Large-scale statistical computing Hack-a-thon

17-18th March Atlanta

Agenda

- Introduction Hack-a-thon
- PatientLevelPrediction R-package

Track 1: Unit testing and continuous integration

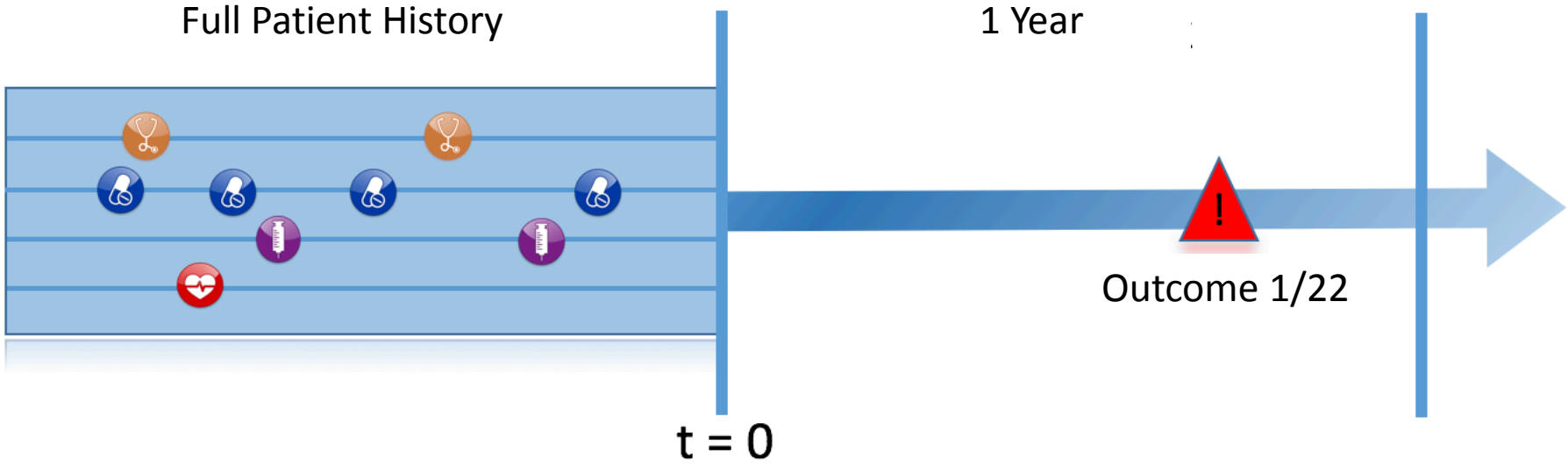
Track 2: Code base improvements

Track 3: Learning Curves, search space reduction

Introduction

Peter R. Rijnbeek, PhD
Erasmus MC Rotterdam
The Netherlands

Work done in 2016

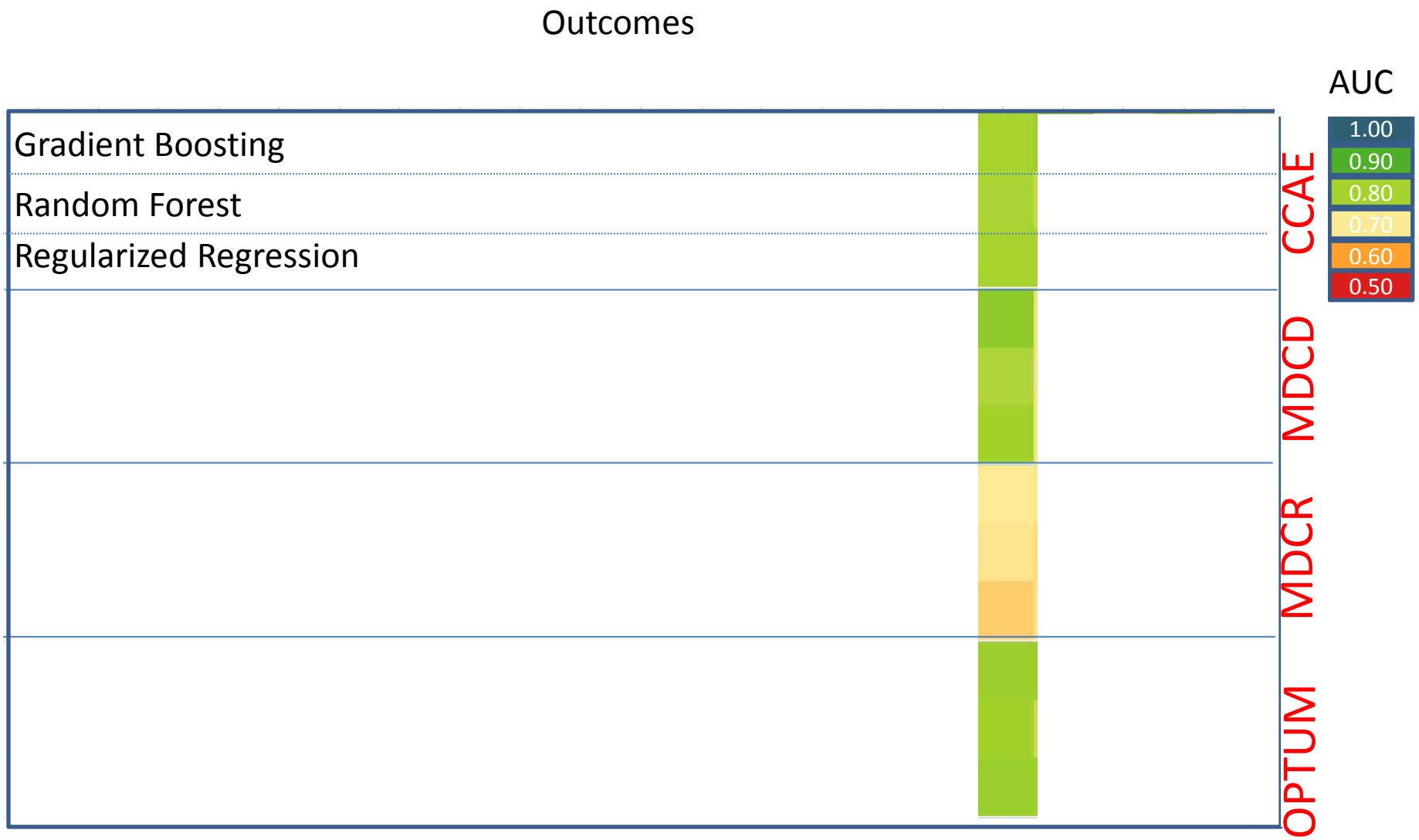


First Pharmaceutically Treated Depression

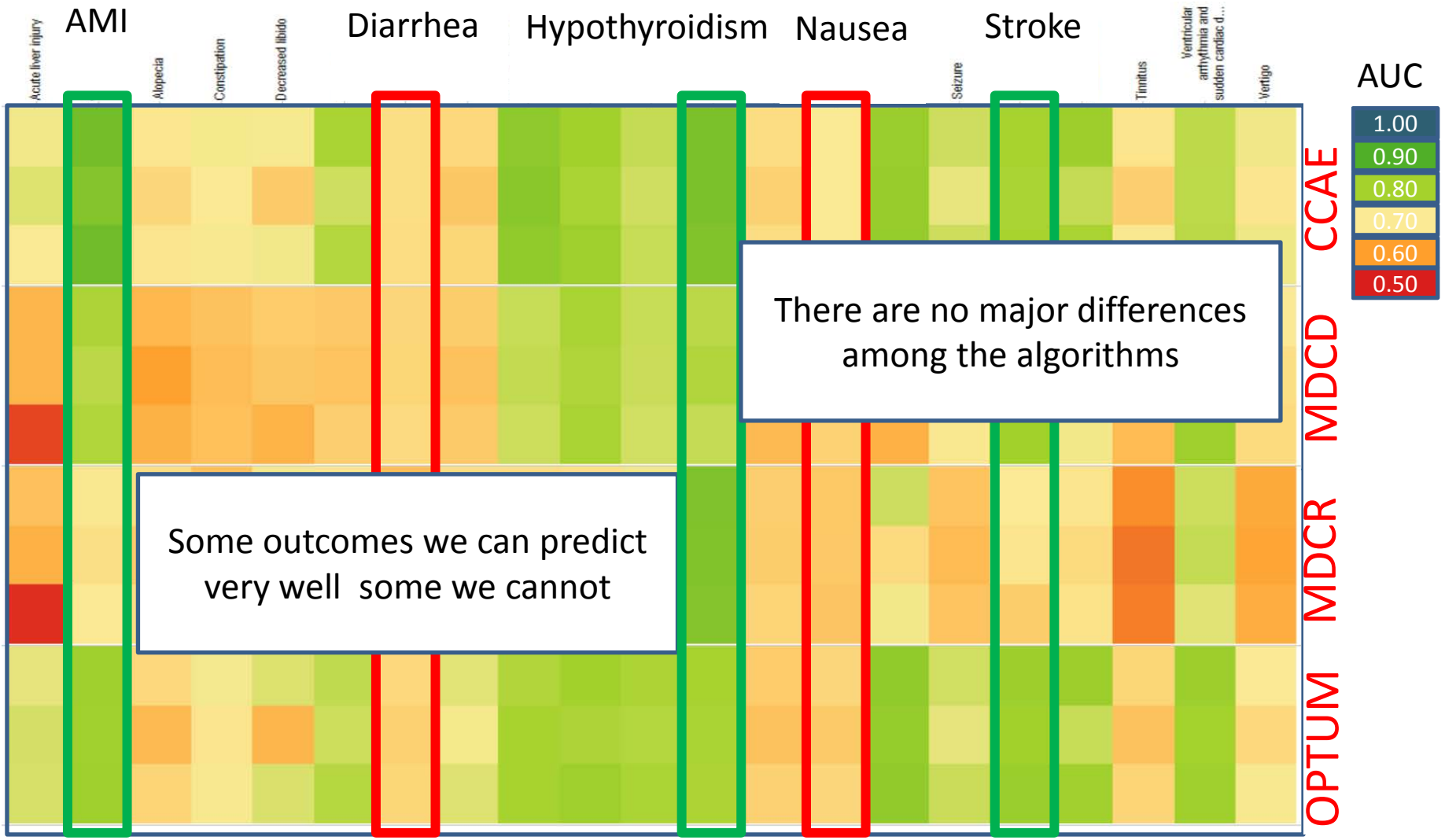
Among patients **in 4 different databases**, we aim to develop prediction models to predict which patients at a defined moment in time (**First Pharmaceutically Treated Depression Event**) will experience one out of **22 different outcomes** during a time-at-risk (**1 year**). Prediction is done using **all demographics, conditions, and drug use** data prior to that moment in time.

Full pipeline in R on top of the OMOP-CDM

Model Discrimination

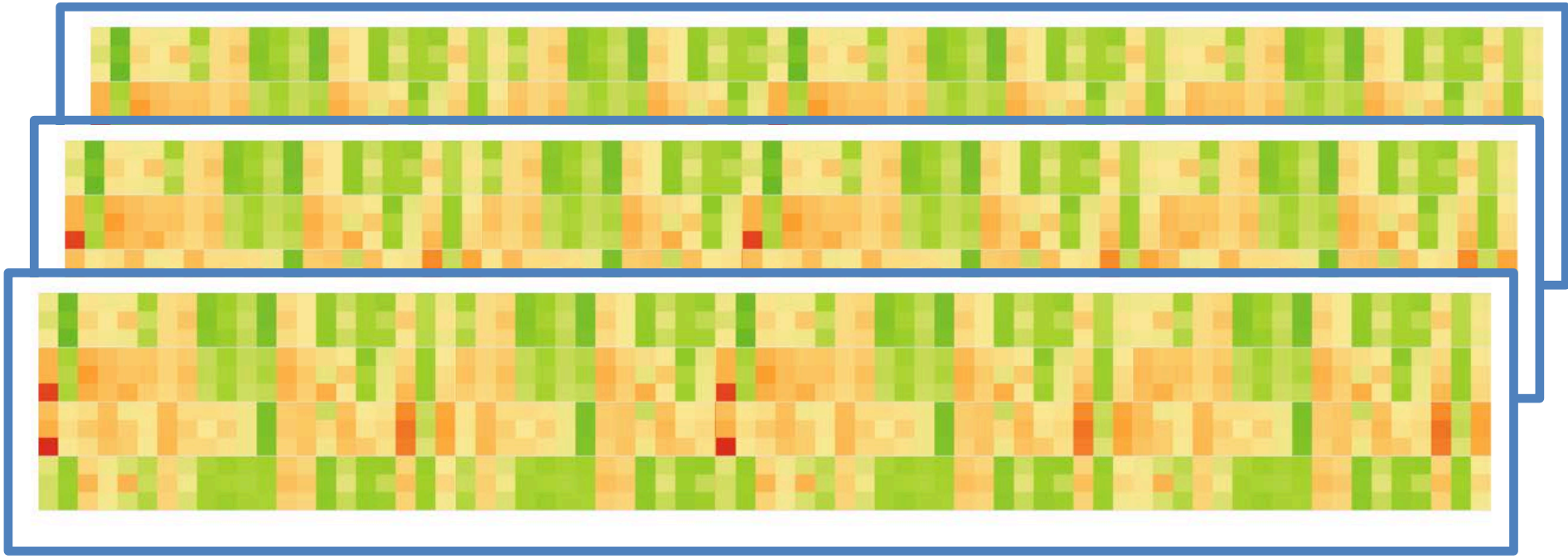


Model Discrimination



What do we want to do in 2017?

Scale up: more cohorts of interest, more outcomes, (on more databases)
Extend: feature engineering, addition of models etc.



Do we need to spend much effort in less promising prediction problems?

Can we transfer knowledge between cohorts of interest and between outcomes?

Agenda

- Introduction Hack-a-thon
- PatientLevelPrediction R-package

Track 1: Unit testing and continuous integration

Track 2: Code base optimization

Track 3: Learning Curves, search space reduction

PatientLevelPrediction R-package

Jenna Reps, PhD

Janssen Research and Development

Slides and Code Explanation Jenna

Track 1: Unit testing and continuous integration

Marc Suchard, PhD
UCLA

Slides Marc

Track 2: Code base optimization

Jenna Reps, PhD

Janssen Research and Development

Slides Jenna

Track 3: Learning curves and search space reduction

Peter Rijnbeek, PhD

Erasmus MC, Rotterdam

The Netherlands

Data extraction

What type of data do we actually need?

- Do we need all conditions, measurements, prescriptions etc or can we take a sequential approach (start with conditions?)
- Can we grow the lookback period?

Experiment: How different would our conclusions be in the POC if we would have only run this on conditions? How much speed would we have gained in the full pipeline?

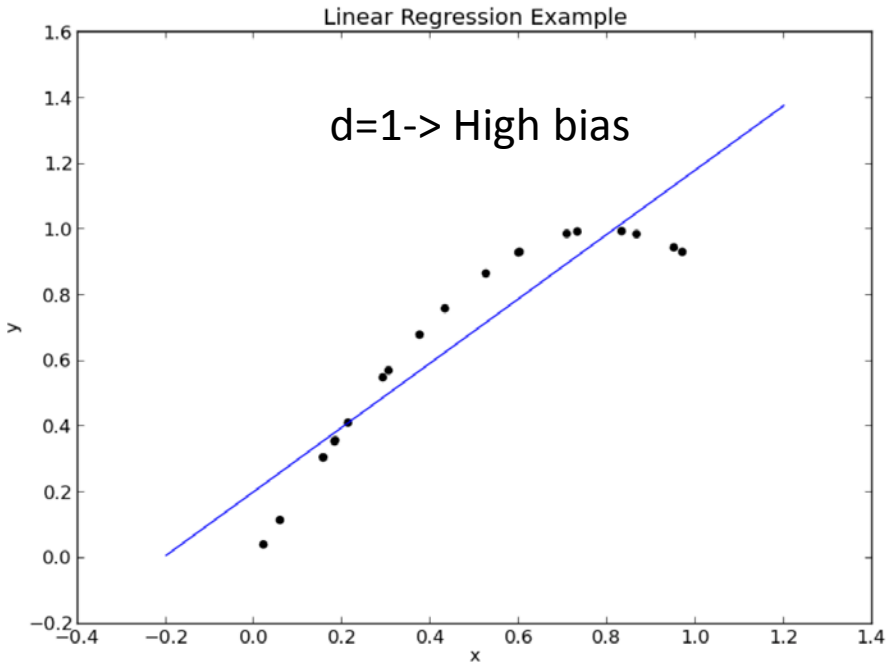
Data partitioning

How much data do we actually need for training and evaluating the models?

- Can we do incremental learning, i.e. start with a smaller set and scale up if this shows increased performance?
- Experiment: take different percentages of the data and compare the performance (learning curve) -> we need code for this to happen!

Background Learning Curves

Question: What is the effect of the training set size on the performance of the models?



Best-fit linear regression to sinusoidal data.

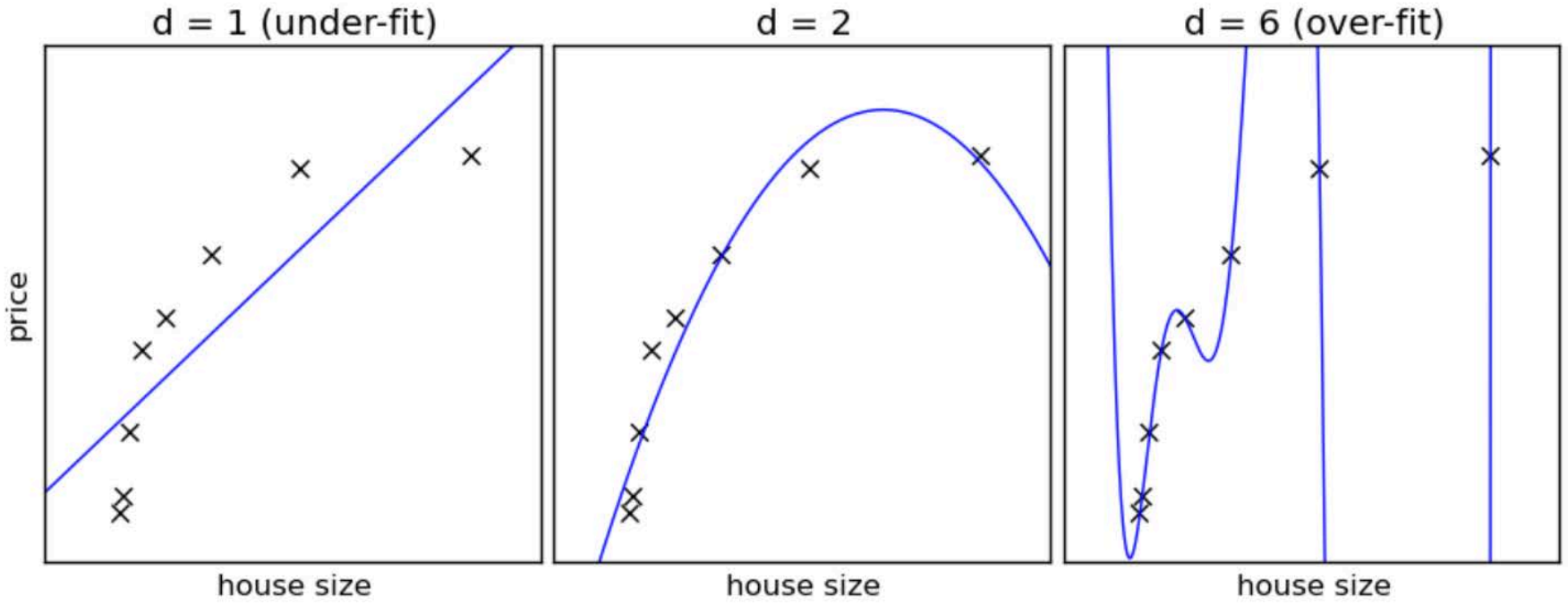
To improve the fit we can:

1. ~~Increase the number of training points *N*. This might give us a training set with more coverage, and lead to greater accuracy~~

2. Increase the degree *d* of the polynomial. This might allow us to more closely fit the training data, and lead to a better result

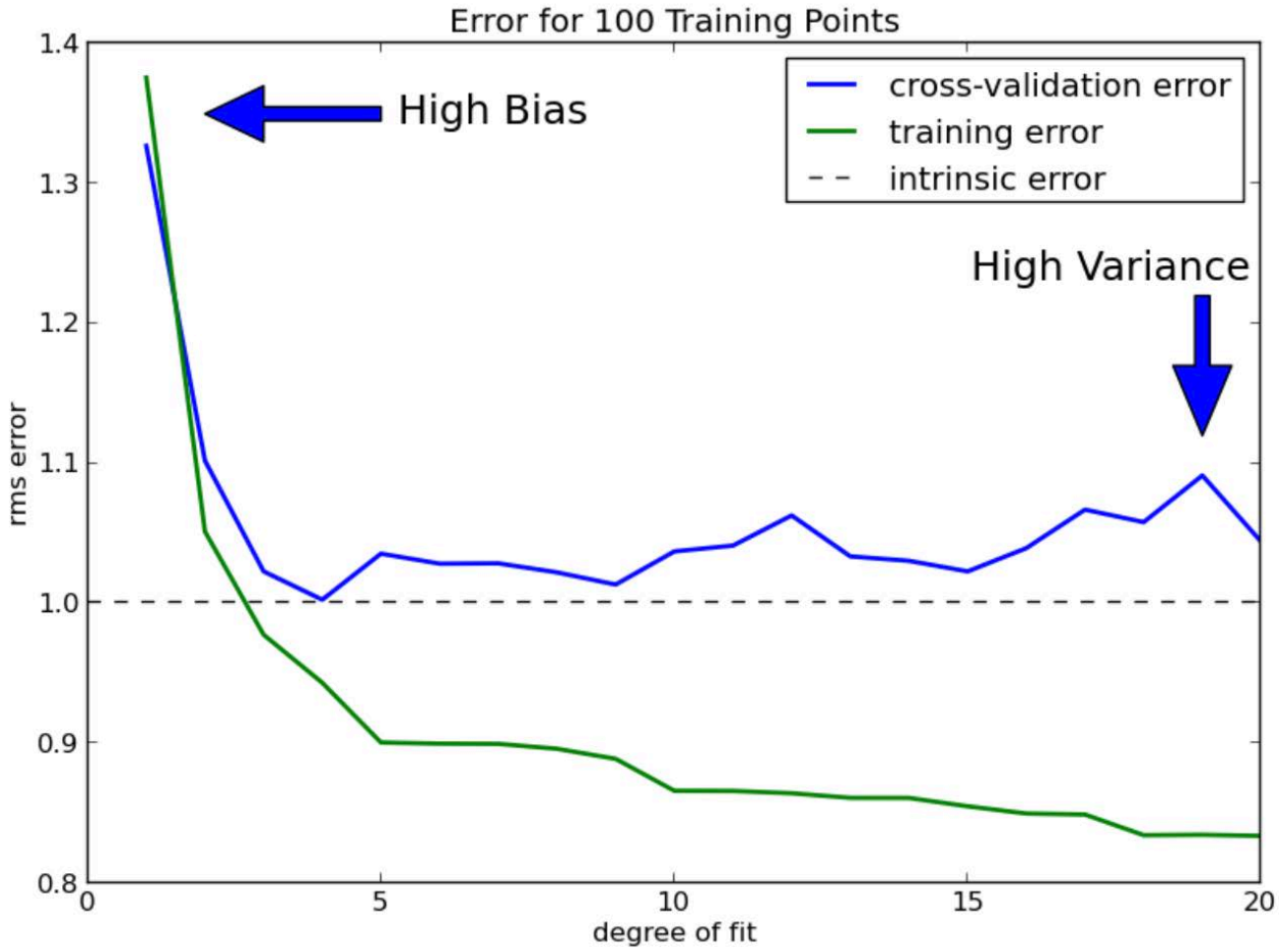
3. Add more features/ complexity, e.g. 1/x

Background Learning Curves



Polynomials of various degrees. $d = 1$ under-fits the data, while $d = 6$ over-fits the data.

Background Learning Curves



The training error and cross-validation error as a function of the polynomial degree *d*.

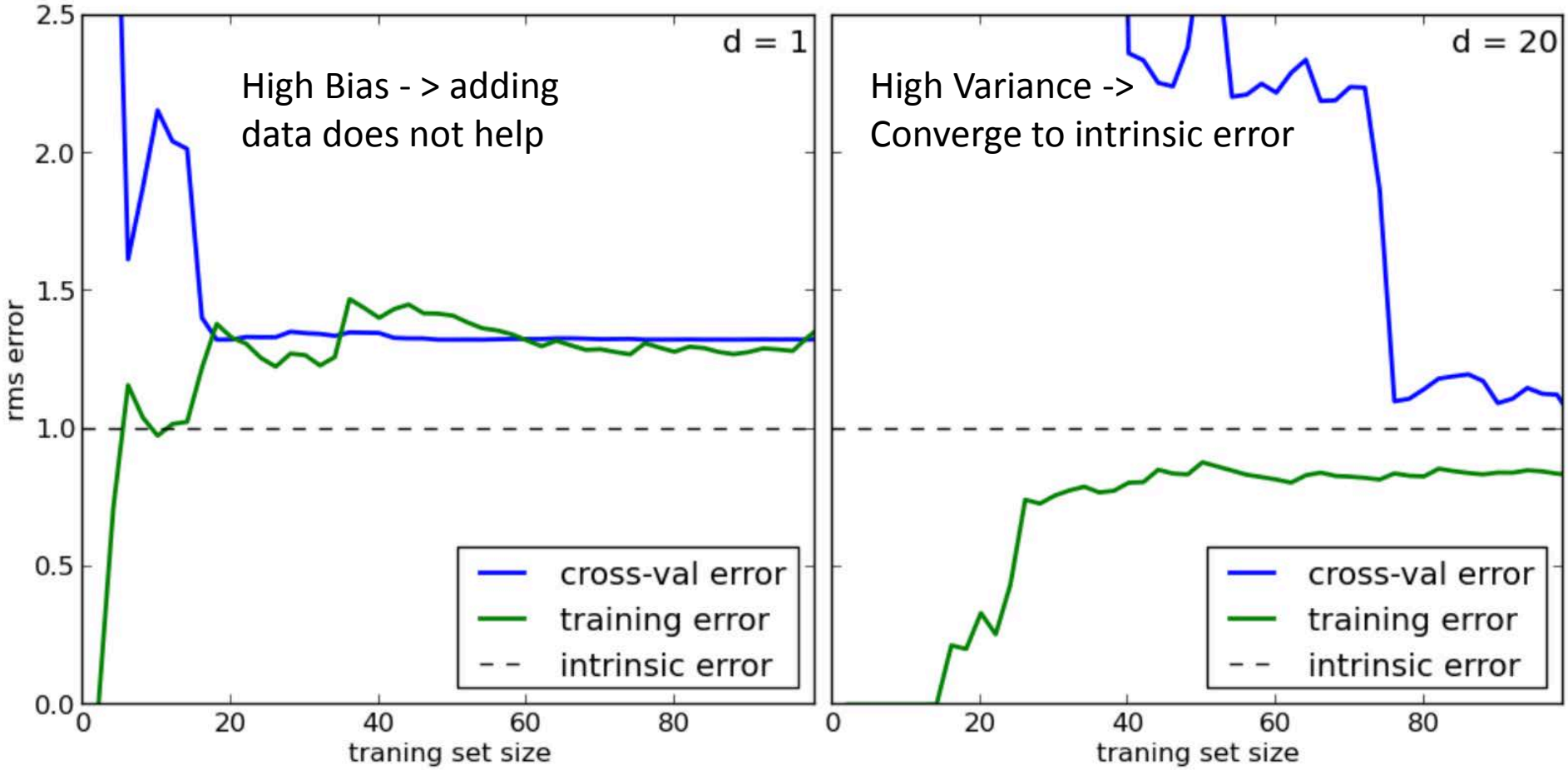
Now d=6 is performing much better than d =2?

Rule of thumb:

The more data points the more complicated model can be used.

Question: But how much is really needed?

Learning Curves



Learning Curves

1. Give insight in the bias and variance of the model
2. Is helpful to determine if getting more data is useful (costly data).

Fitting inverse power laws to empirical learning curves to forecast the performance at larger training sizes.

Progressive sampling: start with a very small batch of instances and progressively increase the training data size until a termination criteria is met.

Figueroa RL. Predicting sample size required for classification performance. BMC Medical Informatics and Decision Making 2012

Learning Curves in Big Data for predictive modelling

We could have the problem we have too much data which increases the computation time too much. Do we need more data? Do we need to make the models more complex to reduce the bias?

A possible focus of a paper could be to define a strategy for this, e.g. by showing that if we have more than >1m(?) cases more data will not help?

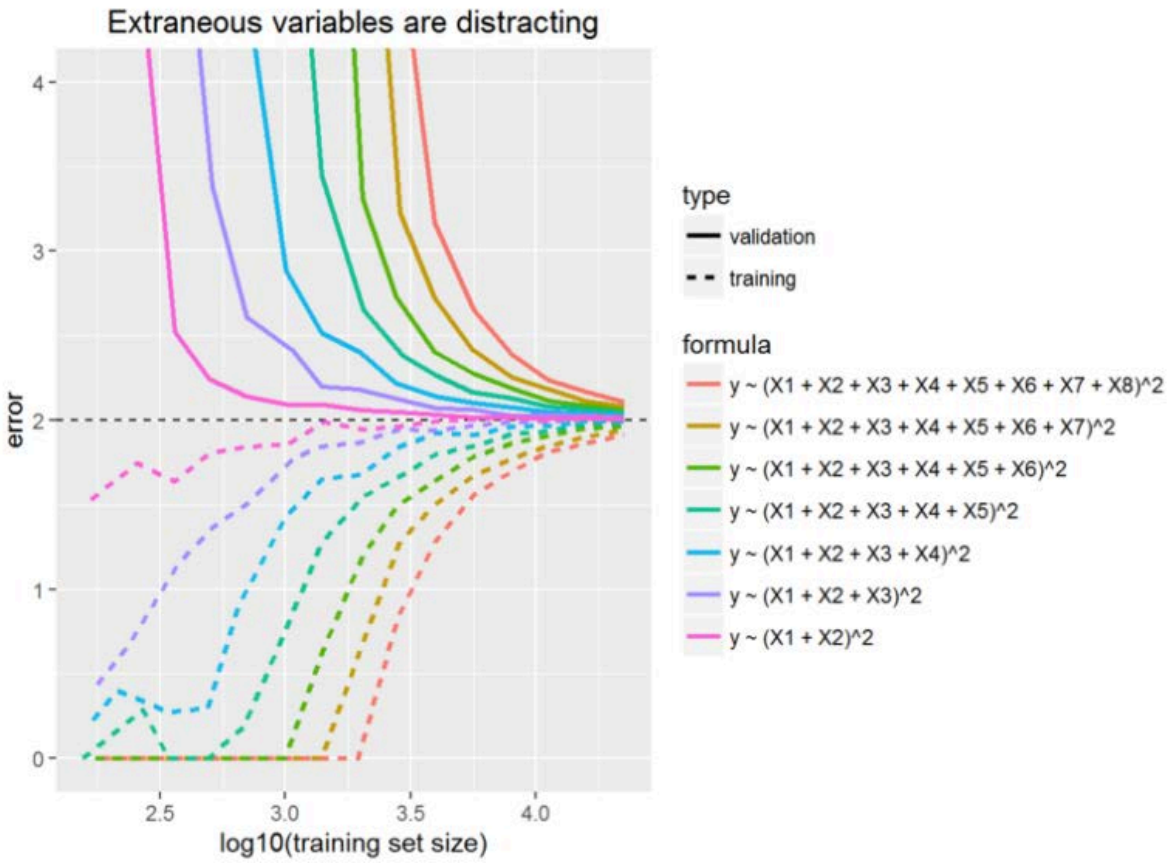
We want to create learning curves for a set of benchmark problems.

We want to do this for different type of models/algorithms using our current PLP Package

Example in R

Simulation experiment with interaction between X1 + X2.

Code is available from www.github.com/mi-erasmusmc/Hack-A-Thon



See Bob Horton : http://blog.revolutionanalytics.com/2016/03/learning-from-learning-curves.html

Model learning

Which type of algorithms will be included and can these be further improved?

- We could start by taking the fasted approach (probably lasso) and only do the others if the performance is above a certain level. We could automate this.
- Can we transfer knowledge between prediction problems? How?

Code optimization

- Can we increase the speed of the code?
- Code profiling etc.

The Hack-a-thon team

Two Slides with the expertise of the group etc from the Google form.

Dinner option…?