# Homework 3 Programming Part Report

Foundations of Artificial Intelligence (CSIE3005)

Department of Information Management, NTU - B11705048 林杰

### (A) PERFORMANCE COMPARISON: DIFFERENT MODELS

(Parameters for this subproblem: learning\_rate = 0.01, iterations = 1000, max\_depth = 5, n estimators = 100.)

#### **Classification Task:**

For the classification task, the evaluation metric is the accuracy.

$$accuracy = \frac{correct\ pedictions}{total\ predictions}$$

Logistic Regression Accuracy: 66.67%

Decision Tree Classifier Accuracy: 88.89% Random Forest Classifier Accuracy: 95.56%

### Analysis:

Decision trees can model complex, non-linear relationships between the features and the target variable, while logistic regression assumes a linear relationship between the input features. Therefore, decision tree classifier can reach higher accuracy than simple logistic regression.

Random forest classifier randomly selects samples from the data and train separate decision trees based on each sample. Then, the predictions from all the individual trees are combined together using a voting mechanism for the final classification result. Therefore, the decision tree classifier can reach the highest accuracy among the three models.

### **Regression Task:**

For the classification task, the evaluation matric is the mean square error (MSE):

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \widehat{y}_i)$$

Linear Regression MSE: 43.41

Decision Tree Regressor MSE: 28.34 Ransom Forest Regressor MSE: 22.96

#### Analysis:

Similar to the previous analysis, decision trees can model more complex and

non-linear relationships between the features. Therefore, the accuracy rate is higher than the simple linear regression model.

Random forest classifier randomly selects samples from the data and train separate decision trees based on each sample. For regression problems, the average of the predictions from all trees is used. Therefore, the decision tree classifier can reach the highest accuracy among the three models.

## (B) PERFORMANCE COMPARISON: NORMALIZATION AND STANDARDIZATION

(Parameters for this subproblem: learning\_rate = 0.01, iterations = 1000)
For this subproblem, I analyze the impact of normalization and standardization techniques on the **logistic regression model**.

## Normalization (Min-Max Scaling):

This method will rescale the value to the range of [0,1]:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Min-Max scaling can be useful when we want the value to be on the same scale. This can sometimes improve performance and the convergence speed. However, this method may be skewed to the minimum and maximum value, resulting to poor scaling.

Setting	With Normalization	Without Normalization	
Statistics			
Accuracy	66.67%	100%	
Execution Time (s)	0.0178	0.0296	

From the above table, we can find out that using the normalization method cannot improve performance. However, the execution time with normalization is faster.

## **Standardization (Z-Score Standardization):**

This method can center the feature values around the mean with a unit standard deviation:

$$x' = \frac{x - \mu}{\sigma}$$

This method can be useful when we assume the data is normally distributed and can deal with outliers properly. However, this method may not lead to good result when the data distribution is skewed.

Setting	With Standardization	Without Standardization	
Statistics			
Accuracy	88.88%	100%	
Execution Time (s)	0.0277	0.0296	

From the above table, we can find out that using the standardization method cannot improve performance based on the given data. The execution time is slighted faster.

## (C) PERFORMANCE COMPARISON: LEARNING RATE AND # OF ITERATIONS

For this subproblem, I analyze the impact of the changes of learning rate and number of iterations on the **linear regression model**.

**Configuration 1:** (Learning rate = 0.005, iterations = 4000)

Linear Regression MSE: 39.7053163178137

Execution Time: 0.015612 seconds

**Configuration 2:** (Learning rate = 0.01, iterations = 1000)

Linear Regression MSE: 43.41392463386301

Execution Time: 0.039103 seconds

From the above statistics, we can find out that using lower learning rate and more iterations can result in lower error (assume overfitting does not happen). A lower learning rate allows the model to make more precise updates to the weights during each iteration, leading to a more stable convergence process. A higher number of iterations give the model more chances to adjust the parameters and get closer to optimal. However, more iterations require more execution time. But if overfitting is observed, maybe lower learning rate and less iterations will be preferred.

### (D) IMPACT OF HYPERPARAMETERS ON THE RANDOM FOREST MODEL

### Results of using different hyperparameters on the random forest model:

Setting	# of trees = 50	# of trees = 100	# of trees = 50
Task	Max depth = 5	Max depth = 5	Max depth = 10
Classification Task	93.33%	95.56%	93.33%
(Accuracy)			
Regression Task (MSE)	22.6455	22.9648	22.7834

From the above table, we can see that in the given data, a random forest with more trees can result in a better prediction in terms of classification task. Deeper trees can also give a better result, assume overfitting does not happen.

How ever, there is no significance difference between the MSE of the regression task. The following are the discussion about the two hyperparameters.

- # of trees: Generally, increasing the number of trees can improve model
  performance by making the predictions more stable. A higher number of trees
  can improve generalization, but with higher computational complexity and
  longer training time.
- Maximum Depth: Deeper trees have the ability to capture more complex
  patterns but may overfit. Shallow trees may underfit. Deeper trees increase the
  model's complexity and computational cost.

How I select the hyperparameters: By trial and error, use a loop to find the hyperparameters that gives the best result. (Best parameters I find: learning\_rate = 0.05, iterations = 4000, max\_depth = 5, n\_estimators = 100. This is implemented in hw3.py)

## (E) STRENGTH AND WEAKNESSES OF THE IMPLEMENTED MODELS

### **Logistic Regression, Linear Regression:**

Best for the following cases:

- When the relationship between the input features and the target variable is nearly linear.
- When you need a model that is easy to interpret and explain.

#### Advantages:

- Fast and efficient to train and predict.
- Less possible to overfit.

#### Limitations:

- It assumes a linear relationship between the input and outcome.
- Not suitable for complex relationships (e.g. non-linear) or interactions between features.

#### **Decision Tree:**

Best for the following cases:

- When the relationship between features and the target is non-linear.
- When you need a model that can be visualized and easy to understand.

### Advantages:

- Easy to interpret and visualize.
- Can deal with non-linear relationships and interactions between features.

#### Limitations:

- When the tree is deep, the model can overfit easily.
- Sensitive to small changes in the data

#### **Decision Tree:**

Best for the following cases:

- When dealing with complex data with non-linear relationships.
- When you want a more robust model that is less possible to overfit than a single decision tree

### Advantages:

- Reduces overfitting compared to individual decision trees.
- Higher prediction accuracy.

#### Limitations:

- Less interpretable compared to single decision trees.
- Needs lots of computational resources.

We can select the suitable model based on the points listed above based on our problem.

#### Reference:

https://medium.com/aimonks/harmonizing-the-scales-the-essential-role-of-feature-scaling-in-machine-learning-0b49e1e99aac

#### ChatGPT