Machine Learning & Pattern Recognition

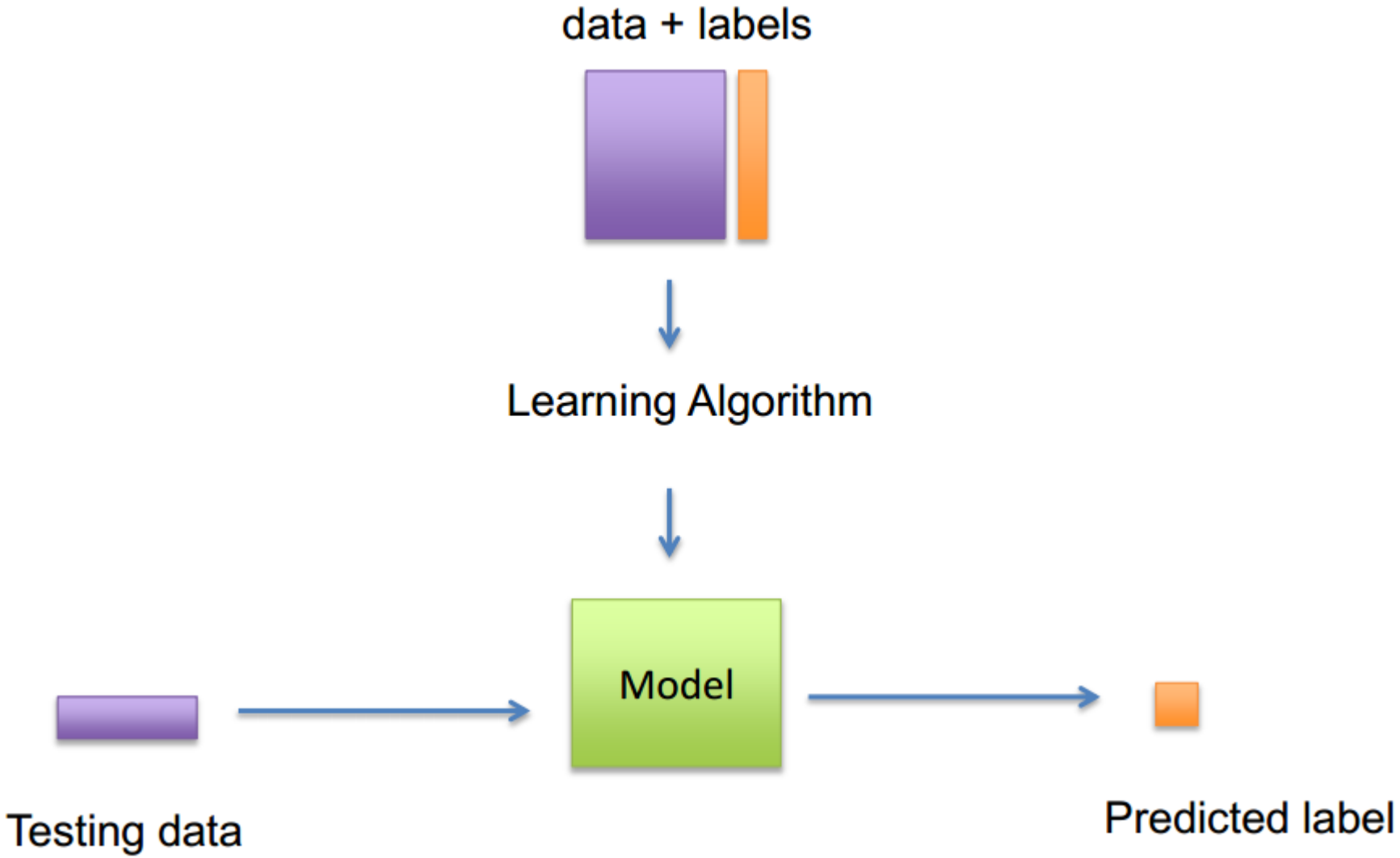
SONG Xuemeng

songxuemeng@sdu.edu.cn

http://xuemeng.bitcron.com/

Feature Selection

Supervised Learning



Text Classification

Is the story "interesting"?

It was a bright cold day in April, and the clocks were striking thirteen. Winston Smith, his chin nuzzled into his breast in an effort to escape the vile wind, slipped quickly through the glass doors of Victory Mansions, though not quickly enough to prevent a swirl of gritty dust from entering along with him.

Bag-of-words representation:

$x = \{0,2,0,0,1, \dots, 4,0,0,0,1\}$

One entry per word!

Easily 50,000 words! With big data…

- **Time complexity**
- **Computational cost**
- **Overfitting**
- **Lack of interpretability**

Feature selection

Some Things Matter, Some Do Not

- ***Relevant*** features
  - Those that we **need** to perform well
- ***Irrelevant*** features
  - Those that are simply **unnecessary**
- ***Redundant*** features
  - Those that **become** irrelevant in the presence of others

Feature Selection

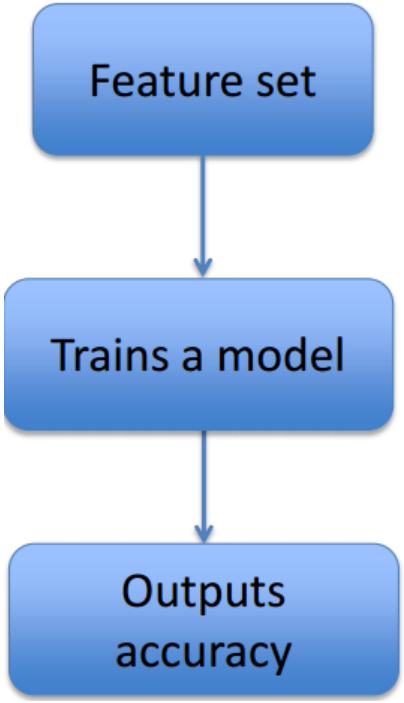
Given: a set of features $\boldsymbol{X} = \{X_1, X_2, \ldots X_D\}$ and a target variable Y.
Find: minimum set $\boldsymbol{S}$ that achieves maximum classification performance of $\boldsymbol{Y}$ (for a given set of classifiers and classification performance metrics)

Feature Selection Techniques

- **Wrappers methods**
- **Filters methods**
- **Embedded methods**

Feature Section (1): Wrapper Methods

Principle: We want to predict Y given the smallest possible subset of $\boldsymbol{X} = \{X_1, X_2, \ldots X_D\}$ while achieving maximal performance (e.g., accuracy).



- **Pros:**
  - Model-oriented
  - Usually gets good performance for the model you choose
- **Cons:**
  - Hugely computationally expensive

Feature Section (1): Wrapper Methods

With an exhaustive search

- With M features, 2^M possible feature subsets.

$$10111000000100010000100000000100101010$$

- 20 features… 1 million feature sets
- 25 features… 33.5 million sets
- 30 features…1.1 billion sets

Feature Section (1): Wrapper Methods

With an exhaustive search

- With M features, 2^M possible feature subsets.

$$10111000000100010000100000000100101010$$

- 20 features… 1 million feature sets
- 25 features… 33.5 million sets
- 30 features…1.1 billion sets

Need for a heuristic search strategy

1. **Sequential forward selection**
   - ❖ Keep adding features one at a time until no further improvement can be achieved
2. **Recursive backward elimination**
   - ❖ Start with the full set of predictors and keep removing features one at a time until no further improvement can be achieved

Wrappers: Sequential Forward Selection

Start with the empty set $S = \emptyset$

While stopping criteria not met

For each feature X_f not in S

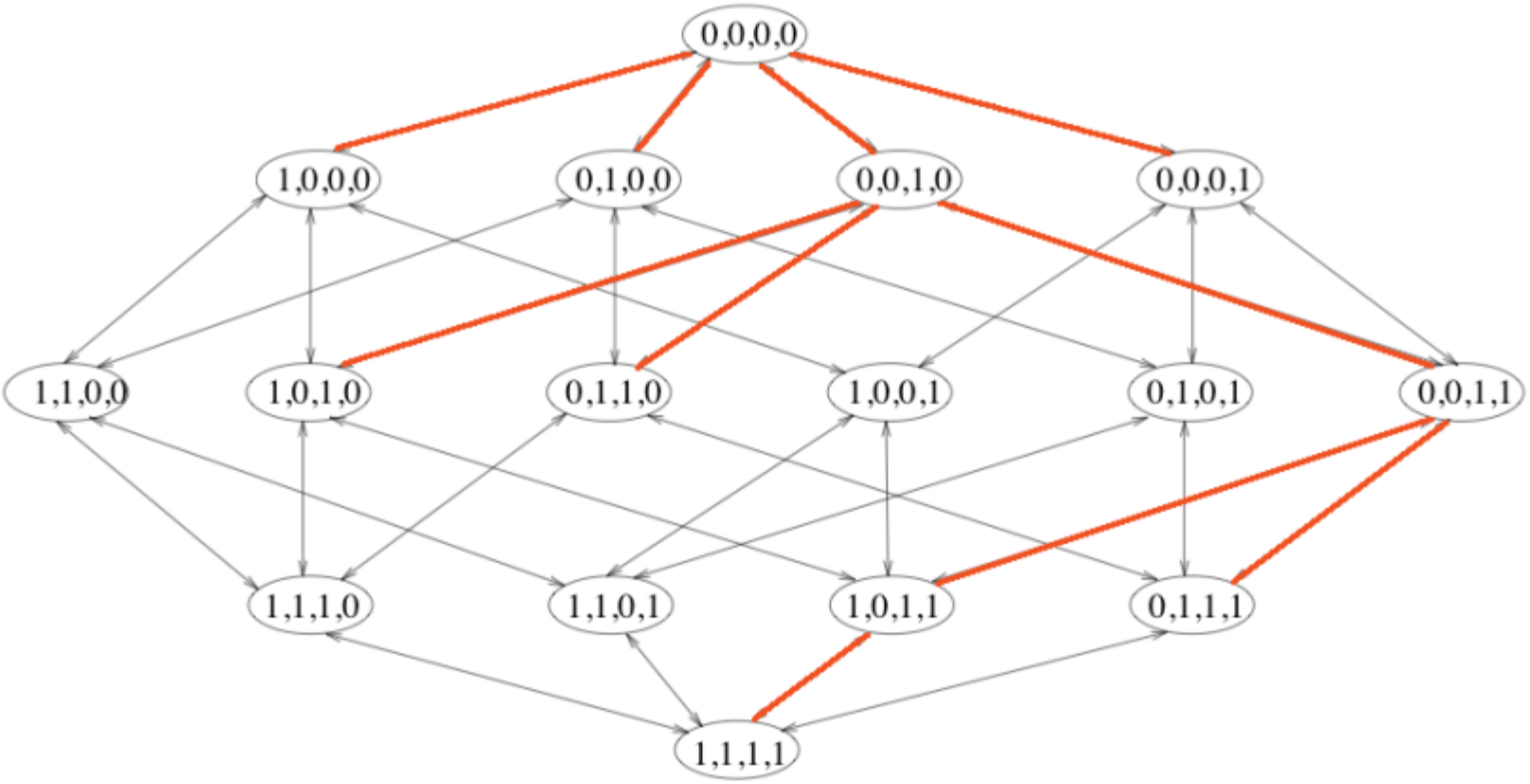
- Define $S' = S \cup \{X_f\}$
- Train model using the features in S'
- Compute the testing accuracy

End

$S = S'$ where S' is the feature set with the greatest accuracy

End

Search Complexity for Sequential Forward Selection



Evaluates $\frac{\mathrm{M(M+1)}}{2}$ feature sets instead of 2^M !

Feature Section (2): Filter Methods

Principle: replace evaluation of model with quick to compute statistics $J(\mathrm{X_f})$

k	$J(X_k)$
35	0.846
42	0.811
10	0.810
654	0.611
22	0.443
59	0.388
...	...
212	0.09
39	0.05

1. Score each feature $\mathrm{X_f}$ individually based on the f-th column of the data matrix and label vector Y.

   For each feature $\mathrm{X_f}$

   Compute $J(\mathrm{X_f})$

   End
2. Rank features according to $J(\mathrm{X_f})$.
3. Choose the top k features with the highest scores.

Feature Section (2): Filter Methods

Principle: replace evaluation of model with quick to compute statistics $J(\mathrm{X_f})$

k	$J(X_k)$
35	0.846
42	0.811
10	0.810
654	0.611
22	0.443
59	0.388
...	...
212	0.09
39	0.05

Examples of filtering criterion $J(\mathrm{X_f})$

- The mutual information
- Pearson r
- χ^2-statistic
- Gini index

Feature Section (2): Filter Methods

Principle: replace evaluation of model with quick to compute statistics $J(\mathrm{X_f})$

k	$J(X_k)$
35	0.846
42	0.811
10	0.810
654	0.611
22	0.443
59	0.388
...	...
212	0.09
39	0.05

Examples of filtering criterion $J(\mathrm{X_f})$

- The mutual information

  A measure of the mutual dependence between the two variables.

  Score $\mathrm{X_f}$ based on the MI with Y.

$$J(\mathrm{X_f}) = I(\mathrm{X_f}, Y)$$

Feature Section (2): Filter Methods

Principle: replace evaluation of model with quick to compute statistics $J(\mathrm{X_f})$

k	$J(X_k)$
35	0.846
42	0.811
10	0.810
654	0.611
22	0.443
59	0.388
...	...
212	0.09
39	0.05

Examples of filtering criterion $J(\mathrm{X_f})$

- Pearson r

  A measure of the linear correlation between two variables X and Y.

$$J(X_k) = \frac{cov(X_k, Y)}{\sqrt{var(X_k)}\sqrt{var(Y)}} \approx \frac{\sum_{i=1}^{N}\left(x_k^{(i)} - \overline{x_k}\right)\left(y^{(i)} - \bar{y}\right)}{\sqrt{\sum_{i=1}^{N}\left(x_k^{(i)} - \overline{x_k}\right)^2 \sum_{i=1}^{N}(y^{(i)} - \bar{y})^2}}$$

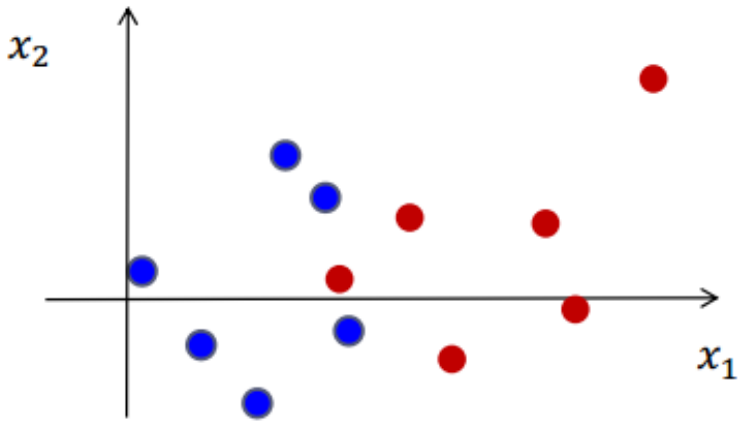
Feature Section (2): Filter Methods

Principle: replace evaluation of model with quick to compute statistics $J(\mathrm{X_f})$

k	$J(X_k)$
35	0.846
42	0.811
10	0.810
654	0.611
22	0.443
59	0.388
...	...
212	0.09
39	0.05

Examples of filtering criterion $J(\mathrm{X_f})$

- Pearson r



$$J(x_1) \quad ? \quad J(x_2)$$

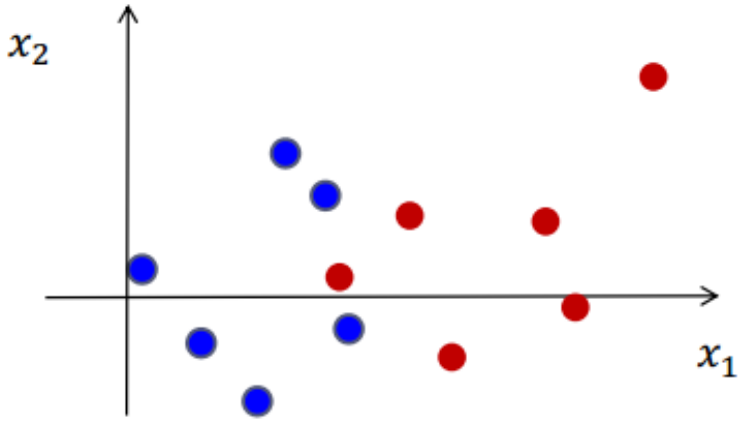
Feature Section (2): Filter Methods

Principle: replace evaluation of model with quick to compute statistics $J(\mathrm{X_f})$

k	$J(X_k)$
35	0.846
42	0.811
10	0.810
654	0.611
22	0.443
59	0.388
...	...
212	0.09
39	0.05

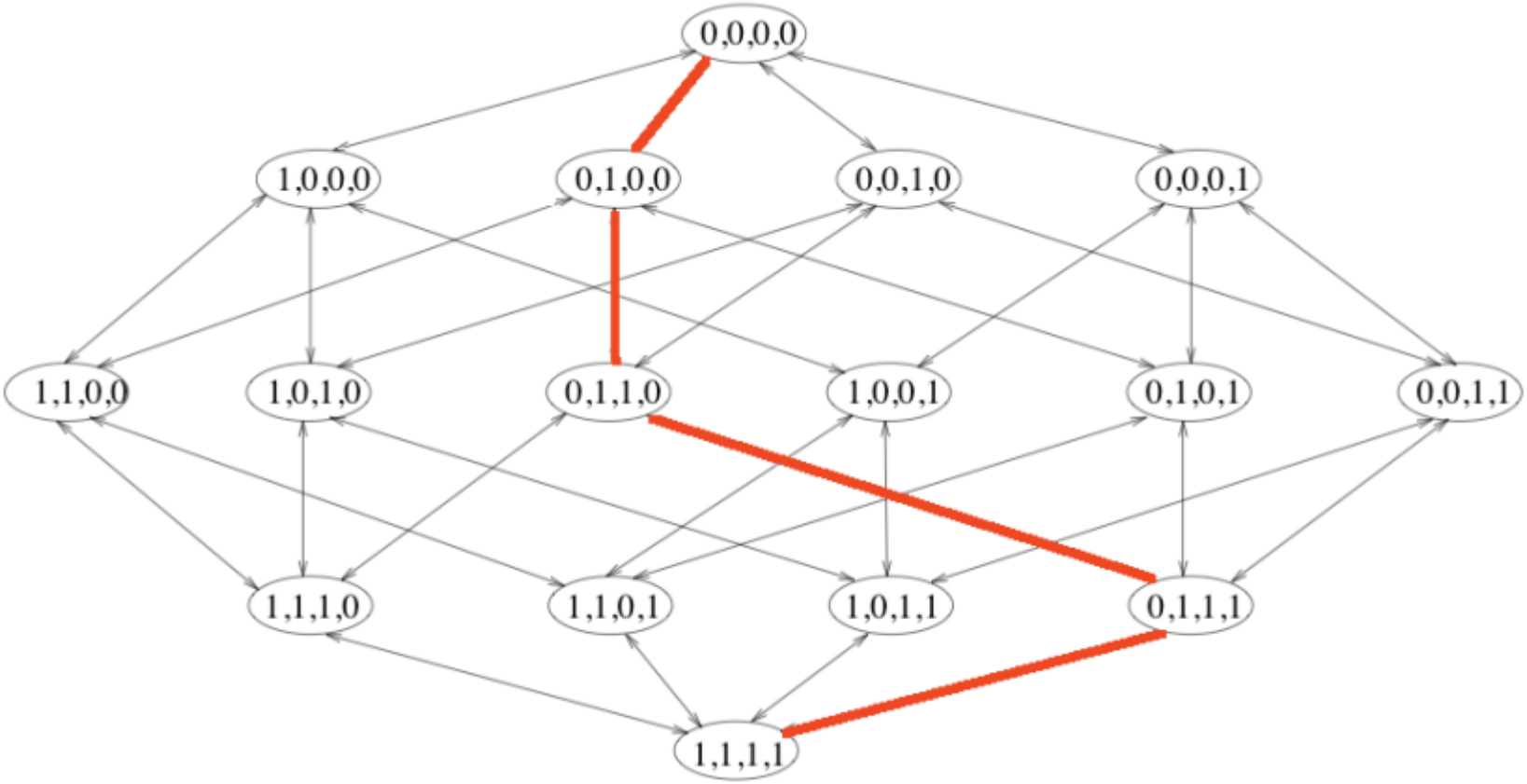
Examples of filtering criterion $J(\mathrm{X_f})$

- Pearson r



$$J(x_1) > J(x_2)$$

Search Complexity for Filter Methods



Pros:

- A lot less expensive!

Cons:

- Not model-oriented

Feature Section (3): Embedded methods

Principle: the classifier performs feature selection as part of the learning procedure.

Example: add the regularization term in the objective function

$$E_D(w) + \lambda E_W(w)$$

Underfitting & Overfitting

- The central challenge in machine learning is that we must perform well on **new, previously unseen** inputs—not just those on which our model was trained.

- The ability to perform well on previously unobserved inputs is called ***generalization***.
- We typically estimate the generalization error (also called test error) of a model by measuring its performance on a *test set* which is collected separately from the training set.

Underfitting & Overfitting

- **Typically,**
  1. We sample the training set
  2. Then use it to choose the parameters to reduce training set error
  3. Then evaluate the model with the test set.
- **Under this process,**
  - the expected test error $\geq$ the training error.

- The factors determining how well a model will perform are its ability to:
  - Make the training error small.
  - Make the gap between training and test error small.

Underfitting & Overfitting

- The factors determining how well a model will perform are its ability to:
  - Make the training error small.
  - Make the gap between training and test error small.

- **Underfitting** occurs when the model is not able to obtain a sufficiently low error value on the training set.
- **Overfitting** occurs when the gap between the training error and test error is too large.

Underfitting & Overfitting

- We can control whether a model is more likely to overfit or underfit by altering its ***capacity***.
- Informally, a model's capacity is its ability to fit a wide variety of functions.
- One way to control the capacity of a learning algorithm is by choosing its ***hypothesis space***, the set of functions that the learning algorithm is allowed to select as being the solution.

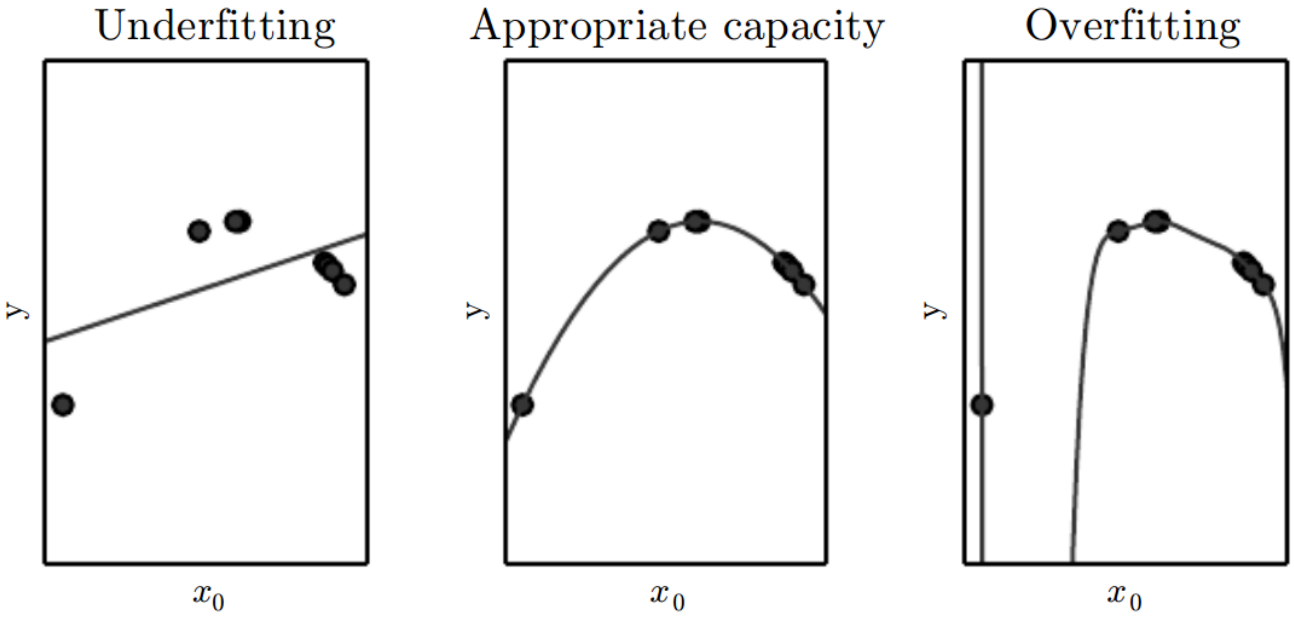
Linear regression	$y = b + wx$
Introduce x^2 (quadratic model)	$y = b + w_1 x + w_2 x^2$
Continue to add more powers of x	$y = b + \sum_{i=1}^{9} w_i x^i$

Underfitting & Overfitting

Machine learning algorithms will generally perform best when their capacity is appropriate in regard to the true complexity of the task they need to perform and the amount of training data they are provided with.

Underfitting & Overfitting

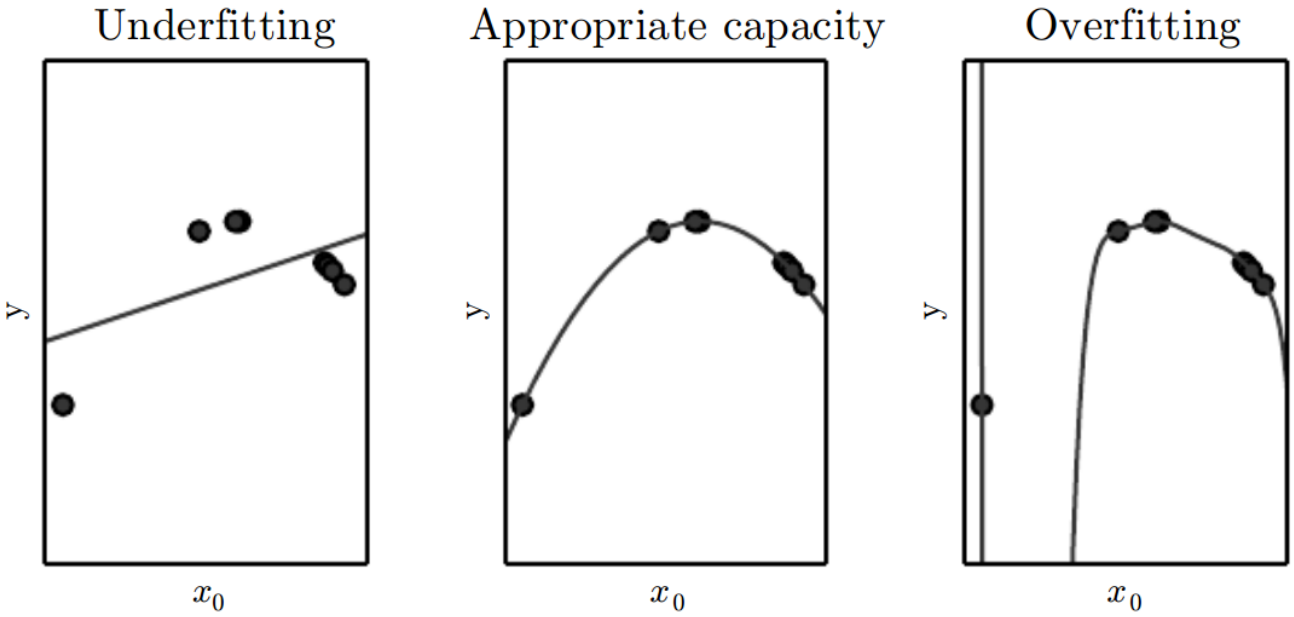
Machine learning algorithms will generally perform best when their capacity is appropriate in regard to the true complexity of the task they need to perform and the amount of training data they are provided with.



We compare a linear, quadratic and degree-9 predictor attempting to fit a problem where the true underlying function is quadratic.

Underfitting & Overfitting

- Models with low capacity may struggle to fit the training set.
- Models with high capacity can overfit by memorizing properties of the training set that do not serve them well on the test set.



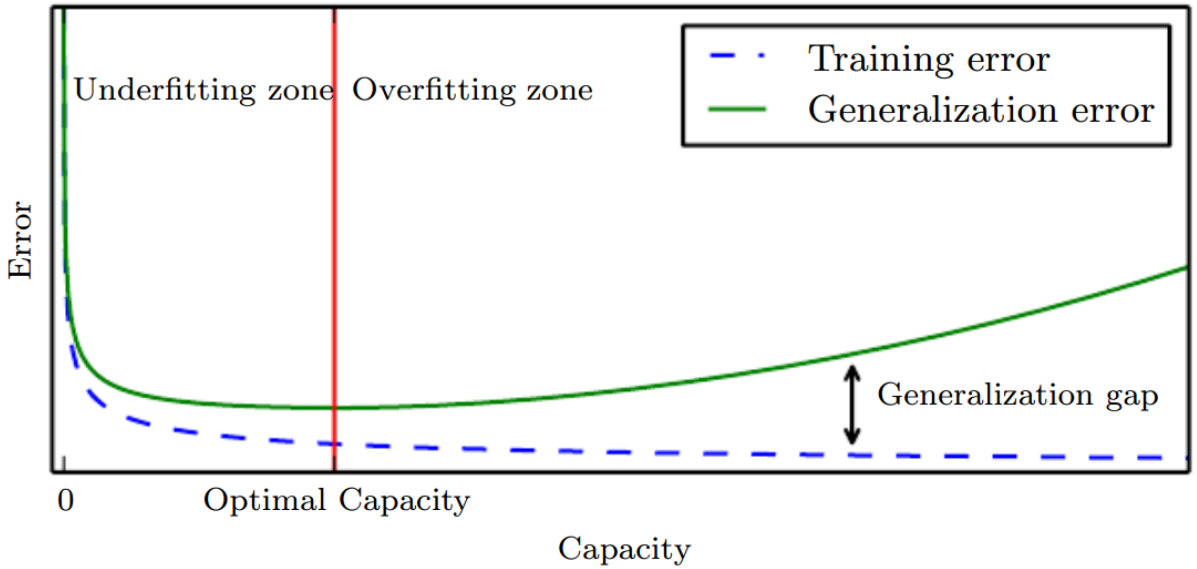
We compare a linear, quadratic and degree-9 predictor attempting to fit a problem where the true underlying function is quadratic.

Underfitting & Overfitting

- Simpler functions are more likely to generalize.
- Also need to choose a sufficiently complex hypothesis to achieve low training error.

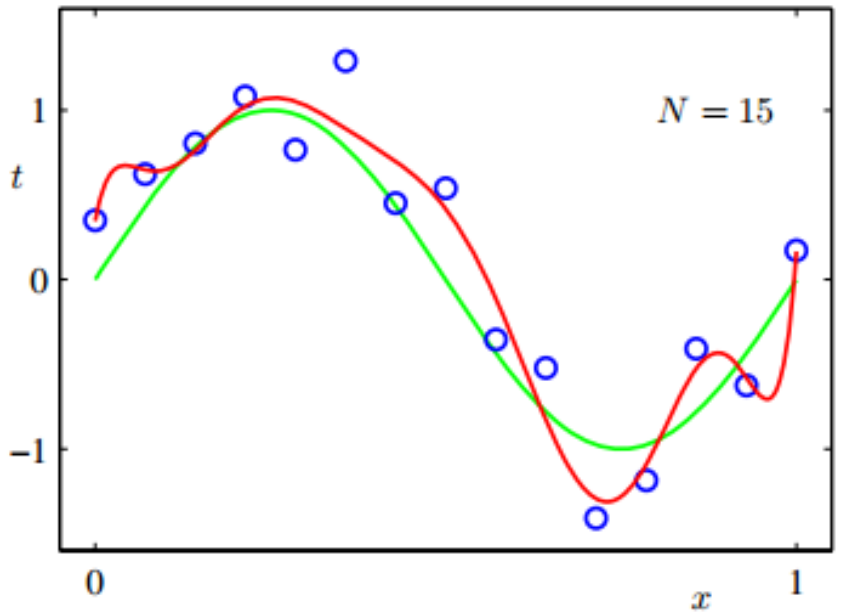
- Training error decreases until it asymptotes to the minimum possible error value as model capacity increases.
- Generalization error has a U-shaped curve as a function of model capacity.

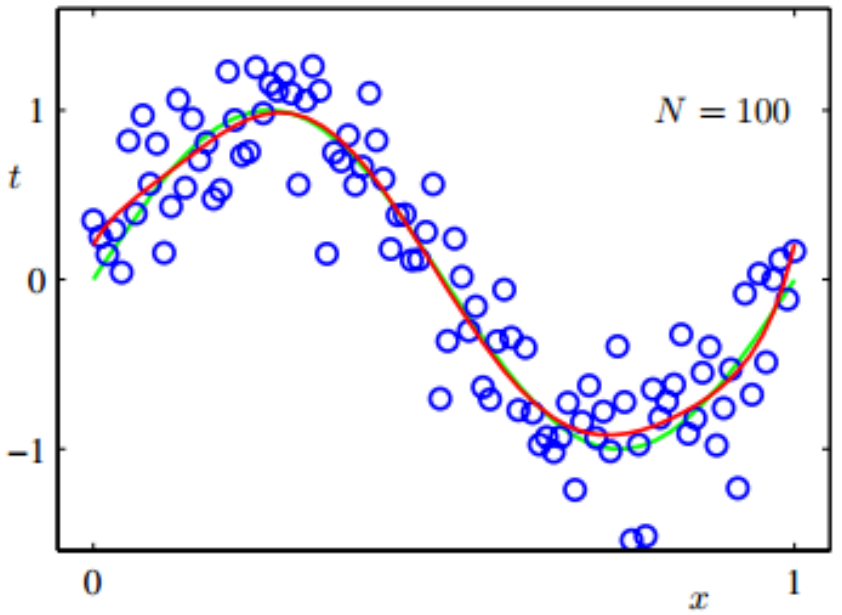
Typical relationship between capacity and error.



Underfitting & Overfitting

- Increasing the size of dataset reduces the over-fitting problem.
- The larger the data set, the more complex the model that we can afford to fit to the data.





Plots of the solutions obtained by minimizing the sum-of-squares error function using the M = 9 polynomial for N = 15 data points (left plot) and N = 100 data points (right plot).

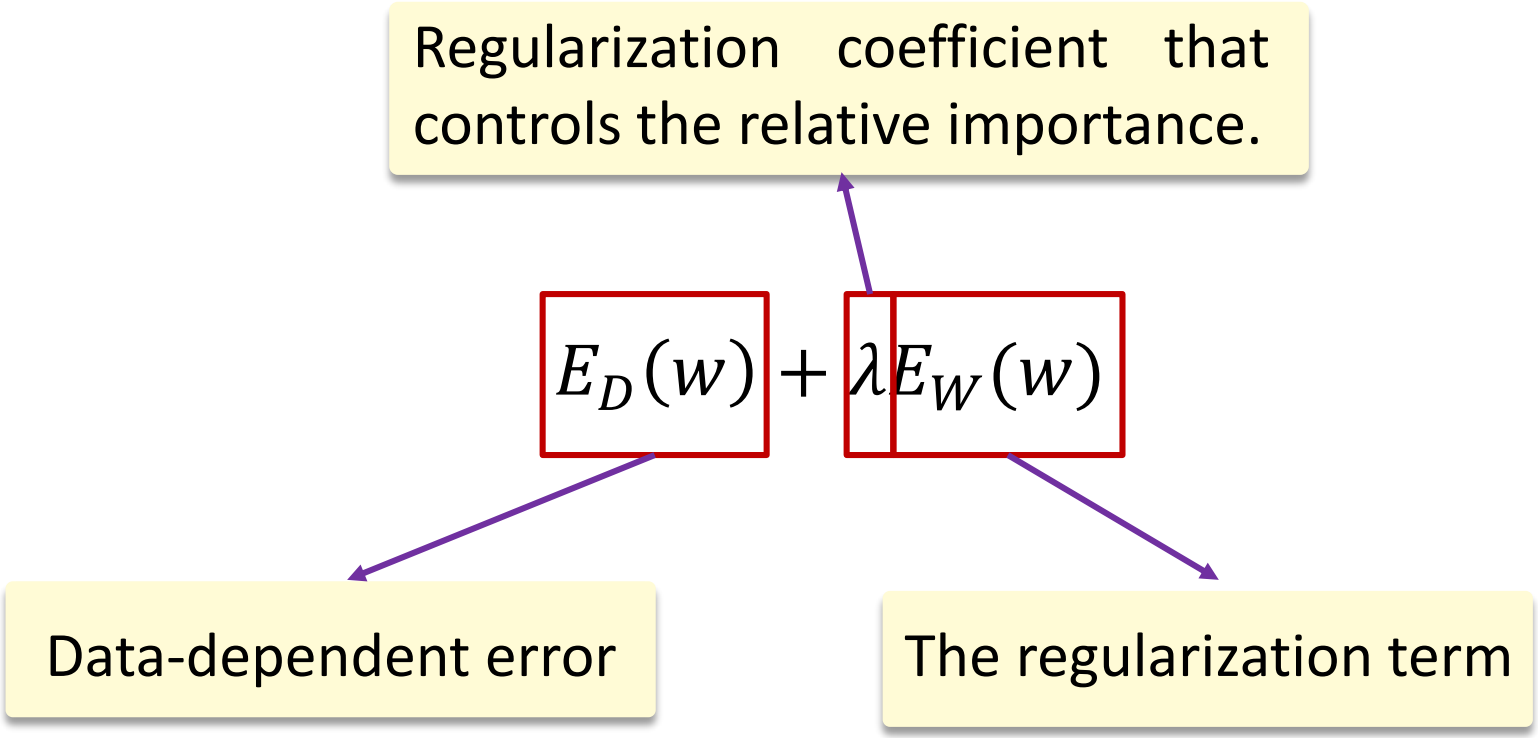
Avoid Overfitting

Adding a regularization term

$$E_D(w) + \lambda E_W(w)$$

Avoid Overfitting

Adding a regularization term



Avoid Overfitting

Adding a regularization term

Regularization coefficient that controls the relative importance.

$$E_D(w) + \lambda E_W(w)$$

Data-dependent error

The regularization term

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

We simply define
$\phi(\boldsymbol{x}) = \boldsymbol{x}$

Avoid Overfitting

Adding a regularization term

$$E_D(w) + \lambda E_W(w)$$

One simple form of regularizer

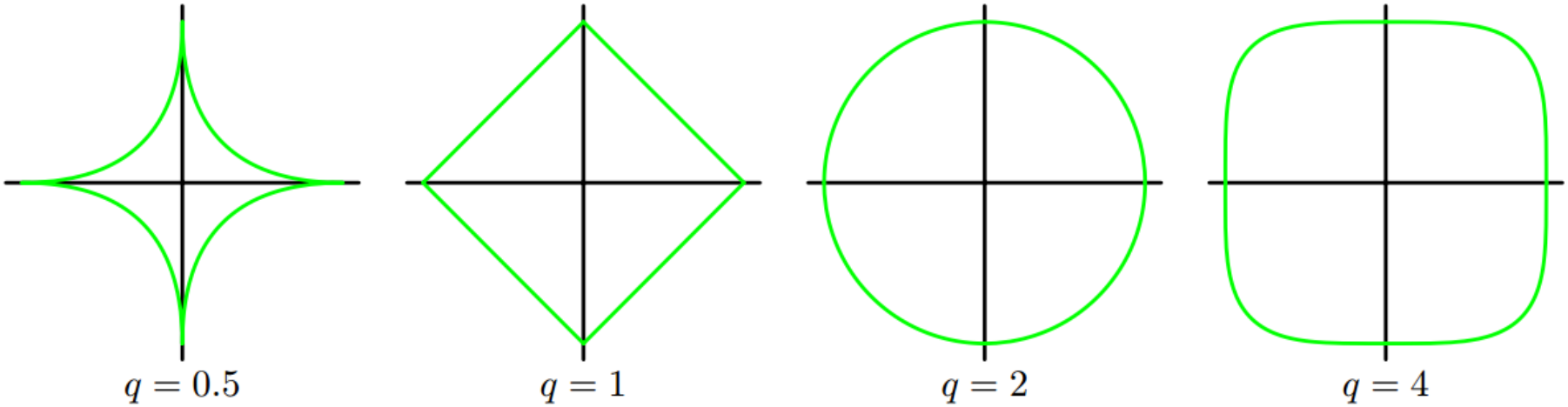
$$E_W(w) = \frac{1}{2} w^T w \qquad E_T(w) = E_D(w) + \frac{\lambda}{2} w^T w$$

weight decay: encourages weight values to decay towards zero.
parameter shrinkage: shrinks parameter values towards zero.

- Advantage： Remains a quadratic function of w, so its exact minimizer can be found in closed form.

More General Regularizer

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$$

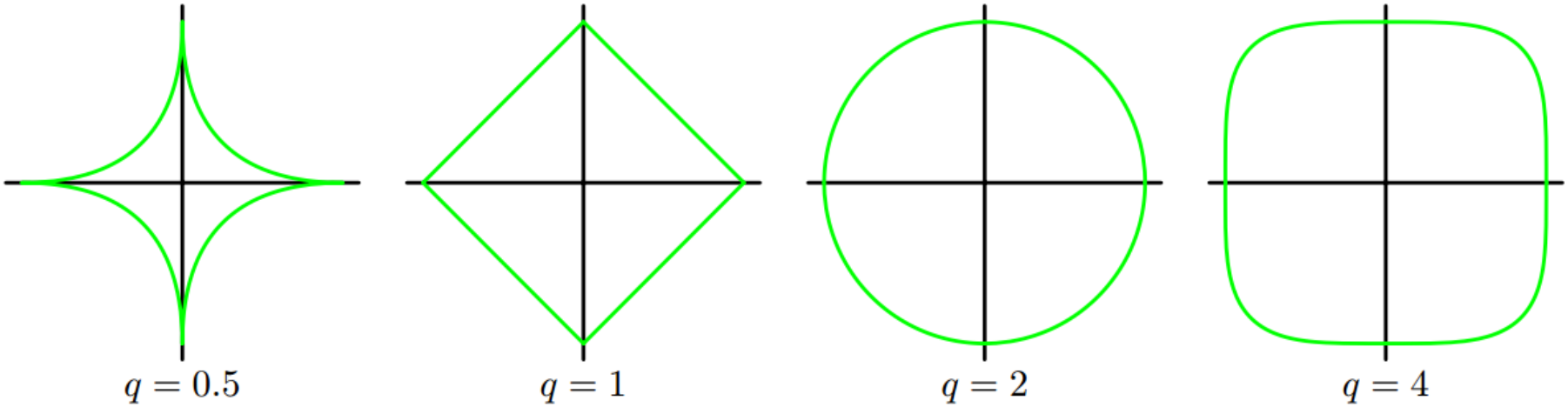


Contours of the regularization term for various values of the *q*.

- *q* = 2 corresponds to the *weight decay*.
- *q* = 1 corresponds to the *lasso*.
  - If λ is sufficiently large, some of the coefficients w_j are driven to zero, leading to a *sparse* model.

More General Regularizer

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$$



Contours of the regularization term for various values of the *q*.

- q = 2 corresponds to the *weight decay*.
- q = 1 corresponds to the *lasso*.
  - If λ is sufficiently large, some of the coefficients w_j are driven to zero, leading to a *sparse* model.

WHY?

More General Regularizer

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$$

Note that minimizing the above function is equivalent to minimizing the unregularized sum-of-squares error $E_D(w)$ subject to the constraint

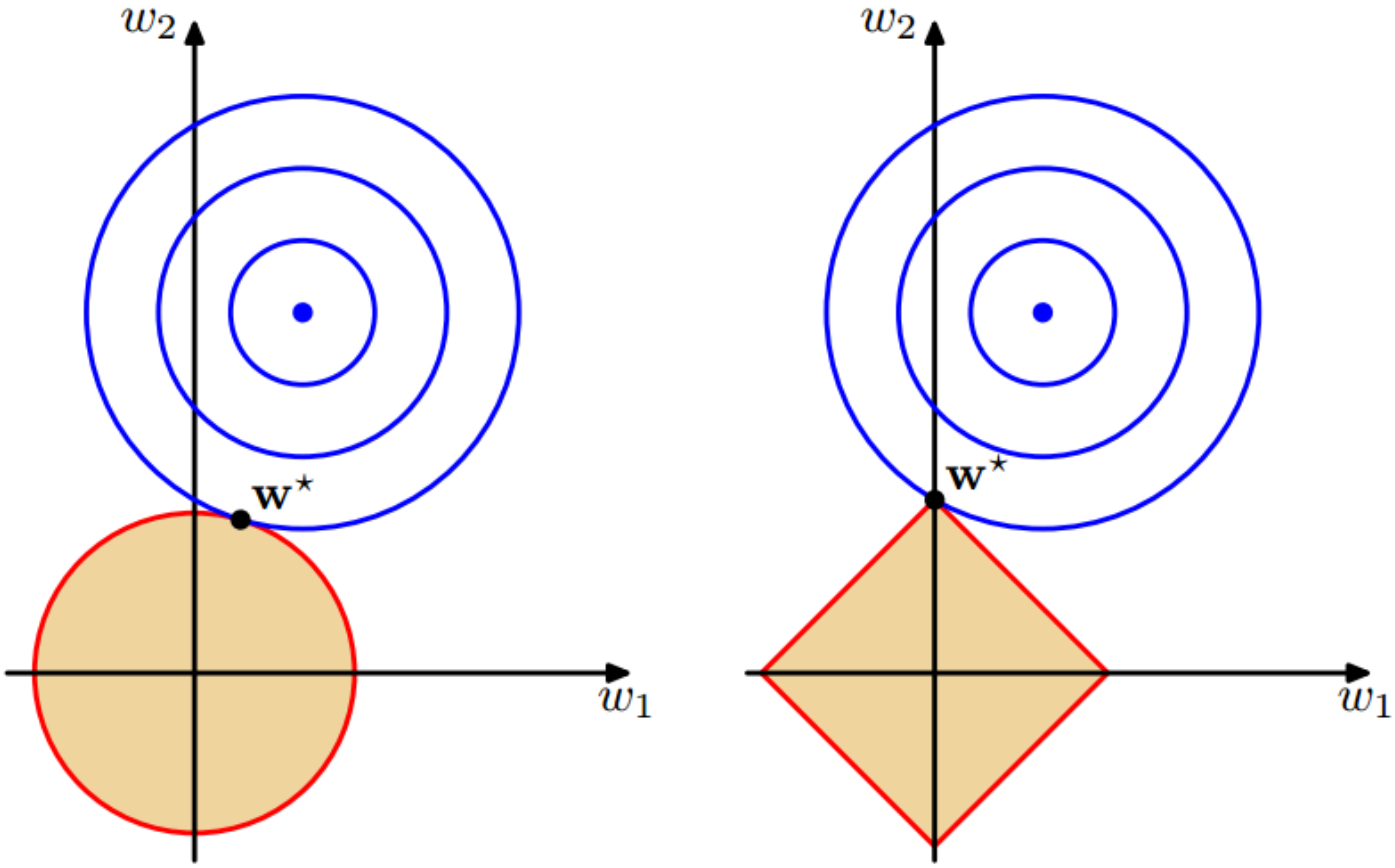
$$\sum_{j=1}^{M}|w_j|^q \leqslant \eta$$

for an appropriate value of the parameter η *(Lagrange multipliers)*.

L1 VS L2 Regularization

w^* is the optimum value for w.

The lasso gives a sparse solution ($w_1^* = 0$).

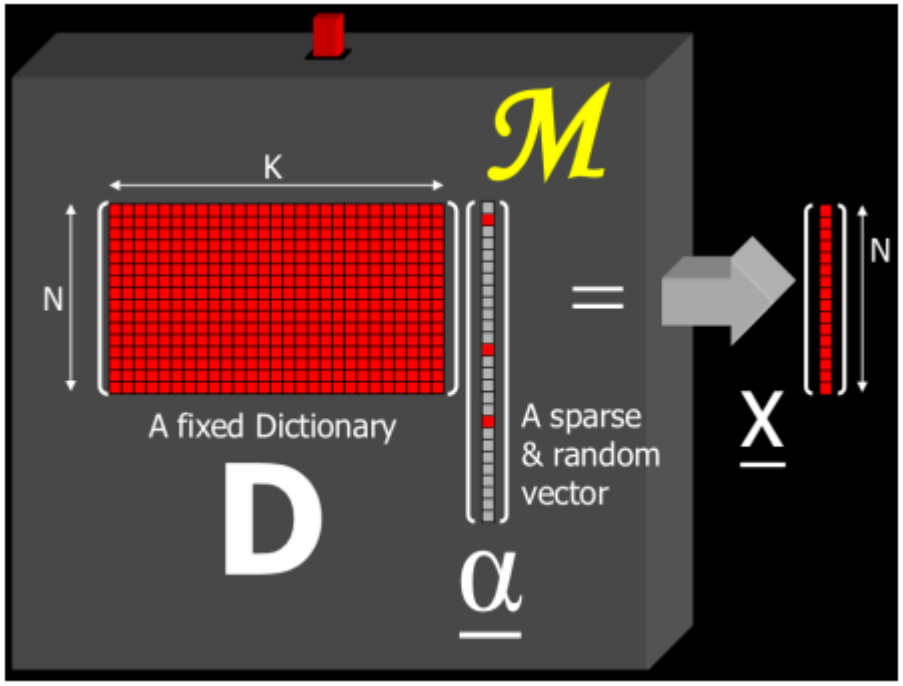


The contours of the unregularized error function (blue) along with the constraint region for the weight decay q = 2 (left) and the lasso q = 1 (right).

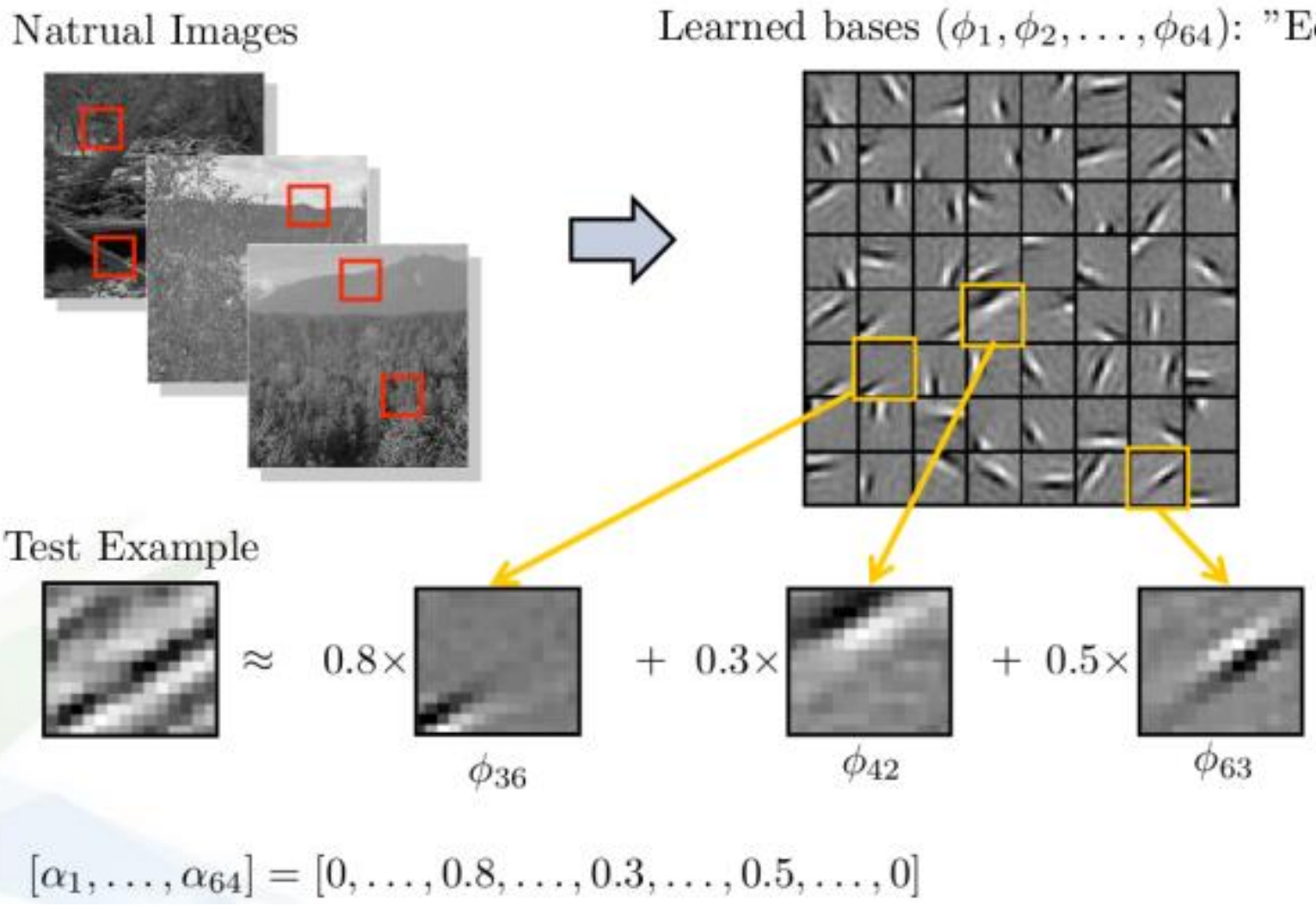
Sparse Coding

Every column of $\mathbf{D}$ is a prototype (basis).

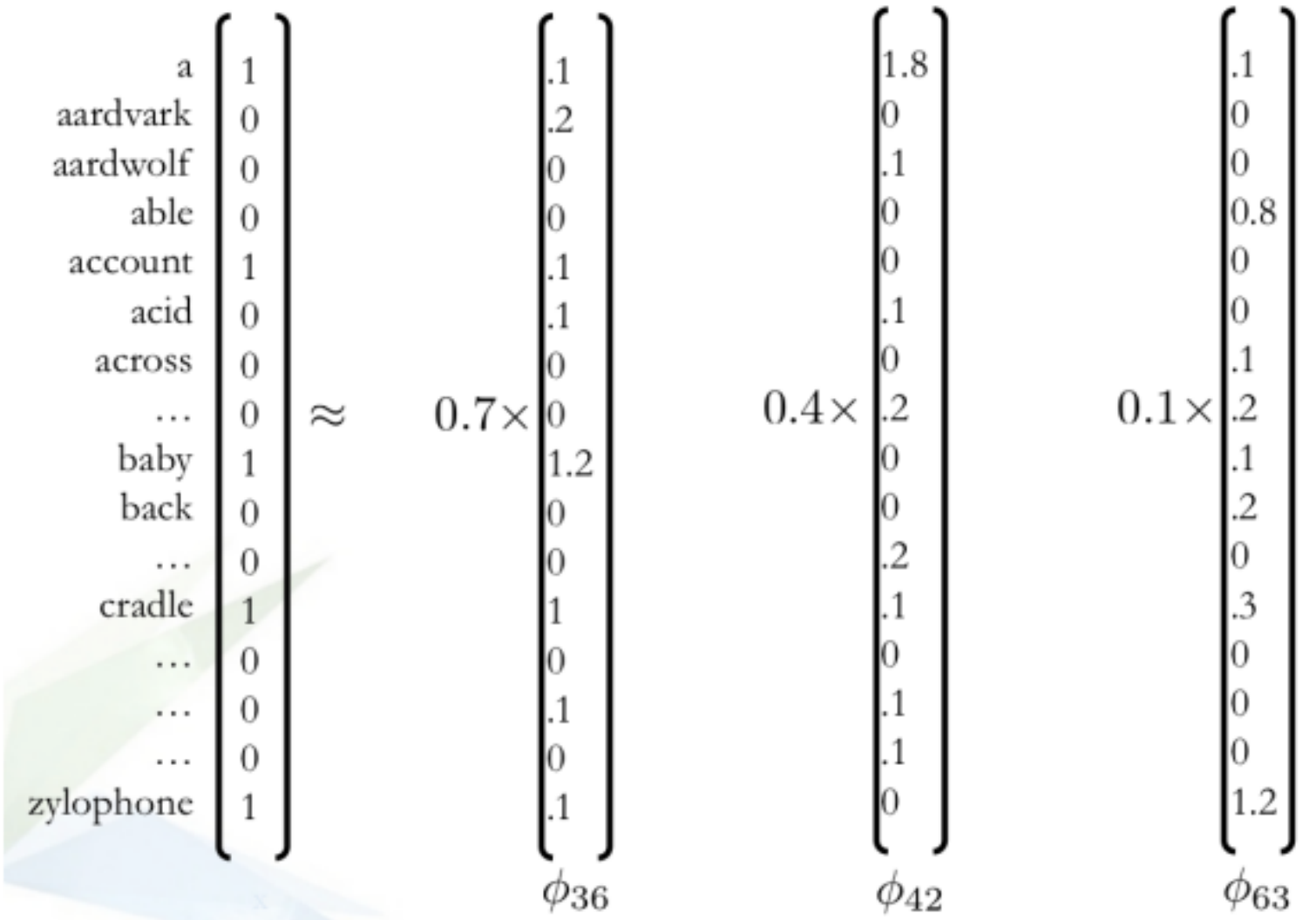
Similar to, but more general than, PCA.



Example: Sparse Coding---Images



Example: Sparse Coding---NLP



Sparse Coding

We assume our data $\boldsymbol{x}$ satisfies

$$\boldsymbol{x} = \sum_{\boldsymbol{i=0}}^{k} \alpha_i \boldsymbol{\phi}_i = \boldsymbol{\alpha D}$$

Learning:

Given training data $\boldsymbol{x}^j, j \in \{1, \ldots, m\}$

Learn dictionary $\boldsymbol{D}$ and **sparse** code $\boldsymbol{\alpha}$ (i.e., α_i are mostly zero)

Encoding:

Given test data $\boldsymbol{x}$, learned dictionary $\boldsymbol{D}$

Compute the sparse code $\boldsymbol{\alpha}$

Sparse Coding

Input: Images $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^m$ (each in $\mathbb{R}^{n \times n}$)

Learn: Dictionary of bases $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_k$ (also $\mathbb{R}^{n \times n}$).

$$\min_{\alpha, \phi} \sum_{i=1}^{m} \left(\left\| \boldsymbol{x}^i - \sum_{j=1}^{k} \alpha_j^i \boldsymbol{\phi}_j \right\|^2 + \lambda \sum_{j=1}^{k} |\alpha_j^i| \right)$$

L1 sparsity term

Causes most α_j^is to be 0.

Sparse Coding

Input: Images $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^m$ (each in $\mathbb{R}^{n \times n}$)

Learn: Dictionary of bases $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_k$ (also $\mathbb{R}^{n \times n}$).

$$\min_{\alpha, \phi} \sum_{i=1}^{m} \left(\left\| \boldsymbol{x}^i - \sum_{j=1}^{k} \alpha_j^i \boldsymbol{\phi}_j \right\|^2 + \lambda \sum_{j=1}^{k} |\alpha_j^i| \right)$$

L1 sparsity term

Causes most α_j^is to be 0.

How to optimize α and ϕ?

Sparse Coding

Input: Images $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^m$ (each in $\mathbb{R}^{n \times n}$)

Learn: Dictionary of bases $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_k$ (also $\mathbb{R}^{n \times n}$).

$$\min_{\alpha, \phi} \sum_{i=1}^{m} \left(\left\| \boldsymbol{x}^i - \sum_{j=1}^{k} \alpha_j^i \boldsymbol{\phi}_j \right\|^2 + \lambda \sum_{j=1}^{k} |\alpha_j^i| \right)$$

L1 sparsity term

Causes most α_j^is to be 0.

Aleternating minimization:

Alternately minimize with respect to $\boldsymbol{\phi}_j$'s (easy) and α_j's (harder).

Sparse Coding

Training time

Input: Images $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^m$ (each in $\mathbb{R}^{n \times n}$)
Learn: Dictionary of bases $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_k$ (also $\mathbb{R}^{n \times n}$).

$$\min_{\boldsymbol{\alpha}, \boldsymbol{\phi}} \sum_{i=1}^{m} \left(\left\| \boldsymbol{x}^i - \sum_{j=1}^{k} \alpha_j^i \boldsymbol{\phi}_j \right\|^2 + \lambda \sum_{j=1}^{k} \left| \alpha_j^i \right| \right)$$

Test time

Input: New image $\boldsymbol{x}$ (in $\mathbb{R}^{n \times n}$) and previously learned $\boldsymbol{\phi}_i$'s.
Output: Representation $[\alpha_1, \alpha_2, \ldots \alpha_n]$ of image $\boldsymbol{x}$.

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{m} \left(\left\| \boldsymbol{x} - \sum_{j=1}^{k} \alpha_j \boldsymbol{\phi}_j \right\|^2 + \lambda \sum_{j=1}^{k} \left| \alpha_j \right| \right)$$

Sparse Coding

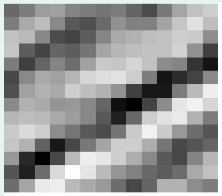
Training time

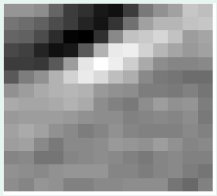
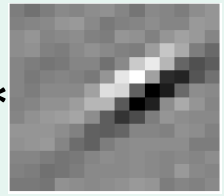
Input: Images $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^m$ (each in $\mathbb{R}^{n \times n}$)
Learn: Dictionary of bases $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_k$ (also $\mathbb{R}^{n \times n}$).

$$\min_{\boldsymbol{\alpha}, \boldsymbol{\phi}} \sum_{i=1}^{m} \left(\left\| \boldsymbol{x}^i - \sum_{j=1}^{k} \alpha_j^i \boldsymbol{\phi}_j \right\|^2 + \lambda \sum_{j=1}^{k} \left| \alpha_j^i \right| \right)$$

Test time

Input: New image $\boldsymbol{x}$ (in $\mathbb{R}^{n \times n}$) and previously learned $\boldsymbol{\phi}_i$'s.
Output: Representation $[\alpha_1, \alpha_2, \ldots \alpha_n]$ of image $\boldsymbol{x}$.


$\approx 0.8 *$

$+ 0.3 *$

$+ 0.5 *$


$\boldsymbol{x} \approx 0.8 * \boldsymbol{\phi}_{36} + 0.3 * \boldsymbol{\phi}_{42} + 0.5 * \boldsymbol{\phi}_{63}$

Represent as: [0, 0, …, 0, **0.8**, 0, …, 0, **0.3**, 0, …, 0, **0.5**, …]

Optimization with L1 Norm

A simple form of the problem with L1 norm is defined as follows,

$$\boldsymbol{\alpha}^* = \min_{\boldsymbol{\alpha}} h(\boldsymbol{\alpha}) = \lambda \|\boldsymbol{\alpha}\|_1 + \frac{1}{2}\|\boldsymbol{\alpha} - \boldsymbol{s}\|^2$$

$$= \sum_{j=1}^{d} \lambda |\alpha_j| + \sum_{j=1}^{d} \frac{1}{2}(\alpha_j - s_j)^2$$

where $\boldsymbol{\alpha}^*$ is the optimal solution. We can conclude that,

(1) If $\alpha_j > 0$, then $h(\alpha_j) = \lambda\alpha_j + \frac{1}{2}(\alpha_j - s_j)^2$ and its derivative is $h'(\alpha_j) = \lambda + \alpha_j^* - s_j = 0 \Rightarrow \alpha_j^* = s_j - \lambda$,which indicates $s_j > \lambda$;

Optimization with L1 Norm

A simple form of the problem with L1 norm is defined as follows,

$$\boldsymbol{\alpha}^* = \min_{\boldsymbol{\alpha}} h(\boldsymbol{\alpha}) = \lambda \|\boldsymbol{\alpha}\|_1 + \frac{1}{2}\|\boldsymbol{\alpha} - \boldsymbol{s}\|^2$$

$$= \sum_{j=1}^{d} \lambda |\alpha_j| + \sum_{j=1}^{d} \frac{1}{2}(\alpha_j - s_j)^2$$

where $\boldsymbol{\alpha}^*$ is the optimal solution. We can conclude that,

(1) If $\alpha_j > 0$, then $h(\alpha_j) = \lambda\alpha_j + \frac{1}{2}(\alpha_j - s_j)^2$ and its derivative is $h'(\alpha_j) = \lambda + \alpha_j^* - s_j = 0 \Rightarrow \alpha_j^* = s_j - \lambda$,which indicates $s_j > \lambda$;

(2) If $\alpha_j < 0$, then $h(\alpha_j) = -\lambda\alpha_j + \frac{1}{2}(\alpha_j - s_j)^2$ and its derivative is $h'(\alpha_j) = -\lambda + \alpha_j^* - s_j = 0 \Rightarrow \alpha_j^* = s_j + \lambda$, which indicates $s_j < -\lambda$;

Optimization with L1 Norm

A simple form of the problem with L1 norm is defined as follows,

$$\boldsymbol{\alpha}^* = \min_{\boldsymbol{\alpha}} h(\boldsymbol{\alpha}) = \lambda \|\boldsymbol{\alpha}\|_1 + \frac{1}{2}\|\boldsymbol{\alpha} - \boldsymbol{s}\|^2$$

$$= \sum_{j=1}^{d} \lambda |\alpha_j| + \sum_{j=1}^{d} \frac{1}{2}(\alpha_j - s_j)^2$$

where $\boldsymbol{\alpha}^*$ is the optimal solution. We can conclude that,

(1) If $\alpha_j > 0$, then $h(\alpha_j) = \lambda\alpha_j + \frac{1}{2}(\alpha_j - s_j)^2$ and its derivative is $h'(\alpha_j) = \lambda + \alpha_j^* - s_j = 0 \Rightarrow \alpha_j^* = s_j - \lambda$,which indicates $s_j > \lambda$;

(2) If $\alpha_j < 0$, then $h(\alpha_j) = -\lambda\alpha_j + \frac{1}{2}(\alpha_j - s_j)^2$ and its derivative is $h'(\alpha_j) = -\lambda + \alpha_j^* - s_j = 0 \Rightarrow \alpha_j^* = s_j + \lambda$, which indicates $s_j < -\lambda$;

(3) If$-\lambda \leq s_j \leq \lambda$, and then $\alpha_j^* = 0$.

Shrink Operator

$\boldsymbol{\alpha}^*$ can be summarized as follows,

$$\alpha_j^* = \begin{cases} s_j - \lambda, & \text{if } s_j > \lambda \\ s_j + \lambda, & \text{if } s_j < -\lambda \\ 0 & \text{otherwise} \end{cases}$$

The equivalent expression is $\boldsymbol{\alpha}^* = shrink(\boldsymbol{s}, \lambda)$, where the j-th component of $shrink(\boldsymbol{s}, \lambda)$,

$$shrink(s, \lambda)_j = sign(s_j)\max\{|s_j| - \lambda, 0\}$$

ISTA (Iterative Shrinkage-Thresholding Algorithms)

The objective function of ISTA has the form of

$$arg\min F(\boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1 = f(\boldsymbol{\alpha}) + g(\boldsymbol{\alpha})$$

which is usually difficult to solve. We convert it to the easy one.

First, the second order Taylor expansion is used to approximate $f(\boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{y}\|_2^2$ at point $\boldsymbol{\alpha}^{k-1}$:

$$f(\boldsymbol{\alpha}) = f\left(\boldsymbol{\alpha}^{k-1}\right) + \left(\boldsymbol{\alpha} - \boldsymbol{\alpha}^{k-1}\right)^T \nabla f\left(\boldsymbol{\alpha}^{k-1}\right) + \frac{1}{2}\left(\boldsymbol{\alpha} - \boldsymbol{\alpha}^{k-1}\right)^T H_f\left(\boldsymbol{\alpha}^{k-1}\right)\left(\boldsymbol{\alpha} - \boldsymbol{\alpha}^{k-1}\right)$$

Hessian matrix of $f(\boldsymbol{\alpha})$ at $\boldsymbol{\alpha}^{k-1}$

ISTA

A real-valued function $f : \mathrm{R} \rightarrow \mathrm{R}$ is called Lipschitz continuous if there exists a positive real constant L such that, for all real x_1 and x_2,

$$|f(x_1) - f(x_2)| \leq L|x_1 - x_2| \qquad L\text{: Lipschitz constant.}$$

Simple algebra shows that given any $\boldsymbol{\alpha}^{k-1}$ and $\boldsymbol{\alpha}$, we always have,

$$f(\boldsymbol{\alpha}) \leq f\left(\boldsymbol{\alpha}^{k-1}\right) + \left(\boldsymbol{\alpha} - \boldsymbol{\alpha}^{k-1}\right)^T \nabla f\left(\boldsymbol{\alpha}^{k-1}\right) + \frac{L}{2}\left\|\boldsymbol{\alpha} - \boldsymbol{\alpha}^{k-1}\right\|^2$$

We thus approximate $\arg\min_{\boldsymbol{\alpha}} F(\boldsymbol{\alpha}) = \arg\min_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) + g(\boldsymbol{\alpha})$ with

$$\arg\min_{\boldsymbol{\alpha}} \frac{L}{2}\left\|\boldsymbol{\alpha} - \left(\boldsymbol{\alpha}^{k-1} - \frac{1}{L}\nabla f\left(\boldsymbol{\alpha}^{k-1}\right)\right)\right\|^2 + g(\boldsymbol{\alpha})$$

ISTA

To optimize $F(\boldsymbol{\alpha}) = f(\boldsymbol{\alpha}) + g(\boldsymbol{\alpha})$, we adopt the following approximation model, and define

$$Q_L(\boldsymbol{\alpha}, \boldsymbol{\alpha}^{k-1}) = f(\boldsymbol{\alpha}^{k-1}) + (\boldsymbol{\alpha} - \boldsymbol{\alpha}^{k-1})^T \nabla f(\boldsymbol{\alpha}^{k-1}) + \frac{L}{2} \|\boldsymbol{\alpha} - \boldsymbol{\alpha}^{k-1}\|^2 + g(\boldsymbol{\alpha})$$

which admits a unique minimizer,

$$P_L(\boldsymbol{\alpha}^{k-1}) = \arg\min_{\boldsymbol{\alpha}} \{Q_L(\boldsymbol{\alpha}, \boldsymbol{\alpha}^{k-1})\}$$

The basic step of ISTA for the above problem thus reduces to

$$\boldsymbol{\alpha}^k = P_L(\boldsymbol{\alpha}^{k-1})$$

ISTA

The objective function of ISTA has the form of

$$arg\min F(\boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{X\alpha} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1 = f(\boldsymbol{\alpha}) + g(\boldsymbol{\alpha})$$

ISTA with constant stepsize

Input: L: Lipschitz constant of ∇f.

Step 0: Take $\alpha^0 \in \mathbb{R}^n$

Step k: $(k \geq 1)$ Compute

$$\boldsymbol{\alpha}^k = P_L\left(\boldsymbol{\alpha}^{k-1}\right)$$

ISTA

Recall the objective function of ISTA,

$$arg\ \min F(\boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{X\alpha} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1 = f(\boldsymbol{\alpha}) + \lambda g(\boldsymbol{\alpha})$$

For the function $f(\boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{X\alpha} - \boldsymbol{y}\|^2$, we have $\nabla f(\boldsymbol{\alpha}) = \boldsymbol{X}^T(\boldsymbol{X\alpha} - \boldsymbol{y})$

Then we have

$$Q_L(\boldsymbol{\alpha}, \boldsymbol{\alpha}^{k-1}) = \frac{L}{2}\left\|\boldsymbol{\alpha} - \left(\boldsymbol{\alpha}^{k-1} - \frac{1}{L}\boldsymbol{X}^T(\boldsymbol{X\alpha}^{k-1} - \boldsymbol{y})\right)\right\|^2 + \lambda\|\boldsymbol{\alpha}\|_1$$

As a result, we have the equivalent problem,

$$\boldsymbol{\alpha}^k = P_L(\boldsymbol{\alpha}^{k-1}) = arg\min\frac{L}{2}\left\|\boldsymbol{\alpha} - \theta(\boldsymbol{\alpha}^{k-1})\right\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1$$

$$\theta(\boldsymbol{\alpha}^{k-1}) = \boldsymbol{\alpha}^{k-1} - \frac{1}{L}\boldsymbol{X}^T(\boldsymbol{X\alpha}^{k-1} - \boldsymbol{y})$$

ISTA

$$\boldsymbol{\alpha}^k = P_L\left(\boldsymbol{\alpha}^{k-1}\right) = arg\min \frac{L}{2}\left\|\boldsymbol{\alpha} - \theta(\boldsymbol{\alpha}^{k-1})\right\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1$$

$$\theta\left(\boldsymbol{\alpha}^{k-1}\right) = \boldsymbol{\alpha}^{k-1} - \frac{1}{L}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\alpha}^{k-1} - \boldsymbol{y})$$

Remember that for the problem with L1 norm,

$$\boldsymbol{\alpha}^* = \min_{\boldsymbol{\alpha}} h(\boldsymbol{\alpha}) = \lambda\|\boldsymbol{\alpha}\|_1 + \frac{1}{2}\|\boldsymbol{\alpha} - \boldsymbol{s}\|^2$$

We have the solution as $\boldsymbol{\alpha}^* = shrink(\boldsymbol{s}, \lambda)$, where $shrink(\boldsymbol{s}, \lambda)_j = sign\left(\boldsymbol{s}_j\right)\max\left\{\left|\boldsymbol{s}_j\right| - \lambda, 0\right\}$.

Therefore, the solution of $\boldsymbol{\alpha}^k = arg\min \frac{L}{2}\left\|\boldsymbol{\alpha} - \theta(\boldsymbol{\alpha}^{k-1})\right\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1$

Is?

ISTA

$$\boldsymbol{\alpha}^k = P_L\left(\boldsymbol{\alpha}^{k-1}\right) = arg\min \frac{L}{2}\left\|\boldsymbol{\alpha} - \theta(\boldsymbol{\alpha}^{k-1})\right\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1$$

$$\theta\left(\boldsymbol{\alpha}^{k-1}\right) = \boldsymbol{\alpha}^{k-1} - \frac{1}{L}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\alpha}^{k-1} - \boldsymbol{y})$$

Remember that for the problem with L1 norm,

$$\boldsymbol{\alpha}^* = \min_{\boldsymbol{\alpha}} h(\boldsymbol{\alpha}) = \lambda\|\boldsymbol{\alpha}\|_1 + \frac{1}{2}\|\boldsymbol{\alpha} - \boldsymbol{s}\|^2$$

We have the solution as $\boldsymbol{\alpha}^* = shrink(\boldsymbol{s}, \lambda)$, where

$shrink(\boldsymbol{s}, \lambda)_j = sign\left(\boldsymbol{s}_j\right)\max\left\{\left|\boldsymbol{s}_j\right| - \lambda, 0\right\}$.

Therefore, the solution of $\boldsymbol{\alpha}^k = arg\min \frac{L}{2}\left\|\boldsymbol{\alpha} - \theta(\boldsymbol{\alpha}^{k-1})\right\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1$

is $\boldsymbol{\alpha}^k = shrink\left(\theta\left(\boldsymbol{\alpha}^{k-1}\right), \lambda/L\right)$.

ISTA with backtracking

Drawback: the Lipschitz constant L is not always easily computable (for large-scale problems).

ISTA with backtracking

Step 0. Take $L_0 > 0$, some $\eta > 1$, and $\alpha^0 \in \mathbb{R}^n$

Step k. ($k \geq 1$) Find the smallest nonnegative integers i_k such that with $\bar{L} = \eta^{i_k} L_{k-1}$

$$F\left(P_{\bar{L}}\left(\boldsymbol{\alpha}^{k-1}\right)\right) \leq Q_{\bar{L}}\left(P_{\bar{L}}\left(\boldsymbol{\alpha}^{k-1}\right), \boldsymbol{\alpha}^{k-1}\right)$$

Set $L_k = \eta^{i_k} L_{k-1}$ and compute

$$\boldsymbol{\alpha}^k = P_{L_k}\left(\boldsymbol{\alpha}^{k-1}\right)$$

$$P_L\left(\boldsymbol{\alpha}^{k-1}\right) = \arg\min_{\boldsymbol{\alpha}}\{Q_L\left(\boldsymbol{\alpha}, \boldsymbol{\alpha}^{k-1}\right)\} \qquad \boldsymbol{\alpha}^k = P_L\left(\boldsymbol{\alpha}^{k-1}\right)$$

FISTA (Fast ISTA)

Main Difference: The $P_L(\cdot)$ is not employed on the previous point $\boldsymbol{\alpha}^{k-1}$, but rather at the point $\boldsymbol{y}^k$ which uses a very specific linear combination of the previous two points $\{\boldsymbol{\alpha}^{k-1}, \boldsymbol{\alpha}^{k-2}\}$.

FISTA with constant stepsize

Input: L: Lipschitz constant of of ∇f.

Step 0. Take $\boldsymbol{y}^1 = \boldsymbol{\alpha}^0 \in \mathbb{R}^n, t_1 = 1$

Step k. ($k \geq 1$) Compute

$$\boldsymbol{\alpha}^k = P_L\left(\boldsymbol{y}^k\right)$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{\alpha}^k + \left(\frac{t_k - 1}{t_{k+1}}\right)\left(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}\right)$$

FISTA

FISTA with backtracking

Step 0. Take $L_0 > 0$, some $\eta > 1$, and $\boldsymbol{y}^1 = \boldsymbol{\alpha}^0 \in \mathbb{R}^n,\ t_1 = 1$

Step k. ($k \geq 1$) Find the smallest nonnegative integers i_k such that with $\bar{L} = \eta^{i_k} L_{k-1}$

$$F\left(P_{\bar{L}}\left(\boldsymbol{y}^k\right)\right) \leq Q_{\bar{L}}\left(P_{\bar{L}}\left(\boldsymbol{y}^k\right), \boldsymbol{y}^k\right)$$

Set $L_k = \eta^{i_k} L_{k-1}$ and compute

$$\boldsymbol{\alpha}^k = P_L\left(\boldsymbol{y}^k\right)$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{\alpha}^k + \left(\frac{t_k - 1}{t_{k+1}}\right)\left(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}\right)$$

Conclusions

Potential benefits

- To improve accuracy
- Reduce computation
- Reduce space
- Reduce cost of future measurements
- Improved data/model understanding

Conclusions

- **Wrappers methods**
  - Use machine learning algorithm as black box to find best subset of features.
  - Generally infeasible on the model ‘big data’ problem.
- **Filters methods**
  - Features selected before machine learning algorithm is run.
- **Embedded methods**
  - Feature selection occurs naturally as part of the machine learning algorithm.

Conclusions

- **Overfitting and Underfitting**
- **L1-norm and L2-norm**
- **Sparse Coding**
- **Shrink Operator**
- **ISTA and FISTA (optional)**