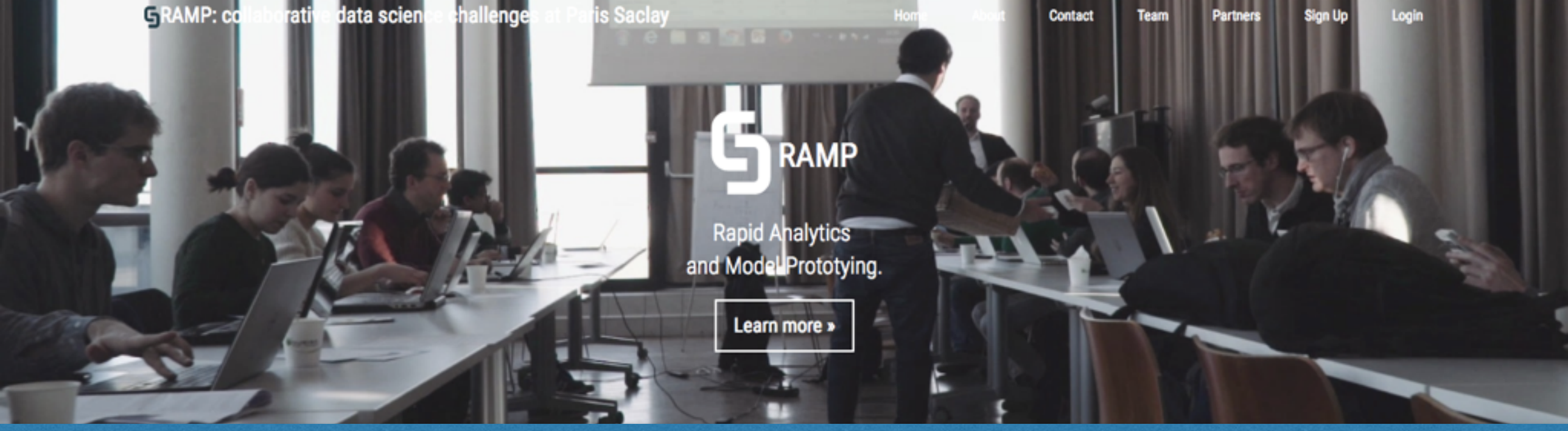
A rapid introduction to RAMP

Akin KAZAKCI
Associate Professor
MINES ParisTech - PSL Research University
osmanakin@gmail.com

RAPID 2018, Dortmund

Who am I?

- Associate Professor in Management Sciences, MINES ParisTech, France
- 15+ years of research experience
  - Operations Research / Optimisation / Decision under uncertainty
  - Innovation Management and Design Theory
  - Management of data science process / Digital transformation
- Part of the RAMP team



Go register at **www.ramp.studio**

Collaborate

During the RAMP, the participants submit predictive solutions (code). The models are trained on our back-end. The scores are displayed on a leaderboard. In the open phase, all participants have access to all code, and they are encouraged to look at and to reuse each other's solutions. This accelerates the development process since good ideas spread fast.

Prototype

During the RAMP we blend the best models, usually achieving a better score than the best submission. Since code is submitted, the blended prototype can be delivered to the organizer, ready to be inserted into a production pipeline, either as code or by exposing it through and API.

Learn from fellow data scientists

A great tool to learn data science! Since all code is open, novice participants can learn from the pros. RAMPs are used in the MS Big Data at Telecom ParisTech, in three UPSaclay M2 programs (Data Science, AIC, Data and Knowledge), in M1 at Polytechnique, and in various in courses beyond Saclay.

Be part of a growing community

RAMPs attract participants coming from different backgrounds and carreer stages who usually meet for the first time. They develop a working relationship in a relaxed environment, and sometimes keep working together after the event.

User - RAMP

https://ramp.studio/problems

RAMP

Current RAMPs

- **Mechanics classification**
  - Saclay M2 Data Camp 2018/19, number of participants = **143**, number of submissions = **0**, combined score = **None**, click here for score vs time plot
- **Solar wind classification**
  - Saclay M2 Data Camp 2018/19, number of participants = **144**, number of submissions = **31**, combined score = **0.24**, click here for score vs time plot
  - Solar wind hackathon, number of participants = **28**, number of submissions = **33**, combined score = **0.21**, click here for score vs time plot
- **Storm intensity forecast**
  - initial hackaton, number of participants = **14**, number of submissions = **22**, combined score = **11.8**, click here for score vs time plot
  - CI2018 hackaton, number of participants = **41**, number of submissions = **63**, combined score = **8.2**, click here for score vs time plot
- **Autism Spectrum Disorder classification**
  - 2018 data challenge, number of participants = **142**, number of submissions = **725**, combined score = **0.834**, click here for score vs time plot
- **Aircraft classification from radar trajectories**
  - Ecole des Mines 2017/18, number of participants = **91**, number of submissions = **511**, combined score = **1.696**, click here for score vs time plot
- **Kaggle Porto-Seguro safe driver prediction**
  - Kaggle RAMP team, number of participants = **35**, number of submissions = **138**, combined score = **0.2881**, click here for score vs time plot
- **Mars craters detection and classification**
  - General RAMP for benchmarking solutions, number of participants = **11**, number of submissions = **0**, combined score = **None**, click here for score vs time plot
  - Saclay M2 Data Camp 2018/19, number of participants = **143**, number of submissions = **0**, combined score = **None**, click here for score vs time plot
  - Saclay M2 Data Camp 2017/18, number of participants = **56**, number of submissions = **78**, combined score = **0.269**, click here for score vs time plot
- **Fake news: classify statements of public figures**
  - Tbilisi DataFest 2017, number of participants = **6**, number of submissions = **3**, combined score = **0.347**, click here for score vs time plot
  - Saclay M2 Data Camp 2017/18, number of participants = **131**, number of submissions = **740**, combined score = **0.491**, click here for score vs time plot
- **Pollenating insect classification (403 classes), simplified workflow**
  - Journées Nationales de l'Ingénieur data challenge collaborative phase, number of participants = **24**, number of submissions = **10**, combined score = **0.884**, click here
- **MNIST classification**
  - test event, number of participants = **13**, number of submissions = **8**, combined score = **0.966**, click here for score vs time plot
- **California winter extreme rainfall prediction**
  - single-day RAMP at Climate Informatics Workshop 2017, number of participants = **23**, number of submissions = **33**, combined score = **0.164**, click here for score vs
- **Pollenating insect classification (403 classes)**
  - Journées Nationales de l'Ingénieur 2017 data challenge, number of participants = **47**, number of submissions = **74**, combined score = **0.883**, click here for score vs t
- **MNIST classification**
  - test event, number of participants = **5**, number of submissions = **1**, combined score = **0.594**, click here for score vs time plot
- **Cell population identification from single-cell mass cytometry data**

Collaborative challenge with code submission

drug_spectra_mines_201617

Leaderboard

Combined score: 0.023

Show 100 entries

Search:

team	submission	contributivity	historical contributivity	combined	err	mare	train time [s]	test time [s]	submitted at (UTC)
hugueslhomas218	correct_concent_v1	8	4	0.028	0.014	0.054	304	1	2017-03-24 15:54:06 Fri
clement.acher	2classifiers	10	6	0.029	0.013	0.052	14	4	2017-03-25 10:42:41 Sat
clement.acher	pipeline_norm3	0	0	0.032	0.015	0.055	347	5	2017-03-24 14:39:38 Fri
clement.acher	pipeline_norm2	0	0	0.032	0.015	0.055	124	4	2017-03-24 14:18:48 Fri
antoine.godet	factor_anl	0	0	0.032	0.015	0.056	174	1	2017-03-24 14:22:44 Fri
jeang	test09	4	8	0.032	0.015	0.055	139	2	2017-03-23 23:50:58 Thu
nicolas.zhang	StdScaler	0	0	0.032	0.015	0.056	393	2	2017-03-24 14:55:11 Fri
jeang	testFalseFalse-36	1	1	0.032	0.015	0.054	395	5	2017-03-24 14:34:25 Fri
clement.acher	pipeline_norm	2	0	0.032	0.015	0.055	235	5	2017-03-24 13:23:35 Fri
jeang	testFalseQuadra-38	0	0	0.032	0.015	0.055	576	9	2017-03-24 14:48:47 Fri
nicolas.zhang	FastOptim	0	0	0.034	0.018	0.058	391	1	2017-03-24 15:59:03 Fri
hugueslhomas218	optimize_params_v4	0	0	0.034	0.014	0.073	393	1	2017-03-24 14:10:16 Fri
nicolas.zhang	FastPCA	1	0	0.034	0.018	0.057	424	2	2017-03-24 15:12:58 Fri
nicolas.zhang	FastOptim	0	0	0.034	0.018	0.058	327	1	2017-03-24 15:09:05 Fri
nicolas.zhang	optimPCA	0	0	0.034	0.018	0.058	428	2	2017-03-24 15:38:16 Fri
jeang	essai01	0	0	0.035	0.015	0.073	218	3	2017-03-23 20:59:38 Thu
majed.somai	aaaTwo	6	6	0.035	0.017	0.073	177	12	2017-03-24 13:48:20 Fri
hugueslhomas218	optimize_params_v3	0	0	0.035	0.017	0.072	340	1	2017-03-24 13:50:36 Fri

RAMP

Sandbox

You can either edit and save the code in the left column or upload the files in the right column. You can also import code from other submissions when the leaderboard links are open.

Edit and save your code!

object_detector

```python
import numpy as np


class ObjectDetector(object):
    def __init__(self):
        pass

    def fit(self, X, y):
        return self

    def predict(self, X):
        # loop over all events saved in X
        # list of of lists of all reconstructed vertices
        all_rec_vertex_list = []
        # loop over events
        for event in X:
            # random number of vertices according to Poisson distribution
            number_vertex = np.random.poisson(6)
            rec_vertex_in_event = []
            for i in range(0, number_vertex):
                x = np.random.normal(0., 0.05)
                y = np.random.normal(0., 0.05)
```

Don't forget to save the files!

Save

Productivity in Data Science:

Hackatons

- Collaborative
- Ill-defined and open

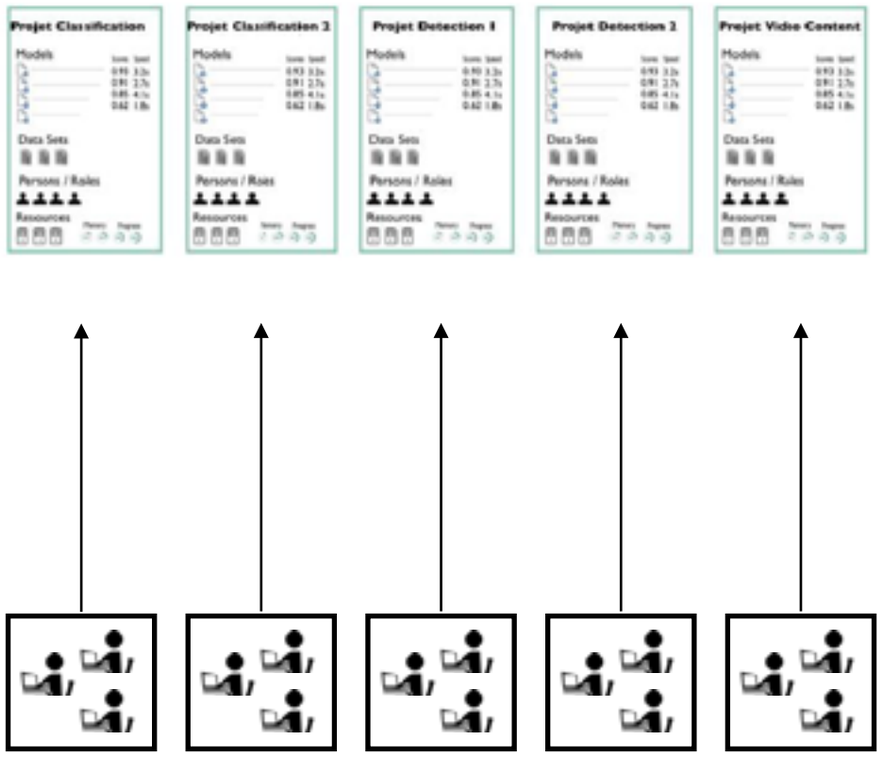
Challenges

- Well-defined (score)
- Boosts productivity
- Competitive

Using all the available brainpower for each problem

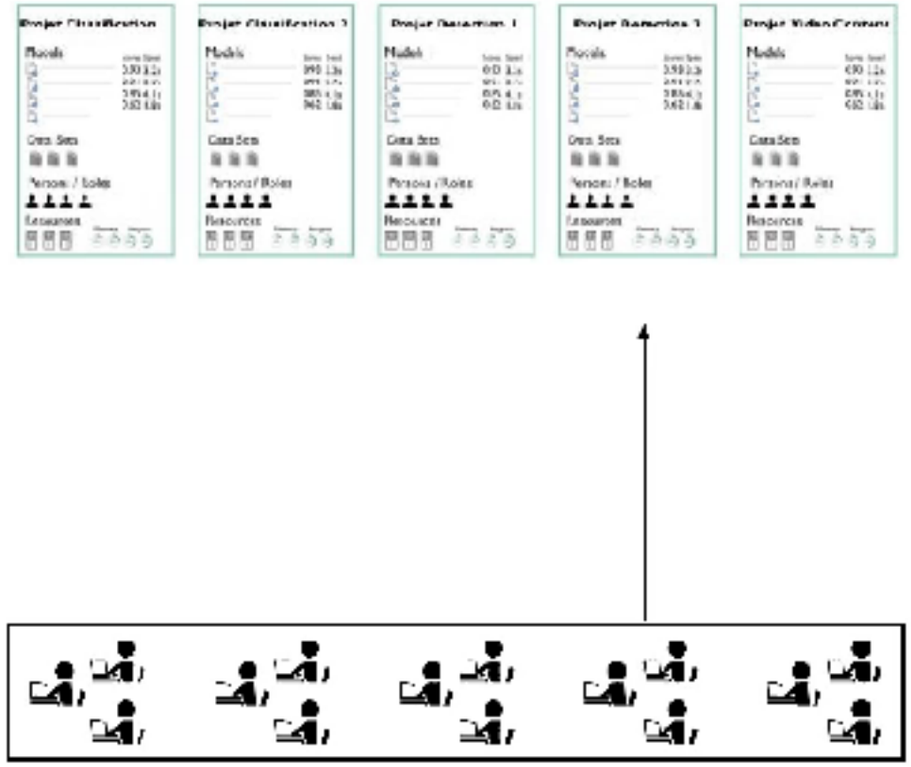
Traditional “labour division"

Small teams,
highly specialised



RAMP

Anybody can lead,
everybody can contribute



akin.kazakci@mines-paristech.fr

Separating problem formulation from problem solving

akin.kazakci@mines-paristech.fr

Lowering the entry barrier

RAMP on Primary Vertex reconstruction

1. Introduction
2. Preprocessing
3. Workflow
4. Evaluation
5. Local testing(expert)version
6. Submission

Introduction

The prediction task

References

Preprocessing

With no further do, let's have a look at the data.

Required dependencies and downloads

Installation of libraries and ramp-workflow

To get this notebook running and test your models locally using the `ramp_test_submission`, we recommend that you use the Python distribution from Anaconda or Miniconda.

```
In [1]: # !conda env create -f environment.yml  # use the environment.yml file to create the 'marc-cratering' env
        # !source activate vertex_finding         # activate the virtual environment
```

OR if you have Python already installed but are **not using Anaconda**, you'll want to use `pip`

```
In [2]: # !pip install -r requirements.txt
```

Download script (optional)

If the data has not yet been downloaded locally, uncomment the following cell and run it.

There are ~700Mb of images.

```
In [3]: # !python download_data.py
```

The input data

The input data consists of a json file for each event (where an event is a bunch crossing). The json file contains the hits in the Velo detector, Velo states from a simplified Kalman filter of the reconstructed velo tracks, as well as the MC truth information.

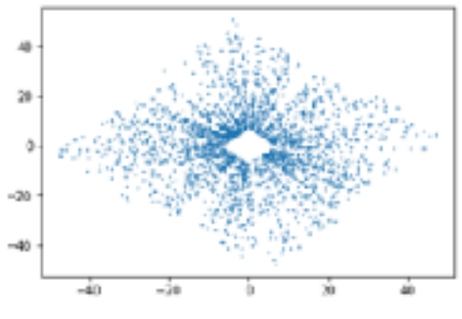
```
In [4]: import json
        from matplotlib import pyplot as plt
        from pprint import pprint
```

```
In [5]: jdata = json.load(open('data/train/RapidVPData_5719285_74665.json'))
        for key in jdata:
            print(key)
```

We can, for example, plot the x-y distribution of all hits:

```
In [9]: hits_x = [item['x'] for  key, item in VeloHits.items()]
        hits_y = [item['y'] for  key, item in VeloHits.items()]
        plt.plot(hits_x,hits_y, 'o', markersize=1.)
```

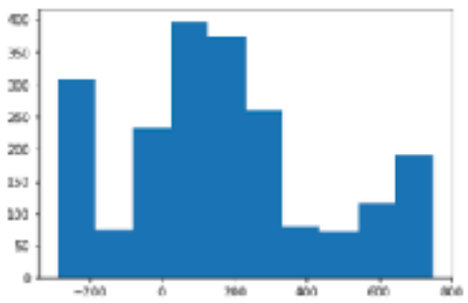
```
Out[9]: [<matplotlib.lines.Line2D at 0x7f28834d1550>]
```



or z distribution

```
In [10]: hits_z = [item['z'] for  key, item in VeloHits.items()]
         plt.hist(hits_z)
```

```
Out[10]: (array([308.,  74., 233., 397., 375., 261.,  79.,  73., 117., 389.]),
          array([-288.08369365, -188.21479492,  -88.34589619,   11.52300254,
                  111.39190127,  211.26080001,  311.12969874,  410.99859746,
                  510.86749619,  610.73639492,  710.60529365]),
          <a list of 10 Patch objects>)
```



The data is transformed from the json format to a (hopefully) more convenient form:

```
In [11]: import problem
```

```
In [12]: X, y = problem.get_train_data('.')
```

see data

X contains the training/test data, y the truth information.

```
In [13]: X.shape
```

```
Out[13]: (6,)
```

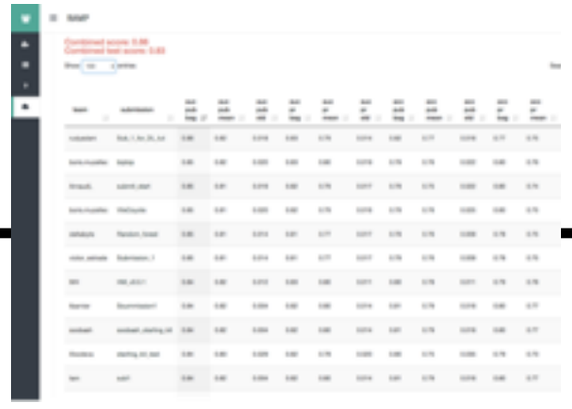
RAMP process

starting kit

a baseline solution

- usually built by an expert
- explains the domain problem, variables and the objectives
- enable other data scientists to become operational

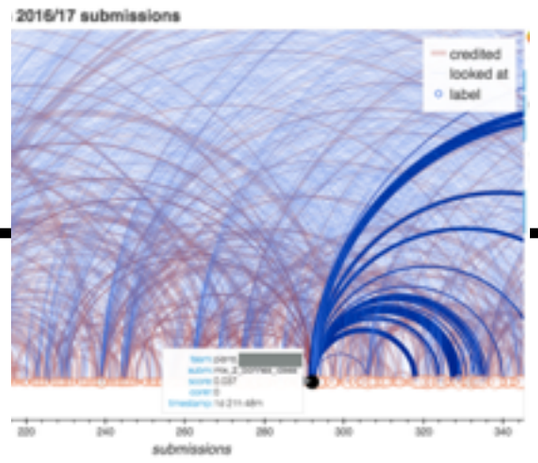
competition



a leaderboard

- gamifies model development
- boosts effort & exploration
- creates a diversity of models
- enhance problem understanding

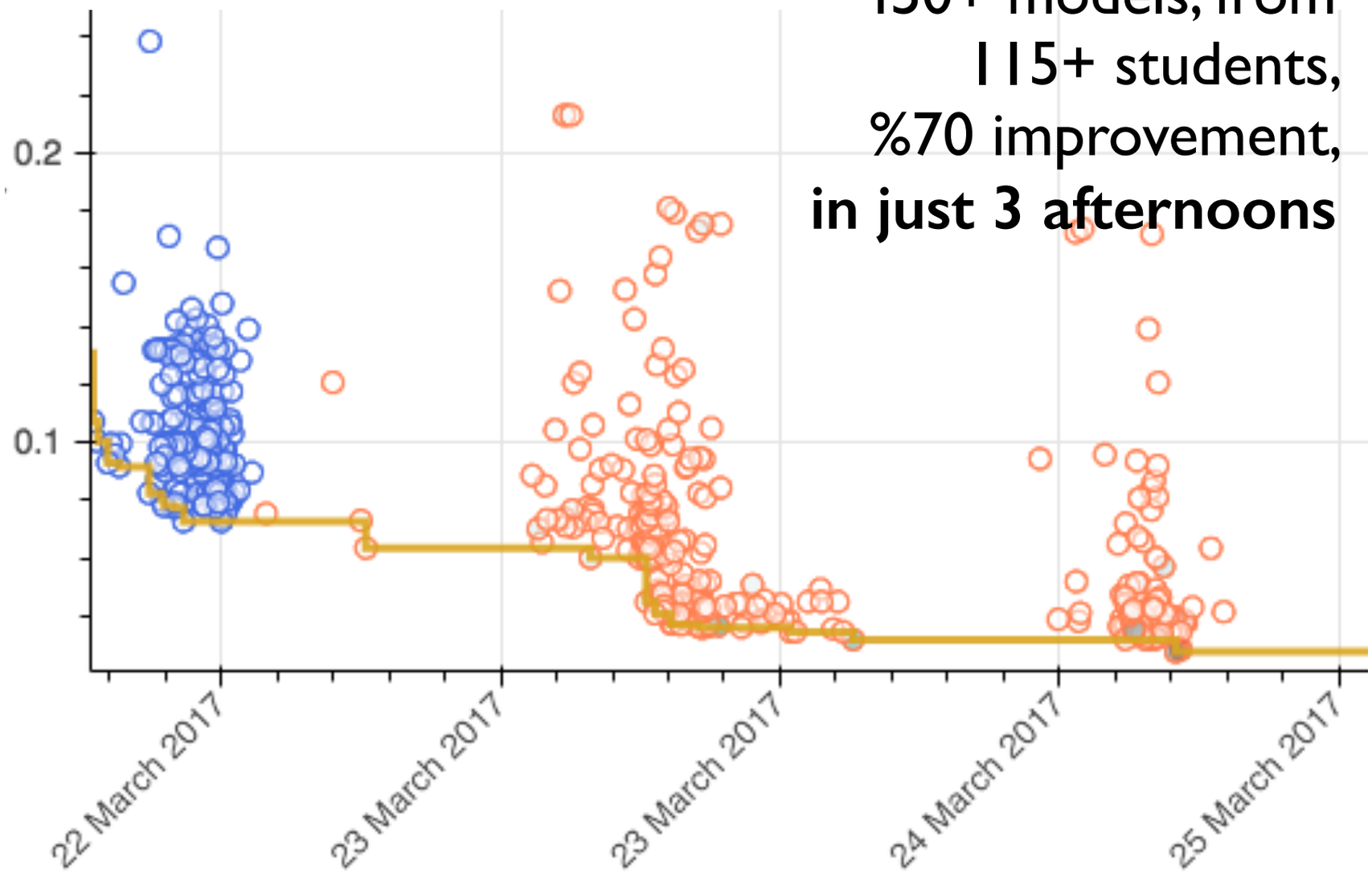
collaboration



propagating best ideas

- all models are open
- a wealth of information becomes accessible
- ideas & models are combined
- synergy & learning & creativity

What R.A.M.P does…

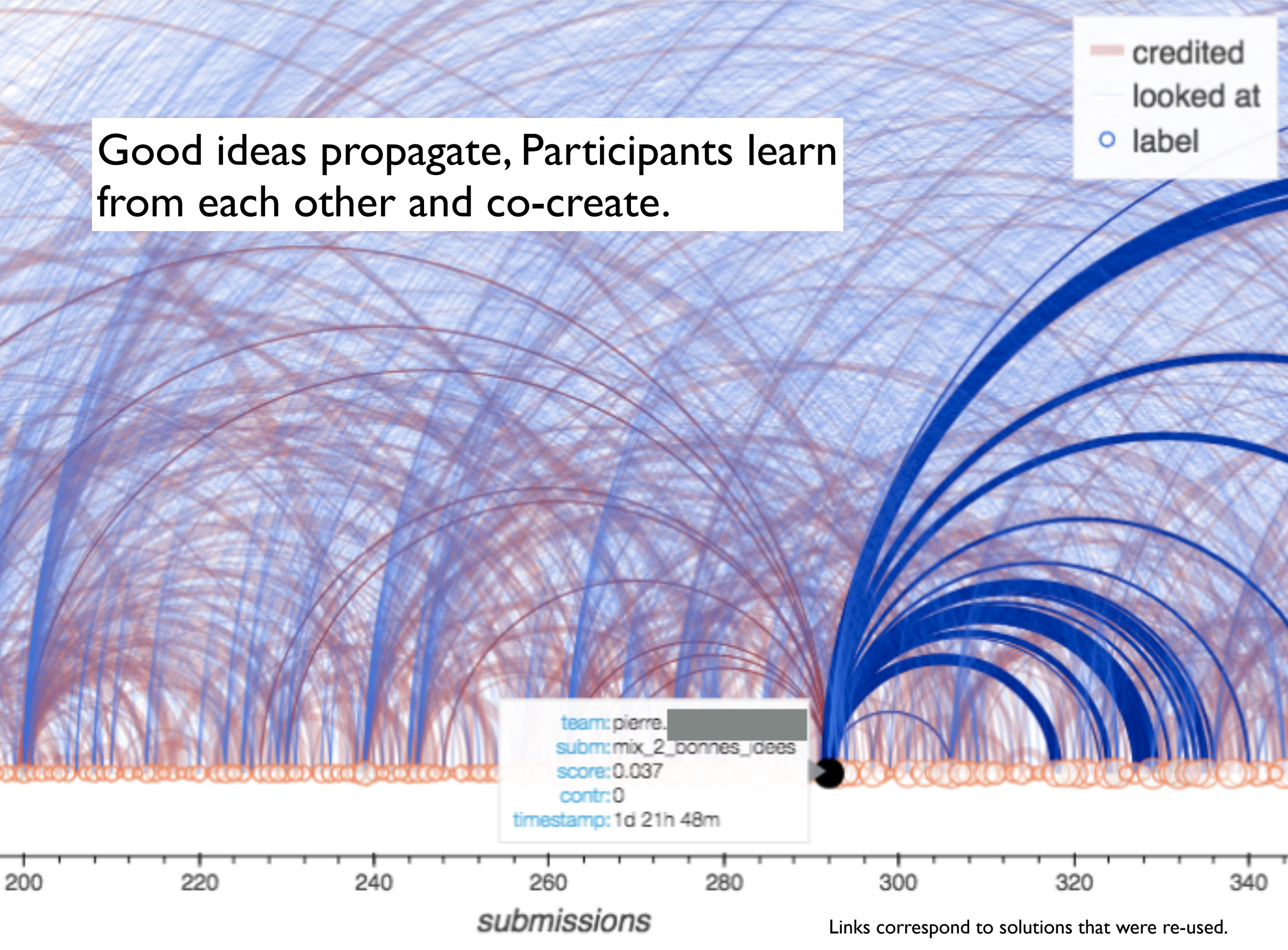


450+ models, from
115+ students,
%70 improvement,
in just 3 afternoons

better models. faster.

thanks to collaboration, competition, learning, experimenting

Good ideas propagate, Participants learn from each other and co-create.



Links correspond to solutions that were re-used.

Thank you

Akin KAZAKCI
osmanakin@gmail.com