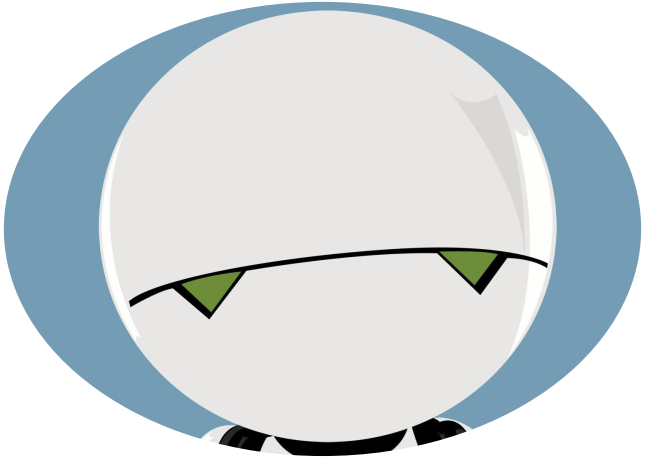
Deeplearning in production

the Data Engineer part

whoami



Scauglog
Data Engineer, Xebia

b

Part 1: Predict at scale

DataXDay 2019

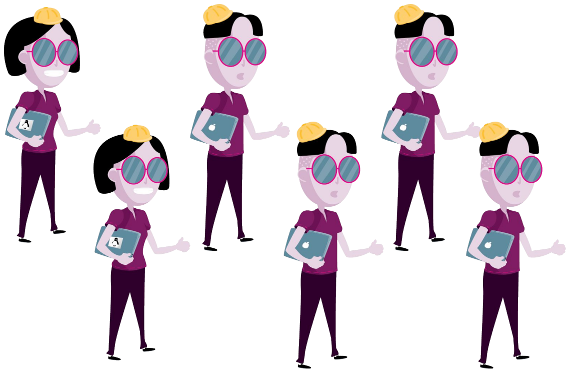
Part 2: Training And Monitoring

init project

Team Astro

Product Owner

Scrum Master

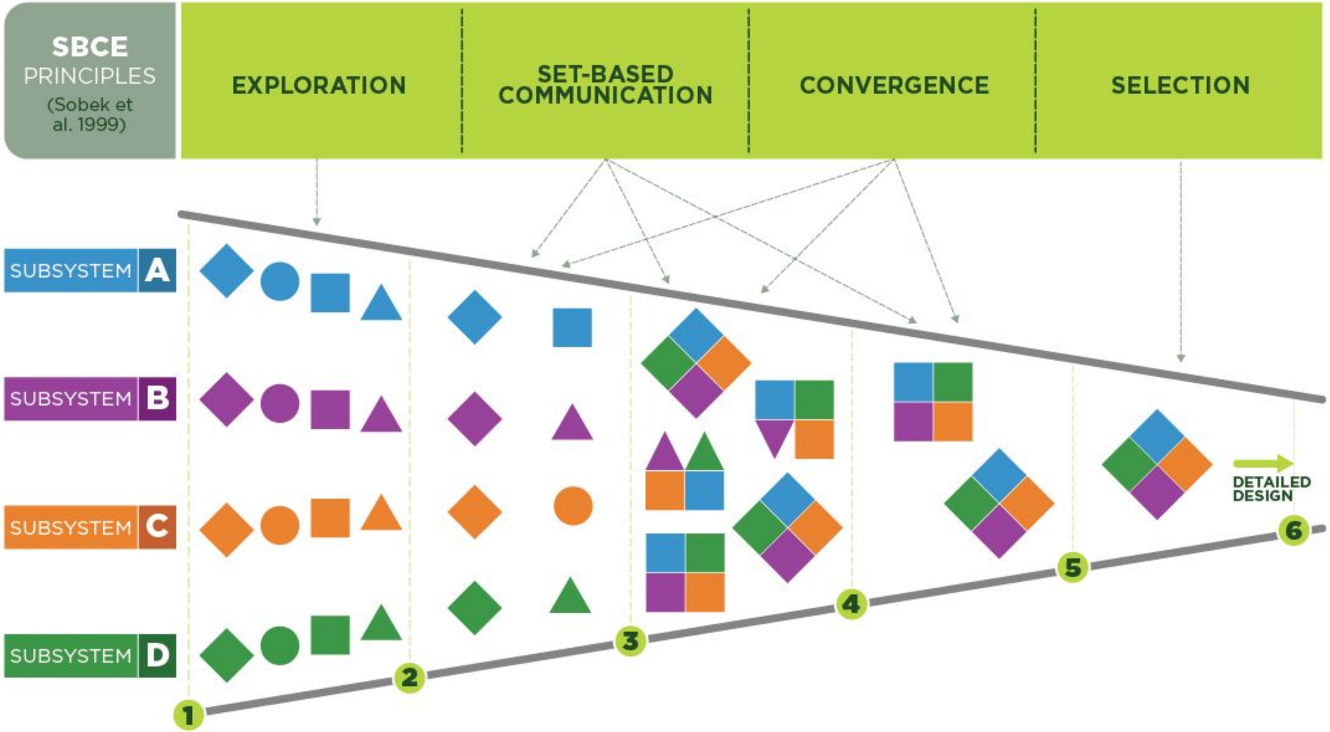


Data Scientists, Data Engineers,
Machine Learning Engineers

Business

- Buy sponsored link on google adwords
- 10M Predictions in less than 1 hour (~2700/s)
- Bid each day
- Each bid should cost less than what we earn

Choose your model

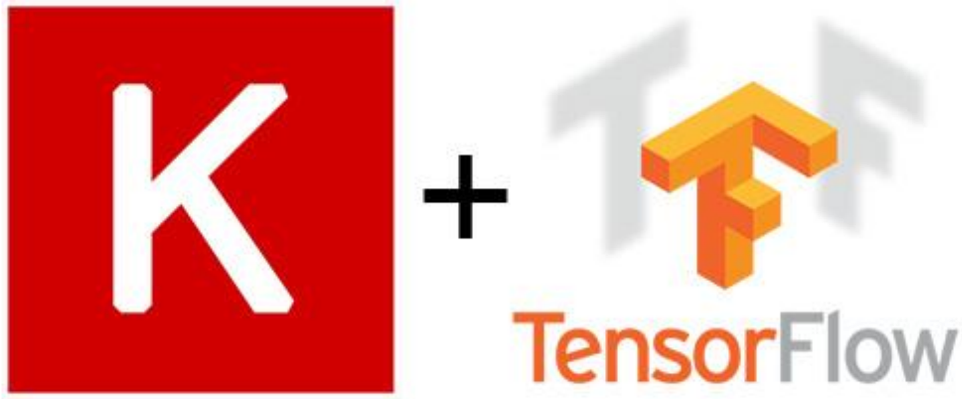


And the winner is

XGBoost

And the winner is

What is Deep Learning?

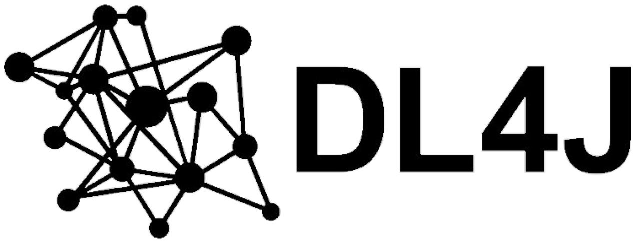


What is a Deep Learning model?

Deeplearning in Production at Scale

?

Choose your Framework



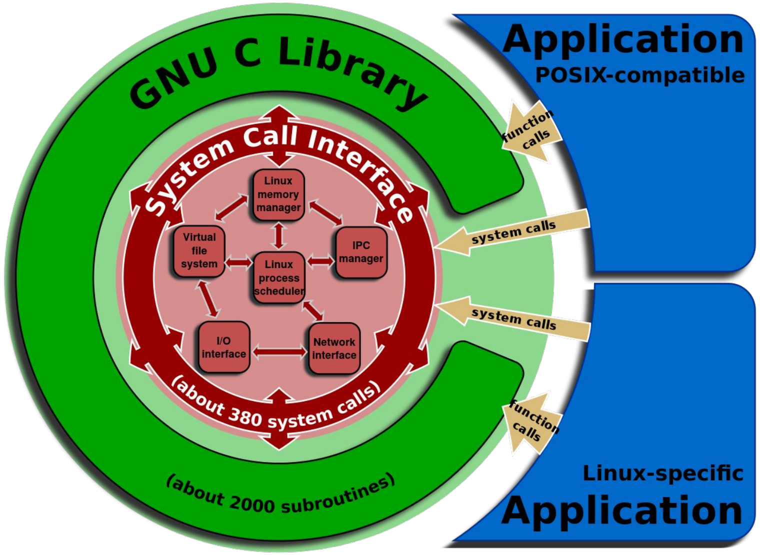
- Distributed Prediction
- Can create complex network
- Documentation
- Community

And the winner is

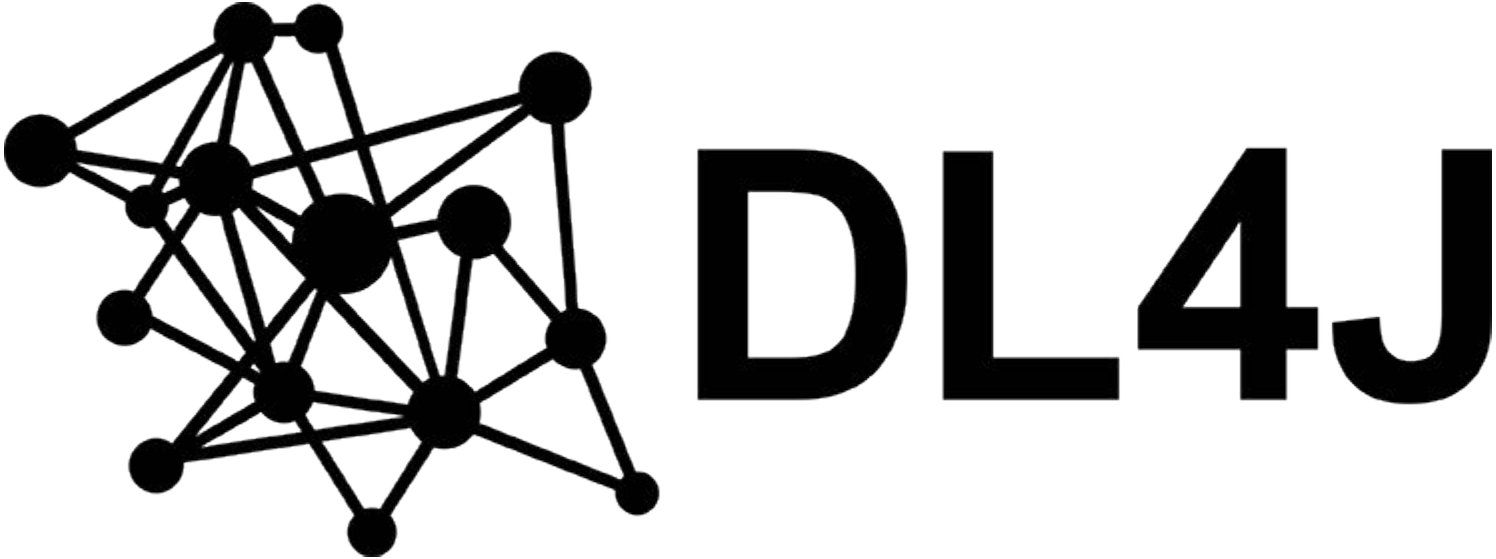
TensorFlow

Wait

CentOS

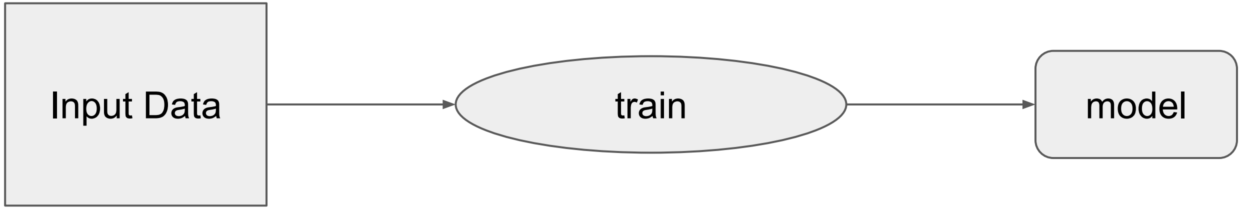

GNU C Library
(about 2000 subroutines)
System Call Interface
(about 380 system calls)
Linux memory manager
Virtual file system
IPC manager
Linux process scheduler
I/O interface
Network interface
Application
POSIX-compatible
function calls
system calls
system calls
function calls
Linux-specific
Application

And the winner is

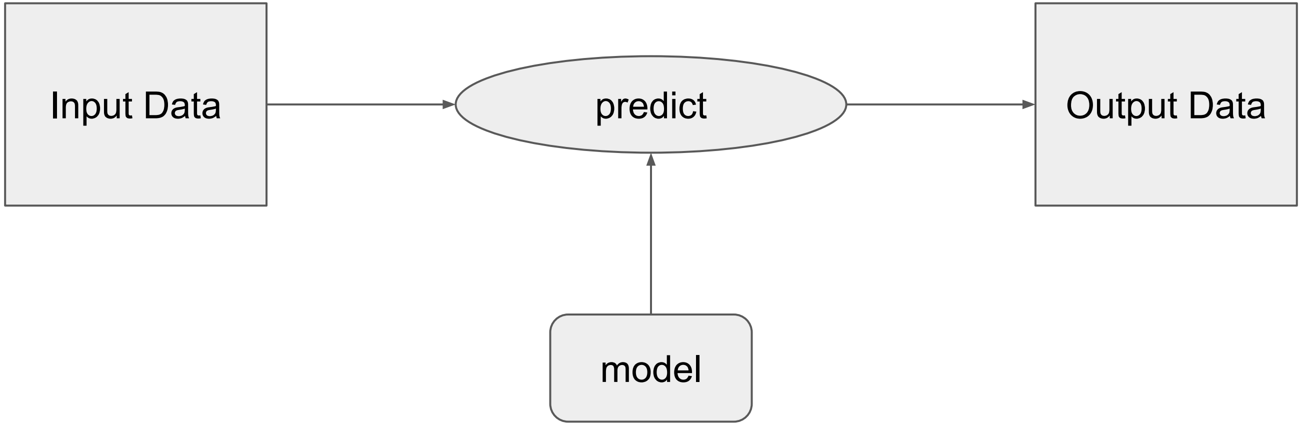


How to Deep Learn

Train Workflow



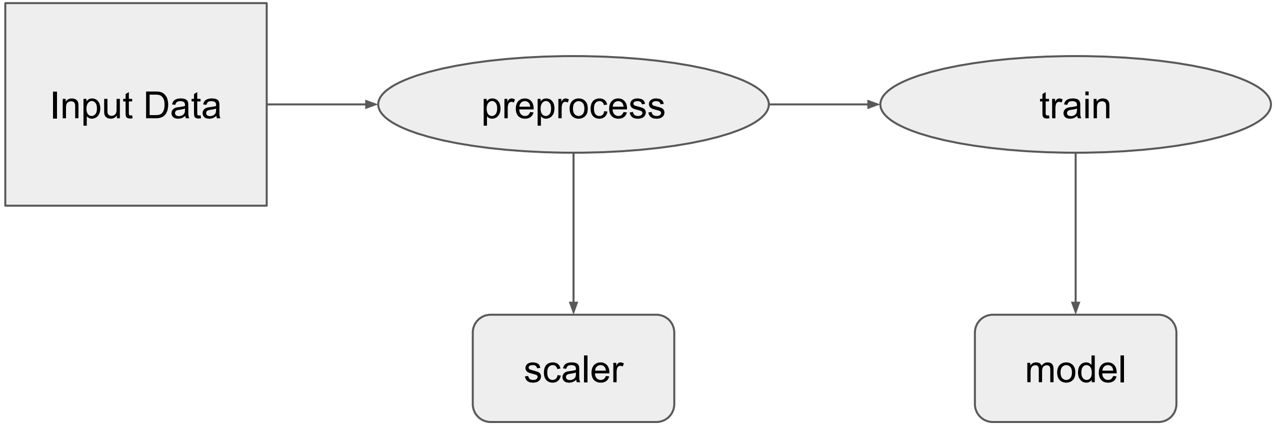
Predict Workflow



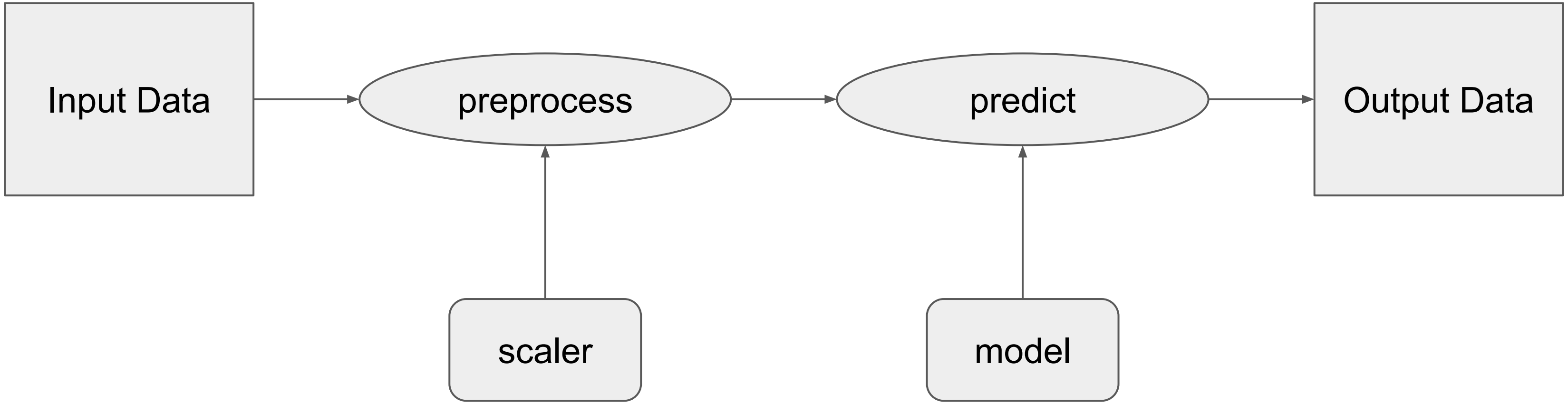
Preprocessing

- Scaling (normalisation, min max, …)
- Replace null
- Lagging

Train Workflow



Predict Workflow

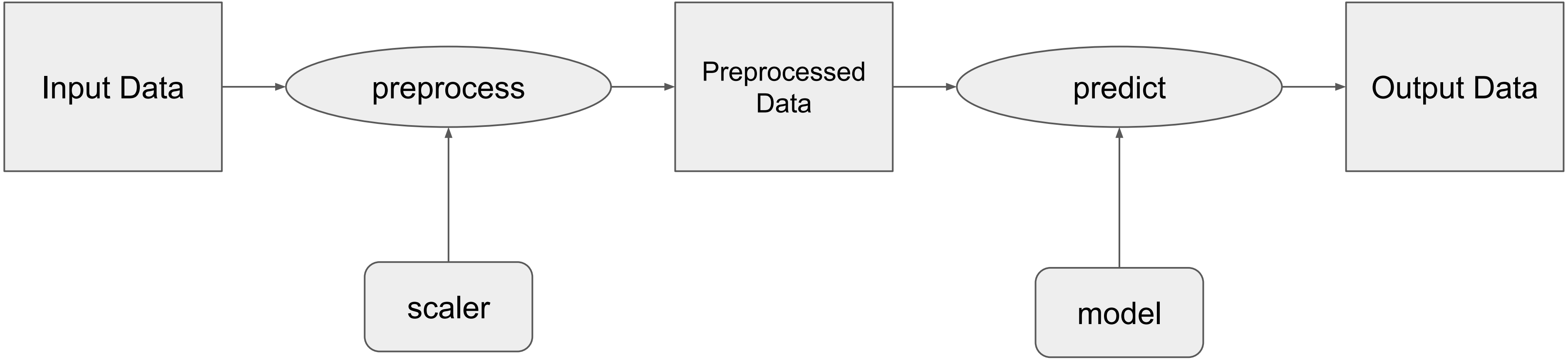


Training at scale

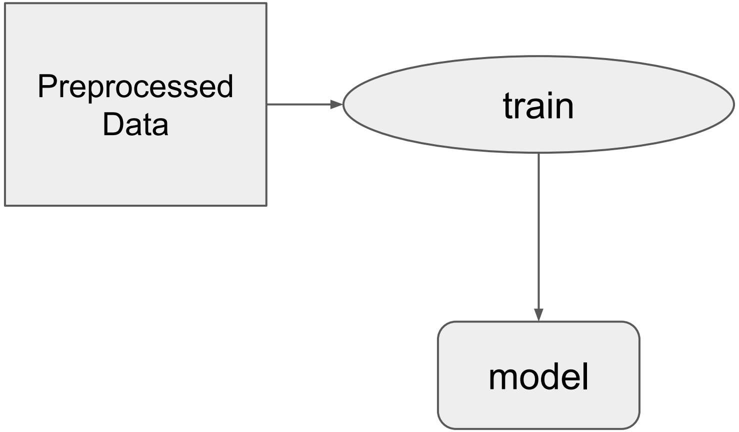
Retrain again and again and again...

- Model performance decline over time
- Hyperparameter tuning
- Deep Learning model rarely comes alone (clustering)

Predict Workflow



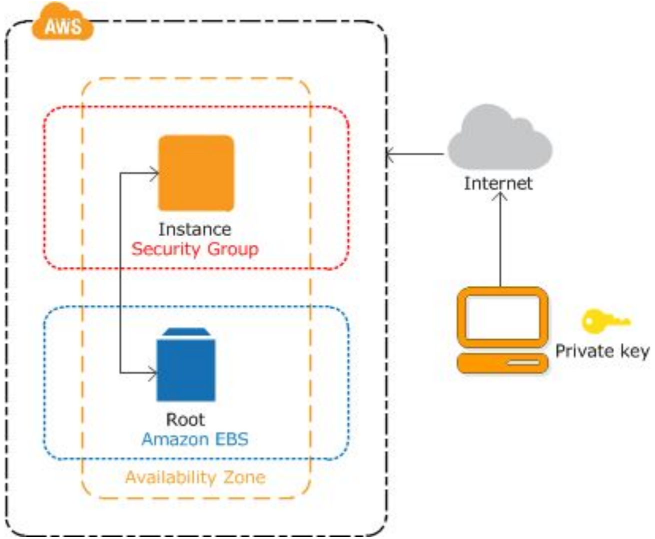
Train Workflow



Training at scale: AWS EC2

Training at scale: AWS EC2

- Create VPC
- Create Subnet associated to VPC
- Create an IGW associated to VPC
- Create a route table associated to IGW
- Create a Security Group associated to VPC
  - Authorize ssh only for my IP
- Create a key pair
- Create EC2 server with EBS volume



Training at scale: AWS EC2

- Add ssh keys of team members
- Install cuda, cudnn, nccl and configure them
- Deploy train jar to EC2 instance
- Deploy train pipeline to EC2 instance
- Deploy preprocessed data to EC2 instance
- Deploy auto shutdown script

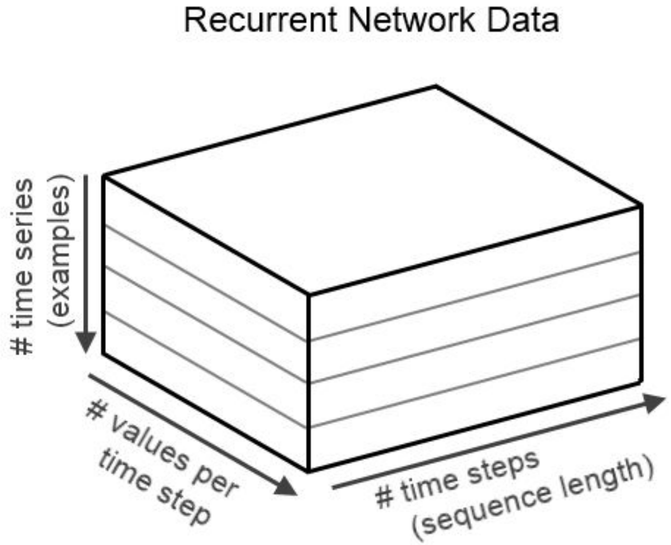
Training at scale: AWS EC2

- Ansible
- Transfert preprocess data to S3
- Store model in S3
- Check CPU vs GPU training time
- Keep track of training config and performance
- Share knowledge with Data Scientist
- Put your data in EBS volume if they fit

ANSIBLE

Training with DL4J: Lessons learned

- Beware of tensor shape
- Prefetch data in memory (InMemoryDatasetIterator)
- Add listener to monitor your training compute performance
- Use the UI

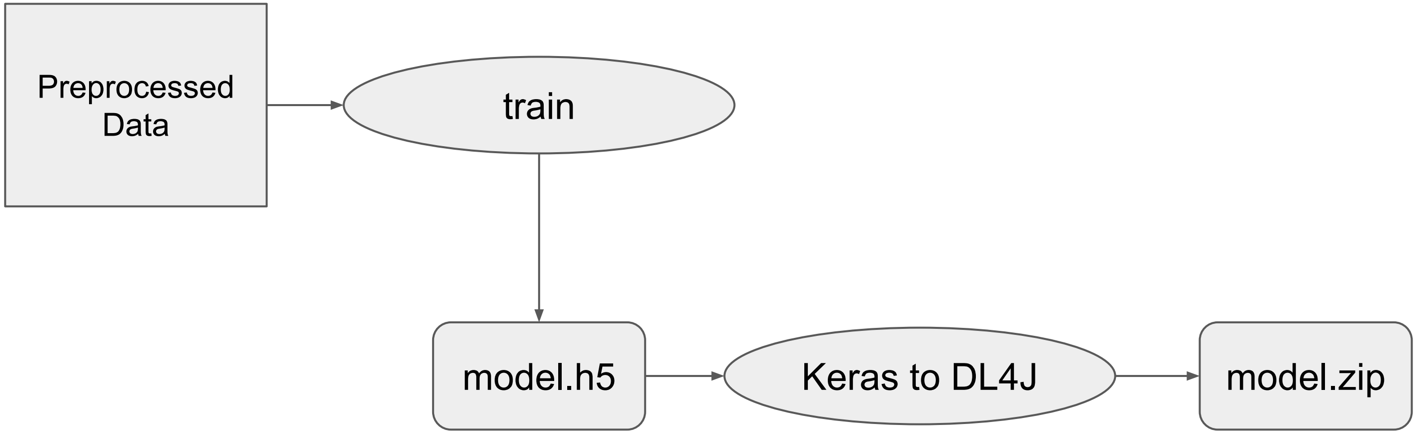


Keras to DL4J

- Data Scientist loves Keras
- Keras is easier to import on notebook
- Training on Keras is faster
- Keras is compliant with cloud training (Sagemaker, CloudML)

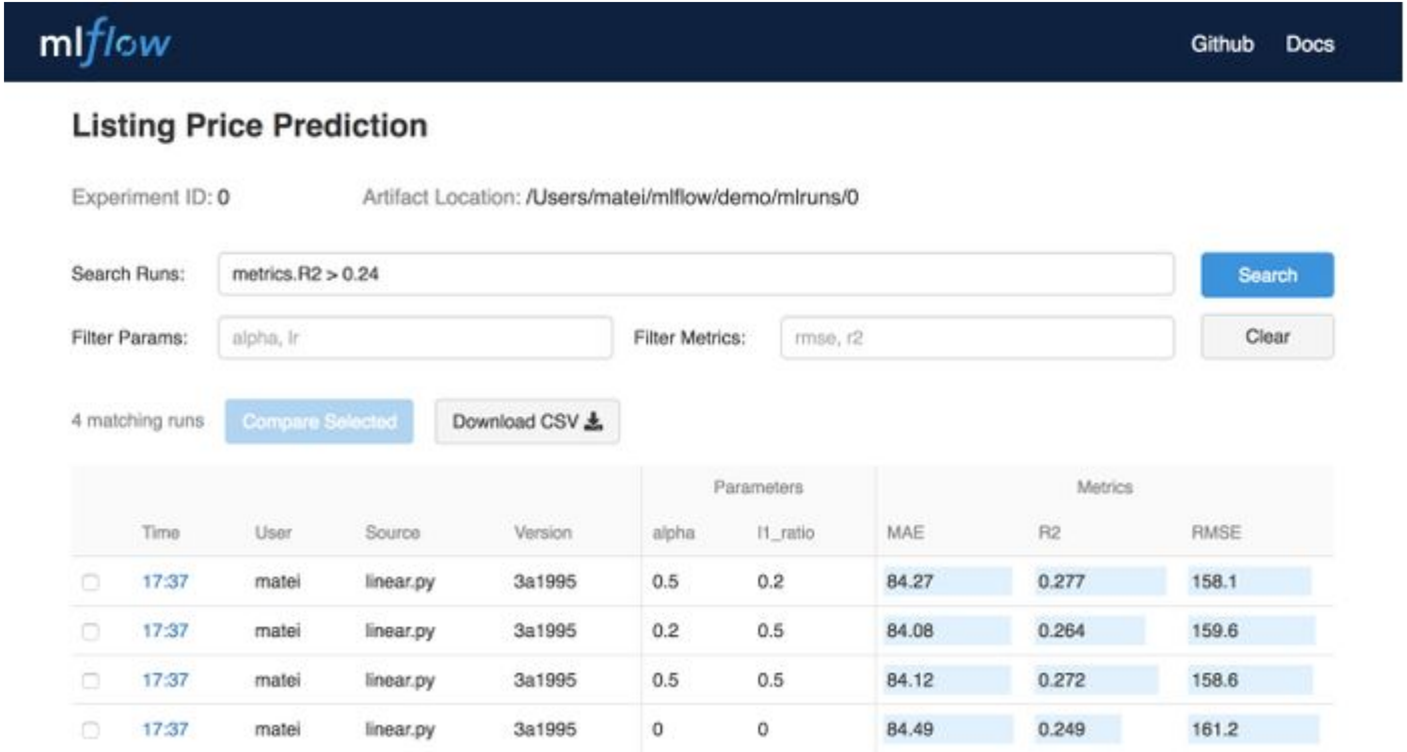
```
def execute(config: Config): Unit = {
 val kerasModel = KerasModelImport.importKerasModelAndWeights(
  config.kerasModelPath, false)
 ModelSaver.writeModel(kerasModel, config.outputModelPath)
}
```

Workflow Train



Monitoring

Monitoring: mlflow



Listing Price Prediction

Experiment ID: 0 Artifact Location: /Users/matei/mlflow/demo/mlruns/0

Search Runs: metrics.R2 > 0.24

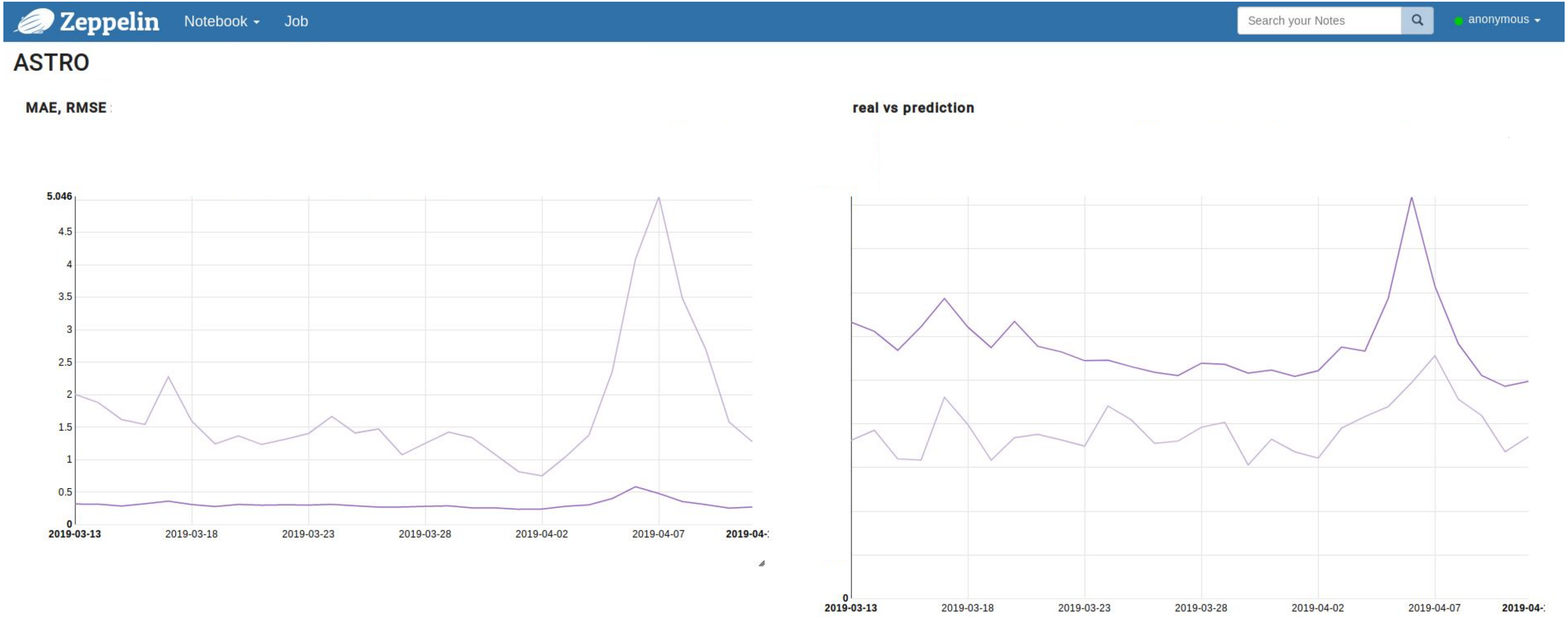
4 matching runs

					Parameters		Metrics		
	Time	User	Source	Version	alpha	l1_ratio	MAE	R2	RMSE
□	17:37	matei	linear.py	3a1995	0.5	0.2	84.27	0.277	158.1
□	17:37	matei	linear.py	3a1995	0.2	0.5	84.08	0.264	159.6
□	17:37	matei	linear.py	3a1995	0.5	0.5	84.12	0.272	158.6
□	17:37	matei	linear.py	3a1995	0	0	84.49	0.249	161.2

Monitoring: mlflow

- Ensure your training machine can reach mlflow server
- Keep track of your experiment
  - Training parameter
  - Performance
- Compare results
- (model repository, standardize model packaging, easy deployment)

Monitoring: Zeppelin



ASTRO

MAE, RMSE

real vs prediction

Monitoring: Zeppelin

- Already in HDP
- Authentication
- Scheduling
- Report View
- Auto shutdown
- Can mix sources (Scala, JDBC, C*, …)
- API to automate deployment

Thank you for your attention

Any questions?

https://github.com/scauglog/prez