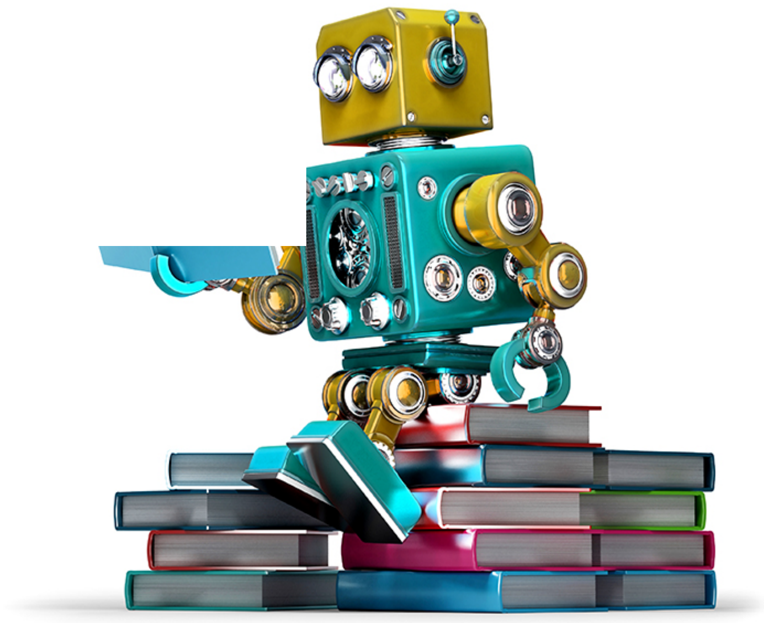


Machine Learning

Lecture 3

NUACA

2017

Vector Norms

Vector norms allow us to talk about the length of vectors

- The L^p norm of $\mathbf{v} = (v_1, \ldots, v_D) \in \mathbb{R}^D$ is given by

$$\|\mathbf{v}\|_p = \left(\sum_{1 \leq i \leq D} |v_i|^p \right)^{1/p}$$

- Properties of L^p (which actually hold for any norm):
  - $\|\mathbf{v}\|_p = 0$ implies $\mathbf{v} = \mathbf{0}$
  - $\|\mathbf{v} + \mathbf{w}\|_p \leq \|\mathbf{v}\|_p + \|\mathbf{w}\|_p$
  - $\|r \cdot \mathbf{v}\|_p = |r| \cdot \|\mathbf{v}\|_p$ for all $r \in \mathbb{R}$

- Popular norms:
  - Manhattan norm L^1
  - Eucledian norm L^2
  - Maximum norm L^∞ where $\|\mathbf{v}\|_\infty = \max_{1 \leq i \leq D} |v_i|$

Literature

- Goodfellow, Bengio, Courville: Deep Learning (2016) https://www.deeplearningbook.org (Chapter 5.2-5.4)
- Murphy: Machine Learning: A Probabilistic Prospective (2012) – Download here - Chap. 7.5

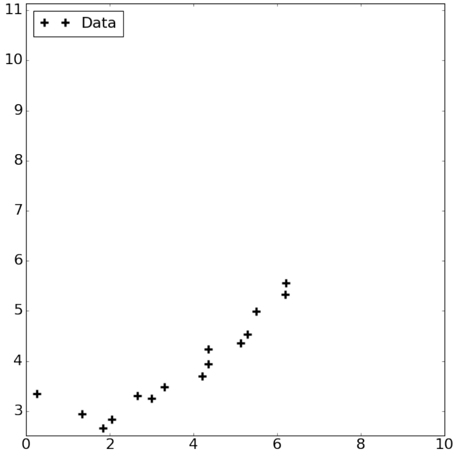
Outline

Basis Function Expansion

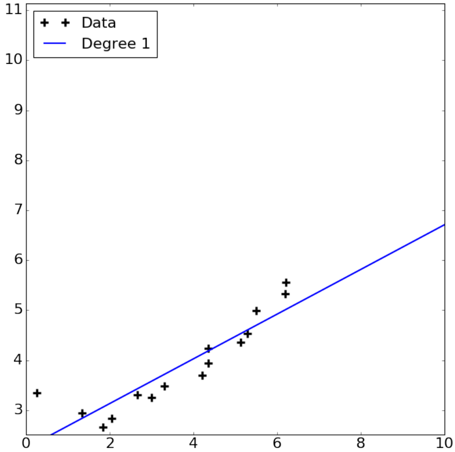
Overfitting and the Bias-Variance Tradeoff

Sources of Overfitting

Linear Regression : Polynomial Basis Expansion



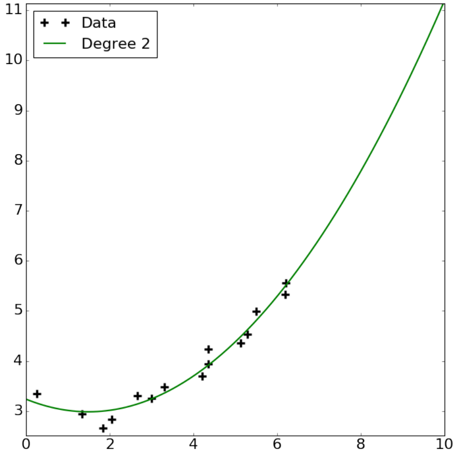
Linear Regression : Polynomial Basis Expansion



Linear Regression : Polynomial Basis Expansion

$$\phi(x) = [1, x, x^2]$$

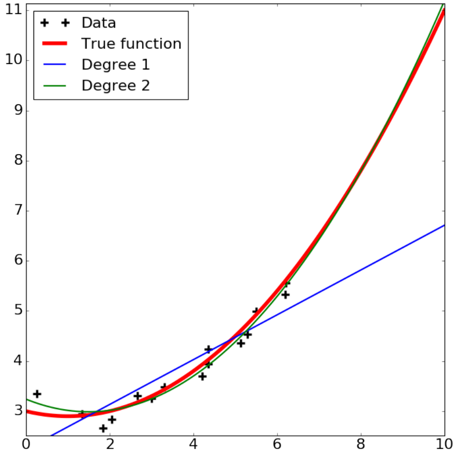
$$w_0 + w_1 x + w_2 x^2 = \phi(x) \cdot [w_0, w_1, w_2]$$



Linear Regression : Polynomial Basis Expansion

$$\boldsymbol{\phi}(x) = [1, x, x^2]$$

$$w_0 + w_1 x + w_2 x^2 = \boldsymbol{\phi}(x) \cdot [w_0, w_1, w_2]$$

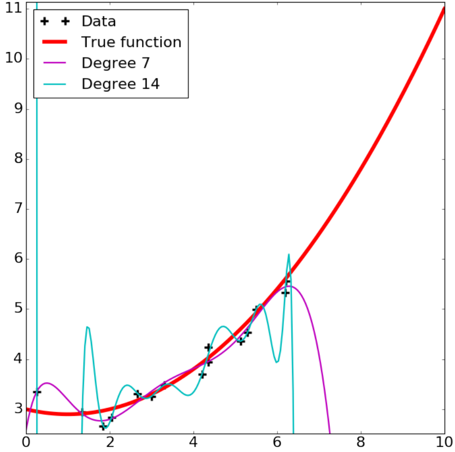


Linear Regression : Polynomial Basis Expansion

$\boldsymbol{\phi}(x) = [1, x, x^2, \cdots, x^d]$

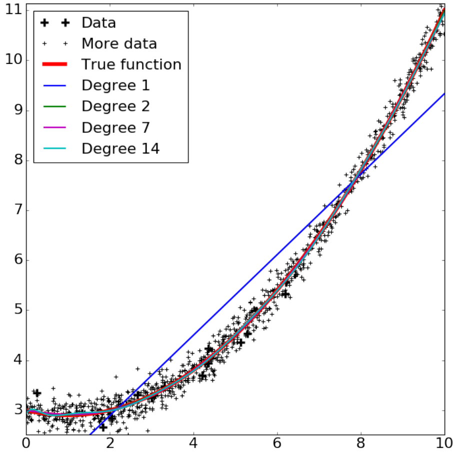
Model $y = \mathbf{w}^\mathsf{T}\boldsymbol{\phi}(x) + \epsilon$

Here $\mathbf{w} \in \mathbb{R}^M$, where M is the number for expanded features



Linear Regression : Polynomial Basis Expansion

Getting more data can avoid overfitting!



Polynomial Basis Expansion in Higher Dimensions

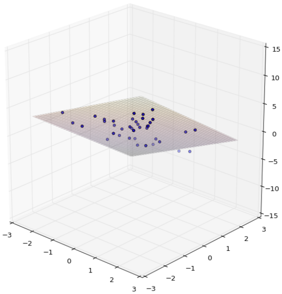
Basis expansion can be performed in higher dimensions

We're still fitting linear models, but using more features

$$y = \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}) + \epsilon$$

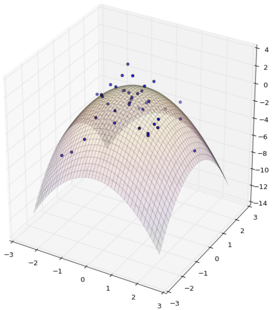
Linear Model

$$\phi(\mathbf{x}) = [1, x_1, x_2]$$



Quadratic Model

$$\phi(\mathbf{x}) = [1, x_1, x_2, x_1^2, x_2^2, x_1 x_2]$$



Using degree d polynomials in D dimensions results in $\approx D^d$ features!

Outline

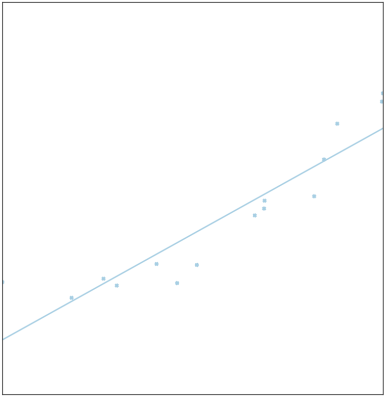
Basis Function Expansion

Overfitting and the Bias-Variance Tradeoff

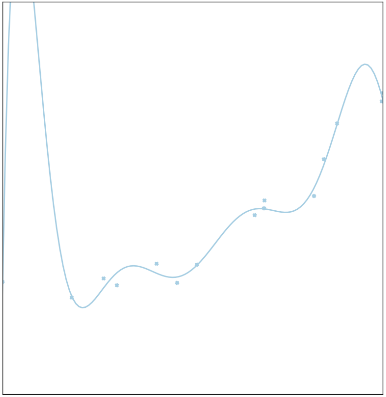
Sources of Overfitting

The Bias Variance Tradeoff

High Bias

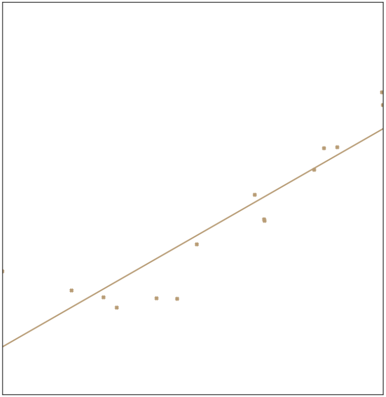


High Variance

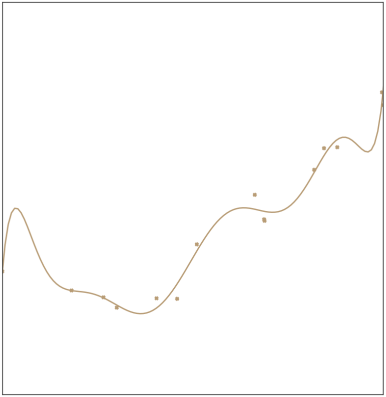


The Bias Variance Tradeoff

High Bias

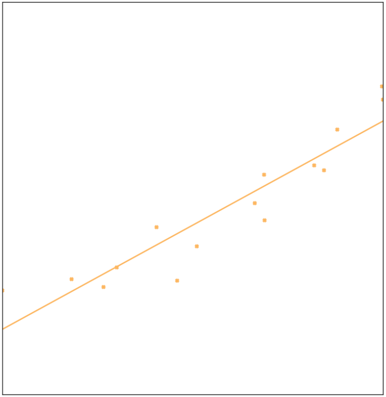


High Variance

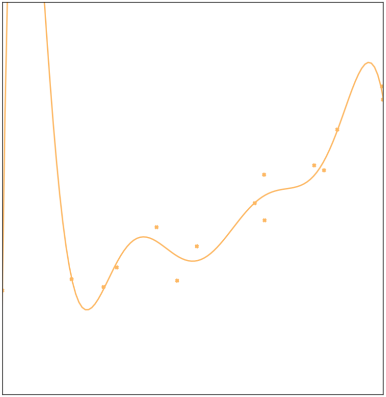


The Bias Variance Tradeoff

High Bias

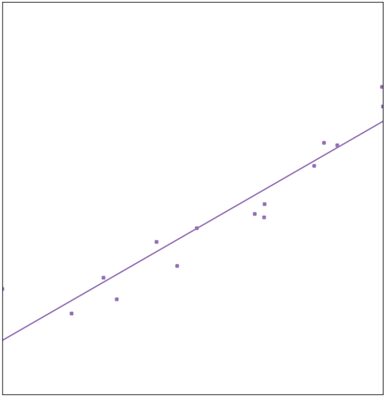


High Variance

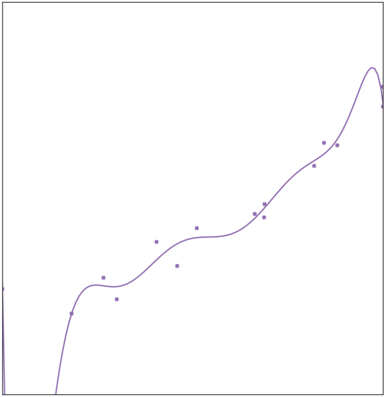


The Bias Variance Tradeoff

High Bias

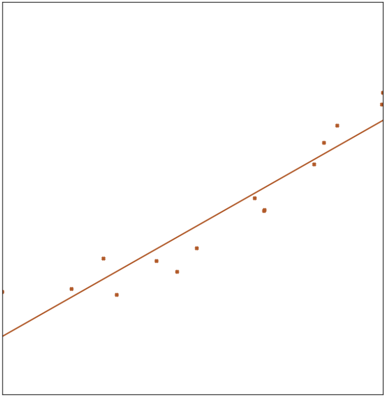


High Variance

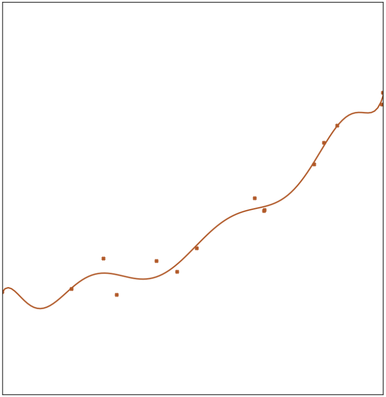


The Bias Variance Tradeoff

High Bias

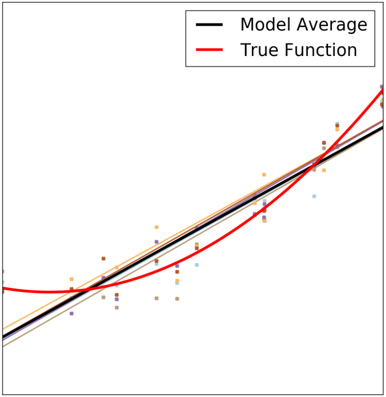


High Variance

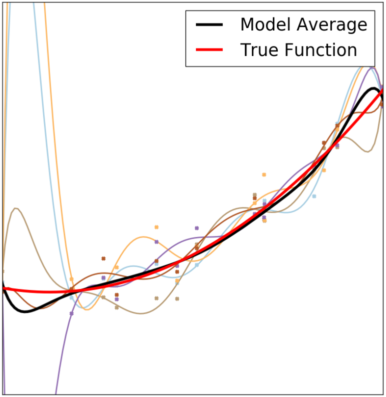


The Bias Variance Tradeoff

High Bias



High Variance



The Bias Variance Tradeoff

- Having high bias means that we are underfitting
- Having high variance means that we are overfitting
- The terms bias and variance in this context are precisely defined statistical notions

- See Sec. 5.4 in the GBC book for a much more detailed description

Learning Curves

Suppose we've trained a model and used it to make predictions

But in reality, the predictions are often poor

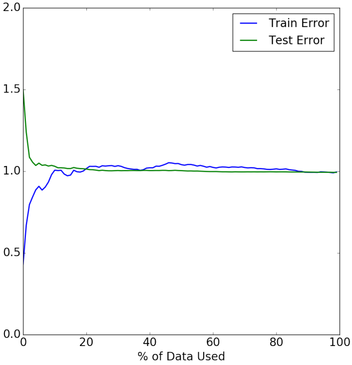
- How can we know whether we have high bias (underfitting) or high variance (overfitting) or neither?
  - Should we add more features (higher degree polynomials, lower width kernels, etc.) to make the model more expressive?
  - Should we simplify the model (lower degree polynomials, larger width kernels, etc.) to reduce the number of parameters?
- Should we try and obtain more data?
  - Often there is a computational and monetary cost to using more data

Learning Curves

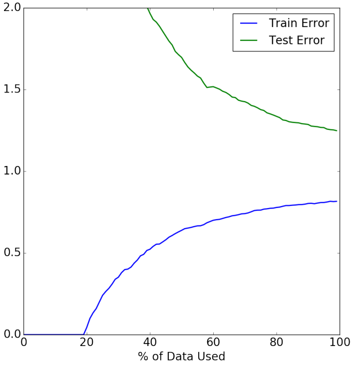
Split the data into a training set and testing set

Train on increasing sizes of data

Plot the training error and test error as a function of training data size



More data is not useful



More data would be useful

Outline

Ridge Regression and Lasso

Model Selection

Ridge Regression

Suppose we have data $\langle(\mathbf{x}_i, y_i)\rangle_{i=1}^{N}$, where $\mathbf{x} \in \mathbb{R}^D$ with $D \gg N$

One idea to avoid overfitting is to add a penalty term for weights

Least Squares Estimate Objective

$$\mathcal{L}(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^{\mathsf{T}}(\mathbf{X}\mathbf{w} - \mathbf{y})$$

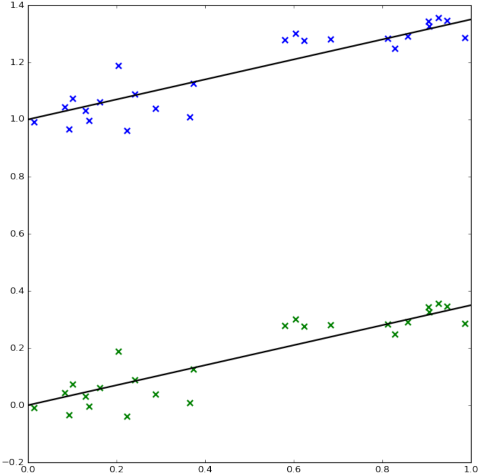
Ridge Regression Objective

$$\mathcal{L}_{\text{ridge}}(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^{\mathsf{T}}(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \sum_{i=1}^{D} w_i^2$$

Ridge Regression

We add a penalty term for weights to control model complexity

Should not penalise the constant term w_0 for being large



Ridge Regression

Should translating and scaling inputs contribute to model complexity?

Suppose $\widehat{y} = w_0 + w_1 x$

Supose x is temperature in $^\circ C$ and x' in $^\circ F$

So $\widehat{y} = \left(w_0 - \frac{160}{9} w_1\right) + \frac{5}{9} w_1 x'$

In one case "model complexity" is w_1^2, in the other it is $\frac{25}{81} w_1^2 < \frac{w_1^2}{3}$

Should try and avoid dependence on scaling and translation of variables

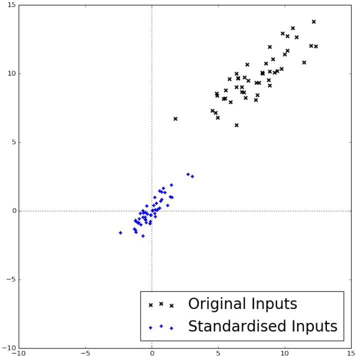
Ridge Regression

Before optimising the ridge objective, it's a good idea to standardise all inputs (mean 0 and variance 1)

If in addition, we center the outputs, *i.e.,* the outputs have mean 0, then the constant term is unnecessary (Exercise on Sheet 2)

Then find $\mathbf{w}$ that minimises the objective function

$$\mathcal{L}_{\text{ridge}}(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^{\mathsf{T}}(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^{\mathsf{T}}\mathbf{w}$$



Deriving Estimate for Ridge Regression

Suppose the data $\langle(\mathbf{x}_i, y_i)\rangle_{i=1}^{N}$ with inputs standardised and output centered

We want to derive expression for $\mathbf{w}$ that minimises

$$\begin{aligned}\mathcal{L}_{\text{ridge}}(\mathbf{w}) &= (\mathbf{X}\mathbf{w} - \mathbf{y})^{\mathsf{T}}(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^{\mathsf{T}}\mathbf{w} \\ &= \mathbf{w}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{w} - 2\mathbf{y}^{\mathsf{T}}\mathbf{X}\mathbf{w} + \mathbf{y}^{\mathsf{T}}\mathbf{y} + \lambda \mathbf{w}^{\mathsf{T}}\mathbf{w}\end{aligned}$$

Let's take the gradient of the objective with respect to $\mathbf{w}$

$$\begin{aligned}\nabla_{\mathbf{w}}\mathcal{L}_{\text{ridge}} &= 2(\mathbf{X}^{\mathsf{T}}\mathbf{X})\mathbf{w} - 2\mathbf{X}^{\mathsf{T}}\mathbf{y} + 2\lambda\mathbf{w} \\ &= 2\left(\left(\mathbf{X}^{\mathsf{T}}\mathbf{X} + \lambda\mathbf{I}_D\right)\mathbf{w} - \mathbf{X}^{\mathsf{T}}\mathbf{y}\right)\end{aligned}$$

Set the gradient to $\mathbf{0}$ and solve for $\mathbf{w}$

$$\left(\mathbf{X}^{\mathsf{T}}\mathbf{X} + \lambda\mathbf{I}_D\right)\mathbf{w} = \mathbf{X}^{\mathsf{T}}\mathbf{y}$$

$$\boxed{\mathbf{w}_{\text{ridge}} = \left(\mathbf{X}^{\mathsf{T}}\mathbf{X} + \lambda\mathbf{I}_D\right)^{-1}\mathbf{X}^{\mathsf{T}}\mathbf{y}}$$

Summary : Ridge Regression

In ridge regression, in addition to the residual sum of squares we penalise the sum of squares of weights

Ridge Regression Objective

$$\mathcal{L}_{\text{ridge}}(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^{\mathsf{T}} (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^{\mathsf{T}} \mathbf{w}$$

This is also called ℓ_2-regularization or weight-decay

Penalising weights "encourages fitting signal rather than just noise"

The Lasso

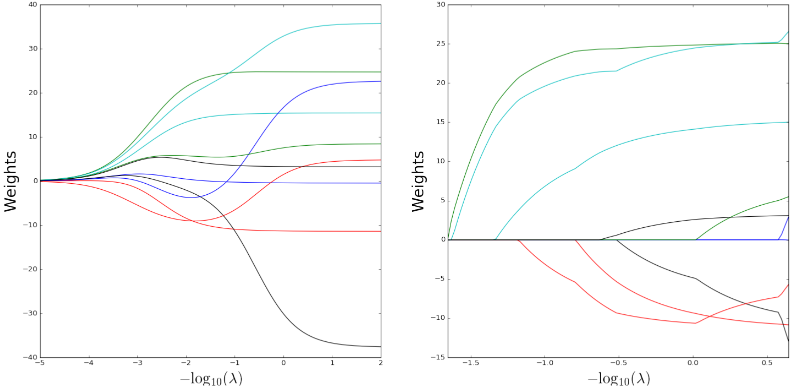
Lasso (least absolute shrinkage and selection operator) minimises the following objective function

Lasso Objective

$$\mathcal{L}_{\text{lasso}}(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^{\mathsf{T}} (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \sum_{i=1}^{D} |w_i|$$

- As with ridge regression, there is a penalty on the weights
- The absolute value function does not allow for a simple close-form expression (ℓ_1-regularization)
- However, there are advantages to using the lasso as we shall see next

Comparing Ridge Regression and the Lasso



When using the Lasso, weights are often <u>exactly</u> 0.

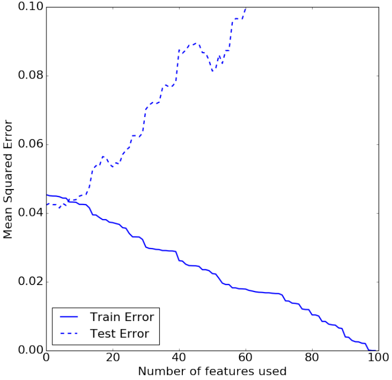
Thus, Lasso gives sparse models.

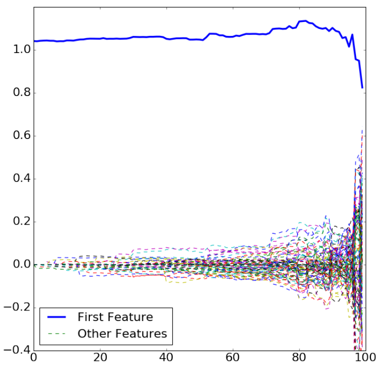
Overfitting: How does it occur?

We have $D = 100$ and $N = 100$ so that $\mathbf{X}$ is 100×100

Every entry of $\mathbf{X}$ is drawn from $\mathcal{N}(0, 1)$

$y_i = x_{i,1} + \mathcal{N}(0, \sigma^2)$, for $\sigma = 0.2$





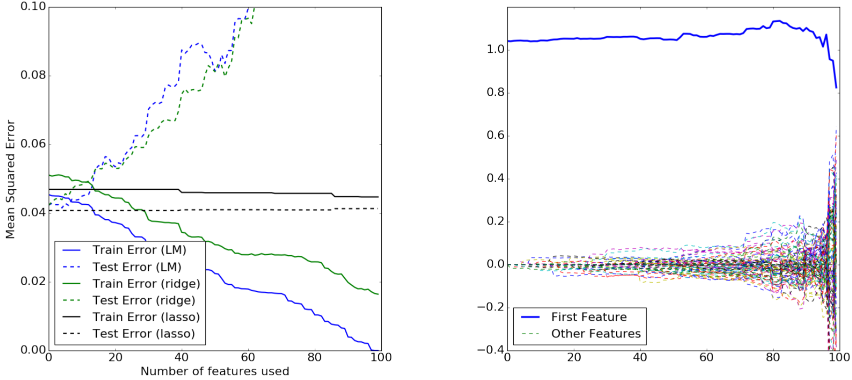
No regularization

Overfitting: How does it occur?

We have $D = 100$ and $N = 100$ so that $\mathbf{X}$ is 100×100

Every entry of $\mathbf{X}$ is drawn from $\mathcal{N}(0, 1)$

$y_i = x_{i,1} + \mathcal{N}(0, \sigma^2)$, for $\sigma = 0.2$



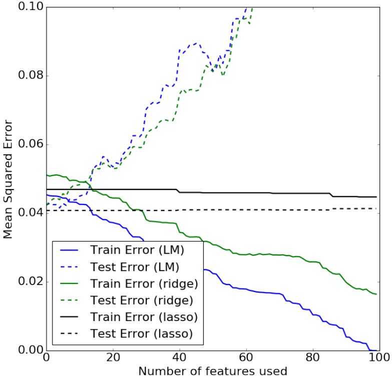
No regularization

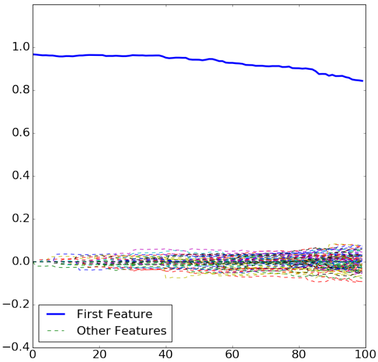
Overfitting: How does it occur?

We have $D = 100$ and $N = 100$ so that $\mathbf{X}$ is 100×100

Every entry of $\mathbf{X}$ is drawn from $\mathcal{N}(0, 1)$

$y_i = x_{i,1} + \mathcal{N}(0, \sigma^2)$, for $\sigma = 0.2$





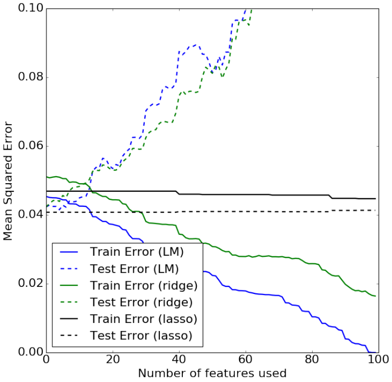
Ridge

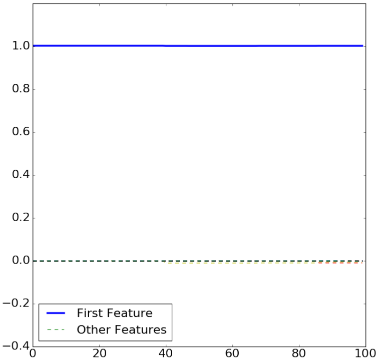
Overfitting: How does it occur?

We have $D = 100$ and $N = 100$ so that $\mathbf{X}$ is 100×100

Every entry of $\mathbf{X}$ is drawn from $\mathcal{N}(0, 1)$

$y_i = x_{i,1} + \mathcal{N}(0, \sigma^2)$, for $\sigma = 0.2$





Lasso

Outline

Ridge Regression and Lasso

Model Selection

How to Choose Hyper-parameters?

- So far, we were just trying to estimate the parameters $\mathbf{w}$
- For Ridge Regression or Lasso, we need to choose λ
- If we perform basis expansion
  - For polynomials, we need to pick degree d
- For more complex models there may be more hyperparameters

Using a Validation Set

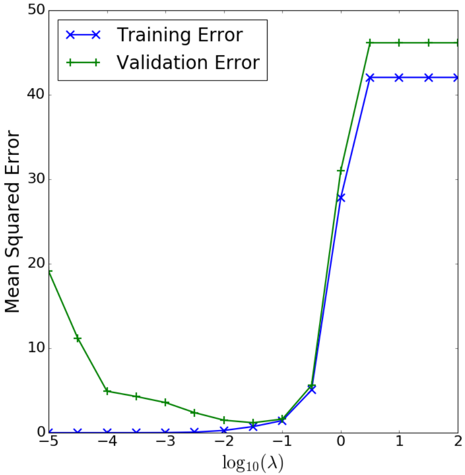
- Divide the data into parts: training, validation (and testing)
- Grid Search: Choose values for the hyperparameters from a finite set
- Train model using training set and evaluate on validation set

λ	training error(%)	validation error(%)
0.01	0	89
0.1	0	43
1	2	12
10	10	8
100	25	27

- Pick the value of λ that minimises validation error
- Typically, split the data as 80% for training, 20% for validation

Training and Validation Curves

- Plot of training and validation error vs λ for Lasso
- Validation error curve is U-shaped



K-Fold Cross Validation

When data is scarce, instead of splitting as training and validation:

- Divide data into K parts
- Use $K - 1$ parts for training and 1 part as validation
- Commonly set $K = 5$ or $K = 10$
- When $K = N$ (the number of datapoints), it is called LOOCV (Leave one out cross validation)

Run 1	train	train	train	train	valid
Run 2	train	train	train	valid	train
Run 3	train	train	valid	train	train
Run 4	train	valid	train	train	train
Run 5	valid	train	train	train	train