Random Forest
Group 22

- Amanda Aleong
- Francis Cruz
- Han Hu
- Youngmok Ko
- Hao Xing
- Shi Yu

Random Forest

- Supervised ML algorithm used for classification or regression
- Ensemble technique which utilizes multiple decision trees

- The power of crowd
- Another example of an ensemble technique:
  - Bagging

Exploring a Random Forest

Planting a Random Forest: n_estimator = 3

Sample#	F1	F2	F3
1			
2			
3			
4			
5			
6			

Set aside as test data

Planting a Random Forest: n_estimator = 3

Sample#	F1	F2	F3
1			
2			
3			
4			
5			

Step 1: Bootstrap sample

Planting a Random Forest: n_estimator = 3

Sample#	F1	F2	F3
1			
2			
3			
4			
5			

Step 1: Bootstrap sample

Sample#	F1	F2	F3
1			
2			
4			

Step 2: Select m features

Setting Up Estimator #1

Planting a Random Forest: n_estimator = 3

Sample#	F1	F2	F3
1			
2			
3			
4			
5			

Step 1: Bootstrap sample

Sample#	F1	F2	F3
1			
2			
4			

Step 2: Select m features

Sample#	F1	F3
1		
2		
4		

Step 3: Pass as input to tree# 1

Setting Up Estimator #1

Planting a Random Forest

- Create n bootstrap sample sets
- Select m features
- Train each tree in parallel
- Aggregate outcome of each tree by taking
  - the mode of the classes (classification) or
  - the mean of the prediction (regression)

Sample#	F1	F2	F3
1			
2			
3			
4			
5			

Step 1: Bootstrap sample

Sample#	F1	F2	F3
2			
4			
5			

Step 2: Select m features

Sample#	F1	F2
2		
4		
5		

Step 3: Pass as input to tree# 2

Setting Up Estimator #2

Planting a Random Forest

- Create n bootstrap sample sets
- Select m features
- Train each tree in parallel
- Aggregate outcome of each tree by taking
  - the mode of the classes (classification) or
  - the mean of the prediction (regression)

Sample#	F1	F2	F3
1			
2			
3			
4			
5			

Step 1: Bootstrap sample

Sample#	F1	F2	F3
2			
3			
4			

Step 2: Select m features

Sample#	F2	F3
2		
3		
4		

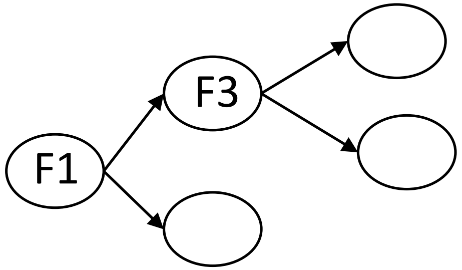
Step 3: Pass as input to tree# 3

Setting Up Estimator #3

Growing a Random Forest

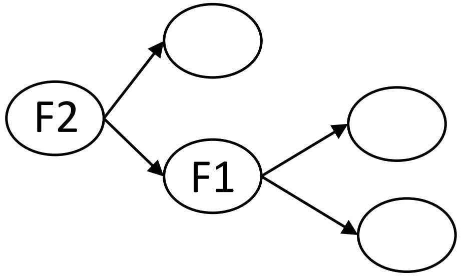
n1

Sample#	F1	F3
1		
2		
4		



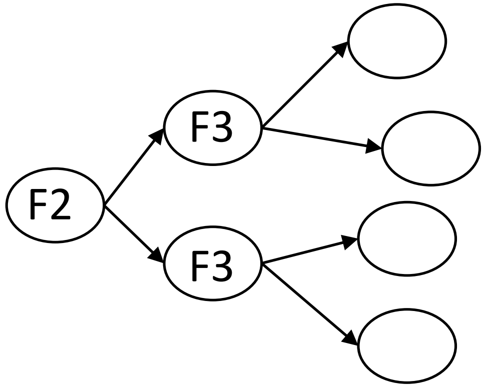
n2

Sample#	F1	F2
2		
4		
5		



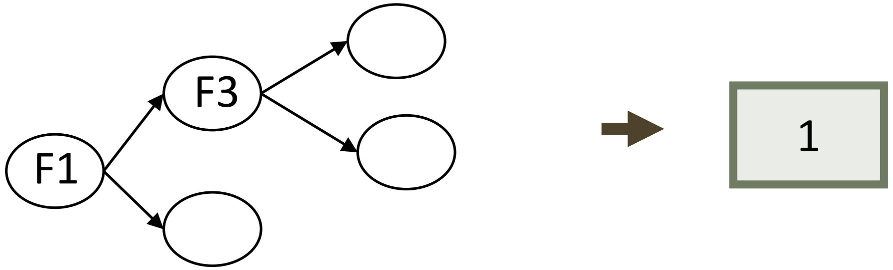
n3

Sample#	F2	F3
2		
3		
4		

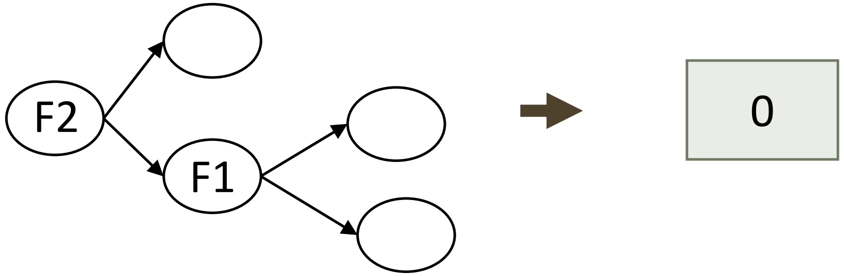


Roaming a Random Forest

Sample#	F1	F2	F3
6			

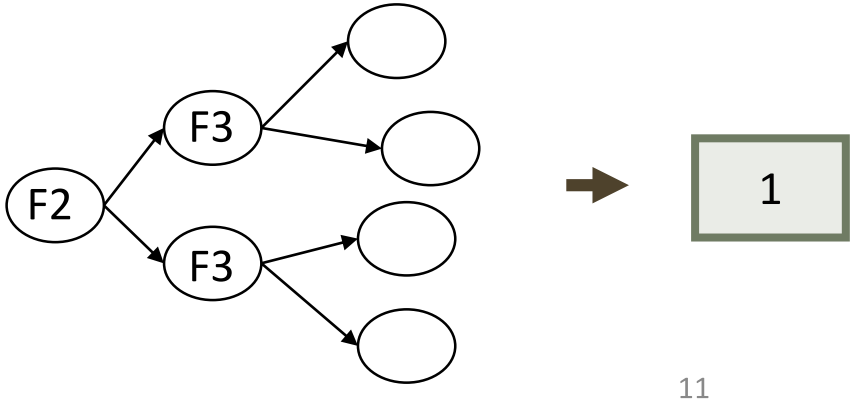


Sample#	F1	F2	F3
6			



Hyperparameters:
Min_samples_leaf
Criterion
Max_depth
N_estimators
Max_features

Sample#	F1	F2	F3
6			

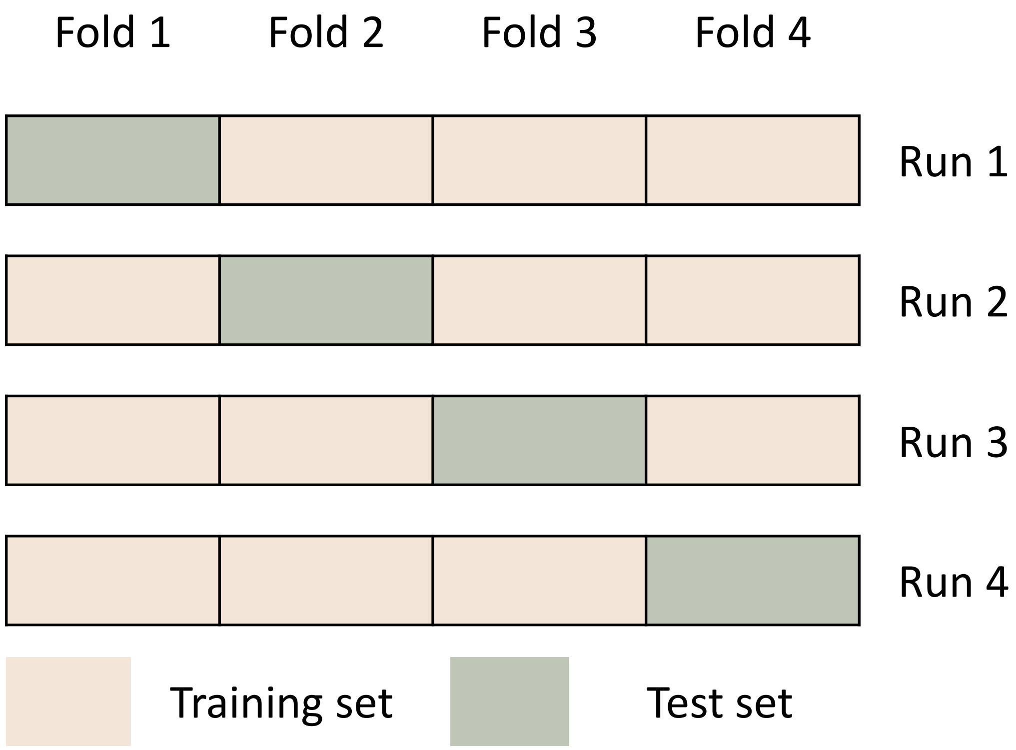


Difference Between RF and Bagging

	Bagging	Random Forest
Ensemble Technique	Yes	Yes
Uses Decision Trees	Sometimes	Always
Bootstrap Sampling	Yes	Yes
Feature space per tree	Uses all M features	Uses m features <M
Reduces Variance	Yes	Yes
Reduces correlation	No	Yes

Random Forest and Cross Validation

Cross-Validation:



Random Forest

- Random forest already subsamples the data set to train each tree.
  - Out-of-bag score: aggregated accuracy from trees where a sample was not a part of the subset used for training.

Random Forest

Advantages

Handle missing values and maintains accuracy for missing data

Less likely to overfit the model

Handle large data set with higher dimensionality

Disadvantages

Computational complexity (Computationally expensive/takes long to solve)

Feels like a black box approach (Loss of explainability)

Python Code: Import Libraries and Data

- Library

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import DecisionTreeRegressor
```

- Import Data

```
UCI_data_URL = 'https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data'
names = ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'num']
heart_disease = pd.read_csv(urlopen(UCI_data_URL), names=names)
```

```
# load testing and training data provided in https://www.kaggle.com/c/house-prices-advanced-regression-techniques
train_housing_data = pd.read_csv('train_housing.csv')
test_housing_data = pd.read_csv('test_housing.csv')
```

Python Code: Decision Tree Classification

- Instance

```
fit_dt = DecisionTreeClassifier(random_state=42)
```

- Set Parameters

```
fit_dt.set_params(criterion = 'gini',
                  max_depth = 4,
                  max_features = 'sqrt',
                  min_samples_leaf = 10)
```

- Training

```
fit_dt.fit(X_train, y_train)
```

- Predict

```
y_test_predict = fit_dt.predict(X_test)
```

- Assess

```
scores_DT_gridsearch_classifier = cross_val_score(fit_dt, X, y, cv = 5)
ave_score_DT_gridsearch_classifier = np.mean(scores_DT_gridsearch_classifier)*100
print('The average score using CV is %.2f%%' % ave_score_DT_gridsearch_classifier)
```

```
The average score using CV is 73.29%
```

Python Code: Random Forest Classification

- Instance

```
fit_rf = RandomForestClassifier(random_state=42,oob_score = False)
```

- Set Parameters

```
fit_rf.set_params(bootstrap = True,
                  criterion = 'entropy',
                  max_depth = 3,
                  max_features = 'sqrt',
                  min_samples_leaf = 2,
                  n_estimators=90,
                  oob_score = True)
```

- Training

```
fit_rf.fit(X_train, y_train)
```

- Predict

```
y_test_predict = fit_rf.predict(X_test)
```

```
ave_oob_score_RF_best_classifier = (fit_rf.oob_score_)*100
print('The average score using Out-of-Bag estimate is %.2f%%' % ave_oob_score_RF_best_classifier)
```

```
The average score using Out-of-Bag estimate is 84.62%
```

- Assess

```
scores_RF_best_classifier = cross_val_score(fit_rf, X, y, cv = 5)
ave_cv_score_RF_best_classifier = np.mean(scores_RF_best_classifier)*100
print('The average score using CV is %.2f%%' % ave_cv_score_RF_best_classifier)
```

```
The average score using CV is 83.78%
```

Python Code: Decision Tree Regression

- Instance
- Set Parameters
- Training
- Predict
- Assess

```
fit_dtr = DecisionTreeRegressor(random_state=42)
fit_dtr.set_params(max_depth= 45,
                   max_features= None,
                   min_samples_leaf= 8)
fit_dtr.fit(X_train,y_train)
y_test_predict = fit_dtr.predict(X_test)
```

```
# cross validation with cv = 5
scores_DT_regressor = cross_val_score(fit_dtr, X_features_train_file, y_train_file, cv = 5)
ave_score_DT_regressor = np.mean(scores_DT_regressor)*100
print('The average score using CV is %.2f%%' % ave_score_DT_regressor)
```

```
The average score using CV is 78.04%
```

Python Code: Random Forest Regression

- Instance
- Set Parameters
- Training
- Predict
- Assess

```
RF_regressor = RandomForestRegressor(random_state=42)
```

```
RF_regressor.set_params(max_depth= 20,
                        max_features= 'sqrt',
                        min_samples_leaf= 2,
                        n_estimators=90,
                        oob_score = True,
                        criterion = 'mse')
```

```
RF_regressor.fit(X_features_train_file, y_train_file)
```

```
y_test_file = RF_regressor.predict(X_features_test_file)
```

```
# Out-of-Bag score
ave_oob_score_RF_best_classifier = (RF_regressor.oob_score_)*100
print('The average score using Out-of-Bag estimate is %.2f%%' % ave_oob_score_RF_best_classifier)
```

```
The average score using Out-of-Bag estimate is 83.61%
```

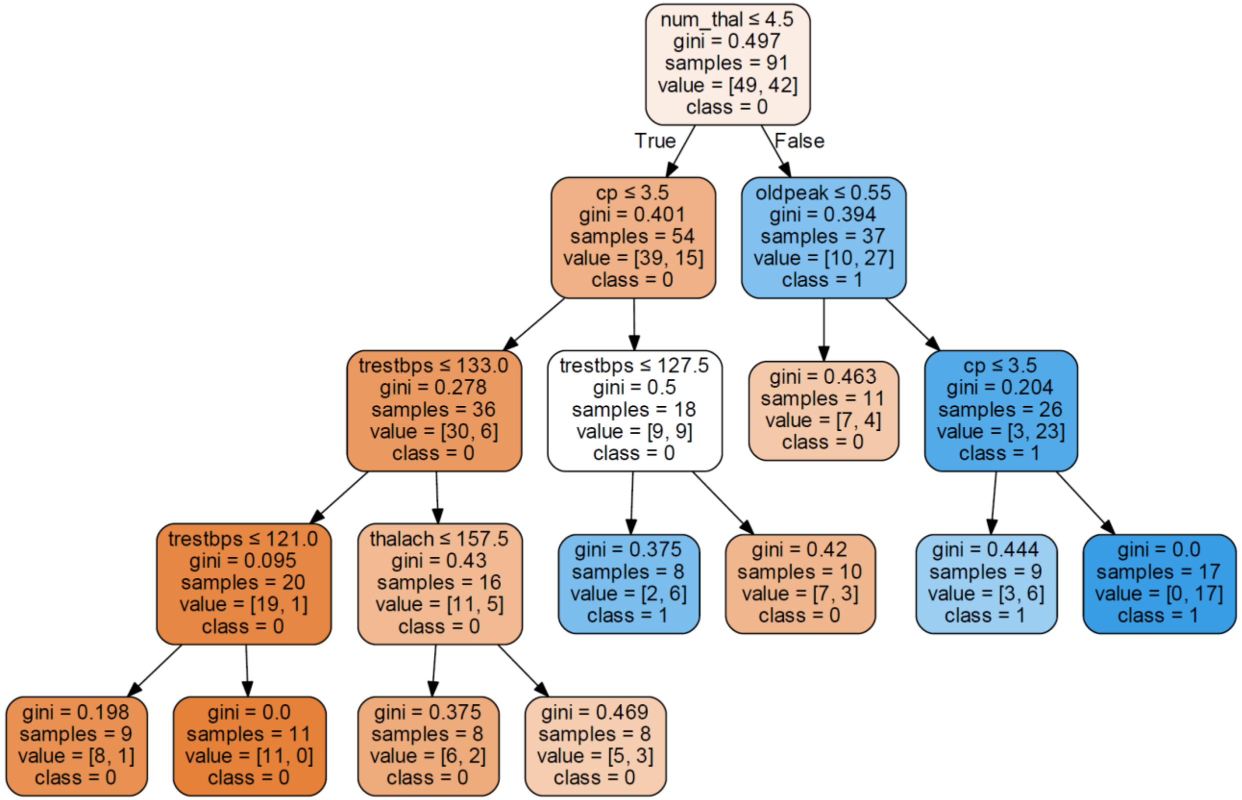
```
# cross validation with cv = 5
scores_RF_regressor = cross_val_score(RF_regressor, X_features_train_file, y_train_file, cv = 5)
ave_score_RF_regressor = np.mean(scores_RF_regressor)*100
print('The average score using CV is %.2f%%' % ave_score_RF_regressor)
```

```
The average score using CV is 83.85%
```

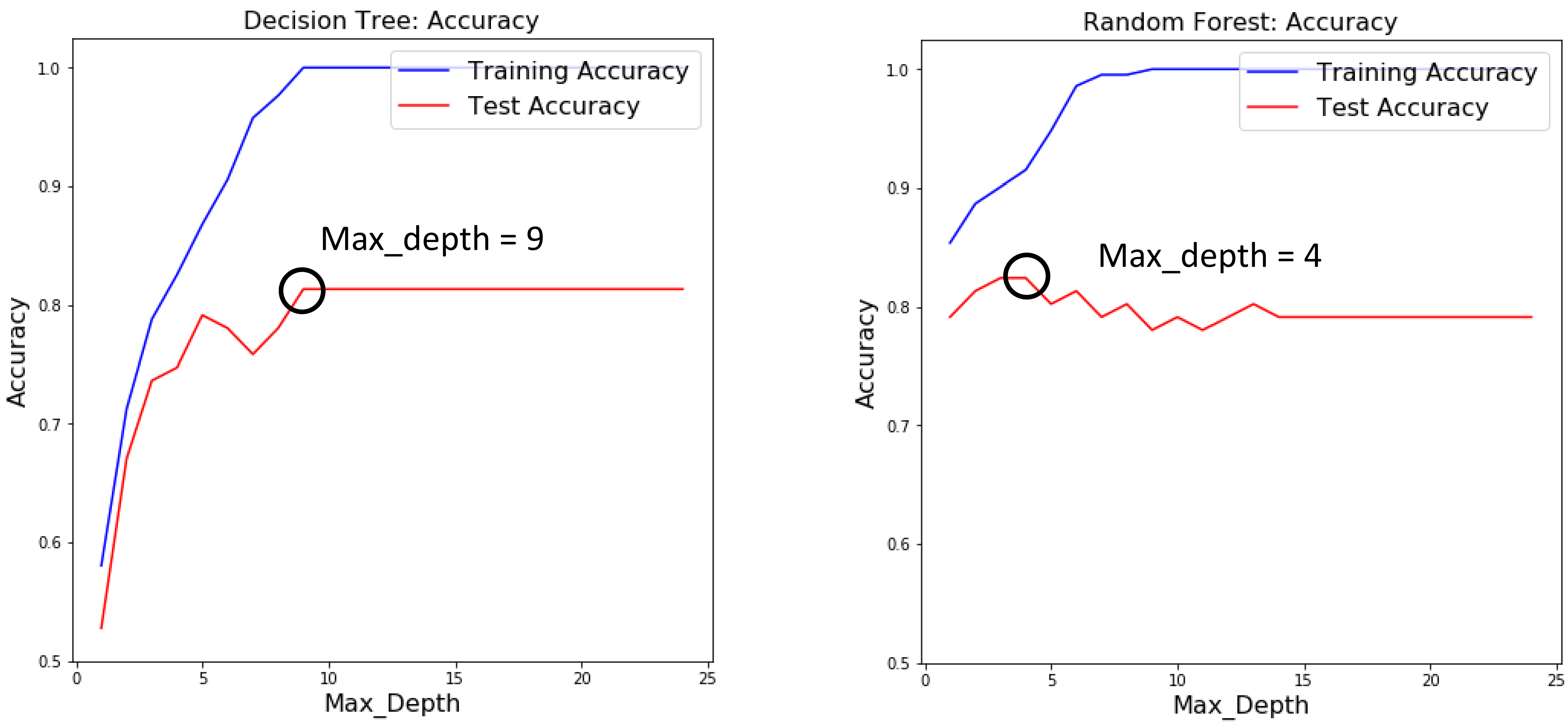
Classification

Heart Disease Prediction

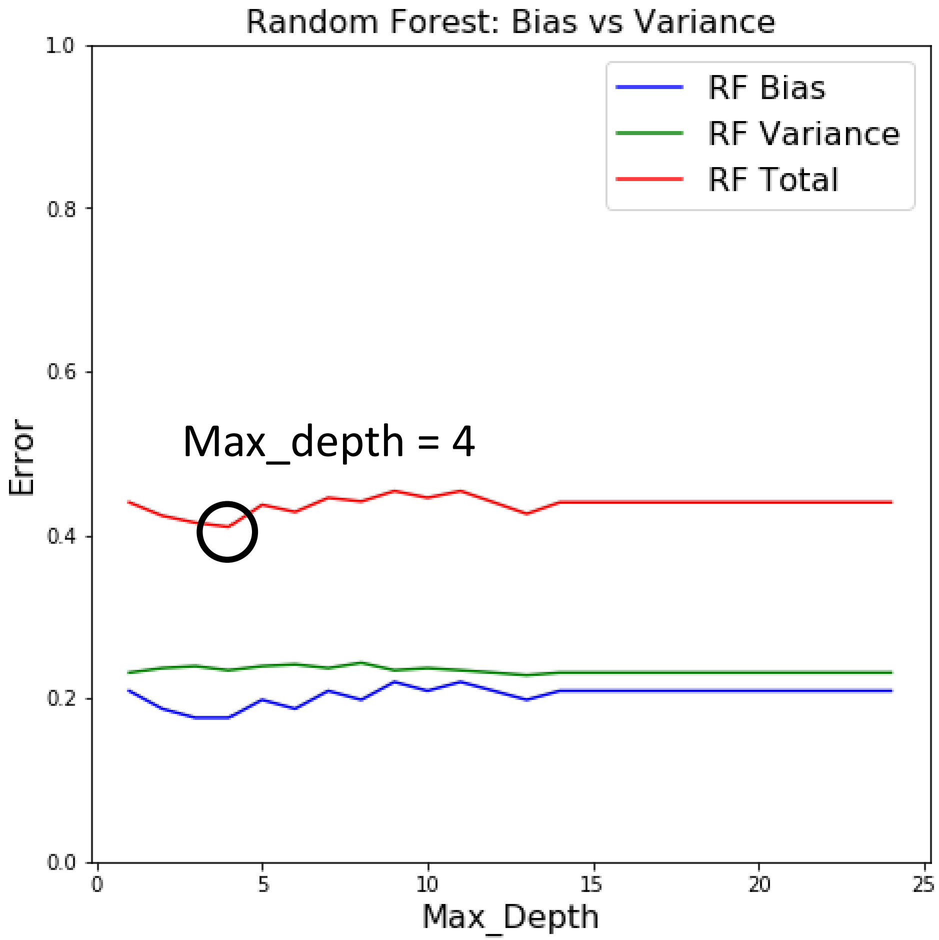
- Input features:
  - Medical history
    - i.e. Age, sex, num (previous heart diseases), cp (chest pain), thal (heart defect), etc.
- Binary output:
  - 1: Disease present
  - 0: Disease not present



Classification: Maximize Test Accuracy

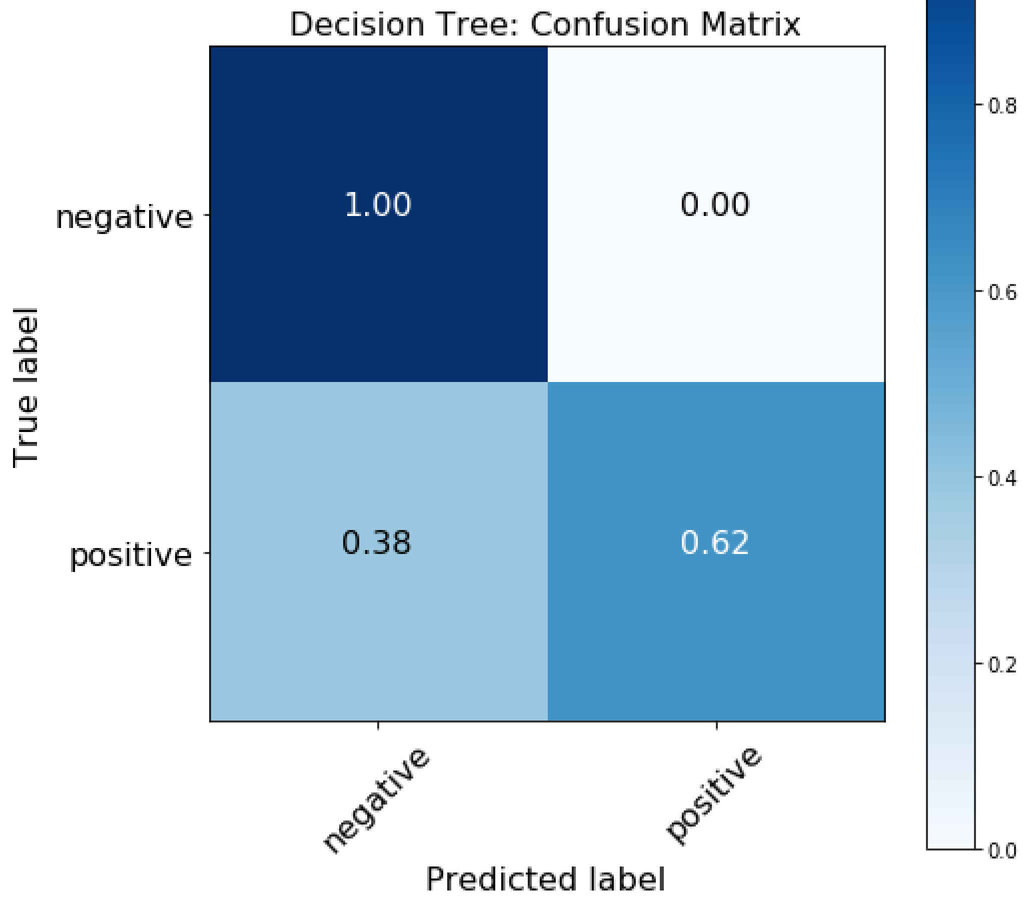


Classification: Minimize Bias & Variance

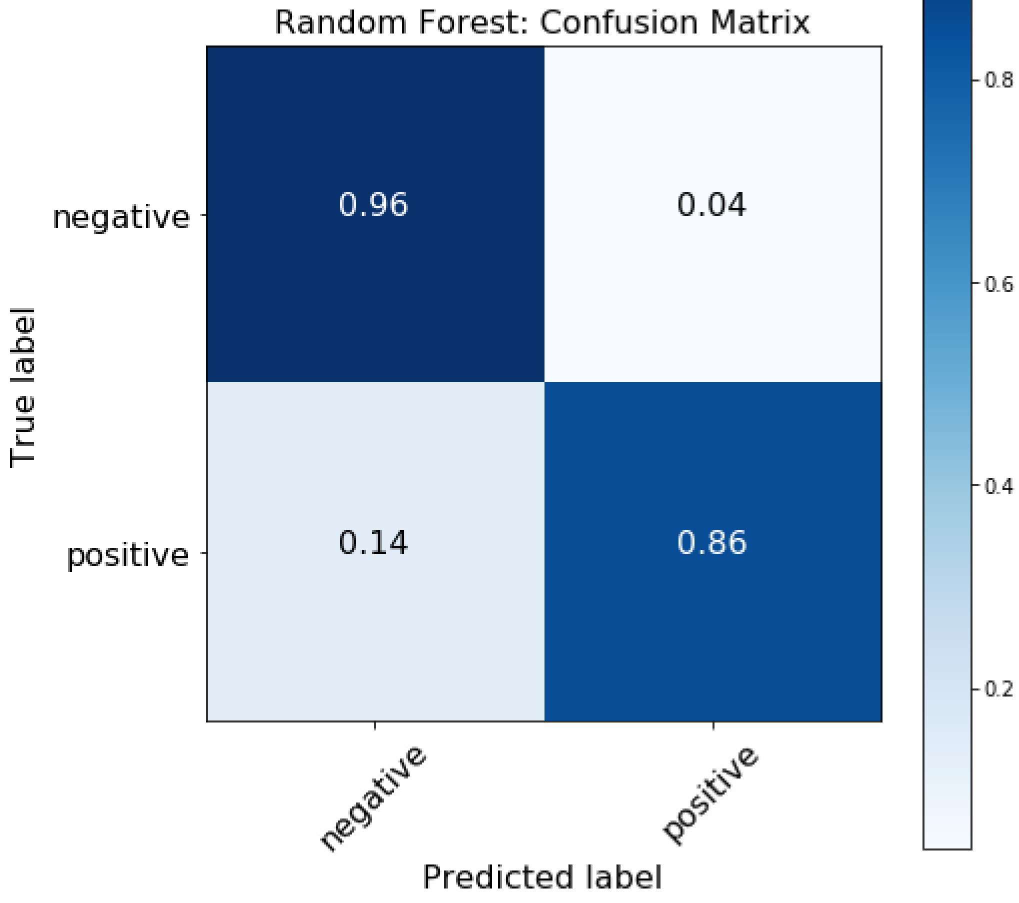


Confusion Matrix: Post-tuning

Decision Tree



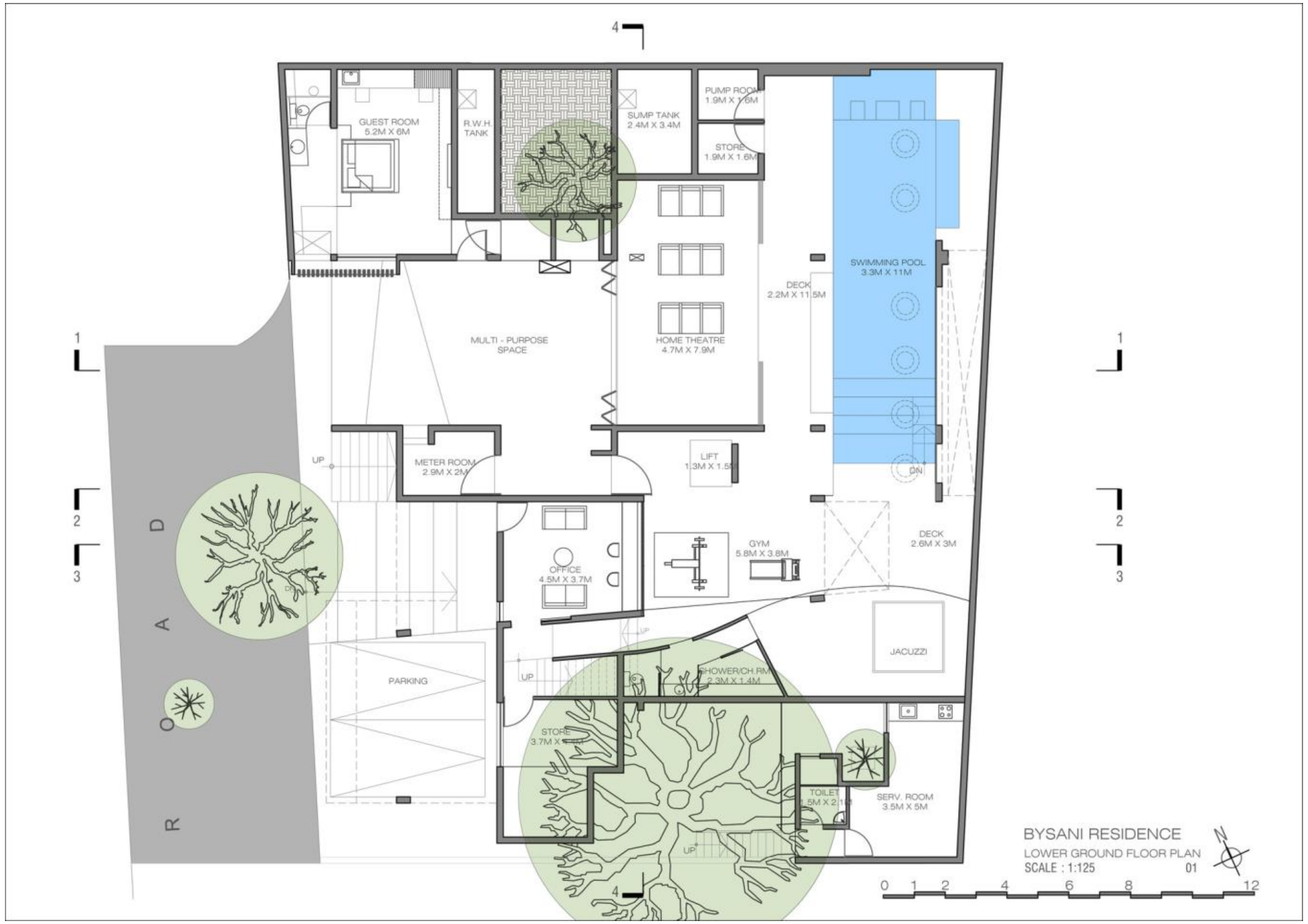
Random Forest



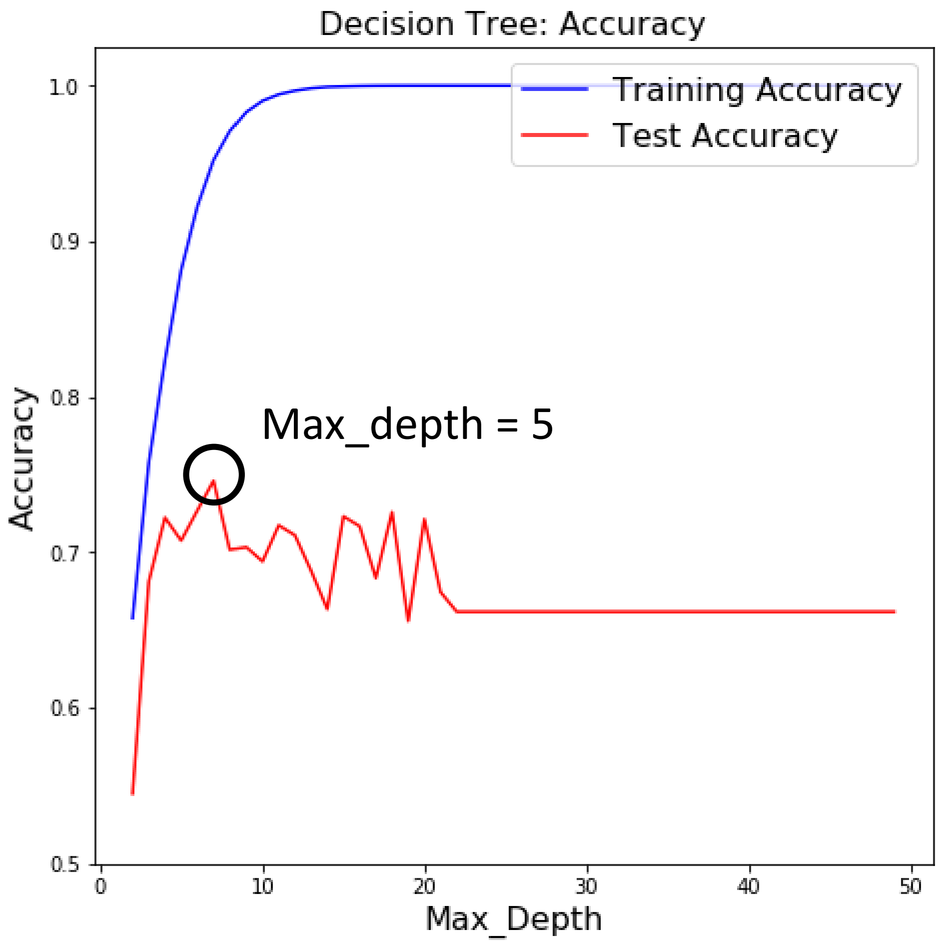
Regression

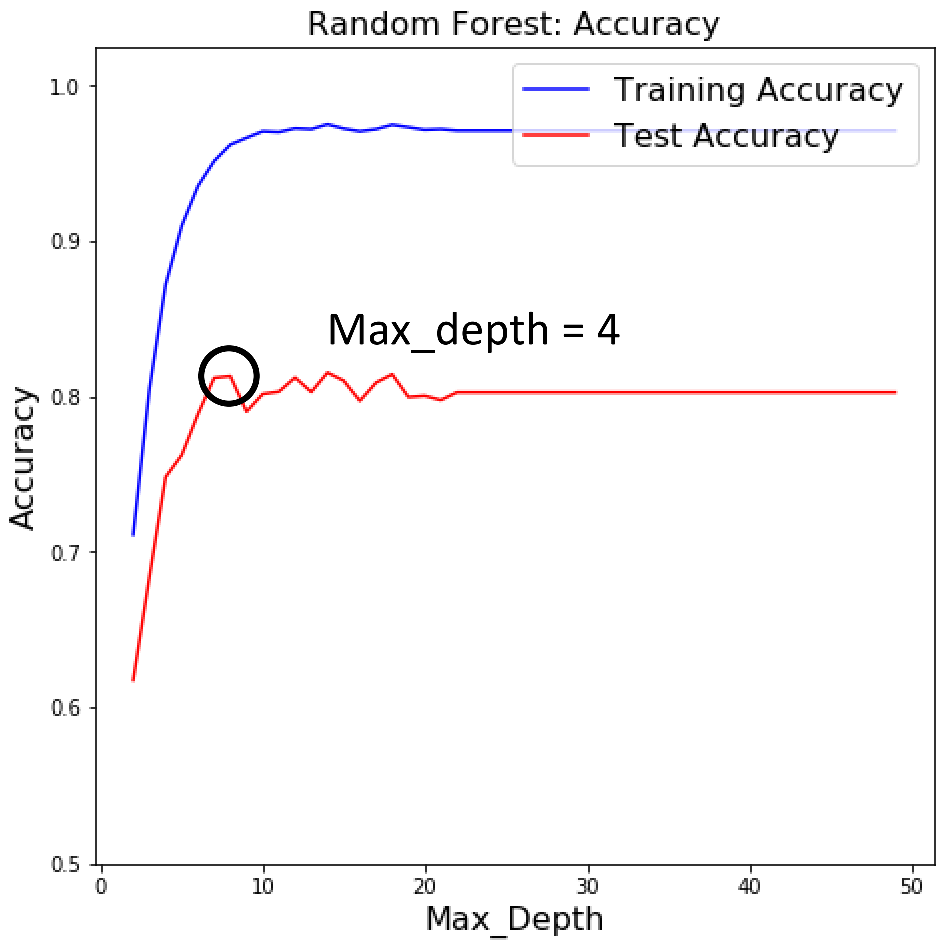
Housing Attribute Price Estimation

- Input features:
  - House amenities i.e. pool, garage, lot area, patio space, etc.
- Continuous output:
  - Estimated price of the house in thousands of USD

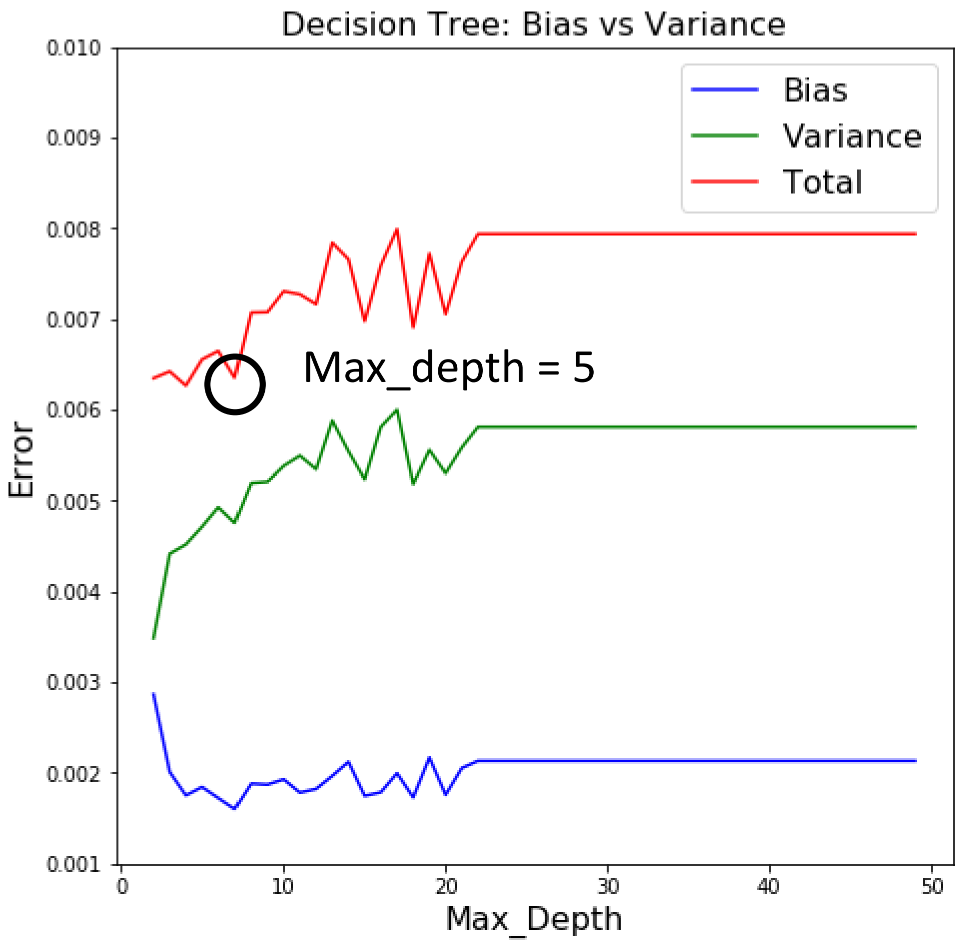


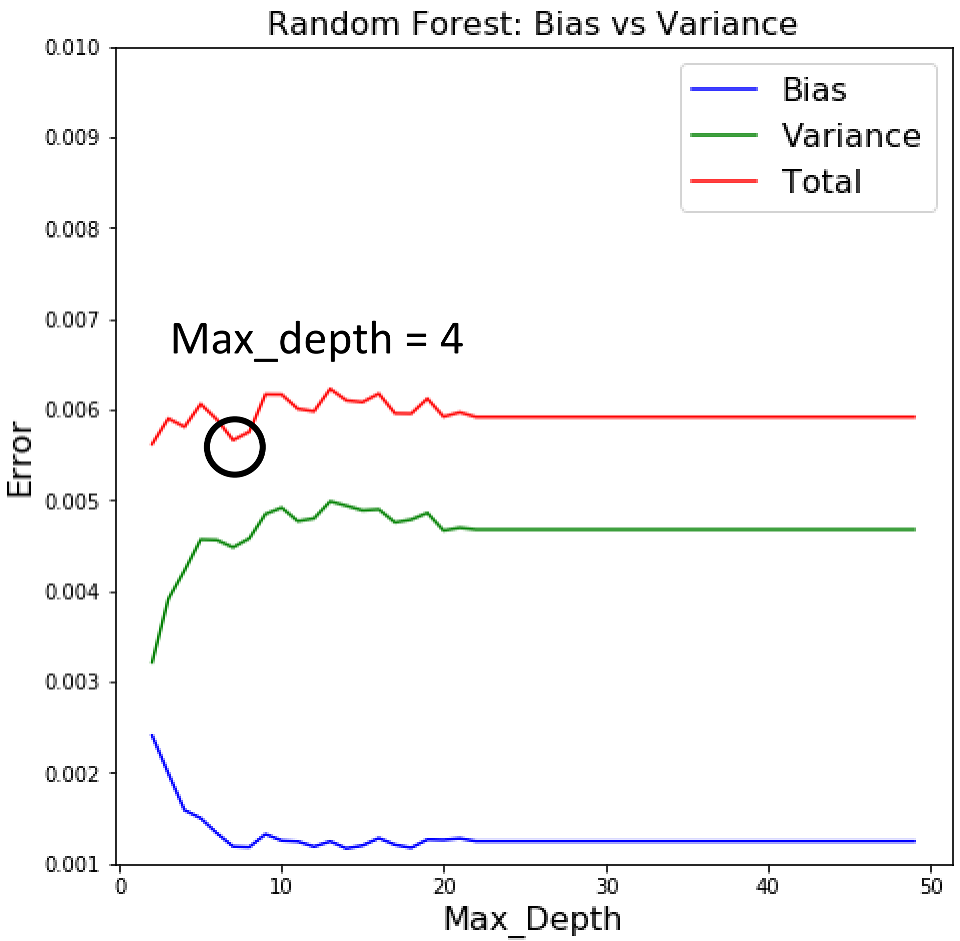
Regression: Maximize Test Accuracy





Regression: Minimize Bias & Variance





Conclusion

- ✓Demonstrated the use of both RF and DT for classification and regression problems
- ✓Discussed differences between bagging and RF
- ✓Discussed similarities and differences between cross validation and random forest

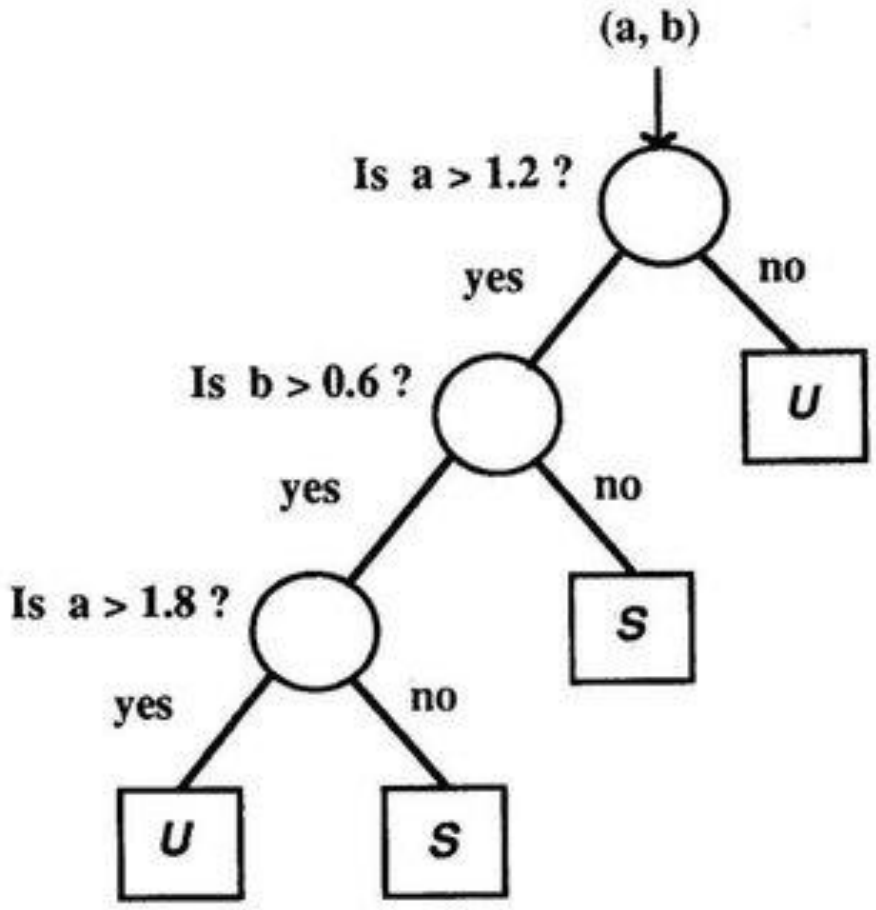

Thank you!

References

- https://scikit-learn.org/stable/auto_examples/ensemble/plot_ensemble_oob.html
- https://scikit-learn.org/stable/modules/tree.html#tree
- Cross Validation: http://blog.citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics
- Evaluation Metrics: https://scikit-learn.org/0.15/model_selection.html
- Hyperparameter Selection:
  - https://towardsdatascience.com/random-forests-and-the-bias-variance-tradeoff-3b77fee339b4
  - https://medium.com/@mohtedibf/indepth-parameter-tuning-for-decision-tree-6753118a03c3
- Decision Tree Score: https://github.com/scikit-learn/scikit-learn/blob/ef5cb84a/sklearn/base.py#L358
- https://stackoverflow.com/questions/8961586/do-i-need-to-normalize-or-scale-data-for-randomforest-r-package

Decision Trees: Basic Unit of a Random Forest

- Supervised learning used mostly for classification but can work for continuous inputs and outputs
- Uses a series of logic statements to determine the variable you are trying to predict.
- The goal of a decision tree to produce homogeneous subsets of the predicted variable.
- Advantages:
  - Easy to understand
  - Useful in data exploration
- Disadvantages
  - Overfitting



Appendix A: Boosting

- Ensemble technique where multiple models are trained in sequence.
- The misclassified samples of each tree are up-weighted when passed to the next tree so that each tree is building off the last.

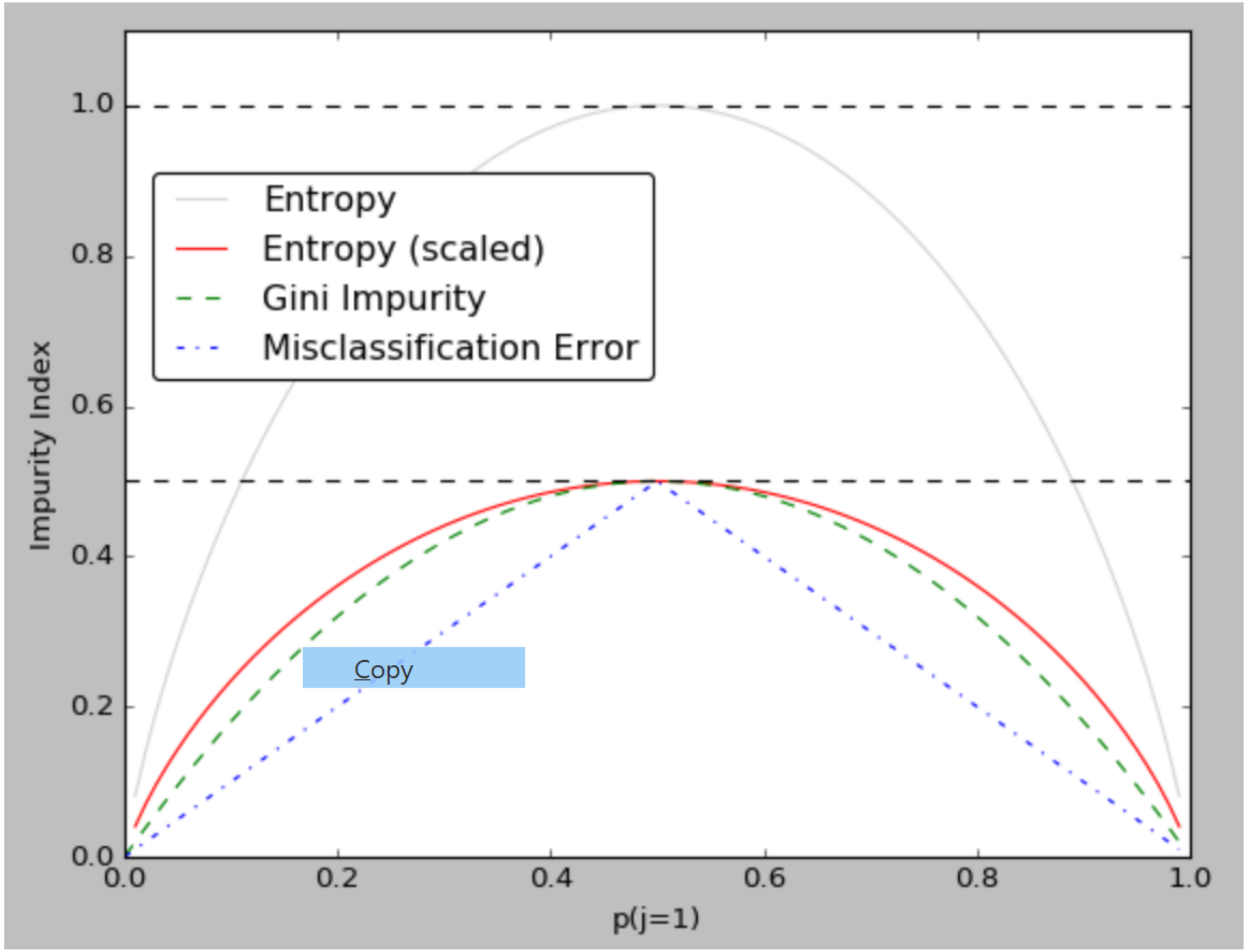
- https://www.analyticsvidhya.com/blog/2015/11/quick-introduction-boosting-algorithms-machine-learning/

Appendix B: Bias & Variance Formula

- Zero-one-loss function: $L(y, y') = \begin{cases} 1\ if\ y \neq y' \\ 0\ if\ y = y' \end{cases}$
- Bias Regression : $\frac{\sum(y - \hat{y})^2}{n}$
- Bias Classification: $\frac{\sum L(y, \hat{y})^2}{n}$
- Variance Regression : $\frac{\sum(\hat{y} - \bar{\hat{y}})^2}{n}$
- Variance Classification : $\frac{\sum L(\hat{y}, \bar{\hat{y}})^2}{n}$

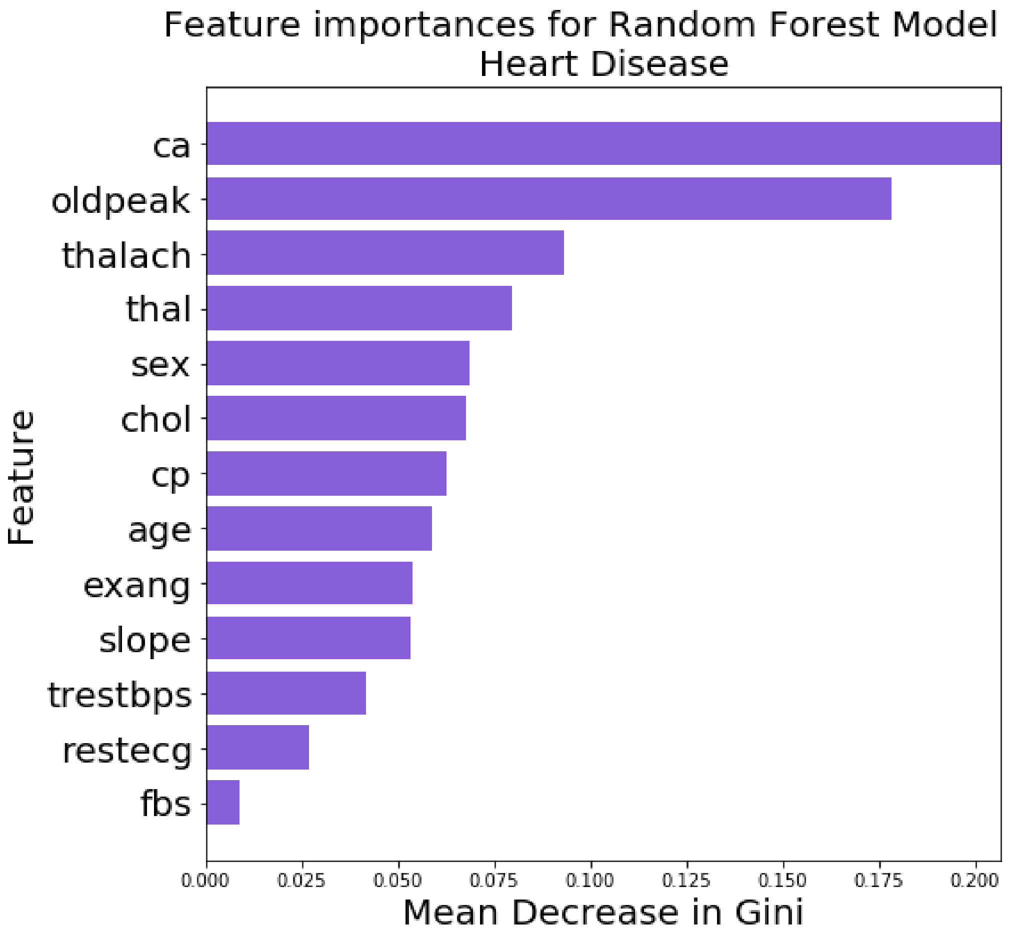
Appendix C: Loss Functions

- Entropy
- Gini Index
- Misclassification Error

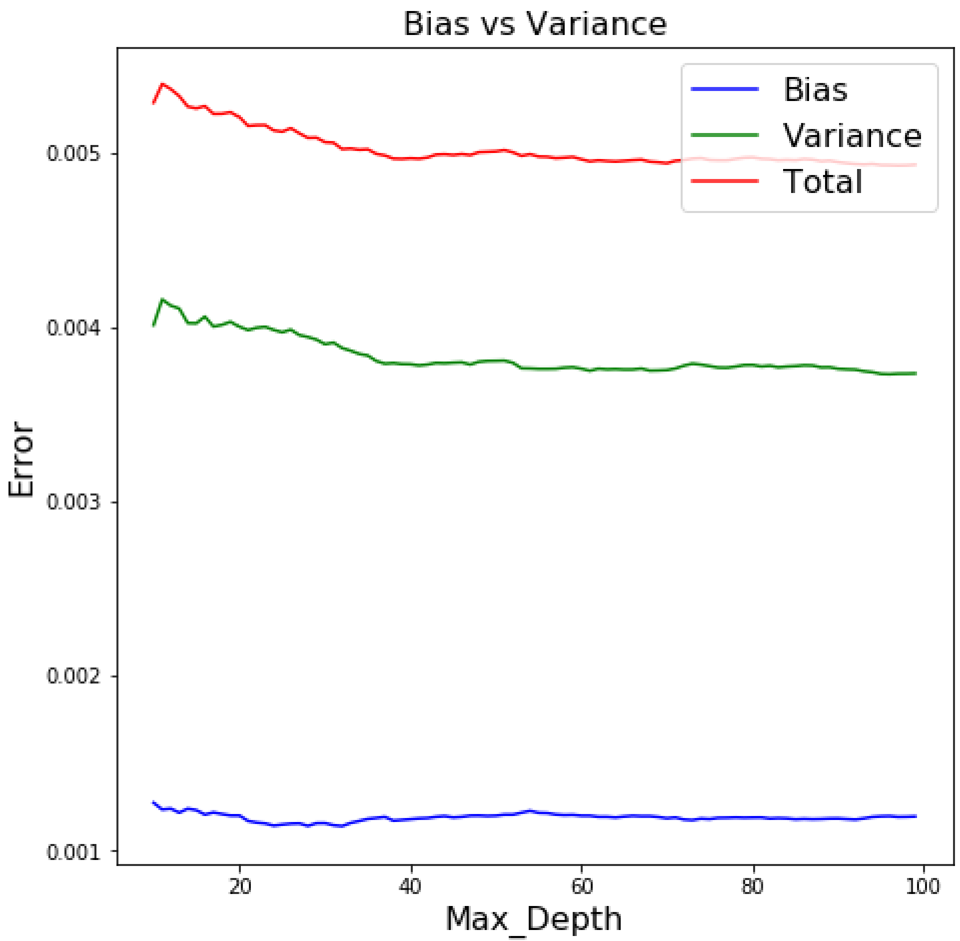


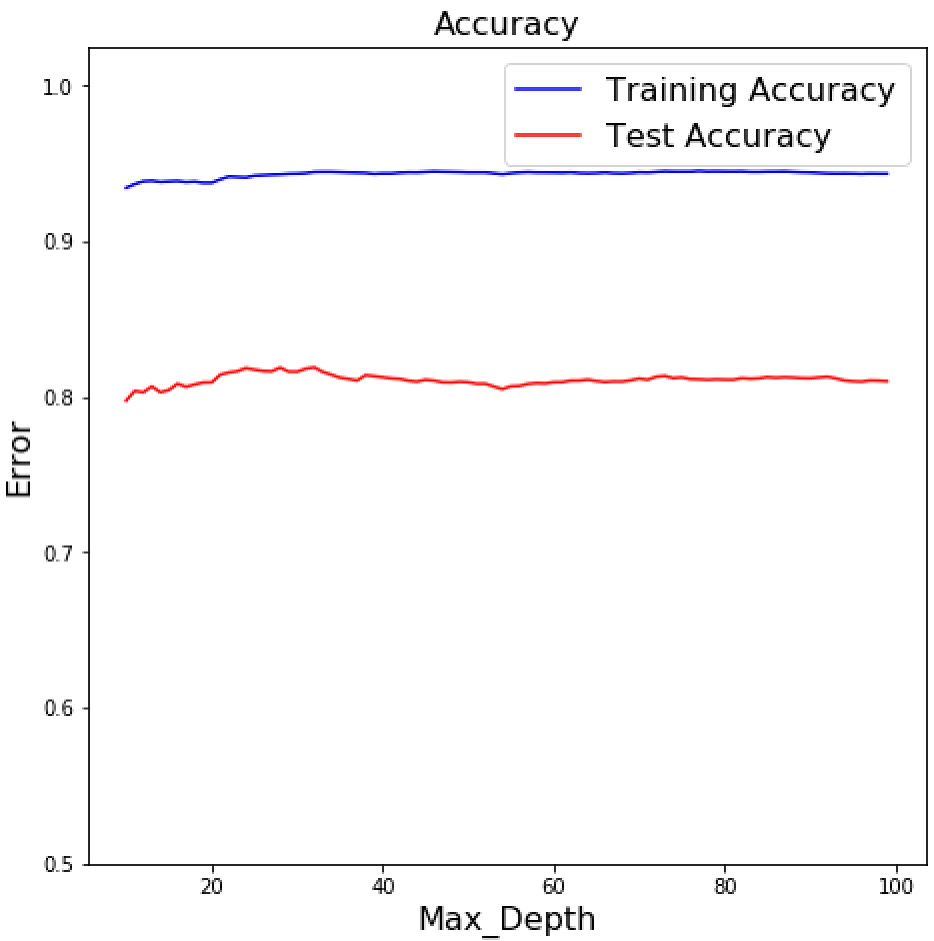
https://www.bogotobogo.com/python/scikit-learn/scikt_machine_learning_Decision_Tree_Learning_Informatioin_Gain_IG_Impurity_Entropy_Gini_Classification_Error.php

Appendix D: Variable Importance

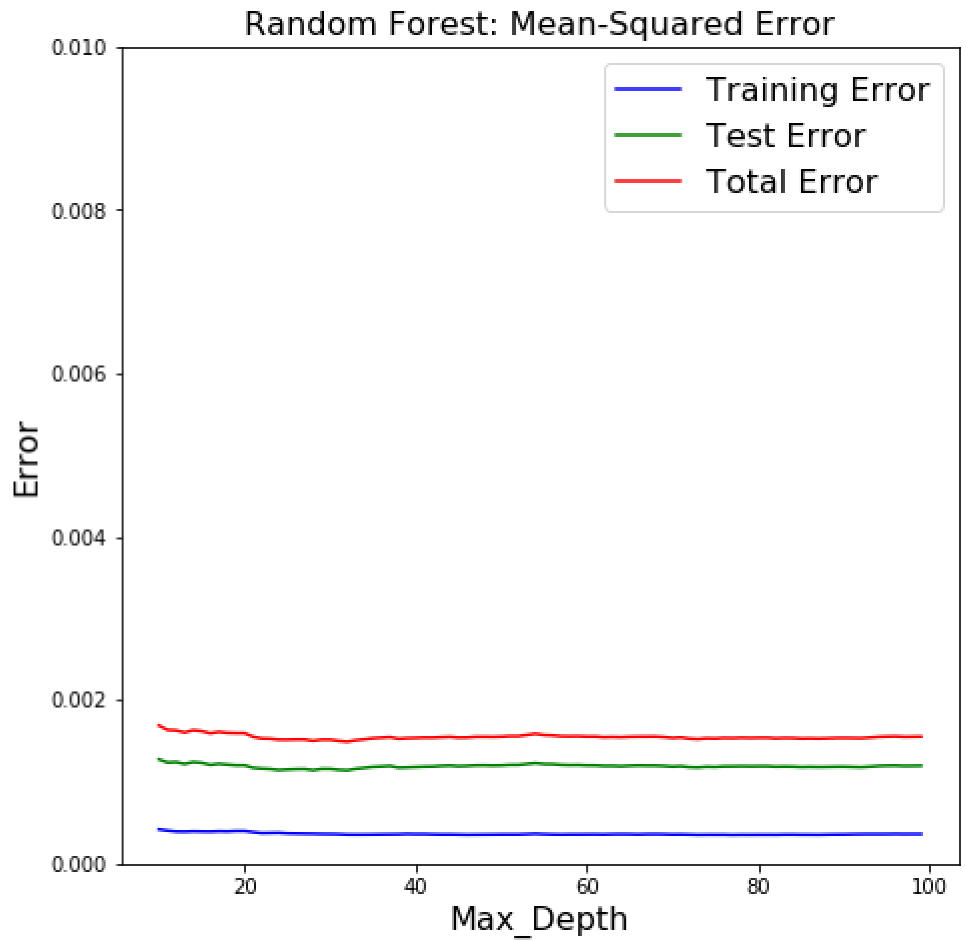


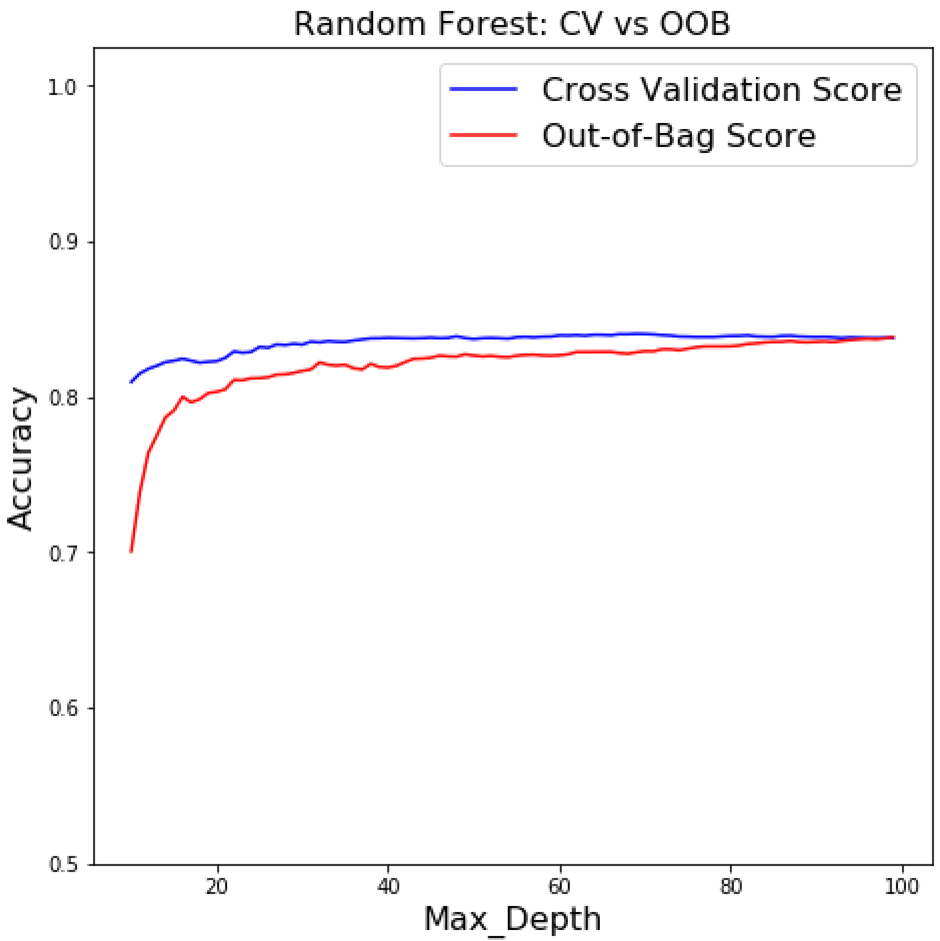
Regression: N_estimators tuning



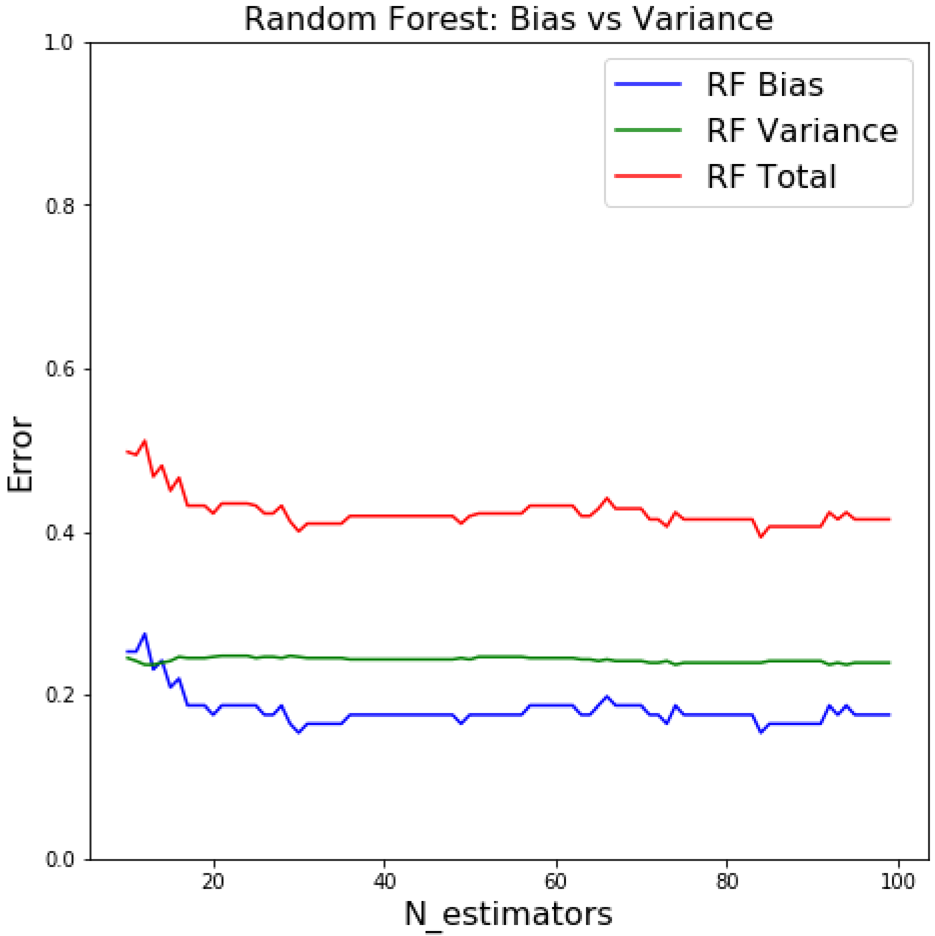


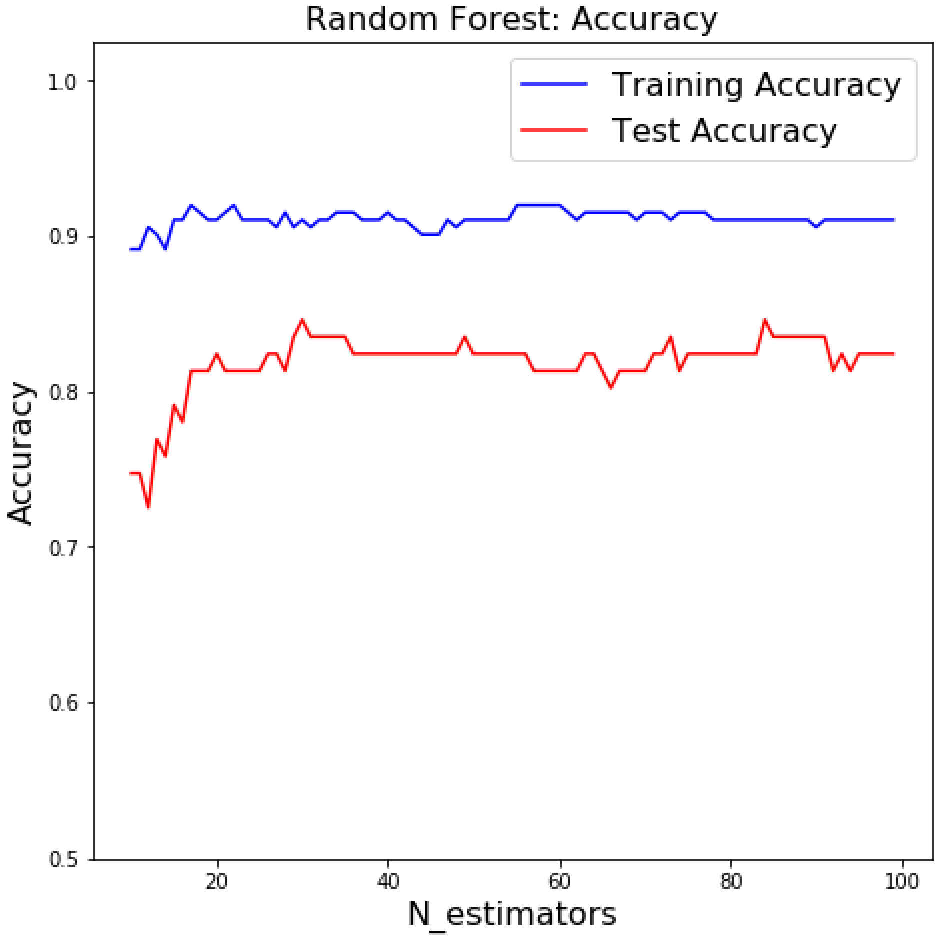
Regression: N_estimators tuning



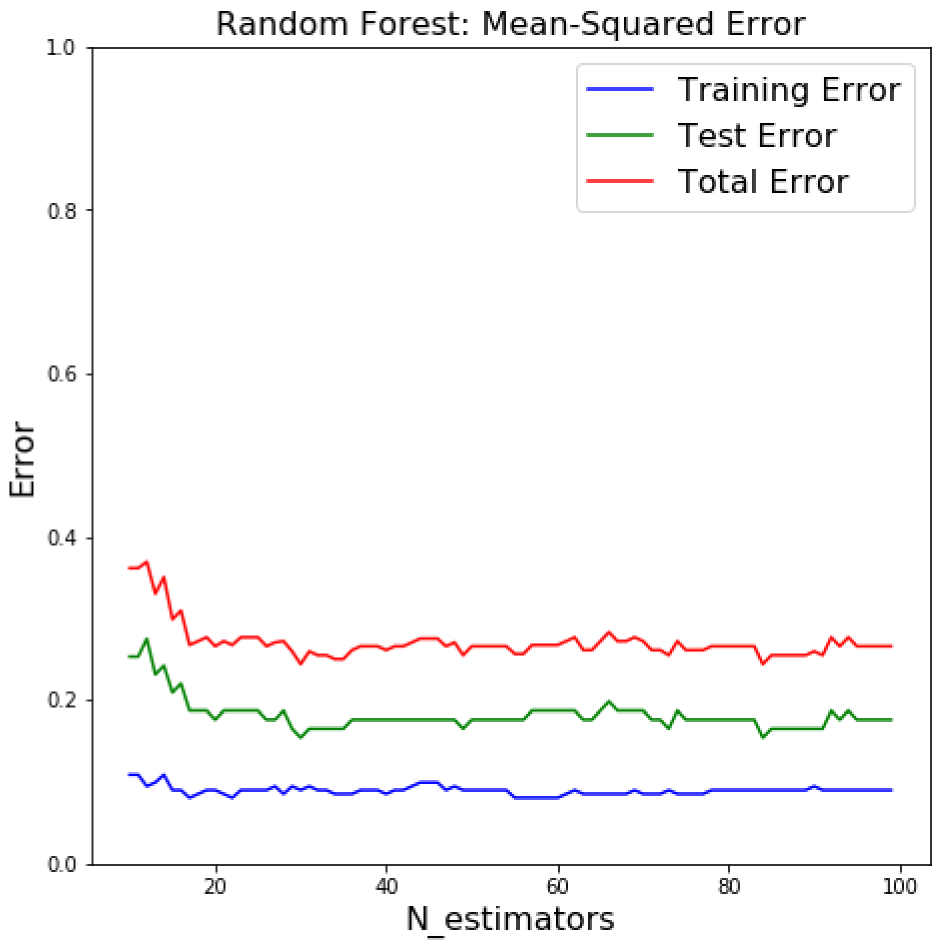


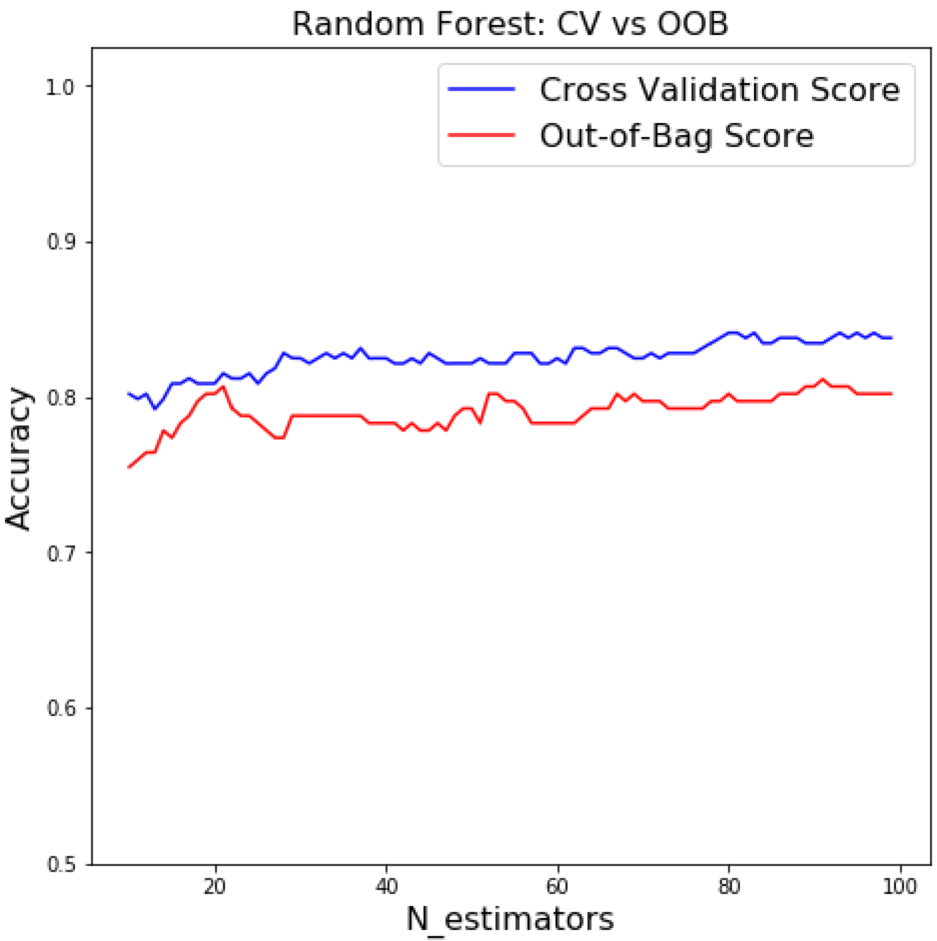
Classification: N_estimator Tuning



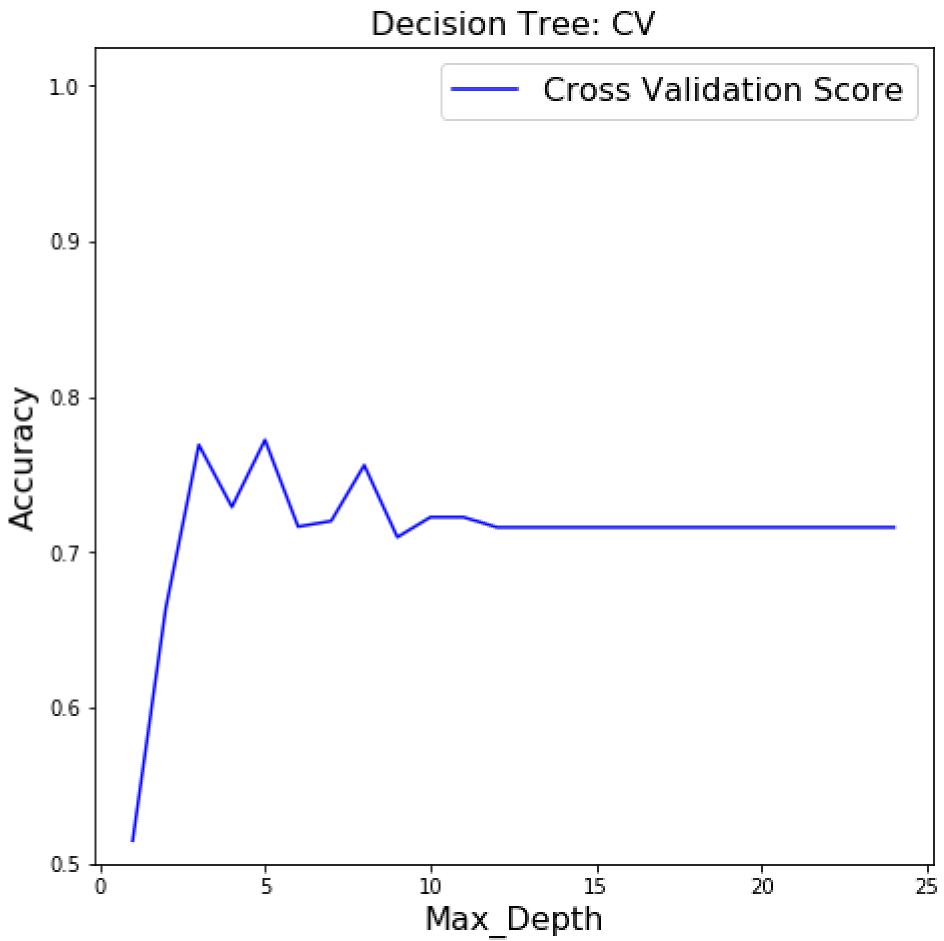


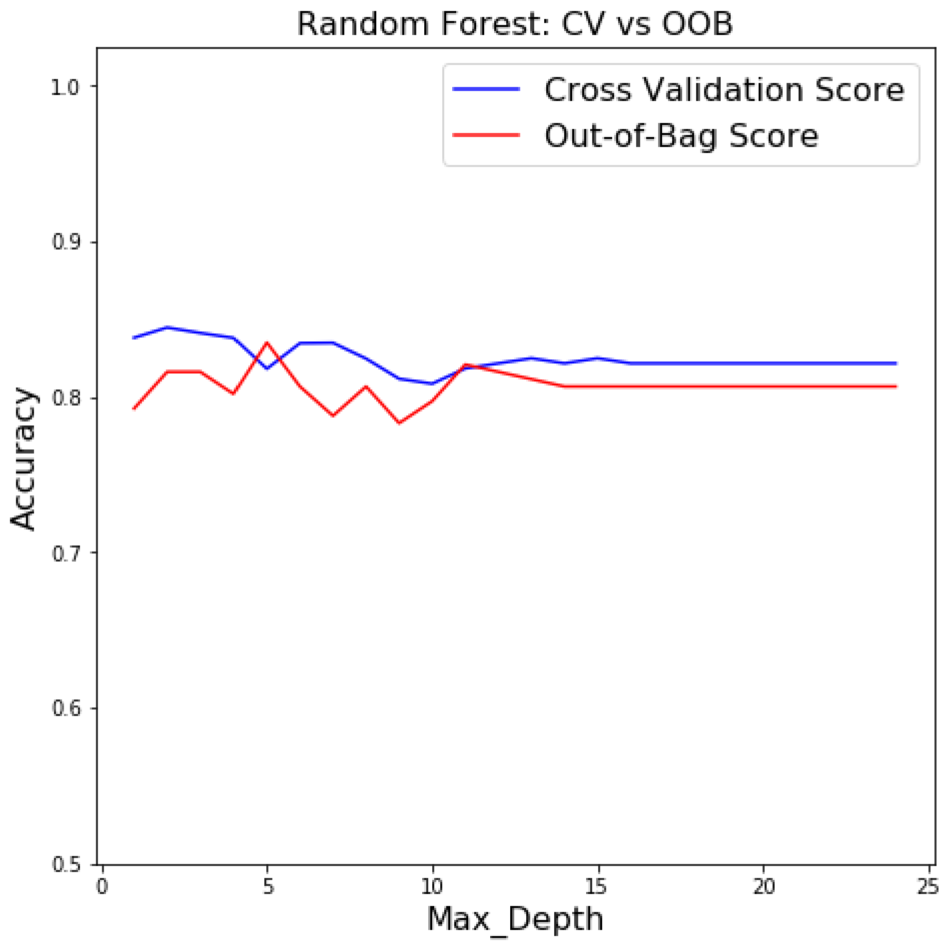
Classification: N_estimator Tuning





Classification: Cross Validation & OOB





Regression: Cross Validation & OOB