Darts
Unifying time series forecasting models from ARIMA to Deep Learning

Europython 2021
Francesco: Data Scientist @ Unit8
- One of the main contributors to Darts.

Gaël: Data Scientist @ Unit8
- 6 years experience in industry in Data science and Machine Learning
1. Intro to Forecasting & Darts
2. Forecasting basics
3. Training on multiple time-series
4. Probabilistic forecasting
5. Try Darts!
Time series are everywhere!
Why Darts?

- Lack of unified library in Python for time series forecasting
- To create a useful tool for ourselves
How did Darts come about?

- **2018/09**
  - Initial commit by hrzn committed on 13 Sep 2018

- **2020/06**
  - First public release: Darts 0.1.0

- **Today**
  - Current version: Darts 0.9.1

- **End of 2021 and onwards**
  - ...
1 Intro to Forecasting & Darts
2 Forecasting basics
3 Training on multiple time-series
4 Probabilistic forecasting
5 Try Darts!
Darts Overview

Goal

MAPE = 0.84%

TimeSeries  Forecasting  Evaluating
The **TimeSeries** object

```python
from darts import TimeSeries

df = pd.read_csv('monthly-milk.csv')
series = TimeSeries.from_dataframe(df,
    time_col='Month',
    value_cols=['Pounds per cow'])
```
Training / validation split

```
training, validation = series.split_before(pd.Timestamp('1973-01-01'))
```
Forecasting – Exponential Smoothing

```python
from darts.models import ExponentialSmoothing
model = ExponentialSmoothing()
model.fit(training)
forecast = model.predict(len(validation))
```
Forecasting – **Theta**

```python
from darts.models import Theta
model = Theta()
model.fit(training)
forecast = model.predict(len(validation))
```
Specifying **parameters**

```python
from darts.models import Theta
from darts import SeasonalityMode

model = Theta(theta=2,
               seasonality_period=12,
               season_mode=SeasonalityMode.MULTIPLICATIVE)

model.fit(training)
forecast = model.predict(len(validation))
```
Evaluating predictions – Which one is better?
Metrics

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

```python
from darts.metrics import mape
score = mape(validation, forecast)
```

```python
from darts.metrics import mase
score = mase(validation, forecast, training)
```
Which one is better?

MAPE: ~3.44%

MAPE: ~2.42%
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.
Historical forecasts

![Diagram showing historical forecasts with training, validation, start, forecast_horizon, and stride labels.]

**TimeSeries**  
**Forecasting**  
**Evaluating**
Historical forecasts

```python
forecasts = model.historical_forecasts(training_series=series,
                                        start=0.5,
                                        forecast_horizon=12,
                                        stride=5,
                                        last_points_only=False)
```
Backtesting

```python
backtest_errors = model.backtest(training_series=series,
                               start=0.5,
                               forecast_horizon=12,
                               stride=1,
                               last_points_only=False,
                               metric=mape,
                               reduction=None)
```
Backtesting

```python
import numpy as np

mean_error = model.backtest(training_series=series,
                          start=0.5,
                          forecast_horizon=12,
                          stride=1,
                          last_points_only=False,
                          metric=mape,
                          reduction=np.mean)
```

Individual error scores (histogram)
The `last_points_only` parameter
From evaluating to optimizing

How can we find the best hyperparameters to maximize accuracy?
Gridsearch

```python
parameters = {
    "theta": [0.5, 1, 1.5, 2, 2.5, 3],
    "season_mode": [SeasonalityMode.MULTIPLICATIVE, SeasonalityMode.ADDITIVE]
}

best_model = Theta.gridsearch(parameters=parameters,
                                training_series=training,
                                forecast_horizon=12,
                                start=0.5,
                                last_points_only=False,
                                metric=mape,
                                reduction=np.mean)

best_model.fit(training)
best_model_forecast = best_model.predict(len(validation))
```
Gridsearch

Default model
(theta = 2, SeasonalityMode.MULTIPLICATIVE)

Best model
(theta = 3, SeasonalityMode.ADDITIVE)

MAPE: ~2.42%

MAPE: ~2.32%
1. Intro to Forecasting & Darts
2. Forecasting basics
3. Training on multiple time-series
4. Probabilistic forecasting
5. Probabilistic forecasting
Supported data types

**Known**
- Input (one/multi timestep)
- Output or target, $dt > 1$ multi horizon

**Forecast (unknown)**
- Meta learning

**Covariate ts**
- Can be also known in the future (weekday etc)

- Univariate ts
- Multivariate ts
Meta-learning on multiple time series

want to predict

Could this help?
Meta-learning on multiple time series

Train on air traffic data only

```python
model_air = NBEATSModel(**kwargs)
model_air.fit(train_air)
pred = model_air.predict(n)
```

MAPE: 8.9%

Train on air traffic and milk production data

```python
model_air_milk = NBEATSModel(**kwargs)
model_air_milk.fit([train_air, train_milk])
pred = model_air_milk.predict(n, serlies=train_air)
```

MAPE: 5.5%
Try Darts!

Forecasting basics

Intro to Forecasting & Darts

Training on multiple time-series

4 Probabilistic forecasting

5 Try Darts!
Unpredictable components in time series
Unpredictable components in time series

Attempt 1

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

Attempt 2

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

Attempt 3

```python
model = TCRMModel(likelihood=GaussianLikelihoodModel(), **kwargs)
model.fit(train, covariates)
pred = model.predict(n, covariates=covariates, num_samples=100)
```

.safe forecast

.uncertain forecast
Probabilistic forecasts

Deterministic forecasting model

Probabilistic forecasting model
Probabilistic forecasts

Model \( \theta \) \rightarrow \text{Probabilistic time series (distribution-agnostic)} \rightarrow \text{Confidence intervals}

```python
forecast.plot(low_quantile=0.01, high_quantile=0.99)
forecast.plot(low_quantile=0.2, high_quantile=0.8)
```
<table>
<thead>
<tr>
<th></th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Intro to Forecasting &amp; Darts</td>
</tr>
<tr>
<td>2</td>
<td>Forecasting basics</td>
</tr>
<tr>
<td>3</td>
<td>Training on multiple time-series</td>
</tr>
<tr>
<td>4</td>
<td>Probabilistic forecasting</td>
</tr>
<tr>
<td>5</td>
<td><strong>Try Darts!</strong></td>
</tr>
</tbody>
</table>
Darts

Discovery
- ACF plots
- Seasonality/trend checks
- Visualisations
- Naive models

Preprocessing
- Normalization
- Interpolation of missing values
- Seasonality/trend removal
- Up/downsampling

Forecasting models
- Classic models: ARIMA, ETS, Theta, ...
- Neural nets: RNN/TCN
- Facebook Prophet
- ... and many others!

Model Selection and Evaluation
- Backtesting
- Residual analysis
- Grid search
- Metrics
If you want to try darts, here are some steps!

Check out the library yourself! As easy as: `pip install darts`

Look through one of our tutorial notebooks or intro blog post

- [https://github.com/unit8co/darts/tree/master/examples](https://github.com/unit8co/darts/tree/master/examples)

Contacting us directly on github or via: [info@unit8.co](mailto:info@unit8.co). We’re always happy to answer questions or discuss time series problems!
thank you
Base Data Transformer
Box-Cox Transformer
Fittable Data Transformer
Invertible Data Transformer
Mappers
Missing Values Filler
Scaler

Models
- ARIMA
- Auto-ARIMA
- Baseline Models
- Block Recurrent Neural Networks
- Ensemble Model Base Class
- Exponential Smoothing
- Fast Fourier Transform
- Filtering Model Base Class
- Forecasting Model Base Classes
- Standard Regression model
- N-BEATS
- Facebook Prophet
- Random Forest
- Regression ensemble model
- Regression Model
- Recurrent Neural Networks
- Temporal Convolutional Network
- Theta Method
- Torch Forecasting Model Base Class
- Transformer Model
- VARIMA

Utils
- TimeSeries Datasets
- Likelihood Models
- Utils for filling missing values
- Model selection utilities
- Utils for time series statistics
- Utils for time series generation
- Utils for Pytorch and its usage
- Additional util functions
Intro to DARTS
Our motivations

- While working on multiple time series projects we noticed a lack of a unified framework for forecasting and analysis.
- First started to create a useful tool for ourselves.
- Contribute and give back to open-source community:

  github.com/unit8co/darts
Time series are everywhere!

Finance
- Financial & sales forecasting
- Cash flow forecasting

Business
- Revenue forecasting
- Web traffic forecasting

Climate
- Energy usage forecasting
- Temperature forecasting

Healthcare
- Hospitalization forecasting
- Regularity of heart beats

...