Advanced topics in AML

2.4.1 Leakage detection

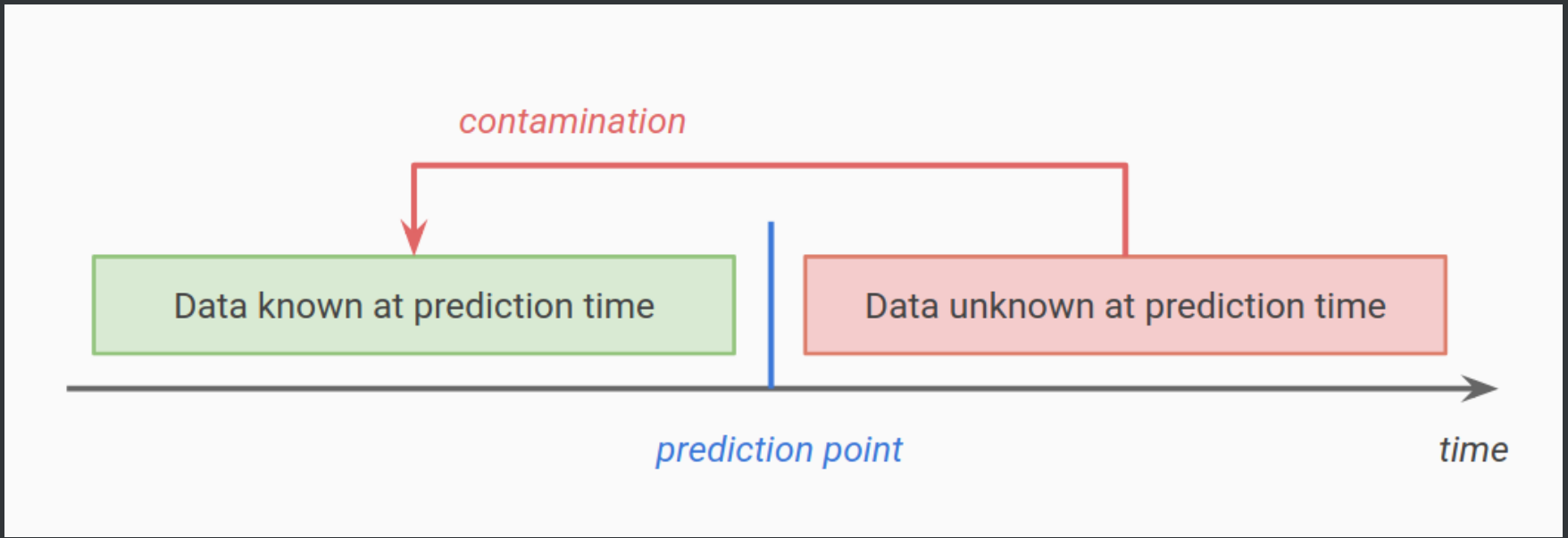
2.4.2 Automated feature engineering

2.4.3 Transfer learning

2.4.4 Model selection at scale

2.4.5 Open problems in AML

Leakage detection



Training on contaminated data leads to overly optimistic expectations about model performance in production

ODSC 2018 talk by Yuriy Guts on target leakage

How can AML help prevent target leakage?

Data Collection

Assist Data Scientist to flag spurious columns (e.g. very hight univariate correlation)

Feature engineering

Pipeline DSLs can provide safety, even when doing manual ML.

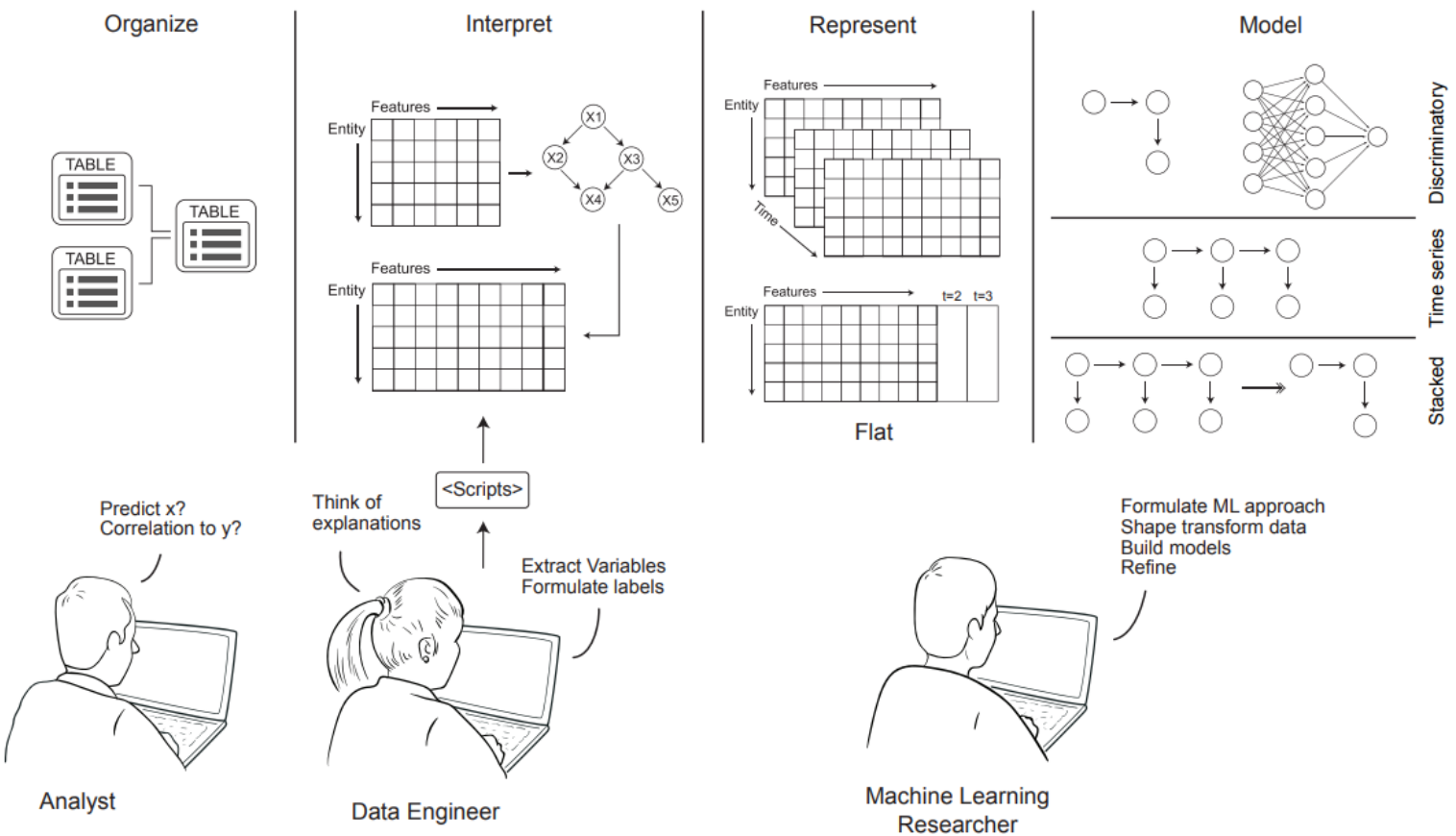
Partitioning

Assist Data Scientist by providing good error diagnostics and model introspection

Training & Tuning

Human-out-of-the-loop: implement best practices, no short cuts, no peeking at the holdout.

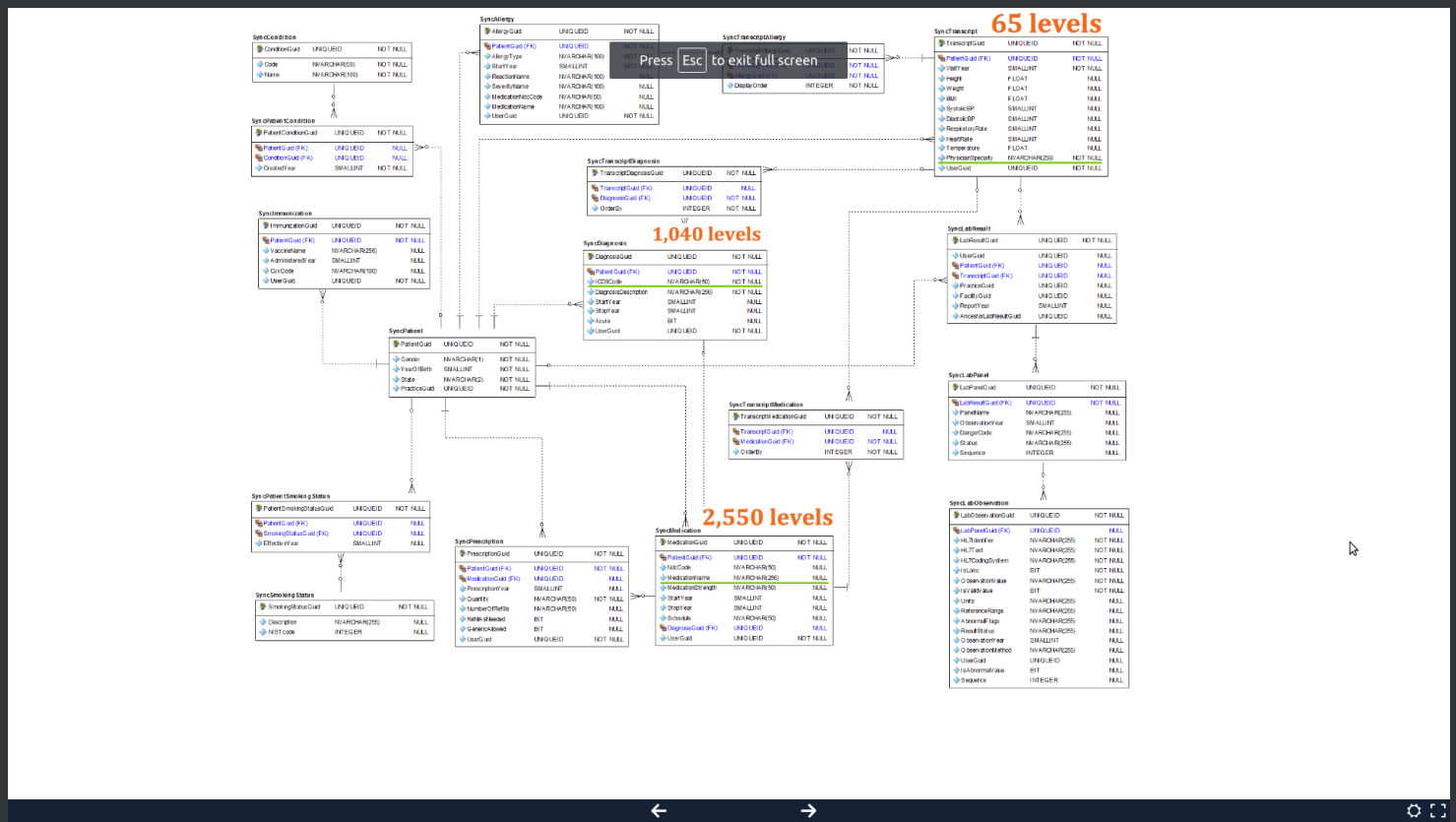
Automated Feature Engineering



Source: J. Kanter, K. Veeramachaneni (2015), Deep feature synthesis: Towards automating data science endeavors. DSAA 2015

Real-world use-case

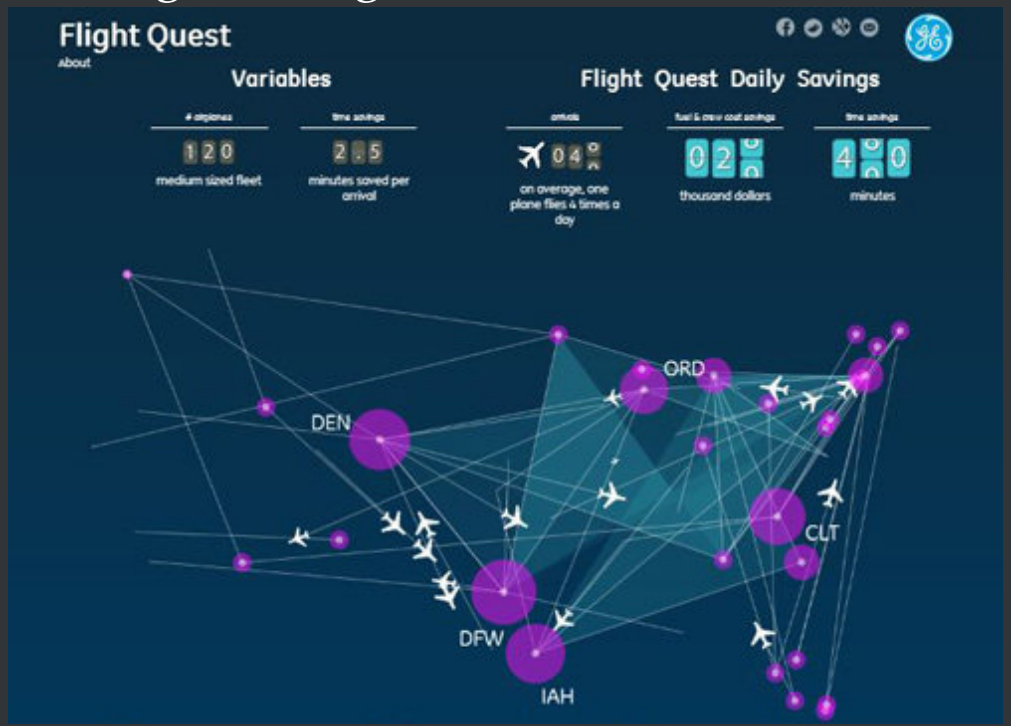
Many real-world use cases operate on relational & transactional data



Most ML feature engineering we've seen so far operate on de-normalized data (i.e. single table)

Time-aware feature engineering...

You also may need to take time into account when doing model selection & feature engineering



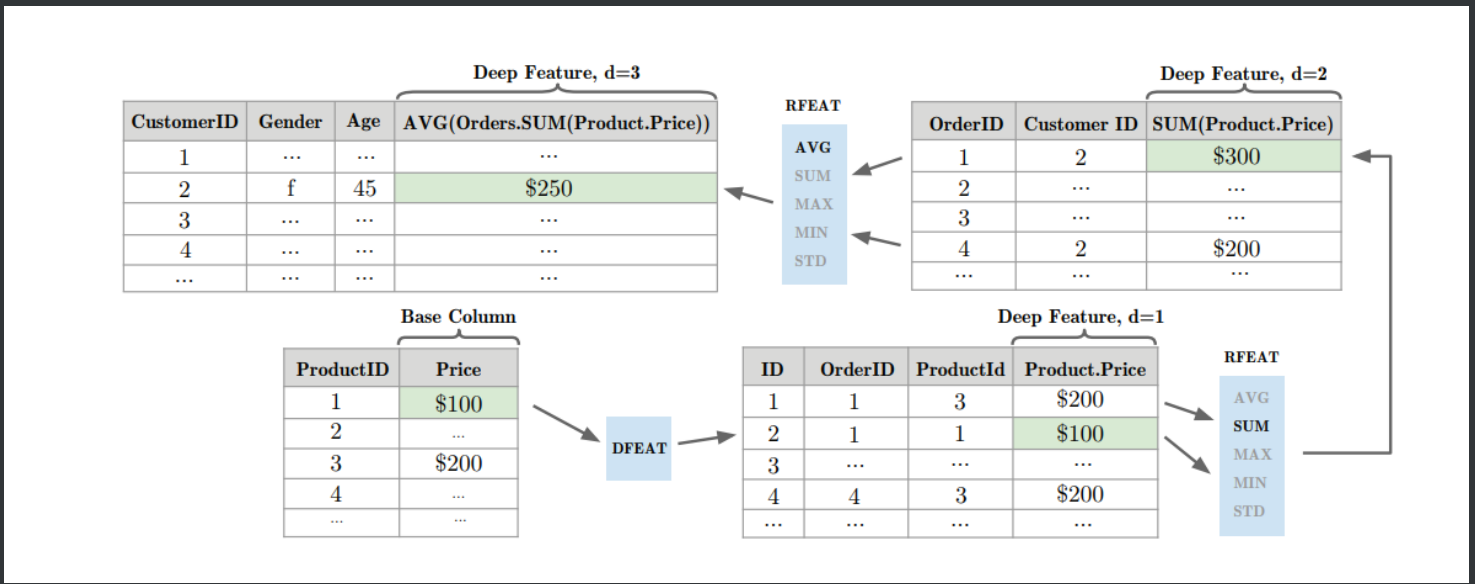
Example: predict the gate / runway arrival time for flight in mid-air (t0 aka cutoff); you must not use records from table weather station which correspond to sensor reading at time t1 > t0

...matters


FLIGHT QUEST
CHALLENGE WINNER:
GXAV &*
TEAM WINNERS
Kenny Chua
Xavier Conort
Cao Hong
Chua Hon Nian
Clifton Phua
Ghim-Eng Yap
WHAT DID THEY DO?
Built a mathematical model that helps improve runway and gate arrival time estimates as much as
40 PERCENT
over industry standards.
WHY?
A 1-minute reduction per departure could save an average-sized airline
1,700 HOURS PER YEAR.
This is equivalent to reducing crew costs by
$1.2 MILLION
and fuel costs by
$5 MILLION

Deep feature Synthesis

DFS is an algorithm for automatically generating features for relational datasets. DFS operates on a snowflake schema and follows relationships in the data to a base field, and then sequentially applies mathematical functions along that path to create the final feature.



Source: J. Kanter, K. Veeramachaneni (2015), Deep feature synthesis: Towards automating data science endeavors. DSAA 2015

Featuretools

Open source implementation of DFS on top of pandas:

```
import featuretools as ft

data = ft.demo.load_mock_customer()
entities = {
   "customers" : (data['customers'], "customer_id"),
   "sessions" : (data['sessions'], "session_id", "session_start"),
   "transactions" : (data['transactions'], "transaction_id", "transaction_time")
   }

relationships = [("sessions", "session_id", "transactions", "session_id"),
                 ("customers", "customer_id", "sessions", "customer_id")]

feature_matrix_customers, features_defs = ft.dfs(entities=entities,
     relationships=relationships,
     target_entity="customers")
```

AFE summary

Generic Extract-Transform-Load (ETL) tools are not great for feature engineering

- Join operations need to take cutoffs into account.
- Feature extractors need to be part of your ML pipeline to guard against leakage and train-test skew.

Data needs to be stored in a way AFE can be applied

- AFE operates on a Transaction Log rather than a static snapshot (e.g. customer table).

Automated feature preprocessing and missing values imputation

Different preprocessing/imputation techniques should be used for different types (families) of models.

Incorrect preprocessing may ruin model performance no matter how good that model is and how long you tune it.

Types of features to consider:

- Numerical features
- Categorical features
- Text features

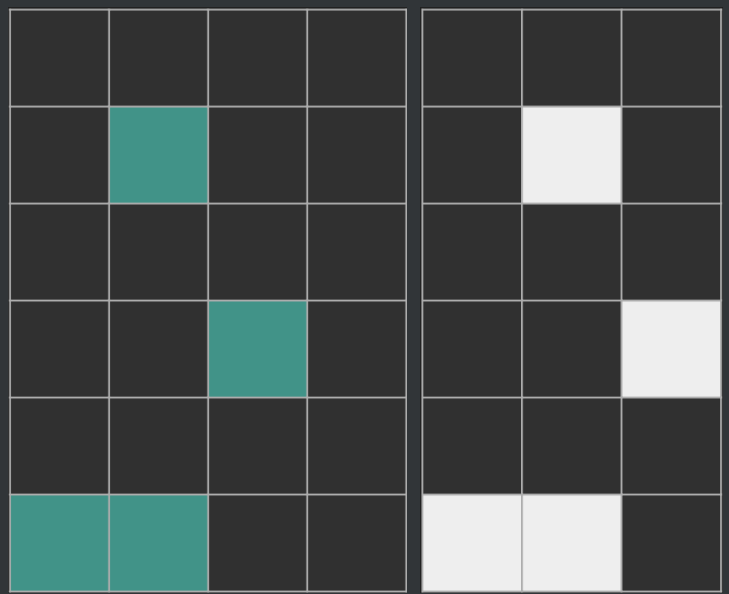
Numerical features

Imputation:

- Impute with mean value (linear models / neural nets)
- Impute with very large number like 9999 (tree based)

Preprocessing:

- Scaling with standard, minmax, etc. (linear models / neural nets)
- Do nothing (tree based)



Categorical features

Imputation:

- Depends on preprocessing, but generally - allocate "missing" category like 0 or -1

Preprocessing:

- Ordinal encoding (tree based)
- One hot encoding (linear/NNs)
- Frequency encoding
- Categorical embedding (easy to learn with NNs)

missing
UA
FR
US

0
1
2
3

0	0	0
1	0	0
0	1	0
0	0	1

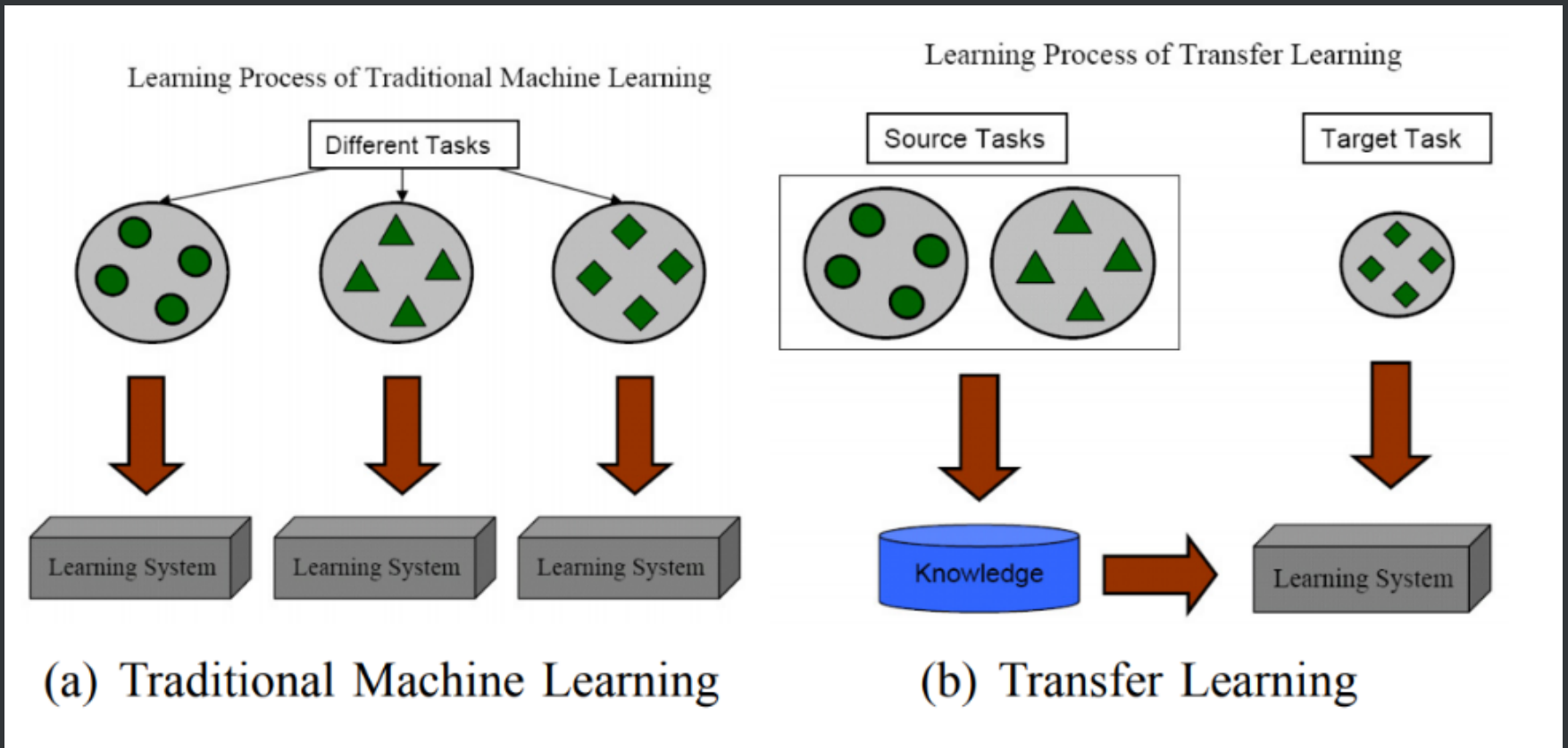
0.04
0.34
0.28
0.44

-0.43	0.12
0.65	0.34
-0.5	-0.91
0.59	0.1

Text

- TF-IDS
- Encoding with word2vec
- Encoding with Universal Sentence Encoder
- Using pre-trained language models
- Stacking with specialized text model

Transfer Learning



Source: S. J. Pan, Q. Yang (2010), "A Survey on Transfer Learning" IEEE

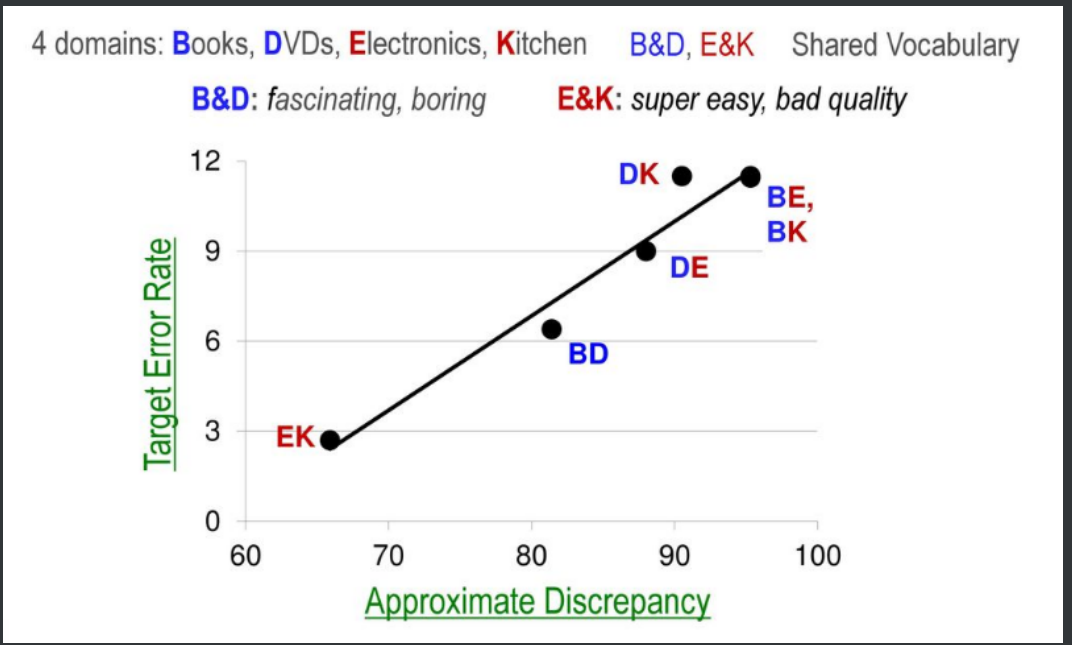
Automatically Select the best Source Task

Goal: Build a sentiment classifier for book reviews but little to no data for the target task is available, however, we have plenty of reviews for dvds, electronics and kitchen appliances.

Which source domain (dvd, electronics, kitchen appliances) should we use to train our sentiment classifier?

Discriminative Distance (aka Discrepancy)

Use a classifier to distinguish source and target domains. The source domain that is the hardest to separate from the target has the smallest discriminative distance and thus most resembles the target.

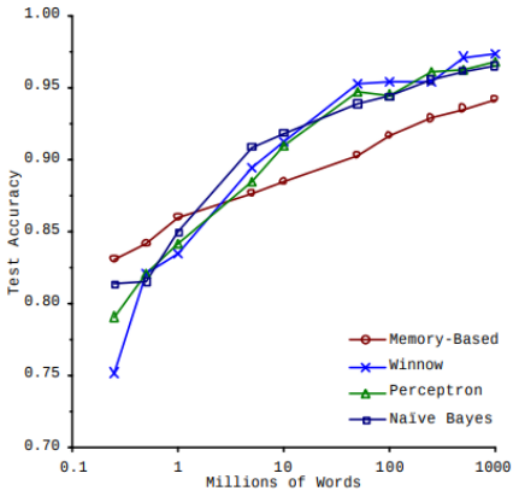


Source: J. Blitzer, H. Daume III, Domain Adaptation

Model selection at scale

The Unreasonable Effectiveness of Data

Alon Halevy, Peter Norvig, and Fernando Pereira, *Google*



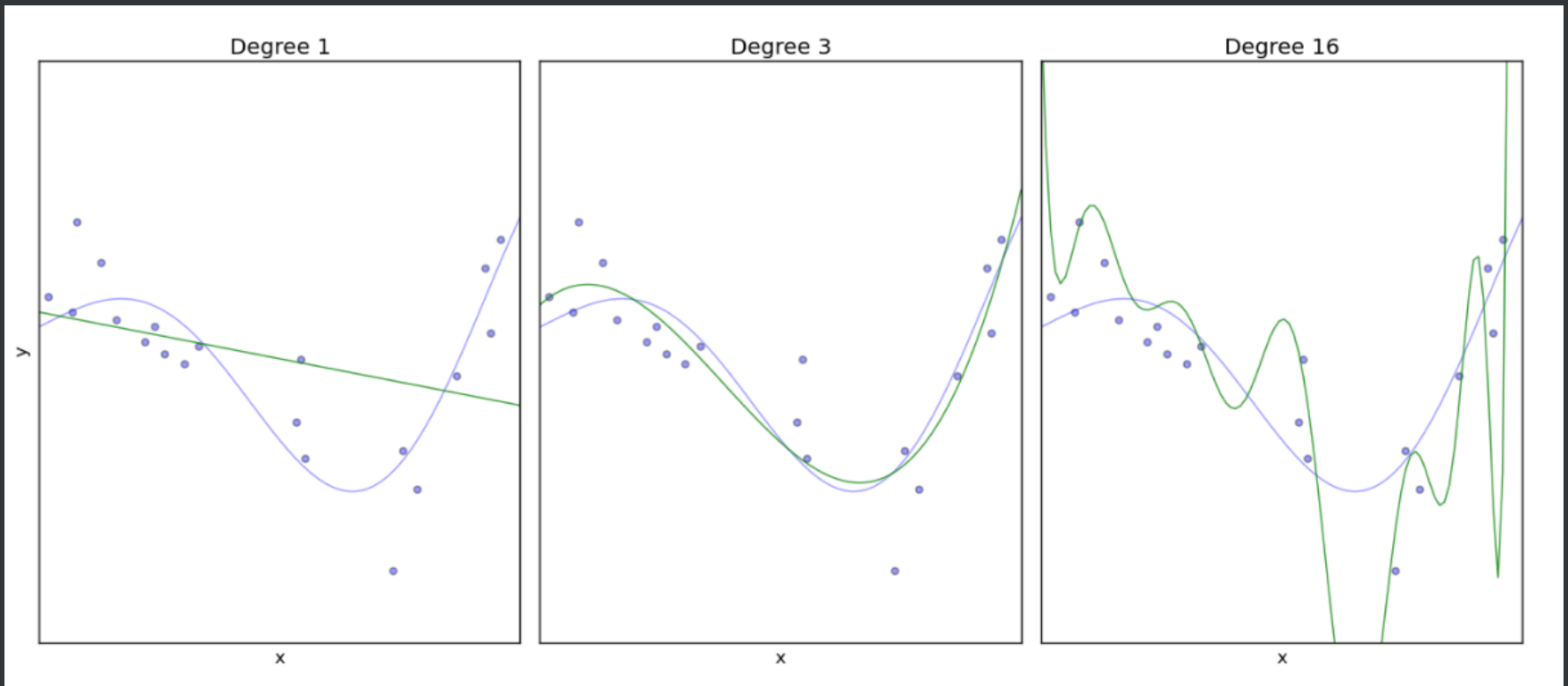
Source: M. Banko, E. Brill (2001), Scaling to Very Very Large Corpora for Natural Language Disambiguation, ACL

More data trumps better algorithms?

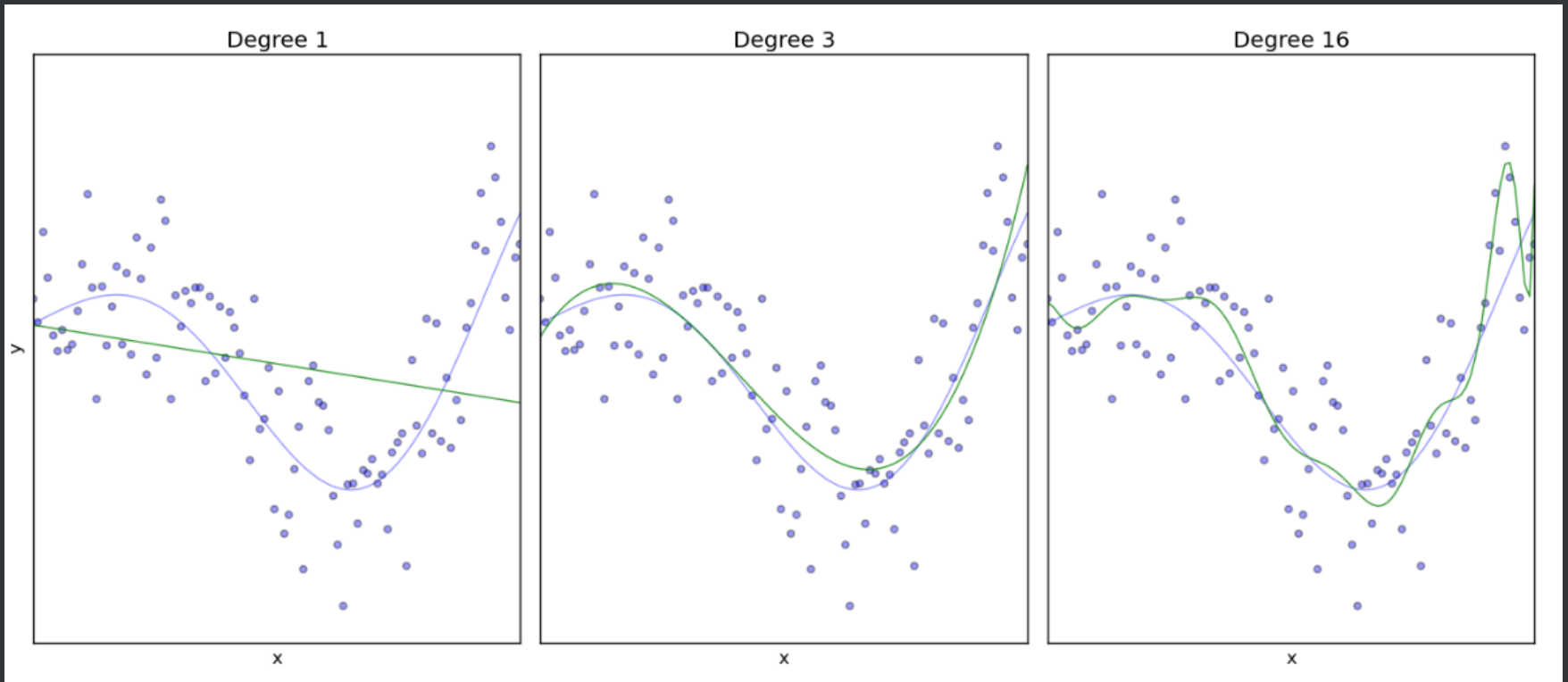
Both Norvig et al. and Banko & Brill work have often been misinterpreted to mean: more data trumps better algorihtms.

We cannot make such a general statement, the answer lies in whether you have a bias or a variance problem.

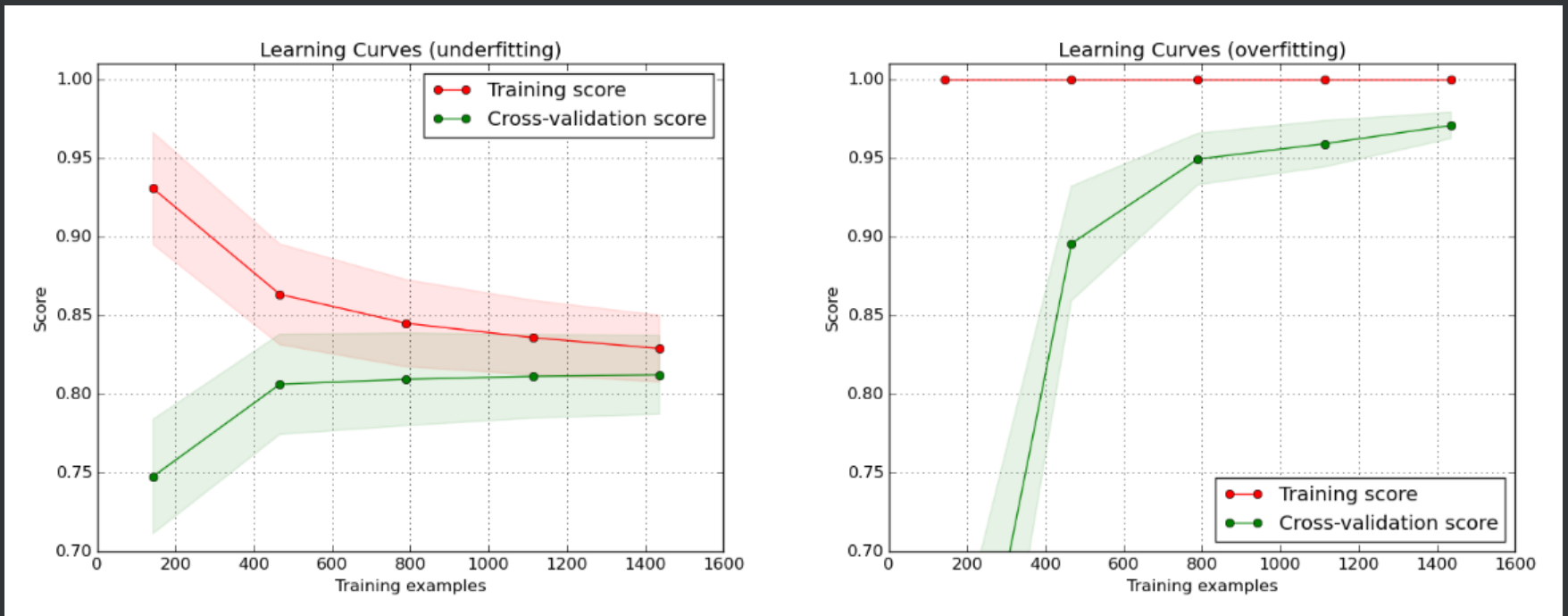
Model Complexity and Overfitting



More data to the rescue?



Underfitting or Overfitting?



Challenges at Scale

- Why learning with more data is harder?
  - Paradox: we could use more complex models due to more data but we cannot because of computational constraints [1].
    - => we need more efficient ways for creating complex models!
- Need to account for the combined cost: model fitting + model selection / tuning
  - Smart hyperparameter tuning tries to decrease the # of model fits
  - we can accomplish this with fewer hyperparameters too[2]

[1] P. Domingos, A few useful things to know about machine learning, 2012

[2] Practitioners often favor algorithms with few hyperparameters such as RandomForest or AveragedPerceptron

Case-study: binary classification on 1TB of data

- Criteo click through data
- Down sampled ads impression data on 24 days
- Fully anonymized dataset:
  - 1 target
  - 13 integer features
  - 26 hashed categorical features
- Experiment setup:
  - Using day 0 - day 22 data for training, day 23 data for testing

Big Data?

Data size:

- ~46GB/day
- ~180,000,000/day

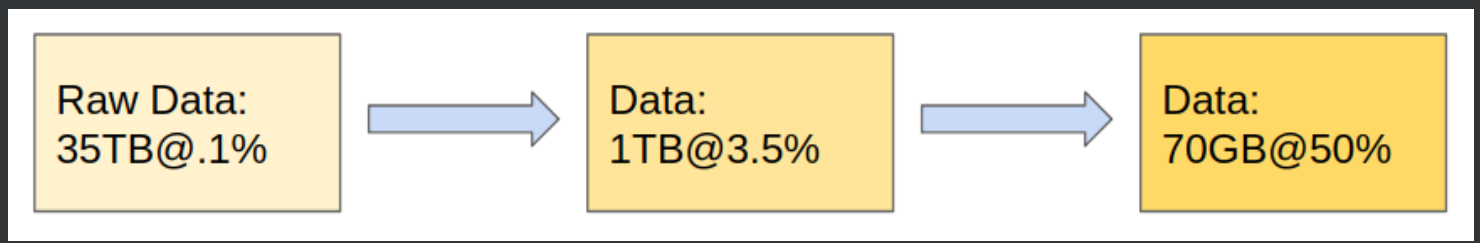
However it is very imbalanced (even after downsampling non-events)

- ~3.5% events rate

Further downsampling of non-events to a balanced dataset will reduce the size of data to ~70GB

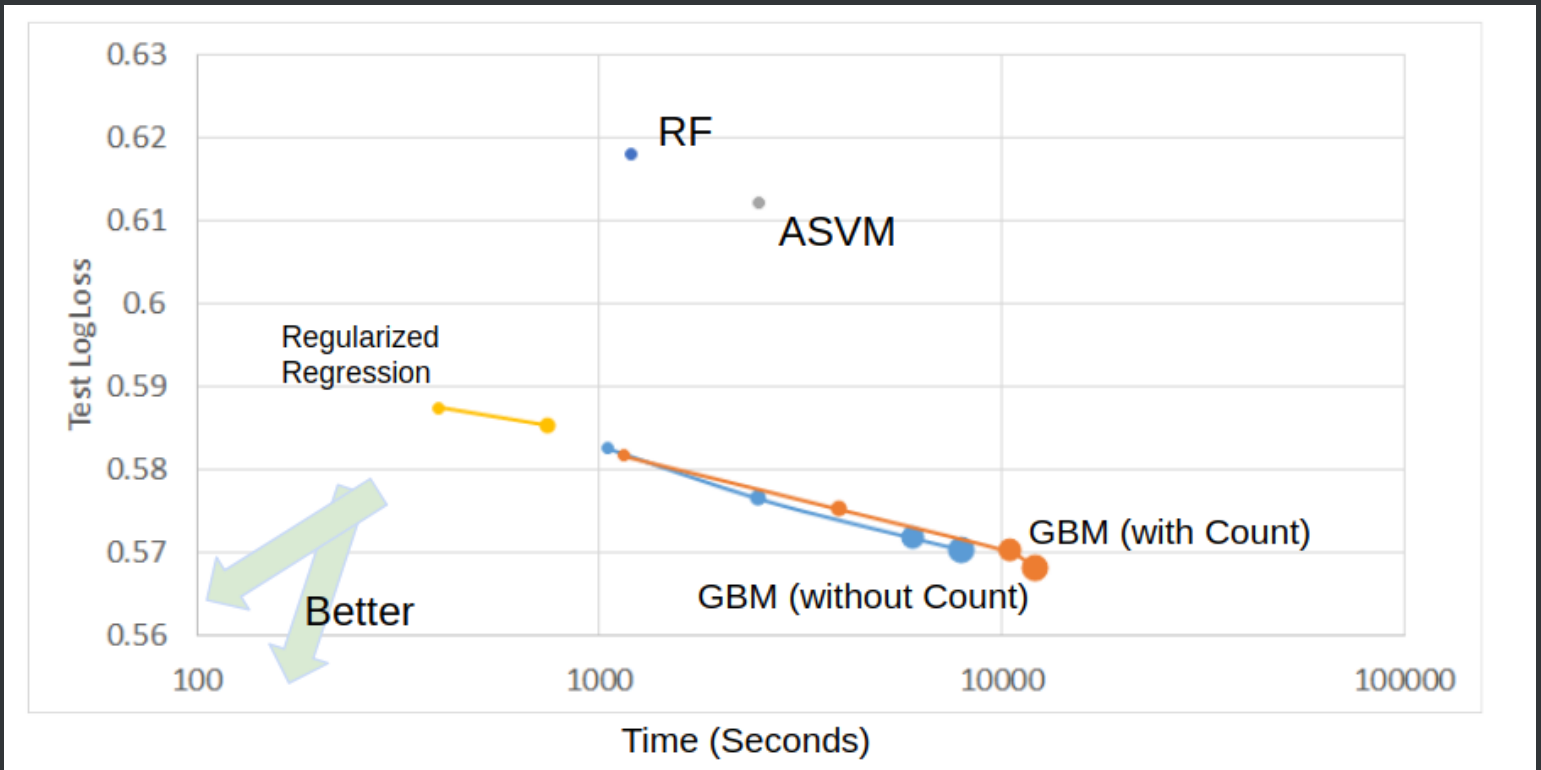
- Will fit into a memory on a beefy machine
- Loss of model accuracy is negligible in most situations

Assuming 0.1% raw event (click through) rate:

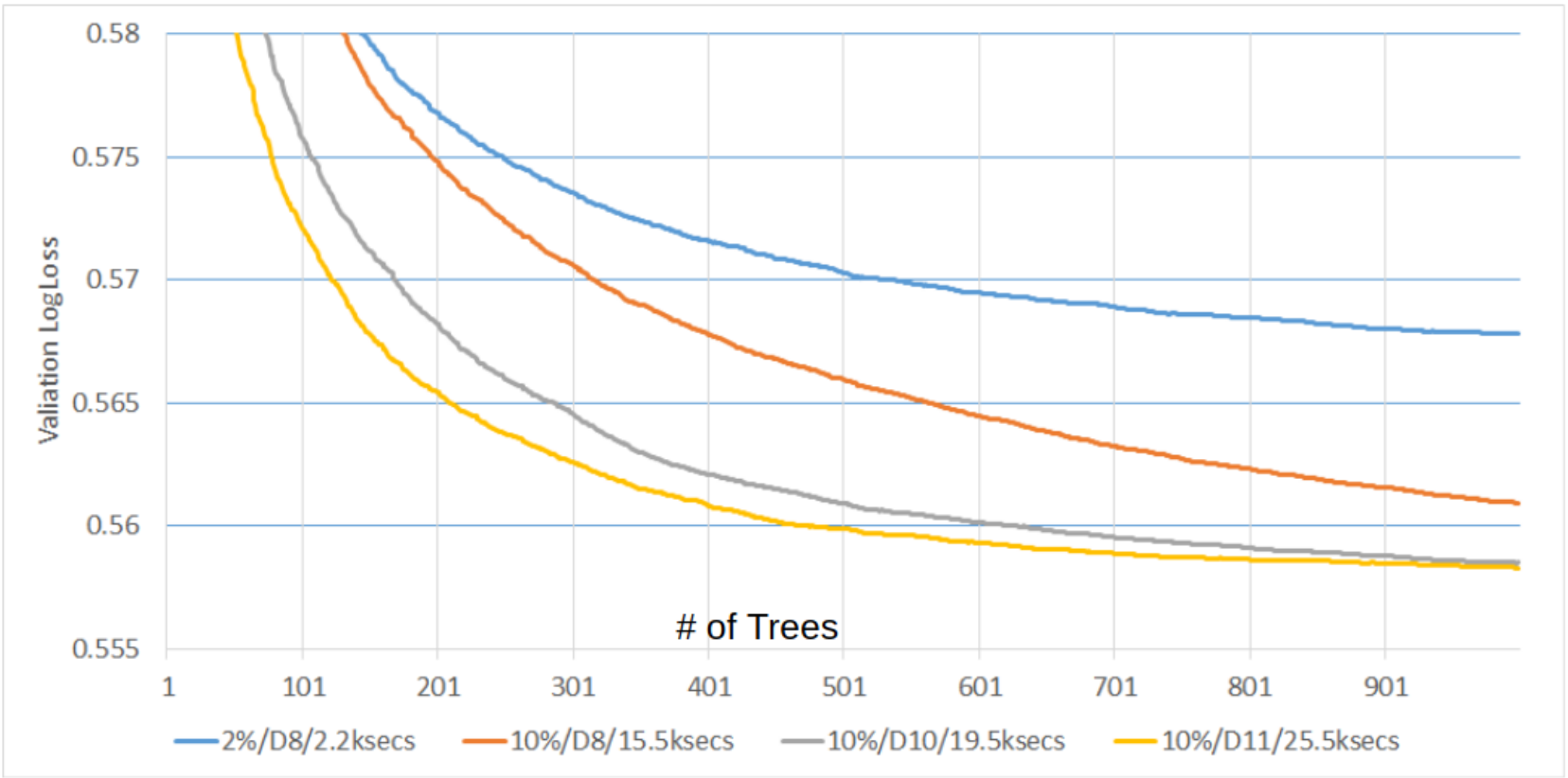


Where to start?

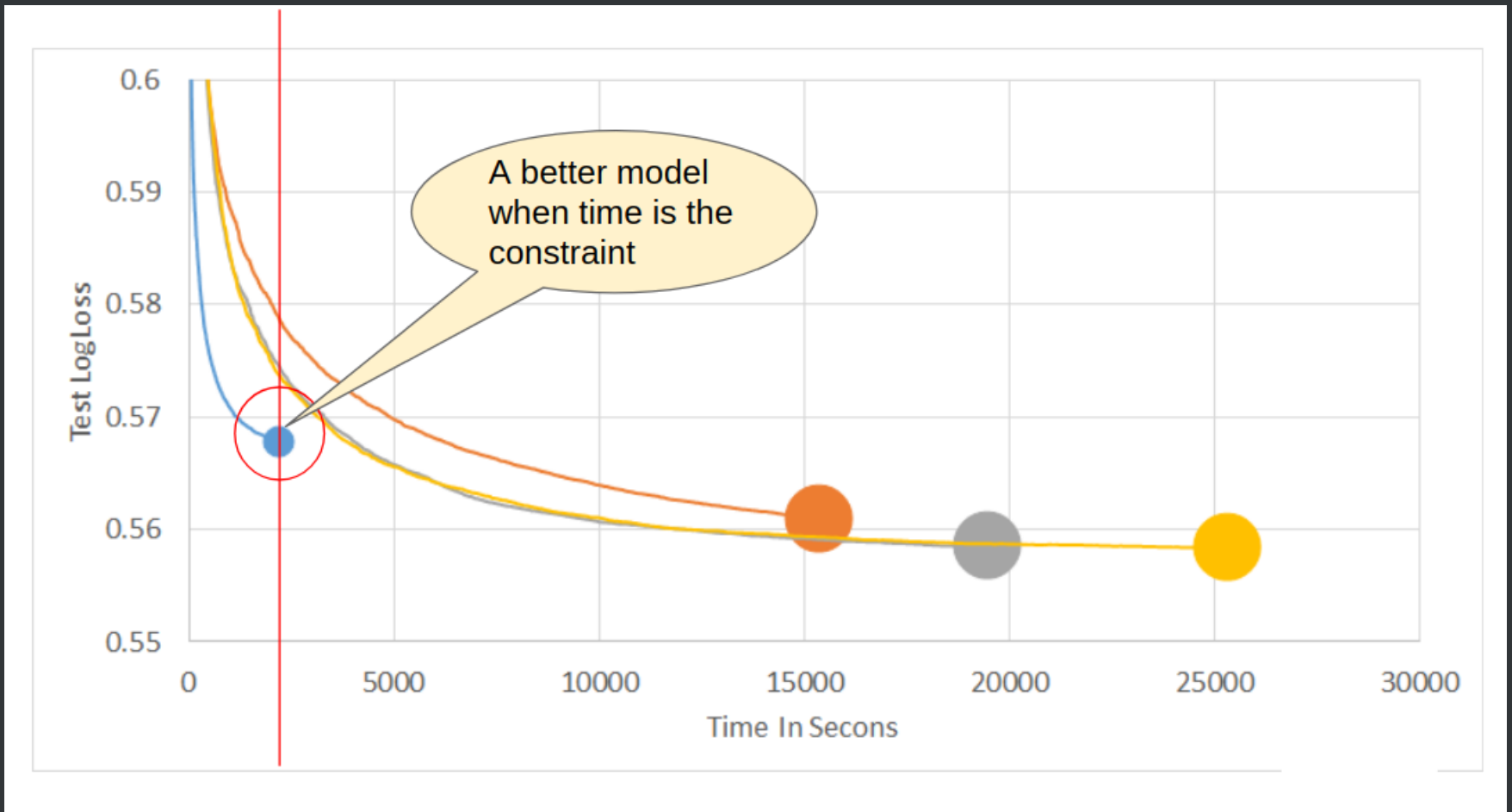
- 70GB (~260,000,000 data points) is still a lot of data
- Let's take a tiny slice of that to experiment
  - Take 0.25%, then .5%, then 1%, and do grid search on them



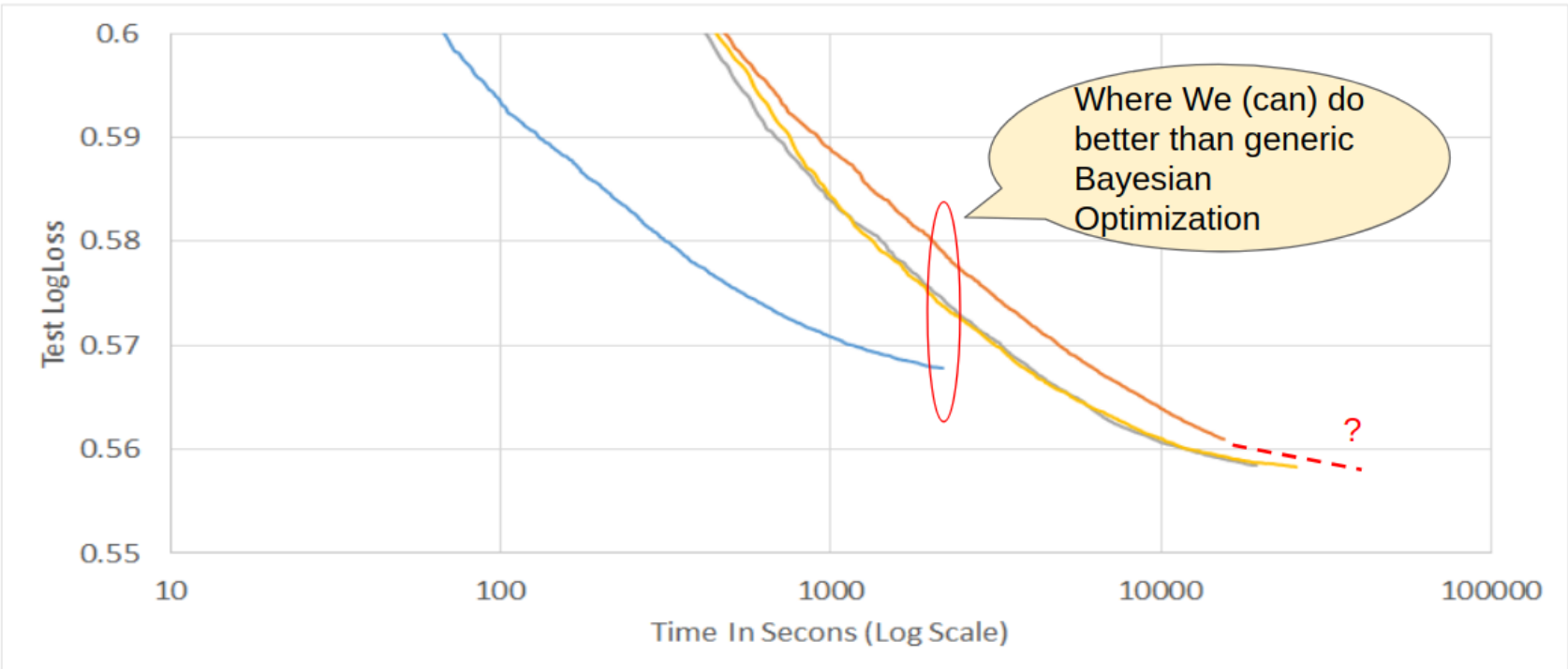
GBM is the way to go, let's go up to 10% data



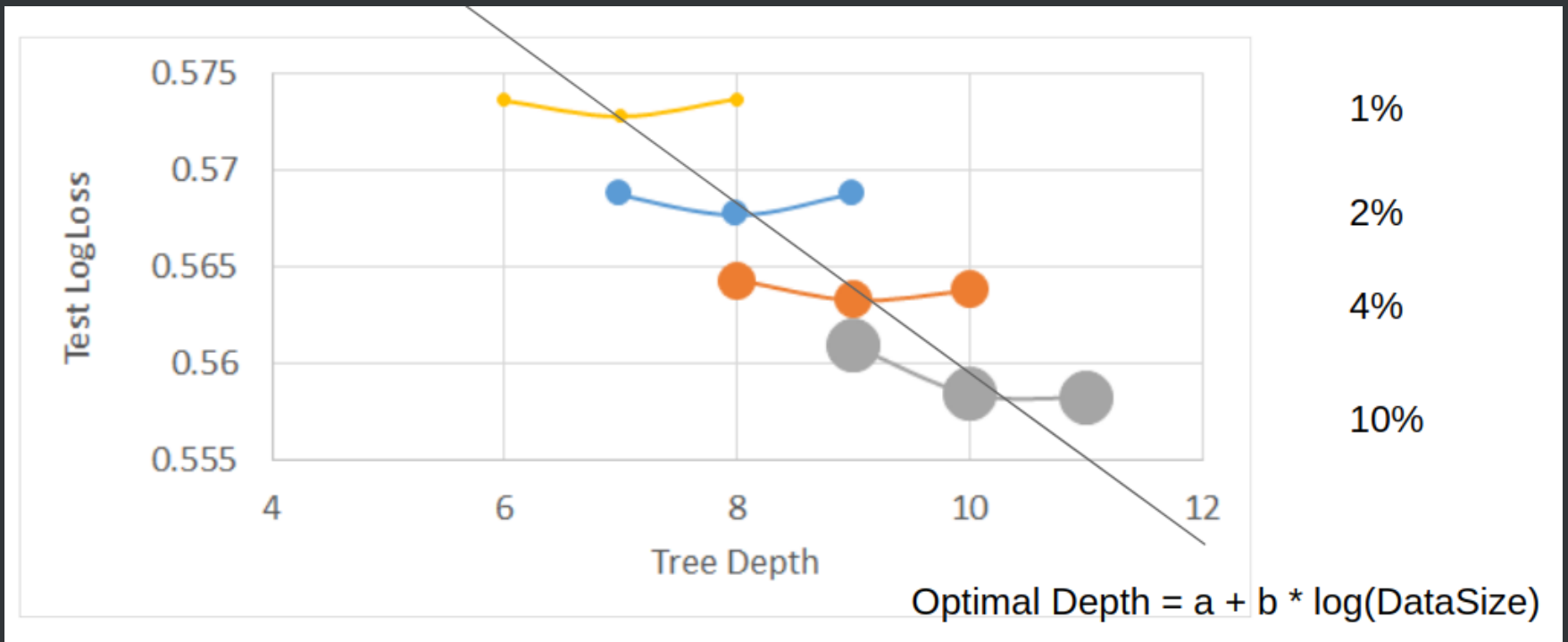
A “Fairer” Way of Comparing Models



Can We Extrapolate?



Tree Depth vs Data Size



Open Research Topics in AML

- __ at scale
  - Pipeline Optimization
  - Automated Feature Engineering
- Automated Leakage Detection
- Automated Partitioning/Cross-Validation
- Automated Metric Selection
- Automated Transfer Learning
- Automated Data Drift Detection
  - the world never stops changing...
- Automated Feedback Cycle Detection
  - ... and we never stop changing the world

More see Rich Caruna's IMCL 2015 talk on AutoML