Generalised learning of time-series
Reconstructive Cross-Validation

Mehmet Süzen PhD MInstP
Alper Yegenoglu

Based on the working article

Generalised learning of time-series: Ornstein-Uhlenbeck processes

M.Suzen & A. Yegenoglu

https://arxiv.org/abs/1910.09394

Based on the working article

Generalised learning of time-series: Ornstein-Uhlenbeck processes

M.Suzen & A. Yegenoglu

https://arxiv.org/abs/1910.09394

Prototype implementation

https://nbviewer.jupyter.org/urls/arxiv.org/src/1910.09394v1/anc/rCV_prototype.ipynb

Agenda

- Background: Generalisation for temporal learning

Agenda

- Background: Generalisation for temporal learning

- Reconstructive Cross-Validation (rCV) : A meta-algorithm

Agenda

- Background: Generalisation for temporal learning
- Reconstructive Cross-Validation (rCV) : A meta-algorithm
- Generating learning curves for time-series via rCV

Agenda

- Background: Generalisation for temporal learning
- Reconstructive Cross-Validation (rCV) : A meta-algorithm
- Generating learning curves for time-series via rCV
- A proof of concept implementation
    - rCV on Ornstein-Uhlenbeck process

Agenda

- Background: Generalisation for temporal learning
- Reconstructive Cross-Validation (rCV) : A meta-algorithm
- Generating learning curves for time-series via rCV
- A proof of concept implementation
  - rCV on Ornstein-Uhlenbeck process
- Outlook

Background: Generalisation for temporal learning

What is learning?

Background: Generalisation for temporal learning

What is learning?

*A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*

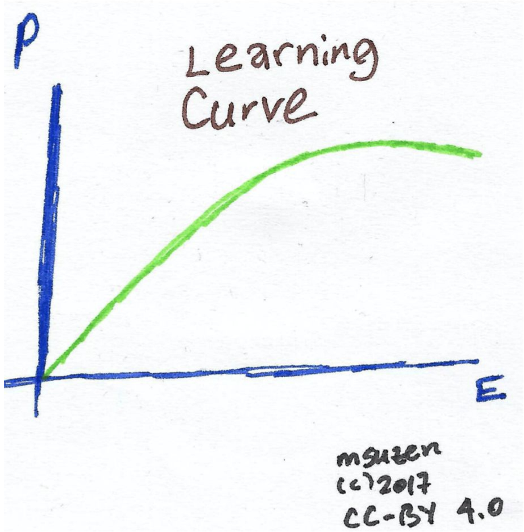
Tom Mitchell (1997)

Background: Generalisation for temporal learning

What is learning?

*A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*

Tom Mitchell (1997)

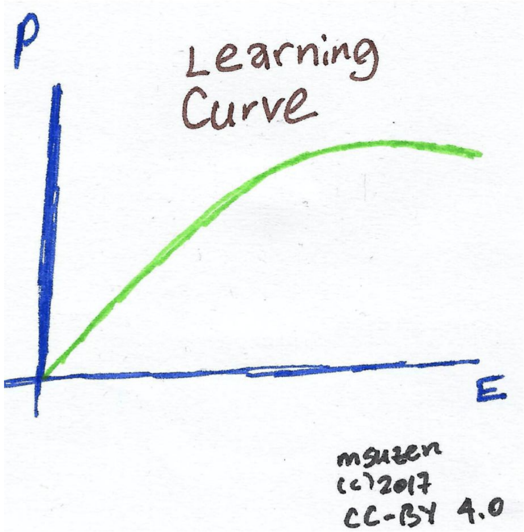


Background: Generalisation for temporal learning

What is learning?

*A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*

Tom Mitchell (1997)

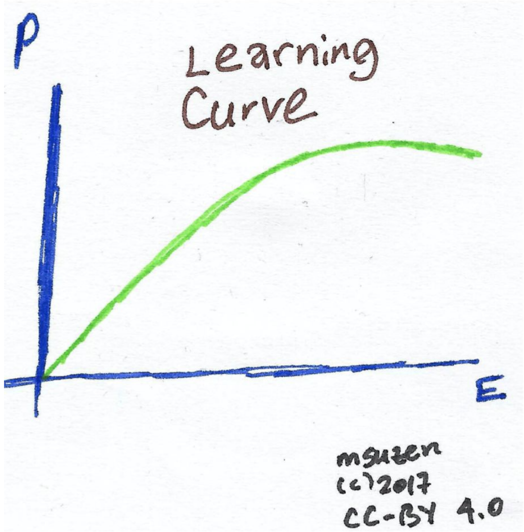


Hermann Ebbinghaus(1913) Learning/Forgetting curves

Background: Generalisation for temporal learning

What is learning?

*A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*
Avoid overtraining (do not memorize) and avoid overfitting (Occam's Razor).
Tom Mitchell (1997)



Hermann Ebbinghaus(1913) Learning/Forgetting curves

Background: Generalisation for temporal learning

Overtraining (generalisation)

Occam's Razor (the least complex model possible)

Background: Generalisation for temporal learning

Overtraining (generalisation)

Occam's Razor (the least complex model possible) [*]

[*] Ignored in deep learning and modern times

Background: Generalisation for temporal learning

Overtraining (generalisation) ***How to do cross-validation for time-series?***

Occam's Razor (the least complex model possible) [*]

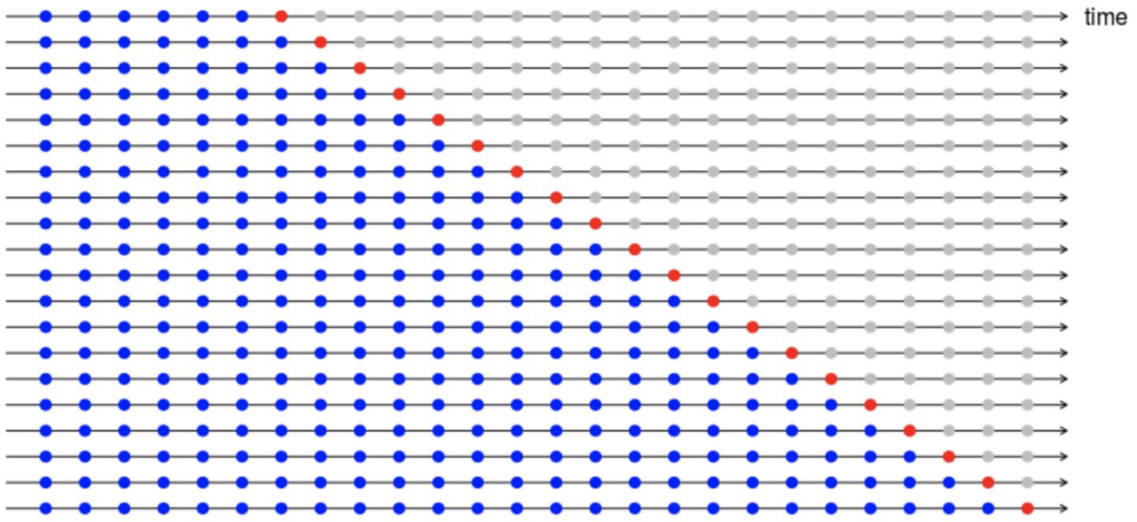
[*] Ignored in deep learning and modern times

Background: Generalisation for temporal learning

Overtraining (generalisation) ***How to do cross-validation for time-series?***

Occam's Razor (the least complex model possible) [*]

- Out-of-Sample (OOS) Evaluation



Hyndmann 2016

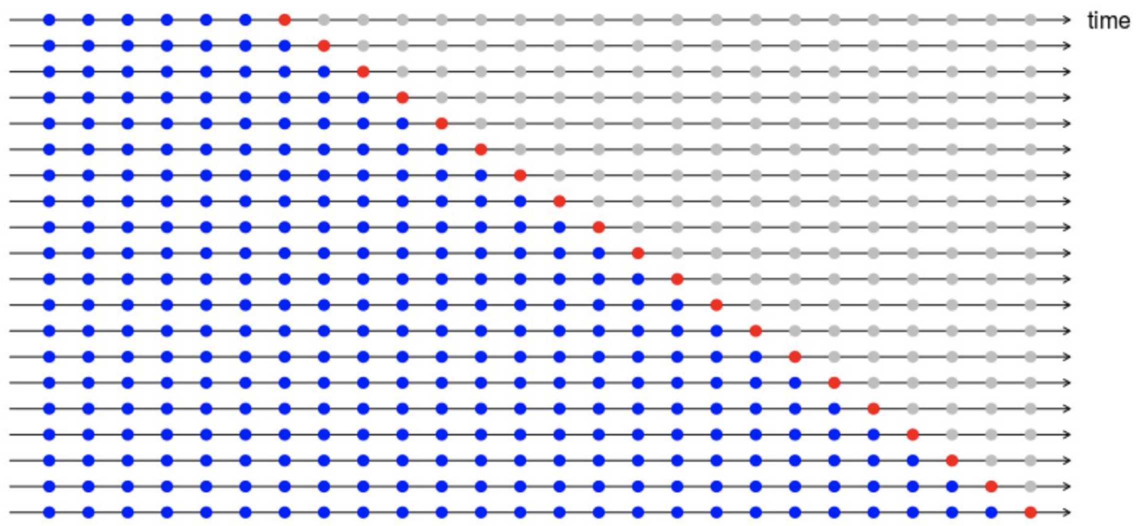
[*] Ignored in deep learning and modern times

Background: Generalisation for temporal learning

Overtraining (generalisation) ***How to do cross-validation for time-series?***

Occam's Razor (the least complex model possible) [*]

- Out-of-Sample (OOS) Evaluation
- Block-resampling Politis-Romano (1994)



Hyndmann 2016

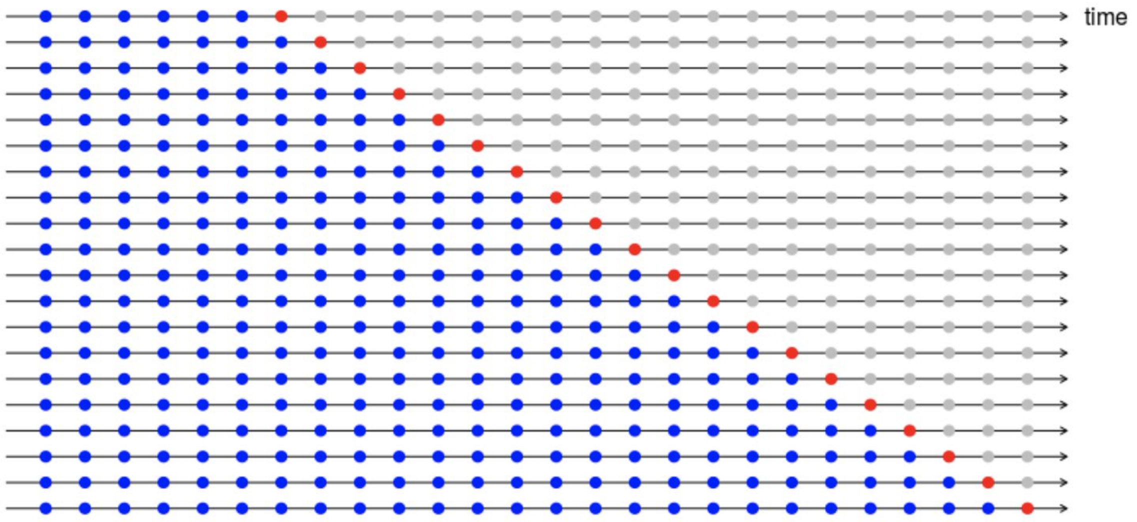
[*] Ignored in deep learning and modern times

Background: Generalisation for temporal learning

Overtraining (generalisation) ***How to do cross-validation for time-series?***

Occam's Razor (the least complex model possible) [*]

- Out-of-Sample (OOS) Evaluation
- Block-resampling Politis-Romano (1994)
- Naive CV: Stationarity Correlations



Hyndmann 2016

[*] Ignored in deep learning and modern times

Reconstructive Cross-Validation (rCV) : A meta-algorithm

Reconstructive Cross-Validation (rCV) : A meta-algorithm

Design principles of rCV for time-series

Reconstructive Cross-Validation (rCV) : A meta-algorithm

Design principles of rCV for time-series

- Logically close to standard cross-validation:
  Arbitrary test-set size and number of folds.

Reconstructive Cross-Validation (rCV) : A meta-algorithm

Design principles of rCV for time-series

- Logically close to standard cross-validation:
  Arbitrary test-set size and number of folds.

- Preserve correlations and data order.

Reconstructive Cross-Validation (rCV) : A meta-algorithm

Design principles of rCV for time-series

- Logically close to standard cross-validation:
  Arbitrary test-set size and number of folds.

- Preserve correlations and data order.

- Does not create absurdity of predicting past from the future data.

Reconstructive Cross-Validation (rCV) : A meta-algorithm

Design principles of rCV for time-series

- Logically close to standard cross-validation:
  Arbitrary test-set size and number of folds.

- Preserve correlations and data order.

- Does not create absurdity of predicting past from the future data.

- Applicable in generic fashion regardless of learning algorithm.

Reconstructive Cross-Validation (rCV) : A meta-algorithm

Design principles of rCV for time-series

- Logically close to standard cross-validation:
  Arbitrary test-set size and number of folds.
- Preserve correlations and data order.
- Does not create absurdity of predicting past from the future data.
- Applicable in generic fashion regardless of learning algorithm.
- Applicable to multi-dimensional time-series.

Reconstructive Cross-Validation (rCV) : A meta-algorithm

Design principles of rCV for time-series

- Logically close to standard cross-validation:
  Arbitrary test-set size and number of folds.
- Preserve correlations and data order.
- Does not create absurdity of predicting past from the future data.
- Applicable in generic fashion regardless of learning algorithm.
- Applicable to multi-dimensional time-series.
- Evaluation metric agnostic.

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- Given training and OOS set: $(y_i, t_i)\ (w_j, t_j), j \geq i$

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- Given training and OOS set: $(y_i, t_i)\ (w_j, t_j), j \geq i$
- Generating k-folds out of training

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- Given training and OOS set: $(y_i, t_i)\ (w_j, t_j), j \geq i$
- Generating k-folds out of training
  - Random k partitions $|\mathbf{y}^1| \approx |\mathbf{y}^2| \approx \ldots \approx |\mathbf{y}^k|$

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- Given training and OOS set: $(y_i, t_i)\ (w_j, t_j), j \geq i$
- Generating k-folds out of training
  - Random k partitions $|\mathbf{y}^1| \approx |\mathbf{y}^2| \approx \ldots \approx |\mathbf{y}^k|$
  - Generate k new training partitions with missing at random

$$Y^m = \bigcup_{l=1, l \neq m}^{k} \mathbf{y}^l$$

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- K-fold reconstruction i.e., any good imputation, filtering technique.

$$R^m = Y^m + \hat{\mathbf{y}}^m$$

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- K-fold reconstruction i.e., any good imputation, filtering technique.

$$R^m = Y^m + \hat{\mathbf{y}}^m$$

- Total error due to reconstruction, for example MAPE, component-wise

$$g_r = \frac{1}{k} \sum_{m=1}^{k} |(\mathbf{y}^m - \hat{\mathbf{y}}^m)|/\mathbf{y}^m.$$

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- A primary predictive model on K reconstructions

$$R^m = Y^m + \hat{\mathbf{y}}^m$$

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- A primary predictive model on K reconstructions

$$R^m = Y^m + \hat{\mathbf{y}}^m$$

- Test primary predictive model on OOS data set: difference component-wise $g_p = \frac{1}{k}\sum_{q=1}^{k}(\mathbf{w} - \hat{\mathbf{w}}^m)$

Reconstructive Cross-Validation (rCV) : A meta-algorithm

The formulation in 1-D ordered data: time-series

- A primary predictive model on K reconstructions

$$R^m = Y^m + \hat{\mathbf{y}}^m$$

- Test primary predictive model on OOS data set: difference component-wise. $$g_p = \frac{1}{k}\sum_{q=1}^{k}(\mathbf{w} - \hat{\mathbf{w}}^m)$$

- Total rCV error: Prediction and reconstruction errors, not-unique!

$$g_{rCV} = g_r \cdot g_p$$

Reconstructive Cross-Validation (rCV) : A meta-algorithm

How to produce learning curve for temporal prediction method?

- Usually not produced for temporal prediction methods.

Reconstructive Cross-Validation (rCV) : A meta-algorithm

How to produce learning curve for temporal prediction method?

- Usually not produced for temporal prediction methods.
- Use rCV performance with varying k-fold reconstructions.

$$L^{err}(k) = g_{err}^{k}$$

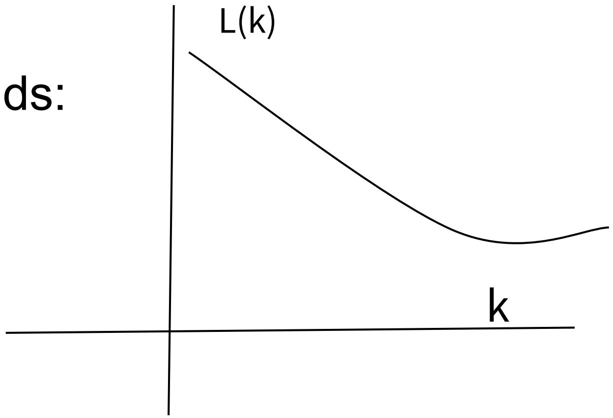
Generating learning curves for time-series via rCV

How to produce learning curve for temporal prediction method?

- Usually not produced for temporal prediction methods.
- Use rCV performance with varying k-fold reconstructions.

$$L^{err}(k) = g_{err}^{k}$$

- Increasing data size by increasing number of folds:
  More non-reconstructed points



A proof of concept implementation

- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
  - Brownian motion in statistical physics

A proof of concept implementation

- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
  - Brownian motion in statistical physics

- OU can be generated via multivariate Gaussian process via exponential Kernel

$$y_{ou}(t_i) \sim \mathcal{N}(\mathbf{5.0}, \exp(-\mathbf{D}^{ii}/2.0))$$

A proof of concept implementation

- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
  - Brownian motion in statistical physics

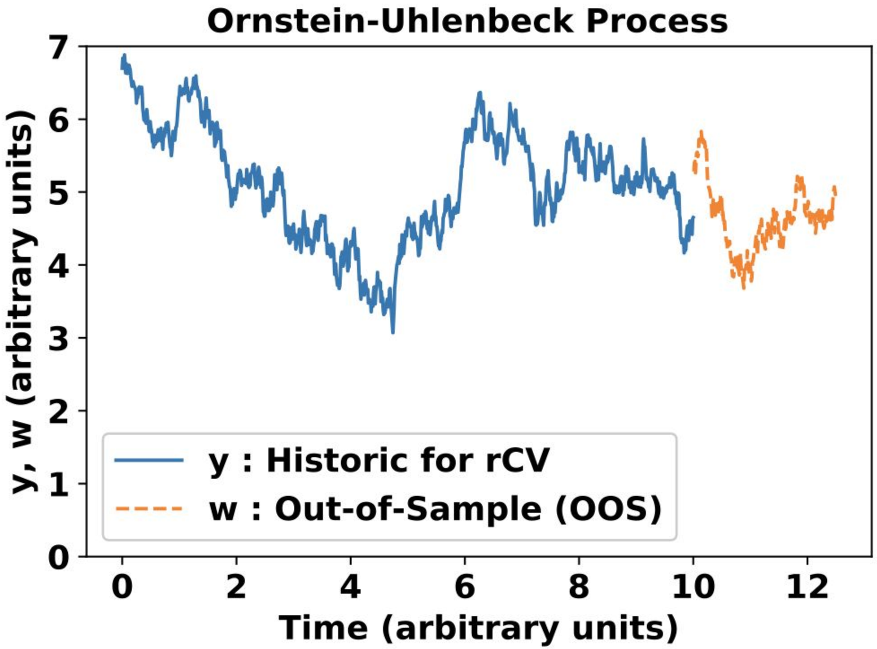
- OU can be generated via multivariate Gaussian process via exponential Kernel

$$y_{ou}(t_i) \sim \mathcal{N}(\mathbf{5.0}, \exp(-\mathbf{D}^{ii}/2.0))$$

- Distance matrix
  1000 points with 0.1 spacing and 500 points for OOS

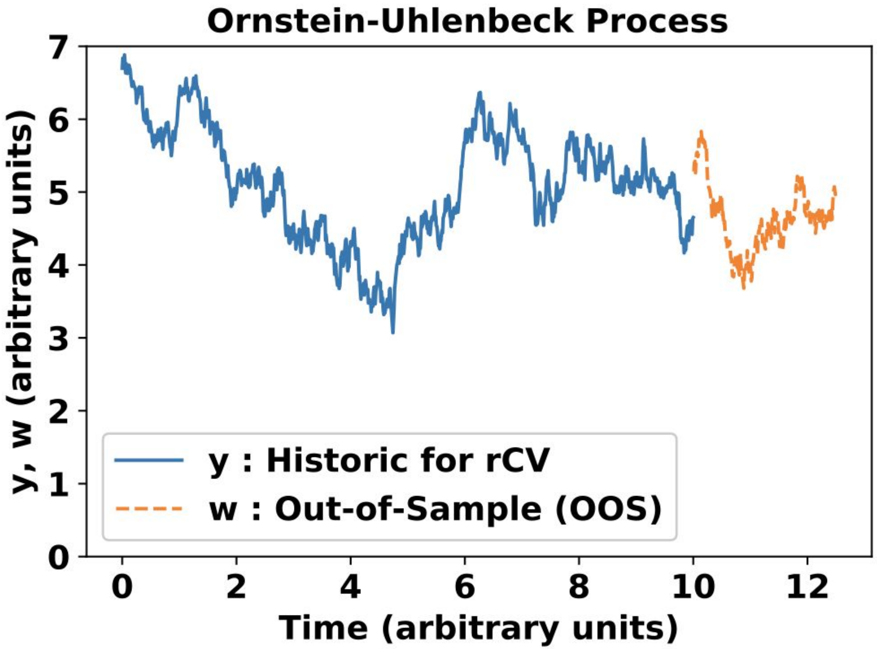
A proof of concept implementation

- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
- *Apply k-fold rCV*



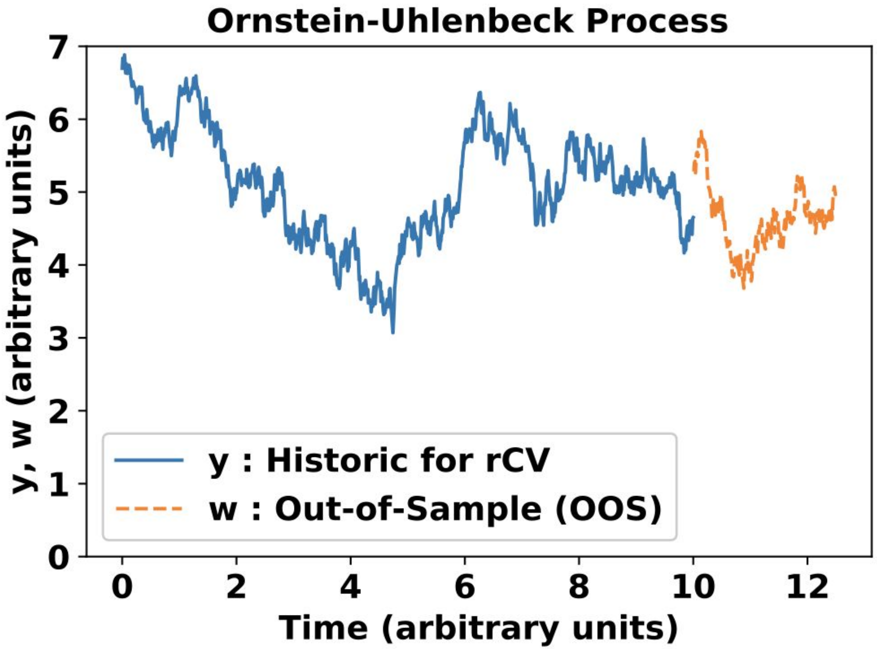
A proof of concept implementation

- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
- *Apply k-fold rCV*
    - *Use Gaussian Process for both reconstruction and temporal prediction model.*



A proof of concept implementation

- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
- *Apply k-fold rCV*
  - *Use Gaussian Process for both reconstruction and temporal prediction model.*
  - *No learning of GP parameters for temporal prediction as PoC only.*

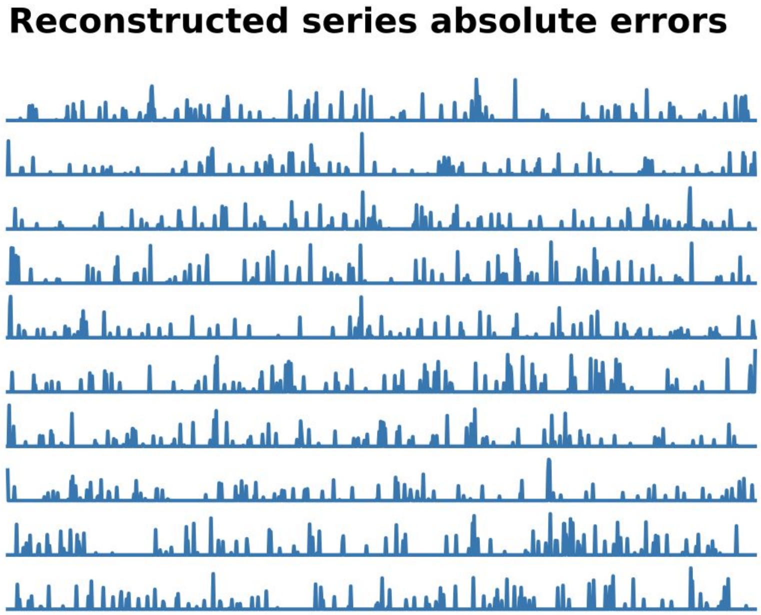


A proof of concept implementation

- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
- *Apply k-fold rCV*
  - *Gaussian Process*
    10-fold imputation
    Mean error 0.014.
    OU Kernel is used.

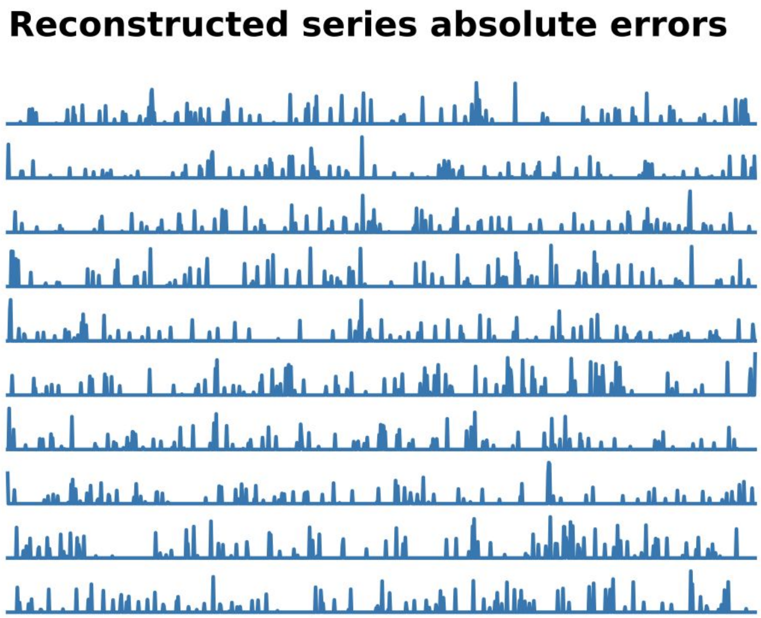
A proof of concept implementation

- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
- *Apply k-fold rCV*
  - *Gaussian Process*
    10-fold imputation
    Mean error 0.014.
    OU Kernel is used.



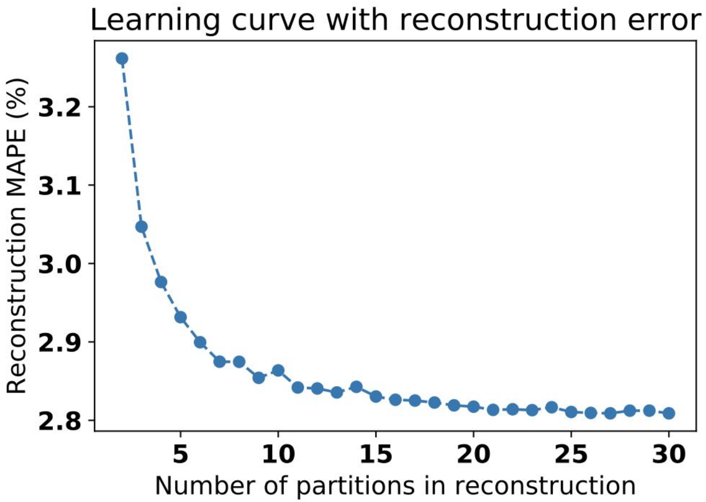
A proof of concept implementation

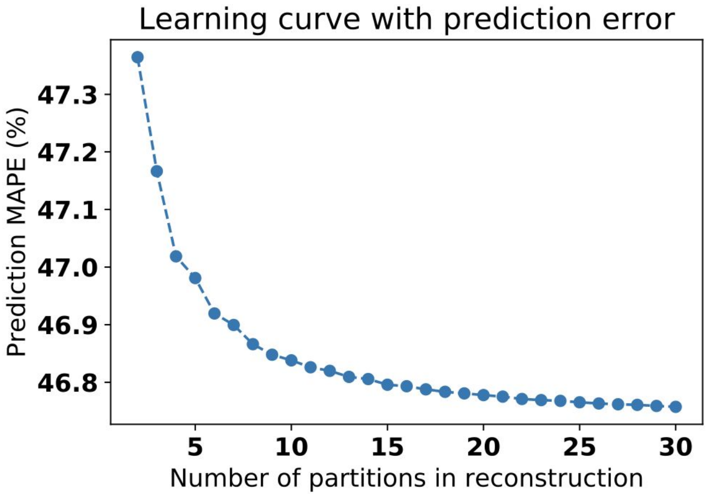
- Synthetic data : *Ornstein-Uhlenbeck Process (OU)*
- *Apply k-fold rCV*
  - *Gaussian Process*
    10-fold imputation
    Mean error 0.014.
    OU Kernel is used.
  - *rCV error : 0.013*
    MAPEs
    Reconstruction: 0.029
    Prediction: 0.468



A proof of concept implementation

- *Ornstein-Uhlenbeck Process (OU) : rCV Learning Curve*





Outlook

- To be submitted: in progress.

- A meta-algorithm rCV:
  Generic implementation as a reusable package (time-permits)

  - *Independent of imputation and prediction models used*
  - *Compatible interface with scikit-learn/statmodels/R-forecast*
  - *Include real data-set examples*
  - *Multivariate series examples*