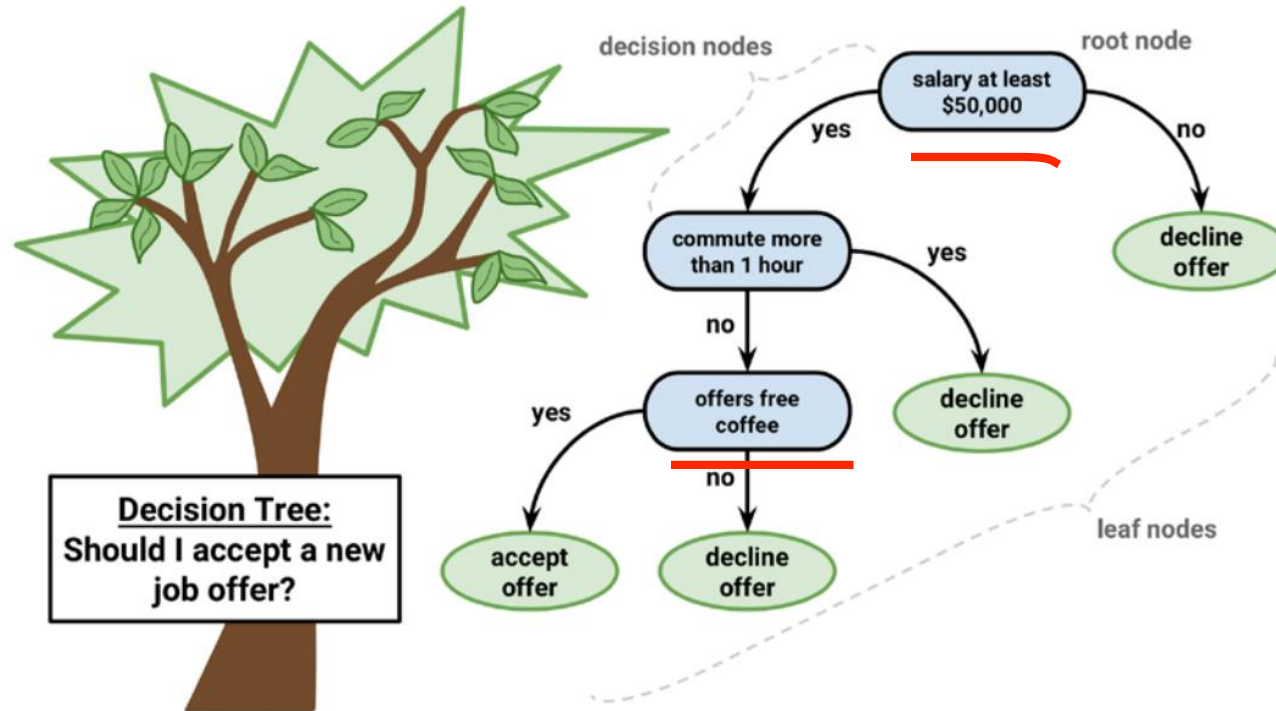# 2022년 가을학기

앙상블 러닝

앙상블 러닝

CONTENTS

What is Decision Tree?

Decision Tree

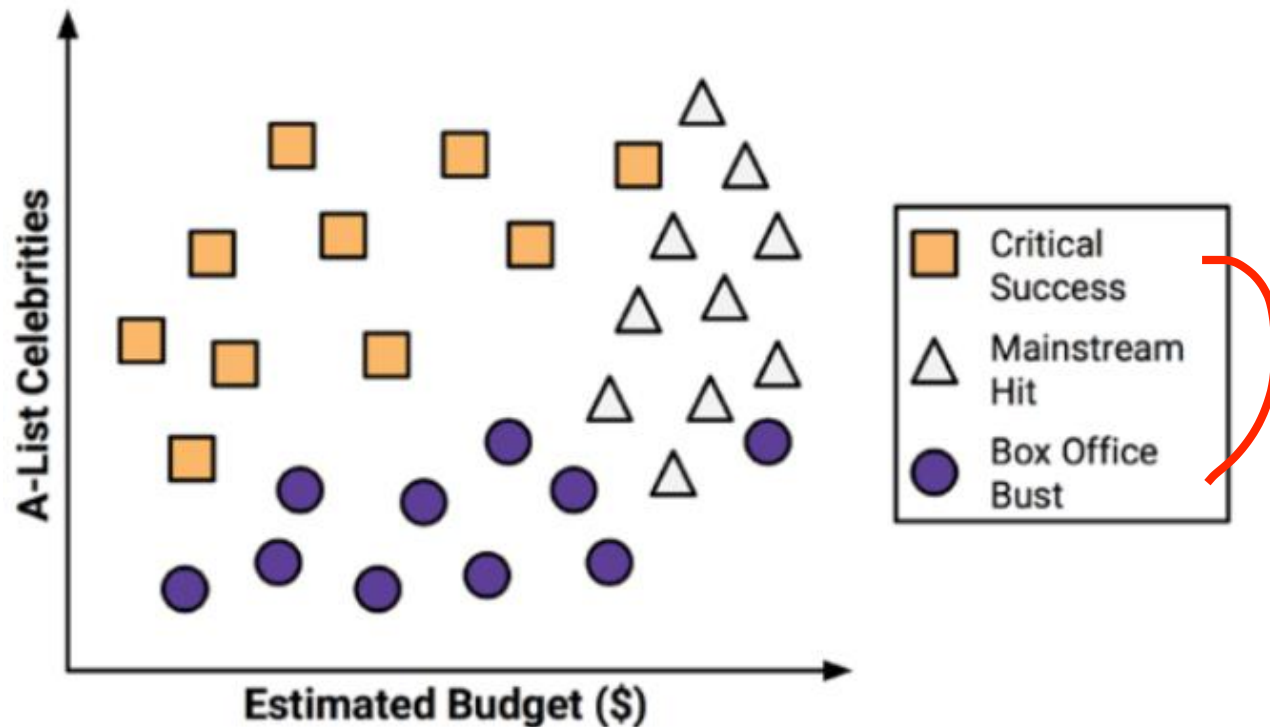❖ **What is Decision Tree?**
- A tree based classification

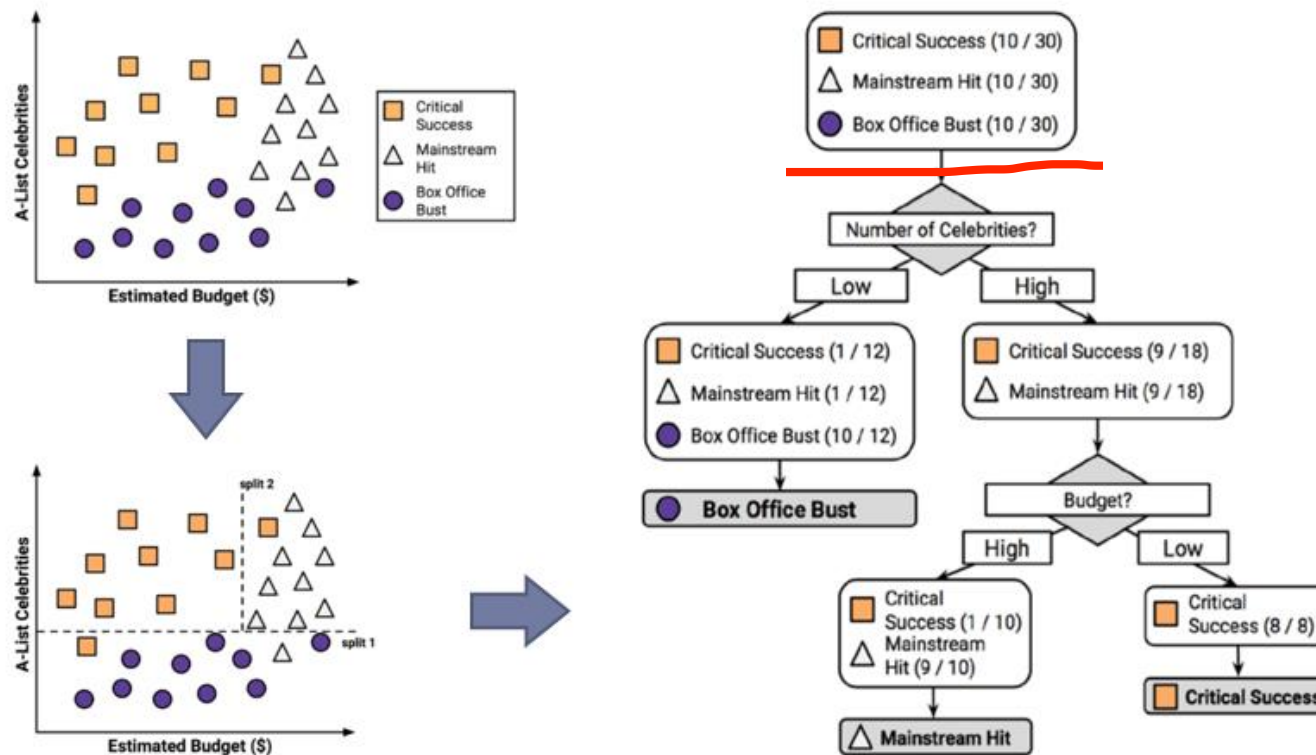# Decision Tree

❖ **Divide and Conquer**

- Decision trees are built using a recursive partitioning
  - Commonly known as divide and conquer

Decision Tree

❖ **Divide and Conquer**

▪ Decision trees are built using a recursive partitioning

• Commonly known as divide and conquer

Decision Tree

❖ **Decision Tree in Python**

 ▪ Loading dataset

```
#Import scikit-learn dataset library
from sklearn.datasets import load_iris

#Load dataset
iris = load_iris()

# print the label species(setosa, versicolor,virginica)
print(iris.target_names)

# print the names of the four features
print(iris.feature_names)
```

```
['setosa' 'versicolor' 'virginica']
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

Decision Tree

❖ **Decision Tree in Python**

▪ Split into train and test dataset

```
# Import train_test_split function
from sklearn.model_selection import train_test_split

train_points = iris.data
train_labels = iris.target

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(train_points, train_labels,
                                            test_size=0.2, random_state=15)
```

Decision Tree

❖ **Decision Tree in Python**

  ▪ Training with Decision Tree

```python
#Import Decision Tree
from sklearn.tree import DecisionTreeClassifier

#Create a Gaussian Classifier
classifier = DecisionTreeClassifier(random_state=0)

#Train the model using the training sets
classifier.fit(X_train,y_train)

guesses = classifier.predict(X_test)
```

Decision Tree

❖ **Decision Tree in Python**

▪ Check out accuracy

```
#Import scikit-learn metrics module for checking confusion matrix and accuracy
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

# Confusion matrix
print("Confusion Matrix: ", confusion_matrix(y_test, guesses))

# Model Accuracy, how often is the classifier correct?
print("Accuracy:", accuracy_score(y_test, guesses))
```

```
Confusion Matrix:    [[ 8  0  0]
                      [ 0  9  2]
                      [ 0  0 11]]

Accuracy: 0.9333333333333333
```
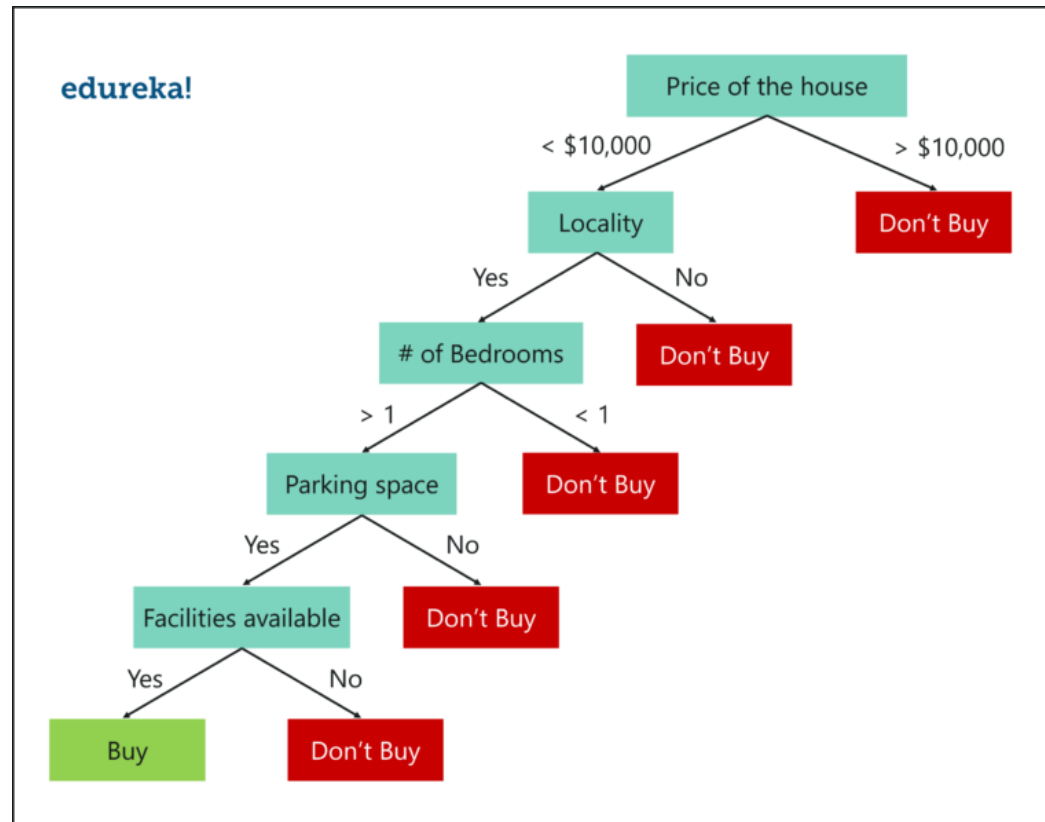
B

What is Random Forest

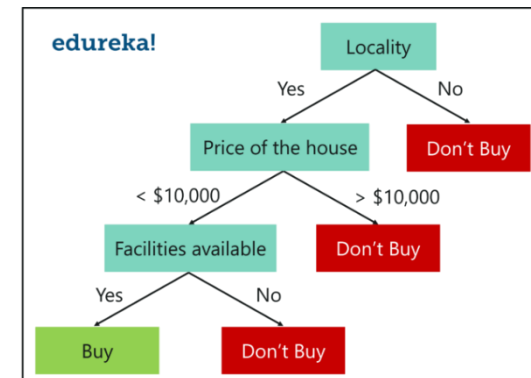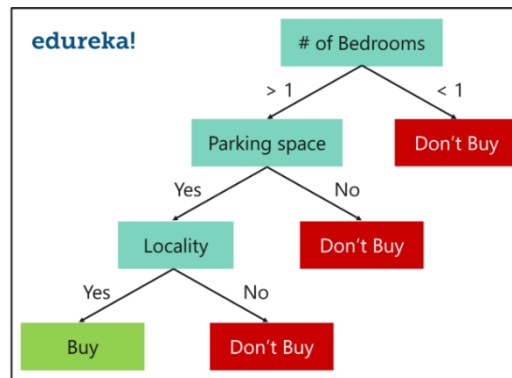# Random Forest

❖ **Problems of Decision Tree?**

 ▪ Built on the entire dataset by using the same parameters and the same features

Random Forest

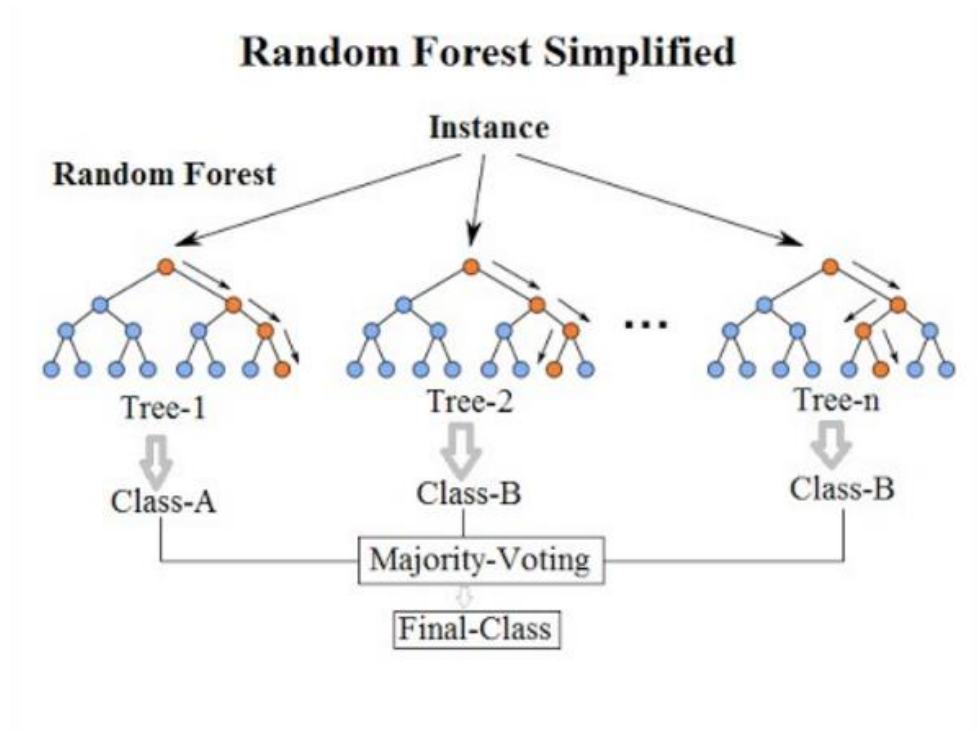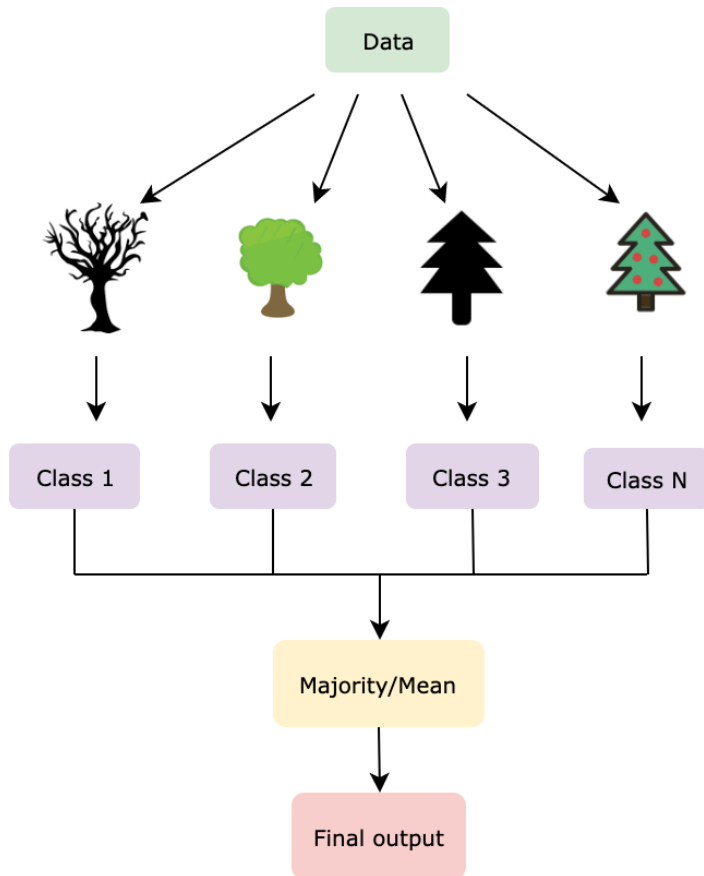❖ **What is Random Forest?**

- Is an ensemble of decision trees

- Randomly selects a set of parameters and creates a decision tree for each set of chosen parameters

- Only a selected a set of features are taken into consideration
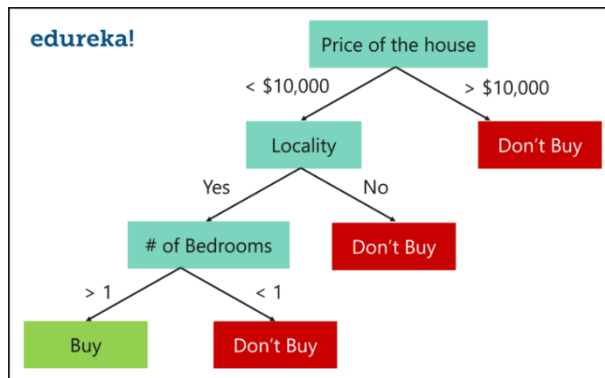
Random Forest

❖ **What is Random Forest?**

▪ Based on majority voting for every predicted result

Random Forest

❖ **Difference between Decision Tree and Random Forest**
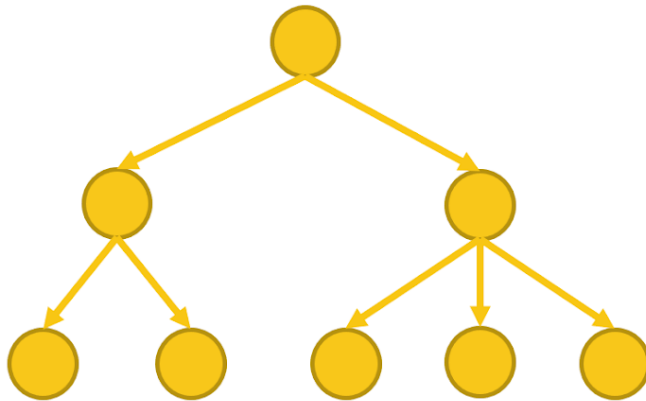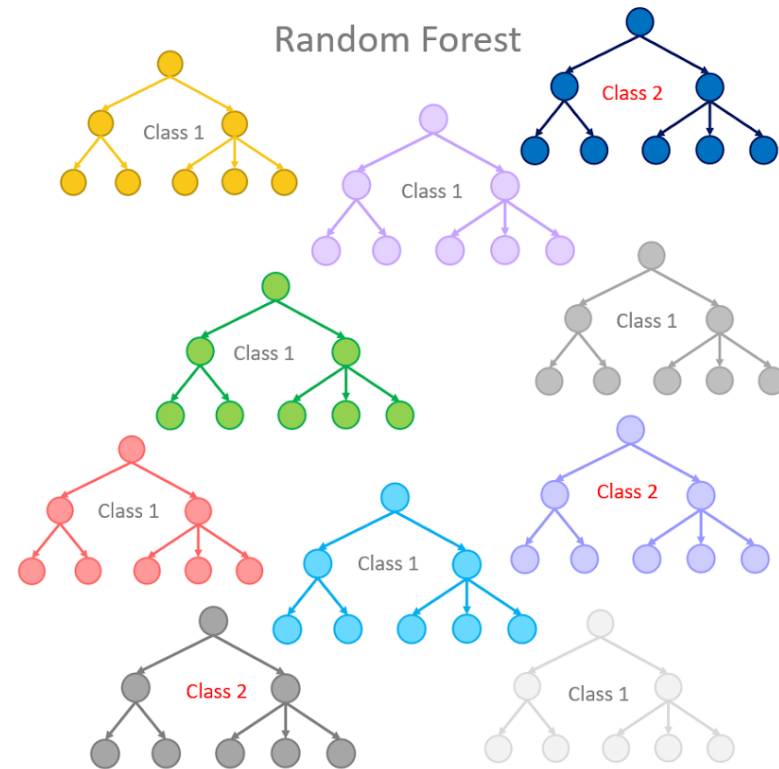
Random Forest

❖ **Random Forest in Python**

▪ Loading dataset

```
#Import scikit-learn dataset library
from sklearn.datasets import load_iris

#Load dataset
iris = load_iris()

# print the label species(setosa, versicolor,virginica)
print(iris.target_names)

# print the names of the four features
print(iris.feature_names)
```

```
['setosa' 'versicolor' 'virginica']
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

Random Forest

❖ **Random Forest in Python**

- Split into train and test dataset

```
# Import train_test_split function
from sklearn.model_selection import train_test_split

train_points = iris.data
train_labels = iris.target

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(train_points, train_labels,
                                            test_size=0.2, random_state=15)
```

Random Forest

❖ **Random Forest in Python**

- Training with Random Forest

```
#Import Random Forest
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
classifier = RandomForestClassifier(random_state=0, n_estimators=10)

#Train the model using the training sets y_pred=clf.predict(X_test)
classifier.fit(X_train,y_train)

guesses = classifier.predict(X_test)
```

Decision Tree

❖ **Decision Tree in Python**

▪ Check out accuracy

```
#Import scikit-learn metrics module for checking confusion matrix and accuracy
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

# Confusion matrix
print("Confusion Matrix: ", confusion_matrix(y_test, guesses))

# Model Accuracy, how often is the classifier correct?
print("Accuracy:", accuracy_score(y_test, guesses))
```

```
Confusion Matrix:    [[ 8  0  0]
                      [ 0 13  0]
                      [ 0  2  7]]

Accuracy: 0.9333333333333333
```

C

앙상블 러닝 실습

앙상블 러닝 실습

❖ **Dataset**
  ▪ pima-indians-diabetes.csv

❖ **Practice**
  ▪ Step 1: Loading dataset
  ▪ Step 2: Normalize and split into train and test dataset
  ▪ Step 3: Training with Decision Tree/Random Forest
  ▪ Step 4: Checking accuracy
  ▪ Step 5: Improving accuracy
  ▪ Step 6: Visualize accuracy

앙상블 러닝 실습

❖ **Step 1: Loading dataset**

```
import pandas as pd

data = pd.read_csv("D:/pima-indians-diabetes.csv")

print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   col1    768 non-null    int64
 1   col2    768 non-null    int64
 2   col3    768 non-null    int64
 3   col4    768 non-null    int64
 4   col4.1  768 non-null    int64
 5   col5    768 non-null    float64
 6   col6    768 non-null    float64
 7   col7    768 non-null    int64
 8   class   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

앙상블 러닝 실습

❖ **Step 2:Normalize and split into train and test dataset**

```
train_points = data.drop("class", axis=1)
train_labels = data["class"]

# standardize the dataset
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
train_points_scaled = scaler.fit_transform(train_points)

# Import train_test_split function
from sklearn.model_selection import train_test_split

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(train_points_scaled, train_labels,
                                                    test_size=0.2, random_state=5)
```

앙상블 러닝 실습

❖ **Step 3: Training with Random Forest**

```
#Import Random Forest
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
classifier = RandomForestClassifier(random_state=0, n_estimators=3)

#Train the model using the training sets y_pred=clf.predict(X_test)
classifier.fit(X_train,y_train)

guesses = classifier.predict(X_test)
```

앙상블 러닝 실습

❖ **Step 4: Checking accuracy**

```
#Import scikit-learn metrics module for checking confusion matrix and accuracy
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

# Confusion matrix
print("Confusion Matrix: ", confusion_matrix(y_test, guesses))

# Model Accuracy, how often is the classifier correct?
print("Accuracy:", accuracy_score(y_test, guesses))
```

Confusion Matrix: [[83 17]
*   [26 28]]*

Accuracy: 0.7207792207792207

앙상블 러닝 실습

❖ **Step 5: Improving accuracy**

```
#Import Random Forest
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
classifier = RandomForestClassifier(random_state=0, n_estimators=5)

#Train the model using the training sets y_pred=clf.predict(X_test)
classifier.fit(X_train,y_train)

guesses = classifier.predict(X_test)
```

Confusion Matrix: [[83 17]
                                 [22 32]]

Accuracy: 0.7467532467532467

앙상블 러닝 실습

❖ **Step 5: Improving accuracy**

```
#Import Random Forest
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
classifier = RandomForestClassifier(random_state=0, n_estimators=10)

#Train the model using the training sets y_pred=clf.predict(X_test)
classifier.fit(X_train,y_train)

guesses = classifier.predict(X_test)
```

Confusion Matrix: [[89 11]
[23 31]]

Accuracy: 0.7792207792207793

앙상블 러닝 실습

❖ **Step 5: Improving accuracy**

```
#Import Random Forest
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
classifier = RandomForestClassifier(random_state=0, n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
classifier.fit(X_train,y_train)

guesses = classifier.predict(X_test)
```

Confusion Matrix: [[87 13]
*   [19 35]]*

Accuracy: 0.7922077922077922

앙상블 러닝 실습

❖ **Step 6: Visualize accuracy**

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

n_range = range(1, 100)
accuracy_scores = []

for n in n_range:
    classifier = RandomForestClassifier(random_state=0, n_estimators=n)
    classifier.fit(X_train, y_train)
    guesses = classifier.predict(X_test)
    accuracy_scores.append(accuracy_score(y_test, guesses))
print(accuracy_scores)

import matplotlib.pyplot as plt

plt.plot(n_range, accuracy_scores)
plt.xlabel('Value of n for Random Forest')
plt.ylabel('Testing Accuracy')
plt.show()
```
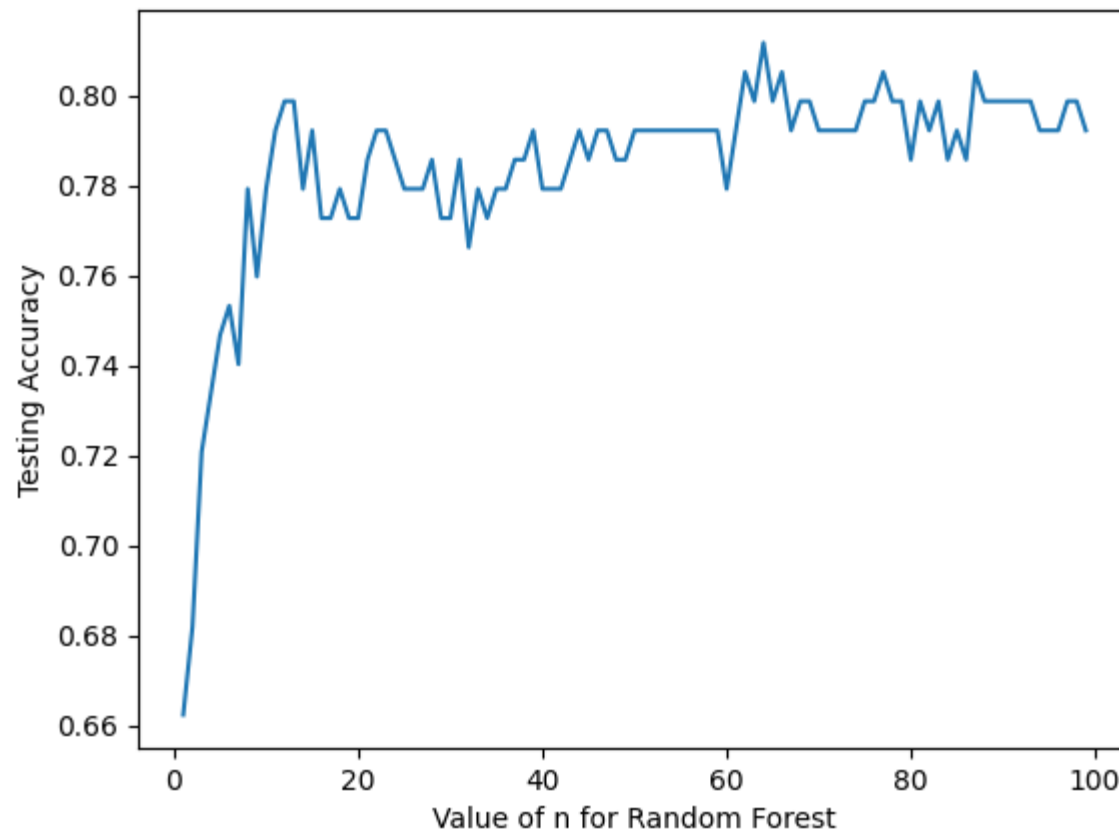
앙상블 러닝 실습

❖ Step 6: Visualize accuracy

Final Task

❖ **Submit your source code for the following task:**

1. Try all source code in the lecture

❖ **Submission: source code, result screenshots and result explanation**

감사합니다!