MONASH University

ETC3250

Business Analytics

Statistical learning

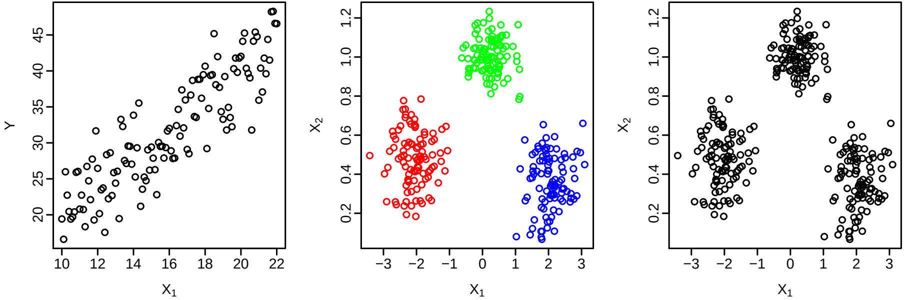
28 February 2018

Outline

Learning from data

- **Better understand** or **make predictions** about a certain phenomenon under study
- **Construct a model** of that phenomenon by finding relations between several variables
- If phenomenon is complex or depends on a large number of variables, an **analytical solution** might not be available
- However, we can **collect data** and learn a model that **approximates** the true underlying phenomenon

Learning from a dataset



$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N} \text{ with } x_i = (x_{i1}, \ldots, x_{ip})^T$$

Statistical learning provides a framework for constructing models from $\mathcal{D}$.

Different learning problems

- Supervised learning
  - Regression (or prediction)
  - Classification

  $\rightarrow y_i$ available for all x_i
- Unsupervised learning

  $\rightarrow y_i$ unavailable for all x_i
- Semi-supervised learning

  $\rightarrow y_i$ available only for few x_i
- Other types of learning: reinforcement learning, online learning, active learning, etc.

Identification of the best learning problem is important in practice

Supervised learning

$$\mathcal{D} = \{(y_i, x_i)\}_{i=1}^{N},$$

where

$$(y_i, x_i) \sim P(Y, X) = P(X)\underbrace{P(Y|X)}.$$

- Y: response (output)
- $X = (X_1, \ldots, X_p)$: set of p predictors (input)

We seek a function $h(X)$ for predicting Y given values of the input X. This function is computed using $\mathcal{D}$.

Supervised learning

$$\mathcal{D} = \{(y_i, x_i)\}_{i=1}^N \text{ where } (y_i, x_i) \sim P(Y, X)$$

We are interested in minimizing the expected *out-of-sample* prediction error:

$$\mathrm{Err}_{\mathrm{out}}(h) = E[L(Y, h(X))]$$

where $L(y, \hat{y})$ is a non-negative real-valued *loss function*. Examples include $L(y, \hat{y}) = (y - \hat{y})^2$ and $L(y, \hat{y}) = I(y \neq \hat{y})$. In other words, the goal is to compute

$$h^* = \mathrm{argmin}_{h \in \mathcal{H}} \mathrm{Err}_{\mathrm{out}}(h).$$

Since we do not know P, we can compute

$$\hat{h} = \mathrm{argmin}_{h \in \mathcal{H}} \mathrm{Err}_{\mathrm{in}}(h) \text{ where } \mathrm{Err}_{\mathrm{in}}(h) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, h(x_i)).$$

Supervised learning - regression

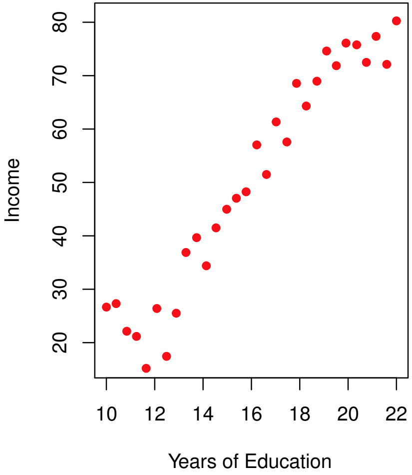
We often assume that our data arose from a statistical model

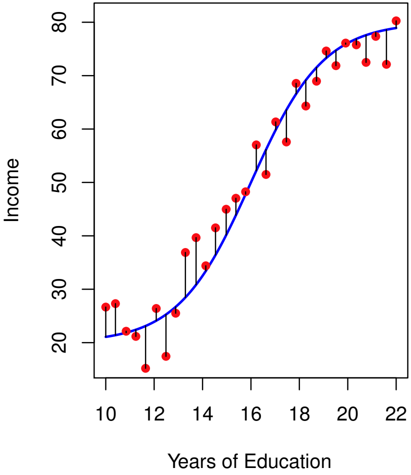
$$Y = f(X) + \varepsilon,$$

where f is the true unknown functoin, ε is the random error term with $E[\varepsilon] = 0$ and is independent of X.

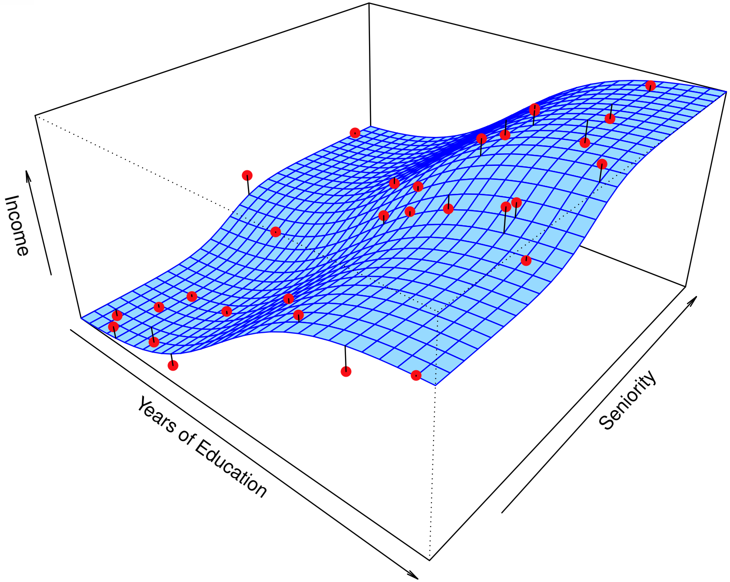
- The additive error model is a useful approximation to the truth
- $f(x) = E[Y|X = x]$
- Not a deterministic relationship: $Y = f(X)$

Supervised learning - regression





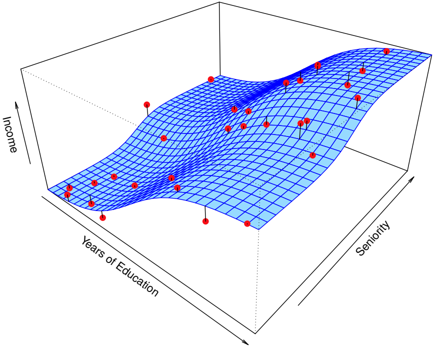
Supervised learning - regression

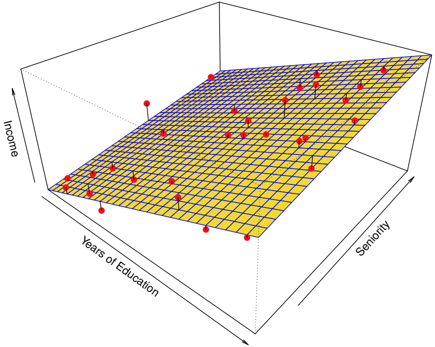


Why estimate f?

- **Prediction**:
  - $\hat{y}_* = \hat{f}(x_*)$ for a new observation x_*
- **Inference (or explanation)**:
  - Which predictors are associated with the response?
  - What is the relationship between the response and each predictor?

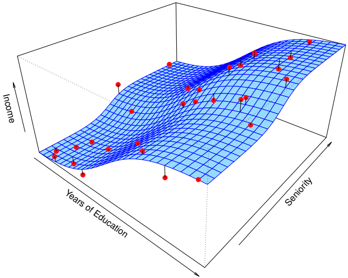
Regression - estimation of f?

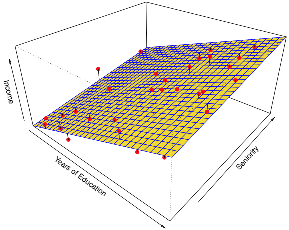


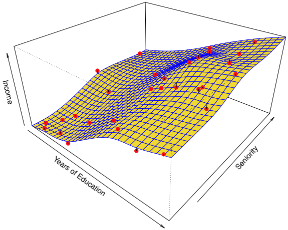


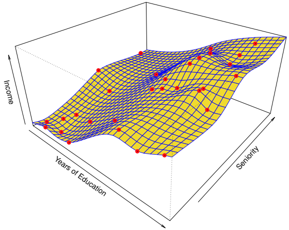
$$\hat{f}(\text{education}, \text{seniority}) = \hat{\beta}_0 + \hat{\beta}_1 \times \text{education} + \hat{\beta}_2 \times \text{seniority}$$

Regression - estimation of f?



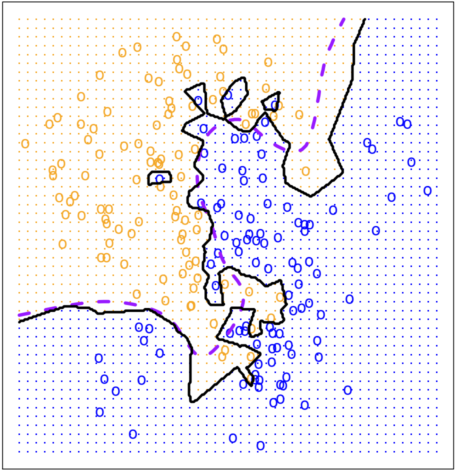




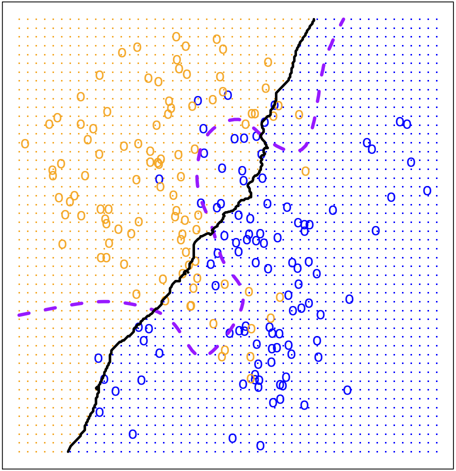


Classification - estimation of f?

KNN: K=1



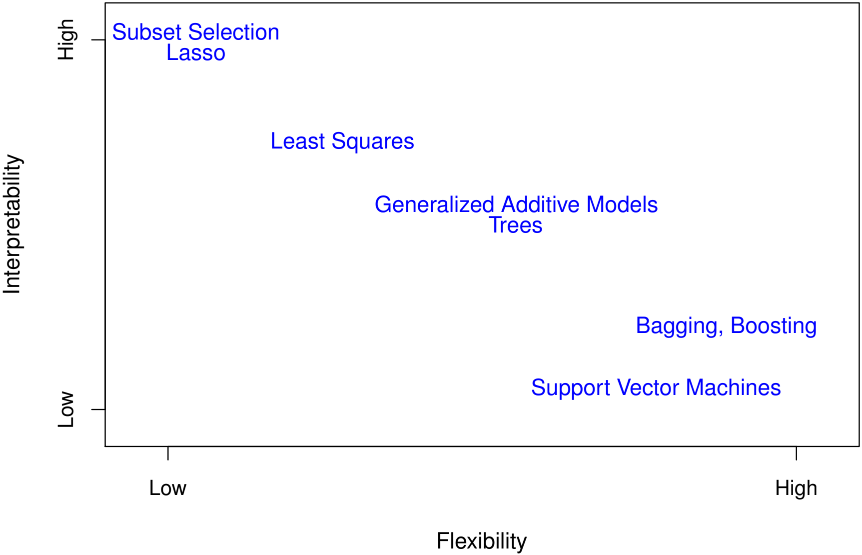
KNN: K=100



Estimation methods

- **Parametric methods**
  - Assumption about the form of f, e.g. linear: $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$ and $\hat{Y}(x) = \hat{f}(x)$
  - ☺ The problem of estimating f reduces to estimating a set of parameters
  - ☺ Usually a good starting point for many learning problems
  - ☹ Poor performance if linearity assumption is wrong
- **Non-parametric methods**
  - No *explicit* assumptions about the form of f, e.g. nearest neighbours: $\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$
  - ☺ High flexibility: it can potentially fit a wider range of shapes for f
  - ☹ A large number of observations is required to estimate f with good accuracy

Model Interpretability vs flexibility



Outline

Regression problems

Suppose we have a regression model $y = f(x) + \varepsilon$.

Estimate $\hat{f}$ from some **training data**, $Tr = \{x_i, y_i\}_1^n$.

One common measure of accuracy is:

Training Mean Squared Error

$$\mathrm{MSE}_{Tr} = \underset{i \in Tr}{\mathrm{Ave}}[y_i - \hat{f}(x_i)]^2 = \frac{1}{n}\sum_{i=1}^{n}[(y_i - \hat{f}(x_i)]^2$$

Regression problems

Suppose we have a regression model $y = f(x) + \varepsilon$.

Estimate $\hat{f}$ from some **training data**, $Tr = \{x_i, y_i\}_1^n$.

One common measure of accuracy is:

Training Mean Squared Error

$$\mathrm{MSE}_{Tr} = \underset{i \in Tr}{\mathrm{Ave}}[y_i - \hat{f}(x_i)]^2 = \frac{1}{n}\sum_{i=1}^{n}[(y_i - \hat{f}(x_i)]^2$$

Measure real accuracy using **test data** $Te = \{x_j, y_j\}_1^m$

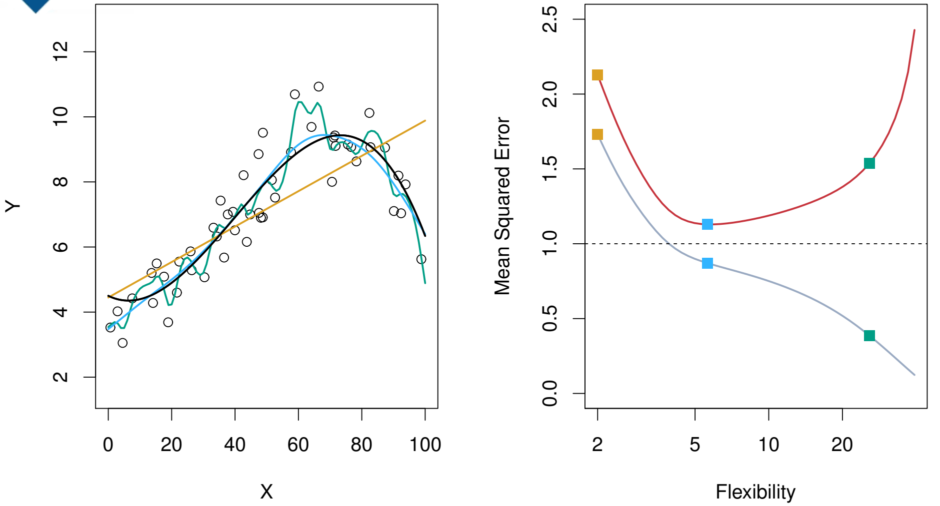
Test Mean Squared Error

$$\mathrm{MSE}_{Te} = \underset{j \in Te}{\mathrm{Ave}}[y_j - \hat{f}(x_j)]^2 = \frac{1}{m}\sum_{j=1}^{m}[(y_j - \hat{f}(x_j)]^2$$

Training vs Test MSEs

- In general, the more flexible a method is, the lower its training MSE will be. i.e. it will “fit” the training data very well.
- However, the test MSE may be higher for a more flexible method than for a simple approach like linear regression.
- Flexibility also makes interpretation more difficult. There is a trade-off between flexibility and model interpretability.

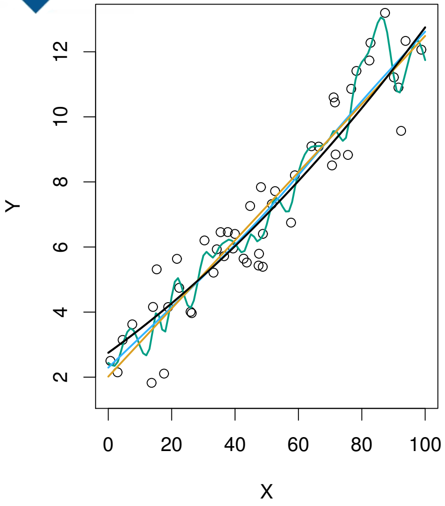
Example: splines

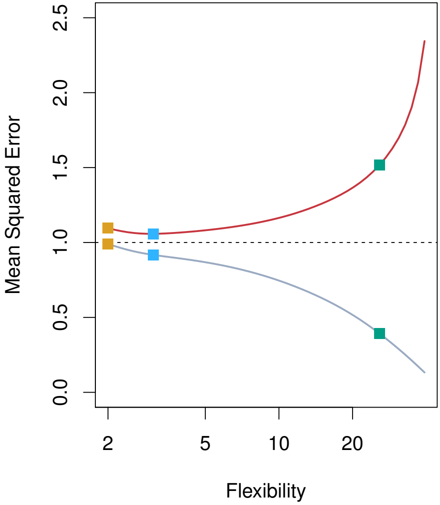


Black: true curve
Orange: linear regression
Blue/green: Smoothing splines

Grey: Training MSE
Red: Test MSE
Dashed: Minimum test MSE

Example: splines

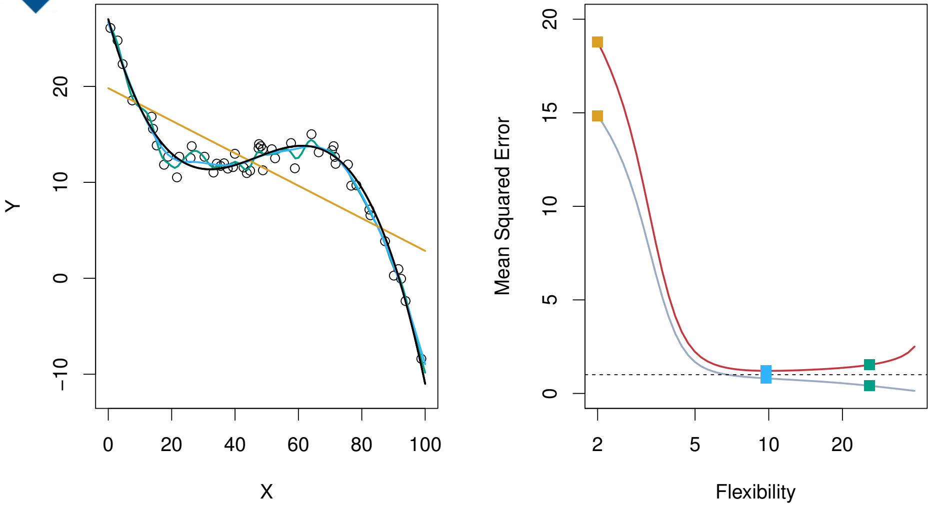




Black: true curve
Orange: linear regression
Blue/green: Smoothing splines

Grey: Training MSE
Red: Test MSE
Dashed: Minimum test MSE

Example: splines



Black: true curve
Orange: linear regression
Blue/green: Smoothing splines

Grey: Training MSE
Red: Test MSE
Dashed: Minimum test MSE

Bias-variance tradeoff

There are two competing forces that govern the choice of learning method: **bias** and **variance**.

Bias

is the error that is introduced by modeling a complicated problem by a simpler problem.

- For example, linear regression assumes a linear relationship when few real relationships are exactly linear.
- In general, the more flexible a method is, the less bias it will have.

Bias-variance tradeoff

There are two competing forces that govern the choice of learning method: **bias** and **variance**.

Variance

refers to how much your estimate would change if you had different training data.

- In general, the more flexible a method is, the more variance it has.
- The size of the training data has an impact on the variance

The bias-variance tradeoff

MSE decomposition

If $Y = f(x) + \varepsilon$ and $f(x) = \mathrm{E}[Y \mid X = x]$, then the expected **test** MSE for a new Y at x_0 will be equal to

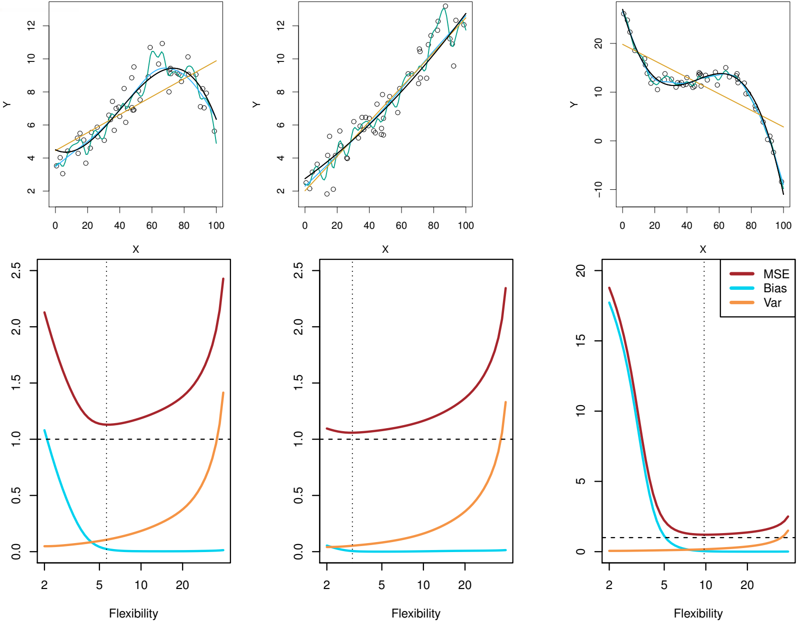
$$\mathrm{E}[(Y - \hat{f}(x_0))^2] = [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\hat{f}(x_0)) + \mathrm{Var}(\varepsilon)$$

$\rightarrow$ see proof of MSE decomposition

Test MSE = Bias^2 + Variance + Irreducible variance

- The expectation averages over the variability of Y as well as the variability in the training data.
- As the flexibility of $\hat{f}$ increases, its variance increases and its bias decreases.
- Choosing the flexibility based on average test MSE amounts to a **bias-variance trade-off**.

Bias-variance trade-off



Optimal prediction

MSE decomposition

If $Y = f(x) + \varepsilon$ and $f(x) = \mathrm{E}[Y \mid X = x]$, then the expected **test** MSE for a new Y at x_0 will be equal to

$$\mathrm{E}[(Y - \hat{f}(x_0))^2] = [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\hat{f}(x_0)) + \mathrm{Var}(\varepsilon)$$

The optimal MSE is obtained when

$$\hat{f} = f = \mathrm{E}[Y \mid X = x].$$

Then bias=variance=0 and

$$\text{MSE} = \text{irreducible variance}$$

This is called the "oracle" predictor because it is not achievable in practice.

Outline

The classification problem

Here the response variable Y is qualitative.

- e.g., email is one of $\mathcal{C} = (\text{spam}, \text{ham})$
- e.g., voters are one of $\mathcal{C} = (\text{Liberal}, \text{Labor}, \text{Green}, \text{National}, \text{Other})$

Our goals are:

1. Build a classifier $C(x)$ that assigns a class label from $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$ to a future unlabeled observation x.
2. Such a classifier will divide the input space into regions $\mathcal{R}_k$ called decision regions, one for each class, such that all points in $\mathcal{R}_k$ are assigned to class $\mathcal{C}_k$
3. Assess the uncertainty in each classification (i.e., the probability of misclassification).
4. Understand the roles of the different predictors among $X = (X_1, X_2, \ldots, X_p)$.

The classification problem

Recall that we want to minimize the expected prediction error

$$E_{(Y,X)}[L(Y, C(X))]$$

where $L(y, \hat{y})$ is a non-negative real-valued *loss function*.

In classification, the output Y is a **categorical variable**, and our loss function can be represented by a $K \times K$ matrix $\boldsymbol{L}$, where $K = \text{card}(\mathcal{C})$. $L(k, l)$ is the price paid for classifying C_k as C_l.

Optimal classifier

We want to minimize the expected prediction error

$$E_{(Y,X)}[L(Y, C(X))] = E_X \left[\sum_{k=1}^{K} L(C_k, C(X)) P(C_k|X) \right]$$

It suffices to minimize the error pointwise:

$$C^*(x) = \text{argmin}_{c \in \mathcal{C}} \sum_{k=1}^{K} L(C_k, c) P(C_k|X = x)$$

Optimal classifier

If we use the zero-one loss, i.e. $L(y, \hat{y}) = I(y \neq \hat{y})$, then

$$\sum_{k=1}^{K} L(C_k, c) P(C_k|X = x) = \sum_{k=1}^{K} P(C_k|X = x) - P(c|X = x)$$
$$= 1 - P(c|X = x)$$
$$C^*(x) = \text{argmin}_{c \in \mathcal{C}} \sum_{k=1}^{K} L(C_k, c) P(C_k|X = x)$$
$$= \text{argmin}_{c \in \mathcal{C}}\, 1 - P(c|X = x)$$
$$= \text{argmax}_{c \in \mathcal{C}}\, P(c|X = x)$$

or

$$C^*(x) = C_k \text{ if } P(C_k|X = x) = \max_{c \in \mathcal{C}} P(c|X = x)$$

Optimal classifier

Let $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$, and let

$$p_k(x) = \mathrm{P}(Y = C_k \mid X = x), \qquad k = 1, 2, \ldots, K.$$

These are the conditional class probabilities at x.

Then the Bayes classifier at x is

$$C(x) = C_j \quad \text{if } p_j(x) = \max\{p_1(x), p_2(x), \ldots, p_K(x)\}$$

- This gives the minimum average test error rate.
- It is an "oracle predictor" because we do not usually know $p_k(x)$.

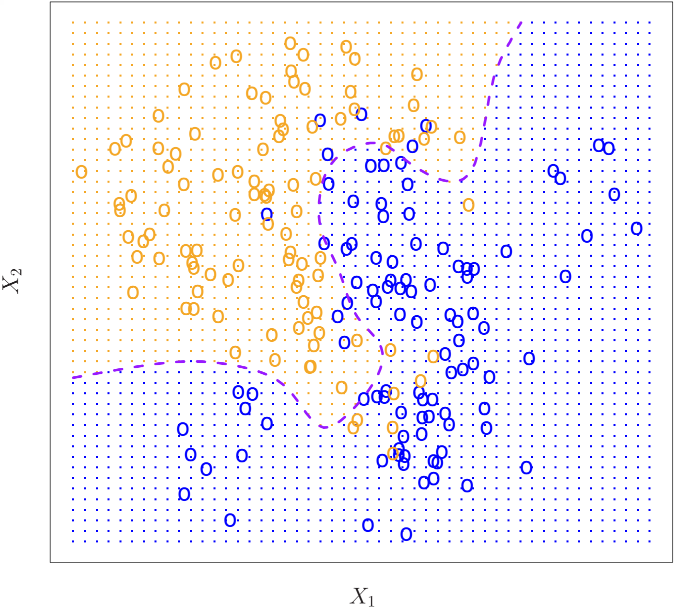
Bayes error rate

Bayes error rate

$$1 - \mathrm{E}\left(\max_j \mathrm{P}(Y = C_j | X)\right)$$

- The "Bayes error rate" is the lowest possible error rate that could be achieved if we knew exactly the "true" probability distribution of the data.
- It is analagous to the "irreducible error" in regression.
- On test data, no classifier can get lower error rates than the Bayes error rate.
- In reality, the Bayes error rate is not known exactly.

Bayes optimal classifier



The classification problem

Compute $\hat{C}$ from some **training data**, $Tr = \{x_i, y_i\}_1^n$.
In place of MSE, we now use the error rate (fraction of misclassifications).

Training Error Rate

$$\text{Error rate}_{Tr} = \frac{1}{n} \sum_{i=1}^{n} I(y_i \neq \hat{C}(x_i))$$

Measure real accuracy using **test data** $Te = \{x_j, y_j\}_1^m$

Test Error Rate

$$\text{Error rate}_{Te} = \frac{1}{m} \sum_{j=1}^{m} I(y_j \neq \hat{C}(x_j))$$

k-Nearest Neighbours

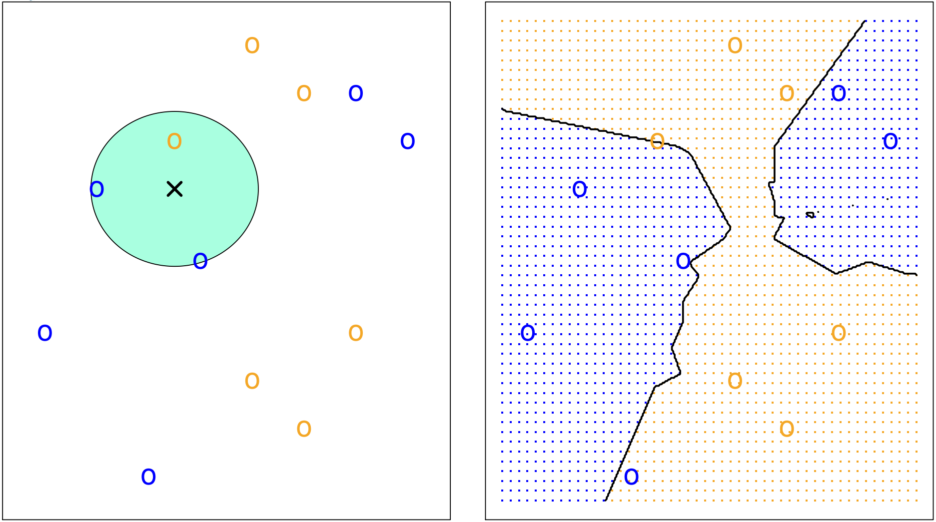
One of the simplest classifiers. Given a test observation x_0:

- Find the K nearest points to x_0 in the training data: $\mathcal{N}_0$.
- Estimate conditional probabilities

$$\mathrm{P}(Y = C_j \mid X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = C_j).$$

- Classify x_0 to class with largest probability.

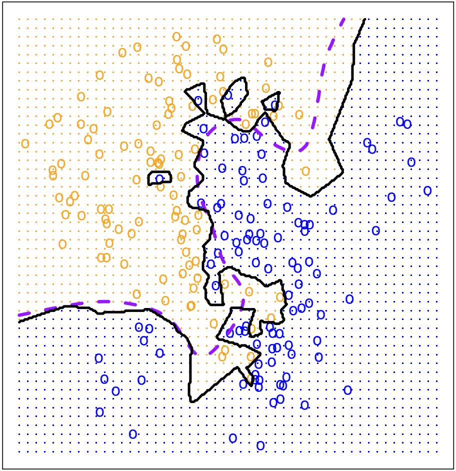
kNN Classifier



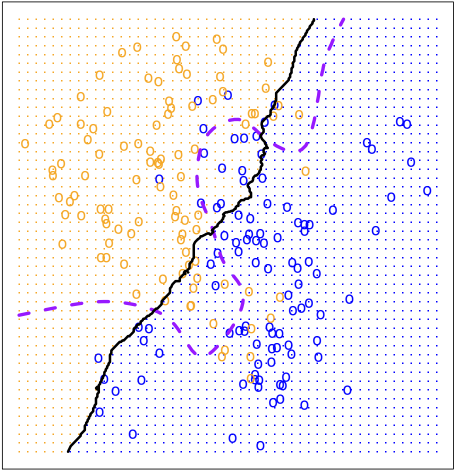
$K = 3$.

kNN Classifier

KNN: K=1

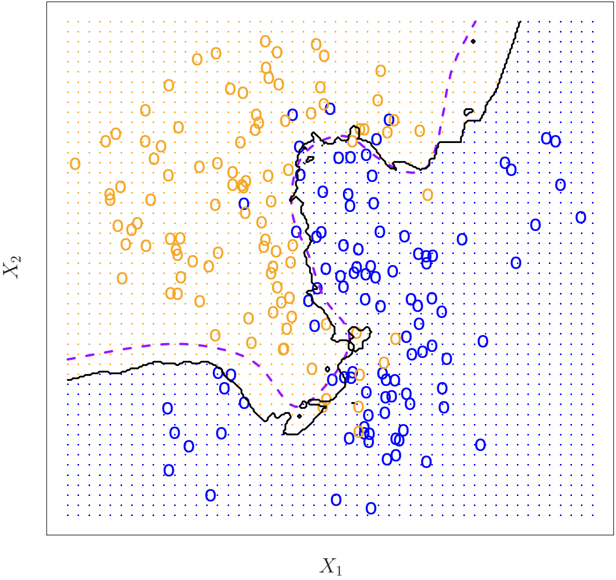


KNN: K=100

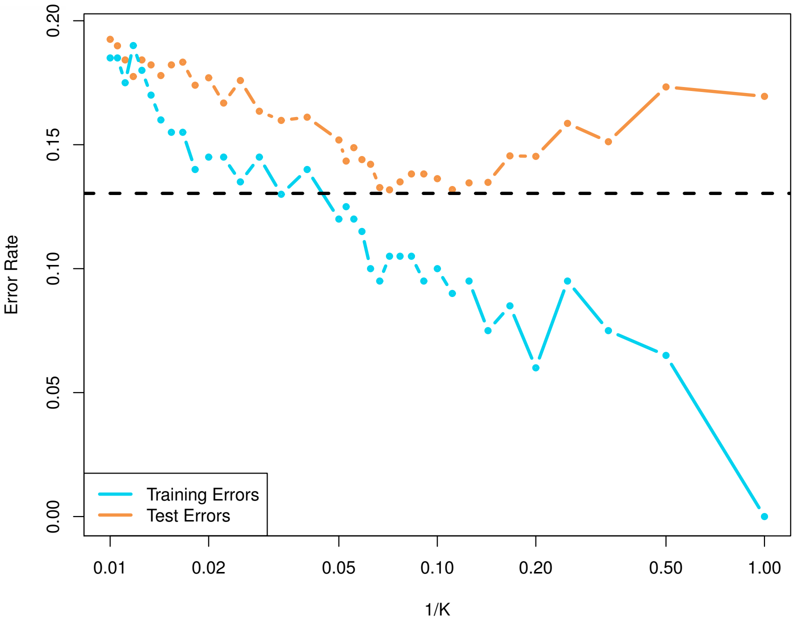


kNN Classifier

KNN: K=10



kNN Classifier



A fundamental picture