

# Decision Tree

อ. ปรัชญ์ ปิยะวงศ์วิศาล

Pratch Piyawongwisal

# Today

- SVM Lab
- Decision Tree
- Homework

# Lab: ใช้ SVM จำแนกพันธุ์ดอกไม้



เป้าหมาย: จำแนก Iris-Virginica ออกจากชนิดอื่น โดยใช้ขนาดของกลีบ Sepal/Petal - width/length

# Recap: Supervised Learning

Decision Tree

Neural Net

kNN

Logistic  
Regression

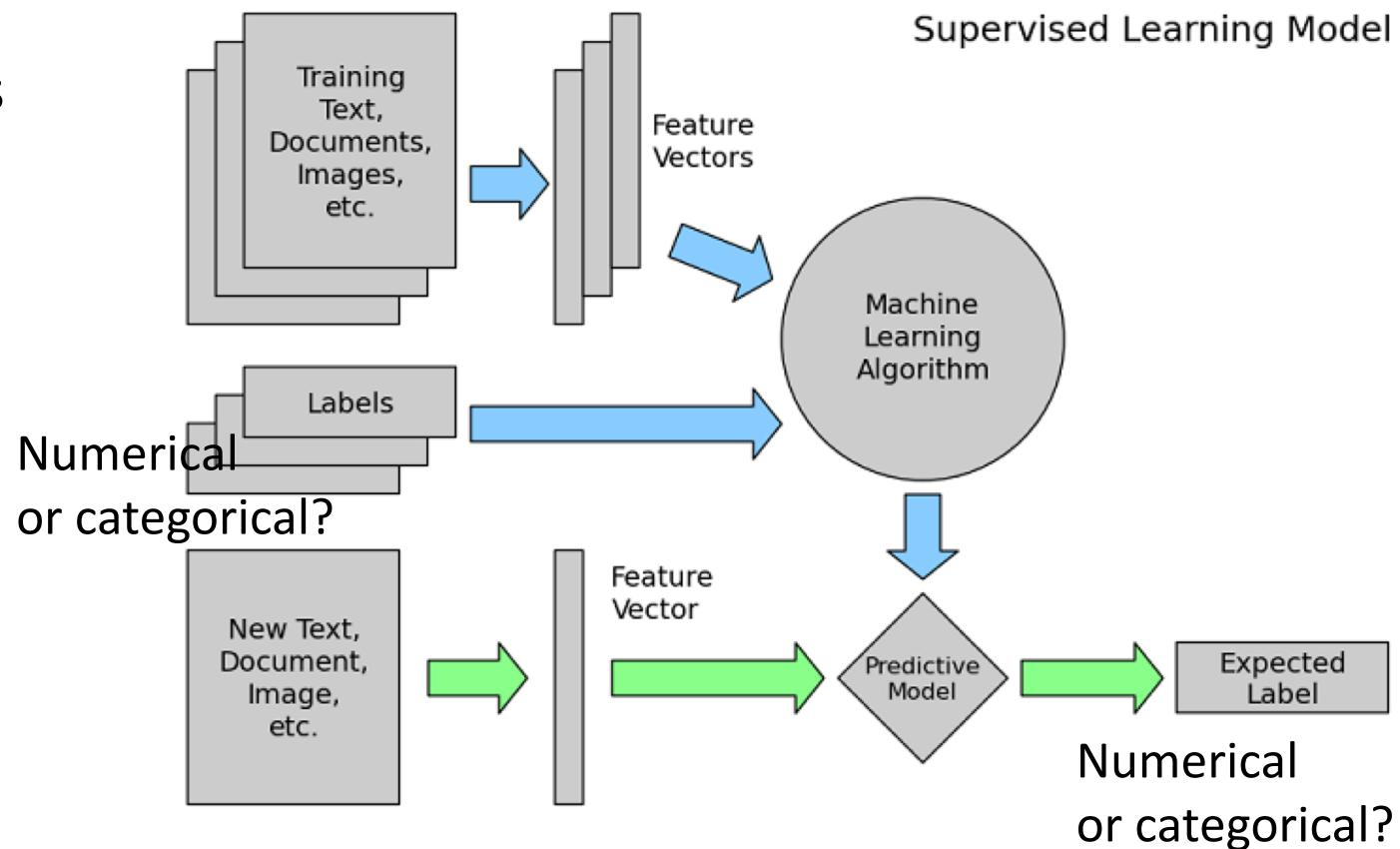
SVM

## Classification

- Predicts class labels/categories
- คำนวณค่าที่เป็นหมวดหมู่ = จำแนกประเภท
- อาจมองเป็นการหา **boundary** ที่แบ่งข้อมูลในแต่ละหมวดหมู่ ออกจากกัน

## Regression

- Predicts continuous values
- คำนวณค่าที่เป็นจำนวนจริง
- อาจมองเป็นการหา **hyperplane** ที่ fit กับข้อมูลที่มีมากที่สุด



# Decision Tree

# Decision Tree

- เป็น supervised learning ใช้ทำ classification เป็นหลัก
- ข้อดี
  - เข้าใจง่าย
  - สามารถตีความและอธิบายที่มาที่ไปได้ง่าย (**white box model**) -- ต่างจาก **black box model** อ่อนง Neural Network
  - ใช้กับข้อมูลที่เป็น numerical หรือ categorical ก็ได้
  - สามารถประมาณความน่าจะเป็นที่จะเป็นแต่ละคลาส (**class probability**) ได้
- ข้อเสีย
  - decision tree เป็นปัญหา NP-complete ไม่สามารถหาคำตอบที่ดีที่สุดใน poly time ได้ จึงต้องใช้ อัลกอริทึมแบบ **heuristic** เช่น **greedy** ในการหาคำตอบแบบ **local optima**

# ລອງ Implement Decision Tree

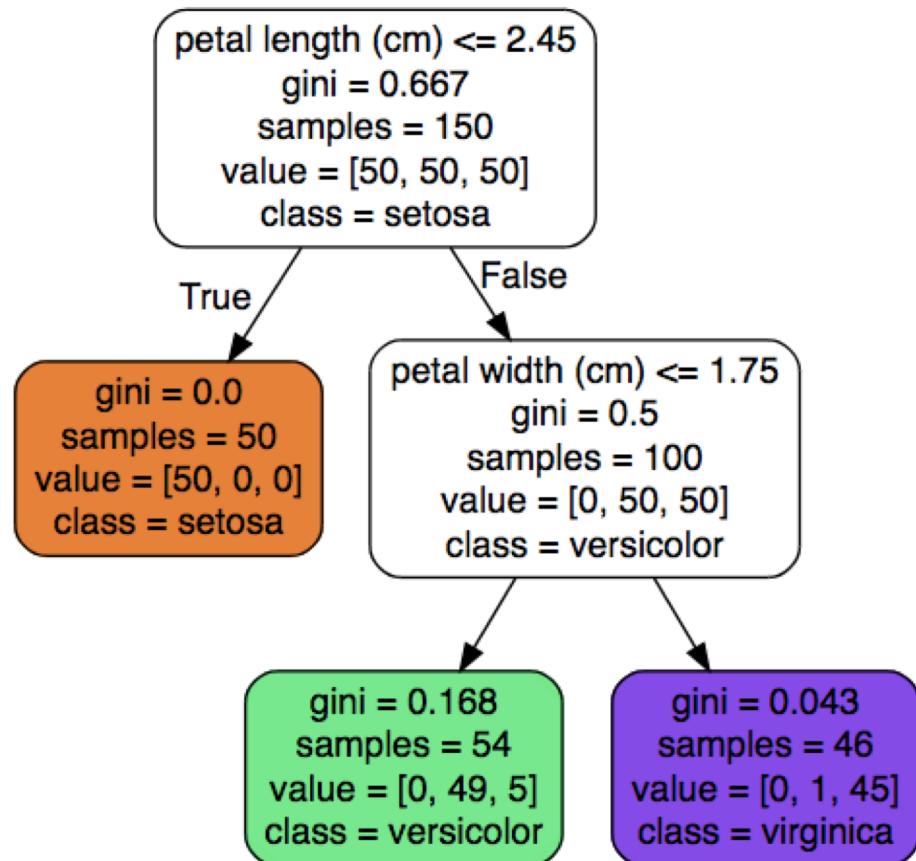
```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
import numpy as np

iris = load_iris()
X = iris.data[:, 2:]          # feature: petal length/width
y = iris.target                # label: 0, 1, 2
np.random.seed(42)
tree_clf = DecisionTreeClassifier(max_depth=2)
tree_clf.fit(X, y)
```

# Visualizing the Decision Tree

```
from sklearn.tree import export_graphviz

tree_dot = export_graphviz(
    tree_clf,
    out_file=None,  # or out_file="iris_tree.dot"
    feature_names=iris.feature_names[2:],
    class_names=iris.target_names,
    rounded=True,
    filled=True
)
print(tree_dot)
```



จากนั้นนำผลลัพธ์ไปแปลงเป็นແນກພາບໄດ້ທี่ <http://www.webgraphviz.com/>

# Alternative: using python-graphviz

Once trained, we can export the tree in `Graphviz` format using the `export_graphviz` exporter. If you use the `conda` package manager, the `graphviz` binaries and the python package can be installed with

```
conda install python-graphviz
```

Alternatively binaries for `graphviz` can be downloaded from the `graphviz` project homepage, and the Python wrapper installed from pypi with `pip install graphviz`.

Below is an example `graphviz` export of the above tree trained on the entire `iris` dataset; the results are saved in an output file `iris.pdf`:

```
>>> import graphviz  
>>> dot_data = tree.export_graphviz(clf, out_file=None)  
>>> graph = graphviz.Source(dot_data)  
>>> graph.render("iris")
```

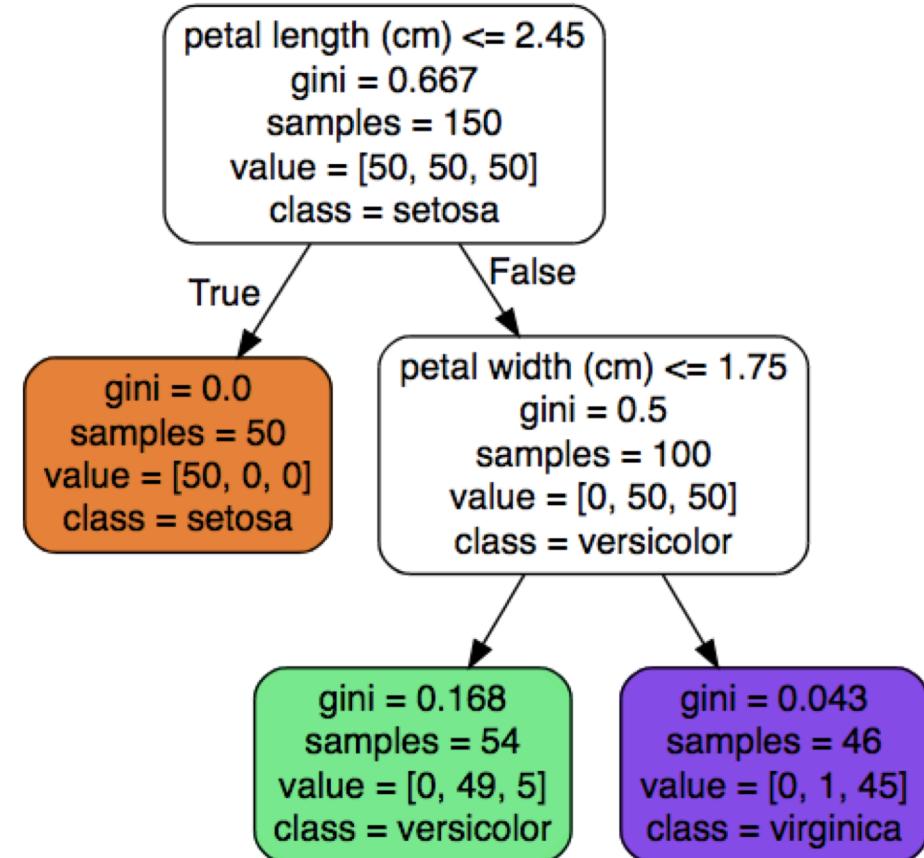
The `export_graphviz` exporter also supports a variety of aesthetic options, including coloring nodes by their class (or value for regression) and using explicit variable and class names if desired. Jupyter notebooks also render these plots inline automatically:

```
>>> dot_data = tree.export_graphviz(clf, out_file=None,  
                                 feature_names=iris.feature_names,  
                                 class_names=iris.target_names,  
                                 filled=True, rounded=True,  
                                 special_characters=True)  
>>> graph = graphviz.Source(dot_data)  
>>> graph
```

# ตีความภาพ decision tree

- เปรียบเสมือน flowchart ของกฎที่อยู่ในรูป if-else ดังนี้

```
if petal_length <= 2.45:  
    class = "setosa"  
  
elif petal <= 1.75:  
    class = "versicolor"  
  
else:  
    class = "virginica"
```



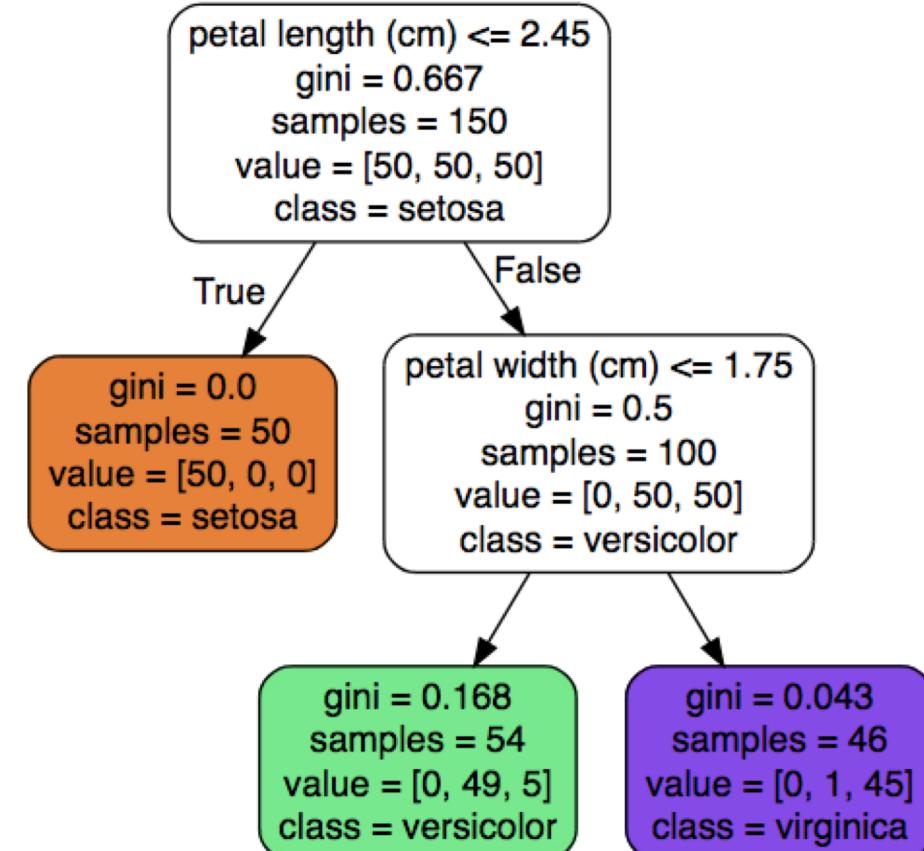
# ตีความภาพ decision tree

- node สี่ข้าง คือ ตัวแบ่ง (splitting attribute)
- ค่า gini คือค่าความปะปน (Impurity) ของ node
  - gini = 0.0 แสดงว่าข้อมูล train ใน node นั้นเป็นคลาสเดียวกันหมด
  - gini มาก แสดงว่า node มีข้อมูลหลายคลาสปะปนกัน

$$G_i = 1 - \sum_{k=1}^n p_{i,k}^2$$

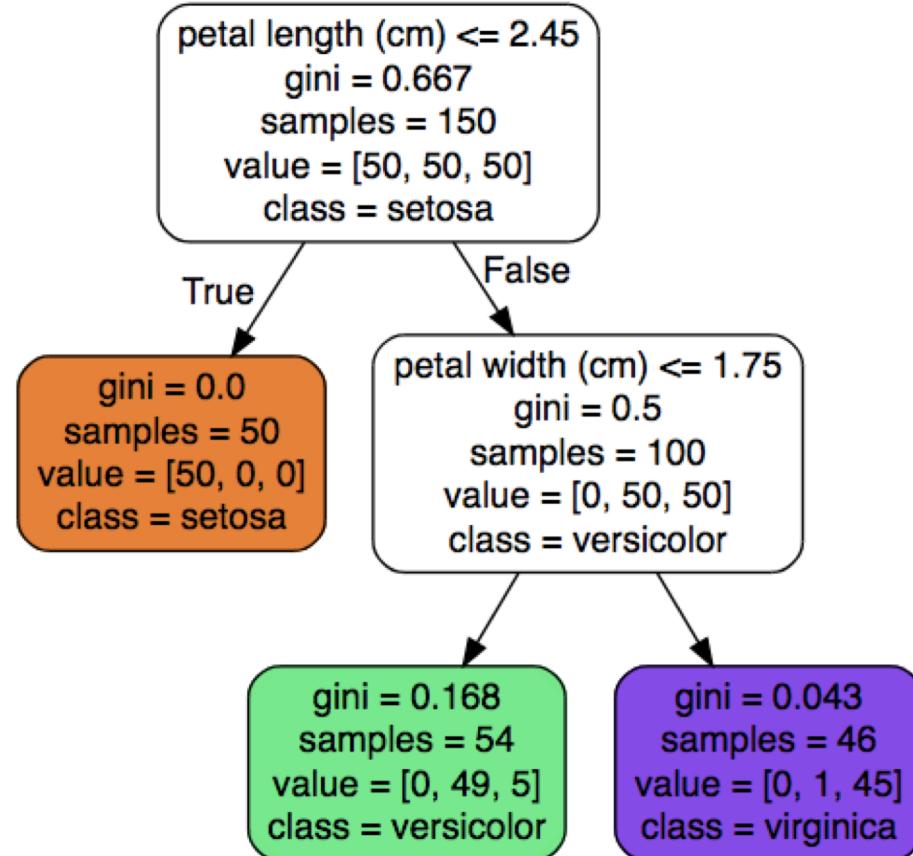
$p_{i,k}$  คืออัตราส่วนของข้อมูลคลาส  $k$  ต่อข้อมูลทั้งหมด ณ node ที่ |

- เช่น  $G(\text{โหนดเขียว}) = 1 - \left(\frac{0}{54}\right)^2 - \left(\frac{49}{54}\right)^2 - \left(\frac{5}{54}\right)^2 = 0.168$



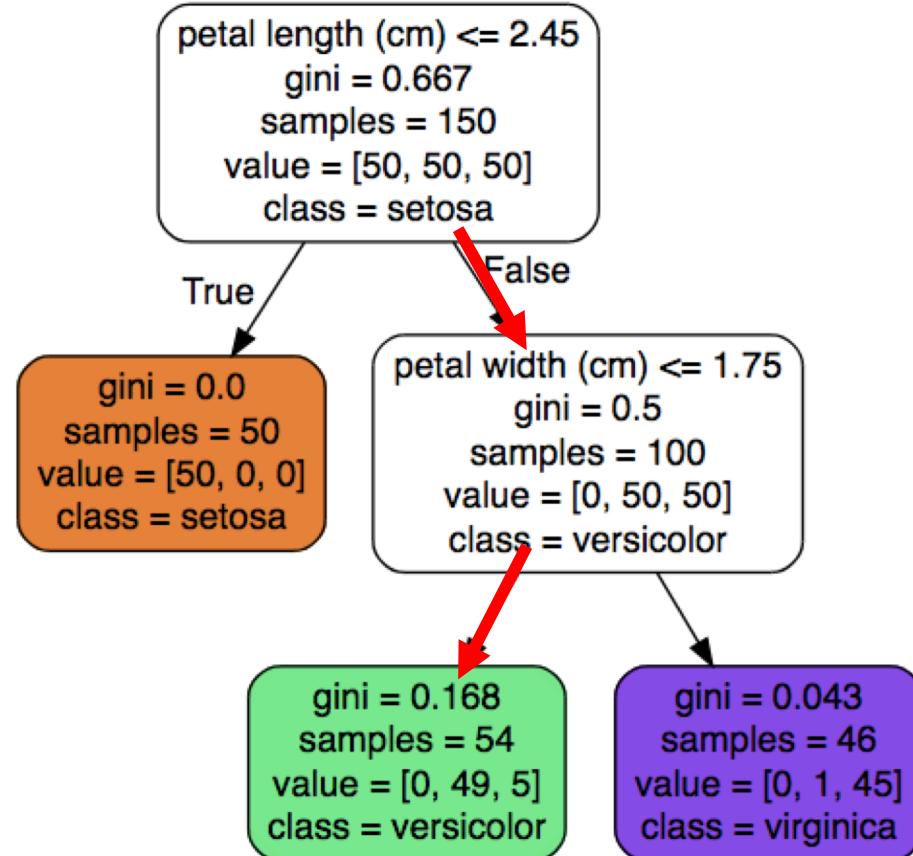
# การจำแนกด้วย decision tree

- ตัวอย่าง: จะใช้ decision tree นี้ classify ดอกกล้วยไม้ที่มีความยาวกลีบ 5cm และมีความกว้างกลีบ 1.5cm



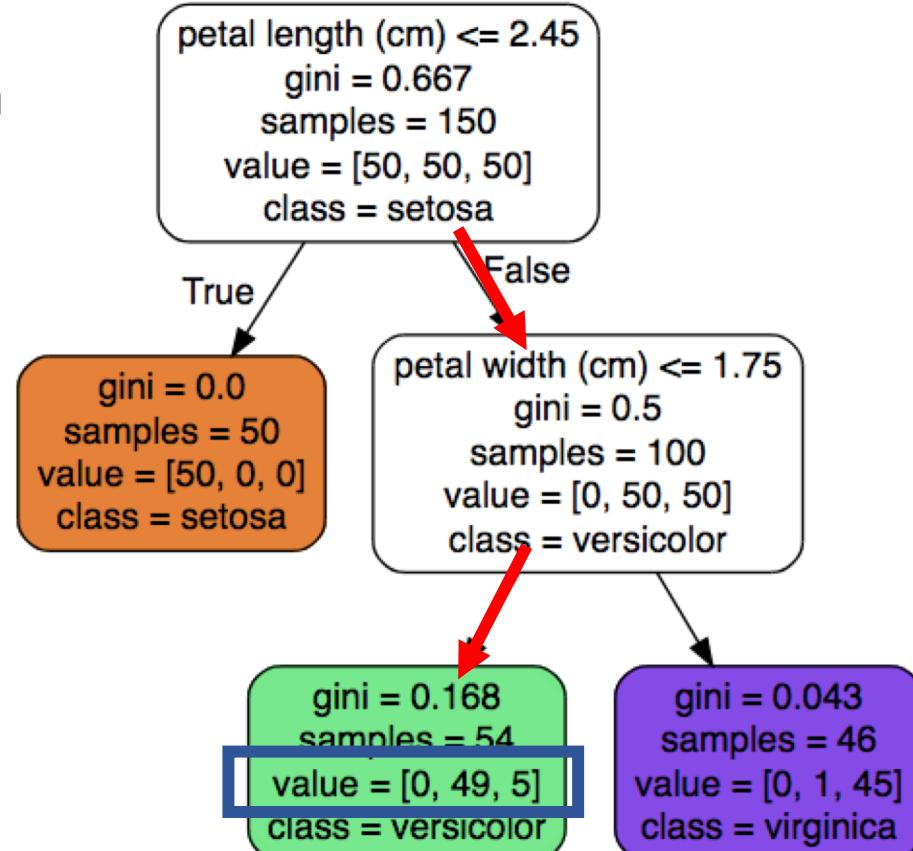
# การจำแนกด้วย decision tree

- ตัวอย่าง: จะใช้ decision tree นี้ classify ดอกกล้วยไม้ที่มีความยาวกลีบ 5cm และมีความกว้างกลีบ 1.5cm
- คำตอบ: predict class="versicolor"
- Q: จะหา class probability ต่อไปนี้
  - ความน่าจะเป็นที่ดอกนี้จะเป็น setosa?
  - ความน่าจะเป็นที่ดอกนี้จะเป็น virginica?



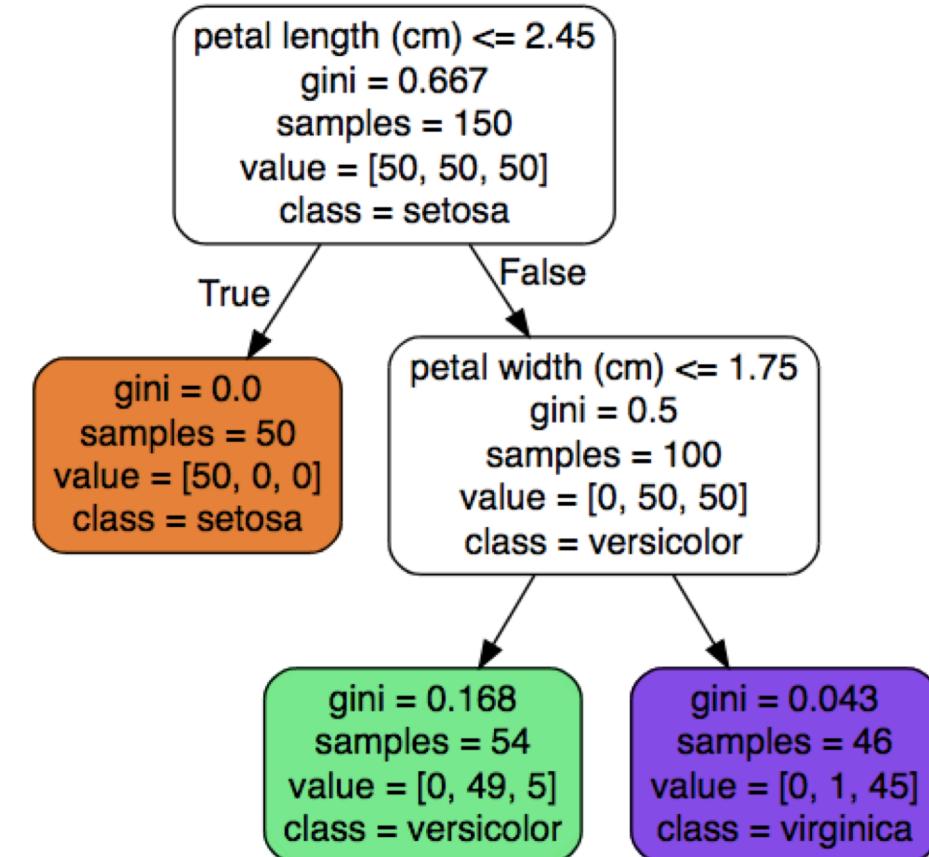
# การจำแนกด้วย decision tree

- ตัวอย่าง: จะใช้ decision tree นี้ classify ดอกกล้วยไม้ที่มีความยาวกลีบ 5cm และมีความกว้างกลีบ 1.5cm
- คำตอบ: predict class="versicolor"
- Q: จงหา class probability ต่อไปนี้
  - ความน่าจะเป็นที่ดอกนี้จะเป็น setosa?  
 $P(\text{setosa}) = 0/54 = 0\%$
  - ความน่าจะเป็นที่ดอกนี้จะเป็น virginica?  
 $P(\text{virginica}) = 5/54 = 9.3\%$



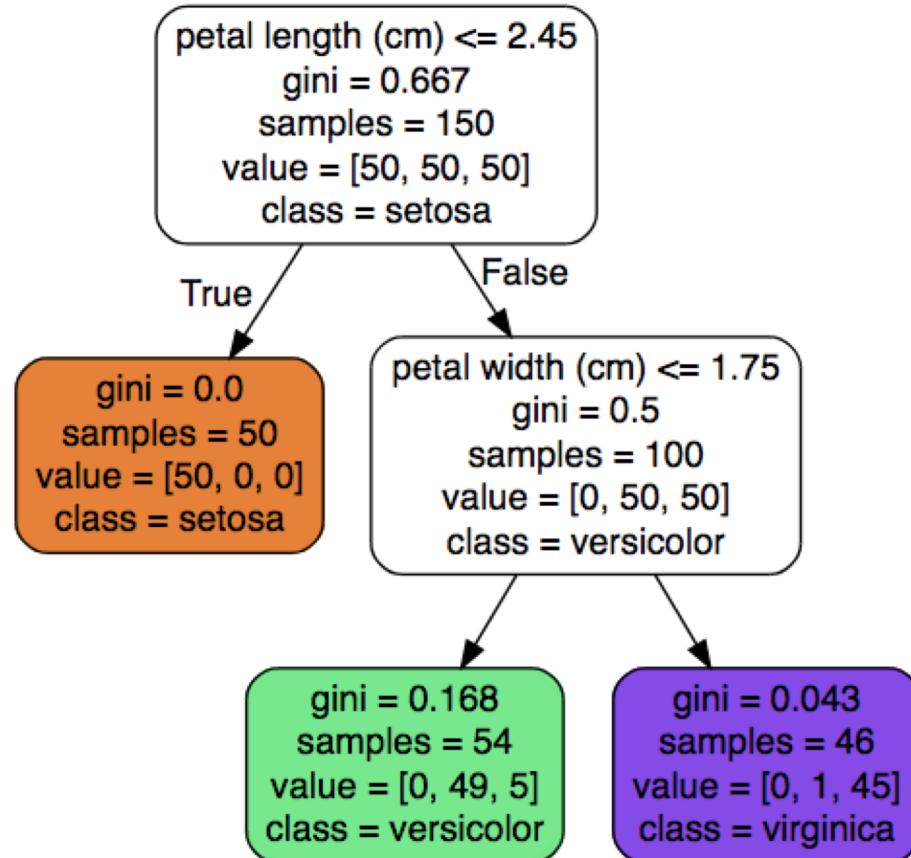
# CART Algorithm สำหรับสร้าง decision tree

- การสร้าง **decision tree** จะเริ่มจาก **root node**
- ในแต่ละ **level** เราจะเลือก  $(k, tk)$  ที่ดีที่สุดมาเป็นตัว **split**
  - attribute  $k$
  - threshold  $tk$
  - กฎ:  $k \leq tk$
- จะเลือก  $(k, tk)$  ที่ดีที่สุดได้อย่างไร?



# CART Algorithm สำหรับสร้าง decision tree

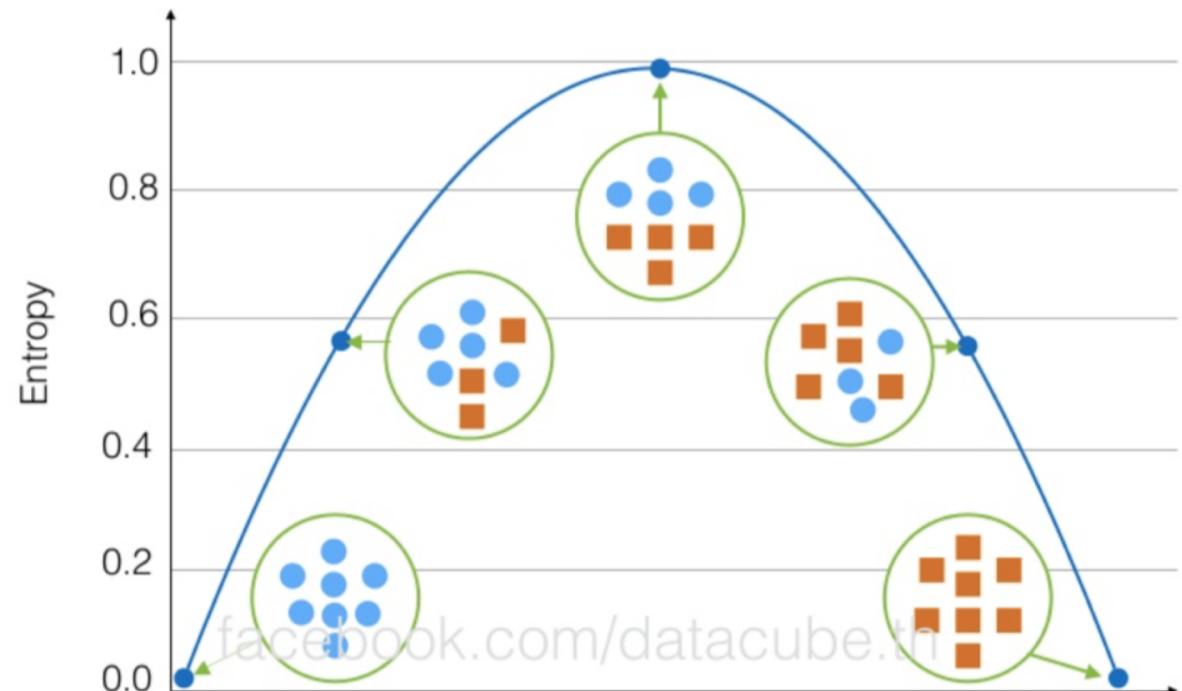
- จะเลือก  $(k, tk)$  ที่ดีที่สุดได้อย่างไร?
- เป้าหมาย: ต้องการ  $(k, tk)$  ที่ทำให้ subset ข้อมูลหลังการ split บริสุทธิ์ (pure) ที่สุด
  - = Gini ต่ำที่สุด
- ดังนั้น Cost function ที่เราต้องการ minimize คือ:
  - $$J(k, t_k) = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right}$$
- ในแต่ละชั้นต้นไม้มี อัลกอริทึม CART จะหา  $(k, tk)$  ที่ทำให้  $J$  ต่ำสุด



\*ปัญหาการหา optimal tree เป็น NP-complete ไม่สามารถหาใน poly time ได้  
จึงต้องใช้วิธีแบบ greedy คือหา  $(k, tk)$  ที่ทำให้ cost น้อยสุด ทีละชั้น แม้ว่าลงไปชั้นล่างๆ แล้วอาจทำให้ cost กลับมาสูงขึ้นได้ก็ตาม

# Choosing Splitting Criteria

- นอกจากราช Gini Impurity แล้ว ยังสามารถใช้ Entropy/Information Gain เป็นเกณฑ์ในการวัดประสิทธิภาพของ splitting attribute ได้ด้วย
  - ID3 algorithm
- splitting criterion ถือเป็น hyperparameter หนึ่งของ decision tree



$$H(T) = I_E(p_1, p_2, \dots, p_J) = - \sum_{i=1}^J p_i \log_2 p_i$$

$$\overbrace{IG(T, a)} = \overbrace{H(T)} - \overbrace{H(T|a)}$$

Information Gain      Entropy(parent)      Weighted Sum of Entropy(Children)

# Homework – Decision Tree

จะ apply decision tree กับชุดข้อมูล student\_major.csv เพื่อทำนาย major ของนักศึกษา

1. แบ่งข้อมูลออกเป็นชุด train, test ในอัตราส่วน 75:25 โดยใช้ `train_test_split` (ตัวอย่างโค้ดในสไลด์ถัดไป)
2. ใช้ข้อมูลชุด train (`X_train, y_train`) ในการ `train/validate` โมเดล decision tree
  - ใช้ `GridSearchCV` เพื่อค้นหาค่า hyperparameter ของ model (เช่น `max_depth, criterion`) ที่ให้ค่า accuracy สูงที่สุด
3. นำ model ที่ประสิทธิภาพสูงที่สุดไปใช้ทำนาย major ของนักศึกษา กับข้อมูลชุดทดสอบ (`X_test, y_test`) และแสดงผลการทำนายทั้งหมด
4. ประเมินประสิทธิภาพของโมเดลที่ได้โดยใช้ `metric` ต่อไปนี้
  - `accuracy score`
  - `confusion matrix (optional)`

\*ใน Jupyter พยายามให้ code ทั้งหมดอยู่ใน cell เดียว (หากจำเป็น สามารถใช้ได้ 4 cell max)

# Homework – Decision Tree

5. จากการทดลองในข้อ 4. ให้

- แสดง line plot ของ accuracy v.s. max\_depth จากการทดลอง ในกรณีที่ใช้ Gini criterion
- แสดง line plot ของ accuracy v.s. max\_depth จากการทดลอง ในกรณีที่ใช้ entropy criterion โดยที่ค่า accuracy ของแต่ละค่า hyperparameter จะถูกเก็บอยู่ใน GridSearchCV.cv\_results\_[“mean\_test\_score”]

6. แสดงแผนผัง decision tree ของโมเดลที่ดีที่สุด

- นำผลจาก export\_graphviz ของโมเดลที่ดีที่สุดไปแสดงบนเว็บ webgraphviz
- capture screen ภาพ decision tree เช่นในไฟล์เดอร์ของ Jupyter
- ใน Jupyter สร้าง cell ล่างสุดให้เป็นแบบ markup แล้วแทรกรูปดังกล่าวด้วย
  - ![caption](path/to/image.png)

# Homework – Decision Tree

วิธีการทำ `train_test_split`

```
from sklearn.model_selection import train_test_split  
  
X = ...  
y = ...  
X_train, X_test, y_train, y_test = train_test_split(X, y,  
test_size=0.25, random_state=33)
```

จากนั้นสามารถนำ `X_train, y_train` ไปใช้ในการฝึก และนำ `X_test, y_test` ไปใช้ในการทำนาย เพื่อวัดประสิทธิภาพได้

# Homework – Decision Tree

## วิธีส่งงาน

- ใน Jupyter Notebook ให้เขียนโค้ดทำ decision tree โดยพยายามให้ห้องทดลองใน cell เดียว (หากจำเป็น สามารถใช้ได้ 4 cell max)
- ทำการ export html โดยไปที่ File -> Download as -> HTML
- rename ชื่อไฟล์เป็น ชื่อ สกุล ภาษาไทย ไม่มีคำนำหน้า
- ส่งไฟล์ html ไปที่ <https://www.dropbox.com/request/e8rvQXyDbJfX323iuljW>
- กรอกชื่อ สกุล ภาษาไทย ไม่ใส่คำนำหน้าชื่อ นาย/นางสาว