DSF-R

Vienna 2018

Scaling R in the Cloud

with Azure Batch and Docker

Christoph Bodner & Thomas Laber

Post

AGENDA

	01	02	03
Topics	Who we are (obligatory marketing stuff…)	Scaling R with Azure	Case study
	Data Science@Post AG: - Overview: Post AG - Our team	Using: - AzureBatch and - Docker for development & production	doAzureParallel + Caret: - Distributed ML - Easy setup

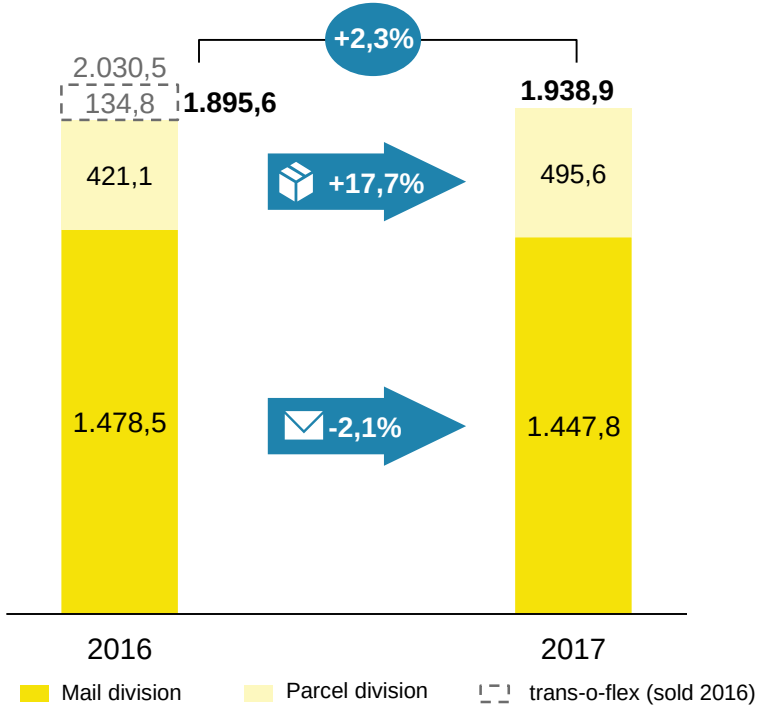
AGENDA

	01	02	03
Topics	Who we are (obligatory marketing stuff…)	Scaling R with Azure	Case study
	Data Science@Post AG: - Overview: Post AG - Our team	Using: - AzureBatch and - Docker for development & production	doAzureParallel + Caret: - Distributed ML - Easy setup

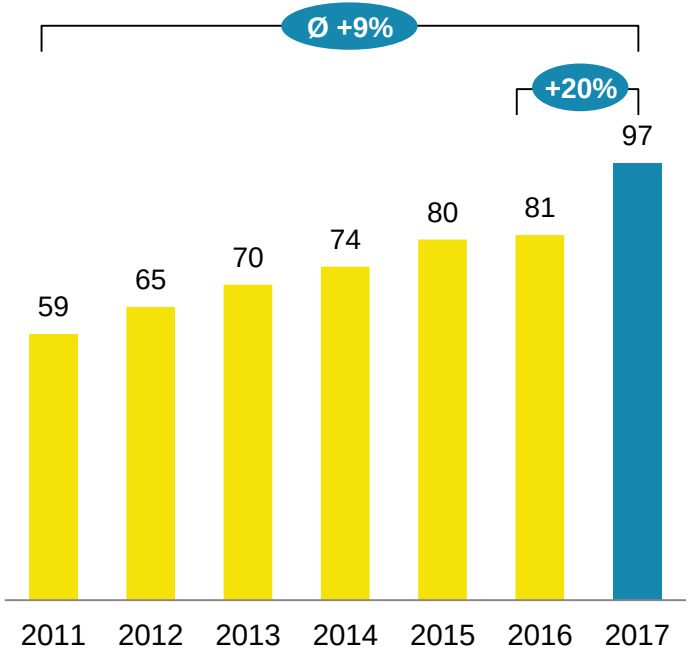
OVERVIEW: POST AG

COMPANY PERFORMANCE & PARCEL VOLUMES OVER TIME

SALES PERFORMANCE
mio EUR



PARCEL VOLUMES OF AUSTRIAN POST
mio parcels



OUR TEAM
PEOPLE WHO LIKE $\pi z^2 a$ IN EVERY FORM

Christoph Bodner

Lead Data Scientist

Quantitative Finance (WU)
Prev.: KPMG

Thomas Laber

Senior Data Scientist

Business Informatics (TU)
Prev.: Accenture

Martin Blöschl

Junior Data Scientist

Computational Intelligence (TU)

Raphael Pesl

Junior Data Scientist

Mathematics (TU)

AGENDA

	01	02	03
Topics	Who we are (obligatory marketing stuff…)	Scaling R with Azure	Case study
	Data Science@Post AG: - Overview: Post AG - Our team	Using: - AzureBatch and - Docker for development & production	doAzureParallel + Caret: - Distributed ML - Easy setup

How can we run complex experiments quickly?

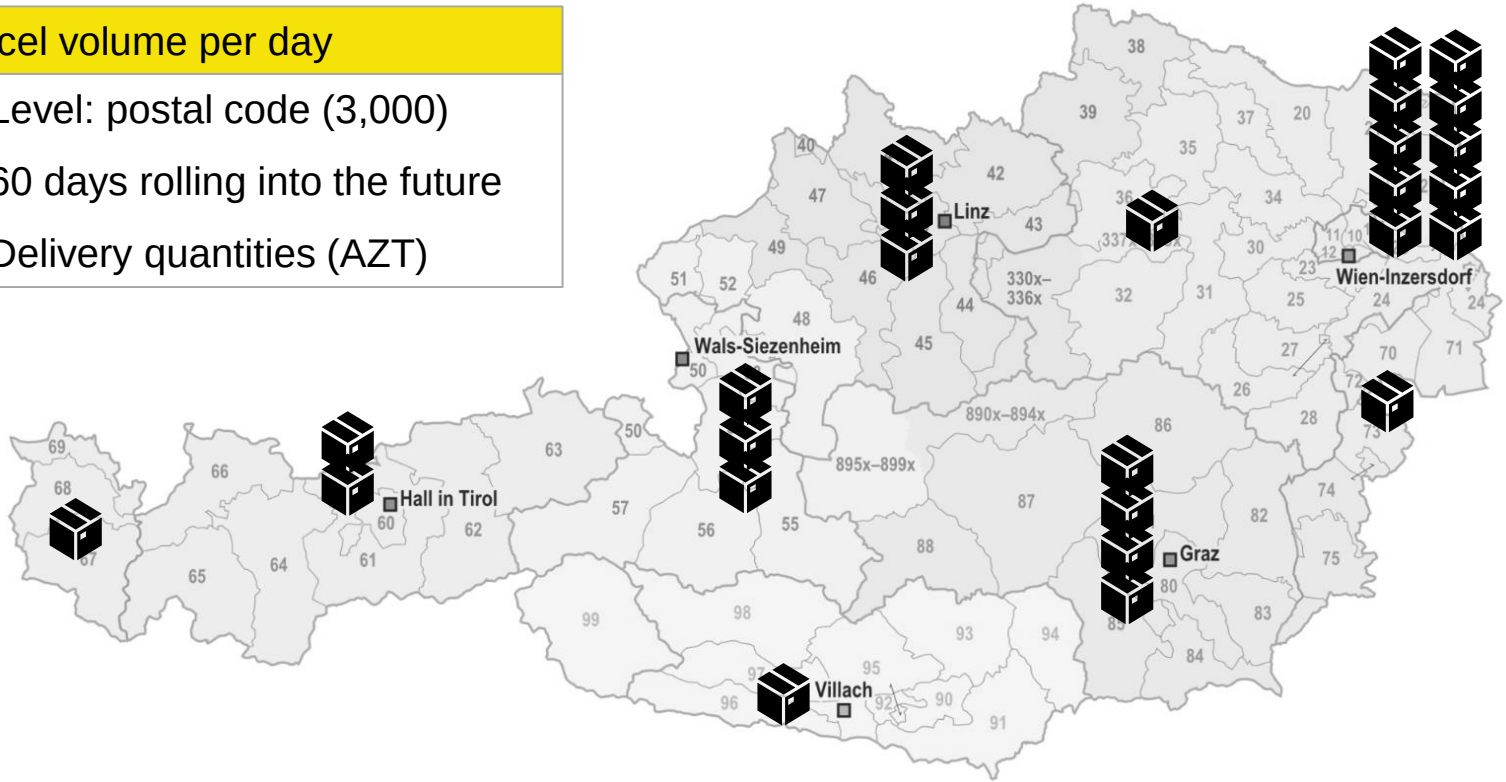
How can we put those models in production?

A SAMPLE PROBLEM

HOW MANY PARCELS WILL WE NEED TO DELIVER IN THE FUTURE?

Parcel volume per day

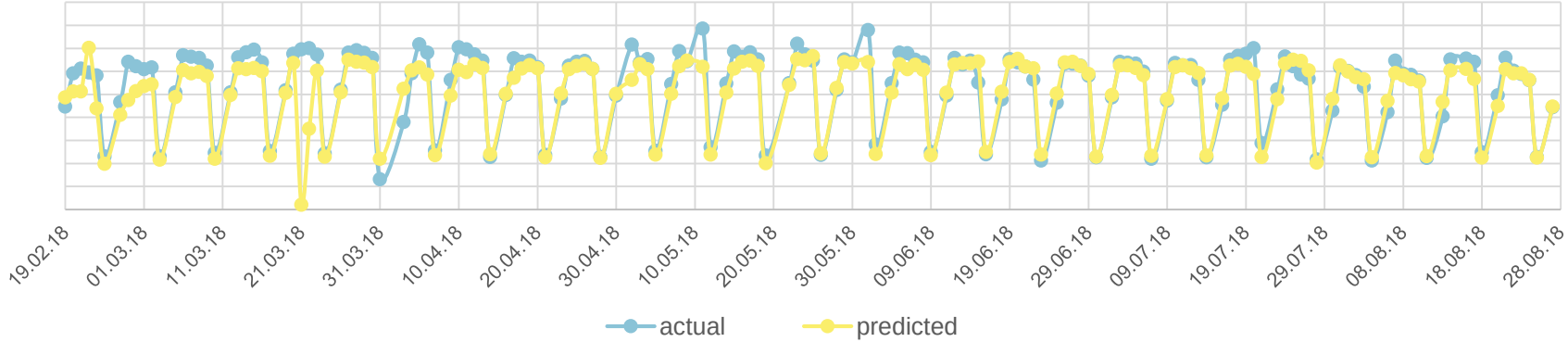
- Level: postal code (3,000)
- 60 days rolling into the future
- Delivery quantities (AZT)



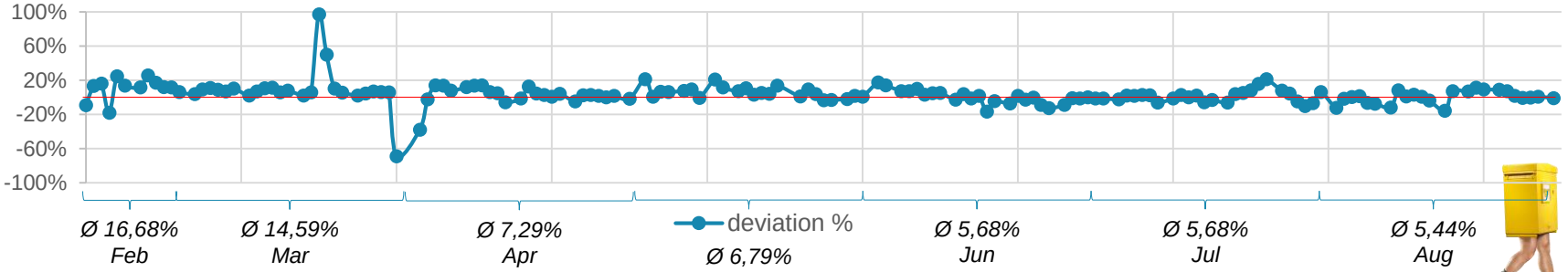
FORECASTING PERFORMANCE
7-DAYS-AHEAD

Performance

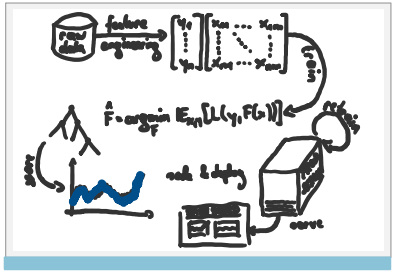
Parcel volumes per day Austria (7 days ahead)



Deviation in percent

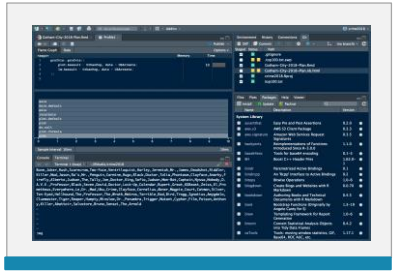


FROM EXPERIMENT TO DEPLOYMENT
OUR JOURNEY



Whiteboard phase

- Draft architecture
- Modeling strategy
- Everything seems easy☺

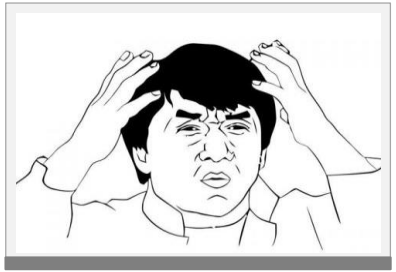


Experiment phase

- Build model on sample
- Model improves
- Happy Data Scientists

Deadline phase

- Model training takes too long
- Model gets simplified to speed up training
- Performance suffers

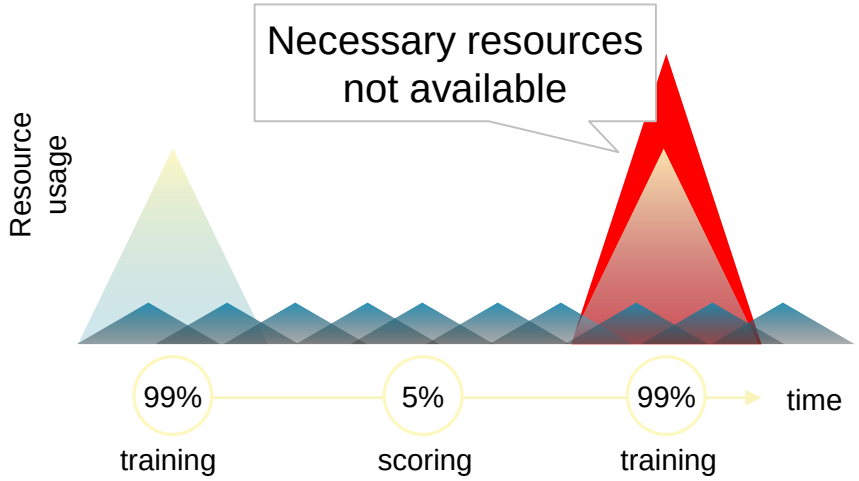


Deployment phase

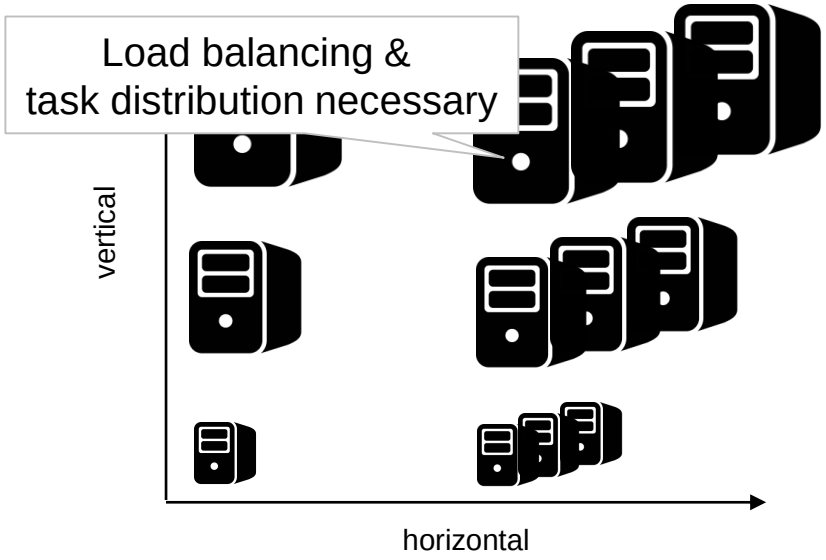
- Dev != Prod
- Server procurement slow
- Server not powerful enough for training
- Server too powerful for scoring

PROBLEMS WE ENCOUNTERED
SCALING COMPUTE RESOURCES ON DEMAND

Resource utilisation



Scaling options

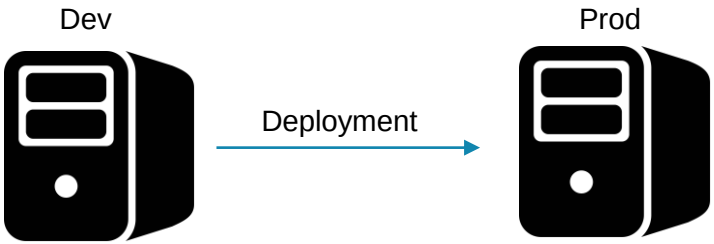


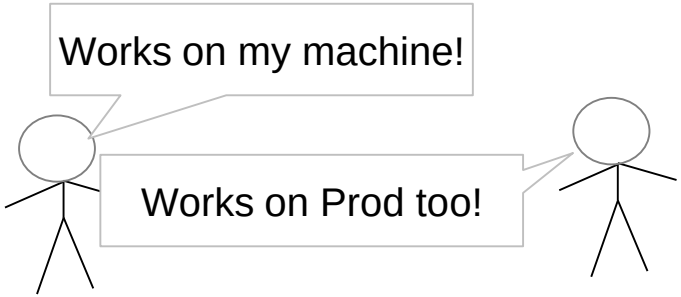
We want to be able to elastically scale **up** and **out** to meet our needs

DEPLOYMENT WAS A PAIN …
DEV-PROD DISPARITY

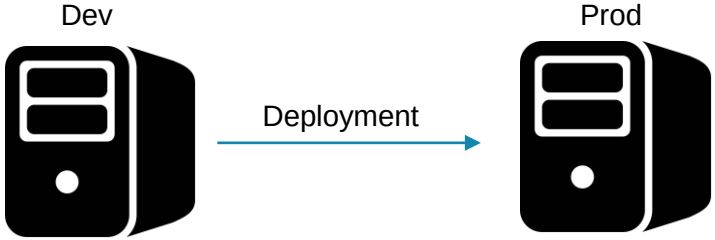
Post

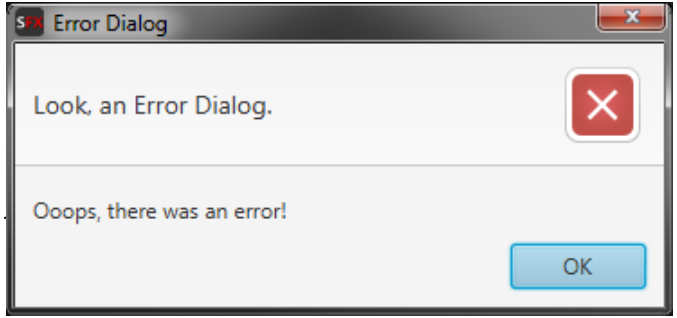
Happy Path Deployments





~~Happy Path~~ Deployments





▶ Keeping Dev and Prod aligned is hard even with best intentions

How can we fix these problems?

HOW CAN WE SOLVE OUR PROBLEMS?
WE HAVE A LONG WISH LIST

Post

Our wish list:

deploy automatically
scale compute
elastically
No vendor lock-in
uniform
architecture
Easy scheduling
Job Management Portal
eliminate dev-prod disparities
cheap
FPGAs
Easy management
no dependencies
GPUs
programming
language agnostic

Solutions we considered:

We have two dominant stacks:

- Microsoft .net + Azure
- SAP

Since we have SAP and .net/Azure support in-house, we looked primarily at these two stacks.

SCALE R WITH AZURE BATCH
DEVELOPMENT & PRODUCTION WORKLOADS

Post

Azure Container Registry

Gated checkin triggers:

Container 1

Container 2

...

Container n

Pull image

Build

Test

Publish

Gitflow Struktur

Hot Fix

Master/Prod

Release

Dev

Feature branches

Gated Checkin

Gated Checkin

Commit

Merge

Container Orchestration & Deployment

AZ Batch:
- doAzureParallel
- aztk

Container

Specify jobs

ETL + Storage

SQL

Write results:
- models
- scores

Job Agent & autoscaling

Batch Explorer

Start compute cluster

Data Scientist 1

Data Scientist 2

SOLUTIONS WE CONSIDERED
COMPARISON

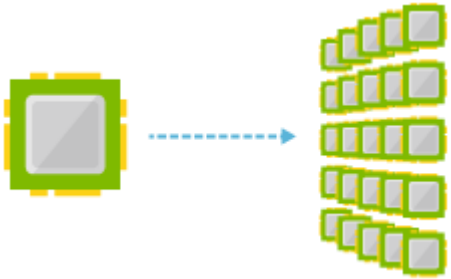
Also take a look at Azure Databricks for Spark workloads

This looks very promising

	Cloud vs. On-Premise Solution			
	Azure ML Studio	Azure ML Workbench	Azure Batch	SAP HANA (PAL)
Supported Languages	R/Python	Python	All	R (Python)
Supports GPUs		Yes	Yes	No
Dev-Prod parity	Easy	Easy	Easy	Hardish
R package available	AzureML	-	doAzureParallel	-
Independent upgrades	partially	partially	Yes	partially
Elastic scaling	-	Yes	Yes	No
Scheduling included	No	No	Yes	Yes
SCORE				

AZURE BATCH OVERALL WINNER

FLEXIBLE, HIGH VALUE/MONEY AND LOW VENDOR LOCK-IN

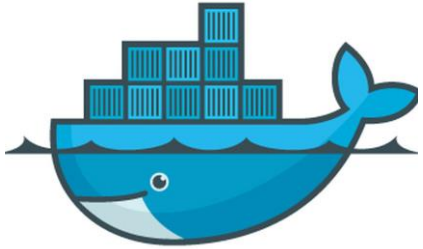


Scale up and out

Specify node sizes and types, e.g. GPU/CPU, RAM and get large discount on low-prio nodes

Scheduling integrated

Specify job schedule and resize pool based on number of outstanding tasks



Docker support

Dev and Prod parity
Fits into CD pipeline

Source images: Microsoft and Docker homepage

DOCKER BRIEFLY EXPLAINED
BUILD ONCE – RUN ANYWHERE

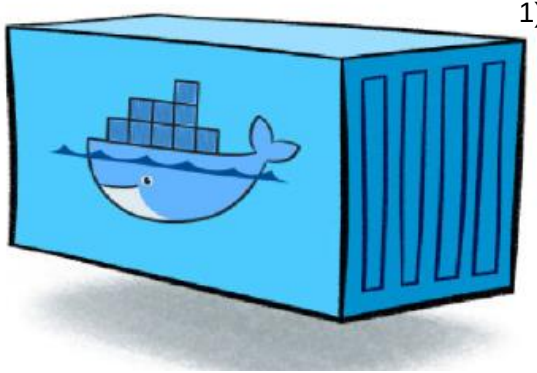
Example Dockerfile

```
## Description: https://hub.docker.com/r/rocker/tidyverse/
FROM rocker/tidyverse

# Install your R package
## Copy R package to docker
RUN mkdir -p /usr/r_package
COPY . /usr/r_package

## Install package dependencies
RUN r -e 'devtools::install_deps(pkg = "/usr/r_package/", dependencies = T)'
## Roxygenize
RUN r -e 'devtools::document(pkg = "/usr/r_package/")'
## Install package
RUN R CMD INSTALL /usr/r_package
```

R image used

Here we install our package

Build Container



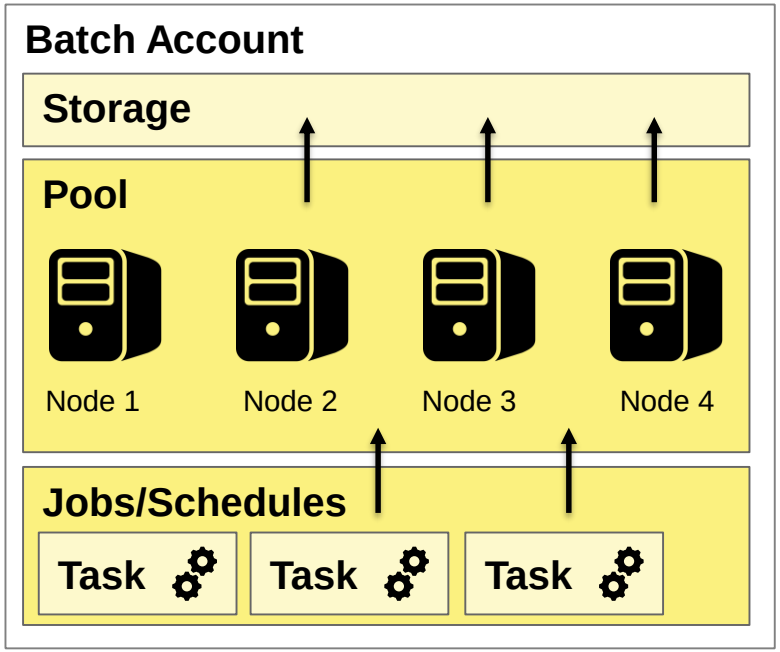
1)

Container can be deployed to docker runtime in production: Dev = Prod (mostly☺)

Source image: 1) https://stephenafamo.com/blog/understanding-docker-containers/

AZURE BATCH
CONCEPTS

Components overview



Description

Pool …… Definition of compute resources
- Node number/type
- Container settings
- Autoscaling
- Task number per node

Schedule … Recurring job for a pool
- Job specification

Job …….. Contains pool and task specs
- Job manager task

Task ……. Work specification
- Command line
- upload options

AZURE BATCH
SETUP OVERVIEW

Pool configuration

```python
# setup batch pool
def create_pool(batch_service_client, pool_id):
    image_ref_to_use = batch.models.ImageReference(
            publisher='microsoft-azure-batch',
            offer='ubuntu-server-container',
            sku='16-04-lts',
            version='latest'
            )

    # Private docker repo specification
    container_conf = batch.models.ContainerConfiguration(
        ...
     )

    # create pool specs with pre-fetched images
    new_pool = batch.models.PoolAddParameter(
        id=pool_id,
        virtual_machine_configuration=batch.models.VirtualMachineConfiguration(
            image_reference=image_ref_to_use,
            container_configuration=container_conf,
            node_agent_sku_id='batch.node.ubuntu 16.04'),
        vm_size=_POOL_VM_SIZE,
        max_tasks_per_node=_MAX_TASKS_PER_NODE
        enable_auto_scale=True,
        auto_scale_evaluation_interval='PT5M',
        auto_scale_formula= "some formula for node type and count"
    )

    # submit pool creation to azure batch
    batch_service_client.pool.add(new_pool)
```

Pool base image

Here using Python SDK

Docker image & autoscaling

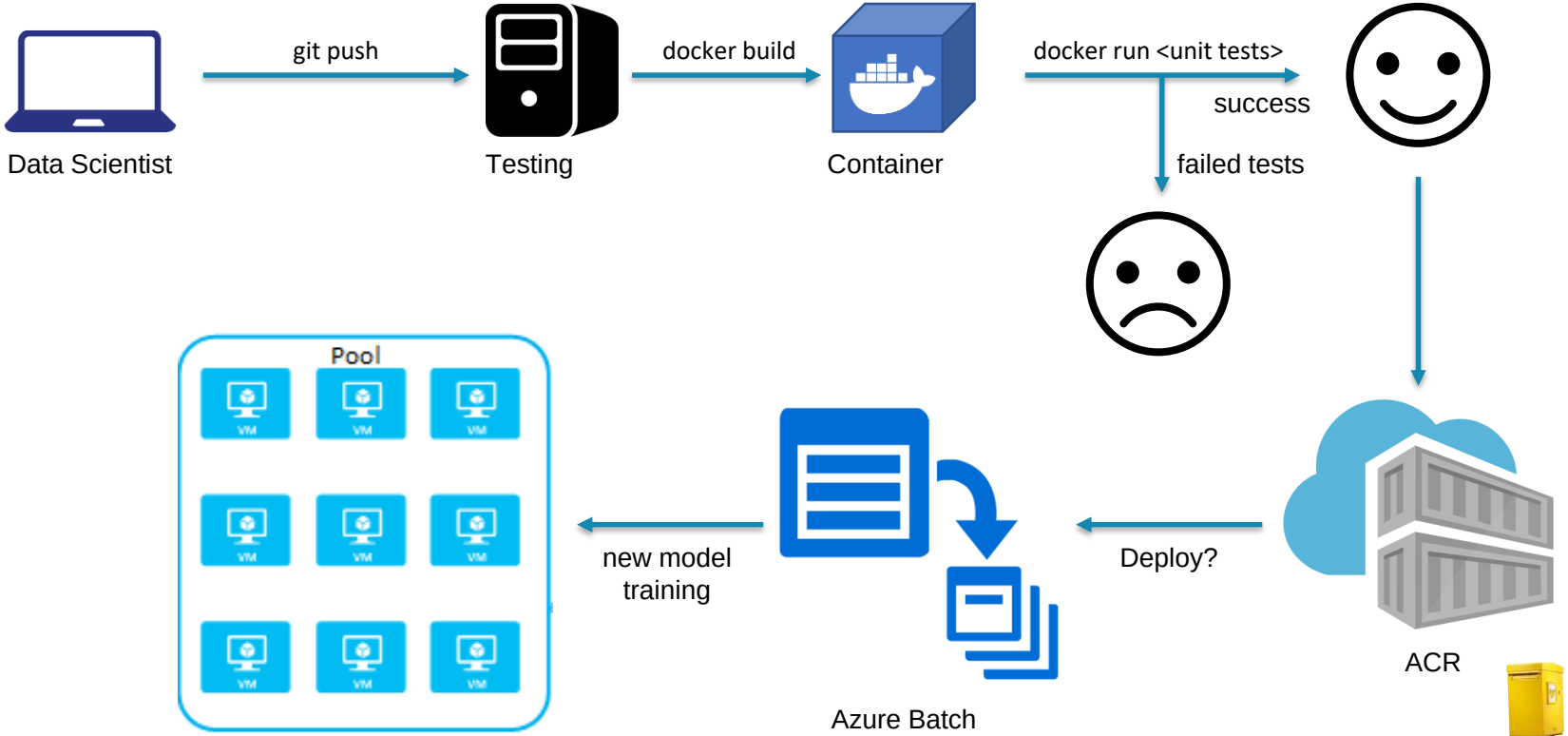
Job Manager Task

```python
        job_manager = batchmodels.JobManagerTask(
            id=_AZ_JOB_MANAGER_ID,
            command_line=_JOB_MANAGER_CMD,
            container_settings=task_container_conf,
            authentication_token_settings=batchmodels.AuthenticationTokenSettin
                access=[batchmodels.AccessScope
                ),
            environment_settings=[
                ...
                ],
            kill_job_on_completion=True,
            run_exclusive=False
        )

        job_spec = batchmodels.JobSpecification(
            pool_info=batchmodels.PoolInformation(pool_id=_POOL_ID),
            display_name='test_schedule',
            #on_all_tasks_complete='noaction',
            job_manager_task=job_manager,
            #common_environment_settings=No
        )

        schedule = batchmodels.Schedule(
            recurrence_interval='PT1M'
        )

        job_schedule = batchmodels.JobScheduleAddParameter(
            id=_JOB_SCHEDULE_ID,
            schedule=schedule,
            job_specification=job_spec
        )

        # submit job schedule creation
        batch_client.job_schedule.add(job_schedule)
```

Command to run

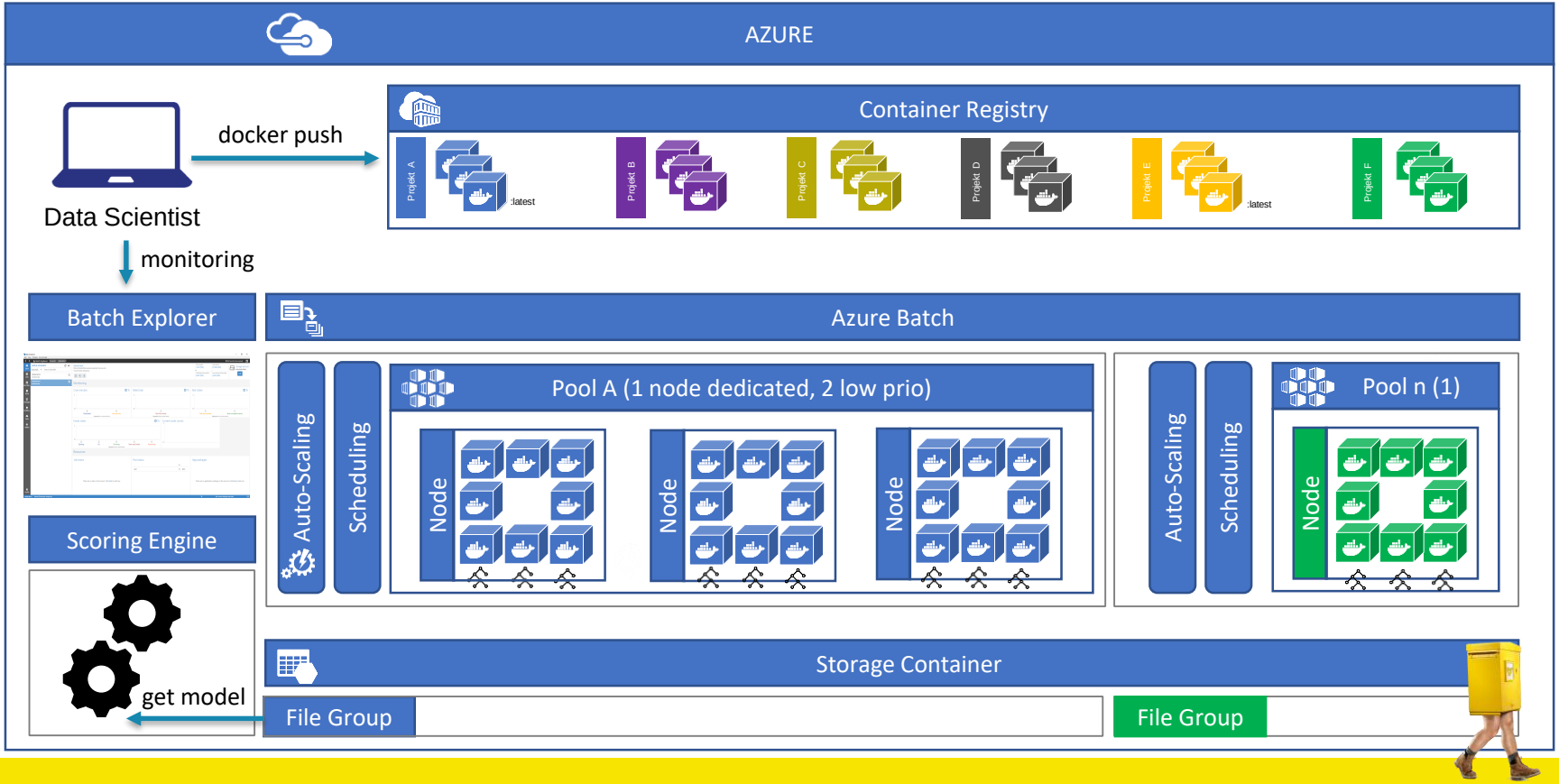
Job schedule bundles everything

VSTS
CONTINUOUS DELIVERY PIPELINE (SIMPLIFIED)



OUR STACK

WE ARE CURRENTLY BUILDING A HPC ENVIRONMENT



MONITORING BATCH JOBS
WITH BATCH EXPLORER

Overview

Currently active jobs

Status of your pools

Output from your jobs

AGENDA

	01	02	03
Topics	Who we are (obligatory marketing stuff…)	Scaling R with Azure	Case study
	Data Science@Post AG: - Overview: Post AG - Our team	Using: - AzureBatch and - Docker for development & production	doAzureParallel + Caret: - Distributed ML - Easy setup

OPEN ISSUES

SECURITY & EVALUATING DIFFERENT REAL-TIME SCORING OPTIONS

Security

- ~~Secret management with KeyVault~~
- ~~Azure Active Directory integration~~
- ~~VNET integration of pools~~
- Docker security best practices audit/check
- Disable public endpoints

Real-time scoring options

- Using AzureML Studio
- Using Azure Model Management
- Kubernetes Cluster
- Azure Functions
- Azure Container Instances + Logic Apps
- …

What we are currently looking into

A bit of work is still open, but we plan to have everything wrapped up end of September

Case Study: Caret + Azure

LET'S DO A QUICK CASE STUDY
CARET + DOAZUREPARALLEL

What we will show you:

- How to setup a Batch Account in Azure
  - Things to consider: VNET, AAD, region…
- Download BatchExplorer
- Get credentials
- Switch to R:
  - Using a custom Docker image
  - Train models in parallel using caret+doAzureParallel as backend

SUMMARY
AZUREBATCH OFFERS EFFICIENT AND FLEXIBLE HPC

Development

- Use doAzureParallel to test multiple models quickly
- Create Docker container for your colleagues to use
- Push to container registry

Production

- Use container image from development
- Setup schedule and cluster specification
- Configure autoscaling and
- Violá ☺

▶ You get a cost efficient, flexible and scalable architecture for your team

Thank you for your attention!

Any questions?

You can contact us at:

- linkedin.com/in/christoph-bodner
- linkedin.com/in/thomas-laber/