# Seminar: Modern Data Analytics

https://www.ifi.uzh.ch/en/dast/teaching/SMDA.html

---

**Prof. Michael Böhlen & Dr. Ahmet Kara &**
**Prof. Dan Olteanu & Dr. Nils Vortmeier**


University of Zurich[UZH]

Sept 14, 2020

Department of Informatics

# Seminar Topic

Recent research development on DB $\cap$ ML

1. ML for DB
    - Use ML models to predict DB runtime performance
    - Replace traditional modules in a DB system with ML models
2. DB for ML
    - Use DB techniques to improve the ML runtime performance

# Assessment

Each participant

- Writes a self-contained report (10 pages)
  - First version due **FOUR weeks before the presentation**
  - Final version due **ON the presentation day**
- Gives a 30 minutes presentation
- Answers follow-up technical questions in a 15-minutes Q&A session

Assessment depends on

- Quality of the report
- Presentation
- Active participation during the seminar
- Input as a buddy

# Buddies

Each participant has a **buddy**. They:

- Read the report
- Make suggestions for improvements
- Help with the presentation (e.g., dry runs)

Please let us know at `dast@ifi.uzh.ch` who is your buddy!

The first version of the report and presentation will be discussed with the buddy and the supervisor a few weeks before the presentation.

# Seminar Schedule

**The presentations will be in person, attendance is compulsory**

- Starting 8am on **Saturday, December 5, 2020**
- Starting 8am on **Saturday, December 12, 2020**
- Room: **2.A.01**

# Preparation for Seminar

- How to read papers and give talks
  - Resources on the seminar web page
- Papers to be read by all students
  - MLSys Whitepaper
  - A Few Useful Things to Know About Machine Learning
- **Each participant is assigned to a selected paper**

# MLSys: The New Frontier of Machine Learning Systems

Significant obstacle: Design & Implementation of systems that support ML models in real-world deployments

- Radically different application requirements
  - Requirements: collect, preprocess, label, and reshape training datasets
  - Replace write & deploy code approach
  - Modern ML applications referred to as a new "Software 2.0"
- Increasingly-prevalent systems-level concerns
- Rising tide of interest and adoption

New Systems ∩ ML research community working on:

- Hardware & software systems for ML
  - Low-level systems for executing ML models
  - Interfaces for embedding learned components in traditional systems
- ML optimised for metrics beyond predictive accuracy

# Questions to be Addressed by the New Research Community

1. How should software systems be designed to

   support the full machine learning lifecycle,

   from programming interfaces and data preprocessing to output interpretation, debugging and monitoring?

2. How should hardware systems be designed for machine learning?

3. How should machine learning systems be designed to

   satisfy metrics beyond predictive accuracy,

   such as power and memory efficiency, accessibility, cost, latency, privacy, security, fairness, and interpretability?

# A few useful Things to Know about Machine Learning

Summary of key lessons that ML researchers and practitioners have learned

$$\text{Learning} = \text{Representation} + \text{Evaluation} + \text{Optimisation}$$

- Representation: Hypothesis space of the learner
  - Choose the set of classifiers that it can possibly learn
- Evaluation: Objective or scoring function
  - Distinguish good classifiers from bad ones
- Optimisation: Search method for the highest-scoring classifier.
  - Key to the efficiency of the learner
  - Unlike in most optimisation problems, we do not have access to the function we want to optimise!
  - Use training error as a surrogate for test error :(
  - Objective function is only a proxy for the true goal $\Rightarrow$ no need to fully optimise it, local optimum may be OK

# Lesson 1: Fundamental Goal of ML

ML's Goal: Generalise beyond the examples in the training set

- Test data is highly important
- Data alone is not enough
  - Every learner must embody some knowledge or assumptions beyond the data
  - Wolpert's "no free lunch": no learner can beat random guessing over all possible functions to be learned.
  - Hope: Functions to learn in the real world are not drawn uniformly from the set of all mathematically possible functions
  - Common reasonable assumptions behind ML's success: similar examples having similar classes, limited dependences, or limited complexity

    Choose a representation in which available knowledge is easily expressible

# Programming vs Learning

*Programming, like all engineering, is a lot of work:*

*We have to build everything from scratch.*

*Learning is more like farming, which lets nature do most of the work. Farmers combine seeds with nutrients to grow crops.*

*Learners combine knowledge with data to grow programs.*

# Lesson 2: Generalisation Error Decomposes into Bias vs. Variance

Generalisation error = bias + variance

- Bias: learner's tendency to consistently learn the same wrong thing
  - Linear learner has high bias: when the frontier between two classes is not a hyperplane the learner is unable to induce it.
  - Decision trees do not have this problem because they can represent any Boolean function.
- Variance: tendency to learn random things irrespective of the real signal
  - Decision trees suffer from high variance: when learned on different training sets generated by the same phenomenon they are often very different, when in fact they should be the same.

# Lesson 2: Generalisation Error Decomposes into Bias vs. Variance

Generalisation error = bias + variance

- Bias: learner's tendency to consistently learn the same wrong thing
  - Linear learner has high bias: when the frontier between two classes is not a hyperplane the learner is unable to induce it.
  - Decision trees do not have this problem because they can represent any Boolean function.
- Variance: tendency to learn random things irrespective of the real signal
  - Decision trees suffer from high variance: when learned on different training sets generated by the same phenomenon they are often very different, when in fact they should be the same.

More powerful learners are not necessarily better than less powerful ones!

In other words: Strong false assumptions can be better than weak true ones

- A learner with the latter needs more data to avoid overfitting

# How to Combat High Variance aka Overfitting

- Cross-validation
- Regularisation term to the evaluation function
  - Penalise classifiers with more structure
  - Favour smaller classifiers with less room to overfit
- Statistical significance test like chi-square before adding new structure
  - Decide whether the distribution of the class is different w/o this structure.

Avoiding overfitting may lead to high bias aka underfitting

# Lesson 3: Curse of Dimensionality

The biggest problem in machine learning (after overfitting)

Bellman (1961): Many algorithms that work fine in low dimensions become intractable when the input is high-dimensional.

- The similarity-based reasoning that machine learning algorithms depend on breaks down in high dimensions (many features).

Most of the volume of a high-dimensional orange is in the skin, not the pulp.

Generalisation becomes exponentially harder as the dimensionality of the examples grows: Training set covers a dwindling fraction of the input space.

Hope: "blessing of non-uniformity" in most applications

- Examples are not spread uniformly throughout the instance space
- Concentrated on or near a lower-dimensional manifold.

# Lesson 4: Theoretical Guarantees are not What They Seem

Induction (learning) vs deduction (database):

- In deduction you can guarantee that the conclusions are correct
- In induction all bets are off

Major development: (probabilistic) guarantees on the results of induction

- The bounds obtained are usually extremely loose

"Asymptopia": asymptotic theoretical guarantee:

- Given infinite data, the learner is guaranteed to output the correct classifier.
- We are seldom in the asymptotic regime
- Bias-Variance trade-off: If learner A is better than learner B given infinite data, B is often better than A given finite data.

Role of theoretical guarantees in ML:

- Source of understanding and driving force for algorithm design

# Lesson 5: Feature Engineering is the Key

Holy Grail of ML: Automate the feature engineering process

Easy, unrealistic learning: Many already known independent features that each correlate well with the class

In real world:

- Done today: automatically generate large numbers of candidate features and select the best by e.g., their information gain with respect to the class.

- Yet: Features irrelevant in isolation may be relevant in combination (XOR)!

  Complex function of the features $\Rightarrow$ unlikely to learn it

- Using many features to find out which ones are useful in combination may be too time-consuming, or cause overfitting.

# Lesson 6: Most effort in an ML project NOT on the Learner

Typical time-consuming tasks:

- Gather, integrate, clean, preprocess data
- Trial & error for feature design.
- Iterative process: Learn, analyse the results, modify the data/learner

# Lesson 7: More Data Beats a Cleverer Algorithm

ML is all about letting data do the heavy lifting $\Rightarrow$ Scalability = challenge

Limited resources: time and memory in CS; + training data in ML.

Today's bottleneck is often time

- Mountains of data available but not used! Not enough time to process it
- Paradox
  - In principle: More data $\Rightarrow$ more complex classifiers can be learned
  - In practice: Complex classifiers take too long to learn $\Rightarrow$ simpler ones are used
- GOAL: Come up with fast ways to learn complex classifiers

# More Data Beats a Cleverer Algorithm

- All learners essentially work by grouping nearby examples into the same class; the key difference is in the meaning of "nearby."
- Learners produce widely different frontiers while still making the same predictions in the regions that matter.
- Try the simplest learners first: naïve Bayes before logistic regression, k-nearest neighbour before support vector machines
- Sophisticated learners are harder to use, they have more knobs you need to turn to get good results and their internals are more opaque.

Clever algorithms make the most of the data and computing resources available

- Often pay off in the end, provided you are willing to put in the effort

Biggest bottleneck: Human cycles

Measures of success beyond accuracy and computational cost: saved human effort, insight gained, human-understandable output.

# Lesson 8: Learn many Models, not Just ONE

Observation: The best learner depends on the application

Reaction: Support many different learners

- Try out several learners, select the best
- Model ensembles: Combine many learner variations

Model ensembles are now standard

- Bagging: Generate random variations of the training set by resampling, learn a classifier on each, and combine the results by voting.
  - Greatly reduces variance while only slightly increasing bias.
- Boosting: Training examples have weights, and these are varied so that each new classifier focuses on the examples the previous ones tended to get wrong.
- Stacking: The outputs of individual classifiers become the inputs of a "higher-level" learner that figures out how best to combine them.

Best solutions for Netflix prize competition: stacked ensembles of 100+ learners

# Lesson 9: Simplicity Does not imply Accuracy

Claim: Given two classifiers with the same training error, the simpler of the two will likely have the lowest test error.

# Lesson 9: Simplicity Does not imply Accuracy

Claim: Given two classifiers with the same training error, the simpler of the two will likely have the lowest test error.

- Counterexamples: model ensembles; support vector machines, which can effectively have an infinite number of parameters without overfitting.
- "No free lunch" $\Rightarrow$ It cannot be true
- Counter-intuitive: No necessary connection between the number of parameters of a model and its tendency to overfit

Simpler hypotheses preferred because simplicity is a virtue in its own right

Occam's razor

(not because of a hypothetical connection with accuracy)