COMP30027 Machine Learning
Discrete & Continuous Data

Semester 1, 2019

Jeremy Nicholson & Tim Baldwin & Karin Verspoor

THE UNIVERSITY OF MELBOURNE

Supervised ML Recap I

- The input to a (supervised) machine learning system consists of:
  - Instances
  - Attributes
  - Class labels
- Possible attribute types (levels of measurement):
  - nominal/categorical
  - ordinal
  - continuous/numeric

Supervised ML Recap II

Class is just a (special) attribute:

- Class is nominal: **Classification**
- Class is numeric: Regression (more in a few weeks)
- (Class is ordinal: ?)

More than Supervised ML, of course:

- Instances aren't labelled: Unsupervised ML (more in a few weeks)
- Not enough instances are labelled: Semi-Supervised ML (more in a few weeks)
- Instances are ordered: Sequence labelling/learning (more in a few weeks)

Supervised ML Recap III

Classification:

- For the learners that we have seen so far, we have made some assumptions about the **type** of attributes in our data set:
  - Naive Bayes, 1-R, Decision Trees: all attributes are **nominal**
  - k-Nearest Neighbour, Nearest Prototype, Support Vector Machines: all attributes are **numeric**
  - 0-R: who cares about attributes?
- What if we have data with the wrong attribute types?
- What if we have a mixture of attribute types?
- Discarding attributes is always possible, but often a bad idea...

Lecture Outline

Our attributes are nominal I

If we have (only) nominal attributes:

- This is how we discussed NB, 1-R, DT
- Need to alter the data into a format suitable for k-NN, NP, SVM

For k-NN and NP, we can fall back to Hamming distance:

$$d_H(A, B) = \sum_i \begin{cases} 0 & \text{if } a_i == b_i \\ 1 & \text{otherwise} \end{cases}$$

(But constructing a prototype is not well defined.)

Our attributes are nominal II

We can randomly assign numbers to attribute values:

- If scale is constant between attributes, this is not as bad an idea as it sounds! (But still undesirable)
- Worse with higher-arity attributes (more attribute values)
- Imposes an attribute ordering which may not exist

Our attributes are nominal III

Most common ML solution: "one–hot encoding"

- If nominal attribute takes m values, replace it with m Boolean attributes
- Example: attribute `temperature`

  $\texttt{hot} = [1, 0, 0]$
  $\texttt{mild} = [0, 1, 0]$
  $\texttt{cool} = [0, 0, 1]$

This is the method of choice, and it "solves" the problem of nominal attributes (by massively increasing the feature space)

Lecture Outline

Naive Bayes I

- To build a Naive Bayes classifier, we must be able to find $P(x_i|c_j)$

a	...	Class
3	...	Y
4	...	Y
7	...	Y
8	...	Y
5	...	Y
...	...	...

What's $P(a = 5|Y)$?

Naive Bayes II

a	...	Class
4.99	...	Y
5.03	...	Y
5.01	...	Y
4.98	...	Y
0.15	...	N
0.21	...	N
0.08	...	N
...	...	...

What's $P(a = 5.02|Y)$?

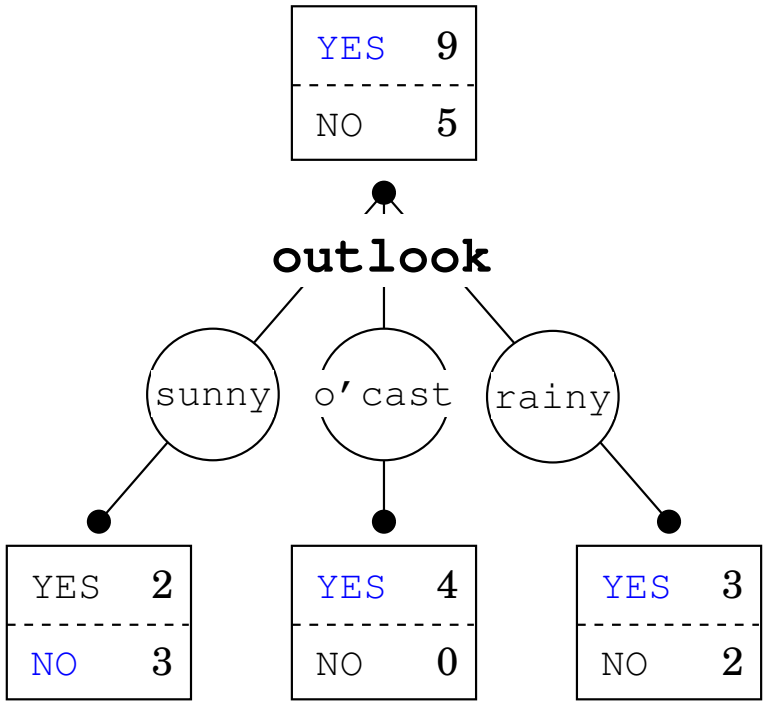
Naive Bayes III

Types of Naive Bayes:

- **Multivariate** NB: attributes are nominal, and can take any (fixed) number of values
- **Binomal** (or Bernoulli) NB: attributes are binary (special case of MV)
- **Multinomial** NB: attributes are natural numbers (usually corresponding to frequencies)
  - Probability distribution is constructed by considering $P(a_k = m|c_j) \approx P(a_k = 1|c_j)^m/(m!)$
- **Gaussian** NB: attributes are real numbers
  - Instead of a Probability Mass Function, we can use a Probability Density Function!
  - More on this later...

Decision Trees I

- To build a Decision Tree, we label a node with an attribute, and branches with corresponding attribute values



Decision Trees II

If the attribute(s) is/are numerical, one popular strategy is **Binarisation**:

- Each node is labelled with a_k, and has two branches: one branch is $a_k \leq m$, one branch is $a_k > m$
- Info Gain/Gain Ratio must be calculated for each non-trivial "split point" for each attribute
  - Naively, this is each unique attribute value in the dataset
  - Faster implementations constrain the number of "split points" we need to consider
- Otherwise equivalent to ID3
- One downside: Binarisation leads to arbitrarily large trees

Decision Trees III

Sometimes the data suggests a richer representation than Binarisation:

a	...	Class
14.99	...	Y
15.03	...	Y
0.01	...	Y
0.48	...	Y
3.15	...	N
5.21	...	N
4.08	...	N
...	...	...

Ranges!

Lecture Outline

What is Discretisation?

- **Discretisation** is the translation of continuous attributes onto nominal attributes:

Continuous	Discretised
0.90	high
0.13	low
0.60	medium
0.35	medium
0.87	high
⋮	⋮

- Discretisation is generally performed as a two-step process:

  decide how many values (= intervals) to map the feature on to

  $$\{(x_0, x_1], (x_1, x_2], \ldots, (x_{n-1}, x_n)\}$$

  map each continuous value onto a discrete value

Source(s): Tan et al. [2006, pp57–63]

Naive Unsupervised Discretisation I

- Treat each unique value as a discrete nominal value:

Continuous	Discretised
0.90	a
0.13	b
0.60	c
0.35	d
0.87	e
⋮	⋮

Naive Unsupervised Discretisation II

Continuous values are nominal values:

- Advantages:
  - simple to implement
- Disadvantages:
  - loss of generality
  - no sense of "numeric proximity"/ordering
  - describes the training data, but nothing more (**overfitting**)

A dataset to Discretise

Consider the following (supervised) instances:

A	C
64	yes
68	yes
69	yes
70	yes
72	yes
75	yes
75	yes
83	yes

A	C
65	no
71	no
72	no
80	no
81	no

Unsupervised Discretisation I

- Partition the values into bins of equal size
- Procedure 1: identify the upper and lower bounds and partition the overall space into n equal intervals = **equal width**:

64	65	68	69	70	71	72	75	80	81	83
yes	no	yes	yes	yes	no	no yes	yes yes	no	no	yes

Unsupervised Discretisation II

Equal width discretisation:

- Advantages:
  - simple
- Disadvantages:
  - badly effected by outliers
  - arbitrary n

Unsupervised Discretisation III

- Procedure 2: sort the values, and identify breakpoints which produce n (roughly) equal-sized partitions = **equal frequency**:

64	65	68	69	70	71	72	75	80	81	83
yes	no	yes	yes	yes	no	no yes	yes yes	no	no	yes

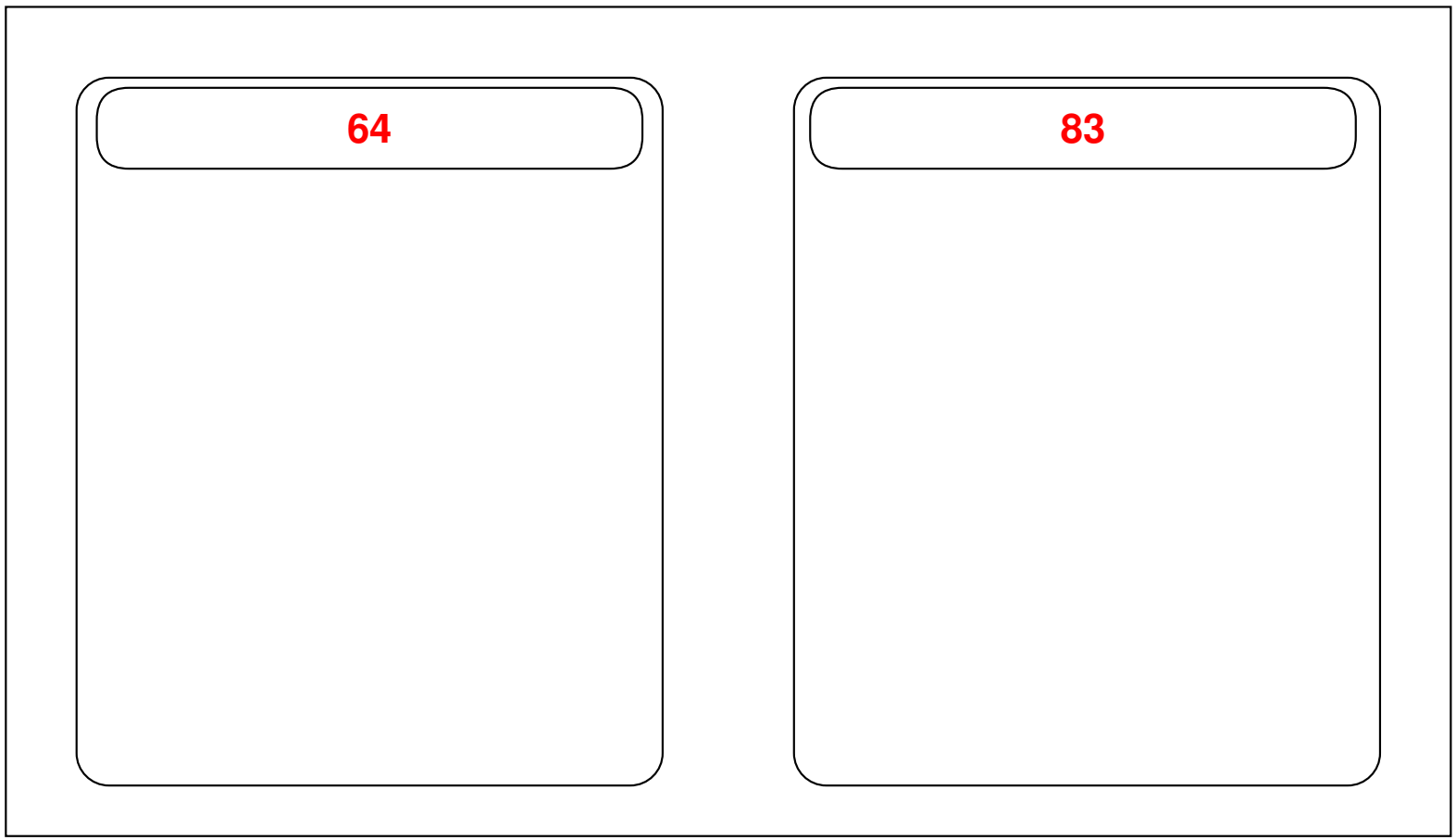
  Equal frequency discretisation
- Advantages:
  - somewhat simple
- Disadvantages:
  - arbitrary n

k-means Clustering

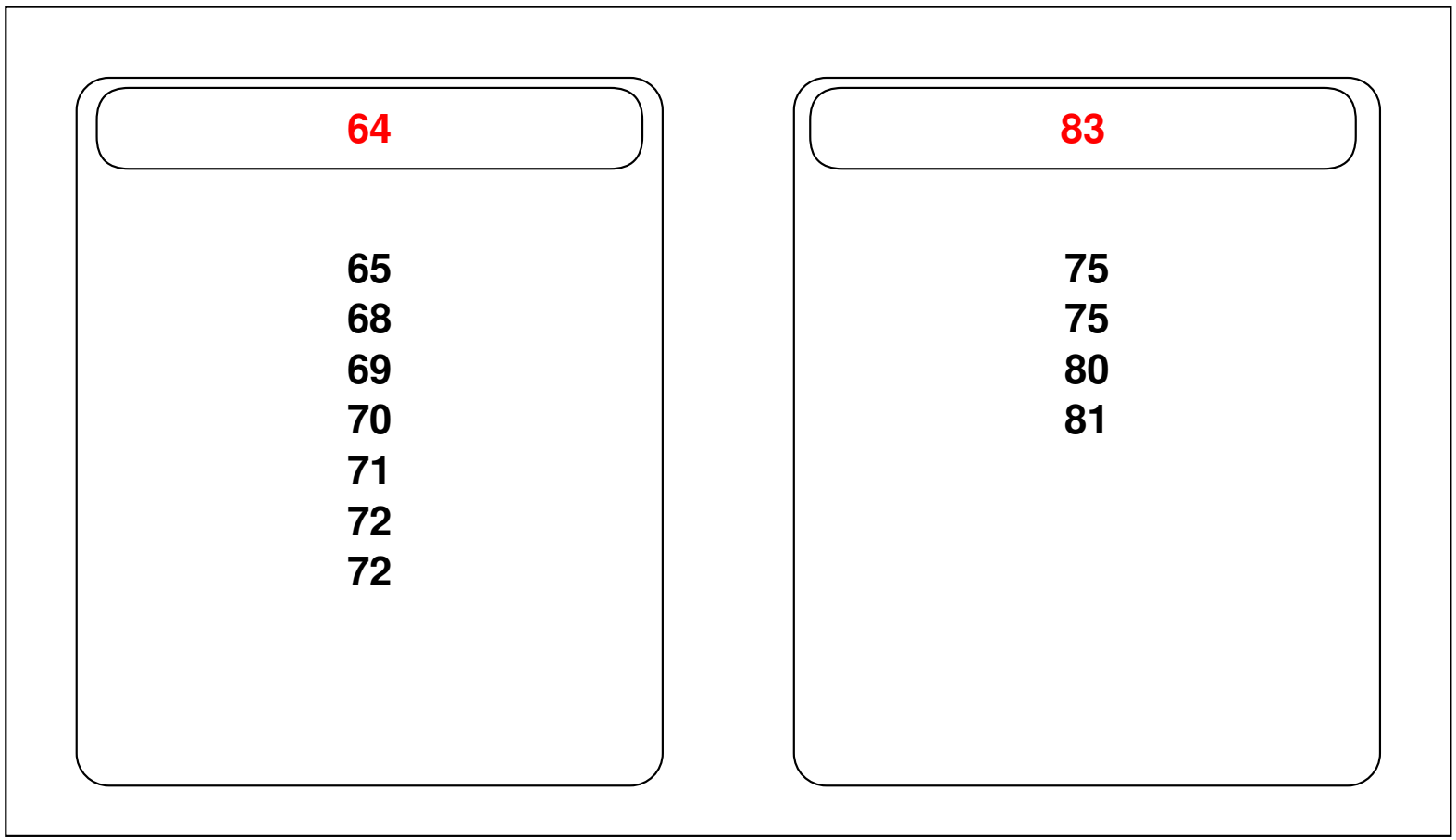
- Given k, the k-means algorithm is implemented in four steps:
  (1) Select k points at random (or otherwise) to act as seed clusters
  (2) Assign each instance to the cluster with the nearest centroid
  (3) Compute seed points as the centroids of the clusters of the current partition (the **centroid** is the centre, i.e., mean point, of the cluster)
  (4) Go back to 2, stop when the assignment of instances to clusters is stable

Source(s): Tan et al. [2006, pp496–515], Jain et al. [1999]

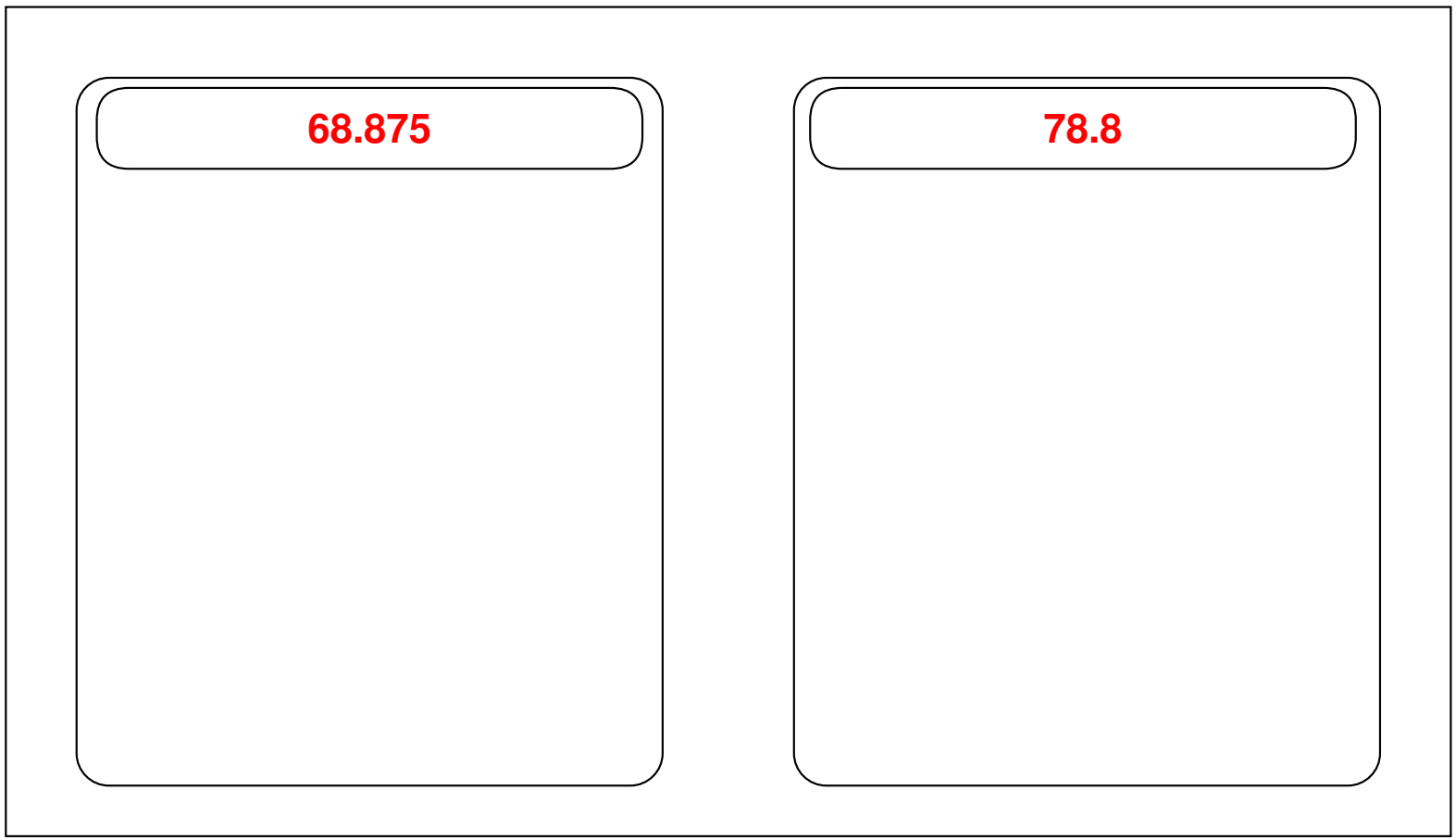
k-means Clustering: Initialisation



k-means Clustering: Iteration 1 (Assignment)



k-means Clustering: Iteration 1 (Centroid)



k-means Clustering: Iteration

- This example converges immediately
  - theoretically, k-means can require any number of iterations up to $C(n, k)$ to converge
  - or it may never converge
  - but fast convergence is fairly typical
- One typical improvement runs k-means multiple times (with random seeds), looking for a common clustering
  - we can simply ignore runs which don't converge within τ iterations

k-means Clustering: Reflections

- Strengths:
  - in this context, ?
- Weaknesses:
  - need multiple runs to have any certainty about convergence
  - random behaviour
  - sensitive to outliers
  - still need to know number of intervals (k)

Naive Supervised Discretisation I

- "Group" values into class-contiguous intervals
- Procedure:

  (1) Sort the values, and identify breakpoints in class membership:

64	65	68	69	70	71	72	72	75	...
yes	no	yes	yes	yes	no	no	yes	yes	...

  (2) Reposition any breakpoints where there is no change in numeric value:

64	65	68	69	70	71	72	72	75	...
yes	no	yes	yes	yes	no	no	yes	yes	...

Naive Supervised Discretisation II

(3) Set the breakpoints midway between the neighbouring values

- Advantages:
  - simple to implement
- Disadvantages:
  - usually creates too many categories (**overfitting**)

Naive Supervised Discretisation III

- Modified procedure to avoid overfitting (v1): delay inserting a breakpoint until each "cluster" contains at least n instances of a single class:

64	65	68	69	70	71	72	72	75	...
yes	no	yes	yes	yes	no	no	yes	yes	...

Naive Supervised Discretisation IV

- Modified procedure to avoid overfitting (v2): merge neighbouring clusters until they reach a certain size/at least n instances of a single class:

64	65	68	69	70	71	72	72	75	...
yes	no	yes	yes	yes	no	no	yes	yes	...

Lecture Outline

Probability Mass Functions

- So far, we have assigned probabilities (NB) to discrete events, like: $P(Temp = hot|Y) = 0.25$
- Summing over every event gives 1: $\sum_i P(a_k = i|c_j) = 1$
- For numerical attributes, we assign probabilities to ranges, like: $P(a_k \in [2.06, 3.12]|Y) = 0.38$
- This is formally an **integral** over some function: $\int_{2.06}^{3.12} \mathrm{pdf}(a_k = x; c_j)dx = 0.38$
- Integrating over the domain of $a_k, [a, b]$ gives 1: $\int_a^b \mathrm{pdf}(a_k = x; c_j)dx = 1$

Probability Density Function

- One popular PDF is based on the **Gaussian distribution** (or **normal distribution**):
- Given the mean μ and standard deviation σ of a distribution X, it is possible to estimate the probability density for an observation $X = x$ (n.b. not a range):

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Some relevant properties of Gaussian distributions:
  - symmetric about the mean
  - area under the curve = 1

Descriptive Statistics 101

- The **mean** of a sample X (= continuous feature) is the average value it takes:

$$\mu_X = \sum_{i=1}^{N} \frac{x_i}{N}$$

- The **standard deviation** of a sample X (= continuous feature) is:

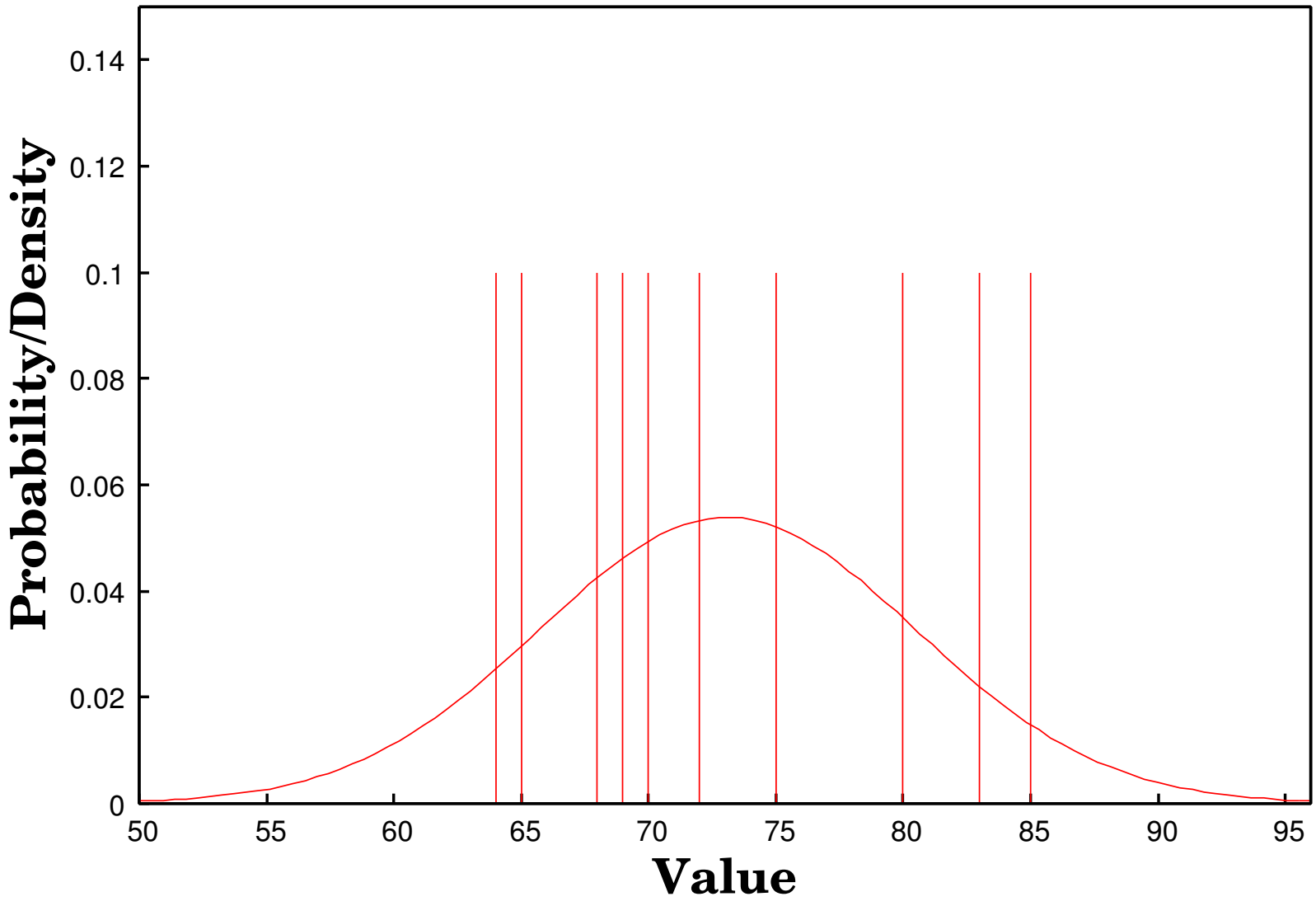
$$\sigma_X = \left| \sqrt{\frac{\sum_{i=1}^{N} (x_i - \mu)^2}{N-1}} \right|$$

Descriptive Statistics 102

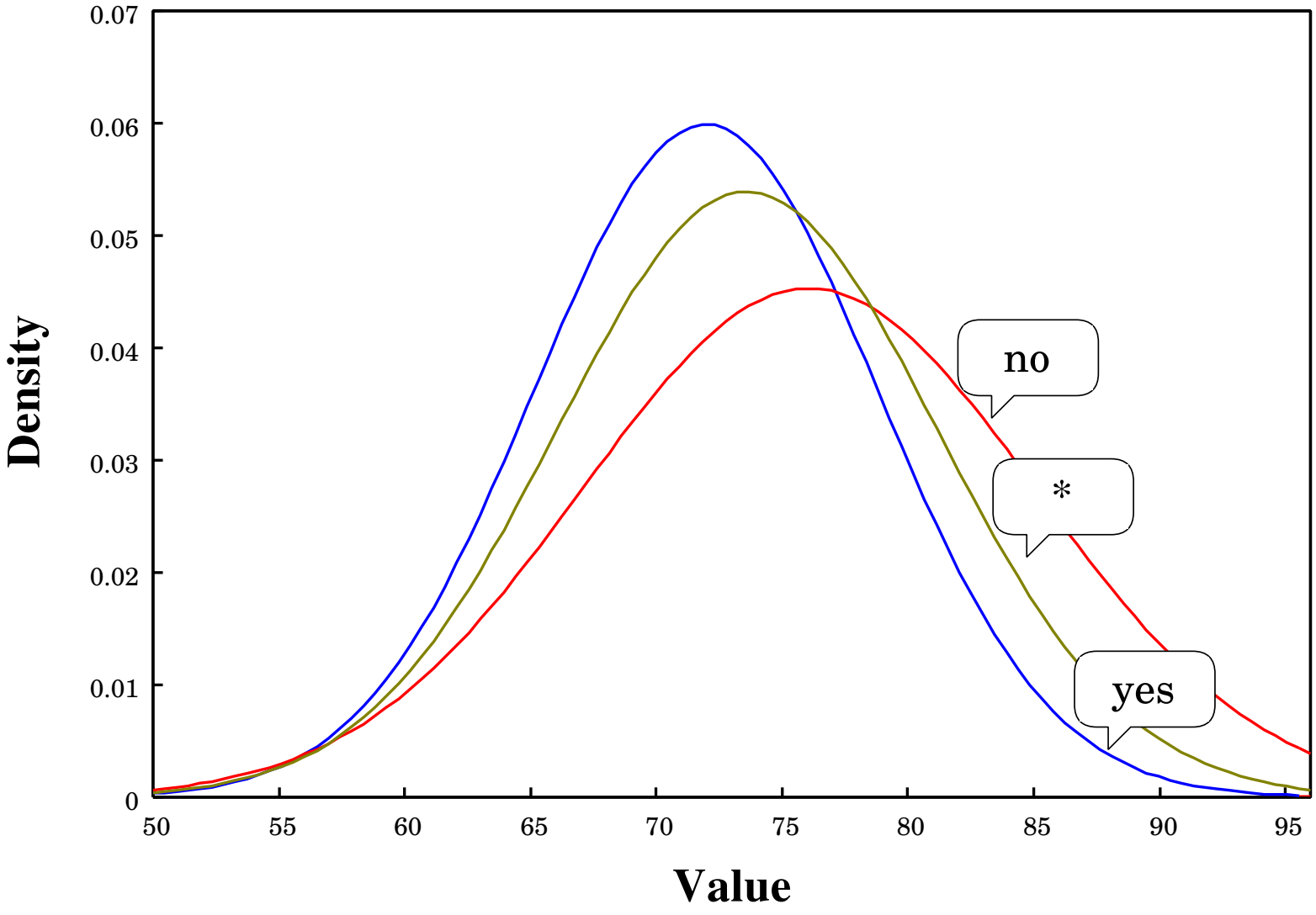
Why Gaussians?

- In practice, a normal distribution is a reasonable approximation for many events
  - including binomial, Poisson, chi-square, and Student's t-distributions
  - e.g. height, weight, shoe size, GPA, ...
- This is a (somewhat surprising) consequence of the **Central Limit Theorem**
- More careful analysis shows that the mean is (almost) always normally distributed, but **outliers** can wreak havoc on our probability estimates

PDF for $P(\texttt{Temperature}|*)$



PDF for $P(\text{Temperature}|X)$



Continuous Naive Bayes in Action

Outlook	Temperature	Humidity	Windy	Play
		Training Data		
sunny	85	high	FALSE	no
sunny	80	high	TRUE	no
overcast	83	high	FALSE	yes
rainy	70	high	FALSE	yes
rainy	68	normal	FALSE	yes
rainy	65	normal	TRUE	no
overcast	64	normal	TRUE	yes
sunny	72	high	FALSE	no
sunny	69	normal	FALSE	yes
rainy	75	normal	FALSE	yes
		Test Data		
sunny	75	normal	TRUE	(yes)

- Priors:

  $P(\texttt{yes}) = \frac{6}{10}$
  $P(\texttt{no}) = \frac{4}{10}$
- Conditional probs:

  $P(\texttt{Temperature = 75}|\texttt{yes}) =$ 0.052
  $P(\texttt{Temperature = 75}|\texttt{no}) =$ 0.045
- Classification of test instance T:

  $P(\texttt{yes}|T) =$
  $\frac{6}{10} \times (\frac{1}{6} \times 0.052 \times \frac{4}{6} \times \frac{1}{6}) \approx 0.0006$
  $P(\texttt{no}|T) =$
  $\frac{4}{10} \times (\frac{3}{4} \times 0.045 \times \frac{1}{4} \times \frac{2}{4}) \approx 0.0017$

Lecture Outline

Summary

- What is the relative fit between the models learned to date and continuous-valued attributes?
- What is discretisation?
- How can we carry out unsupervised and supervised discretisation?
- What are probability density functions, and when do we use them?

References I

A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Addison Wesley, 2006.