Computational Methods in Astrophysics

Dr Rob Thacker (AT319E)

thacker@ap.smu.ca

Intro to machine learning

- Excellent & useful text: Introduction to Statistical Learning by James et al (6th edition, 2015)
  - Many examples in R (book built around it)
  - Highly recommend purchasing a copy
  - Most of these notes are taken from it
- Key point – machine learning and statistical learning are very similar, but the latter is often more formal
  - Being good at either (lot of overlap!) requires both a detailed knowledge of stats and algorithms
  - Machine learning community generally more interested in new techniques rather than validation, tends to come from more CS angle
  - Also suggestion that ML more interested in prediction, SL more interested in inference
- With "Big Data" such a hot topic both communities are extensively interested in it

Statistical learning

- Formally began in 1960s
  - But one can legitimately argue least squares going all the way back to Gauss/Legendre is where things started (~1805)
  - Note: neural networks predate – 1940s, 50s
- Original focus on function estimation based on data
- Blossomed in 1990s
  - i.e. Support vector machine algorithms spurred interest in multidimensional fitting and algorithm development
- Papers are formal, lots of concerns about proof and model validity
  - Evolution from math community is clear

Data & notation

- Helpful to formalize things just a little bit
- Denote all data by **X**, p variables, n samples so x_{ij} are components of the matrix **X:**

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix}.$$

  - **Remember:** row vector is ith sample (left example)
  - **Column vector:** subsample of a given variable (right)

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix}.$$

$$\mathbf{x}_j = \begin{pmatrix} x_{1j} \\ x_{2j} \\ \vdots \\ x_{nj} \end{pmatrix}.$$

Data II

- The bolding of the subsample leads to the following notation with bold **x (i.e. column vector of variables)**:

$$\boldsymbol{X} = (\boldsymbol{x}_1 \quad \dots \quad \boldsymbol{x}_n)$$

- Or equivalently (T=transpose) and x are not bolded, so representing a single sample vector but transposed into row

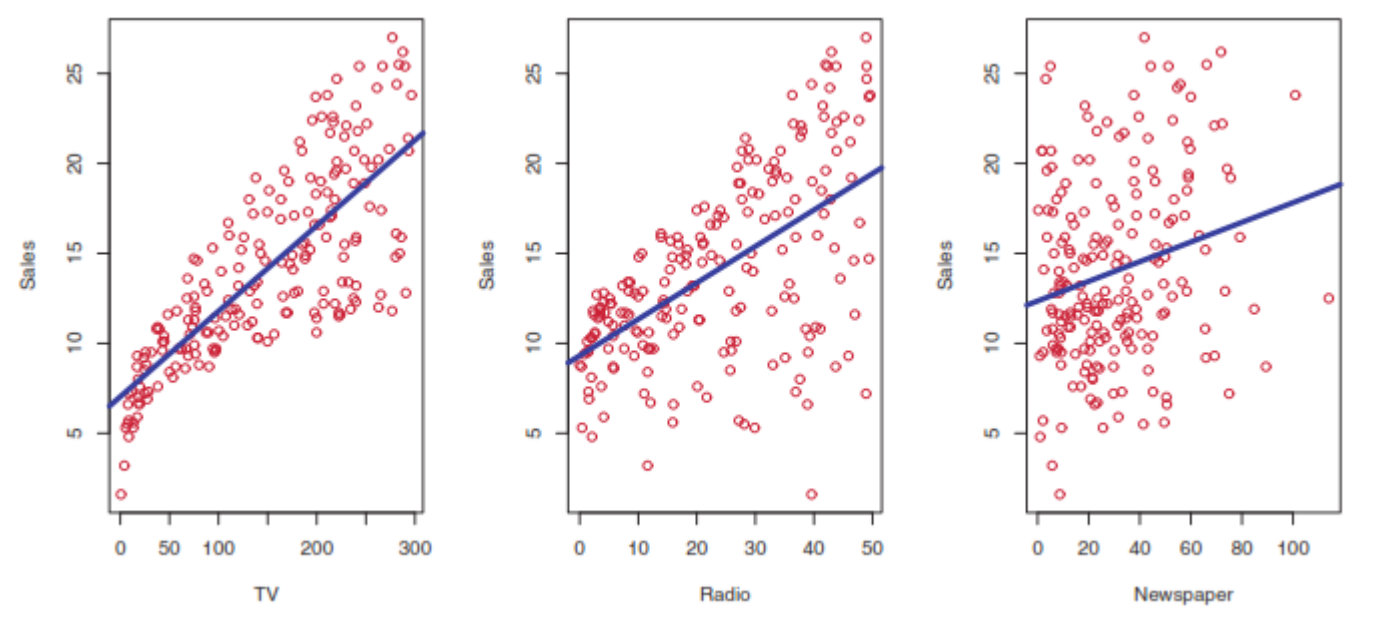
$$\boldsymbol{X} = \begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix}$$

Nomenclature warnings

- Note it's more important that you understand the data representations than the precise definitions
- Different sub-fields may have different representations
- So don't get wedded to one particular formalism
- In particular – capitals are often used to denote the set of all values of a given variable
  - If you're not sure what's being represented don't carry on reading - figure it out first!

Example

- Consider a situation where sales volume is to be predicted on the basis of marketing budgets (TV, radio, newspaper)
  - Sales is the output variable denoted Y
  - Marketing budgets are the input variables denoted X_1,X_2,X_3



Example II

- Nomenclature (used interchangeably):
  - output variable = "response" = "dependent variable"
  - Input variables = "indep. Variables" = "predictors" = "features"
- Y values obviously pair with values of X_1,X_2,X_3 and we assume there is some relationship X=(combination of X values)

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

- Epsilon represents a random error term (indep of X) of mean zero
  - f encompasses the systematic information X provides about Y
  - The previous plots are not considering f's that depend on more than one predictor

Inference vs prediction

- There is a subtle difference
  - In essence inference wants to know about "process" while prediction is more focused on "outcomes"
- Consider predicting (adverse) patient reactions (Y) to a given drug based on blood variables (X)
  - The precise Y=f(X) really can't be studied in detail
  - So since error term averages to zero, we assume a prediction

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

  - $\hat{Y}$ is the prediction of Y, $\hat{f}$ is an estimate of f
- What would we expect if we do this?

Prediction: Reducible errors

- Expectation of squares of differences is then

$$E[(Y - \hat{Y})^2] = E[(f(X) + \epsilon - \hat{f}(X))^2]$$

$$E[(Y - \hat{Y})^2] = E[(f(X) - \hat{f}(X))^2] + Var(\epsilon)$$

- First term on RHS is reducible error
  - Better and better estimates of f reduce that error
  - That's the goal of prediction
- Second term is the irreducible error
  - No matter how good $\hat{f}$ is you can't remove it

Inference

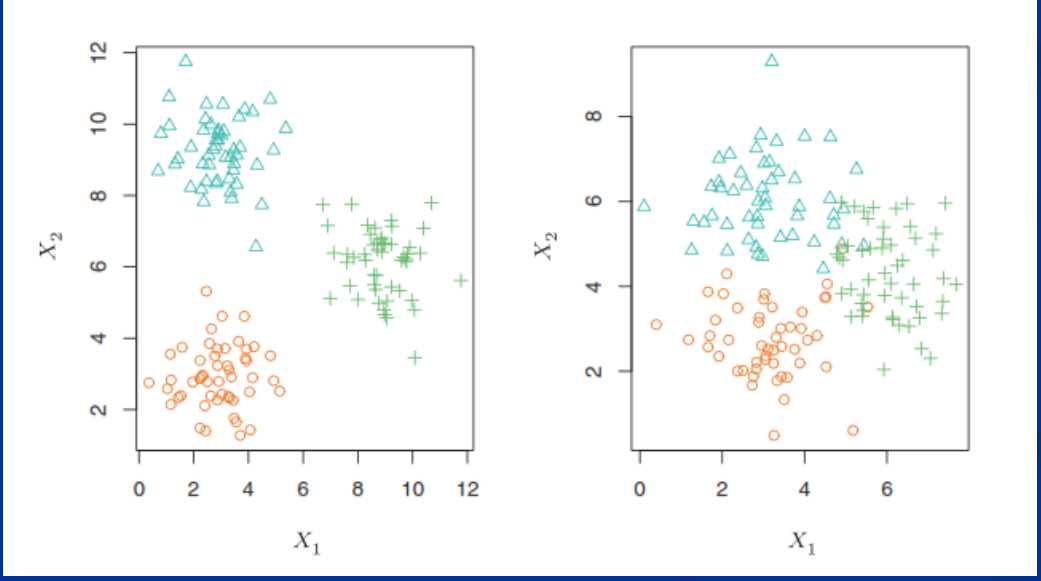
- Suppose rather than knowing Y we want to know how it changes or is impacted by the various X
  - Which predictors are associated with the response?
    - Hopefully only a small number are
  - What is the precise relationship between predictors and response?
    - Could we consider linear? Do we need more complex?
- Consider the advertising example. Inference allows you to ask:
  - Which variables are most important?
  - Which sales medium produces biggest boost in sales?
- Caution: there is a somewhat blurry line once you start using inference to make predictions.

Supervised learning

- Thus far we've considered situations with (Y,X) pairs
  - Every predictor has an associated response
  - Frequently called the training data
- This is known as supervised learning
  - Whether or not you are doing inference or prediction
- Linear regression is a good example of supervised learning
  - More complicated algorithms like support vector machines are also examples of supervised learning
- But what if we don't know what the response function is – can we learn anything?
- Can also have semi-supervised learning where only some Y are known

Unsupervised learning

- Without response variable you can't fit to anything – regression is not possible, for example
- But we can learn about
  - Relationships between variables
  - Relationships between observations (samples)



Clustering is a good example.
Do variables fall into distinct ranges, and how do these relate to the underlying data?
Quite what the separate clusters may mean is not clear.
This is a discovery process.

Stats recap: Para vs non-para

- Two approaches to calculating response function
  - Physics/astro relies on parametric approach a lot of the time
- Parametric methods are essentially a two-step process
  - 1) Assume a functional for f

    e.g. linear model over p variables:

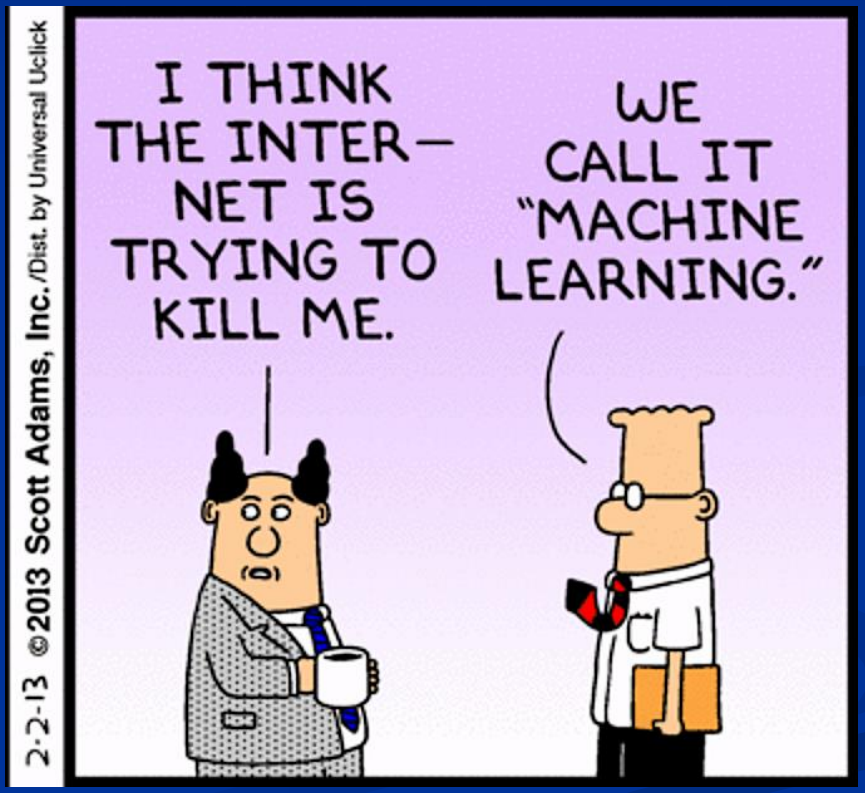
    $$f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$$
  - 2) Given training data need a way to constrain the β values
    - Least squares is one simple way of doing this
- This approach usually makes things straightforward (in principle)
  - Drawback is that rarely is f(X) truly a linear function
- More complex functions are possible but can suffer problems
  - "overfitting" – where function follows features that are really noise
  - More parameters need to be fit as well
    - But the number of parameters is fixed

Non-Parametric

- What if we fit models that can change their functional form as we add more samples?
  - Sounds great – except remember the "over-fitting" concern
- Simple example: Fit an n-point moving average
  - Drawbacks: need ordered data, wiggly, poorly defined at end points (need equally spaced data too)
  - Function is defined by all the samples, gets progressively larger with increase in sample space
  - Can make more complicated using different weights
    - Weight more distant points less (`kernel smoothers')
  - Splines can be used as well lots of choices

How can we not have Dilbert?

- A little "bit" of humour



Non-parametric – a note

- Many people associate "non parametric" with ranked tests
  - e.g. Spearman's rank correlation
- Or with ordinal data, that while having a distinct order, doesn't have an obvious measure between categories
  - e.g. stress scale in patients
- Parametric models require data to be specified on an interval, or at least have defined intervals between data
- Non parametric methods also have advantages in smaller samples
  - Not a concern here
- My take away is that "non parametric" in the machine/statistical learning sense, often means a non parametric model

Regression vs classification

- Data can be quantitative or categorical (qualitative)
  - Quantitative: height, mass
  - Categorical: gender, eye colour (often called "classes")
- If data is quantitative we usually talk about regression
  - Least squares is a simple example
- For categorical data, the task is classification
  - Frustratingly, some approaches are still called regressions (consider yourselves warned)
  - e.g. "logistic regression" uses categorical data
    - The subtlety arises from estimating class probabilities – that allows you to define continuous functions that sample to discrete outcomes

Accuracy

- "No free lunch"
- No single statistical test or method is "best"
  - Inevitably we tend to use what we know!
- Of course we need measures of quality of fit: e.g. Mean square error:

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

- This is easily evaluated for the training data – but who cares?
  - We want to know about non-training data denoted y_0,x_0 (need sample)
- The test MSE rather than training MSE is what matters!
  - You may have test data available to use
  - If you don't just minimizing the MSE on training data may not be a good idea? Why?

Beautiful example from ISL

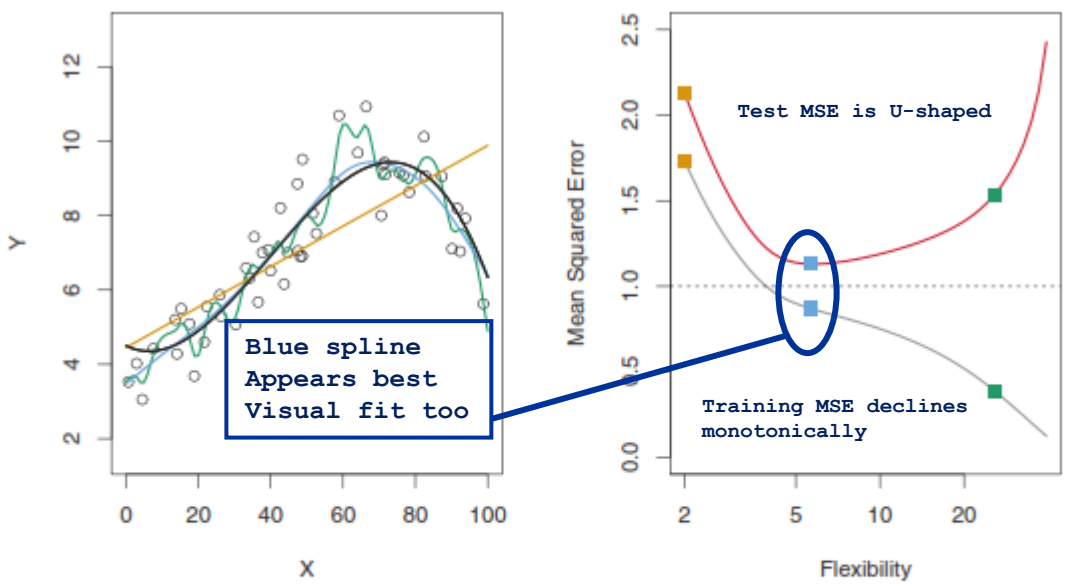


FIGURE 2.9. Left: *Data simulated from f, shown in black. Three estimates of f are shown: the linear regression line (orange curve), and two smoothing spline fits (blue and green curves).* Right: *Training MSE (grey curve), test MSE (red curve), and minimum possible test MSE over all methods (dashed line). Squares represent the training and test MSEs for the three fits shown in the left-hand panel.*

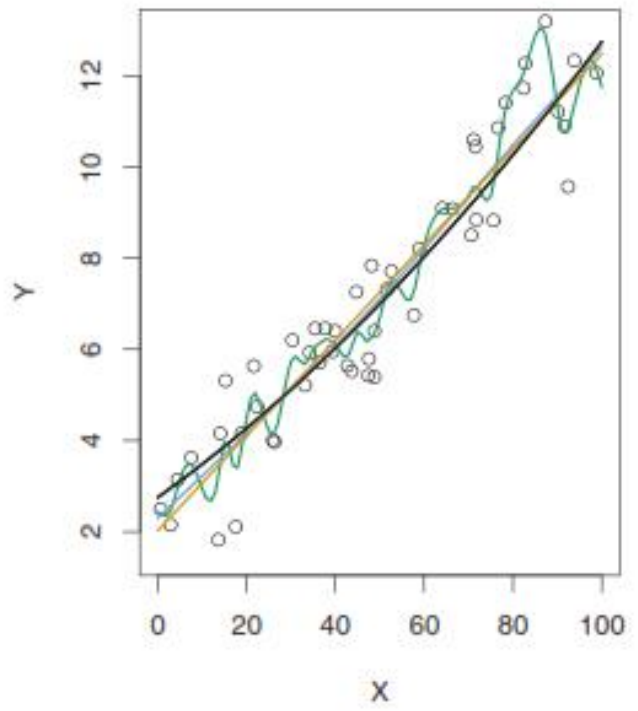
- True data = black+random
- Orange = least squares linear
- Blue, green progressively higher order smoothing splines

To much flexibility is not good – too much movement, too little is not either…

Huh!? Don't try too hard!

- Lesson #1: The training MSE will be (except for freak situations) lower than the test MSE
- Lesson #2: The U-shape is a general feature of the MSE on test vs training data
- Lesson #3: Overfitting is a problem – noise is interpreted as signal
- How can we estimate the test MSE from the training MSE?
  - Tough question – so-called "cross validation" is one approach
    - Idea is to sub-divide training set into training and test set

Linear model example



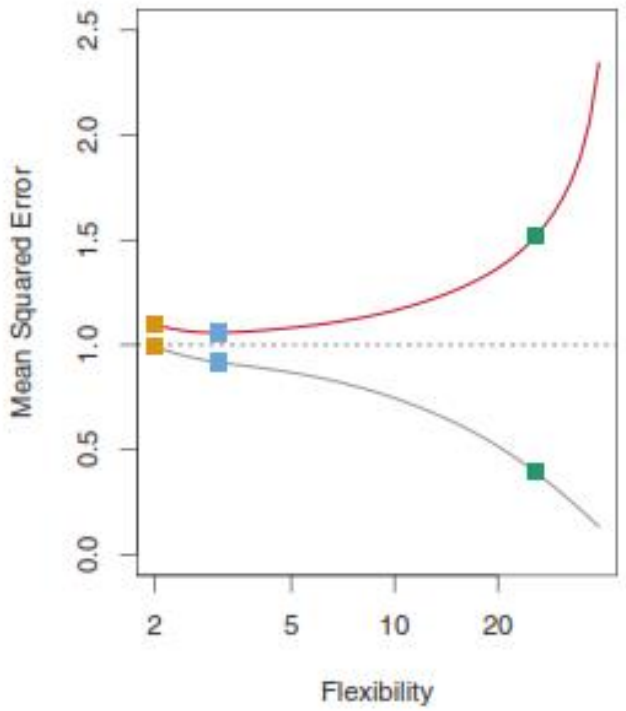
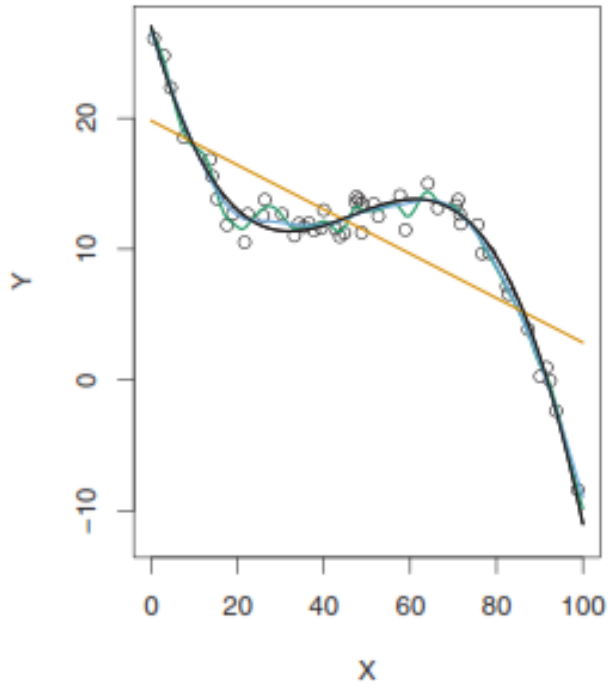


FIGURE 2.10. *Details are as in Figure 2.9, using a different true f that is much closer to linear. In this setting, linear regression provides a very good fit to the data.*

Strongly non-linear example



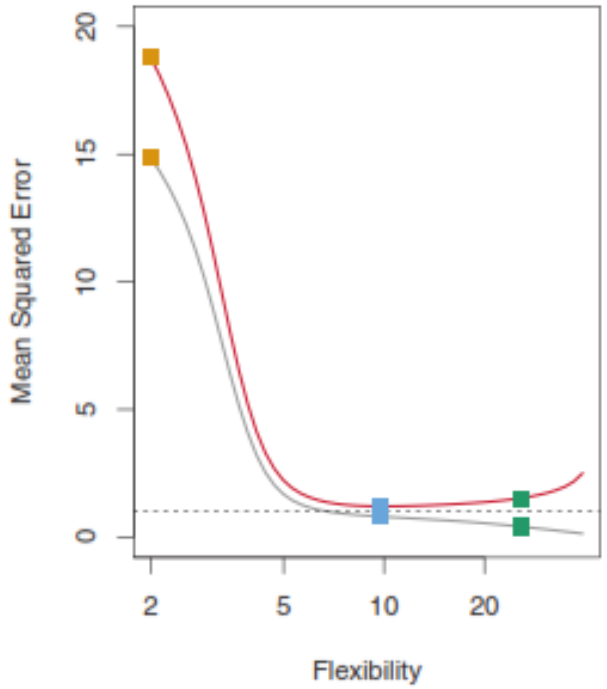


FIGURE 2.11. *Details are as in Figure 2.9, using a different f that is far from linear. In this setting, linear regression provides a very poor fit to the data.*

- This example also has smaller random component

Bias/variance tradeoffs

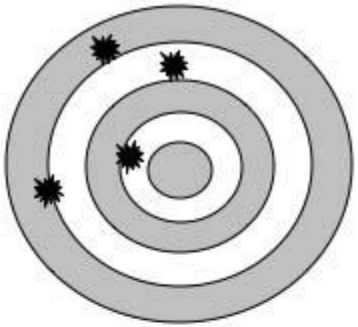
- "Beyond the scope of the book" to show that the MSE on repeated test data, x_0, can be decomposed as follows:

$$E[(y_0 - \hat{f}(x_0))^2]$$
$$= Var\left(\hat{f}(x_0)\right) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon)$$

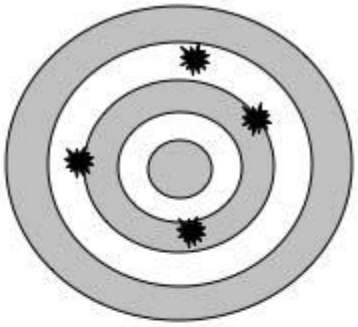
- Where $$Bias\left(\hat{f}(x_0)\right) = E\left(\hat{f}(x_0)\right) - y_0$$

- Variance determines how much things would change if we choose a different training set
- Bias reflects the accuracy of the underlying model assumptions
  - e.g. A highly non-linear problem isn't well fit by a linear model

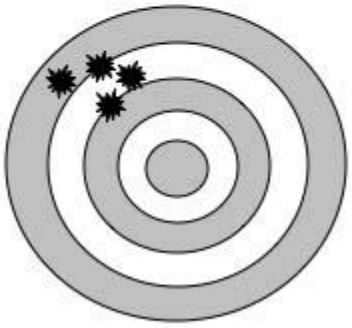
Bias vs variance



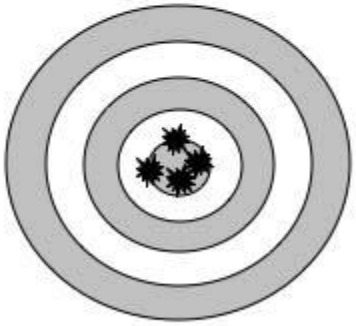
High Bias
High variance



Low Bias
High variance



High Bias
Low variance



Low Bias
Low variance

Bias/variance tradeoffs II

- As a general rule:
  - The more flexibility in the method the more the variance will increase
  - The bias will decrease
- From a low amount of flexibility, bias decreases faster than variance increases
  - Initially, as flexibility is increased, MSE usually declines
  - Beyond a certain point bias improvements "stop" and variance takes over
  - Produces a U-shaped distribution

Bias/variance tradeoffs III

- Here are graphs of bias & variance for the "sine-like", linear and non-linear examples considered earlier

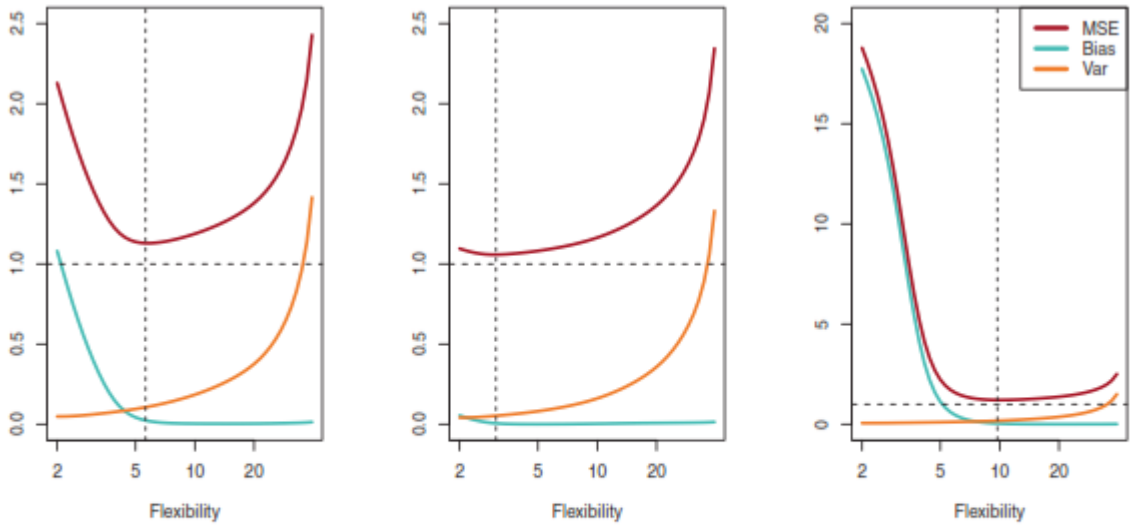


FIGURE 2.12. *Squared bias (blue curve), variance (orange curve), Var(ϵ) (dashed line), and test MSE (red curve) for the three data sets in Figures 2.9–2.11. The vertical dotted line indicates the flexibility level corresponding to the smallest test MSE.*

Highlights the challenges of fitting well:
Easy to find low bias – just choose very flexible solution.
Finding lowest variance is harder.

Excellent way to think about the general problem of fitting!

Here's the kicker – in general we don't have f so we can't do this explicitly anyway!

Categorical variables

- For categorically variables we can test the number of times a class is predicted correctly – training error rate
  - Define the indicator I to be 0 if training data *matches* predicted, 1 if it fails

$$TER = \frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

- Once we use the test data, test error rate is

$$\text{Ave}(I(y_0 \neq \hat{y}_0))$$

- Clearly a good classifier is one for which the test error rate is close to minimal

Bayes Classifier

- The test error rate is minimized by a simple idea:
  - Assign each observation to the most likely class given its predictor values
  - i.e. we put a test observation in the class j for which

$$P(Y = j|X = x_0)$$

    is largest
  - i.e. the conditional probability that Y=j given X=x_0
- This is the "Bayes Classifier"
- For 2 classes P=0.5 into one class, P=0.5 into another defines the Bayes Decision Boundary
- For simulated data we know how they were generated, and thus can evaluate conditional probabilities
- For real data we cannot know the distributions, and so we must approximate

Fully computed Bayes Classifier

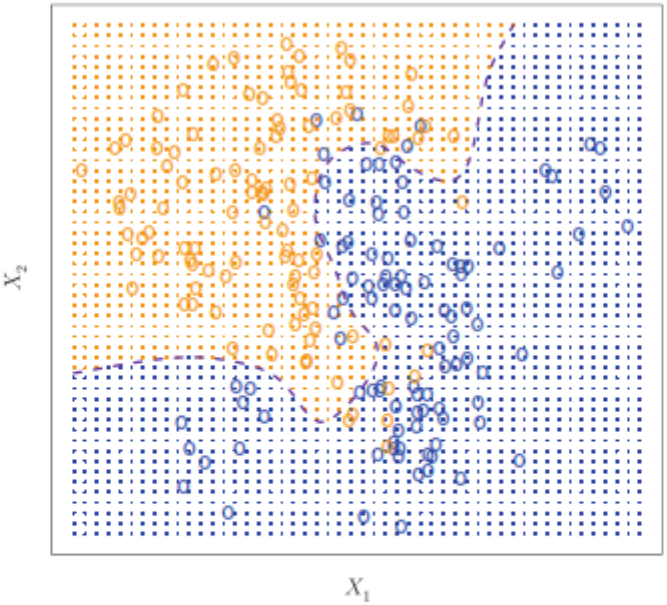


FIGURE 2.13. *A simulated data set consisting of* 100 *observations in each of two groups, indicated in blue and in orange. The purple dashed line represents the Bayes decision boundary. The orange background grid indicates the region in which a test observation will be assigned to the orange class, and the blue background grid indicates the region in which a test observation will be assigned to the blue class.*

K-nearest neighbour

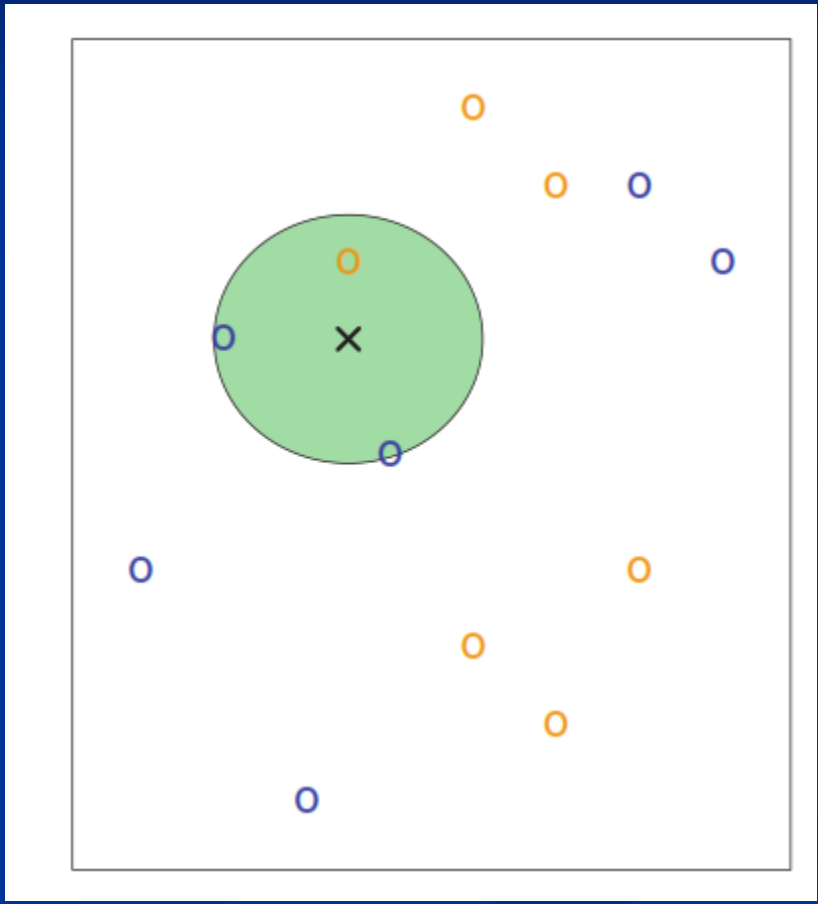
- Goal: estimate the conditional distribution of Y given X, then classify observation to class with highest estimated probability
- Consider an estimator that uses K nearest samples to a point

$$P(Y = j|X = x_0) = \frac{1}{K} \sum_{i \in nearest} I(y_i = j)$$

- Consider example

K-nearest neighbour II

- Point X has 2 blue and 1 orange nearest
- 2/3 prob for blue, 1/3 for orange
- Assign X to blue!
- Can vary number of points as well



Comparison KNN vs BC

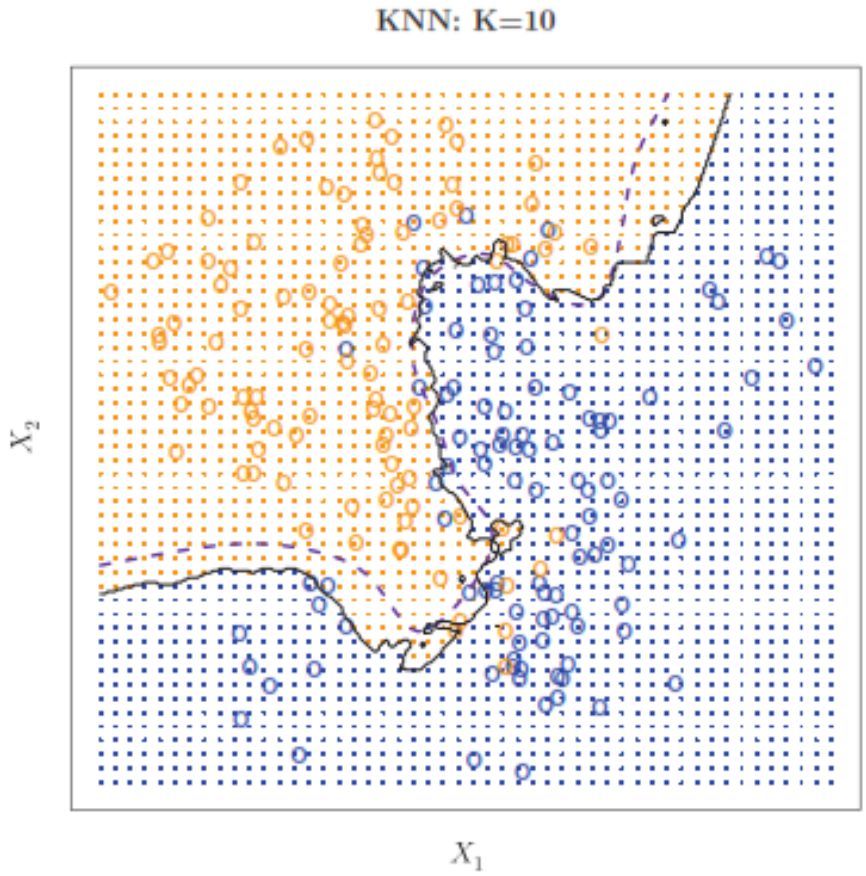


FIGURE 2.15. *The black curve indicates the KNN decision boundary on the data from Figure 2.13, using $K = 10$. The Bayes decision boundary is shown as a purple dashed line. The KNN and Bayes decision boundaries are very similar.*

KNN is sensitive to K choice

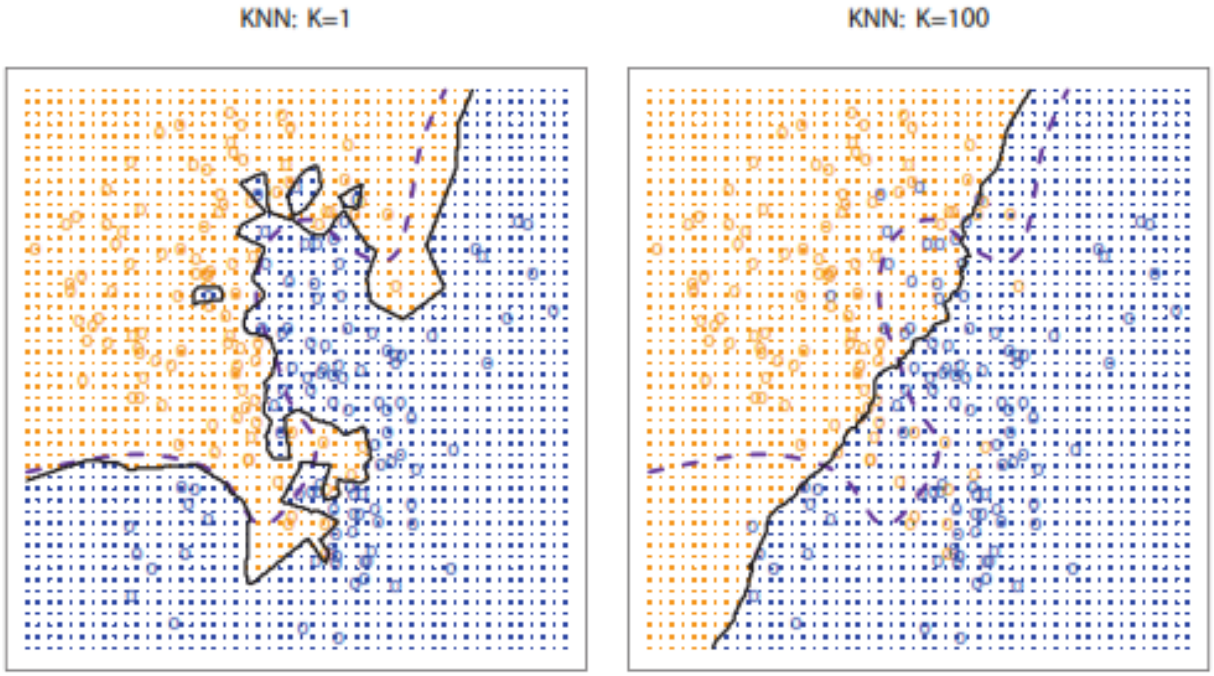
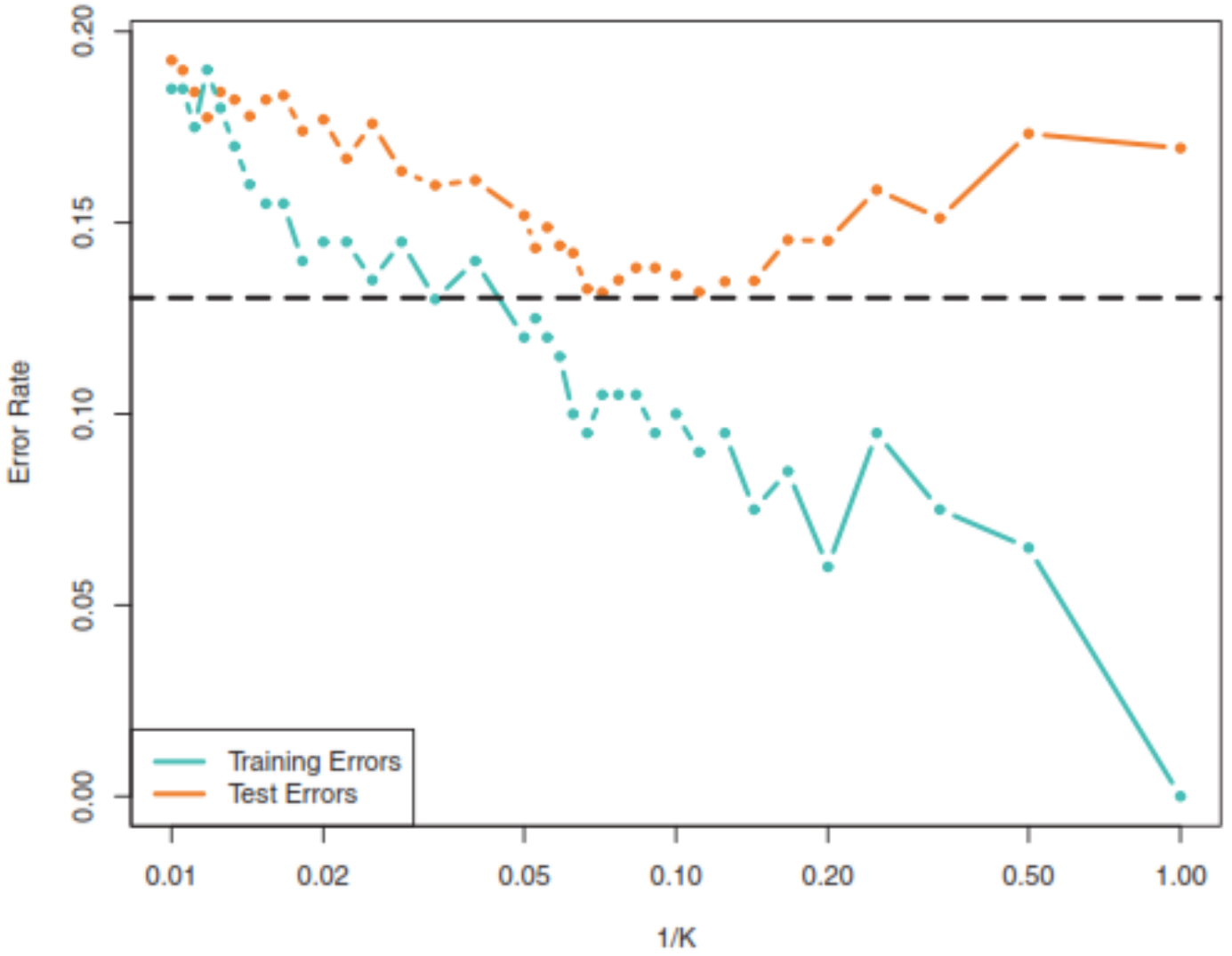


FIGURE 2.16. *A comparison of the KNN decision boundaries (solid black curves) obtained using $K = 1$ and $K = 100$ on the data from Figure 2.13. With $K = 1$, the decision boundary is overly flexible, while with $K = 100$ it is not sufficiently flexible. The Bayes decision boundary is shown as a purple dashed line.*

KNN sensitive to K choice

- K=1 is "too flexible" finds structure that is not present in the true BDB
  - High variance, low bias
- Increasing to K=10 showed good agreement
- K=100 is the opposite to K=1, low variance, high bias
- The error rates follow a similar pattern to what we saw for MSE
  - K=1 is 0.1695, K=100 is 0.1925
  - BUT! K=10 is 0.1363 – lower than both

Test and training errors



Summary

- SL/ML have a lot in common – one more formal than the other
- SL methods focusing on improving the reducible error – the irreducible error is beyond our means to change
- Most problems we consider are supervised learning – we have a training set of values to learn from
  - Unsupervised learning is about discovering relationships
- Model fitting breaks down into bias and variance
  - Increasing flexibility increases variance, but reduces bias producing U shaped curve
- For categorical variables, where the form of f is unknown, we can use KNN methods to explore parameter space