COMP90051 **Statistical Machine Learning**

Semester 2, 2015

Lecturer: Ben Rubinstein

4. Regularisation as conditioning; Regularisation as limiting model complexity

THE UNIVERSITY OF MELBOURNE

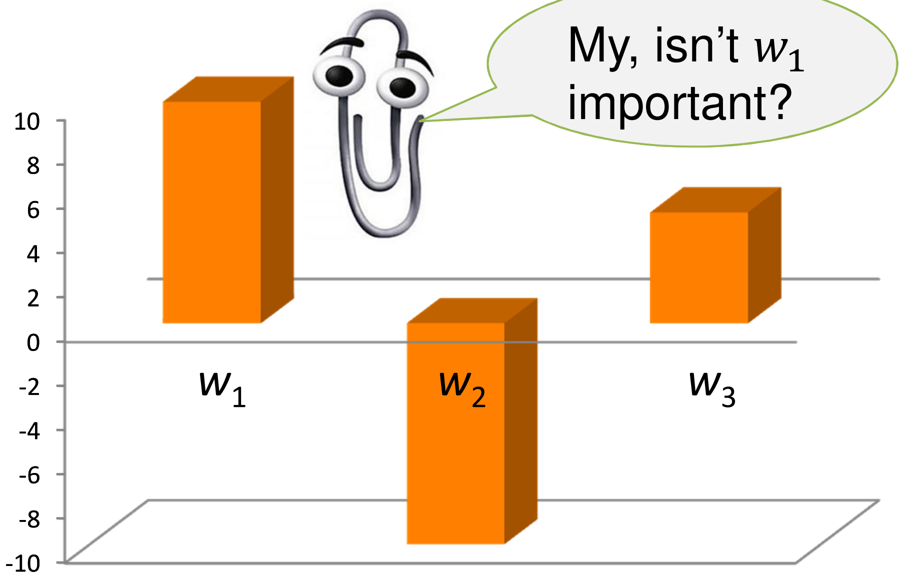
Ill-Conditioned Learning and Regularisation

Many machine learning methods can overfit or take a long time to train if given too many features or features that are too similar (i.e. ***irrelevant features****). These learning problems are called by some, ill-posed inverse problems.* ***Regularisation*** *re-conditions them.*

Irrelevant Features: An Xtreme Example

- Linear model on *d*=3 features, first two same
  - If **X** is *n* x 3 matrix of the *n* instances
  - First two columns of **X** identical
  - Feature 2 is **irrelevant** (alt. 1)
  - Model: $f(\mathbf{x}) = \mathbf{x}'\mathbf{w} = \sum_{i=1}^{d} w_i x_i$

3	3	7
6	6	9
21	21	79
34	34	2



- Effect of perturbations on model predictions?
  - Add Δ to w_1
  - Subtract Δ from w_2

…identical predictions

…no interpretability

Irrelevant Features in General

- Xtreme case: features complete clones
- For linear models, more generally
  - Feature $\mathbf{X}_{\cdot i}$ is irrelevant if
  - $\mathbf{X}_{\cdot i}$ is a linear combination of other columns

$$\mathbf{X}_{\cdot i} = \sum_{j \neq i} \alpha_j \mathbf{X}_{\cdot j}$$

  … for any constants α_j

- Even *near*-irrelevance can be problematic
- Not just a pathological xtreme; *easy to happen!*

Irrelevant Features: …and the ugly

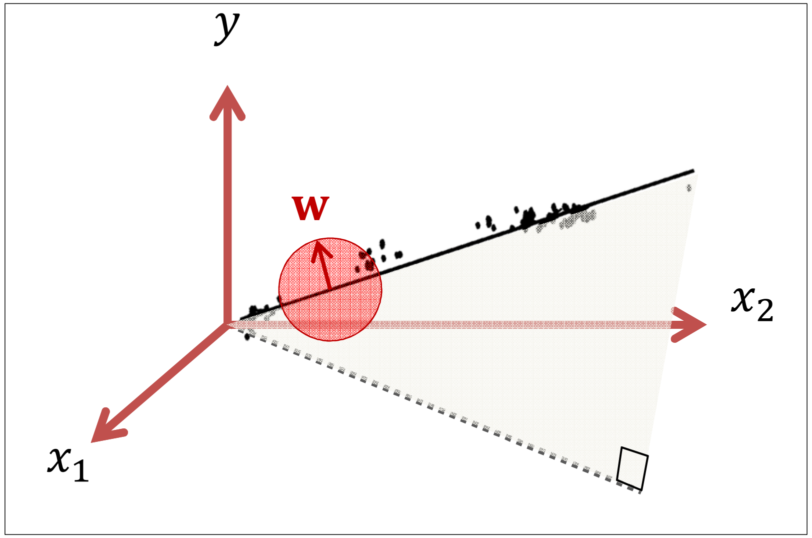
Ugly: computation

- Linear regression fits $\min_{\mathbf{w}} \sum_i (y_i - \mathbf{X}_{i\cdot}\mathbf{w})^2$
- Solution: $\mathbf{w}^* = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$, an inverse problem
- Irrelevance → rank deficient

  i.e. some eigenvalues zero/negative

  → no inverse $\cancel{(\mathbf{X}'\mathbf{X})^{-1}}$

  This is an ill-posed inverse problem



No uniqueness

What can we do about it?

Re-Condition (aka Regularise)

"Re-condition' $\mathbf{X'X}$:

- I.e. use $\mathbf{w}^* = (\mathbf{X'X} + \lambda\mathbf{I})^{-1}\mathbf{X'y}$
- Adds $\lambda > 0$ to each eigenvalue
- For big enough λ we are 👍

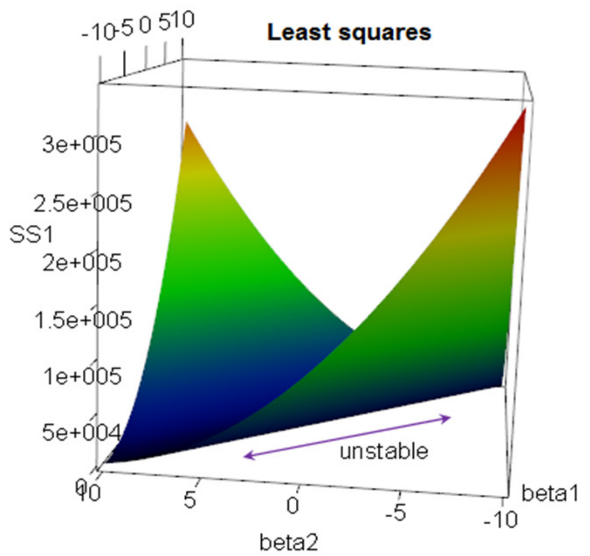
This is **ridge regression**!

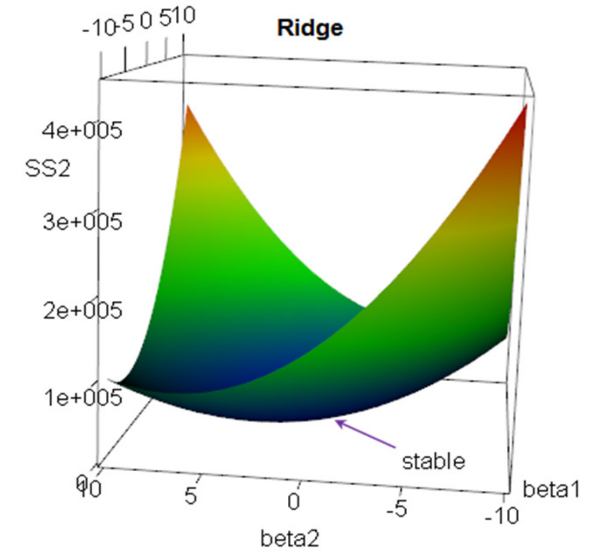
$$\min_{\mathbf{w}} \sum_i (y_i - \mathbf{X}_{i.}\mathbf{w})^2 + \lambda \|\mathbf{w}\|_2{}^2$$

- Added part is a regularisation term

Regularisation is a win-win

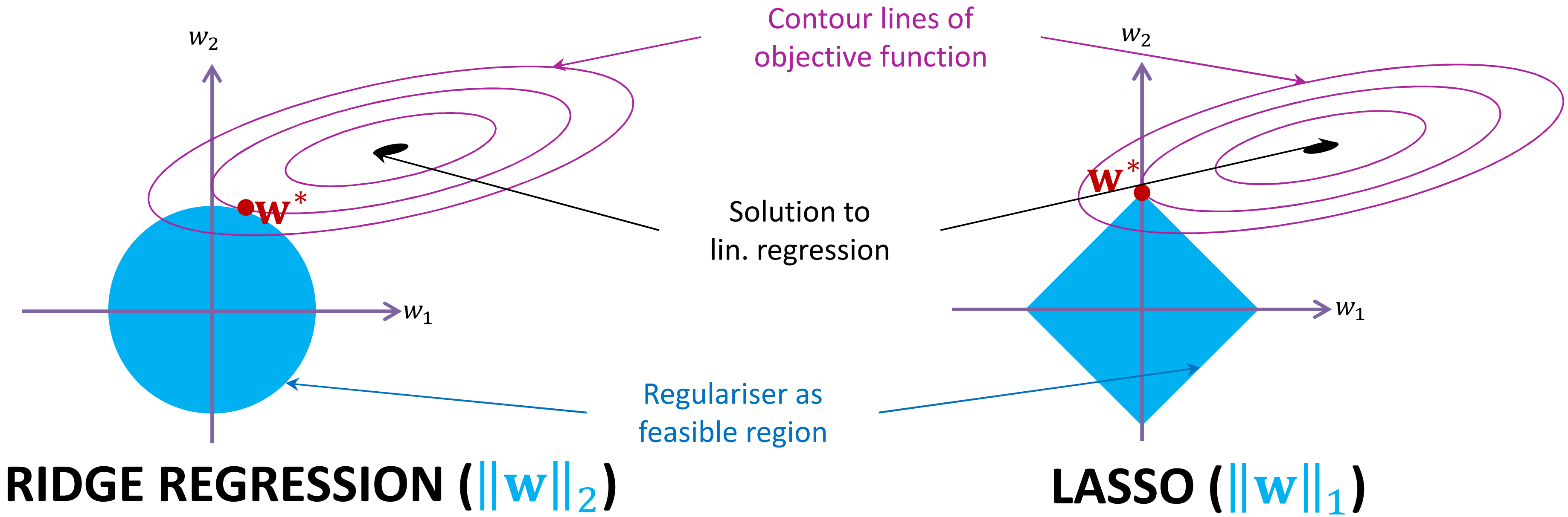
- Good for inference (lowers variance)
- Good computation ("convex" like a bowl)





Equivalent View: Regulariser as Constraint

$$\min_{\mathbf{w}} \sum_i (y_i - \mathbf{X}_i.\mathbf{w})^2 \ \text{s.t.}\ \|\mathbf{w}\|_2 \leq \mu$$



L_1-regularisation encourages solutions $\mathbf{w}^*$ to sit on axes

→ $\mathbf{w}^*$ will have components equal zero → $\mathbf{w}^*$ will be sparse!

Sparsity-Regularised Learning

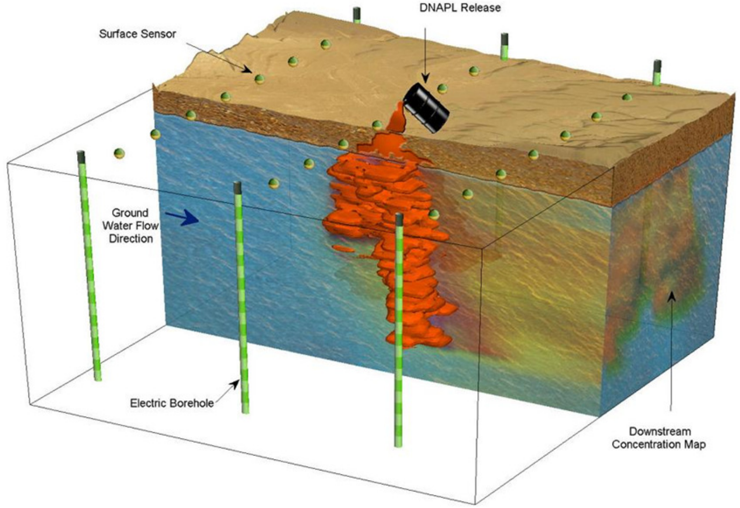
Lasso a special case of "compressed sensing"

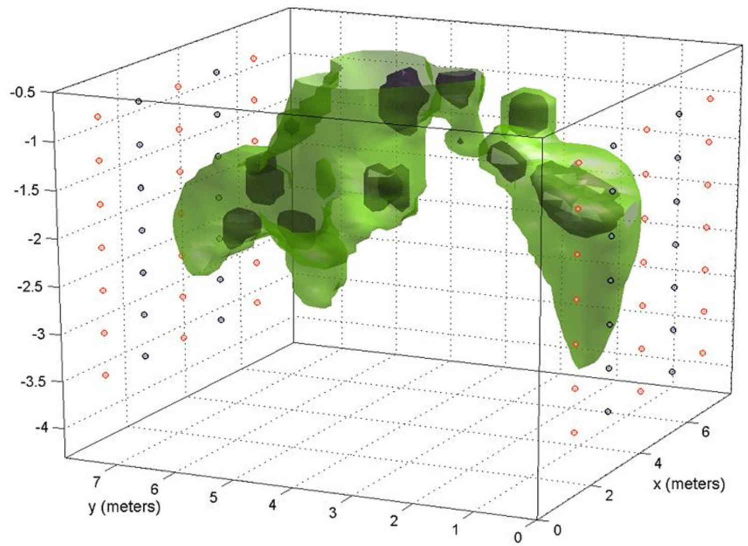
- Encourage sparsity through regulariser
- Many learners can be modified

State-of-the-art for high-dim. data

- Where *d>>n*
- Cannot hope to even have O(d) parameters
- Sparsity like simultaneous feature selection and learning

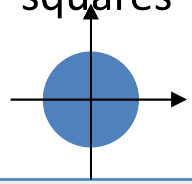
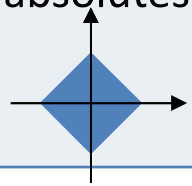
Many applications (e.g. tomography)





Aghasi et al. (2013)

Regularised Linear Regression

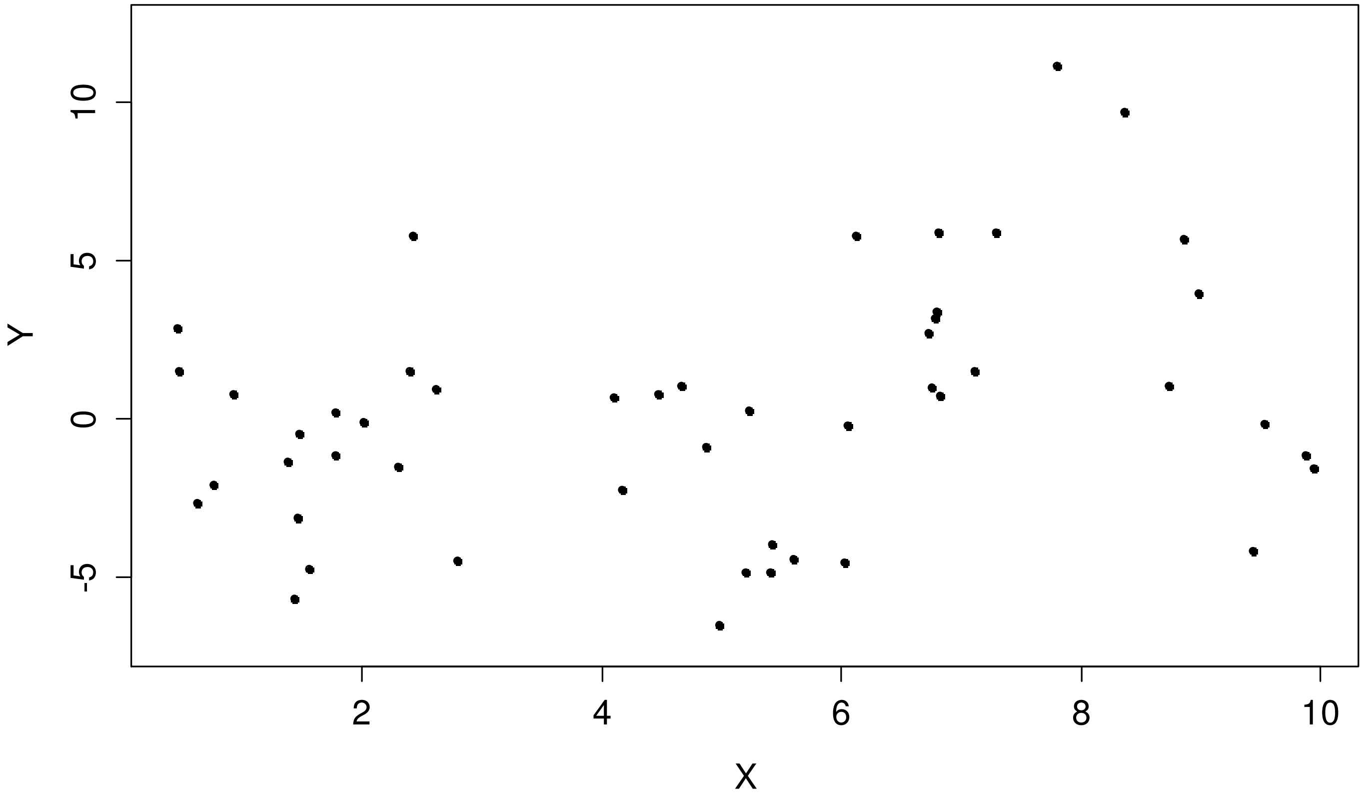
Algorithm	Minimises	Regulariser on w?	Notes
Linear regression	$\sum_{i=1}^{n}(y_i - \mathbf{X}_{i}.\mathbf{w})^2$	None	Solution is $(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$ if inverse exists
Ridge regression	$\sum_{i=1}^{n}(y_i - \mathbf{X}_{i}.\mathbf{w})^2 + \lambda\lVert\mathbf{w}\rVert_2{}^2$	Sum of squares 	Solution is $(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{y}$
Lasso	$\sum_{i=1}^{n}(y_i - \mathbf{X}_{i}.\mathbf{w})^2 + \lambda\lVert\mathbf{w}\rVert_1$	Sum of absolutes 	No closed-form, but solutions are sparse and suitable for high-dim data

Exercise: How would you do this for logistic regression?

Model Complexity and Regularisation

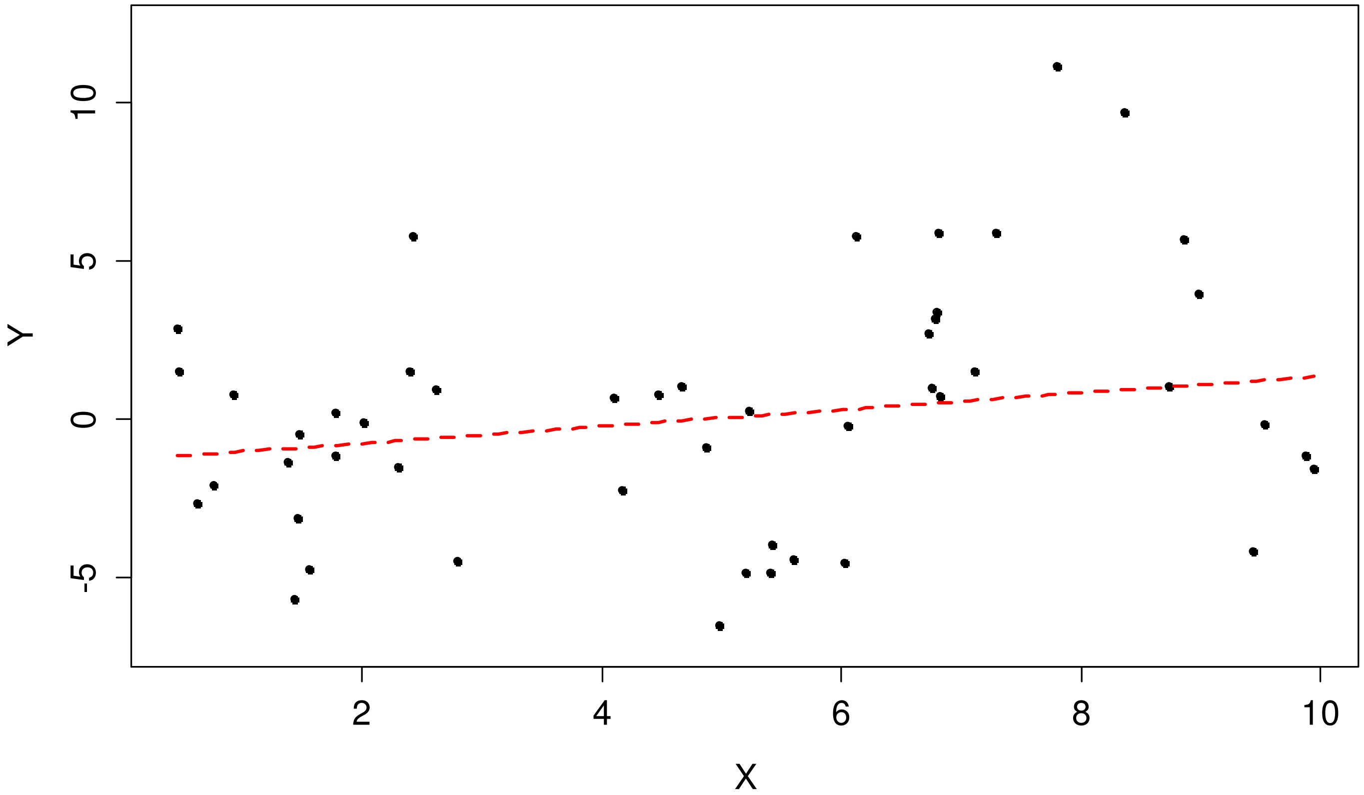
Model complexity measures "number of effective parameters". More complexity requires more data, lest we overfit. Limiting the "complexity" – regularisation – limits overfitting!

Example regression problem



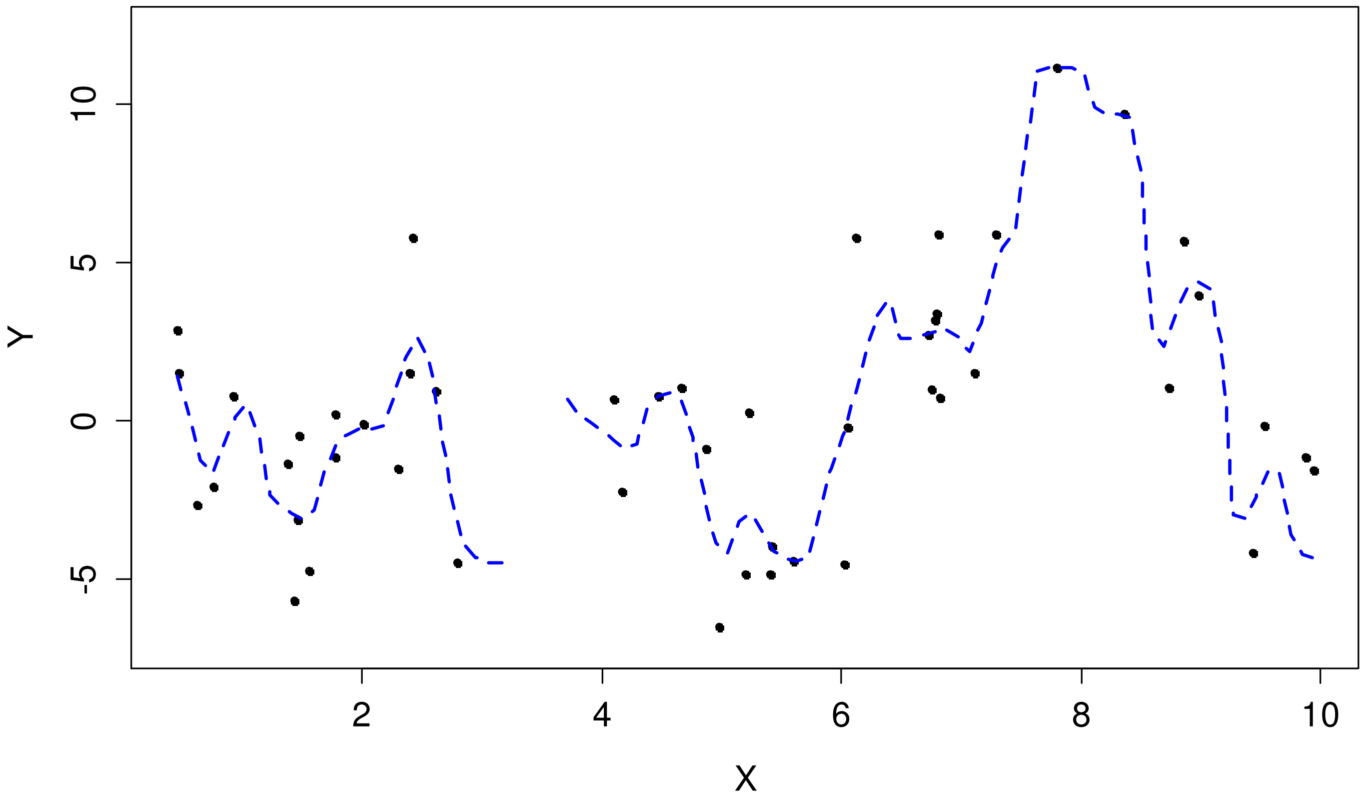
How complex a model should we use?

Underfitting (linear regression)



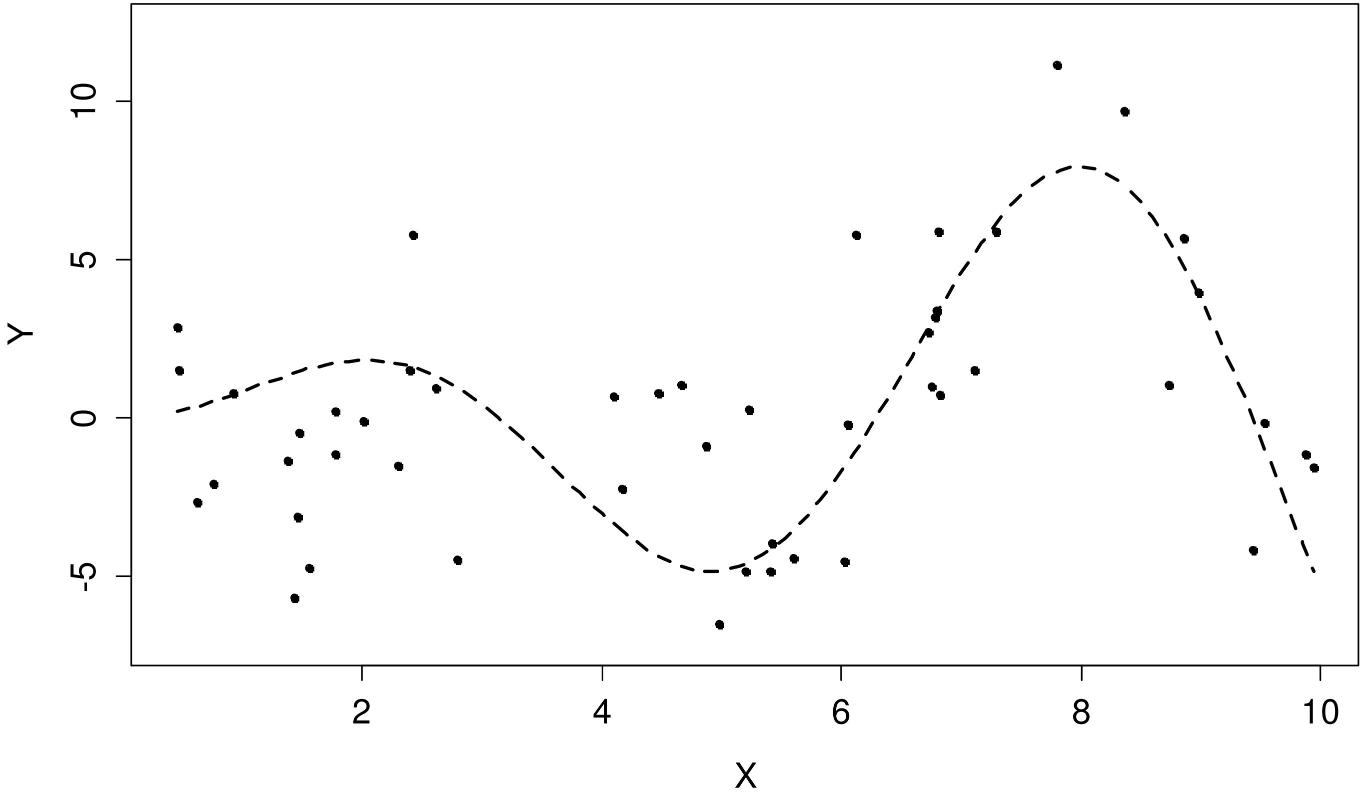
Model class Θ can be **too simple** to possibly fit true model.

Overfitting (non-parametric smoothing)



Model class Θ can be **so complex** it can fit true model + noise

Actual model ($x \sin x$)

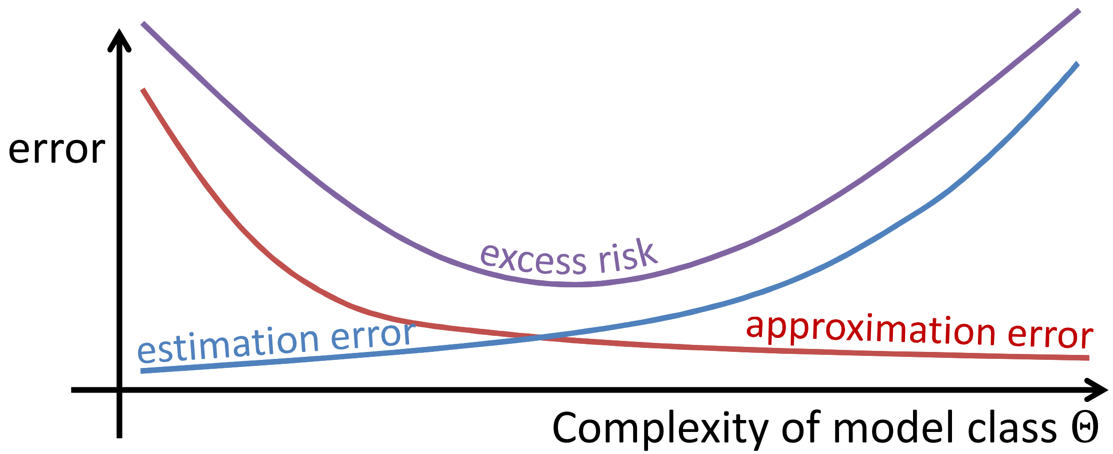


The **right model class** Θ will sacrifice some training error, for test error.

Model complexity

- Test error == Expected loss == Risk
- Best possible risk R^*? How far is estimate? $R(\hat{\theta}) - R^*$ Bayes risk

$$\underbrace{\left(R(\hat{\theta}) - \min_{\theta\in\Theta} R(\theta)\right)}_{\text{estimation error}} + \underbrace{\left(\min_{\theta\in\Theta} R(\theta) - R^*\right)}_{\text{approximation error}}$$



Inherent trade-off!

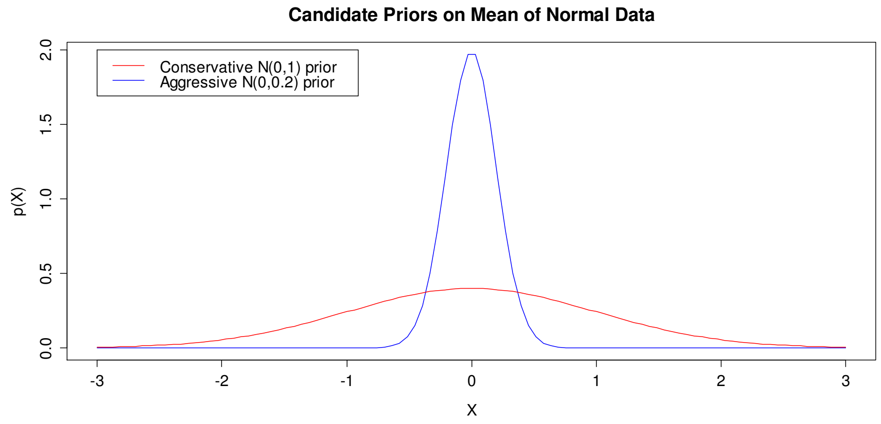
But how do we "vary" model complexity?

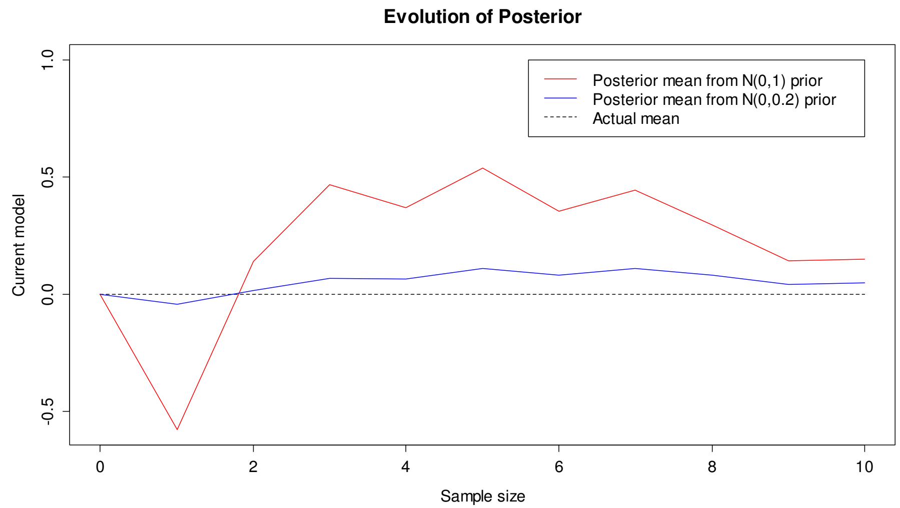
- **Regularise,** baby, regularise (change that λ parameter)
- Cross-validate to set amount of regularisation

1. Split training data into D_{train}, $D_{validation}$ sets
2. For each potential parameter value
   a) Train using parameter on D_{train}
   b) Test on $D_{validation}$
3. Pick parameter with best test score
4. Retrain using best parameter, on all data

One more slide: Bayesians regularise too!

- Know $X|\theta \sim N(\theta, 1)$, find θ
- Candidate priors
  * Conservative $\theta \sim N(0,1)$
  * Aggressive $\theta \sim N(0,0.2)$
- Train on observed $X_1,\ldots,X_{10}$
  * Really came from $\theta = 0$
  * -1.159, 1.578 1.451, -0.020, 1.385, -0.759, 1.061, -0.876, -1.244, 0.215
- More concentrated (less variable) priors regularise posteriors more





Summary

- Regularisation
  * What can go wrong with irrelevant features
  * Regularisation as conditioning ill-posed problems
  * What is model complexity? What happens with low/high?
  * How regularisation controls model complexity
  * Regularised linear regressors (ridge, lasso); logisticR too!
  * Priors as Bayesian regularisation