Workshop 9

COMP20008 Elements of Data Processing

Learning outcomes

By the end of this class, you should be able to:

- explain the similarities/differences between classification and regression
- explain how predictions are made for decision trees and k-nearest neighbours classifiers
- use decision trees and k-nearest neighbours classifiers in Python

Q1: Classification and regression

What is classification? What is regression? What is the difference between the two?

Q1: Classification and regression

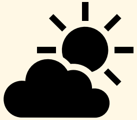
What is classification? What is regression? What is the difference between the two?

	Classification	Regression
Commonality	Both are prediction problems—learn a function that maps an input to an output	
Difference	Output is a class label	Output is a continuous value

Q1: Classification and regression

What is classification? What is regression? What is the difference between the two?

Classification



E.g. predicting the weather conditions: “sunny”, “cloudy”, “rainy”, “windy”, “snowy”

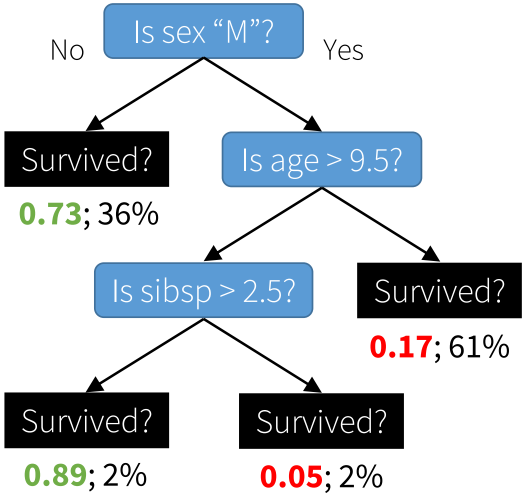
Regression

E.g. predicting the temperature in degrees Celsius

Decision trees

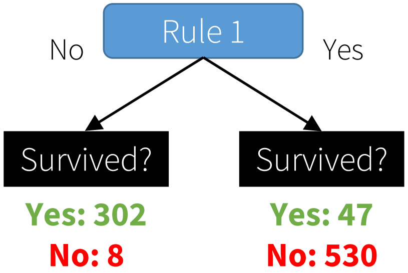
Example: predicting survival of passengers on the Titanic

survived	name	sex	age	sibsp	parch	fare	pclass
No	Mr. Owen Harris…	M	22	1	0	7.25	3
Yes	Mrs. John Bradl…	F	38	1	0	71.283	1
Yes	Miss. Laina Hei…	F	26	0	0	7.925	3
Yes	Mrs. Jacques He…	F	35	1	0	53.1	1
No	Mr. William Hen…	M	35	0	0	8.05	3
No	Mr. James Moran	M	27	0	0	8.4583	3
No	Mr. Timothy J M…	M	54	0	0	51.862	1
No	Master. Gosta L…	M	2	3	1	21.075	3
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

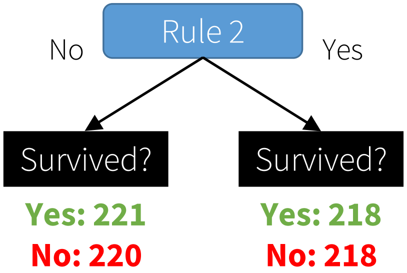


Decision tree splits

- Need to choose a splitting rule when adding a node to the tree
- Splitting rule should achieve high purity



High node purity



Low node purity

Entropy as a measure of impurity

The entropy at a node t is

$$H(t) = -\sum_{y=1}^{n_y} p_y \log p_y$$

where p_y is the relative frequency of class y at node t.

- High node purity when $H(t) = 0$
- Low node purity when $H(t) = \log n_y$ where n_y is the number of classes

Information gain as a splitting criterion

- *Information gain* can be used to measure the quality of a split
- It compares the entropy of the parent node (before splitting) with the entropy of the child nodes (after splitting)

The information gain at node $\boldsymbol{t}$ is

$$IG(t|\text{children}) = H(t) - H(t|\text{children})$$

where

$$H(t|\text{children}) = \sum_{c \in \text{children}} \frac{N(c)}{N} H(c)$$

data points in child node $\boldsymbol{c}$

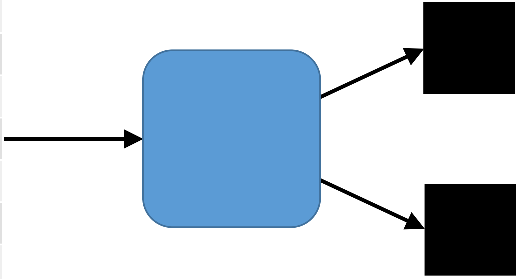
entropy at node $\boldsymbol{c}$

data points in parent node $\boldsymbol{t}$

Q2: Decision tree splits

Suppose we would like to insert a node in a decision tree. Decide which feature should be used for splitting based on the information gain.

Feature A	Feature B	Class label
T	F	+
T	T	+
T	T	+
T	F	–
T	T	+
F	F	–
F	F	–
F	F	–
T	T	–
T	F	–



Q2: Decision tree splits

Feature A	Feature B	Class label
T	F	+
T	T	+
T	T	+
T	F	–
T	T	+
F	F	–
F	F	–
F	F	–
T	T	–
T	F	–

Begin computing the entropy of parent node $\boldsymbol{t}$

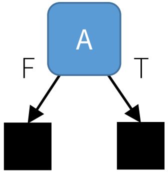
$$H(t) = -\sum_{y \in \text{classes}} p_y \log p_y$$

$$= -\frac{4}{10}\log\frac{4}{10} - \frac{6}{10}\log\frac{6}{10}$$

$$= 0.9710$$

y	p_y
+	4/10
–	6/10

Q2: Decision tree splits



Feature A	Feature B	Class label
T	F	+
T	T	+
T	T	+
T	F	−
T	T	+
F	F	−
F	F	−
F	F	−
T	T	−
T	F	−

Now split on A and compute $H(t|A)$

$$H(t|A) = \frac{N(A=T)}{N} H(A=T) + \frac{N(A=F)}{N} H(A=F)$$

$$= \frac{7}{10} \times 0.9852 + \frac{3}{10} \times 0 = 0.6897$$

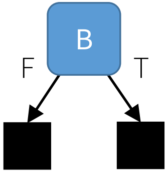
$$H(A=T) = -\frac{4}{7}\log\frac{4}{7} - \frac{3}{7}\log\frac{3}{7} = 0.9852$$

A	y	p_y
T	+	4/7
T	−	3/7

$$H(A=F) = -\frac{3}{3}\log\frac{3}{3} = 0$$

A	y	p_y
F	+	0/3
F	−	3/3

Q2: Decision tree splits



Feature A	Feature B	Class label
T	F	+
T	T	+
T	T	+
T	F	−
T	T	+
F	F	−
F	F	−
F	F	−
T	T	−
T	F	−

Now split on A and compute $H(t|B)$

$$H(t|B) = \frac{N(B=T)}{N} H(B=T) + \frac{N(B=F)}{N} H(B=F)$$

$$= \frac{4}{10} \times 0.8113 + \frac{6}{10} \times 0.6500 = 0.7145$$

$$H(B=T) = -\frac{3}{4}\log\frac{3}{4} - \frac{1}{4}\log\frac{1}{4} = 0.8113$$

B	y	p_y
T	+	3/4
T	−	1/4

$$H(B=F) = -\frac{1}{6}\log\frac{1}{6} - \frac{5}{6}\log\frac{5}{6} = 0.6500$$

B	y	p_y
F	+	1/6
F	−	5/6

Q2: Decision tree splits

For split on A:

$$IG(t, A) = H(t) - H(t|A) = 0.9710 - 0.6897 = 0.2813$$

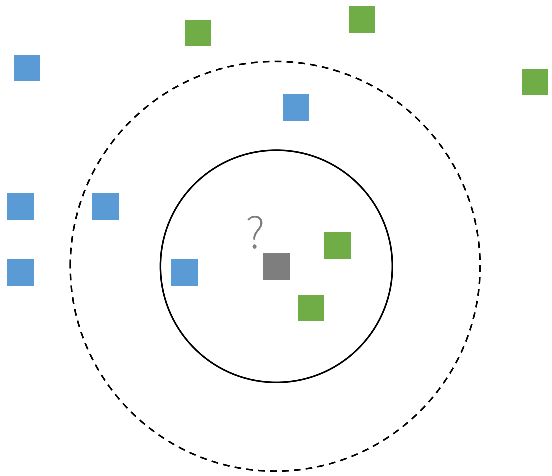
For split on B:

$$IG(t, B) = H(t) - H(t|B) = 0.9710 - 0.7145 = 0.2565$$

So we split on feature A as it maximises the information gain

k-nearest neighbours (kNN) classifier

- Predict the class of an instance based on the majority class of the k nearest neighbours
- Need to specify a distance function to determine the k nearest neighbours
- Feature scaling is often important to achieve good performance



Q3: 1D kNN example

x	0.5	3.0	4.5	4.6	4.9	5.2	5.3	5.5	7.0	9.5
y	−	−	+	+	+	−	−	+	−	−

Classify $\boldsymbol{x = 5.0}$ according to its 1-, 3-, 5-, and 9-nearest neighbours

Q3: 1D kNN example

$k = 1$

x	0.5	3.0	4.5	4.6	4.9	5.0	5.2	5.3	5.5	7.0	9.5
y	−	−	+	+	+	?	−	−	+	−	−

$\hat{y}(5.0) = +$

$k = 3$

x	0.5	3.0	4.5	4.6	4.9	5.0	5.2	5.3	5.5	7.0	9.5
y	−	−	+	+	+	?	−	−	+	−	−

$\hat{y}(5.0) = -$

$k = 5$

x	0.5	3.0	4.5	4.6	4.9	5.0	5.2	5.3	5.5	7.0	9.5
y	−	−	+	+	+	?	−	−	+	−	−

$\hat{y}(5.0) = +$

$k = 9$

x	0.5	3.0	4.5	4.6	4.9	5.0	5.2	5.3	5.5	7.0	9.5
y	−	−	+	+	+	?	−	−	+	−	−

$\hat{y}(5.0) = -$

Q3: 1D kNN example

How does the parameter k affect the k-NN classifier? What would be the behaviour as $k \to \infty$

Q3: 1D kNN example

How does the parameter k affect the k-NN classifier? What would be the behaviour as $k \rightarrow \infty$

- Larger k reduces the affect of noise, but can smooth the decision boundaries between classes too aggressively
- In the limit $k \rightarrow \infty$, the predicted label is the majority class w.r.t. the entire dataset

Q4: Decision trees and missing values

Describe two ways a decision tree could be used to classify a test instance when it has missing features.

Q4: Decision trees and missing values

Describe two ways a decision tree could be used to classify a test instance when it has missing features.

<u>Option 1</u>

Impute the missing features. Then use the decision tree as normal.

Q4: Decision trees and missing values

Describe two ways a decision tree could be used to classify a test instance when it has missing features.

Option 2

Marginalize over the missing features. When we encounter a node that splits on a missing feature:

- The test instance is split among the child nodes according to the split proportions of the training set
- Continue traversing the tree
- End up with a distribution over the class labels
- Choose the class with the highest probability