CS342 Machine Learning: Lab #3
k-NN and Decision Tree Learning

Labs on Week 3 of Term 2

Week 17

Instructor:
Dr Theo Damoulas (T.Damoulas@warwick.ac.uk)

Tutors:
Helen McKay (H.McKay@warwick.ac.uk),
Joe Meagher (J.Meagher@warwick.ac.uk)
Karla Monterrubio-Gomez (K.Monterrubio-Gomez@warwick.ac.uk),
Jevgenij Gamper (J.Gamper@warwick.ac.uk)

In the third Lab we will explore the use and implementation of k-NN and the ID3 Decision Tree. Refer to the module slides and T. Mitchell's book for the supporting material.

1 Machine Learning Component

The material here builds on lectures 4-6. We will be providing the Python (unoptimised) "solutions" a week after each Lab.

1.1 k-NN

Go to the UCI Machine Learning Repository and download the Diabetes dataset: `http://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes`. Our goal is to predict if female patients will test positive for diabetes given 8 attributes such as age, plasma glucose concentration, etc. For details on the dataset click on the "Data Set Description":
`http://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes/pima-indians-diabetes.names`.

→ Import the dataset into a pandas data frame and standardise the attributes (subtract the mean and divide by the standard deviation, see Lab1).

→ Assume one class is the"negative" class and the second class is the "positive" class. Write a function that would take as input your predicted targets and the true targets, and estimate the *Accuracy* of your classifier defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \tag{1}$$

where, TP = True Positives (model predicts positive and true value is positive), FP = False Positives (model predicts positive and true value is negative), TN = True Negatives (model predicts negative and true value is negative), and FN = False Negatives (model predicts negative and true value is positive).

→ Perform k-NN classification using the scikit implementation (*sklearn.neighbors.KNeighborsClassifier*) for $k = 1, 3, 5$ and use 10-fold Cross-Validation (*sklearn.cross_validation*) and the measure of *Accuracy* to choose the best model.

Hint: You may find that the *KFold* function within *sklearn.cross_validation* is useful for keeping track of which instances are within each fold when perfoming k-fold cross validation.

1.2 ID3 Decision Tree

In Figure 1 you can find the Tennis data from the example we used at Lecture 5 on describing the ID3 algorithm.

→ Write a function that computes the Entropy of a set S with N_{pos} positive observations and N_{neg} negative observations.

The Entropy of a set S is given by $\text{Entropy}(S) = -\sum_{c=1}^{C} p_c \log_2 p_c$, where C is the number of classes (2 in our case since we have a positive and a negative class) and p_c is the frequency of class c in the set S.

→ Write a function that takes as input a set S of observations and an attribute A from these observations, and calculates the Information Gain, denoted as $\text{Gain}(S, A)$, as if we were to split on that attribute in the context of the ID3 decision tree algorithm.

The Information Gain is defined as: $\text{Gain}(S, A) = \text{Entropy}(S) - \sum_i \frac{|S_{u_i}|}{|S|}\text{Entropy}(S_{u_i})$

where S_{u_i} is the subset of observations from S whose attribute A is equal to the i^{th} possible value that A takes. For example, the attribute Temperature takes three possible values as either Hot, Mild, or Cool. So $S_{u_1} = S_{\text{Hot}}$ is all the observations that have the Hot value in the Temperature attribute. The $\sum_i$ summation over i indicates we are summing over all the possible values u_i of attribute A. Finally, $|S|$ and $|S_{u_i}|$ denote the cardinality of the set S and subset S_{u_i} respectively (cardinality = number of elements of the set). So $|S| = 14$ in our example dataset.

→ Estimate the Information Gain of all the attributes. Which one would you choose for the root node of your decision tree?

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Figure 1: Toy dataset for ID3 that can be downloaded from the module website. The task we are given is to predict if we should Play Tennis or not given the rest of the attributes. We have 14 training examples (observations and their targets) to use.

1.3 Decision Tree induction

Implement the ID3 Decision Tree from the pseudocode (recursive algorithm) below and induce/learn the tree from the data in Figure 1.

Algorithm 1 ID3 Algorithm

ID3 (Observations, Targets, Attributes)

if all Observations are class $+1$ **then**
  return single-node tree Root with label $+1$
if all Observations are class -1 **then**
  return single-node tree Root with label -1
if Attributes is empty **then**
  return the single-node tree Root with label the most common value in Targets
else
  Begin
  A$\leftarrow$ best attribute from Attributes (highest Information Gain)
  The decision attribute for Root $\leftarrow A$
  for each possible vale u_i of A: **do**
    Add a new tree branch below Root for $A = u_i$
    $S_{u_i} \leftarrow$ Subset of Observations with $A = u_i$
    if S_{u_i} is empty **then**
      Add leaf node with label the most common value in Targets
    else
      add below branch ID3(S_{u_i}, Targets, Attributes - {A}))
  End
return Root