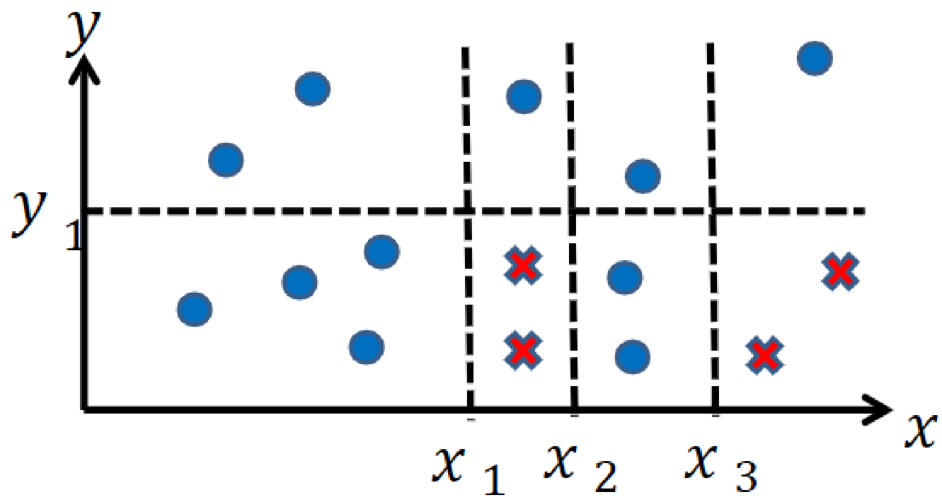
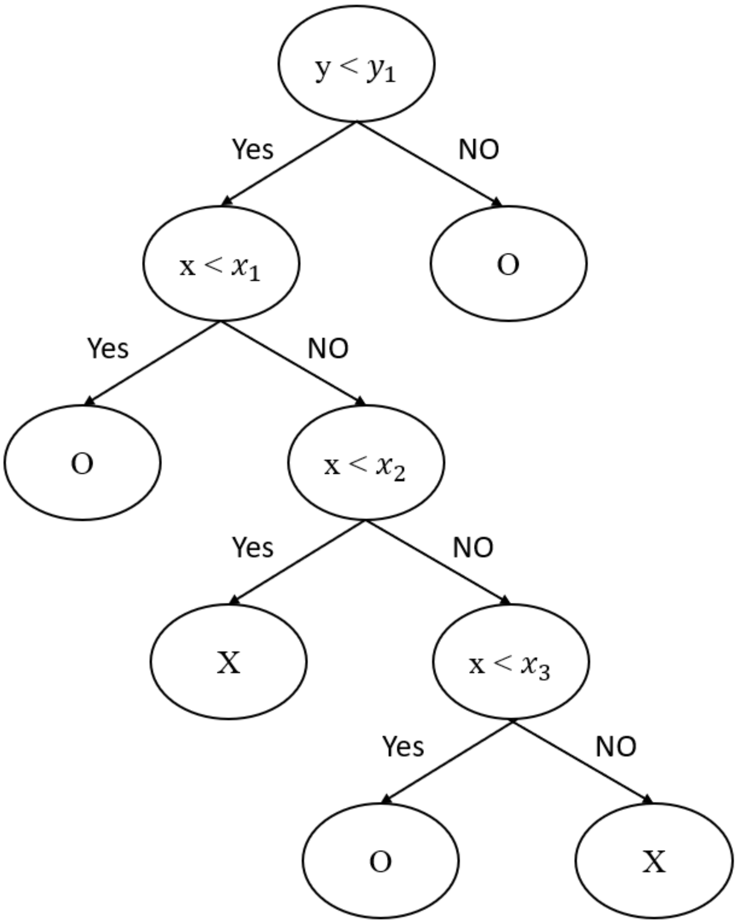
HW#4 Due:4/19/2022

110598095 歐俊杰

1. Plot a decision tree for the following data points. If you carefully design your tree, you will just need to use one “>” or “<” in a vertex.





2. We have G = 0.048 for H_0 = 65 and G = 0.102 for H_0 = 80 on pp. 46 of the decision-tree PPT file. Confirm these G values are correct by hand calculation.

Outlook	Temperature	Humidity	Windy	Play
Sunny	85	85	false	Don't Play
Sunny	80	90	true	Don't Play
Overcast	83	78	false	Play
Rainy	70	96	false	Play
Rainy	68	80	false	Play
Rainy	65	70	true	Don't Play
Overcast	64	65	true	Play
Sunny	72	95	false	Don't Play
Sunny	69	70	false	Play
Rainy	75	80	false	Play
Sunny	75	70	true	Play
Overcast	72	90	true	Play
Overcast	81	75	false	Play
Rainy	71	80	true	Don't Play

$$E_n(S) = -\frac{5}{14} * log_2\frac{5}{14} - \frac{9}{14} * log_2\frac{9}{14} \cong 0.94$$

$H_0 = 65$

$$E_n(H \leq H_0) = -\frac{1}{1} * log_2\frac{1}{1} - \frac{0}{1} * log_2\frac{0}{1} = 0 \text{，} P_{H \leq H_0} = \frac{1}{14}$$

$$E_n(H > H_0) = -\frac{8}{13} * log_2\frac{8}{13} - \frac{5}{13} * log_2\frac{5}{13} \cong 0.961 \text{，} P_{H > H_0} = \frac{13}{14}$$

$$G(S,H) = 0.94 - (\frac{1}{14} * 0 + \frac{13}{14} * 0.961) \cong 0.048$$

$H_0 = 80$

$$E_n(H \leq H_0) = -\frac{7}{9} * log_2\frac{7}{9} - \frac{2}{9} * log_2\frac{2}{9} \cong 0.764 \text{，} P_{H \leq H_0} = \frac{9}{14}$$

$$E_n(H > H_0) = -\frac{2}{5} * log_2\frac{2}{5} - \frac{3}{5} * log_2\frac{3}{5} \cong 0.971 \text{，} P_{H > H_0} = \frac{5}{14}$$

$$G(S,H) = 0.94 - (\frac{9}{14} * 0.764 + \frac{5}{14} * 0.971) \cong 0.102$$

---

3. We have a dataset $S = \left\{\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}2\\2\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1\\1\end{bmatrix}\right\}$. Follow the k-means algorithm to complete the assignment step and the update step for one run. Use $k = 2$, initial $\boldsymbol{\mu}_1 = \begin{bmatrix}-1\\-1\end{bmatrix}$ and $\boldsymbol{\mu}_2 = \begin{bmatrix}3\\3\end{bmatrix}$ in the computation.

$$s = \left\{\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}2\\2\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1\\1\end{bmatrix}\right\}$$

$$\mu_1 = \begin{bmatrix}-1\\-1\end{bmatrix} \text{，} \mu_2 = \begin{bmatrix}3\\3\end{bmatrix}$$

$$\|x_1-\mu_1\|=\sqrt{2}\text{，}\|x_1-\mu_2\|=\sqrt{18} => \begin{matrix} b(1,1)=1 \\ b(1,2)=0 \end{matrix}$$

$$\|x_2-\mu_1\|=\sqrt{18}\text{，}\|x_2-\mu_2\|=\sqrt{2} => \begin{matrix} b(2,1)=0 \\ b(2,2)=1 \end{matrix}$$

$$\|x_3-\mu_1\|=\sqrt{5}\text{，}\|x_3-\mu_2\|=\sqrt{13} => \begin{matrix} b(3,1)=1 \\ b(3,2)=0 \end{matrix}$$

$$\|x_4-\mu_1\|=\sqrt{8}\text{，}\|x_4-\mu_2\|=\sqrt{8} => \begin{matrix} b(4,1)=1 \\ b(4,2)=1 \end{matrix}$$

$$=> \mu_1=\frac{x_1+x_3+x_4}{3}=\begin{bmatrix} \frac{1}{3} \\ \frac{2}{3} \end{bmatrix}\text{，}\mu_2=\frac{x_2+x_4}{2}=\begin{bmatrix} \frac{3}{2} \\ \frac{3}{2} \end{bmatrix}$$

$$\|x_1-\mu_1\|=\sqrt{\frac{5}{9}}\text{，}\|x_1-\mu_2\|=\sqrt{\frac{18}{4}} => \begin{matrix} b(1,1)=1 \\ b(1,2)=0 \end{matrix}$$

$$\|x_2-\mu_1\|=\sqrt{\frac{41}{9}}\text{，}\|x_2-\mu_2\|=\sqrt{\frac{1}{2}} => \begin{matrix} b(2,1)=0 \\ b(2,2)=1 \end{matrix}$$

$$\|x_3-\mu_1\|=\sqrt{\frac{2}{9}}\text{，}\|x_3-\mu_2\|=\sqrt{\frac{5}{2}} => \begin{matrix} b(3,1)=1 \\ b(3,2)=0 \end{matrix}$$

$$\|x_4-\mu_1\|=\sqrt{\frac{5}{9}}\text{，}\|x_4-\mu_2\|=\sqrt{\frac{1}{2}} => \begin{matrix} b(4,1)=0 \\ b(4,2)=1 \end{matrix}$$

$$=> \mu_1=\frac{x_1+x_3}{2}=\begin{bmatrix} 0 \\ \frac{1}{2} \end{bmatrix}\text{，}\mu_2=\frac{x_2+x_4}{2}=\begin{bmatrix} \frac{3}{2} \\ \frac{3}{2} \end{bmatrix}$$

$$\|x_1-\mu_1\|=\sqrt{\frac{1}{4}}\text{，}\|x_1-\mu_2\|=\sqrt{\frac{18}{4}} => \begin{matrix} b(1,1)=1 \\ b(1,2)=0 \end{matrix}$$

$$\|x_2-\mu_1\|=\sqrt{\frac{25}{4}}\text{，}\|x_2-\mu_2\|=\sqrt{\frac{1}{2}} => \begin{matrix} b(2,1)=0 \\ b(2,2)=1 \end{matrix}$$

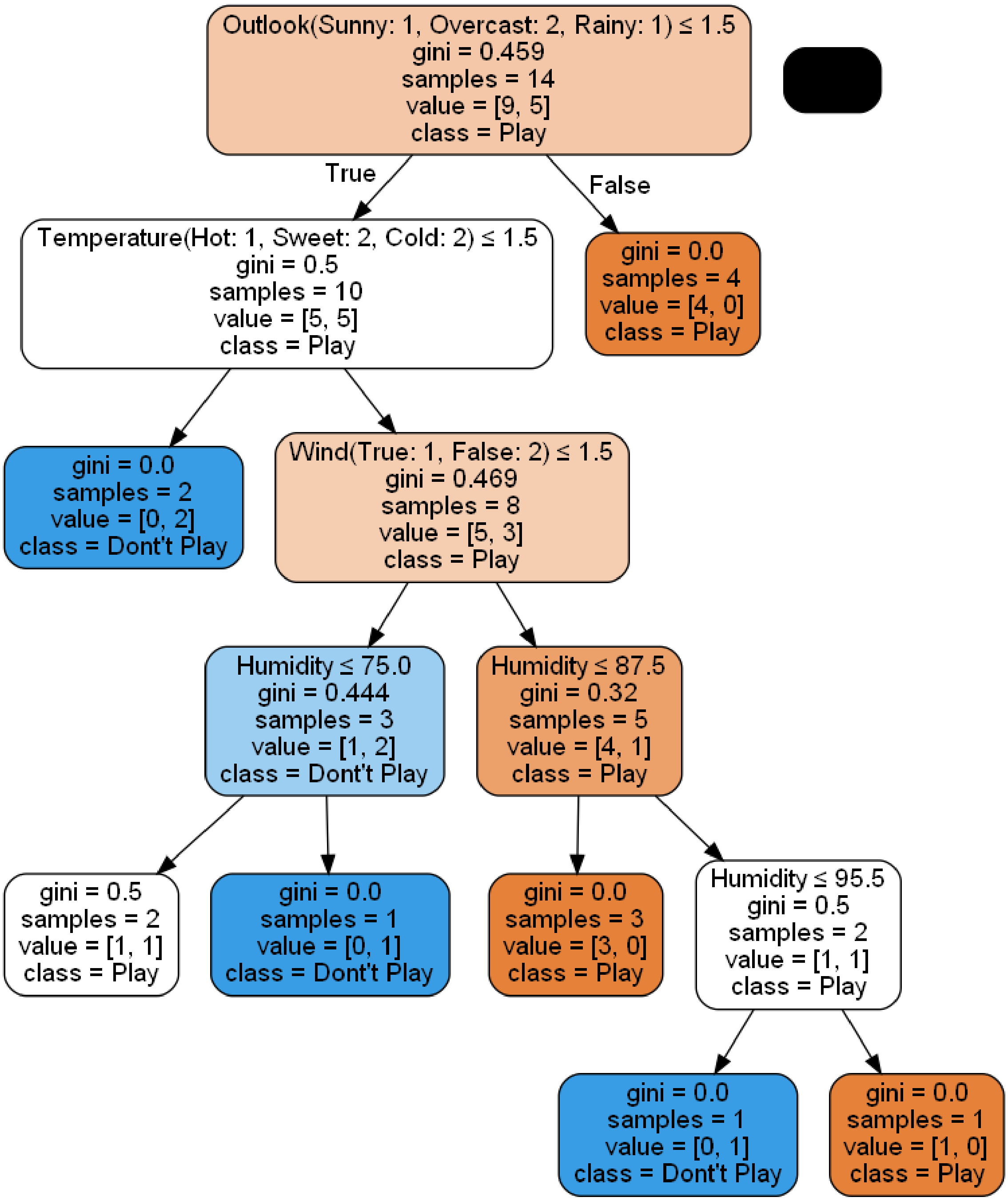
$$\|x_3-\mu_1\|=\sqrt{\frac{1}{4}}\text{，}\|x_3-\mu_2\|=\sqrt{\frac{5}{2}} => \begin{matrix} b(3,1)=1 \\ b(3,2)=0 \end{matrix}$$

$$\|x_4-\mu_1\|=\sqrt{\frac{5}{4}}\text{，}\|x_4-\mu_2\|=\sqrt{\frac{1}{2}} => \begin{matrix} b(4,1)=0 \\ b(4,2)=1 \end{matrix}$$

$$=> \mu_1=\frac{x_1+x_3}{2}=\begin{bmatrix} 0 \\ \frac{1}{2} \end{bmatrix}\text{，}\mu_2=\frac{x_2+x_4}{2}=\begin{bmatrix} \frac{3}{2} \\ \frac{3}{2} \end{bmatrix}$$

4. Write a program by using the sklearn to construct a CART tree for the play/no_play data on the lecture notes (16_decison_trees). You need to think a way to deal with the categorical data. If a particular day is sunny, high temperature, low humidity, and no wind, what is the decision based on your plotted tree?

Base on my plotted tree answer is <u>**Don’t Play**</u>



```python
from sklearn import tree
from six import StringIO
import numpy as np
import pydot
import os
```

```python
os.environ['PATH'] += os.pathsep + 'C:/Program Files/Graphviz/bin'
```

```python
feature_names = ['Outlook(Sunny: 1, Overcast: 2, Rainy: 1)',
                 'Temperature(Hot: 1, Sweet: 2, Cold: 2)',
                 'Humidity',
                 'Wind(True: 1, False: 2)']

target_names = ['Play', 'Dont\'t Play']

data = np.array([[1, 1, 85, 2], [1, 1, 90, 1], [2, 1, 78, 2], [1, 2, 96, 2],
                 [1, 2, 80, 2], [1, 2, 70, 1], [2, 2, 65, 1], [1, 2, 95, 2],
                 [1, 2, 70, 2], [1, 2, 80, 2], [1, 2, 70, 1], [2, 2, 90, 1],
                 [2, 1, 75, 2], [1, 2, 80, 1]])

target = np.array([2, 2, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 2])
```

```python
clf = tree.DecisionTreeClassifier()
clf = clf.fit(data, target)

dot_data = StringIO()

tree.export_graphviz(clf, out_file = dot_data, feature_names = feature_names,
                     class_names = target_names, filled = True, rounded = True,
                     special_characters = True)

(graph,) = pydot.graph_from_dot_data(dot_data.getvalue())
graph.write_png('DecisionTree.png')
```

5. Repeat the classification of the Iris dataset by using the random forest method with K = 50. To simplify the problem, just do 7:3 splitting for training and testing sets. Remember to repeat the trials 10 times to calculate the average. Of the methods you used in HW1, HW2, HW 3 and this HW, which method seems the best?

KNN's Accuracy is the best => KNN model seems best.

```python
from sklearn import datasets
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

```python
iris_data = datasets.load_iris()
```

```python
data = iris_data.data
target = iris_data.target
```

```python
accuracy = 0

for times in range(10):
    rfc_data = RandomForestClassifier(n_estimators = 50, random_state = 42)

    x_train, x_test, y_train, y_test = train_test_split(data, target, test_size = 0.3)

    rfc_data.fit(x_train, y_train)

    pre = rfc_data.predict(x_test)

    accuracy = accuracy + accuracy_score(y_test, pre)

print("Accuracy : %.2f" % round(accuracy * 0.1, 2))
```

```
Accuracy : 0.95
```