Random Forest

The Overfitting-Underdogs
(Will, Sam, Adi)

What is Random Forest?

Random Forest is an algorithm that combines multiple decision trees to make more accurate predictions than any individual tree.

Random Forest works by creating a forest of decision trees, where each tree is trained on a random subset of the training data and a random subset of the columns.

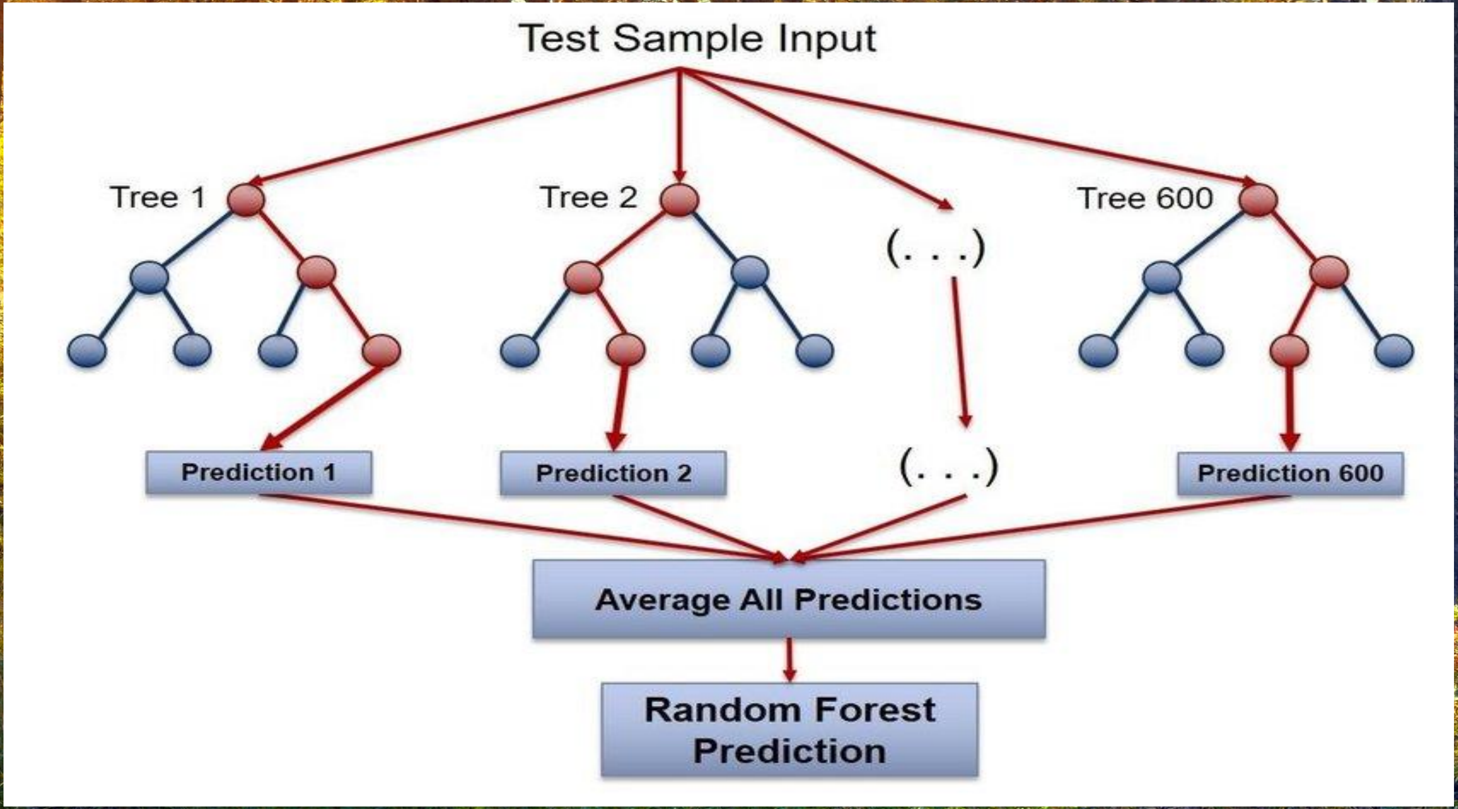
Random Forest (Regression) makes predictions by averaging the predictions from all the independent trees created.

Target DataSets?

Random Forest is considered one of the most powerful and versatile machine learning algorithms, and is used in a wide range of applications such as:

- Bioinformatics
- Finance
- Image Classification

How does the algorithm work intuitively?

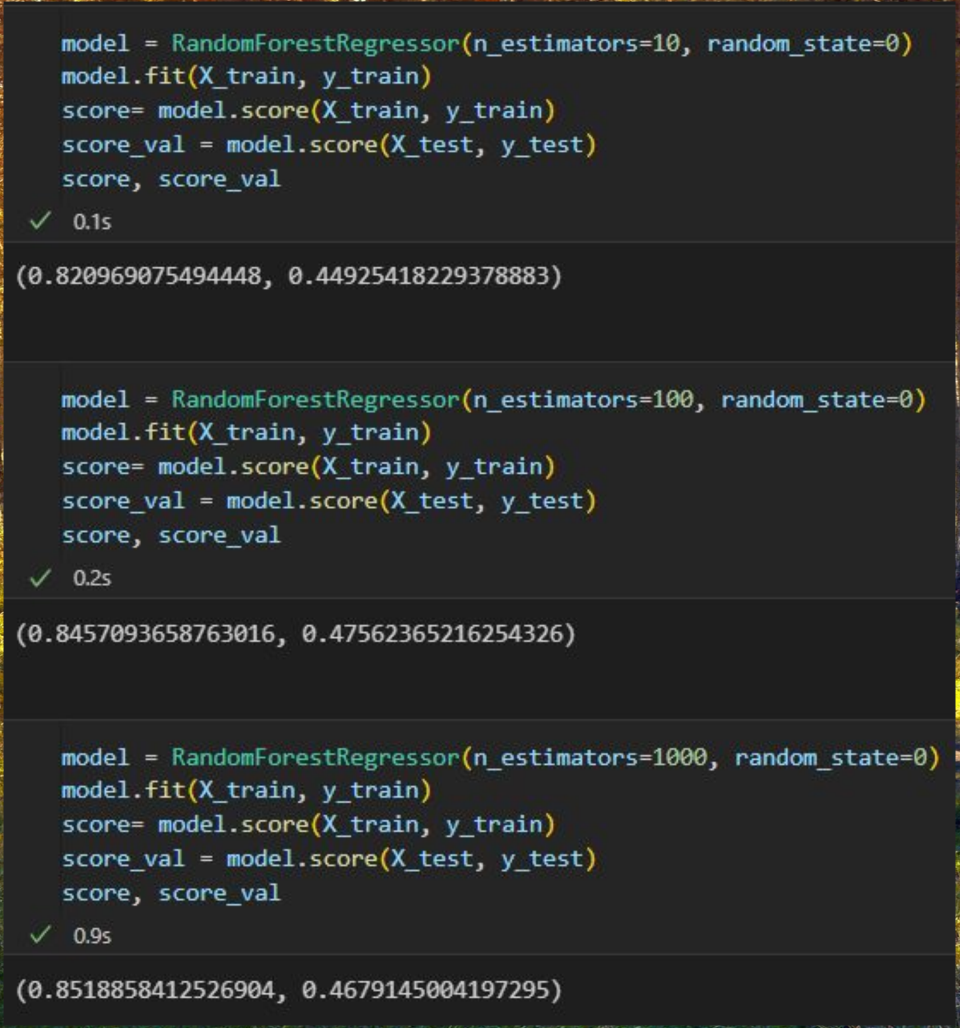


Advantages of Random Forest

1. Reduces the overfitting problem in decision trees. reducing variance and and therefore improving accuracy.
2. Less sensitive to changes in training data than decision tree.
3. Can be used for both classification and regression problems
4. Works well on both categorical and continuous variables.
5. No feature scaling is required as it uses a rule based approach rather than distance calculation (Linear Regression)
6. Some implementations can handle missing values and outliers automatically (Not SciKit Learn version)

Disadvantages of Random Forest

1. **Complexity** -It is common to create a large number of trees in a random forest (SciKit Learn version defaults to 1000 trees). This requires much more computational power and resources compared to decision tree.
2. **Longer Training Period** - Also due to the fact that it generates many trees, random forest requires much more time to train.
3. **Bias** towards categorical variables with many categories - which can result in lower accuracy for these variables
4. **Lack of extrapolation** - it is not suitable for predicting values outside of the range of the training data
5. **Difficult to interpret** - challenging to understand how the model makes its predictions
6. **Overfitting** - This problem can still easily occur if the random forest is not well-tuned
7. **Model Selection** - Due to multiple hyperparameters that can be tuned, Model selection becomes more difficult



```
model = RandomForestRegressor(n_estimators=10, random_state=0)
model.fit(X_train, y_train)
score= model.score(X_train, y_train)
score_val = model.score(X_test, y_test)
score, score_val
```

✓ 0.1s

```
(0.820969075494448, 0.44925418229378883)
```

```
model = RandomForestRegressor(n_estimators=100, random_state=0)
model.fit(X_train, y_train)
score= model.score(X_train, y_train)
score_val = model.score(X_test, y_test)
score, score_val
```

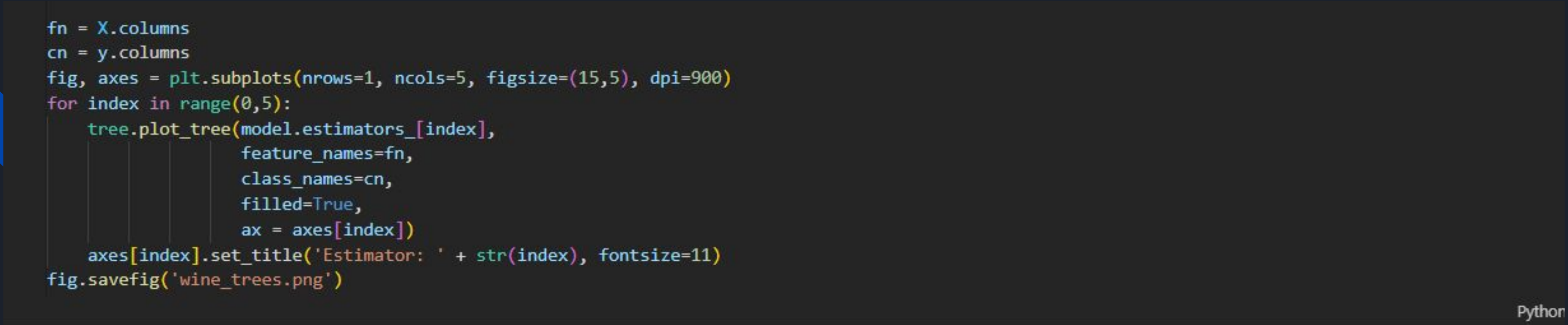
✓ 0.2s

```
(0.8457093658763016, 0.47562365216254326)
```

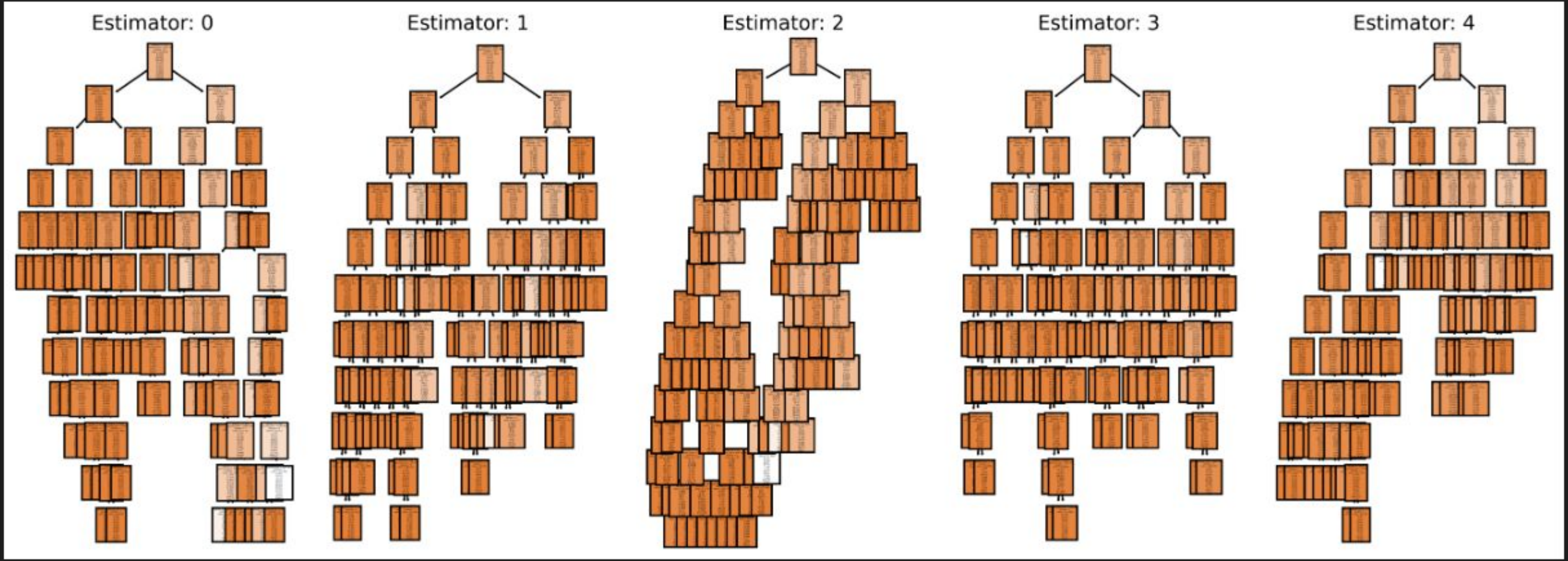
```
model = RandomForestRegressor(n_estimators=1000, random_state=0)
model.fit(X_train, y_train)
score= model.score(X_train, y_train)
score_val = model.score(X_test, y_test)
score, score_val
```

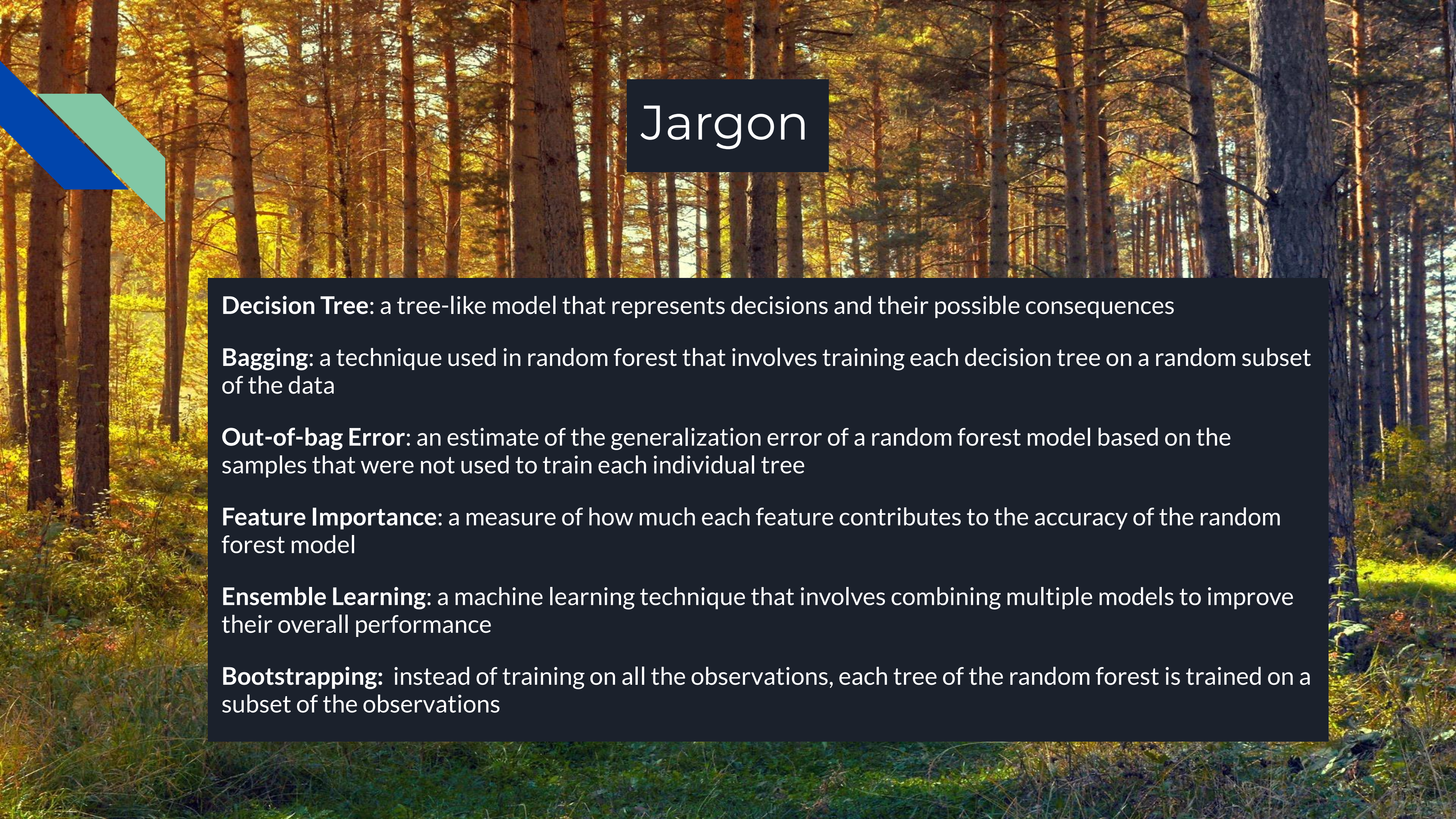
✓ 0.9s

```
(0.8518858412526904, 0.4679145004197295)
```



```
fn = X.columns
cn = y.columns
fig, axes = plt.subplots(nrows=1, ncols=5, figsize=(15,5), dpi=900)
for index in range(0,5):
    tree.plot_tree(model.estimators_[index],
                   feature_names=fn,
                   class_names=cn,
                   filled=True,
                   ax = axes[index])
    axes[index].set_title('Estimator: ' + str(index), fontsize=11)
fig.savefig('wine_trees.png')
```





Jargon

Decision Tree: a tree-like model that represents decisions and their possible consequences

Bagging: a technique used in random forest that involves training each decision tree on a random subset of the data

Out-of-bag Error: an estimate of the generalization error of a random forest model based on the samples that were not used to train each individual tree

Feature Importance: a measure of how much each feature contributes to the accuracy of the random forest model

Ensemble Learning: a machine learning technique that involves combining multiple models to improve their overall performance

Bootstrapping: instead of training on all the observations, each tree of the random forest is trained on a subset of the observations

What hyperparameters exist?

N_estimators: the number of trees in the forest

Criterion: The function to measure the quality of a split ex(“squared_error”, “absolute_error”, “friedman_mse”, “poisson”). Defaults to “squared_error”

Max Depth: The maximum depth of the tree. If none is selected then nodes are all expanded until leaves contain less than the min_samples_split split,

Min_samples_split: the minimum number of samples required to split an internal node

Min_samples_leaf: the minimum number of samples needed to be a leaf node

Max_features: The number of features to consider when looking for a best split

Links & Resources

Overview of the Random Forest Algorithm:
https://www.mygreatlearning.com/blog/random-forest-algorithm/

Random Forest Algorithm Explained (Youtube):
https://www.youtube.com/watch?v=v6VJ2RO66Ag

Random Forest Classifier vs Regressor:
https://pianalytix.com/random-forest-classifier-and-regressor/

When to use Random Forests: https://www.kaggle.com/questions-and-answers/97925

Advantages and Disadvantages of Random Forest Algorithm in Machine Learning:
http://theprofessionalspoint.blogspot.com/2019/02/advantages-and-disadvantages-of-random.html