HI Marina and Sandra please read the following document very carefully to understand everything for the evaluatuion

Whatever is in red is code syntax that is later explained

GridSearchCV:

1- Defining a grid of hyperparameter combinations (in this case, parameters like max_depth, min_samples_leaf, etc.).

2-Training the model on multiple subsets of the data (using cross-validation).

3-Evaluating the performance of each combination based on a specified metric (here, roc_auc).

why we use it because:

The Random Forest algorithm has several hyperparameters that can significantly impact its performance, such as:

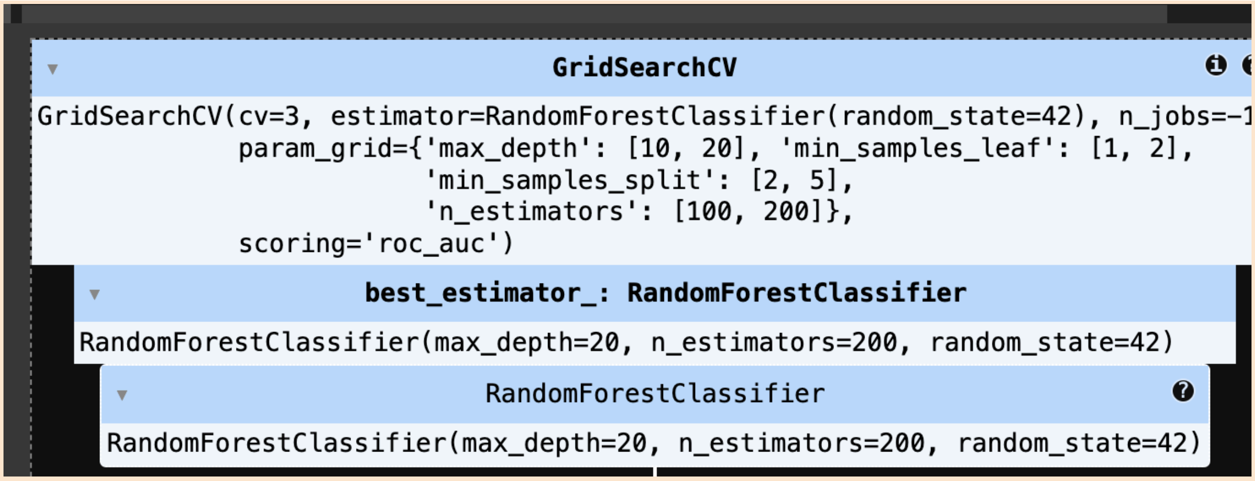
n_estimators: The number of decision trees in the forest.

max_depth: The maximum depth of each decision tree.

min_samples_split: The minimum number of samples required to split an internal node.

min_samples_leaf: The minimum number of samples required to be at a leaf node.

Manually trying different combinations is not only time-consuming but also prone to error. GridSearchCV automates this process, ensuring all combinations are tried and the best parameters are selected.



```
GridSearchCV(cv=3, estimator=RandomForestClassifier(random_state=42), n_jobs=-1
             param_grid={'max_depth': [10, 20], 'min_samples_leaf': [1, 2],
                         'min_samples_split': [2, 5],
                         'n_estimators': [100, 200]},
             scoring='roc_auc')
```

```
RandomForestClassifier(max_depth=20, n_estimators=200, random_state=42)
```

```
RandomForestClassifier(max_depth=20, n_estimators=200, random_state=42)
```

What does the output show?

-GridSearchCV(cv=3, ...): This means a 3-fold cross-validation was used to train and validate the model on different splits of the data.

param_grid: This is the grid of hyperparameters that GridSearchCV tested:

n_estimators: 100, 200

max_depth: 10, 20

min_samples_split: 2, 5

min_samples_leaf: 1, 2

- scoring='roc_auc': This specifies that the ROC AUC score (a performance metric for classification models) was used to evaluate each model.

-best_estimator_: The output RandomForestClassifier(max_depth=20, n_estimators=200, random_state=42) indicates that the best hyperparameters found were:

max_depth = 20

n_estimators = 200

y_pred_proba = model.predict_proba(X_val)[:, 1] if hasattr(model, 'predict_proba') else model.decision_function(X_val)

- **Purpose**:
    - Compute the predicted probabilities of the positive class for the validation set (X_val).
    - Some models support predict_proba, while others use decision_function.
- **Syntax Explanation**:
    - **hasattr(model, 'predict_proba')**:
        - Checks if the model has the predict_proba method.
        - Models like RandomForestClassifier and LogisticRegression support predict_proba.
    - **predict_proba(X_val)[:, 1]**:
        - predict_proba returns a 2D array where:
            - Column 0: Probability of the negative class.
            - Column 1: Probability of the positive class.
        - [:, 1] extracts the probabilities of the positive class.
    - **decision_function(X_val)**:
        - For models like SVM, which don't provide predict_proba, decision_function is used to compute decision scores.
    - **if-else**:

- Uses predict_proba if available; otherwise, uses decision_function.

```
roc_auc = roc_auc_score(y_val, y_pred_proba)
```

```
f1 = f1_score(y_val, y_pred)
```

roc_auc_score(y_val, y_pred_proba):

- Calculates the ROC AUC score, which measures the model's ability to distinguish between positive and negative classes.
- It uses the predicted probabilities (y_pred_proba) and the true labels (y_val).

f1_score(y_val, y_pred):

- Calculates the F1 Score, which is the harmonic mean of precision and recall.
- It uses the predicted class labels (y_pred) and the true labels (y_val).

```
results.append({'Model': name, 'ROC AUC': roc_auc, 'F1 Score': f1})
```

- **Purpose**:
    - Adds a dictionary to the results list with the model's name and its computed metrics.
- **Dictionary Structure**:
    - 'Model': The name of the model (e.g., 'Logistic Regression').
    - 'ROC AUC': The computed ROC AUC score for the model.
    - 'F1 Score': The computed F1 Score for the model.