Holly Cohan,Yesh Onipede, Yu Xia

# Week 8 Report: Model Training, Hyperparameter Tuning, and Model Evaluation

The objective of this week's assignment is to implement a more complex modeling approach compared to the approach selected in week 7. We applied a Random Forest model to our dataset and evaluated different hyperparameter settings to improve the model's performance. The Random Forest model is an ensemble learning technique that improves upon decision trees by combining multiple decision trees to form a more robust and accurate predictive model.

In this assignment, we are tuning several key hyperparameters of the Random Forest model to enhance its performance. The n_estimators parameter, which controls the number of decision trees in the ensemble, is varied to balance computational cost and model accuracy, as more trees generally lead to better performance but increased training time. The max_depth parameter limits the depth of each tree, helping to control overfitting by preventing the trees from growing too complex. Additionally, the max_features parameter, which determines the number of features considered at each split, introduces randomness and can reduce correlation among trees. Setting max_features to values such as 'sqrt' or 'log2' helps make individual trees less similar and the ensemble more robust. These hyperparameters collectively shape the model's ability to generalize to new data and reduce overfitting while maintaining predictive performance.

In preparation for modeling, we needed to ravel the dataframe into a 1-dimensional numpy array. Random Forest can be a computationally intensive model, making cross-validation difficult when tuning hyperparameters. To streamline the process of experimenting with multiple hyperparameters, we created a function *best_rf* that automates the training and evaluation steps on the validation set, helping us identify the best model.

## Random Forest 1: (n_estimators=100, max_depth=15, max_features=None)

The first random forest we built aims to explore how each parameter contributes to the model's performance. While more estimators and larger numbers of max_depth would improve model performance, we found that max_features has a very significantly larger impact on model performance. As we are tuning the value for max_features, the best model has a significantly higher RMSE on the validation set.

**Hyperparameters**

- **n_estimators**: 100
    - We selected 100 trees because with max_depth = 10 and max_features = 'log2', 100 estimators performed the best.
- **max_depth**: 15
    - Between 5, 10, and 15 as options for max_depth, a max_depth of 15 performs the best when n_estimators = 100 and max_features = 'log2'.

- **max_features**: None
    - ‘max_features’ determines the maximum number of features considered when splitting a node in each decision tree. We are experimenting with different values to see how this parameter affects model performance, aiming to find a balance between diversity in the trees and overall accuracy.
    - With max_features = none, the model utilizes all features in training. We have observed that the model’s performance significantly improved when max_features was tuned to None. In addition, for a regression task, max_features = None is typically a good rule of thumb.

**Random Forest 2: ('n_estimators': 200, 'max_depth': 15, 'max_features': None)**

For the second tree, we want to tune max_depth and max_features to a larger number to see if it will improve the model’s performance.

**Hyperparameters**

- **n_estimators**: 200
    - Since this is a regression task, the final prediction is the average of the predictions from all 200 trees.
- **max_depth**: 15
    - ‘max_depth’ controls the maximum depth of each decision tree in the random forest. We are experimenting with this parameter to understand its impact on model performance and to find an optimal value that balances complexity and accuracy.
    - This parameter limits the depth of each decision tree to a maximum of 15 levels. Constraining the depth helps control overfitting by preventing the trees from growing too complex and capturing noise in the training data.
    - A lower max_depth can increase bias because the trees are shallower and may not fully capture the patterns in the data. However, it reduces variance, making the model more generalizable.
- **Max_features**: **None**

**Random Forest 3: ('n_estimators': 200, 'max_depth': 10, 'max_features': None)**

The model configuration for Regression Tree 3 uses a Random Forest with n_estimators=200, max_depth=10, and max_features=None'. This setup aims to address the issue with overfitting. We want to reduce the max_depth for each tree and increase the number of estimators to balance the tradeoff between variance and bias to ensure that the model can perform well on unseen data.
**Hyperparameters**

- **n_estimators**: 200

    - This parameter indicates that the Random Forest is composed of 200 individual decision trees. Each tree is trained on a different bootstrap sample of the training data, introducing variability and reducing overfitting.
    - Since this is a regression task, the final prediction is the average of the predictions from all 200 trees. Using more trees increases the stability of the predictions, as individual tree errors are averaged out.

- **max_depth**: 10
    - This parameter limits the depth of each decision tree to a maximum of 10 levels, allowing the trees to capture more complex patterns in the data compared to shallower trees.
    - A deeper tree (with a higher max_depth) reduces bias because the model can better fit the training data. However, deeper trees are more prone to overfitting. By setting max_depth=10, we aim to allow the model to capture meaningful patterns while avoiding capturing noise in the training set.

- **max_features**: None
    - We kept the max_features to None since using all features significantly improves the model's ability to make accurate predictions.

## Conclusion

**Random Forest 1 (n_estimators=100, max_depth=15, max_features=None):**
This model achieved a strong balance between predictive accuracy and computational efficiency, with relatively low MSE and RMSE values on both training and validation sets. Using 100 trees allowed the model to reduce variance, while setting a max depth of 15 limited overfitting and helped capture underlying patterns in the data. The results suggest good predictive performance and reasonable generalization to new data.

**Random Forest 2 (n_estimators=200, max_depth=15, max_features=None):**
In this model, increasing the number of trees to 200 and setting max_depth to 15 allowed the model to capture more complex patterns, resulting in slightly improved performance metrics. However, this configuration showed signs of overfitting, with higher accuracy on the training set than the validation set. The inclusion of all features (max_features=None) enhanced model accuracy by fully utilizing the available information, though it slightly increased variance.

**Random Forest 3 (n_estimators=200, max_depth=10, max_features=None): [BEST MODEL]**
This model sought to find a middle ground by using 200 trees while keeping max_depth at 10. This setup provided a tradeoff between variance and bias, with reasonably low MSE and RMSE on both the training and validation sets. The reduced depth helped control overfitting, making this model effective for capturing key data patterns without high risk of overfitting, enhancing its generalizability. Due to the balance that this model achieved, Random Forest 3 with 200 estimators, a maximum depth of 10, and no maximum number of features is our best model.

Appendix 1.A Metrics Table

| Model Type | Metrics | Training Set Values | Validation Set Values | Testing Set Values |
|---|---|---|---|---|
| **Random Forest 1**<br><br>**{'n_estimators': 100, 'max_depth': 15, 'max_features': None}** | **MSE** | 1.7703 | 3.0264 | 2.9568 |
| | **RMSE** | 1.3305 | 1.7397 | 1.7195 |
| | **$R^2$** | 0.9998 | 0.9996 | 0.9997 |
| **Random Forest 2**<br><br>**{'n_estimators': 200, 'max_depth': 15, 'max_features': None}** | **MSE** | 1.7664 | 3.0131 | 2.9543 |
| | **RMSE** | 1.3290 | 1.7358 | 1.7188 |
| | **$R^2$** | 0.9998 | 0.9996 | 0.9997 |
| **Random Forest 3**<br><br>**{n_estimators': 200, 'max_depth': 10, 'max_features': None}** | **MSE** | 3.3915 | 3.8244 | 3.8193 |
| | **RMSE** | 1.8416 | 1.9556 | 1.9543 |
| | **$R^2$** | 0.9996 | 0.9995 | 0.9996 |