NYCU Introduction to Machine Learning, Homework 3

[111550151], [徐嘉亨]

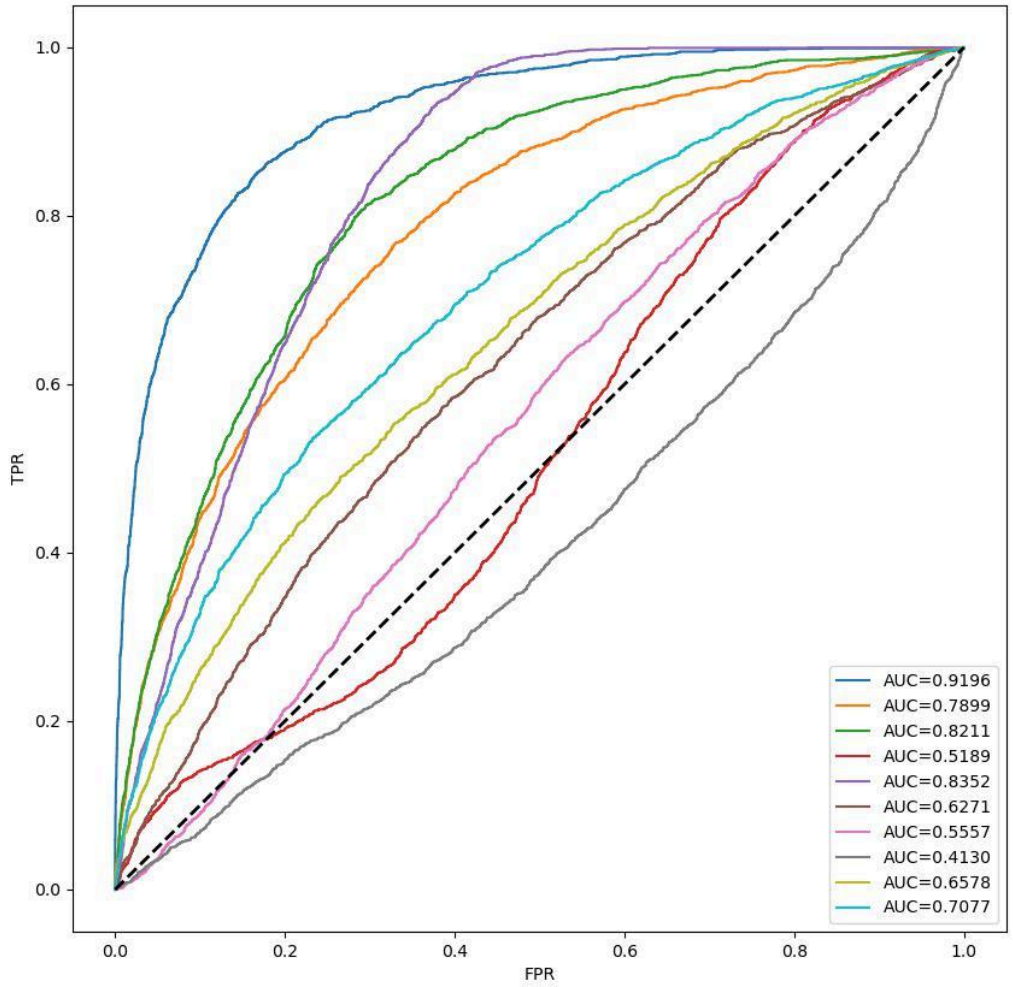
Part. 1, Coding (60%):

(20%) Adaboost

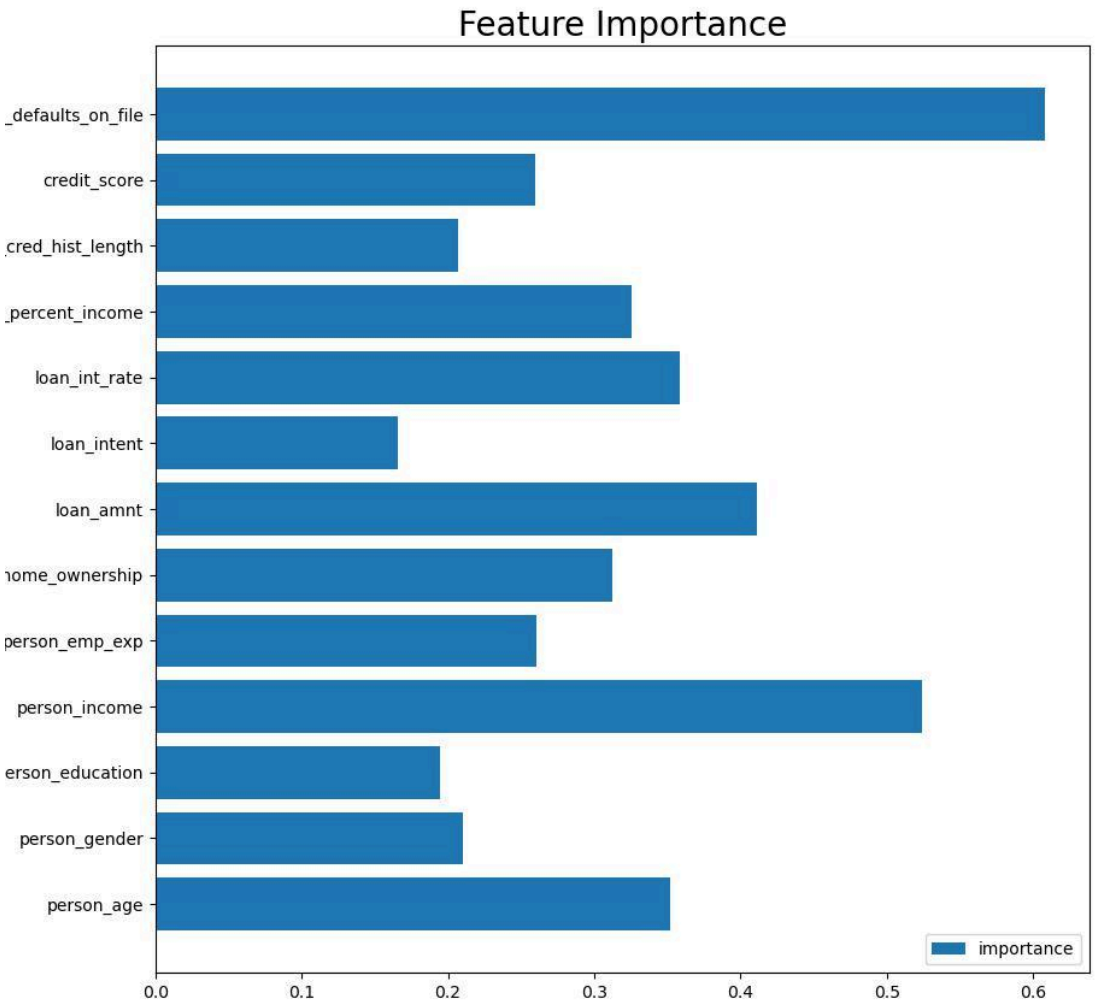
1. (10%) Show your accuracy of the testing data (n_estimators = 10)

```
2024-11-19 01:06:59.822 | INFO     | __main__:main:55 - AdaBoost - Accuracy: 0.8481
```

2. (5%) Plot the AUC curves of each weak classifier.



3. (5%) Plot the feature importance of the AdaBoost method. Also, you should snapshot the implementation to calculate the feature importance.



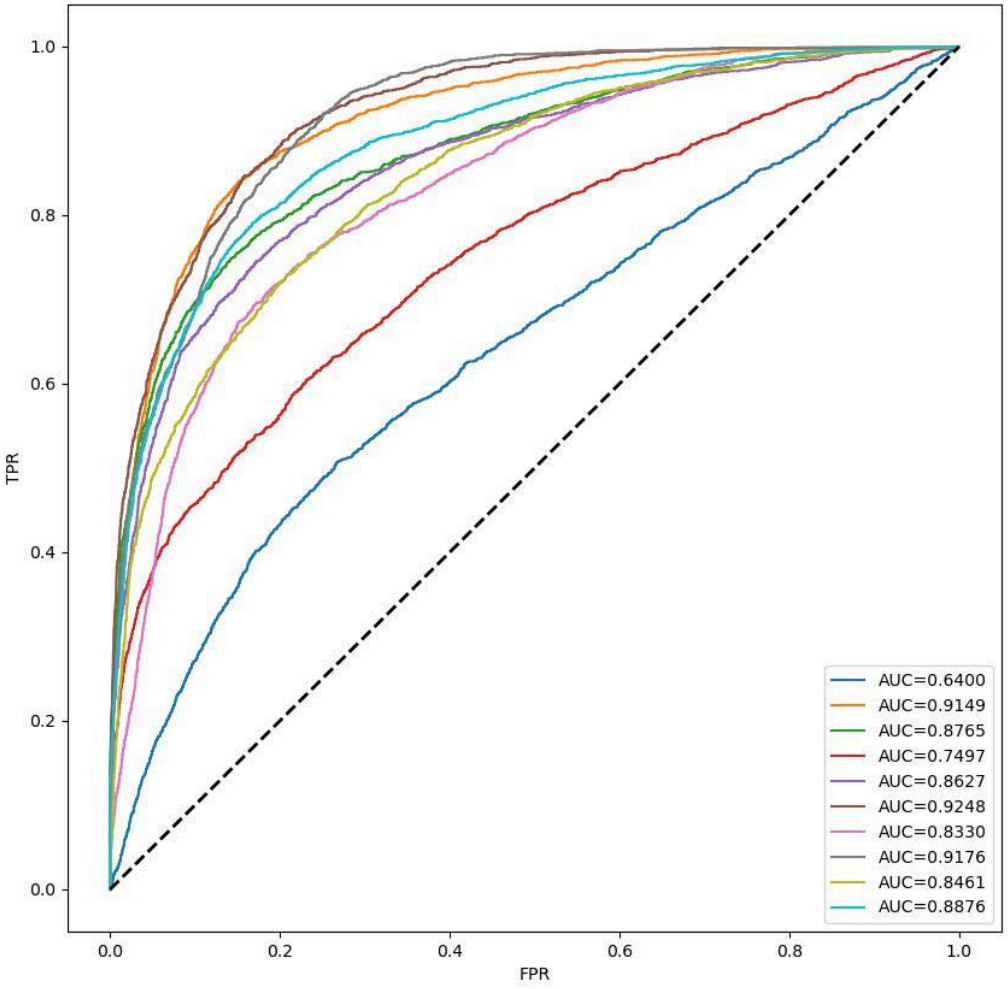

```python
def compute_feature_importance(self) -> t.Sequence[float]:
    """Implement your code here"""
    feature_importance = []
    for learner, alpha in zip(self.learners, self.alphas):
        feature_weights = learner.layer.weight.data.abs().numpy()
        importance = feature_weights * alpha
        feature_importance.append(importance)
    feature_importance = np.array(feature_importance)
    feature_importance = np.sum(feature_importance, axis=0)
    return feature_importance.reshape(-1)
```

(20%) Bagging

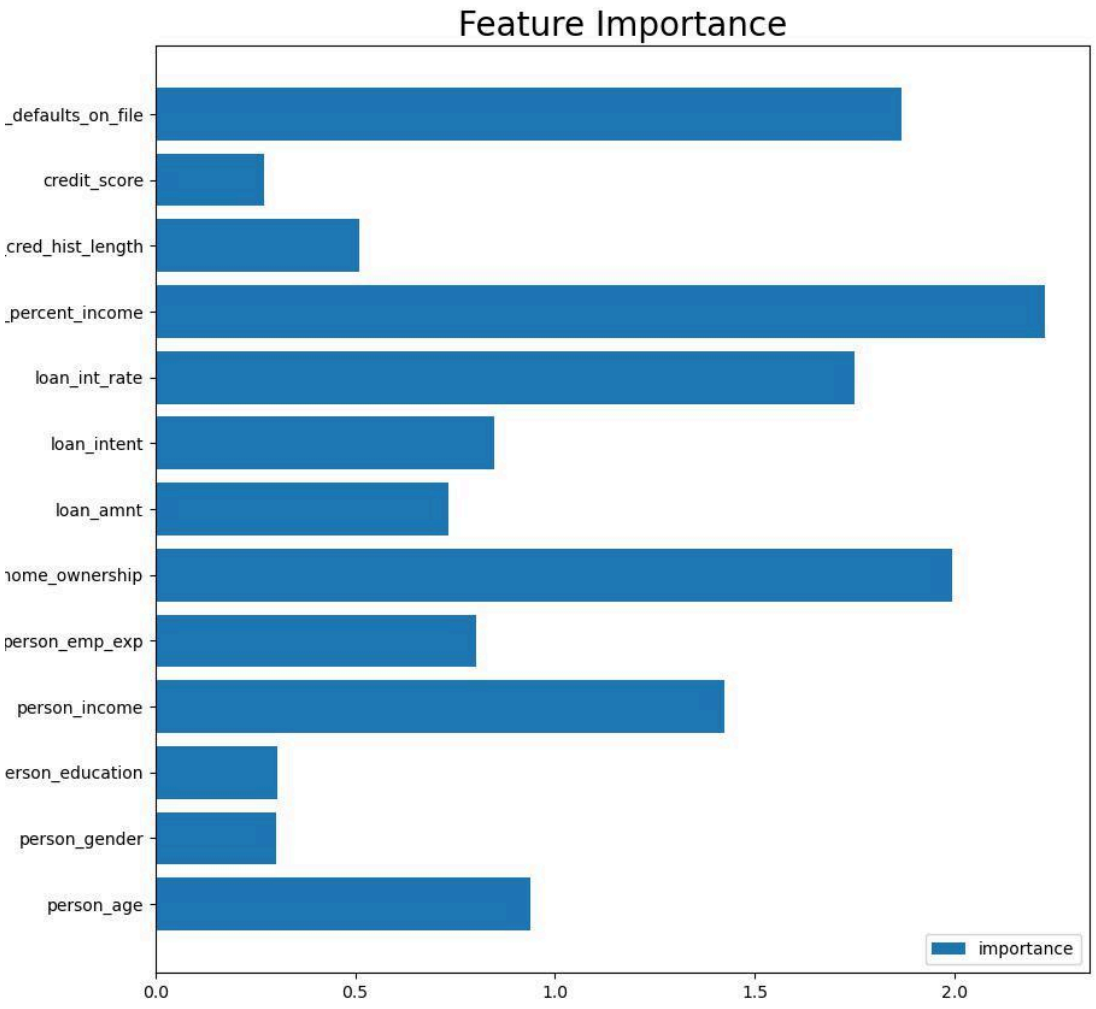
4. (10%) Show your accuracy of the testing data with 10 estimators. (n_estimators=10)

```
2024-11-19 01:07:14.154 | INFO     | __main__:main:83 - Bagging - Accuracy: 0.8470
```

5. (5%) Plot the AUC curves of each weak classifier.



6. (5%) Plot the feature importance of the Bagging method. Also, you should snapshot the implementation to calculate the feature importance.




```python
def compute_feature_importance(self) -> t.Sequence[float]:
    """Implement your code here"""
    feature_importance = []
    for learner in self.learners:
        feature_weights = learner.layer.weight.data.abs().numpy()
        feature_importance.append(feature_weights)
    feature_importance = np.array(feature_importance)
    feature_importance = np.sum(feature_importance, axis=0)
    return feature_importance.reshape(-1)
```

(15%) Decision Tree

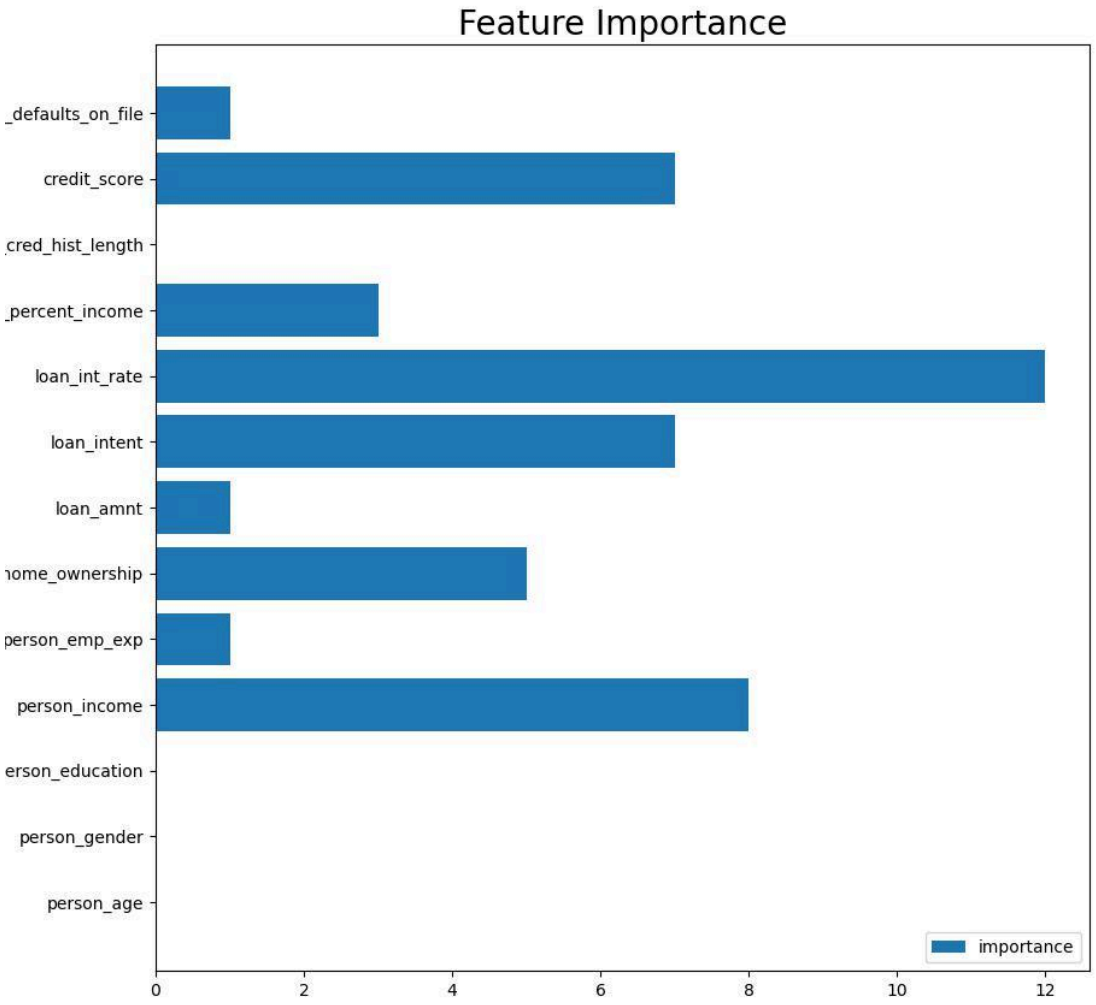
7. (5%) Compute the Gini index and the entropy of the array [0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1].

```
2024-11-19 01:07:43.088 | INFO     | __main__:main:104 - Gini index of data is 0.4628
2024-11-19 01:07:43.088 | INFO     | __main__:main:105 - Entropy of data is 0.9457
```

8. (5%) Show your accuracy of the testing data with a max-depth = 7

```
2024-11-19 01:18:24.545 | INFO     | __main__:main:109 - DecisionTree - Accuracy: 0.9044
```

9. (5%) Plot the feature importance of the decision tree.



(5%) Code Linting

10. Show the snapshot of the flake8 linting result (paste the execution command even when there is no error).

```
(mlhw1) jxea666@LAPTOP-AR88PBDG:~/Intro2ML/hw3$ flake8 main.py
(mlhw1) jxea666@LAPTOP-AR88PBDG:~/Intro2ML/hw3$ cd src
(mlhw1) jxea666@LAPTOP-AR88PBDG:~/Intro2ML/hw3/src$ flake8 __init__.py
(mlhw1) jxea666@LAPTOP-AR88PBDG:~/Intro2ML/hw3/src$ flake8 adaboost.py
(mlhw1) jxea666@LAPTOP-AR88PBDG:~/Intro2ML/hw3/src$ flake8 bagging.py
(mlhw1) jxea666@LAPTOP-AR88PBDG:~/Intro2ML/hw3/src$ flake8 decision_tree.py
(mlhw1) jxea666@LAPTOP-AR88PBDG:~/Intro2ML/hw3/src$ flake8 utils.py
(mlhw1) jxea666@LAPTOP-AR88PBDG:~/Intro2ML/hw3/src$
```

Part. 2, Questions (40%):

1. (10%) What are Bagging and Boosting, and how are they different? Please list their differences and compare the two methods in detail.

   Bagging is an ensemble method that trains multiple models independently in parallel on different subsets of the data and averaging their predictions (for regression) or voting (for classification). Boosting is an ensemble method that builds models sequentially, with each model focusing on correcting the errors made by the previous ones. The final model combines all the models to make a strong predictor. About their differences: 1. Bagging aims for reducing the impact of noise and overfitting by averaging predictions. Boosting aims for reducing bias and focusing on harder examples. 2. Bagging models are trained independently. Boosting models are trained sequentially. 3. Bagging random samples subsets (bootstrapped samples) of data. Boosting samples subsets of data by the sample weights. 4. Bagging treats all models equally and averages. Boosting focuses on correcting previous errors by reweighting misclassified samples. 5. Bagging combines results by majority vote (classification) or averaging (regression). Boosting let weighted sum of model outputs to be final prediction based on their performance.

2. (15%) What criterion do we use to decide when we stop splitting while performing the decision tree algorithm? Please list at least three criteria.

   1. The tree stops splitting when a predefined maximum depth is reached. 2. A split is allowed only if each child node contains at least a specified minimum number of samples. 3. Splitting stops if the data subset is pure, which means all data in that subset belongs to the same class. 4. The information gain is less than a predefined threshold.

3. (15%) A student claims to have a brilliant idea to make random forests more powerful: since random forests prefer trees which are diverse, i.e., not strongly

correlated, the student proposes setting m = 1, where m is the number of random features used in each node of each decision tree. The student claims that this will improve accuracy while reducing variance. Do you agree with the student's claims? Clearly explain your answer.

(In this question, I suppose this random forest is the same as defined in the lecture: In each node of each decision tree, it will randomly select *m* attributes(this m has the same meaning with the m in the question) from the *M* attributes, and it will pick the **best** attribute and its threshold as the split) In my opinion, this setting is unlikely to improve accuracy but might decrease the variance. Let's talk about the choice of m, if m is large, which allows trees to choose the most relevant features with higher probability than smaller m(because it has many choices), improving the quality of each tree. If m is small, the trees have higher probability to be forced to use different features because it only chooses one feature from many features to use. That means it might increase the diversity among the trees in the forest. In other words, it reduces correlation between trees, which reduces the overall variance of the forest. So I think the accuracy will not increase or even decrease but variance will decrease if m is 1(small).