CS 6220 Data Mining — Assignment 8 — Decision Trees — Samuel Steiner

In [ ]:

```python
# Import packages
import numpy as np
import graphviz
from sklearn.datasets import load_boston
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, export_graphviz
```

In [ ]:

```python
# load data
X, y = load_boston(return_X_y=True)

# Split the range of target values into three equal parts - low, mid, and high.
# Reassign the target values into into three categorical values 0, 1, and 2, representing low, mid and high range of values, respectively
diff = (y.max() - y.min())/3
d = y.min()
split = []
for _ in range(3):
    split.append([d, d+diff])
    d += diff

def categorize_target(value):
    for idx, rangx in enumerate(split):
        if value >= rangx[0] and value <= rangx[1]:
            return idx

y_ = np.array(list(map(categorize_target, y)))
```

In [ ]:

```python
# 1. Split the dataset into 70% training set and 30% test set.
X_train, X_test, y_train, y_test = train_test_split(X, y_, test_size=0.3)
```

In [ ]:

```python
# 2. Using scikit-learn’s DecisionTreeClassifier, train a supervised learning model that can be used to generate predictions for your data.
# A reference to how you can do that can be found in the users manual at
# https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier.

dt_clf = DecisionTreeClassifier()
dt_clf.fit(X_train, y_train)
```

Out[ ]:

```
DecisionTreeClassifier()
```

In [ ]:

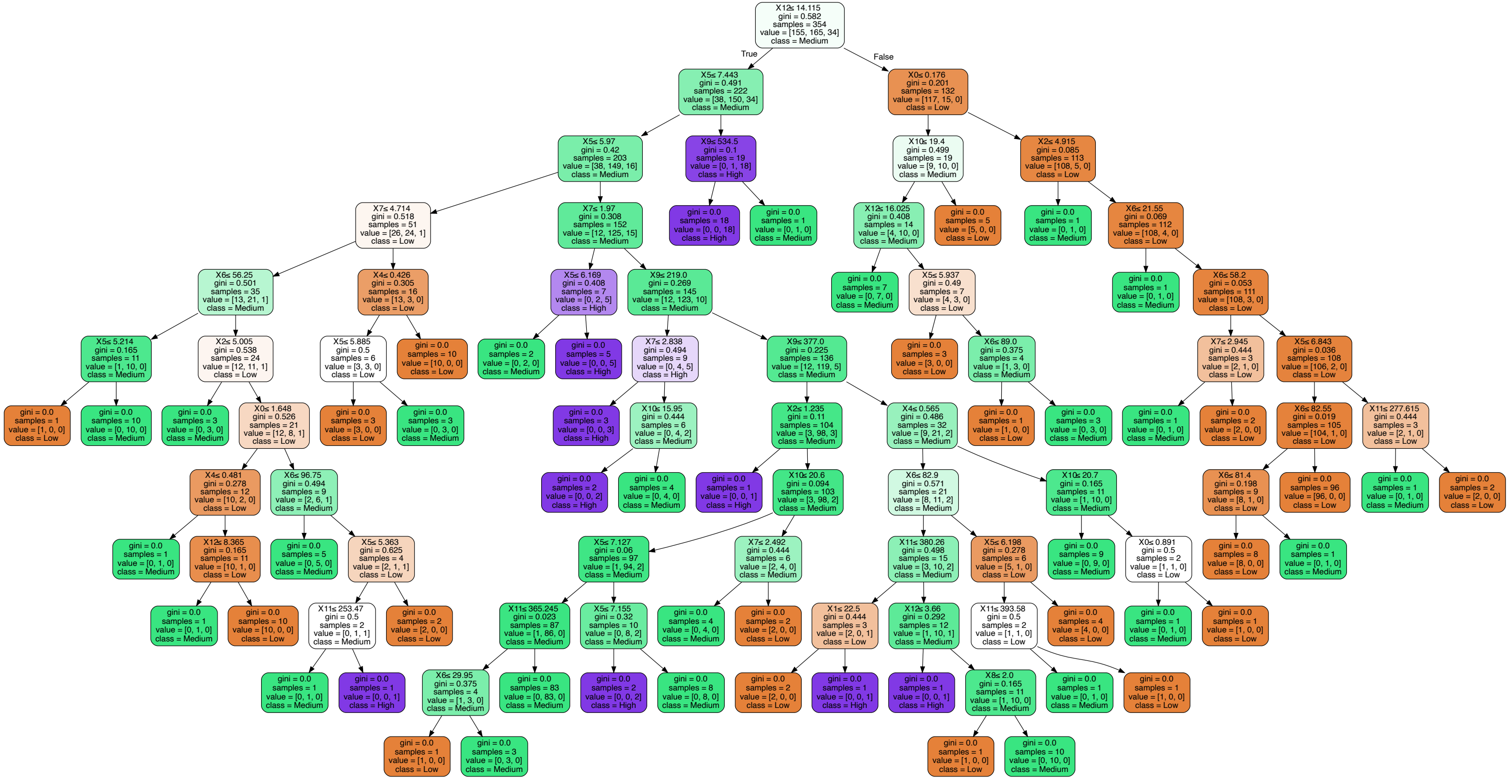
```python
# 3. Report the tree depth, number of leaves, feature importance, train score, and test score of the tree. Let the tree depth be Td.

td = dt_clf.get_depth()

def report_on_dt(dt_):
        print(f"""
        Tree Depth: {dt_.get_depth()}
        Number of Leaves: {dt_.get_n_leaves()}
        Feature importance: {' '.join([f'Feature {idx+1}: {val:.3f}' for idx, val in enumerate(dt_.feature_importances_)])}
        Train score: {dt_.score(X_train, y_train)}
        Test score: {dt_.score(X_test, y_test):.3f}""")

report_on_dt(dt_clf)
```

```
        Tree Depth: 11
        Number of Leaves: 53
        Feature importance: Feature 1: 0.057 Feature 2: 0.006 Feature 3: 0.027 Feature 4: 0.000 Feature 5: 0.025 Feature 6: 0.243 Feature 7: 0.073 Feature 8: 0.072 Feature 9: 0.009 Feature 10: 0.046 Feature 11: 0.041 Feature 12: 0.031 Feature 13: 0.370
        Train score: 1.0
        Test score: 0.816
```

In [ ]:

```python
# 4. Show the visual output of the decision tree.
dot_data = export_graphviz(dt_clf, class_names=['Low', 'Medium', 'High'],
                           filled=True, rounded=True,  special_characters=True)
graph = graphviz.Source(dot_data)
graph.render(f"boston_housing_{dt_clf.get_depth()}")
graph
```

Out[ ]:


X12≤ 14.115 gini = 0.582 samples = 354 value = [155, 165, 34] class = Medium
True
False
X5≤ 7.443 gini = 0.491 samples = 222 value = [38, 150, 34] class = Medium
X0≤ 0.176 gini = 0.201 samples = 132 value = [117, 15, 0] class = Low
X5≤ 5.97 gini = 0.42 samples = 203 value = [38, 149, 16] class = Medium
X9≤ 534.5 gini = 0.1 samples = 19 value = [0, 1, 18] class = High
X10≤ 19.4 gini = 0.499 samples = 19 value = [9, 10, 0] class = Medium
X2≤ 4.915 gini = 0.085 samples = 113 value = [108, 5, 0] class = Low
X7≤ 4.714 gini = 0.518 samples = 51 value = [26, 24, 1] class = Low
X7≤ 1.97 gini = 0.308 samples = 152 value = [12, 125, 15] class = Medium
gini = 0.0 samples = 18 value = [0, 0, 18] class = High
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
X12≤ 16.025 gini = 0.408 samples = 14 value = [4, 10, 0] class = Medium
gini = 0.0 samples = 5 value = [5, 0, 0] class = Low
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
X6≤ 21.55 gini = 0.069 samples = 112 value = [108, 4, 0] class = Low
X6≤ 56.25 gini = 0.501 samples = 35 value = [13, 21, 1] class = Medium
X4≤ 0.426 gini = 0.305 samples = 16 value = [13, 3, 0] class = Low
X5≤ 6.169 gini = 0.408 samples = 7 value = [0, 2, 5] class = High
X9≤ 219.0 gini = 0.269 samples = 145 value = [12, 123, 10] class = Medium
gini = 0.0 samples = 7 value = [0, 7, 0] class = Medium
X5≤ 5.937 gini = 0.49 samples = 7 value = [4, 3, 0] class = Low
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
X6≤ 58.2 gini = 0.053 samples = 111 value = [108, 3, 0] class = Low
X5≤ 5.214 gini = 0.165 samples = 11 value = [1, 10, 0] class = Medium
X2≤ 5.005 gini = 0.538 samples = 24 value = [12, 11, 1] class = Low
X5≤ 5.885 gini = 0.5 samples = 6 value = [3, 3, 0] class = Low
gini = 0.0 samples = 10 value = [10, 0, 0] class = Low
gini = 0.0 samples = 2 value = [0, 2, 0] class = Medium
gini = 0.0 samples = 5 value = [0, 0, 5] class = High
X7≤ 2.838 gini = 0.494 samples = 9 value = [0, 4, 5] class = High
X9≤ 377.0 gini = 0.225 samples = 136 value = [12, 119, 5] class = Medium
gini = 0.0 samples = 3 value = [3, 0, 0] class = Low
X6≤ 89.0 gini = 0.375 samples = 4 value = [1, 3, 0] class = Medium
X7≤ 2.945 gini = 0.444 samples = 3 value = [2, 1, 0] class = Low
X5≤ 6.843 gini = 0.036 samples = 108 value = [106, 2, 0] class = Low
gini = 0.0 samples = 1 value = [1, 0, 0] class = Low
gini = 0.0 samples = 10 value = [0, 10, 0] class = Medium
gini = 0.0 samples = 3 value = [0, 3, 0] class = Medium
X0≤ 1.648 gini = 0.526 samples = 21 value = [12, 8, 1] class = Low
gini = 0.0 samples = 3 value = [3, 0, 0] class = Low
gini = 0.0 samples = 3 value = [0, 3, 0] class = Medium
gini = 0.0 samples = 3 value = [0, 0, 3] class = High
X10≤ 15.95 gini = 0.444 samples = 6 value = [0, 4, 2] class = Medium
X2≤ 1.235 gini = 0.11 samples = 104 value = [3, 98, 3] class = Medium
X4≤ 0.565 gini = 0.486 samples = 32 value = [9, 21, 2] class = Medium
gini = 0.0 samples = 1 value = [1, 0, 0] class = Low
gini = 0.0 samples = 3 value = [0, 3, 0] class = Medium
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
gini = 0.0 samples = 2 value = [2, 0, 0] class = Low
X6≤ 82.55 gini = 0.019 samples = 105 value = [104, 1, 0] class = Low
X11≤ 277.615 gini = 0.444 samples = 3 value = [2, 1, 0] class = Low
X4≤ 0.481 gini = 0.278 samples = 12 value = [10, 2, 0] class = Low
X6≤ 96.75 gini = 0.494 samples = 9 value = [2, 6, 1] class = Medium
gini = 0.0 samples = 2 value = [0, 0, 2] class = High
gini = 0.0 samples = 4 value = [0, 4, 0] class = Medium
gini = 0.0 samples = 1 value = [0, 0, 1] class = High
X10≤ 20.6 gini = 0.094 samples = 103 value = [3, 98, 2] class = Medium
X6≤ 82.9 gini = 0.571 samples = 21 value = [8, 11, 2] class = Medium
X10≤ 20.7 gini = 0.165 samples = 11 value = [1, 10, 0] class = Medium
X6≤ 81.4 gini = 0.198 samples = 9 value = [8, 1, 0] class = Low
gini = 0.0 samples = 96 value = [96, 0, 0] class = Low
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
gini = 0.0 samples = 2 value = [2, 0, 0] class = Low
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
X12≤ 8.365 gini = 0.165 samples = 11 value = [10, 1, 0] class = Low
gini = 0.0 samples = 5 value = [0, 5, 0] class = Medium
X5≤ 5.363 gini = 0.625 samples = 4 value = [2, 1, 1] class = Low
X5≤ 7.127 gini = 0.06 samples = 97 value = [1, 94, 2] class = Medium
X7≤ 2.492 gini = 0.444 samples = 6 value = [2, 4, 0] class = Medium
X11≤ 380.26 gini = 0.498 samples = 15 value = [3, 10, 2] class = Medium
X5≤ 6.198 gini = 0.278 samples = 6 value = [5, 1, 0] class = Low
gini = 0.0 samples = 9 value = [0, 9, 0] class = Medium
X0≤ 0.891 gini = 0.5 samples = 2 value = [1, 1, 0] class = Low
gini = 0.0 samples = 8 value = [8, 0, 0] class = Low
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
gini = 0.0 samples = 10 value = [10, 0, 0] class = Low
X11≤ 253.47 gini = 0.5 samples = 2 value = [0, 1, 1] class = Medium
gini = 0.0 samples = 2 value = [2, 0, 0] class = Low
X11≤ 365.245 gini = 0.023 samples = 87 value = [1, 86, 0] class = Medium
X5≤ 7.155 gini = 0.32 samples = 10 value = [0, 8, 2] class = Medium
gini = 0.0 samples = 4 value = [0, 4, 0] class = Medium
gini = 0.0 samples = 2 value = [2, 0, 0] class = Low
X1≤ 22.5 gini = 0.444 samples = 3 value = [2, 0, 1] class = Low
X12≤ 3.66 gini = 0.292 samples = 12 value = [1, 10, 1] class = Medium
X11≤ 393.58 gini = 0.5 samples = 2 value = [1, 1, 0] class = Low
gini = 0.0 samples = 4 value = [4, 0, 0] class = Low
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
gini = 0.0 samples = 1 value = [1, 0, 0] class = Low
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
gini = 0.0 samples = 1 value = [0, 0, 1] class = High
X6≤ 29.95 gini = 0.375 samples = 4 value = [1, 3, 0] class = Medium
gini = 0.0 samples = 83 value = [0, 83, 0] class = Medium
gini = 0.0 samples = 2 value = [0, 0, 2] class = High
gini = 0.0 samples = 8 value = [0, 8, 0] class = Medium
gini = 0.0 samples = 2 value = [2, 0, 0] class = Low
gini = 0.0 samples = 1 value = [0, 0, 1] class = High
gini = 0.0 samples = 1 value = [0, 0, 1] class = High
X8≤ 2.0 gini = 0.165 samples = 11 value = [1, 10, 0] class = Medium
gini = 0.0 samples = 1 value = [0, 1, 0] class = Medium
gini = 0.0 samples = 1 value = [1, 0, 0] class = Low
gini = 0.0 samples = 1 value = [1, 0, 0] class = Low
gini = 0.0 samples = 3 value = [0, 3, 0] class = Medium
gini = 0.0 samples = 1 value = [1, 0, 0] class = Low
gini = 0.0 samples = 10 value = [0, 10, 0] class = Medium

In [ ]:

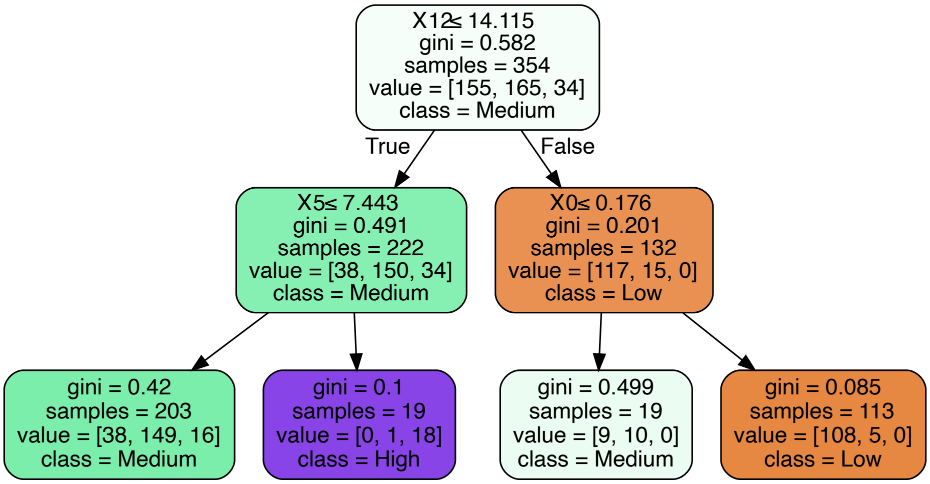
```
# 5. Next, Generate (Td−1) decision trees on the same training set using fixed tree depths
# {1, 2, ...(T d −1)}. The tree depth can be set using max=d, where d is the depth of the tree.

# 6. For each of the (Td−1) trees report, tree depth, number of leaves, feature importance,
# train score, and test score of the tree.
best = 0
best_clf = None
for depth in range(1, td):
    dt_set_clf = DecisionTreeClassifier(max_depth=depth)
    dt_set_clf.fit(X_train, y_train)
    report_on_dt(dt_set_clf)
    score = dt_set_clf.score(X_test, y_test)
    if score > best:
        best = score
        best_clf = dt_set_clf
```

```
        Tree Depth: 1
        Number of Leaves: 2
        Feature importance: Feature 1: 0.000 Feature 2: 0.000 Feature 3: 0.000 Feature 4: 0.000 Feature 5: 0.000 Feature 6: 0.000 Feature 7: 0.000 Feature 8:
0.000 Feature 9: 0.000 Feature 10: 0.000 Feature 11: 0.000 Feature 12: 0.000 Feature 13: 1.000
        Train score: 0.7542372881355932
        Test score: 0.770

        Tree Depth: 2
        Number of Leaves: 4
        Feature importance: Feature 1: 0.076 Feature 2: 0.000 Feature 3: 0.000 Feature 4: 0.000 Feature 5: 0.000 Feature 6: 0.218 Feature 7: 0.000 Feature 8:
0.000 Feature 9: 0.000 Feature 10: 0.000 Feature 11: 0.000 Feature 12: 0.000 Feature 13: 0.706
        Train score: 0.8050847457627118
        Test score: 0.842

        Tree Depth: 3
        Number of Leaves: 8
        Feature importance: Feature 1: 0.079 Feature 2: 0.000 Feature 3: 0.000 Feature 4: 0.000 Feature 5: 0.000 Feature 6: 0.284 Feature 7: 0.000 Feature 8:
0.000 Feature 9: 0.000 Feature 10: 0.000 Feature 11: 0.047 Feature 12: 0.000 Feature 13: 0.590
        Train score: 0.8305084745762712
        Test score: 0.822

        Tree Depth: 4
        Number of Leaves: 12
        Feature importance: Feature 1: 0.057 Feature 2: 0.000 Feature 3: 0.014 Feature 4: 0.000 Feature 5: 0.000 Feature 6: 0.255 Feature 7: 0.014 Feature 8:
0.068 Feature 9: 0.000 Feature 10: 0.000 Feature 11: 0.028 Feature 12: 0.014 Feature 13: 0.549
        Train score: 0.867231638418079
        Test score: 0.836

        Tree Depth: 5
        Number of Leaves: 18
        Feature importance: Feature 1: 0.064 Feature 2: 0.000 Feature 3: 0.013 Feature 4: 0.000 Feature 5: 0.000 Feature 6: 0.268 Feature 7: 0.032 Feature 8:
0.061 Feature 9: 0.013 Feature 10: 0.027 Feature 11: 0.026 Feature 12: 0.000 Feature 13: 0.497
        Train score: 0.884180790960452
        Test score: 0.829

        Tree Depth: 6
        Number of Leaves: 26
        Feature importance: Feature 1: 0.067 Feature 2: 0.000 Feature 3: 0.012 Feature 4: 0.000 Feature 5: 0.000 Feature 6: 0.254 Feature 7: 0.042 Feature 8:
0.066 Feature 9: 0.000 Feature 10: 0.046 Feature 11: 0.047 Feature 12: 0.000 Feature 13: 0.468
        Train score: 0.9152542372881356
        Test score: 0.829

        Tree Depth: 7
        Number of Leaves: 32
        Feature importance: Feature 1: 0.073 Feature 2: 0.000 Feature 3: 0.022 Feature 4: 0.000 Feature 5: 0.027 Feature 6: 0.237 Feature 7: 0.040 Feature 8:
0.062 Feature 9: 0.000 Feature 10: 0.054 Feature 11: 0.048 Feature 12: 0.008 Feature 13: 0.429
        Train score: 0.9378531073446328
        Test score: 0.836

        Tree Depth: 8
        Number of Leaves: 38
        Feature importance: Feature 1: 0.059 Feature 2: 0.000 Feature 3: 0.020 Feature 4: 0.000 Feature 5: 0.020 Feature 6: 0.240 Feature 7: 0.077 Feature 8: 
0.067 Feature 9: 0.000 Feature 10: 0.051 Feature 11: 0.051 Feature 12: 0.007 Feature 13: 0.407
        Train score: 0.9576271186440678
        Test score: 0.822

        Tree Depth: 9
        Number of Leaves: 45
        Feature importance: Feature 1: 0.075 Feature 2: 0.000 Feature 3: 0.019 Feature 4: 0.000 Feature 5: 0.016 Feature 6: 0.252 Feature 7: 0.073 Feature 8: 
0.059 Feature 9: 0.000 Feature 10: 0.048 Feature 11: 0.039 Feature 12: 0.014 Feature 13: 0.405
        Train score: 0.9774011299435028
        Test score: 0.822

        Tree Depth: 10
        Number of Leaves: 51
        Feature importance: Feature 1: 0.069 Feature 2: 0.000 Feature 3: 0.031 Feature 4: 0.000 Feature 5: 0.016 Feature 6: 0.248 Feature 7: 0.077 Feature 8: 
0.060 Feature 9: 0.008 Feature 10: 0.070 Feature 11: 0.031 Feature 12: 0.015 Feature 13: 0.374
        Train score: 0.9943502824858758
        Test score: 0.780
```

In [ ]:

```python
# 7. Show the visual output of the decision tree with highest test score from the (Td-1) trees.
dot_data = export_graphviz(best_clf, class_names=['Low', 'Medium', 'High'],
                        filled=True, rounded=True,  special_characters=True)
graph = graphviz.Source(dot_data)
graph.render(f"boston_housing_{best_clf.get_depth()}")
graph
```

Out[ ]:


X12≤ 14.115
gini = 0.582
samples = 354
value = [155, 165, 34]
class = Medium
True
False
X5≤ 7.443
gini = 0.491
samples = 222
value = [38, 150, 34]
class = Medium
X0≤ 0.176
gini = 0.201
samples = 132
value = [117, 15, 0]
class = Low
gini = 0.42
samples = 203
value = [38, 149, 16]
class = Medium
gini = 0.1
samples = 19
value = [0, 1, 18]
class = High
gini = 0.499
samples = 19
value = [9, 10, 0]
class = Medium
gini = 0.085
samples = 113
value = [108, 5, 0]
class = Low