Machine Learning Ideas

ACTL3142 & ACTL5110 Statistical Machine Learning for Risk and Actuarial Applications

Overview

- The machine learning perspective
- Validation Set Approach
- Cross Validation (CV) Methods
  - Leave-One-Out Cross-Validation
  - k-Fold Cross-Validation
  - CV on classification problems
- Regularisation

Very important
Test error

The remainder of the term

- Week 5:
  - Choosing between models (cross-validation)
  - Avoiding overfitting (regularisation/shrinkage)
- Week 7: Splines and non-linear regressions (with a special guest lecturer Dr. P. Laub!) If more than 12, P Wong will give a big hint for final.
- Week 8: Week off. Happy Easter!
- Week 9: Decision trees
- Week 10: Unsupervised learning and exam preparation

Consultation
regular
times
+
Tuesday
1-2pm.

Machine Learning Perspective

The Two Cultures

GLM
logistic

	Statistical Learning	Machine Learning
Origin	Statistics	Computer Science
f(X)	Model	Algorithm
Emphasis	Interpretability, precision and uncertainty	Large scale application and prediction accuracy
Jargon	Parameters, estimation	Weights, learning
Confidence interval	Uncertainty of parameters	No notion of uncertainty
Assumptions	Explicit a priori assumption	No prior assumption, we learn from the data

See Breiman (2001) and Why a Mathematician, Statistician, & Machine Learner Solve the Same Problem Differently

Training and Test Errors

Questions to answer in ML project

You fit a few models, then ask:

1. **(Selection)** Which of these models is the best?
2. **(Future Performance)** How good should we expect the final model to be on unseen data?

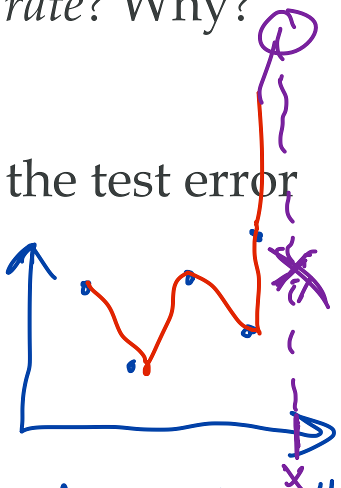
Test error = performance of model on data we have not seen before.

Training error < Test error

by defn, it's usually minimised

Training Error versus Test Error

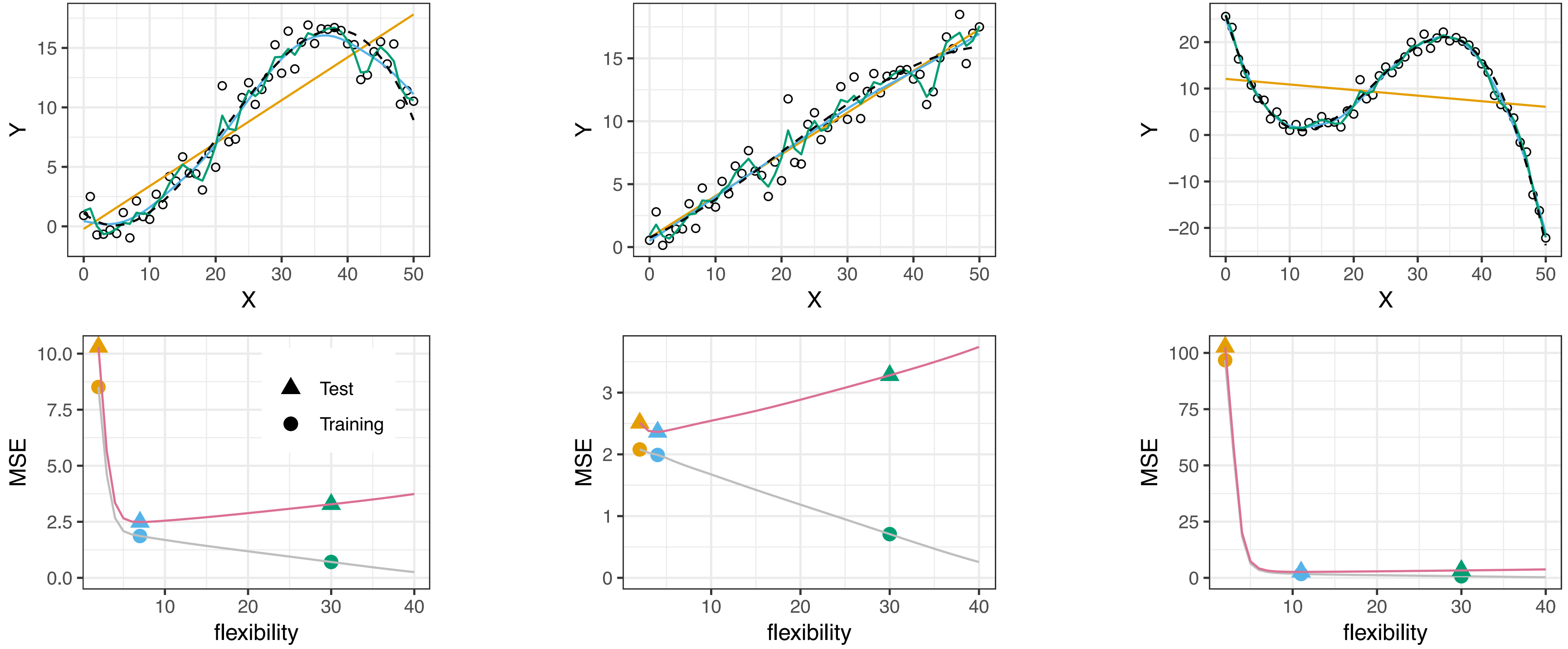
- How do we calculate the *test error rate*?
  - Easy if a designated test set is available ✓ Kaggle competition
  - But the designated test set is usually not available
- Can we use the *training error rate* to approximate the *test error rate*? Why?
  - Easy to calculate the training error
  - But the training error rate can dramatically underestimate the test error rate



- Sometimes test error can become dramatically worse as training error improves (interpolation models).

Simulated Examples

Training & test errors for three problems:



Cf. Module 1 slide 30.

Estimating the Test Error Rate

A number of techniques are available if there is no designated test set:

- Make a mathematical adjustment to the training error rate
  - e.g. C_p statistic, AIC and BIC
- Fit the model to a subset of the training observations
  - use the remaining training observations as the test set

$$AIC = \underbrace{-2\log(L)}_{\text{Training error}} + \overbrace{2k}^{\text{Adjustment}}$$

$k = \#$ params

Set aside a fraction for a test set

Set aside a fraction after *shuffling*.

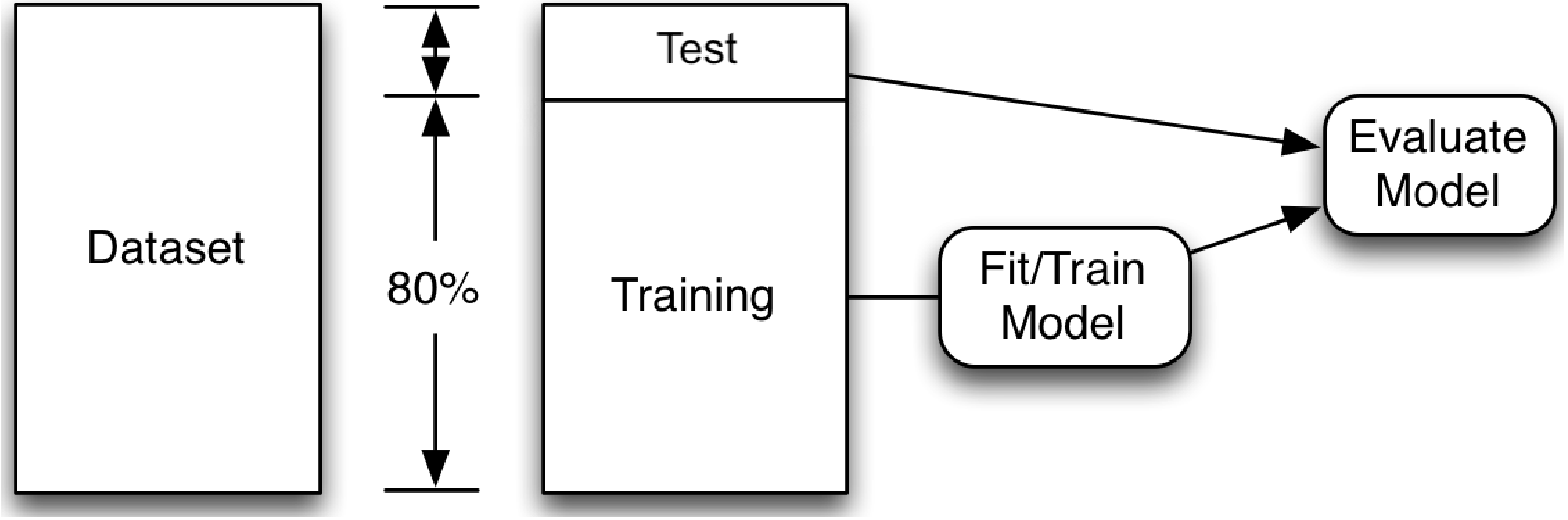
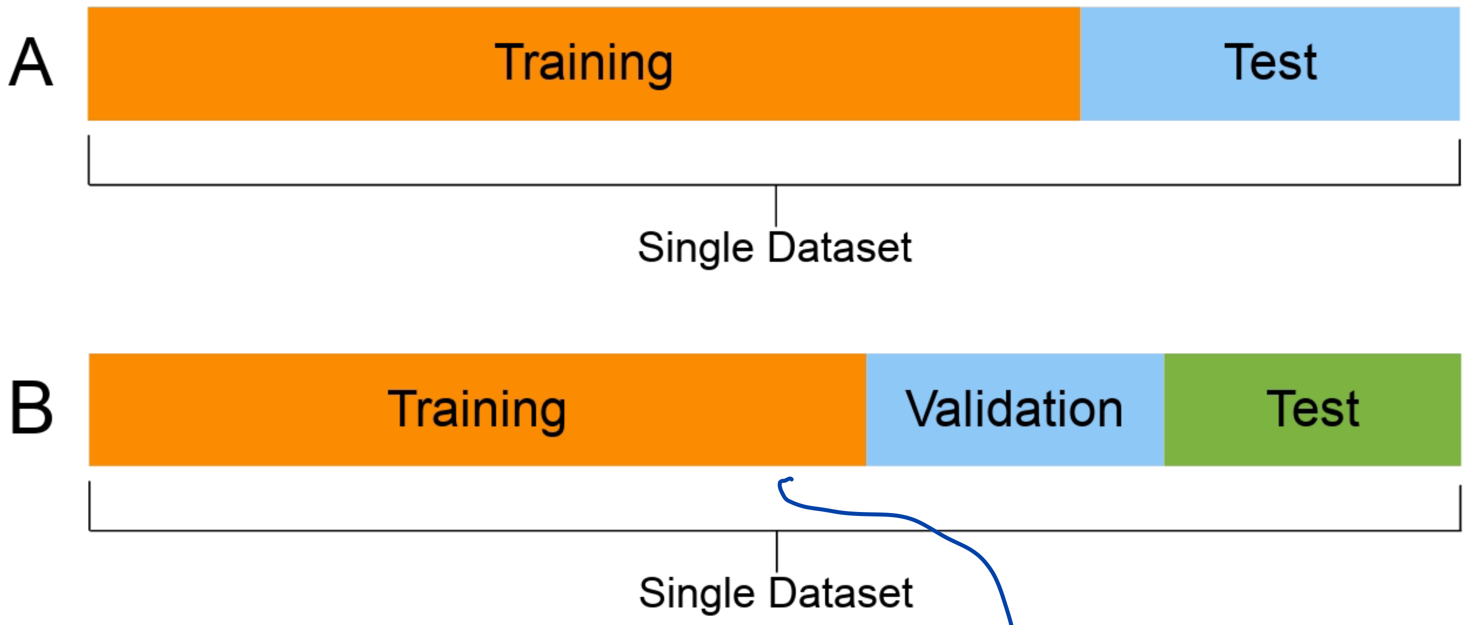


Illustration of a typical training/test split.

Adapted from: Heaton (2022), Applications of Deep Learning, Part 2.1: Introduction to Pandas.

Basic ML workflow


A

Training

Test

Single Dataset

B

Training

Validation

Test

Single Dataset

fit model on training, view performance on val. set as you tune, final check is on test.

Splitting the data.

1. For each model, fit it to the *training set*.
2. Compute the error for each model on the *validation set*.
3. Select the model with the lowest validation error.
4. Compute the error of the final model on the *test set*.

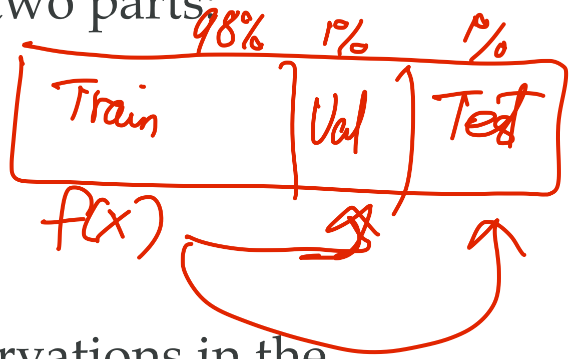
Source: Wikipedia.

Validation Set Approach

Idea

A simple approach to estimate the test error:

- (Randomly) divide the available set of observations into two parts:
  - a training set
  - a validation set or hold-out set (i.e. a ‘testing set’)
- Fit the model on the training set
- Use the fitted model to predict the responses for the observations in the validation set
- Validation set error rate provides an estimate of the test error rate



Discussion

Briefly suggest some benefits and drawbacks of the validation set approach.

Pros:

- Much simpler than the C_p/AIC/BIC methods (no σ^2 estimation).
- * No need to make any assumptions about the data.
- Popular in the deep learning world (lots of data). 98 - 1 - 1

Cons:

- Have less data to train on.
- Randomly putting data into training or validation set.

- Outliers or high leverage points are problematic

Validation Set Approach

> In statistics, sometimes we only use a single data set. To still be able to evaluate the performance of the developed prediction model on the same data, sophisticated methods have developed over a long period of time and are still in use in some parts of the statistics community. These methods account for the fact that the model saw the data during fitting and applied corrections to account for that. These methods include, for example, the Akaike Information Criterion (AIC) or the Bayesian Information Criterion (BIC). Don't get confused. If you have a validation set, you don't need these methods.

Source: Sic & Duerr (2020), Probabilistic Deep Learning, Chapter 5.

Auto Dataset Example

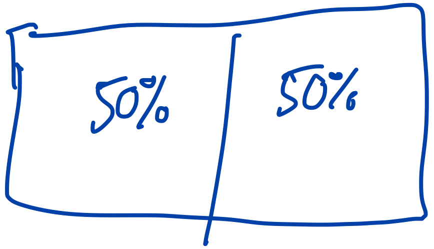
Auto dataset

We want to choose the optimal degree of polynomial n

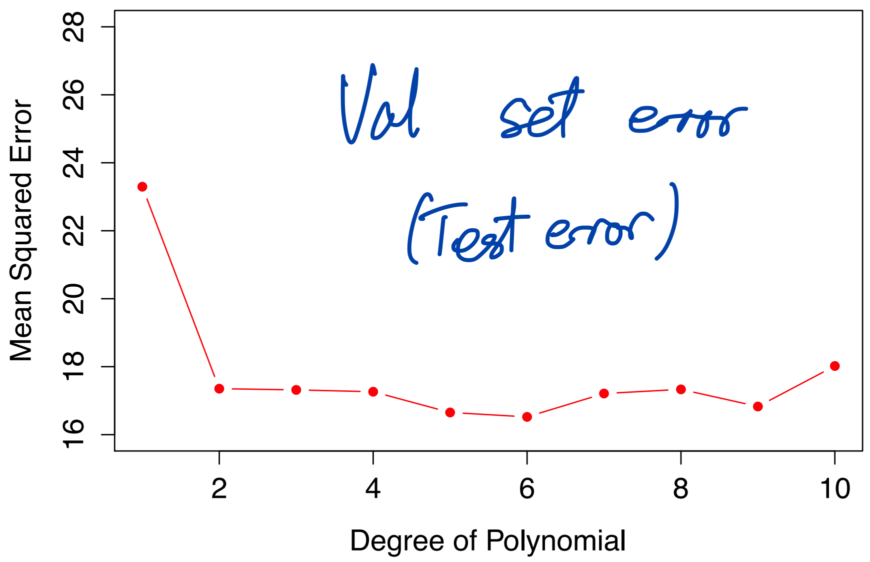
$$\texttt{mpg} = \sum_{i=0}^{n} \beta_i \cdot \texttt{horsepower}^i$$

- Randomly split 392 observations into two sets
  - a training set with 196 data points
  - a validation set with the remaining 196 observations
- Fit models of different degrees of polynomial on the training sample
- Predict $\widehat{\texttt{mgp}}$ using estimated β_i and **horsepower** in the validation sample
  - use MSE as a measure of validation set error

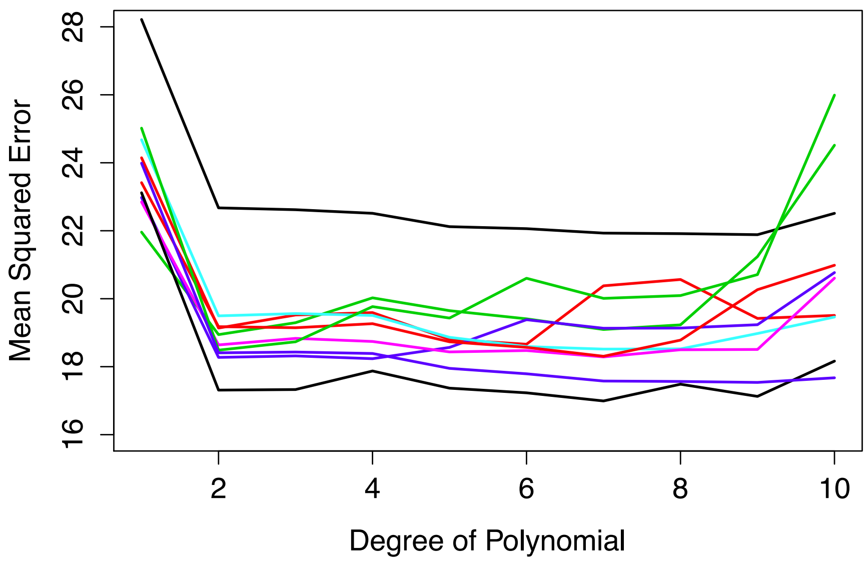
Auto dataset - Discussion



$$\texttt{mpg} = \sum_{i=0}^{n} \beta_i \cdot \texttt{horsepower}^i$$



Validation error estimates for one split



Validation error estimates for ten splits

Drawbacks of the validation set approach?

- Lot of variance in estimator of test error.

Drawbacks - Discussion

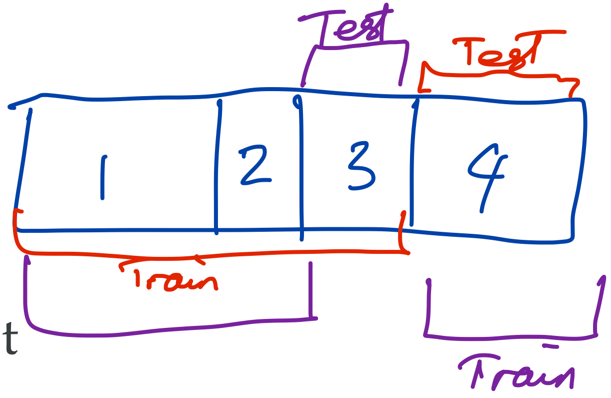
- The test error rate can be highly **variable**, depending on which observations are included in the training set.
  - Example: validation method repeated ten times on the Auto data set (previous slide)
- The validation error rate may tend to **overestimate** the test error rate:
  - only a subset of observations are used to fit the model,
  - statistical models tend to perform worse when trained on fewer observations.

10% Train	Val 90%

⇒ Suggestions to refine/improve the Validation Set approach?

Leave-One-Out Cross-Validation

LOOCV: Idea



- Split the set of observations into two parts:
  - a single observation (x_i, y_i) for the validation set
  - the remaining observations make up the training set: $\{(x_1, y_1), \cdots, (x_{i-1}, y_{i-1}), (x_{i+1}, y_{i+1}), \cdots, (x_n, y_n)\}$
- Fit the model on the $n-1$ training observations
- Predict $\hat{y}_i$ for the validation set
  - $\mathrm{MSE}_i = (y_i - \hat{y}_i)^2$ provides an approximately unbiased estimate for the test error
- Repeat the procedure n times
- The resulting LOOCV estimate for the test error is:

$$\mathrm{CV}_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \mathrm{MSE}_i$$

✓ Deterministic

$$\mathrm{Var}\left(\frac{X+Y}{n}\right) = \frac{1}{n^2}\left(\mathrm{Var}(X) + \mathrm{Var}(Y) + 2\,\mathrm{Cov}(X,Y)\right)$$

Validation set vs LOOCV - Discussion

Discuss how LOOCV performs relative to the Validation set approach - focusing in particular on the drawbacks of the Validation set approach identified previously.

Pros:

- LOOCV is less variable since we estimate the test error n times and average over that. Each observation is in both treated as a training data point ($n-1$ times) and as a validation data point.
- With LOOCV we train on many more observations ($n-1$) so the overestimation is minimised.

Cons:

- Training n times as many models as before!
- Each MSE$_i$ is highly correlated (training sets are almost identical).

- Variance of LOOCV is high
(New dataset)

LOOCV for least square linear models

- For linear (or polynomial) regression, LOOCV is extremely cheap to compute:

$$\boxed{\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2}$$

- where
  - $\hat{y}_i$ is the original least squares fit
  - h_i is the leverage defined in Chapter 3 (M2)
- This simplification generally does not apply in general

Note

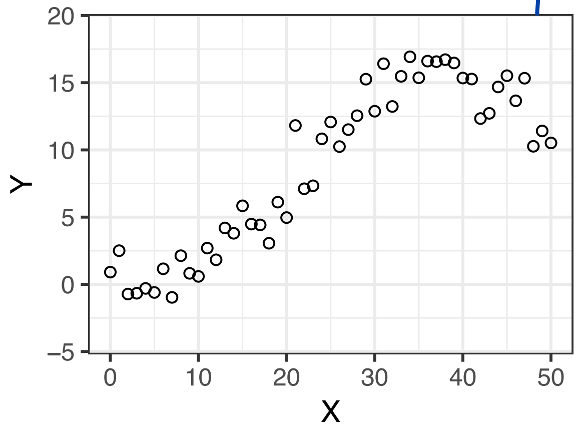
LOOCV and this simplificaiton is cute & historically important, but less practically so.

k-Fold Cross-Validation

Idea

Fold 1	Fold 2	.	.	.

- Randomly divided the set of observations into K groups, or folds of about equal size
  - the k^{th} fold is treated as a validation set
  - the remaining $K-1$ folds make up the training set
- Repeat K times resulting K estimates of the test error

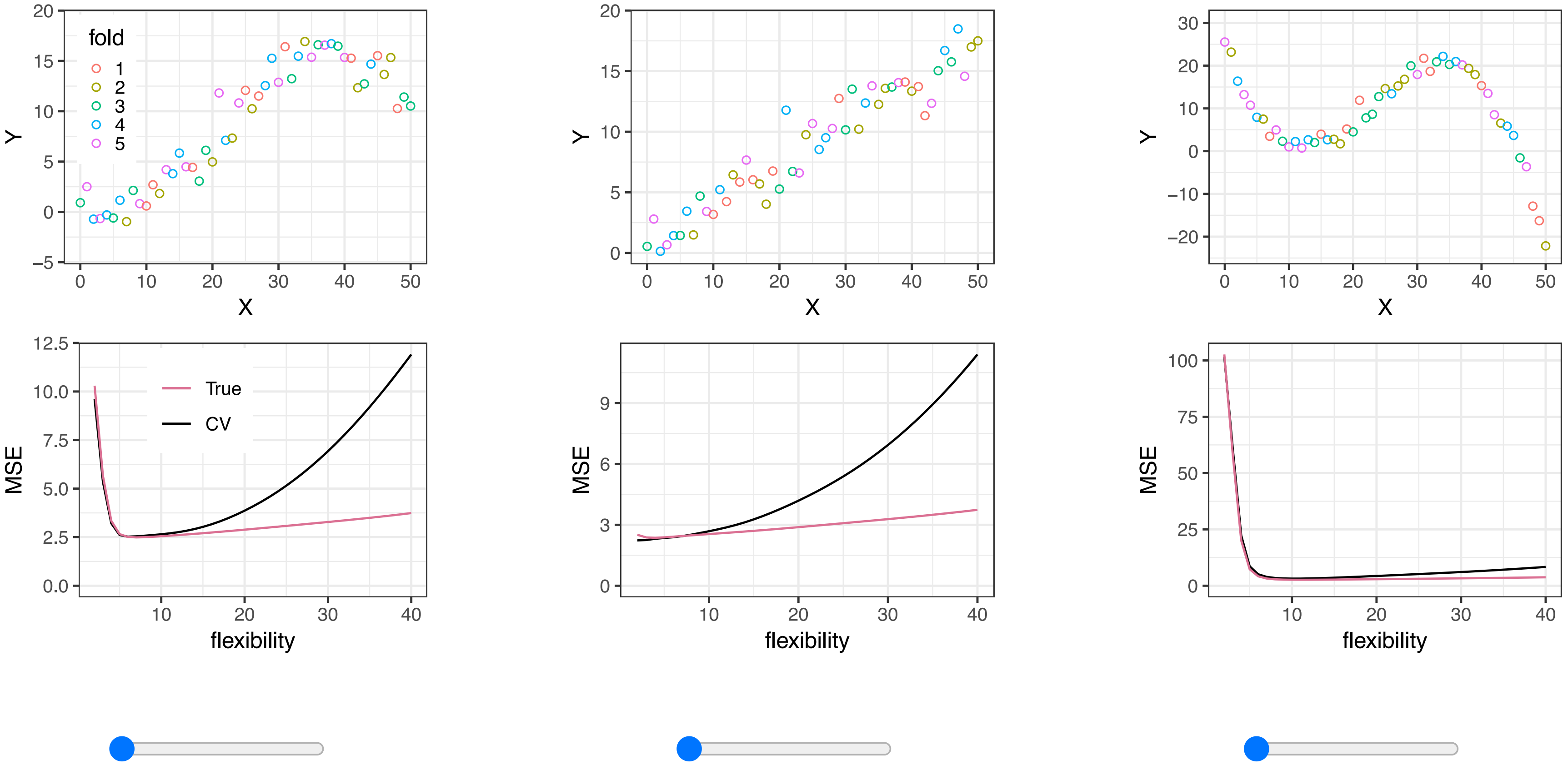


$$\mathrm{CV}_{(K)} = \frac{1}{K}\sum_{k=1}^{K}\mathrm{MSE}_k$$

$$= \frac{\sum (y_i - \hat{y}_i)^2}{n_k}$$

- In practice $K = 5$ or $K = 10$
- LOOCV is a special case of k-fold CV where $K = n$

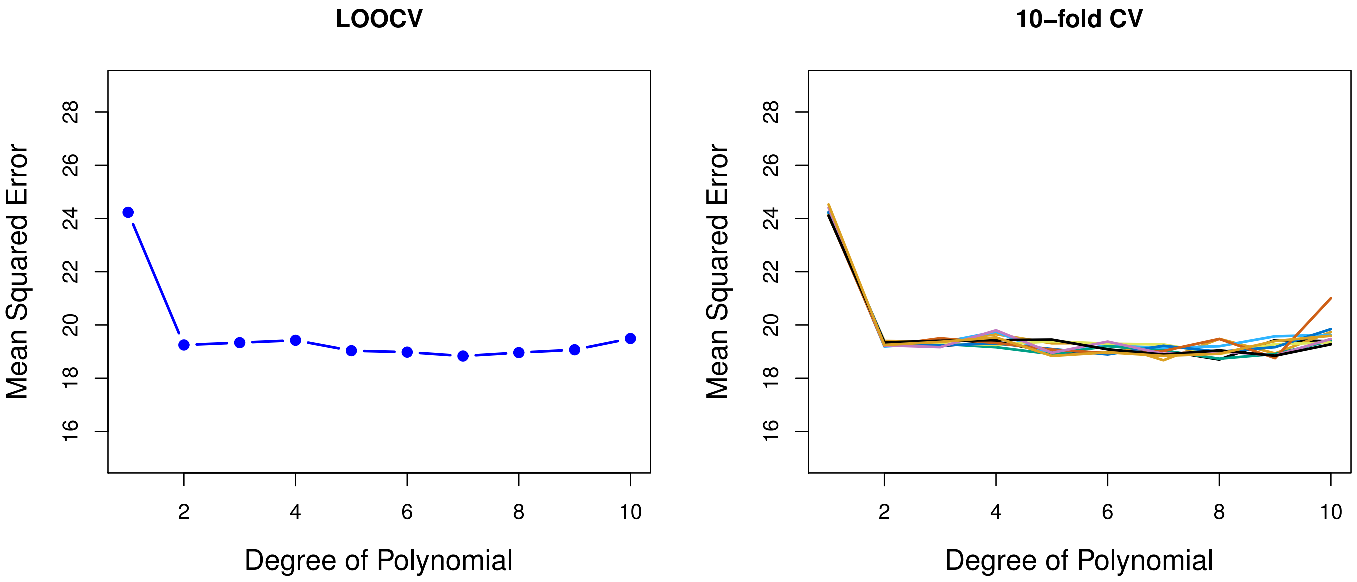
Simulated Examples



Example: **`Auto`** data set

We want to choose the optimal degree of polynomial, n

- Randomly split the observations into 10 folds
- Use **`cv.glm`**() function (part of the **`boot`** library) to calculate the cross validation error



(Left) LOOCV. (Right) 10-fold cross validation method repeated nine times.

LOOCV vs k-fold CV - Discussion

We can use LOOCV or k-fold CV to estimate the test error rate.

- Which estimation method gives more biased test error, and why?
- Which estimation method gives results of higher variances, and why?
- Which estimation method is faster?

- LOOCV has less bias since it is trained on a much larger dataset ↓ Bias
- LOOCV has higher variance since the individual MSE_k test error estimates are highly correlated ↑ variance
- k-fold fits fewer models, so is faster/cheaper

Cross-validation for classification

- Use percentage of misclassified observations instead of MSE
- Otherwise similar to the regression setting where the outcome Y is quantitative

$$\text{LOOCV error rate:} \quad \text{CV}_{(n)} = \frac{1}{n} \sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

$$k\text{-fold CV error rate:} \quad \text{CV}_{(K)} = \frac{1}{K} \sum_{k=1}^{K} \frac{\sum_i I(y_i \neq \hat{y}_i)}{n_k}$$

where n_k is the number of observations in the k^{th} fold

If n, we would be underestimating the test error

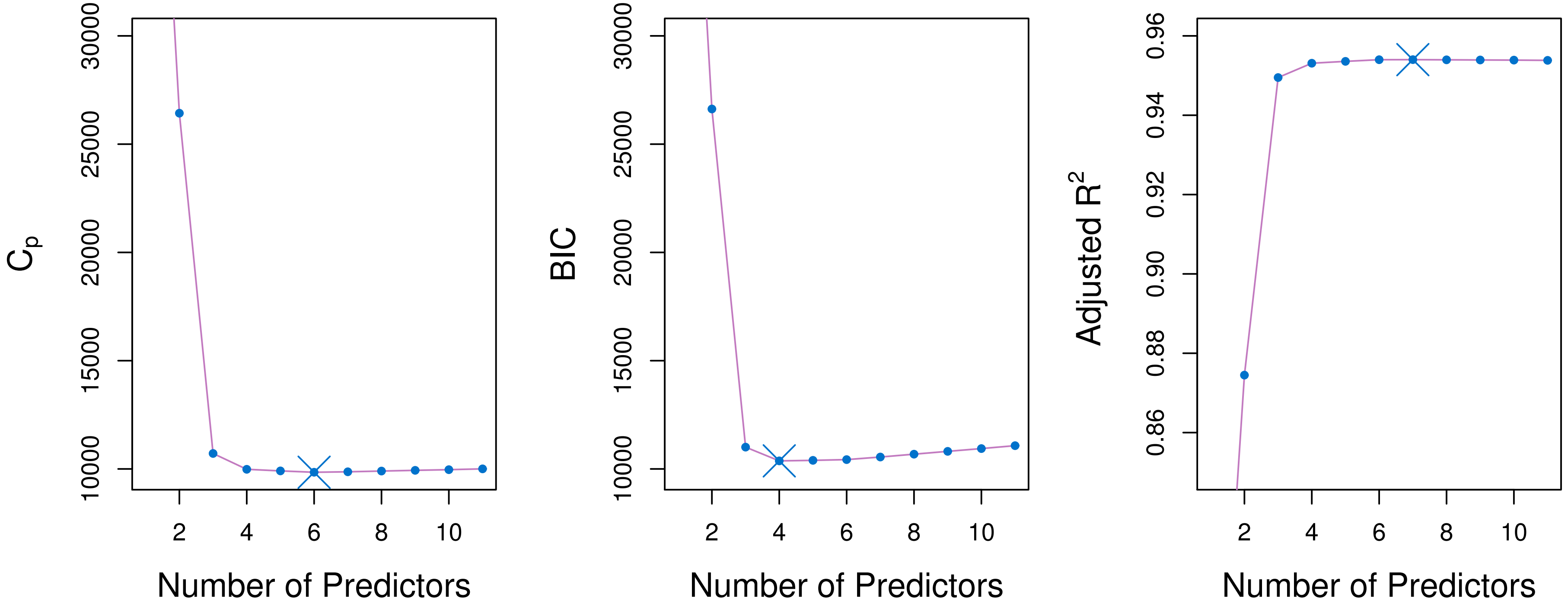
Validation and Cross-Validation

- Directly computes test error – assumes less about the data's underlying structure
- More versatile: does not rely on σ^2 being estimated, or knowing the model's degrees of freedom
- More computationally expensive, but not a major issue now - computers are fast (except for deep learning!)

- The data across folds / val / test sets all come from the same underlying distribution

`Credit` dataset I

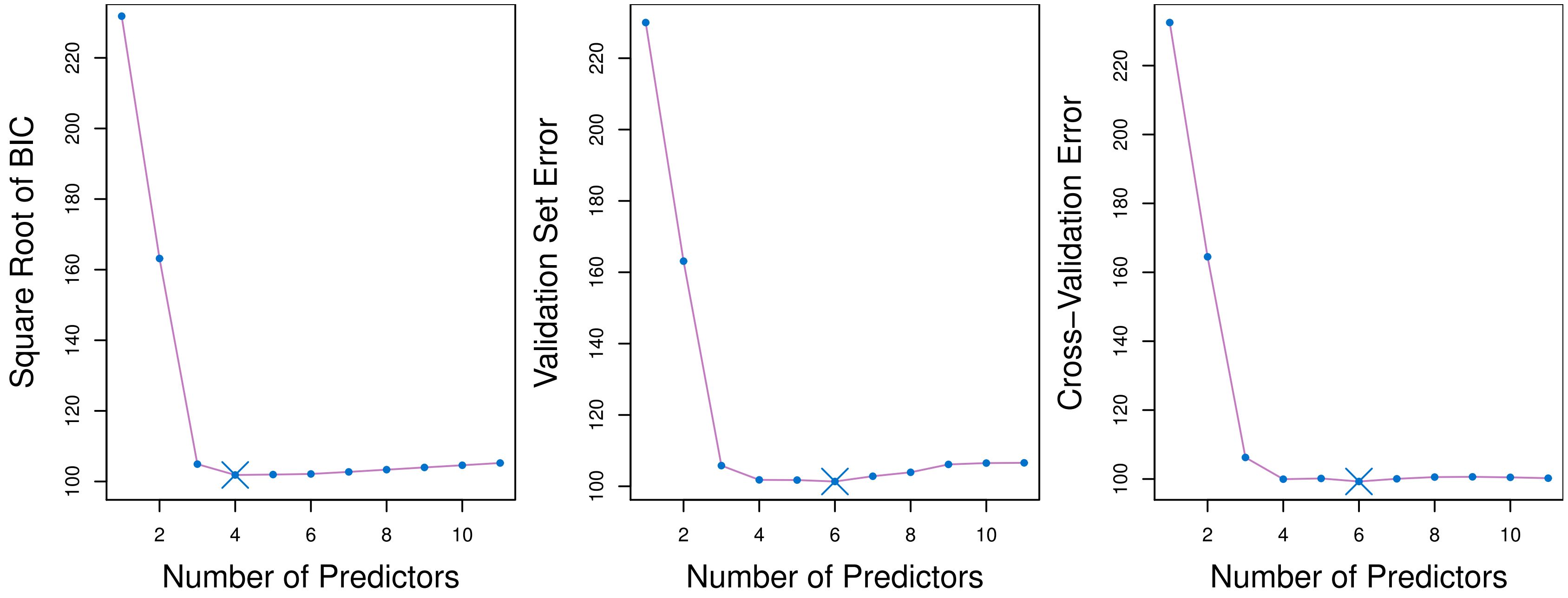
How do you determine the best number of predictors?



Comparing C_p, BIC & Adjusted R^2 for different number of predictors d.

Credit dataset II

How do you determine the best number of predictors?



Comparing root of BIC, validation set error and cross-validation error for different *d*.

Note

Here, can use the 'one standard error' rule to put a smaller model.

Regularisation

Remember BIC?

Bayesian Information Criterion

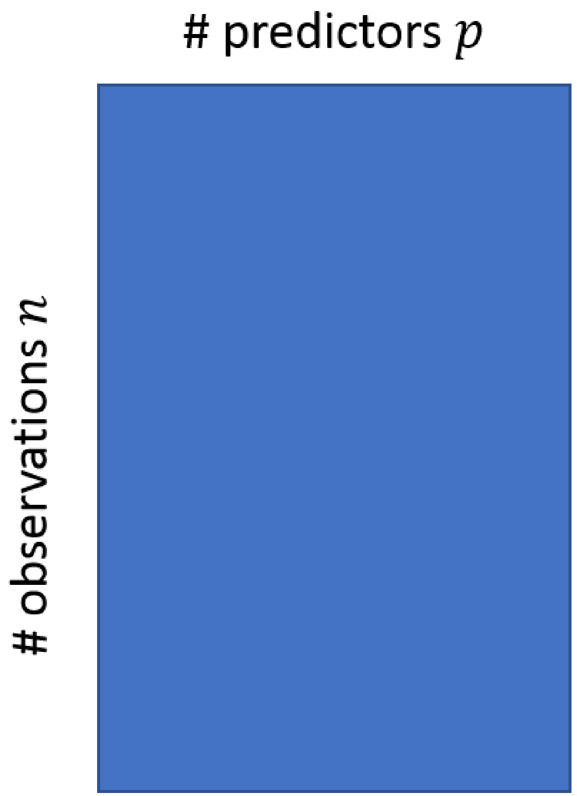
$$\text{BIC} = \frac{1}{n}\left(\underbrace{\text{RSS}}_{\text{Training error}} + \log(n)\, d\hat{\sigma}^2\right)$$

If we minimise this, we tradeoff

- prediction accuracy (RSS), and
- complexity (number of predictors)

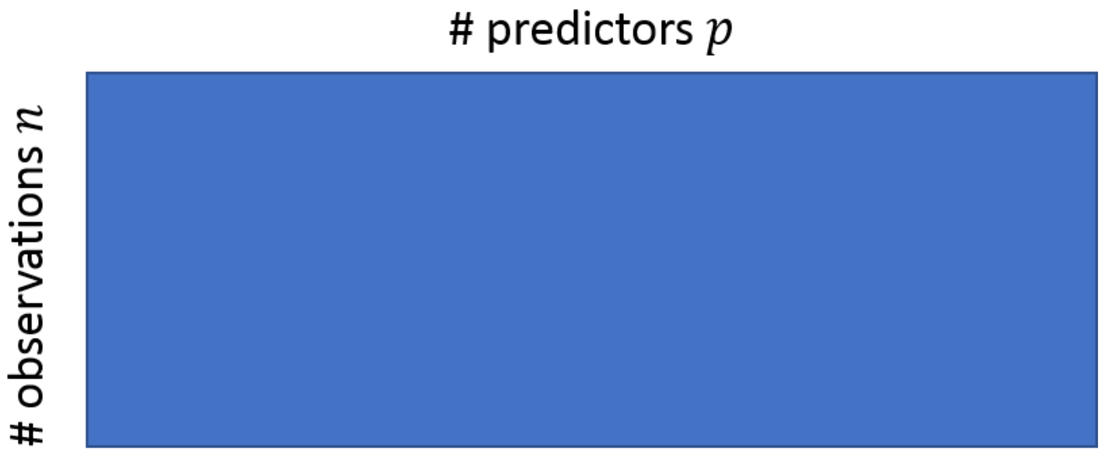
Traditional data vs. “Big” data

- Alzheimers



Traditional Data $(n >> p)$

- Traditional linear regression methods



“Big Data” $(p >> n)$

- Traditional methods break
- Need new approaches

If $p>>n$,
the training error
will be 0.

Shrinkage methods

- Alternative to subset selection is to fit model using all p predictors, but *constraints*, or *regularizes* the coefficients (='shrinks' the coefficients)
- Two types:
  - Ridge regression: pushes estimates towards zero, but all predictors included
  - Lasso regression: pushes estimates towards zero, some predictors excluded

Ridge regression I

Minimise on β:

$$\overbrace{\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)}^{\text{OLS}} + \overbrace{\lambda \sum_{j=1}^{p}}^{\text{Penalty}} \beta_j^2 = \text{RSS} + \lambda \sum_{j=1}^{p} \beta_j^2$$

- $\lambda \sum_{j=1}^{p} \beta_j^2$, the shrinkage penalty, restricts the growth of coefficient estimates
  - $\lambda \to \infty$: Parameter estimates heavily penalised, coefficients pushed to zero, model is $y_i = \hat{\beta}_0$
  - $\lambda = 0$: Parameter estimates not penalised at all, reduces to simple linear regression - obtain the best model which includes all parameters

Ridge regression II

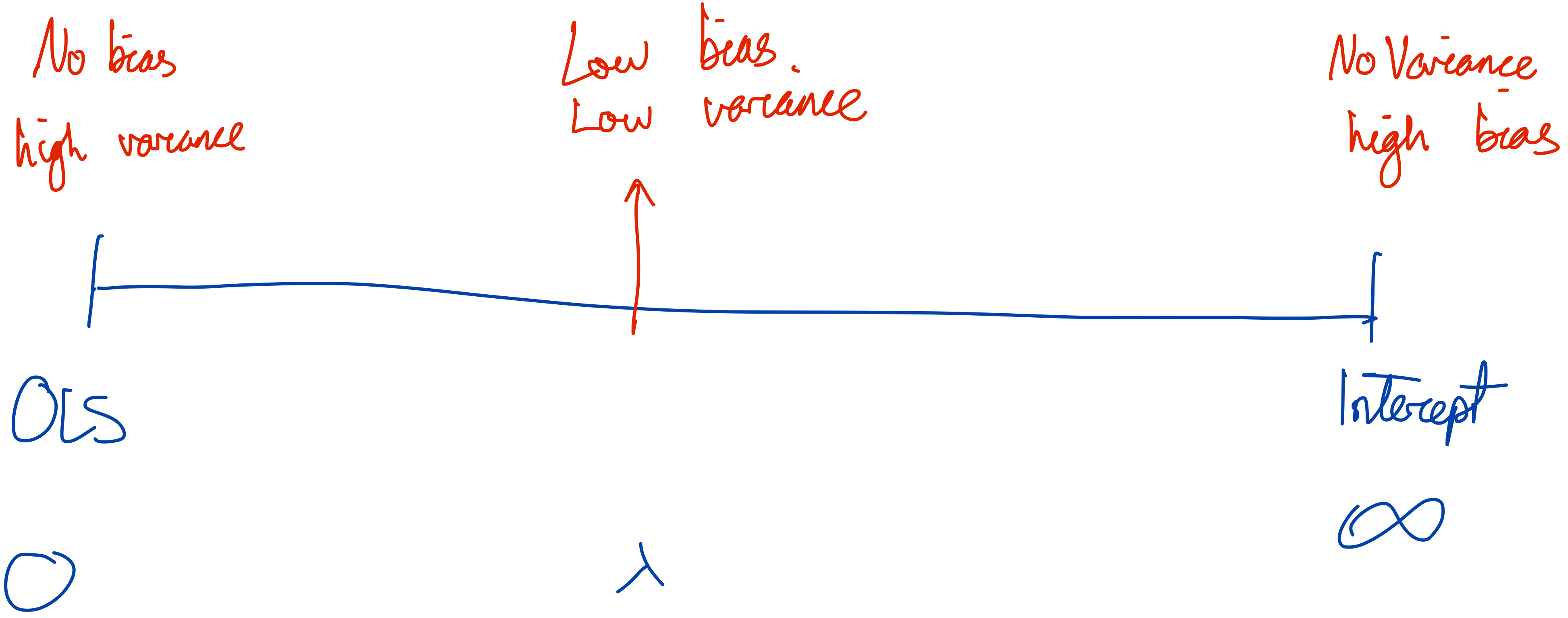
- Note that β_0's estimate is not penalised: Coefficient estimates are heavily scale-variant
- Need to standardise all predictors so their sample variance is 1:

$$x'_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2}}$$

Standardise first!

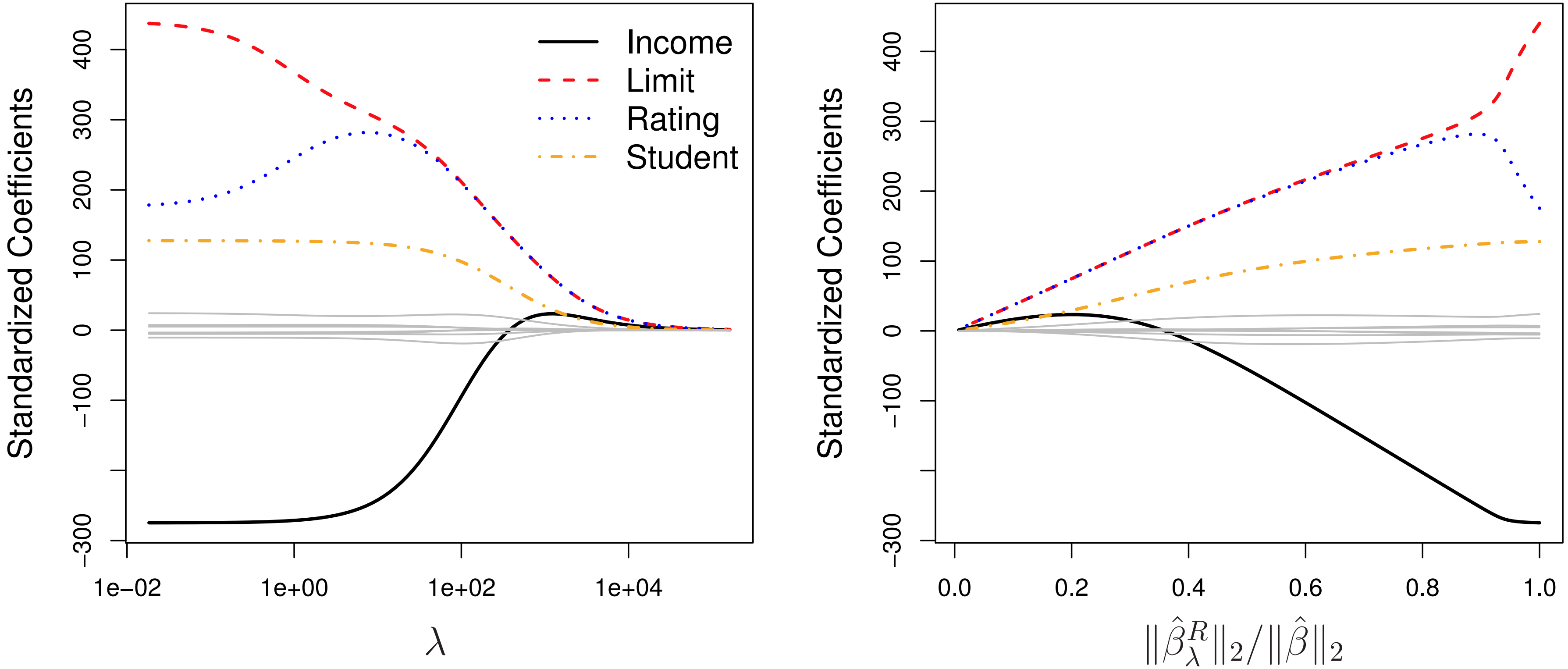
Tuning Parameter λ

- Different λ will give different estimates. Use cross-validation to find the best λ
- Lower λ leads to more flexible errors: lower bias but higher variance
- But λ can be changed: flexibility can be modified to get a model minimising MSE



Ridge regression on `Credit` dataset

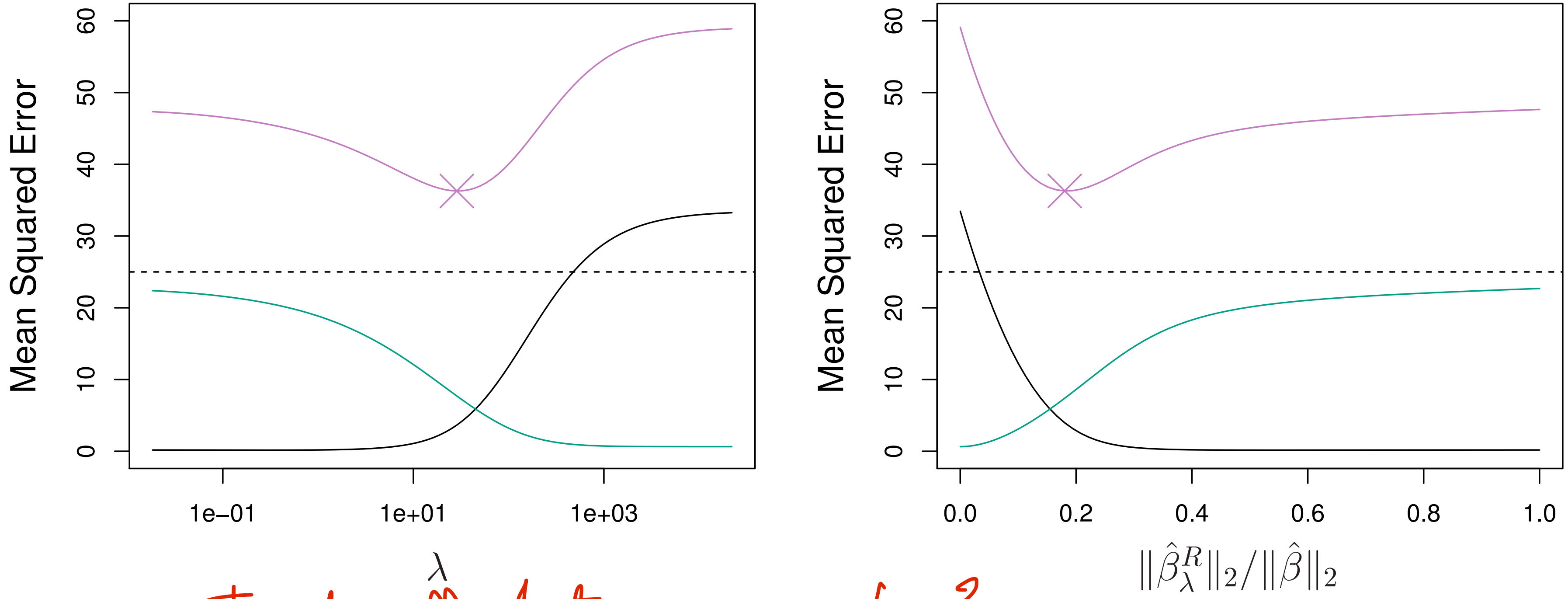
What do you observe / recommend as the best model?



The standardised ridge regression coefficients

Ridge regression on simulated dataset

What do you observe / recommend as the best model?



Trade off between var + bias^2 is worth it

Squared bias (black), variance (green), and test mean squared error (purple)

Ridge regression - comments

- Almost all parameters are included, and coefficients are generally low: difficult to interpret model
- Far more efficient than best-subset: only one model for each λ needs to be computed, calculating for *all* λ is almost identical to least squares estimates
- ✓✓ Performs better than least-squares where the relationship is linear, but the estimate variance is high

UNSW SYDNEY

Lasso regression

Minimise on β:

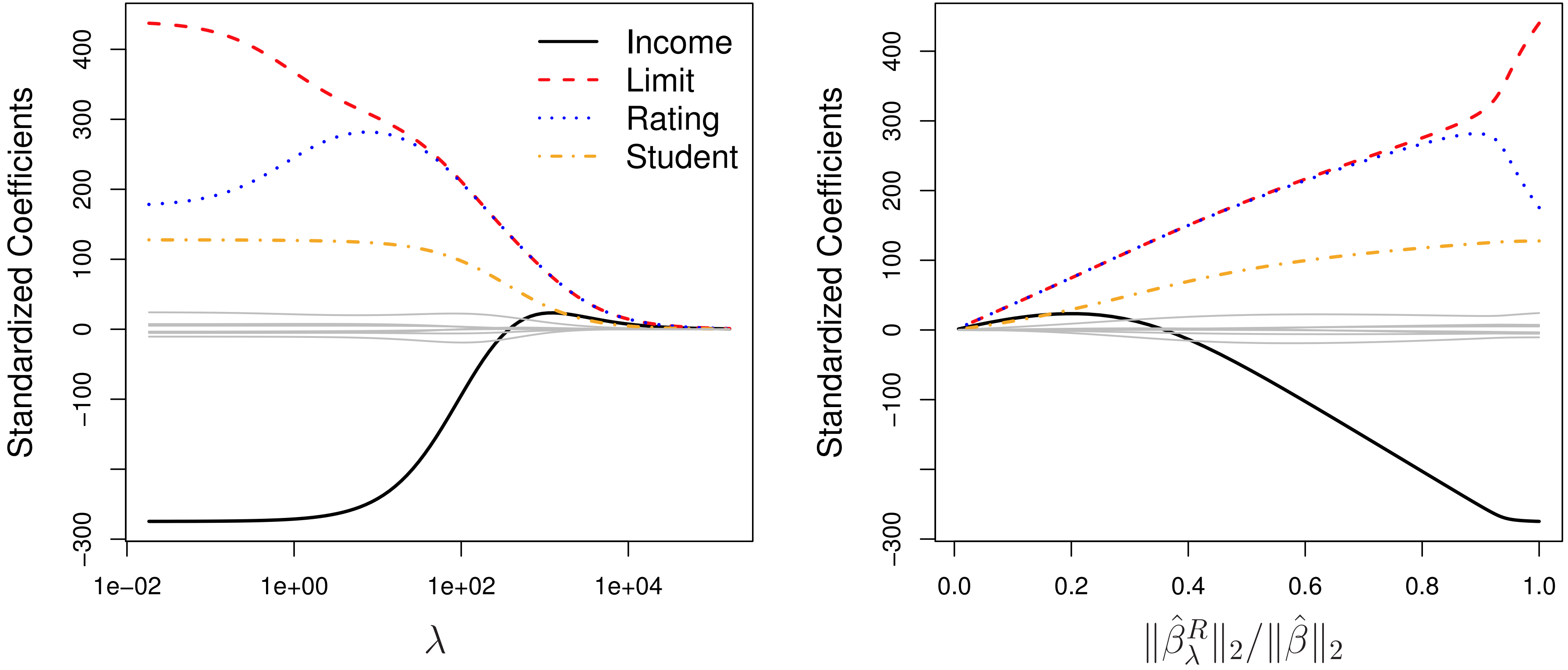
$$\overbrace{\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)}^{\text{OLS}} + \overbrace{\lambda\sum_{j=1}^{p}|\beta_j|}^{\text{Penalty}} = \text{RSS} + \lambda\sum_{j=1}^{p}\boxed{|\beta_j|}$$

, Abs, Ridge was square.

- Only difference: penalties placed on absolute value of coefficient estimates
- Can force some of them to exactly zero: significantly easier to interpret model
- Has the effect of also performing some variable selection, like best-subset

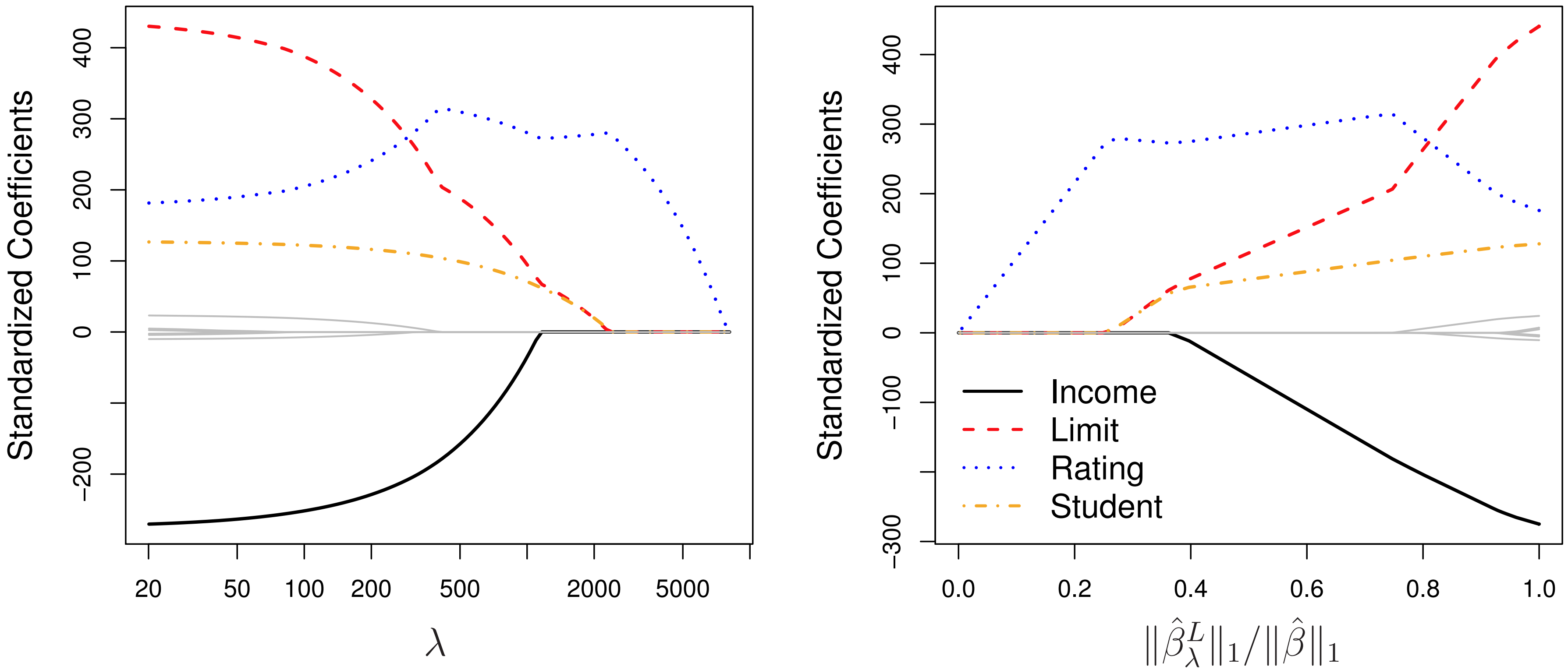
Ridge shrinks to 0, but is never 0.

Lasso vs Ridge on `Credit` data I



Coefficients in the ridge regression for various λ values

Lasso vs Ridge on `Credit` data II



Coefficients in the lasso regression for various λ values

Alternate formulation

Ridge regression: minimise MSE subject to $\min \sum (y_i - \hat{y}_i)^2$

$$\sum_{j=1}^{n} \beta_j^2 \leq s$$

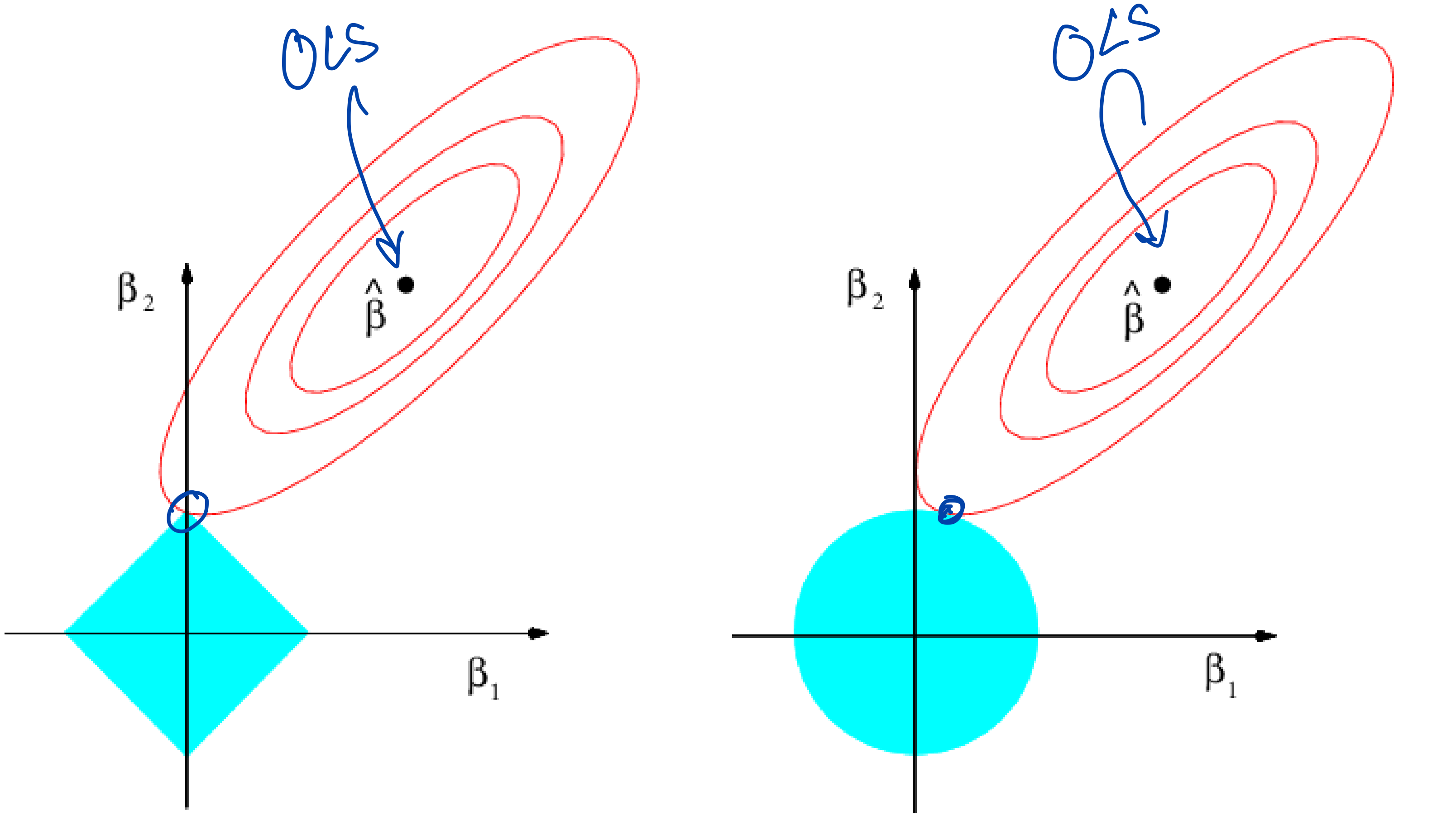
Lasso regression: minimise MSE subject to

$$\sum_{j=1}^{n} |\beta_j| \leq s$$

- Each method is assigns a "budget" to "spend" on the coefficient estimates
- The size of the "budget" is based on λ (ridge and lasso)
- This is like Lagrange multipliers (actually look up Karush–Kuhn–Tucker (KKT) conditions)

UNSW SYDNEY

Ridge vs Lasso: Some intuition



Contours of training MSE against the constraint regions for ridge & lasso.

Lasso leads to a pointier solution space: more likely to set parameters to zero.

Extensions

Ridge & lasso for GLMs

- Ridge and lasso can be applied to any GLM, the penalties are added to the negative log-likelihood

Elastic net penalty

$$\lambda\left[\alpha\sum_{j=1}^{p}|\beta_j| + (1-\alpha)\sum_{j=1}^{p}\beta_j^2\right]$$

- $\alpha = 1$ is lasso, $\alpha = 0$ is ridge
- $0 < \alpha < 1$ is elastic net, a compromise between ridge and lasso

Estimate α, λ by CV.

When to use what?

> So when should you use elastic net regression, or ridge, lasso, or plain linear regression (i.e., without any regularization)? It is almost always preferable to have at least a little bit of regularization, so generally you should avoid plain linear regression. Ridge is a good default, but if you suspect that only a few features are useful, you should prefer lasso or elastic net because they tend to reduce the useless features' weights down to zero, as discussed earlier. In general, elastic net is preferred over lasso because lasso may behave erratically when the number of features is greater than the number of training instances or when several features are strongly correlated.

Source: Géron (2022), Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd Edition, Chapter 4.

Elastic net in `R`

- `glmnet` package
- `alpha` parameter controls the mix of ridge and lasso
- `lambda` parameter controls the strength of the penalty
- `cv.glmnet` function performs cross-validation to find the best λ