Chaitanya M. Nawale
BECOB215

PAGE NO.
DATE :

ML Assignment No. 4

* Title :- To implement decision tree classifier technique.

* Objective :- To learn how to apply decision tree algorithm for given dataset using SK-learn.

* Outcomes :- Ability to apply decision tree classification technique to solve the given problem.

* Software requirements :- python3, jupyter notebook, scikit-learn.

* Problem Statement :- A dataset collected in a cosmetics shop showing details of customers and whether or not they responded to a special offer to buy a new lip-stick is shown in table below. Use this dataset to build a decision tree, with buys as the target variable, to help in buying lip-sticks in the further future find the root node of decision tree. According to the decision tree you have made from previous trainging dataset, what is the decision for the test data : [Age < 21, Income = Low, Gender = Female, Marital Status = Married] ?

Scanned by TapScanner

ID	Age	Income	Gender	Marital Status	Buys
1	<21	High	Male	Single	No
2	<21	High	Male	Married	No
3	21-35	High	Male	Single	Yes
4	>35	Medium	Male	Single	Yes
5	>35	Low	Female	Single	Yes
6	>35	Low	Female	Married	No
7	21-35	Low	Female	Married	Yes
8	<21	Medium	Male	Single	No
9	<21	Low	Female	Married	Yes
10	>35	Medium	Female	Single	Yes
11	<21	Medium	Female	Married	Yes
12	21-35	Medium	Male	Married	Yes
13	21-35	High	Female	Single	Yes
14	>35	Medium	Male	Married	No

* Theory :- • Decision tree is a graph to represent choices and their results in form of a tree. The nodes in the graph represents the decision rules or conditions. It is mostly used in Machine Learning and Data Mining applications using Python.

As you might intuit from the name, decision tree learners, build a model in the form of a tree structure. The model itself comprises a series of logical decisions, similar to a flowchart, with decision nodes that indicate a decision to be made on an attribute. These split into branches that indicate the decision's choices. The tree is terminated by leaf nodes (also known as terminal nodes) that denote the result of following combination of decisions.

- Binary Decision Trees :
  A binary decision tree is a structure based on a sequencial decision process. Starting from the root, a feature is evaluated and one of the two branches is selected. This procedure is repeated until a final leaf is reached, which normally represents the classification target we're looking for. Considering other algorithms, decision tree seems to be simpler in their dynamics; however, if the dataset is splittable while keeping an internal balance, the overall process is intuitive and rather fast in its predictions. Moreover, decision trees can work efficiently with unnormalized datasets because their internal structure is not influenced by the values assumed by each feature.

- Binary decisions
  Let's consider an input dataset X:
  $X = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\}$ where $\bar{x}_i \in \mathbb{R}^m$
  Every vector is made up of m features, so each of them can be a good candidate to create a node based on the (feature, threshold) tuple:
  According to the feature and the threshold,

Scanned by TapScanner

the structure of the tree will change. Intuitively, we should pick the feature that best separates our data in other words, a perfect separating feature will be present only in a node and the two subsequent branches won't be based on it anymore. In real problems, this is often impossible, so it's necessary to find the feature that minimizes the number of decision steps.

• Impurity measures:
To define the most used impurity measures, we need to consider the total number of target classes:
$y = \{Y_1, Y_2, \dots, Y_n\}$ where $y_n \in (0,1,2,\dots$

In a certain node j, we can define the probability $p(i|j)$ where i is an index[1,n] associated with each class. In other words, according to a frequentist approach, this value is the ratio between the number of samples belonging to class i and the total number of samples belonging to the selected node.

• Decision tree classification with scikit-learn:-
Spliting dataset into train and test dataset:

```
>>> X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.3, random_state=100)
```

1) Decision Tree Classifier with criterion Gini Index:

BECOB215

PAGE NO.
DATE :

```
>>> clf_gini = DecisionTreeClassifier( criterion=
"gini", random_state=100, max_depth=3,
~~mean~~ min_samples_leaf =5)
```

Gini Index prediction on test dataset:

```
>>> y_pred = clf_gini.predict (X_test)
```

Decision Tree model with Gini Index accuracy:

```
>>> print "Accuracy is ", accuracy_score(y_test,
y_pred ) * 100
```

2) Decision Tree Classifier with criterion information gain:

```
>>> clf_entropy = DecisionTreeClassifier (
criterion ="entropy", random_state = 100, max
_depth = 3, min_samples_leaf = 5)
>>> clf_entropy.fit( X_train , y_train)
```

Information Gain Decision tree prediction on test dataset:

```
>>> y_pred_en= clf_entropy. predict (X_test)
```

Decision Tree classifier with Information gain accuracy:

```
>>> print "Accuracy is", accuracy_score(y_test,y_pred_en)*100
```

* Conclusion :- Thus we have successfully implemented decision tree algorithm for given datasets using Sk-Learn.

Scanned by TapScanner