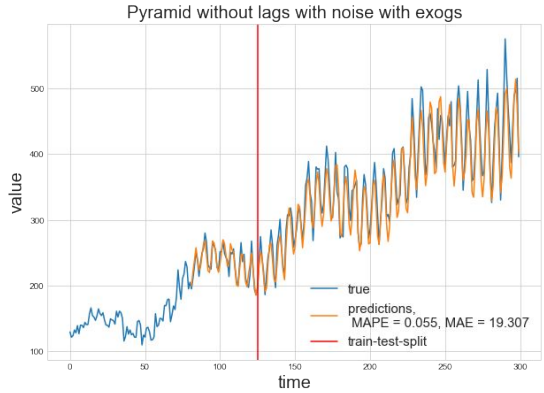


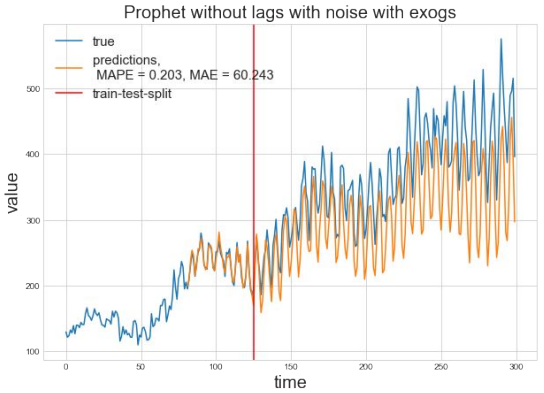


1st dataset with exogenous on small data

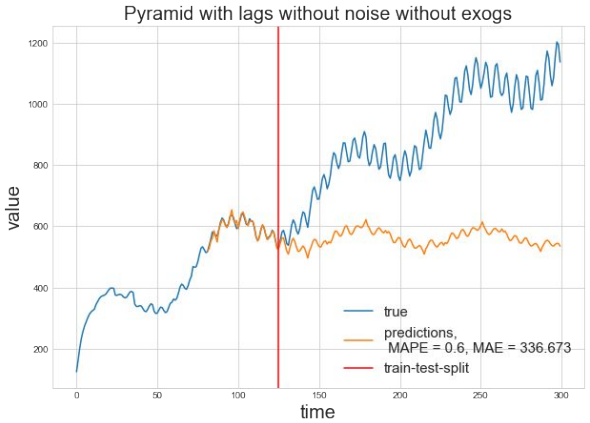


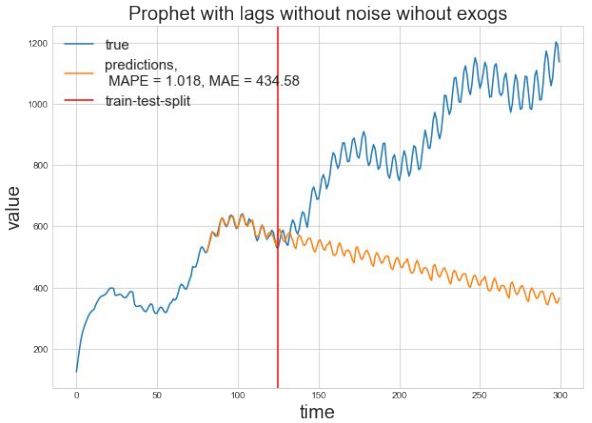


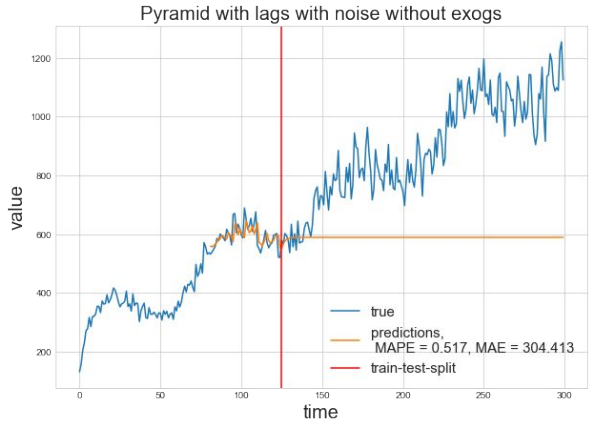


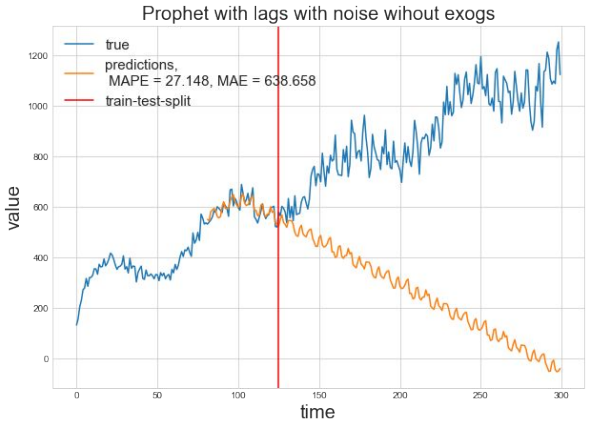


2nd dataset without exogenous on small data

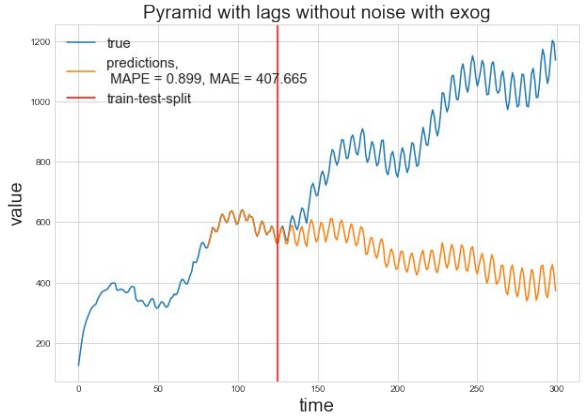


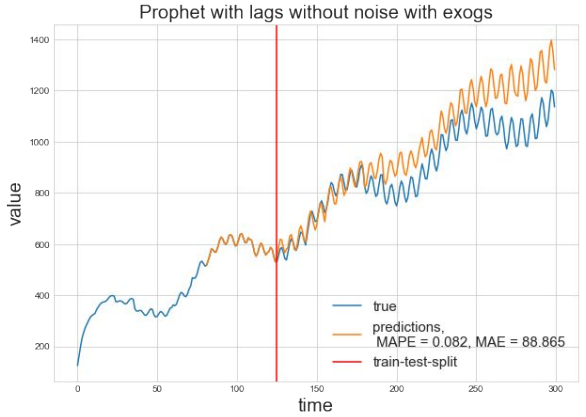


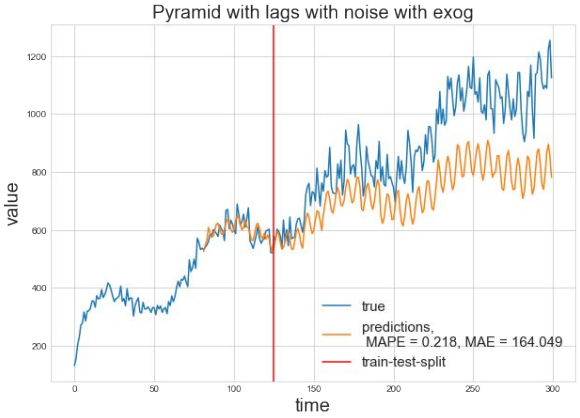


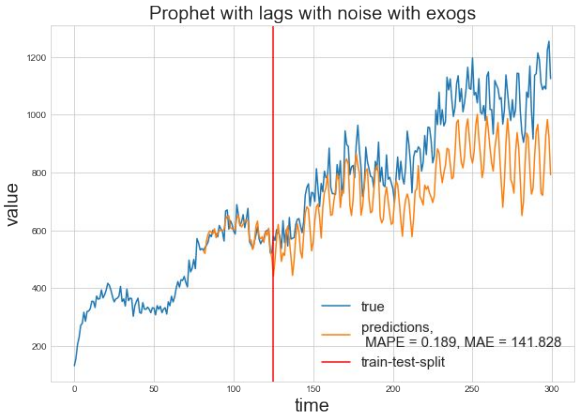


2nd dataset with exogenous on small data







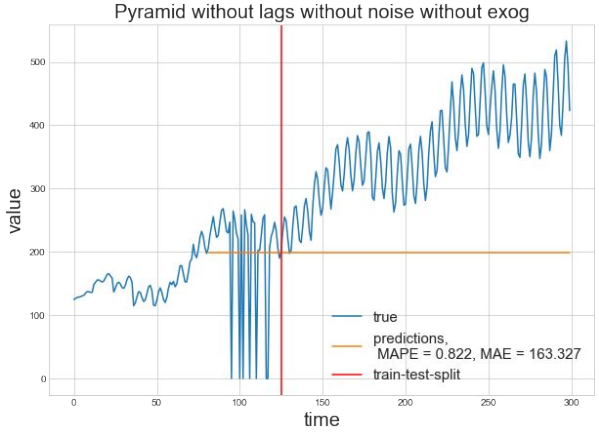


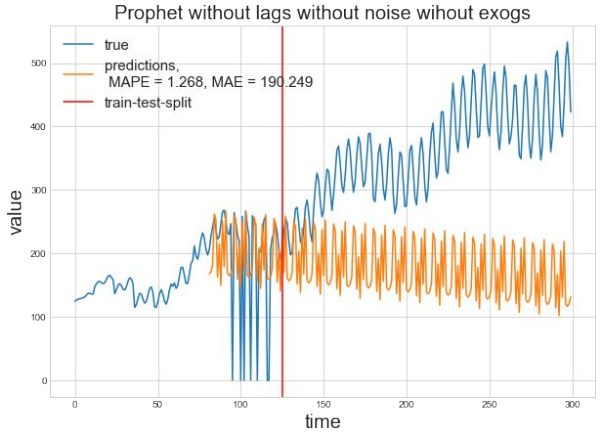
Forecasting on **data with outliers**

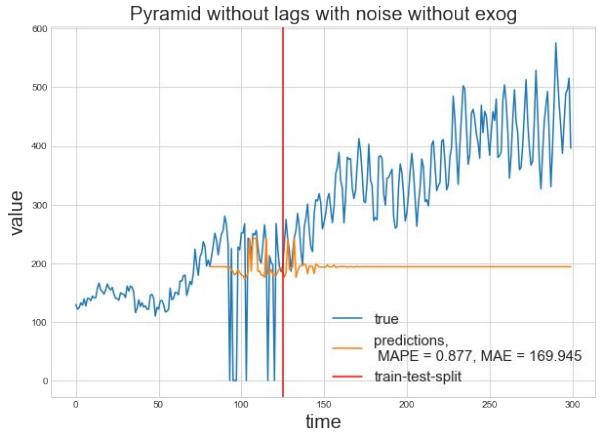
Influence of outliers size on Prophet and Sarimax

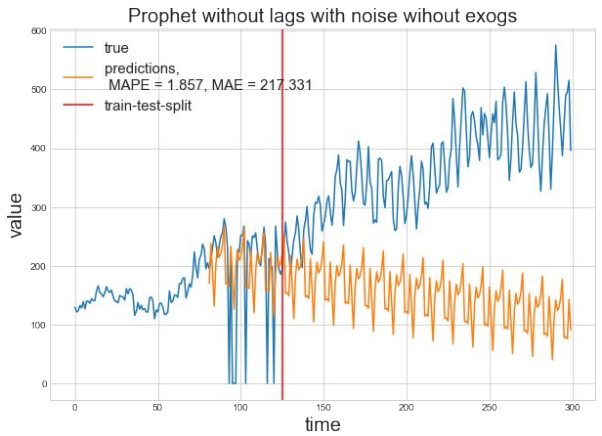
1. Start point = 80
2. Middle point = 125
3. End point = 175
4. 7 zero points

1-st dataset without exogenous with outliers

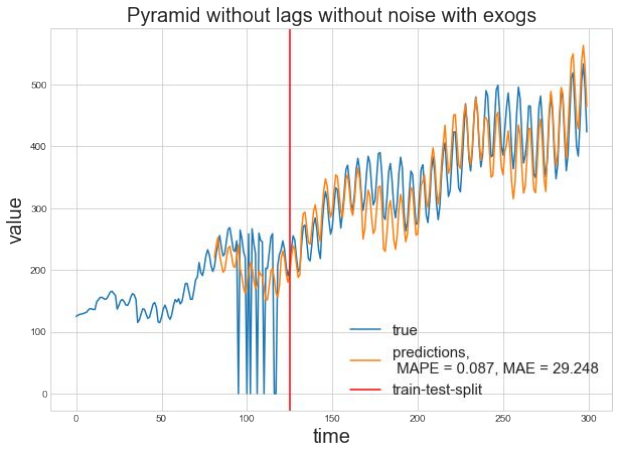


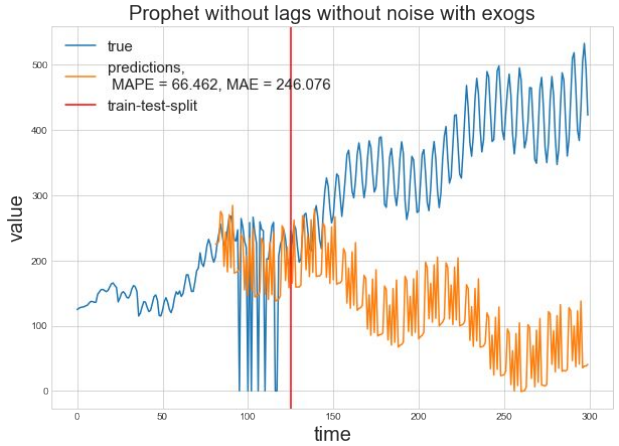


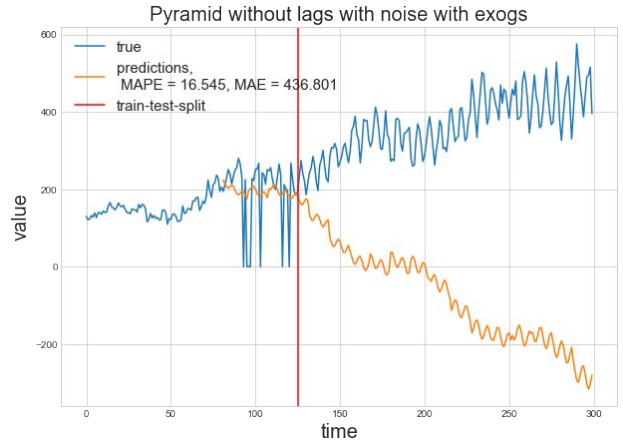


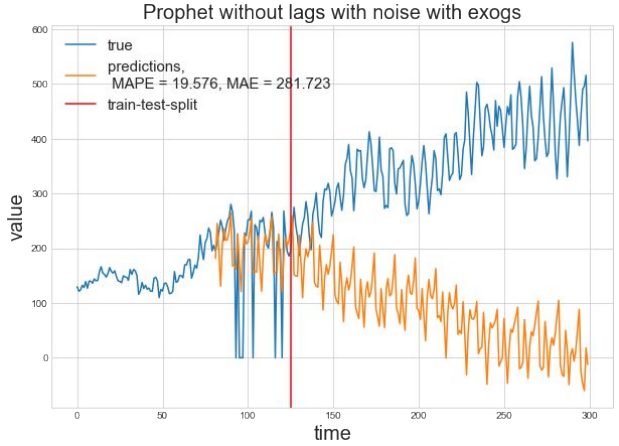


1-st dataset with exogenous with outliers

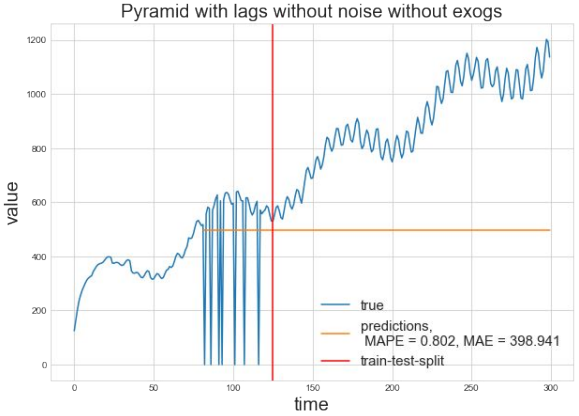


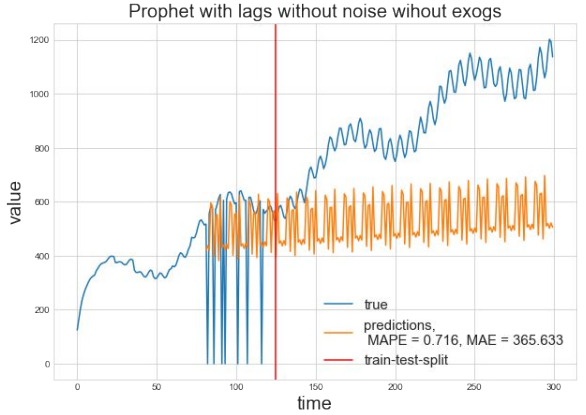


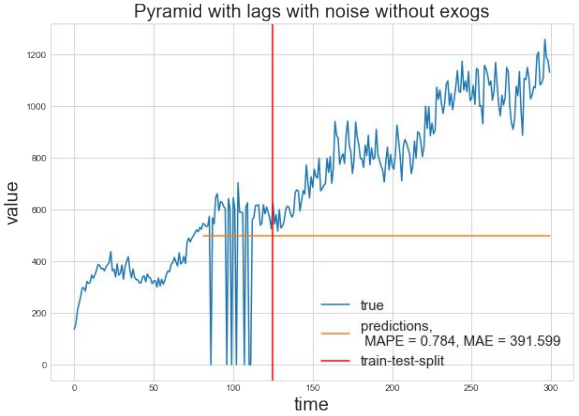


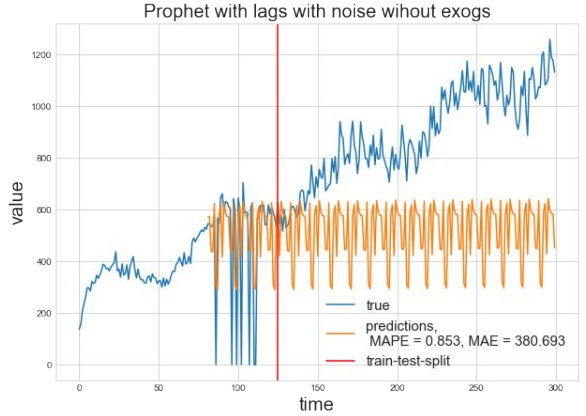


2-nd dataset without exogenous with outliers

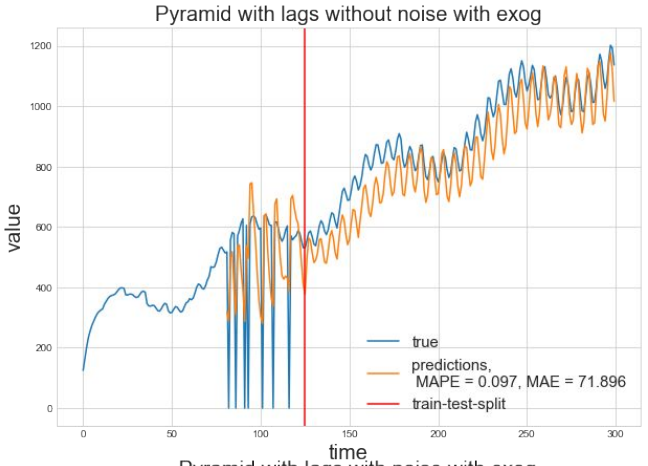


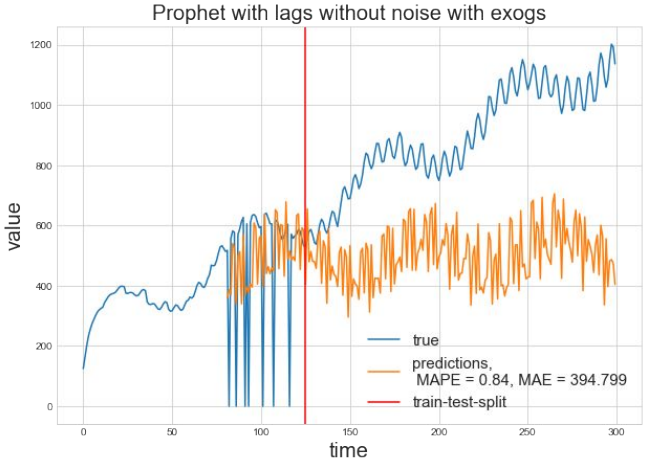


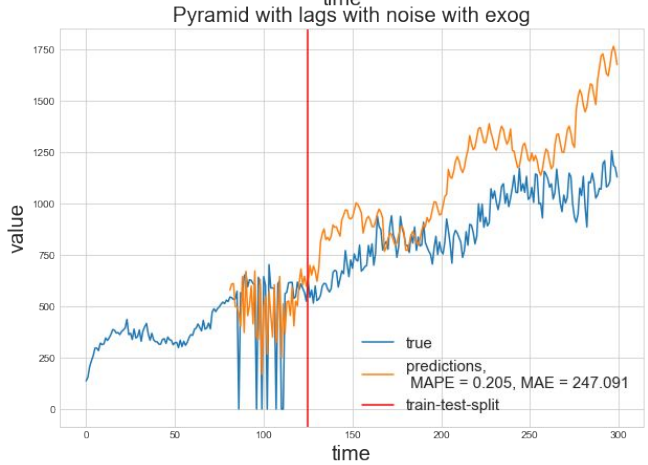


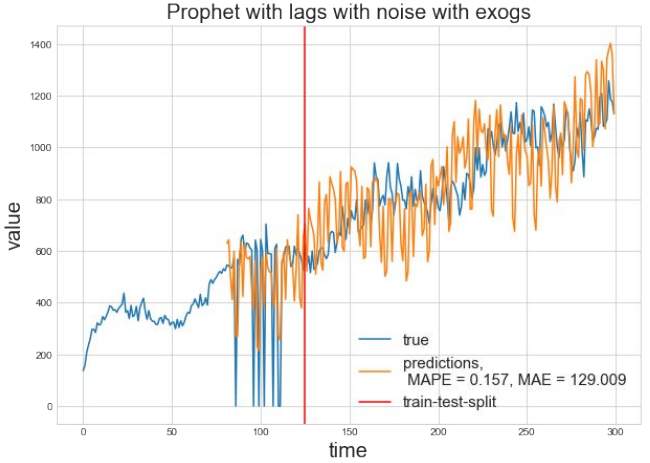


2-nd dataset with exogenous with outliers







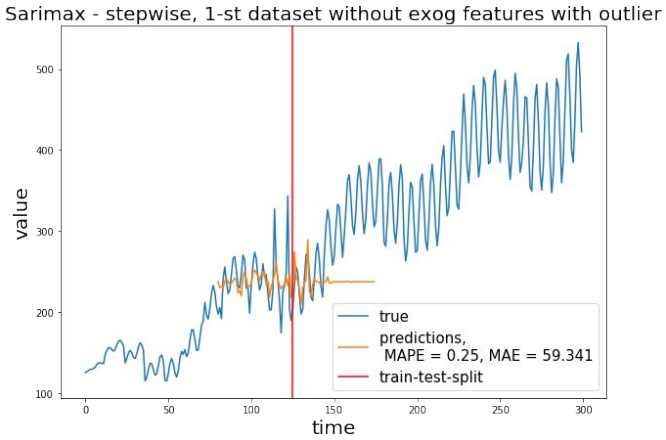


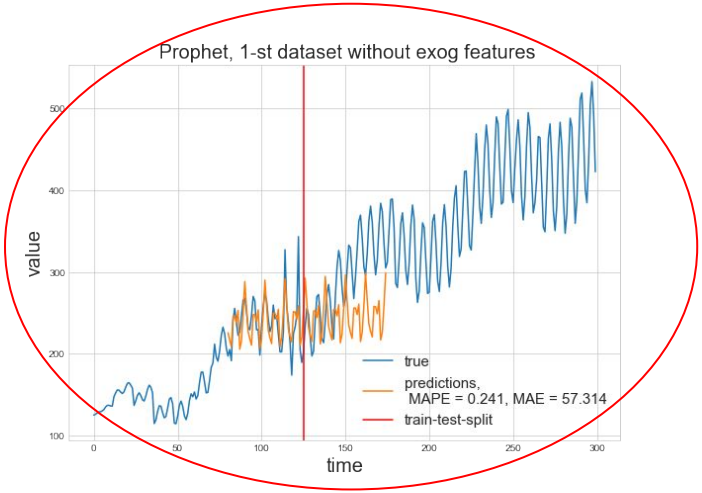
Influence of outliers on Prophet and Sarimax (not so big outliers)

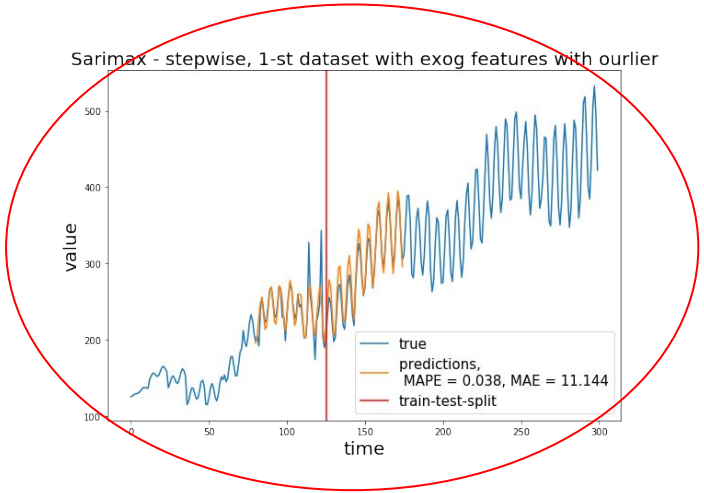
1. Start point = 80
2. Middle point = 125
3. End point = 175
4. Generate randomly outliers

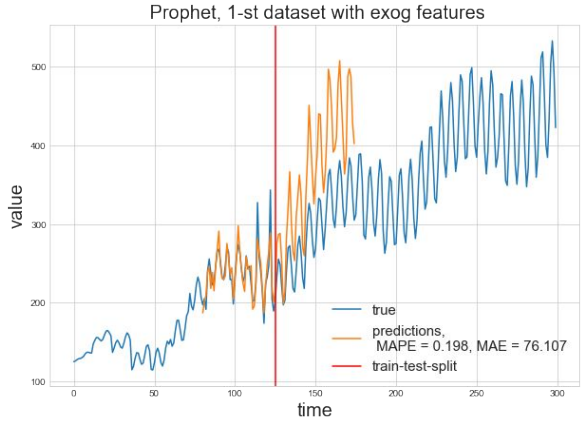
```
np.random.seed(123)

number_of_outliers = 7

idx_1 = np.random.randint(low = start_point, high = middle_point, size = number_of_outliers)
idx_2 = np.random.randint(low = start_point, high = middle_point, size = number_of_outliers)

y_1[idx_1] = y_1[idx_1] + np.random.normal(scale = 50, size = number_of_outliers)
y_2[idx_2] = y_2[idx_2] + np.random.normal(scale = 50, size = number_of_outliers)
```

1-st dataset

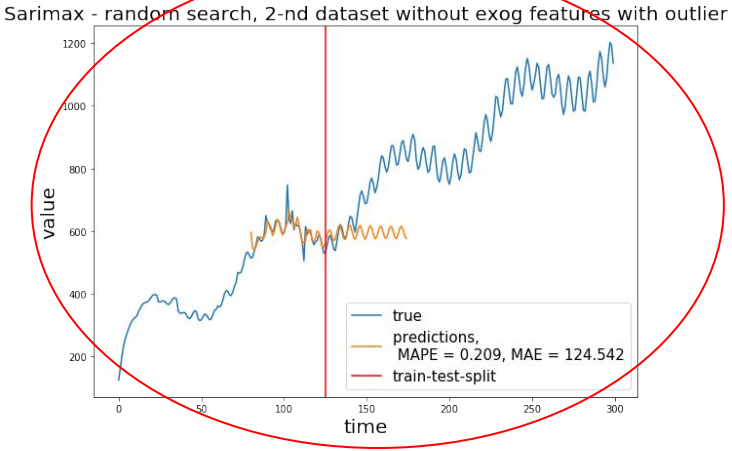

Sarimax - stepwise, 1-st dataset without exog features with outlier
value
time
true
predictions,
MAPE = 0.25, MAE = 59.341
train-test-split

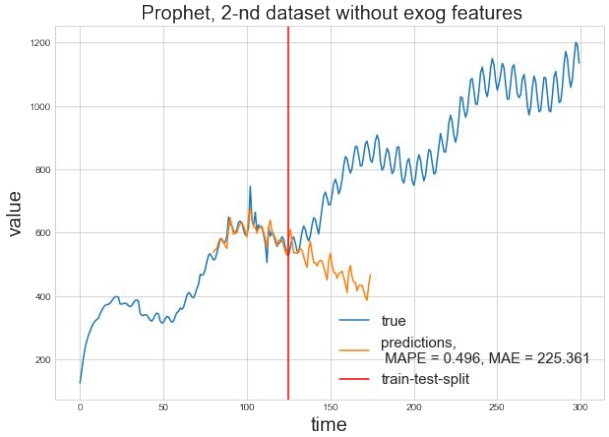

Prophet, 1-st dataset without exog features
value
time
true
predictions,
MAPE = 0.241, MAE = 57.314
train-test-split

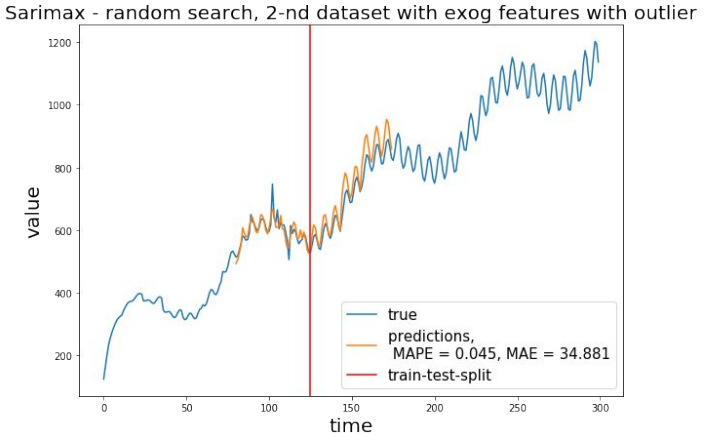

Sarimax - stepwise, 1-st dataset with exog features with ourlier
value
time
true
predictions,
MAPE = 0.038, MAE = 11.144
train-test-split

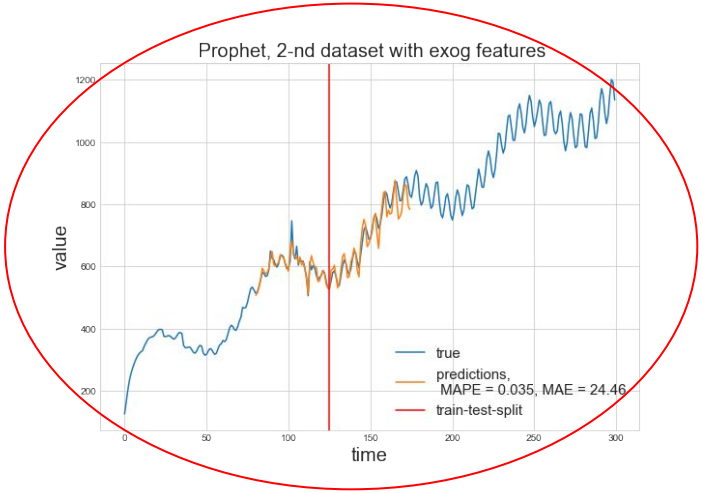

Prophet, 1-st dataset with exog features
value
time
true
predictions,
MAPE = 0.198, MAE = 76.107
train-test-split

2-nd dataset


Sarimax - random search, 2-nd dataset without exog features with outlier
value
time
true
predictions,
MAPE = 0.209, MAE = 124.542
train-test-split


Prophet, 2-nd dataset without exog features
value
time
true
predictions,
MAPE = 0.496, MAE = 225.361
train-test-split


Sarimax - random search, 2-nd dataset with exog features with outlier
value
time
true
predictions,
MAPE = 0.045, MAE = 34.881
train-test-split


Prophet, 2-nd dataset with exog features
value
time
true
predictions,
MAPE = 0.035, MAE = 24.46
train-test-split

Prophet and Sarimax with nonlinear dependency

Target Time Series:

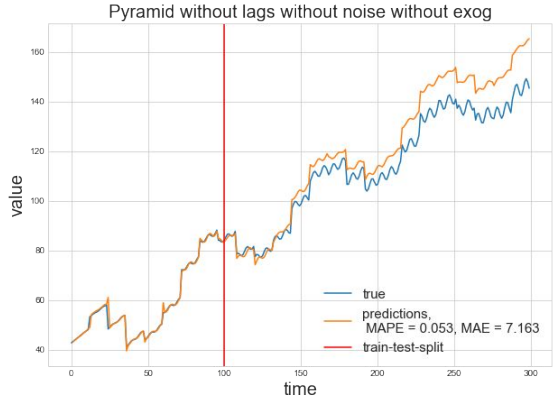
1) $$\frac{3}{2}(f_3 f_4)^{1/3} + \frac{1}{100} f_2 + \frac{1}{10} f_1$$

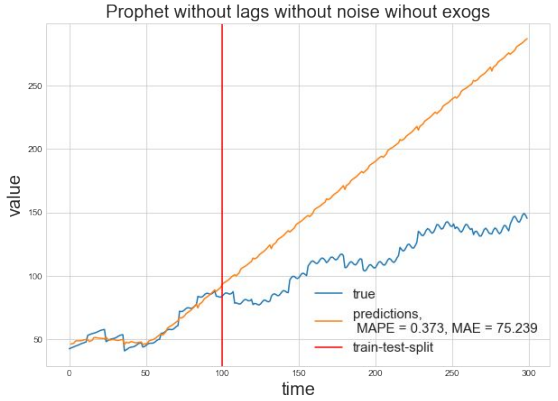
2) $$\frac{3}{2}(f_3 f_4)^{1/3} + \frac{1}{100} f_2 + \frac{1}{10} f_1 + 0.2y_{t-1} - 0.3y_{t-2} - 0.3y_{t-3}$$

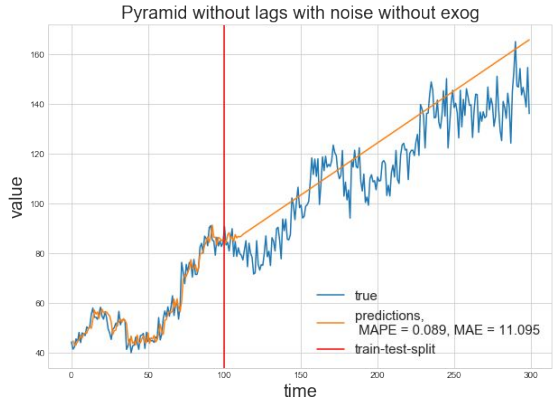
3) $$\frac{3}{2}(f_3 f_4)^{1/3} + \frac{1}{100} f_2 + \frac{1}{10} f_1 + \xi_t$$

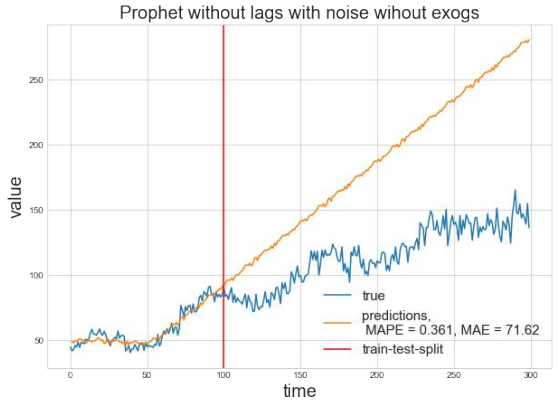
4) $$\frac{3}{2}(f_3 f_4)^{1/3} + \frac{1}{100} f_2 + \frac{1}{10} f_1 + 0.2y_{t-1} - 0.3y_{t-2} - 0.3y_{t-3} + \xi_t$$

1-st dataset without exogenous with nonlinear dependency

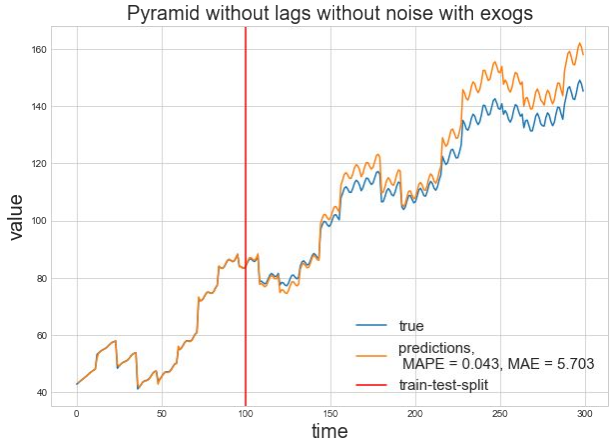


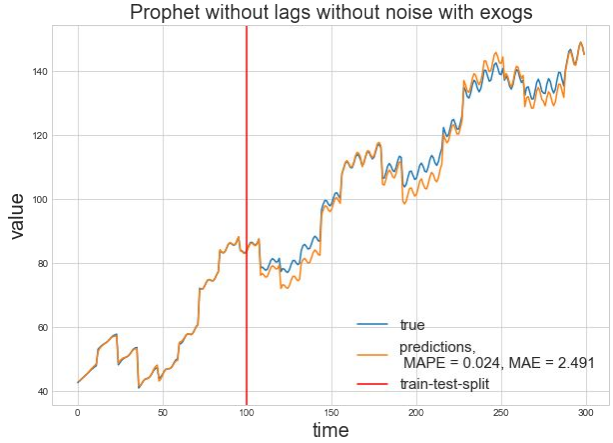


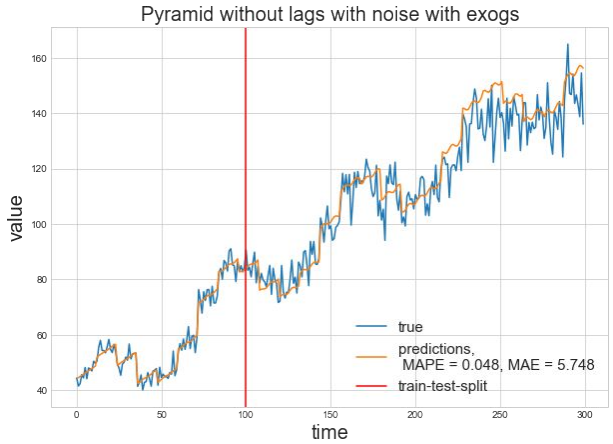


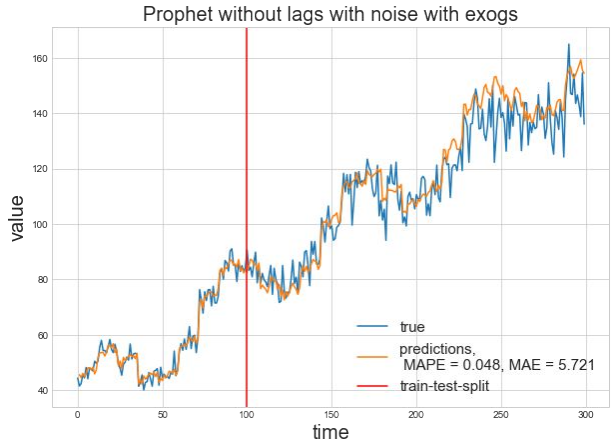


1-st dataset with exogenous with nonlinear dependency

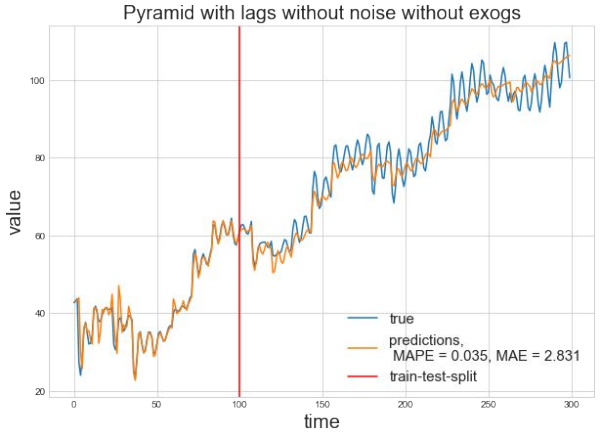


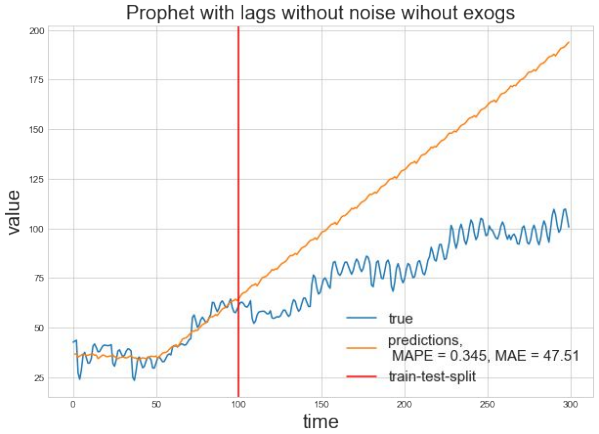


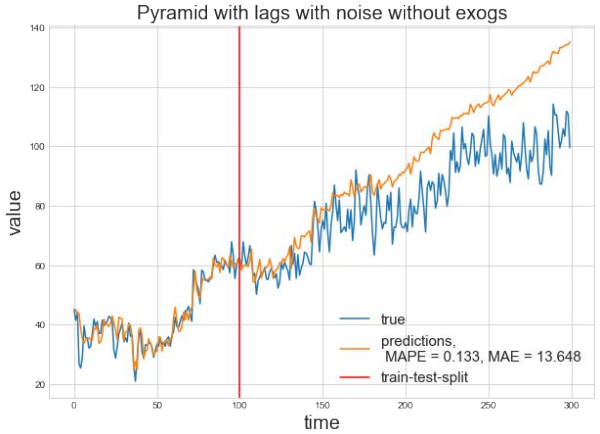


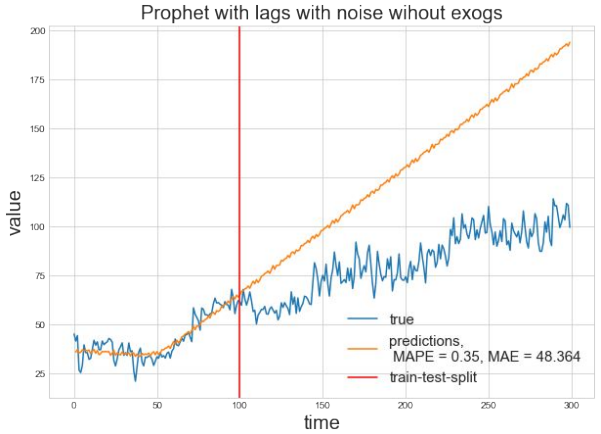


2-nd dataset without exogenous with nonlinear dependency

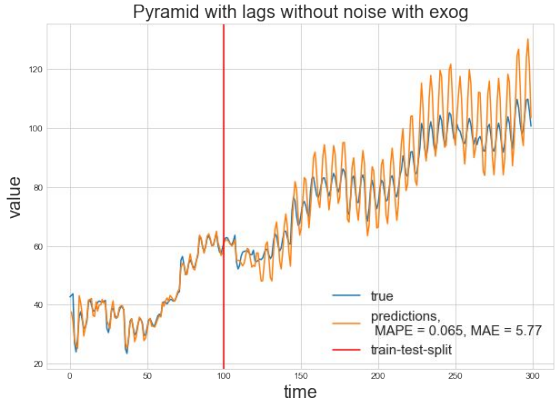


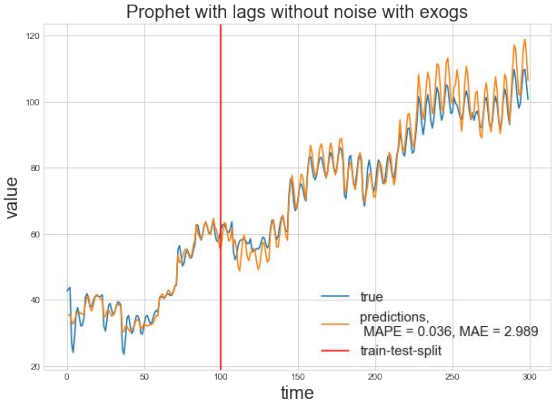


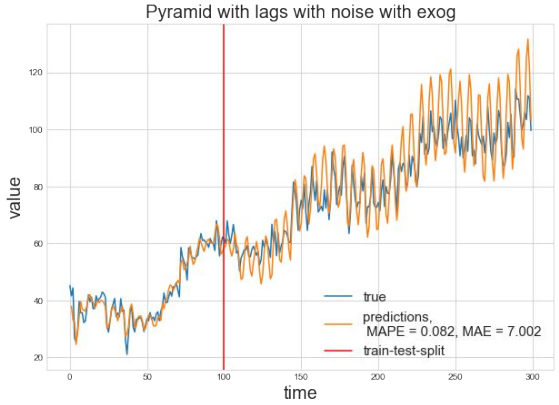


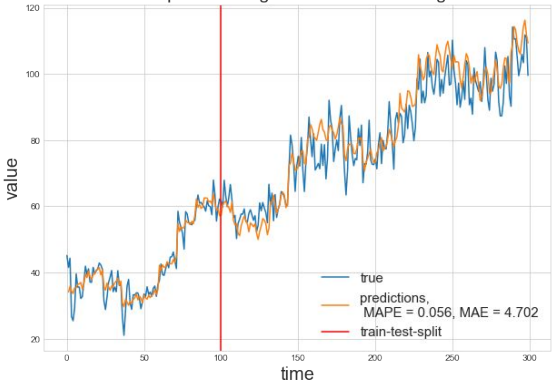


2-nd dataset with exogenous with nonlinear dependency

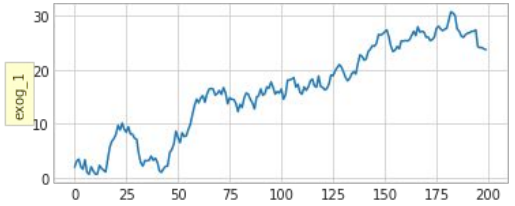






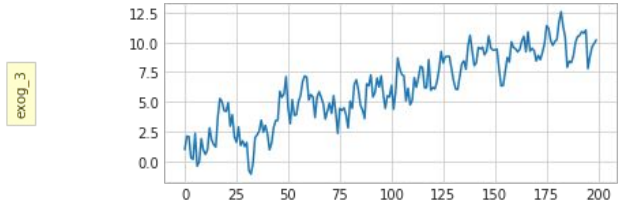


Synthetic test 3



AR (1)

AR (1)



AR (1)

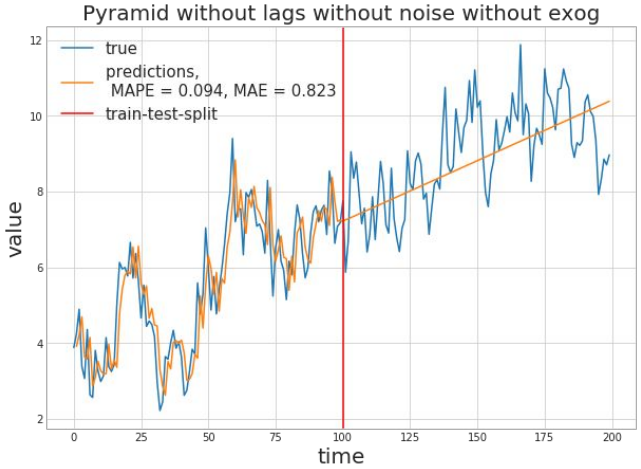
AR (1)

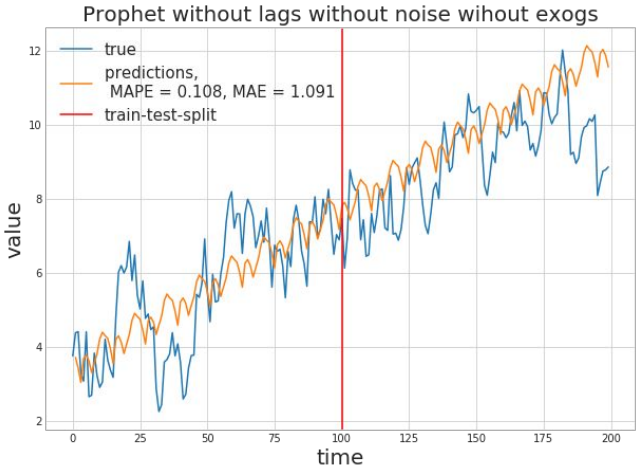
Target Time Series:

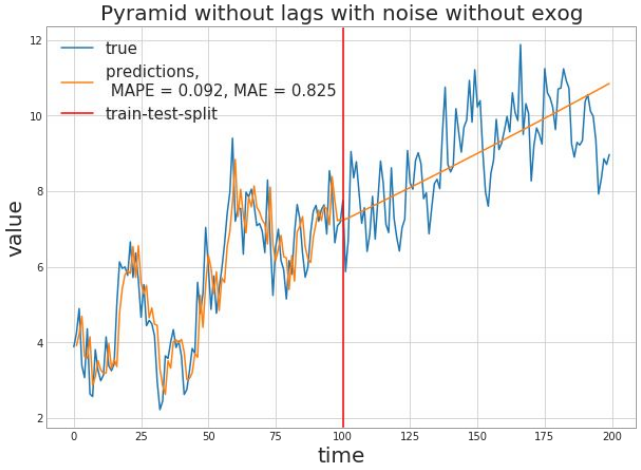
$$y_t = \frac{1}{4}Exog_t^{(1)} + \frac{1}{4}Exog_t^{(2)} + \frac{1}{4}Exog_t^{(3)} + \frac{1}{4}Exog_t^{(4)} + \xi_t$$

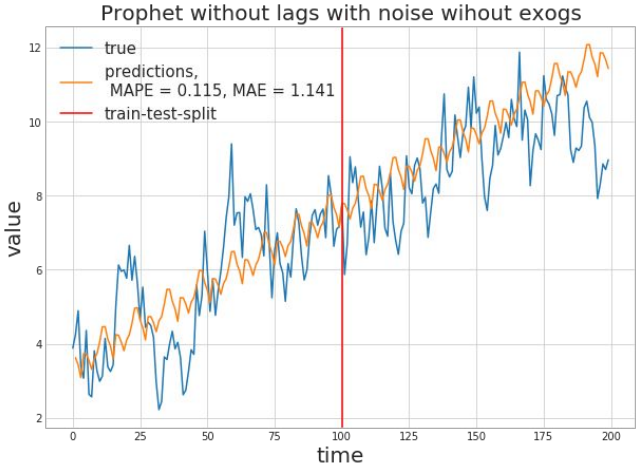
$$y_t = \frac{1}{4}Exog_t^{(1)} + \frac{1}{4}Exog_t^{(2)} + \frac{1}{4}Exog_t^{(3)} + \frac{1}{4}Exog_t^{(4)} + 0.3y_{t-1} + 0.2y_{t-2} + 0.1y_{t-3} + \xi_t$$

Synthetic test 3 (results without exogenous)

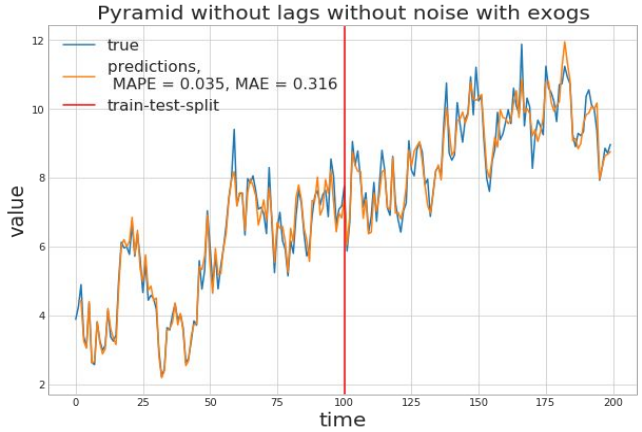


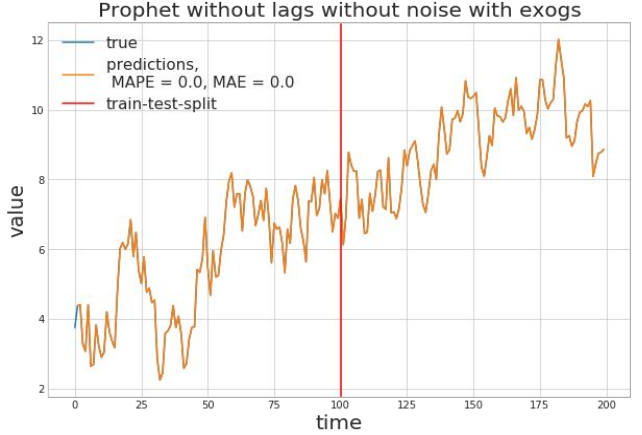


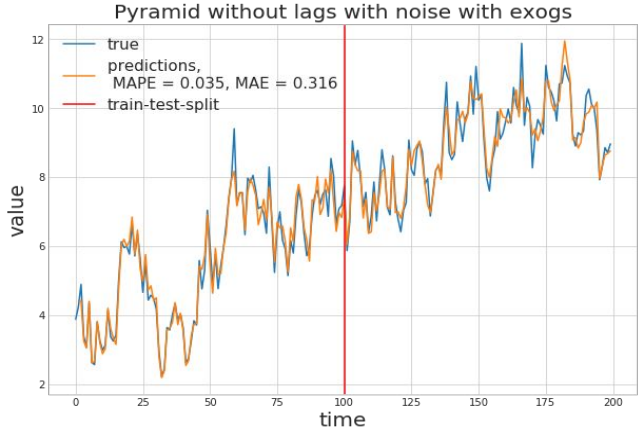


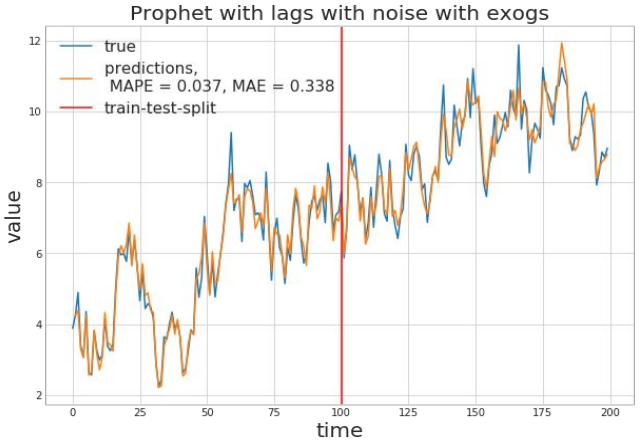


Synthetic test 3 (results with exogenous)

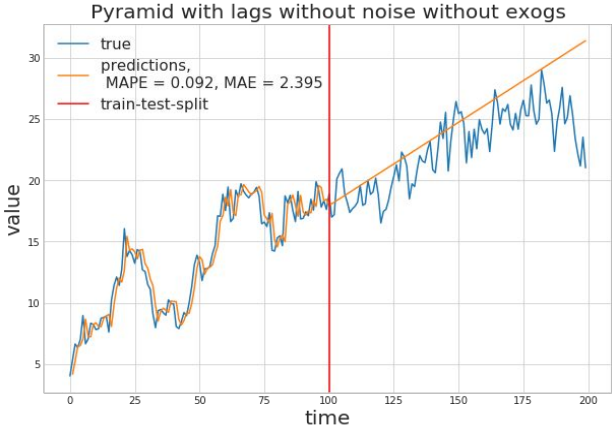


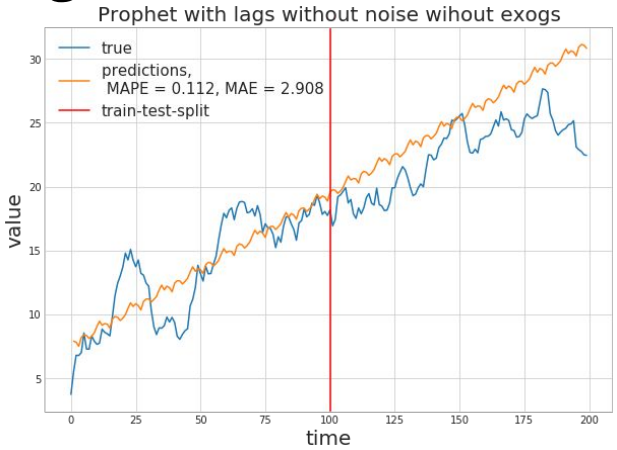


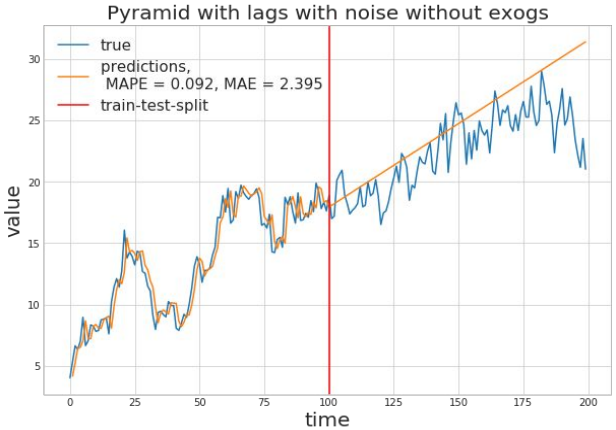


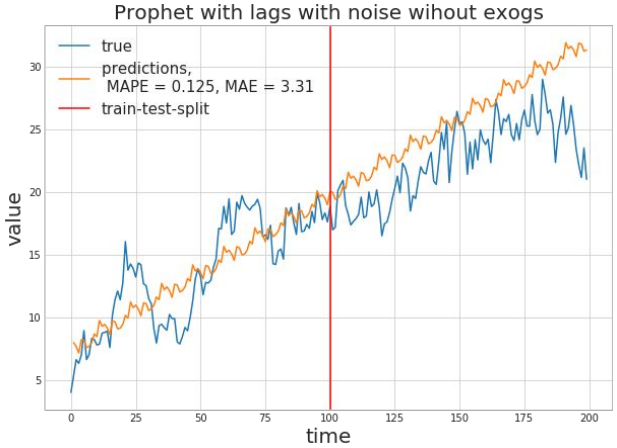


Synthetic test 3 (results without exogenous)

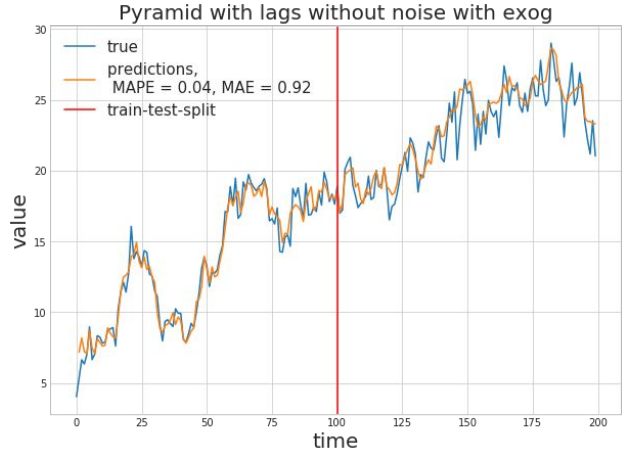


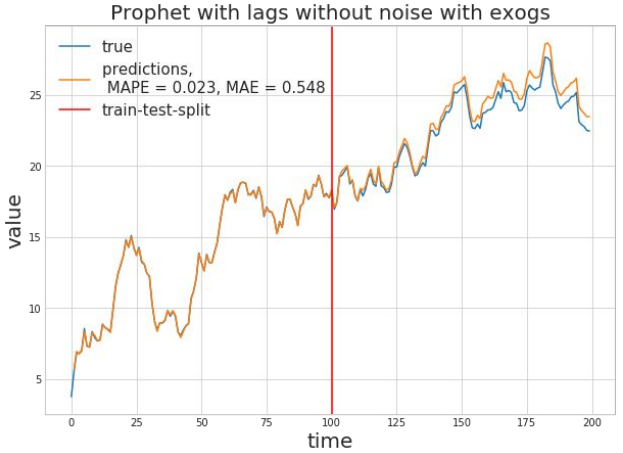


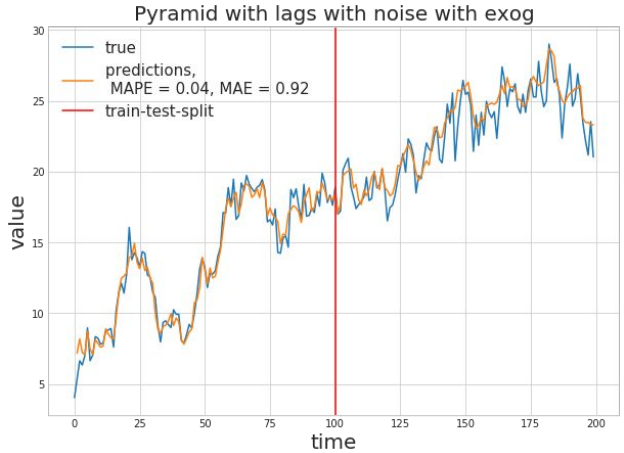


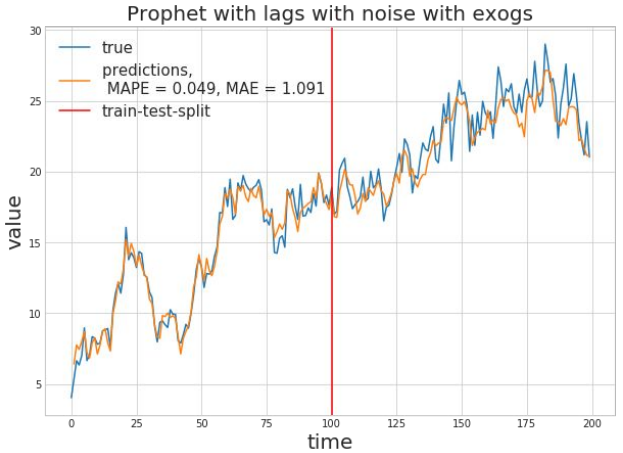


Synthetic test 3 (results with exogenous)

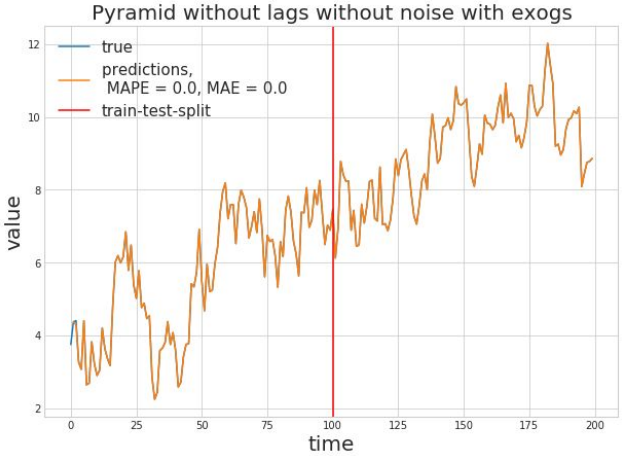


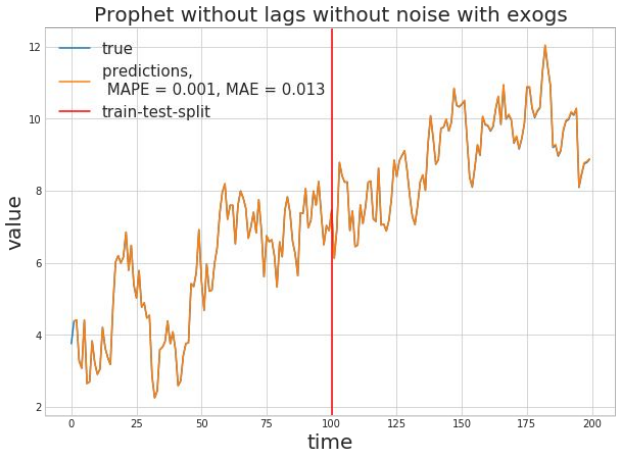


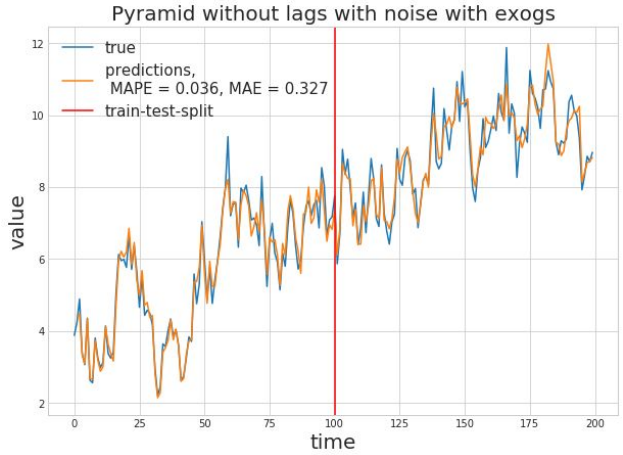


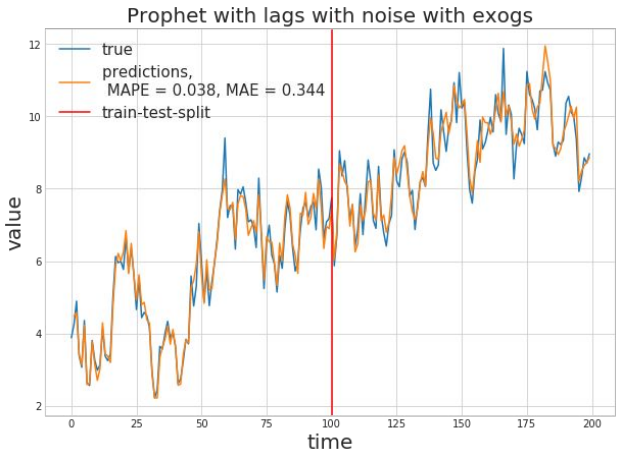


Synthetic test 3 (redundant features)

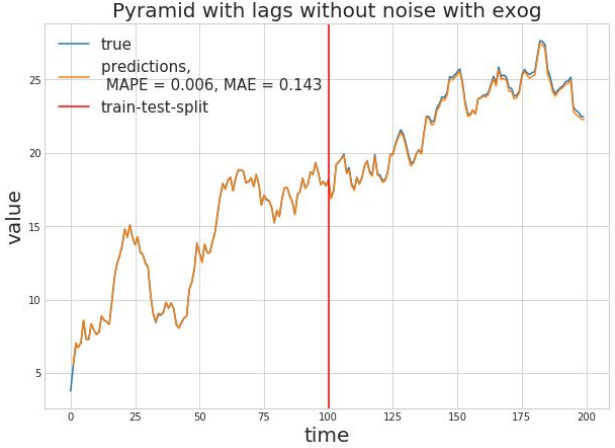


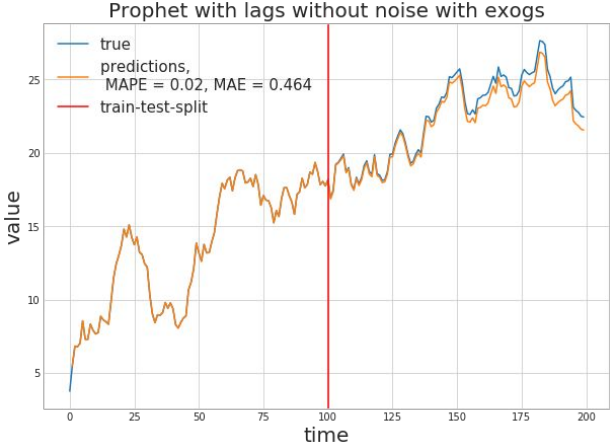


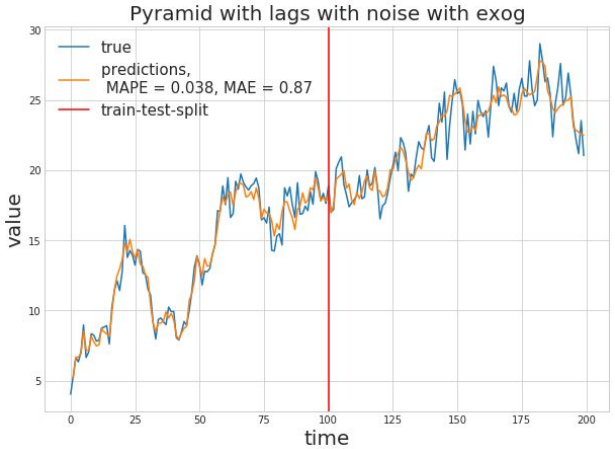


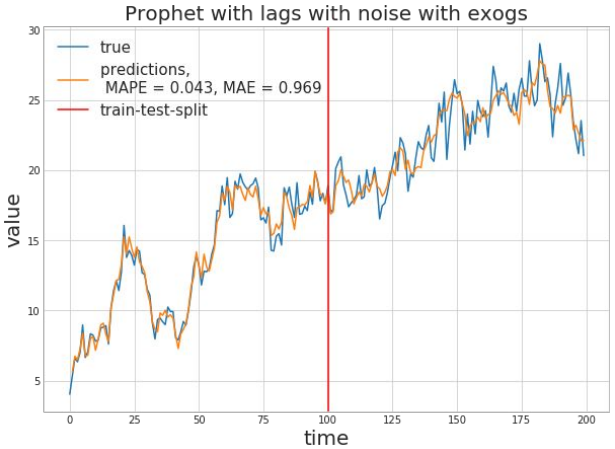


Synthetic test 3 (redundant features)









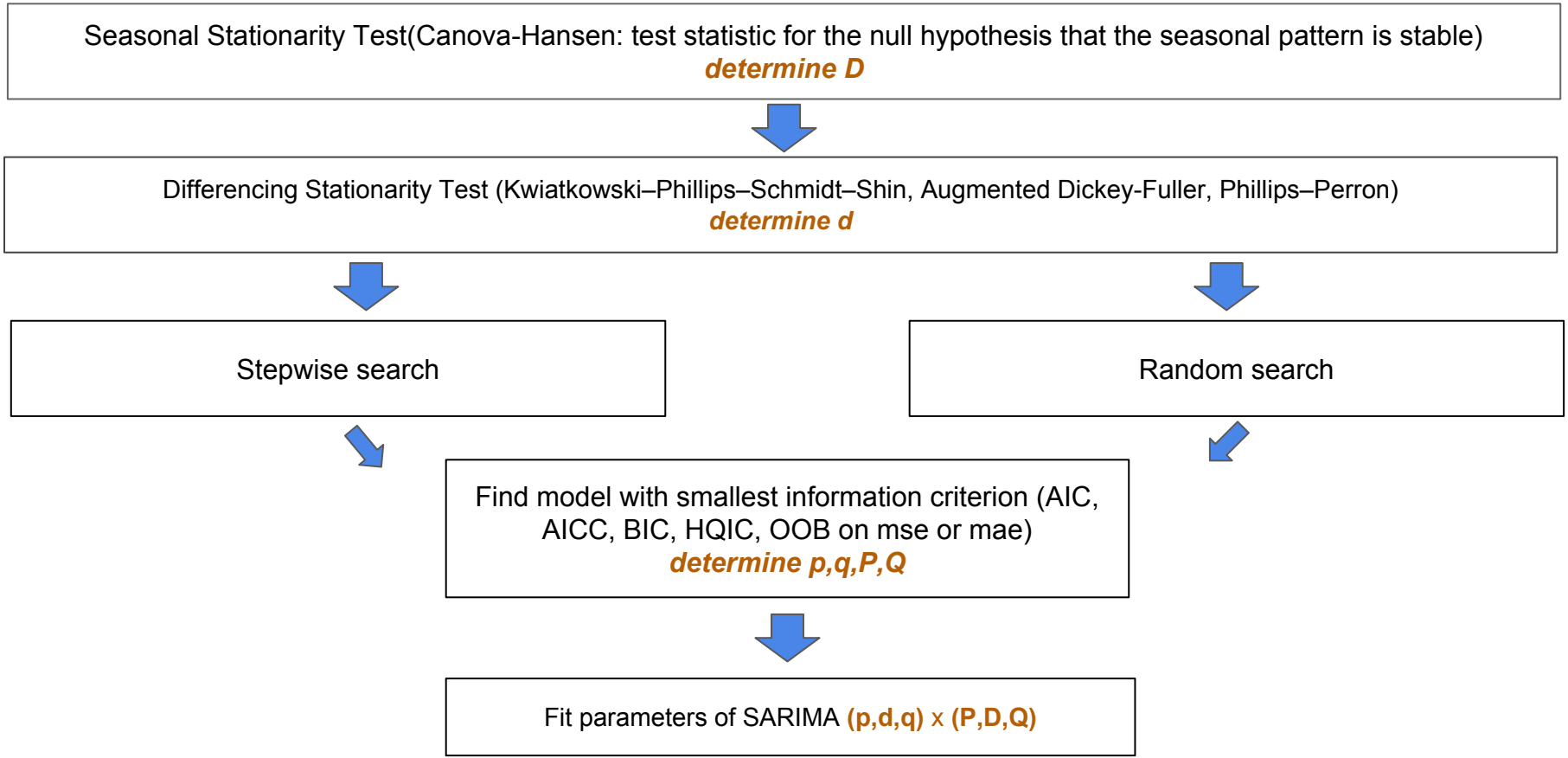
Pyramid library (auto-arima based on statsmodels' ARIMA and SARIMAX models)

The SARIMA model is specified $(p, d, q) \times (P, D, Q)_s$.

$$\phi_p(L)\tilde{\phi}_P(L^s)\Delta^d\Delta_s^D y_t = A(t) + \theta_q(L)\tilde{\theta}_Q(L^s)\zeta_t$$

```
Signature: auto_arima(y, exogenous=None, start_p=2, d=None, start_q=2, max_p=5, max_
d=2, max_q=5, start_P=1, D=None, start_Q=1, max_P=2, max_D=1, max_Q=2, max_order=10,
m=1, seasonal=True, stationary=False, information_criterion='aic', alpha=0.05, test=
'kpss', seasonal_test='ch', stepwise=True, n_jobs=1, start_params=None, trend='c', m
ethod=None, transparams=True, solver='lbfgs', maxiter=50, disp=0, callback=None, off
set_test_args=None, seasonal_test_args=None, suppress_warnings=False, error_action='
warn', trace=False, random=False, random_state=None, n_fits=10, return_valid_fits=Fa
lse, out_of_sample_size=0, scoring='mse', scoring_args=None, **fit_args)
```

Pyramid library (auto-arima based on statsmodels' ARIMA and SARIMAX models)



Pyramid library (auto-arima based on statsmodels' ARIMA and SARIMAX models)

Stepwise search

Journal of Statistical Software
July 2008, Volume 27, Issue 3. *http://www.jstatsoft.org/*

Automatic Time Series Forecasting: The forecast Package for R

Rob J. Hyndman
Monash University

Yeasmin Khandakar
Monash University

Step 1: We try four possible models to start with.

- ARIMA$(2,d,2)$ if $m=1$ and ARIMA$(2,d,2)(1,D,1)$ if $m>1$.
- ARIMA$(0,d,0)$ if $m=1$ and ARIMA$(0,d,0)(0,D,0)$ if $m>1$.
- ARIMA$(1,d,0)$ if $m=1$ and ARIMA$(1,d,0)(1,D,0)$ if $m>1$.
- ARIMA$(0,d,1)$ if $m=1$ and ARIMA$(0,d,1)(0,D,1)$ if $m>1$.

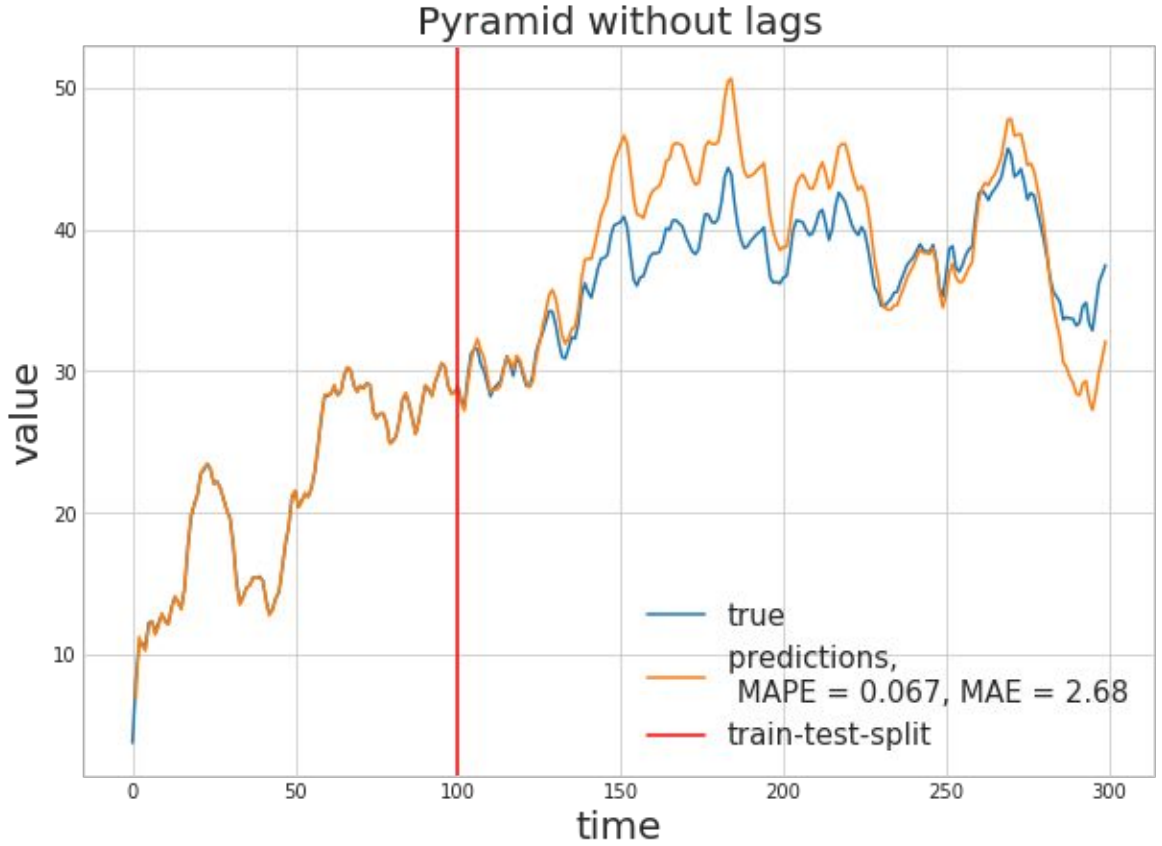
If $d+D\leq 1$, these models are fitted with $c\neq 0$. Otherwise, we set $c=0$. Of these four models, we select the one with the smallest AIC value. This is called the "current" model and is denoted by ARIMA(p,d,q) if $m=1$ or ARIMA$(p,d,q)(P,D,Q)_m$ if $m>1$.

Step 2: We consider up to thirteen variations on the current model:

- where one of p, q, P and Q is allowed to vary by ± 1 from the current model;
- where p and q both vary by ± 1 from the current model;
- where P and Q both vary by ± 1 from the current model;
- where the constant c is included if the current model has $c=0$ or excluded if the current model has $c\neq 0$.

Whenever a model with lower AIC is found, it becomes the new "current" model and the procedure is repeated. This process finishes when we cannot find a model close to the current model with lower AIC.

Pyramid library (auto-arima based on statsmodels' ARIMA and SARIMAX models)



Pyramid library (auto-arima based on statsmodels' ARIMA and SARIMAX models)

Statespace Model Results

Dep. Variable:	y	**No. Observations:**	100
Model:	SARIMAX(1, 0, 3)	**Log Likelihood**	72.633
Date:	Fri, 20 Jul 2018	**AIC**	-123.266
Time:	16:17:41	**BIC**	-94.609
Sample:	0	**HQIC**	-111.668
	- 100		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-0.1213	0.360	-0.336	0.737	-0.828	0.585
drift	0.0044	0.006	0.730	0.465	-0.007	0.016
x1	1.8280	0.166	11.005	0.000	1.502	2.154
x2	-3.4512	0.324	-10.659	0.000	-4.086	-2.817
x3	-1.1977	0.410	-2.920	0.004	-2.002	-0.394
x4	1.4796	1.049	1.411	0.158	-0.576	3.535
ar.L1	0.9742	0.032	30.558	0.000	0.912	1.037
ma.L1	2.8102	0.099	28.348	0.000	2.616	3.004
ma.L2	2.6893	0.186	14.423	0.000	2.324	3.055
ma.L3	0.8733	0.088	9.938	0.000	0.701	1.046
sigma2	0.0101	0.002	4.595	0.000	0.006	0.014

Ljung-Box (Q):	77.69	**Jarque-Bera (JB):**	0.69
Prob(Q):	0.00	**Prob(JB):**	0.71
Heteroskedasticity (H):	0.35	**Skew:**	0.19
Prob(H) (two-sided):	0.00	**Kurtosis:**	2.86

$$SARIMAX(1,0,3) \rightarrow ARMA(1,3)$$

$\Rightarrow$

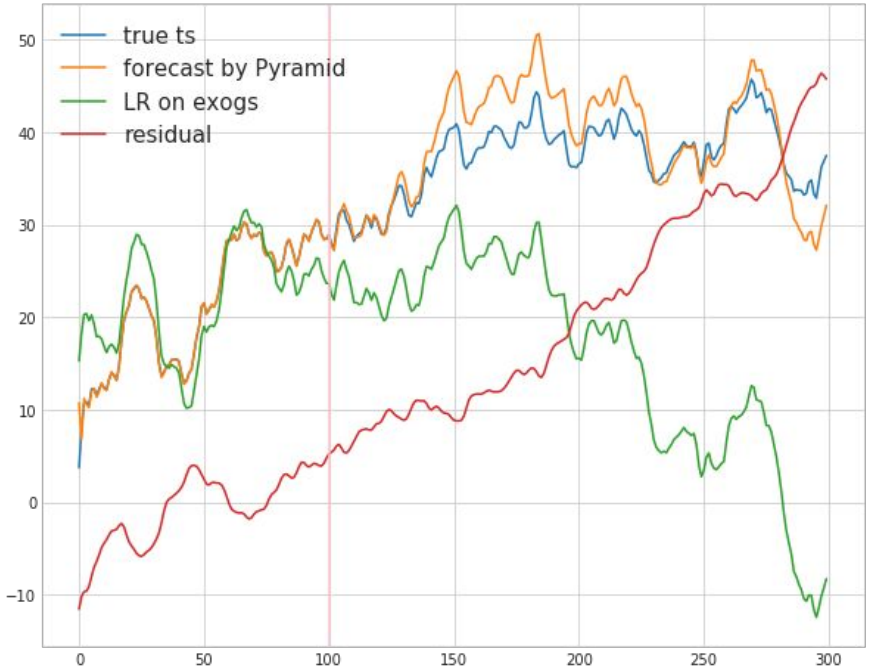
$$y_t = c_0 + c_1 t + x_1 Exog_t^1 + x_2 Exog_t^2 + x_3 Exog_t^3 + x_4 Exog_t^4 + \phi_1 y_{t-1} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \theta_3 \epsilon_{t-3} + \epsilon_t$$

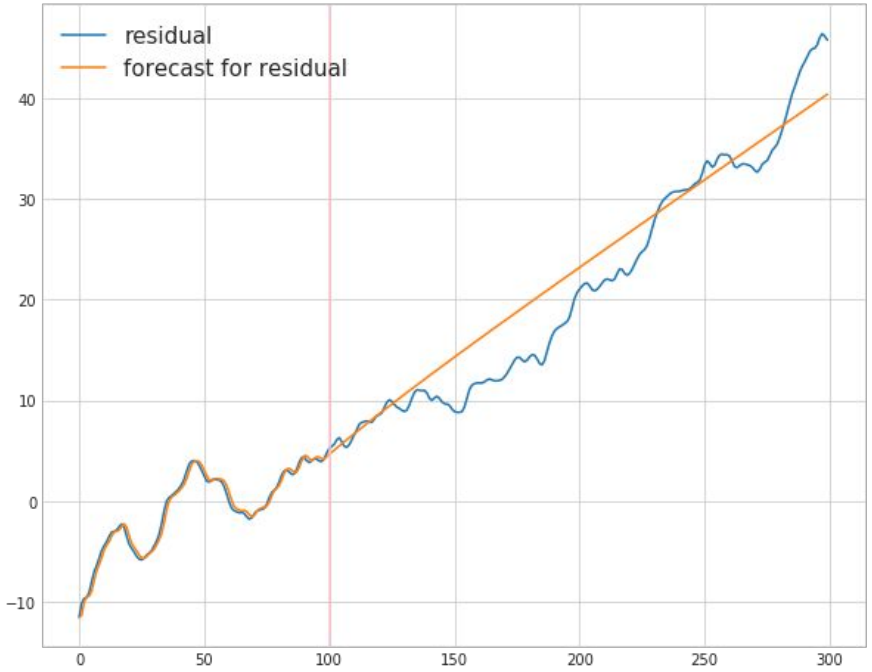
$\Rightarrow$

$$y_t = 0.121 + 0.004t + 1.827 * Exog_t^1 - 3.451 * Exog_t^2 - 1.197 * Exog_t^3 + 1.479 * Exog_t^4 + \\ +0.972 * y_{t-1} + 2.810 * \epsilon_{t-1} + 2.689 * \epsilon_{t-2} + 0.873 * \epsilon_{t-3} + \epsilon_t$$

, where $\epsilon \sim \mathcal{N}(0, 0.010)$

Pyramid library (auto-arima based on statsmodels' ARIMA and SARIMAX models)



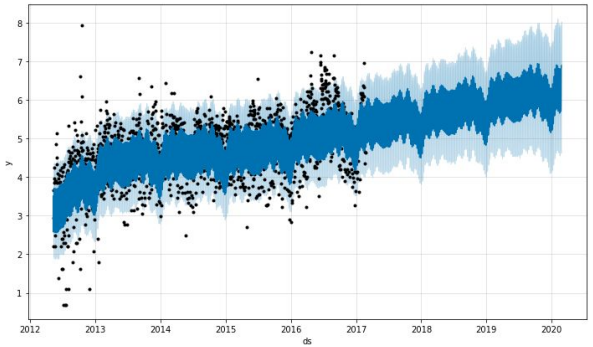


Facebook prophet model

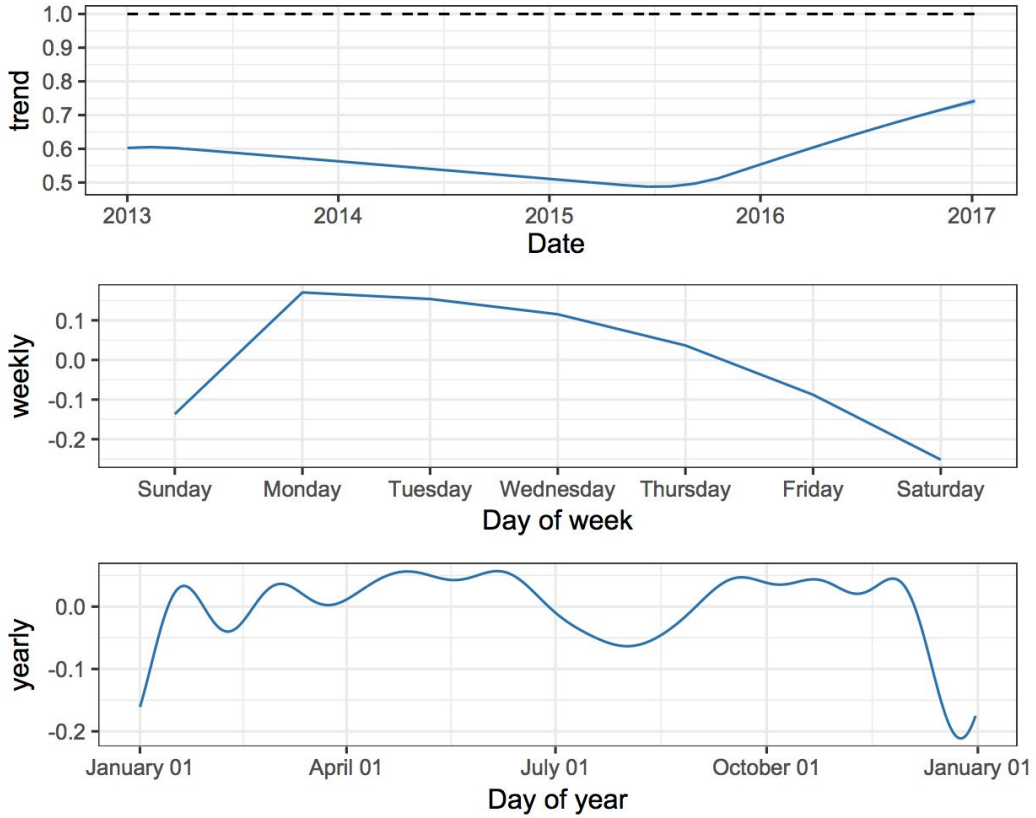
- Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects
- It works best with time series that have strong seasonal effects and several seasons of historical data
- Prophet is robust to missing data and shifts in the trend, and typically handles outliers well

```
m = Prophet()
m.fit(df)
forecast = m.predict(future)
```

	DS	YHAT	YHAT_LOWER	YHAT_UPPER
3265	2017-01-15	8.199274	7.489884	8.969065
3266	2017-01-16	8.524244	7.790682	9.266504
3267	2017-01-17	8.311615	7.553025	9.049803
3268	2017-01-18	8.144232	7.428174	8.864747
3269	2017-01-19	8.156091	7.395160	8.883232



Components of forecasting



Birdview of facebook prophet model

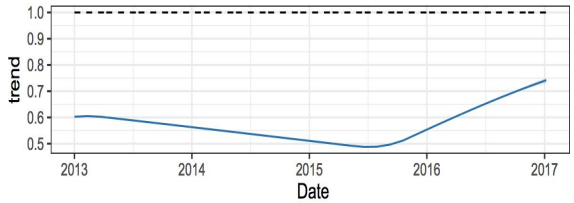
$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

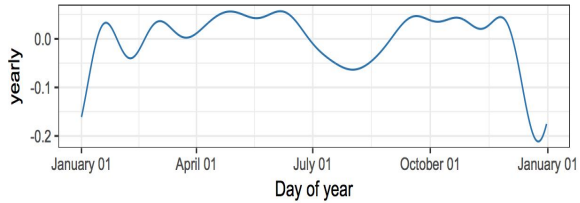
Trend component *models non-periodic changes in the value of the time series*

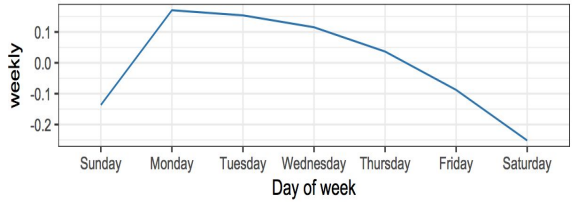
Seasonal component *models periodic changes (e.g., weekly and yearly seasonality)*

Events component *models the effects of holidays which occur on potentially irregular schedules over one or more days.*

Error component *represents any idiosyncratic changes which are not accommodated by the model*







Final formula for prophet model

$$\hat{y} = \text{trend} * (1 + \text{multiplicative terms}) + \text{additive terms}$$

Seasonal features *Exogenous regressors* *Seasonal features* *Exogenous regressors*

$$\hat{y}_t = \left[(k + \sum_{i>=t} \delta_i)t + m + \sum_{i>=t}(-\text{changepoint}_i \delta_i)\right] * y_{\text{scaled}}$$

Trend

$$\left\{1 + \sum_{\substack{i \in I \\ I=\{\text{multiplicative and seasonal features}\}}} \sum_{j=0}^{\text{seasonal order(i)}} \left(\beta_{i+j}[j\%2 == 0]Sin\left(\frac{2\pi t(j+1)}{Period}\right) + \beta_{i+j}[j\%2 == 1]Cos\left(\frac{2\pi t(j+1)}{Period}\right)\right)\right.$$

$$\left. + \sum_{\substack{i \in I \\ I=\{\text{multiplicative and exogenous features}\}}} \left(\beta_i \frac{regressor_i - mu_{train}}{std_{train}}\right)\right\} +$$

Multiplicative terms

$$\left\{1 + \sum_{\substack{i \in I \\ I=\{\text{additive and seasonal features}\}}} \sum_{j=0}^{\text{seasonal order(i)}} \left(\beta_{i+j}[j\%2 == 0]Sin\left(\frac{2\pi t(j+1)}{Period}\right) + \beta_{i+j}[j\%2 == 1]Cos\left(\frac{2\pi t(j+1)}{Period}\right)\right)\right.$$

$$\left. + \sum_{\substack{i \in I \\ I=\{\text{additive and exogenous features}\}}} \left(\beta_i \frac{regressor_i - mu_{train}}{std_{train}}\right)\right\}$$

Additive terms

Multiplicative terms contain both seasonal and exog. regressors

$$1 + \underbrace{\sum_{\substack{i \in I \\ I=\{\text{multiplicative and seasonal features}\}}} \sum_{j=0}^{\text{seasonal order(i)}} \left(\beta_{i+j}[j\%2 == 0] Sin\left(\frac{2\pi t(j+1)}{Period}\right) + \beta_{i+j}[j\%2 == 1] Cos\left(\frac{2\pi t(j+1)}{Period}\right)\right) + \sum_{\substack{i \in I \\ I=\{\text{multiplicative and exogenous features}\}}} \left(\beta_i \frac{regressor_i(t) - \mu_{train}}{std_{train}}\right)}$$

$Sin\left(\frac{2\pi t(1)}{12}\right)$ $Cos\left(\frac{2\pi t(2)}{12}\right)$ $\frac{f_1(t) - \mu_{train}}{std_{train}}$

time

	yearly_delim_1	yearly_delim_2	monthly_delim_1	monthly_delim_2	f1	f2
0	-8.681856e-13	1.000000	-0.951057	0.309017	1.740690	-1.198350
1	-7.832139e-13	-1.000000	-0.951057	0.309017	1.775159	1.106529
2	5.000000e-01	0.866025	-0.866025	0.500000	1.809628	2.469238

$Sin\left(\frac{2\pi t(1)}{30}\right)$ $Cos\left(\frac{2\pi t(2)}{30}\right)$

Training model parameters

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix}$$

seasonal components (Fourier basis coefficients)

exogenous components

Elementwise product

mask for multiplicative features

Additive terms contain both seasonal and exog. regressors

$$1 + \underbrace{\sum_{\substack{i \in I \\ I=\{\text{additive and seasonal features}\}}} \sum_{j=0}^{\text{seasonal order(i)}} \left(\beta_{i+j}[j\%2 == 0] Sin\left(\frac{2\pi t(j+1)}{Period}\right) + \beta_{i+j}[j\%2 == 1] Cos\left(\frac{2\pi t(j+1)}{Period}\right)\right) + \sum_{\substack{i \in I \\ I=\{\text{additive and exogenous features}\}}} \left(\beta_i \frac{regressor_i(t) - \mu_{train}}{std_{train}}\right)}$$

$Sin\left(\frac{2\pi t(1)}{12}\right)$ $Cos\left(\frac{2\pi t(2)}{12}\right)$ $\frac{f_1(t) - \mu_{train}}{std_{train}}$

time

	yearly_delim_1	yearly_delim_2	monthly_delim_1	monthly_delim_2	f1	f2
0	-8.681856e-13	1.000000	-0.951057	0.309017	1.740690	-1.198350
1	-7.832139e-13	-1.000000	-0.951057	0.309017	1.775159	1.106529
2	5.000000e-01	0.866025	-0.866025	0.500000	1.809628	2.469238

$Sin\left(\frac{2\pi t(1)}{30}\right)$ $Cos\left(\frac{2\pi t(2)}{30}\right)$

Training model parameters

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \odot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix}$$

seasonal components (Fourier basis coefficients)

exogenous components

Elementwise product

mask for additive features

Trend component of prediction

$$\left[(k + \sum_{i>=t} \delta_i)t + m + \sum_{i>=t} (-\text{changepoint}_i \delta_i)\right] * y_{\text{scaled}}$$

Base trend rate

Changes in the trend in changepoints

Offset parameter
Shift on target axis

Points in which we calculate the changes in rate

Scaled value of target

The trend model

Base trend

$$k + \sum_{j:t>s_j} \delta_j.$$

Cumulative sum in trend changes

$$a_j(t) = \begin{cases} 1, & \text{if } t \geq s_j, \\ 0, & \text{otherwise.} \end{cases}$$

$$\lambda = \frac{1}{S} \sum_{j=1}^{S} |\delta_j|.$$

$$\forall j > T, \quad \begin{cases} \delta_j = 0 \text{ w.p. } \frac{T-S}{T}, \\ \delta_j \sim \text{Laplace}(0, \lambda) \text{ w.p. } \frac{S}{T} \end{cases}$$

Nonlinear, Saturating Growth.

Linear growth

$$g(t) = \frac{C}{1 + \exp(-k(t-m))}$$

$$g(t) = (k + \mathbf{a}(t)^\intercal \boldsymbol{\delta})t + (m + \mathbf{a}(t)^\intercal \boldsymbol{\gamma})$$

Seasonality

Model

$$s(t) = \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi n t}{P}\right) + b_n \sin\left(\frac{2\pi n t}{P}\right) \right)$$

Define the following vectors

$$X(t) = \left[\cos\left(\frac{2\pi(1)t}{365.25}\right), \ldots, \sin\left(\frac{2\pi(10)t}{365.25}\right) \right] \qquad \boldsymbol{\beta} = [a_1, b_1, \ldots, a_N, b_N]^\intercal$$

Model for training

$$s(t) = X(t)\boldsymbol{\beta}. \qquad \boldsymbol{\beta} \sim \text{Normal}(0, \sigma^2)$$

Holidays and events

Model

$$Z(t) = [\mathbf{1}(t \in D_1), \ldots, \mathbf{1}(t \in D_L)]$$

Model for training

$$h(t) = Z(t)\boldsymbol{\kappa}. \qquad \boldsymbol{\kappa} \sim \text{Normal}(0, \nu^2)$$

Holidays and events

Model

$$Z(t) = [\mathbf{1}(t \in D_1), \ldots, \mathbf{1}(t \in D_L)]$$

Model for training

$$h(t) = Z(t)\boldsymbol{\kappa}. \qquad \boldsymbol{\kappa} \sim \text{Normal}(0, \nu^2)$$

Conclusion

Old version

All experiments' results are reflected in one table

		autoregressive dependence		dependence from exogenous features	
	Model \ *Features*	Y	tsfresh(Y)	Y + exogenous	tsfresh(Y + exogenous)
Baseline models	Linear regression	.	.	.	.
	SARIMAX	.	.	.	.
	RNN	.	.	.	.
AutoML algorithms	Auto-arima (pyramid)	.	.	.	.
	Facebook Prophet	.	.	.	.
	Auto-sklearn	.	.	.	.

Results on synthetic dataset
1-st dataset

		autoregressive dependence		dependence from exogenous features	
	Model \ *Features*	Y	tsfresh(Y)	Y + exogenous	tsfresh(Y + exogenous)
		.	.	.	.
AutoML algorithms	PyFlux	**0.208 43**	.	**?**	.
	Facebook Prophet	**0.29 121**		**0 0.065**	.
	Auto-arima (stepwise)	**0.30 92**	.	**0.0 0**	.
	Auto-arima (random)	**0.04 12**	.	**0.0 0**	.
	PyAf	**0.33 100**	.		.

MAPE
MAE

Results on synthetic dataset
2-nd dataset

		autoregressive dependence		dependence from exogenous features	
	Model \ *Features*	Y	tsfresh(Y)	Y + exogenous	tsfresh(Y + exogenous)
		.	.	.	.
AutoML algorithms	PyFlux	**0.25** **143**	.	**?**	.
	Facebook Prophet	**0.31** **334**		**0.17** **151**	.
	Auto-arima (stepwise)	**0.02** **17**	.	**0.05** **35**	.
	Auto-arima (random)	**0.12** **94**	.	**0.07** **50**	.
	PyAf	**0.17** **139**	.		.

Python Automatic Forecasting (PyAF)

Generic training features

- Signal decomposition as the sum of a trend, periodic and AR(25) component
- PyAF works as a competition between a **comprehensive set of possible signal transformations and linear decompositions**. For each transformed signal , a set of possible trends, periodic components and AR models is generated and all the possible combinations are estimated. The best decomposition in term of performance is kept to forecast the signal (the performance is computed on a part of the signal that was not used for the estimation).
- **Signal transformation** is supported before **signal decompositions**. Four transformations are supported by default. Other transformation are available (Box-Cox etc).
- All Models are estimated using **standard procedures and state-of-the-art time series modeling**. For example, trend regressions and AR/ARX models are estimated using scikit-learn linear regression models.
- Standard performance measures are used (L1, RMSE, MAPE, etc)
- **Hierarchical Forecasting**

PyFlux

PyFlux is a library for time series analysis and prediction.

- Users can choose from a flexible range of modelling and inference options, and use the output for forecasting and retrospection
- Users can build a full probabilistic model where the data y and latent variables (parameters) z are treated as random variables through a joint probability $p(y,z)$.
- The advantage of a probabilistic approach is that it gives a more complete picture of uncertainty, which is important for time series tasks such as forecasting.

```
my_model = pf.ARIMA(data=my_dataframe, ar=2, ma=0, family=pf.Normal())
```