Unit8™

Workshop: Forecasting time series

VisCon, 12 October 2019

digital natives

unit8.co

About us

Krzysiek & Kilian

Software Engineers @ Unit8

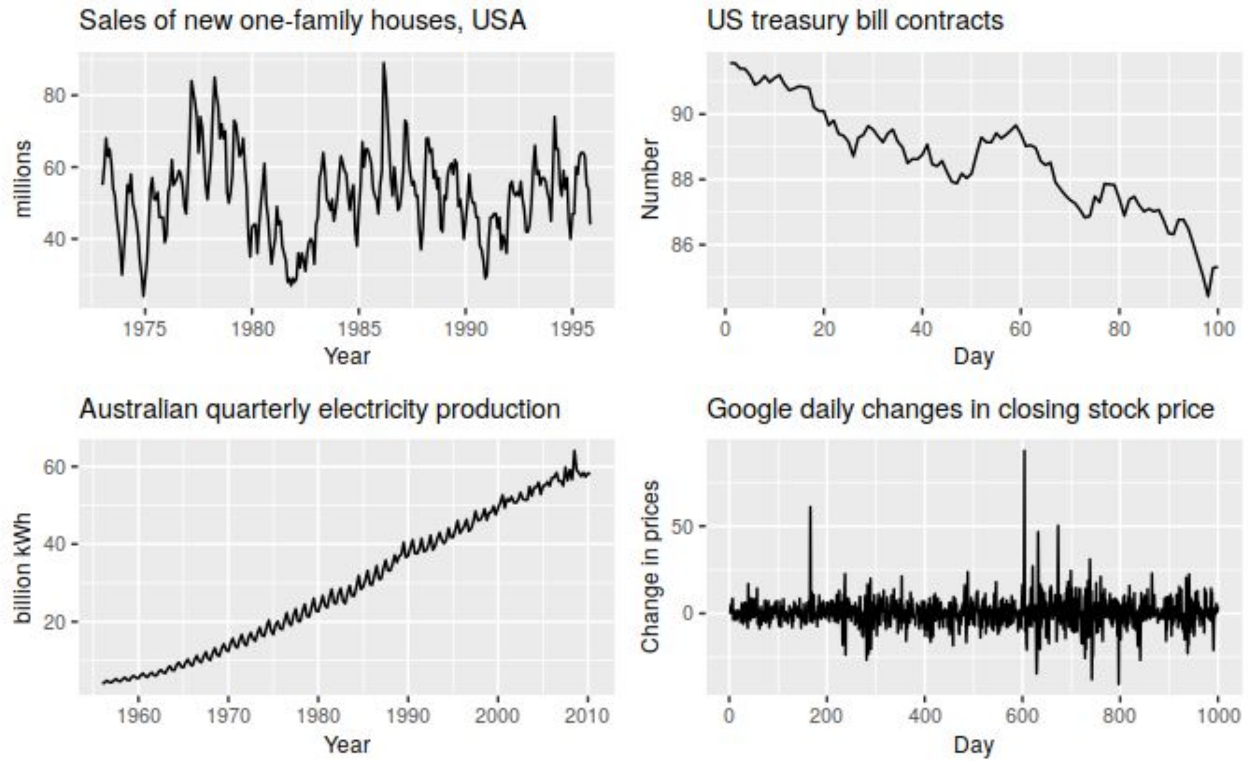
Unit8™

Before we begin... Let's setup!

https://github.com/unit8co/u8timeseries-viscon

Why Time Series Forecasting?

Time Series Examples



The Big Picture

Everyone needs forecasting!

Predicting electricity demand

Predicting product sales

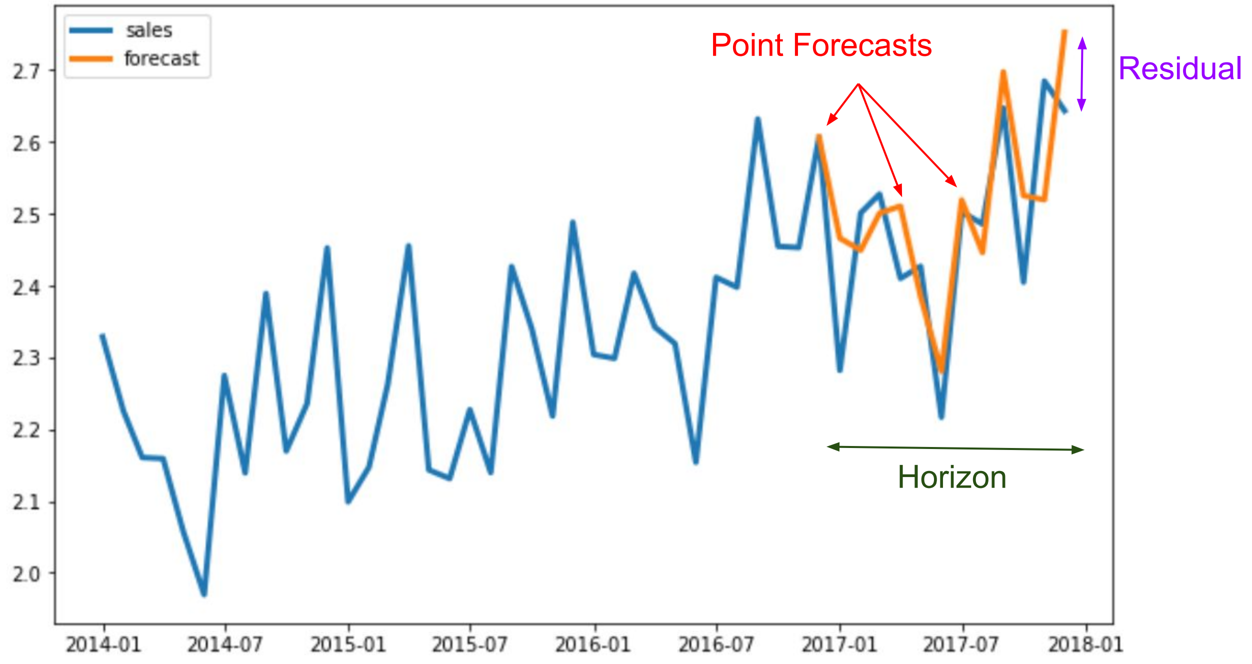
Predicting taxi rides demand

... and many more

Unit8™

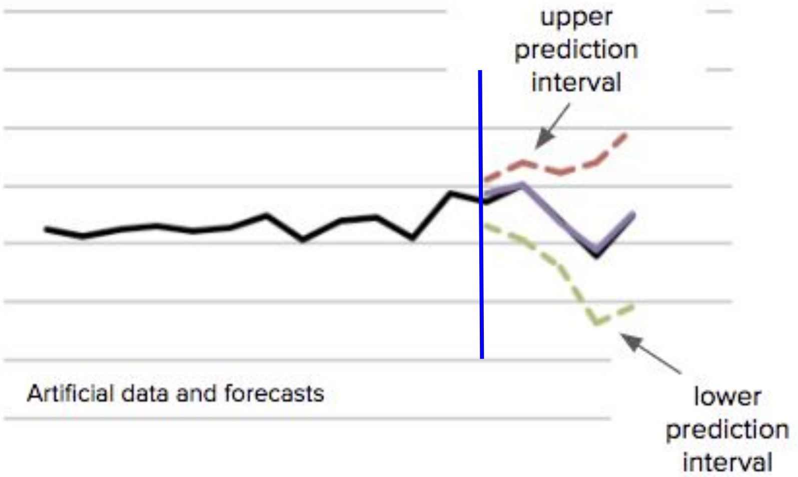
Time Series Concepts

Point Forecasts, Horizon, Residuals

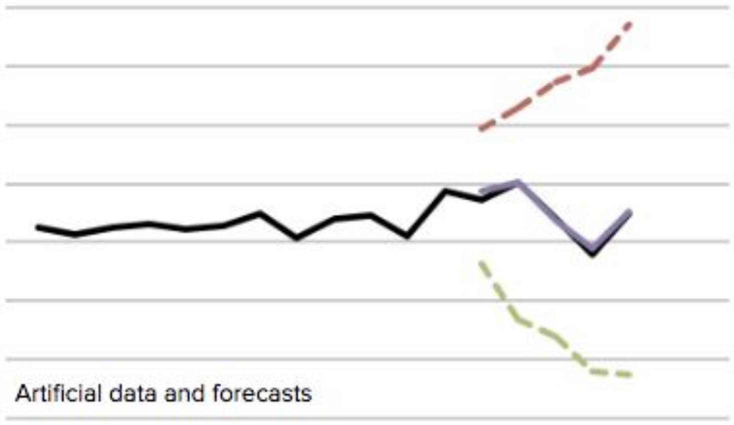


Unit8™

Prediction Intervals



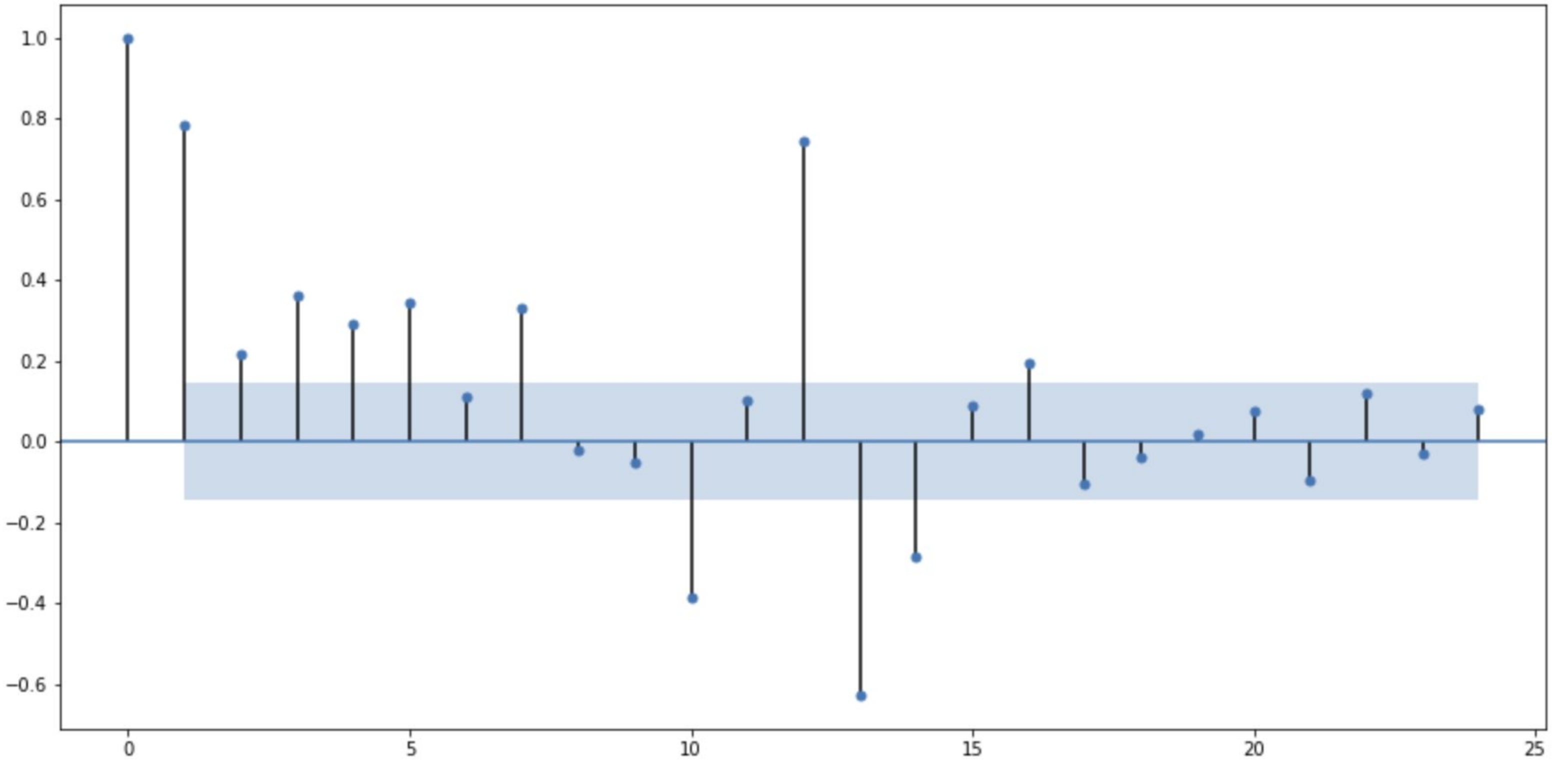
VS.



We're 95% sure that the value is going to be within

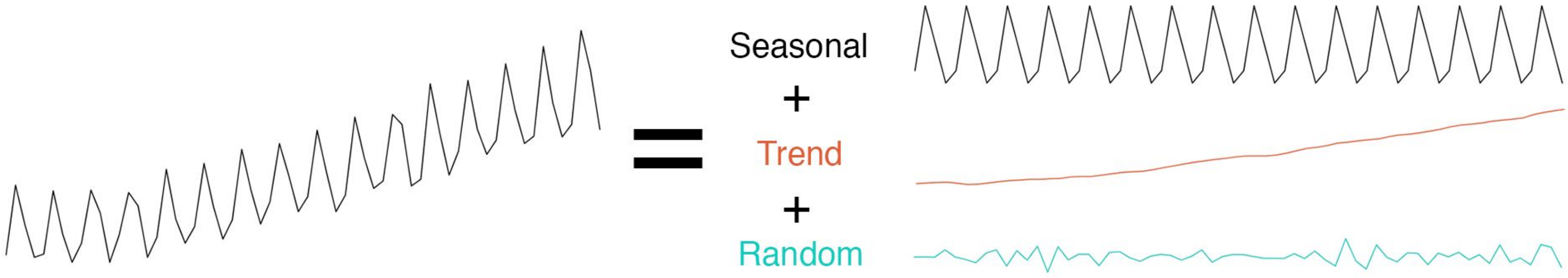
Unit8™

Autocorrelation



Unit8™

Seasonal Trend Decomposition

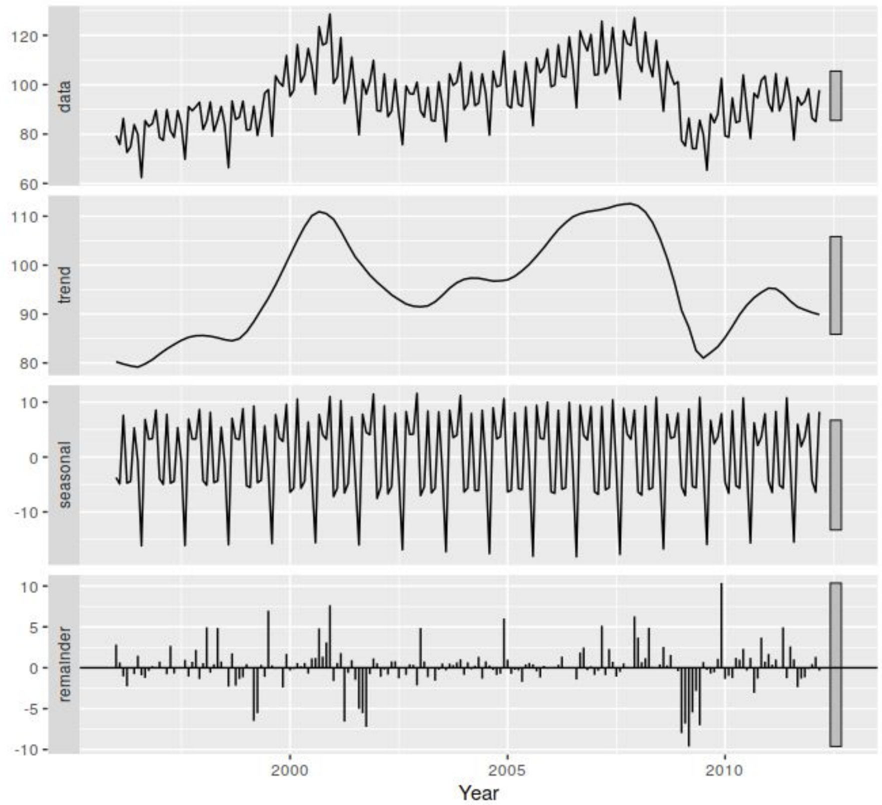


Additive:
Time series = Seasonal + Trend + Random

Multiplicative:
Time series = Trend * Seasonal *Random

Unit8™

More complex trend & seasonality



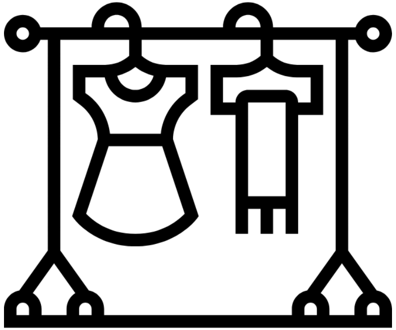
Unit8

Time for Task 1

The dataset

RetailNZ.csv

Monthly total clothing retail sales
Value in dollars
Timespan: May 1995 - September 2010



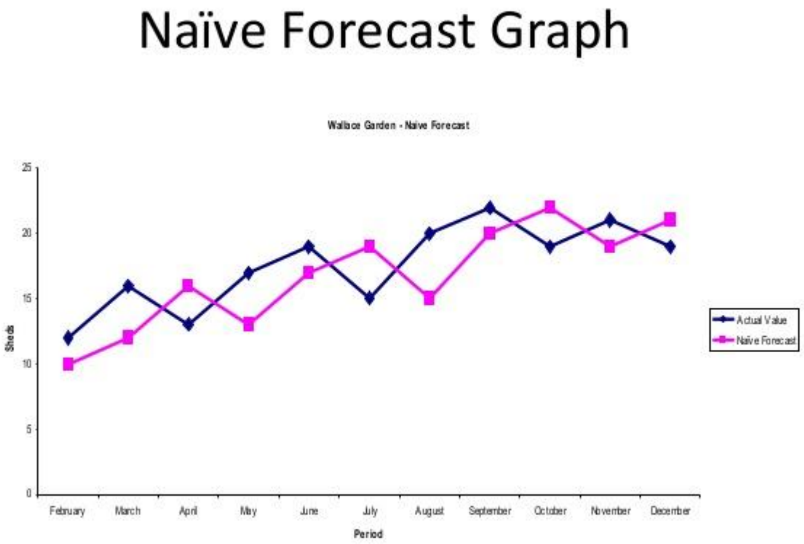
Let’s code!

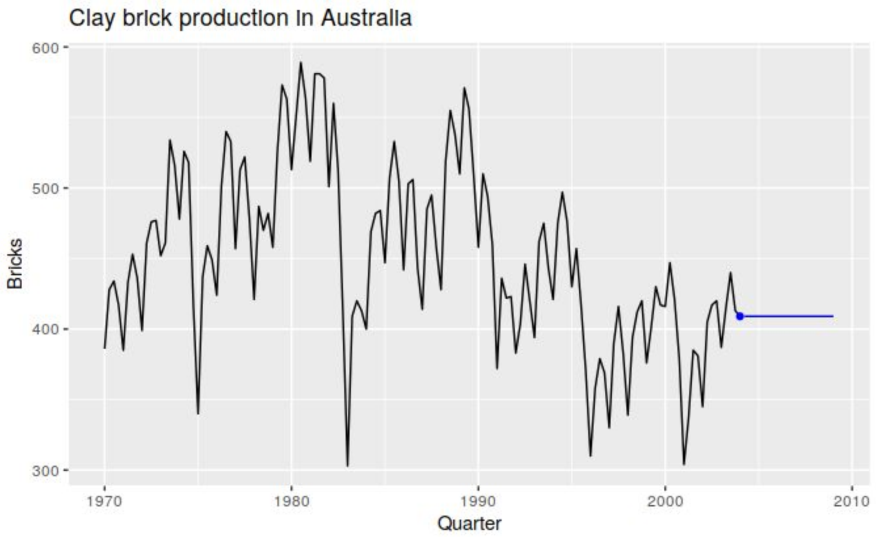
Task 1 in *1_explore_and_forecast_task1-3.ipynb* notebook

Naive

Naive

Just take the last value (or previous year value) as prediction of next value.





Unit8™

Exponential Smoothing

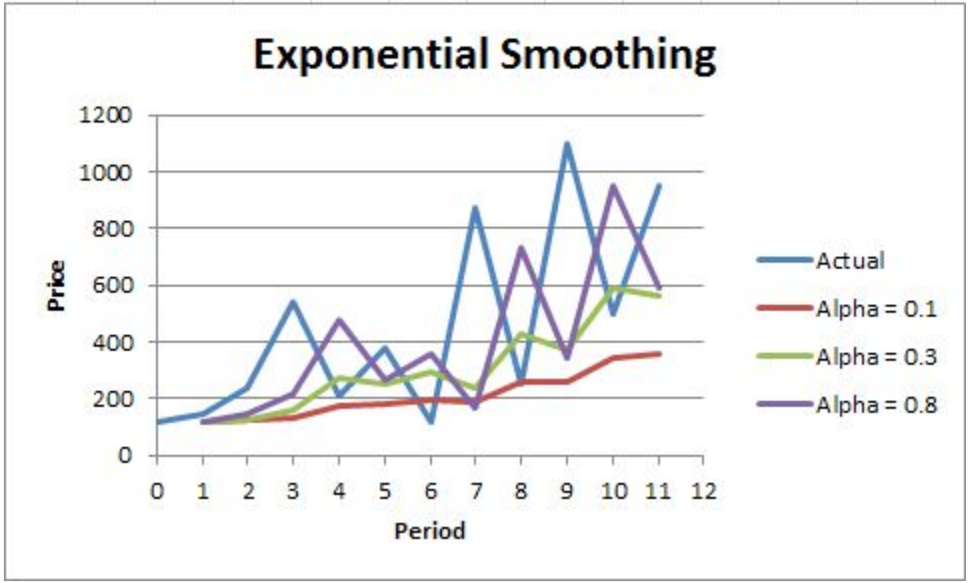
Exponential Smoothing Intuition

Let’s mix a weighted sum of:

- ★ **Previous value**
- ★ **Our previous estimation**

$$\hat{y}_{t+1|t} = \boxed{\alpha y_t} + \boxed{(1-\alpha)\hat{y}_{t|t-1},}$$

Smoothing coefficients
fitted to dataset



Unit8

Exponential Smoothing

There's lots of different methods in this family….

Trend Component	Seasonal Component		
	N (None)	A (Additive)	M (Multiplicative)
N (None)	(N,N)	(N,A)	(N,M)
A (Additive)	(A,N)	(A,A)	(A,M)
A_d (Additive damped)	(A_d,N)	(A_d,A)	(A_d,M)

Simple Exponential Smoothing (SES)

Holt's Method

Holt-Winters' Method

Exponential Smoothing

How to choose the model?

Follow ES modeling procedure

Better to automate the process!

```r
library(forecast)
model <- ets(dataset)
future <- forecast(model, h=3)
```

`ets()` brute-forces all possible models to find the one that best fits the dataset

Unit8™

ARIMA

ARIMA = AR + I + MA

Let's mix together:

- ★ previous p values of the serie (AR)
- ★ previous q residuals of the serie (MA)
- ★ single or double differencing (d) to get rid of seasonality & trend (I) $y'_t = y_t - y_{t-1}$

Autoregressive AR(p) Moving Average MA(q)

$$X_t = c + \varepsilon_t + \boxed{\sum_{i=1}^{p} \varphi_i \boxed{X_{t-i}}} + \boxed{\sum_{i=1}^{q} \theta_i \boxed{\varepsilon_{t-i}}}.$$

Differencing I(d)

Weights that need to be fitted
on the dataset

ARIMA = AR(p) + I(d) + MA(q)

How to choose p, d & q?

Follow ARIMA modeling procedure

Better to automate the process!

```
library(forecast)
model <- auto.arima(dataset)
future <- forecast(model, h=3)
```

`auto.arima()` brute-forces the search space of (p,d,q) that optimizes AIC, AICc or BIC value.

Time for Task 2

Error metrics

- Mean Absolute Percentage Error (MAPE)

$$\mathrm{M} = \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|,$$

- Symmetric MAPE (sMAPE)

$$\mathrm{SMAPE} = \frac{100\%}{n} \sum_{t=1}^{n} \frac{|F_t - A_t|}{|A_t| + |F_t|}$$

- Mean Absolute Scaled Error (MASE)

$$\mathrm{MASE} = \mathrm{mean}\left(\frac{|e_j|}{\frac{1}{T-1} \sum_{t=2}^{T} |Y_t - Y_{t-1}|} \right) = \frac{\frac{1}{J} \sum_j |e_j|}{\frac{1}{T-1} \sum_{t=2}^{T} |Y_t - Y_{t-1}|}$$

Unit8™

Time for Task 3

Which model is the best?

M3 Competition (2000) & NN3 (2006)

Can be expressed as Simple Exponential Smoothing (SES) with drift

- 3003 series
- Data from business, finance & economics
- Seasonality: yearly, monthly, daily, hourly
- Series length between 14 and 126

Method	MAPE	sMAPE	MASE
Theta	17.42	12.76	1.39
ForecastPro	18.00	13.06	1.47
B-J automatic	19.13	13.72	1.54
ETS	17.38	13.13	1.43
AutoARIMA	19.12	13.85	1.47

- Conclusions:
  - Complex methods not necessarily better than simpler ones
  - Methods ranking varies depending on accuracy metric & forecasting horizon
  - ML & Neural Networks unable to get comparable results

u8timeseries

u8timeseries

- Specialized software written by Unit8
- Easy to use collection of forecasting models in Python
- Unified API, on top of *pandas*
- Auto-regressive and regressive models
- Utilities to manipulate time series

Unit8™

Time for Task 4

Ensembling

M4 Competition (2018)

- 100K series
- Objective: compare ML vs statistical
- Point Forecast + Prediction Interval

Ranking	Type of Method	sMAPE	MASE	OWA
1st Ranked: Best	Hybrid	11.374	1.536	0.821
2nd Ranked: 2nd Best	Best Combining	11.720	1.551	0.838
8th Ranked	Best Statistical	11.986	1.601	0.861
19th Ranked	Comb Benchmark	12.555	1.663	0.898
25th Ranked	Best ML	12.894	1.682	0.915
37th Ranked	Naïve 2 Benchmark	13.564	1.912	1.000
48th Ranked	2nd Best ML	16.638	2.056	1.151

- Conclusions:
  - Combination of methods is the king - 12/17 of best methods
  - Surprising winner: hybrid approach ES-RNN submitted by Slawek Smyl from Uber
  - Pure ML methods performed poorly

M4 Competition (2018)

Significant margin

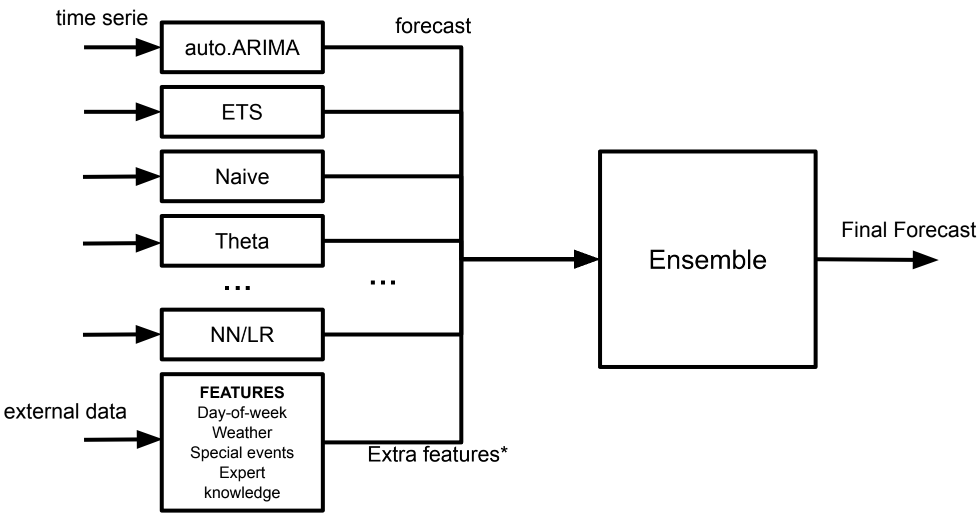
Rank	Team	Affiliation	Method	sMAPE	MASE	OWA	Diff from Comb (%)
1	Smyl	Uber Technologies	Hybrid	11.37	1.54	0.821	-8.52
2	Montero-Manso et al.	University of A Coruña & Monash University	Comb (S & ML)	11.72	1.55	0.838	-6.65
3	Pawlikowski et al.	ProLogistica Soft	Comb (S)	11.84	1.55	0.841	-6.25
4	Jaganathan & Prakash	Individual	Comb (S & ML)	11.70	1.57	0.842	-6.17
5	Fiorucci, J. A. & Louzada	University of Brasilia & University of São Paulo	Comb (S)	11.84	1.55	0.843	-6.10
6	Petropoulos & Svetunkov	University of Bath & Lancaster University	Comb (S)	11.89	1.57	0.848	-5.55

All submitted methods open sourced on GitHub:
https://github.com/M4Competition/M4-methods

All M4-Conference papers available online:
https://www.mcompetitions.unic.ac.cy/presentations/

Unit8

Ensembling



Different ways to combine:

- Simple Average
- Weighted Average by accuracy
- Learned weights (LR, RF, XGBoost)

Benefits:

- Improved accuracy due to additional information
- Safety in numbers - reduced risk of bad forecast
- Ability to incorporate external features

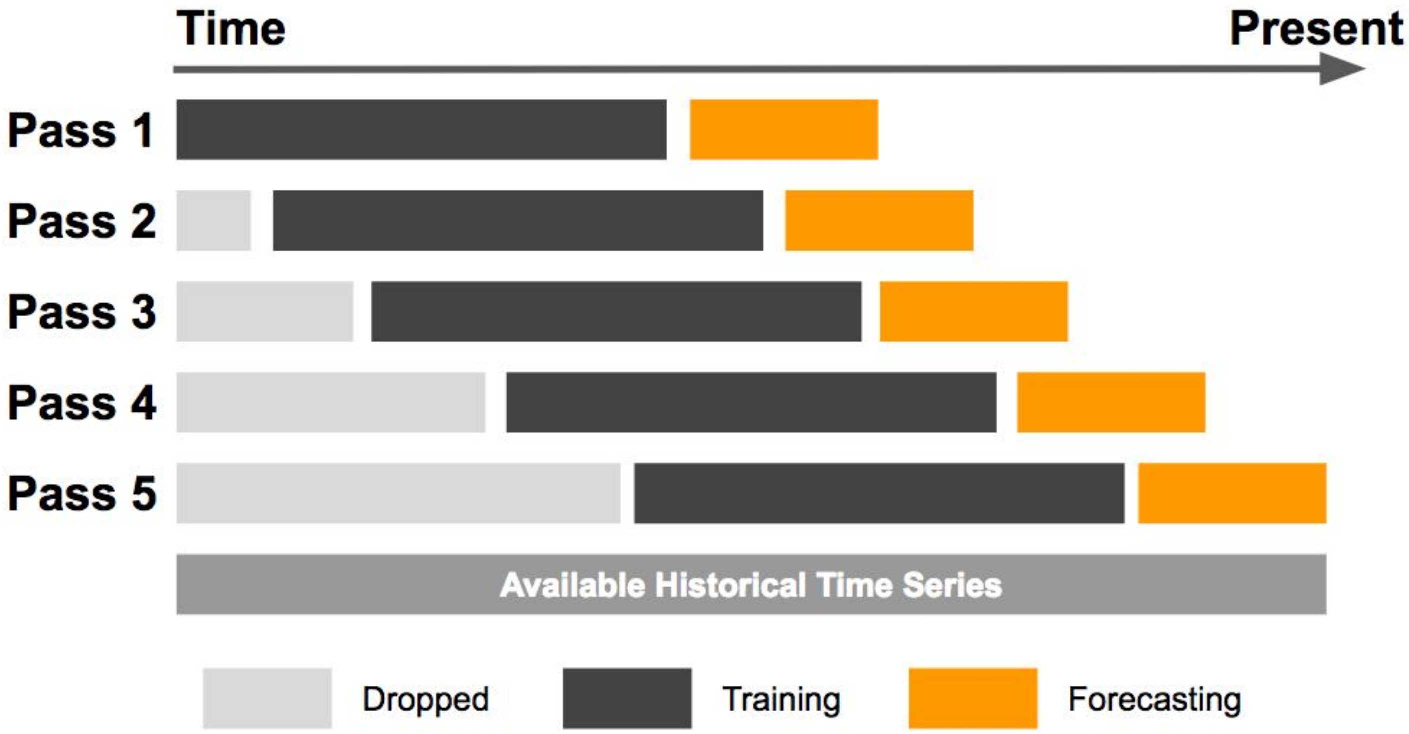
Backtesting

Backtesting

Concept: simulate predictions that would have been obtained historically with certain model(s)

1. Set prediction time $t \leftarrow t_0$
2. Produce point prediction for a fixed time horizon
3. Move pointer $t \leftarrow t+1$
4. Repeat

Backtesting - test your model over time



aka Time Series Cross-Validation

Unit8

Time for Task 5

Let’s code!

Workshop u8timeseries. Jupyter notebook.

Tasks:

- Time series forecasting using u8timeseries
- Backtesting
- Ensemble predictions

Unit8™

Forecasting Research

In Search of Single Forecasting Model

Statistical methods work very well but:

- Infeasible for millions of time series (Uber, Amazon)
- Require frequent retraining
- Do not benefit from cross-learning

Benefits of single ML model:

- Difficult to train & get right, but easy to maintain
- Allow cross-learning & can accomodate for new time series

Hybrid ES-RNN (Uber)

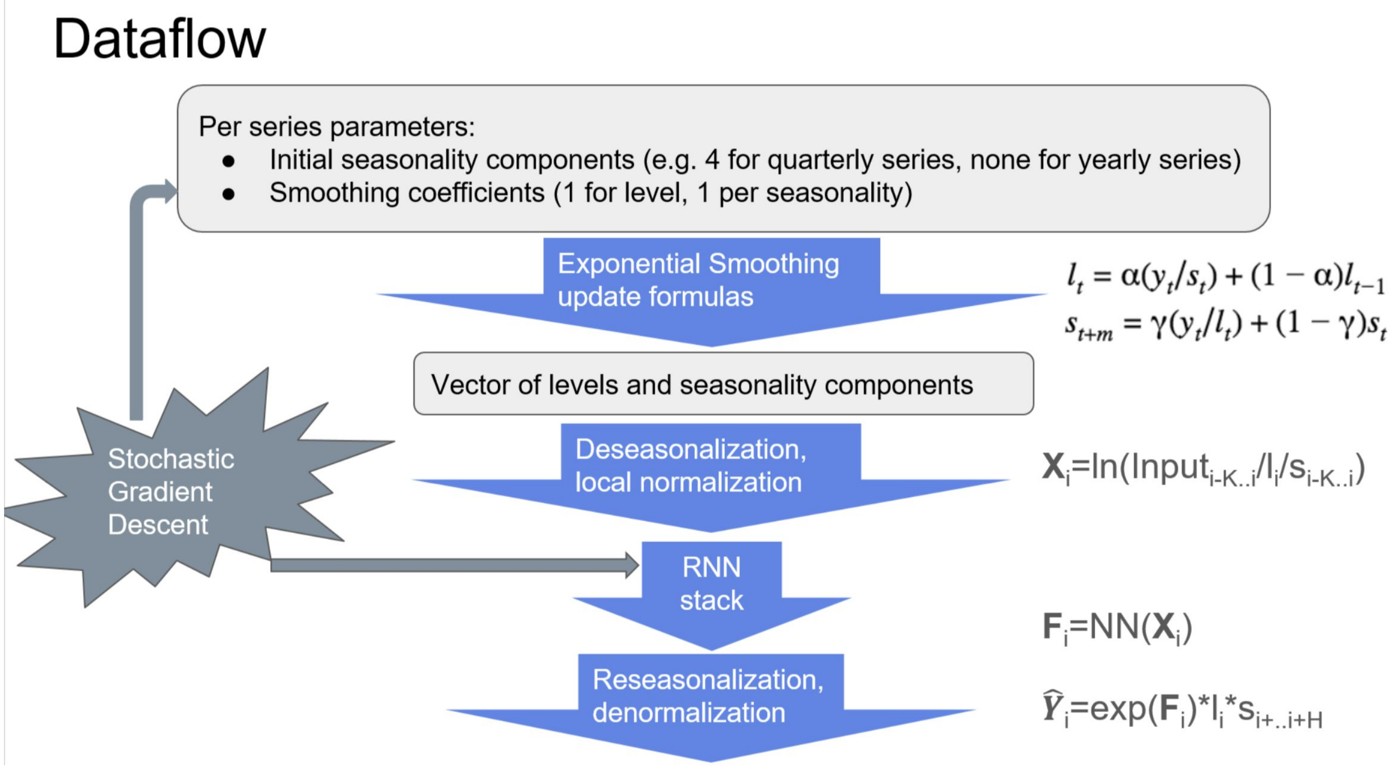
The idea is the following:

- Separate Exponential Smoothing (ES) model per serie
- One global LSTM network that learns on all of the series

Ensemble of Specialists

- Train a pool of models on random subset of the series
- Final forecast for a particular serie is the average of top N models

Hybrid ES-RNN (Uber)



Implemented in C++ using DyNet, but there are open source implementations in PyTorch

DeepAR (Amazon)

- Available as a Service via AWS Forecast
- Also has some open source implementation https://github.com/zhykoties/DeepAR
- Used internally to forecast Amazon.com products demand & AWS utilization

Highlights:

- One global LSTM model for all series
- Predicting likelihood instead of point forecast
- Covariates: item-dependent + time-dependent features

Conclusions

- **Classical methods are hard to beat when you have:**
  - Sufficient history
  - Little external information
  - Few related time series

- **Start with classics and proceed with Combination of Methods:**
  - Reduced risk of bad forecast
  - Ability to incorporate external features

- **Experiment with emerging ML/DL methods:**
  - You know what you are doing
  - Lots of related time series
  - Cross-learning between series
  - One global model

- **Forecasting toolset:**
  - Much better in R (forecast, hts, tsfeatures, tsintermittent, etc..)
  - Bit rusty in Python (statmodels, prophet, etc)

Unit8™

Bibliography

- M4-Conference Presentations
  https://www.mcompetitions.unic.ac.cy/presentations/
- Forecasting at Uber
  https://eng.uber.com/forecasting-introduction/
- Amazon DeepAR+
  https://docs.aws.amazon.com/forecast/latest/dg/aws-forecast-recipe-deeparplus.html
- Automatic Algorithms for Time Series Forecasting
  https://www.slideshare.net/hyndman/automatic-time-series-forecasting
- Exploring Time Series Feature Space
  https://www.slideshare.net/hyndman/exploring-the-feature-space-of-large-collections-of-time-series
- Facebook Prophet
  https://research.fb.com/prophet-forecasting-at-scale/
- Guru of Time Series Forecasting Homepage
  https://robjhyndman.com/
- DeepAR+ Open Source Implementation
  https://github.com/zhykoties/DeepAR
- ES-RNN Open Source Implementation
  https://github.com/damitkwr/ESRNN-GPU

Unit8

Twitter: unit8co

Linkedin: unit8.co

Instagram: unit8.co

follow us!

unit8.co

thank you