Machine Learning in the Data Revolution Era

Shai Shalev-Shwartz

School of Computer Science and Engineering
The Hebrew University of Jerusalem

Machine Learning Seminar Series,
Google & University of Waterloo,
July 2010

10 Years Ago ...

- ALLWEIN, SCHAPIRE & SINGER, 2000 (received Best 10-year paper at ICML 2010)

Problem	#Examples Train	Test	#Attributes	#Classes
dermatology	366	-	34	6
satimage	4435	2000	36	6
glass	214	-	9	7
segmentation	2310	-	19	7
ecoli	336	-	8	8
pendigits	7494	3498	16	10
yeast	1484	-	8	10
vowel	528	462	10	11
soybean	307	376	35	19
thyroid	9172	-	29	20
audiology	226	-	69	24
isolet	6238	1559	617	26
letter	16000	4000	16	26

Table 1: Description of the datasets used in the experiments.

Data Revolution

In 2009, more data has been generated by individuals than in the entire history of mankind through 2008 ...

WIKIPEDIA

facebook

Gmail by Google BETA

amazon.com

REUTERS

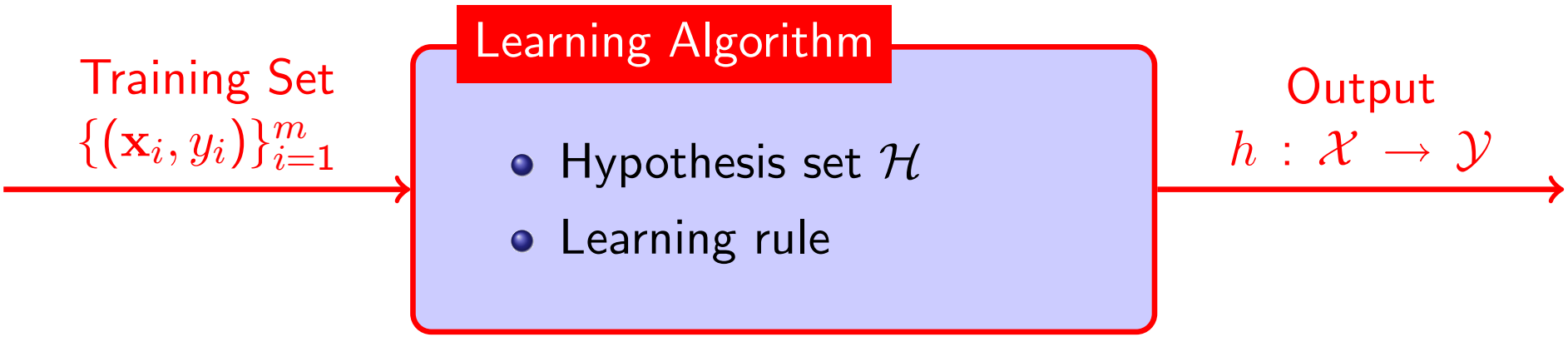
Discriminative Learning

Document

If only Tiger Woods could bond with fans as well as he did with his caddie on the Old Course practice range on Monday he might find the road to redemption all the more simple...

About Sport?

Discriminative Learning



E.g.

- $\mathcal{X}$ - instance domain (e.g documents)
- $\mathcal{Y}$ - label set (e.g. document category – is about sport?)
- Goal: learn a predictor $h : \mathcal{X} \to \mathcal{Y}$
- $\mathcal{H}$ - Set of possible predictors
- Learning rule - (e.g. "find $h \in \mathcal{H}$ with minimal error on training set")

Large Scale Learning

$10k$ training examples

1 hour

2.3% error

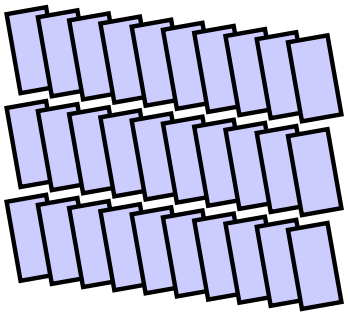
Large Scale Learning

$10k$ training examples

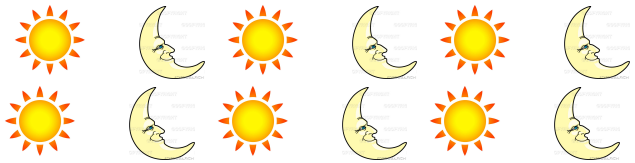
1 hour

2.3% error

$1M$ training examples



1 week



2.29% error

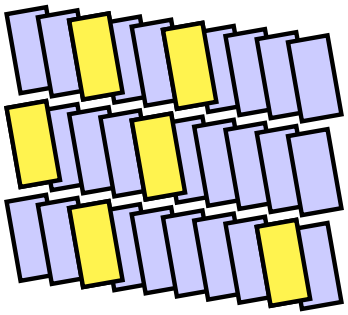
Large Scale Learning

$10k$ training examples

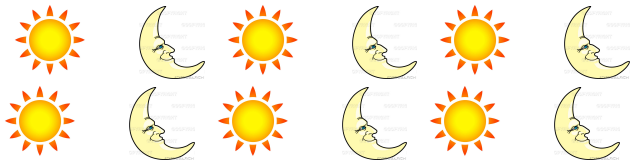
1 hour

2.3% error

$1M$ training examples



1 week



2.29% error

- Can always sub-sample and get error of 2.3% using 1 hour

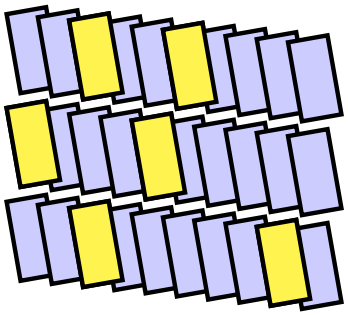
Large Scale Learning

$10k$ training examples

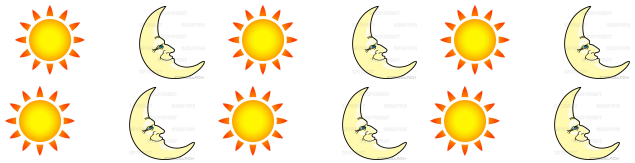
1 hour

2.3% error

$1M$ training examples



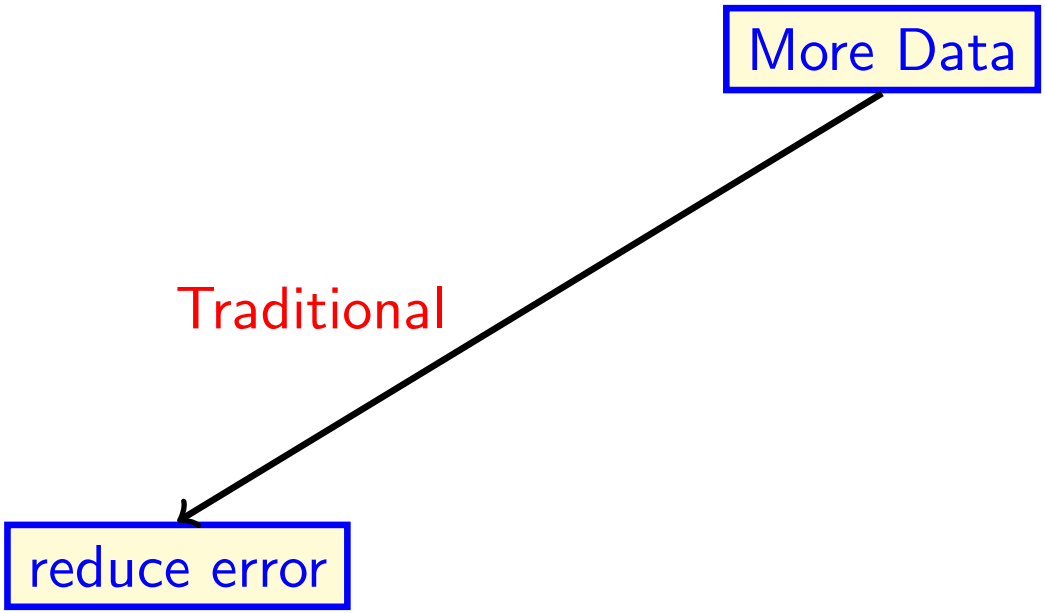
1 week



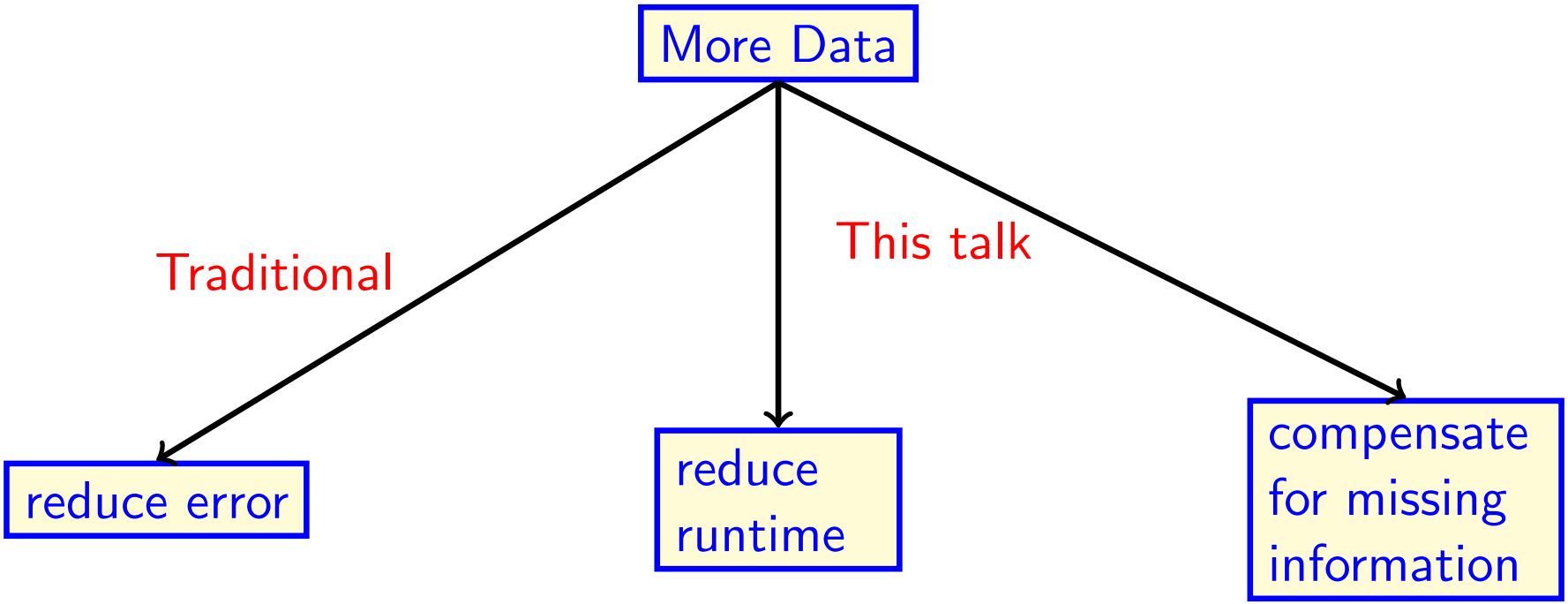
2.29% error

- Can always sub-sample and get error of 2.3% using 1 hour
- Say we're happy with 2.3% error
  What else can we gain from the larger data set?

How can more data help?



How can more data help?



Outline

How can more data reduce runtime?

- Learning using Stochastic Optimization (S. & Srebro 2008)
- Injecting Structure (S, Shamir, Sirdharan 2010)

How can more data compensate for missing information?

- Attribute Efficient Learning (Cesa-Bianchi, S., Shamir 2010)
  Technique: Rely on Stochastic Optimization
- Ads Placement (Kakade, S., Tewari 2008)
  Technique: Inject Structure by Exploration

Background on ML: Vapnik's General Learning Setting

Probably Approximately Solve:

$$\min_{h \in \mathcal{H}} \mathop{\mathbb{E}}_{z \sim \mathcal{D}} [f(h, z)]$$

where:

- $\mathcal{D}$ is an unknown distribution
- Can only get samples $z_1, \ldots, z_m \overset{\text{i.i.d.}}{\sim} \mathcal{D}$
- $\mathcal{H}$ is a known hypothesis class
- $f(h, z)$ is the loss of using hypothesis h on example z

Background on ML: Vapnik's General Learning Setting

Probably Approximately Solve:

$$\min_{h \in \mathcal{H}} \mathop{\mathbb{E}}_{z \sim \mathcal{D}} [f(h, z)]$$

where:

- $\mathcal{D}$ is an unknown distribution
- Can only get samples $z_1, \ldots, z_m \overset{\text{i.i.d.}}{\sim} \mathcal{D}$
- $\mathcal{H}$ is a known hypothesis class
- $f(h, z)$ is the loss of using hypothesis h on example z

Main question studied by Vapnik:
How many examples are required to find $\hat{h} = A(z_1, \ldots, z_m)$ s.t.

$$\mathop{\mathbb{E}}_{z \sim \mathcal{D}} \left[f(\hat{h}, z) \right] - \min_{h \in \mathcal{H}} \mathop{\mathbb{E}}_{z \sim \mathcal{D}} [f(h, z)] \leq \epsilon \quad \text{w.p.} \geq 1 - \delta$$

Examples

- Document Categorization:
  - $z = (\mathbf{x}, y)$, $\mathbf{x}$ represents a document and $y \in [k]$ is the topic
  - $h \in \mathcal{H}$ is a predictor $h : \mathcal{X} \to [k]$
  - $f(h, z) = \mathbf{1}[h(\mathbf{x}) \neq y]$
- k-means clustering:
  - $z \in \mathbb{R}^d$
  - $h \in \mathbb{R}^{k,d}$ specifies k cluster centers
  - $f(h, z) = \min_j \|h_{j,\to} - z\|$
- Density Estimation:
  - h is a parameter of a density $p_h(z)$
  - $f(h, z) = -\log p_h(z)$
- Ads Placement:
  - $z = (\mathbf{x}, \mathbf{c})$, $\mathbf{x}$ represents context and $\mathbf{c} \in [0, 1]^k$ represents the gain of placing each ad
  - $h : \mathcal{X} \to \mathbb{S}^k$ returns the probability to place each ad
  - $f(h, z) = \langle h(x), \mathbf{c} \rangle$ is the expected gain of following policy h

Vapnik's Learning Rule

Empirical Risk Minimization:

Return a hypothesis with smallest risk on the training examples:

$$\hat{h} \in \operatorname*{argmin}_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} f(h, z_i)$$

Vapnik's Learning Rule

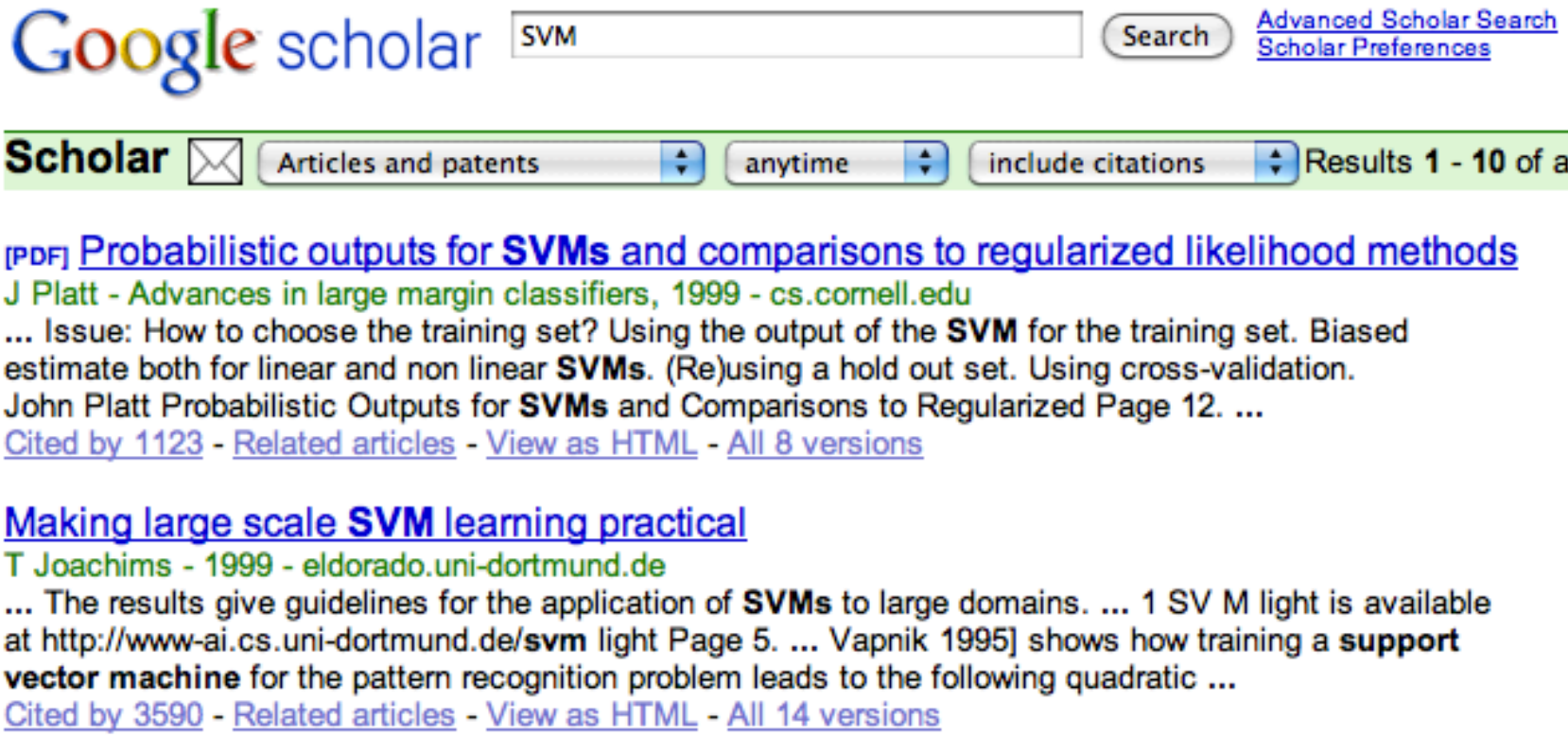
Empirical Risk Minimization:
Return a hypothesis with smallest risk on the training examples:

$$\hat{h} \in \operatorname*{argmin}_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} f(h, z_i)$$

Sample Complexity: If $m \geq \Omega\left(\frac{C(\mathcal{H}) \log(1/\delta)}{\epsilon^2}\right)$, where $C(\mathcal{H})$ is some complexity measure, then $\hat{h}$ is Probably Approximately Correct.

Example: Support Vector Machines

SVM is a very popular Machine Learning algorithm



Support Vector Machines

SVM learning rule:

- Given samples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ solve:

$$\min_{\mathbf{w}} \ \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max\{0, 1 - y_i\langle \mathbf{w}, \mathbf{x}_i\rangle\}$$

Casting in the General Learning Setting:

- $z = (\mathbf{x}, y)$
- $h = \mathbf{w}$
- $f(h, z) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\} + \frac{\lambda}{2}\|\mathbf{w}\|^2$
- SVM learning rule is exactly ERM for the above loss function

Solving the SVM problem

- SVM learning rule can be rewritten as a Quadratic Optimization Problem.
- Solvable in polynomial time by standard solvers. Dedicated solvers also exist. End of story?

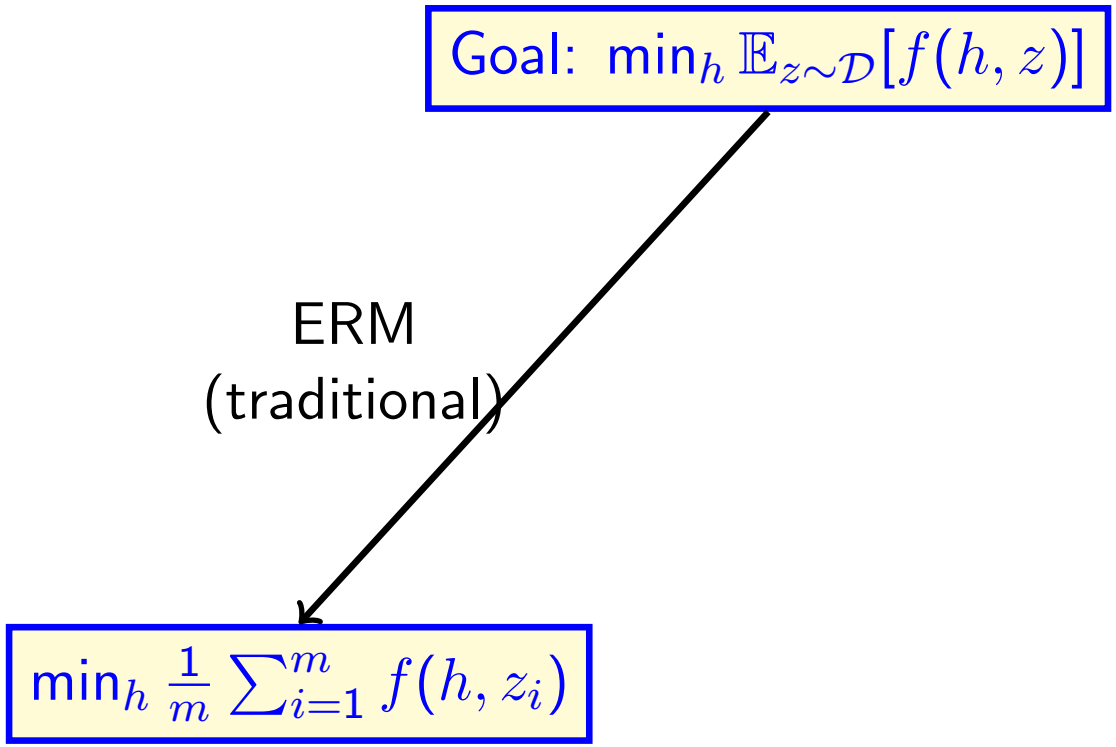
Solving the SVM problem

- SVM learning rule can be rewritten as a Quadratic Optimization Problem.
- Solvable in polynomial time by standard solvers. Dedicated solvers also exist. End of story?
- Not quite... Runtime of standard solvers increases with m
- What if m is very large ?

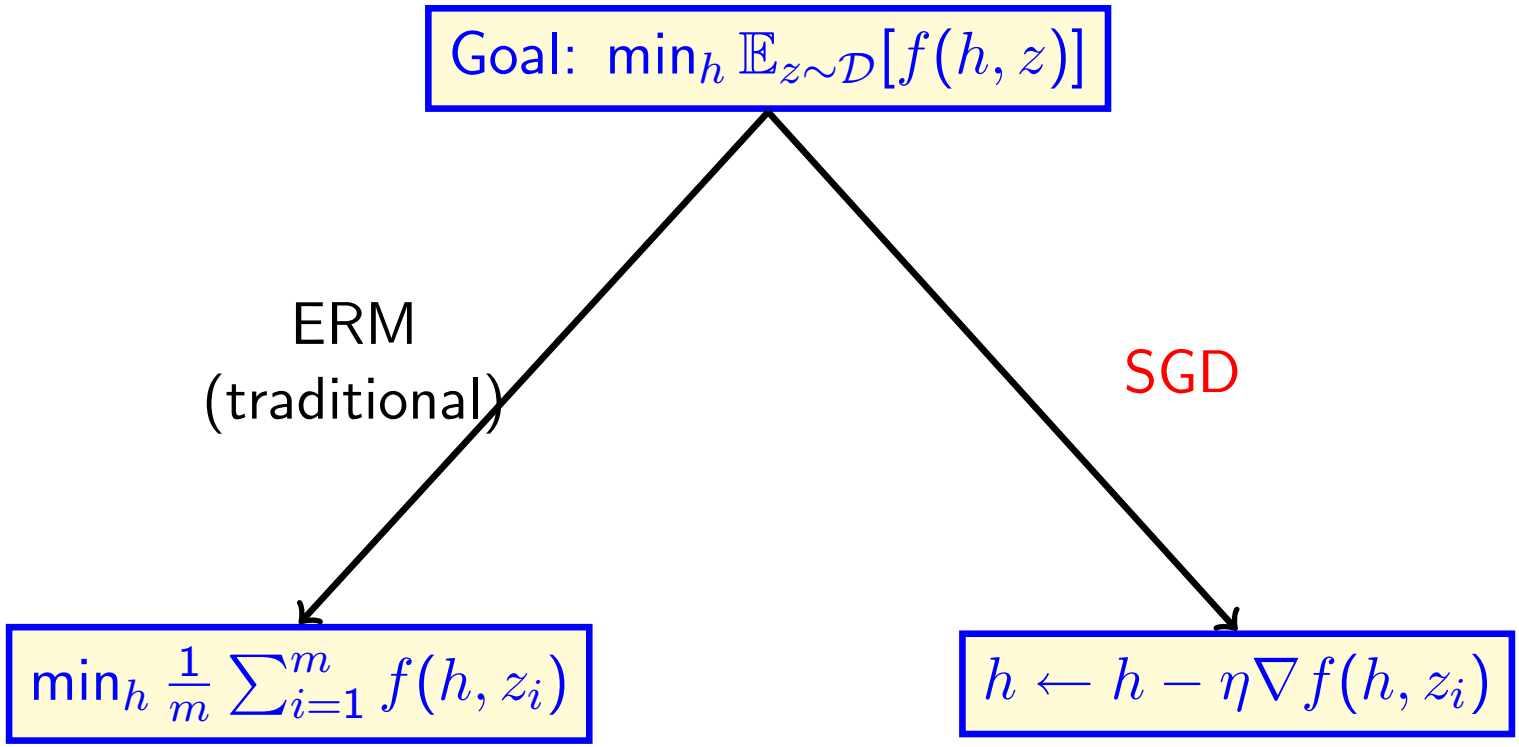
Machine Learning Perspective on Optimization

- Our real goal is *not* to minimize the SVM optimization problem
- SVM optimization problem is just a proxy for achieving the real goal — find a hypothesis with small error on unseen data
- Should study runtime required to solve the *Learning Problem*:
  - $T(m, \epsilon)$ – runtime required to find ϵ-accurate solution w.r.t. unseen data when number of training examples available is m
  - $T(\infty, \epsilon)$ – data laden analysis – data is plentiful and computational complexity is the real bottleneck

ERM vs. Stochastic Optimization



ERM vs. Stochastic Optimization

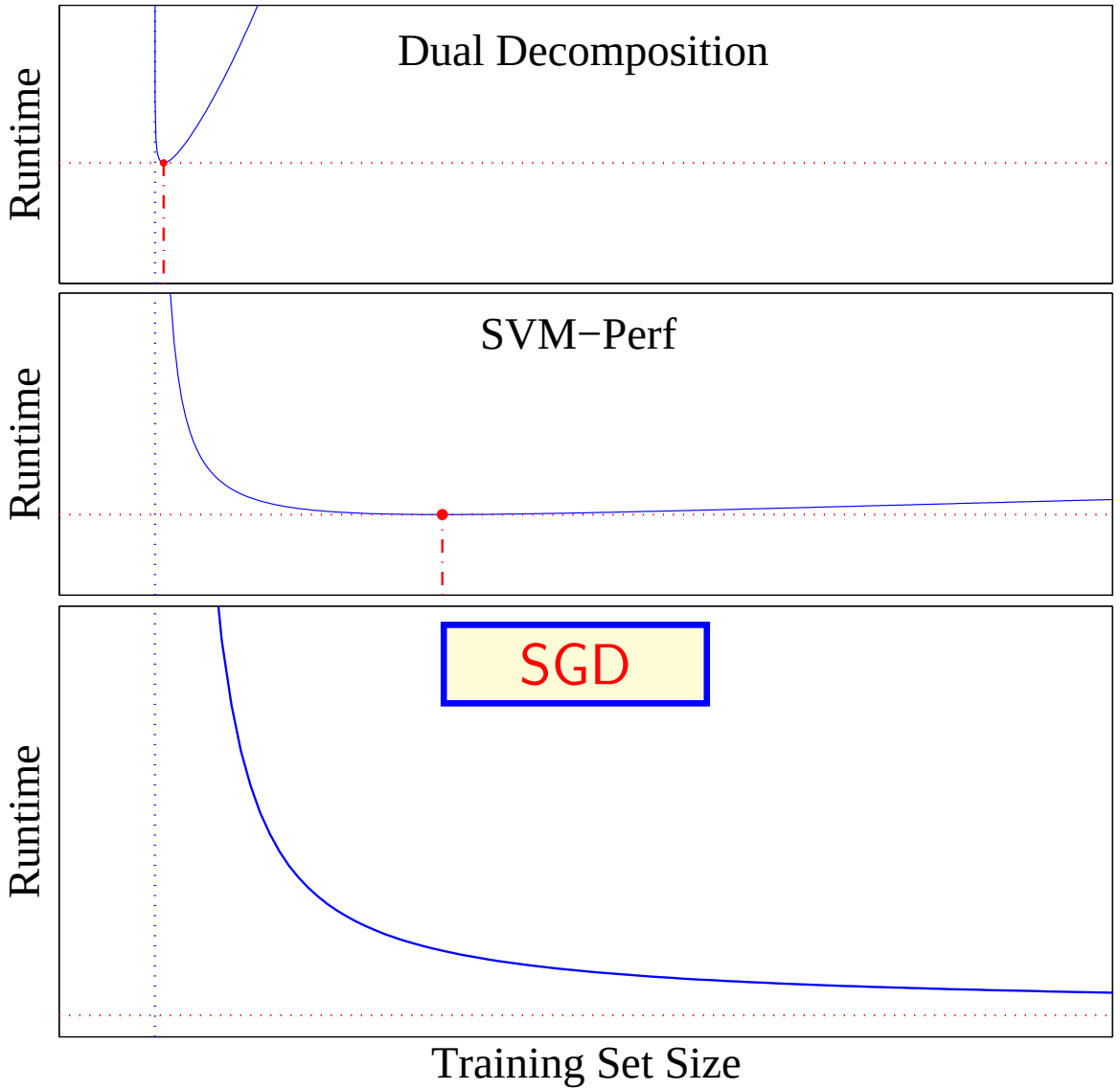


Data Laden Analysis of SVM Solvers

Method	Optimization runtime	$T(\infty, \epsilon)$
Interior Point	$m^{3.5} \log \log \frac{1}{\epsilon}$	$\frac{1}{\epsilon^7}$
Dual decompositoin	$m^2 \log \frac{1}{\epsilon}$	$\frac{1}{\epsilon^4}$
SVM-Perf (Joachims '06)	$\frac{m}{\lambda \epsilon}$	$\frac{1}{\epsilon^4}$
SGD (S, Srbero, Singer '07)	$\frac{1}{\lambda \epsilon}$	$\frac{1}{\epsilon^2}$

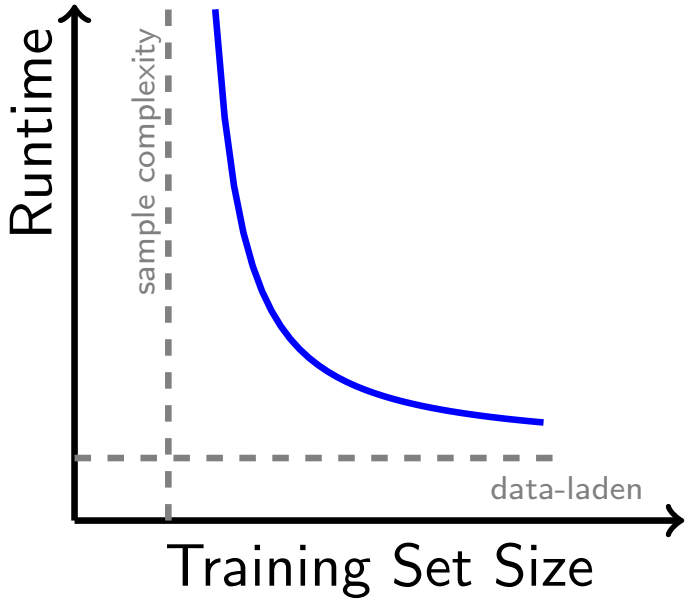
"Best" optimization method is the worst learning method . . .

More Data Less Work

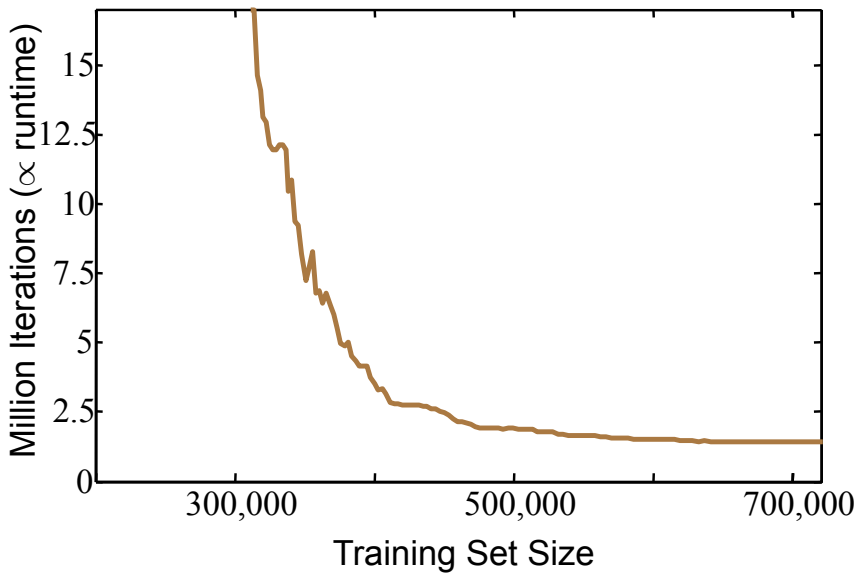


More Data Less Work

Theoretical



Empirical (CCAT)



Intuition: Better to do simple operations on more examples than complicated operations on fewer examples

Outline

How can more data reduce runtime?

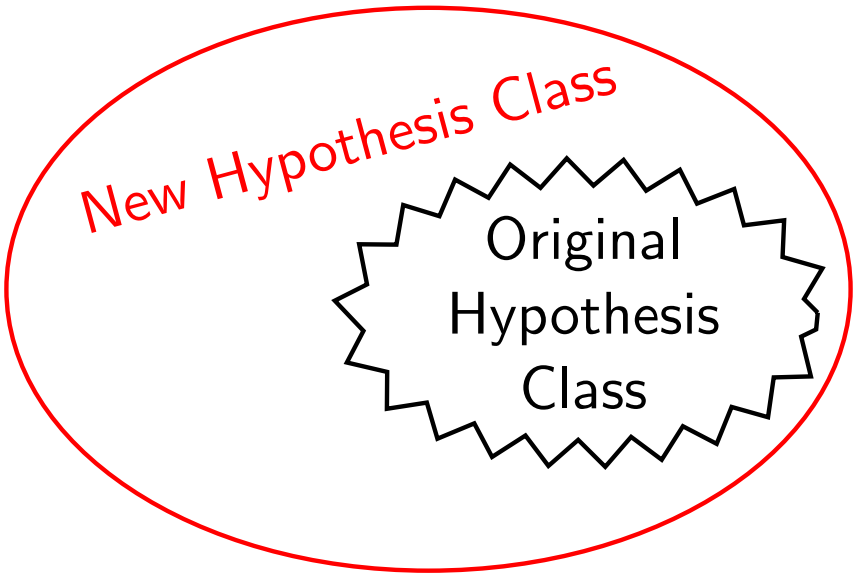
- Learning using Stochastic Optimization (S. & Srebro 2008) ✓
- Injecting Structure (S, Shamir, Sirdharan 2010)

How can more data compensate for missing information?

- Attribute Efficient Learning (Cesa-Bianchi, S., Shamir 2010)
  Technique: rely on Stochastic Optimization
- Ads Placement (Kakade, S., Tewari 2008)
  Technique: Inject Structure by Exploration

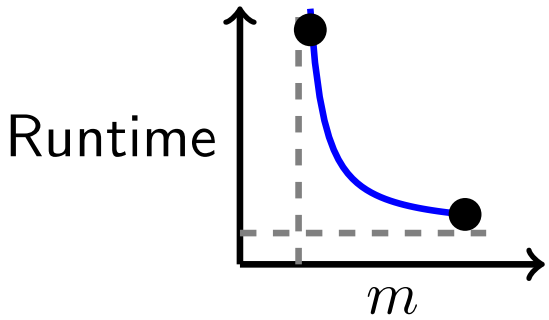
Injecting Structure – Main Idea

- Replace original hypothesis class with a larger hypothesis class
- On one hand: Larger class has more structure $\Rightarrow$ easier to optimize
- On the other hand: Larger class $\Rightarrow$ larger estimation error $\Rightarrow$ need more examples



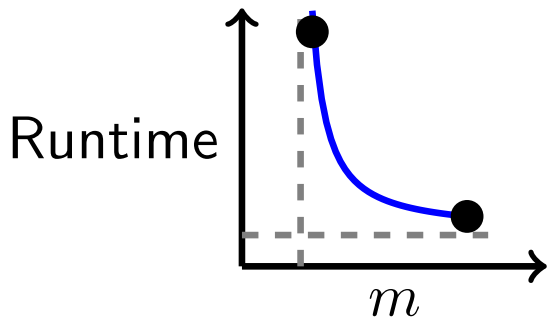
Example — Learning 3-DNF

- Goal: learn a Boolean function $h : \{0,1\}^d \to \{0,1\}$



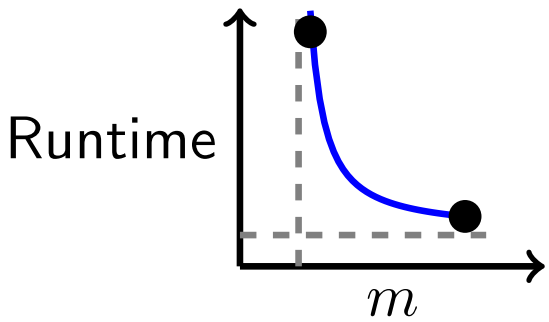
Example — Learning 3-DNF

- Goal: learn a Boolean function $h : \{0,1\}^d \to \{0,1\}$
- Hypothesis class: 3-term DNF formulae:



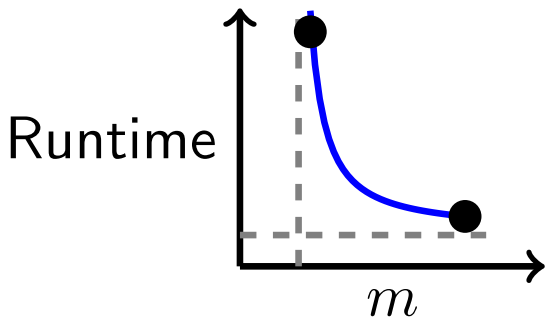
Example — Learning 3-DNF

- Goal: learn a Boolean function $h : \{0,1\}^d \to \{0,1\}$
- Hypothesis class: 3-term DNF formulae:
  - E.g. $h(\mathbf{x}) = (x_1 \wedge \neg x_3 \wedge x_7) \vee (x_4 \wedge x_2) \vee (x_5 \wedge \neg x_9)$



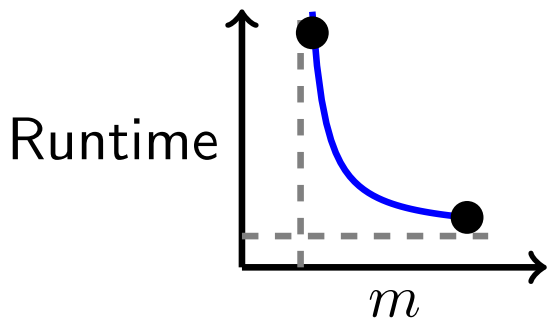
Example — Learning 3-DNF

- Goal: learn a Boolean function $h : \{0,1\}^d \to \{0,1\}$
- Hypothesis class: 3-term DNF formulae:
  - E.g. $h(\mathbf{x}) = (x_1 \wedge \neg x_3 \wedge x_7) \vee (x_4 \wedge x_2) \vee (x_5 \wedge \neg x_9)$
- Sample complexity is order d/ϵ



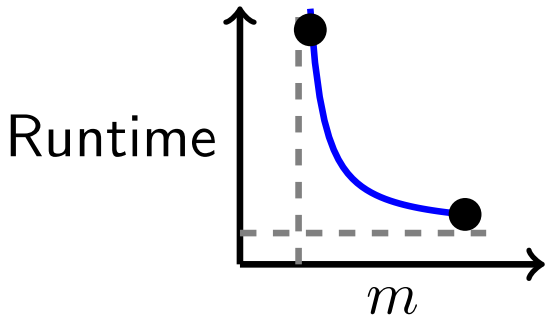
Example — Learning 3-DNF

- Goal: learn a Boolean function $h : \{0,1\}^d \to \{0,1\}$
- Hypothesis class: 3-term DNF formulae:
  - E.g. $h(\mathbf{x}) = (x_1 \wedge \neg x_3 \wedge x_7) \vee (x_4 \wedge x_2) \vee (x_5 \wedge \neg x_9)$
- Sample complexity is order d/ϵ
- Kearns & Vazirani: If RP$\neq$NP, it is not possible to efficiently find an ϵ-accurate 3-DNF formula



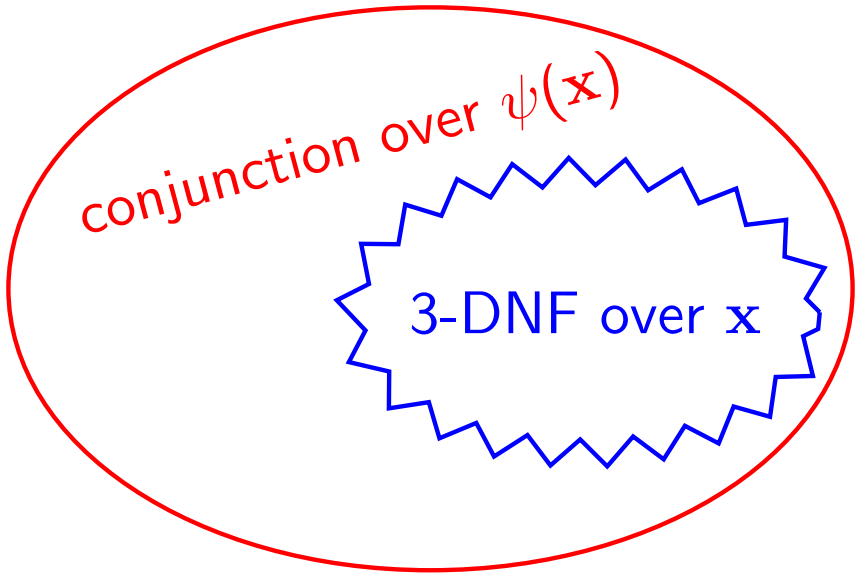
Example — Learning 3-DNF

- Goal: learn a Boolean function $h : \{0,1\}^d \to \{0,1\}$
- Hypothesis class: 3-term DNF formulae:
  - E.g. $h(\mathbf{x}) = (x_1 \wedge \neg x_3 \wedge x_7) \vee (x_4 \wedge x_2) \vee (x_5 \wedge \neg x_9)$
- Sample complexity is order d/ϵ
- Kearns & Vazirani: If RP$\neq$NP, it is not possible to efficiently find an ϵ-accurate 3-DNF formula
- Claim: if $m \geq d^3/\epsilon$ it is possible to find a predictor with error $\leq \epsilon$ in polynomial time



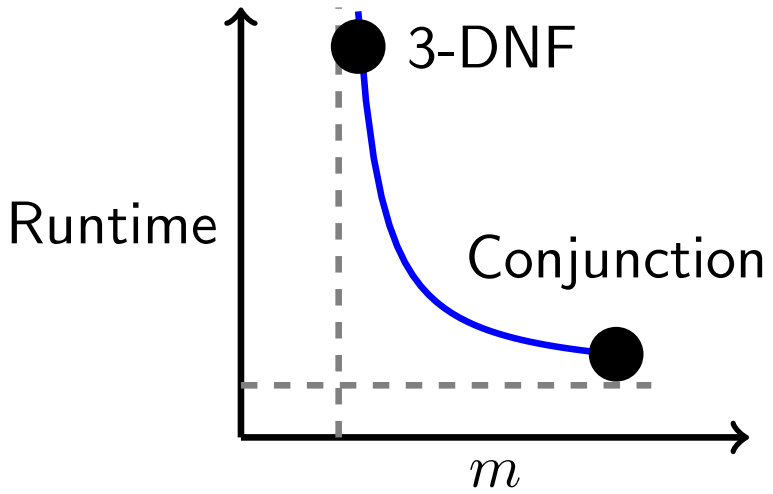
Proof

- Observation: 3-DNF formula can be rewritten as $\wedge_{u\in T_1, v\in T_2, w\in T_3}(u \vee v \vee w)$ for three sets of literals T_1, T_2, T_3
- Define: $\psi : \{0,1\}^d \rightarrow \{0,1\}^{2(2d)^3}$ s.t. for each triplet of literals u, v, w there are two variables indicating if $u \vee v \vee w$ is true or false
- Observation: Each 3-DNF can be represented as a single conjunction over $\psi(\mathbf{x})$
- Easy to learn single conjunction (greedy or LP)

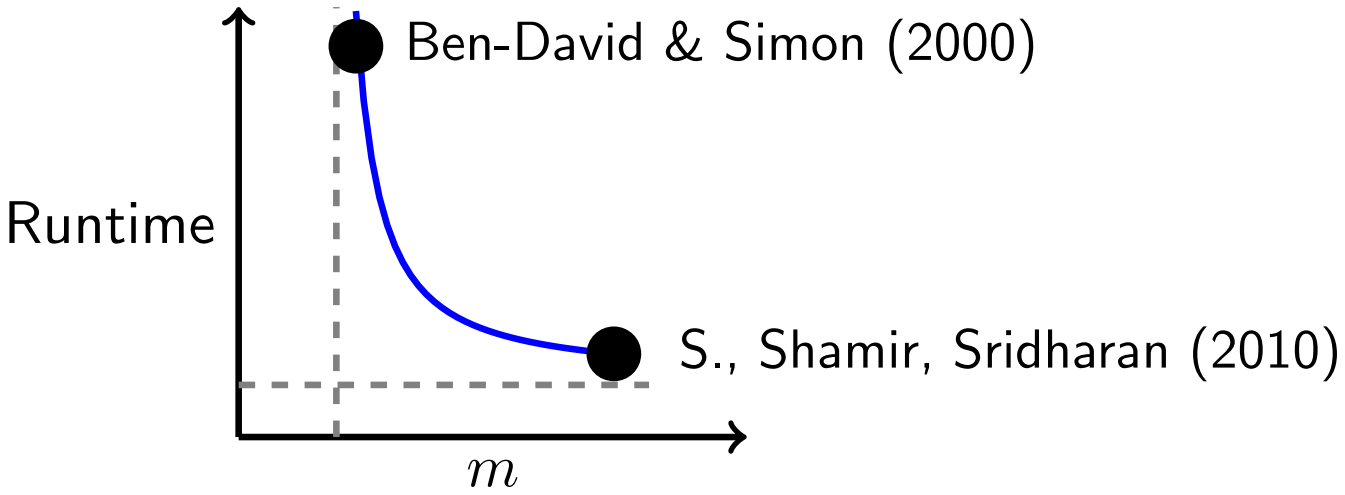


Trading samples for runtime

Algorithm	samples	runtime
3-DNF over $\mathbf{x}$	$\frac{d}{\epsilon}$	2^d
Conjunction over $\psi(\mathbf{x})$	$\frac{d^3}{\epsilon}$	$\text{poly}(d)$



More complicated example: Agnostic learning of Halfspaces with $0-1$ loss



Outline

How can more data reduce runtime?

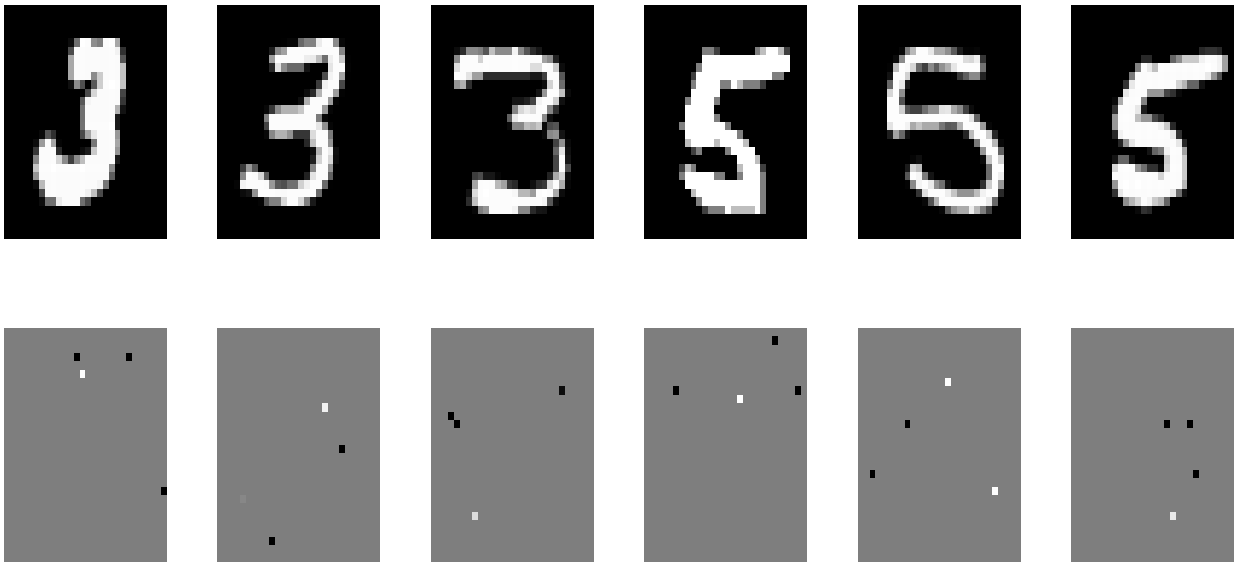
- Learning using Stochastic Optimization (S. & Srebro 2008) ✓
- Injecting Structure (S, Shamir, Sirdharan 2010) ✓

How can more data compensate for missing information?

- Attribute Efficient Learning (Cesa-Bianchi, S., Shamir 2010)
  Technique: rely on Stochastic Optimization
- Ads Placement (Kakade, S., Tewari 2008)
  Technique: Inject Structure by Exploration

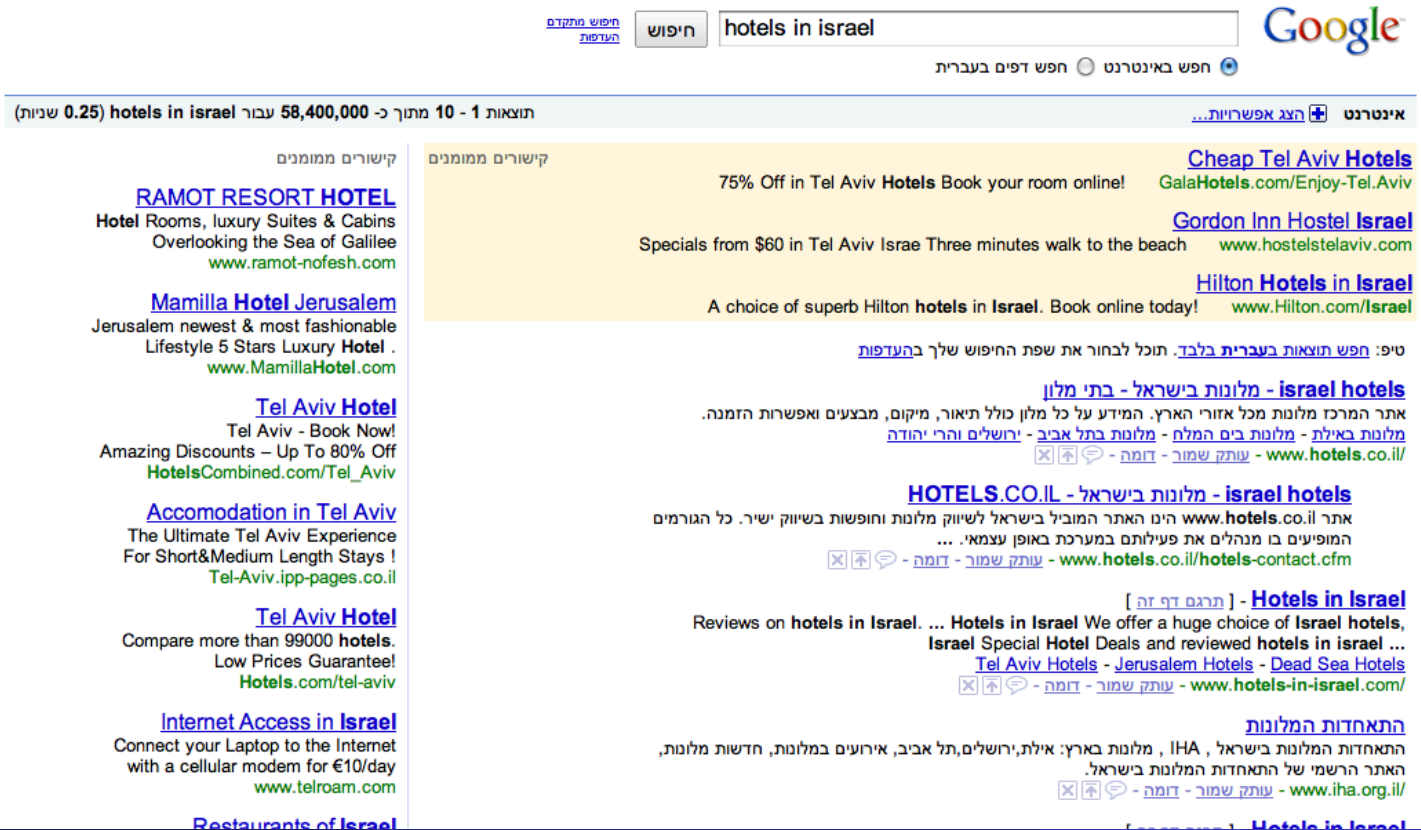
Attribute efficient regression

- Each training example is a pair $(\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$
- Partial information: can only view $O(1)$ attributes of each individual example



Sponsored Advertisement (contextual multi-armed bandit)

- Each training example is a pair $(\mathbf{x}, \mathbf{c}) \in \mathbb{R}^d \times [0,1]^k$
- Partial information: Don't know $\mathbf{c}$. Can only guess some y and know the value of c_y



How more data helps?

Three main techniques:

1. Missing information as noise
2. Active Exploration — try to "fish" the relevant information
3. Inject structure — problem hard in the original representation but becomes simple in another representation (different hypothesis class)

More data helps because:

1. It reduces variance — compensates for the noise
2. It allows more exploration
3. It compensates for larger sample complexity due to using larger hypotheses classes

Attribute efficient regression

Formal problem statement:

- Unknown distribution $\mathcal{D}$ over $\mathbb{R}^d \times \mathbb{R}$
- Goal: learn a linear predictor $\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle$ with low risk:
- Risk: $L_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{\mathcal{D}}[(\langle \mathbf{w}, \mathbf{x} \rangle - y)^2]$
- Training set: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
- Partial information: For each $(\mathbf{x}_i, y_i)$, learner can view only k attributes of $\mathbf{x}_i$
- Active selection: learner can choose which k attributes to see

Similar to "Learning with restricted focus of attention" (Ben-David & Dichterman 98)

Dealing with missing information

- Usually difficult — exponential ways to complete the missing information
- Popular approach — Expectation Maximization (EM)

Previous methods usually do not come with guarantees
(neither sample complexity nor computational complexity)

Ostrich approach

- Simply set the missing attributes to a default value
- Use your favorite regression algorithm for full information, e.g.,

$$\min_{\mathbf{w}} \sum_{i=1}^{m} (\langle \mathbf{w}, \tilde{\mathbf{x}}_i \rangle - y_i)^2 \; + \; \lambda \|\mathbf{w}\|_p^p$$

  - Ridge Regression: $p = 2$
  - Lasso: $p = 1$

Ostrich approach

- Simply set the missing attributes to a default value
- Use your favorite regression algorithm for full information, e.g.,

$$\min_{\mathbf{w}} \sum_{i=1}^{m} (\langle \mathbf{w}, \tilde{\mathbf{x}}_i \rangle - y_i)^2 \; + \; \lambda \|\mathbf{w}\|_p^p$$

  - Ridge Regression: $p = 2$
  - Lasso: $p = 1$

- Efficient
- Correct? Sample complexity? How to choose attributes ?

Partial information as noise

- Observation:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} = \frac{1}{d} \begin{pmatrix} dx_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \ldots + \frac{1}{d} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ dx_d \end{pmatrix}$$

- Therefore, choosing i uniformly at random gives

$$\mathop{\mathbb{E}}_{i}[dx_i \mathbf{e}^i] = \mathbf{x} \ .$$

- If $\|\mathbf{x}\| \leq 1$ then $\|dx_i \mathbf{e}^i\| \leq d$ (i.e. variance increased)
- Reduced missing information to unbiased noise
- Many examples can compensate for the added noise

Many examples compensates for noise

- True goal: minimize over $\mathbf{w}$ the true risk $L_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{(\mathbf{x},y)}[(\langle \mathbf{w}, \mathbf{x} \rangle - y)^2]$
- Loss on one example, $(\langle \mathbf{w}, \mathbf{x} \rangle - y)^2$, gives an unbiased estimate for $L_{\mathcal{D}}(\mathbf{w})$
- Averaging loss on many examples reduces variance
- In our case, we construct an unbiased estimate of the loss of each single example
- Variance increases but many examples still reduces it back

Loss-Based Attribute Efficient Regression (LaBAER)

Theorem (Cesa-Bianchi, S, Shamir)

Let $\hat{\mathbf{w}}$ be the output of LaBAER. Then, with overwhelming probability

$$L_{\mathcal{D}}(\hat{\mathbf{w}}) \leq \min_{\mathbf{w}:\|\mathbf{w}\|_1 \leq B} L_D(\mathbf{w}) + \tilde{O}\left(\frac{d^2\, B^2}{\sqrt{m}}\right) \ ,$$

where d is dimension and m is number of examples.

Loss-Based Attribute Efficient Regression (LaBAER)

Theorem (Cesa-Bianchi, S, Shamir)

Let $\hat{\mathbf{w}}$ be the output of LaBAER. Then, with overwhelming probability

$$L_{\mathcal{D}}(\hat{\mathbf{w}}) \leq \min_{\mathbf{w}:\|\mathbf{w}\|_1 \leq B} L_D(\mathbf{w}) + \tilde{O}\left(\frac{d^2\, B^2}{\sqrt{m}}\right) \ ,$$

where d is dimension and m is number of examples.

- Factor of d^4 more examples compensates for the missing information !

Loss-Based Attribute Efficient Regression (LaBAER)

Theorem (Cesa-Bianchi, S, Shamir)

Let $\hat{\mathbf{w}}$ be the output of LaBAER. Then, with overwhelming probability

$$L_{\mathcal{D}}(\hat{\mathbf{w}}) \leq \min_{\mathbf{w}:\|\mathbf{w}\|_1 \leq B} L_D(\mathbf{w}) + \tilde{O}\left(\frac{d^2\, B^2}{\sqrt{m}}\right) \ ,$$

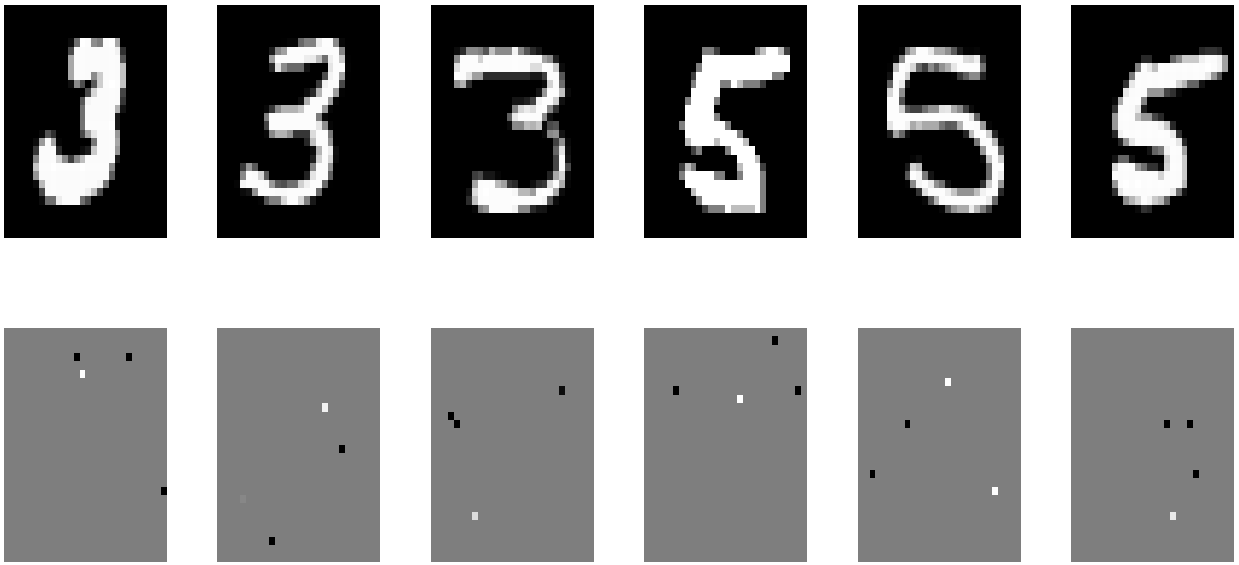
where d is dimension and m is number of examples.

- Factor of d^4 more examples compensates for the missing information !
- Can we do better?

Pegasos Attribute Efficient Regression (PAER)

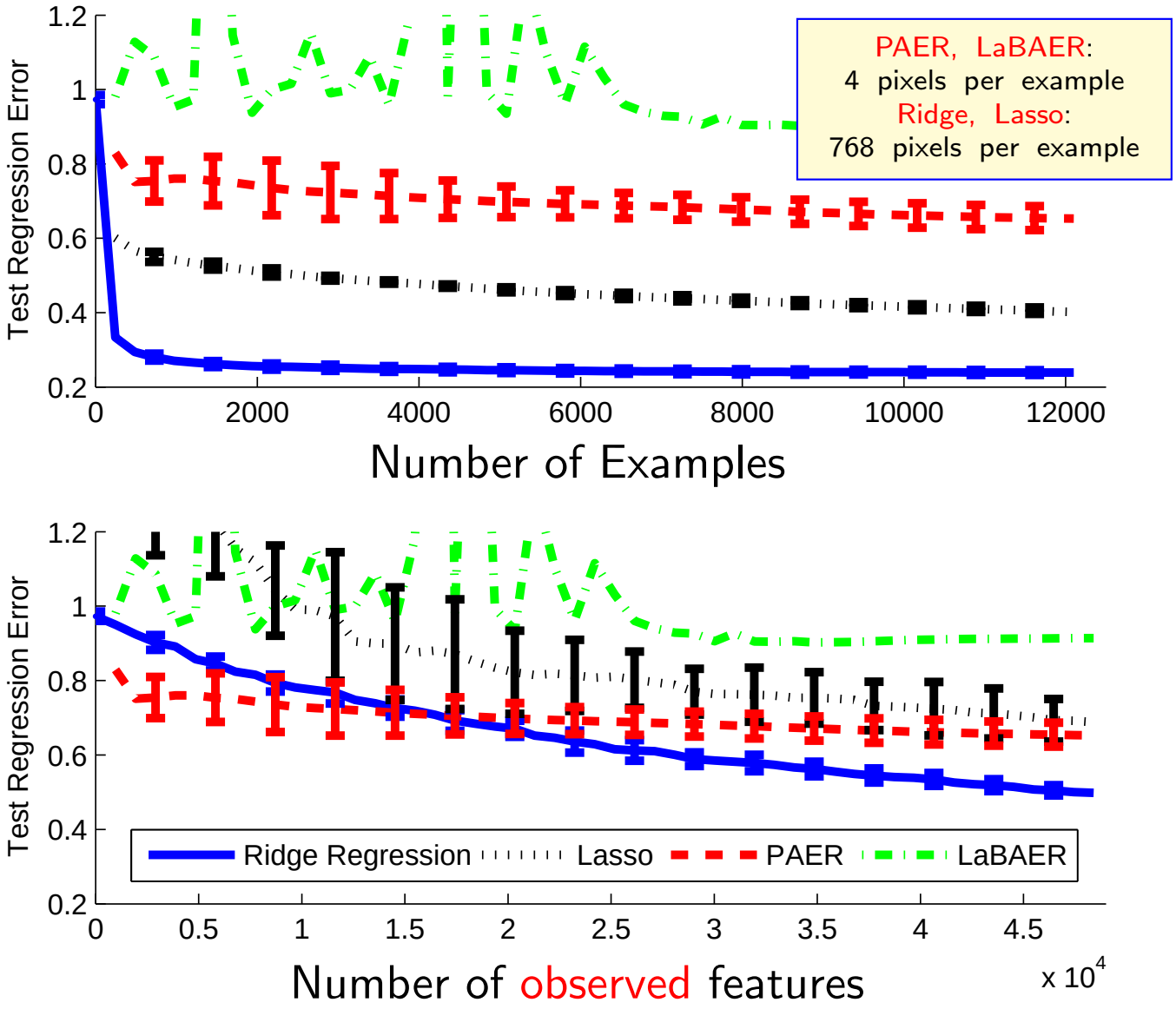
- The factor d^2 in the bound — because we estimate a matrix $\mathbf{x}\mathbf{x}^T$
- Can we avoid estimating the matrix ?
- Yes ! Estimate the gradient of the loss instead of the loss
- Follow the Stochastic Gradient Descent approach
- Leads to a much better bound

Demonstration



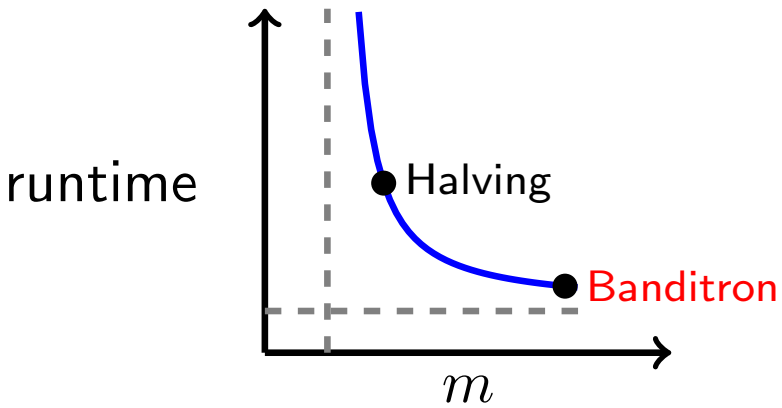
- Full information classifiers (top line) $\Rightarrow$ error of $\sim 1.5\%$
- Our algorithms (bottom line) $\Rightarrow$ error of $\sim 4\%$

Demonstration



Ads Placement: More data improves runtime

- Partial supervision
- Technique: Active Exploration
- Larger regret can be compensated by having more examples



Summary

- Learning theory: Many examples $\Rightarrow$ smaller error
- This work: Many examples $\Rightarrow$
  - higher efficiency
  - compensating for missing information
- Techniques:
  1. Stochastic optimization
  2. Inject structure
  3. Missing information as noise
  4. Active Exploration