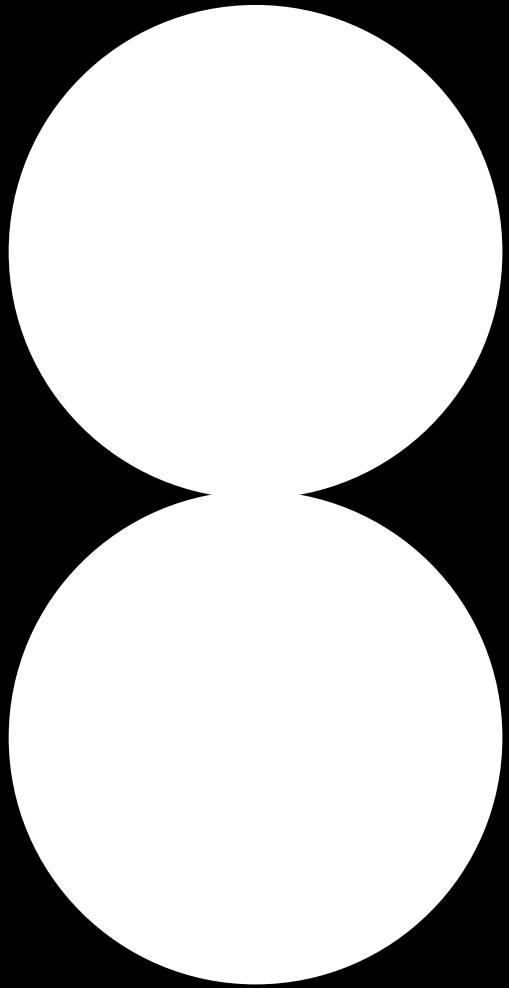
Unit8™

Darts for Time Series Forecasting

Julien Herzen and Francesco Lässig



Francesco

- Data Scientist @ Unit8
- Darts core contributor

Julien

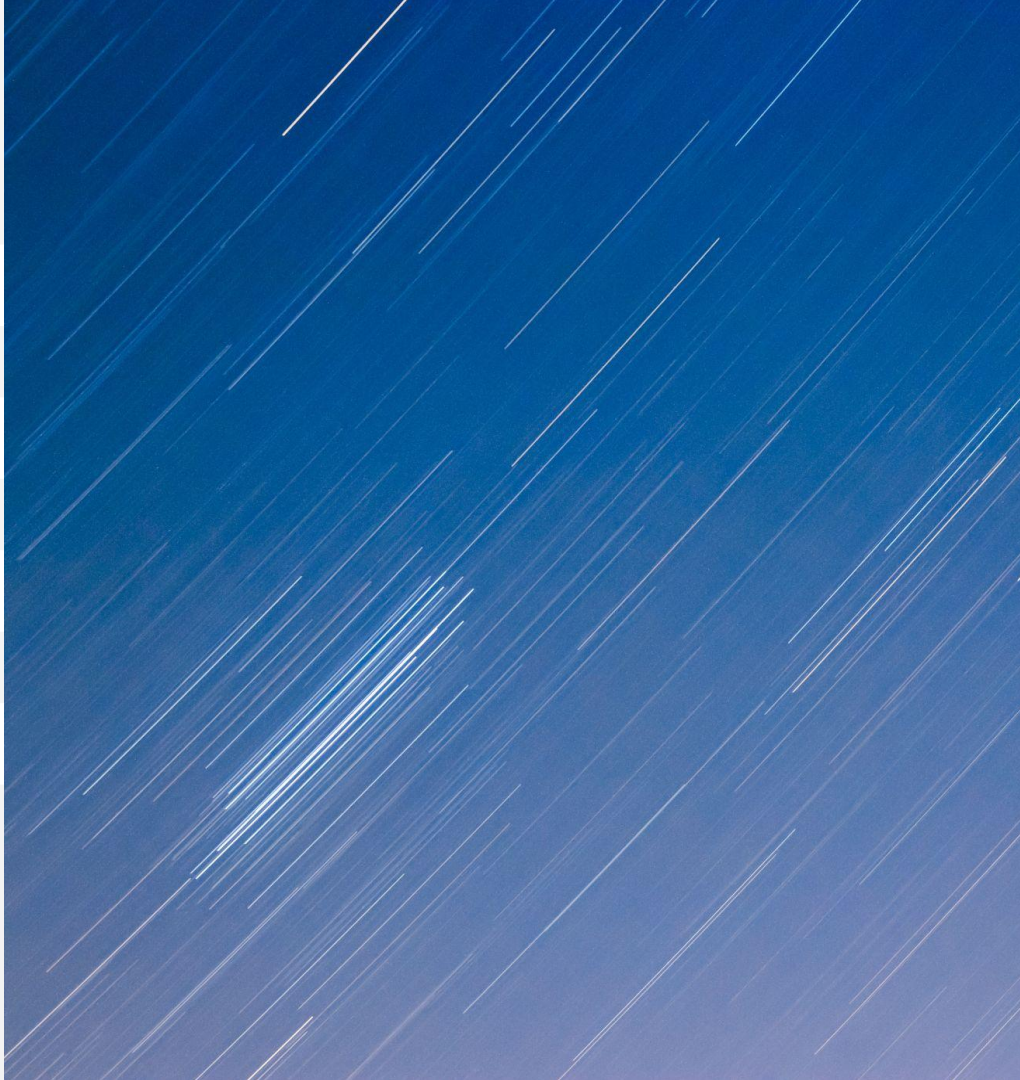
- Data Scientist & Area Director @ Unit8
- Darts core contributor

1 **Intro to Forecasting & Darts**

2 Basic forecasting using Darts

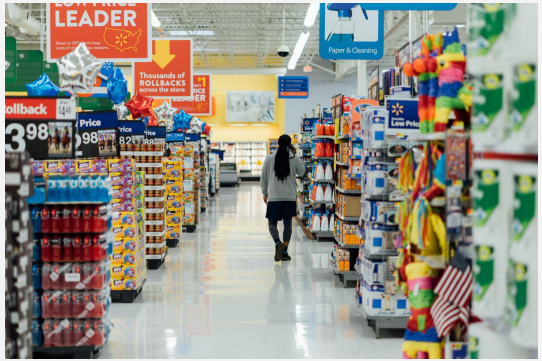
3 More advanced features

4 Conclusions & next steps



Why Forecasting?

Retail



How can we ensure that the consumer demand for every product is met without overspending and creating waste?

Energy

How can much energy should we produce?

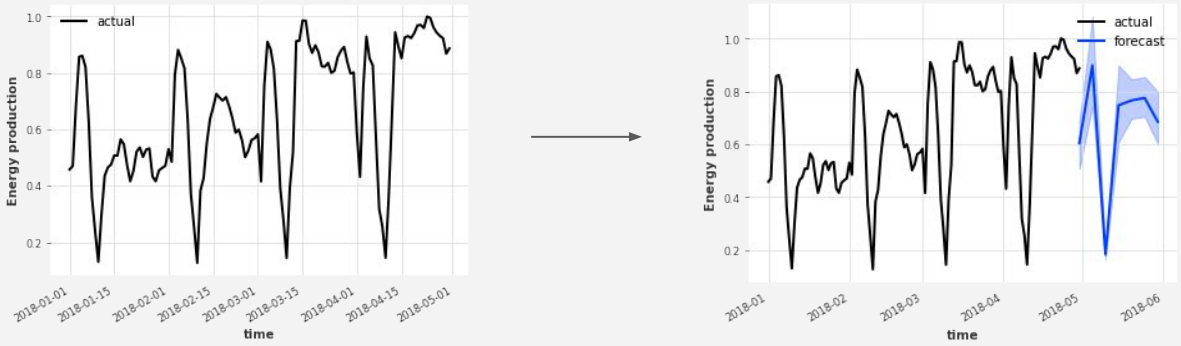
How much energy will we produce given a certain setup?

Telecommunications

Will our current infrastructure support the traffic in the near and far future?

Unit8

Time Series Forecasting



Future = *f*(history + external data)

Consideration: Do we have a valuable signal in the data?

Unit8

Darts

Darts is a Python library for easy manipulation and forecasting of time series.

- Immutable `TimeSeries` class as basic building block.
- Unified `fit()`, `predict()` interface.
- Classical models & state-of-the-art ML approaches.
- User-friendliness: intuitive API and reasonable defaults.

Unit8

https://github.com/unit8co/darts

Darts

Discovery

- Statistical analysis
- Visualizations

Preprocessing

- Missing value interp.
- Normalizing, scaling
- Seas./trend removal

Forecasting

- Statistical models
- Deep-learning models
- Probabilistic forecasting
- Ensembling
- Multiple TS support (meta learning)
- Multivariate and covariate support

Model Evaluation and Selection

- Historical forecasting / backtesting
- Residual analysis
- Metrics
- Grid search

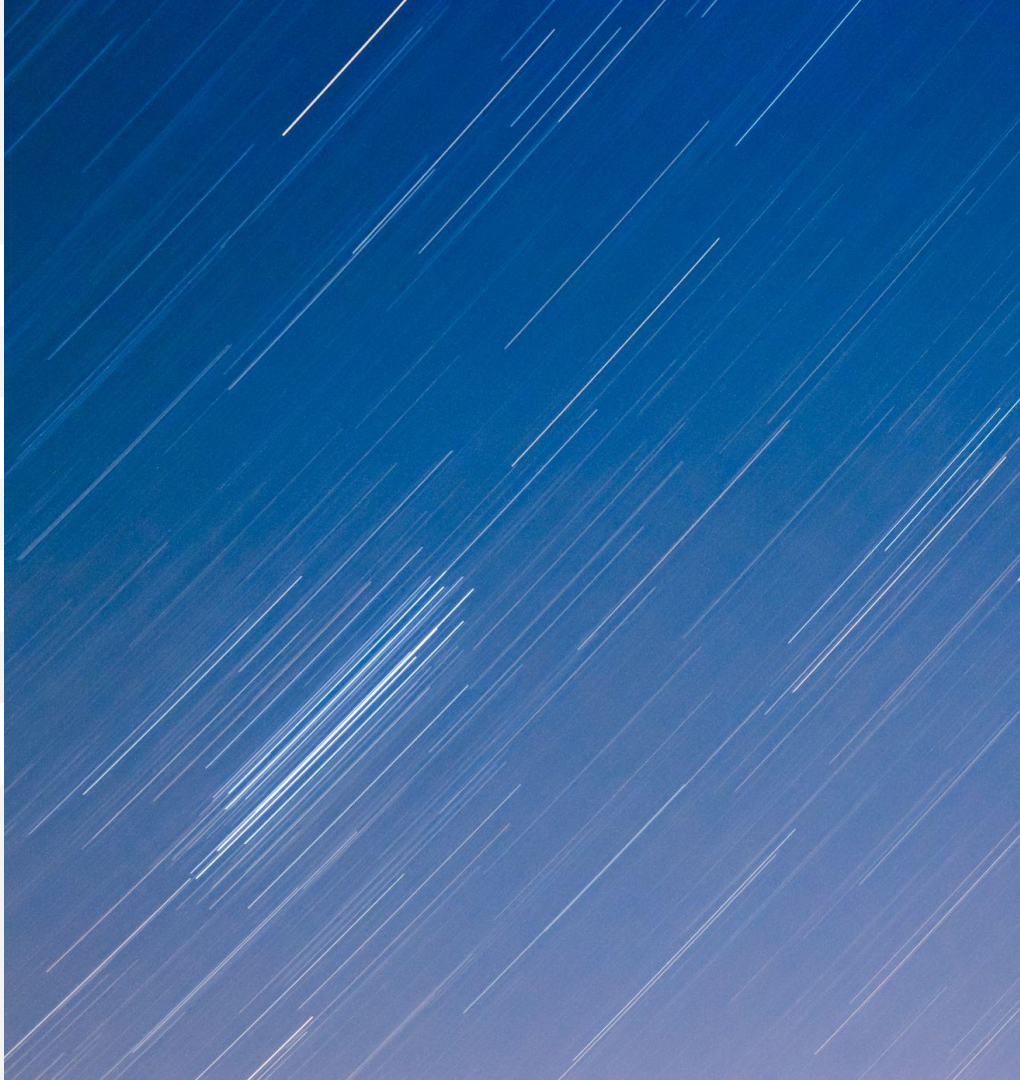
Unit8

1 Intro to Forecasting & Darts

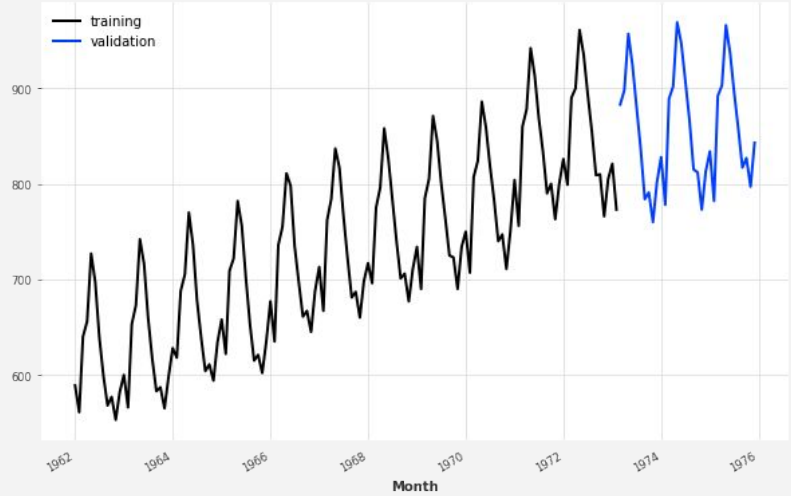
2 Basic forecasting using Darts

3 More advanced features

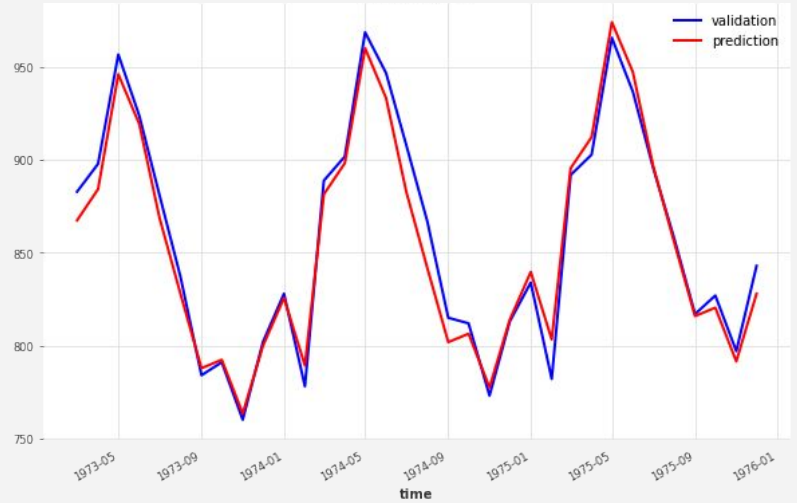
4 Conclusions & next steps



Darts Overview



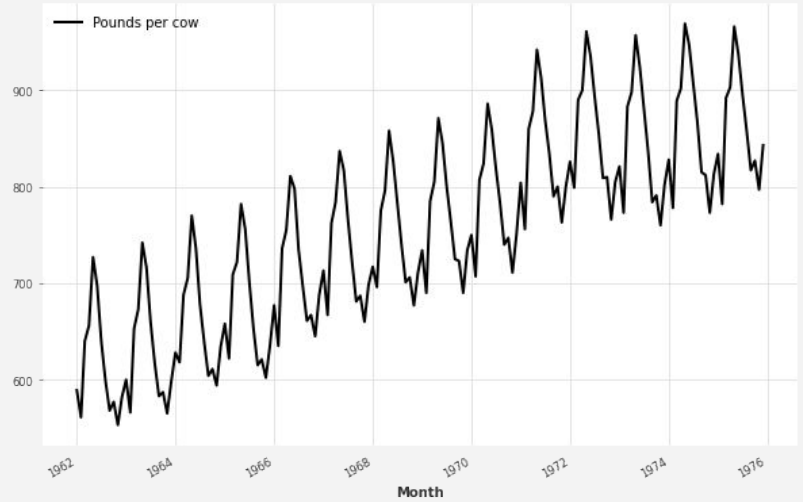
Goal



TimeSeries → Forecasting → Evaluating → Tuning

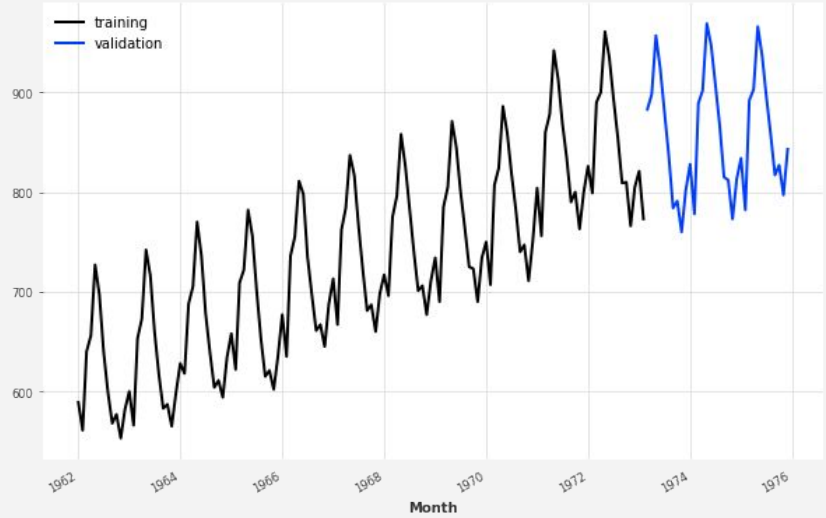
The TimeSeries object

```python
from darts import TimeSeries

series = TimeSeries.from_csv('monthly-milk.csv', time_col='Month')
series.plot()
```



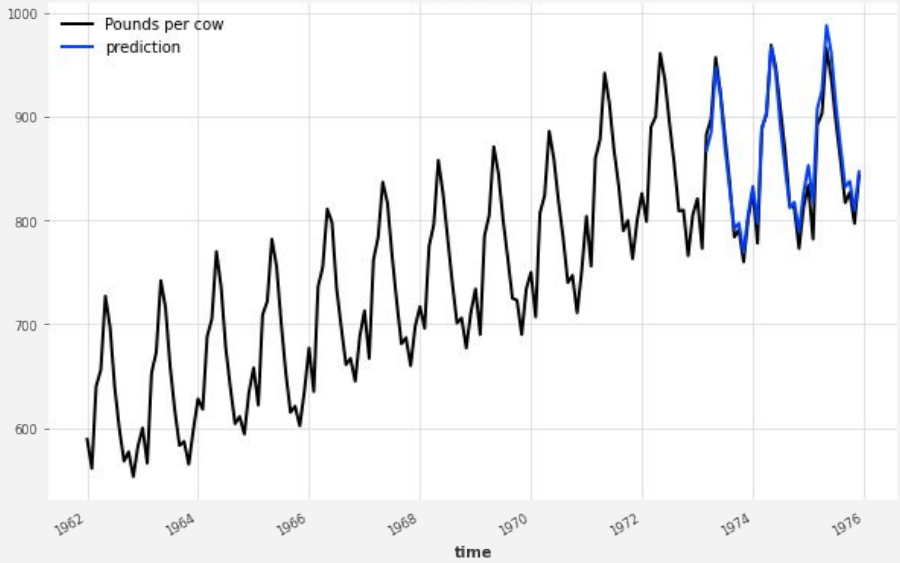
Training / validation split

```
training, validation = series.split_after(0.8)
```



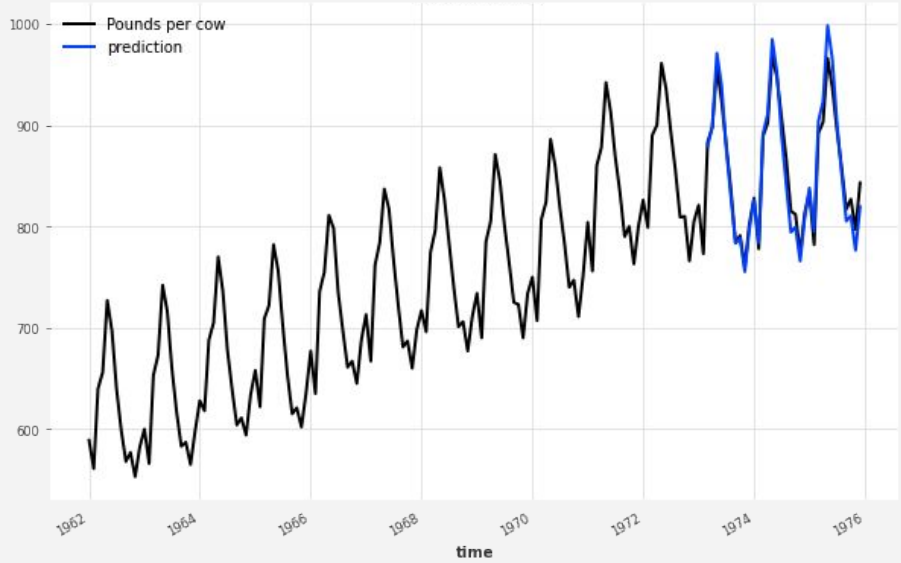
Forecasting – Exponential Smoothing

```python
from darts.models import ExponentialSmoothing

model = ExponentialSmoothing()
model.fit(training)
pred = model.predict(len(validation))
```



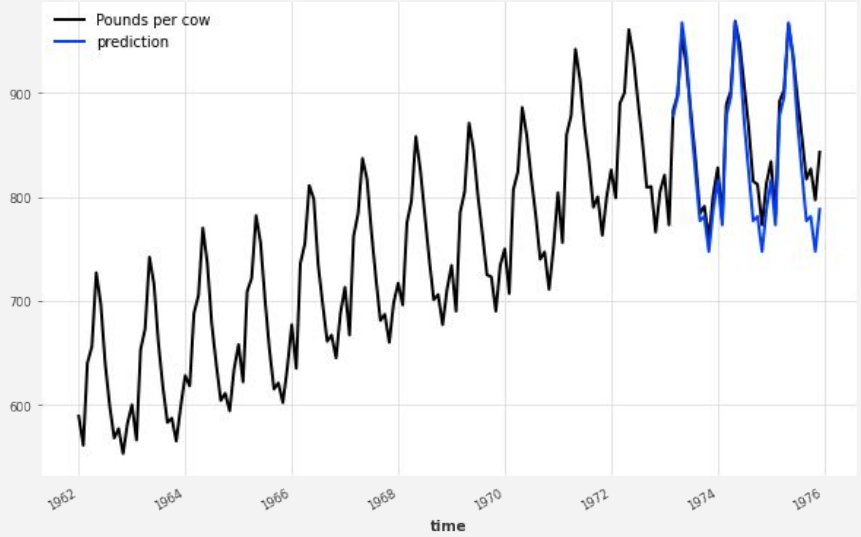
Forecasting – Theta

```python
from darts.models import Theta

model = Theta()
model.fit(training)
pred = model.predict(len(validation))
```



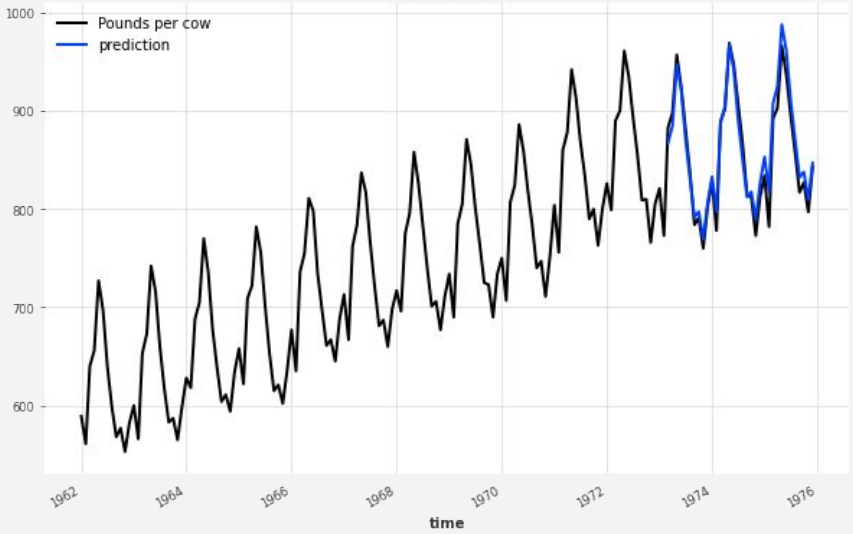
Specifying parameters

```
model = Theta(theta=1)
model.fit(training)
pred = model.predict(len(validation))
```

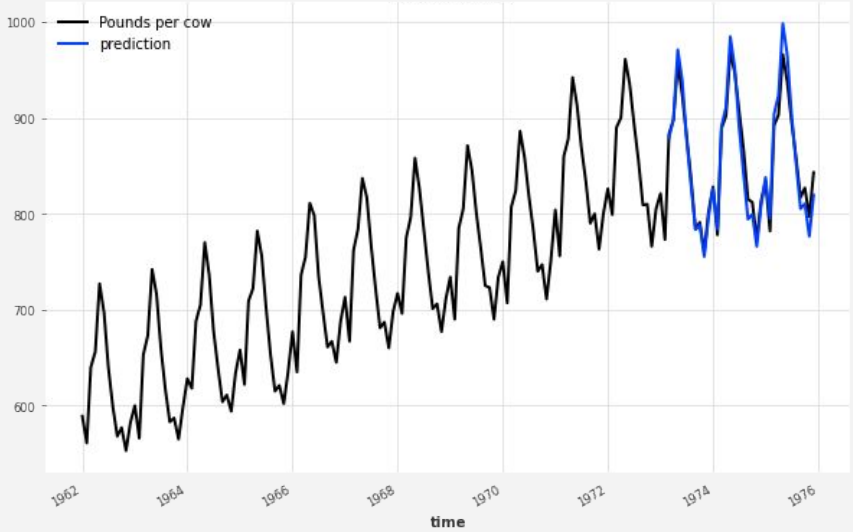


Evaluating predictions – Which one is better?

Exponential Smoothing



Theta



Metrics

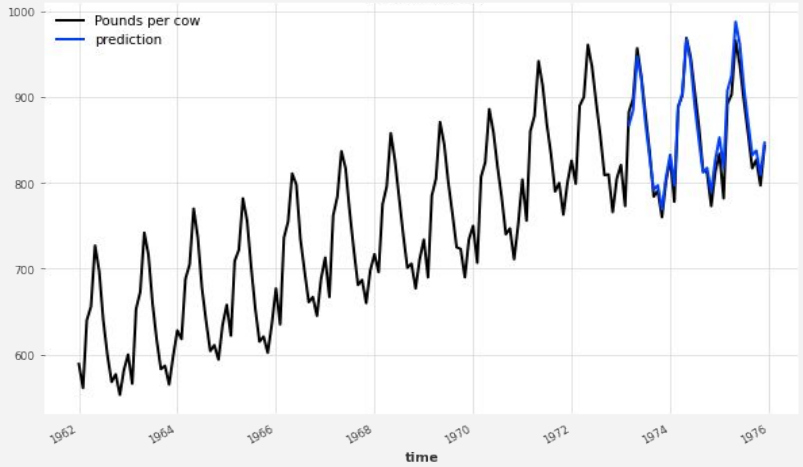
Many different scores can be computed – Darts lets you import the one you need.

```python
from darts.metrics import mape

score = mape(validation, pred)
```

```python
from darts.metrics import mase

score = mase(validation, pred, training)
```

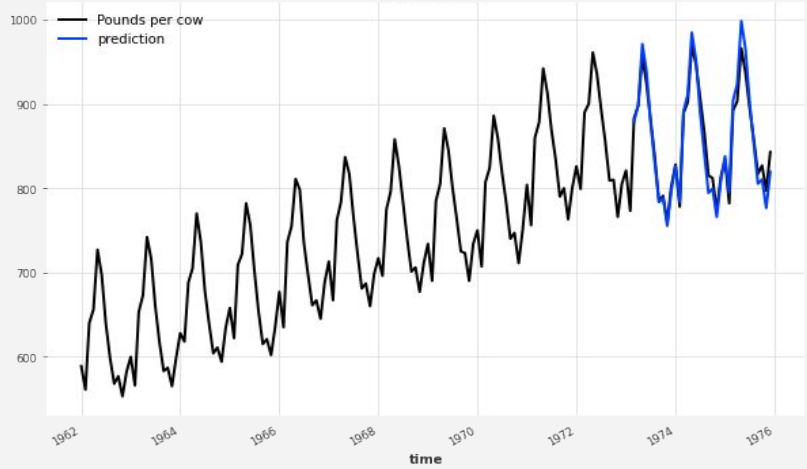
Which one is better?

Exponential Smoothing



MAPE: ~1.39%

Theta



MAPE: ~1.28%

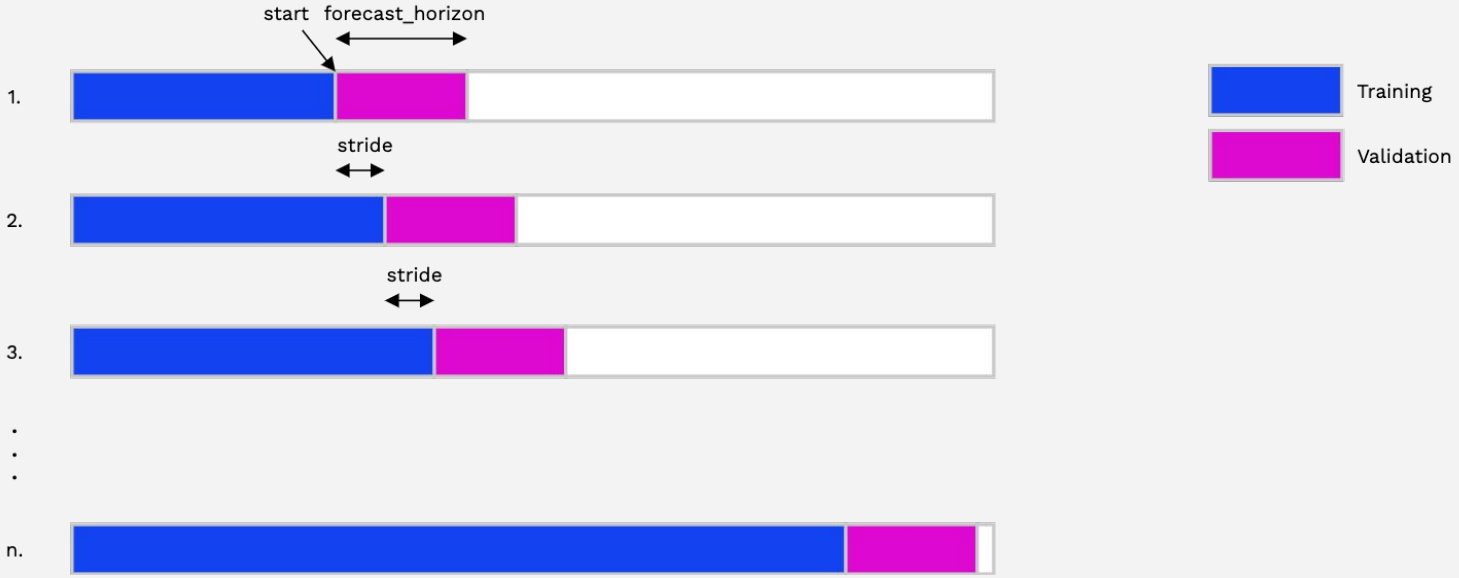
TimeSeries | Forecasting | **Evaluating** | Tuning

Evaluating model performance

`historical_forecasts()` and `backtest()`

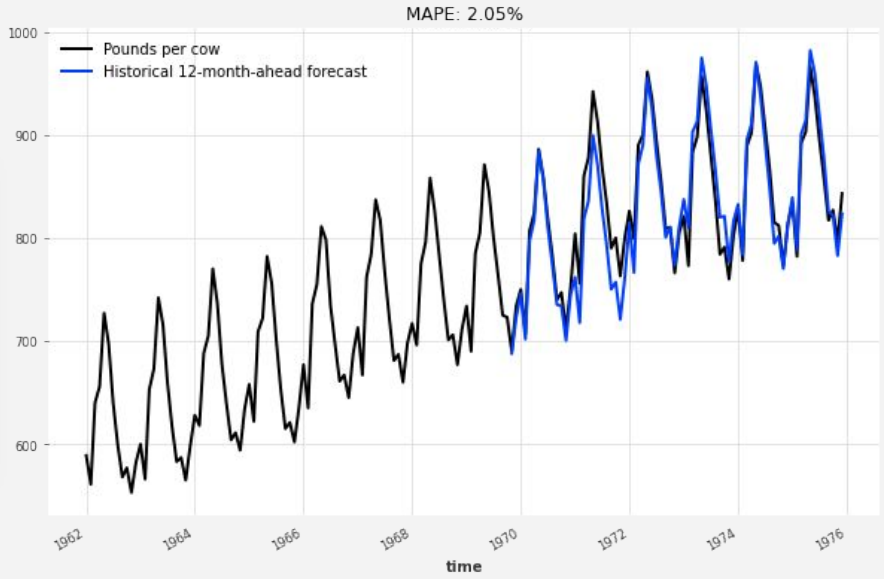
Simulate how a model **would have performed**
if it had been historically used to forecast a time series.

Predicting historical forecasts



Historical forecasts

```
historical_forecast = model.historical_forecasts(
    series=series,
    start=0.5,
    forecast_horizon=12
)
```



From evaluating to optimizing

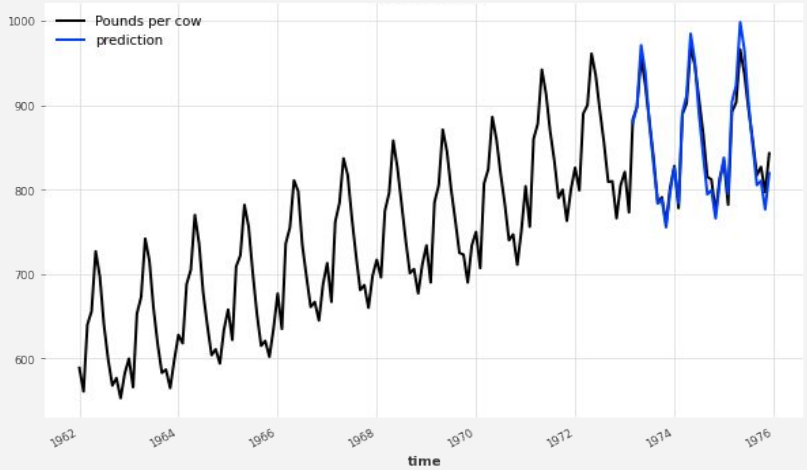
How can we find the best hyperparameters to maximize accuracy ?

Gridsearch

```
parameters = {
    'theta': [0.5, 1, 1.5, 2, 2.5],
    'season_mode': [SeasonalityMode.MULTIPLICATIVE,
                    SeasonalityMode.ADDITIVE]
}

best_model, best_parameters = Theta.gridsearch(
    parameters=parameters,
    series=training,
    start=0.5,
    forecast_horizon=12
)
```

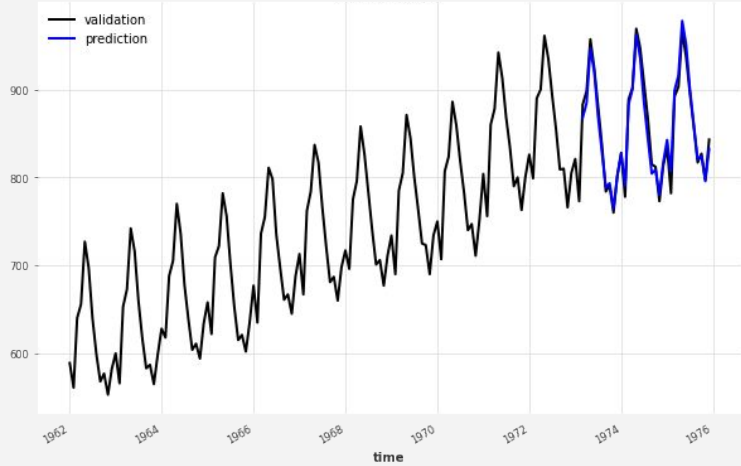
Gridsearch

`Theta(theta=2, season_mode=MULTIPLICATIVE)`



MAPE: ~1.28%

`Theta(theta=3, season_mode=ADDITIVE)`



MAPE: ~0.98%

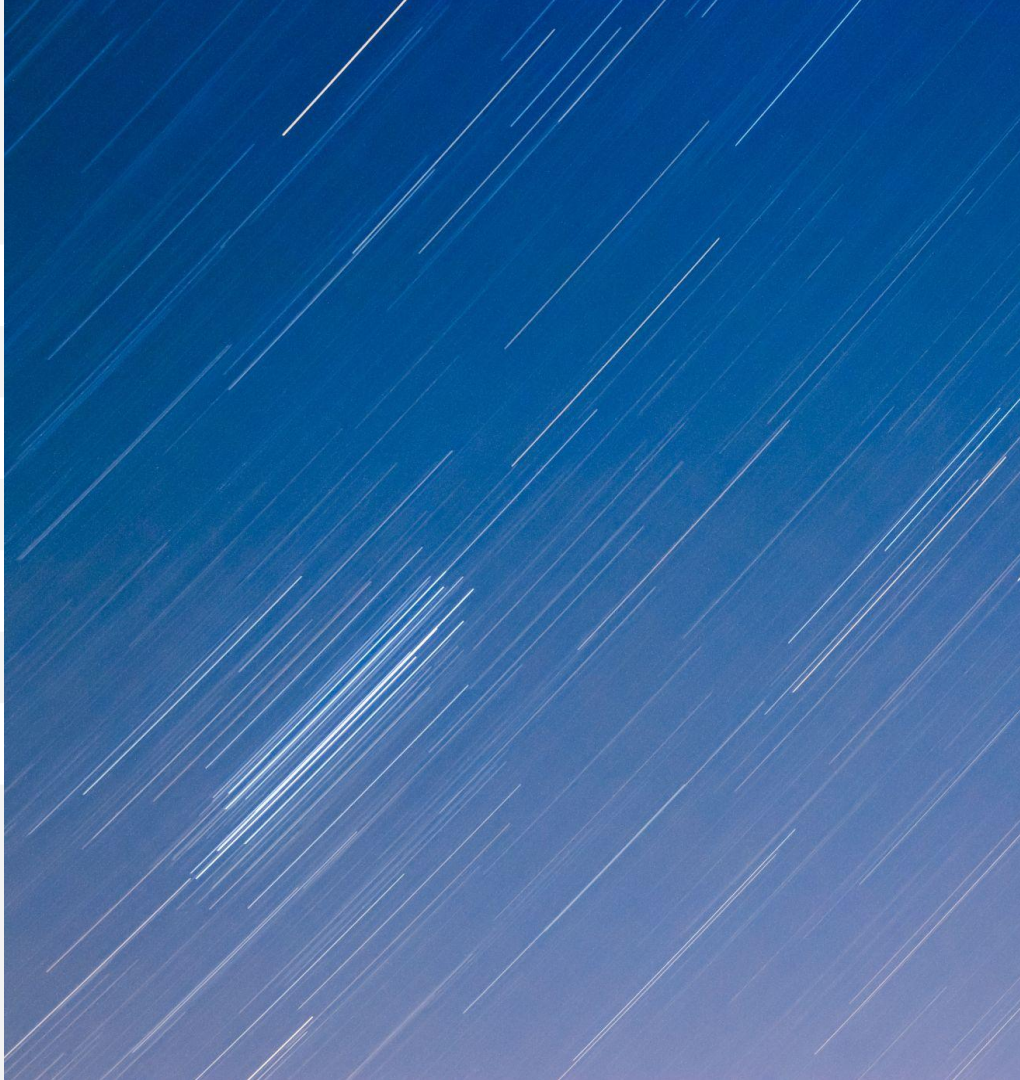
TimeSeries → Forecasting → Evaluating → **Tuning**

1 Intro to Forecasting & Darts

2 Basic forecasting using Darts

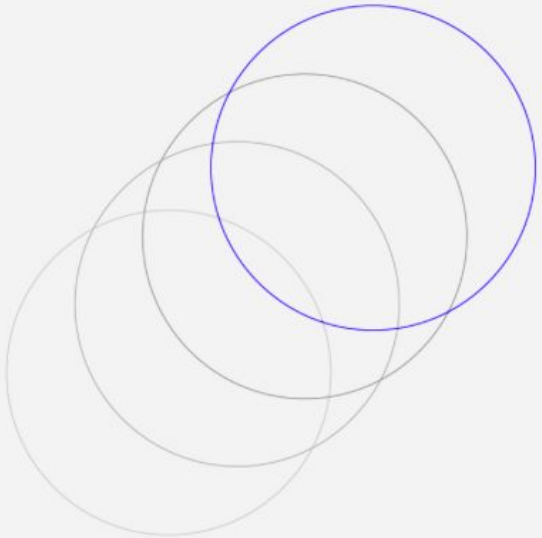
3 More advanced features

4 Conclusions & next steps



Modern Machine Learning for Time Series

- Classical ML & deep learning
- Training on multiple series & large datasets
- Support for multi-dimensional series
- Including external past & future data
- Probabilistic forecasting



Meta-learning on 48,000 monthly series (M4 dataset)

Unit8

Spyros Makridakis et al., “The M4 Competition: 100,000 time series and 61 forecasting methods”, International Journal of Forecasting, 2020.

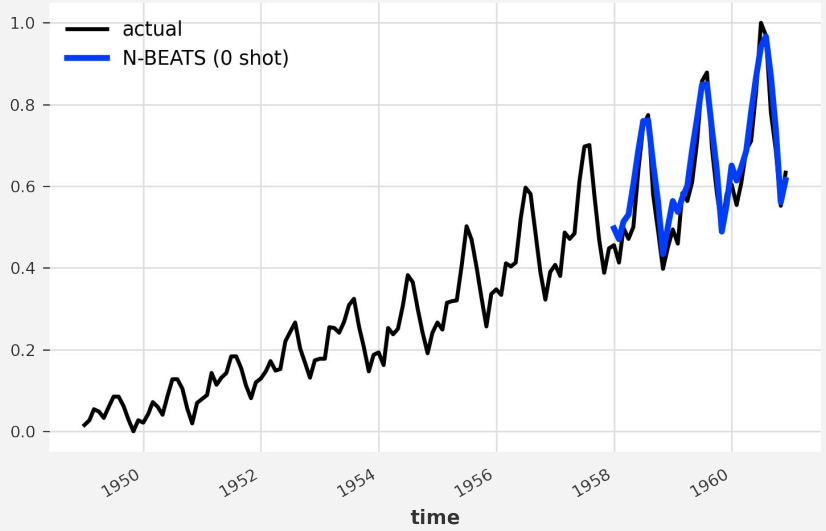
Meta-learning with N-BEATS

```python
from darts.models import NBEATSModel

model = NBEATSModel(**kwargs)
model.fit(all_train_series)

pred = model.predict(n=36, series=air_train)
```

Sequence of 48,000 TimeSeries

Zero shot forecasting

Inference on a series **never seen during training** (takes a few ms)



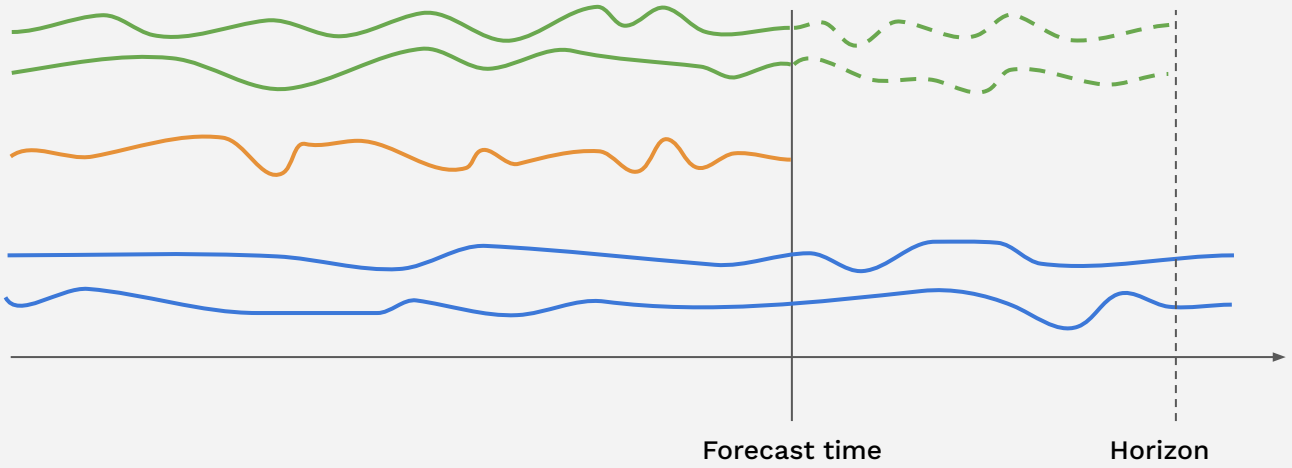
Unit8

Including Past & Future External Data

Target series: what we want to forecast

Past covariates: Unknown into the future e.g. measurements

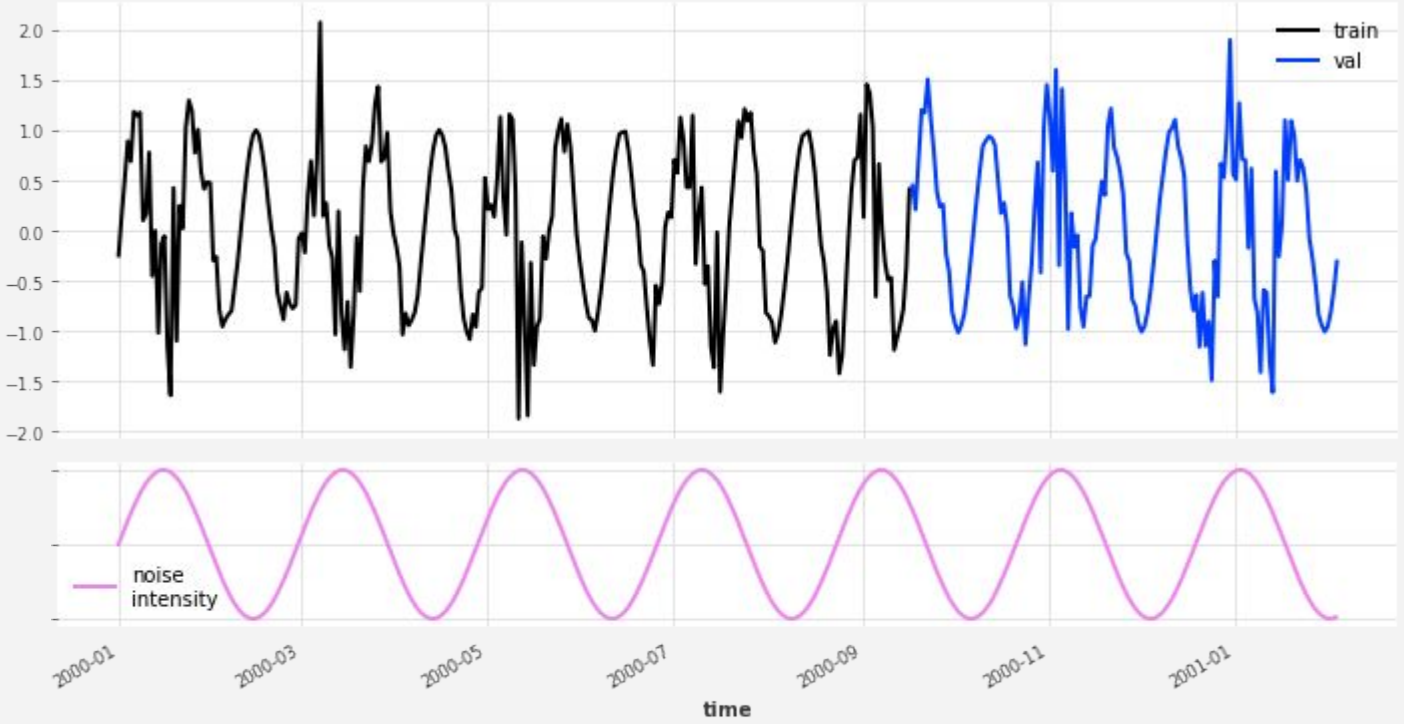
Future covariates: Known into the future e.g. calendar, weather forecasts, actions



- `fit()` and `predict()` can accept `past_covariates` and/or `future_covariates`, depending on model.
- If `future_covariates` are given, future values will be required at inference time.
- Alignment of covariates with target is automatic.

Unit8

Probabilistic Forecasts

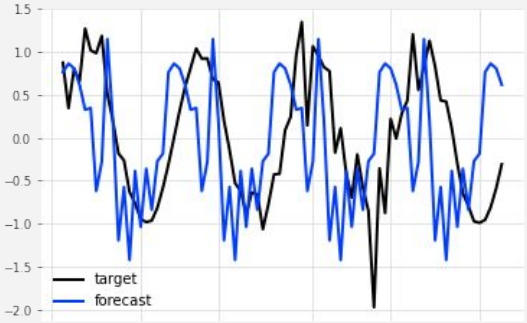


Unit8

Capturing Series Stochasticity

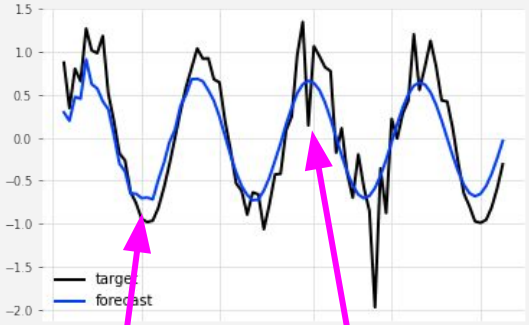
Attempt 1

```
model = NaiveSeasonal(seasonal_period)
model.fit(train)
pred = model.predict(n)
```



Attempt 2

```
model = ARIMA(seasonal_period, 0, 0)
model.fit(train)
pred =  model.predict(n)
```

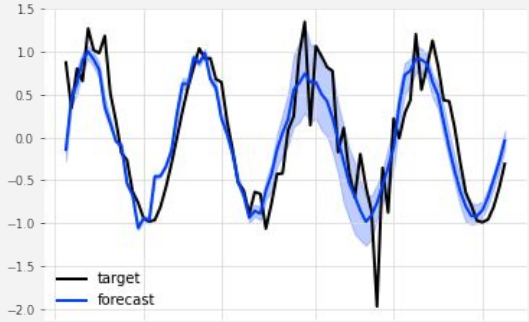


"safe" forecast

uncertain forecast

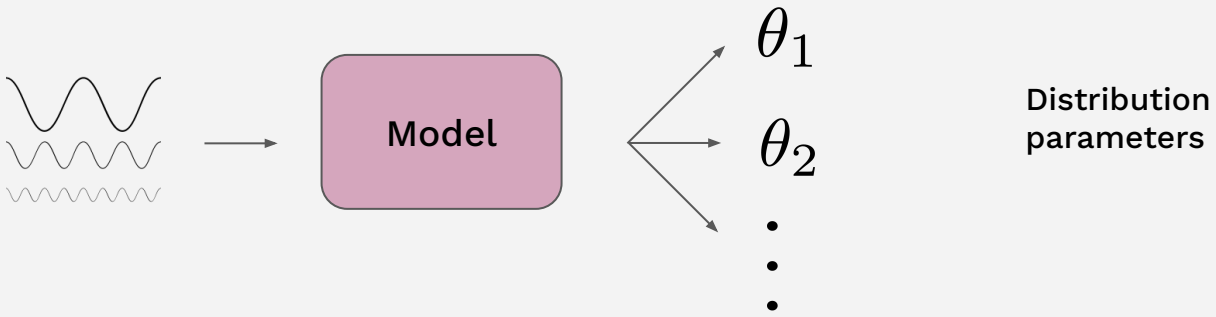
Attempt 3

```
model = TCNModel(likelihood=GaussianLikelihood(),
                 **kwargs)
model.fit(train, past_covariates=noise_intensity)
pred = model.predict(n,
                     past_covariates=noise_intensity,
                     num_samples=100)
```

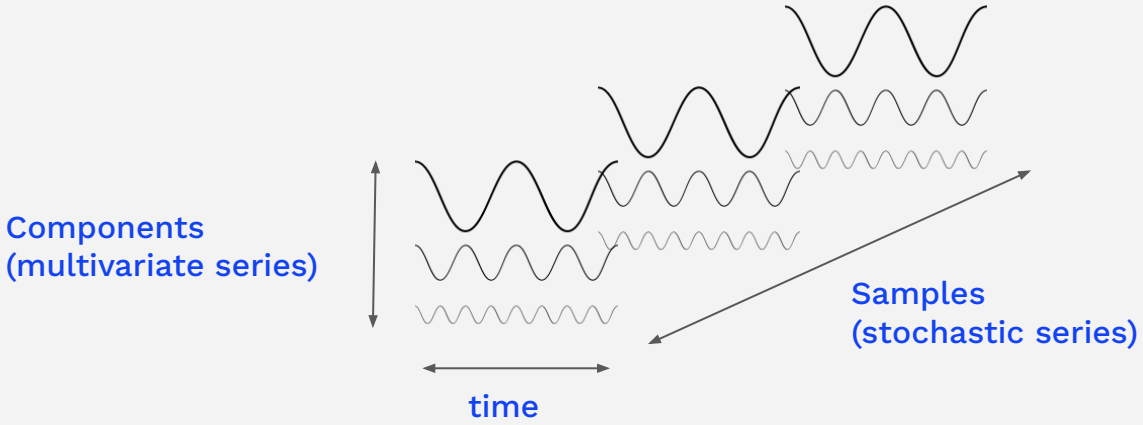


Unit8

Probabilistic forecasts

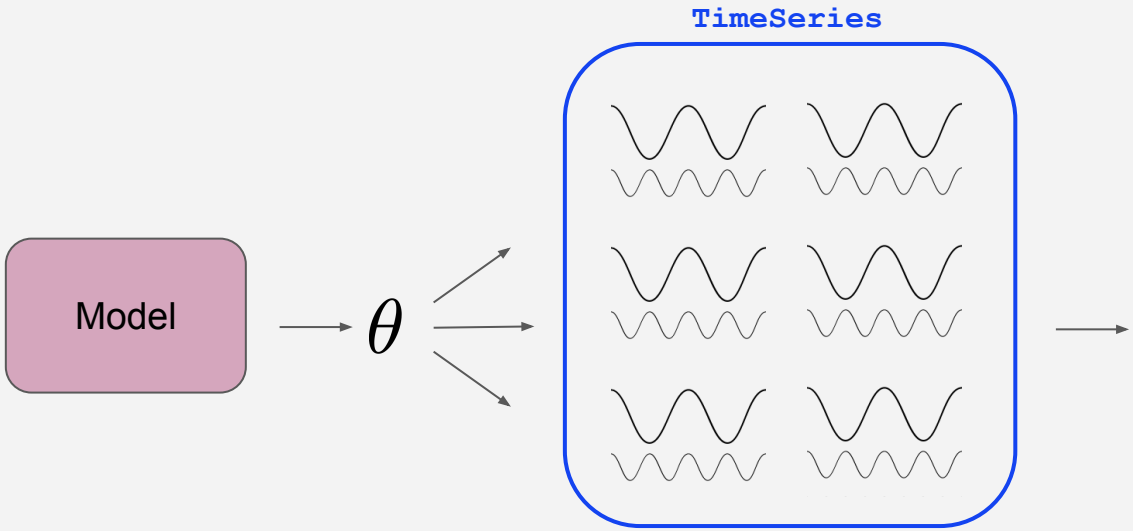


A `TimeSeries` contains 3 dimensions

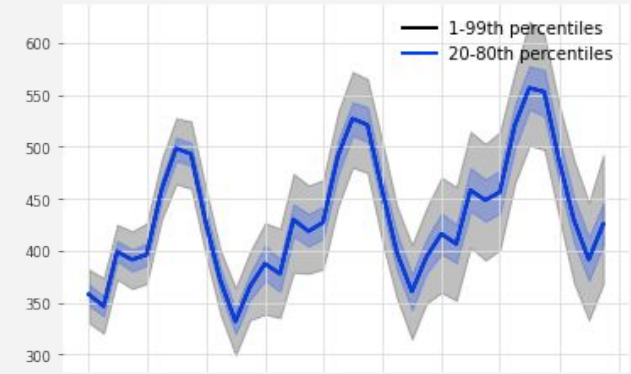


Unit8

Probabilistic forecasts



Probabilistic time series
(distribution-agnostic)




```
forecast.plot(low_quantile=0.01, high_quantile=0.99)
forecast.plot(low_quantile=0.2, high_quantile=0.8)
```

Confidence intervals

Unit8

Probabilistic forecasts

```
from darts.utils.likelihood_models import (GaussianLikelihood,
                                           PoissonLikelihood,
                                           NegativeBinomialLikelihood,
                                           BernoulliLikelihood,
                                           GammaLikelihood,
                                           GumbelLikelihood,
                                           LaplaceLikelihood,
                                           BetaLikelihood,
                                           ExponentialLikelihood,
                                           DirichletLikelihood,
                                           GeometricLikelihood,
                                           CauchyLikelihood,
                                           ContinuousBernoulliLikelihood,
                                           HalfNormalLikelihood,
                                           LogNormalLikelihood,
                                           WeibullLikelihood)
```

\+ Priors on distributions' parameters

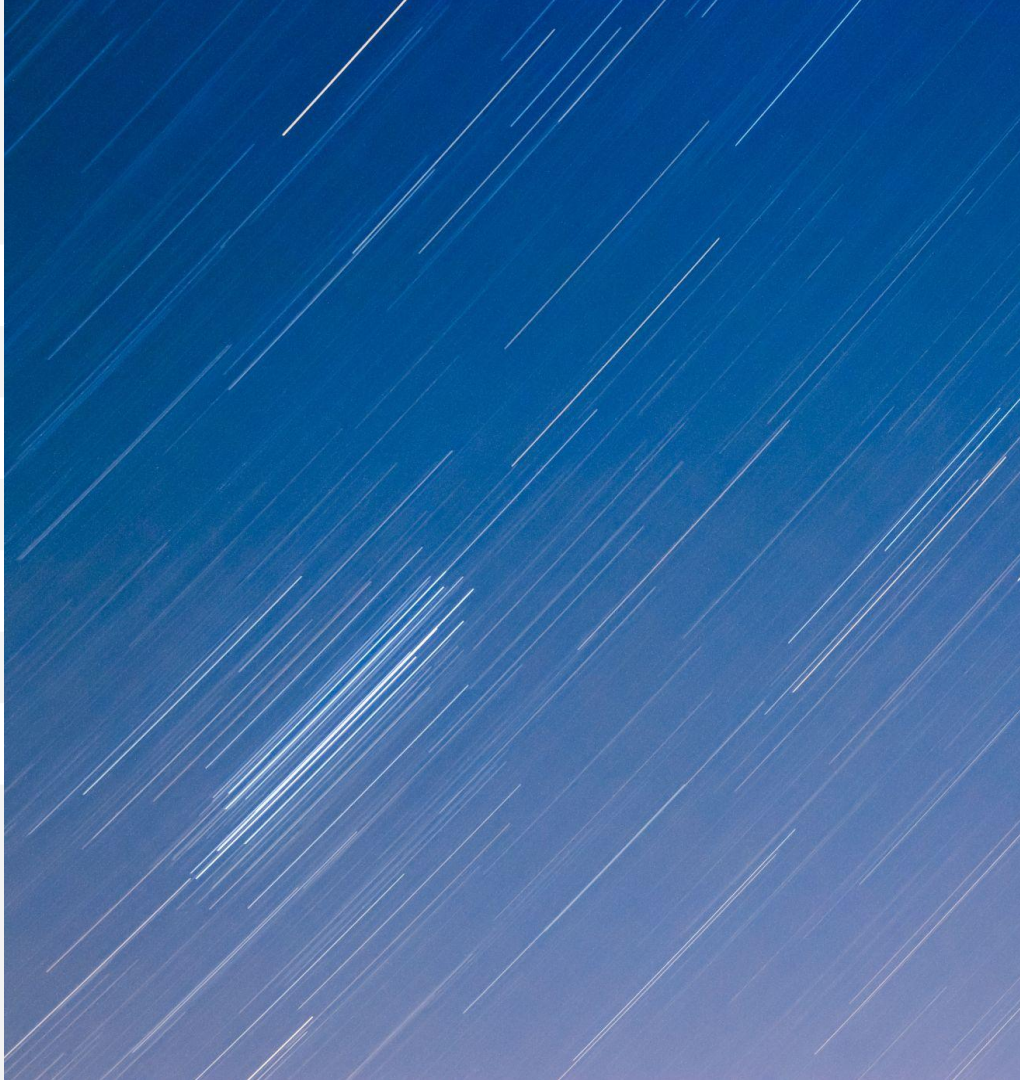
Unit8

1 Intro to Forecasting & Darts

2 Basic forecasting using Darts

3 More advanced features

4 Conclusions & next steps



Darts: User-friendly & Modern ML for Time Series

Try it out for yourself :)

- `pip install darts`
- https://github.com/unit8co/darts/

Some next steps:

- Anomaly detection
- Static covariates
- AutoML
- Pre-trained models
- ...

Contact: info@unit8.co. We're always happy to discuss time series problems!

Unit8

unit8.co

Francesco
francesco.laessig@unit8.co
–
Julien
julien@unit8.co

thank you