Unit Schedule: Modules

Module	Week	Content	Ross
1.	1	introduction to modelling	1,2
2.	2	probability refresher	3
	3	random vars & expected values	4
	4	special distributions	5
3.	5	statistical inference	6&7
	6	confidence intervals	7
	7	hypothesis testing	8
4.	8	**dependence & linear regression**	9
	9	**classification, clustering & mixtures**	
5.	10	random numbers & simulation	15(bits)
	11	basic machine learning	
6.	12	modelling, validation & review	

Revision at *https://flux.qa/43FMK4*

FIT5197 Statistical Data Modelling

Module 4

Classification, Clustering & Mixtures

2020 Lecture 9

Monash University

Classification

(ePub sections 4.3; Ross 9.11)

Outline

The Iris Data

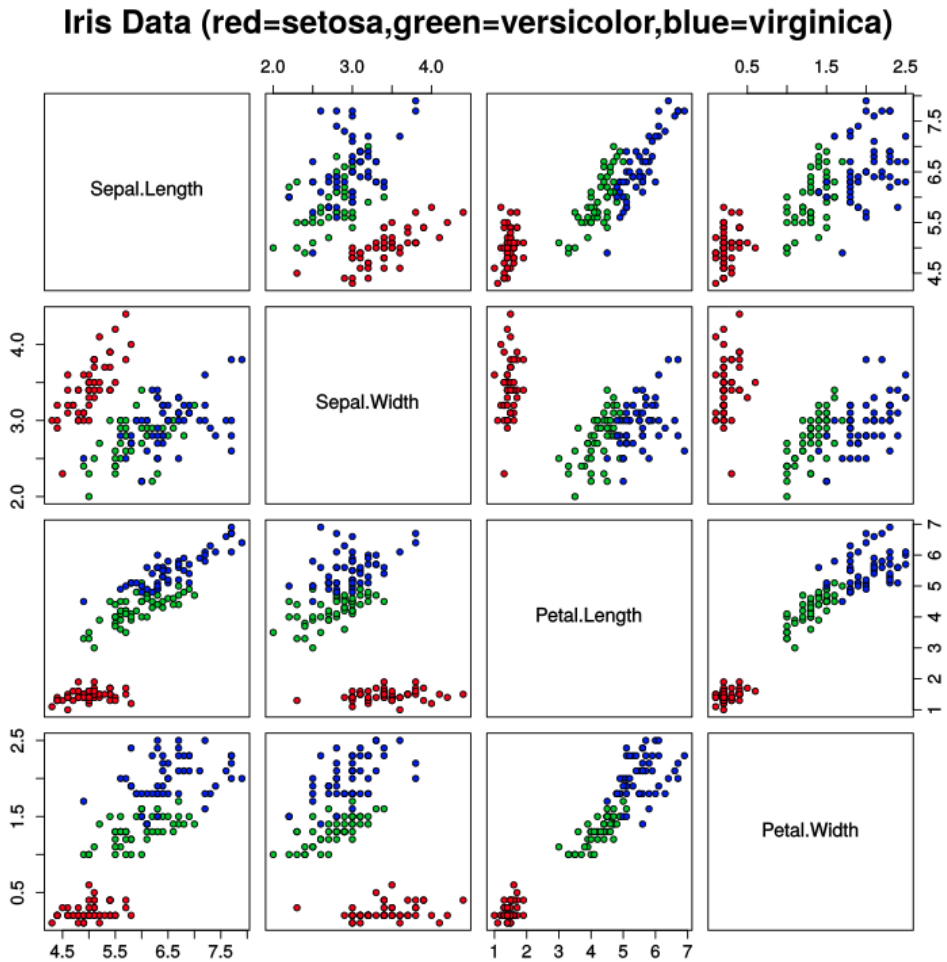
In 1936 Ronald Fisher wished to use statistical methods to determine Iris species based on measurements of petal and sepal length and width.

Iris Setosa

Iris Versicolor

Iris Virginica

The Iris Data, Classify?

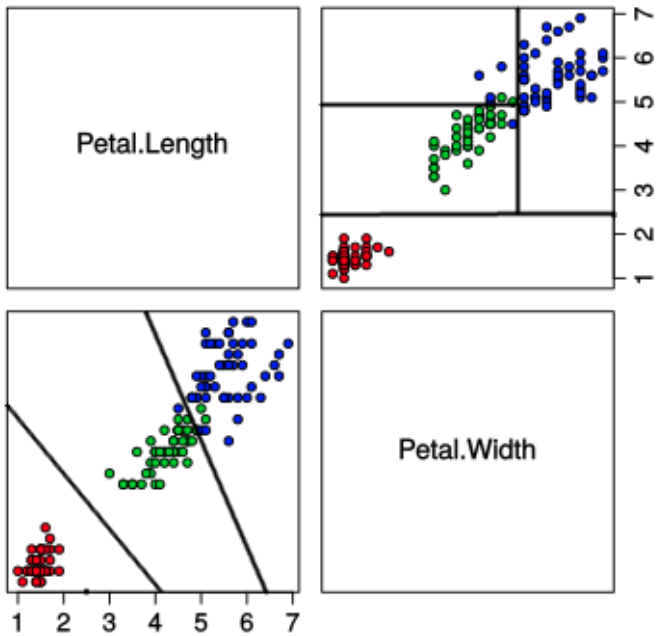


by Nicoguaro [CC BY 4.0], from Wikimedia Commons

if we had any two real valued features,

how would we predict the species (setosa, versicolor, virginica) ?

The Iris Data



from previous

Top right: axis parallel lines, e.g., decision tree

```
if Petal.Length < 2.45
then class = setosa
else if Petal.Width > 1.6
then class = virginica
else Petal.Length < 4.97
then class = versicolor
etc.
```

Bottom left: arbitrary lines, e.g., linear classifier

```
if (Petal.Width + 0.362*Petal.Length) < 1.7
then class = setosa
else (Petal.Width + Petal.Length) > 6.4
then class = virginica
etc.
```

But ...

Sometimes we don't just want a class from a classifier!

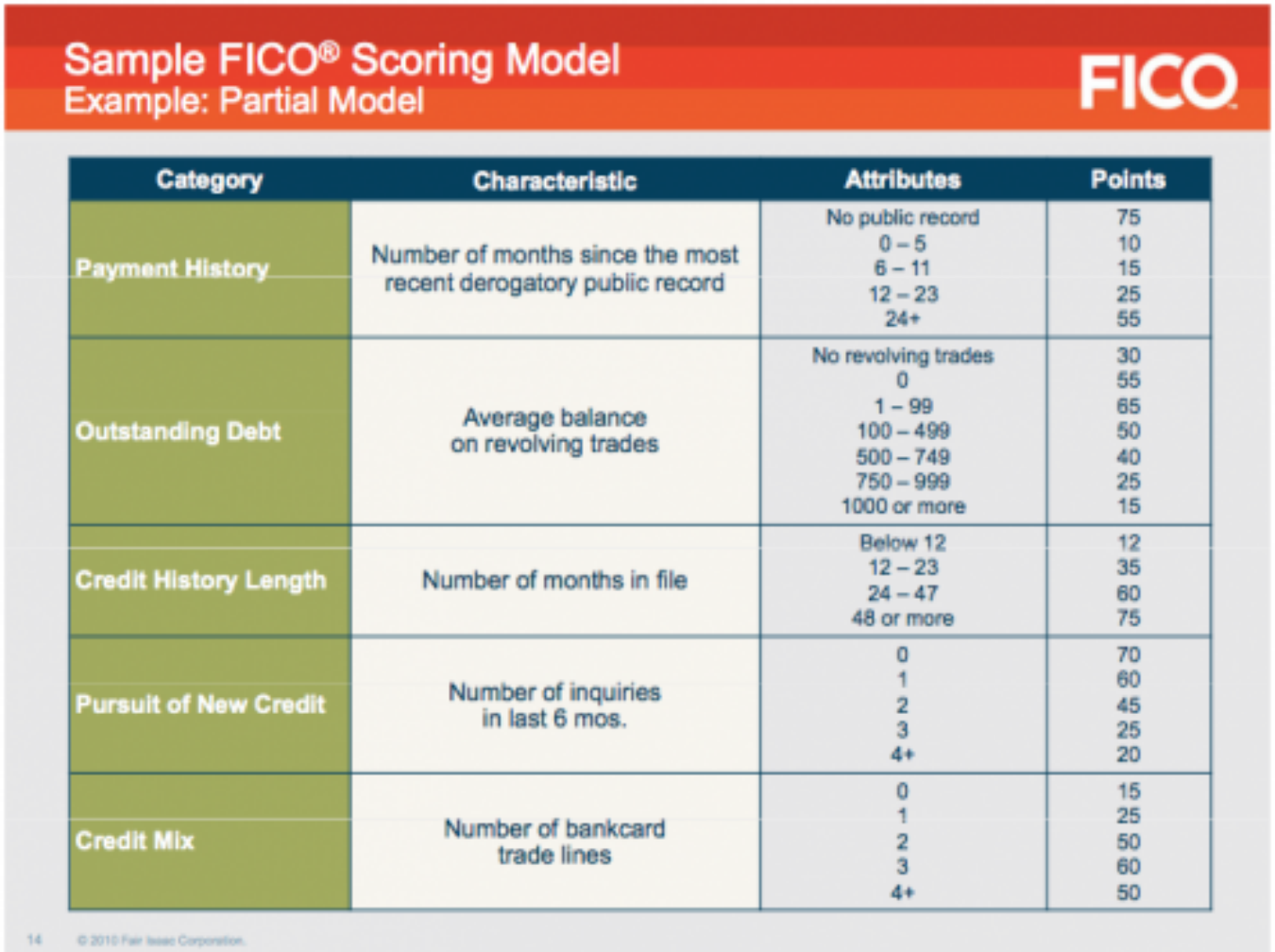
consider Cancer testing:

- Medical screening: Needs to make sure high probability of finding positive cases, less important to avoid false negatives

consider FICO scoring (for car/auto loans):

- tunable: bank manager wants to tune things to accept/reject more or less

Computing FICO Scores



Sample FICO® Scoring Model
Example: Partial Model

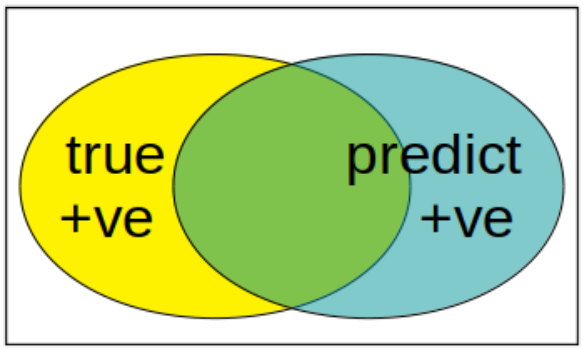
FICO

Category	Characteristic	Attributes	Points
Payment History	Number of months since the most recent derogatory public record	No public record 0 – 5 6 – 11 12 – 23 24+	75 10 15 25 55
Outstanding Debt	Average balance on revolving trades	No revolving trades 0 1 – 99 100 – 499 500 – 749 750 – 999 1000 or more	30 55 65 50 40 25 15
Credit History Length	Number of months in file	Below 12 12 – 23 24 – 47 48 or more	12 35 60 75
Pursuit of New Credit	Number of inquiries in last 6 mos.	0 1 2 3 4+	70 60 45 25 20
Credit Mix	Number of bankcard trade lines	0 1 2 3 4+	15 25 50 60 50

14 © 2010 Fair Isaac Corporation.

its a linear model, but we want to adjust the acceptance score

Evaluating Binary Predictions



$$\text{Precision} = \frac{\blacksquare}{\blacksquare\blacksquare} \qquad \text{Accuracy} = \frac{\blacksquare\square}{\blacksquare\blacksquare\blacksquare\square}$$

$$\text{Sensitivity} = \text{Recall} = \frac{\blacksquare}{\blacksquare\blacksquare} \qquad \text{Specificity} = \frac{\square}{\blacksquare\square}$$

$$\text{Prevalence} = \text{Base rate} = \frac{\blacksquare\blacksquare}{\blacksquare\blacksquare\blacksquare\square}$$

we can be more or less sensitive about about different error measures: recall, accuracy, sensitivity, etc.

see *precision and recall* on Wikipedia

Hard and Soft Classifiers

- A Classifier attempts to predict the value of a categorical variable Y, based on predictors $X_1, \ldots, X_p$.
- A Hard classifier gives a specific predicted value of Y - it predicts the class of each individual.
- A Soft classifier gives a *score* for each class based on the predictors
  - A common example is the probability that the individual is in the class given the values of the predictors.

Hard and Soft Classifiers

- A Classifier attempts to predict the value of a categorical variable Y, based on predictors $X_1, \ldots, X_p$.
- A Hard classifier gives a specific predicted value of Y - it predicts the class of each individual.
- A Soft classifier gives a *score* for each class based on the predictors
  - A common example is the probability that the individual is in the class given the values of the predictors.
- Fisher used a hard classifier for classifying irises.
- FICO uses a soft classifier for predicting defaults.
- We will look at soft classifiers.

Outline

Classification – Setup

- We consider soft classifiers and interpret them probabilistically
- Imagine we have a categorical outcome variable Y
- We also have p predictor variables $X_1, \ldots, X_p$
  $\Longrightarrow$ often called features in classification literature
- A classifier is function $\hat{p}(y, x_1, x_2, , \ldots, x_P)$ which estimates the probability of being in class y given the values for the predictors $x_1, \ldots, x_p$.

Classification – Setup

- We consider soft classifiers and interpret them probabilistically
- Imagine we have a categorical outcome variable Y
- We also have p predictor variables $X_1, \ldots, X_p$
  $\Longrightarrow$ often called features in classification literature
- A classifier is function $\hat{p}(y, x_1, x_2, , \ldots, x_P)$ which estimates the probability of being in class y given the values for the predictors $x_1, \ldots, x_p$.
- Ideally we want:

$$\hat{p}(y, x_1, x_2, , \ldots, x_P) = p(Y{=}y \mid X_1{=}x_1, X_2{=}x_2, \ldots, X_p{=}x_p)$$

- This is the conditional probability of $Y = y$ given $X_1, \ldots, X_p$.

Classification – Setup

- We consider soft classifiers and interpret them probabilistically
- Imagine we have a categorical outcome variable Y
- We also have p predictor variables $X_1, \ldots, X_p$
  $\Longrightarrow$ often called features in classification literature
- A classifier is function $\hat{p}(y, x_1, x_2, , \ldots, x_P)$ which estimates the probability of being in class y given the values for the predictors $x_1, \ldots, x_p$.
- Ideally we want:

$$\hat{p}(y, x_1, x_2, , \ldots, x_P) = p(Y{=}y \mid X_1{=}x_1, X_2{=}x_2, \ldots, X_p{=}x_p)$$

- This is the conditional probability of $Y = y$ given $X_1, \ldots, X_p$.
- The classifier which satisfies this equation exactly is called the Bayes classifier.

Classifiers

- Let us now specialise our problem
- Assume that all the predictors are also categorical
- The formula for conditional probability is:

$$p(Y{=}y \mid X_1{=}x_1, \ldots, X_p{=}x_p) = \frac{p(Y{=}y, X_1{=}x_1, \ldots, X_p{=}x_p)}{p(X_1{=}x_1, \ldots, X_p{=}x_p)}$$

where

  - the numerator is the joint probability of $(Y{=}y, X_1{=}x_1, \ldots, X_p{=}x_p)$;
  - the denominator is the marginal probability of $(X_1{=}x_1, \ldots, X_p{=}x_p)$.

Bayes Classifier

- So if we have the joint probability we can build the Bayes classifier
- Consider the following example:

	No Heart Disease (H=0)	Heart Disease (H=1)
No Mutation (M=0)	0.35	0.30
Mutation (M=1)	0.10	0.25

Table: Population joint probabilities of heart disease/LDLR mutation.

- Then we have

$$p(H{=}1 \mid M{=}0) = \frac{p(H{=}1, M{=}0)}{p(M{=}0)} = 0.4615$$

$$p(H{=}1 \mid M{=}1) = \frac{p(H{=}1, M{=}1)}{p(M{=}1)} = 0.7143$$

Bayes Classifier, cont.

- In our example we were told the population joint probabilities
- But in practice we don't know these – we just have data
- We can try to estimate them from the data
- For our example:
  - our target is heart disease, $H \in \{0, 1\}$,
  - predictor is LDLR mutation, $M \in \{0, 1\}$

Classifiers, Example Learning

- Imagine we have n realisations of the random variables H and M: $\mathbf{m} = (m_1, \ldots, m_n)$ and $\mathbf{h} = (h_1, \ldots, h_n)$
- We could estimate joint probability by proportions

$$\hat{P}(H{=}h, M{=}m) = \frac{1}{n} \sum_{i=1}^{n} I(h_i = h \text{ and } m_i = m)$$

- Note $I(\cdot)$ is the Indicator function which is 1 if its argument is true and 0 otherwise.
- Weak law of large numbers guarantees this will converge on population proportions for large enough n

Classifiers, Example Learning

- Example, imagine we had
  - $\mathbf{m} = (1, 1, 0, 1, 1, 1, 0, 0)$ and
  - $\mathbf{h} = (1, 0, 1, 1, 0, 0, 1, 0)$
- Then estimated joint probabilities are

	No Heart Disease (H=0)	Heart Disease (H=1)
No Mutation (M=0)	$1/8$	$2/8$
Mutation (M=1)	$3/8$	$2/8$

Table: Estimated joint probabilities of heart disease/LDLR mutation

Classifiers, Example Learning

- Example, imagine we had
  - $\mathbf{m} = (1, 1, 0, 1, 1, 1, 0, 0)$ and
  - $\mathbf{h} = (1, 0, 1, 1, 0, 0, 1, 0)$
- Then estimated joint probabilities are

	No Heart Disease (H=0)	Heart Disease (H=1)
No Mutation (M=0)	1/8	2/8
Mutation (M=1)	3/8	2/8

Table: Estimated joint probabilities of heart disease/LDLR mutation

- Now we can estimate $p(H{=}h \mid M{=}m)$ using

$$\frac{\hat{P}(H{=}h, M{=}m)}{\hat{P}(H{=}0, M{=}m) + \hat{P}(H{=}1, M{=}m)}$$

- For large n the proportions will be close to population probabilities

Classifiers, Iris Data

- Let us do a similar approach with the Iris dataset.
- To make it manageable, transform the features to Booleans, in R

```
> data(iris)
> iris$SLC = iris$Sepal.Length < 6
> iris$SWC = iris$Sepal.Width < 3
> iris$PLC = iris$Petal.Length < 5
> iris$PWC = iris$Petal.Width < 1.6
```

- called discretising the continuous attributes,
  - usually use 3-8 cut-points, not one!
  - clever algorithms exist to do it well
- now we can get frequency counts of pairwise entries for Species versus the new Boolean variables SLC, SWC, PLC, PWC to do an estimate

Classifiers, Scaling Learning

- Simple enough – but there is a problem
- For our simple problem H and M were binary
  $\Longrightarrow$ only need to estimate $2 \times 2 = 4$ joint probabilities
- What if we had two binary genetic mutations, M_1 and M_2?
- Now have $2 \times 2 \times 2 = 8$ probabilities to estimate
- For p predictors, there are 2^{p+1} probabilities to estimate
  $\Longrightarrow$ exponential growth in p
- This rapidly outstrips our sample size n no matter how big n is

Classifiers, Scaling Learning

- We need to constrain the problem
- Two simple approaches popular in literature

Classifiers, Scaling Learning

- We need to constrain the problem
- Two simple approaches popular in literature

- Naïve Bayes classifiers
  - Make independence assumptions about conditional probabilities
  - Easily handle categorical predictors
  - Easily handles multi-class targets
  - Popular in text mining and classification

Classifiers, Scaling Learning

- We need to constrain the problem
- Two simple approaches popular in literature

- Naïve Bayes classifiers
  - Make independence assumptions about conditional probabilities
  - Easily handle categorical predictors
  - Easily handles multi-class targets
  - Popular in text mining and classification

- Logistic regression
  - Adaptation of the linear model, widely used
  - Directly estimates conditional probabilities
  - Handles categorical and continuous predictors
  - More difficult to handle multi-class targets

Naïve Bayes Classifiers

- The Naïve Bayes solves the problem of too many probabilities to estimate by making a very strong assumption
- Let $X_1, \ldots, X_p$ be p categorical predictors (features) $\Longrightarrow$ do not have to be binary
- Use the shorthand notation $p(Y{=}y, X{=}x) \equiv p(y, x)$
- Naïve Bayes assumes predictors are conditionally independent, given the value of the target

$$p(x_1, \ldots, x_p \,|\, y) = \prod_{j=1}^{p} p(x_j \,|\, y)$$

Naïve Bayes Classifiers

- This assumption lets us write joint probability as

$$\begin{aligned} p(y, x_1, \ldots, x_p) &= p(y)p(x_1, \ldots, x_p \mid y) \\ &= p(y) \prod_{j=1}^{p} p(x_j \mid y) \end{aligned}$$

Naïve Bayes Classifiers

- This assumption lets us write joint probability as

$$\begin{aligned} p(y, x_1, \ldots, x_p) &= p(y)p(x_1, \ldots, x_p \mid y) \\ &= p(y) \prod_{j=1}^{p} p(x_j \mid y) \end{aligned}$$

- Using this joint probability we can write

$$p(y \mid x_1, \ldots, x_p) = \frac{p(y) \prod_{j=1}^{p} p(x_j \mid y)}{p(x_1, \ldots, x_p)}$$

where we note $p(x_1, \ldots, x_p)$ does not depend on y
$\Longrightarrow$ constant for given values of predictors

Bayes Classifiers

Compare the (full) Bayes Classifier:

$$p(y \mid x_1, \ldots, x_p) = \frac{p(y)p(x_1, \ldots, x_p \mid y)}{p(x_1, \ldots, x_p)}$$

which gives the posterior probability by Bayes Theorem

with the Naïve Bayes Classifier:

$$p(y \mid x_1, \ldots, x_p) = \frac{p(y) \prod_{j=1}^{p} p(x_j \mid y)}{p(x_1, \ldots, x_p)}$$

which also adds the independence assumption

Naïve Bayes for iris Data

Compare the Bayes Classifier:

$$p(set. \mid SLC, SWC, PLC, PWC) = \frac{p(set.)p(SLC, SWC, PLC, PWC \mid set.)}{p(SLC, SWC, PLC, PWC)}$$

with the Naïve Bayes Classifier:

$$p(set. \mid SLC, SWC, PLC, PWC) = \frac{p(set.)p(SLC \mid set.)p(SWC \mid set.)p(PLC \mid set.)p(PWC \mid set.)}{p(SLC, SWC, PLC, PWC)}$$

Naïve Bayes for iris Data

Compare the Bayes Classifier:

$$p(set. \mid SLC, SWC, PLC, PWC) = \frac{p(set.)p(SLC, SWC, PLC, PWC \mid set.)}{p(SLC, SWC, PLC, PWC)}$$

with the Naïve Bayes Classifier:

$$p(set. \mid SLC, SWC, PLC, PWC) = \frac{p(set.)p(SLC \mid set.)p(SWC \mid set.)p(PLC \mid set.)p(PWC \mid set.)}{p(SLC, SWC, PLC, PWC)}$$

- we need probability models for SLC, SWC, PLC, PWC for each of the 3 species
- so we need 12 univariate models instead of 3 quad-variate models

Naïve Bayes Classifiers, cont.

- Reduced number of probabilities we need to estimate
  - Let predictor j be categorical with K_j categories
  - Let target Y be categorical with K_y categories
- Conditional independence means we need only estimate

$$p(X_j{=}x \mid Y{=}y),\ x \in \{1, \ldots, K_j\},\ y \in \{1, \ldots, K_y\}$$

for $j = 1, \ldots, p$, and

$$p(Y{=}y),\ y \in \{1, \ldots, K_y\}$$

Naïve Bayes Classifiers, cont.

- Without assumptions we estimate $K_y \prod_{j=1}^{p} K_j$ probabilities
- With assumptions we only estimate

$$K_y + (K_y \times K_1) + (K_y \times K_2) + \ldots + (K_y \times K_p)$$

probabilities, and usually

$$K_y \left(1 + \sum_{j=1}^{p} K_j\right) \ll K_y \prod_{j=1}^{p} K_j$$

- So Naïve Bayes assumptions reduce problem size to linear in p

Naïve Bayes Classifiers, cont.

- Given n samples of target-predictor pairs
  - $\mathbf{y} = (y_1, \ldots, y_n)$
  - $\mathbf{x}_j = (x_{1,j}, \ldots, x_{n,j}),\ j = 1, \ldots, p$

  we can learn a Naïve Bayes model from the proportions:

$$\hat{P}(X_j{=}x \mid Y{=}y) = \frac{\sum_{i=1}^{n} I(x_{i,j}{=}x \text{ and } y_i{=}y)}{\sum_{i=1}^{n} I(y_i{=}y)}$$

  and

$$\hat{P}(Y{=}y) = \frac{1}{n} \sum_{i=1}^{n} I(y_i{=}y)$$

Naïve Bayes Classifiers, cont.

- Given n samples of target-predictor pairs
  - $\mathbf{y} = (y_1, \ldots, y_n)$
  - $\mathbf{x}_j = (x_{1,j}, \ldots, x_{n,j}),\ j = 1, \ldots, p$

  we can learn a Naïve Bayes model from the proportions:

$$\hat{P}(X_j{=}x \mid Y{=}y) = \frac{\sum_{i=1}^{n} I(x_{i,j}{=}x \text{ and } y_i{=}y)}{\sum_{i=1}^{n} I(y_i{=}y)}$$

  and

$$\hat{P}(Y{=}y) = \frac{1}{n} \sum_{i=1}^{n} I(y_i{=}y)$$

- That is, we estimate probabilities from the observed frequencies.
- Learning a Naïve Bayes classifer is linear time complexity in the sample size n.

Naïve Bayes Classifiers, cont.

- So Naïve Bayes is efficient, but only by being quite restrictive
- No real reason to think that predictors should be conditionally independent in real life
- However, often works sufficiently well in practice

Naïve Bayes Classifiers, Other Features

- So far assumed categorical predictors
- How to handle continuous predictors?
- There are two approaches:

  1. We assume parametric distributions for each predictor, i.e.,

     $$p(x_j \mid y) \equiv p(x_j \mid \theta_{j,y})$$

     and estimate $\hat{\theta}_{j,y}$ from the data using maximum likelihood
  2. We discretize the predictor into categories, i.e., use histogram

- We do not examine this further in this subject

Classifiers, Iris Data

We tabulate the various occurrences (using R's `table()`):

Species	setosa	versicolor	virginica
SLC=Sepal.Length$<$6	50	26	7
$\overline{SLC}$=Sepal.Length$\geq$6	0	24	43

Species	setosa	versicolor	virginica
SWC=Sepal.Width$<$3	2	34	21
$\overline{SWC}$=Sepal.Width$\geq$3	48	16	29

- now we can compute estimates for p(Species=setosa), $p(SLC|\text{Species=setosa})$, $p(\overline{SWC}|\text{Species=versicolor})$, etc.

Naïve Bayes for Iris Data

To find the probability a flower's species is setosa, start with the formula

$$p(set. \mid SLC, SWC, PLC, PWC) = \frac{p(set.)p(SLC, SWC, PLC, PWC \mid set.)}{p(SLC, SWC, PLC, PWC)}$$

and assume conditional independence to get

$$p(set. \mid SLC, SWC, PLC, PWC) = \frac{p(set.)p(SLC \mid set.)p(SWC \mid set.)p(PLC \mid set.)p(PWC \mid set.)}{p(SLC, SWC, PLC, PWC)}$$

- we simplified the 4-dimensional joint tables to 4 single dimensional vectors
- all these terms are estimated as per the previous slide

Outline

Logistic Regression

- The Naïve Bayes approach made assumptions about the features to simplify estimation of the joint probabilities
- It then used the joint probabilities to find the conditional probability of the targets using Bayes rule.
- This is a round-about way of solving problem
- Logistic regression directly models the conditional probabilities
  - Extends the linear regression model to binary data

Logistic Regression

- Given predictors $x_{i,1}, \ldots, x_{i,p}$ multiple linear regression predicts the target as

$$\mathbb{E}[y_i] = \eta_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j}$$

  where η_i is shorthand for our linear predictor for individual i
- We find $\beta_0, \beta_1, \ldots, \beta_j$ by least-squares
- If our target is binary, we *could* fit a linear model using least-squares and approximate

$$p(Y_i = 1 \mid x_{i,1}, \ldots, x_{i,p}) \approx \eta_i$$

Logistic Regression

- Given predictors $x_{i,1}, \ldots, x_{i,p}$ multiple linear regression predicts the target as

$$\mathbb{E}[y_i] = \eta_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j}$$

  where η_i is shorthand for our linear predictor for individual i
- We find $\beta_0, \beta_1, \ldots, \beta_j$ by least-squares
- If our target is binary, we *could* fit a linear model using least-squares and approximate

$$p(Y_i = 1 \mid x_{i,1}, \ldots, x_{i,p}) \approx \eta_i$$

- Serious problem: our predicted value η_i could be less than zero, or greater than one, for certain values of the features!

Logistic Regression Trick

- One solution is to bound η_i to (0,1)
- There exist a lot of ways of bounding η_i
- In Logistic regression we use the logistic function

$$p(Y_i{=}1 \mid x_{i,1}, \ldots, x_{i,p}) = \frac{1}{1 + \exp(-\eta_i)}$$

- This function smoothly
  - tends to 0 as $\eta_i \equiv \beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} \to -\infty$;
  - tends to 1 as $\eta_i \equiv \beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} \to \infty$.

Logistic Function

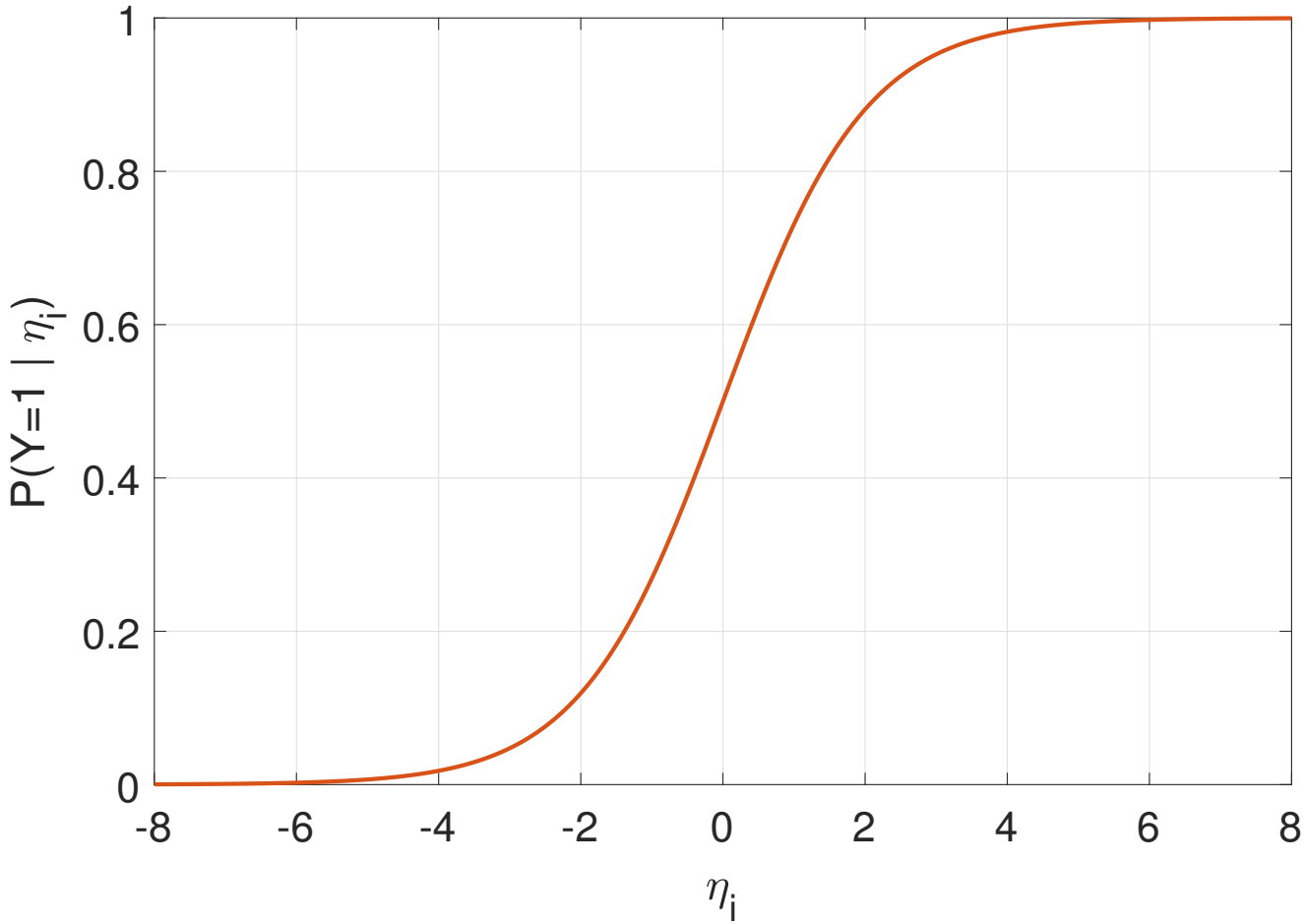


Figure: The logistic function. As $\eta_i \to -\infty$, then $P(Y{=}1) \to 0$, and as $\eta_i \to \infty$, then $P(Y{=}1) \to 1$.

Log-Odds

- We can interpret the linear model in terms of log-odds
- Given the $p(Y{=}1)$ and $p(Y{=}0)$, the odds for $Y{=}1$ are

$$p(Y{=}1)/p(Y{=}0)$$

- They reflect how many more times likely the event $Y{=}1$ is to occur than the event $Y{=}0$

Log-Odds

- We can interpret the linear model in terms of log-odds
- Given the $p(Y{=}1)$ and $p(Y{=}0)$, the odds for $Y{=}1$ are

$$p(Y{=}1)/p(Y{=}0)$$

- They reflect how many more times likely the event $Y{=}1$ is to occur than the event $Y{=}0$
- Example: probability of a heads from coin toss is 0.75; then
  - the odds for seeing a head are $0.75/0.25 = 3$;
  - the odds for seeing a tail are $0.25/0.75 = 1/3$.
- The log-odds make this symmetric:
  - the log-odds for seeing a head are $\log(0.75/0.25) = \log 3$;
  - the log-odds for seeing a tail are $\log(0.25/0.75) = -\log 3$.

Logistic Regression

- A logistic regression models the conditional log-odds as

$$\log\left(\frac{p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}{p(Y_i{=}0 \mid x_{i,1},\ldots,x_{i,p})}\right) = \beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} \equiv \eta_i$$

- So the log-odds of a success, given the values of the predictors, is equal to the linear predictor η_i
  - the intercept β_0 is the log-odds when all the predictors are zero, i.e., $x_{i,1} = x_{i,2} = \ldots = x_{i,p} = 0$;
  - the coefficient β_j is the increase in log-odds per unit change of predictor x_j

- The odds for $Y{=}1$ are $\exp(\eta_i)$
  - when $\eta_j > 0$, $Y{=}1$ is more likely than $Y{=}0$;
  - when $\eta_j < 0$, $Y{=}0$ is more likely than $Y{=}1$

Logistic Regression

(optional)

- To see that setting log-odds equal to η_i leads to logistic regression, write:

$$\log\left(\frac{p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}{1-p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}\right) = \eta_i$$

Logistic Regression

(optional)

- To see that setting log-odds equal to η_i leads to logistic regression, write:

$$\log\left(\frac{p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}{1-p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}\right) = \eta_i$$

- Now exponentiate both sides

$$\frac{p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}{1-p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})} = \exp(\eta_i)$$

Logistic Regression

(optional)

- To see that setting log-odds equal to η_i leads to logistic regression, write:

$$\log\left(\frac{p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}{1-p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}\right) = \eta_i$$

- Now exponentiate both sides

$$\frac{p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})}{1-p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p})} = \exp(\eta_i)$$

- Solving for $p(Y_i{=}1 \mid \cdots)$ yields

$$p(Y_i{=}1 \mid x_{i,1},\ldots,x_{i,p}) = \frac{\exp(\eta_i)}{1+\exp(\eta_i)} = \frac{1}{1+\exp(-\eta_j)}$$

which is the logistic function (noting $1/e^a = e^{-a}$).

Fitting Log.Reg. Models

- How to estimate the regression coefficients?
- Many packages use maximum likelihood
- Let $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)$ be our parameters (intercept, coeffs.)
- Assume our targets are independent RVs;
- Each Y_i is then distributed as per a Bernoulli distribution

$$Y_i \sim \mathrm{Be}\left(\theta_i(\boldsymbol{\beta})\right)$$

where

$$\theta_i(\boldsymbol{\beta}) = \frac{1}{1 + \exp\left(-\beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j}\right)}$$

is the probability of success for individual i, given the predictors $x_{i,1}, \ldots, x_{i,p}$ and the parameters $\beta_0, \beta_1, \ldots, \beta_p$.

Fitting Log.Reg. Models

(optional)

- If $\mathbf{y} = (y_1, \ldots, y_n)$ are binary targets, the likelihood for a logistic regression is then

$$\begin{aligned} p(\mathbf{y} \mid \beta) &= \prod_{i=1}^{n} p(y_i|\beta) \\ &= \prod_{i=1}^{n} \theta_i(\beta)^{y_i} \left(1 - \theta_i(\beta)\right)^{1-y_i} \end{aligned}$$

  from the pdf of the Bernoulli distribution.

- The negative log-likelihood is then

$$L(\mathbf{y} \mid \beta) = -\sum_{i=1}^{n} \left[y_i \log \theta_i(\beta) + (1 - y_i) \log \left(1 - \theta_i(\beta)\right)\right]$$

Fitting Log.Reg. Models

- The negative log-likelihood is then

$$L(\mathbf{y} \mid \beta) = -\sum_{i=1}^{n} [y_i \log \theta_i(\beta) + (1 - y_i) \log (1 - \theta_i(\beta))]$$

- The values $\hat{\beta}$ of β that minimise this quantity are the maximum likelihood estimates

- No closed form solution exists, must be found numerically
  $\Longrightarrow$ But luckily always only a single, global minimum

- Time complexity roughly cubic in number of predictors p
  $\Longrightarrow$ potentially slower than Naïve Bayes

Goodness-of-Fit

- The minimised negative log-likelihood

$$L(\mathbf{y} \mid \hat{\boldsymbol{\beta}})$$

  is a measure of goodness-of-fit of a model.
- The difference in minimised negative log-likelihoods

$$L(\mathbf{y} \mid \hat{\beta}_0) - L(\mathbf{y} \mid \hat{\boldsymbol{\beta}})$$

  is a measure of fit
  - bigger differences $\Rightarrow$ better fit
- Relative to model with an intercept only (no predictors); equivalent to assuming $P(Y{=}1)$ is same for all individuals
- Sometimes maximised log-likelihood is reported instead; depends on package

Predicting with Log.Reg.

- Once we have found estimates $\hat{\beta}$ it is easy to predict with a logistic regression
- For some new values of features $x'_1, \ldots, x'_p$, we calculate

$$\hat{\eta} = \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j x'_j$$

- Then we can estimate the probability that $Y' = 1$ for these features:

$$p(Y'{=}1 \mid x'_1, \ldots, x'_p) = \frac{1}{1 + \exp(-\hat{\eta})}$$

- If we need to guess at most likely class, choose value of Y' that maximises this probability

Logistic Regressions are Linear

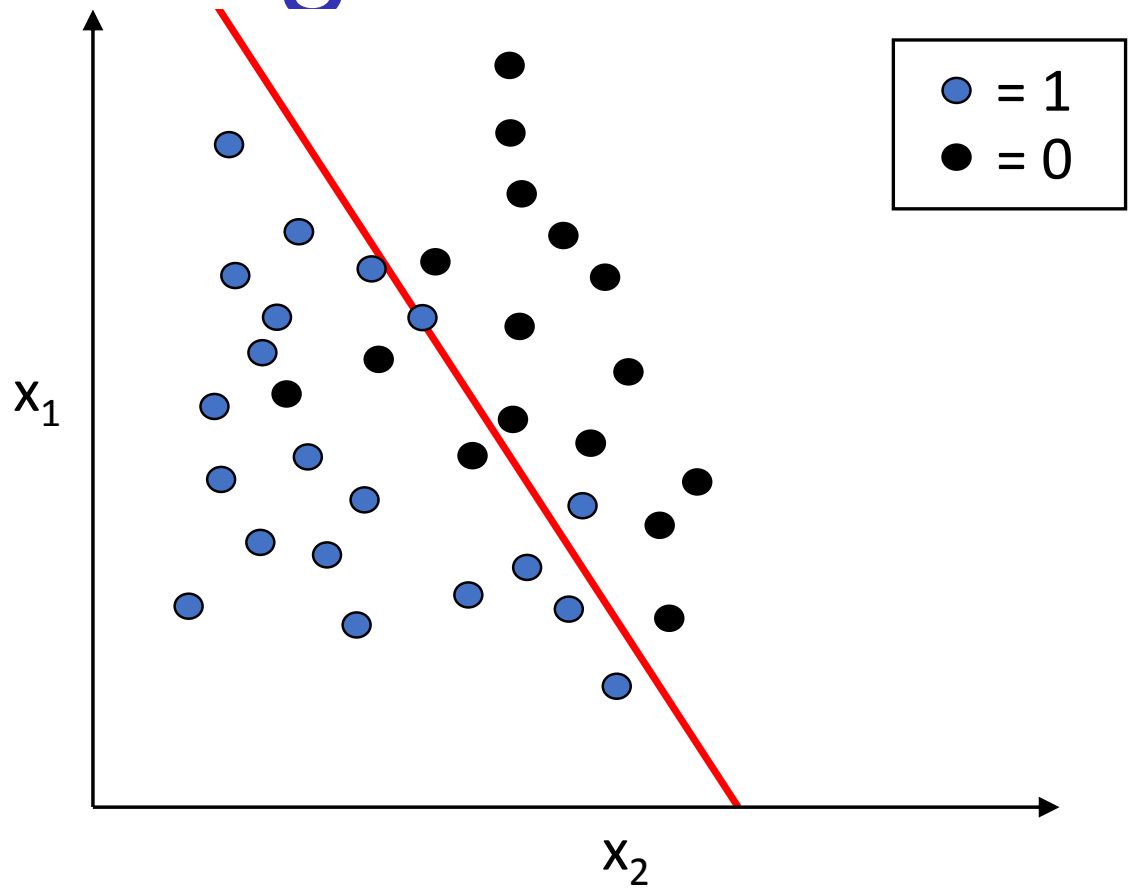


Figure: A logistic regression seperates successes from failures by using a linear seperation surface. The line is defined by the values of the two features x_1 and x_2 that satisfy $p(Y{=}1 \mid x_1, x_2) = 1/2$. For models with p features, this becomes a p-dimensional plane.

Extensions

- A strength of logistic regression is that it builds on the tools used in linear regression
- Handle categorical predictors same as linear regression
  $\Longrightarrow$ form new indicator variables for each category
- They can also handle non-linearities the same way as linear regressions, e.g.,
  - logarithmic transformations of predictors;
  - polynomial transformations of predictors.

Finding Log.Reg. Models

(optional)

- Same approaches as for linear models
  $\Longrightarrow$ try to avoid under/over-fitting
- Hypothesis testing $H_0 : \beta_j = 0$ vs $H_A : \beta_j \neq 0$
  - Smaller p-value $\Rightarrow$ more likely predictor j is important
- Model selection
  - Use penalized likelihood:

    $$L(\mathbf{y} \mid \hat{\boldsymbol{\beta}}) + k\alpha_n$$

    where:
    - k is number of predictors in model;
    - $\alpha_n = 1$ for Akaike information criterion (AIC);
    - $\alpha_n = 3/2$ for Kullback information criterion (KIC);
    - $\alpha_n = (1/2) \log n$ for Bayesian information criterion (BIC).
  - Sometimes two times these quantities are used (e.g., in R)
- Forward/backwards selection of predictors

Clustering & Mixtures
(ePub sections 4.5)

Revision at *https://flux.qa/43FMK4*

Outline

Unsupervised Learning

- We have n items, each with q associated attributes, formed into a matrix

$$\mathbf{Y} = \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_n \end{pmatrix} = \begin{pmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,q} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,q} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n,1} & y_{n,2} & \cdots & y_{n,q} \end{pmatrix}$$

- Each $\mathbf{y}_i$ is a "data-point" in q-dimensional space
- Unlike supervised learning, we do not nominate any one of these as a "target"
- Instead we want to discover structure in the data

Unsupervised Learning, cont.

- What is unsupervised learning used for?
- Classifying or categorising objects (taxonomy)
  - For example, species of animals
- Filling in missing entries in the data matrix
  - Matrix completion problem
  - Recommender systems
  - Imputation (estimating missing data in predictor matrix before supervised learning)
- Image processing
  - Noise removal
  - Compression
  - Image analysis and recognition

Clustering

- Assumptions
  - Population consists of K sub-populations ($K > 1$)
  - We are given observations from the pooled population only
    - No sub-population information is available
- Aim
  - Sometimes descriptive: divide data into *clusters* which are similar
  - Sometimes inferential: infer data comes from K populations and estimate models for each of the sub-populations

Clustering

- Assumptions
  - Population consists of K sub-populations ($K > 1$)
  - We are given observations from the pooled population only
    - No sub-population information is available
- Aim
  - Sometimes descriptive: divide data into *clusters* which are similar
  - Sometimes inferential: infer data comes from K populations and estimate models for each of the sub-populations
- Sometimes called intrinsic classification or segmentation
  $\Longrightarrow$ Class labels are learned from the data

K-means Clustering

- Perhaps most commonly used clustering technique
- Attempts to find K clusters, so that points within each cluster are as similar as possible
- In particular partitions the data into K clusters $C_1, \ldots, C_K$, so as to minimise the sum of squared distances between each point and the mean of its cluster (called the centroids).

K-means Clustering

- Models data as having K "centroids" defined by mean vectors

$$\mathbf{M} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_K \end{pmatrix} = \begin{pmatrix} \mu_{1,1} & \cdots & \mu_{1,q} \\ \vdots & \ddots & \vdots \\ \mu_{K,1} & \cdots & \mu_{K,q} \end{pmatrix}$$

- Assigns items to class with minimum squared distance to mean.

K-means Clustering

- Models data as having K "centroids" defined by mean vectors

$$\mathbf{M} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_K \end{pmatrix} = \begin{pmatrix} \mu_{1,1} & \cdots & \mu_{1,q} \\ \vdots & \ddots & \vdots \\ \mu_{K,1} & \cdots & \mu_{K,q} \end{pmatrix}$$

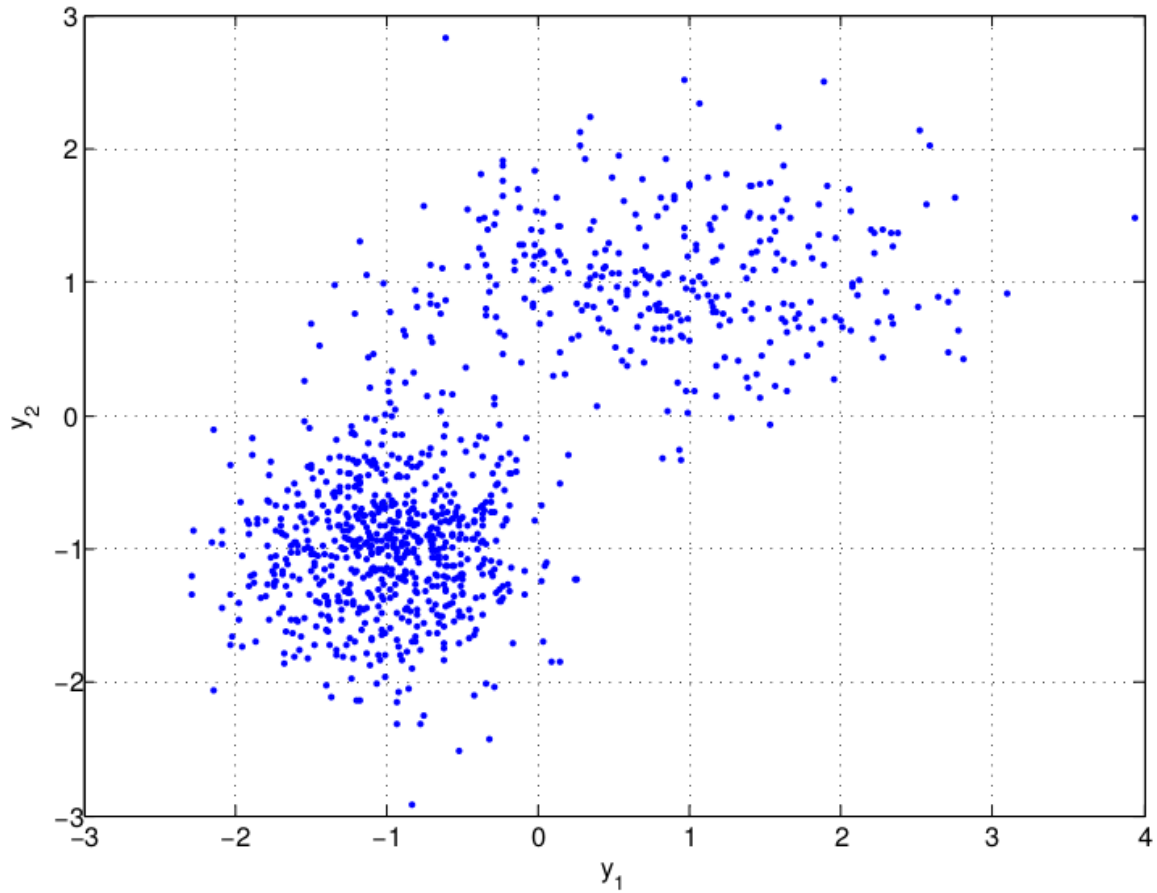
- Assigns items to class with minimum squared distance to mean.
- Similarity between item i and centroid k is

$$d_k^2(i) = \left(\sum_{j=1}^{q} (y_{i,j} - \mu_{k,j})^2 \right)$$

$\Longrightarrow$ Euclidean distance (squared) between the vectors.

K-means Clustering Example

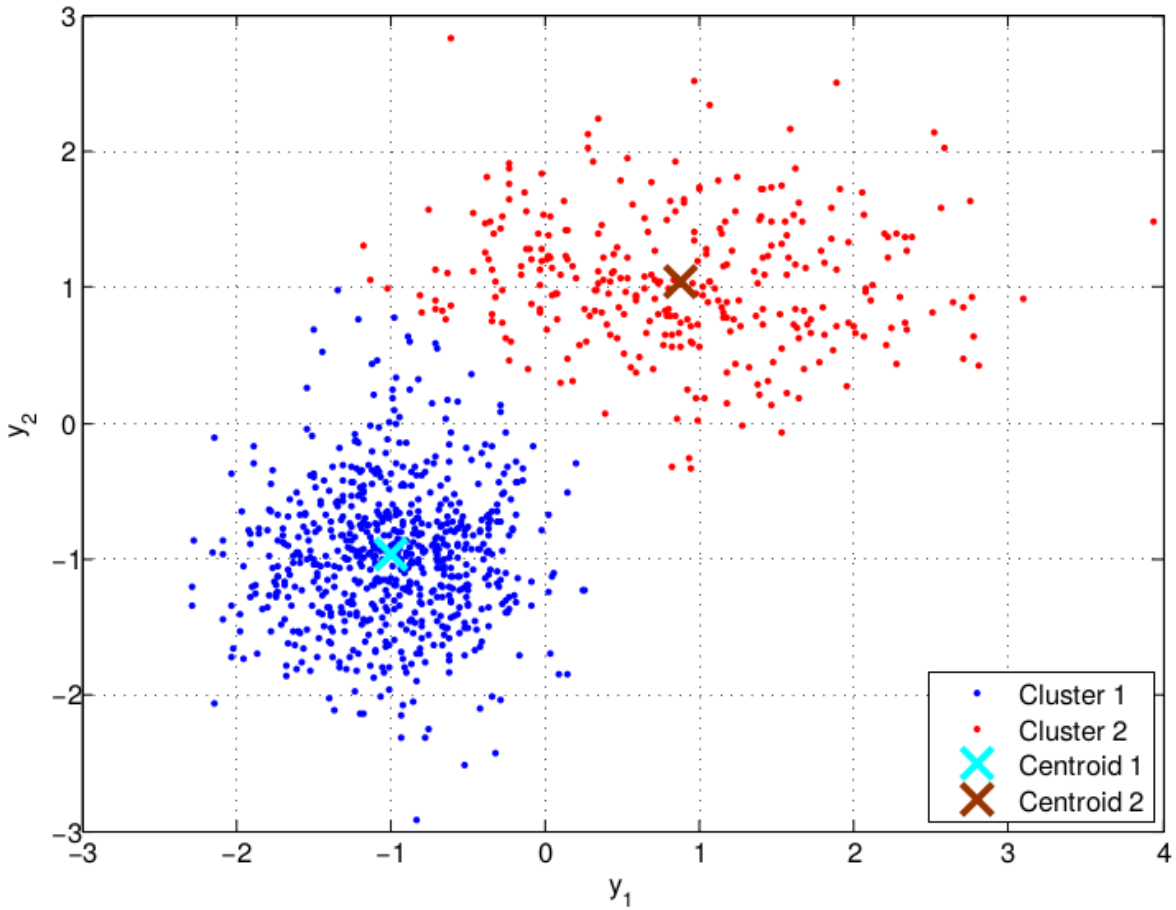
- Artificial data example



- Chosen so that the “clusters” are obvious for demonstration purposes

K-means Clustering Example

- K-means clustering with $K = 2$



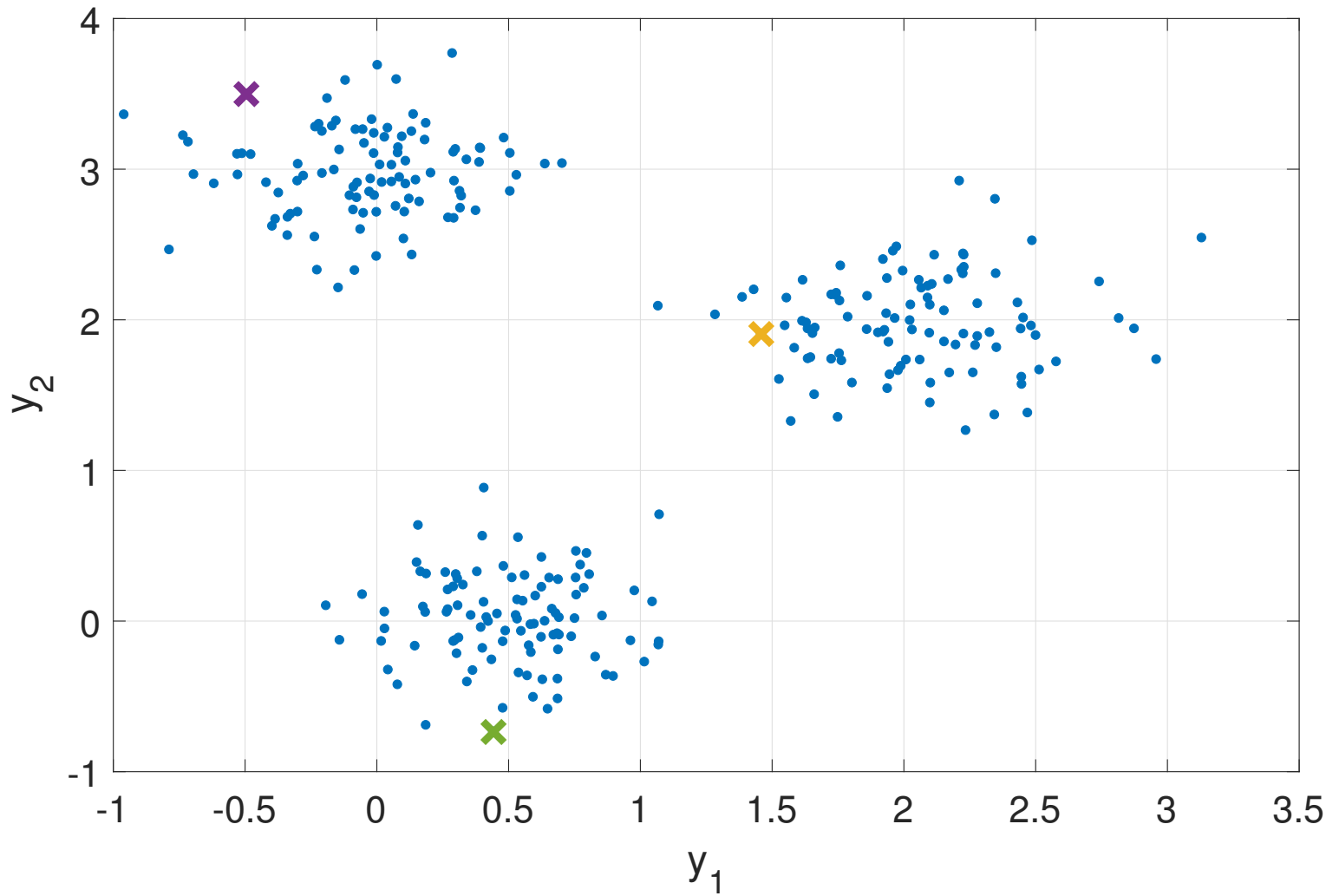
- The centroids are chosen so that the within-cluster sum-of-squares is minimised

K-means Algorithm

- The k-means algorithm is very simple:
  1. Initialise $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$ randomly
  2. Loop until convergence
     - 2.1 Compute distances $d_k(i)$ from each data point $\mathbf{y}_i$ to each centroid $\boldsymbol{\mu}_k$
     - 2.2 Assign datapoints to cluster with closest centroid
     - 2.3 Re-estimate each $\boldsymbol{\mu}_k$ using the datapoints assigned to cluster k
- Converges quickly to a stable solution
  $\Longrightarrow$ might not be the global-minima
- Sensitive to starting points

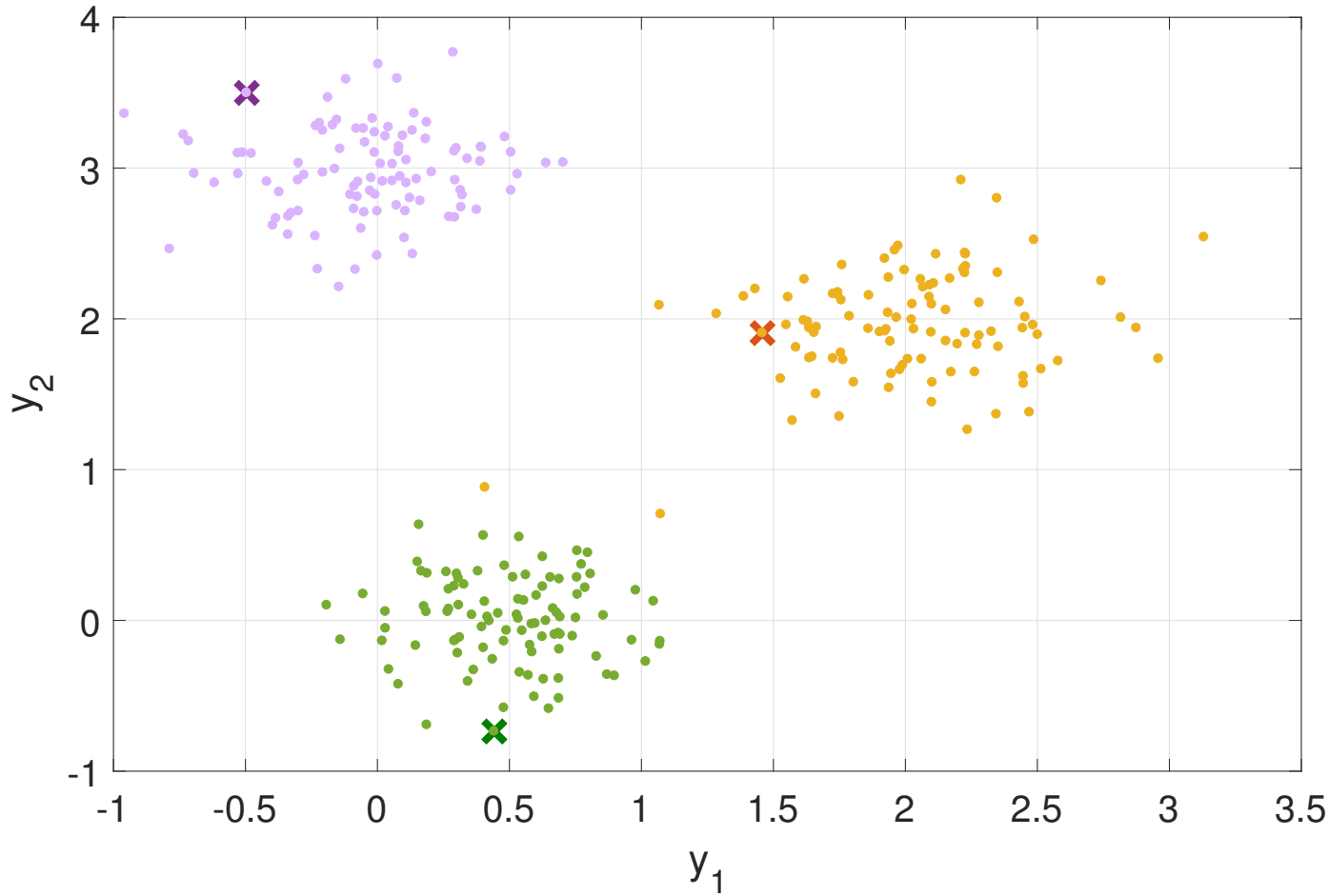
K-means Algor. Example (1)

- Example: $K = 3$, initial starting points for centroids μ_k



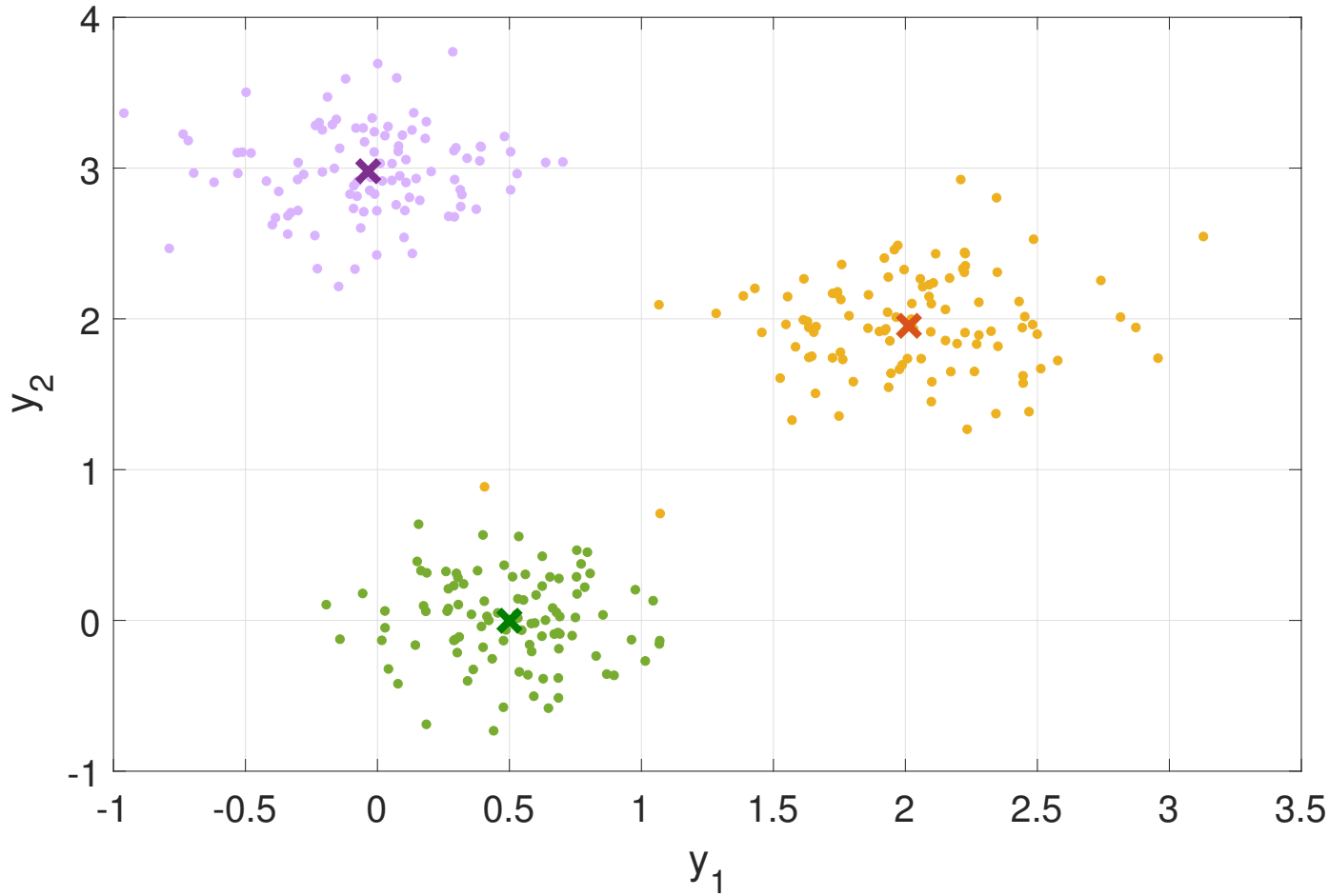
K-means Algor. Example (2)

- Example: assigning points to clusters with closest centroid



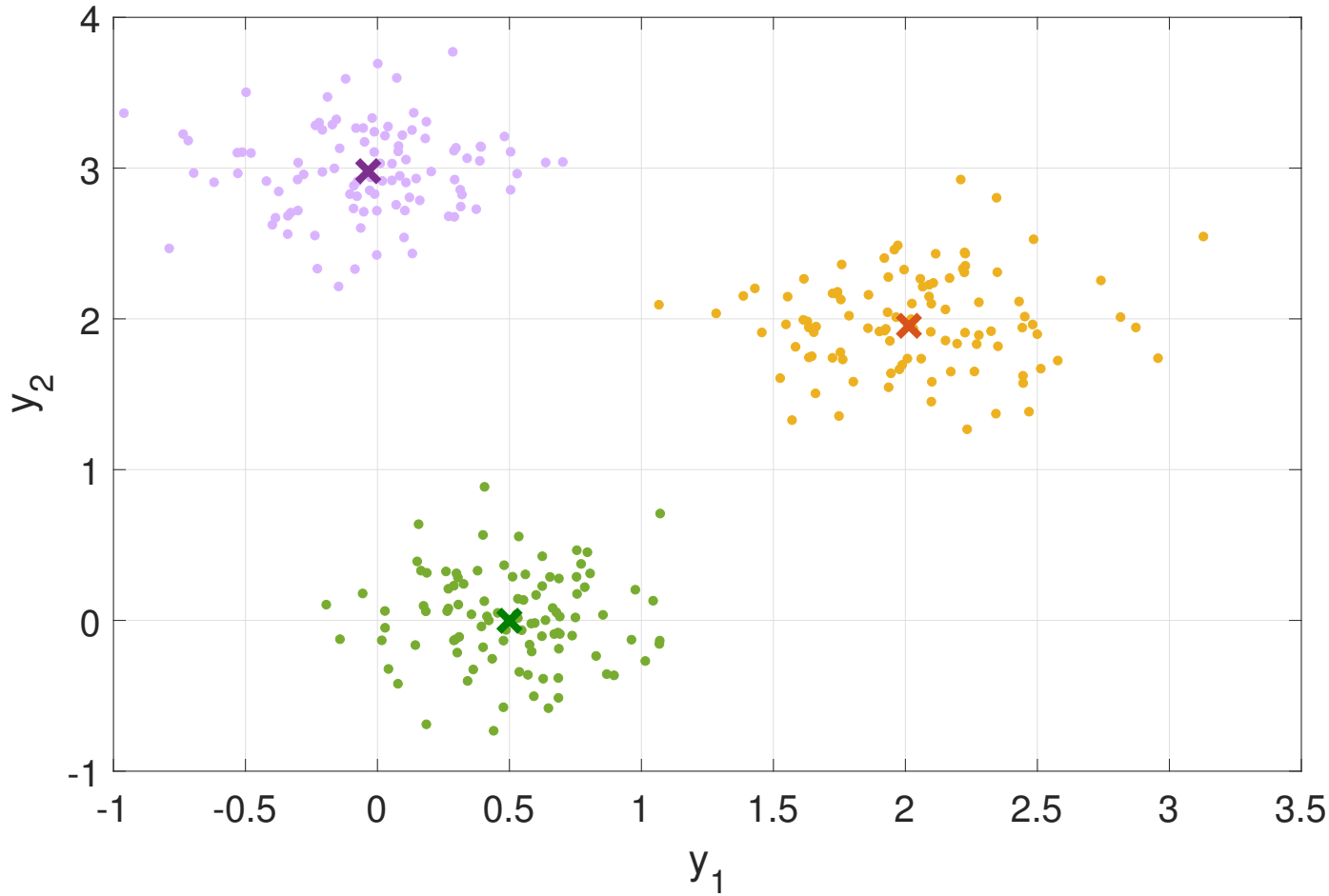
K-means Algor. Example (3)

- Example: re-estimating centroids from data in the clusters



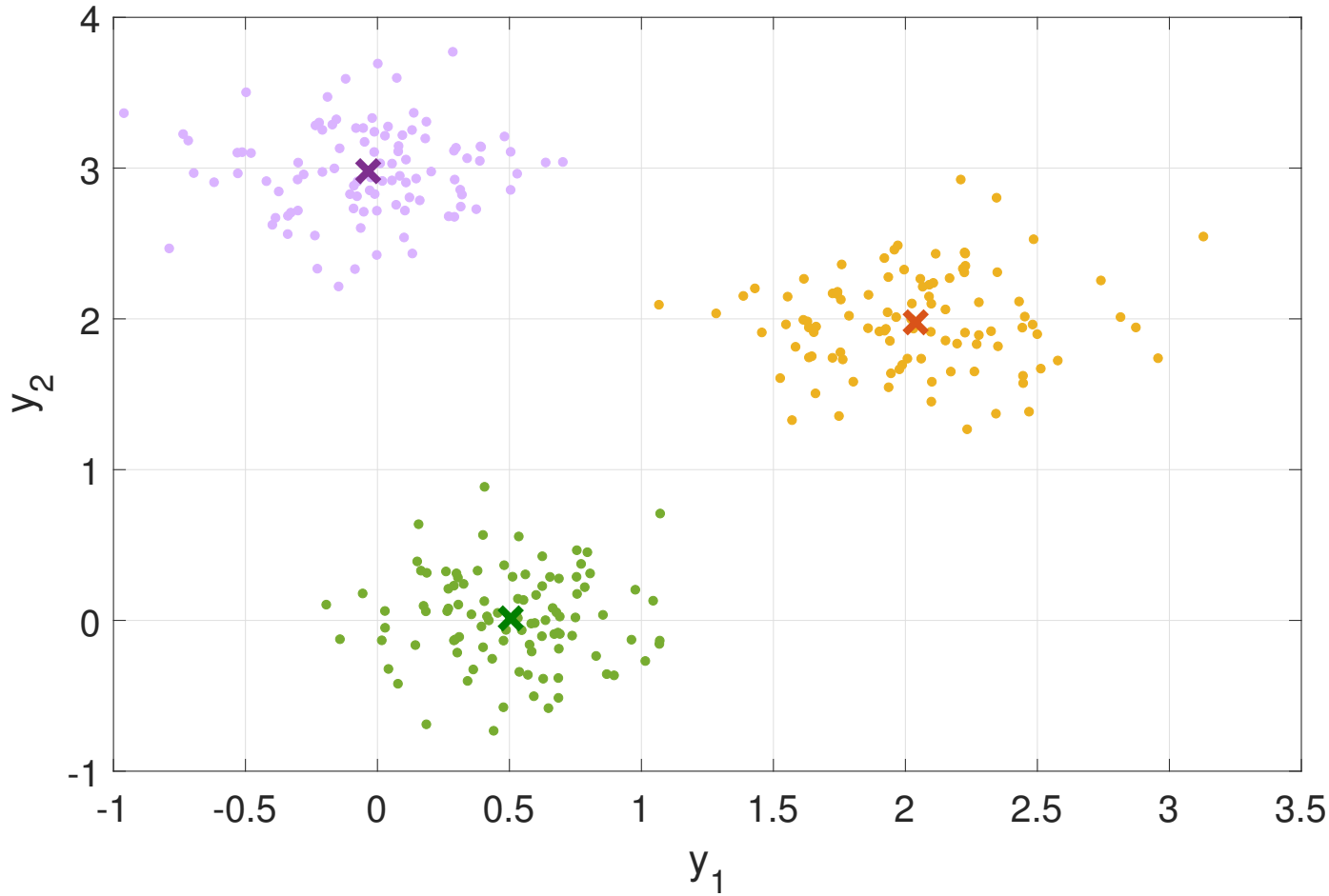
K-means Algor. Example (4)

- Example: assigning points to clusters with closest centroid



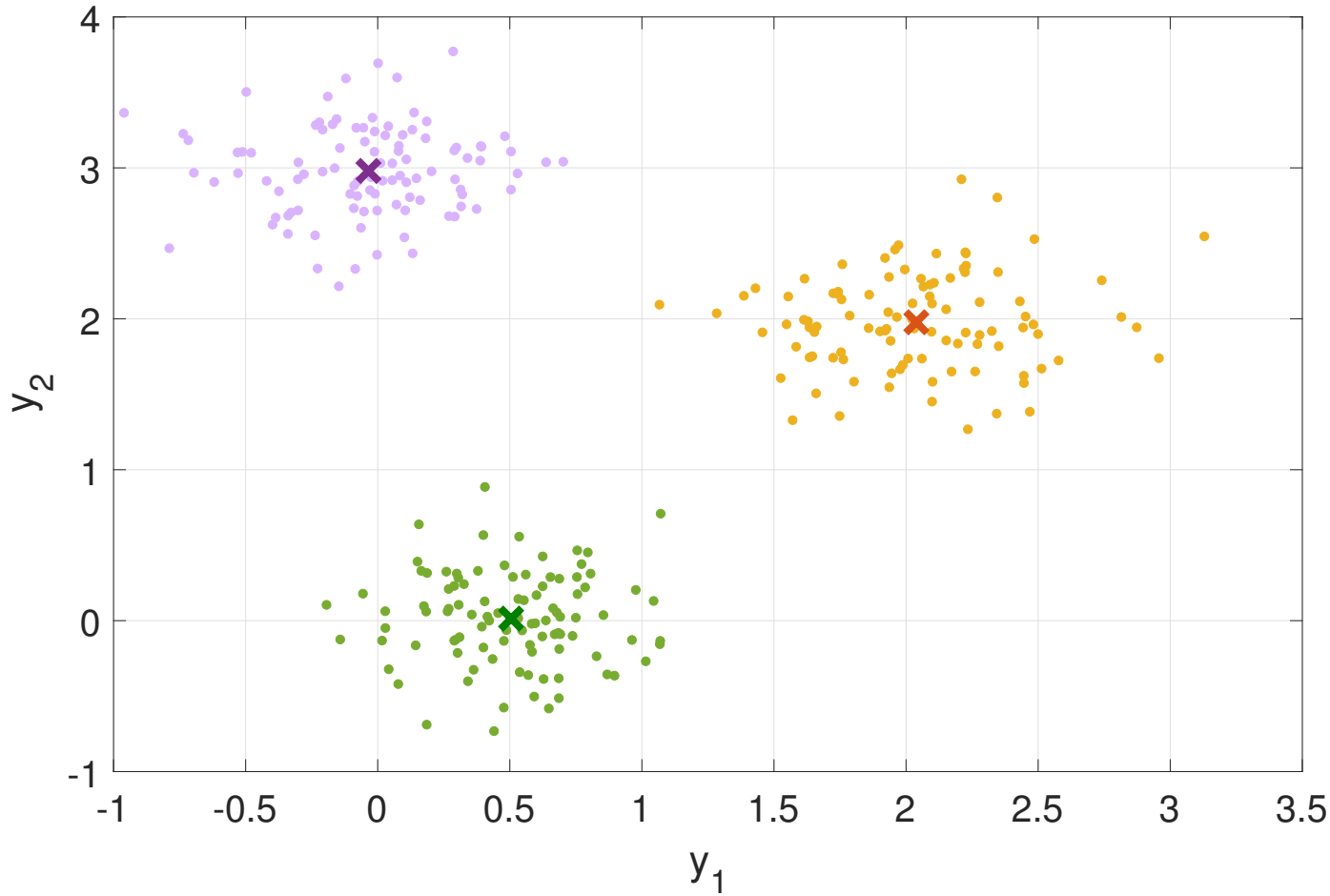
K-means Algor. Example (5)

- Example: re-estimating centroids from data in the clusters



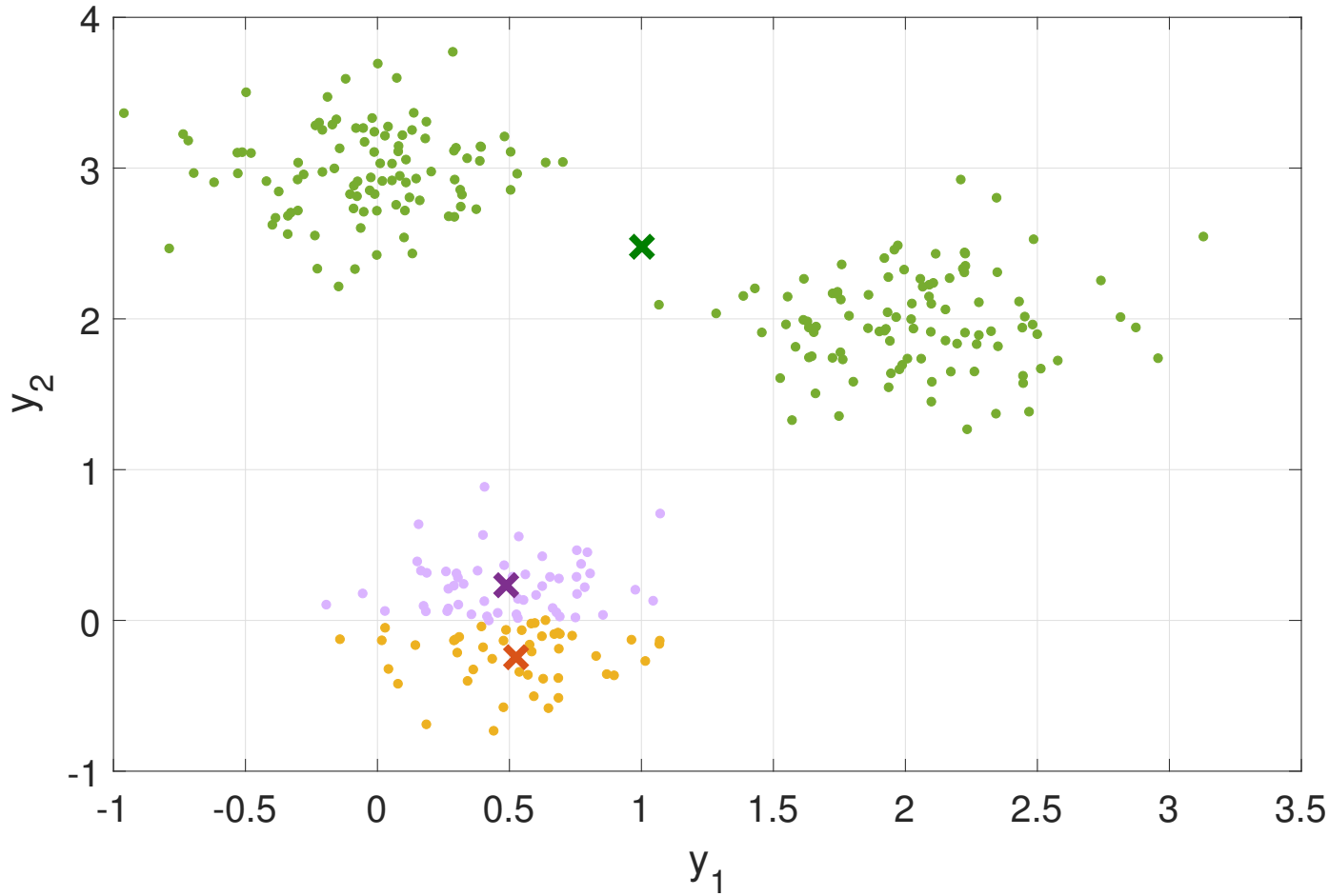
K-means Algor. Example (6)

- Example: after 3 iterations, centroids are stable



K-means Algor. Bad Example

- The k-means algorithm is sensitive to starting points



K-means Algor. Theory

- k-means tries to optimise the function

$$D(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K) = \sum_{i=1}^{n} \min_k \left\{ d_k^2(i) \right\}$$

- That is, it tries to minimise the distances of each point to its nearest centroid.
- k-means has no model in the usual sense: it is defined using a distance function.
- Bad seeding leads to local minima.
- k-means$++$ algorithm improves convergence dramatically $\Longrightarrow$ randomly choose centers to be far apart from each other

Further Clustering

- Alternative similarity measures
  - Weighted Euclidean distance
  - “Cityblock” distance
  - Hamming distance (for pure binary data)
  - and many more ...

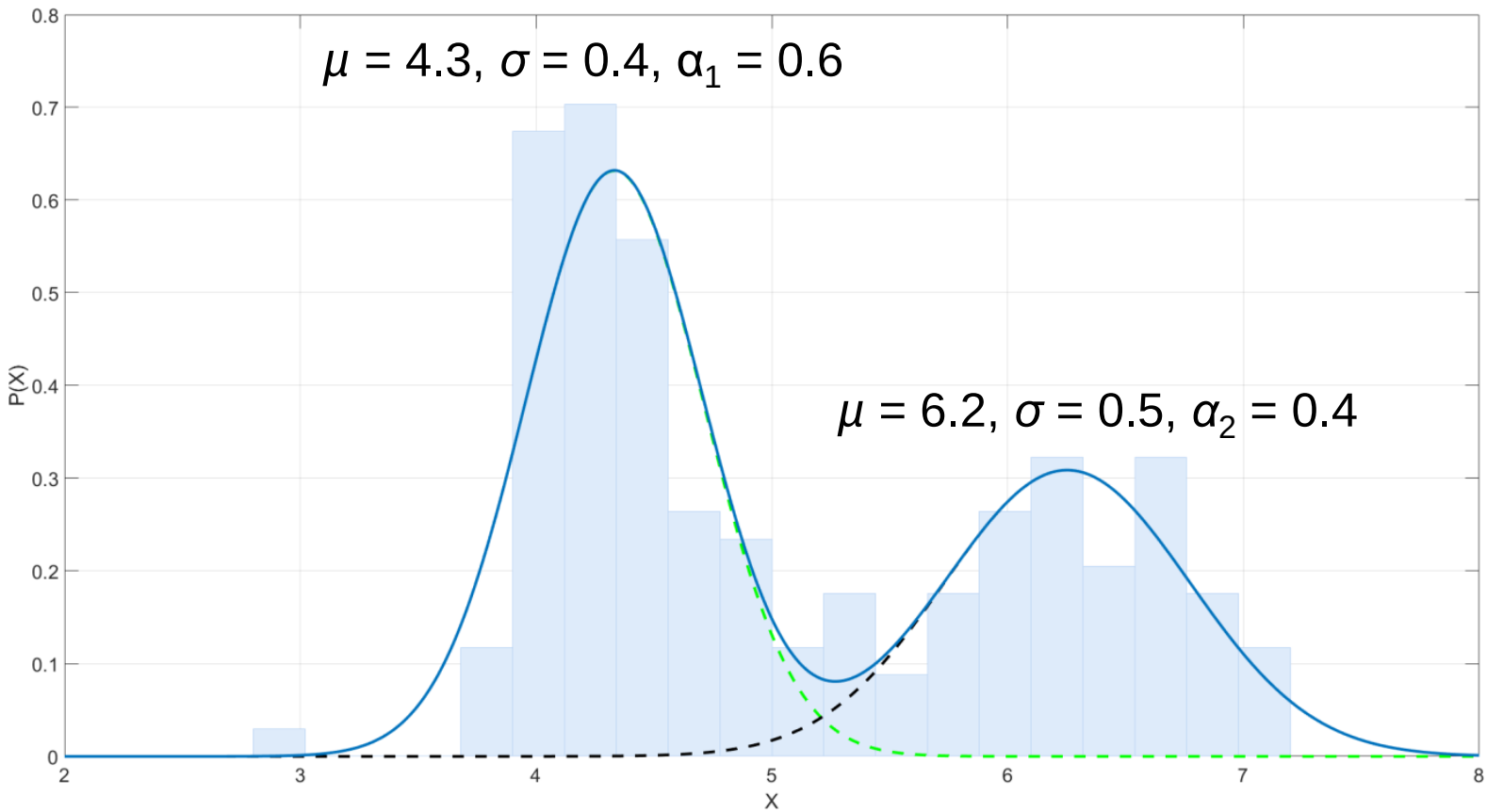
Further Clustering

- Alternative similarity measures
  - Weighted Euclidean distance
  - “Cityblock” distance
  - Hamming distance (for pure binary data)
  - and many more ...
- Some potential issues
  - “Hard” classification of items to clusters
  - Difficult to handle mixed attributes (continuous, discrete)
  - No explicit statistical interpretation
  - How to choose K using just the data?
- Mixture modelling a flexible alternative

Outline

Mixture Modelling Example

- Example: two normal distributions mixed with $\vec{\alpha} = (0.6, 0.4)$



Mixture Modelling

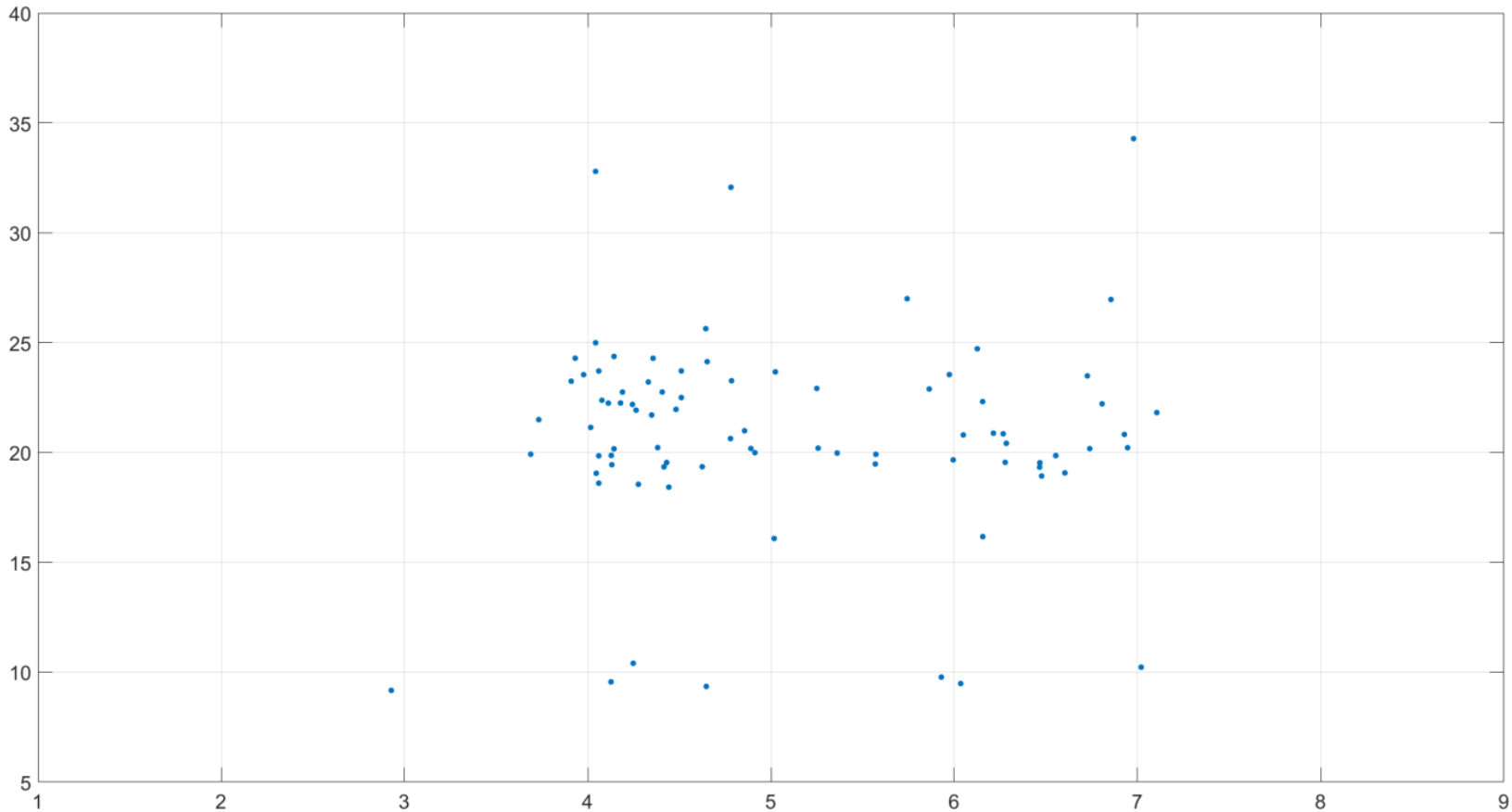
- Models data for each feature as a mixture of probability distributions

$$p(y_{i,j}) = \sum_{k=1}^{K} \alpha_k p(y_{i,j} \mid \theta_{k,j})$$

  where
  - K is the number of classes
  - $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)$ are the mixing (population) weights
  - $\theta_{k,j}$ are the parameters of the distributions
- Has an explicit probabilistic form
  $\Longrightarrow$ allows for statistical interpretion

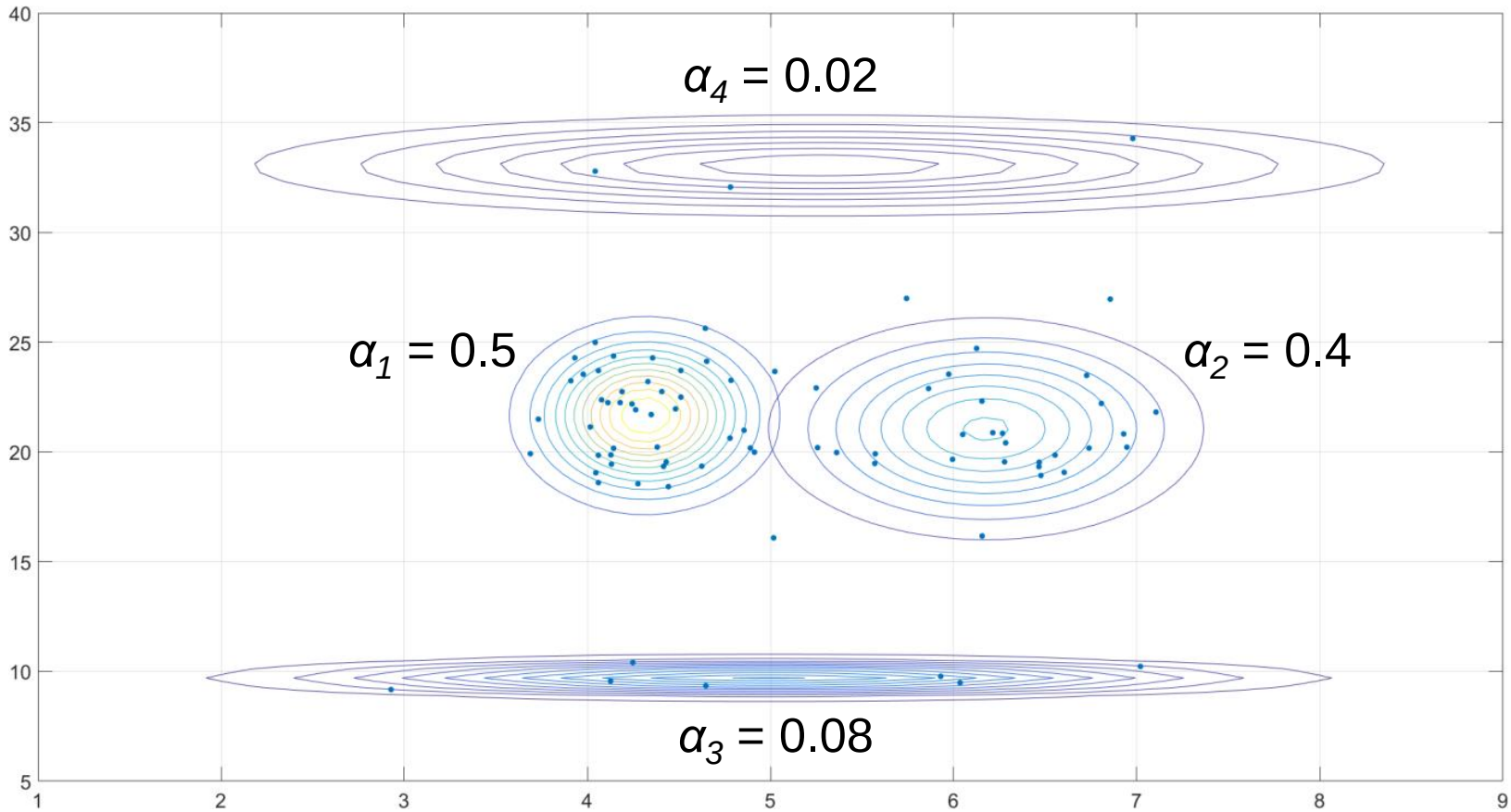
Example Data

- Example: two dimensional dataset



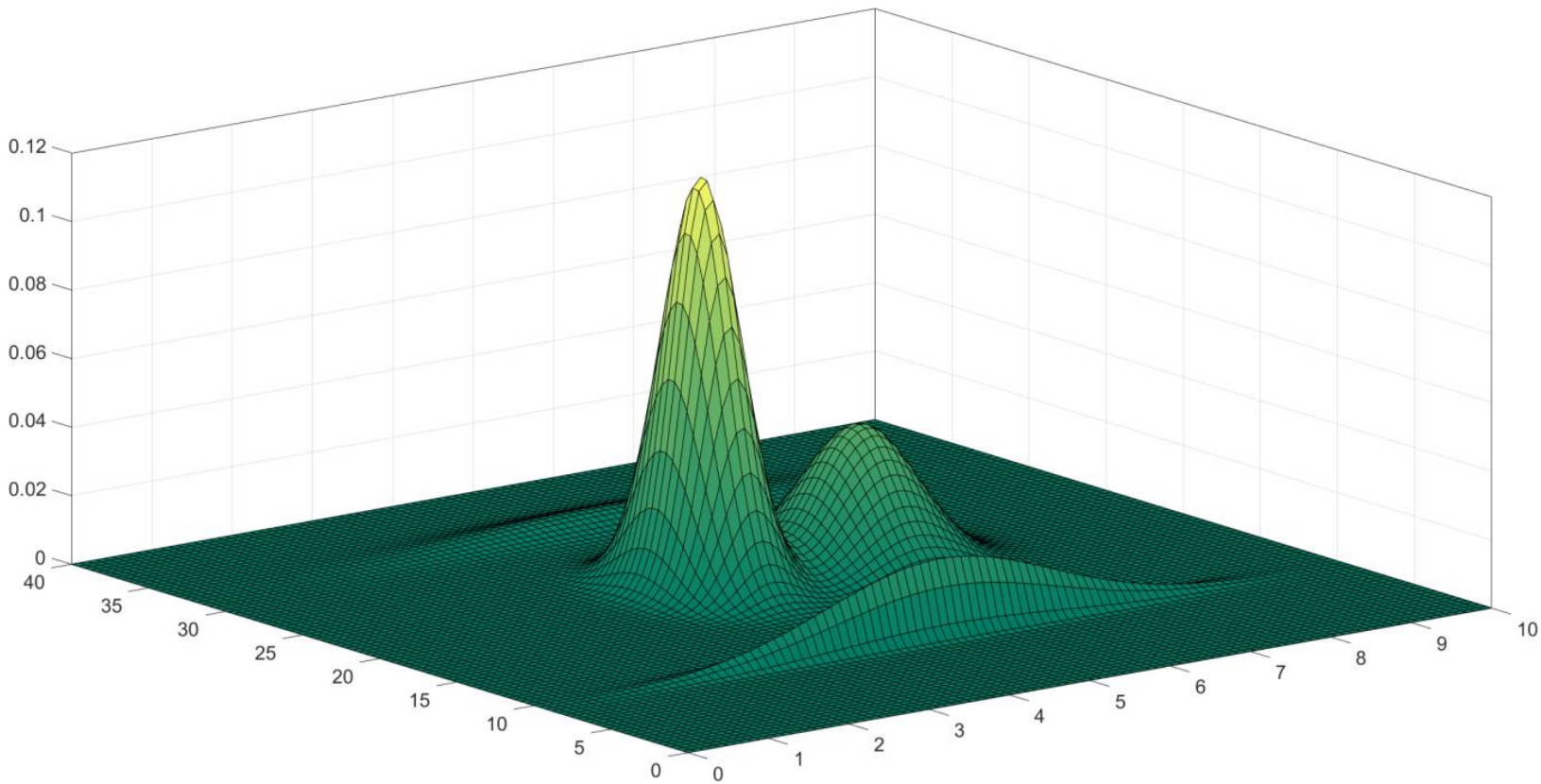
Example Model

- Mixture modelling discovers $K = 4$ classes



Example Plot

- Plot of the mixture model density



Mixture Modelling, cont.

- How is this related to clustering?
- Each class is a cluster
  - Class-specific probability distributions over each attribute
    - e.g., normal, inverse Gaussian, Poisson, etc.
  - Mixing weight is prevalance of items in the class
    - Fraction of our population in that particular subpopulation

Mixture Modelling, cont.

- How is this related to clustering?
- Each class is a cluster
  - Class-specific probability distributions over each attribute
    - e.g., normal, inverse Gaussian, Poisson, etc.
  - Mixing weight is prevalance of items in the class
    - Fraction of our population in that particular subpopulation
- The resulting mixture model has
  - K different classes (subpopulations)
  - q different models for each class, one for each attribute
    - $\theta_{k,j}$ are parameters of model for attribute j in class k
  - $K \times q$ total probability models

End of Week 8