UTM

UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING

SCHOOL OF COMPUTING

SECB4313-01

BIOINFORMATICS MODELING AND SIMULATION

ASSIGNMENT 2 - HYPERPARAMETER OPTIMIZATION TECHNIQUES

LECTURER:

DR. AZURAH BINTI A SAMAH

GROUP MEMBERS:

SHAHRIL BIN SAIFUL BAHRI (A20EC0144)

MUHAMMAD AIMAN BIN ABDUL RAZAK (A20EC0082)

NAVINTHRA RAO A/L VENKATAKUMAR (A20EC0104)

1. Identify 4 hyperparameters and propose two values for each hyperparameter. Justify the selection of the hyperparameters and its corresponding values.

a) Number Of Trees (‘n_estimators’) : value (100, 200)

Justification: Increasing the value of trees can increase model performance by reducing variance but at the cost of increased computation

b) Maximum Depth of Trees (‘max_depth’) : value (10, 20)

Justification: Controlling the maximum depth of trees can prevent overfitting and underfitting of the trees.

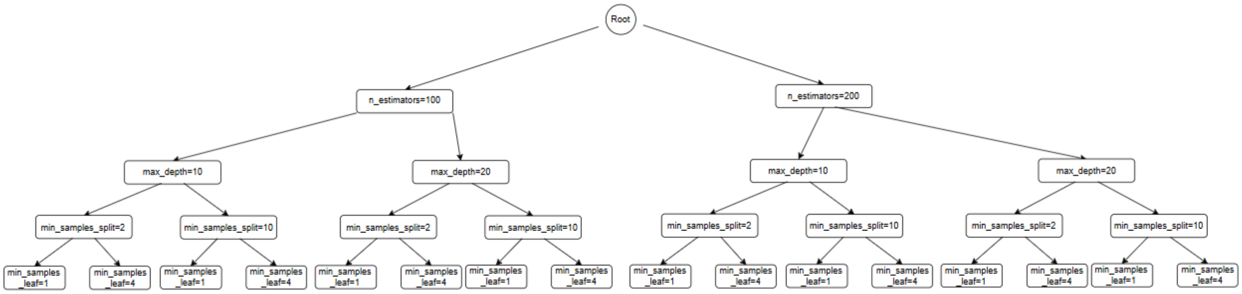
c) Minimum Samples Split (‘min_samples_split’) : value (2, 10)

Justification: By having control over the samples split, we can determine the model’s complexity and performance of the internal node.

d) Minimum Samples Leaf (‘min_samples_leaf’) : value (1, 4)

Justification: Having the control on how many samples required on a leaf node, which smoothen the model.

2. Construct a tree diagram that display the proposed hyperparameters and its corresponding values.



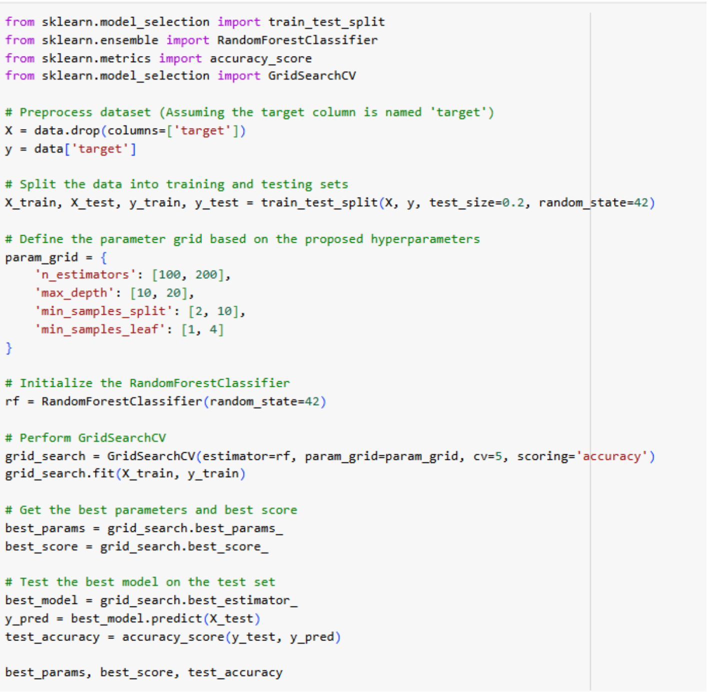
3. Tabulate the proposed experimental design based on your tree diagram

Experiment	‘n_estimators’	‘max_depth’	‘min_samples_split’	‘min_samples_leaf’
1	100	10	2	1
2	100	10	2	4
3	100	10	10	1
4	100	10	10	4
5	100	20	2	1
6	100	20	2	4
7	100	20	10	1
8	100	20	10	4
9	200	10	2	1
10	200	10	2	4

11	200	10	10	1
12	200	10	10	4
13	200	20	2	1
14	200	20	2	4
15	200	20	10	1
16	200	20	10	4

4. Perform hyperparameter tuning to improve current result. (Simulate the model and collect the results)

The code below will print out the best hyperparameter, best score and test accuracy of the model.



```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV

# Preprocess dataset (Assuming the target column is named 'target')
X = data.drop(columns=['target'])
y = data['target']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Define the parameter grid based on the proposed hyperparameters
param_grid = {
    'n_estimators': [100, 200],
    'max_depth': [10, 20],
    'min_samples_split': [2, 10],
    'min_samples_leaf': [1, 4]
}

# Initialize the RandomForestClassifier
rf = RandomForestClassifier(random_state=42)

# Perform GridSearchCV
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

# Get the best parameters and best score
best_params = grid_search.best_params_
best_score = grid_search.best_score_

# Test the best model on the test set
best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test)
test_accuracy = accuracy_score(y_test, y_pred)

best_params, best_score, test_accuracy
```

5. Tabulate the results based on your proposed experimental design

Experiment	‘n_estimators’	‘max_depth’	‘min_samples_split’	‘min_samples_leaf’	Score	Accuracy
10	200	10	2	4	0.81394	0.85245

6. Analyze the results. Which combination of hyperparameters generate most improved result?

Since the code only will generate one output which is the best hyperparameter with the best score and accuracy, experiment 10 is the chosen winner. This combination resulted in a cross-validation accuracy of approximately 81.39% and a test accuracy of approximately 85.25%. This indicates that the model performed well with this set of hyperparameters, providing a balance between complexity and overfitting.