KU x depa
MACHINE LEARNING
Training Course 2022

Decision Tree

DR. SETHAVIDH GERTPHOL

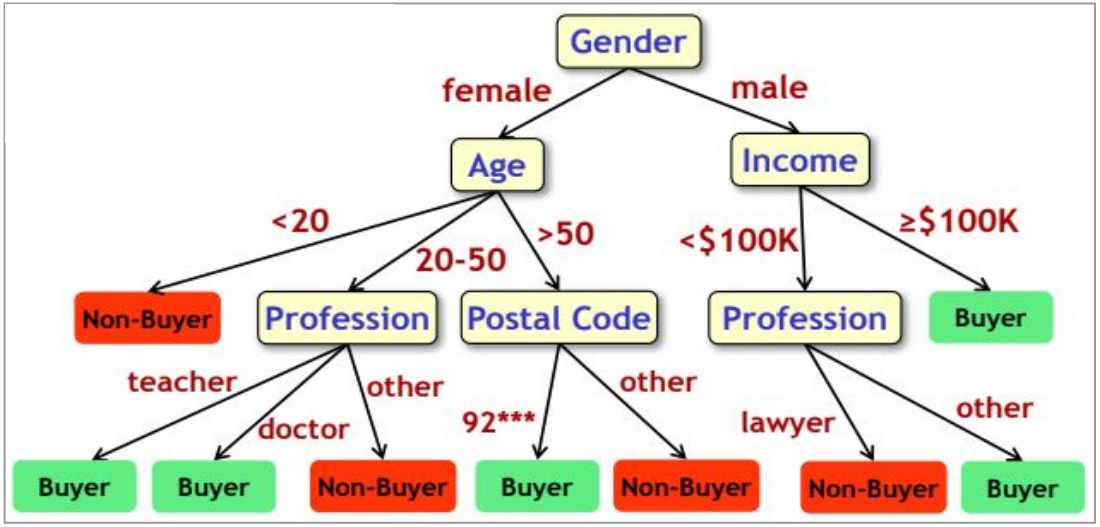
KU KASETSART UNIVERSITY depa

Today's Outline

- Decision Tree Classifier
- Overfitting
- Using Sklearn
- Decision Tree Regressor

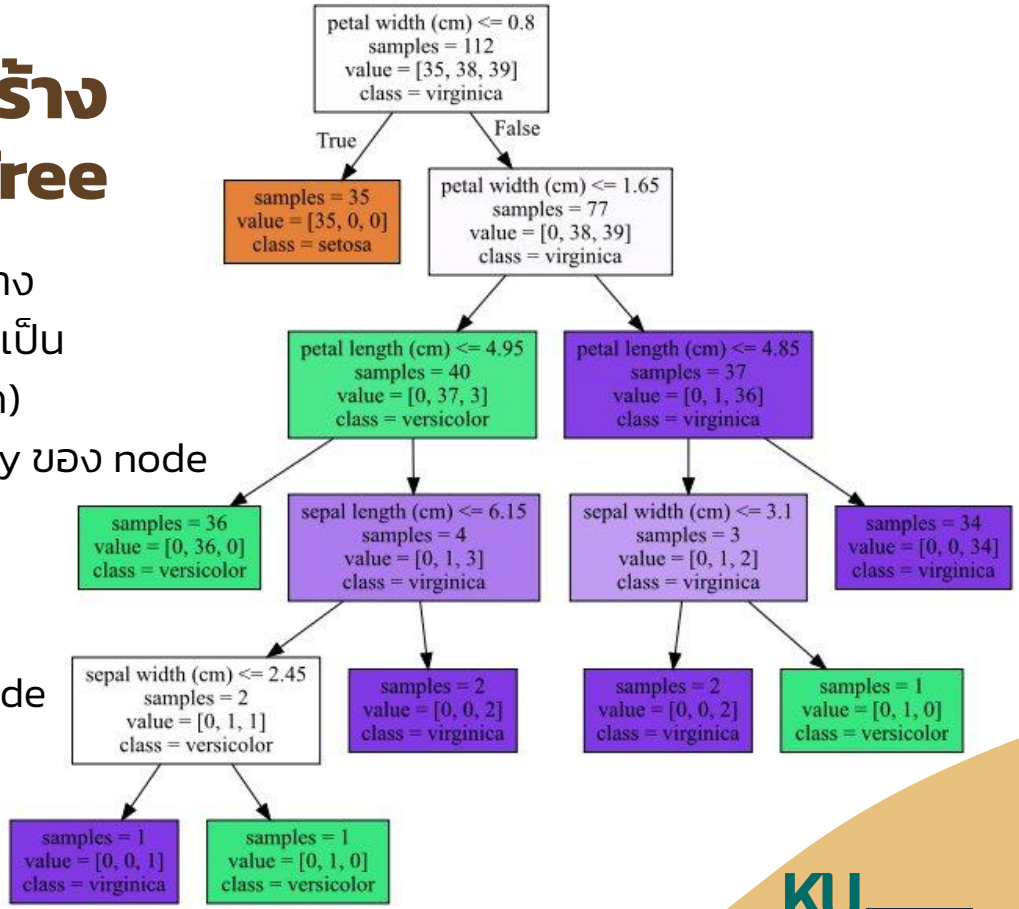
Decision Trees

- หนึ่งใน algorithm ที่เป็นที่นิยมที่สุด และเป็นพื้นฐานของ algorithm ที่ซับซ้อนขึ้น
- นำข้อมูลมาสร้างต้นไม้การตัดสินใจ
- ใช้ต้นไม้การตัดสินใจจำแนกผลเฉลยของข้อมูลในอนาคต
- **Node**: Features
- **Edges**: Feature Values
- **Leaf**: Classes / Labels



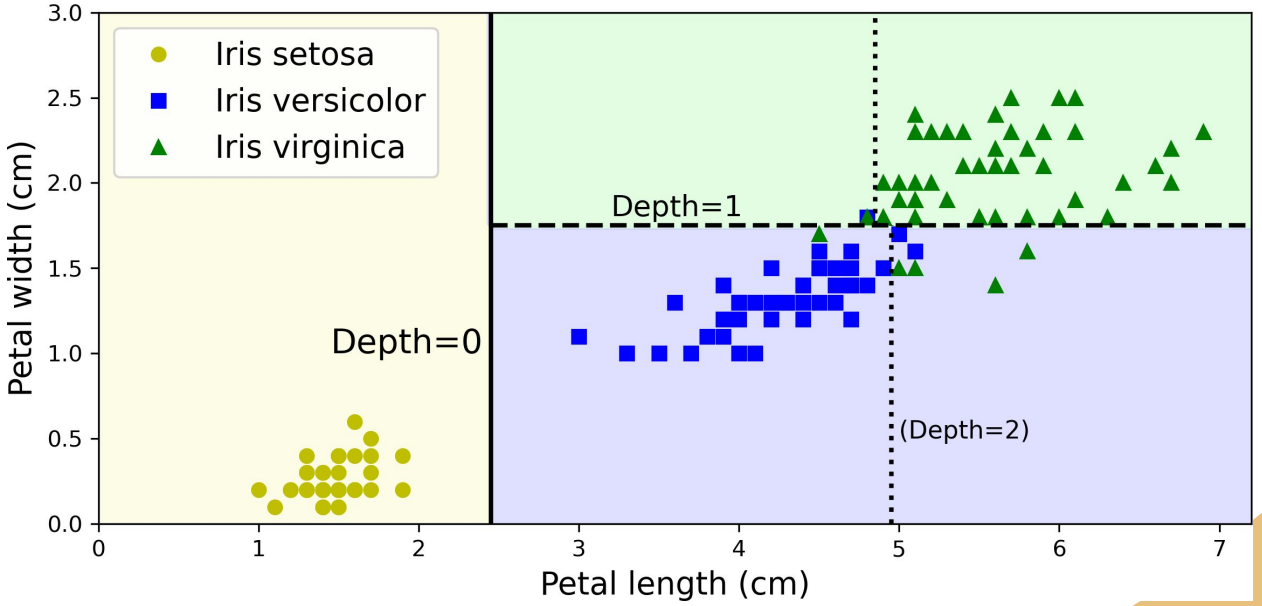
การสร้าง Decision Tree

1. เริ่มต้นจาก decision node ที่มีทุกตัวอย่าง
2. หา feature และค่าที่คัดแยกตัวอย่างออกเป็น Class ที่บริสุทธิ์ที่สุด (impurity น้อยที่สุด)
   - มีมาตรวัดสองแบบที่ใช้วัด impurity ของ node คือ gini impurity กับ entropy
   - ในทางปฏิบัติไม่ค่อยต่างกันมากนัก
3. Node ที่บริสุทธิ์แล้วจะเป็น leaf node
4. Node ที่ไม่บริสุทธิ์จะยังเป็น decision node และถูกตัดแบ่งอีกครั้ง
5. ทำจนไม่มี decision node เหลือ



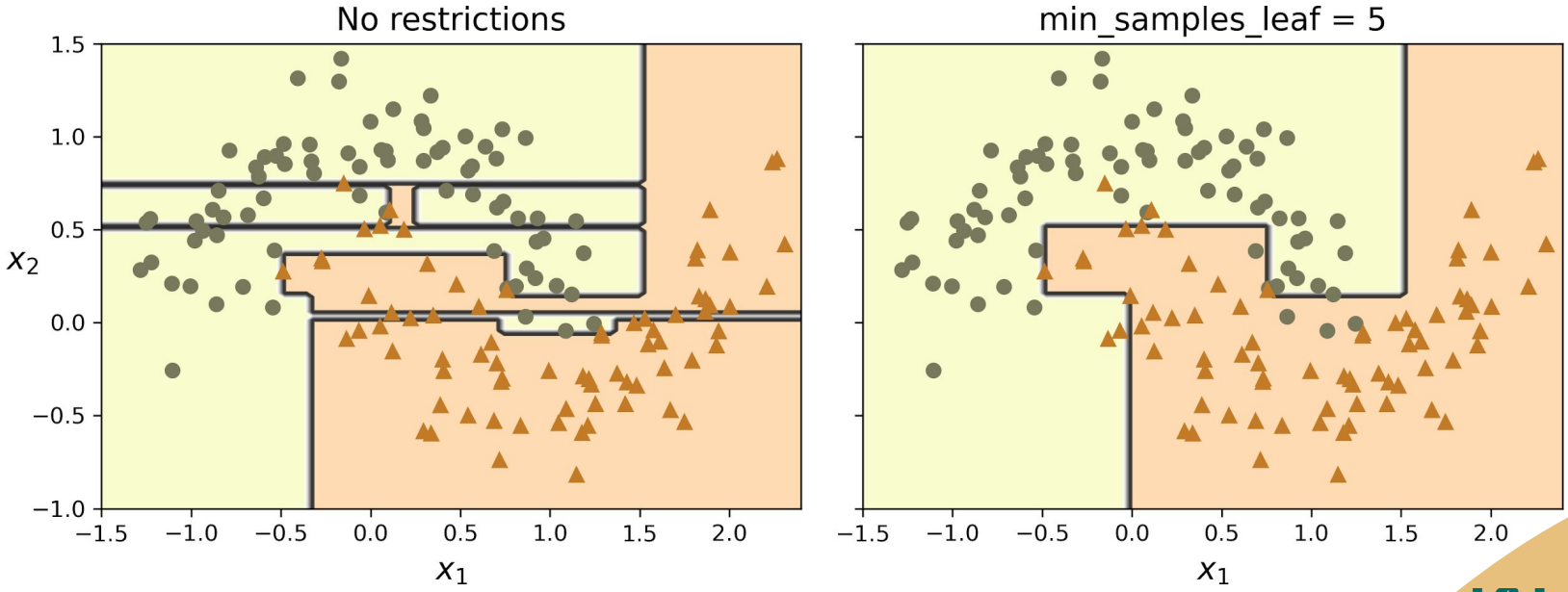
Decision Boundary

- สร้างระนาบตัดชุดข้อมูลหนึ่งระนาบต่อ decision node ระนาบจะตั้งฉากกับแกนของ feature ที่ใช้ตัดแบ่งโหนดนั้น
- ระบุความเชื่อมั่นในการทำนายได้จากสัดส่วนจำนวนตัวอย่างของ class เทียบกับตัวอย่างทั้งหมดใน leaf node



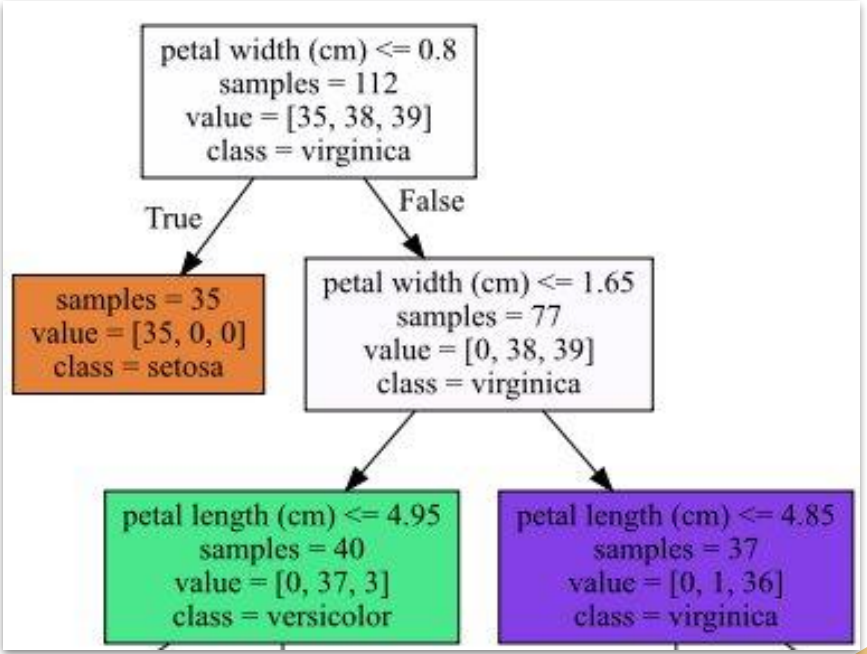
Overfitting

- Decision Tree มักจะหา feature และค่าที่จะตัดแบ่งตัวอย่าง จนกลายเป็น leaf node จนหมด ซึ่งเป็นการ overfit



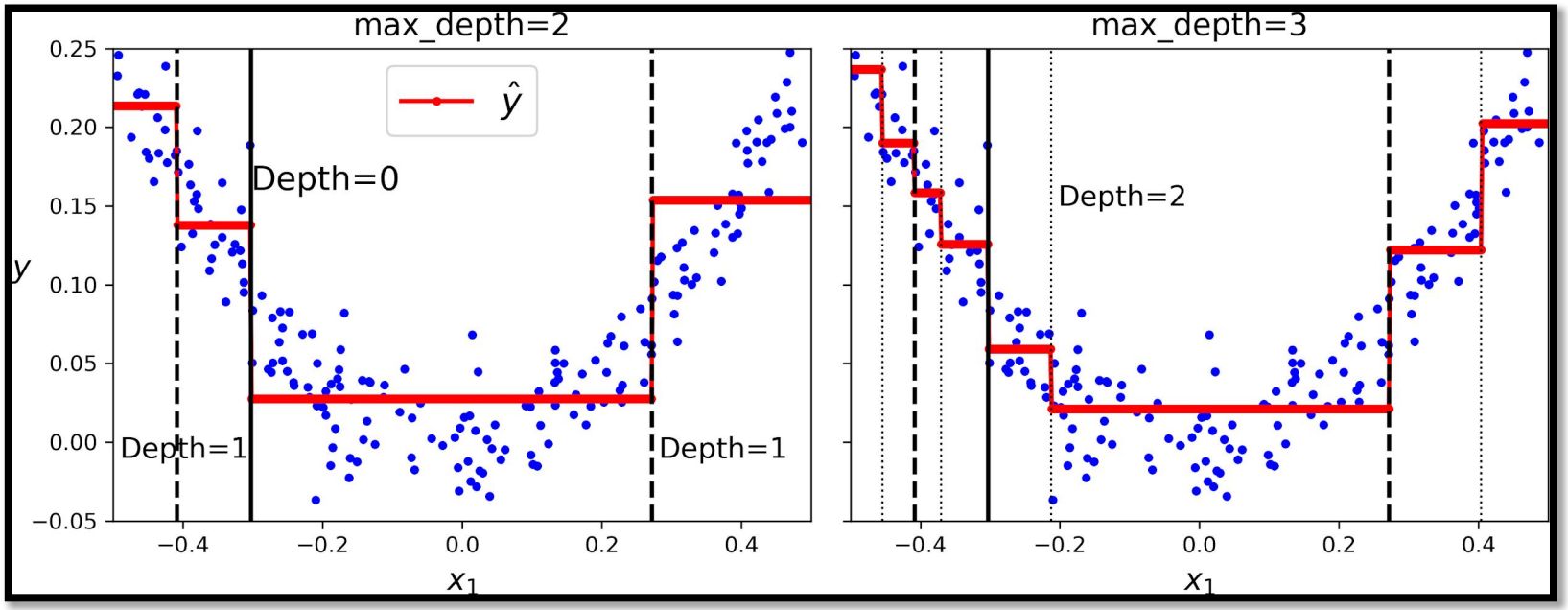
การลดการ overfit ของ Decision Tree

- จำกัดความลึกของต้นไม้
- จำกัดจำนวน leaf node ของต้นไม้
- จำกัดจำนวนตัวอย่างที่น้อยที่สุดที่จะเป็น leaf node ได้
- จำกัดจำนวนตัวอย่างที่น้อยที่สุดใน decision node ที่จะถูกตัดแบ่ง
- ตัดกิ่ง (prune) คือรวม node ใน subtree ทั้งหมดกลับมาเป็น leaf node



Prediction

- คำทำนายคือค่าเฉลี่ยของตัวอย่างทุกตัวใน leaf node (เส้นแดง)



Decision Tree in scikit-learn

```
[2] import pandas as pd
```

```
[3] from sklearn.preprocessing import OneHotEncoder, MinMaxScaler
    from sklearn.compose import make_column_transformer
```

Load training data

```
[4] df = pd.read_csv("./drive/MyDrive/Datasets/01-census-income.csv")
    X_with_cat = df[ ['age', 'edu num', 'marital status', 'sex'
    , 'captial-gain', 'capital-loss', 'hours-per-week'] ]
    y = df[ ['label'] ]

    df2 = pd.read_csv("/content/drive/MyDrive/Datasets/02-future-census.csv")
    unseen_X = df2[ ['age', 'edu num', 'marital status', 'sex'
    , 'captial-gain', 'capital-loss', 'hours-per-week'] ]
    unseen_y = df2[ ['label'] ]
```

Load test data

Decision Tree Classifier

```
[5]  transformer = make_column_transformer(
         ( OneHotEncoder(), ['marital status','sex'] ),
         remainder='passthrough'
     )
     X_transformed = transformer.fit_transform(X_with_cat)
     unseen_X_transformed = transformer.transform(unseen_X)
```

Just one-hot encoding. No need for Scaler.

Set other columns to "passthrough"

```
[7]  from sklearn.tree import DecisionTreeClassifier
```

Import

```
[8]  dtree = DecisionTreeClassifier()
     dtree.fit(X_transformed, y)
     dtree.score(unseen_X_transformed, unseen_y)
```

```
0.8351760851760852
```

Create object, fit, and score

Feature importance

```
[13] transformer.get_feature_names_out()

array(['onehotencoder__marital status_ Divorced',
       'onehotencoder__marital status_ Married-AF-spouse',
       'onehotencoder__marital status_ Married-civ-spouse',
       'onehotencoder__marital status_ Married-spouse-absent',
       'onehotencoder__marital status_ Never-married',
       'onehotencoder__marital status_ Separated',
       'onehotencoder__marital status_ Widowed',
       'onehotencoder__sex_ Female', 'onehotencoder__sex_ Male',
       'remainder__age', 'remainder__edu num', 'remainder__captial-gain',
       'remainder__capital-loss', 'remainder__hours-per-week'],
      dtype=object)
```

ดูชื่อ attribute ของตารางที่ transform แล้วจาก get_feature_names_out()

Feature importance คำนวณจากค่า impurity ที่ลดลงเมื่อทำการแบ่ง node ด้วย feature นั้น

```
[17] dtree.feature_importances_

array([0.00452221, 0.0014887 , 0.27406815, 0.00132442, 0.00545897,
       0.00210979, 0.00136966, 0.00779005, 0.00984762, 0.16981044,
       0.17950391, 0.16074455, 0.06057386, 0.12138768])
```

ดูความสำคัญของ feature จาก attribute feature_importance_

Other options

- `criterion`: กำหนดมาตรวัด impurity ว่าจะเป็น 'gini' หรือ 'entropy'
- `max_depth`: สร้าง Tree อย่างมากสุดเท่านี้ชั้น
- `min_samples_split`: node ใดที่มีตัวอย่างน้อยกว่านี้จะไม่ถูกแบ่ง
- `min_samples_leaf`: leaf node ต้องมีจำนวนตัวอย่างอย่างน้อยเท่านี้
- `max_features`: จะพิจารณากี่ feature ในการตัดแบ่ง node
- `max_leaf_nodes`: สร้าง leaf node ได้มากสุดเท่านี้
- `min_impurity_decrease`: การตัดแบ่ง node ต้องลด impurity ได้อย่างน้อยเท่านี้
- `ccp_alpha`: ระบุค่า cost-complexity. Subtree ที่มีค่า ccp มากที่สุด ที่น้อยกว่า ค่านี้ จะโดนยุบรวมเป็น leaf node

Hyperparameters tuning (a bit)

```
[31] param_grid = {'max_depth':[5,10],
                   'min_samples_split':[30,60],
                   'min_samples_leaf':[100,200]
                   }
```

```
[32] dtree_search = DecisionTreeClassifier()
     grid_search = GridSearchCV(dtree_search, param_grid, cv=5)
```

```
[33] grid_search.fit(X_transformed, y)
     grid_search.best_params_, grid_search.best_score_
```

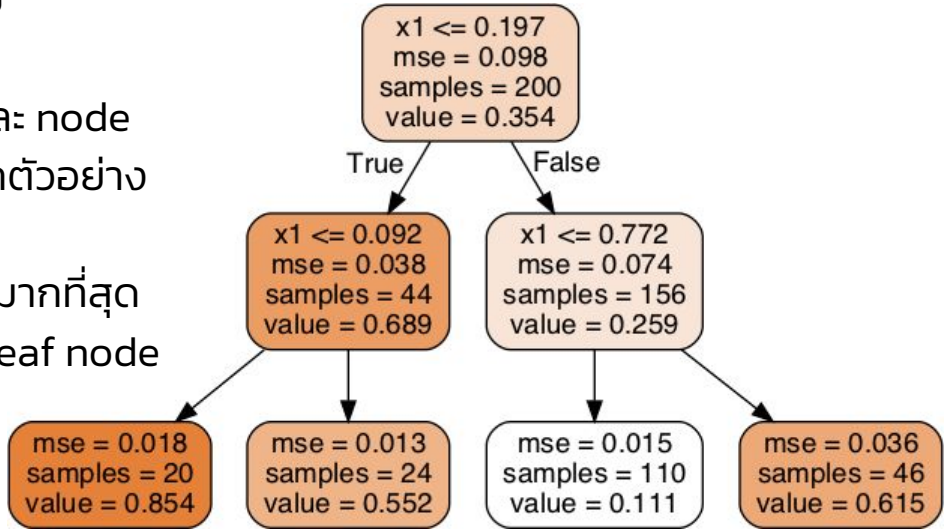
```
({'max_depth': 10, 'min_samples_leaf': 200, 'min_samples_split': 30},
 0.8534865172566695)
```

```
[34] grid_search.score(unseen_X_transformed, unseen_y)
```

```
0.8530917280917281
```

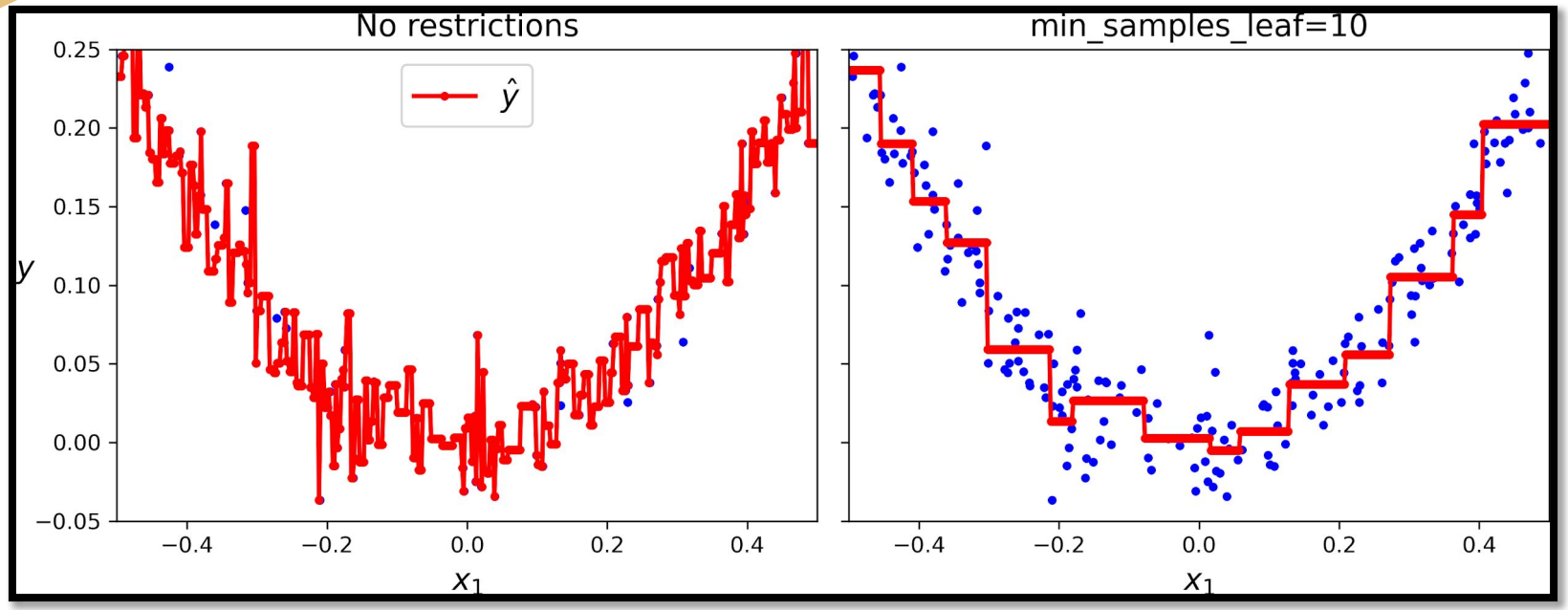
Decision Tree Regressor

- ใช้วิธีการสร้างและทำนายเหมือนกับ Decision Tree Classifier
- มาตรวัดความ"ไม่สมบูรณ์"ของแต่ละ node คือค่า mean square error ของทุกตัวอย่างเทียบกับค่าเฉลี่ย
- ตัดแบ่งโหนดโดยให้ค่า mse ลดลงมากที่สุด
- ใช้ค่าเฉลี่ยของตัวอย่างทั้งหมดใน leaf node เป็นค่าทำนาย



Overfitting

- Decision Tree Regressor เกิดการ overfit ได้ง่าย ควรปรับ hyperparameter ลดการ overfit



DecisionTreeRegressor in sklearn

Load data

encoding 'tis620' คือรหัส encoding (อันเก่า) ของภาษาไทย

```
[44] df3 = pd.read_csv("/content/drive/MyDrive/Datasets/03-cities.csv", encoding='tis620')
     df3.head()
```

	city	country	latitude	longitude	temperature	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unr
0	Aalborg	Denmark	57.03	9.92	7.52	NaN	NaN	NaN	NaN	
1	Aberdeen	United Kingdom	57.17	-2.08	8.10	NaN	NaN	NaN	NaN	
2	Abisko	Sweden	63.35	18.83	0.20	NaN	NaN	NaN	NaN	
3	Adana	Turkey	36.99	35.32	18.67	NaN	NaN	NaN	NaN	
4	Albacete	Spain	39.00	-1.87	12.62	NaN	NaN	NaN	NaN	

5 rows × 23 columns

DecisionTreeRegressor in sklearn

```
[41] from sklearn.model_selection import train_test_split
     from sklearn.tree import DecisionTreeRegressor
```

import

```
[45] X_train, X_test, y_train, y_test = train_test_split(df3[['latitude','longitude']], df3[['temperature']])
```

Split train-test data

```
[47] dt_reg = DecisionTreeRegressor()
     dt_reg.fit(X_train, y_train)
     dt_reg.score(X_test, y_test)
```

fit และ score ค่า score คือ R^2

```
0.7089025339912114
```

```
dt_reg.score(X_train, y_train)
```

```
1.0
```

เช็ค R^2 ของ training set จะเห็นว่าได้คะแนนเต็ม แสดงว่า overfit มาก

ลองวัดผลด้วยมาตรวัดอื่น

```
[51] from sklearn.metrics import mean_absolute_percentage_error, mean_squared_error, mean_absolute_error, r2_score
```

```
[52] y_predicted = dt_reg.predict(X_test)
```

```
[56] r2_score(y_test, y_predicted)
```

```
0.7089025339912114
```

```
[53] mean_absolute_percentage_error(y_test, y_predicted)
```

```
0.5476670334236472
```

```
[58] mean_squared_error(y_test, y_predicted)
```

```
4.600701851851852
```

```
[55] mean_absolute_error(y_test, y_predicted)
```

```
1.346851851851852
```

Reduce overfitting

```
[69] param_reg = {'max_depth':[3,4,5,10,20],
                  'min_samples_split':[5,10,15,20],
                  'min_samples_leaf':[5,10,20,30]
                  }
```

```
[70] gs_reg = GridSearchCV(DecisionTreeRegressor(), param_reg, cv=5)
     gs_reg.fit(X_train, y_train)
     gs_reg.best_params_, gs_reg.best_score_
```

```
({'max_depth': 10, 'min_samples_leaf': 5, 'min_samples_split': 5},
 0.7010082780849783)
```

```
[71] y_predicted_gs = gs_reg.predict(X_test)
     gs_reg.score(X_test, y_test)
```

```
0.803616203305658
```

```
[72] mean_absolute_percentage_error(y_test,
```

```
0.37183643661175797
```

```
[73] mean_squared_error(y_test, y_predicted_gs)
```

```
3.1037827622246597
```

```
[74] mean_absolute_error(y_test, y_predicted_gs)
```

```
1.1801645355673136
```

ข้อดีข้อเสียของ Decision Tree

ข้อดี

- เข้าใจง่าย ตีความการทำนายได้ง่าย
- ไม่ต้องทำ feature scaling
- ใช้ feature ที่เป็น category กับ numerical ผสมกันได้
- สร้างโมเดลได้ค่อนข้างรวดเร็ว
- ใช้เวลาในการทำนายน้อย

ข้อเสีย

- มักจะ overfit ถึงแม้จะพยายามลดแล้วก็ตาม
- อาจต้องใช้ต้นไม้หลายต้นและการสุ่มเลือก feature เพื่อให้โมเดล generalize ได้ (random forest)