```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier, export_graphviz
%matplotlib inline
play_data = pd.read_csv('/content/tennis.csv')
play_data
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **0** | D1 | Sunny | Hot | High | Weak | No |
| **1** | D2 | Sunny | Hot | High | Strong | No |
| **2** | D3 | Overcast | Hot | High | Weak | Yes |
| **3** | D4 | Rain | Mild | High | Weak | Yes |
| **4** | D5 | Rain | Cool | Normal | Weak | Yes |
| **5** | D6 | Rain | Cool | Normal | Strong | No |
| **6** | D7 | Overcast | Cool | Normal | Strong | Yes |
| **7** | D8 | Sunny | Mild | High | Weak | No |
| **8** | D9 | Sunny | Cool | Normal | Weak | Yes |
| **9** | D10 | Rain | Mild | Normal | Weak | Yes |
| **10** | D11 | Sunny | Mild | Normal | Strong | Yes |
| **11** | D12 | Overcast | Mild | High | Strong | Yes |
| **12** | D13 | Overcast | Hot | Normal | Weak | Yes |
| **13** | D14 | Rain | Mild | High | Strong | No |

```
play_data.play.value_counts()
```

```
Yes    9
No     5
Name: play, dtype: int64
```

```
Entropy_Play = -(9/14)*np.log2(9/14) -(5/14)*np.log2(5/14)
```

```
Entropy_Play
```

```
0.9402859586706309
```

```
play_data[play_data.outlook == 'Sunny']
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **0** | D1 | Sunny | Hot | High | Weak | No |
| **1** | D2 | Sunny | Hot | High | Strong | No |
| **7** | D8 | Sunny | Mild | High | Weak | No |
| **8** | D9 | Sunny | Cool | Normal | Weak | Yes |
| **10** | D11 | Sunny | Mild | Normal | Strong | Yes |

```
# Entropy(Play|Outlook=Sunny)
Entropy_Play_Outlook_Sunny =-(3/5)*np.log2(3/5) -(2/5)*np.log2(2/5)
```

```
Entropy_Play_Outlook_Sunny
```

```
0.9709505944546686
```

```
play_data[play_data.outlook == 'Overcast']
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **2** | D3 | Overcast | Hot | High | Weak | Yes |
| **6** | D7 | Overcast | Cool | Normal | Strong | Yes |
| **11** | D12 | Overcast | Mild | High | Strong | Yes |
| **12** | D13 | Overcast | Hot | Normal | Weak | Yes |

```
play_data[play_data.outlook == 'Rain']
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **3** | D4 | Rain | Mild | High | Weak | Yes |
| **4** | D5 | Rain | Cool | Normal | Weak | Yes |
| **5** | D6 | Rain | Cool | Normal | Strong | No |
| **9** | D10 | Rain | Mild | Normal | Weak | Yes |
| **13** | D14 | Rain | Mild | High | Strong | No |

```
Entropy_Play_Outlook_Rain = -(2/5)*np.log2(2/5) - (3/5)*np.log2(3/5)
```

```
Entropy_Play_Outlook_Rain
```

```
0.9709505944546686
```

```
Entropy_Play - (5/14)*Entropy_Play_Outlook_Sunny - (4/14)*0 - (5/14) * Entropy_Play_Outloo
```

```
0.2467498197744391
```

```
play_data[play_data.outlook == 'Overcast']
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **2** | D3 | Overcast | Hot | High | Weak | Yes |
| **6** | D7 | Overcast | Cool | Normal | Strong | Yes |
| **11** | D12 | Overcast | Mild | High | Strong | Yes |
| **12** | D13 | Overcast | Hot | Normal | Weak | Yes |

```
play_data[play_data.outlook == 'Sunny']
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **0** | D1 | Sunny | Hot | High | Weak | No |
| **1** | D2 | Sunny | Hot | High | Strong | No |
| **7** | D8 | Sunny | Mild | High | Weak | No |
| **8** | D9 | Sunny | Cool | Normal | Weak | Yes |
| **10** | D11 | Sunny | Mild | Normal | Strong | Yes |

```
Entropy_Play_Outlook_Sunny =-(3/5)*np.log2(3/5) -(2/5)*np.log2(2/5)
Entropy_Play_Outlook_Sunny
```

```
0.9709505944546686
```

```
Entropy_Play_Outlook_Sunny - (3/5)*0 - (2/5)*0
```

```
0.9709505944546686
```

```
Entropy_Wind_False = -(1/3)*np.log2(1/3) - (2/3)*np.log2(2/3)
Entropy_Wind_False
```

```
0.9182958340544896
```

```
Entropy_Play_Outlook_Sunny - (3/5)* Entropy_Wind_False - (2/5)*1
```

```
0.01997309402197489
```

```
Entropy_Play_Outlook_Sunny - (2/5)*0 - (1/5)*0 - (2/5)* 1
```

```
0.5709505944546686
```

```
play_data[(play_data.outlook == 'Sunny') & (play_data.humidity == 'High')]
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **0** | D1 | Sunny | Hot | High | Weak | No |
| **1** | D2 | Sunny | Hot | High | Strong | No |
| **7** | D8 | Sunny | Mild | High | Weak | No |

```
play_data[(play_data.outlook == 'Sunny') & (play_data.humidity == 'Normal')]
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **8** | D9 | Sunny | Cool | Normal | Weak | Yes |
| **10** | D11 | Sunny | Mild | Normal | Strong | Yes |

```
play_data[play_data.outlook == 'Rain']
```

| | day | outlook | temp | humidity | wind | play |
|---|---|---|---|---|---|---|
| **3** | D4 | Rain | Mild | High | Weak | Yes |
| **4** | D5 | Rain | Cool | Normal | Weak | Yes |
| **5** | D6 | Rain | Cool | Normal | Strong | No |
| **9** | D10 | Rain | Mild | Normal | Weak | Yes |
| **13** | D14 | Rain | Mild | High | Strong | No |

```
Entropy_Play_Outlook_Rainy =-(3/5)*np.log2(3/5) -(2/5)*np.log2(2/5)
Entropy_Play_Outlook_Rainy
```

```
0.9709505944546686
```

```
Entropy_Play_Outlook_Rainy - (3/5)*0.918 - (2/5)*1
```

```
0.020150594454668602
```

```
Entropy_Play_Outlook_Rainy - (2/5)*0 - (3/5)*0
```

```
0.9709505944546686
```

```
Entropy_Play_Outlook_Rainy_Normal = -(1/3)*np.log2(1/3) - (2/3)*np.log2(2/3)
Entropy_Play_Outlook_Rainy_Normal
```

```
0.9182958340544896
```

```
Entropy_Play_Outlook_Rainy - (2/5)*1 - (3/5)*Entropy_Play_Outlook_Rainy_Normal
```

```
0.01997309402197489
```

## Decision Tree for Classication

```
from sklearn.datasets import load_breast_cancer
from sklearn.tree import DecisionTreeClassifier, export_graphviz, ExtraTreeClassifier
cancer = load_breast_cancer()
cancer.data[:5]
```

```
array([[1.799e+01, 1.038e+01, 1.228e+02, 1.001e+03, 1.184e-01, 2.776e-01,
        3.001e-01, 1.471e-01, 2.419e-01, 7.871e-02, 1.095e+00, 9.053e-01,
        8.589e+00, 1.534e+02, 6.399e-03, 4.904e-02, 5.373e-02, 1.587e-02,
        3.003e-02, 6.193e-03, 2.538e+01, 1.733e+01, 1.846e+02, 2.019e+03,
        1.622e-01, 6.656e-01, 7.119e-01, 2.654e-01, 4.601e-01, 1.189e-01],
       [2.057e+01, 1.777e+01, 1.329e+02, 1.326e+03, 8.474e-02, 7.864e-02,
        8.690e-02, 7.017e-02, 1.812e-01, 5.667e-02, 5.435e-01, 7.339e-01,
        3.398e+00, 7.408e+01, 5.225e-03, 1.308e-02, 1.860e-02, 1.340e-02,
        1.389e-02, 3.532e-03, 2.499e+01, 2.341e+01, 1.588e+02, 1.956e+03,
        1.238e-01, 1.866e-01, 2.416e-01, 1.860e-01, 2.750e-01, 8.902e-02],
       [1.969e+01, 2.125e+01, 1.300e+02, 1.203e+03, 1.096e-01, 1.599e-01,
        1.974e-01, 1.279e-01, 2.069e-01, 5.999e-02, 7.456e-01, 7.869e-01,
        4.585e+00, 9.403e+01, 6.150e-03, 4.006e-02, 3.832e-02, 2.058e-02,
        2.250e-02, 4.571e-03, 2.357e+01, 2.553e+01, 1.525e+02, 1.709e+03,
        1.444e-01, 4.245e-01, 4.504e-01, 2.430e-01, 3.613e-01, 8.758e-02],
       [1.142e+01, 2.038e+01, 7.758e+01, 3.861e+02, 1.425e-01, 2.839e-01,
        2.414e-01, 1.052e-01, 2.597e-01, 9.744e-02, 4.956e-01, 1.156e+00,
        3.445e+00, 2.723e+01, 9.110e-03, 7.458e-02, 5.661e-02, 1.867e-02,
        5.963e-02, 9.208e-03, 1.491e+01, 2.650e+01, 9.887e+01, 5.677e+02,
        2.098e-01, 8.663e-01, 6.869e-01, 2.575e-01, 6.638e-01, 1.730e-01],
       [2.029e+01, 1.434e+01, 1.351e+02, 1.297e+03, 1.003e-01, 1.328e-01,
        1.980e-01, 1.043e-01, 1.809e-01, 5.883e-02, 7.572e-01, 7.813e-01,
        5.438e+00, 9.444e+01, 1.149e-02, 2.461e-02, 5.688e-02, 1.885e-02,
        1.756e-02, 5.115e-03, 2.254e+01, 1.667e+01, 1.522e+02, 1.575e+03,
        1.374e-01, 2.050e-01, 4.000e-01, 1.625e-01, 2.364e-01, 7.678e-02]])
```

```
cancer.target[:5]
```

```
array([0, 0, 0, 0, 0])
```

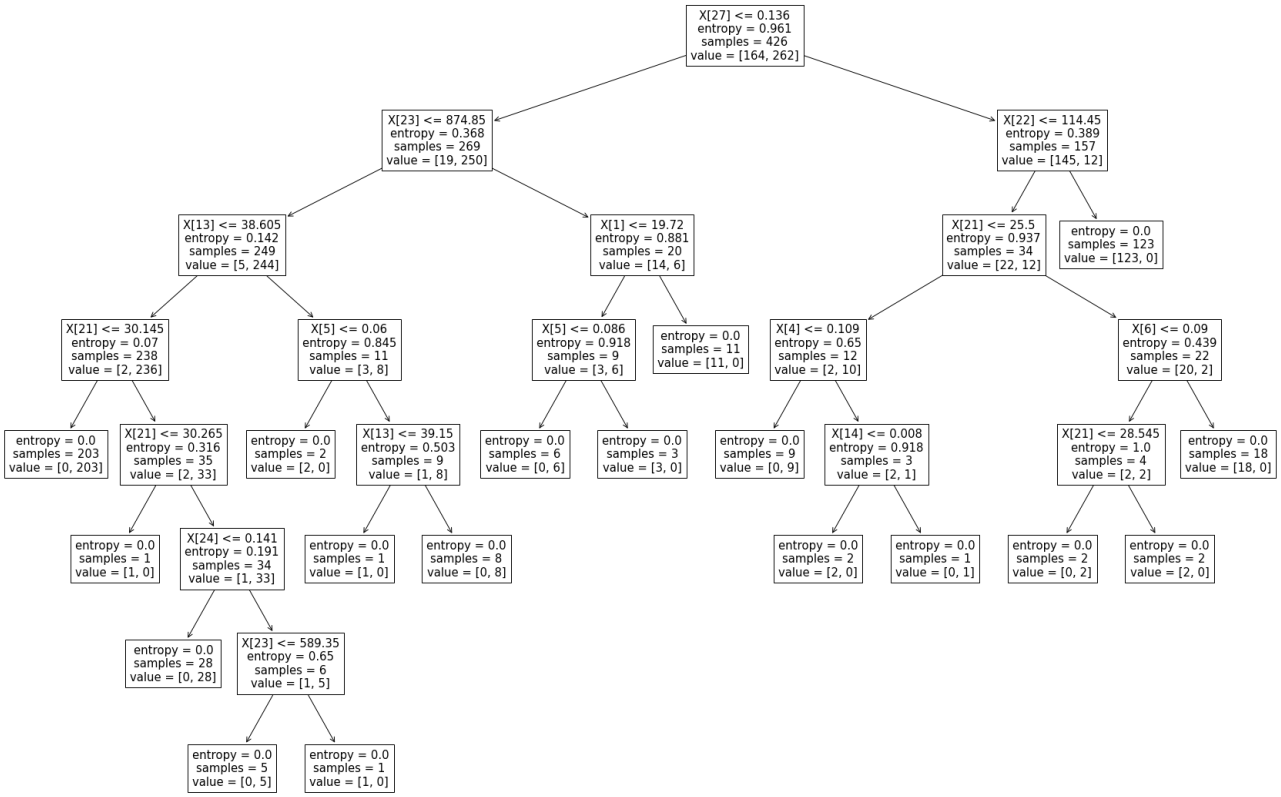
```
dt = DecisionTreeClassifier(criterion='entropy')
```

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

```
trainX,testX,trainY,testY = train_test_split(cancer.data, cancer.target)
dt.fit(trainX,trainY)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

```
fig = plt.figure(figsize=(30,20))
tree.plot_tree(dt);
```




```
dt.predict(testX)
```

```
array([0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1,
       1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0,
       1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0])
```

```
dt.feature_importances
```

```
dt.feature_importances_
```

```
array([0.        , 0.02285489, 0.        , 0.        , 0.0123181 ,
       0.03182248, 0.01384048, 0.        , 0.        , 0.        ,
       0.        , 0.        , 0.        , 0.0339327 , 0.00672595,
       0.        , 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.06965883, 0.07154064, 0.12209759, 0.00636878,
       0.        , 0.        , 0.60883957, 0.        , 0.        ])
```

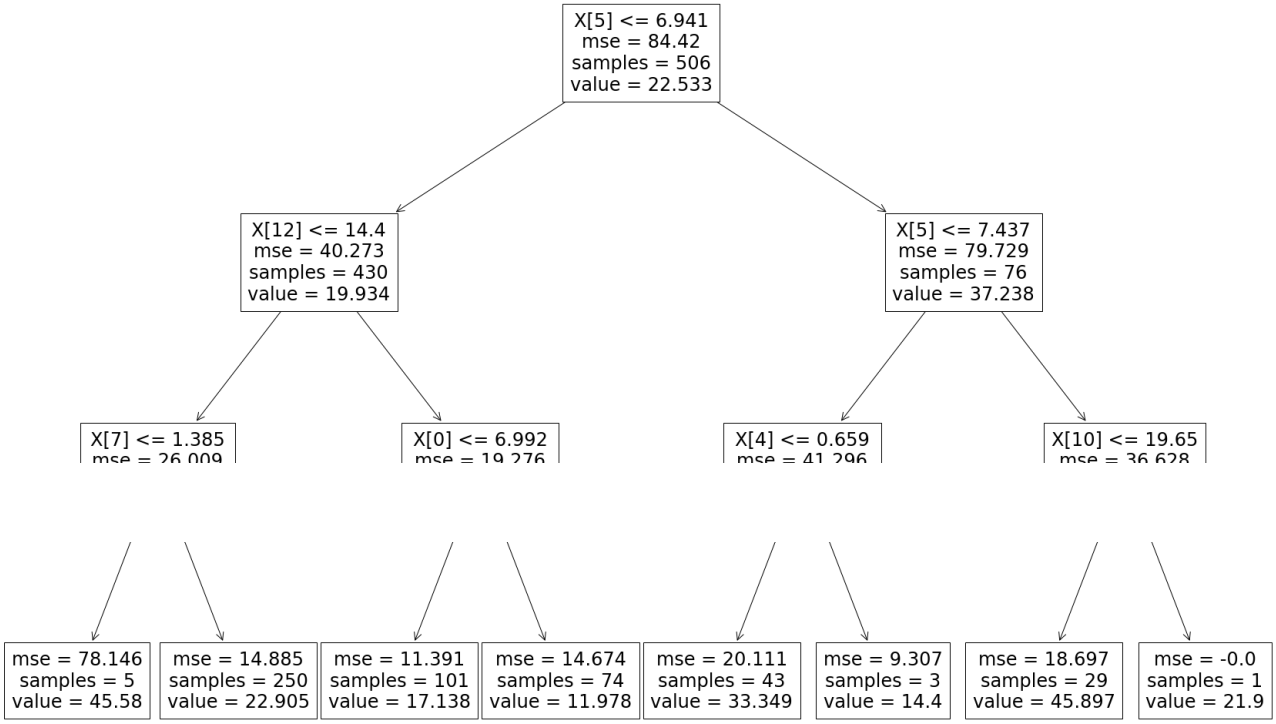
## Decision Tree for Regression

```
from sklearn.datasets import load_boston
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor, export_graphviz, E
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

```
boston = load_boston()
X = boston.data
y = boston.target
```

```
regr = DecisionTreeRegressor(max_depth=3, random_state=1234)
model = regr.fit(X, y)
```

```
fig = plt.figure(figsize=(30,20))
tree.plot_tree(model);
```


X[5] <= 6.941
mse = 84.42
samples = 506
value = 22.533
X[12] <= 14.4
mse = 40.273
samples = 430
value = 19.934
X[5] <= 7.437
mse = 79.729
samples = 76
value = 37.238
X[7] <= 1.385
X[0] <= 6.992
X[4] <= 0.659
X[10] <= 19.65
mse = 78.146
samples = 5
value = 45.58
mse = 14.885
samples = 250
value = 22.905
mse = 11.391
samples = 101
value = 17.138
mse = 14.674
samples = 74
value = 11.978
mse = 20.111
samples = 43
value = 33.349
mse = 9.307
samples = 3
value = 14.4
mse = 18.697
samples = 29
value = 45.897
mse = -0.0
samples = 1
value = 21.9