UNIVERSITY OF PORTSMOUTH

Python for Data Analysis

Modeling in Python – Decision Tree and Random Forest (TB2 - Week 2)

Atefeh Khazaei

atefeh.khazaei@port.ac.uk

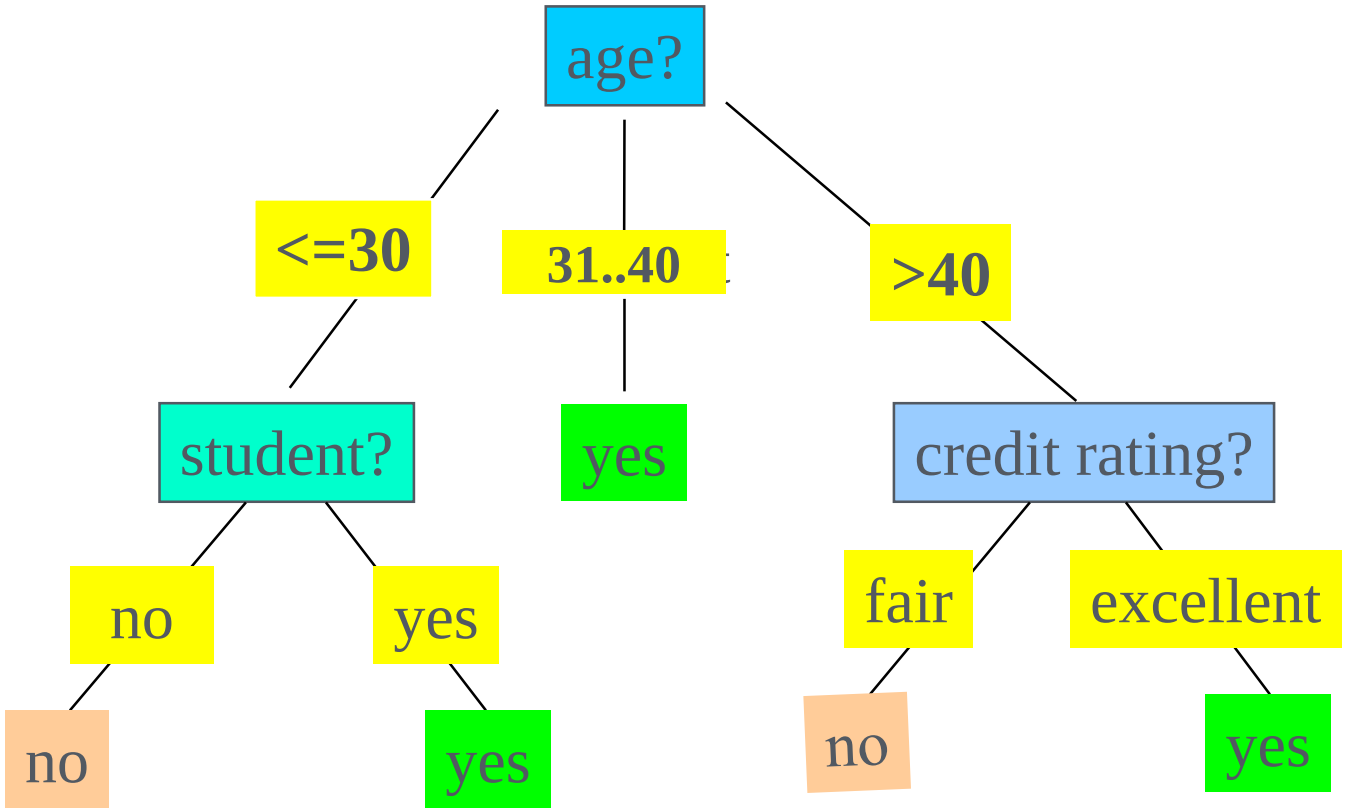
What we will learn this week?

- Modelling Algorithms
  - Decision Tree
  - Random Forest

Decision Tree

- Decision Tree is one of the useful supervised learning algorithms.
  - In supervised learning the data is already labelled and you are aware which target you want to predict for new samples.
- Advantage of DT:
  - Being interpretability
    - Trees can be visualised
- Disadvantage of DT:
  - DT can create biased trees if some classes are imbalanced.
  - It is suggested that balance the dataset prior to fitting with decision tree.

Decision Tree (cont.)



age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Internal nodes (non-leaf nodes) denote a test on an attribute.
Branches represent outcomes of tests.
Leaf nodes (terminal nodes) hold class labels.
Root node is the topmost node.

Decision Tree (cont.)

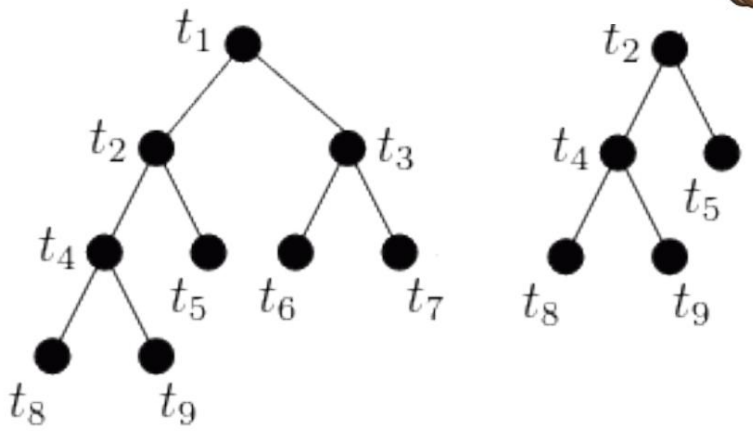
- Basic Decision Tree algorithm is a **greedy** algorithm.
- Tree is constructed in a **top-down recursive divide-and-conquer manner**.
- Decision Tree Algorithm main steps are:
  - At start, all the training tuples are at the root.
  - Attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain, Gain Ratio, and GiniIndex).
  - Tuples are partitioned recursively based on selected attributes.
  - When an stopping condition is met, one leaf is constructed

Decision Tree (cont.)

- Conditions for **stopping** partitioning:
  1. All samples for a given node belong to the same class.
  2. There are no remaining attributes for further partitioning – majority voting is employed.
  3. There are no samples left – majority voting on the parent's samples is employed.

Overfitting and Tree Pruning

- **Overfitting:** An induced tree may overfit the training data:
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- **Pruning:** To make the model simpler to reduce the chance of overfitting.
- **Principle of Occam's razor**: given two explanations for something, the explanation most likely to be correct is the simplest one.



sklearn.tree.DecisionTreeClassifier()

```
from sklearn import tree
from sklearn import metrics
from sklearn.model_selection import train_test_split

# Training data features, skip the first column 'Survived'
train_features = train_data[:, 1:]

# 'Survived' column values
train_target = train_data[:, 0]

# Split 80-20 train vs test data
train_x, test_x, train_y, test_y = train_test_split(train_features,
                                                    train_target,
                                                    test_size=0.20,
                                                    random_state=0)

clf = tree.DecisionTreeClassifier() #I will keep default values for this
clf = clf.fit(train_x, train_y)
predict_y = clf.predict(test_x)

from sklearn.metrics import accuracy_score
print ("Accuracy = %.2f" % (accuracy_score(test_y, predict_y)))
```

sklearn.tree.DecisionTreeClassifier()

- You can find more details in the following link:
  - https://scikit-learn.org/stable/modules/tree.html
  - https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
- About:
  - Decision Tree Algorithm
  - DecisionTreeClassifier() parameters
  - plot_tree(): to plot the obtained tree

Wisdom of the crowd

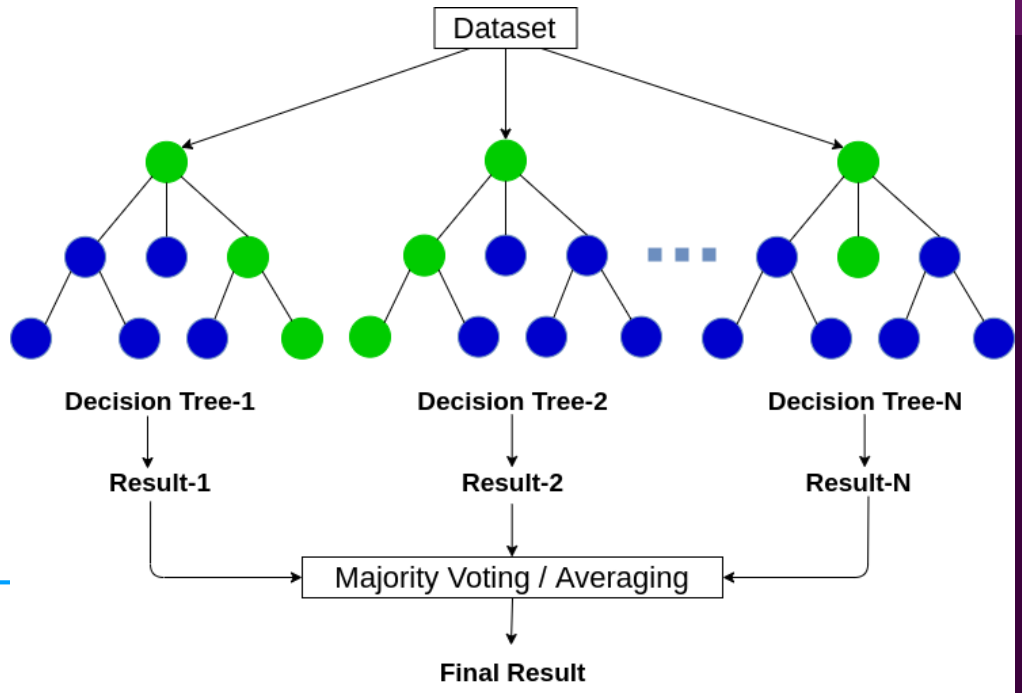
- The wisdom of the crowd is the collective opinion of a group of individuals rather than that of a single expert.
- An explanation for this phenomenon is that there is idiosyncratic noise associated with each individual judgment, and taking the average over a large number of responses will go some way toward removing the effect of this noise.
- Among the current applications of this phenomenon can be mentioned to social information sites:
  - Quora, Wikipedia, Yahoo!

Ensemble learning

- Ensemble Learning is similar to wisdom of the crowd.
- Ensemble model improves **accuracy** and **robustness** over single model methods.
- Popular methods:
  - Bagging
    - **Random Forest** is a famous example of the bagging method
  - Boosting
  - Stacking

Random forest

- Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.
- Random decision forests correct for decision trees' habit of overfitting to their training set.
- Random forests generally outperform DTs.



sklearn.ensemble.RandomForestClassifier()

```
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split
```

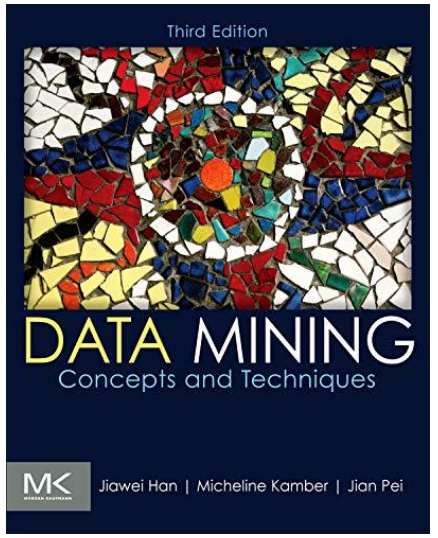
```
# Training data features, skip the first column 'Survived'
train_features = train_data[:, 1:]

# 'Survived' column values
train_target = train_data[:, 0]

# Split 80-20 train vs test data
train_x, test_x, train_y, test_y = train_test_split(train_features,
                                                    train_target,
                                                    test_size=0.20,
                                                    random_state=0)

clf = RandomForestClassifier(n_estimators=100)

clf = clf.fit(train_x, train_y)
predict_y = clf.predict(test_x)

from sklearn.metrics import accuracy_score
print ("Accuracy = %.2f" % (accuracy_score(test_y, predict_y)))
```

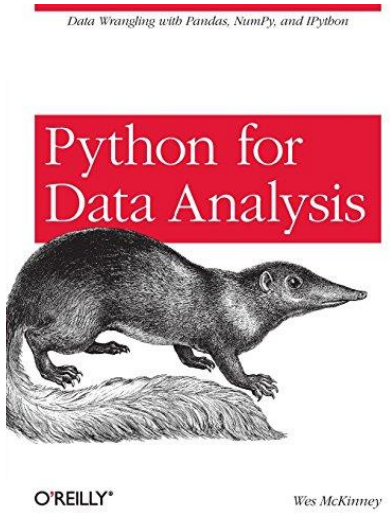
sklearn.ensemble.RandomForestClassifier()

- You can find more details in the following link:
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier
- About:
  - RandomForestClassifier() parameters
  - Some Examples

References & More Resources

- References:
  - McKinney, Wes. *Python for data analysis: Data wrangling with Pandas, NumPy, and Ipython*, O'Reilly Media, Inc., 2012.
  - Han, Jiawei, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.





Practical Session

- Revise the Titanic Case Study (Last session of TB1) and build some **Decision Trees** and **Random Forest** models for Titanic. Try **different parameters** for these models and **compare them together**.