Problem 1

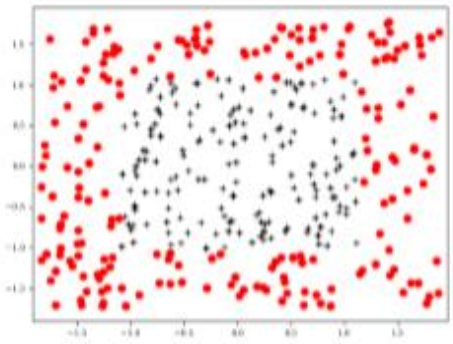


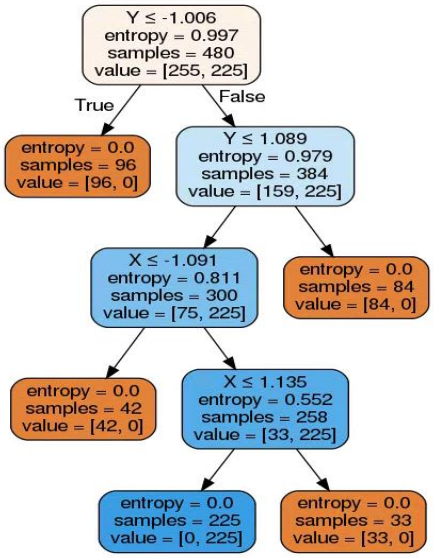
Figure 1: A 2-class classification problem

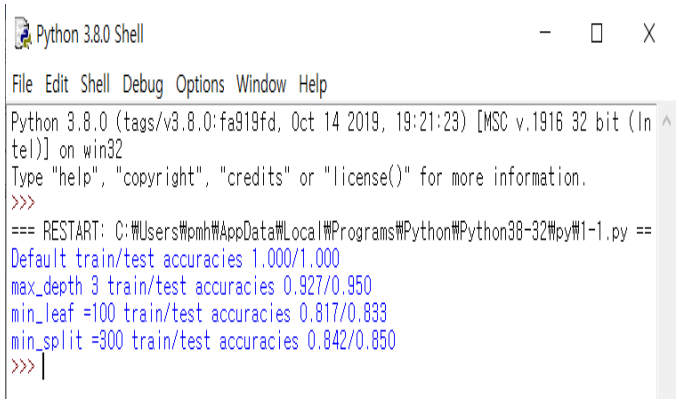
Answer the following question on a 2-class classification problem. The training set is given in Figure 1. Check out the accuracy of a training and testing sets when 80% data are used for training and the rest for testing. The file name is dstest.txt.

(a) Build and evaluate decision trees in terms of max_depth, min_samples_split, and min_samples_leaf.

```
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn import tree
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score
import pandas as pd
import numpy as np
import os
```

```
report1 = pd.read_csv("/Users/pmh/AppData/Local/Programs/Python/Python38-32/py/dstest.csv")

X = np.array(pd.DataFrame(report1, columns=['X','Y']))

y = np.array(pd.DataFrame(report1, columns=['C']))

X_train, X_test, y_train, y_test = train_test_split(X,y, train_size = 0.8)


dt_clf = DecisionTreeClassifier(criterion='entropy')

dt_clf = dt_clf.fit(X_train, y_train)

y_train_pred = dt_clf.predict(X_train)

y_test_pred = dt_clf.predict(X_test)

tree_train = accuracy_score(y_train, y_train_pred)

tree_test = accuracy_score(y_test, y_test_pred)

print('Default train/test accuracies %.3f/%.3f' % (tree_train, tree_test))


dt_clf = DecisionTreeClassifier(criterion='entropy', max_depth=3)

dt_clf = dt_clf.fit(X_train, y_train)

y_train_pred = dt_clf.predict(X_train)

y_test_pred = dt_clf.predict(X_test)

tree_train = accuracy_score(y_train, y_train_pred)

tree_test = accuracy_score(y_test, y_test_pred)

print('max_depth 3 train/test accuracies %.3f/%.3f' % (tree_train, tree_test))


dt_clf = DecisionTreeClassifier(criterion='entropy', min_samples_leaf=100)

dt_clf = dt_clf.fit(X_train, y_train)

y_train_pred = dt_clf.predict(X_train)

y_test_pred = dt_clf.predict(X_test)
```





(b) Build and evaluate an AdaBoost ensemble where max_depth is 2 and n_estimators varies from 10 to 50 in 5 increments.

```
from sklearn.metrics import classification_report, confusion_matrix

from sklearn.ensemble import AdaBoostClassifier

from sklearn.metrics import accuracy_score

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier

from sklearn import tree

from sklearn.model_selection import train_test_split

import pandas as pd

import numpy as np

import os

import warnings

warnings.filterwarnings(action='ignore')

est = 10;

report1 = pd.read_csv("/Users/pmh/AppData/Local/Programs/Python/Python38-
32/py/dstest.csv")
```

```
X = np.array(pd.DataFrame(report1, columns=['X','Y']))

y = np.array(pd.DataFrame(report1, columns=['C']))

X_train, X_test, y_train, y_test = train_test_split(X,y, train_size = 0.8)



for i in range(20):

    ada = AdaBoostClassifier(n_estimators=est, random_state = 5)

    ada.fit(X_train, y_train)

    y_train_pred = ada.predict(X_train)

    y_test_pred = ada.predict(X_test)

    ada_train = accuracy_score(y_train, y_train_pred)

    ada_test = accuracy_score(y_test, y_test_pred)

    print('%d est AdaBoost train/test accuracies %.3f/%.3f' % (est, ada_train, ada_test))

    est = est +10;
```

```
=== RESTART: C:₩Users₩pmh₩AppData₩Local₩Programs₩Python₩Python38-32₩py₩1-2.py ==
10 est AdaBoost train/test accuracies 1.000/1.000
20 est AdaBoost train/test accuracies 1.000/1.000
30 est AdaBoost train/test accuracies 1.000/1.000
40 est AdaBoost train/test accuracies 1.000/1.000
50 est AdaBoost train/test accuracies 1.000/1.000
60 est AdaBoost train/test accuracies 1.000/1.000
70 est AdaBoost train/test accuracies 1.000/1.000
80 est AdaBoost train/test accuracies 1.000/1.000
90 est AdaBoost train/test accuracies 1.000/1.000
100 est AdaBoost train/test accuracies 1.000/1.000
110 est AdaBoost train/test accuracies 1.000/1.000
120 est AdaBoost train/test accuracies 1.000/1.000
130 est AdaBoost train/test accuracies 1.000/1.000
140 est AdaBoost train/test accuracies 1.000/1.000
150 est AdaBoost train/test accuracies 1.000/1.000
160 est AdaBoost train/test accuracies 1.000/1.000
170 est AdaBoost train/test accuracies 1.000/1.000
180 est AdaBoost train/test accuracies 1.000/1.000
190 est AdaBoost train/test accuracies 1.000/1.000
200 est AdaBoost train/test accuracies 1.000/1.000
>>>
```

(c) Build and evaluate a random forest ensemble where max_depth is 2 and n_estimators varies from 10 to 50 in 5 increments.

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
import warnings


warnings.filterwarnings(action='ignore')


est = 10;


report1 = pd.read_csv("/Users/pmh/AppData/Local/Programs/Python/Python38-32/py/dstest.csv")
X = np.array(pd.DataFrame(report1, columns=['X','Y']))
y = np.array(pd.DataFrame(report1, columns=['C']))
X_train, X_test, y_train, y_test = train_test_split(X,y, train_size = 0.8)


for i in range(20):
    ran = RandomForestClassifier(n_estimators=est, criterion='entropy', max_depth=2,
                                 min_samples_split=10, min_samples_leaf=10,
                                 max_leaf_nodes=100, bootstrap=True)
    ran.fit(X_train, y_train)
```

```
=== RESTART: C:₩Users₩pmh₩AppData₩Local₩Programs₩Python₩Python38-32₩py₩1-3.py ==
10 est RandomForest train/test accuracies 0.933/0.925
20 est RandomForest train/test accuracies 0.933/0.925
30 est RandomForest train/test accuracies 0.998/1.000
40 est RandomForest train/test accuracies 0.933/0.925
50 est RandomForest train/test accuracies 0.998/1.000
60 est RandomForest train/test accuracies 0.933/0.925
70 est RandomForest train/test accuracies 1.000/1.000
80 est RandomForest train/test accuracies 1.000/1.000
90 est RandomForest train/test accuracies 1.000/1.000
100 est RandomForest train/test accuracies 0.998/1.000
110 est RandomForest train/test accuracies 1.000/1.000
120 est RandomForest train/test accuracies 0.933/0.925
130 est RandomForest train/test accuracies 1.000/1.000
140 est RandomForest train/test accuracies 0.933/0.925
150 est RandomForest train/test accuracies 1.000/1.000
160 est RandomForest train/test accuracies 1.000/1.000
170 est RandomForest train/test accuracies 0.933/0.925
180 est RandomForest train/test accuracies 0.933/0.925
190 est RandomForest train/test accuracies 0.933/0.925
200 est RandomForest train/test accuracies 0.998/1.000
>>>
```

(d) Disscuss the results between decision tree and ensemble model.

Problem 2

Answer the questions.

(a) Write a generic procedure for building a decision tree.

S1 : Start at the top of the tree.

S2 : Grow it by splitting attributes one by one to determine which attribute to split, look at node impurity.

S3 : Assign leaf nodes the majority vote in the leaf.

S4 : When getting to the bottom, prune the tree to prevent overfitting.

(b) Discuss the advantages and drawbacks of decision tree applications for solving real-world problems.

장점 : decision tree를 통한 data 분석의 결과는 tree 구조로 표현되기 때문에 분석가가 결과를 쉽게 이해하고 설명할 수 있다.

단점 : decision tree는 Hill Climbing 방식 및 Greedy 방식을 사용하고 있다. 일반적인 Greedy 방식의 알고리즘이 그렇듯이 이 방식은 최적의 해를 보장하지는 못한다.

레코드의 개수가 약간의 차이만 있어도 tree의 모양이 달라질 수 있다.

Problem 3

Answer the questions on information gain.

(a) Define information $\mathrm{I(p)}$ from observing the occurrence of an event with probability p.

$\text{no. of bits} = -log_2 p\,, 0 \le p \le 1$

(b) What is the meaning of information $\mathrm{I(p)}$?

(Interpretation) The expected number of bits needed to encode p

$\mathrm{I(p)} = -log_2 p$

(c) Prove the following subproblems with probability p and integer m and n.

(1) $\mathrm{I}(p^n) = n \cdot I(p)$

$\mathrm{I}(p^n) = -log_2 p^n = -nlog_2 p = \mathrm{n} \cdot \mathrm{I(p)}$

(2) $\mathrm{I(p)} = \mathrm{I}\left(p^{(1/m)^m}\right) = mI(p^{1/m})$

$\mathrm{I(p)} = \mathrm{I}\left(p^{(1/m)^m}\right) = -log_2 p^{(1/m)^m} = -mlog_2 p^{(1/m)} = mI(p^{1/m})$

(3) $\mathrm{I}\left(p^{m/n}\right) = \frac{m}{n} I(p)$

$$\mathrm{I}\left(p^{m/n}\right) = -log_2 p^{(m/n)} = -\frac{m}{n} log_2 p = \frac{m}{n} I(p)$$

Problem 4

Table 1: A toy problem

X_1	X_2	X_3	Y
T	T	F	T
T	T	T	F
T	F	T	F
T	F	F	T
F	T	F	T
F	T	T	F
F	F	F	F
F	F	T	T

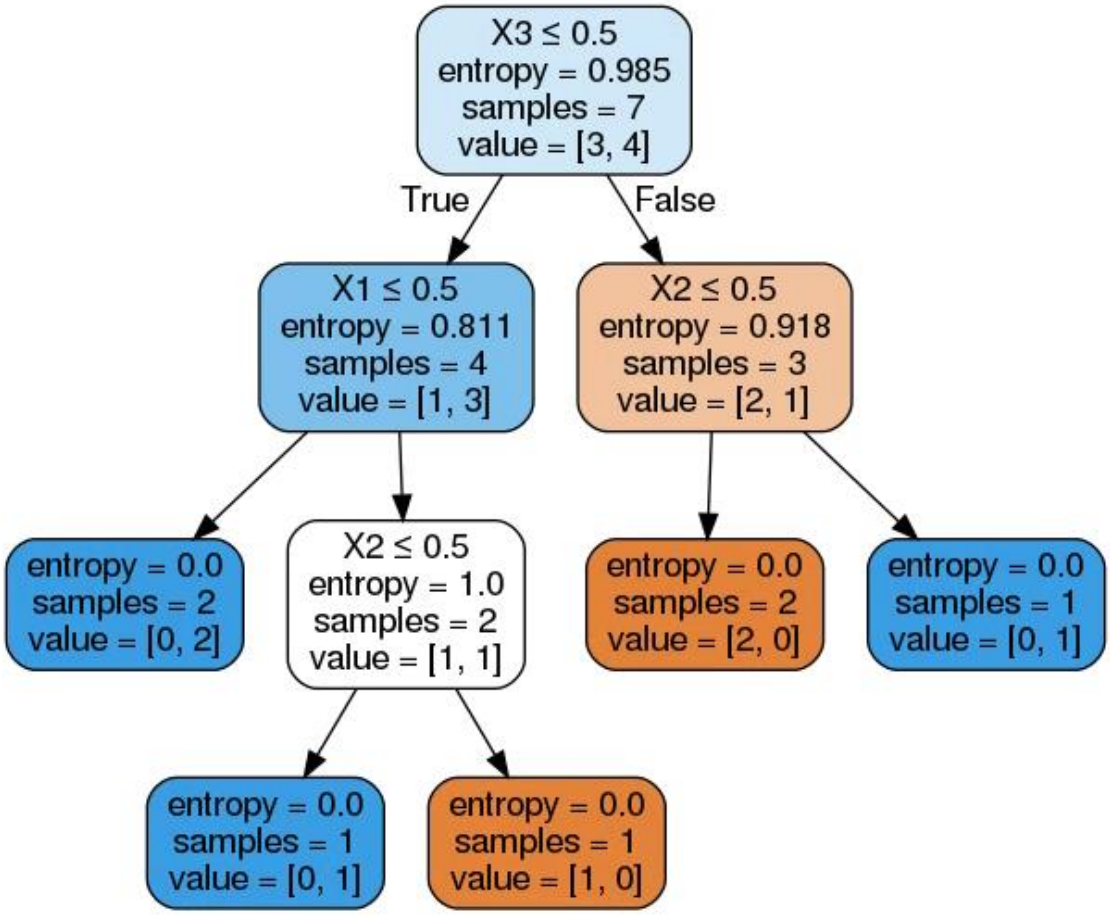
Draw a decision tree using information gain $\mathrm{IG(X)}$ from Table 1. The decision tree is based on the pseudo code.

S1 Start from empty decision tree

S2 Split on the best attribute selected by computing

$$\arg\max\ IG(X_i) = \arg\ \max\ (H(Y) - H(Y|X_i))$$

S3 Do S2 until the termination condition is met



Problem 5

Describe and discuss an overfitting in decision tree practice.

Overfitting은 학습데이터에서는 높은 정확도를 보이지만, 학습데이터가 아닌 실제 테스트 데이터에 대한 결과에서는 학습데이터에 비해 낮은 정확도를 보여주는 것을 말한다.

Decision tree에서의 Overfitting 문제가 발생하는 이유는 Decision tree의 tree 구성이 세분화되고 깊어지면 깊어 질수록, 학습 데이터에 특화되어지고, 이와 같은 이유로 실제 테스트 데이터에 대해서는 정확도가 떨어질 위험이 크기 때문이다.

그러므로, Decision tree에서의 Overfitting 문제를 해결하기 위한 가장 간단한 방법으로는, Pruning 방법이 있다. Pruning은 트레이닝 단계에서 tree의 구성을 너무 세분화하지 않고 tree의 깊이를 조절하는 것을 말한다. 이 방법을 쓰게 되면, 트레이닝 데이터에 대해서는 정확도가 낮아지지만, 실제 테스트 데이터에서는 오히려 좋은 정확도를 보여줄 수 있다.

Problem 6

The ensemble approach has become a cutting edge model in artificial intelligence.

Below are some questions on ensemble model.

(a) Why do we use an ensemble model instead of a single model?

앙상블 학습은 하나의 모델만을 학습시키는 것이 아니라 여러 모델(weak learners)을 조합하여 최종 모델을 만들어 내는 것을 말한다. 이와 같은 방법으로 모델을 설계하게 되면, 일반적인 경우 single model로 데이터를 학습시켰을 때 얻을 수 있는 성능보다 좋은 성능을 얻을 수 있다.

$$\text{ensemble} = \mathrm{P}(\mathrm{y} \geq \mathrm{k}) = \sum_{k=1}^{n} \binom{n}{k} \epsilon^{k} (1-\epsilon)^{n-k}$$

(b) Discuss bagging and boosting strategies to construct an ensemble model.

Bagging : Bootstrap Aggregating 1. Pick training set random sampling(with replacement), 2. Learn Models parallelly by picked training set 3. Getting final model from trained model through majority vote

1. Parallel execute
2. Minimize Variance
3. Random sampling

Boosting : 1. Init state, all training data are weighted equally 2. At each step, a new learner is trained by increasing the weights of the incorrectly predicted examples 3. Final prediction is computed with the weighted sum of all the learners.

1. Sequential execute
2. Decrease bias
3. Weight misclassified data

(c) Point out the drawback of ensemble models.

1. Harder to tune than a single model

2. Lack of interpretability, compared to linear classifiers

3. Need hing computation time

(d) Give model examples using bagging or boosting strategy for an ensemble model.

bagging : random forest

Boosting : adaboost, xgboost