STA4026S – Honours Analytics
Section A – Theory and Application of Supervised Learning
Lecture 1 – Introduction to Supervised Learning

Stefan S. Britz
stefan.britz@uct.ac.za

Department of Statistical Sciences
University of Cape Town

Overview

Overview

In this section we will:

1. cover some of the fundamental theoretical principles underpinning **supervised statistical and machine learning**
2. explore various models, algorithms, and heuristics to analyse different types of data, both for **regression** (continuous target variable) and **classification** (categorical target variable) problems
3. **apply** these methods in **R**.

The aim is to find a balance between breadth of topics, depth of theory, and practical application.

Outline

- Lecture 1: Introduction to supervised learning
  - Introduction
  - Bias-Variance trade-off
  - Model validation
- Lecture 2: Model Selection & Regularisation
  - L_1 & L_2 regularisation
  - ElasticNet
- Lecture 3: Classification Models
  - Logistic regression
  - Model evaluation
  - ROC curves

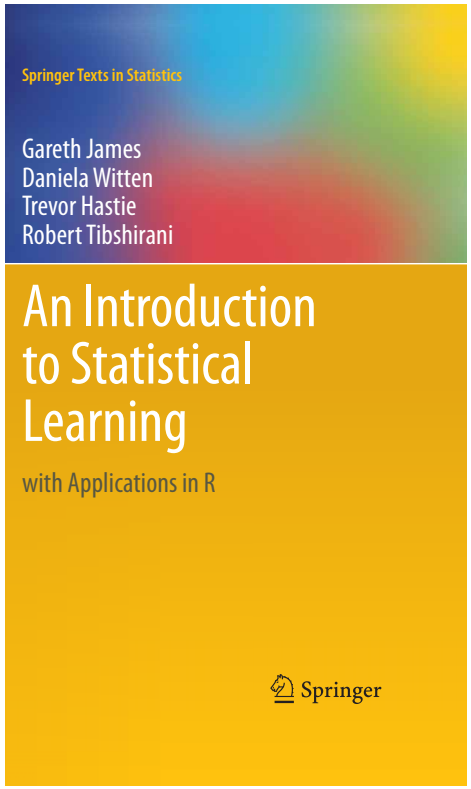
Outline

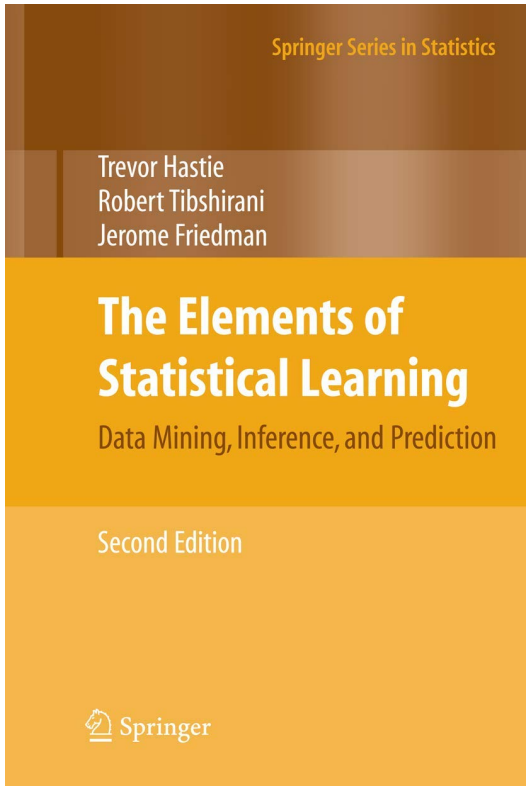
- Lecture 4: Beyond Linearity
  - Polynomial regression
  - KNN
- Lecture 5 & 6: Tree-based Methods
  - CART
  - Random forests
  - Boosting

Resources

The course notes are available here and on Vula

The following texts are the main sources for the theoretical components:

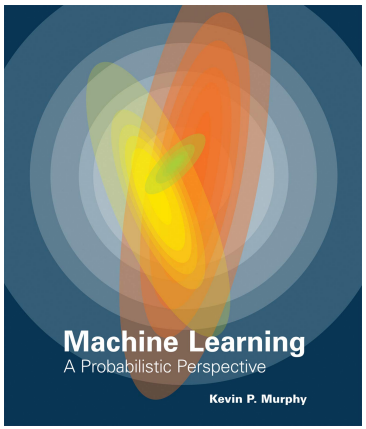


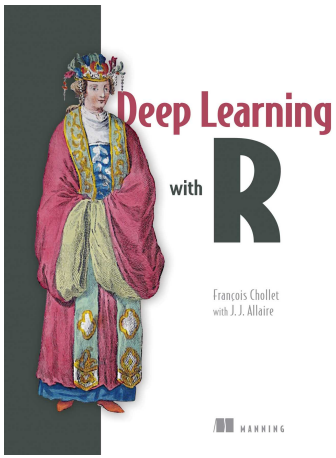


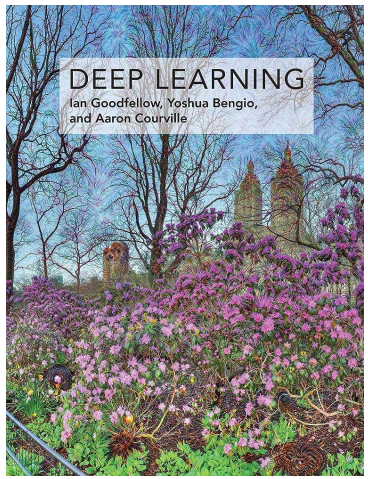
Downloadable here and here

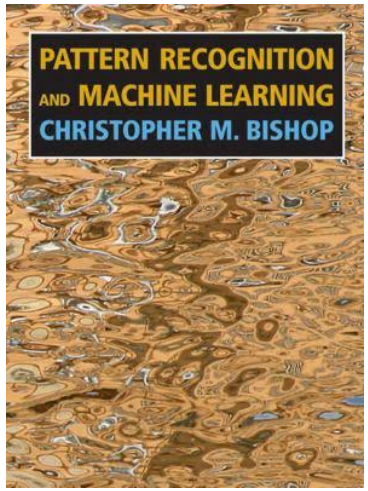
Beyond this course

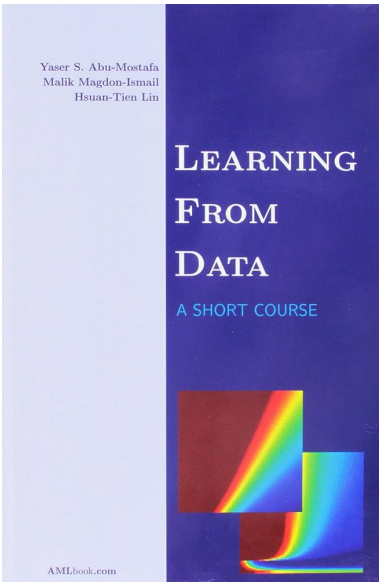
There is a lot of excellent content out there to continue with. Recommendations will depend on your goals and focus. Here are just a few suggestions:











Introduction

What is Analytics?

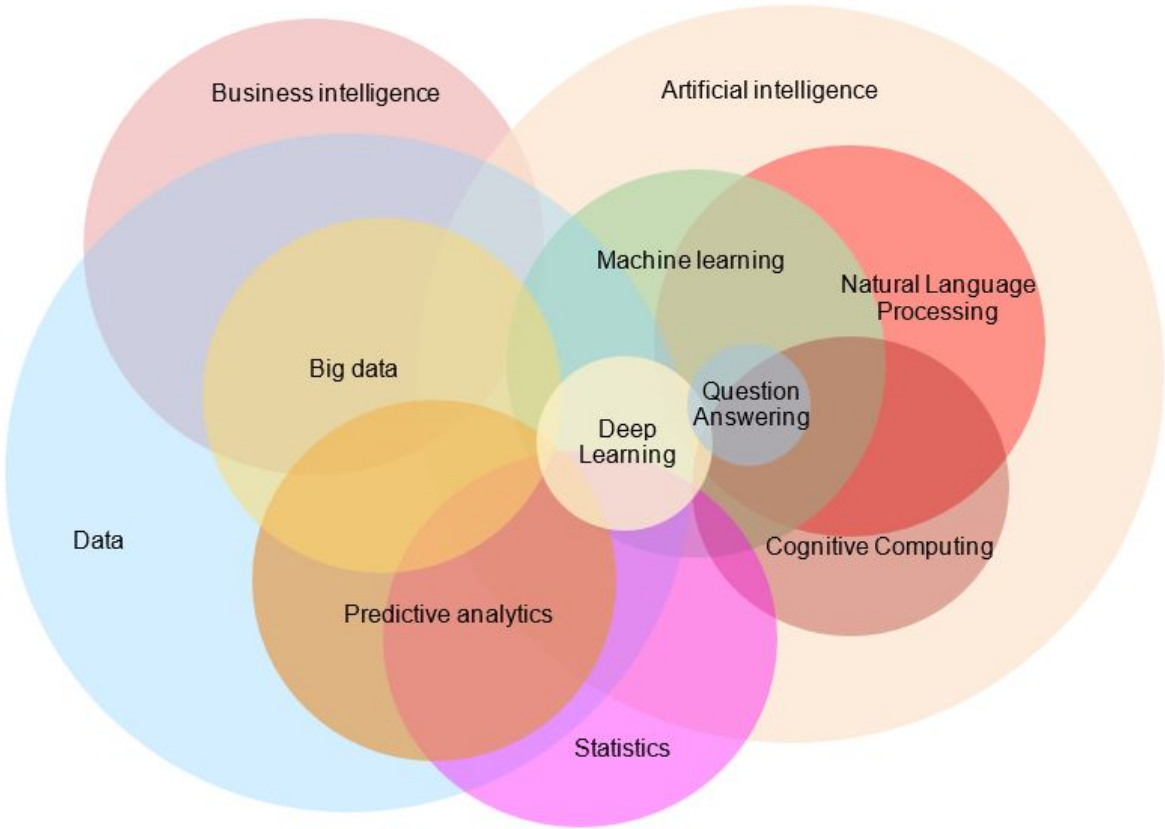
Analytics is the scientific process of transforming typically large amounts of data into information to guide decision making

Analytics techniques can be grouped into three categories:

1. Descriptive Analytics
   Data summary and visualisation
2. Predictive Analytics
   Statistical learning techniques to identify relationships in the data that are important for prediction and classification
3. Prescriptive Analytics
   Using data patterns and predictions to prescribe optimal decisions

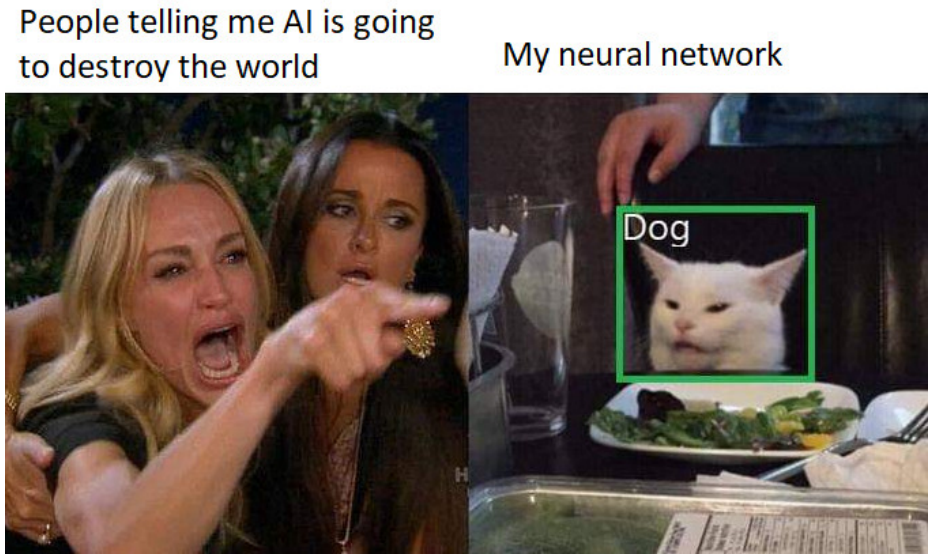
Sounds simple enough...

...wait, what now?!



Buzzwords come and go, can sometimes blur the boundaries, and can be hard to distinguish from the established terms... Until they are established.

Some tongue-in-cheek LOLs



Statistical Learning or Machine Learning?

Statistical Learning and **Machine Learning** are both concerned with extracting insights from data and making predictions, albeit with slightly different philosophical and methodological perspectives.

Statistical Learning

- Model relationships between variables
- Make inferences about population and underlying data-generating process
- Often makes explicit assumptions about distributions and relationships
- More interpretable, focus on the impact of individual variables on the outcome

Machine Learning

- Develop algorithms to automatically learn relationships
- D-G process is important, but more focused on achieving optimal predictive performance
- More agnostic, not relying heavily on explicit statistical assumptions
- Can be highly complex, limiting the interpretability

Statistical Learning or Machine Learning?

Statistical Learning and **Machine Learning** are both concerned with extracting insights from data and making predictions, albeit with slightly different philosophical and methodological perspectives.

Statistical Learning

- Model relationships between variables
- Make inferences about population and underlying data-generating process
- Often makes explicit assumptions about distributions and relationships
- More interpretable, focus on the impact of individual variables on the outcome

Machine Learning

- Develop algorithms to automatically learn relationships
- D-G process is important, but more focused on achieving optimal predictive performance
- More agnostic, not relying heavily on explicit statistical assumptions
- Can be highly complex, limiting the interpretability

Statistical Learning or Machine Learning?

Statistical Learning and **Machine Learning** are both concerned with extracting insights from data and making predictions, albeit with slightly different philosophical and methodological perspectives.

Statistical Learning

- Model relationships between variables
- Make inferences about population and underlying data-generating process
- Often makes explicit assumptions about distributions and relationships
- More interpretable, focus on the impact of individual variables on the outcome

Machine Learning

- Develop algorithms to automatically learn relationships
- D-G process is important, but more focused on achieving optimal predictive performance
- More agnostic, not relying heavily on explicit statistical assumptions
- Can be highly complex, limiting the interpretability

Statistical Learning

There are three broad types of statistical learning methods:

1. Supervised learning
   - Methods that relate a **response variable** y to a set of predictors X, with the aim of predicting the response for future observations
   - Includes linear, non-linear and logistic regression; tree-based methods; KNN; artificial neural networks; and SVMs (amongst many others)
2. Unsupervised learning
   - Methods of identifying patterns and structure in the data **without a response variable**. Often used to identify latent groupings of the observations
   - Includes principal component analysis, all clustering algorithms and self-organising maps
3. Reinforcement learning
   - Agents learn to make sequential decisions through interactions with an environment in order to maximize cumulative rewards

Statistical Learning

There are three broad types of statistical learning methods:

1. Supervised learning
   - Methods that relate a **response variable** y to a set of predictors X, with the aim of predicting the response for future observations
   - Includes linear, non-linear and logistic regression; tree-based methods; KNN; artificial neural networks; and SVMs (amongst many others)
2. Unsupervised learning
   - Methods of identifying patterns and structure in the data **without a response variable**. Often used to identify latent groupings of the observations
   - Includes principal component analysis, all clustering algorithms and self-organising maps
3. Reinforcement learning
   - Agents learn to make sequential decisions through interactions with an environment in order to maximize cumulative rewards

Statistical Learning

There are three broad types of statistical learning methods:

1. Supervised learning
   - Methods that relate a **response variable** y to a set of predictors X, with the aim of predicting the response for future observations
   - Includes linear, non-linear and logistic regression; tree-based methods; KNN; artificial neural networks; and SVMs (amongst many others)
2. Unsupervised learning
   - Methods of identifying patterns and structure in the data **without a response variable**. Often used to identify latent groupings of the observations
   - Includes principal component analysis, all clustering algorithms and self-organising maps
3. Reinforcement learning
   - Agents learn to make sequential decisions through interactions with an environment in order to maximize cumulative rewards

Statistical Learning

There are three broad types of statistical learning methods:

1. Supervised learning
   - Methods that relate a **response variable** y to a set of predictors X, with the aim of predicting the response for future observations
   - Includes linear, non-linear and logistic regression; tree-based methods; KNN; artificial neural networks; and SVMs (amongst many others)
2. Unsupervised learning
   - Methods of identifying patterns and structure in the data **without a response variable**. Often used to identify latent groupings of the observations
   - Includes principal component analysis, all clustering algorithms and self-organising maps
3. Reinforcement learning
   - Agents learn to make sequential decisions through interactions with an environment in order to maximize cumulative rewards

Supervised Learning Example

- Dataset containing percentages of words and punctuation marks appearing in 4,601 emails marked as spam or not spam
- Goal: predict whether an incoming email is spam

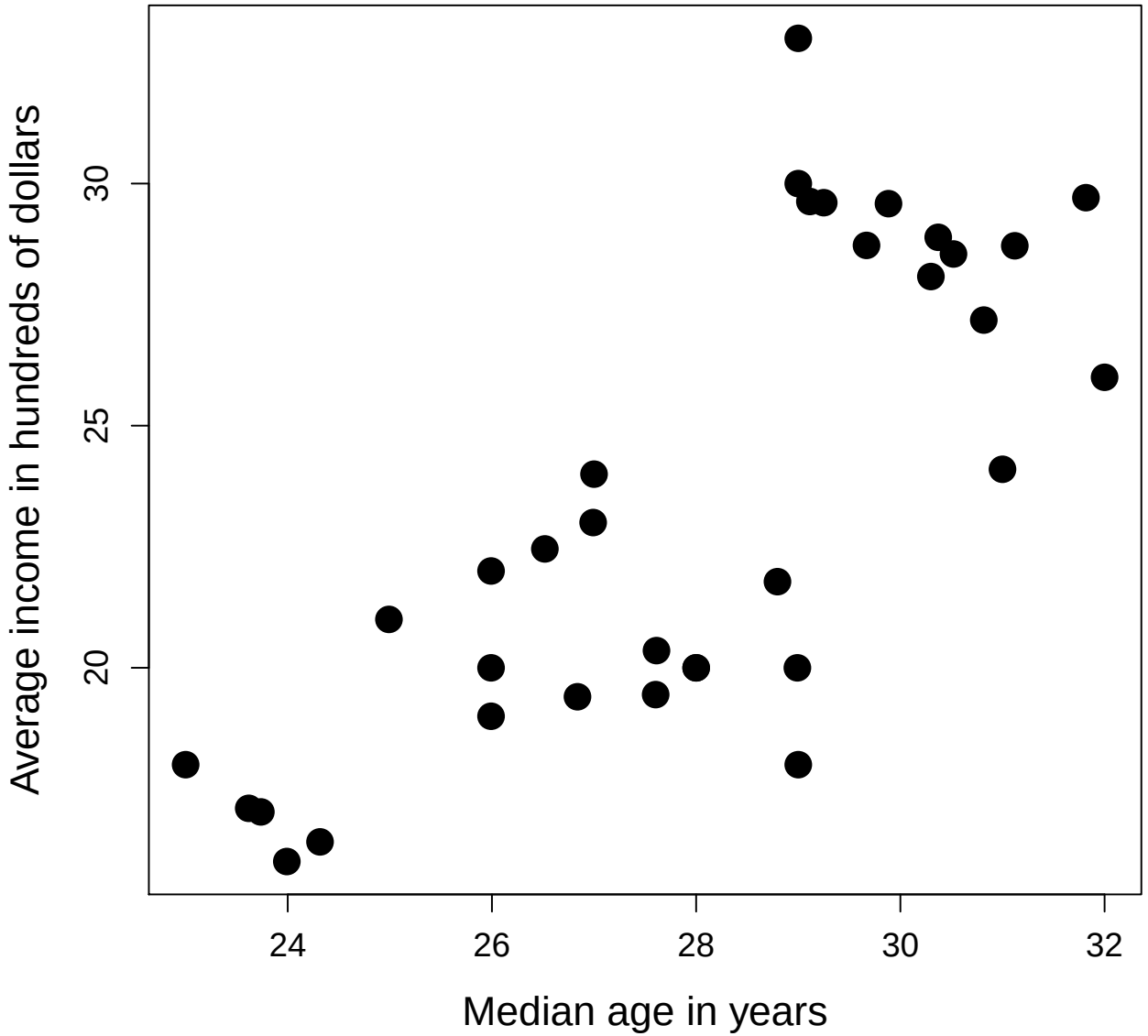
type	free	your	our	mail	order	$	!	(
email	0.000	0.000	0.270	0.550	0.000	0.000	0.000	0.549
email	0.000	1.780	0.890	0.000	0.000	0.000	0.000	0.298
email	0.000	0.000	0.000	1.960	0.000	0.000	0.000	0.373
email	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
email	0.000	0.000	0.600	0.000	0.100	0.000	0.000	0.049
spam	0.000	2.830	0.940	0.940	0.000	0.000	0.000	0.000
spam	1.050	2.100	0.000	0.000	0.000	0.182	0.365	0.365
email	1.380	1.380	0.000	0.690	0.000	0.000	2.378	0.000
email	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.098
email	0.930	0.000	0.000	0.000	0.000	0.000	0.000	0.163

Unsupervised Learning Example

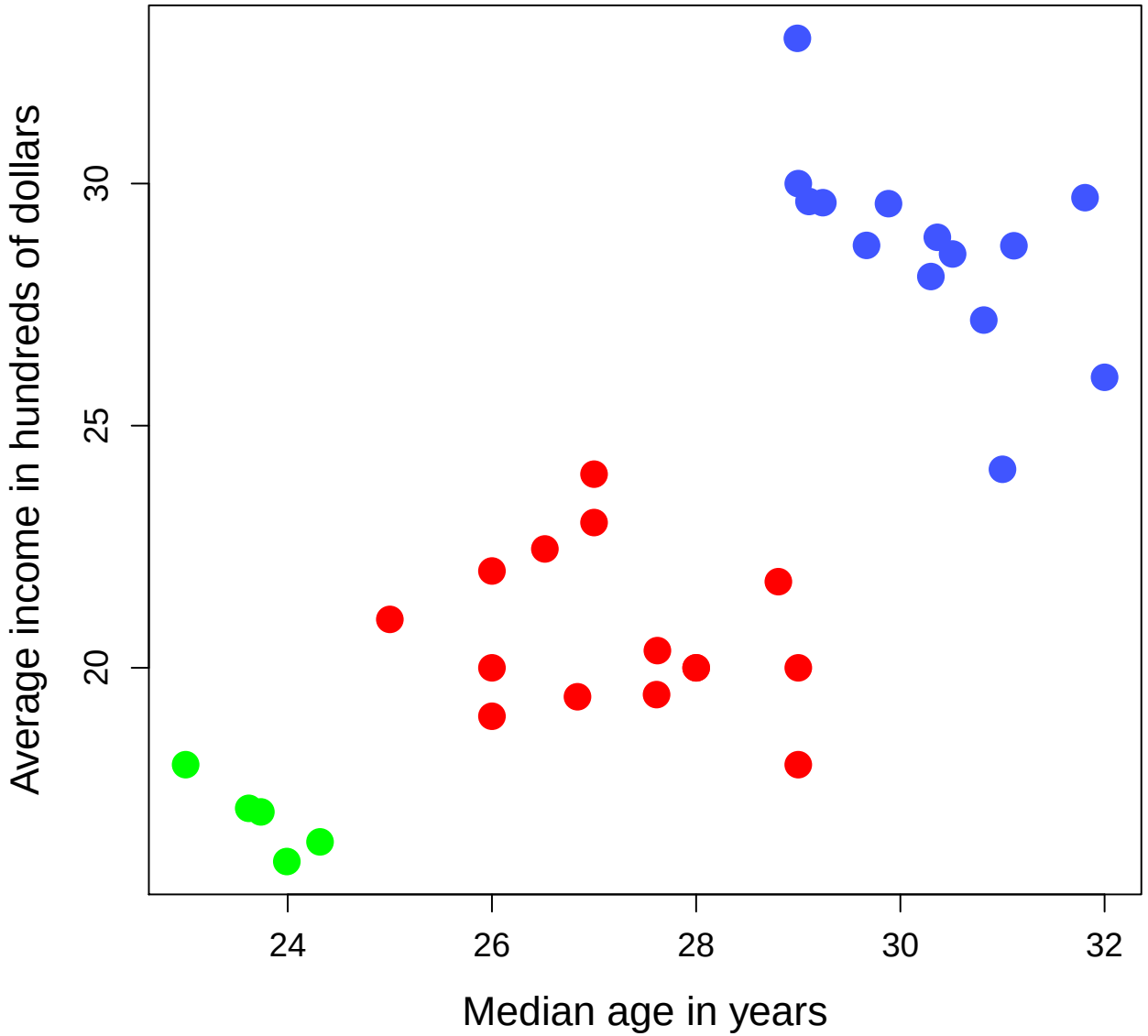
- Dataset of demographic variables for US states
- Goal: identify distinct groups of states for targeted marketing

State	Income	Age	Persons	Births	Employ	Urban	Educ
AL	19.0	26	64	55	31	22	89
AS	18.0	29	34	43	29	22	87
DE	33.0	29	226	66	37	23	108
FL	24.1	31	91	74	15	20	106
GA	22.0	26	68	55	32	24	88
KY	20.0	28	76	45	27	22	85
LA	21.0	25	72	63	17	25	86
MD	30.0	29	314	73	25	20	10
MI	16.0	24	46	38	31	24	86
MO	26.0	32	63	67	28	21	93

Unsupervised Learning Problem



Unsupervised Learning Solution/SL Problem



Supervised Learning

Intro

Given a quantitative response Y and a set of p predictors $X_1, X_2, \ldots, X_p$, we are interested in the assumed, unobserved function that maps the inputs to the outputs:

$$Y = \underbrace{f(X)}_{systematic} + \underbrace{\epsilon}_{random},$$

where $f(.)$ represents the fixed, but unknown function and ϵ is a random error term, independent of X, with $E(\epsilon) = 0$

Intro

By estimating f such that

$$\hat{Y} = \hat{f}(X),$$

we allow for:

- Prediction of Y – the primary goal in forecasting
- and inference – describing how Y is affected by changes in X.

Hypothesising $\hat{f}$ can be done in two ways:

- Parametric approach
- Non-parametric approach

Parametric Approach

An **assumption** is made about the the functional form of f.

For example, the linear model

$$f(\boldsymbol{X}) = \beta_0 + \sum_{j=1}^{p} \beta_j X_j$$

The best estimate of f is defined as the set of parameters $\hat{\boldsymbol{\beta}}$ that minimise some specified **loss function**.

Given a **training set** $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{n}$ use, say, OLS to minimise the MSE:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \Big[y_i - \hat{f}(x_i) \Big]^2$$

The problem of estimating an arbitrary p-dimensional function is now simplified to fitting a set of parameters.

Non-Parametric Approach

- Makes no explicit assumptions regarding the functional form of f
- Allows one to fit a wide range of possible forms for f
- Estimation is not reduced to estimating a set of parameters, hence this approach generally requires more data than parametric estimation
- The objective remains to find f that best fits the training data, whilst avoiding **overfitting** to ensure that the model **generalises** well to unseen data

Generalisation

The primary goal of prediction is to accurately predict the outcomes of data not yet observed by the model, i.e. **out-of-sample** observations

Consider the following case:

- The estimated function $\hat{f}$ is fixed
- Out-of-sample observations – $\{\boldsymbol{x}_0, y_0\}$ – are introduced, referred to as the **test set**

We are interested in the expected **test MSE**:

$$E\left[y_0 - \hat{f}(\boldsymbol{x}_0)\right]^2$$

which can be deconstructed as follows.

Test MSE Decomposition

$$
\begin{aligned}
E\left[y_0 - \hat{f}(\boldsymbol{x}_0)\right]^2 &= E\left[f(\boldsymbol{x}_0) + \epsilon - \hat{f}(\boldsymbol{x}_0)\right]^2 \\
&= E\left[\left(f(\boldsymbol{x}_0) - \hat{f}(\boldsymbol{x}_0)\right)^2 + 2\epsilon\left(f(\boldsymbol{x}_0) - \hat{f}(\boldsymbol{x}_0)\right) + \epsilon^2\right] \\
&= E\left[f(\boldsymbol{x}_0) - \hat{f}(\boldsymbol{x}_0)\right]^2 + 2E[\epsilon]E\left[f(\boldsymbol{x}_0) - \hat{f}(\boldsymbol{x}_0)\right] + E\left[\epsilon^2\right] \\
&= \underbrace{E\left[f(\boldsymbol{x}_0) - \hat{f}(\boldsymbol{x}_0)\right]^2}_{reducible} + \underbrace{Var(\epsilon)}_{irreducible} \qquad (1)
\end{aligned}
$$

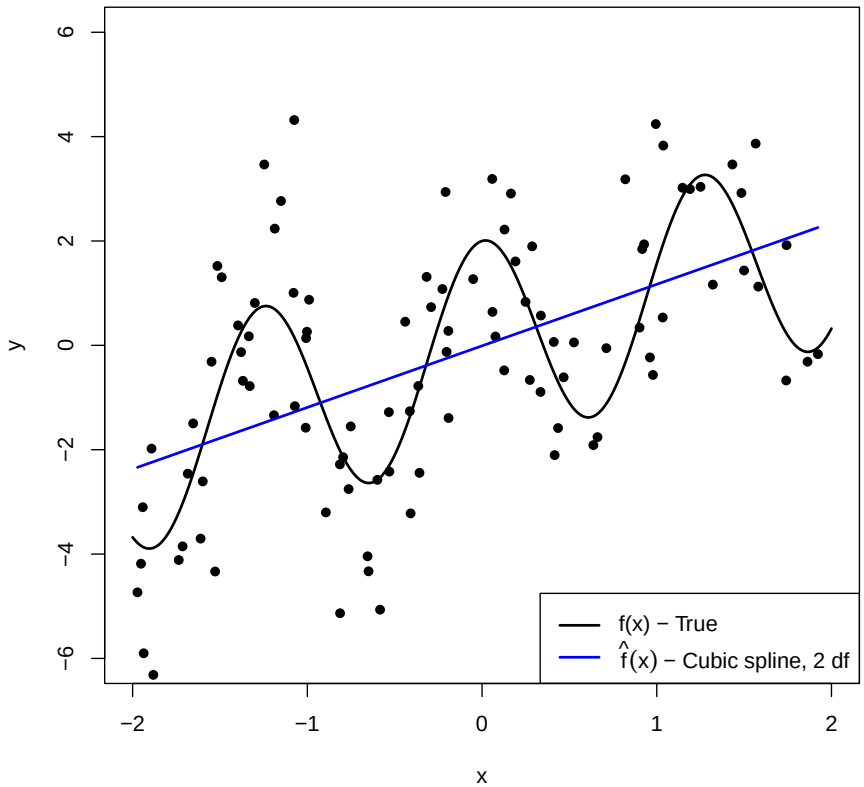
The primary goal of machine learning is to find an $\hat{f}$ that minimises the **reducible error**.

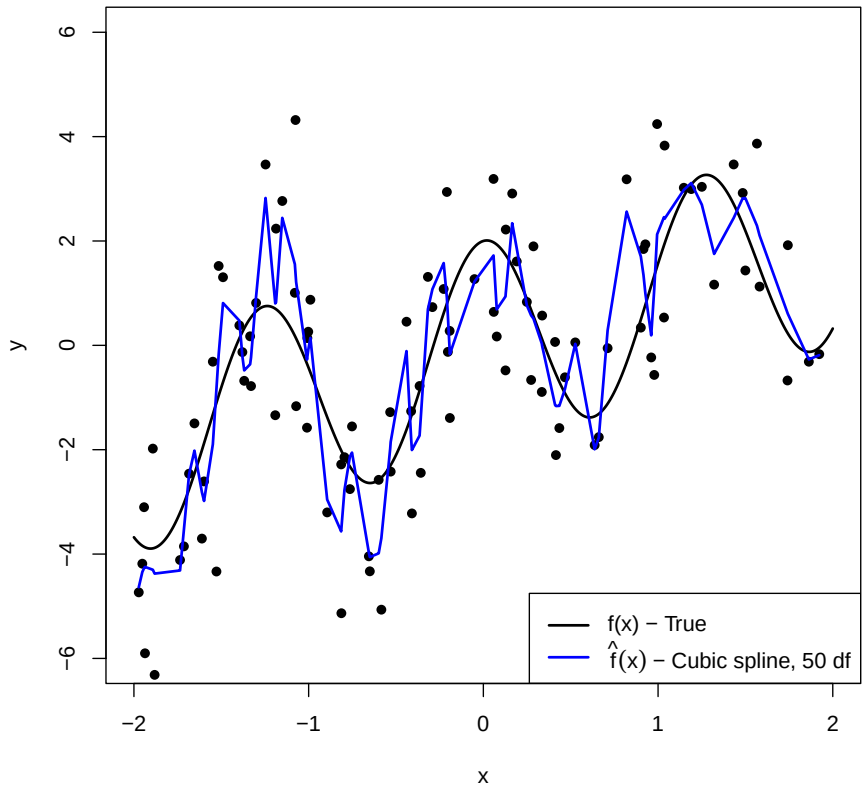
Because of the **irreducible error** (noise), the theoretical MSE will have some lower bound, which is (almost) **always unknown in practice**.

Bias-Variance Trade-Off

Bias-Variance Trade-Off

To find an $\hat{f}$ that minimises the reducible error, we need to identify a **suitably complex** model.

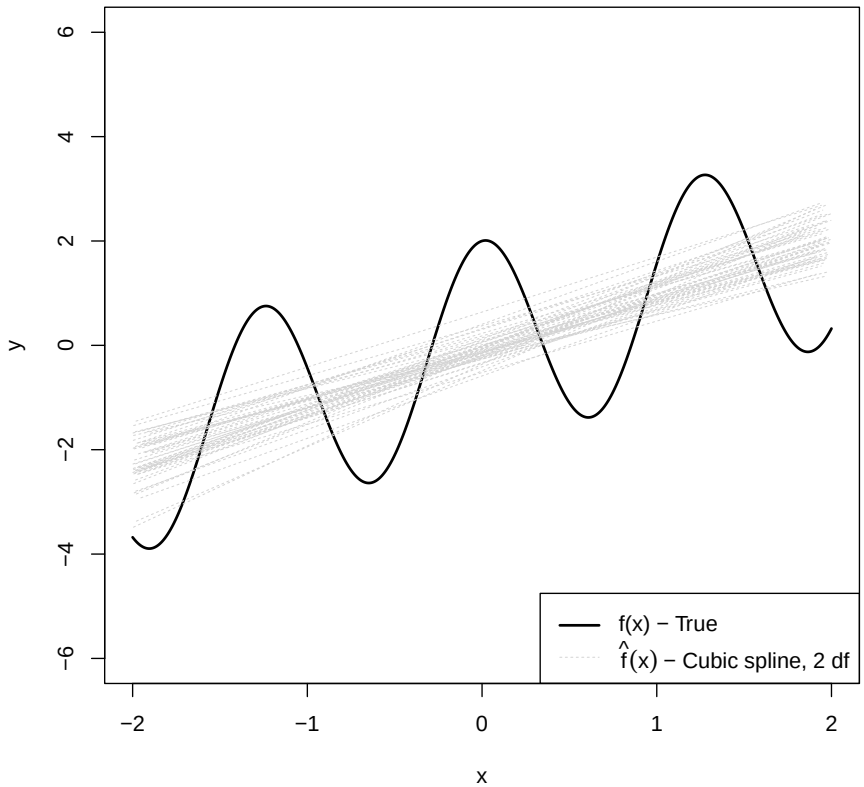


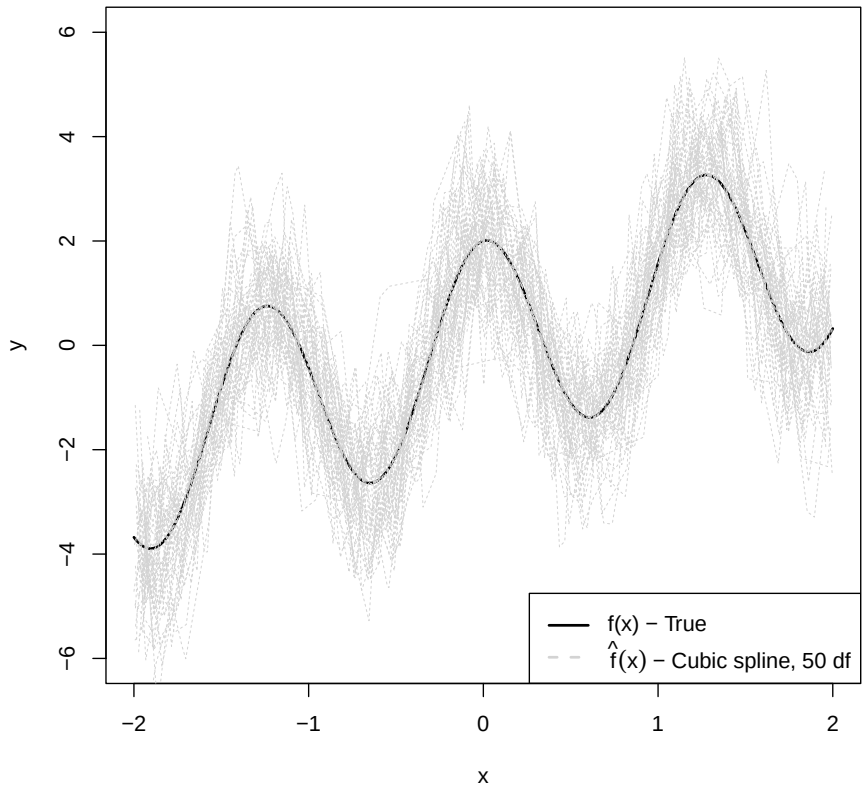


If the model is too simple [left], it won't capture the pattern closely enough. If it's too complex [right], it fits the noise too much and does not generalise well.

Bias-Variance Trade-Off

To find an $\hat{f}$ that minimises the reducible error, we need to identify a **suitably complex** model.





If the model is too simple [left], it won't capture the pattern closely enough. If it's too complex [right], it fits the noise too much and does not generalise well.

Bias-Variance Trade-Off

- Consider again a fixed $\hat{f}$ and out-of-sample observations $\{\boldsymbol{x}_0, y_0\}$
- Let $f = f(\boldsymbol{x}_0)$ and $\hat{f} = \hat{f}(\boldsymbol{x}_0)$
- Note that f is deterministic such that $E[f] = f$

Let us now further deconstruct the **reducible error** in Equation 1:

$$E\left[f - \hat{f}\right]^2$$

Bias-Variance Trade-Off

$$
\begin{aligned}
E\left[f-\hat{f}\right]^2 &= E\left[\hat{f}-f\right]^2 \\
&= E\left[\hat{f}-E(\hat{f})+E(\hat{f})-f\right]^2 \\
&= E\left\{\left[\hat{f}-E(\hat{f})\right]^2+2\left[\hat{f}-E(\hat{f})\right]\left[E(\hat{f})-f\right]+\left[E(\hat{f})-f\right]^2\right\} \\
&= E\left[\hat{f}-E(\hat{f})\right]^2+2E\left\{\left[\hat{f}-E(\hat{f})\right]\left[E(\hat{f})-f\right]\right\} \\
&\quad +E\left[E(\hat{f})-f\right]^2 \\
&= Var\left[\hat{f}\right]+0+\left[E(\hat{f})-f\right]^2 \\
&= Var\left[\hat{f}\right]+Bias^2\left[\hat{f}\right] \qquad (2)
\end{aligned}
$$

Bias-Variance Trade-Off

Showing that the crossproduct term equals zero:

$$
\begin{aligned}
E\left\{\left[\hat{f} - E(\hat{f})\right]\left[E(\hat{f}) - f\right]\right\} &= E\left[\hat{f}E(\hat{f}) - E(\hat{f})E(\hat{f}) - \hat{f}f + E(\hat{f})f\right] \\
&= E(\hat{f})E(\hat{f}) - E(\hat{f})E(\hat{f}) - E(\hat{f})f + E(\hat{f})f \\
&= 0
\end{aligned}
$$

Bias-Variance Trade-Off

- From Equation 2 we see that in order to minimise the expected test MSE, we need to find a model that has the lowest combined variance and (squared) bias
- The **variance** represents the extent to which $\hat{f}$ changes between different training samples taken from the same population
- The **bias** of $\hat{f}$ is simply the error that is introduced by approximating the real-world relationship with a simpler representation

NOTE!!

Since f is generally unknown, the bias component cannot be directly observed or measured outside of simulations.

Example 1 – Simulation

Although the concepts of model complexity and flexibility are not necessarily perfectly defined – depending on the class of model being hypothesised – this example should provide an intuitive understanding.

Consider a simple function with only one feature:

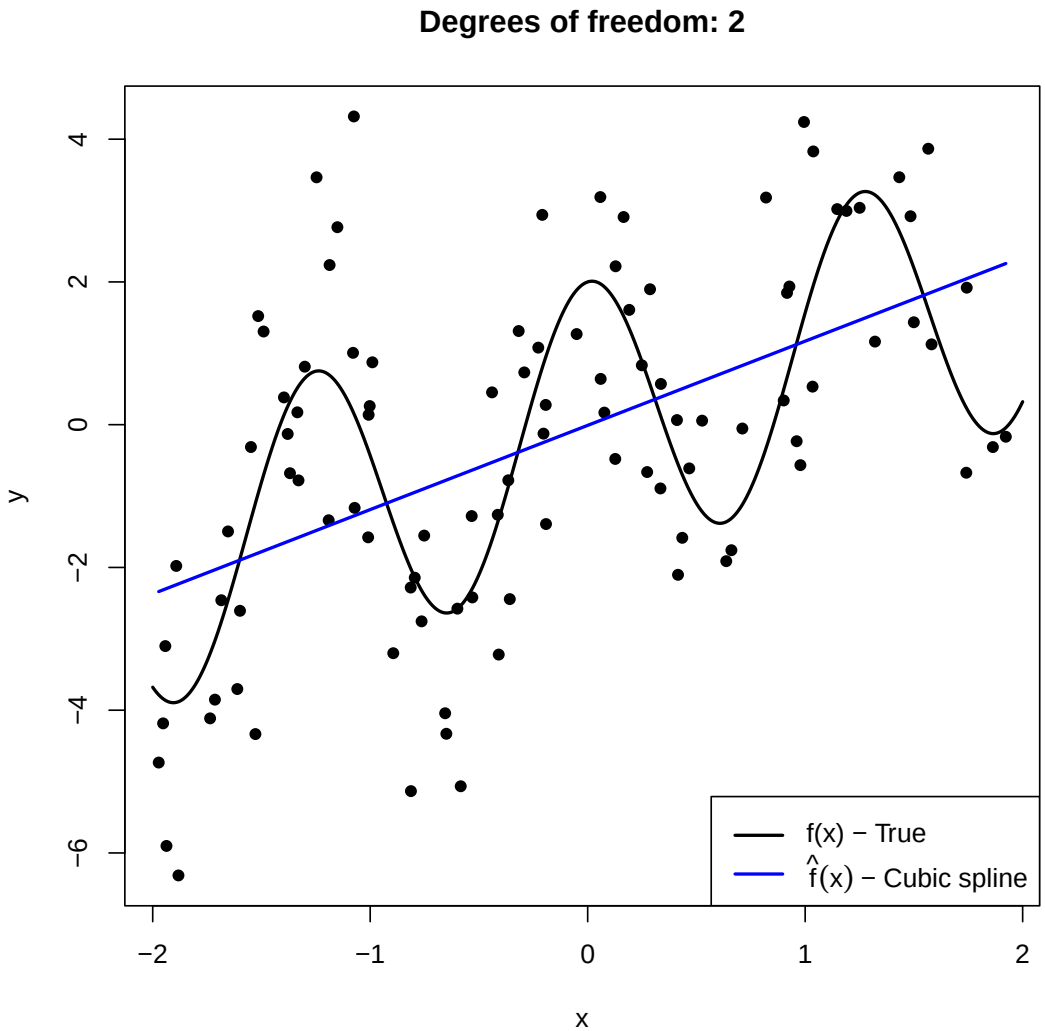
$$Y = X + 2\cos(5X) + \epsilon,$$

where $\epsilon \sim N(0, 2)$

Simulate $n = 100$ observations from $X \sim U(-2, 2)$, to which we fit cubic smoothing splines[1] of increasing complexity.

[1] Splines are beyond the scope of this course, but provide an easy-to-see illustration of "flexibility"

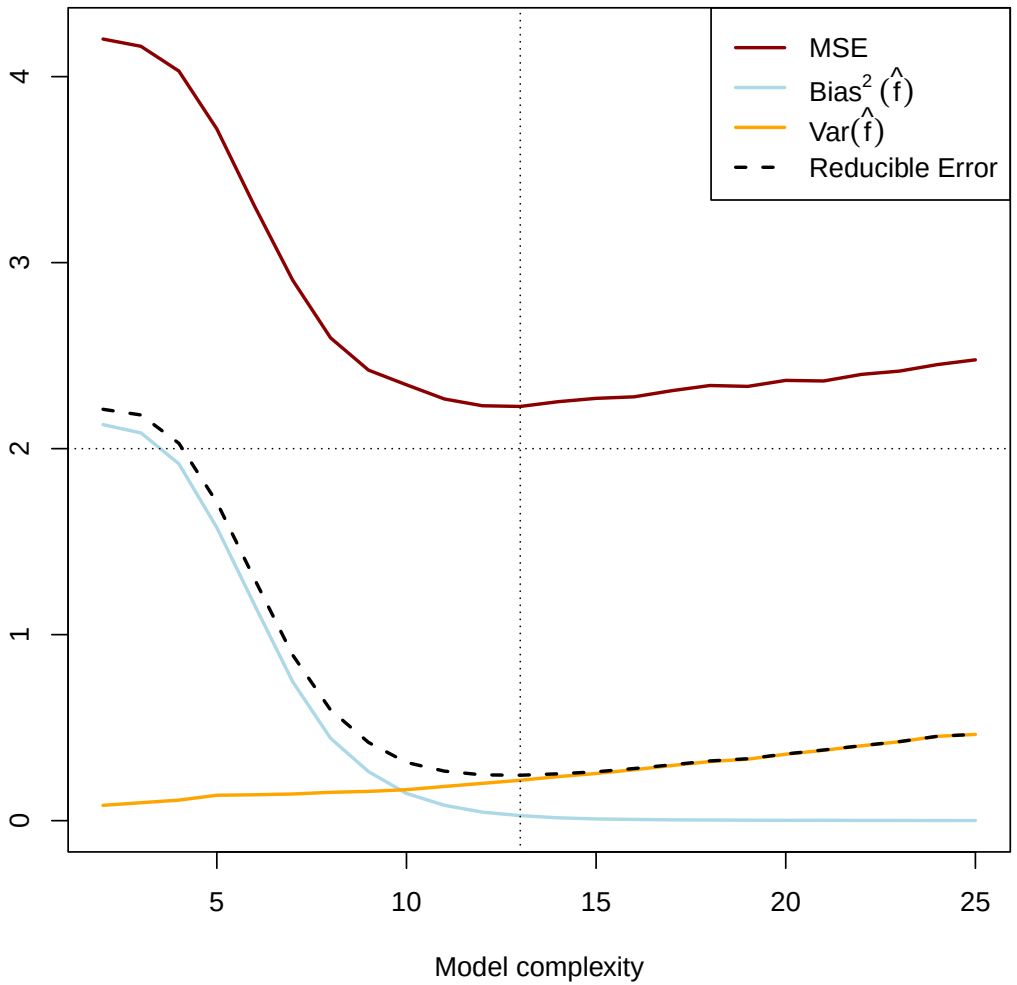
Example 1 – Simulation



This illustrates that, for this model, degrees of freedom is directly proportional to model complexity

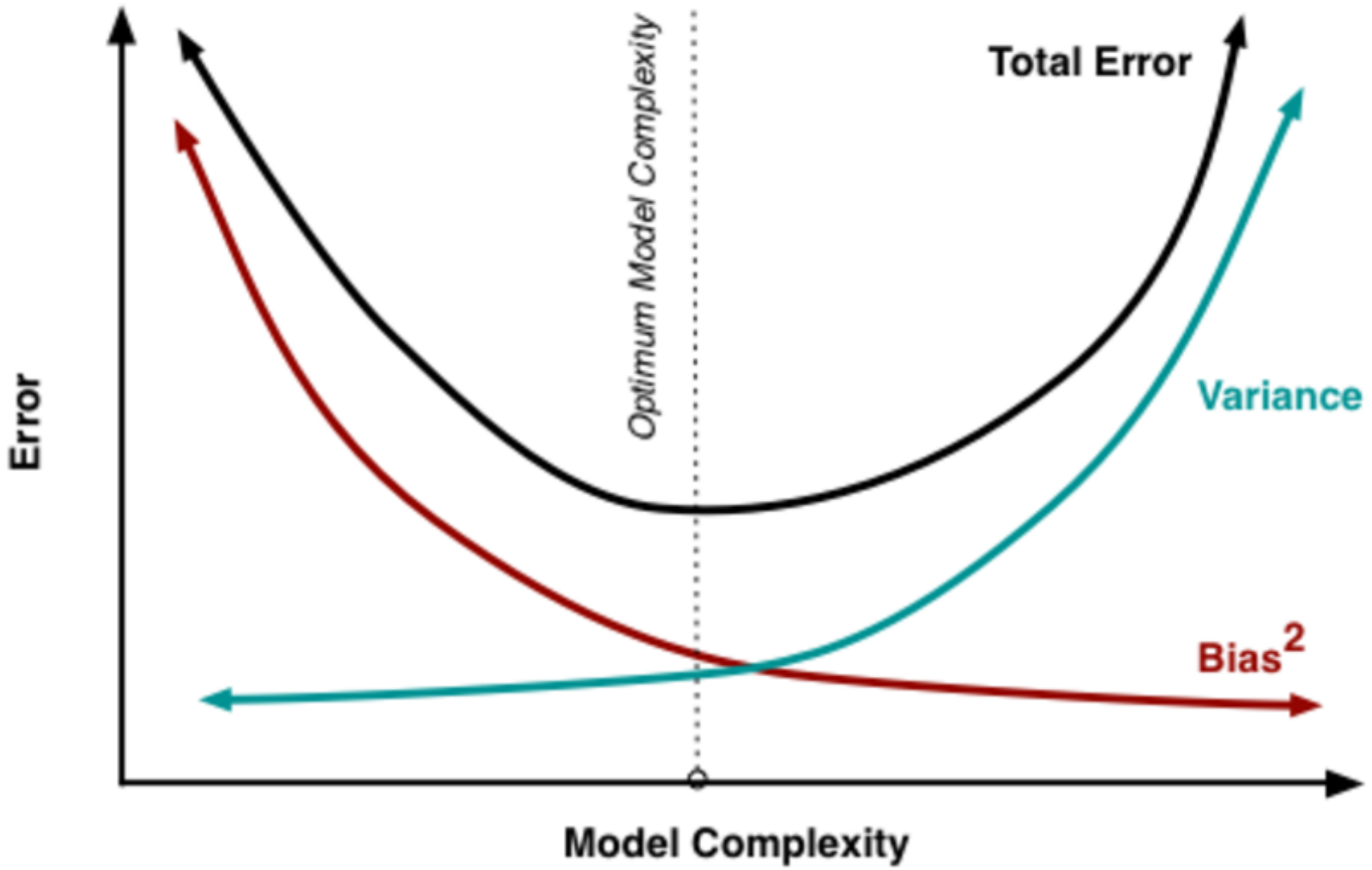
Example 1 – Simulation

To extricate the bias and variance components, we need to observe these models' fit on out-of-sample data across many random realisations of training samples:



Bias-Variance Trade-Off

Here we observed the realisation of a well-known pattern in machine learning:



Bias-Variance Trade-Off

As model complexity/flexibility increases:

- The variance across multiple training samples increases
- The (squared) bias decreases
- $E(\epsilon^2) = Var(\epsilon)$ remains constant

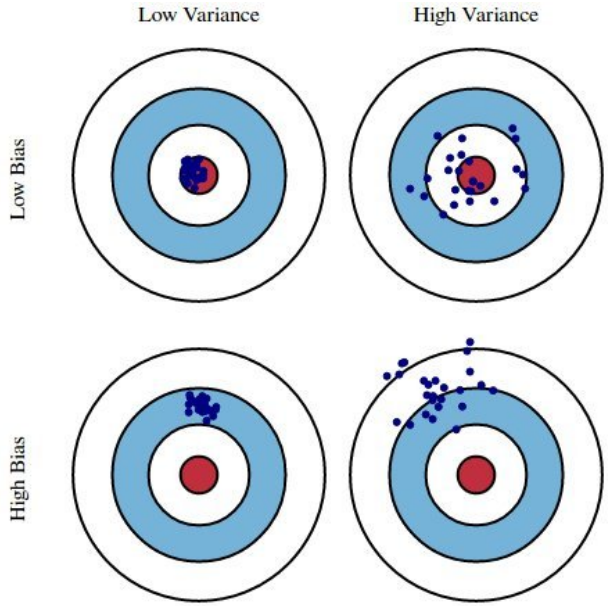
From some complexity/flexibility of $\hat{f}$, the decrease in bias^2 $\left(\hat{f}\right)$ is offset by the increase in $Var\left(\hat{f}\right)$.

At this point the model starts to **overfit** and the test MSE starts increasing.

This is the **bias-variance trade-off**.

Bias-Variance Trade-Off

- The fundamental challenge in statistical learning is to postulate a model of the data that yields both a low bias and variance, whilst policing the model complexity such that the sum of these error components are minimised



- When modelling real-world data, we do not observe f and therefore cannot explicitly compute the test MSE

- To estimate the test MSE, we make use of model validation

Model validation

Two different learning strategies

Model Validation

- When comparing different statistical models, we would like to select the one that we think will work best on unseen test data.
- If we use the test data to make this decision, this will be cheating!
- The result is that we will choose the model best suited to that particular dataset.
- With model validation, we use the training to **validate** our learning, i.e. gauge how well we would do in the test.

Validation Set Approach

One way is to use the validation/hold-out set approach:

1. Leave aside (randomly) a portion of the training data, say 30%
2. Train models on the other 70% of the data only
3. Test the models on the 30%
4. Select the model that yields the lowest validation MSE

Although there are some situations in which this approach is merited, it has two potential flaws:

1. Due to the single random split, the validation estimate of the test error can be highly variable.
2. Since we are reducing our training data, the model sees less information, generally leading to worse performance. Therefore, the validation error may overestimate the test error.

Cross-Validation

To address these issues, we adopt a **cross-validation** (CV) strategy:

1. Randomly divide the entire training set into k groups (folds) of equal size
2. Using fold i as the validation set, train on the other $k-1$ folds and calculate MSE_i
3. Repeat for $i = 1, \ldots, k$ to determine $MSE_1,\ MSE_2, \ldots,\ MSE_k$
4. Calculate the average:

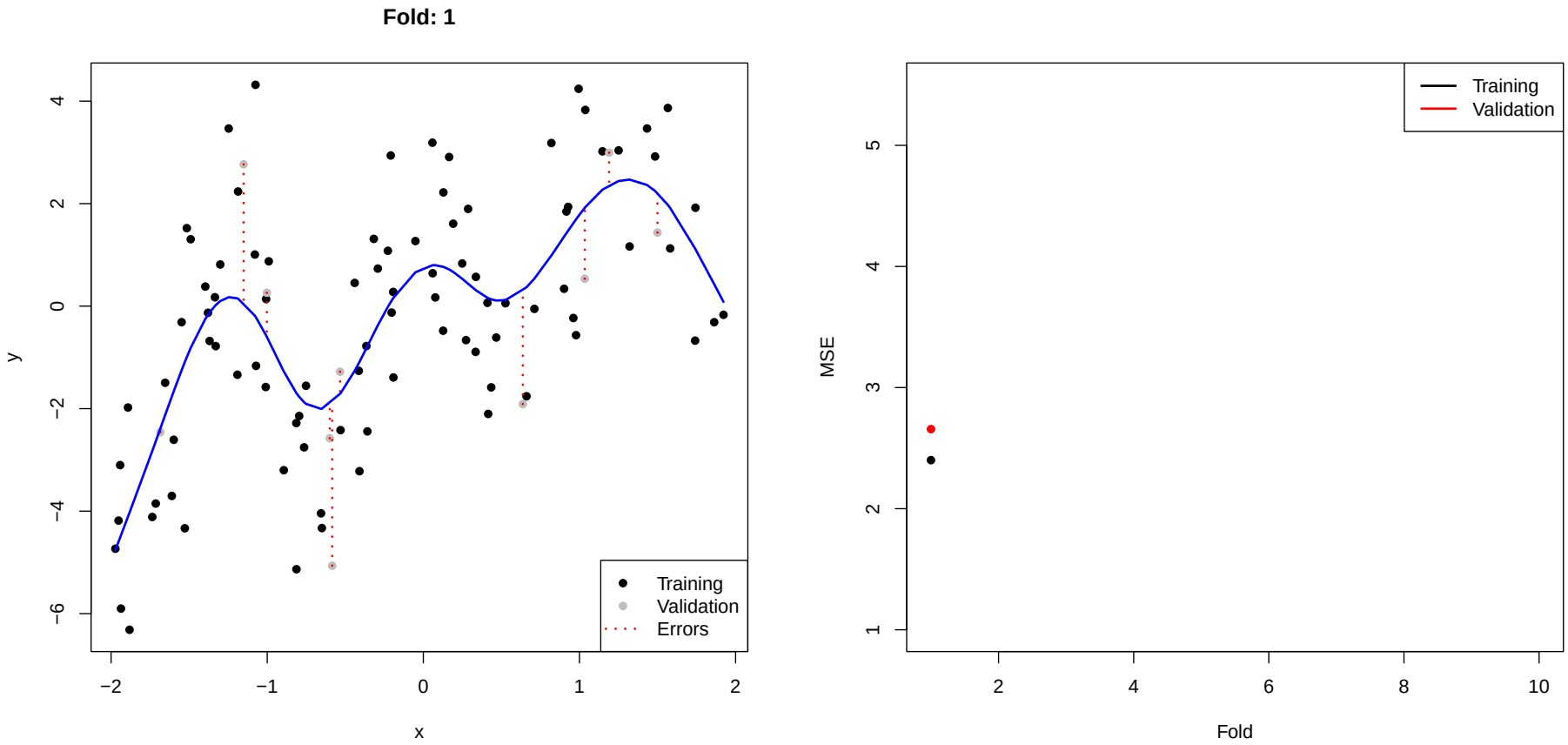
$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} MSE_i$$

Cross-Validation

- Which value of k should we use?
- The same bias-variance dynamics apply to the choice of k in k-fold CV
- It has been shown that $k = 5$ or $k = 10$ yields a good balance such that the test error estimate does not suffer from excessively high bias nor variance
- Setting $k = n$, yields **Leave-one-out cross-validation** (LOOCV)
- Here, the n training sets will be almost identical, such that there will be very high correlation between them
- Larger k also means higher computational costs (more model fits)

Example 1 – Simulation (continued)

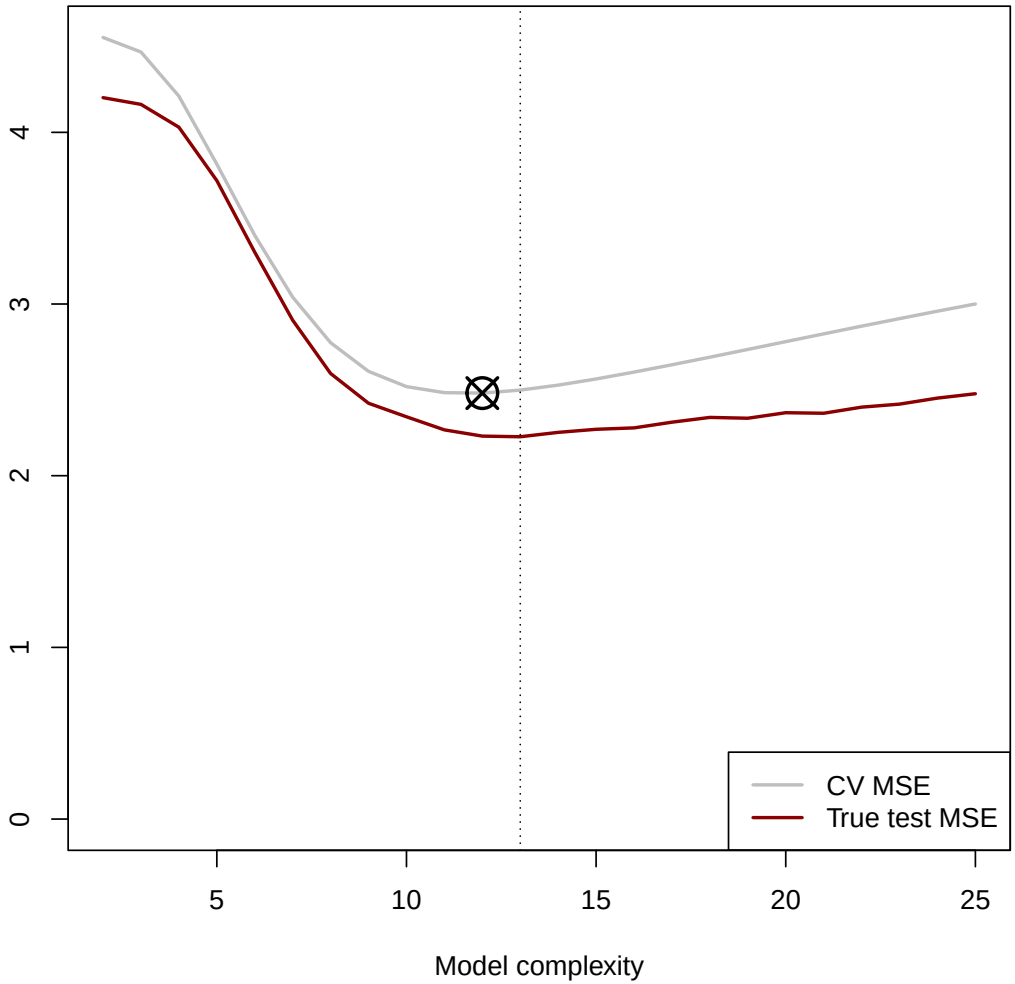
Returning to our previous examples, apply 10-fold CV to a cubic spline with 8 degrees of freedom:



As expected, the validation error is noticeably more variable than the training error across the folds

Example 1 – Simulation (continued)

CV error minimum is reached at $dof = 12$



Because we simulated these data, we know the best model has $dof = 13$

Conclusion

- This chapter focused on the bias-variance trade-off and using cross-validation (CV) to estimate the out-of-sample performance of models of the same form, but different complexity
- Each model was considered a separate hypothesised representation of the underlying function – f – mapping all the explanatory variables (features) to the dependent (target) variable
- In the following chapter we will turn our attention to **variable selection**, i.e. deciding which features to include in the model