

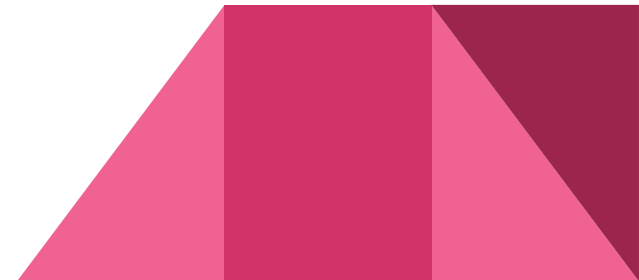
Machine Learning to moderate ads in real world classified's business

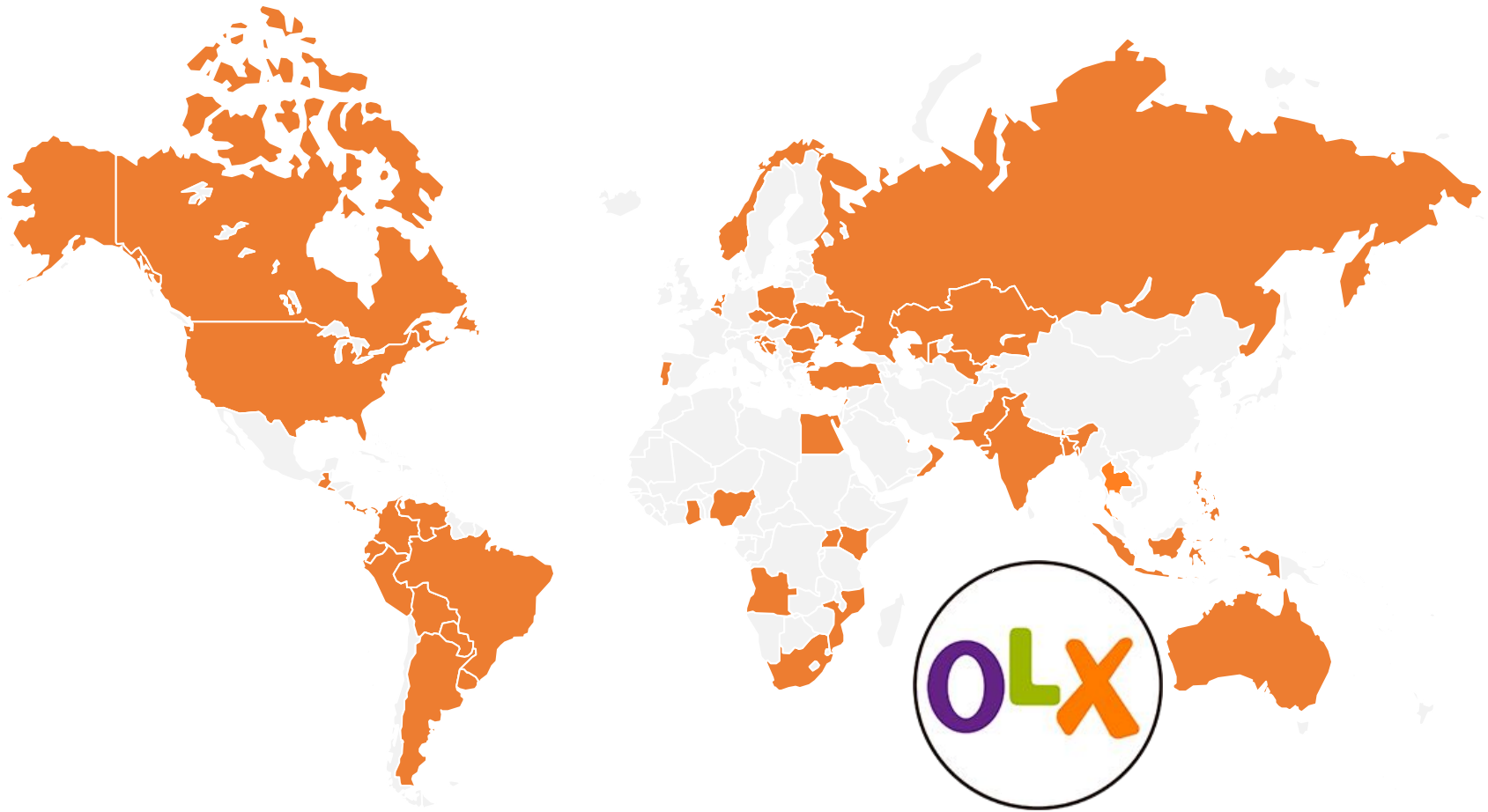
by Vaibhav Singh & Jaroslaw Szymczak



Agenda

- Moderation problem
- Offline model creation
 - feature generation
 - feature selection
 - data leakage
 - the algorithm
- Model evaluation
- Going live with the product
 - is your data really big?
 - automatic model creation pipeline
 - consistent development and production environments
 - platform architecture
 - performance monitoring





50+
countries

60+ million
new monthly listings

18+ million
unique monthly sellers

What do moderators look for?

Avoidance of payment

Sell another item in paid listing by changing its content

Flood site with duplicate posts to increase visibility

Create multiple accounts to bypass free ad per user limit

Violation of ToS

Add Phone numbers, Company information on image rather than in description or dedicated fields

Try to sell forbidden items, very often with title and description that try to evade keyword filters

Miscategorized listings

Item is placed in wrong category

Item is coming from legitimate business, but is marked as coming from individual

'Seek' problem in job offers

Offline model creation

Feature engineering...

... and selection

Feature selection:

- necessary for some algorithms, for others - not so much
- most important features
- avoiding leakage

Feature generation - one-hot-encoding

	category
1	cars
2	pets
3	phones
4	fashion
5	fashion
6	phones
7	hobby
8	hobby
9	cars
10	pets



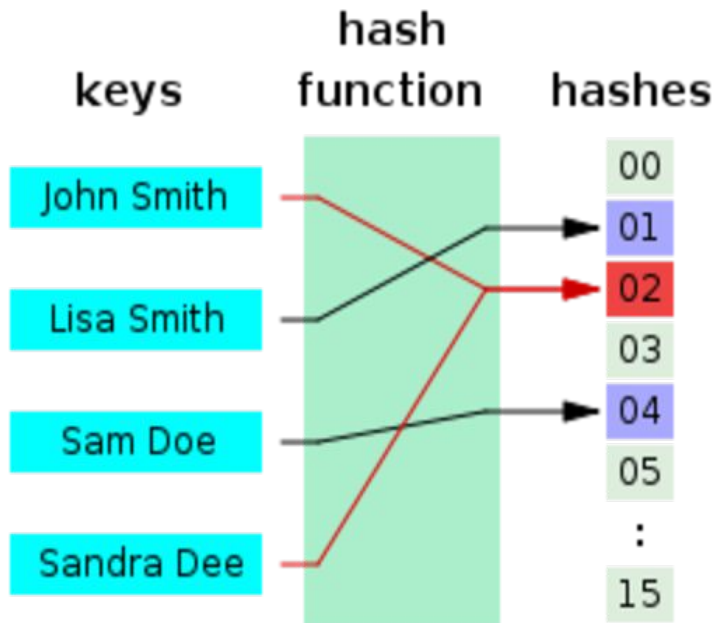
	category_cars	category_fashion	category_hobby	category_pets	category_phones
1	1	0	0	0	0
2	0	0	0	1	0
3	0	0	0	0	1
4	0	1	0	0	0
5	0	1	0	0	0
6	0	0	0	0	1
7	0	0	1	0	0
8	0	0	1	0	0
9	1	0	0	0	0
10	0	0	0	1	0

Feature generation - feature hashing

	category	bucket
1	phones	2
2	hobby	1
3	fashion	2
4	fashion	2
5	fashion	2
6	pets	1
7	hobby	1
8	phones	2
9	pets	1
10	hobby	1



	category_bucket_1	category_bucket_2
1	0	1
2	1	0
3	0	1
4	0	1
5	0	1
6	1	0
7	1	0
8	0	1
9	1	0
10	1	0



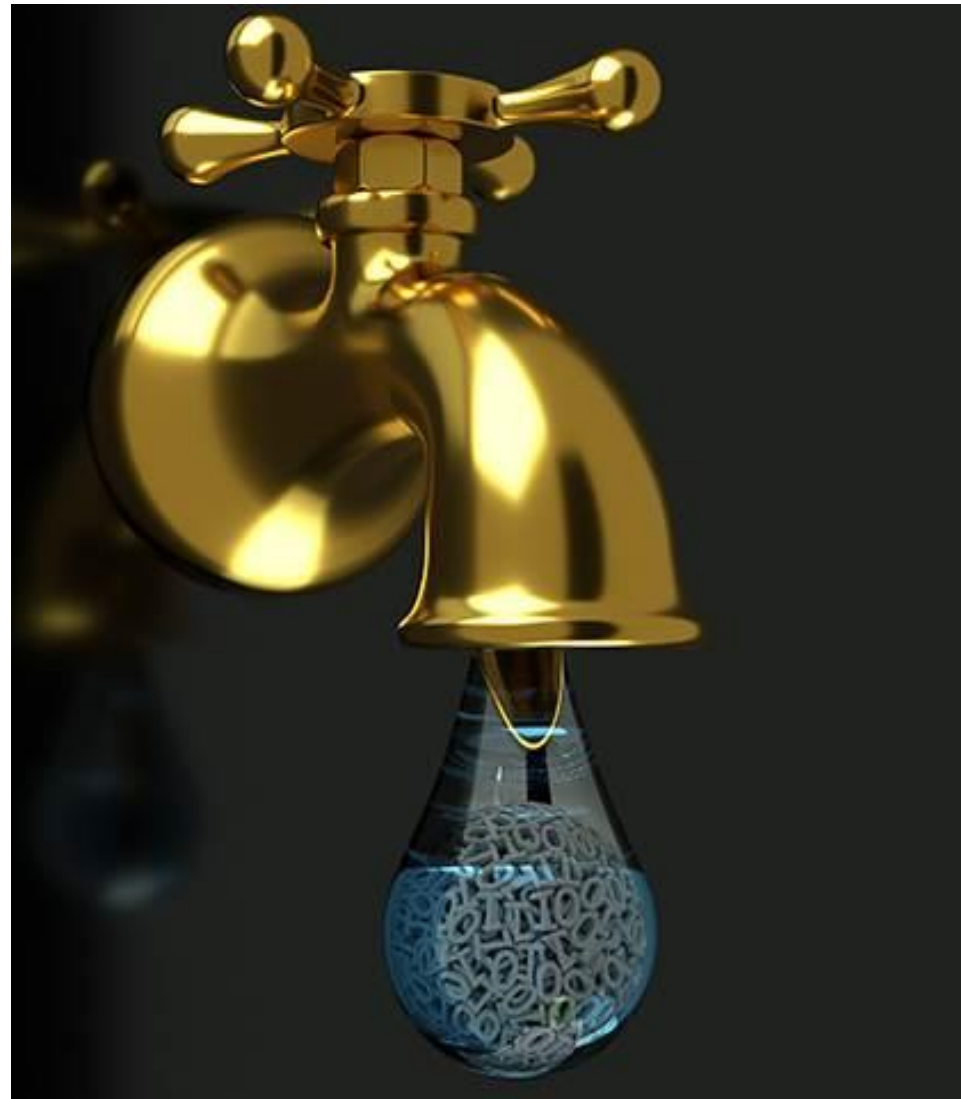
Feature hashing

- Good when dealing high dimensional, sparse features -- dimensionality reduction
- Memory efficient
- Cons - Getting back to feature names is difficult
- Cons - Hash collisions can have negative effects



Data Leakage

- Remove obvious fields
e.g.: id, account numbers
- Check the importance of
the features for any
unusual observations
- Have hold-out set that you
do not process wrt. target
variable
- Closely monitor live
performance



The algorithm

Desired features:

- state-of-the-art structured binary problems
- allowing reducing variance errors (overfitting)
- allowing reducing bias errors (underfitting)
- has efficient implementation

eXtreme Gradient Boosting (XGBoost)



Model evaluation

**I HAVE 99%
ACCURACY**

**I HAVE NO
IDEA WHAT
I'M DOING**



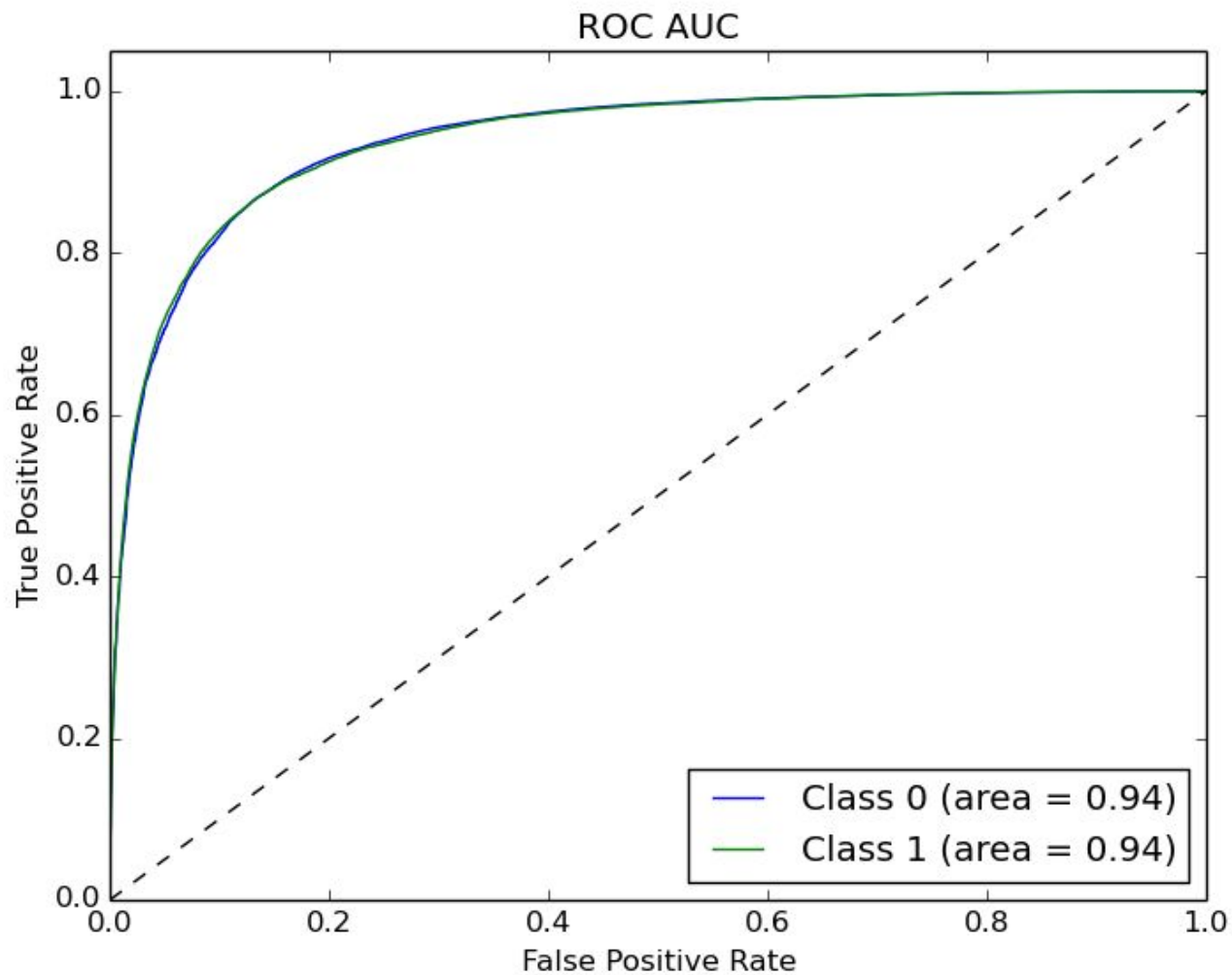
**SO GREAT, MY BOSS WILL
LOVE IT**

Beyond accuracy

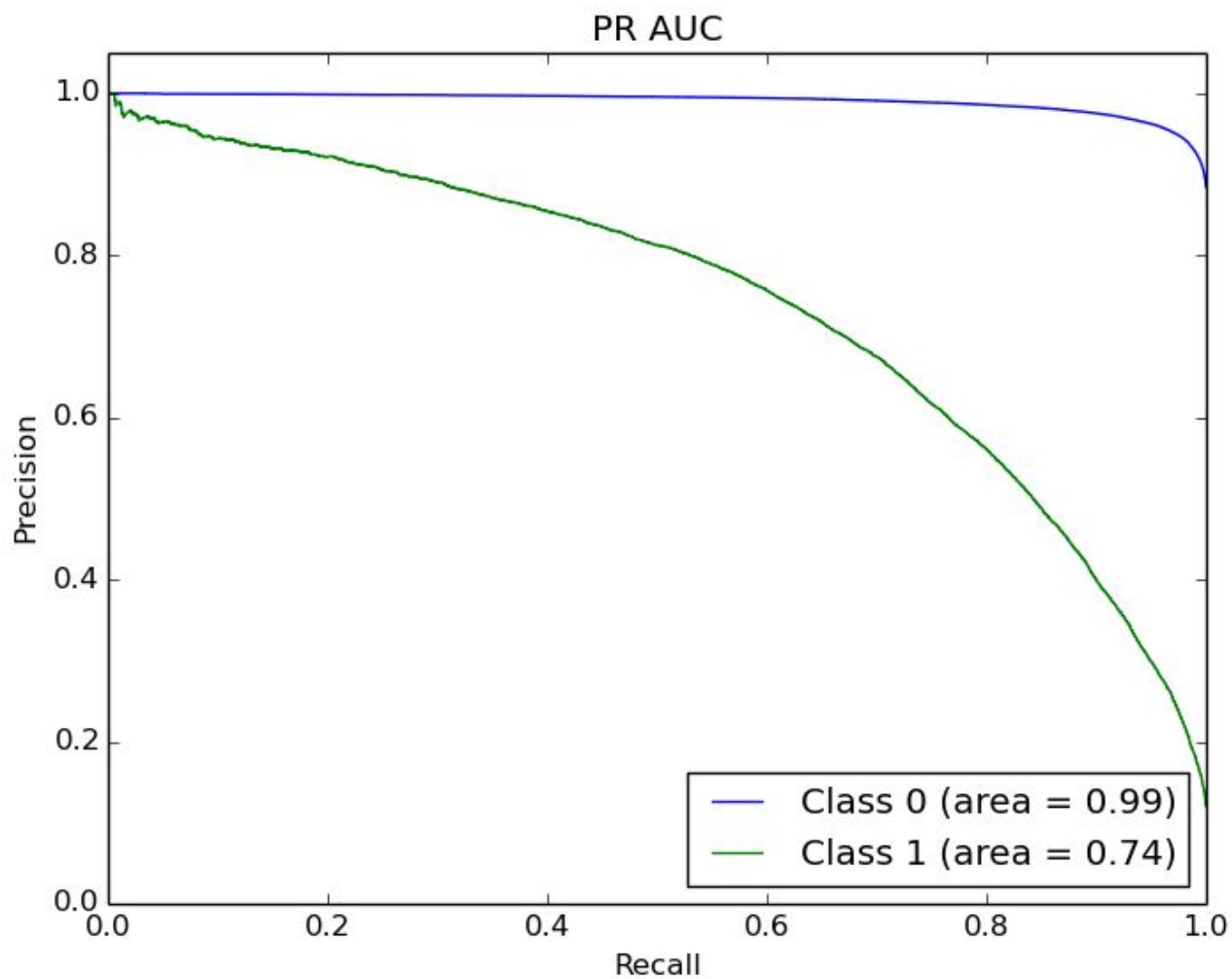
- ROC AUC (Receiver-Operator Curve):
 - can be interpreted as concordance probability (i.e. random positive example has the probability equal to AUC, that it's score is higher)
 - it is too abstract to use as a standalone quality metric
 - does not depend on classes ratio
- PRC AUC (Precision-Recall Curve)
 - Depends on data balance
 - Is not intuitively interpretable
- Precision @ fixed Recall, Recall @ fixed Precision:
 - can be found using thresholding
 - they heavily depend on data balance
 - they are the best to reflect the business requirements
 - and to take into account processing capabilities
(then actually Precision @k is more accurate)
- choose one, and only one as your KPI and others as constraints



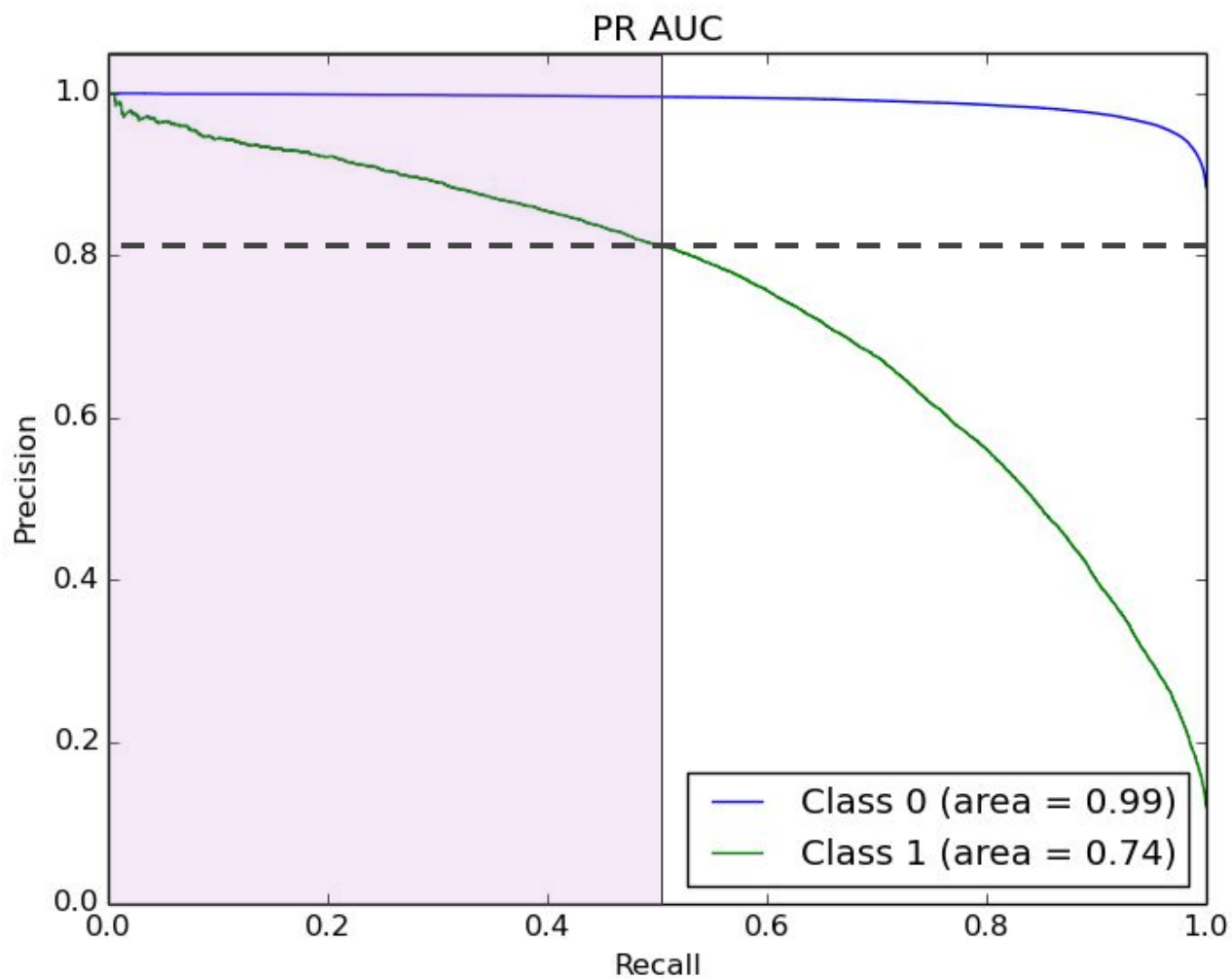
Example ROC for moderation problem



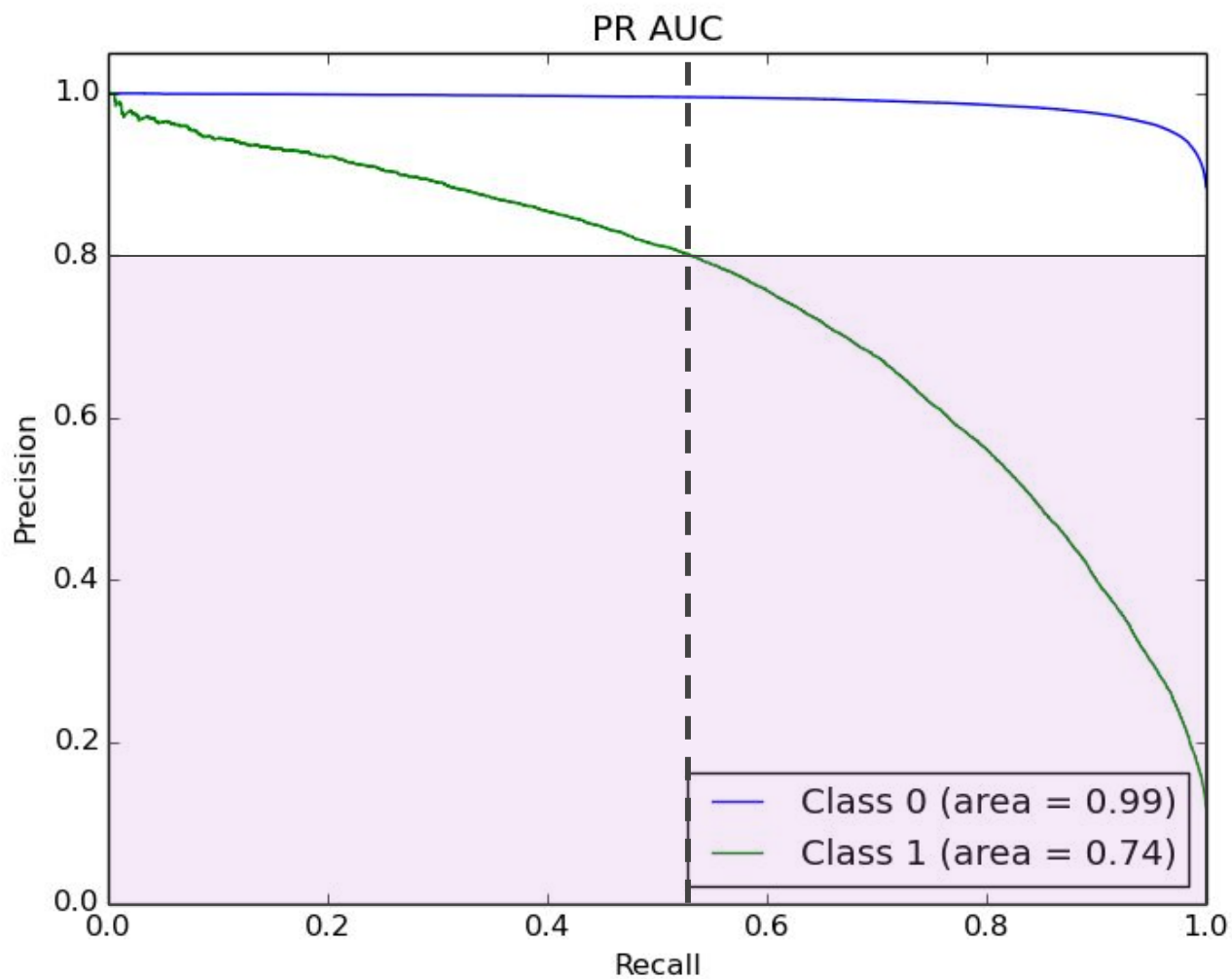
Precision-recall curve example



Precision @recall

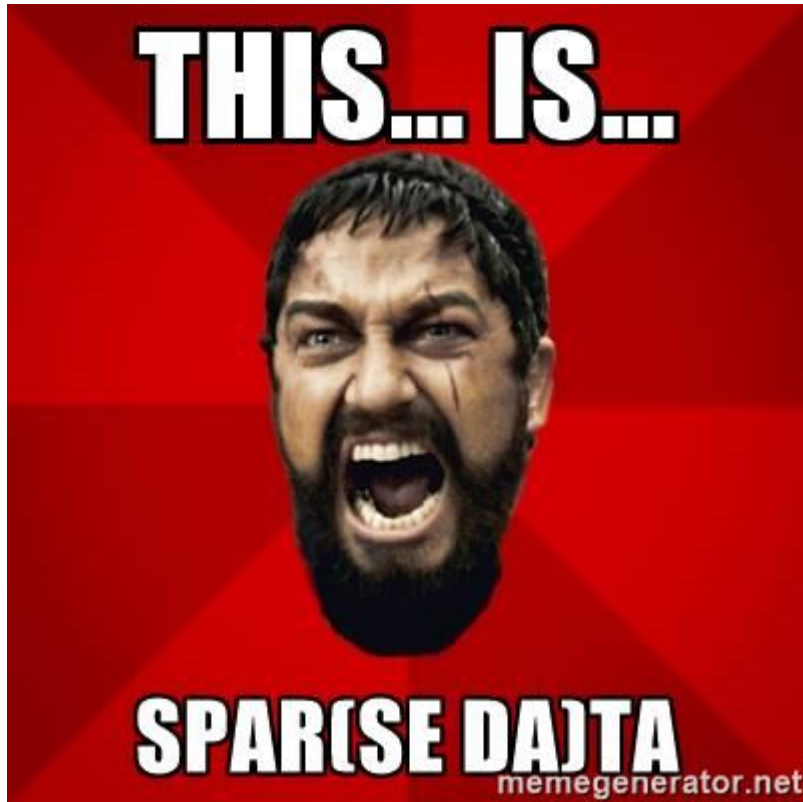


Recall @precision



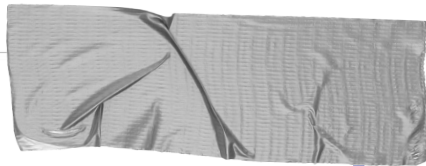


Going live with the product



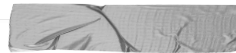
Is your data

really big?

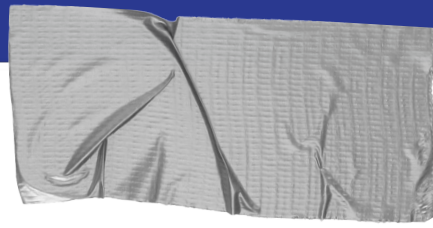


SVM Light Data Format

- Memory Efficient.
Features can be created
on one machine and do
not require huge clusters
- Cons - Number of
features is unknown,
store it separately



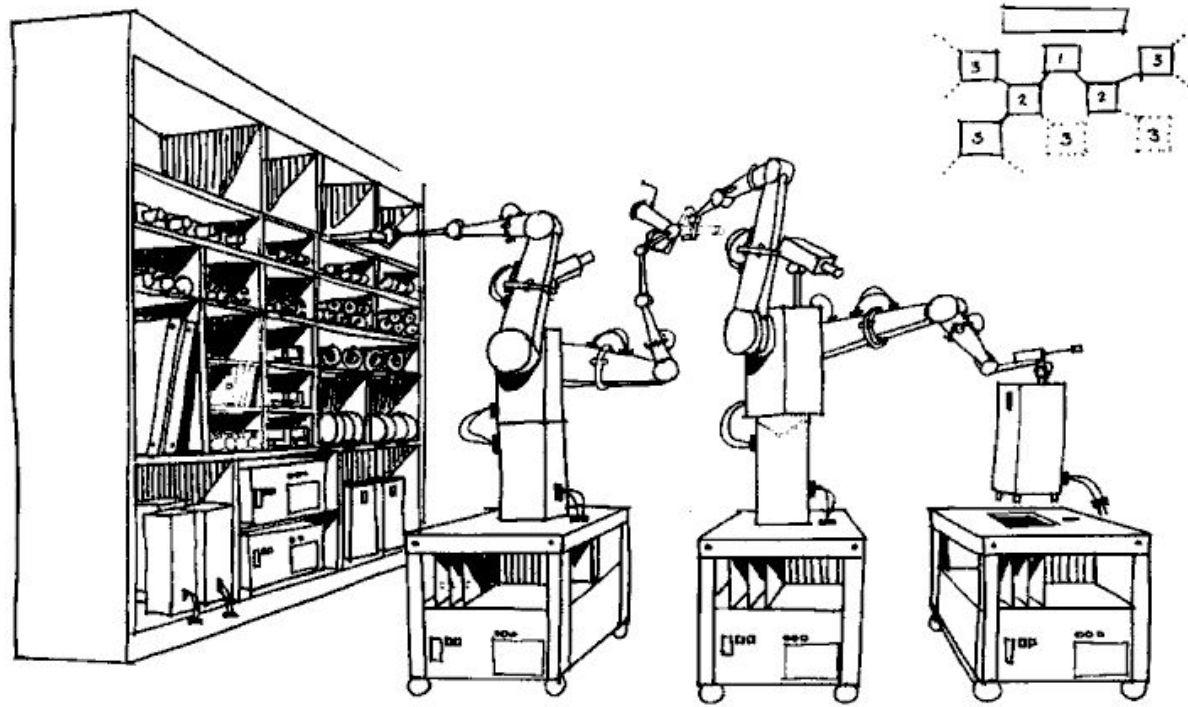
```
1 191:-0.44 87214:-0.44 200004:0.20 200012:1 206976:1 206983:-1 207015:1 207017:1 226201:1
1 1738:0.57 130440:-0.57 206999:0.32 207000:28 207001:6 207013:1 207015:1 207017:1 226300:1
0 2812:-0.63 34755:-0.31 206995:2.28 206997:1 206998:2 206999:0.00 207000:1 207001:28 226192:1
1 4019:0.35 206999:0.43 207000:40 207001:18 207013:1 207014:1 207016:1 226261:1
0 8903:0.37 207000:4 207001:14 207013:1 207014:1 207016:1 226262:1
1 5878:-0.27 206995:2.28 206998:1 206999:5.80 207000:1 207001:24 226187:1
```



Lessons Learnt

- Do not go for distributed learning if you don't need to
- Choose your tech dependent on data size. Do not go for hype driven development
- Your machine does not limit, there's cloud
- Ask yourself: What's the most difficult problem to scale ? → People

Model Generation Pipeline



Automate

Automatic model creation pipeline

- Automation makes things deterministic
- Airflow, Luigi and many others are good choice for Job dependency management

Luigi Dashboard

Luigi Task Status

Task List

Dependency Graph

Workers

Resources

TASK FAMILIES

1 ConsolidateTrainingData

139 DownloadExamplesFromOne

1 DownloadManyDays

1 ExtractClassicFeaturesAndClass


1 GenerateImageFeatures

1 GenerateImageList


1 GenerateNNInput

1 TestXGBoost


1 TrainXGBoost




PENDING TASKS
7




RUNNING TASKS
1




BATCH RUNNING TA...
0




DONE TASKS
139




FAILED TASKS
0



UPSTREAM FAILURE
0



DISABLED TASKS
0







UPSTREAM DISABLED
0

Show 10 entries

Filter table:

Filter on Server ☐

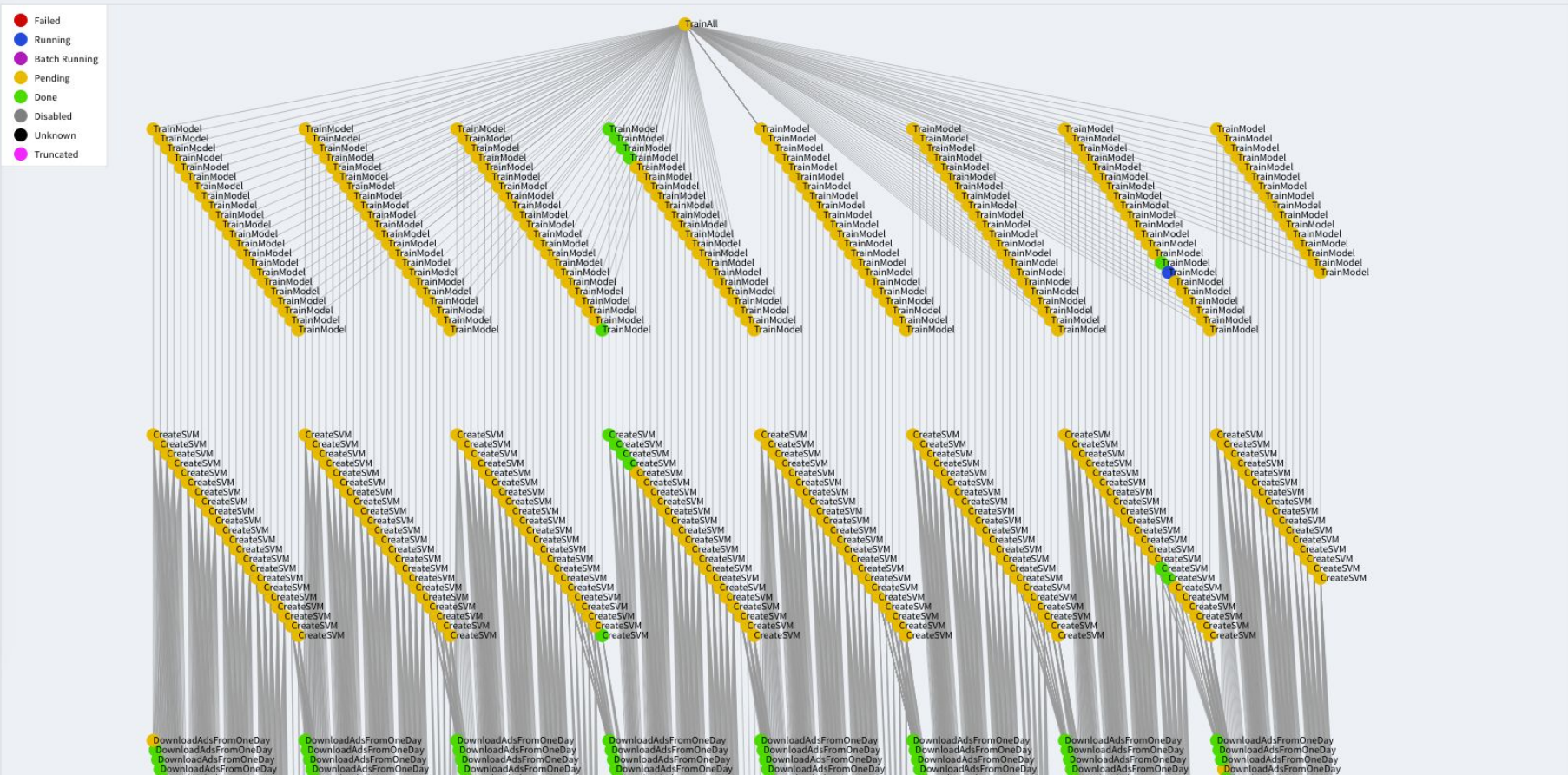
	Name	Details	Priority	Time	Actions
	RUNNING DownloadManyDays	examples_limit=100000, site_code=olxpl, date_interval=2017-01-01-2017-05-20, data_root=/home/batch/bad_images/, neg_examples_pct_retention=15	0	2017-6-9 01:32:54 0 minutes	
	PENDING GenerateNNInput	image_size=224, date_interval=2017-01-01-2017-05-20, neg_examples_pct_retention=15, examples_limit=100000, site_code=olxpl, data_root=/home/batch/bad images/.	0	2017-6-9 01:32:53	

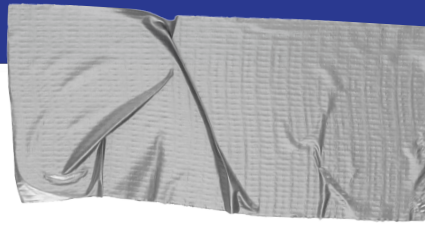
Luigi Task Visualizer

☐ ☐

Visualisation Type D3 SVG

TrainAll()
Dependency Graph





Lessons Learnt

- when you use the output path on your own, create your output at the very end of the task
- you can dynamically create dependencies by yielding the task
- adding workers parameter to your command parallelizes task that are ready to be run (e.g. `python run.py Task ... --workers 15`)



kubernetes



Consistent development
and production
environments

Model Serving Architecture

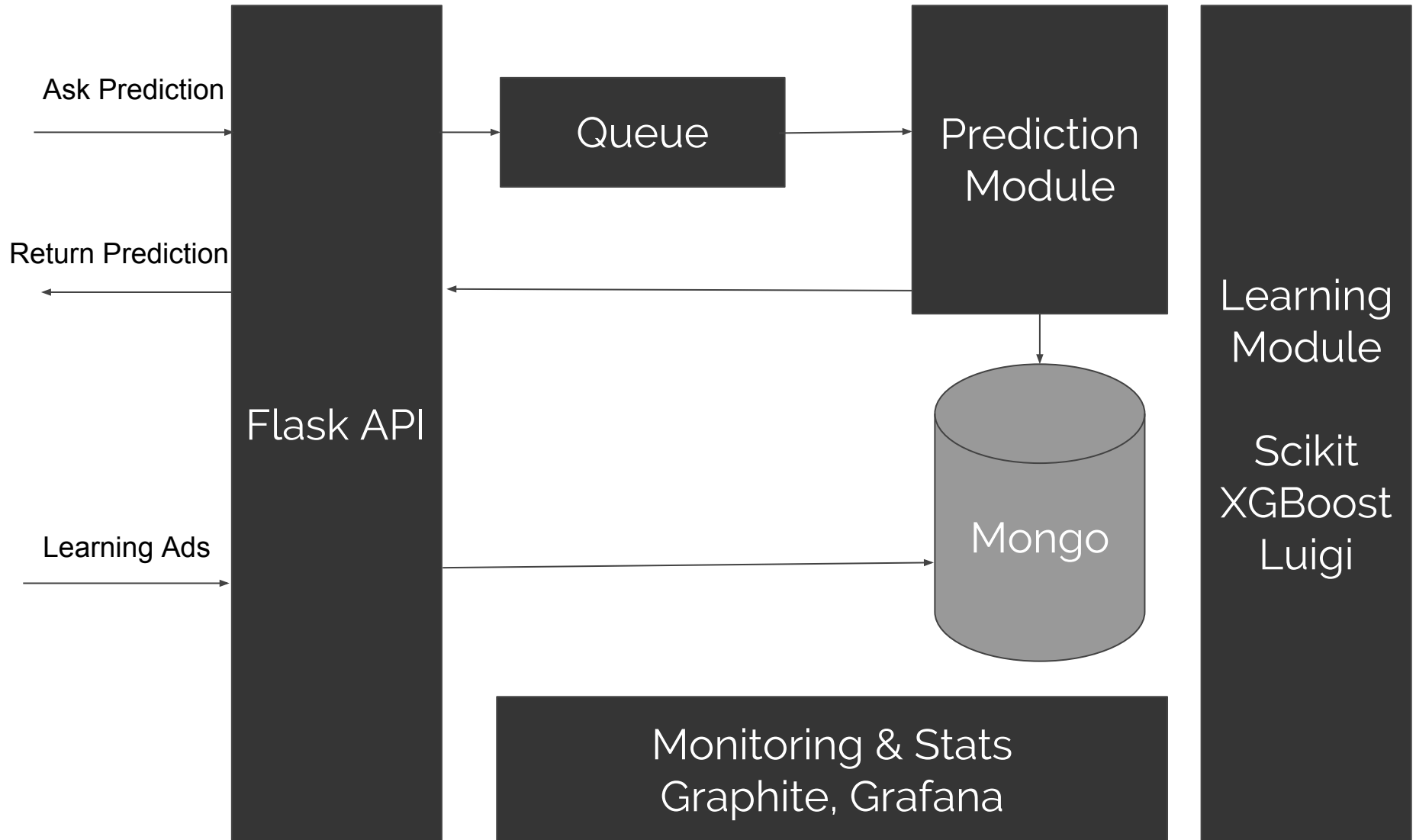
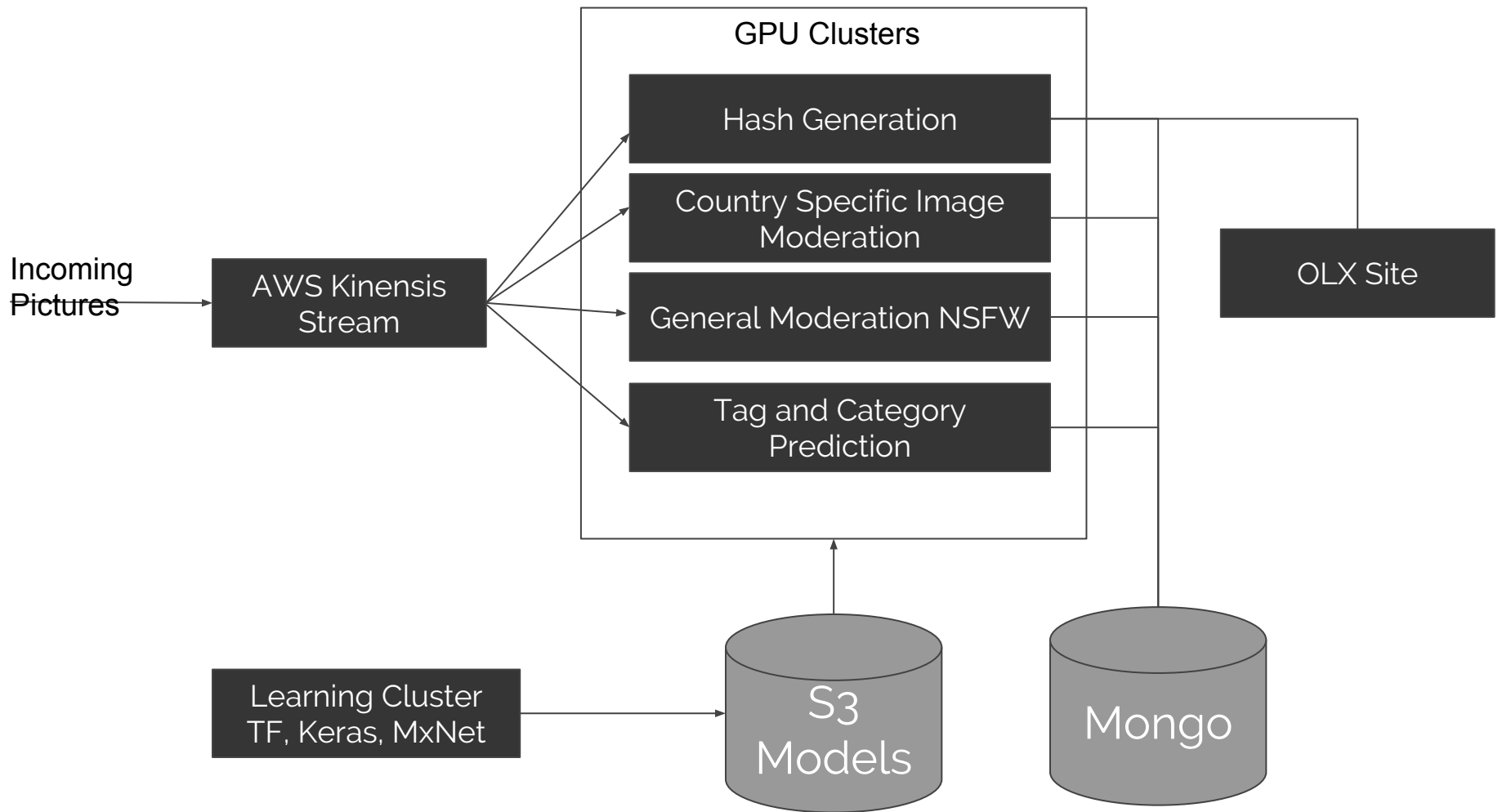
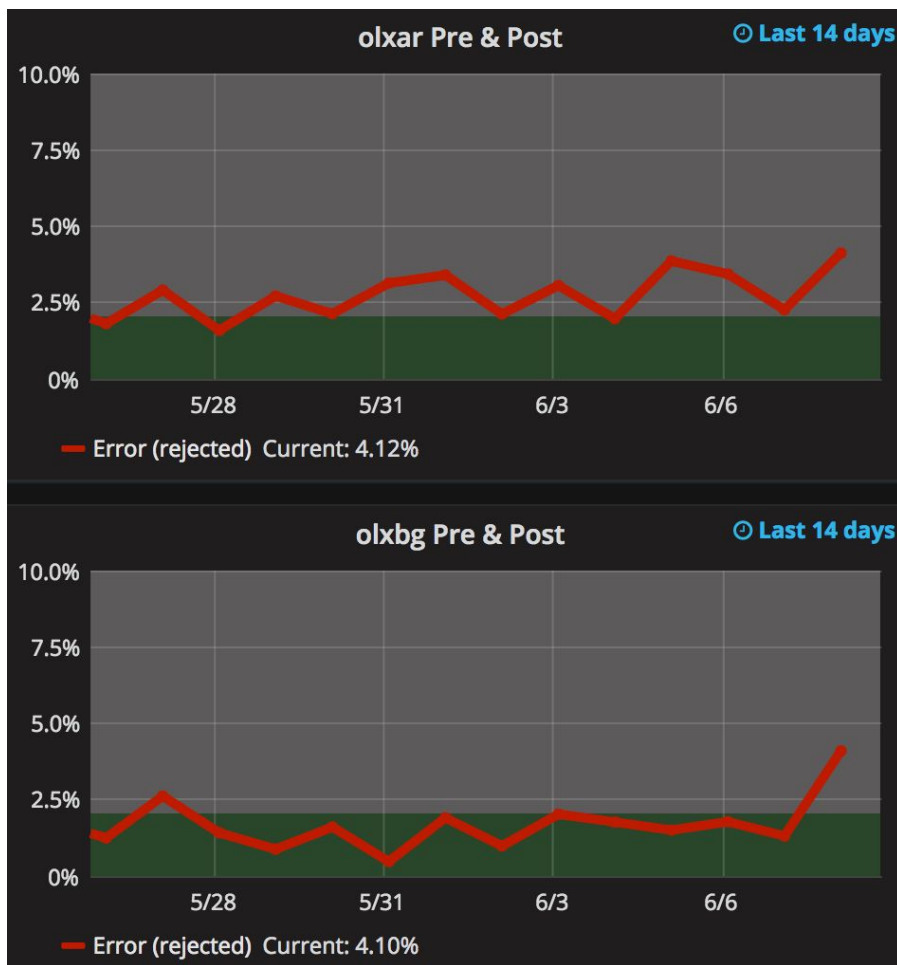


Image Model Serving Architecture



Performance monitoring

Model monitoring and management



Talos APP 7:12 PM ☆

Report of the day: 07-06-2017

olxpl

Accepted decision error

Acceptance precision: 0.951881848499

Acceptance precision with ignored rejection reasons: 0.941564561734

Premoderation and Postmoderation decision precision

Moderation precision: 0.602865761689

Moderation precision with ignored rejection reasons: 0.606956262129

Highlight reasons precision

Highlight reasons precision: 0.579865771812

Highlight reasons recall: 0.271101349231

Highlight categories precision

Highlight categories precision: 0.735283159463

Highlight categories recall: 0.392814490446

olxza

Accepted decision error

Acceptance precision: 0.965517241379

Acceptance precision with ignored rejection reasons: 0.915887850467

Premoderation and Postmoderation decision precision

Moderation precision: 0.873857404022

Moderation precision with ignored rejection reasons: 0.876840696118

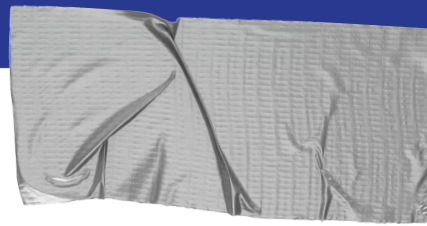
Highlight reasons precision

Highlight reasons precision: 0.871031746032

Highlight reasons recall: 0.163805970149

Highlight categories precision

Too small sample size to evaluate reliable precision and recall



Lessons Learnt

→ **Always Batch**

Batching will reduce CPU Utilization and the same machines would be able to handle much more requests

→ **Modularize, Dockerize and Orchestrate**

Containerize your code so that it is transparent to Machine configurations

→ **Monitoring**

Use a monitoring service

→ **Choose simple and easy tech**

Acknowledgements

- Andrzej Prałat
- Wojciech Rybicki



Vaibhav Singh
vaibhav.singh@olx.com

Jaroslav Szymczak
jaroslav.szymczak@olx.com