Student's Name

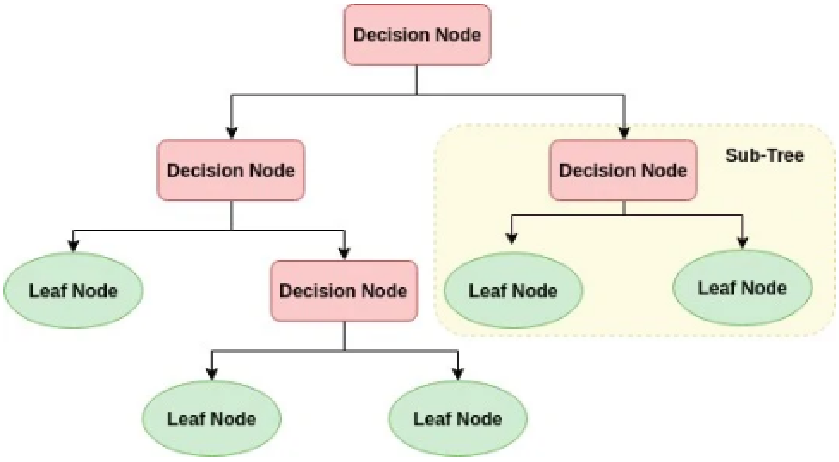
Instructor's Name

Course

Date

Decision Trees

A decision tree is a white box type of machine learning algorithm. It shares its internal decision-making logic, unlike black-box algorithms like Neural Network. It is a flowchart-like structure (Navlani). The tree consists of an internal node representing a feature, a branch representing a decision rule, and leaf nodes representing the outcomes. The topmost node in this structure is referred to as the root node (Kulkarni). It partitions based on the feature values. A decision tree helps in decision-making through visualization. Decision trees are easy to understand and interpret as they mimic human-level thinking. They are also distribution-free, independent of probability distribution assumptions (Navlani). The figure below shows a decision tree structure.



A decision tree algorithm can be used for both classification and regression problems. It follows three primary stages to solve these problems. These stages are described below.

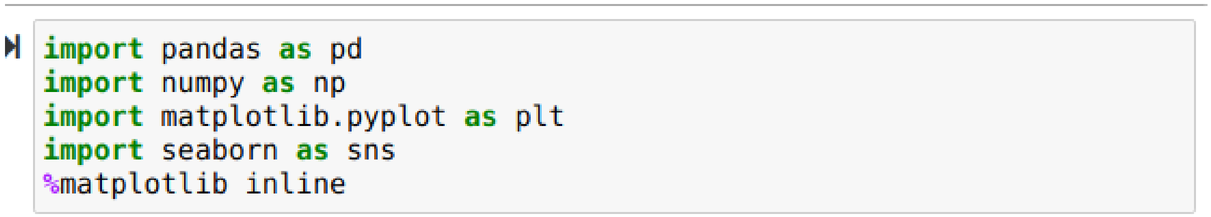
1. Best attribute selection – At this stage, Attribute Selection Measures (ASM) are used to split the provided records. Attribute selection measure refers to a heuristic used for selecting and splitting. It partitions datasets into the best possible categories. It includes splitting rules as it determines breakpoints for tuples on a particular node. Common attribute selection measures include Gini Index, Gain Ratio, and Information Gain.

2. Convert the selected attribute into a decision node. In this phase, it divides the dataset into smaller subsets.

3. Tree building – In this phase, the process is repeated recursively for each child until either all tuples belong to the same attribute value, there are no more remaining attributes, or there are no more instances.

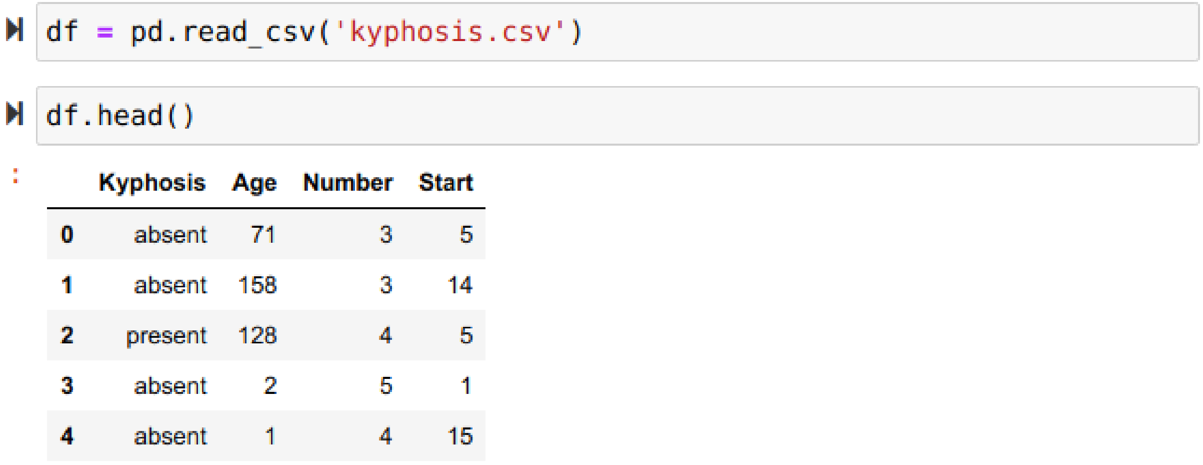
The following figures show a decision tree implementation using Scikit-learn.

1. Importing the required libraries



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

2. Reading and viewing the datasets

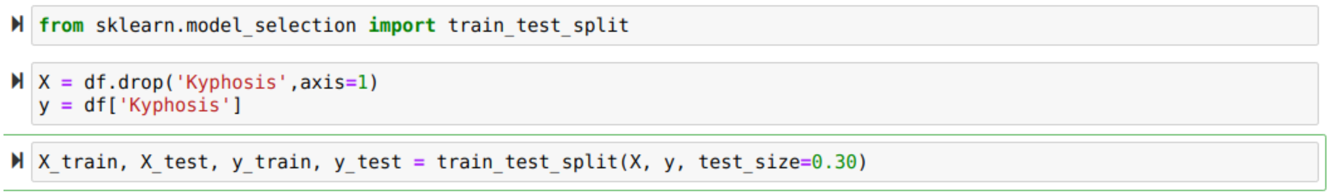


```
df = pd.read_csv('kyphosis.csv')
```

```
df.head()
```

	Kyphosis	Age	Number	Start
0	absent	71	3	5
1	absent	158	3	14
2	present	128	4	5
3	absent	2	5	1
4	absent	1	4	15

3. Splitting the dataset into train and test datasets

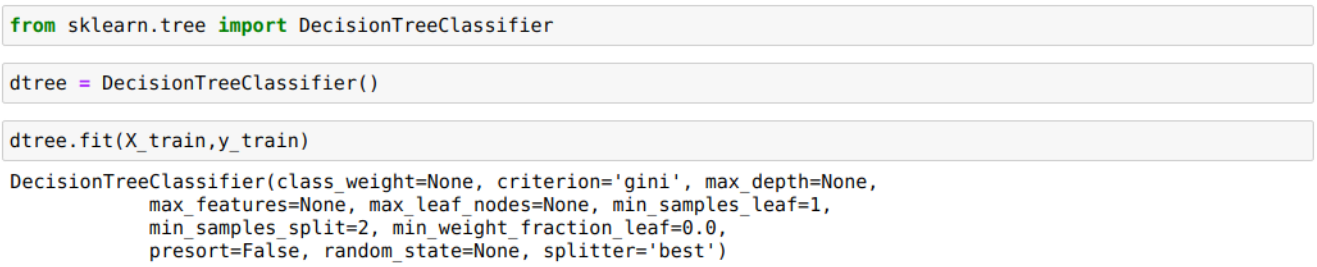


```
from sklearn.model_selection import train_test_split
```

```
X = df.drop('Kyphosis',axis=1)
y = df['Kyphosis']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
```

4. Implementing the decision tree algorithm



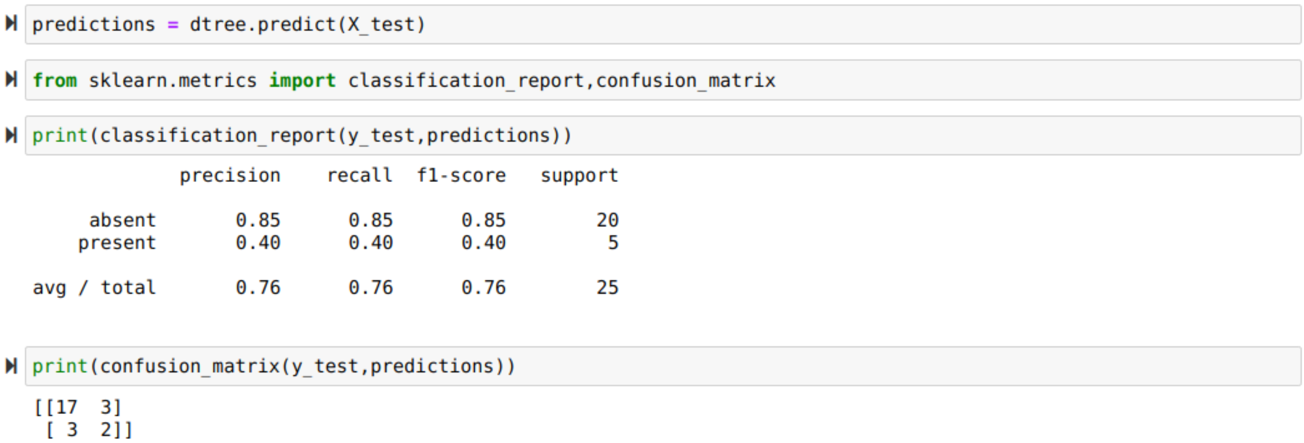
```
from sklearn.tree import DecisionTreeClassifier
```

```
dtree = DecisionTreeClassifier()
```

```
dtree.fit(X_train,y_train)
```

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            presort=False, random_state=None, splitter='best')
```

5. Model Prediction and Evaluation



```
predictions = dtree.predict(X_test)
```

```
from sklearn.metrics import classification_report,confusion_matrix
```

```
print(classification_report(y_test,predictions))
```

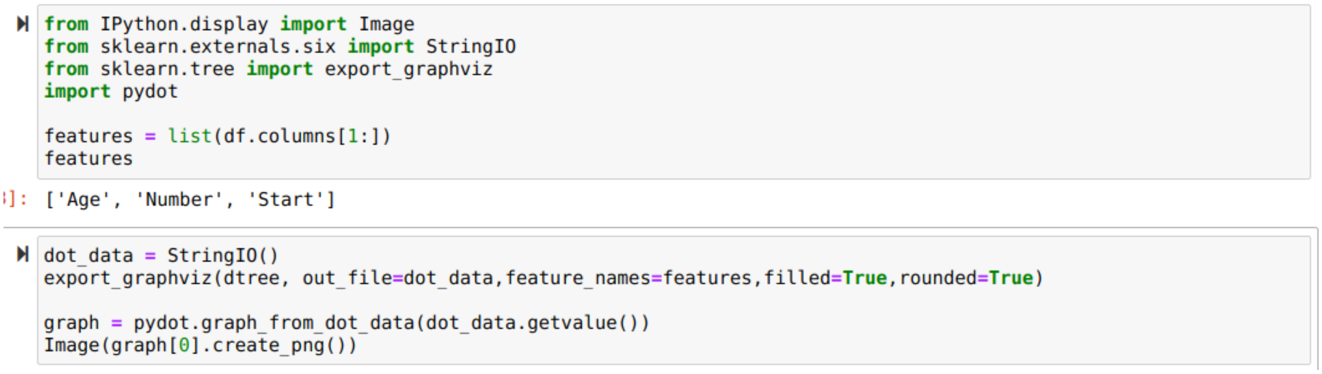
```
             precision    recall  f1-score   support

     absent       0.85      0.85      0.85        20
    present       0.40      0.40      0.40         5

avg / total       0.76      0.76      0.76        25
```

```
print(confusion_matrix(y_test,predictions))
```

```
[[17  3]
 [ 3  2]]
```

6. Tree Visualization

We will use the pydot library to visualize the decision tree.

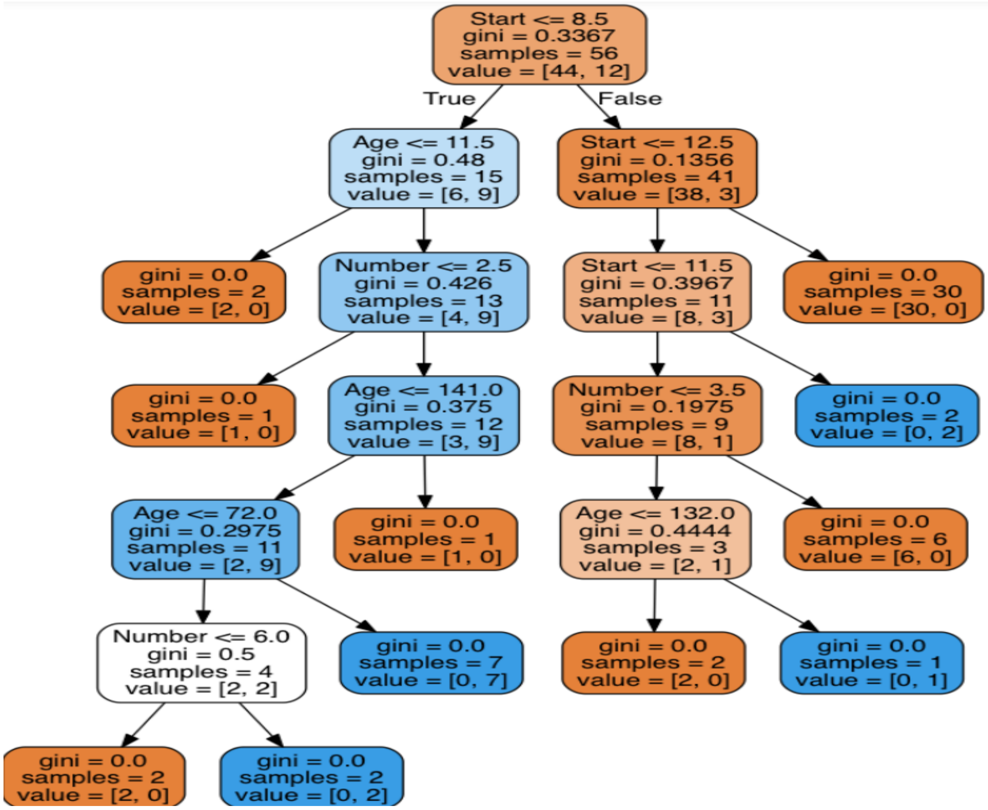


```
from IPython.display import Image
from sklearn.externals.six import StringIO
from sklearn.tree import export_graphviz
import pydot

features = list(df.columns[1:])
features
```

```
]: ['Age', 'Number', 'Start']
```

```
dot_data = StringIO()
export_graphviz(dtree, out_file=dot_data,feature_names=features,filled=True,rounded=True)

graph = pydot.graph_from_dot_data(dot_data.getvalue())
Image(graph[0].create_png())
```

The decision tree from above is shown below.



Works Cited

Kulkarni, Mayur. “Decision Trees for Classification: A Machine Learning Algorithm.” *Xoriant,* 7 Sept. 2017, www.xoriant.com/blog/product-engineering/decision-trees-machine-learning-algorithm.html. Accessed 10 Apr. 2022.

Navlani, Avinash. “Decision Tree Classification in Python Tutorial.” *datacamp,* 28 Dec. 2018, www.datacamp.com/community/tutorials/decision-tree-classification-python. Accessed 10 Apr. 2022.