Lab02: Decision Tree with scikit-learn

In this assignment, you are going to *build a decision tree on the UCI Poker Hand Data Set*, with the support from *scikit-learn library*.

About the Poker Hand Data Set from UCI Machine Learning Repository

There are ***1,025,010 records*** in the data set. Each of which is an example of a hand consisting of five playing cards drawn from a standard deck of 52. Each card is described using two attributes (suit and rank), for a total of ***10 predictive attributes***. There is ***one Class attribute*** that describes the "Poker Hand". The order of cards is important, which is why there are 480 possible Royal Flush hands as compared to 4 (one for each suit - explained in the description).

You can download the data files from here.

Assignment requirements

You are asked to write a Python program and use scikit-learn functions to fulfill the following tasks. Although there is no strict rule on how to organize the code, each task should be noted carefully and it must reflect all the requirements mentioned.

Preparing the data sets

This task prepares the training sets and test sets for the incoming experiments.

Starting with the data provided in this folder, you need to *manually merge* the two files, `poker-hand-training-true.data` and `poker-hand-testing.data`, into a single CSV file, named `poker-hand-data.csv`.

Then, you prepare the following four subsets from the merged data in `poker-hand-data.csv`:

- `feature_train`: a set of training examples, each of which is a tuple of 10 attribute values (target attribute excluded).
- `label_train`: a set of labels corresponding to the examples in `feature_train`.
- `feature_test`: a set of test examples, it is of similar structure to `feature_train`
- `label_test`: a set of labels corresponding to the examples in `feature_test`.

You need to shuffle the data before splitting and split it in a stratified fashion. Other parameters (if there is any) are left by default.

There will be experiments on training sets and test sets of different proportions, including `(train/test) 40/60, 60/40, 80/20,` and `90/10`, and thus you need 16 subsets.

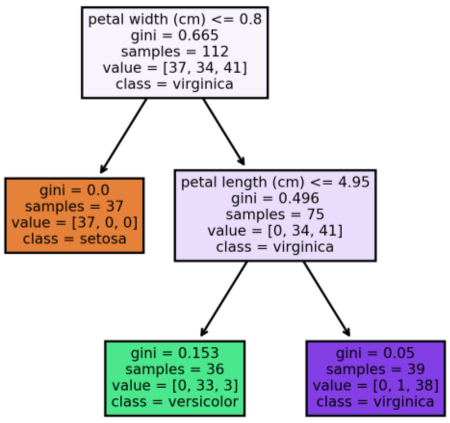
Visualize the distributions of classes in all the data sets (the original set, training set, and test set) of all proportions to show that you have prepared them appropriately.

Building the decision tree classifiers

This task conducts experiments on the designated train/test proportions listed above.

You need to fit an instance of `sklearn.tree.DecisionTreeClassifier` (with ***information gain***) to each training set and visualize the resulting decision tree using `graphviz`.

The aside figure gives an example of a decision tree built on the Iris data set (3 classes).



Evaluating the decision tree classifiers

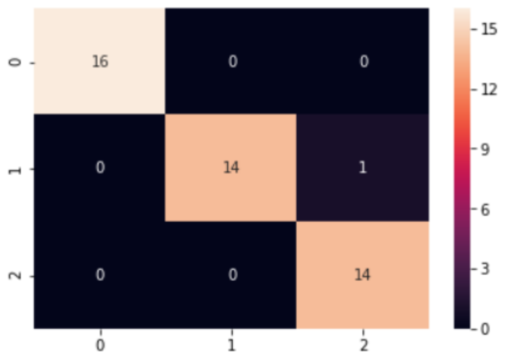
For each of the above decision tree classifiers, predict the examples in the *corresponding test set*, and make a report using `classification_report` and `confusion_matrix`.

The following figure gives an example of classification report and confusion matrix for a classifier on the Iris data set (3 classes).

```
             precision    recall  f1-score   support

     setosa       1.00      1.00      1.00        16
 versicolor       1.00      0.93      0.97        15
  virginica       0.93      1.00      0.97        14

avg / total       0.98      0.98      0.98        45
```



How do you interpret the classification report and the confusion matrix? From that, make your own comments on the performances of those decision tree classifiers.

The depth and accuracy of a decision tree

This task works on the *80/20 training set and test set*. You need to consider how the decision tree's depth affects the classification accuracy.

You can specify the maximum depth of a decision tree by varying the parameter `max_depth` of `sklearn.tree.DecisionTreeClassifier`.

You need to try the following values for parameter `max_depth`: `None, 2, 3, 4, 5, 6,` and `7`. And then,

- Provide the decision tree drawn by `graphviz` for each `max_depth` value
- Report to the following table the `accuracy_score` (on the test set) of the decision tree classifier when changing the value of parameter `max_depth`.

max_depth	None	2	3	4	5	6	7
Accuracy							

- Make your own comment on the above statistics.

References

[1] Scikit-learn decision trees: https://scikit-learn.org/stable/modules/tree.html

[2] Analysis and classification of Mushrooms: https://www.kaggle.com/haimfeld87/analysis-and-classification-of-mushrooms

Grading

No.	Specifications	Scores (%)
1	Preparing the data sets	30
2	Building the decision tree classifiers	20
3	Evaluating the decision tree classifiers	
	Classification report and confusion matrix	10
	Comments	10
4	The depth and accuracy of a decision tree	
	Trees, tables, and charts	20
	Comments	10
Total		**100**

Notice

- This is an **INDIVIDUAL** assignment.
- You must use **Python** language and present the code in a single **ipynb file**.
- Write down your report on a **PDF File.**
- All the required visualizations must be presented in the ipynb file, while statistical results and comments are presented in the report.
- A program with syntax/runtime error(s) will not be accepted.