Department of Computer Engineering
University of Peradeniya

Data Mining and Machine Learning
Lab 05

August 2, 2017

1 Introduction to Decision Trees

Decision Tree is a non-parametric supervised learning method that can be used for both classification and regression. Decision trees essentially encode a set of if-then-else rules which can be used to predict target variable given data features. These if-then-else rules are formed using the training dataset with the aim to satisfy as many training data instances as possible. Various decision tree learning algorithms have been developed and they work best in different situations. An advantage of decision trees is that they can model any type of function for classification or regression which other techniques cannot. But a decision tree is highly prone to overfitting and bias towards training data. So, decision trees are used for very large datasets which are assumed to represent the ground truth well. Additionally, certain tree pruning algorithms are also used to tackle overfitting.

2 Classification

Classification is a data mining technique that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. For example, a classification model could be used to identify loan applicants as low, medium, or high credit risks. `DecisionTreeClassifier` is a class capable of performing multi-class classification on a dataset.

1. As with the other classifiers, DecisionTreeClassifier takes as input two arrays: an array X, sparse or dense, of size [n-samples, n-features] holding the training samples, and an array Y of integer values, size [n-samples], holding the class labels for the training samples.

   ```python
   from sklearn import tree
   X = [[140, 1],[130, 1], [150, 0], [170, 0]]
   Y = [1, 2, 3, 4]
   clf = tree.DecisionTreeClassifier()
   clf = clf.fit(X, Y)
   ```

2. After being fitted, the model can then be used to predict the class of samples:

   ```python
   clf.predict([[170,1]])
   ```

3. Alternatively, the probability of each class can be predicted, which is the fraction of training samples of the same class in a leaf:

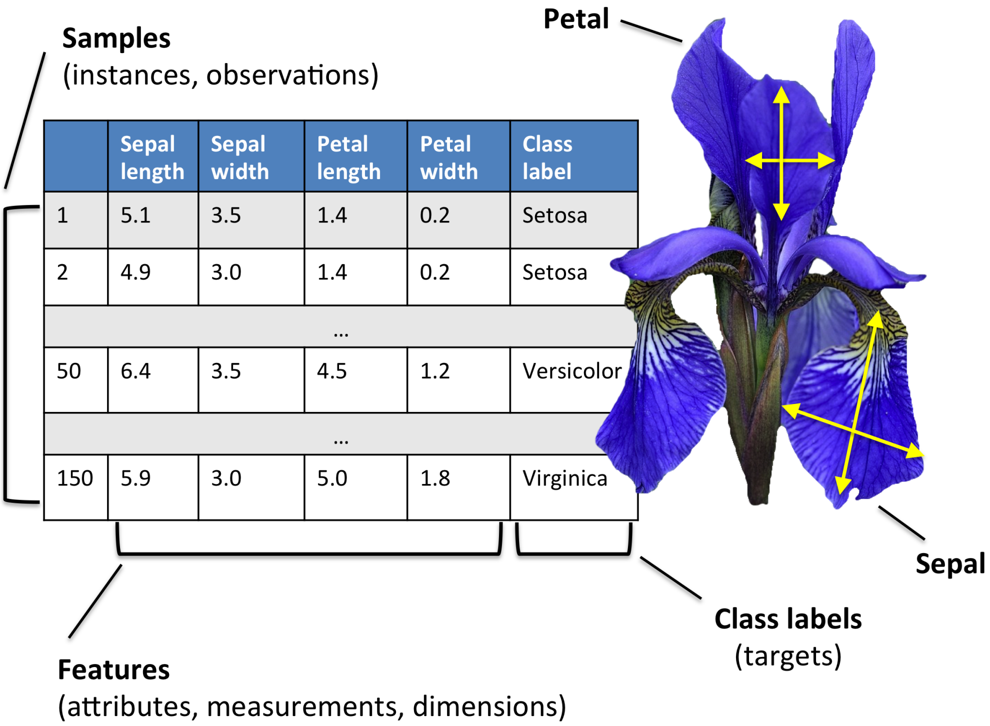
```
clf.predict_proba([[170,1]])
```

4. DecisionTreeClassifier is capable of both binary classification (e.g. class labels as [-1, 1]) and multiclass classification (e.g. class labels as [0, ..., K-1], here K is the number of classes).

3 Try Out

Familiarize decision tree building with scikit-learn

Iris dataset is a famous dataset that contains 150 iris flowers of three different species: Iris-Setosa, Iris-Versicolor and Iris-Virginica and each entry consists of four features: sepal length, sepal width, petal width and petal length which are illustrated in Fig. 1.



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
…					
50	6.4	3.5	4.5	1.2	Versicolor
…					
150	5.9	3.0	5.0	1.8	Virginica

Samples
(instances, observations)
Petal
Sepal
Features
(attributes, measurements, dimensions)
Class labels
(targets)

Figure 1: A sample view of Iris data set

1. Using the Iris dataset, we can construct a tree as follows:

```
from sklearn.datasets import load_iris
from sklearn import tree
iris = load_iris()
clf = tree.DecisionTreeClassifier()
clf = clf.fit(iris.data, iris.target)
```

2. Alternatively, if we have Python module pydotplus installed, we can generate a PDF file (or any other supported file type) directly in Python.The $export - graphviz$ exporter also supports a variety of aesthetic options, including coloring nodes by their class (or value for regression) and using explicit variable and class names if desired

```
import pydotplus
dot_data = tree.export_graphviz(clf, out_file=None)
graph = pydotplus.graph_from_dot_data(dot_data)
graph.write_pdf("iris.pdf")
```

3. IPython notebooks can also render these plots inline using the Image() function.Here,The IPython Notebook is now known as the Jupyter Notebook. It is an interactive computational environment, in which you can combine code execution, rich text, mathematics, plots and rich media. For more details on the Jupyter Notebook, please see the Jupyter website.

```
from IPython.display import Image

dot_data = tree.export_graphviz(clf, out_file=None,
                         feature_names=iris.feature_names,
                         class_names=iris.target_names,
                         filled=True, rounded=True,
                         special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data)
Image(graph.create_png())
graph.write_png("iris.png")
```

4. The Python Imaging Library (PIL) adds image processing capabilities to your Python interpreter. This library supports many file formats, and provides powerful image processing and graphics capabilities.

```
from PIL import Image, ImageDraw
image = Image.open("iris.png")
image.show()
```

5. After being fitted, the model can then be used to predict the class of samples:

```
clf.predict(iris.data[:1, :])
```

6. Alternatively, the probability of each class can be predicted, which is the fraction of training samples of the same class in a leaf:

```
clf.predict_proba(iris.data[:1, :])
```

4 Lab Exercise

Creating a decision tree and predicting class labels for test data

1. Load breast-cancer dataset by loading the data set as given and handle the missing values too. First,import the necessary libraries that you require for the analysis.

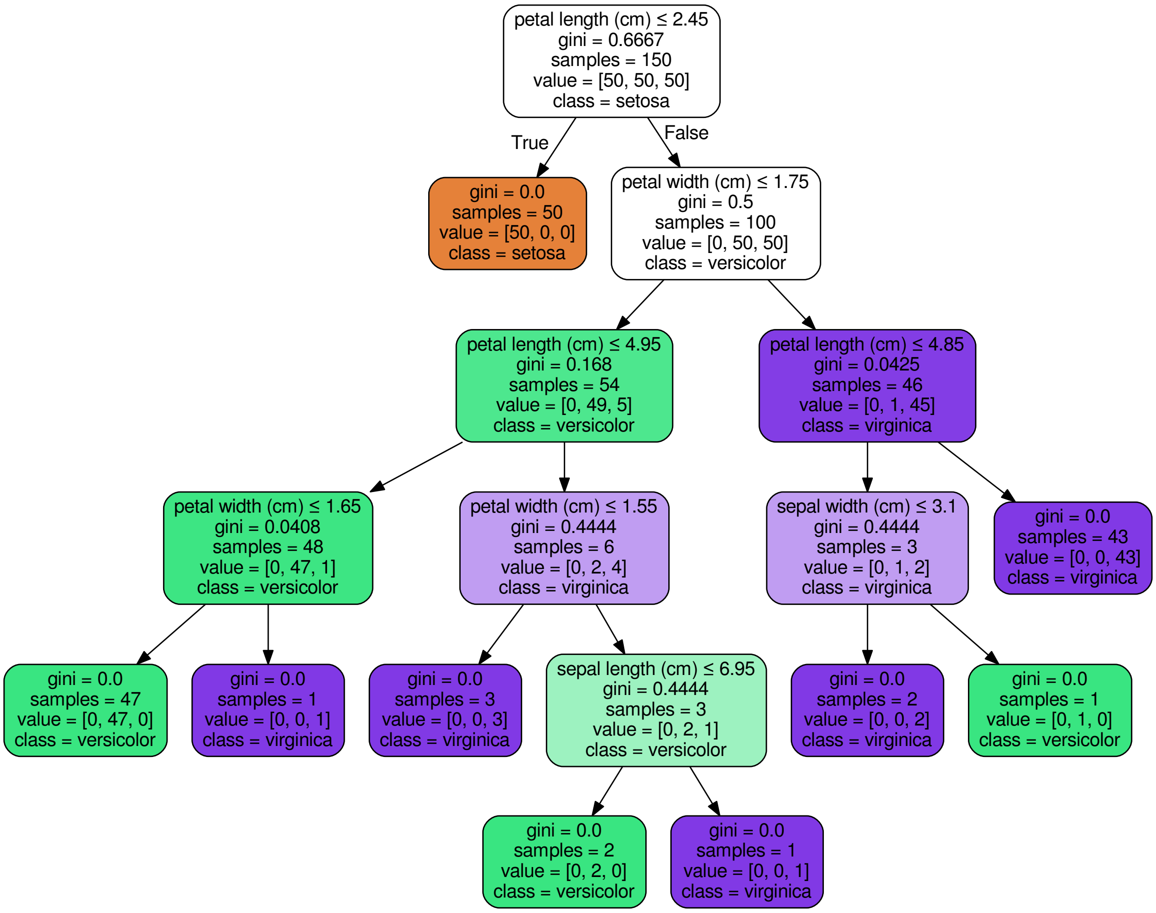


Figure 2: Decision tree for Iris data set

2. Convert the response variable (class attribute) **breastcancerper100th** to binary such that breast cancer cases reported that less than or equal to 20 will be coded as 0 and cases greater than 20 will be coded as 1. Convert the explanatory variables, i.e., the other attributes, to categorical (nominal attributes) appropriately.

3. Train the decision tree classifier by splitting the data as 2/3 for training and 1/3 for testing.

4. Once you create the training and testing data sets, initialize the decision tree classifier from sklearn. Then, build the model on training data set.

5. Calculate a prediction accuracy for test data set.

```
sklearn.metrics.accuracy_score(y_true , y_pred)
```

5 Submission

Submit a single .py file as [12|13]xxxlab05.py where xxx is your registration number. Add answers for questions 3, 4 and 5 as comments in the same file.

6 Important

Make sure that you have the basic understanding of decision trees. This lab is really important for successive labs. If you do not understand any concepts, make sure you get some help from instructors.

7 Deadline

August 04, 23:59:59 GMT+5:30.