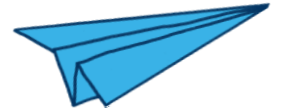


**blueyonder**

Forward looking. Forward thinking.



# Python in the world of mail order and retail

EuroPython 2015, 24.07.2015

Dr. Philipp Mack

DataScientist @ blueyonder

# Agenda

Use cases in the 'real' world for a replenishment solution  
Simplified and purified use case – the essence  
What you need – the ingredients  
Putting it all together – the cook book  
What do you get at the end – the meal  
or a simple replenishment application

# Material

Where can you find everything ?

<https://github.com/philippmack/europython2015-pmack.git>

Includes this presentation, notebooks, examples and the framework

# Real world use cases

Mail order

Estimate what will be ordered by customers the next days and preorder the right amount to minimize delivery time for the customers

$O(100000)$  items

Seasonality / delivery time of goods

Slow selling goods, returns

# Real world use cases

## Retail

Predict the correct order amount of goods to minimize out-of-stock situations as well as the surplus

$O(100)$  items and  $O(100)$  locations

Seasonality / events / weather / expiry date

Fast moving goods

# Simplified use case

What is the purified essence of these examples ?

Know the demand and stocks for certain periods or points in time depending on the delivery time and the frequency of orders

Order the right amount of items considering given conditions like expiry range, availability, excess ...

# Simplified use case

Predict the correct order amount of goods to minimize out-of-stock situations as well as the surplus

Simulated data – timeseries of poisson distributed sales

Order every day in the evening

Throw everything away in the evening

Delivery the next morning



# Framework

What do you we need for a basic/simple solution?

Input/output interfaces -> csv-files

Prediction/simulation/replenishment module

Testing – code and configuration

Plotting

Logging

Documentation

Deployment

Reporting

Monitoring

# Ingredients

## Predictions

We need to know the tomorrow's demand

Model examples :

Take yesterday's sales

Take last week sales of same weekday

Rolling mean

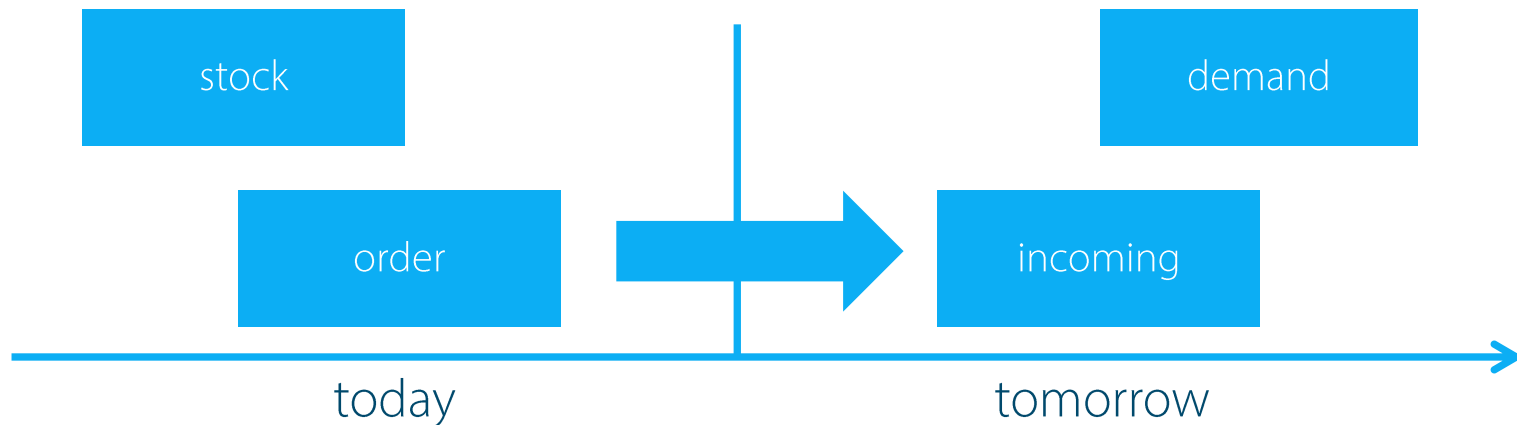
Moving average

Gradient boost ...

# Ingredients

## Order calculation

stock in the evening : 0 ,  
demand interval : 1 day , forecast horizon : 1 day



$$\text{order} = \text{demand of tomorrow} - \text{stock in the evening}$$

# The cook book

## Simulation

Predict demand for each day on test sample

- a) Take yesterday's demand
- b) Rolling mean of the last five days

Calculate order for these demands with a given strategy

- a) Order the expectation value of your prediction
- b) Take quantile  $x$  from a assumed probability density

# The cook book

## *DEMO*

| Prediction            | Replenishment               | Quantiles                           |
|-----------------------|-----------------------------|-------------------------------------|
| -                     | Order 1                     | -                                   |
| -                     | Order 10                    | -                                   |
| Last known sale       | Order expectation           | -                                   |
| Rolling mean (5 days) | Order expectation           | -                                   |
| Last known sale       | Order quantile from poisson | 10,20,30,40,50,60,70,80,90,95,98,99 |
| Rolling mean (5 days) | Order quantile from poisson | 10,20,30,40,50,60,70,80,90,95,98,99 |

# The cook book

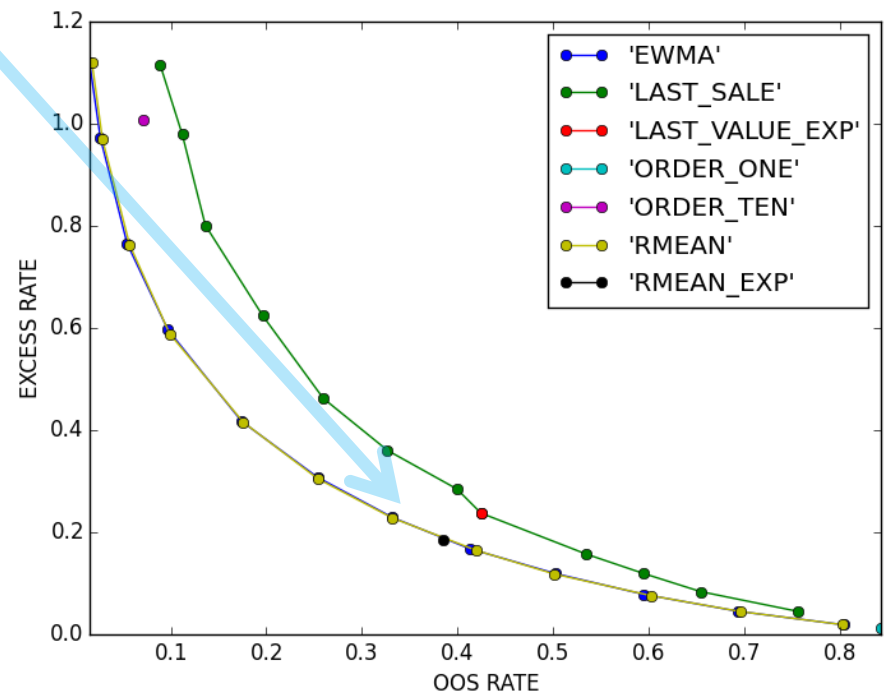
Evaluation of working points for a replenishment solution

Excess rate :

$$\frac{\text{surplus amount}}{\text{sold amount}}$$

OOS rate :

$$\frac{\text{'item' days out of stock}}{\text{all items times days}}$$



# The cook book

Paralellisation :

<https://docs.python.org/2/library/multiprocessing.html>

```
from multiprocessing import Pool  
p = Pool(4)  
result = p.map(f,list_of_values)
```

For multiple hosts :

Redis : <https://redis-py.readthedocs.org/en/latest/>

# The cook book

Testing :

Code :

<http://pytest.org/latest/>

<https://pypi.python.org/pypi/pytest-cov>

Config : yaml / json

<https://pypi.python.org/pypi/voluptuous>



# The cook book

## Replenishment

Chose a strategy and the working point derived from your simulation and calculate an order for the next day for each given product

Working point for Rmean + Poission order  
60% Quantile ->

Simulated OOS : 33%

Simulated excess : 23%

# The cook book

What is missing ?

Logging :

<https://docs.python.org/2/library/logging.html>

Reporting/Monitoring/Plotting :

<http://matplotlib.org/>

<http://stanford.edu/~mwaskom/software/seaborn/>

<http://bokeh.pydata.org/en/latest/>

# The cook book

What is missing ?

Documentation :

<http://sphinx-doc.org/>

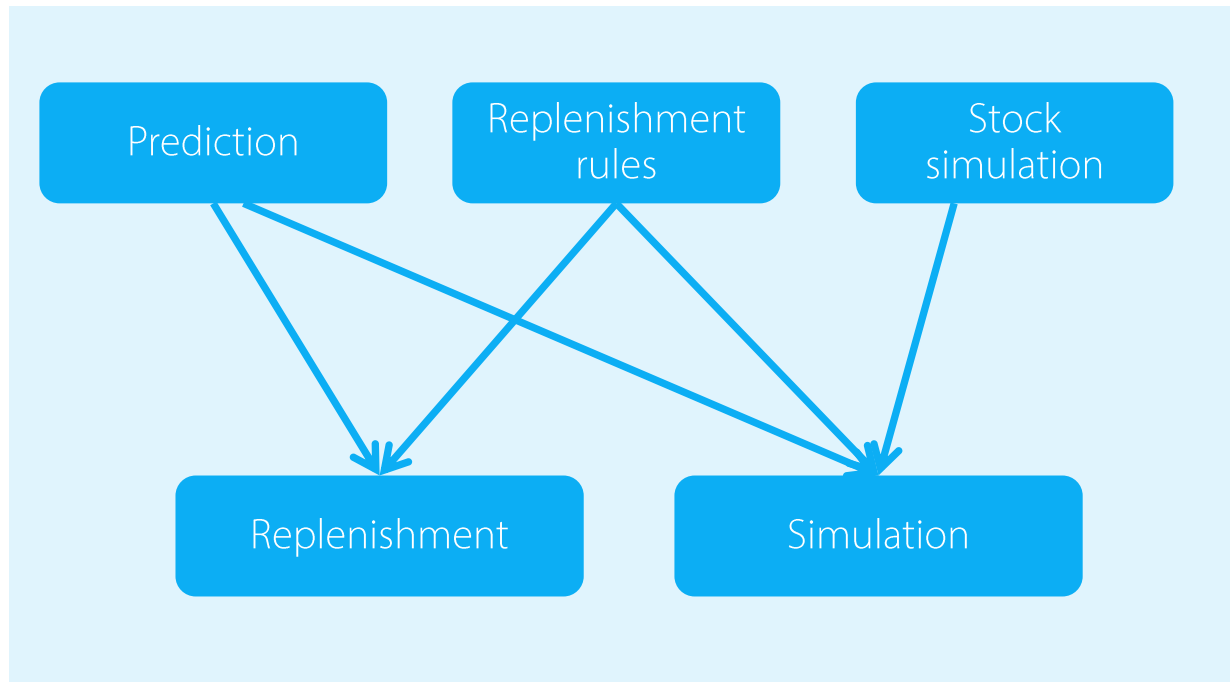
Deployment :

[http://docs.ansible.com/developing\\_api.html](http://docs.ansible.com/developing_api.html)

<http://www.ansible.com/application-deployment>

# The meal

## Putting it all together



Documentation  
Logging  
Monitoring  
Tests  
Configuration  
Deployment  
Reporting

This simple replenishment framework provides a very basic solution for an order forecast system and illustrates the simplicity and strength of python.

## What's the lesson ?

Thank you for your attention and visit us our blueyonder booth