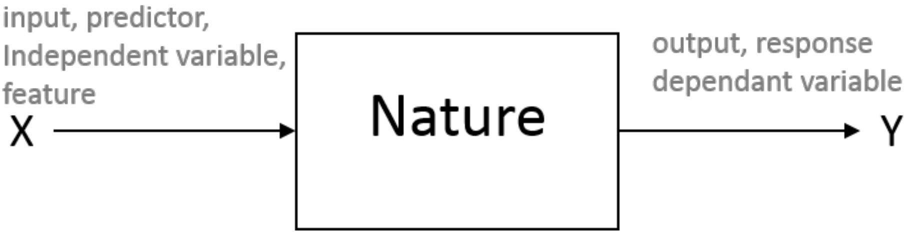
Foundations of Statistical and Machine Learning for Actuaries
Resampling, cross-validation and regularisation

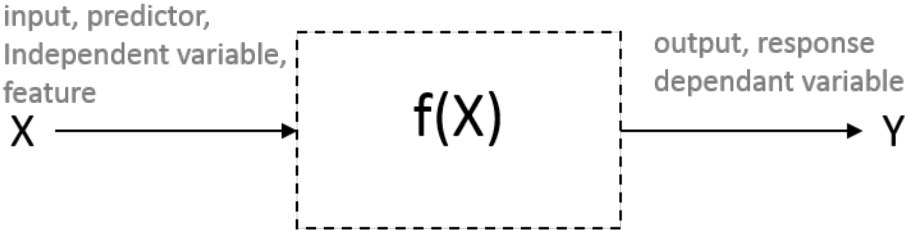
Edward (Jed) Frees, University of Wisconsin - Madison
Andres M. Villegas, University of New South Wales

July 2025

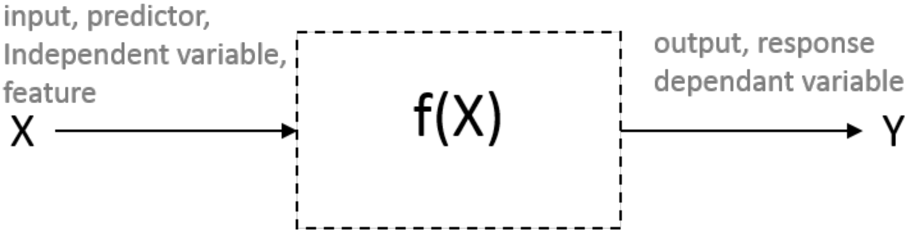
What is statistical (machine) learning?



What is statistical (machine) learning?



What is statistical (machine) learning?



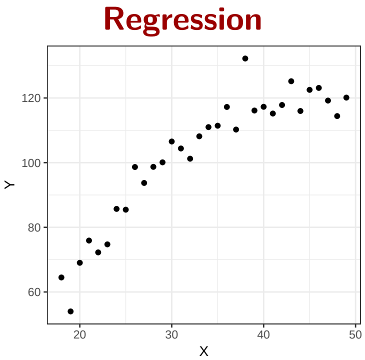
Prediction

- Predict outcomes of Y given X
  - What it means isn't as important, it just needs accurate predictions
- Models tend to be more complex

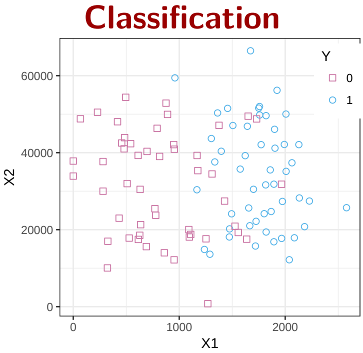
Inference

- Understand how Y is affected by X
- Which predictors do we add? How are they related?
- Models tend to be simpler

Regression vs. classification



- Y is quantitative, continuous
- Examples: Sales prediction, claim size prediction, stock price modelling



- Y is qualitative, discrete
- Examples: Fraud detection, face recognition, accident occurrence, death

More formally in regression we assume

$$Y = f(X) + \epsilon$$

- Y is the outcomes, response, target variable
- $X := (X_1, X_2, \ldots, X_p)$ are the features, inputs, predictors
- ϵ captures measurement error and other discrepancies

Our objective is to **find** an **appropriate** f for the problem at hand

How to estimate f?

Parametrics

- Make an assumption about the shape of f
- Problem reduced down to estimating a few parameters
  - Works fine with limited data, provided assumption is reasonable
- Assumption strong: tends to miss some signal

Non-parametric

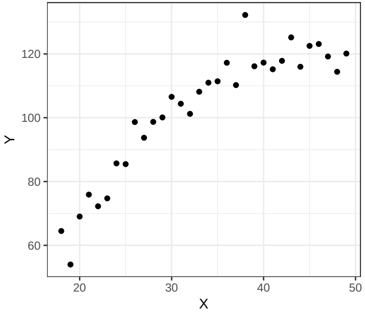
- Make no assumption about f's shape
- Involves estimating a lot of "parameters"
- Need lots of data
- Assumption weak: tends to incorporate some noise
- Be particularly careful re the risk of overfitting

Parametrics example: Linear regression

Approximately a linear relationship between X and Y

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

- The model is specified in terms of $p + 1$ parameters β_0, β_1, ..., β_p
  - Use (training) data to produce estimates $\hat{\beta}_0$, $\hat{\beta}_1$, ..., $\hat{\beta}_p$
  - **Almost never correct**, but serves as a good and interpretable approximation.

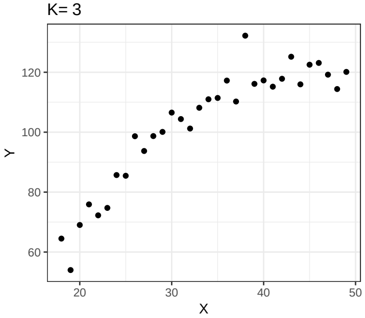


Non-parametrics example: K-nearest neighbours

KNN is one of the simplest non-parametric approaches

$$\hat{f}(x_0) = \frac{1}{K} \sum_{x_i \in \mathcal{N}_0} y_i$$

- can be pretty good for small p and large data sets (big N)
- need to choose the size of the value of K
  - we will discuss other smoother versions such as local linear regression and splines in session 2

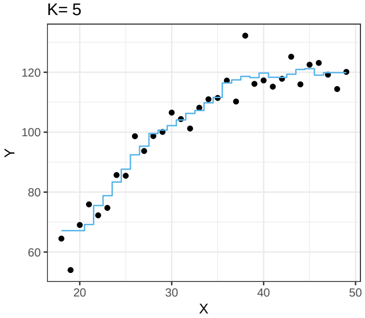


Non-parametrics example: K-nearest neighbours

KNN is one of the simplest non-parametric approaches

$$\hat{f}(x_0) = \frac{1}{K} \sum_{x_i \in \mathcal{N}_0} y_i$$

- can be pretty good for small p and large data sets (big N)
- need to choose the size of the value of K
  - we will discuss other smoother versions such as local linear regression and splines in session 2

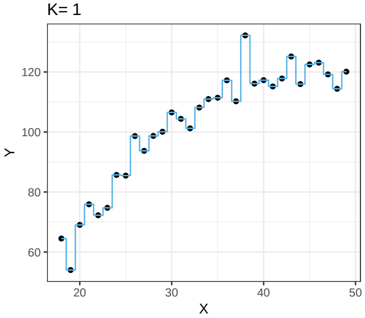


Non-parametrics example: K-nearest neighbours

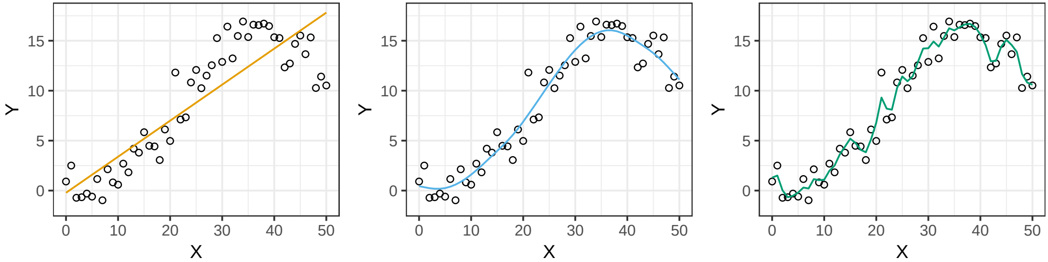
KNN is one of the simplest non-parametric approaches

$$\hat{f}(x_0) = \frac{1}{K} \sum_{x_i \in \mathcal{N}_0} y_i$$

- can be pretty good for small p and large data sets (big N)
- need to choose the size of the value of K
  - we will discuss other smoother versions such as local linear regression and splines in session 2



How to choose f?



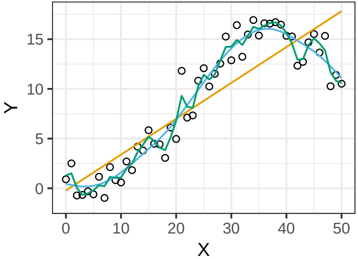
How do we decide which is the best model?

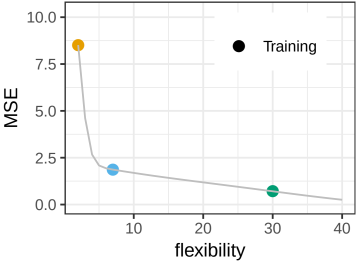
Assessing model accuracy

We fit the model $\hat{f}(x)$ to some **training** data $Tr = \{x_i, y_i\}_{i=1}^n$.

- We can compute the Training Mean Squared Error

$$MSE_{Tr} = \frac{1}{n} \sum_{i \in Tr} (y_i - \hat{f}(x_i))^2$$





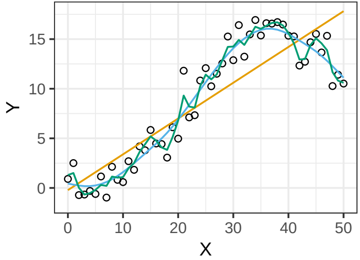
Assessing model accuracy

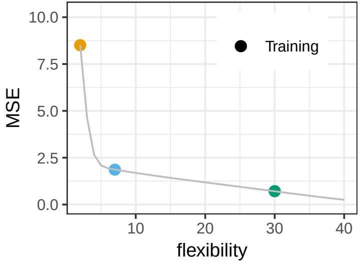
We fit the model $\hat{f}(x)$ to some **training** data $Tr = \{x_i, y_i\}_{i=1}^n$.

- We can compute the Training Mean Squared Error

$$MSE_{Tr} = \frac{1}{n} \sum_{i \in Tr} (y_i - \hat{f}(x_i))^2$$

This tends to be biased to more overfit models!





Assessing model accuracy

We fit the model $\hat{f}(x)$ to some **training** data $Tr = \{x_i, y_i\}_{i=1}^{n}$.

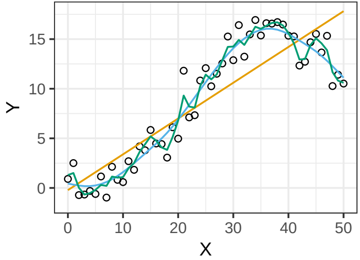
- We can compute the Training Mean Squared Error

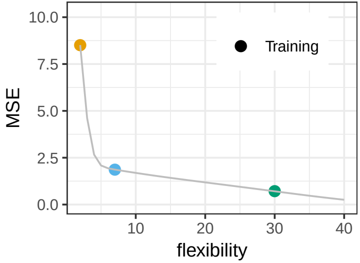
$$MSE_{Tr} = \frac{1}{n} \sum_{i \in Tr} (y_i - \hat{f}(x_i))^2$$

  This tends to be biased to more overfit models!
  We should instead use some fresh **test** data $Te = \{x_i, y_i\}_{i=1}^{m}$.

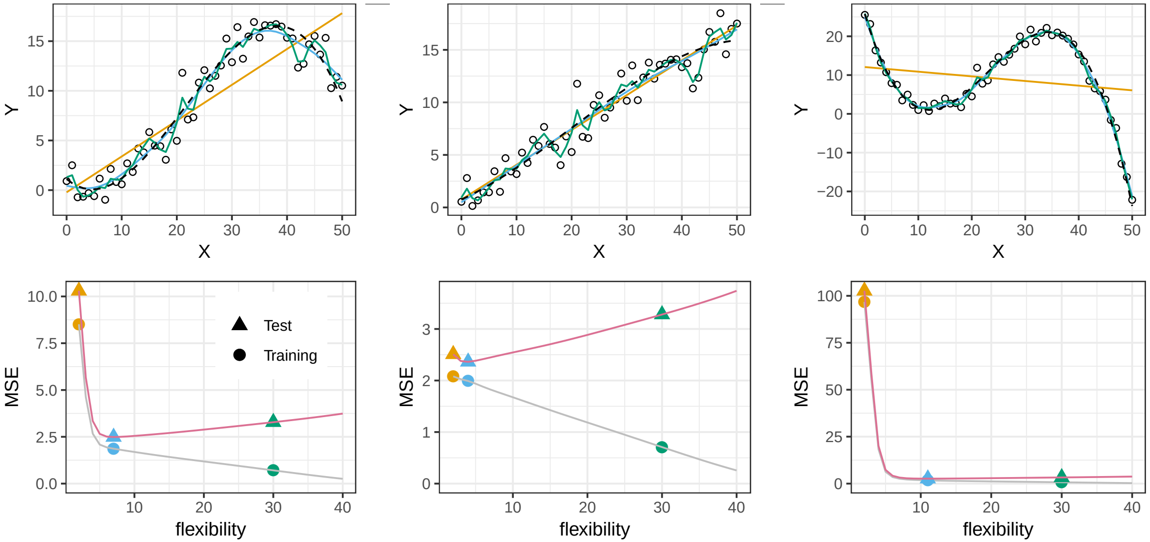
-

$$MSE_{Te} = \frac{1}{m} \sum_{i \in Te} (y_i - \hat{f}(x_i))^2$$





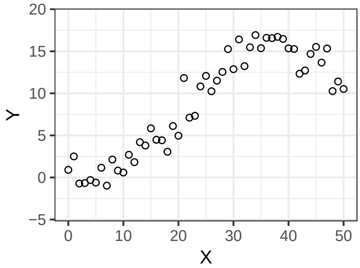
Assessing model accuracy



How do we calculate the test error?

1. The best solution is to use a large designated test set
   - Often not available
2. Make a mathematical adjustment to the training error rate
   - e.g. Cp statistic, AIC and BIC
3. Fit the model to a subset of the training observations
   - Use the remaining training observations as the test set

k-fold Cross-validation

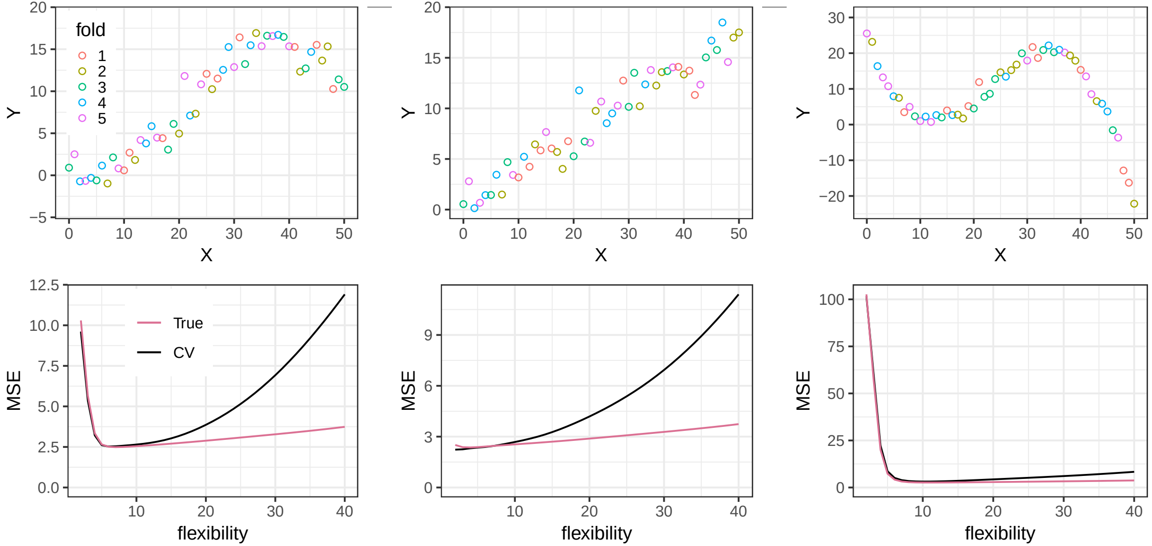


- Randomly divided the set of observations into K groups, or folds of approximately equal size
- the k^{th} fold is treated as a validation set
- the remaining $K-1$ folds make up the training set
- Repeat K times resulting K estimates of the test error

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

  - In practice $K = 5$ or $K = 10$

k-fold Cross-validation



Summary of key concepts

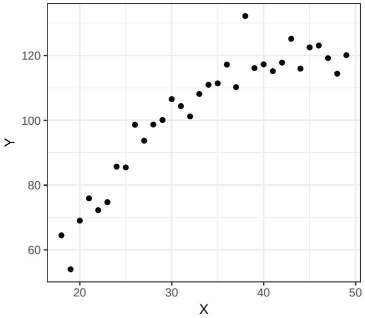
We have discussed key concepts in statistical/machine Learning

- Supervised learning vs. Unsupervised Learning
- Prediction vs. Inference
- Regression vs. Classification
- Parametric Vs. Non-Parametric
- Training MSE vs. Test MSE
- Cross-Validation

Supervised learning: regression

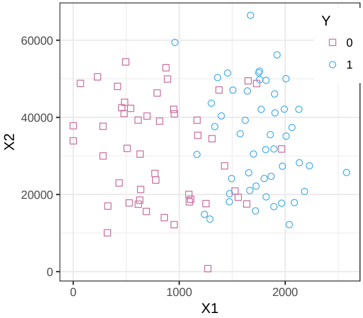
Regression vs. classification

Regression



- Y is quantitative, continuous
- Examples: Sales prediction, claim size prediction, stock price modelling

Classification



- Y is qualitative, discrete
- Examples: Fraud detection, face recognition, accident occurrence, death

More formally in regression we assume

$$Y = f(X) + \epsilon$$

- Y is the outcomes, response, target variable
- $X := (X_1, X_2, \ldots, X_p)$ are the features, inputs, predictors
- ϵ captures measurement error and other discrepancies

Our objective is to **find** an **appropriate** f for the problem at hand

Can we predict house prices?

Source: http://www.abc.net.au/news/2018-03-17/how-to-win-at-house-auction/9547166

Output (Y):

- House price

Input (X):

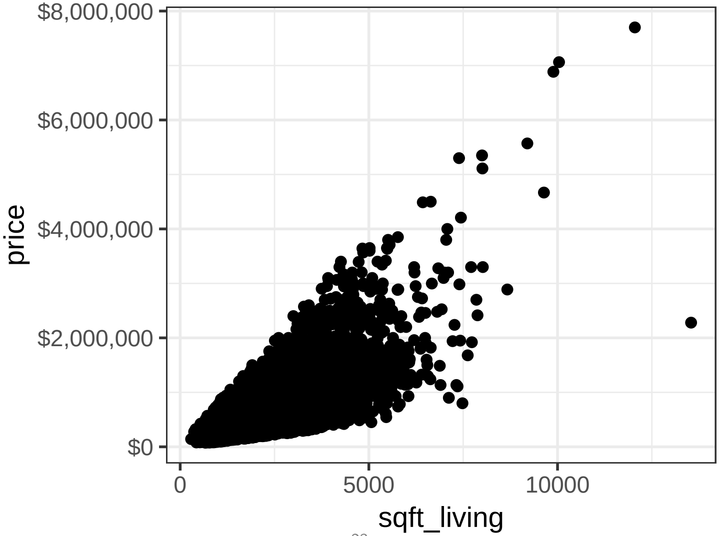
- Home area
- Land area
- # of bedrooms
- # of bathrooms
- Neighbourhood
- Year built
- ...

House Sales in King County, USA

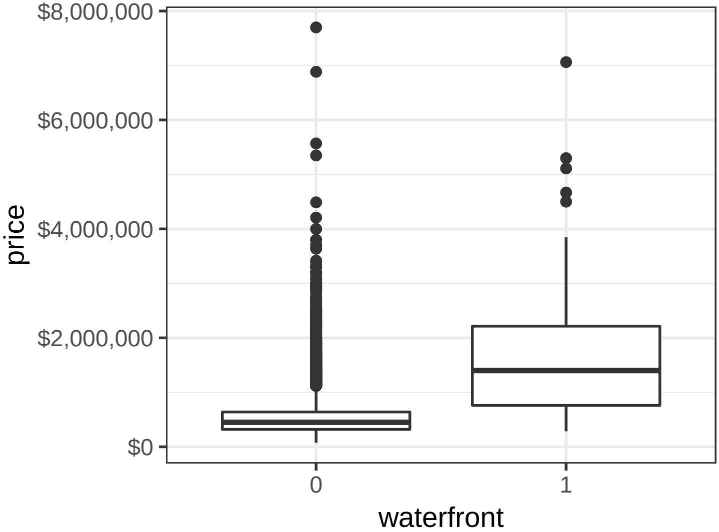
Dataset from Kaggle of 21613 homes sold between May 2014 and May 2015. (https://www.kaggle.com/harlfoxem/housesalesprediction/home)

- `price`: Price is prediction target
- `bedrooms`: Number of Bedrooms
- `bathrooms`: Number of bathrooms/bedrooms
- `sqft_living`: square footage of the home
- `sqft_lot`: square footage of the lot
- `floors`: Total floors (levels) in house
- `yr_built`: Built Year
- `yr_renovated`: Year when house was renovated
- `waterfront`: House which has a view to a waterfront
- `sqft_above`: square footage of house apart from basement

House Sales in King County, USA



House Sales in King County, USA



Simple linear regression

- Approximately a linear relationship between X and Y

$$Y = \beta_0 + \beta_1 X + \epsilon$$

- Use (training) data to produce estimates $\hat{\beta}_0$ and $\hat{\beta}_1$
  - Make predictions given $X = x$

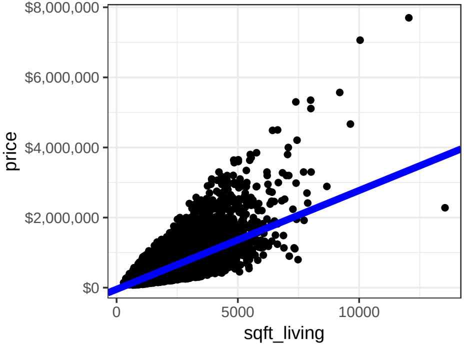
$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

- Minimise the residual sum of squares (RSS)

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x)^2$$

Simple linear regression: House prices

$$\texttt{price} = \beta_0 + \beta_1 \times \texttt{sqft\_living}$$



	Variable	estimate	std.error	p.value
1	(Intercept)	-47116.08	4923.34	0.00
2	sqft_living	281.96	2.16	0.00

Multiple linear regression

- Extend the simple linear regression model to accommodate multiple predictors

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

- β_j: the average effect on Y of a one unit increase in X_j, holding all other predictors fixed

Multiple linear regression: House prices

	Variable	estimate	std.error	p.value
1	(Intercept)	6289259.59	156282.14	0.00
2	bedrooms	-67820.03	2534.30	0.00
3	bathrooms	67280.69	4247.65	0.00
4	sqft_living	281.71	5.22	0.00
5	sqft_lot	-0.29	0.04	0.00
6	floors	43248.82	4526.50	0.00
7	yr_built	-3221.70	80.97	0.00
8	yr_renovated	6.69	4.74	0.16
9	waterfront	740322.15	20947.07	0.00
10	sqft_above	19.19	5.30	0.00

Shortcomings of linear regression

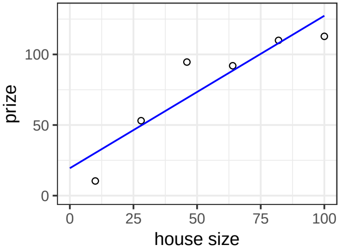
1. **Prediction accuracy**: the linear regression fit often does not predict well, especially when p (the number of predictors) is large
2. **Model Interpretability**: linear regression freely assigns a coefficient to each predictor variable. When p is large, we may sometimes seek, for the sake of interpretation, a smaller set of **important variables**
3. **Non-linearities**: linear assumption is almost **always an approximation** – sometimes bad.

Generalisations of the Linear Model

We discuss methods that expand the scope of linear models and how they are fit:

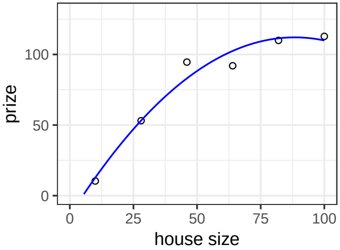
- *Regularised fitting:* Ridge regression and lasso
- *Classification problems:* logistic regression
- *Interactions:* Tree-based methods, bagging, random forests and boosting (these also capture non-linearities)

Motivation: Linear Regression House prices



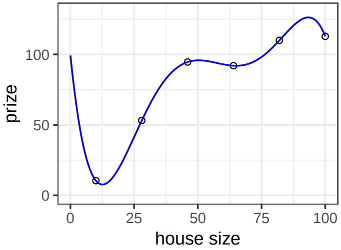
$$y = \beta_0 + \beta_1 x$$

- Underfit
- High bias



$$y = \beta_0 + \beta_1 x + \beta_2 x^2$$

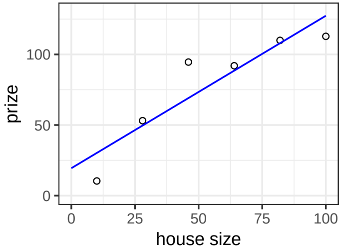
- Just right



$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \beta_5 x^5$$

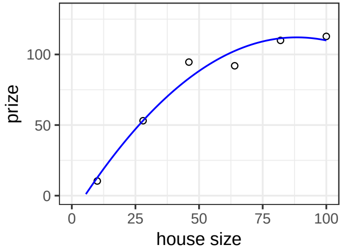
- Overfit
- High variance

Motivation: Linear Regression House prices



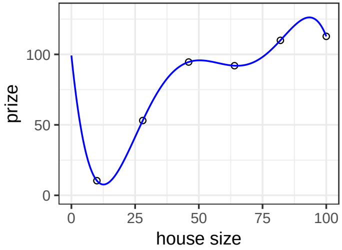
$$y = \beta_0 + \beta_1 x$$

- Underfit
- High bias



$$y = \beta_0 + \beta_1 x + \beta_2 x^2$$

- Just right



$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \beta_5 x^5$$

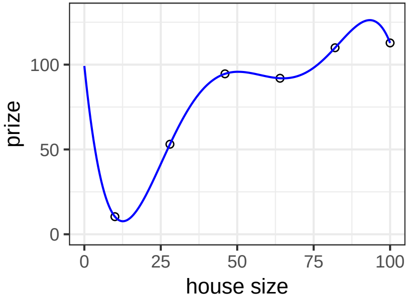
- Overfit
- High variance

Overfitting: We have too many features, the model may fit the training set well ($RSS \approx 0$), but fail to generalise to new cases (predict prices of new example)

Overfitting with many features

Not unique to polynomial regression but also if lots of inputs (p large)

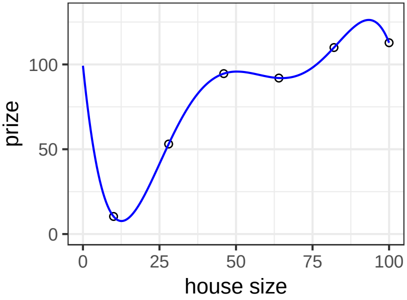
- x_1 = Home area
- x_2 = Land area
- x_3 = # of bedrooms
- x_4 = # of bathrooms
- x_5 = Neighbourhood
- x_6 = Year built
- x_7 = Average income in the neighbouhood
- x_8 = Kitchen size
- $\vdots$
- x_{100}



Addressing Overfitting

There are several several options

1. Reduce number of features/variable
   - Manually
   - Subset selection algorithm
2. Regularisation
   - Keep all the features, but reduce magnitude of parameters β_i
     - Works well when we have a lot of features, each of which contributes a bit to predicting y



Addressing overfitting via regularisation

$$\text{Total cost} \;=\; \begin{array}{c}\text{Measure}\\ \text{of Fit}\end{array} \;+\; \begin{array}{c}\text{Measure of Magnitude}\\ \text{of Coefficient}\end{array}$$

Addressing overfitting via regularisation

$$\text{Total cost} = \underbrace{\begin{array}{c}\text{Measure}\\ \text{of Fit}\end{array}}_{\mathrm{RSS}} + \begin{array}{c}\text{Measure of Magnitude}\\ \text{of Coefficient}\end{array}$$

Addressing overfitting via regularisation

$$\text{Total cost} = \underbrace{\begin{array}{c}\text{Measure}\\ \text{of Fit}\end{array}}_{\mathrm{RSS}} + \underbrace{\begin{array}{c}\text{Measure of Magnitude}\\ \text{of Coefficient}\end{array}}_{\beta_1^2+\beta_2^2+\cdots+\beta_p^2}$$

Ridge Regression

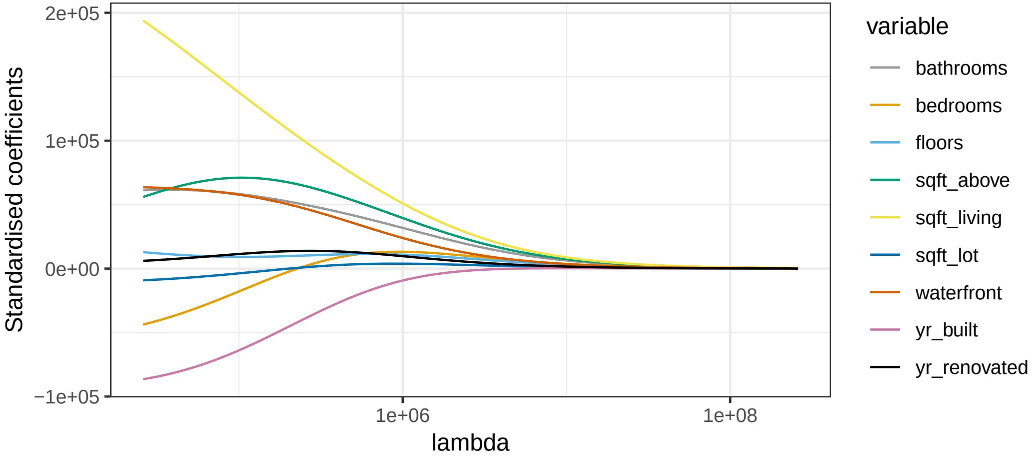
Minimise on β:

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

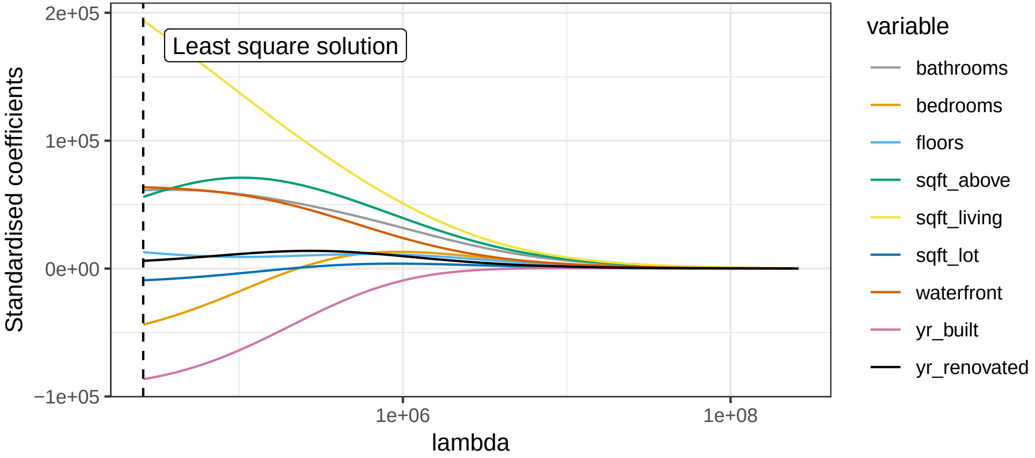
λ: Tuning parameter = balance of fit and magnitude

- $\lambda \to \infty$: 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.

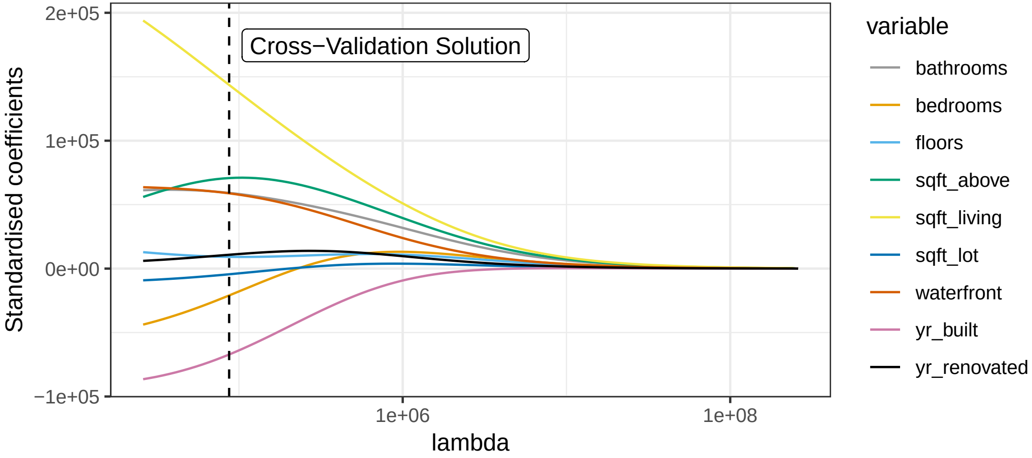
Ridge Solutions paths: The house data



Ridge Solutions paths: The house data



Ridge Solutions paths: The house data

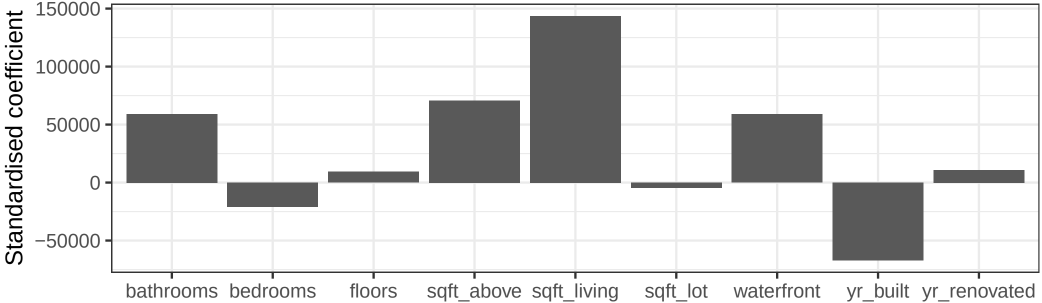


Ridge Cross-Validation Solution: The house data

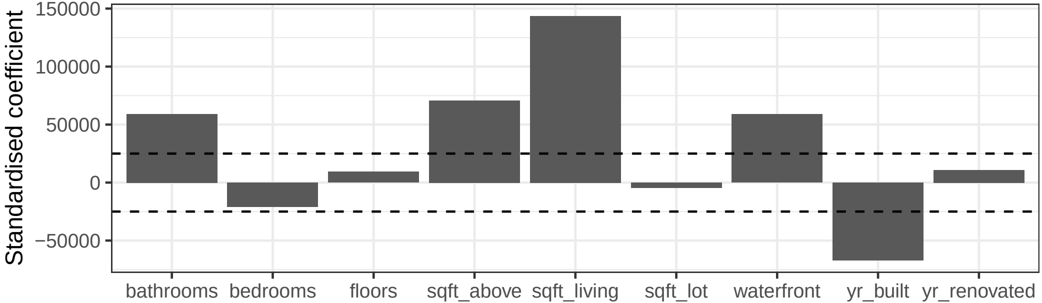
Variable	Estimate
(Intercept)	4461371.36
bedrooms	-23152.81
bathrooms	76655.13
sqft_living	155.79
sqft_lot	-0.11
floors	17043.10
yr_built	-2290.26
yr_renovated	27.11
waterfront	675263.65
sqft_above	85.58

Contains all variables so **still harder to interpret!**

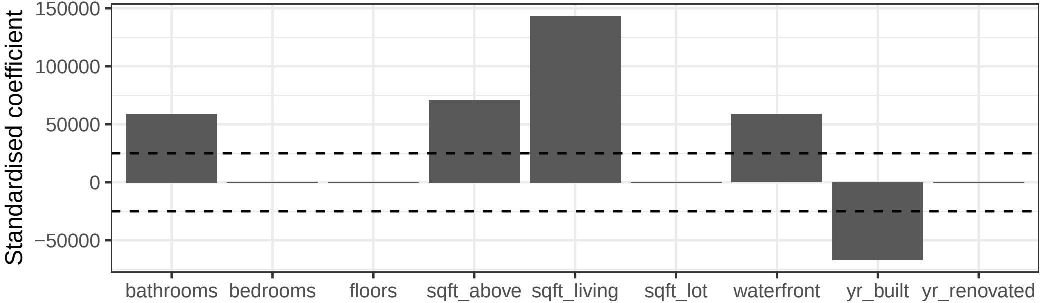
Thresholding ridge coefficients?



Thresholding ridge coefficients?



Thresholding ridge coefficients?



Feature selection via regularisation

$$\text{Total cost} = \underbrace{\begin{array}{c}\text{Measure}\\ \text{of Fit}\end{array}}_{\mathrm{RSS}} + \underbrace{\begin{array}{c}\text{Measure of Magnitude}\\ \text{of Coefficient}\end{array}}_{|\beta_1|+|\beta_2|+\cdots+|\beta_p|}$$

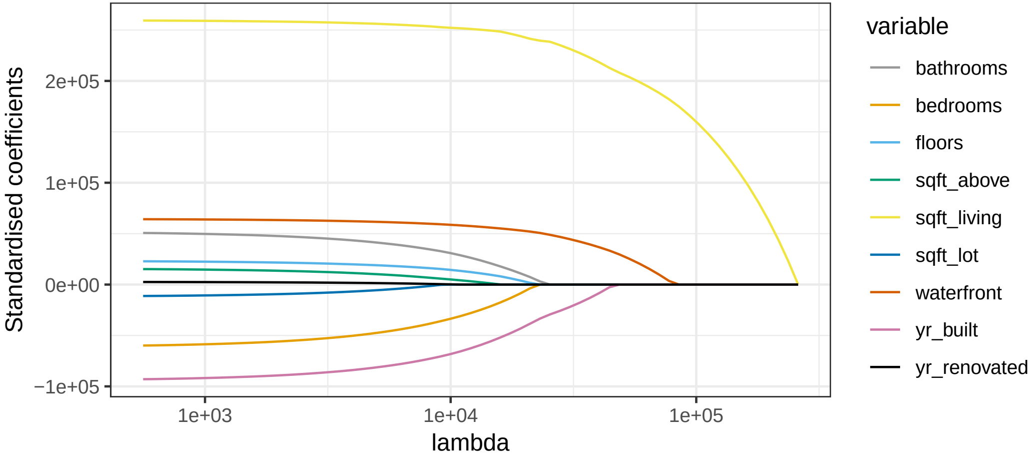
Lasso regression

Minimise on β:

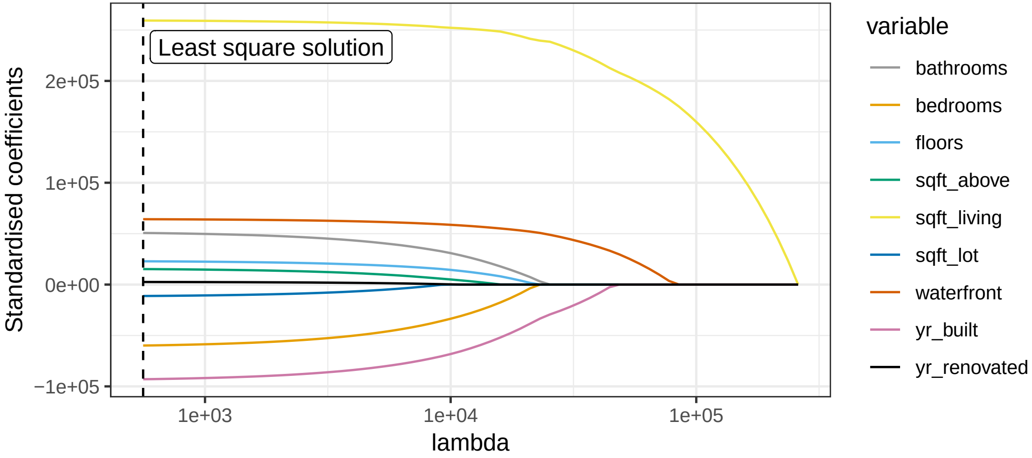
$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right) + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

- 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

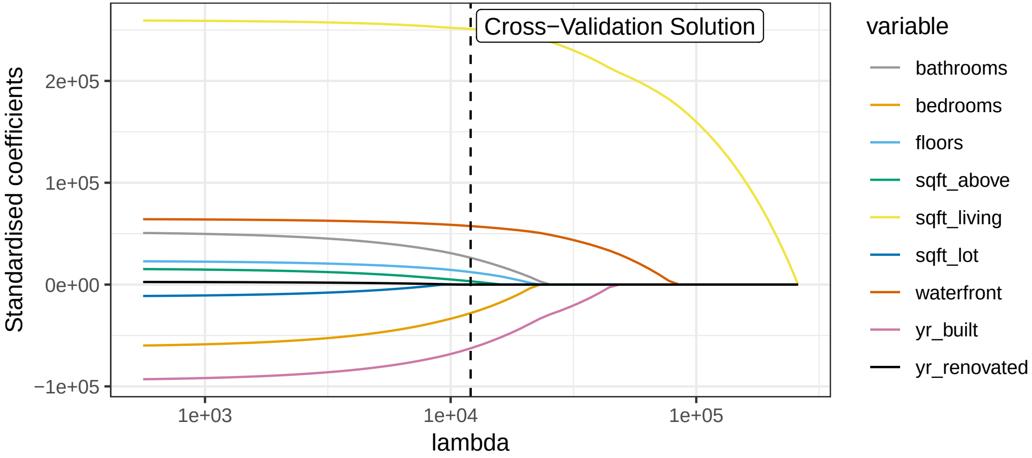
Lasso Solutions paths: The house data



Lasso Solutions paths: The house data



Lasso Solutions paths: The house data



Lasso Cross-Validation Solution: The house data

Variable	estimate
(Intercept)	4166939.79
bedrooms	-30936.45
bathrooms	34095.80
sqft_living	272.38
sqft_lot	–
floors	22706.47
yr_built	-2134.77
yr_renovated	–
waterfront	659380.22
sqft_above	3.90

Lasso vs. Ridge