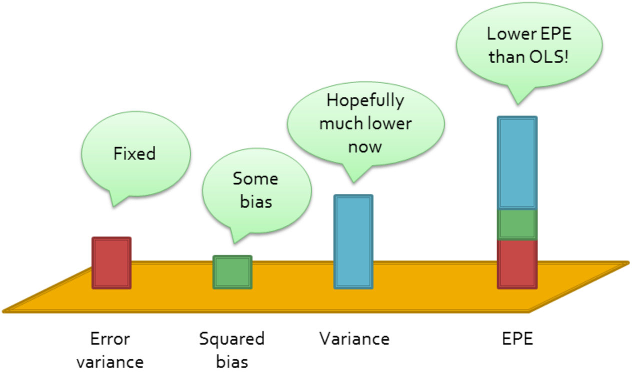
Model Selection

Line Clemmensen

DTU

02582 Computational Data Analysis, 2019

Last week - terminology



- Exoected Prediction Error
- Bias-Variance trade-off

Last week - methods

Methods introduced,

- Ordinary Least Squares
- Ridge Regression
- Fischer Linear Discriminant Analysis
- K-Nearest-Neighbor

What problems are they solving?

Properties?

Today's lecture

- Model Complexity
  - Bias, variance, overfitting and underfitting
- Model Selection
  - Methods for selecting an appropriate model from an ensemble of candidates.
    - Training, test and validation set
    - Cross-validation
    - Methods based on information criteria
- Model Assessment
  - Bootstrap
  - Sensitivity, specificity and ROC curves

Model Complexity

- Overfitting and underfitting
- Regularization
- Bias and variance

Over- and underfitting

When fitting statistical models there are parameters to tune and choices to make

Name a few examples!

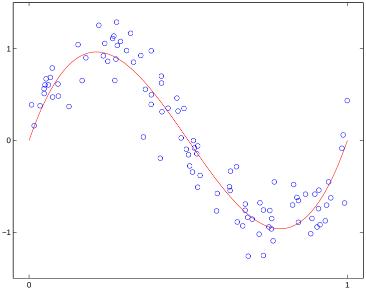
Over- and underfitting

- We prefer a simple model.
- We prefer models that work (low EPE).
- These properties may contradict

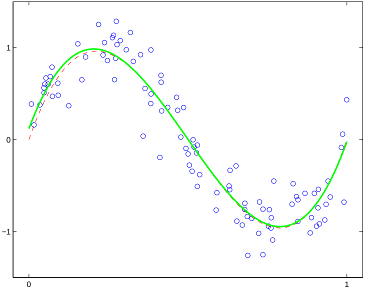
Too simple: Our data set will not be accurately described. Model assumptions are wrong.

Too complex: The model becomes too flexible. We fit noise in data. We need lots of data to support such a model.

Polynomial data example - OLS regression



From Christopher Bishop's: Pattern Recognition and Machine Learning



$$y_i = 20x_i^3 - 30x_i^2 + 10x_i + e_i$$

$$e_i \in N(0, 0.5^2)$$

$$i = 1, ..., n$$

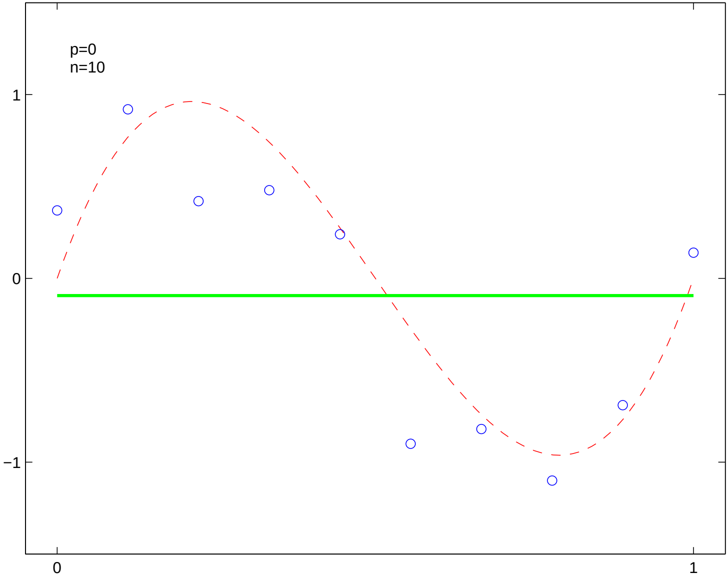
$$n = 100$$

$$\hat{y} = 20.3x^3 - 30.7x^2 + 10.4x - 0.04$$

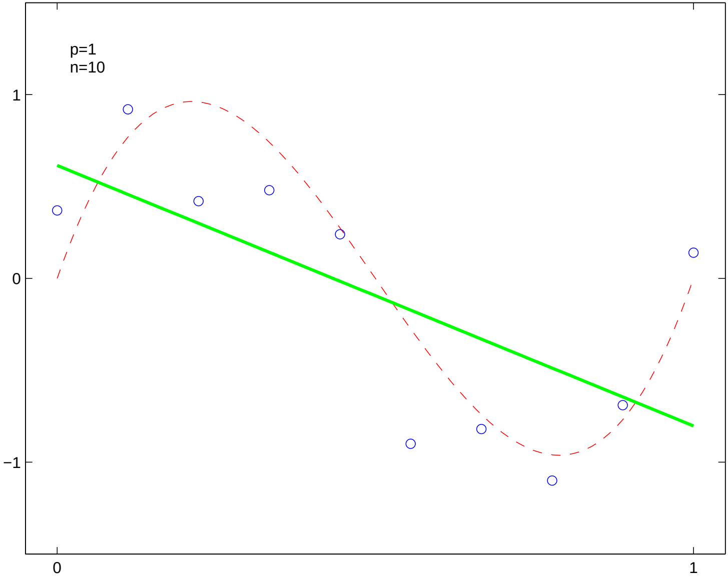
$$\beta_{OLS} = (X^T X)^{-1}(X^T Y)$$

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ \vdots & & & \vdots \\ 1 & x_n & x_n^2 & x_n^3 \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

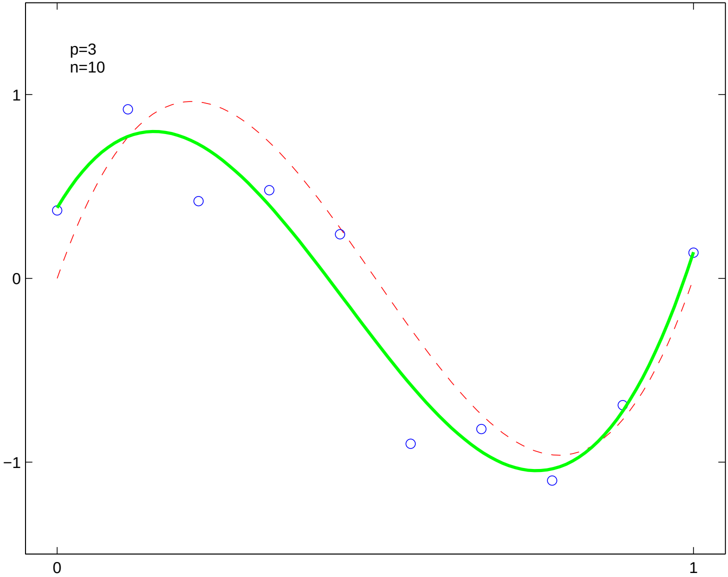
0th order polynomial



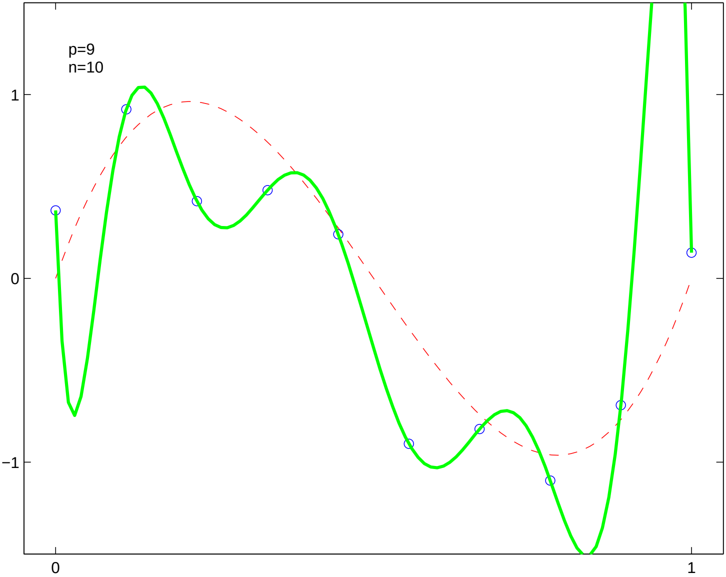
1st order polynomial



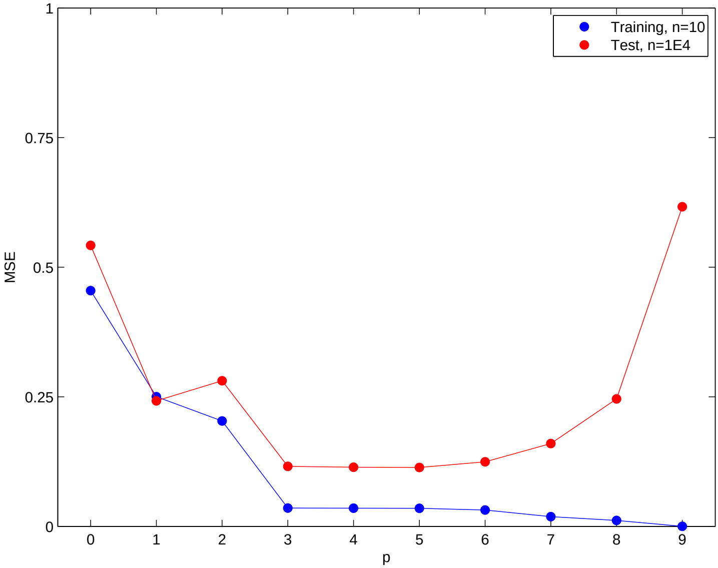
3rd order polynomial



9th order polynomial



Over-fitting

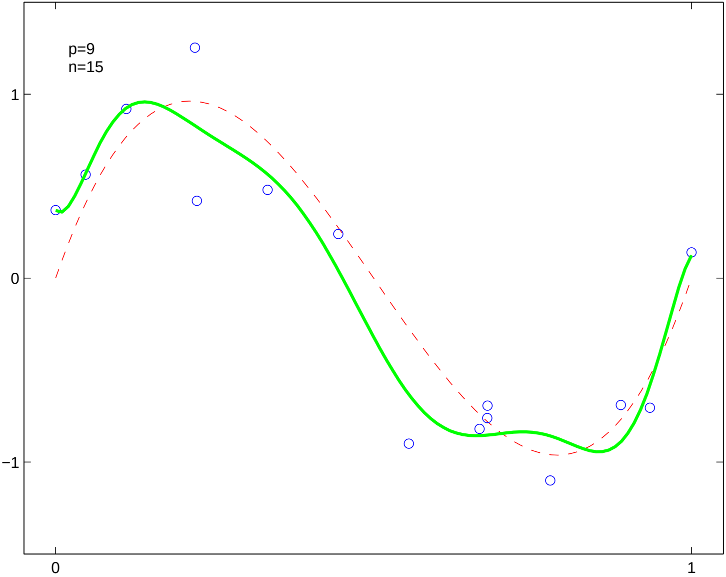


Polynomial coefficients

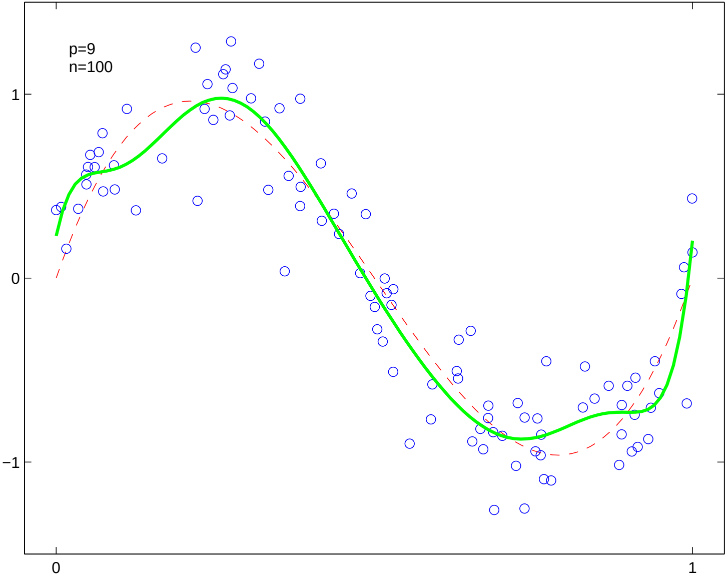
	$p = 0$	$p = 1$	$p = 3$	$p = 9$
$\hat{\beta}_0$	-0.094	0.61	0.38	0.36
$\hat{\beta}_1$		-1.41	5.84	-94.17
$\hat{\beta}_2$			-23.09	2560.25
$\hat{\beta}_3$			17.00	-25599.24
$\hat{\beta}_4$				129867.51
$\hat{\beta}_5$				-375245.91
$\hat{\beta}_6$				642903.57
$\hat{\beta}_7$				-645828.97
$\hat{\beta}_8$				351189.61
$\hat{\beta}_9$				-79752.87

Notice the poor quality of the β estimates even for $p = 3$

Data set size



Data set size



Idea

Observation:

- Over-fitting is associated with high variance in parameter estimates

Occam's razor:

- The smallest suitable model is the best.

Regularization, two approaches

- Bayesian prior to β
- Penalty to models with large β
  - $\hat{\beta} = \arg\min_{\beta} ||Y - X\beta||_2^2 + \lambda||\beta||$
  - $||\cdot||$ suitable norm

Ridge regression

We can use a flexible model (such as the 9th order polynomial before) and **avoid overfitting via regularization.**

Quadratic norm,

$$\beta_{Ridge} = \arg\min_{\beta} ||Y - X\beta||_2^2 + \lambda||\beta||_2^2$$

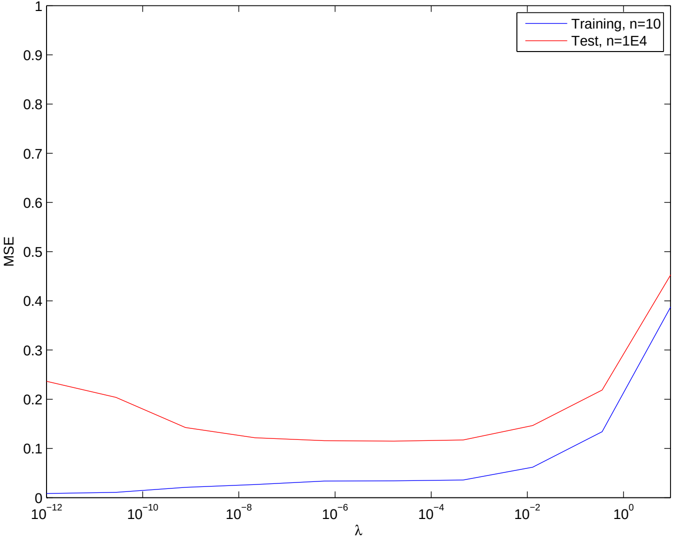
with simple solution,

$$\beta_{Ridge} = (X^T X + \lambda I)^{-1}(X^T Y)$$

Regularization parameter λ,

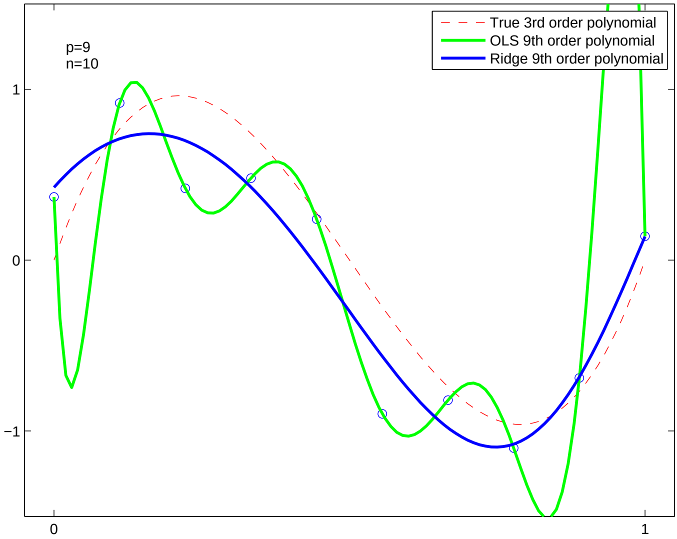
- $\lambda = 0$ gives $\beta_{Ridge} = \beta_{OLS}$
  - No bias
  - High variance
- $\lambda \rightarrow \infty$ gives $\beta_{Ridge} \rightarrow 0$
  - High bias
  - No variance

Tuning the Regularization parameter



What value of **regularization parameter** λ should we chose?

Optimal ridge model



What can we say about **bias** and **variance**?

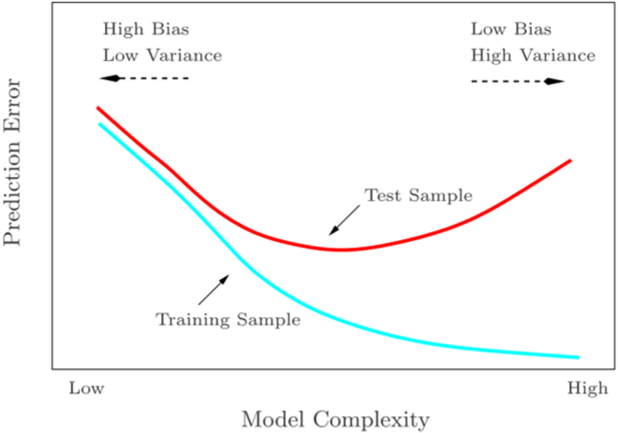
Bias and Variance

Another way of describing the under and overfitting situations.

We usually face two scenarios:

- Getting it right on average but being wrong most of the time
- Never getting it quite right, but usually almost

Bias/variance vs model complexity



Optimal properties

Given some restrictions.... *Unbiased*

Unbiased $E(\beta_{OLS}) = \beta_{true} \quad (= [0, 10, -30, 20]^T)$

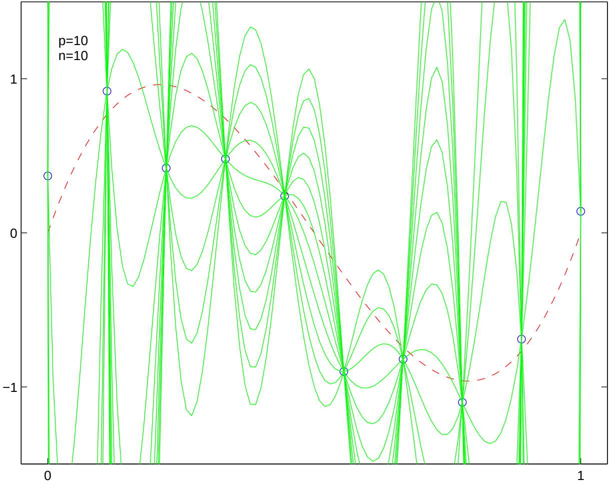
BLUE Best Linear Unbiased Estimator
Minimum variance of all linear unbiased estimators.

UMVUE Uniformly Minimum-Variance Unbiased Estimator
Unbiased estimator with lower variance than any other unbiased estimator for all possible values of the parameter.

EPE Expected Prediction Error

$$\begin{aligned}
EPE(x_0) &= E[(Y - \hat{f}(x_0))^2] | X = x_0] \\
&= \sigma_e^2 + [E\hat{f}(x_0) - f(x_0)]^2 + E[\hat{f}(x_0) - E\hat{f}(x_0)]^2 \\
&= \sigma_e^2 + \text{Bias}^2(\hat{f}(x_0)) + \text{Var}(\hat{f}(x_0)) \\
&= \text{Irreducible Error + Bias}^2 \text{ + Variance} \\
&\text{Unbiased/BLUE/UMVUE} => \\
&= \text{Irreducible Error + Minimal variance|Unbiased}
\end{aligned}$$

10th order polynomial



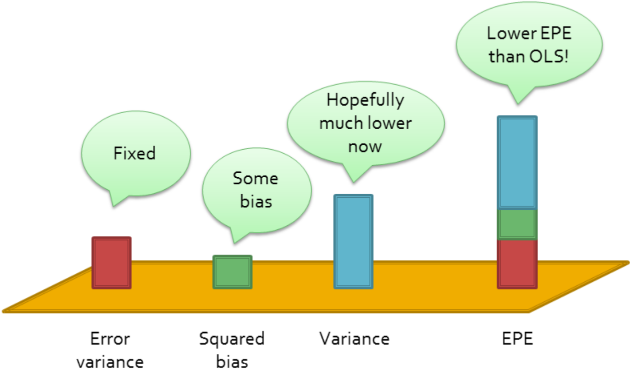
No solution to

$$\beta_{OLS} = (X^T X)^{-1}(X^T Y)$$

infinitely many to

$$\hat{\beta} = \arg\min_{\beta} ||Y - X\beta||_2^2$$

Bias-variance trade-off



- Estimated Prediction Error
- Bias-Variance trade-off

Model selection tools

- Overfitting and underfitting
- Training, validation and test set
- Cross validation
- Informational criteria

Model selection and assessment

The more complex the model, the better we will fit the training data. Often we overfit to the training data.

Overfit models can perform poorly on test data - high variance.

Underfit models can perform poorly on test data - high bias.

We perform

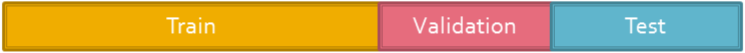
1. **Model selection:** To choose a value of a tuning parameter or to choose between models.
2. **Model assessment:** To assess our chosen model, e.g. estimate the future prediction ability of the chosen model.

For both of these purposes, the best approach is to evaluate the procedure on an independent test set.

If possible one should use different test data for (1) and (2) above. A **validation set** for (1) and a **test set** for (2).

Model selection

For well-conditioned problems, divide data into three parts



Training set
Used to build different models, e.g. with different values of the tuning parameter λ.

Validation set
Used to calculate $\hat{EPE}$ for the different models and select the best, i.e. used to tune parameters, select features and similar *modeling decisions*. Also called **dev** (development) set, or **hold-out cross validation** set.

Test set
Used to estimate/test how well the resulting model performs, i.e. *evaluate performance* of algorithm, but make no decisions based on this data.

Model selection

Andrew Ng, Stanford University, Machine Learning Yearning (a must read for practitioners)

- Dev and test set should reflect data you expect to get in the future (i.e. the goal of the modeling)
- Dev and test sets must come from the same distribution
- Alternatively use a dev$_{train}$ and a dev$_{test}$ set
- Use a single-number metric to evaluate performance

Making sure the split isn't critical

```
for m=1:M
    Randomize data (permute)
    Split data in 3 (train, validation, test)
    Train model on range of tuning parameters using train data
    Select best model based on validation data
    Test model to estimate the error on test set
end
Calculate mean and std error over M test errors
```

Cross-validation

Often there is insufficient data to create a separate validation and test set. In this instance use cross validation instead.

Cross-validation:

- The primary method for estimating a tuning parameter, e.g. λ.
- No underlying assumptions
  - Except, that observations are assumed to be independent.
- Is simple, intuitive and easy to implement

K-fold cross-validation

- Split randomized data into K (roughly) equal parts

1	2	3	4	5
Train	Train	Validation	Train	Train

- For each $k = 1, 2, ..., K$, fit the model with parameter λ to the other $K - 1$ parts, giving $\hat{\beta}^{\neg k}(\lambda)$ and compute its error in predicting the kth part:

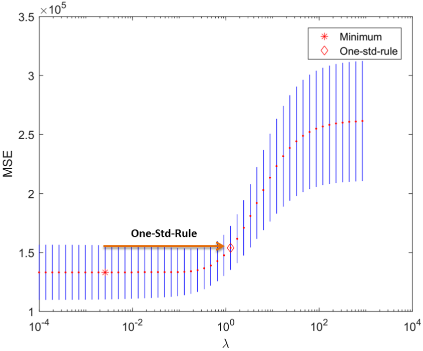
$$Err_k(\lambda) = \sum_{i \in k\text{th part}} (y_i - X_i \hat{\beta}^{\neg k}(\lambda))^2$$

  This gives the cross-validation error

$$CV(\lambda) = \frac{1}{K} \sum_{k=1}^{K} Err_k(\lambda)$$

- Do this for many values of λ and choose the value of λ that makes $CV(\lambda)$ smallest

CV Model Selection



- Estimated prediction error curves, computed via cross validation.
- Bars indicate estimated standard errors.
- Vertical line chosen by “one standard error rule”.

One standard error rule: Choose the smallest model whose error is no more than one standard error above the error of the best model.

This compensates that the CV procedure chooses somewhat too complex models.

Variance estimates in K-fold cross-validation

We often use the standard error to give us a confidence about the mean (the cross-validation error from the previous slide)

$$CV(\lambda) = \frac{1}{K}\sum_{k=1}^{K} Err_k(\lambda)$$

$$S.E.(\lambda) = \frac{1}{\sqrt{K}}\sqrt{\frac{1}{K}\sum_{k=1}^{K}(Err_k(\lambda) - CV(\lambda))^2}$$

NOTE: This is a **biased variance estimate** as the observations are correlated! The variance is underestimated - thus be careful in trying to test for significant differences beteween two models (Bengio 2004).

Considerations

Observations are supposed to be independent. Otherwise information will leak from validation set to training set and we will overfit.

- Permute data before splitting in folds - the data set might be sorted in some way.
- Normalize training data separately for each fold.
- Impute missing values separately for each training set.
- If observations are sampled in groups, let each group go into the same fold.
- Be extremely careful with data sampled from dynamic systems (time-dependent data).
- Be careful to perform ALL pre-processing within the CV folds.

Be careful using leave-one-out CV ($K = N$), the folds are too similar. Use $K = 5$ or 10 as a good compromise between bias and variance.

Validation and test set issues

- Important to have both cross-validation and test sets, since we often run CV many times with different parameters. This can bias the CV results.
- A separate test set provides a convincing, independent assessment of a models performance (use only once!).
- Test set results might still overestimate actual performance, as a real future test set may differ in many ways from todays data.

Tibshirani claims that a test set should be 50 % or 30 % of all data.

Andrew Ng says this depends on the amount of available data.

Watch out for overfitting dev/validation set

- Manually looking at observations gives you an intuition about data and can cause you to overfit the dev set

- Split into eyeball dev set and black box dev set
- Write down *everything* you do to your data during your analysis (Efron guide: log book)

When to get a new dev/validation set

- Test set and dev set distributions differ
- You have overfit to the dev set
- The optimizing metric does not represent the aim of the project

Error analysis and modeling choices

Human-level error	1.0%	8.5%	0.0%	0.0%
Training error	9.0%	9.0%	1.5%	1.0%
Training Dev error	-	-	9%	10%
Dev error	10%	11%	10%	20%
Focus on:	Bias	Variance	Variance	Data mismatch

Andew Ng's Two Knobs

Two fundamental assumptions of supervised learning

- You can fit the training data
  Avoidable bias knob
- Training performance generalizes to dev- and test- set
  Variance knob

Error analysis and modeling choices

Identify biggest error gap:

- Human-level error (expert, similar or baseline)
  - Use more complex model
  - Better optimization algorithms
  - Hyper parameter search/model structure

Bias knob

- Training error
  - Regularization
  - More data
  - Hyper parameter search/model structure

Variance knob

- Dev error

Error analysis and modeling choices

It is good practice, and one can learn from:

- Plotting residuals
- Examining misclassifications

Information criteria

- Optimism and training error
- Information criteria
  - C_p-statistic
  - Akaike Information Criterion, AIC
  - Bayes Information Criterion, BIC

Optimism of the training error

The training error for an OLS regression is

$$\textbf{training-error} = \frac{1}{N}\sum_{i=1}^{N}(y_i - x_i\hat{\beta})^2$$

Think that we for each x_i can get a new measurement y_i^0. What would the error for these new y_i^0 be?

$$\textbf{in-sample-error} = \frac{1}{N}\sum_{i=1}^{N}(y_i^0 - x_i\hat{\beta})^2$$

Selecting the model with the smallest in-sample-error would be an effective model selection tool.

Advantage: We only need to fit one model to the training data (fast!!)

Optimism of the training error

The difference between the **in-sample-error** and the **training-error** is called **optimism.** It is possible to show that quite generally

$$\textbf{expected optimism} = \frac{2}{N}\sum_{i=1}^{N} Cov(\hat{y}_i, y_i).$$

The more we overfit data the greater $Cov(\hat{y}_i, y_i)$ will be and thereby increasing the optimism.

In the linear case with d variables and squared error loss we have

$$\textbf{expected optimism} = d\sigma_e^2.$$

Therefore we have that

$$\textbf{expected in-sample-error} = \textbf{expected training-error} + 2\frac{d}{N}\sigma_e^2$$

C_p-statistic

Given a squared error loss and a model with d variables we can calculate the so-called C_p-statistic

$$C_p = \overline{\text{err}} + 2\frac{d}{N}\hat{\sigma}_e^2$$

Using this metric we can select the best model by choosing the model that minimizes C_p.

Expected training error,
use the actual training error $\frac{1}{N}\sum_{i=1}^{N}(y_i - x_i\hat{\beta})^2$ as an estimate.

Noise variance,
use the mean squared error (MSE) of a low bias model (OLS or KNN) as the estimate $\hat{\sigma}_e^2$.

Akaike Information Criterion

AIC uses a log-likelihood loss function (logL) to estimate the error term (maximized log-likelihood)

$$AIC = -\frac{2}{N}\text{logL} + 2\frac{d}{N}$$

For the Gaussian case with a tuning parameter λ we define

$$AIC(\lambda) = \overline{err}(\lambda) + 2\frac{d(\lambda)}{N}\hat{\sigma}_e^2.$$

- For the Gaussian model C_p and AIC are identical.
- $d(\lambda)$ is the effective number of parameters in the model tuned with λ.

Effective number of parameters

For a linear fitting method

$$\hat{Y} = SY$$

we can calculate the effective number of parameters as

$$df(S) = trace(S)$$

ie the sum of the diagonal elements of S.

Both OLS and ridge regression are linear fitting methods,

$$\begin{aligned}\hat{Y} &= X\hat{\beta} \\ &= X(X^TX)^{-1}X^TY \\ &= SY\end{aligned}$$

Bayes Information Criterion

The Bayes Information Criterion, BIC, is like AIC based on a log likelihood loss function. It is motivated from a Bayesian approach selecting the model with the largest posterior probability.

$$BIC = -2\text{logL} + \log(N)d$$

Under the Gaussian model and squared error loss,

$$BIC(\lambda) = \frac{N}{\hat{\sigma}_e^2}\left(\overline{\text{err}}(\lambda) + \log(N)\frac{d(\lambda)}{N}\hat{\sigma}_e^2\right).$$

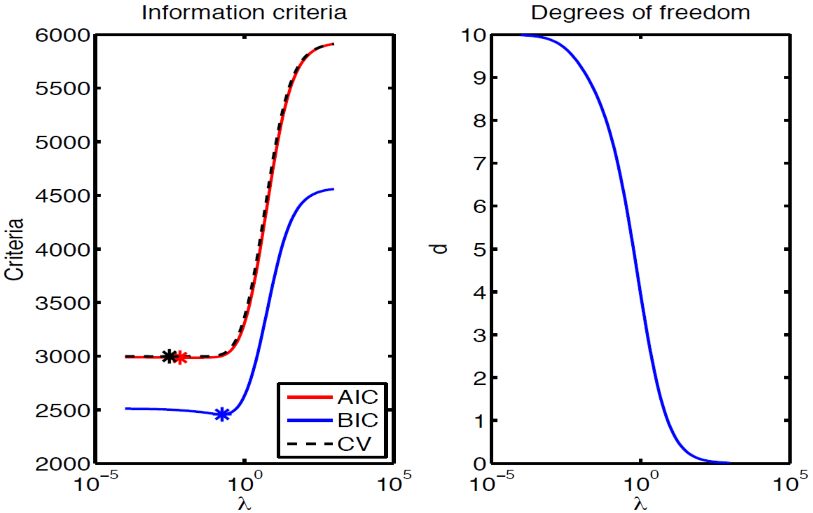
Again, select the model that has the lowest value.

AIC and BIC

For a given set of models (including the true model).

- As $n \rightarrow \infty$ the probability that BIC selects the correct model approaches one, whereas AIC tends to select a model which is too complex.
- For small sample sizes BIC tends to select a model which is too simple.

Example on Diabetes data



Relationship between AIC and CV

- Stone(1977) showed that AIC and leave-one-out cross-validation are asymptotically equivalent.
- Often leave-one-out cross-validation and AIC tend to choose models which are too complex.

Stone M. (1977) An asymptotic equivalence of choice of models by cross-validation and Akaike's criterion. Journal of the Royal Statistical Society Series B. 39, 44-7.

Model Assessment

- Bootstrap Methods
- Classifier performance
  - Confusion Matrix
  - ROC curves

The Bootstrap

Bootstrap is a general method for **assessing statistical accuracy**.

- Term from the story of Baron von Munchhausen “Pulling oneself up by the bootstraps” (though in fact it was by the hair)
- “Pull oneself over a fence by one’s bootstraps”, to mean an absurdly impossible action.
- Bootstrap estimates can be thought of as Monte-Carlo estimates.

The statistical bootstrap is also an absurdly simple and effective way of solving apparently hard problems.

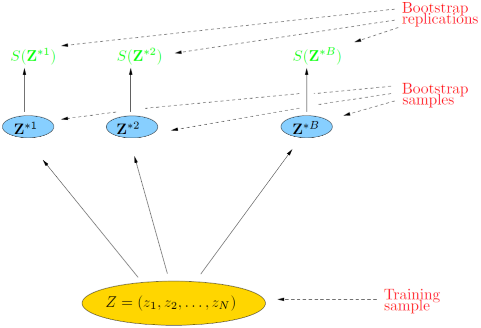
Bootstrap Method

1. Given a training set $Z = (z_1, z_2, ..., z_N)$ where $z_i = (x_i, y_i)$, the basic idea is to randomly draw data sets with replacement from the training data, each sample the same size as the original training set.
2. This is done B times ($B = 100$ say), producing B bootstrap data sets. Then refit the model to each of the bootstrap data sets, and examine the behavior of the fits over B replications.

Instead of repeating the whole data collection 100 times we just draw observations with replacement from our training data and use them as if they were new experiments.

Example

The variance of a parameter, *S*, is estimated by taking the variance over *B* bootstrap replications (the value of the parameter for the specific bootstrap samples).

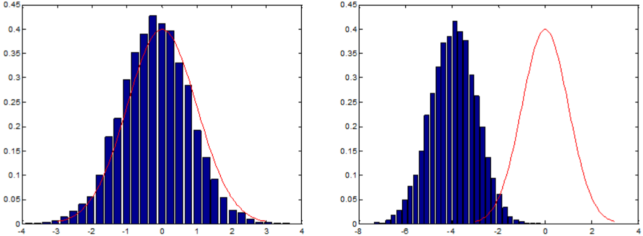


$$\widehat{\mathrm{Var}}[S(Z)] = \frac{1}{B-1} \sum_{b=1}^{B} (S(Z^{*b}) - \bar{S}^*)^2$$

Example

The distribution of two β estimates from OLS regression ($H_0 : \beta_i = 0$).

Compare to the distribution under the null-hypothesis that $\beta_i = 0$.



Example

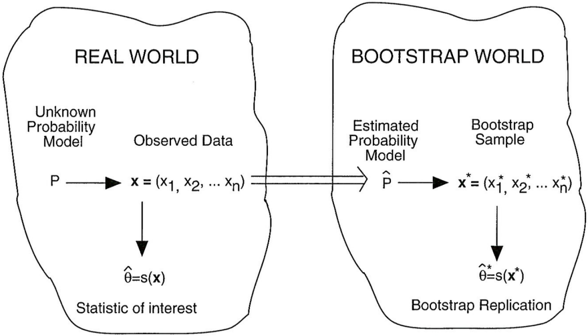
Training error is estimated by taking the mean error over the *B* bootstrap replications.

- Error measured in relation to the original data set.
- However, it is a bad idea. The bootstrap test set is not independent from the training (original) set. (We sample with replacement)
- Better, use **Out-Of-Bag** samples.
  - When measuring the error at (x_i, y_i) select only bootstrap replications which do not contain (x_i, y_i).
  - About 1/3 of the observations will be left out of every bootstrap replicate.

Understanding bootstrap

David Freedman's terminology

We do not know P, therefore are we operating in the world where $\hat{P}$ is the truth. Use it as a mirror copy of the real world!


REAL WORLD
Unknown Probability Model
Observed Data
P
$\mathbf{x} = (x_1, x_2, \dots x_n)$
$\hat{\theta}=s(\mathbf{x})$
Statistic of interest
BOOTSTRAP WORLD
Estimated Probability Model
Bootstrap Sample
$\hat{P}$
$\mathbf{x}^* = (x_1^*, x_2^*, \dots x_n^*)$
$\hat{\theta}^*=s(\mathbf{x}^*)$
Bootstrap Replication

Remarks

1. How many bootstrap replicates do we need?
    - For standard deviation, a couple of hundreds.
    - For confidence intervalls, 1000 - 2000.

   Try different numbers and see if it affects the result.

2. Bootstrap does not work so well for “tail statistics”. It works better for “in the middle” of data.

3. Tibshirani: Do not use bootstrap for model selection. It is intended for other problems.

Classifier performance

- Confusion Matrix
- ROC curves

Classifiers

Once we have decided on a particular model, we are finished ... or are we?

Your (binary) classifier defines a rule,

“if $P(x = \text{red}) \geq c$ then red else blue”

The value of c is often chosen to 0.5 but it is not god-given.

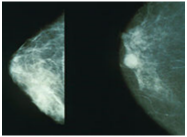
Implications

Obviously, changing the cut value, *c*, will favor one class more.

Give examples of classification problems with a favorable class.

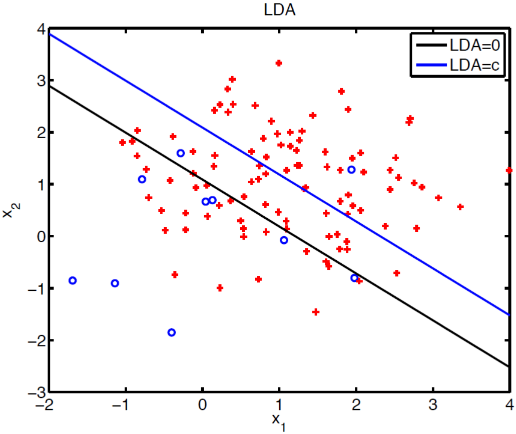
Mammography Task - find characteristic masses and microcalcifications indicative of breast cancer.

- What is the cost of a region falsely classified as high risk?
- What is the cost of a region falsely classified as low risk?



Linear example

Linear discriminant analysis,



Confusion matrix

		True Outcome		
		Postive	Negative	
Test Outcome	Postive	🟩	🟥	🟨
	Negative	🟥	🟩	🟨
		🟨	🟨	

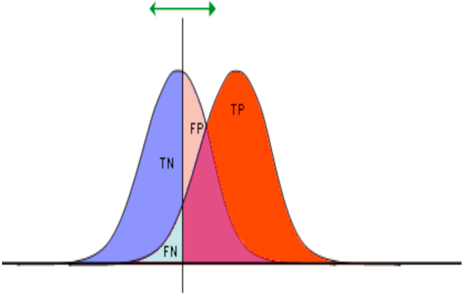
Confusion matrix

		True Outcome		
		Postive	Negative	
Test Outcome	Postive	True Positive (TP) *hits*	False Positive (FP) *Type I error,* *false alarm*	
	Negative	False Negative (FN) *Type II error,* *misses*	True Negative (TN) *correct rejection*	

Confusion matrix

		True Outcome		
		Postive	Negative	
Test Outcome	Postive	True Positive (TP) *hits*	False Positive (FP) *Type I error, false alarm*	Positive Predictive Value PPV = TP/(TP+FP)
	Negative	False Negative (FN) *Type II error, misses*	True Negative (TN) *correct rejection*	Negative Predictive Value NPV = TN/(TN+FN)
		Sensitivity (hit rate, recall) TPR = TP/P	Specificity SPC = TN/N	

Example with gaussian data



TP	FP
FN	TN
1	1

Diagnostics

- **Sensitivity** (TPR)
  - Your method's ability to identify positives
- **Specificity** (SPC)
  - Your method's ability to identify negatives
- **Positive Predictive Value** (PPV)
  - Proportion of positives which are correctly classified
- **Negative Predictive Value** (NPV)
  - Proportion of negatives which are correctly classified
- **False Discovery Rate** (FDR)
  - Proportion of positives which are incorrectly classified

ROC curve

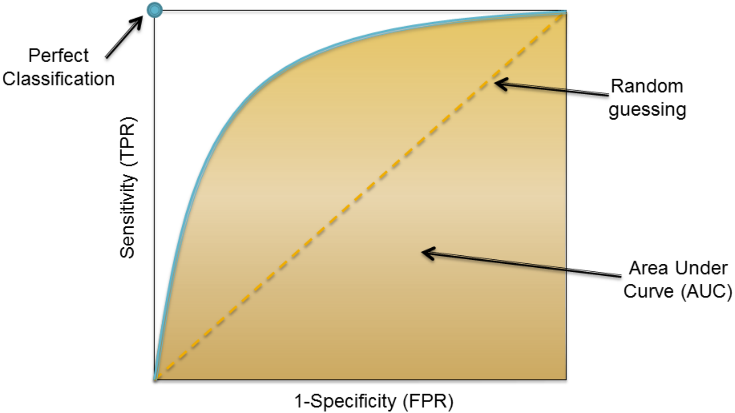
Sensitivity and specificity changes as the cut-off value changes.

Let’s plot all combinations!

The **Receiver Operating Characteristic** (ROC)

- Sensitivity vs (1-Specificity) or equivalently,
- True positive rate vs false positive rate.

ROC curve



Perfect Classification

Random guessing

Area Under Curve (AUC)

Maximizing the AUC

Can we explicitly maximize AUC when creating models?

- AUC is discrete, hard to optimize model parameters directly w.r.t. AUC.
- Most continuous methods such as maximum likelihood corresponds well to AUC.

Reporting
Pick a model, estimate its parameters and report on AUC.

Regularization
Select regularization parameter based on AUC.

Use AUC with caution

Assume you are segmentting a small part, 1%, of an image (e.g. a tumor in a mammography)

- You have a large number of true negatives (TN)
- What is the penalty for a segmentation twice as large as it should be?
  - Specificity = TN/(TN+FP) = 1mio/(1mio+10000)
  - Specificity: 100% => 99%

Result: AUC is very high, even though the segmentation is poor.

Interpret with caution when prevalence is off.

Using a single-number metric

Example: Detection of degradations on train tracks: Degradations = 1s; Normal behavior = 0s.

Classifier	Recall rate (sensitivity)	Specificity (1 - FDR)
A	92 %	85 %
B	90 %	88 %
C	71 %	95 %
D	99 %	70 %

Which model would you choose?

Summary

- Over- and underfitting
- Model selection
  - Training/validation/test set
  - Cross-validation
    - One standard error rule
  - Information criterion
    - C_p-statistics
    - AIC
    - BIC
- Model assessment
  - Bootstrap
  - Sensitivity vs specificity and ROC curves
  - Confusion matrices