Decision tree classifiers

Classification & model validation, random forests

Mathematics and Methods in Machine Learning and Neural Networks
Metropolia University of Applied Sciences
Vesa Ollikainen

Learning goals

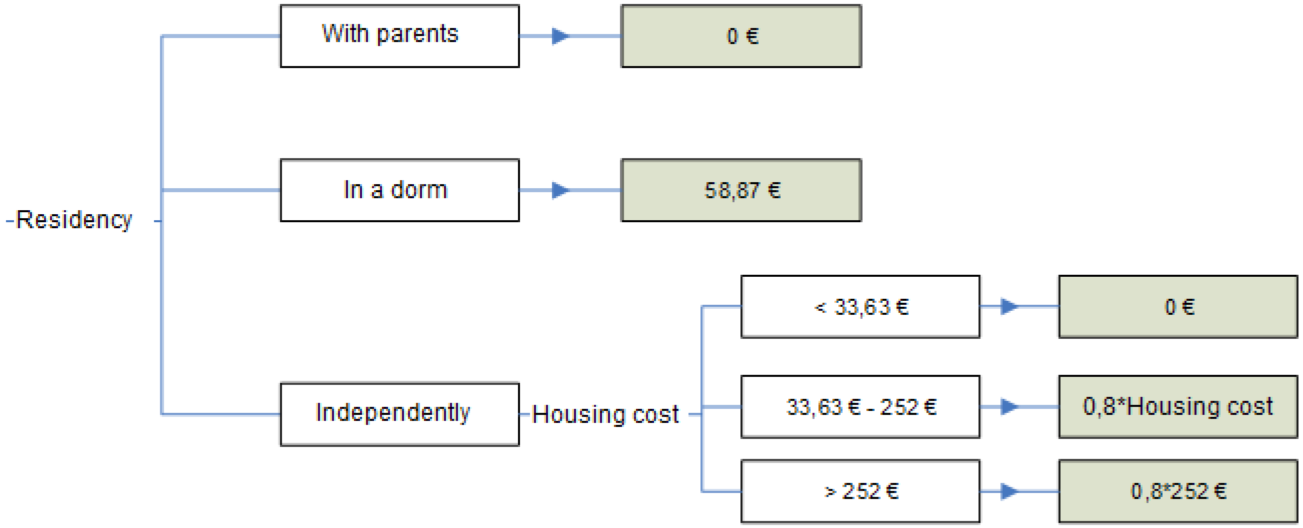
1. Understand how decision trees are used to solve classification problems.
2. Learn the basics of evaluating classification accuracy.
3. Learn to carry out decision tree / random forest analysis.

Idea of decision trees

- Decision trees are a classification method.
- They are suitable for
  - Visualizing decision-making processes
  - Classifying observations into predetermined classes
- The decision trees describe how certain conditions lead into an action or an outcome for each observation
- Decision trees can be used as a tool for prediction.
  - The prediction is based on a decision tree constructed from earlier observations with know outcome.
  - For example, predict occurrence of stroke (yes/no) based on age, smoking, and cholesterol level.
    - The occurence of a stroke is a response variable.
    - The other variables are called explanatory variables.
- The explanatory variables can be of any scale (class, ordinal and/or interval).
- Let’s consider decision trees as a visualization tool first.

A decision tree: an example

- An example depicts the formation of a student's state housing benefit in Finland (until 2017).
- A choice is made in each internal node of the tree.
- The leaf nodes (aka terminal nodes) represent the potential outcomes.

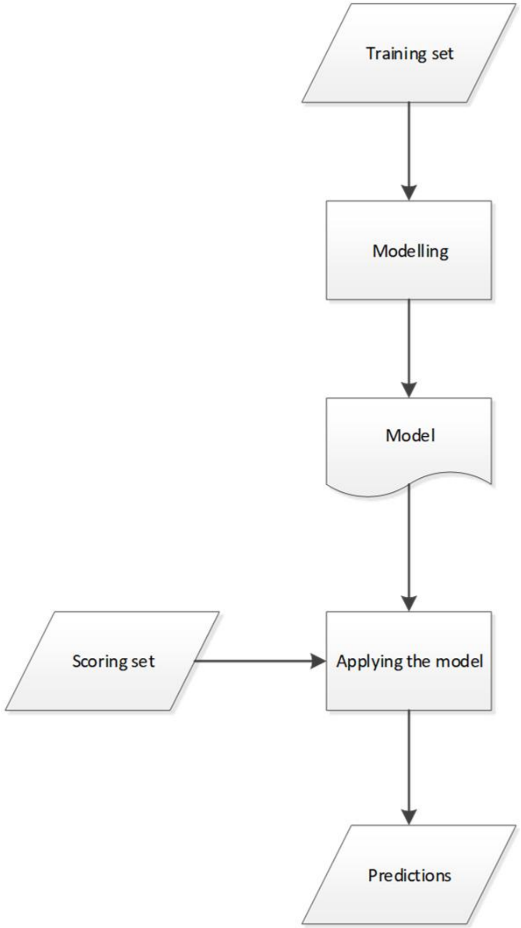


Probability distribution as an outcome of classification

- The decision tree of the previous example produced an absolute outcome (class).
  - The conditions unequivocally determined the class of the observation.
  - There were four classes:
    - No benefit
    - 58,87€
    - 80% of housing costs
    - Maximum benefit (80% × 252€)
- The outcomes of classification can be probability distributions.
- Example: classify fruit into apples and oranges based on peel colour and fruit size.
  - An outcome of a decision tree can be e.g. that an individual fruit has a 93% probability of being an apple and a 7% probability of being an orange.

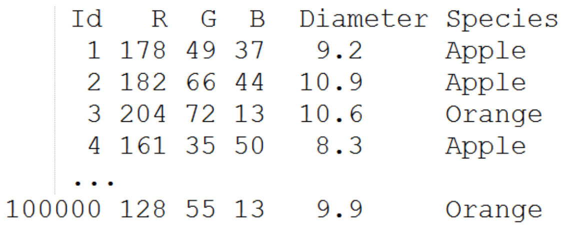
Prediction with decision trees

- Based on a **training set** (aka. *learning set*) a **model** is generated. The model tells the rule how the value of a response variable is deduced based on explanatory variables.
  - For the learning set, the correct answer is known.
- For the **scoring set**, the goal is to **predict** the value of the response variable based on the constructed model.

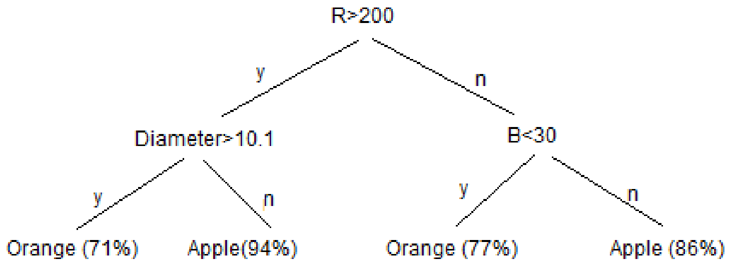


Prediction: an example

- Classify fruit based on peel color (RGB) and diameter.
- Step 1: Build a model (decision tree) based on the training set.
  - Correct answers, i.e. human-classified apple/orange values, are used in the construction

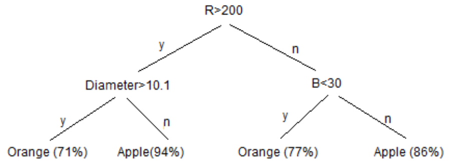


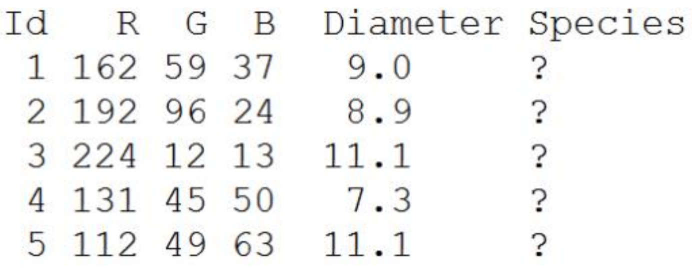
```
    Id    R  G  B  Diameter Species
     1 178 49 37    9.2      Apple
     2 182 66 44   10.9      Apple
     3 204 72 13   10.6      Orange
     4 161 35 50    8.3      Apple
    ...
100000 128 55 13    9.9      Orange
```

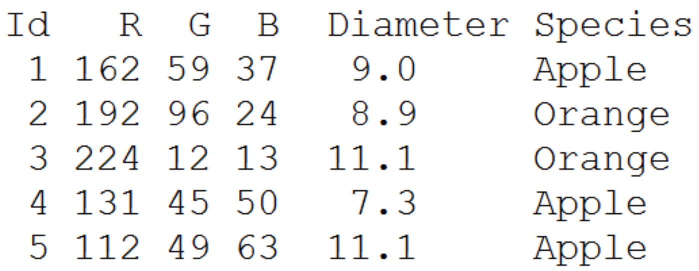


Prediction: an example

- Step 2: In production, the model (the decision tree) is applied for classifying the actual, unknown fruit.



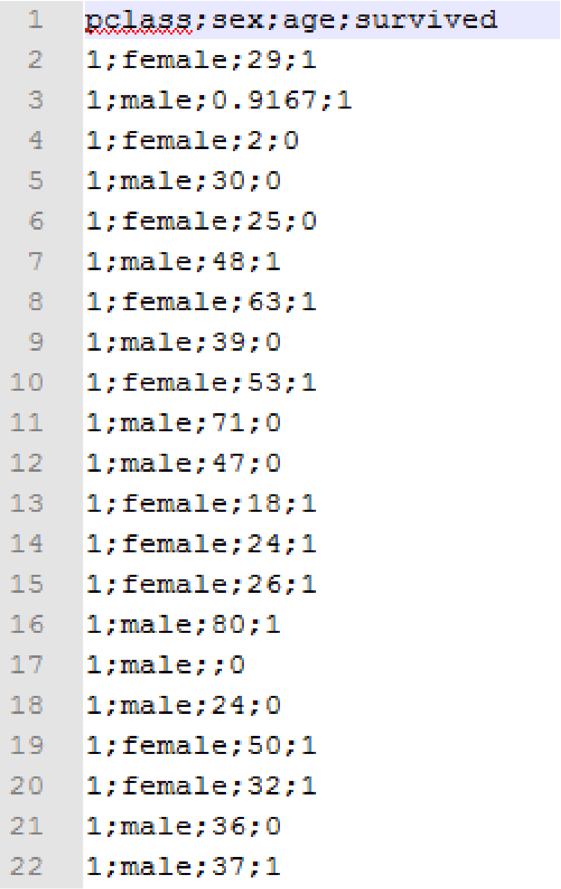

```
Id   R  G  B  Diameter Species
 1 162 59 37    9.0       ?
 2 192 96 24    8.9       ?
 3 224 12 13   11.1       ?
 4 131 45 50    7.3       ?
 5 112 49 63   11.1       ?
```


```
Id   R  G  B  Diameter Species
 1 162 59 37    9.0     Apple
 2 192 96 24    8.9     Orange
 3 224 12 13   11.1     Orange
 4 131 45 50    7.3     Apple
 5 112 49 63   11.1     Apple
```

Construction of a decision tree

- Key question: ”How can we construct the decision tree in such a way that it classifies as well as possible?”
- Good classification referes to the situation where the probability distributions in the leaf nodes are as uneven as possible.
  - This makes the classifications more reliable.
  - E.g. a node with ”93% apples, 7% oranges” is better than a node with ”88% apples, 12% oranges”.
- In the next example, we construct a decision tree for predicting the survival of passengers in RMS Titanic.

Example: RMS Titanic



```
pclass;sex;age;survived
1;female;29;1
1;male;0.9167;1
1;female;2;0
1;male;30;0
1;female;25;0
1;male;48;1
1;female;63;1
1;male;39;0
1;female;53;1
1;male;71;0
1;male;47;0
1;female;18;1
1;female;24;1
1;female;26;1
1;male;80;1
1;male;;0
1;male;24;0
1;female;50;1
1;female;32;1
1;male;36;0
1;male;37;1
```

- RMS Titanic hit an iceberg on its maiden voyage on April 14, 1912.
- There were 1309 passengers onboard.
  - i.e. the data set (passenger record) contains 1309 observations.
- The variables are
  - Travel class (1/2/3)
  - Gender (male/female)
  - Age (integer, except for babies)
  - Survival status (1/0)
- The survival status is considered as a response variable.
  - The remaining variables are explanatory variables.

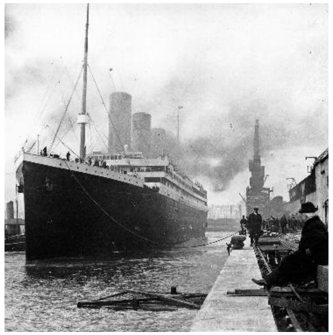
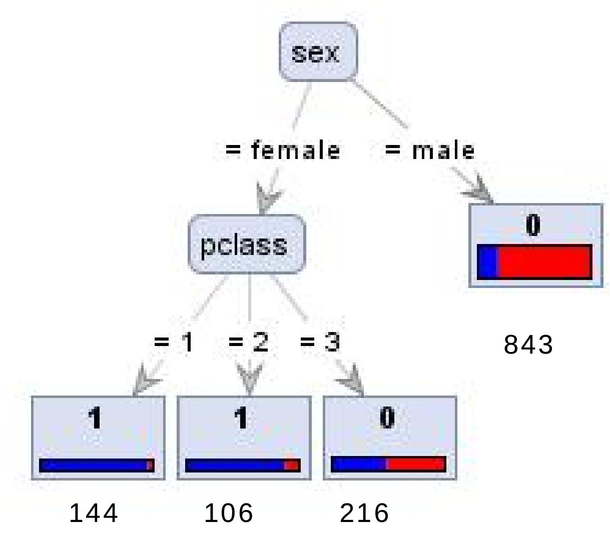
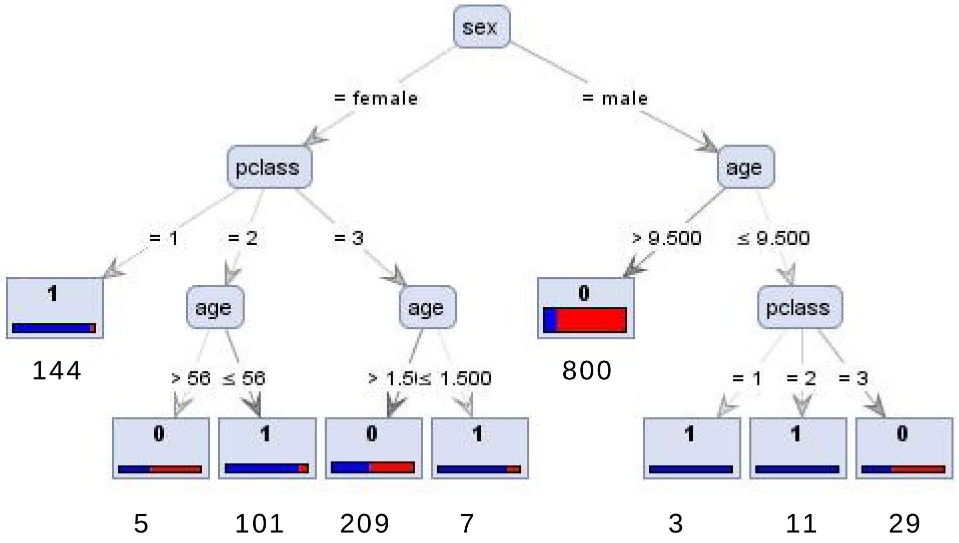


Image: public domain.

RMS Titanic: two decision trees



- The blue vs red color in the bar depicts the proportion of the survived vs deceased passengers.
- The integers below are numbers of observations.



Size of the decision tree

- At first glance, a more complex decision tree automatically seems to produce a more accurate classification.
- However, there’s a danger of model overfitting.
  - There’s always random noise in the data. When the characteristics of the noise are incorporated into the model, the prediction accuracy does not improve.
  - This is revealed by validation, which we will cover shortly.

Hunt’s algorithm

- Hunt’s algorithm is a classic decision-tree construction algorithm.
- It starts from an empty tree that contains only the root. Initially, all observations go to the root node.
- In subsequent steps, the tree is constructed top-down by reiterating the two steps:
    1. Find a division rule that splits the observations in the node into two or more groups in such a way that the distributions of the response variable are as different as possible between the resulting nodes.
    2. Based on the optimal division rule, create two or more child nodes for the node at hand. For each child node, repeat from Step 1 unless the termination criterion is met.
- The termination criterion: quit splitting a node when:
    - All observations fall into the same class, or,
    - There are no differences between the observations that the split can be based on, or
    - The number of observation falls below a predetermined mininum threshold.

Split rules in Hunt’s algorithm

- Initially all passenger of RMS Titanic are in the root node.
- To begin with, all possible split rules are tested:
  A. Split based on gender
  B. Split based on travel class
  C. Split based on age.
    - This is computationally more challenging as there is a infinite number of potential cutoff points to be considered
    - C4.5 algorithm can use non-categorical variables and dynamically find the optimal cutoff point.
- Gini index (see following slide) can be used to find the best split rule.

Gini index in testing split rules

- It is necessary to find a criterion for goodness of split in a decision tree node,
- Gini index (aka. *Gini coefficient* , *Gini impurity*) of a node measures how tightly the observations in a given node fall into the same class.
  - If all observations go strictly into the same class, Gini index equals zero.
  - As the variation increases, Gini index approaches unity.

Gini index

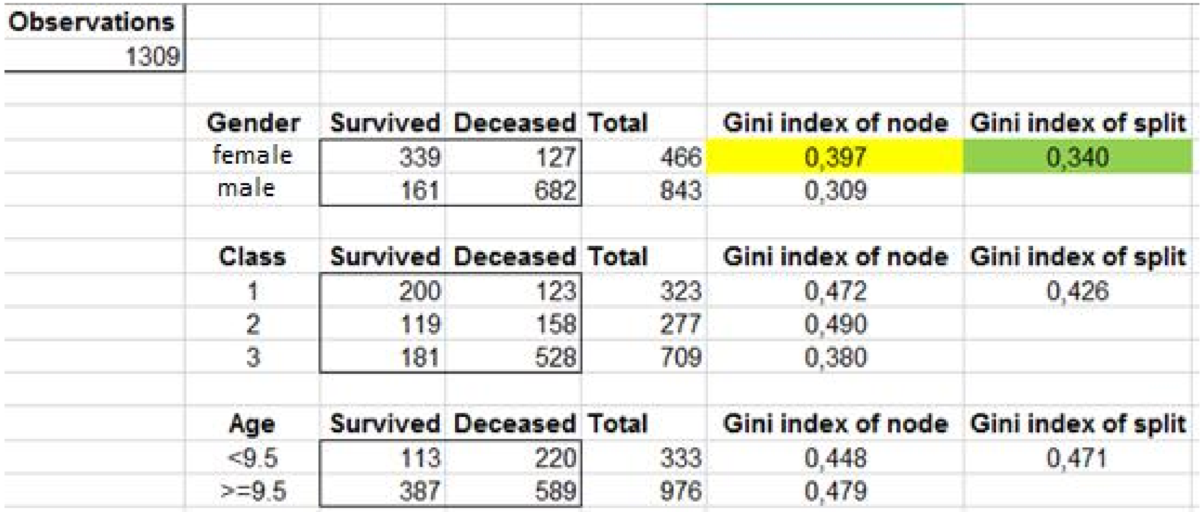
- For a node t: $$g(t) = 1 - \sum_{i=1}^{n} p_i^2$$

where n is the number of classes, and p_i is the probability that an observation falls into class i.

- For a split: $$\sum_{t \in T} \frac{|t|}{|T|} g(t)$$

where T is the set of all child nodes, $|t|$ is the number of observations in a single child node, and $|T|$ is the total number of observations in all child nodes (i.e. the number of observations in the parent node).

Example: selecting a split with Gini index



Observations					
1309					

A

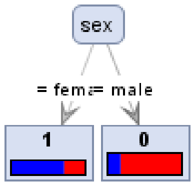
Gender	Survived	Deceased	Total	Gini index of node	Gini index of split
female	339	127	466	0,397	0,340
male	161	682	843	0,309	

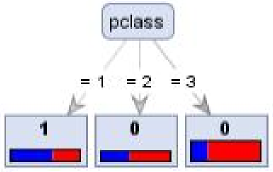
B

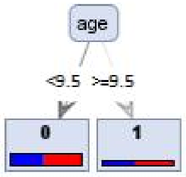
Class	Survived	Deceased	Total	Gini index of node	Gini index of split
1	200	123	323	0,472	0,426
2	119	158	277	0,490	
3	181	528	709	0,380	

C

Age	Survived	Deceased	Total	Gini index of node	Gini index of split
<9.5	113	220	333	0,448	0,471
>=9.5	387	589	976	0,479	







- Calculating the Gini index of a child node in yellow cell:

$$1-\left(\frac{339}{466}\right)^2-\left(\frac{127}{466}\right)^2 = 0{,}397$$

- Gini index for the entire split in the green cell:

$$\frac{466}{1309}\cdot 0{,}397+\frac{843}{1309}\cdot 0{,}309=0{,}340$$

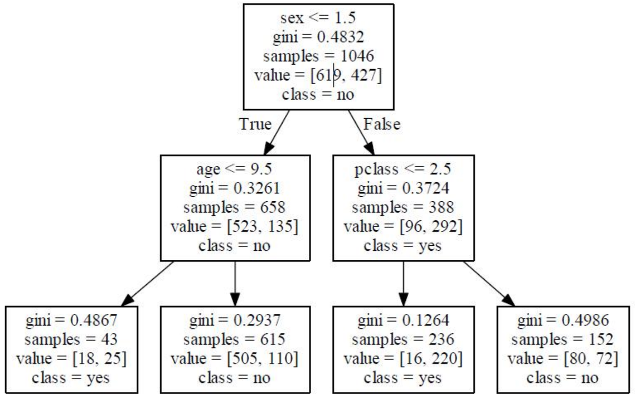
- Choose the split criterion with the lowest Gini index for the entire split (option A, gender).

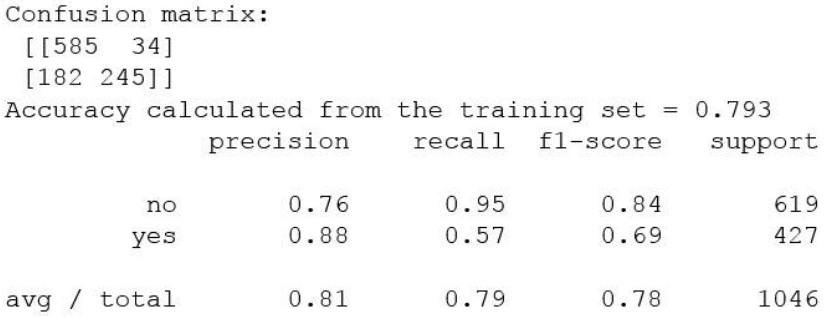
Split criterion and tree size

- The resulting decision tree gets increasingly complex as the nodes are repeatedly split based on Gini index.
- What is an optimal size for the tree?
- The decision tree algorithm can include a distinct pruning phase where the resulting tree is pruned into a simpler shape.

Confusion matrix

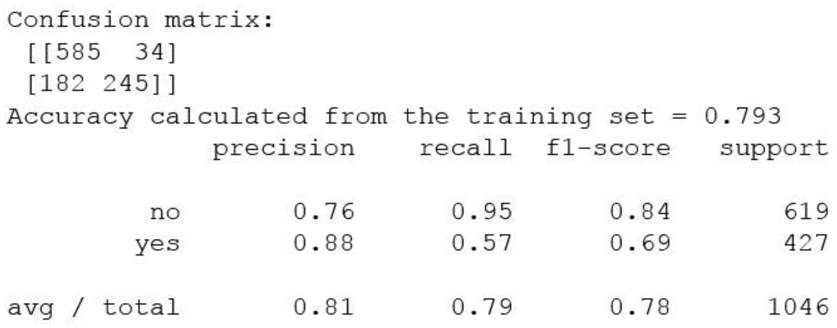
- Confusion matrix is used to evaluate the classification performance of a decision tree.
  - The matrix (aka. contingency table) show how often the true and predicted classifications match.
  - Note that the performance is so far evaluated from the training set.
    - The performance evaluation is likely to be too optimistic.





```
Confusion matrix:
 [[585  34]
 [182 245]]
Accuracy calculated from the training set = 0.793
               precision    recall  f1-score   support

         no       0.76      0.95      0.84       619
        yes       0.88      0.57      0.69       427

avg / total       0.81      0.79      0.78      1046
```

Confusion matrix



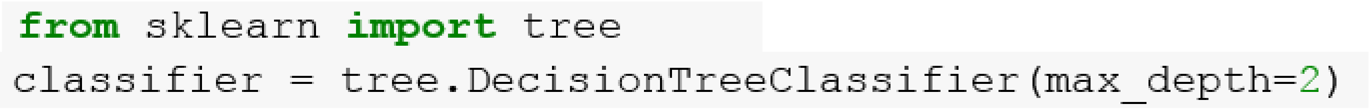
```
Confusion matrix:
 [[585  34]
 [182 245]]
Accuracy calculated from the training set = 0.793
             precision    recall  f1-score   support

         no       0.76      0.95      0.84       619
        yes       0.88      0.57      0.69       427

avg / total       0.81      0.79      0.78      1046
```

- The confusion matrix contains four frequencies.
- Pay attention to the recall and precision figures in the margins.
- E.g. the following results can be seen:
  - The tree classifies correctly 79% of the observations.
  - There were 34 cases where survival was predicted but the passenger died.
  - For survivors, the survival could be predicted with a probability of 57%.
  - For the deceased, the death could be predicted with a probability of 95%.
  - When the decision tree predicts survival, the probability of survival is 88%.
  - When the decision tree predicts death, the probability of death is 76%.
- In Python, use **sklearn.metrics.confusion_matrix()** to compute the confusion matrix.
  - The recalls and the precisions can easily be computed as a post-processing step, or using **sklearn.metrics.classification_report()** .

Decision tree and confusion matrix in Python

- A full example of the decision tree analysis process for the Titanic data can be found in the **Documents/Methods/Data/Titanic** folder of the course's Oma workspace.

Setting parameters in Python



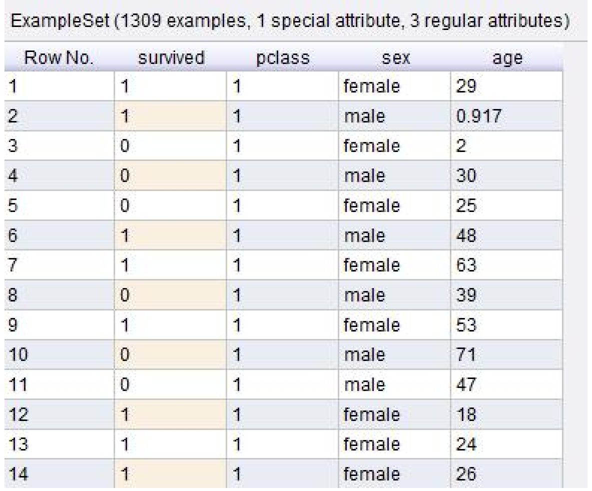
```python
from sklearn import tree
classifier = tree.DecisionTreeClassifier(max_depth=2)
```

- Extreme tree complexity and overfitting is often a problem with decision trees.
- The complexity of a decision tree in Python/**scikit-learn** is mainly controlled by three parameters:
  - **max_depth** defines the maximum depth of the tree.
  - **min_samples_split** and **min_samples_leaf** define the minimum number of observations at any intermediate node, and, respectively, leaf node.
- Any one of them can be used to adjust the size of the resulting tree.

Problem: model overfitting

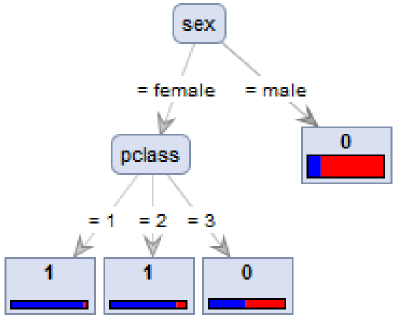
- The data mining software produces the model (e.g. a decision tree) based on the training set.
- When the data set is small, the model can be based on rules that are not applicable in the general population.
- Example: the goal is to construct a decision tree that classifies people into left and right handed persons based on their external characteristics.
  - Let's assume that there are 20 people in the set, 3 of whom are left-handed.
  - It is certain to find a set of characteristics that correctly specifies these 3 persons. For example, it may turn out that all of them have either a hearing aid or shoe size 41, whereas none of the right-handed happens to satisfy this criterion.
  - The resulting decision tree matches the training set perfectly. 100% accuracy!
  - However, as the tree is applied to a new set of individuals, it turns out to be useless.
- Next, we focus on validation that reveals the aforementioned problems.

Problem: Titanic and the decision tree



ExampleSet (1309 examples, 1 special attribute, 3 regular attributes)

Row No.	survived	pclass	sex	age
1	1	1	female	29
2	1	1	male	0.917
3	0	1	female	2
4	0	1	male	30
5	0	1	female	25
6	1	1	male	48
7	1	1	female	63
8	0	1	male	39
9	1	1	female	53
10	0	1	male	71
11	0	1	male	47
12	1	1	female	18
13	1	1	female	24
14	1	1	female	26



- The tree may have adapted to special characteristics of the training set. In a repeated experiment (!) it may not perform as well.

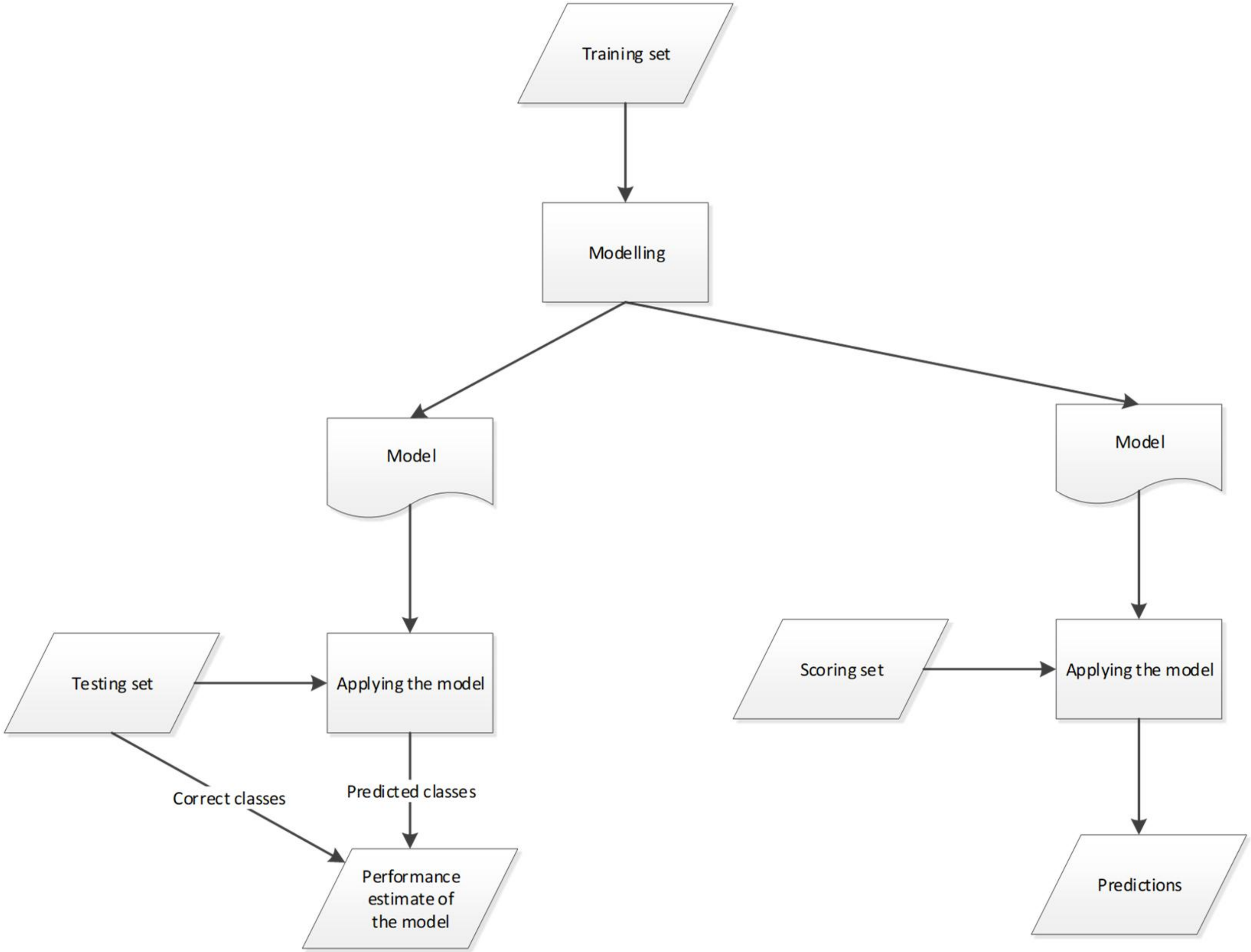
Solution

- Evaluate the goodness of a model by validation.
- Validation relies on three data sets:
  - training set (classes known, used in model construction)
  - testing set (classes known, used in validation)
  - scoring set (classes unknown, predicted)

1. A model is constructed based on the training set.
2. The goodness of the model is evaluated by the testings set.
3. The model is applied to the scoring set.

- The validation methods differ from each other on how the training and testing sets are formed.
  - As the data size is limited, same date needs to be ”recycled” in the training and testing sets.
- The validation methods decribed here generalize to any classifier (neural networks etc.)

Analysis pipeline (with validation)



Validation methods

1. Validation with training set (= no validation)
2. Validation with a separate testing set
3. Cross-validation
4. Split validation

Validation with a separate testing set

- The best option for validation is to do it with new, real data set.
- For this new set, testing set, it is necessary to know the correct classes.
- The predicted classes can then be compared to the known, correct classes.
  - This reveals the true performance of the classifier with a data set that has not been used in the model construction.
- It is not always possible to have a new data set.
  - E.g. in Titanic case, there's just the original passenger data.

Split validation

- Split validation is a straightforward validation strategy.
- The original data set is split into two separate data sets: a training set and a testing set.
  - Thus, not all of the data are used in model construction; a fraction is set apart for validation.
  - The ratio of the sizes of the two data sets is controlled by a parameter: e.g. if 2/3 is used for decision tree construction, then 1/3 can be used as a testing set.
    - A large training set produces a more accurate model, but the estimate of the accuracy is less reliable (due to the small size of the testing set).
    - A small training set may produce a weaker model, but the estimate of the accuracy of the (potentially weaker) model is more reliable.

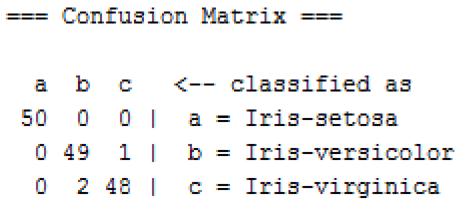
Cross validation

- Cross validation aims at ensuring that a single unlucky split into training and testing set will not skew the validation result.
  - The data set is split into a desired number (*k*) of subsets.
  - The validation procedure comprises *k* rounds.
  - Each of the k subsets acts in turn as a test set.
  - The union of the k-1 remaining subsets makes the training set for that round.
- For example, assuming k=10, in each of the 10 rounds:
  - 90% of the data set acts as a training set. The decision tree is constructed based on that set. The tree can differ from one round to another.
  - The remaining 10% acts as a test set. From this set, it is calculated how well the tree classified in this round.
- Finally, the results obtained from 10 small test sets are combined into a single confusion matrix and a global accuracy estimate.

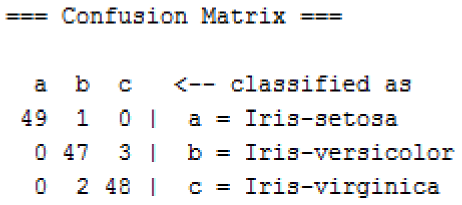
Leave-one-out cross validation

- Leave-on-out cross validation is a special case of cross validation.
- In each round, the testing set contains just one observation.
  - In a data set of n observations, all the remaining $n-1$ observations constitute the training set.
  - Each round produces just one classification result (’correct’ or ’wrong’)
- Finally, the n classification results are combined for a confusion matrix and accuracy estimate.
- Computationally heavy but minimizes the effect of random sampling.

Example



```
=== Confusion Matrix ===

  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 49  1 |  b = Iris-versicolor
  0  2 48 |  c = Iris-virginica
```



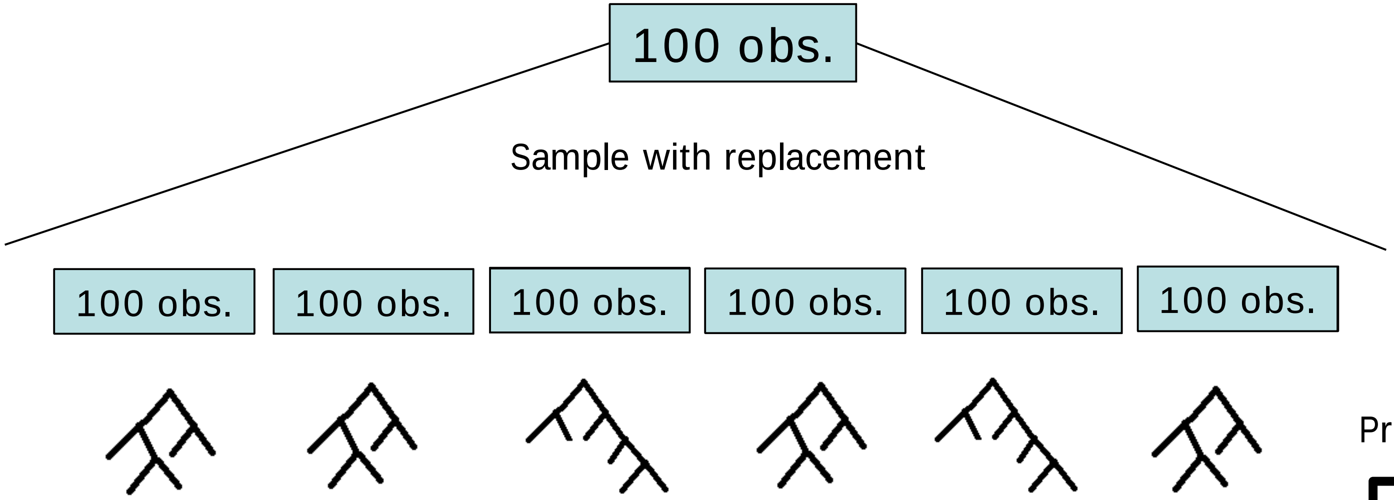
```
=== Confusion Matrix ===

  a  b  c   <-- classified as
 49  1  0 |  a = Iris-setosa
  0 47  3 |  b = Iris-versicolor
  0  2 48 |  c = Iris-virginica
```

- The confusion matrices for a decision tree obtained from the Iris data set:
  - The accuracy calculated from the training set (on the left) is 98%.
    - This corresponds to no validation.
  - The accuracy estimate obtained by cross-validation (k=10, on the right), is 96%.
- The estimate of the accuracy should be based on the results on the right.

Random forests

- The random forest algorithm constructs a set of decision trees simultaneously.
  - It is an example of an ensemble method that creates a collection of models simultaneously.
- Randomness is introduced into the construction of the trees.
- This mitigates model overfitting.
  - The validation is built in the model generation, so a distinct validation phase is not required.
- In **scikit-learn** implementation (**sklearn.ensemble.RandomForestClassifier**):
  - The training set for each tree is of the same size as the original data, but sampled with replacement.
  - A random subset of variables is selected at each intermediate node. The best split for those variables is selected. (**max_features**)
  - The number of trees (e.g. 10) is a parameter. (**n_estimators**)
  - The overall output is the mode of the classifications of the individual trees.
    - That is: if 7 of the trees predict an observation to fall in class 1, and 3 of the trees predict class 2, the “majority vote” wins and the forest outputs class 1.

Random forest



100 obs.

Sample with replacement

100 obs. | 100 obs. | 100 obs. | 100 obs. | 100 obs. | 100 obs.

							Predictions
Obs.1	yes	yes	no	yes	yes	no	yes
Obs.2	yes	no	no	no	no	yes	no
			…				
Obs.100	yes	no	yes	no	yes	yes	yes

Finally

- The decision trees are a classification method that rely on the use of a training set.
- The estimate of the classification accuracy based on the training set is usually too high.
  - This is due to model overfitting (random noise is incorporated in the model).
- Validation provides a means to get an estimate of the accuracy for a data set that has not been included in the model construction.
- The idea is that this estimate holds true for any ’new’ data as well, i.e. the scoring set.
- Ultimately, if the test and scoring sets stem from the same population, the accuracy estimate for the testing set can then be generalized to the scoring set.
  - This estimate acts as a justification for applying the results (e.g. a decision tree) in real life, to achieve the business goals.
- Model validation is easy and straghtforward. It should always be done.
  - The validation aspect is incorporated in the construction of random forests.