Fundamentals of Machine Learning (Fall 2022)

Homework #4 (100 Pts, Due date: Nov 20)

Student ID 2017314712

Name 김우진

Instruction: We provide all codes and datasets in Python. Please write your code to complete Decision Tree and BaggingEnsemble('models/DecisionTree.py', 'models/Bagging.py). You should submit two files as follows:

- 'STUDENTID_HW4.zip': ./models/*.py and your document. Your document must be converted into a pdf.

Note: **You should write your source code in the 'EDIT HERE' and not edit other parts.**

(1) [40 pts] We provide all template codes and datasets in Python. Write your code to implement the decision tree. You must install NumPy, Scikit-learn (sklearn), and Matplotlib libraries.

(a) [10 pts] Implement the function 'compute_entropy' in 'model/DecisionTree.py.' The entropy is defined as follows:

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x)$$

Answer: Write your code here. You also have to submit your code to i-campus.

```
positive = 0
negative = 0
for i in labels:
    if i == 'positive':
        positive += 1
    if i == 'negative' :
        negative += 1

all = positive + negative
positive = positive/all
negative = negative/all
if positive == 0 or negative == 0:
    entropy = 0
else:
    entropy = -(positive*np.log2(positive) + negative*np.log2(negative))
```

(b) [20 pts] Implement the function 'selection_criteria' in 'model/DecisionTree.py.' The conditional entropy and information gain are defined as follows:

$$H(Y|X) = \sum_{x \in X} p(x)H(Y|X = x), IG(Y|A) = H(Y) - H(Y|A)$$

Answer: Write your code here. You also have to submit your code to i-campus.

```
n = 0
x_num = 0
o_num = 0
b_num = 0

for i in distinct_values:
    if i == 'x':
        x_num = counts[n]
    if i == 'o':
        o_num = counts[n]
    if i == 'b':
        b_num = counts[n]
    n += 1

all = x_num + o_num + b_num

x_label = []
o_label = []
b_label = []

num = len(feature_data)
for i in range(0,num):
    if labels[i] == 'positive':
        if feature_data[i] == 'x':
            x_label.append('positive')
        if feature_data[i] == 'o':
            o_label.append('positive')
        if feature_data[i] == 'b':
            b_label.append('positive')
    else :
        if feature_data[i] == 'x':
            x_label.append('negative')
        if feature_data[i] == 'o':
            o_label.append('negative')
        if feature_data[i] == 'b':
            b_label.append('negative')

entropy = compute_entropy(labels)
conditional_entropy = 0
if x_num != 0:
    conditional_entropy += (x_num/all)*(compute_entropy(x_label))
if o_num != 0:
    conditional_entropy += (o_num/all)*(compute_entropy(o_label))
if b_num != 0:
    conditional_entropy += (b_num/all)*(compute_entropy(b_label))

information_gain = entropy - conditional_entropy
```

(c) [10 pts] After training the model, fill in the blank using the code provided in '0_DecisionTree_main.py'.

Answer: Fill in the blank in the table.

Max depth	Accuracy
3	0.7696
4	0.7958
5	0.8586

(2) [40 pts] We provide all template codes and datasets in Python. Write your code to implement the bagging ensemble. You may need to install NumPy, Scikit-learn (sklearn), and Matplotlib libraries.

(a) [10 pts] Implement the function 'create_datasets' in 'model/BaggingSampler.py'. You need to generate m new datasets by randomly sampling the raining dataset with a given sampling ratio in this function.

Answer: Write your code here. You also have to submit your code to i-campus.

```
for i in range(int(n_samples)):
    sampled_df.append(random.choice(df_values))
#print(sampled_df)
```

(b) [10 pts] Implement the function 'random_feature_selection' in 'model/DecisionTree.py'.

Answer: Write your code here. You also have to submit your code to i-campus.

```
random_features = np.random.choice(feature_list,num_features,replace=False)
#
```

(c) [20 pts] Fill in the blank using the code provided in '1_RandomForest_main.py.' Both depth and number of features are fixed as 3.

Answer: Fill in the blank in the table.

Model	Sampling ratio	Num datasets	Accuracy
Decision tree	-	-	
Random Forest	0.6	1	0.6545
		10	0.8010
		50	0.8325
		100	0.8325
	0.1	50	0.8272
	0.3		0.8325
	0.5		0.8325
	0.7		0.7958

(3) [20 pts] Given the "breast_cancer" data and '2_Performance_Comparison.py', draw a plot that depicts performances of decision tree, random forest, AdaBoost. Briefly explain your result. You can use the scikit-learn library

Answer:

Decision Tree는 값이 항상 동일하다는 것을 알수있다

Random Forest는 estimator 개수가 늘어나면 보통 acc가 늘어난다.

이 seed에서는 동일하게 나온부분이있지만 다른seed로 해봤을 때 늘어났다

Adaboost도 estimator 개수가 늘어날수록 성능이 올라갔다

Adaboost가 가장 뛰어난 성능을 보였다.