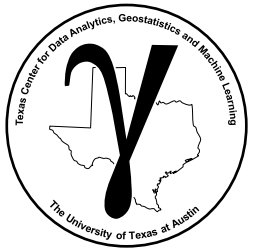
PGE 383 Subsurface Machine Learning

Lecture 14: Decision Tree

Lecture outline:

- **Decision Tree**
- **Predictive Machine Learning Concepts**
- **Decision Tree Training**
- **Decision Tree Tuning**
- **Decision Tree Tuning with K-fold Cross Validation**
- **Decision Tree Shapley Values**
- **Decision Tree Example**
- **Decision Tree Hands-on**



Motivation

Motivation for decision tree

- flexible and easy to understand
- extend to much more powerful ensemble tree methods

Decision Tree

Black spruce forest Boreal Forest, Alaska, https://www.britannica.com/plant/spruce.

Aspects of decision trees

Supervised Learning - decision tree is used for supervised learning, with labels, Y.

$$Y = f(X_1, \ldots, X_m) + \epsilon$$

we are predicting a response, Y, from a set of features, $X_1, \ldots, X_m$

Region-based Prediction, first form regions and then in each region,

Regression or Classification - continuous Y for regression tree or categorical Y for classification tree.

Average of Training Data in Region

$$\hat{y} = \frac{1}{|R_j|} \sum_{X_i \in R_j} y_i$$

Most Common / Plurality of Training Data in Region

$$\hat{y} = \arg \underset{c \in C}{max} \frac{1}{|R_j|} \sum_{X_i \in R_j} I(y_i = c)$$

Note, I is the indicator transform, 1 if $y_i = c$, 0 otherwise.

Decision Tree

Why cover decision trees?

- They are not the most powerful, cutting-edge method in machine learning
- But they are one of the most understandable, interpretable
- Decision trees are expanded with random forests, bagging and boosting to be cutting edge.

Let's learn first about a single tree and then we can comprehend the forest

Black spruce forest Boreal Forest, Alaska, https://www.britannica.com/plant/spruce.

Decision Tree

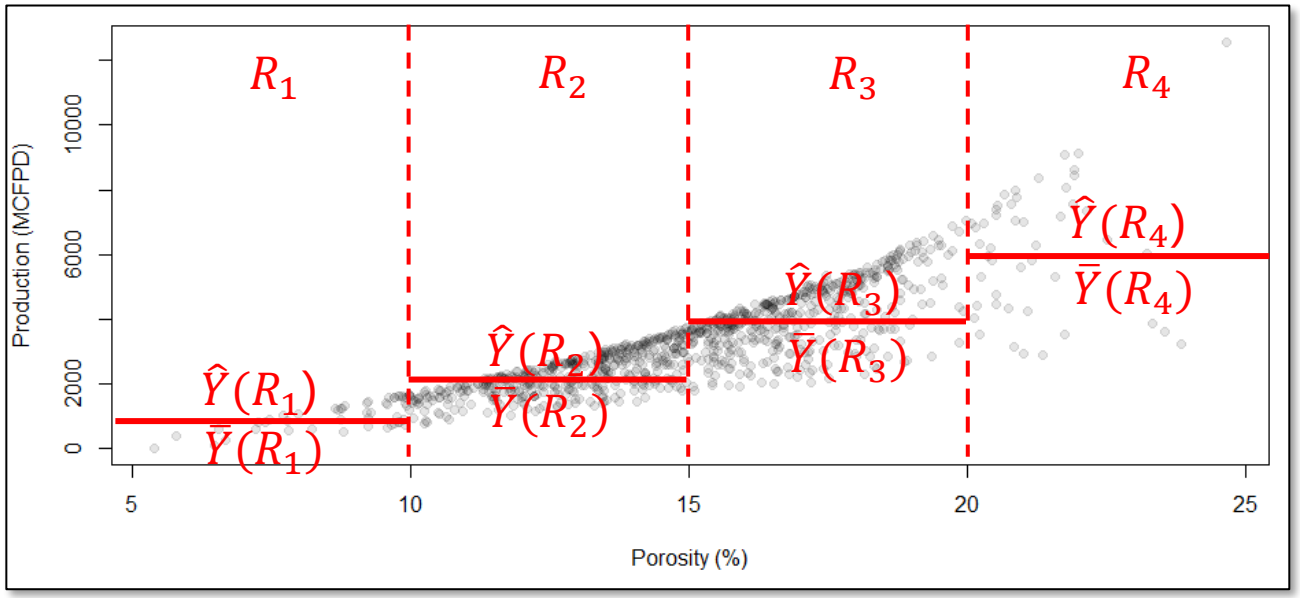
Predictor feature space segmentation-based prediction

The fundamental idea is to divide the predictor space, $X_1, \ldots, X_m$, into J mutually exclusive, exhaustive regions

- **mutually exclusive** – any combination of predictors only belongs to a single region, R_j
- **exhaustive** – all combinations of predictors belong a region, R_j, regions cover entire feature space (range of the variables being considered)

The same prediction in each region, mean of training data in region, $\hat{Y}(R_j) = \bar{Y}(R_j)$ or $\hat{Y}(R_j) = \arg\max(Y(R_j))$

For example, predict production, $\hat{Y}$, from continuous porosity, X_1,



4 region decision tree with data and predictions by region, $R_j, j = 1, \ldots, J$.

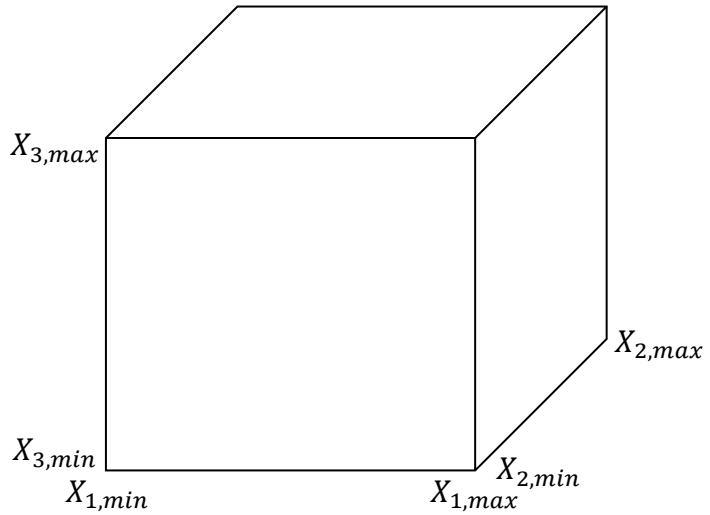
Decision Tree

Predictor Feature Space

The space that includes all possible estimation problems, i.e., the combination of all possible predictor feature values, $x_1, x_2, \ldots, x_m$.

Typically this is defined by the range of possible values, $x_\alpha \in [X_{\alpha,min}, X_{\alpha,max}]$, resulting in,

- line segment – 1 predictor feature
- rectangle – 2 predictor features
- rectangular cuboid - 3 predictor features
- hyperrectangle - >3 predictor features



Schmatic of predictor feature space.

Decision Tree

How do we construct regions, $R_1, R_2, \ldots, R_J$, for our predictions?

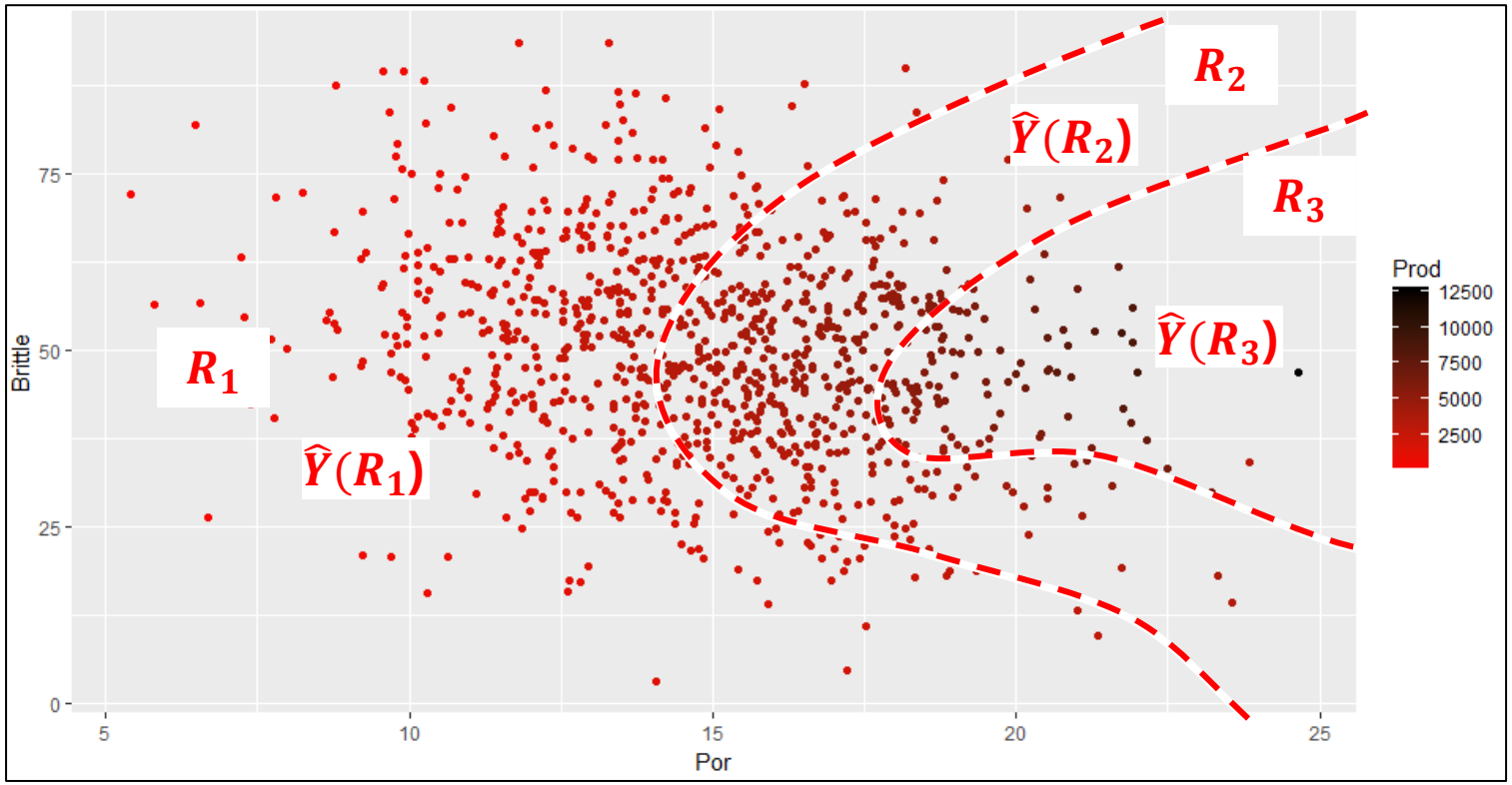
- Our regions could be any shape!
- Consider this 2 predictor feature and 1 response feature problem.

Consider this prediction of unconventional well production (MCFPD) from porosity (%) and brittleness (%)

Efficient Regions!

But,

- complicated
- difficult to calculate
- many parameters



Predict production from porosity and brittleness with 3 regions with data and predictions by region, $R_j, j = 1, \ldots, J$.

Decision Tree

How do we construct regions, $R_1, R_2, \ldots, R_J$, for our predictions?

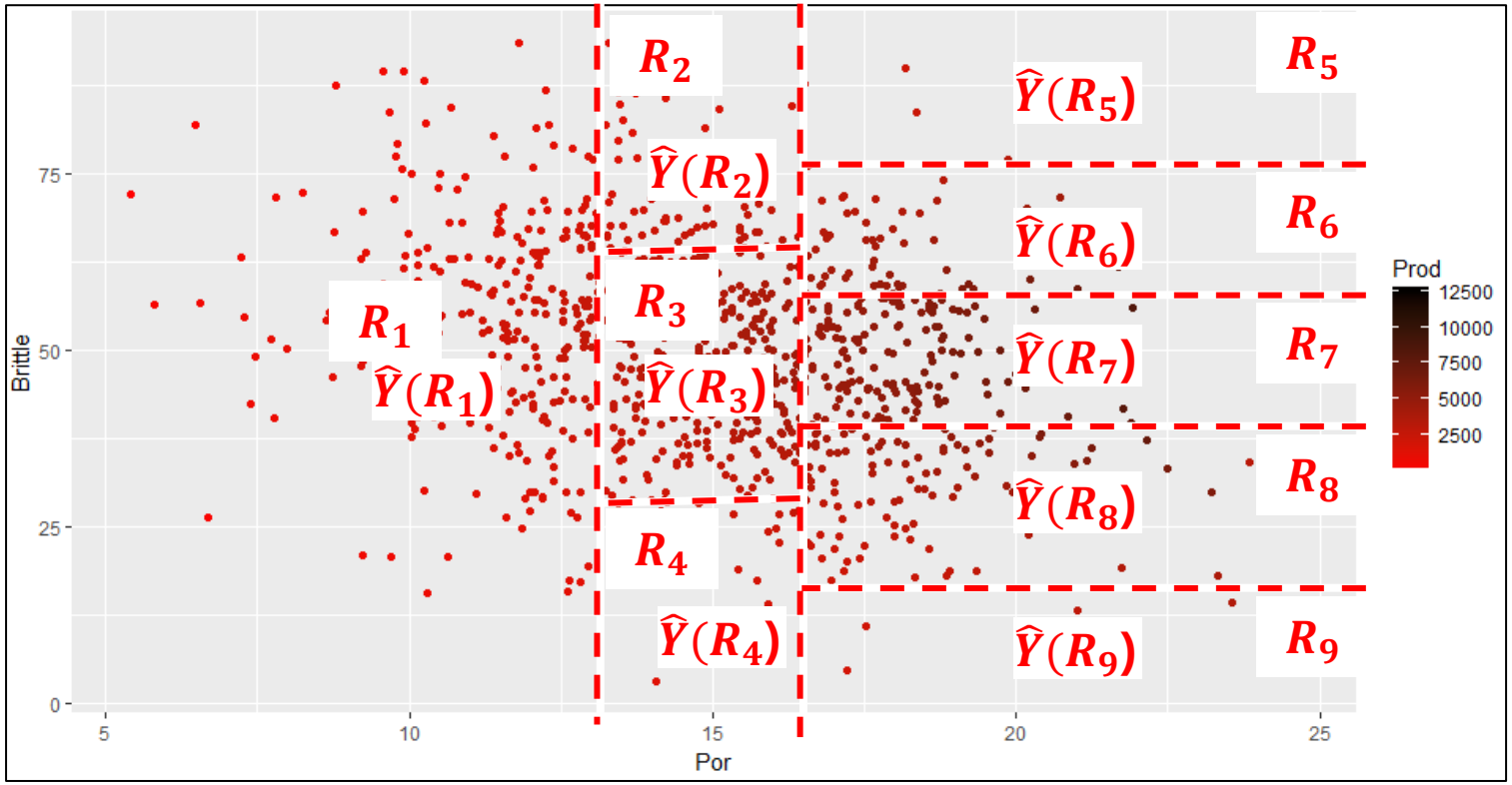
- They could be any shape!
- We decide to use high-dimensional cuboid → simple interpretation / rules

Hierarchical segmentation over the features – somewhat **flexible, compact model!**

Simple Regions!

And,

- easy to calculate
- few parameters
- represented as a tree model



Predict production from porosity and brittleness with 9 cuboid regions with data and predictions by region, $R_j, j = 1, \ldots, J$.

PGE 383 Subsurface Machine Learning

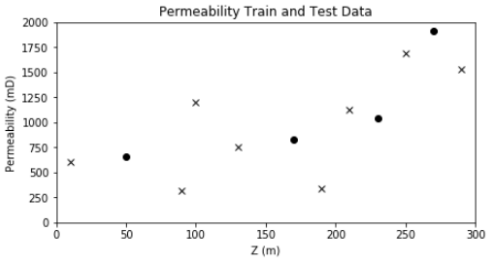
Lecture 14: Decision Tree

Lecture outline:

- **Predictive Machine Learning Concepts**

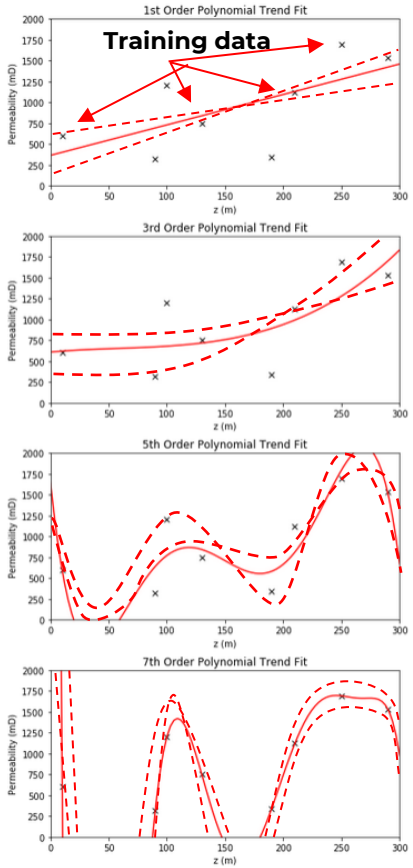
Predictive Model Workflow

1. Split the Data into Train and Test Subsets



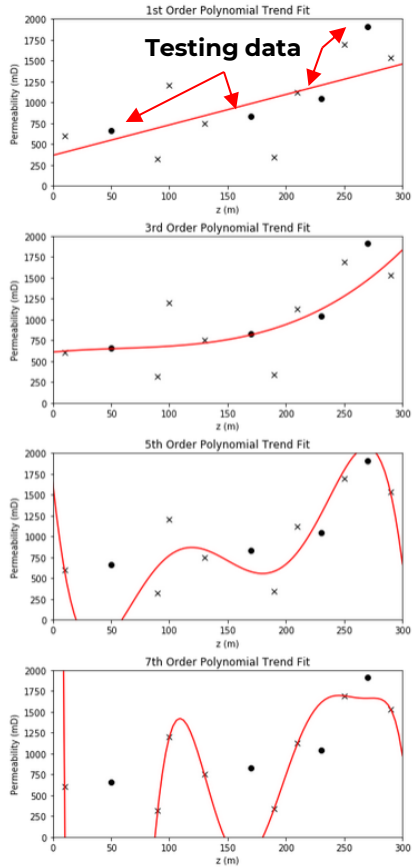
2. Build Models Over a Range of Hyperparameters Increasing Model Complexity

3. Train the Model Parameters with Training Data



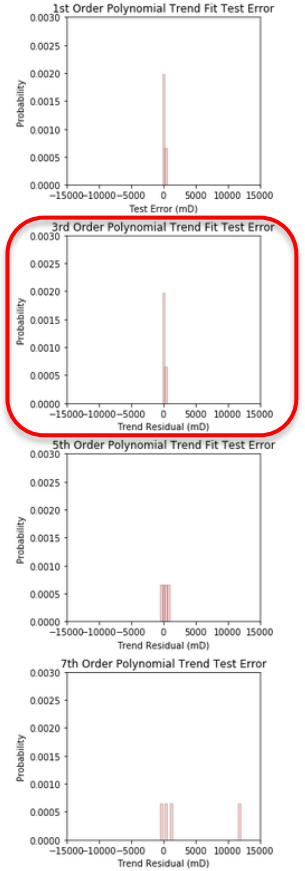
Best Fit Models to Training Data

4. Check Each Best Fit Model with Withheld Testing Data



Error Over the Testing Data

5. Tune the Model Hyperparameters with Testing Data



Hyper-parameters

Retrain the model with tuned hyperparameters and all data, training & testing.

Apply the retrained model.

All of this work to calculate tuned hyperparameters.

Now make our model.

Machine learning model building workflow to avoid overfit.

Model Parameters

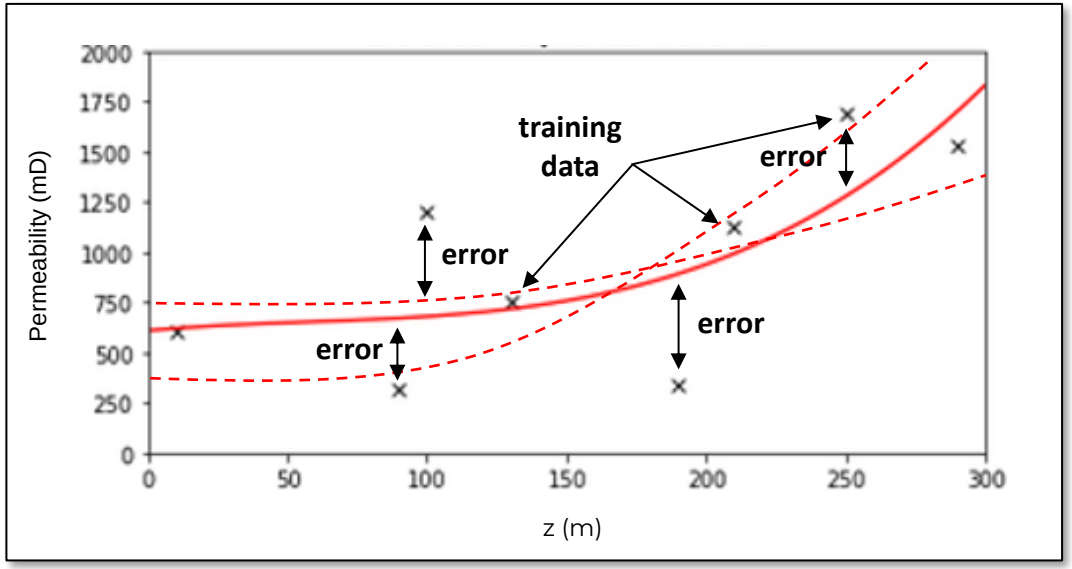
Model Parameters

- Fit during training phase to minimize error at the training data
- For this 3rd order polynomial:

$$y = \boldsymbol{b_3}x^3 + \boldsymbol{b_2}x^2 + \boldsymbol{b_1}x + \boldsymbol{b_0}$$

Parameters:

$\boldsymbol{b_3}$, $\boldsymbol{b_2}$, $\boldsymbol{b_1}$ and $\boldsymbol{b_0}$



Setting model parameters to minimize the error relative to training data.

Model HyperParameters

Model Hyperparameters

- Constrain the model complexity.
- Select hyperparameters that maximize accuracy with the testing data.

- For a polynomial model:

Increasing Complexity

$\vdots$

7th Order: $y = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$

$\vdots$

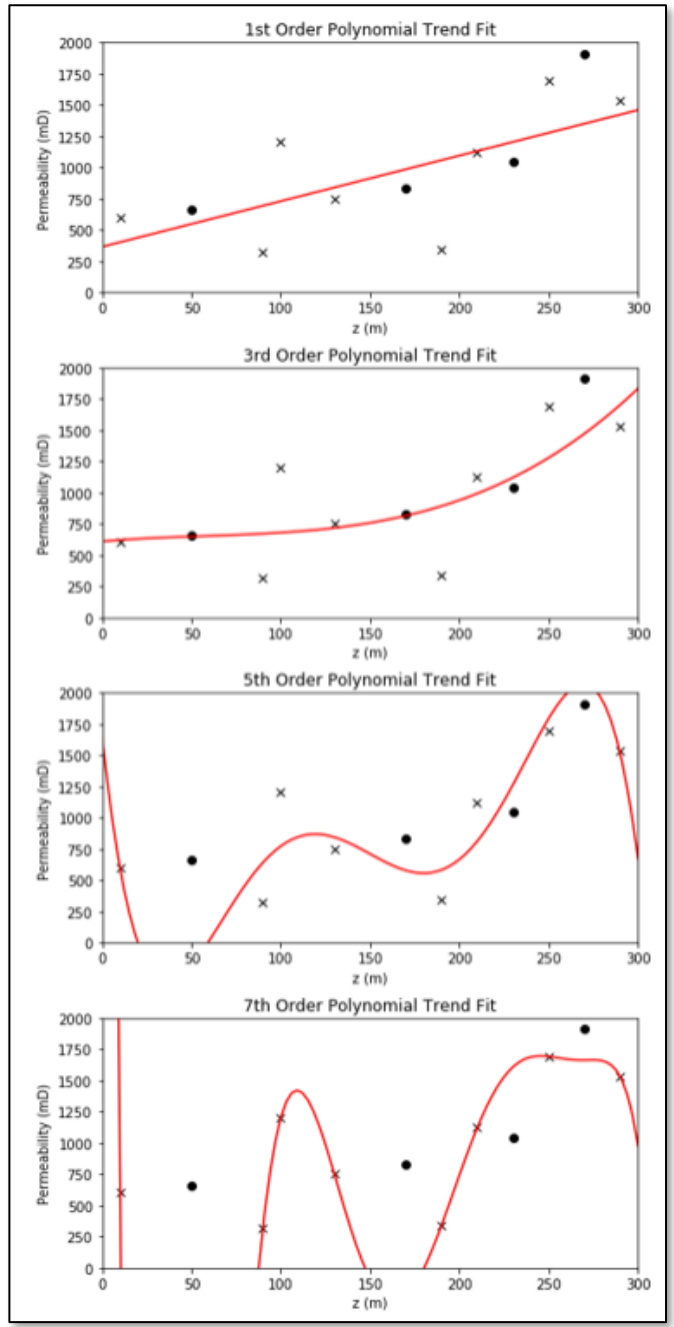
5th Order: $y = b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$

$\vdots$

3rd Order: $y = b_3x^3 + b_2x^2 + b_1x + b_0$

2nd Order: $y = b_2x^2 + b_1x + b_0$

1st Order: $y = b_1x + b_0$



Varying the model complexity, model hyperparameter, to maximize fit with testing data.

PGE 383 Subsurface Machine Learning

Lecture 14: Decision Tree

Lecture outline:

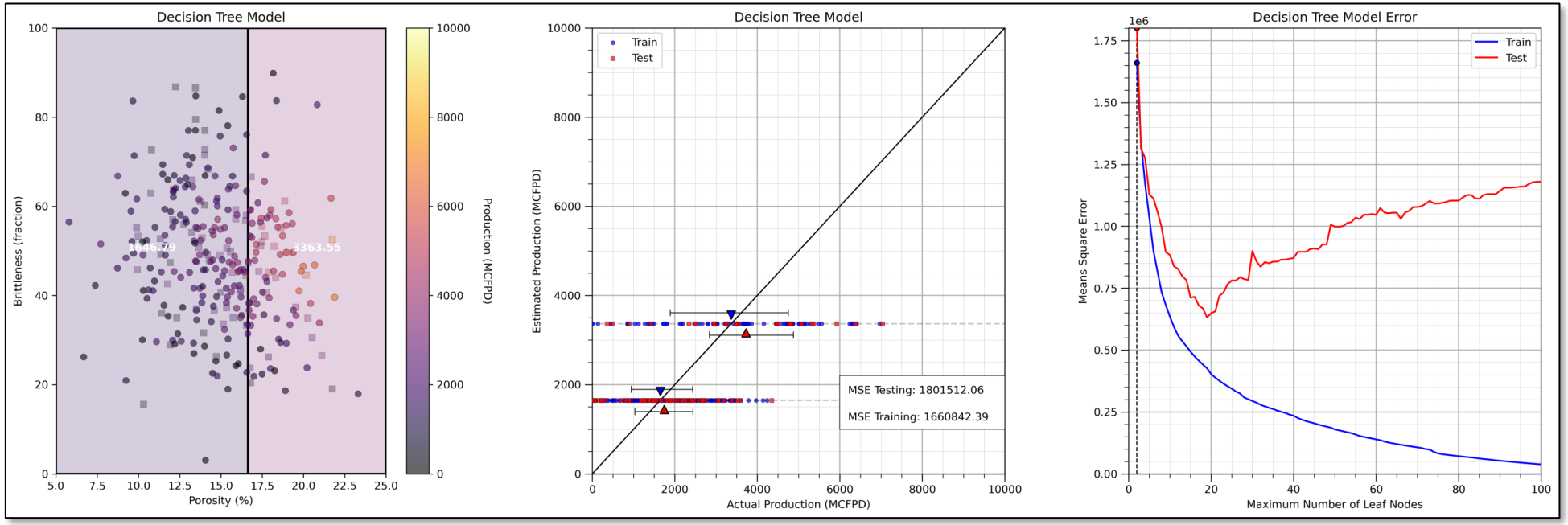
- **Decision Tree Training**

Decision Tree Training

How do we construct regions, $R_1, R_2, \ldots, R_J$, for our predictions?

Recursive, binary splitting

- **Greedy** - at each step, the method selects the choice that minimizes RSS. There is no attempt to look ahead, jointly optimize over multiple choices
- **Top-down** - at the beginning all data belong to a single region, top of the tree, greedy selection of the single best split over any feature that best reduces the RSS



First best split, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Decision Tree Loss

The decision tree loss function

Regression Residual Sum of Squares:

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

where $\hat{y}_{R_j}$ is the estimate and y_i is the training data value.

$$\hat{y} = \frac{1}{|R_j|} \sum_{X_i \in R_j} y_i$$

$|R_j|$ is the number of training data in j region and y_i is a training data, determined by predictor feature values X_i in R_j.

Classification Weighted Average Gini Impurity:

A measure of the level of mixing over all the regions.

$$Gini_{total} = \sum_{j=1}^{J} \frac{N_j}{N} Gini(j)$$

Weighted average Gini Impurity over all Regions (J)

where N_j number of training data in region j and N is the number of training data.

$$Gini(j) = 1 - \sum_{m=1}^{M} p_{j,m}^2 \qquad \left[0{,}1 - \frac{1}{M}\right]$$

where M number of cateogries, and $p_{j,m}$ is the proportion of training data in region j of category m.

No Mixing

	c=1	c=2	c=3
N_j	10	0	0
p_i	1.0	0	0
p_i^2	1.0	0.	0.

$\sum p_i^2 = 1.0$

$Gini(j) = 1 - 1.0 = 0.0$

Maximum Mixing

	c=1	c=2	c=3
N_j	3	3	3
p_i	0.33	0.33	0.33
p_i^2	0.11	0.11	0.11

$\sum p_i^2 = 0.33$

$Gini(j) = 1 - 0.33 = 0.67$

Moderate Mixing

	c=1	c=2	c=3
N_j	6	4	0
p_i	0.66	0.44	0.0
p_i^2	0.44	0.20	0.09

$\sum p_i^2 = 0.64$

$Gini(j) = 1 - 0.64 = 0.36$

Decision Tree Training and Tuning

Here's a Demonstration of Training and Tuning a Decision Tree

First, we perform the first several hierarchical segmentations

- the predictor feature space image is real and the decision tree is hand drawn and estimated for an illustration

Then we walk through the hierarchical segementations and track the training and testing error

- we will see the tuned model complexity, the number of regions / leaf nodes that minimize testing error

Decision Tree Training

The steps for training a decision tree,

0. **Assign All Data to a Single Region** - that covers the entire predictor feature space

1. **Scan All Possible Splits** - over all regions and over all features

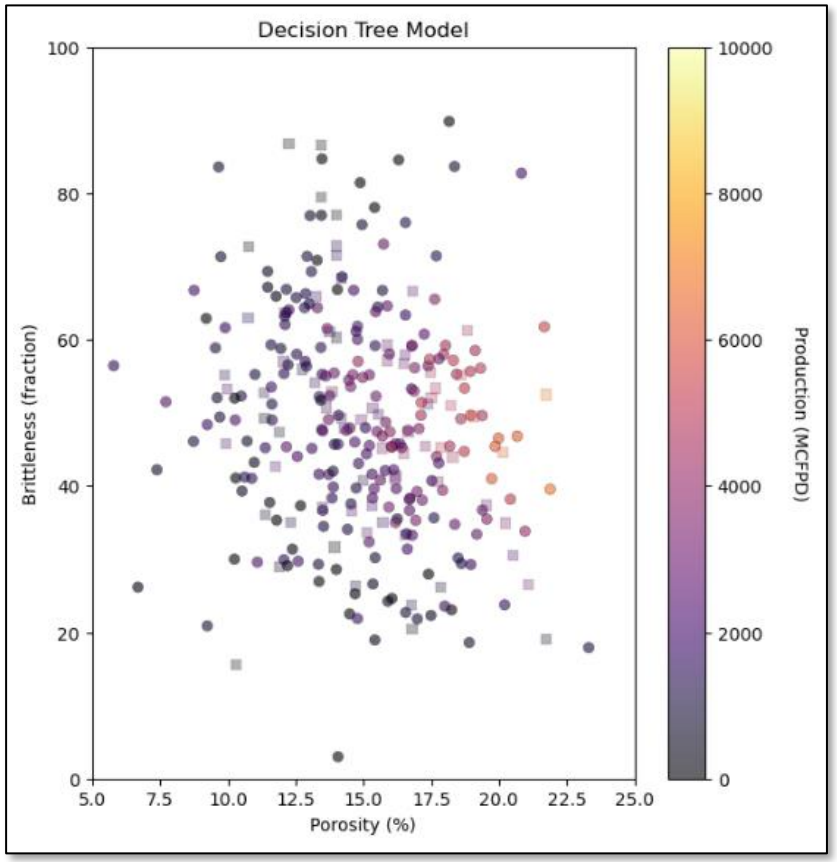
2. **Select the Best Split** - this is greedy optimizatiot

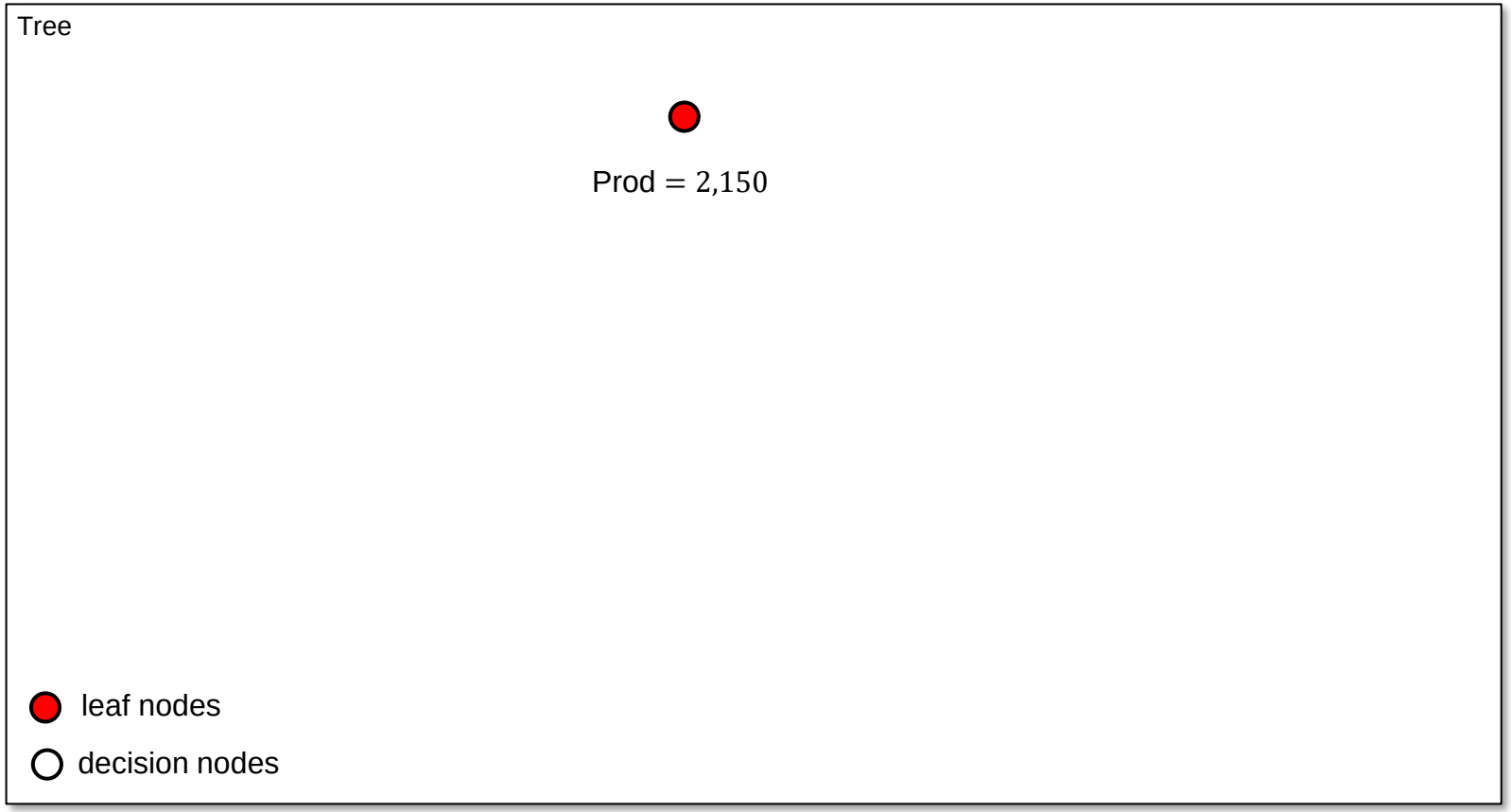
3. **Iterate Until Very Overfit** - return to step one for the next split until the tree is very overfit.

Note, this is a solution heuristic – a practical approximation, since there is no attempt to jointly optimize over all splits

- for example, to select a suboptimal split to maximize training error reduction with a subsequent split

Decision Tree Training

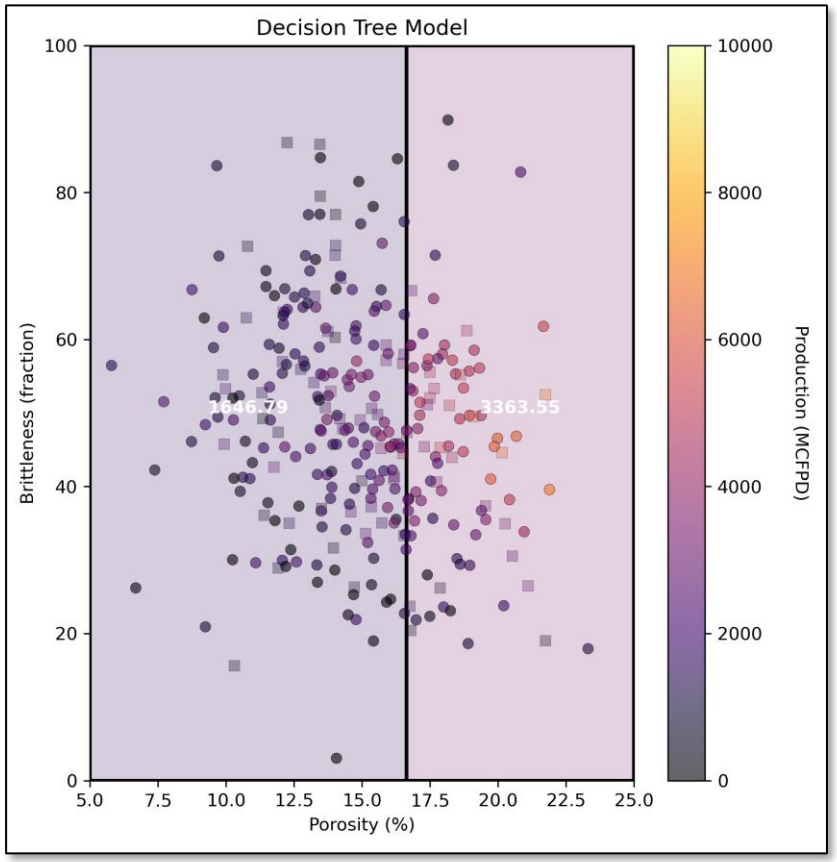


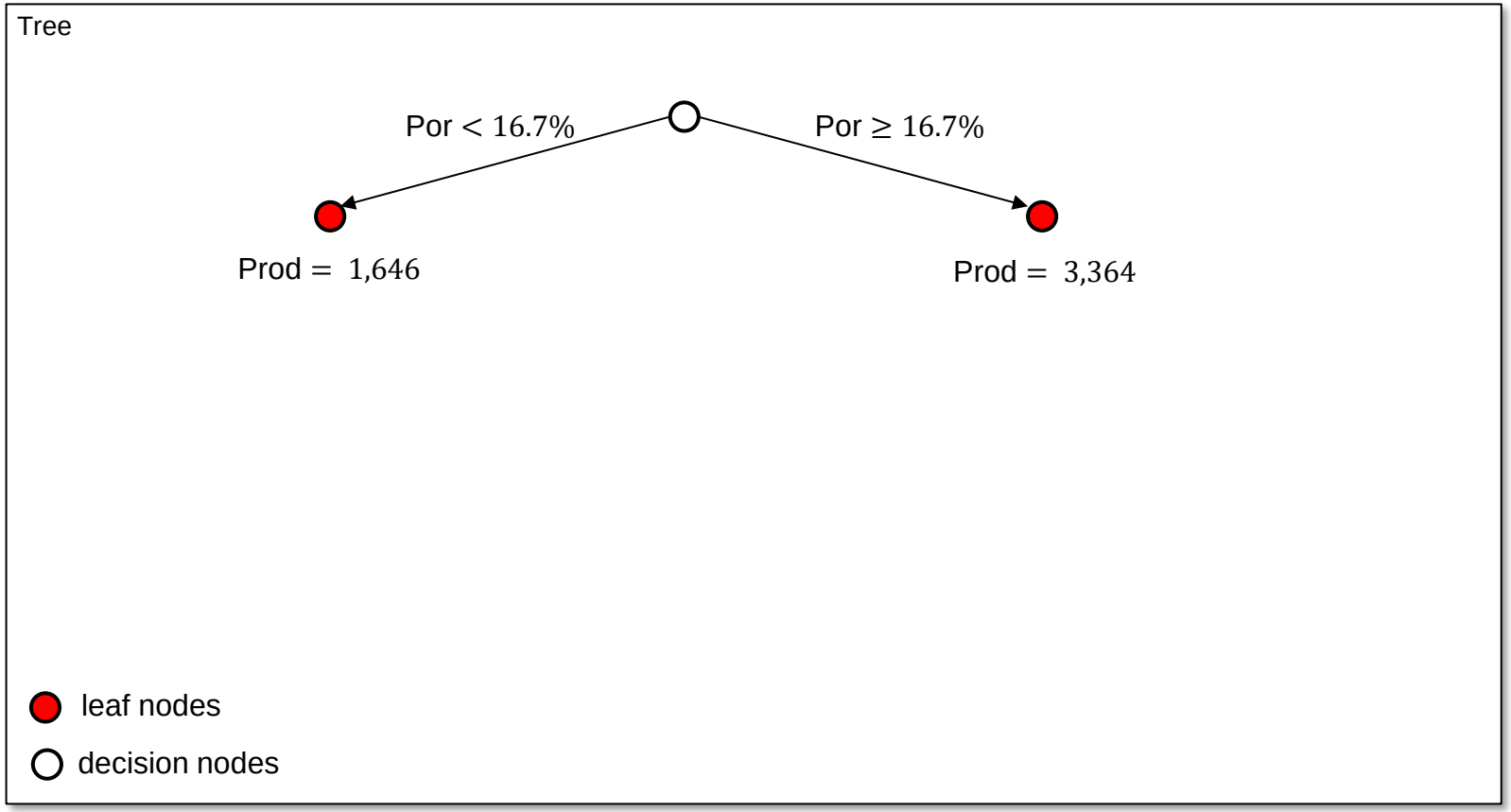


Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 1 region (leaf nodes) – predict with the global mean.

Decision Tree Training

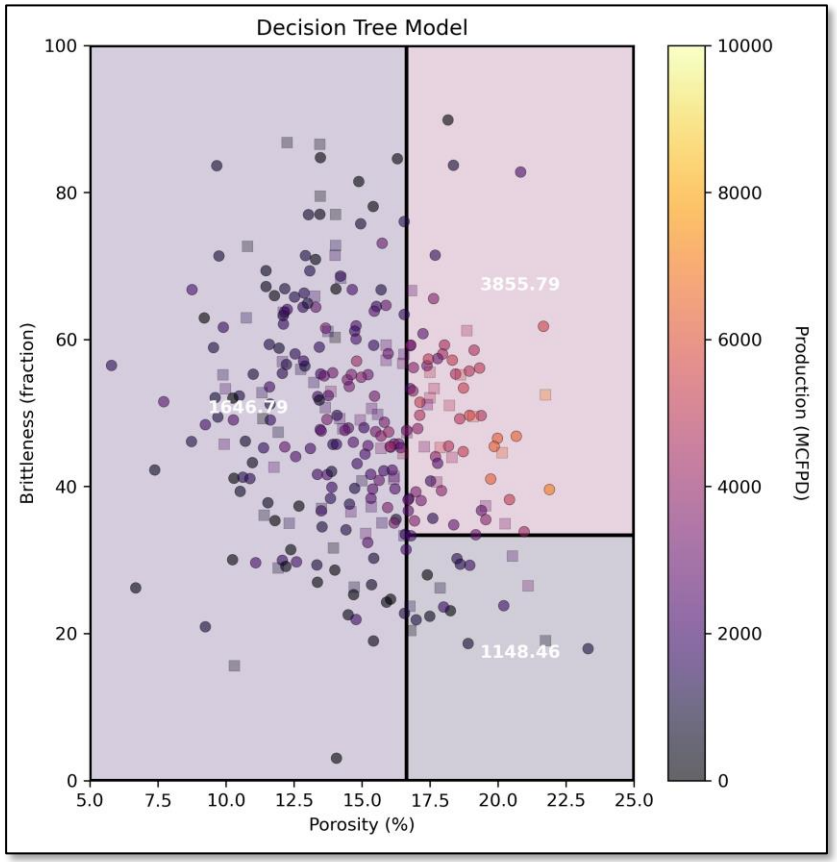


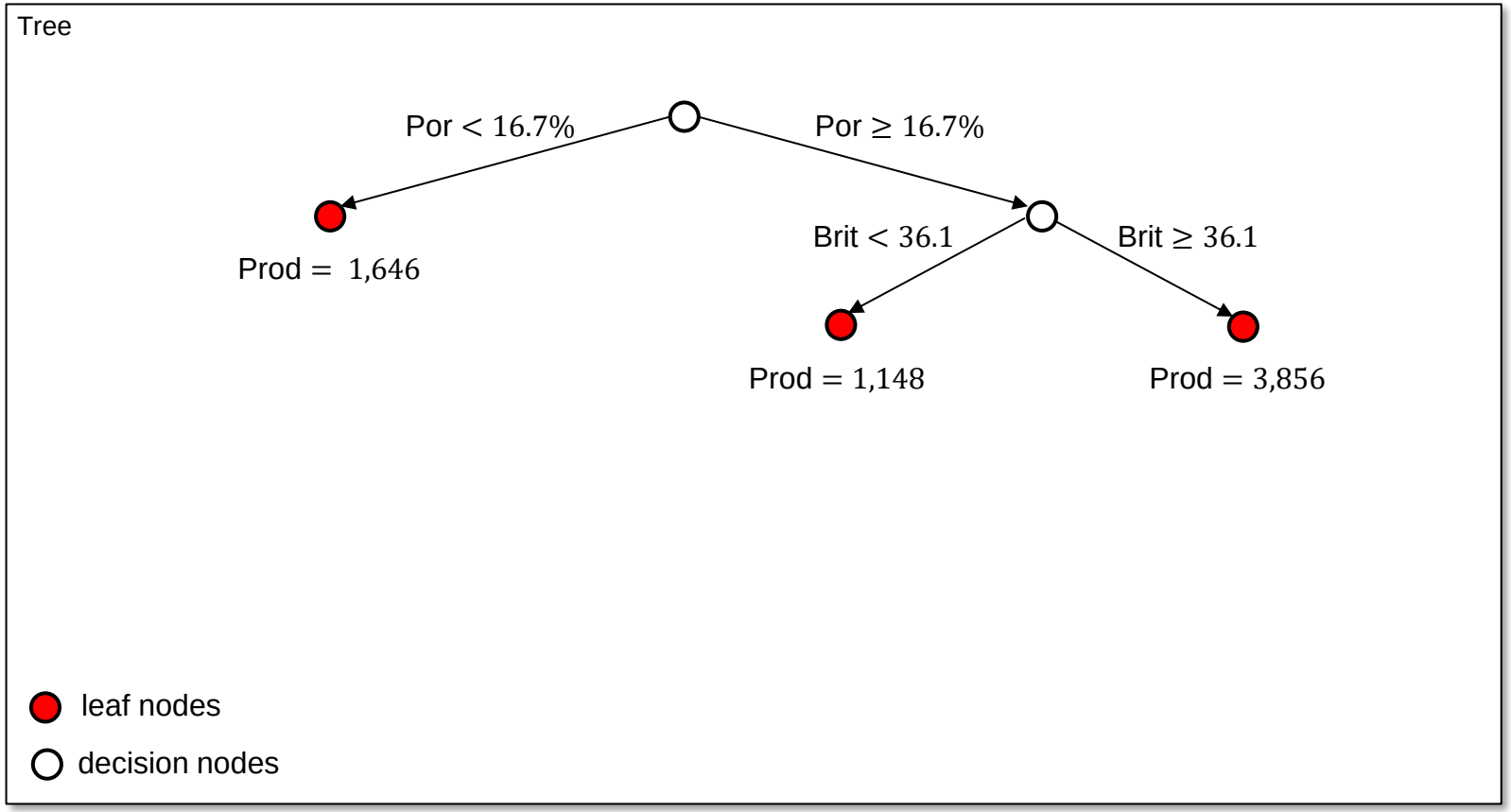


Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 2 regions (leaf nodes) – very, very underfit model

Decision Tree Training

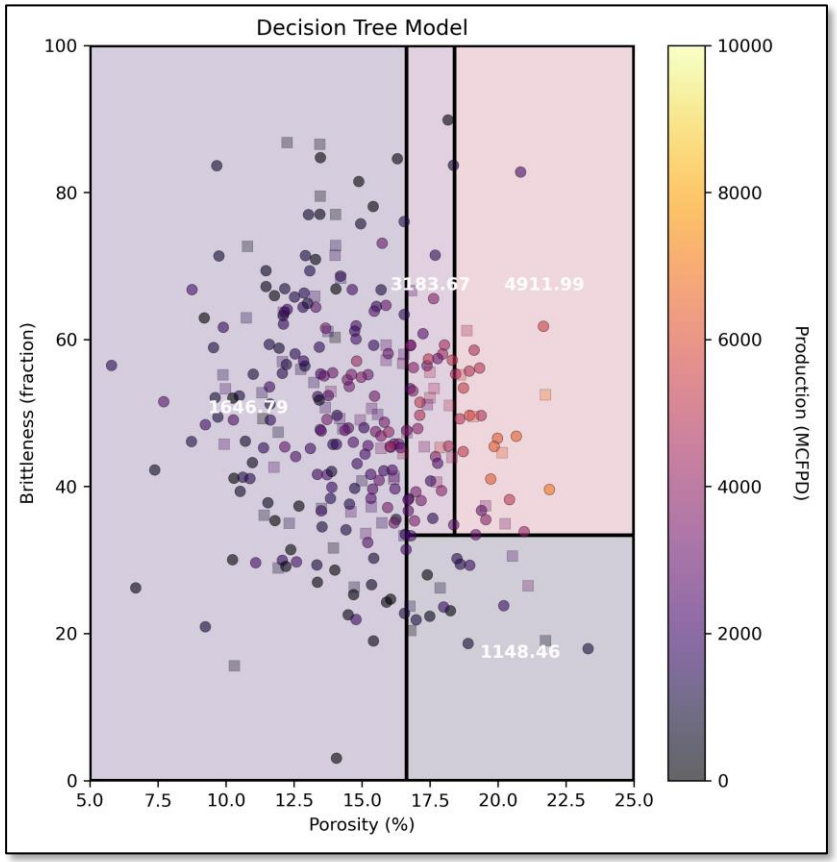


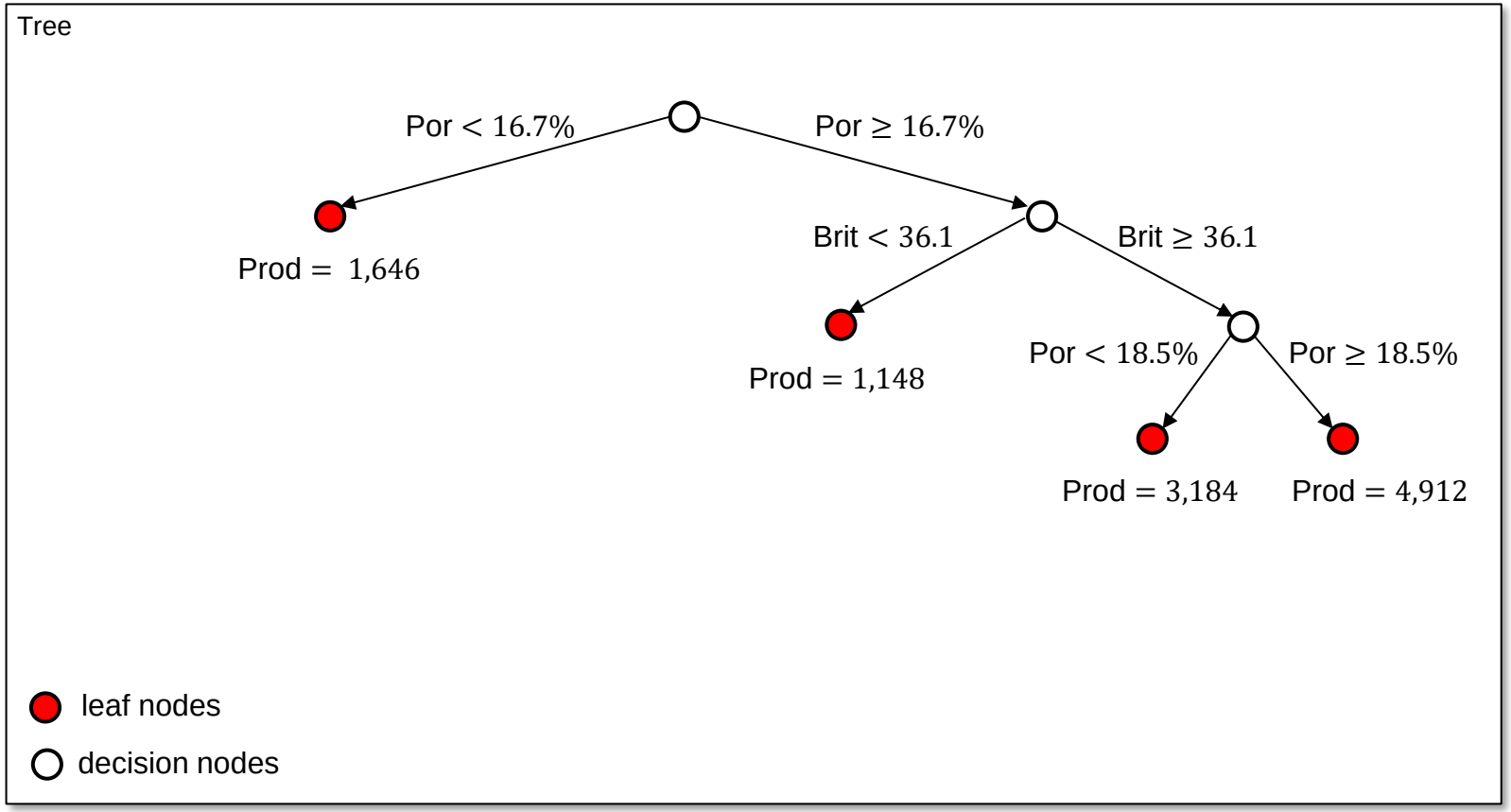


Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 3 regions (leaf nodes) – very, very underfit model

Decision Tree Training

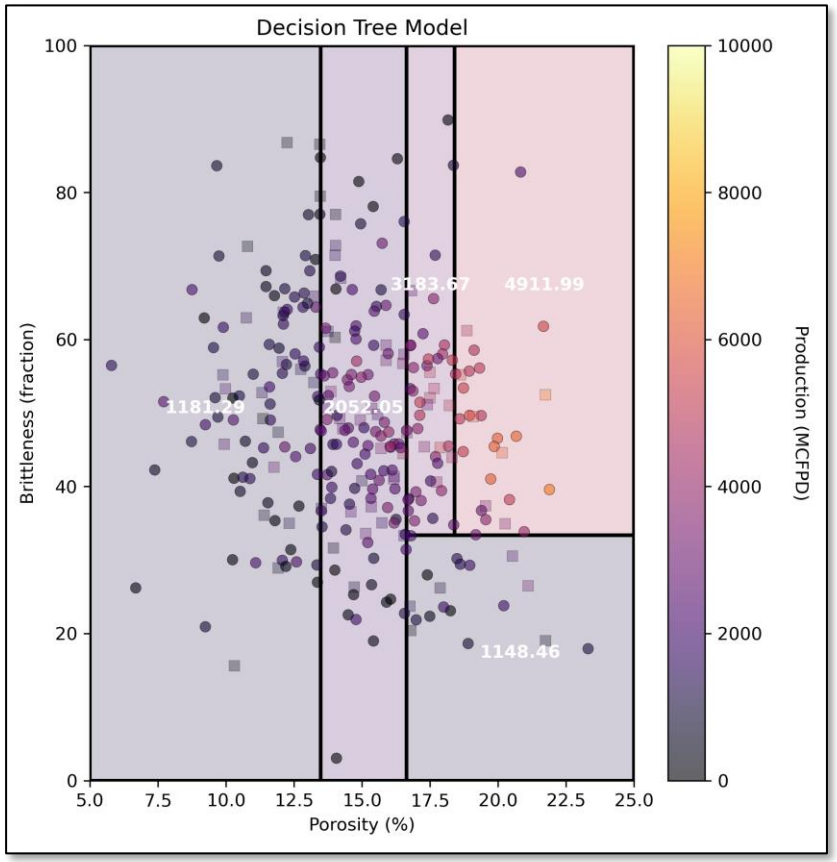


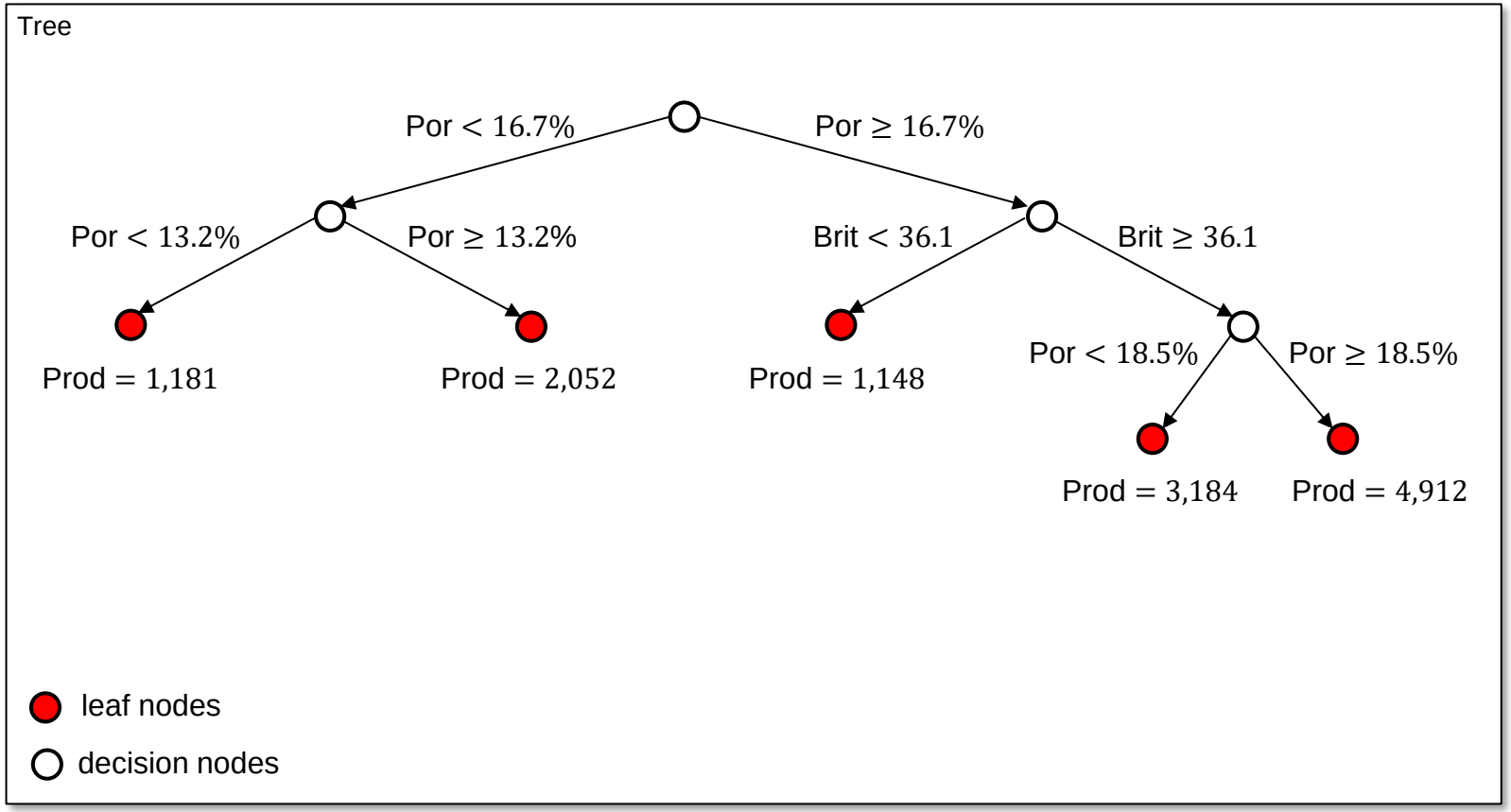


Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 4 regions (leaf nodes) – very underfit model

Decision Tree Training

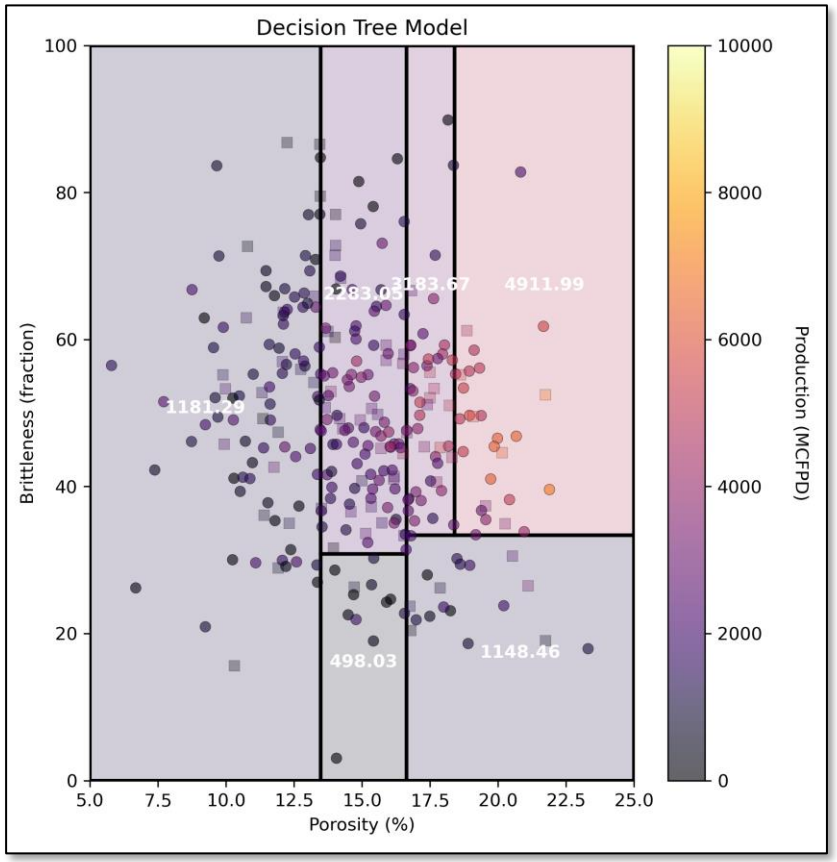


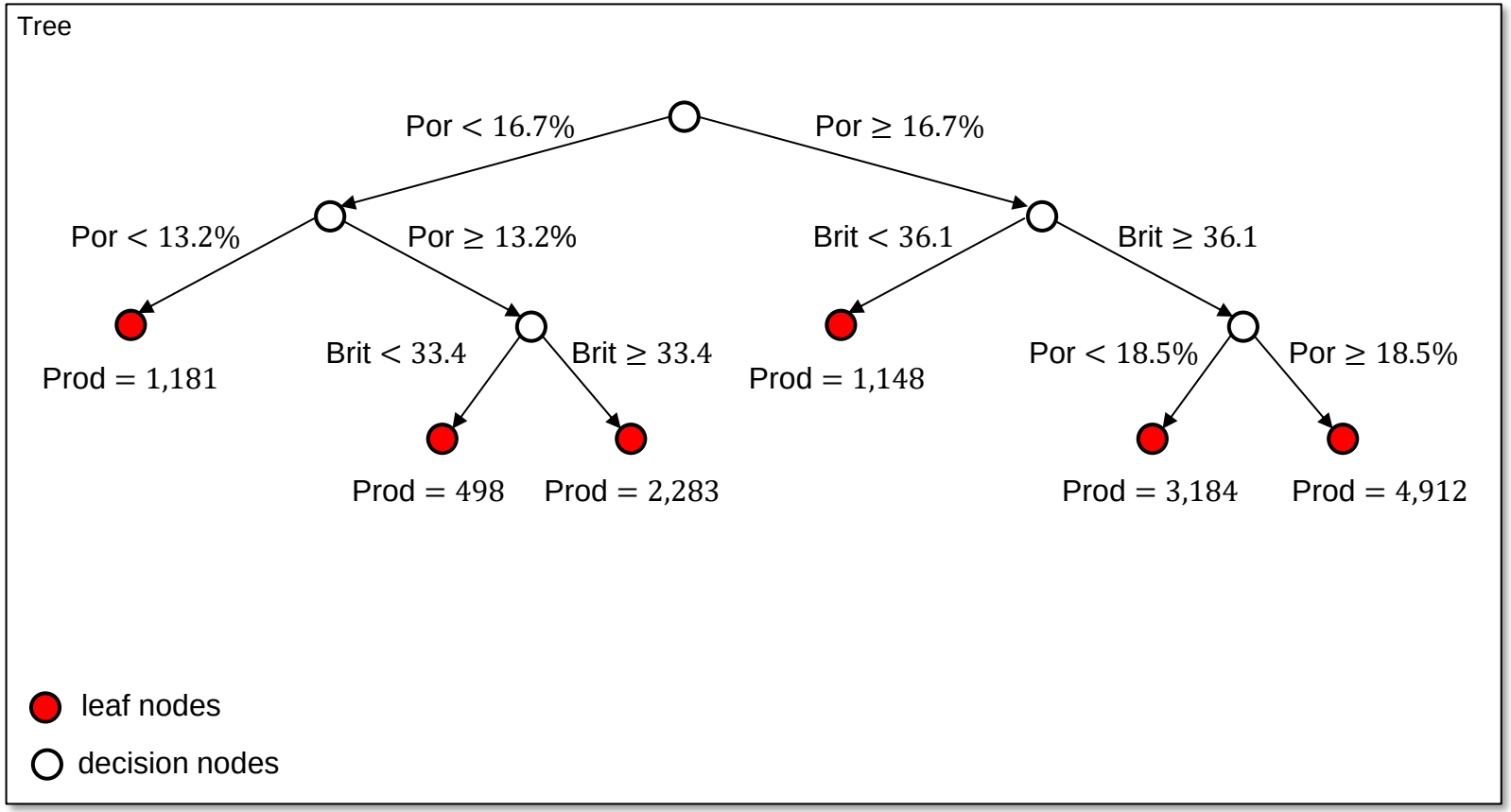


Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 5 regions (leaf nodes) – very underfit model

Decision Tree Training

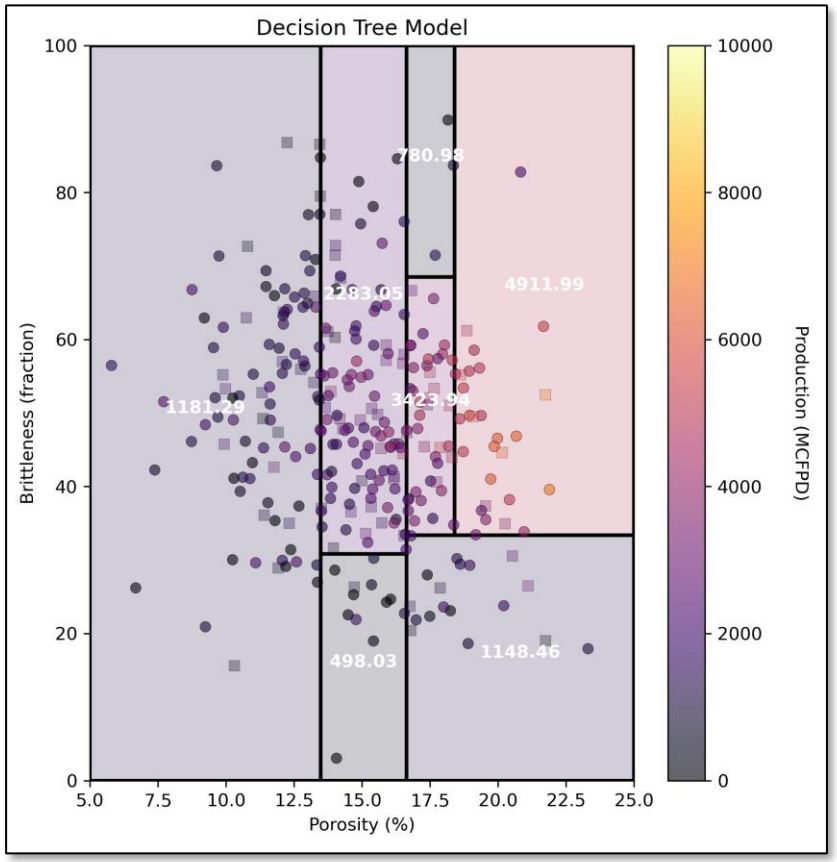


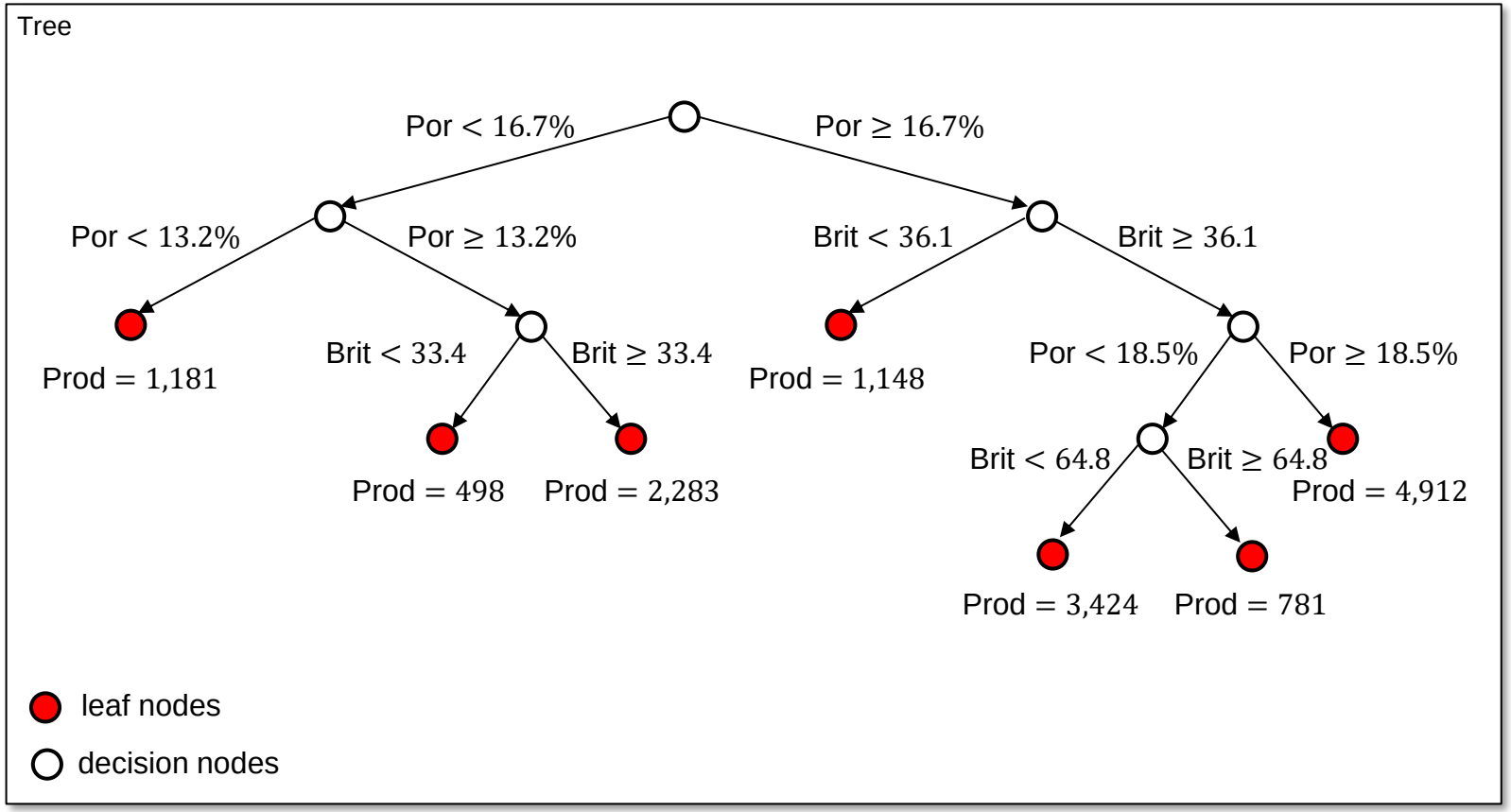


Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 6 regions (leaf nodes) – underfit model

Decision Tree Training

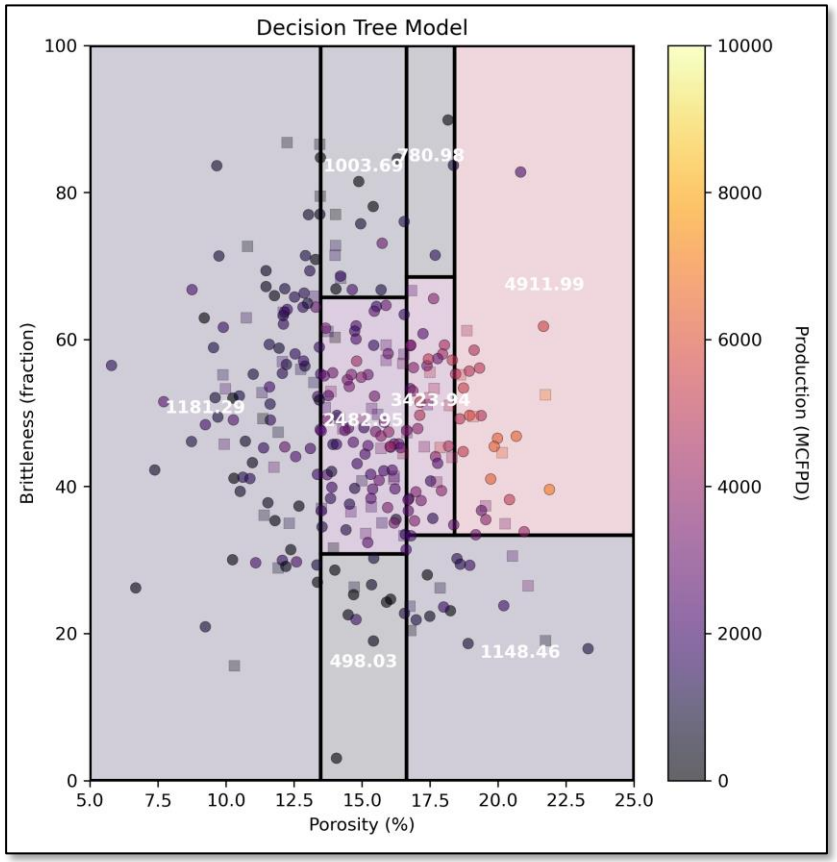


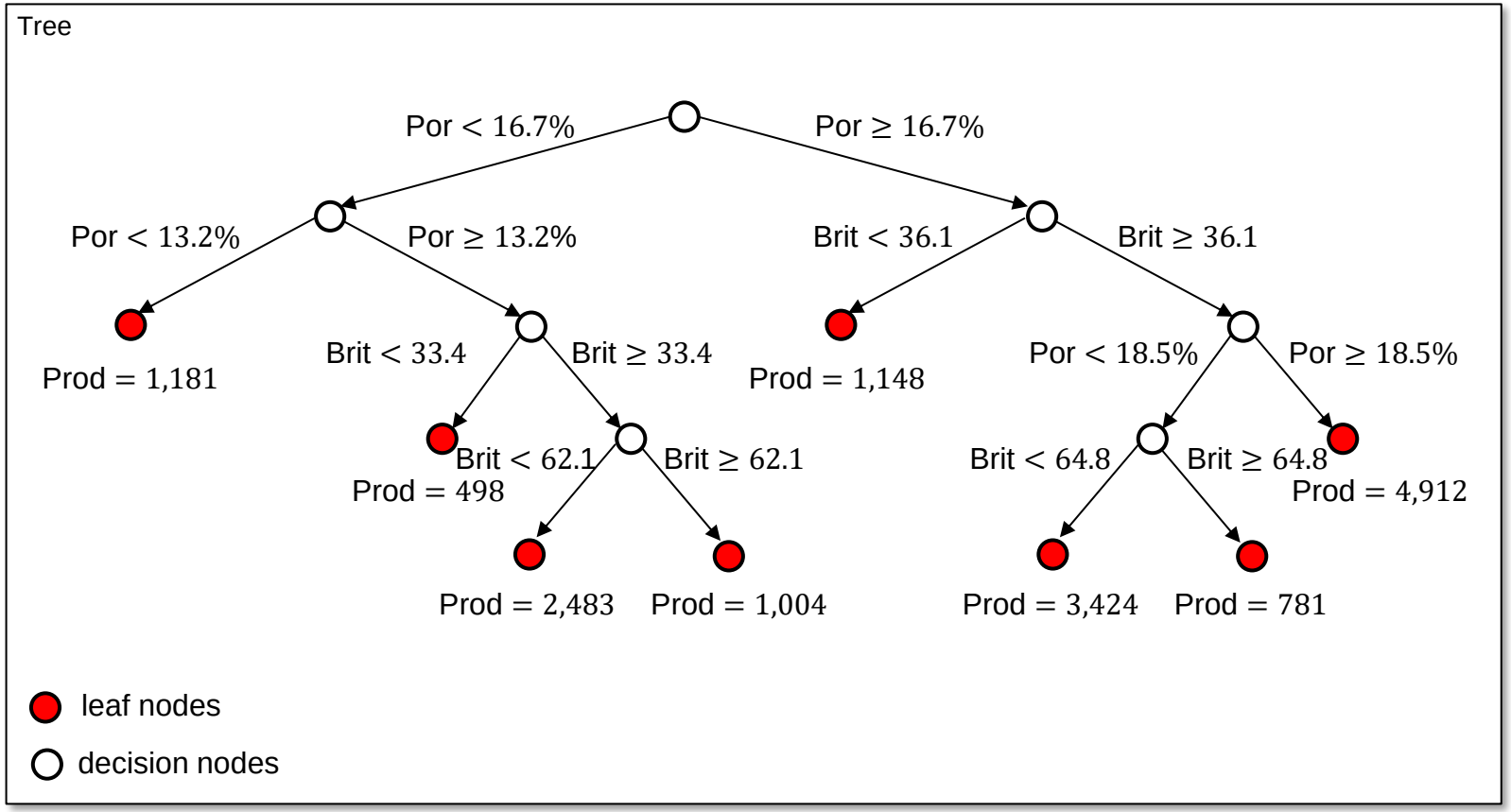


Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 7 regions (leaf nodes) – underfit model

Decision Tree Training

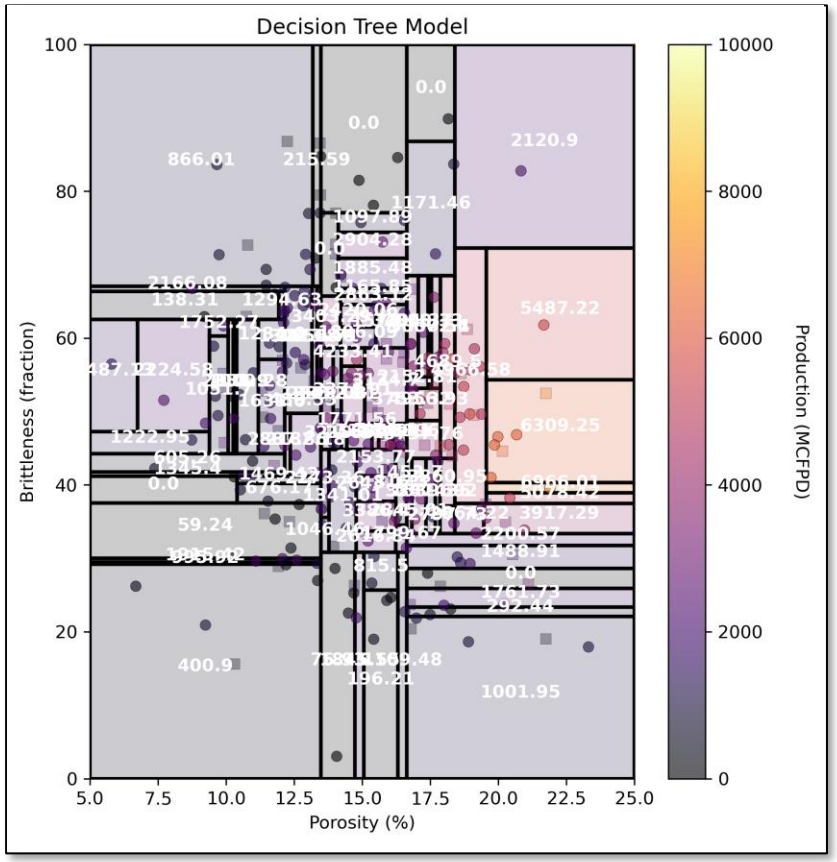


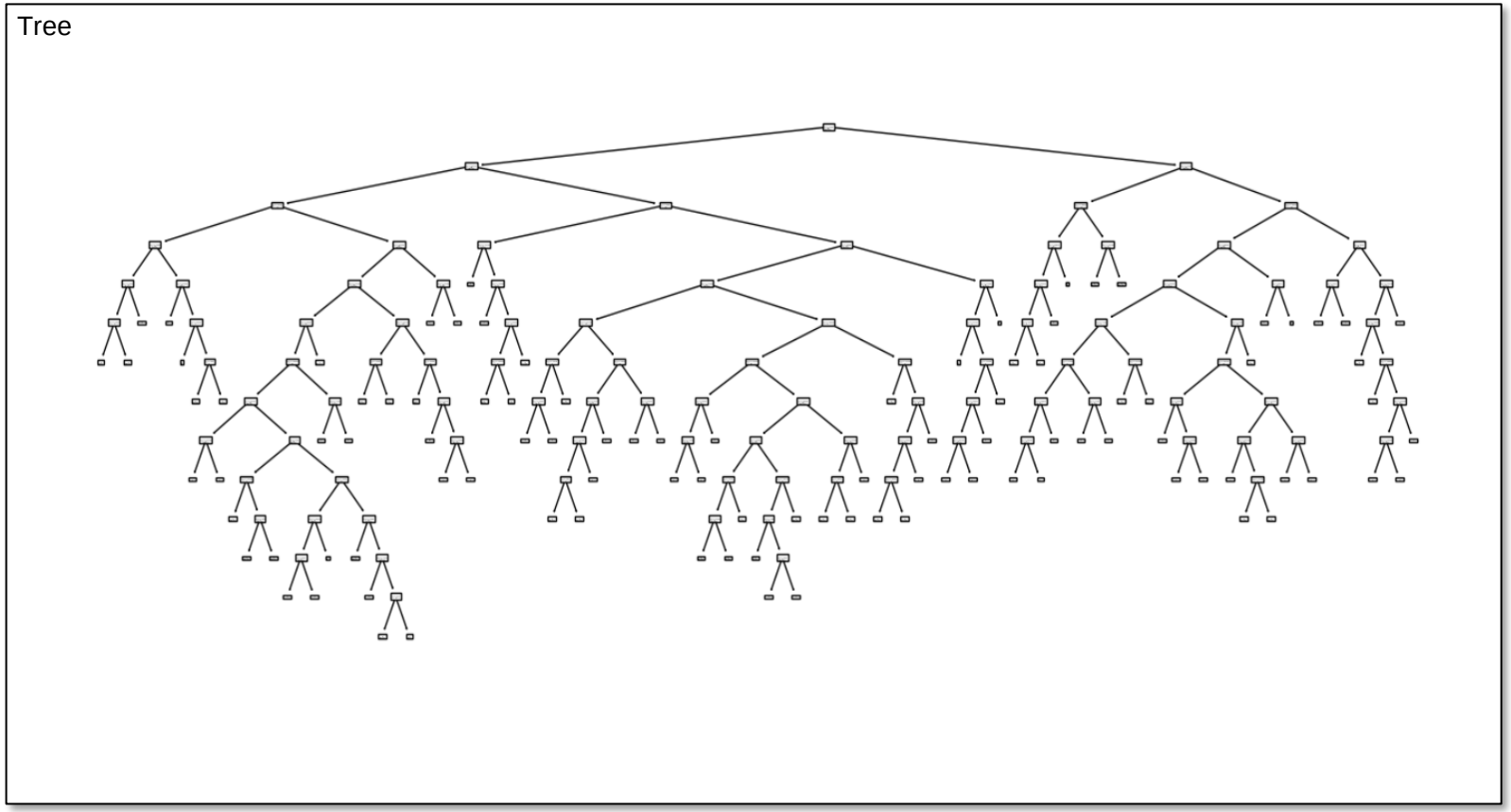


Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 8 regions (leaf nodes) – underfit model

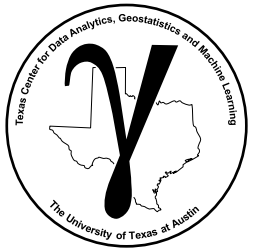
Decision Tree Training





Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 100 regions (leaf nodes) – very, very overfit model, complicated enough to stop.



Determining the Next Best Split

Determining the Next Best Split

- This requires search over all possible features within all regions
- This is not computationally impractical (not too big a space to search), because we actually only attempt the midpoints between the sorted training data.
- For a new condidate split, s, we have 2 potential new regions from the original 1 region

$$R_{1(m,s)} = \{X|X_m < s\} \text{ and } R_{2(m,s)} = \{X|X_m \geq s\}$$

- X_m are the features and s is the threshold for the segmentation into R_1 and R_2
- We segment such that we minimize the Residual Sum of Squares:

$$RSS = \sum_{i:x_i \in R_1(m,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:x_i \in R_2(m,s)} (y_i - \hat{y}_{R_2})^2$$

We add in the RSS components from all the other regions to get the total model RSS.

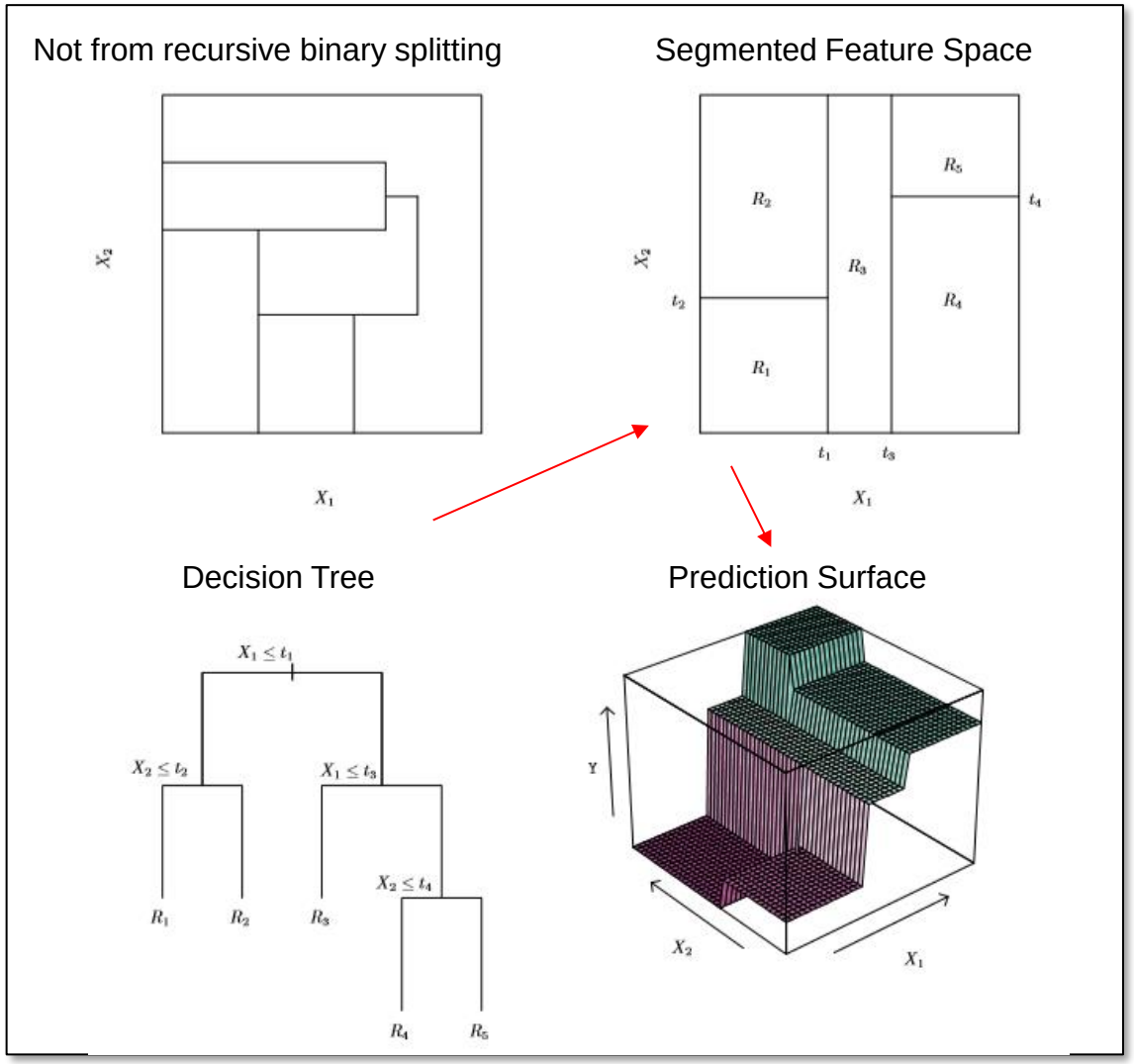
Decision Trees
The Regions

The Shape of our Decision Tree Prediction Models

Example from James et al. (2017)

- Top-left 2D feature space partitioning that could not result from recursive binary splitting

- Top-right feature space partitioning, decision tree and estimation surface for feature space.



Visualization of model from recursive binary splitting and one not (upper left), image from James et al. (2017)

PGE 383 Subsurface Machine Learning

Lecture 14: Decision Tree

Lecture outline:

- **Decision Tree Tuning**

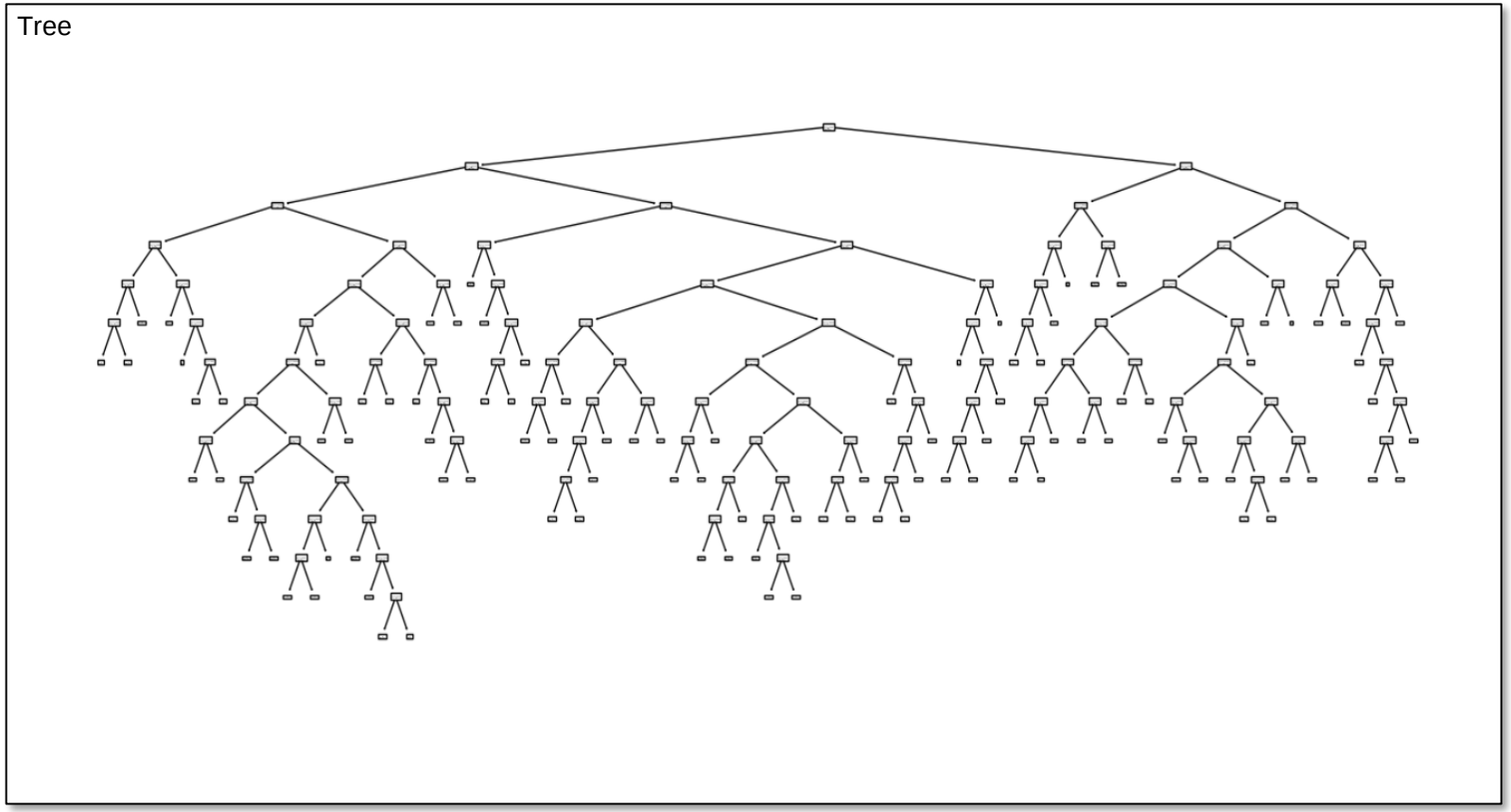
Decision Tree Tuning

To tune the decision tree we take the very overfit model,

- sequentially cut the last decision node
- i.e., prune the last branch of the tree

The simpler trees are inside the complicated tree!

- calculate test error as we prune
- select tree with minimum test error



Our overfit 100 region decision tree model, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

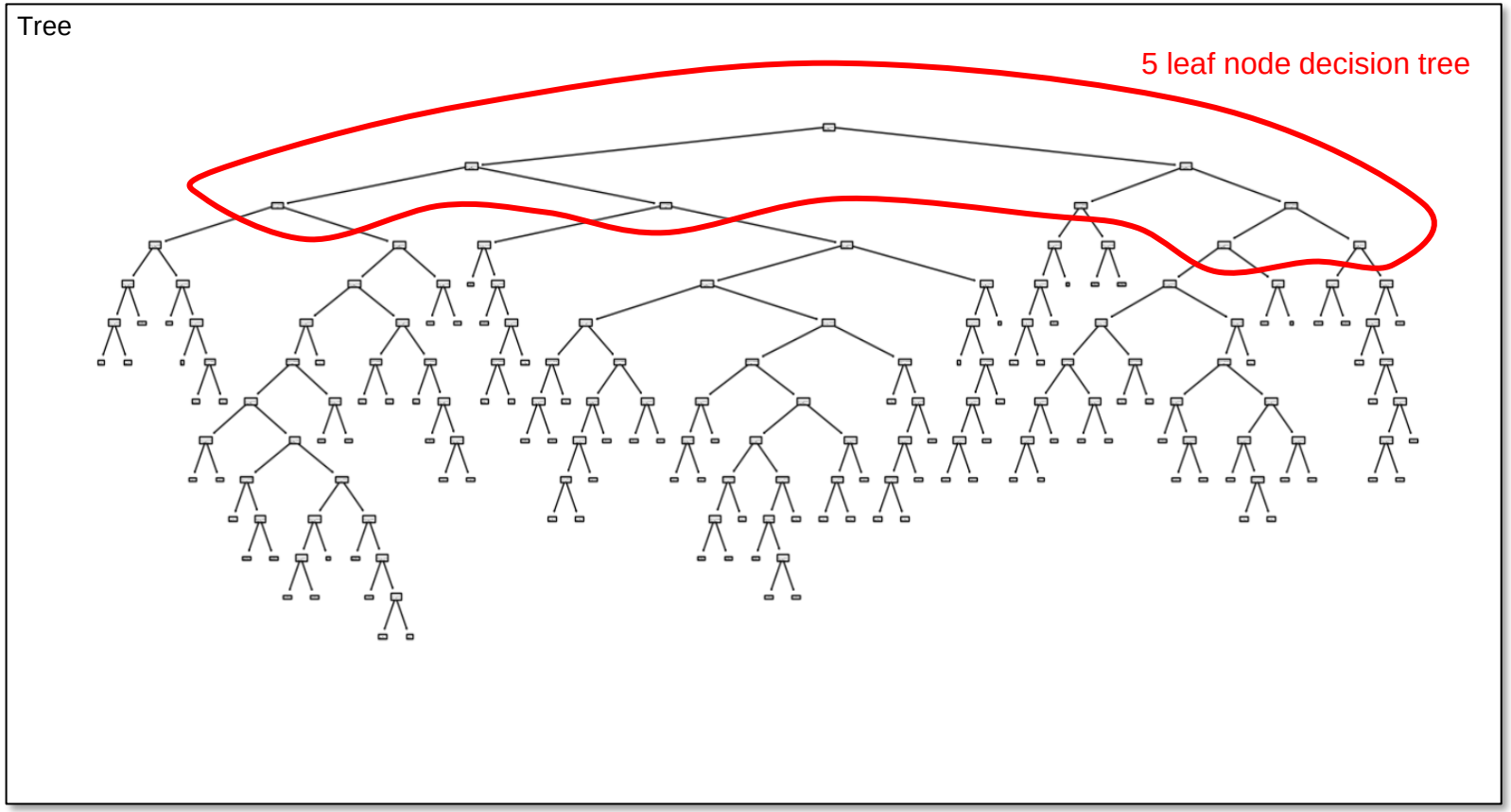
Decision Tree Tuning

To tune the decision tree we take the very overfit model,

- sequentially cut the last decision node
- i.e., prune the last branch of the tree

The simpler trees are inside the complicated tree!

- calculate test error as we prune
- select tree with minimum test error



Our overfit 100 region decision tree model, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

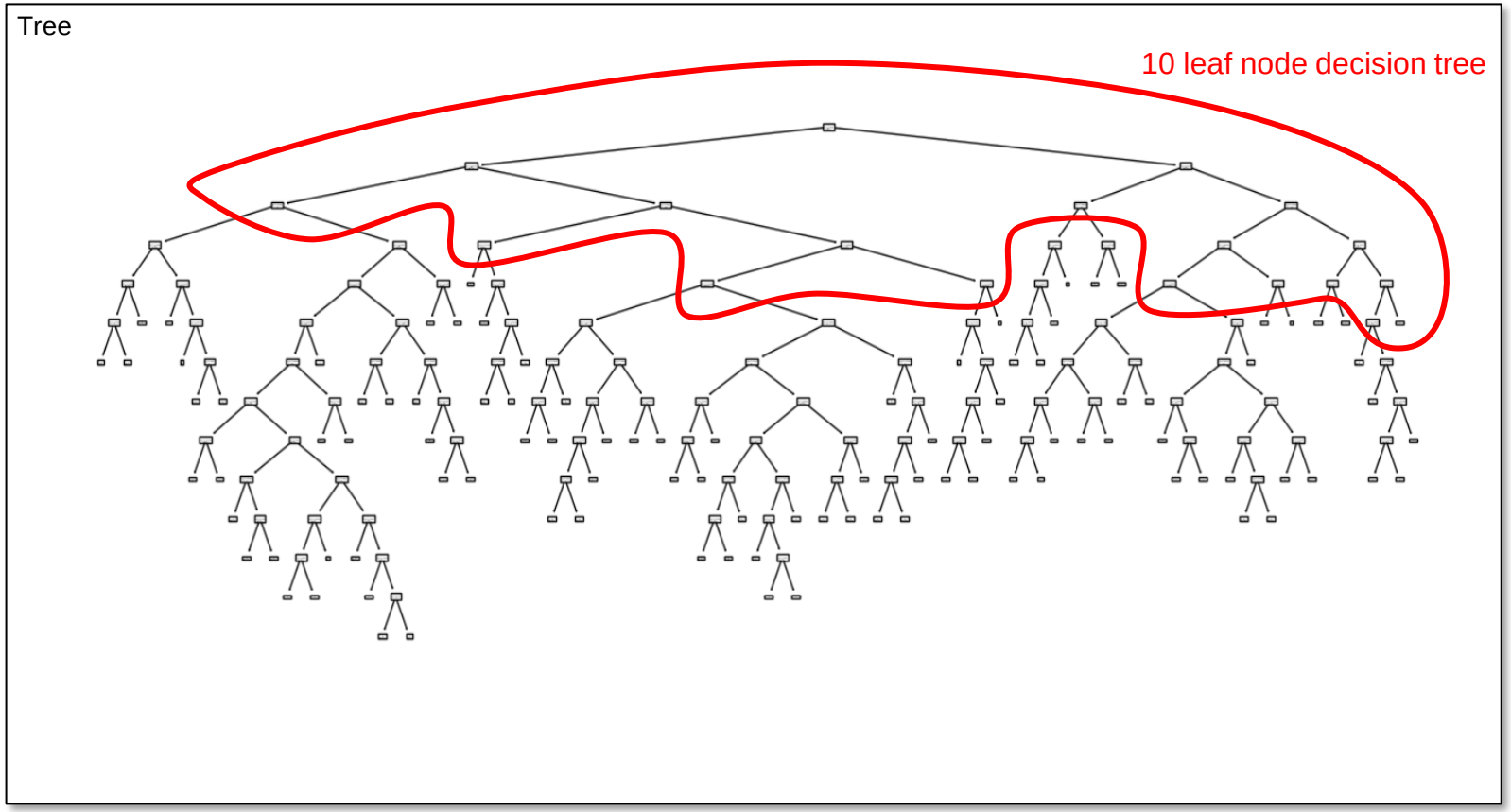
Decision Tree Tuning

To tune the decision tree we take the very overfit model,

- sequentially cut the last decision node
- i.e., prune the last branch of the tree

The simpler trees are inside the complicated tree!

- calculate test error as we prune
- select tree with minimum test error



Our overfit 100 region decision tree model, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

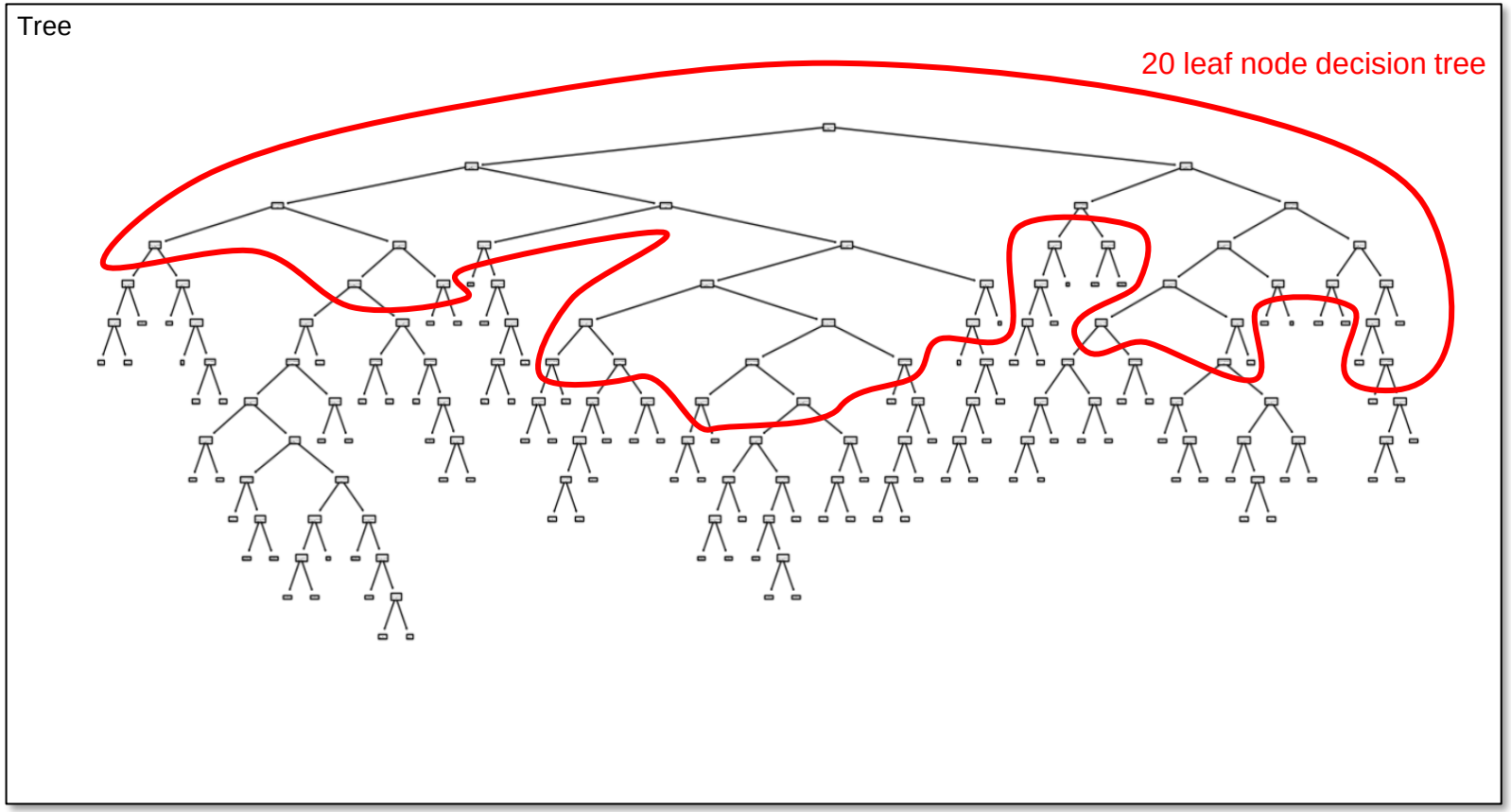
Decision Tree Tuning

To tune the decision tree we take the very overfit model,

- sequentially cut the last decision node
- i.e., prune the last branch of the tree

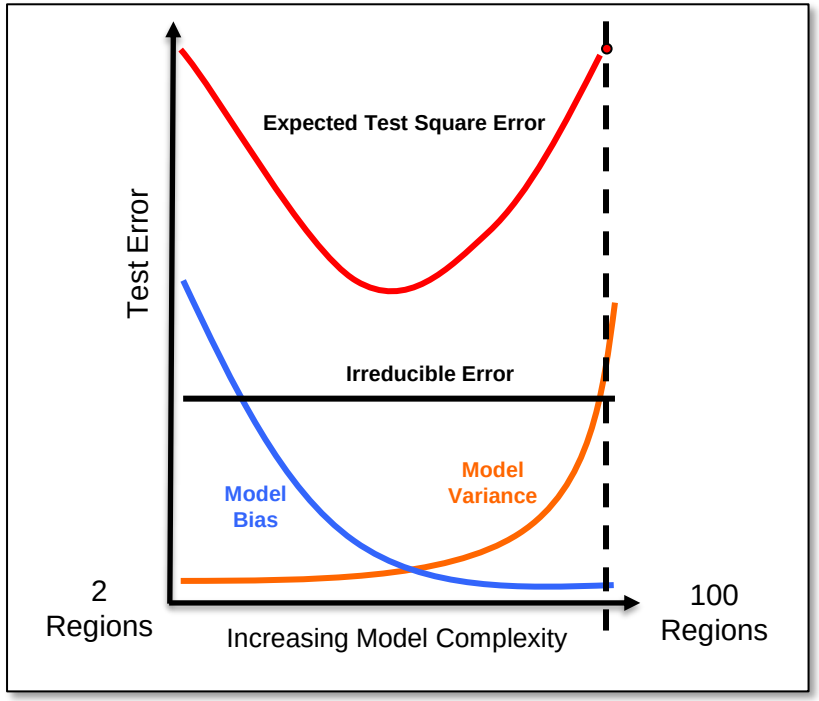
The simpler trees are inside the complicated tree!

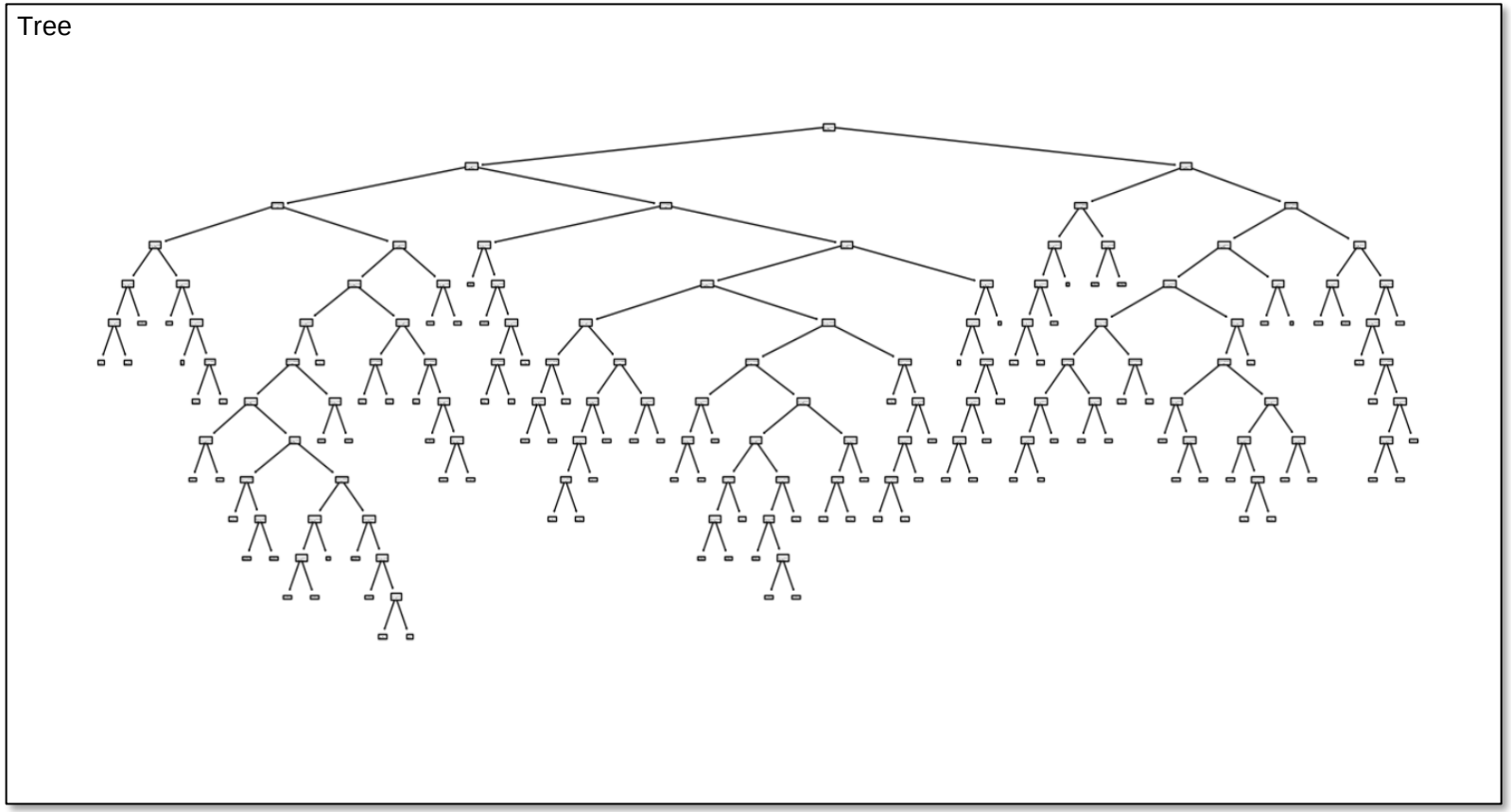
- calculate test error as we prune
- select tree with minimum test error



Our overfit 100 region decision tree model, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Decision Tree Tuning

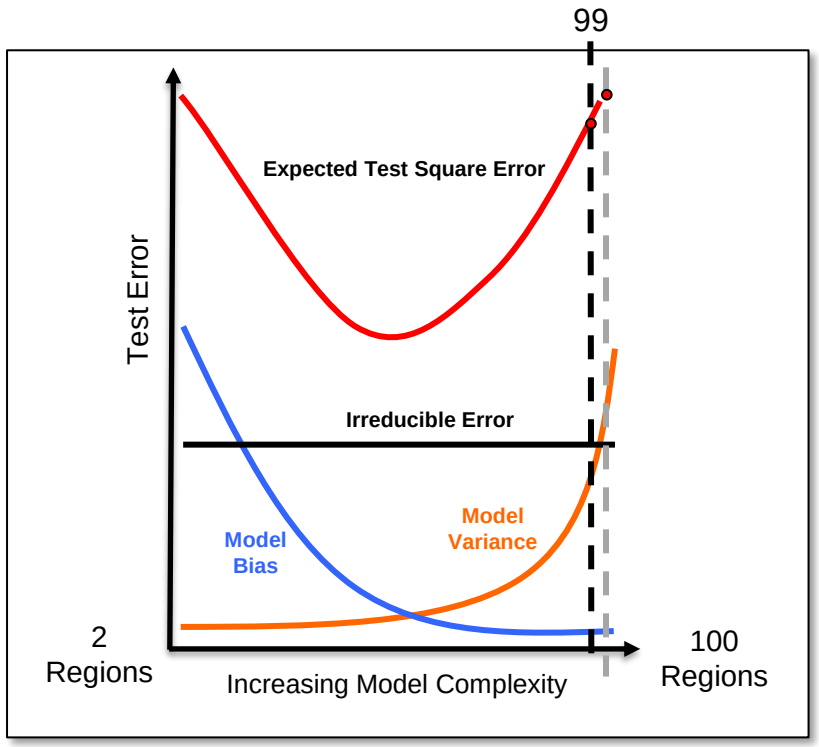


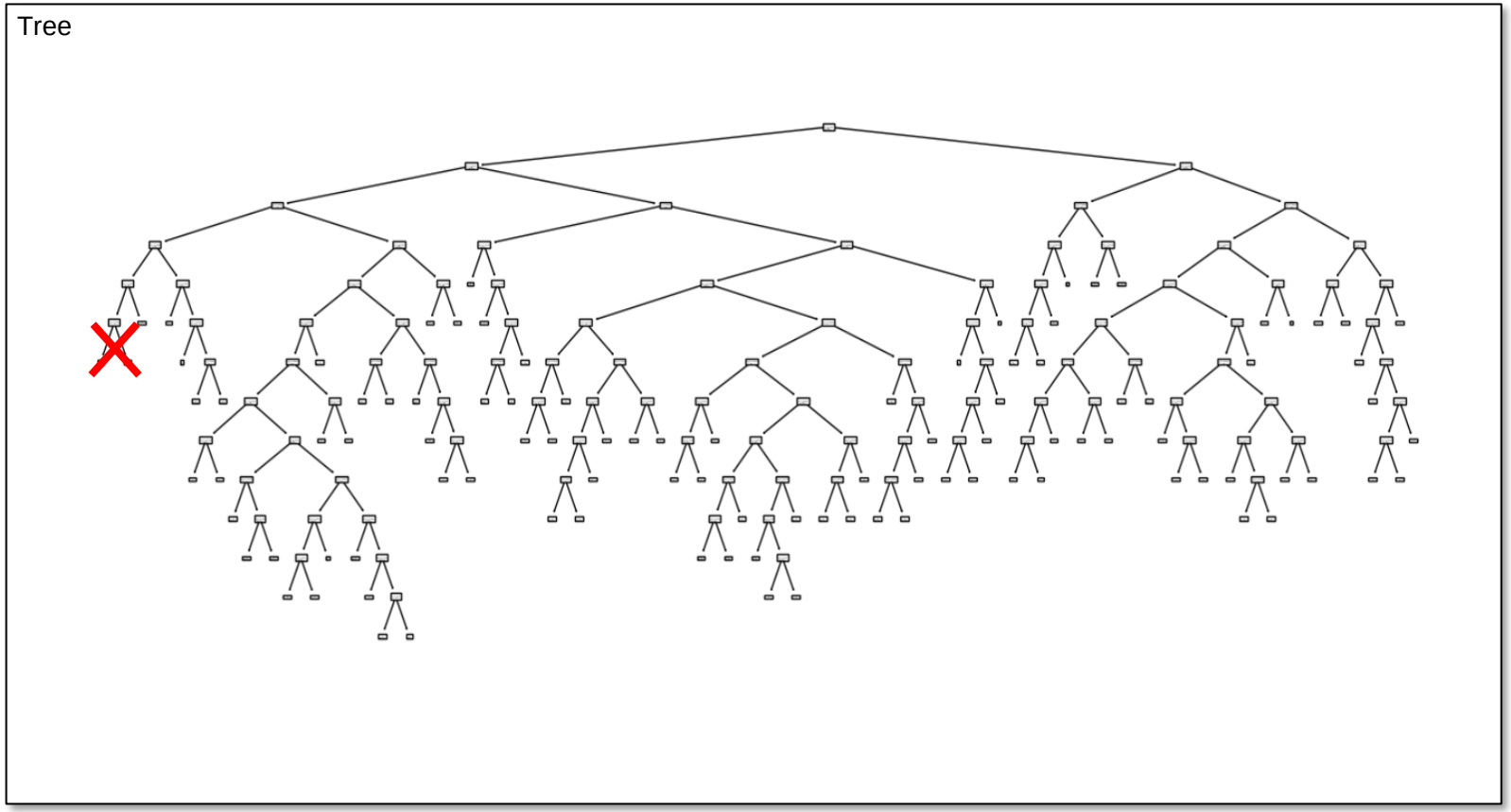


Our overfit 100 region decision tree model, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 100 regions (leaf nodes) – very, very overfit model

Decision Tree Tuning

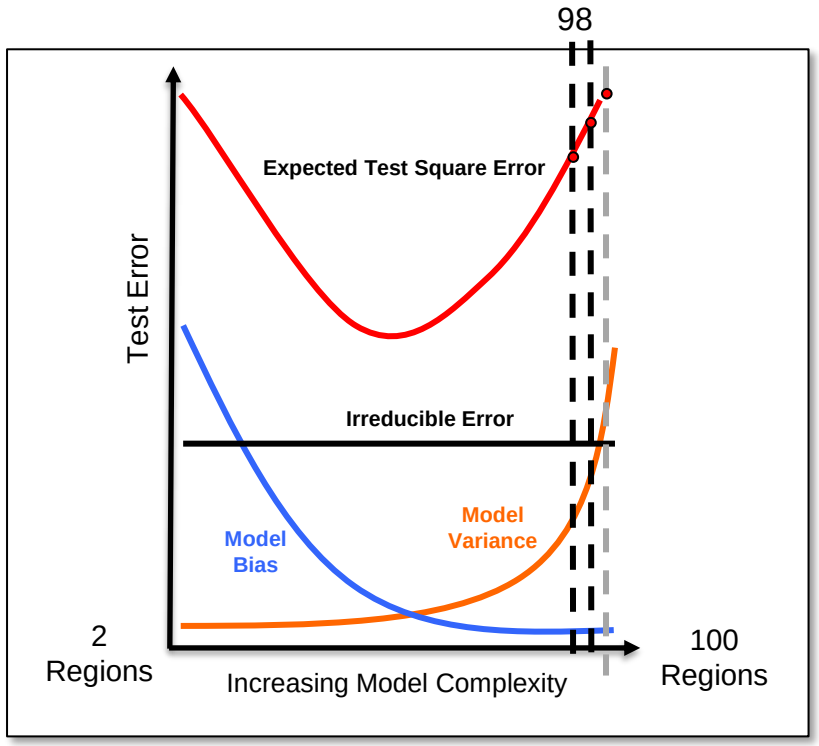


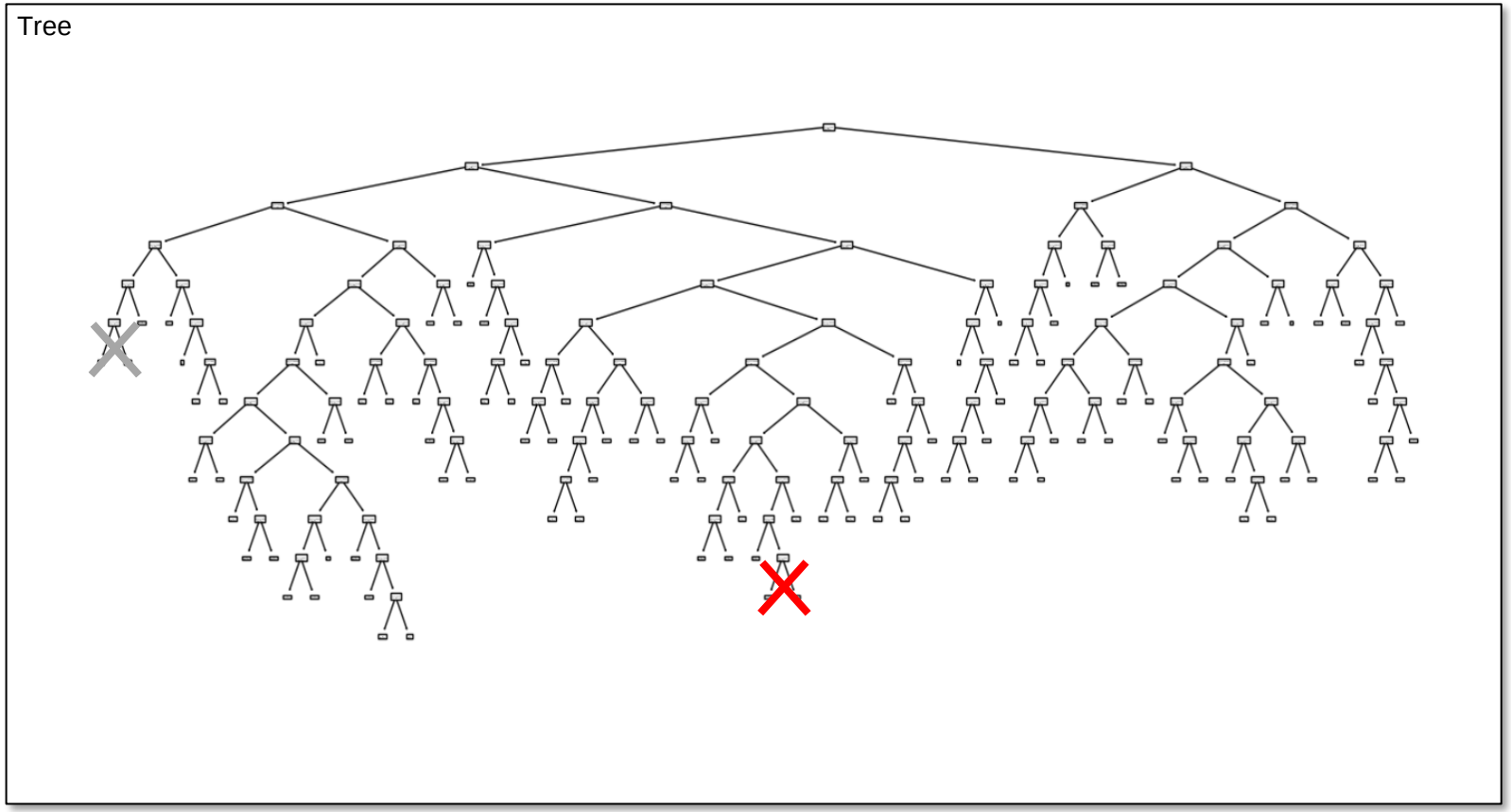


Our overfit 100 region decision tree model pruned to 99 regions, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 99 regions (leaf nodes) – very, very overfit model

Decision Tree Tuning

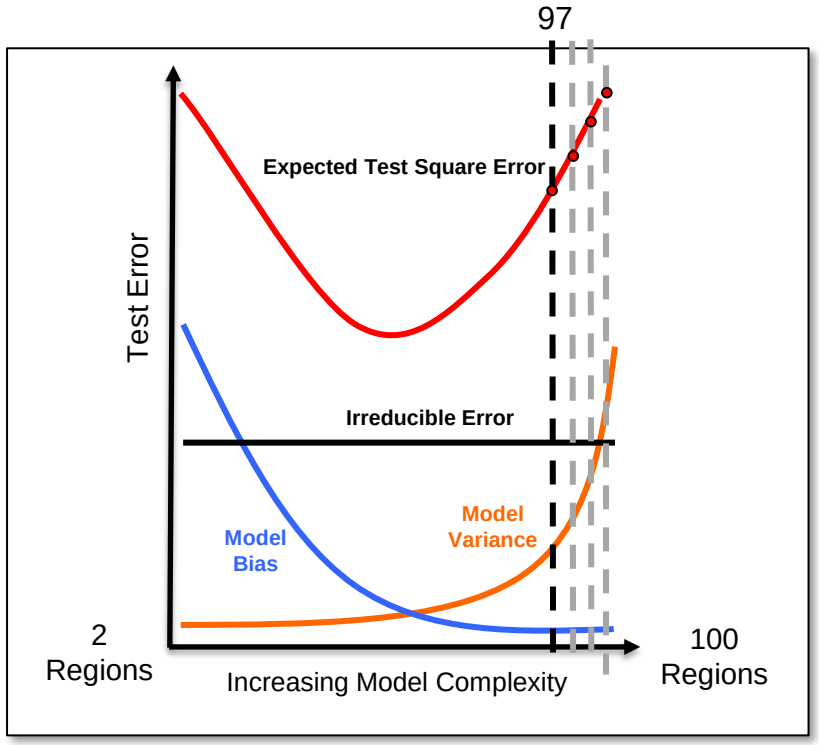


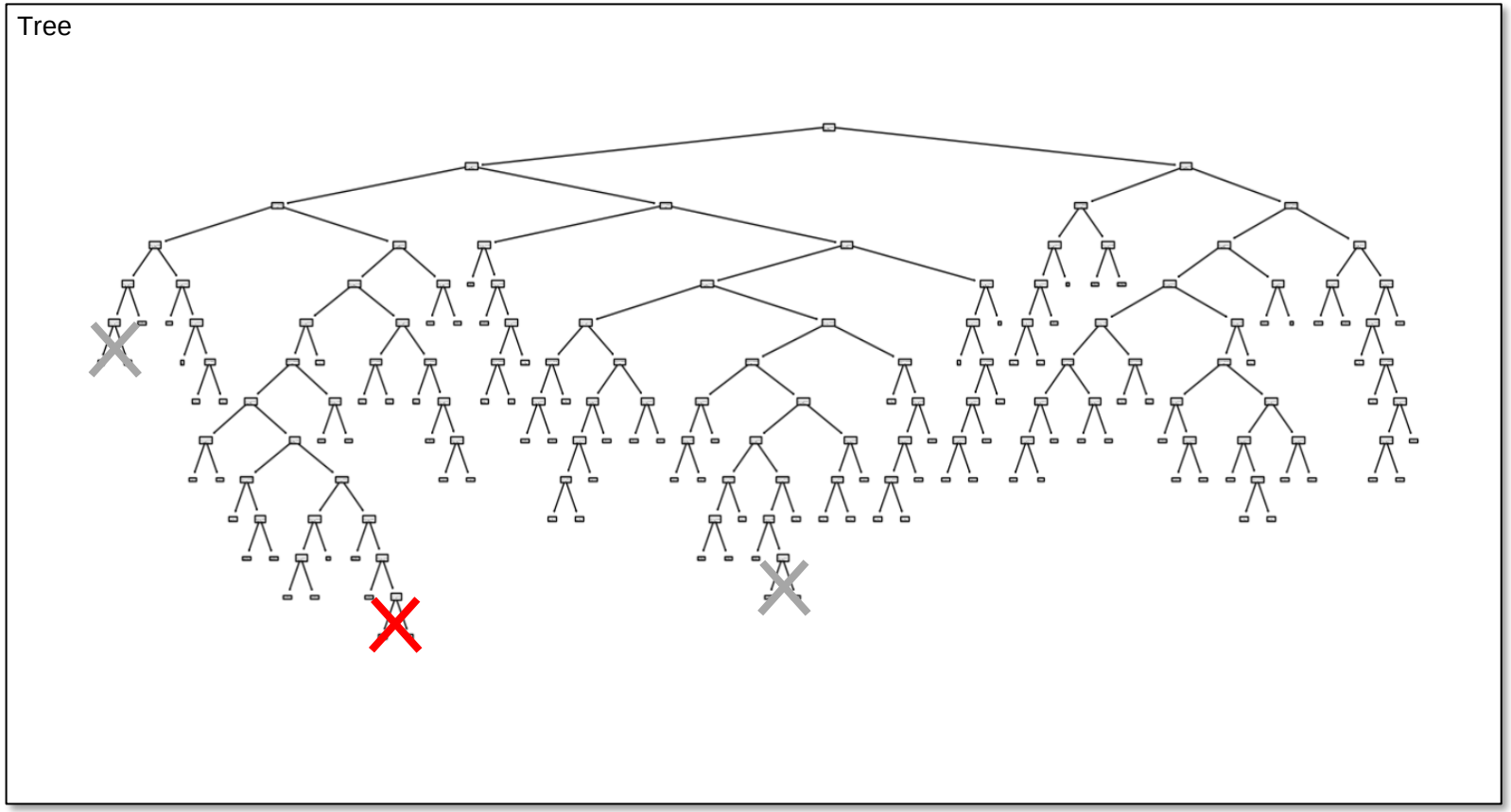


Our overfit 100 region decision tree model pruned to 98 regions, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 98 regions (leaf nodes) – very, very overfit model

Decision Tree Tuning

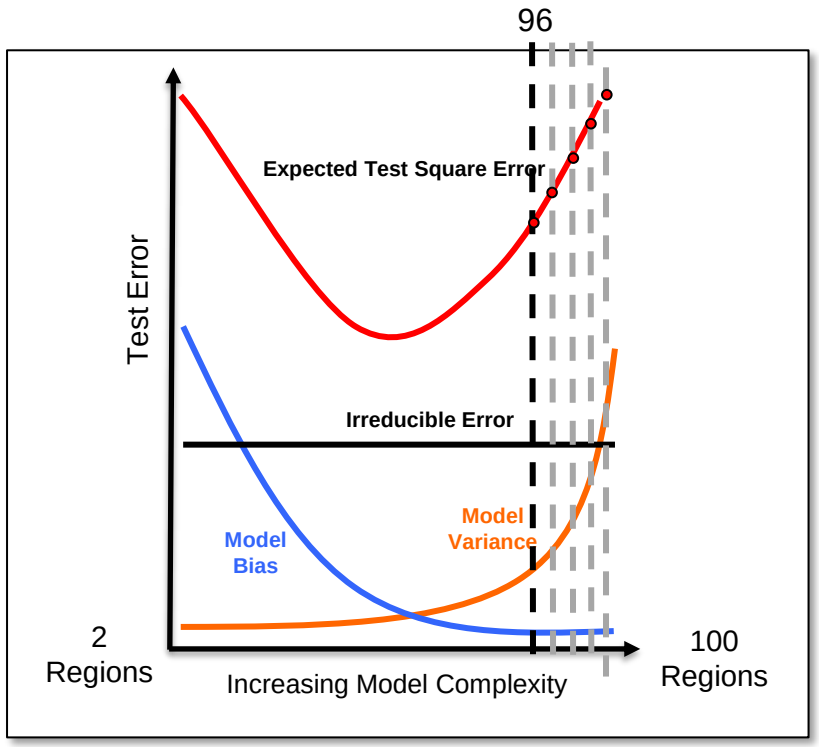


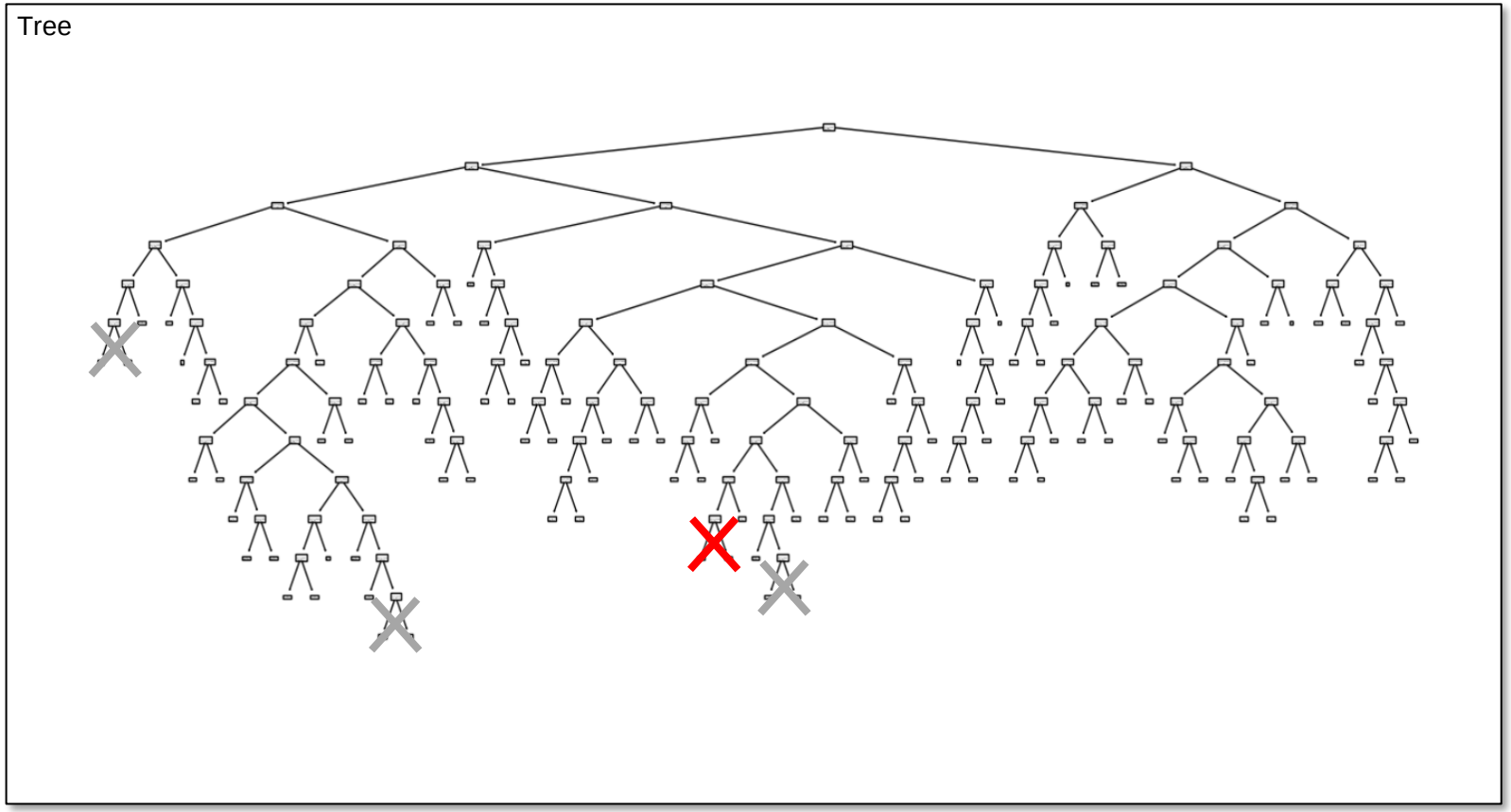


Our overfit 100 region decision tree model pruned to 97 regions, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 97 regions (leaf nodes) – very, very overfit model

Decision Tree Tuning

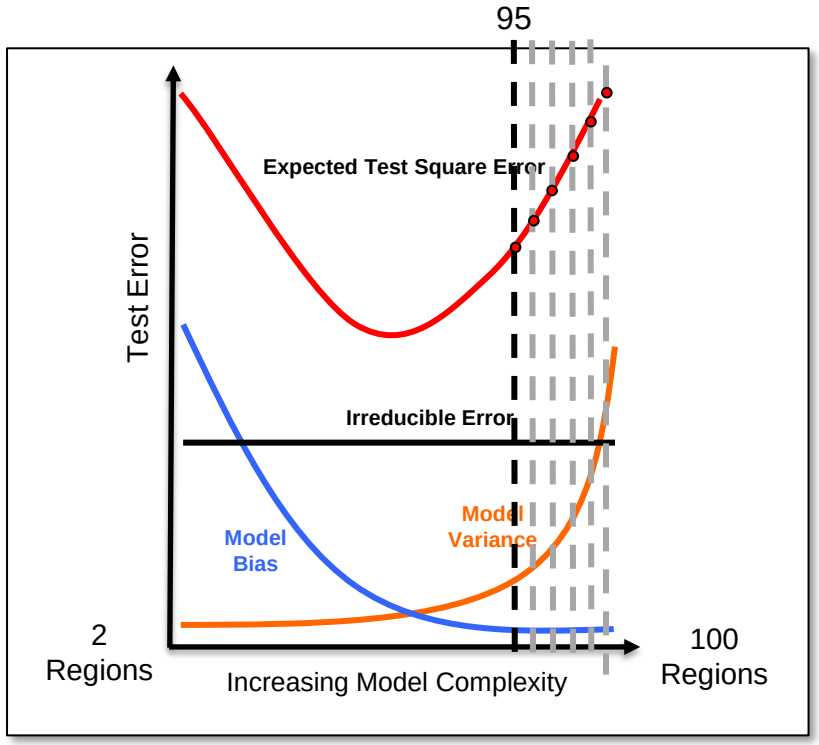


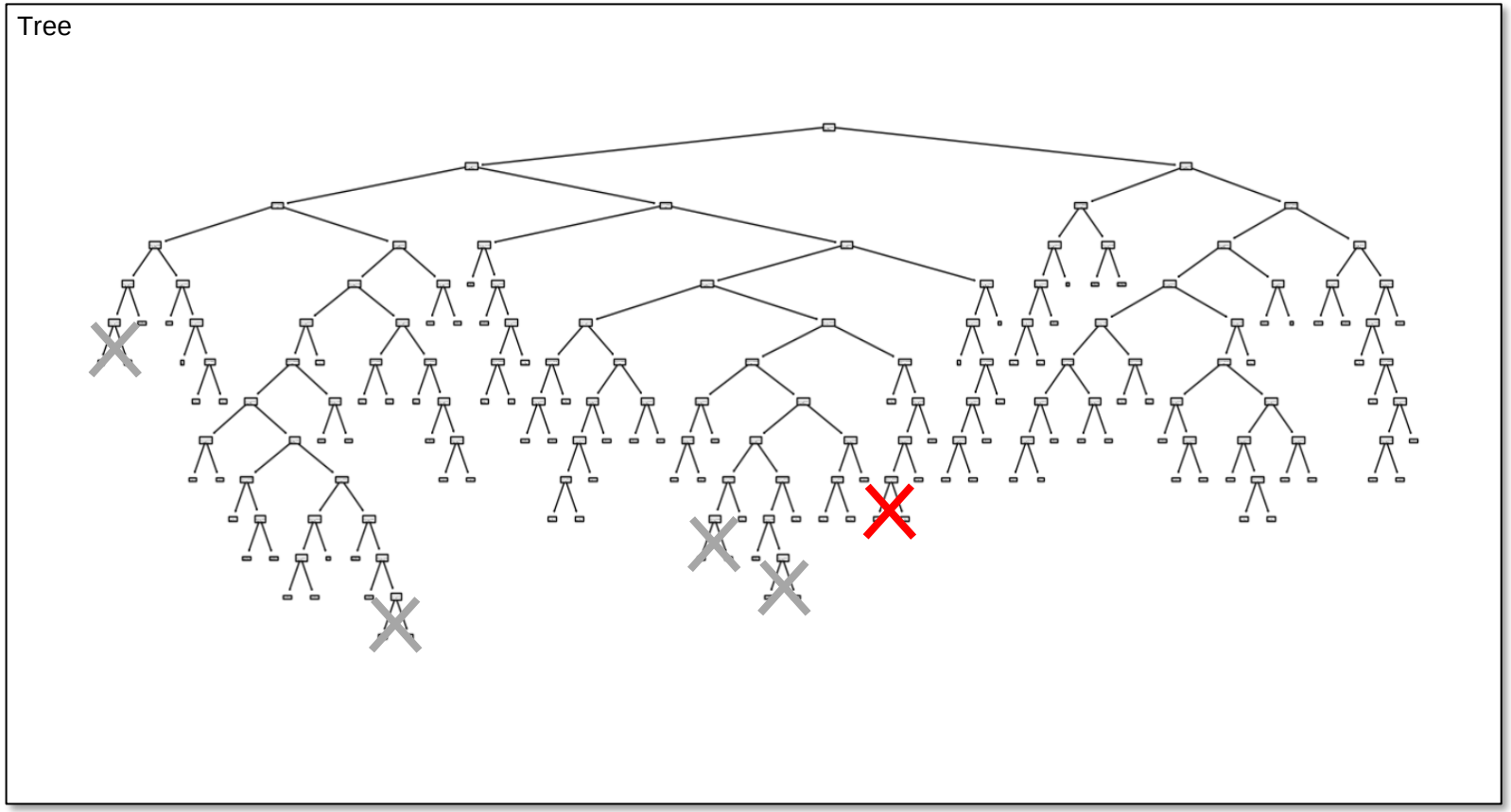


Our overfit 100 region decision tree model pruned to 96 regions, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 96 regions (leaf nodes) – very, very overfit model

Decision Tree Tuning

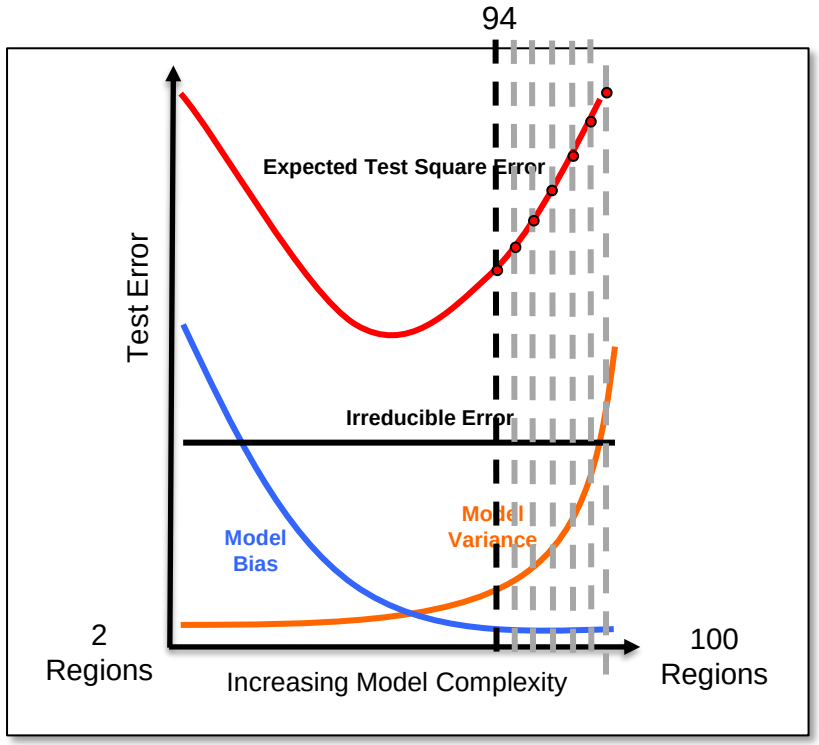


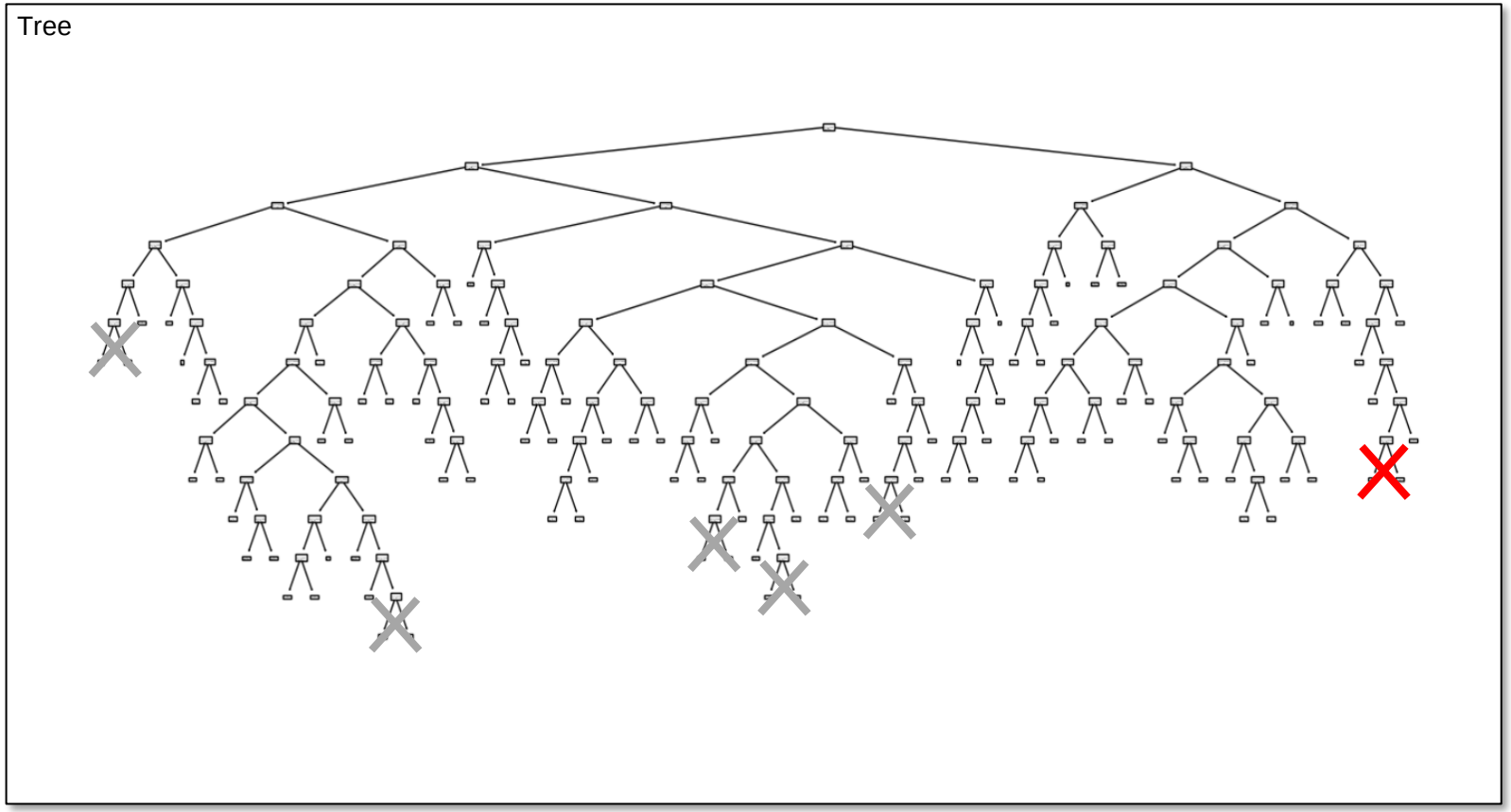


Our overfit 100 region decision tree model pruned to 95 regions, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 95 regions (leaf nodes) – very, very overfit model

Decision Tree Tuning





Our overfit 100 region decision tree model pruned to 94 regions, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 94 regions (leaf nodes) – very, very overfit model

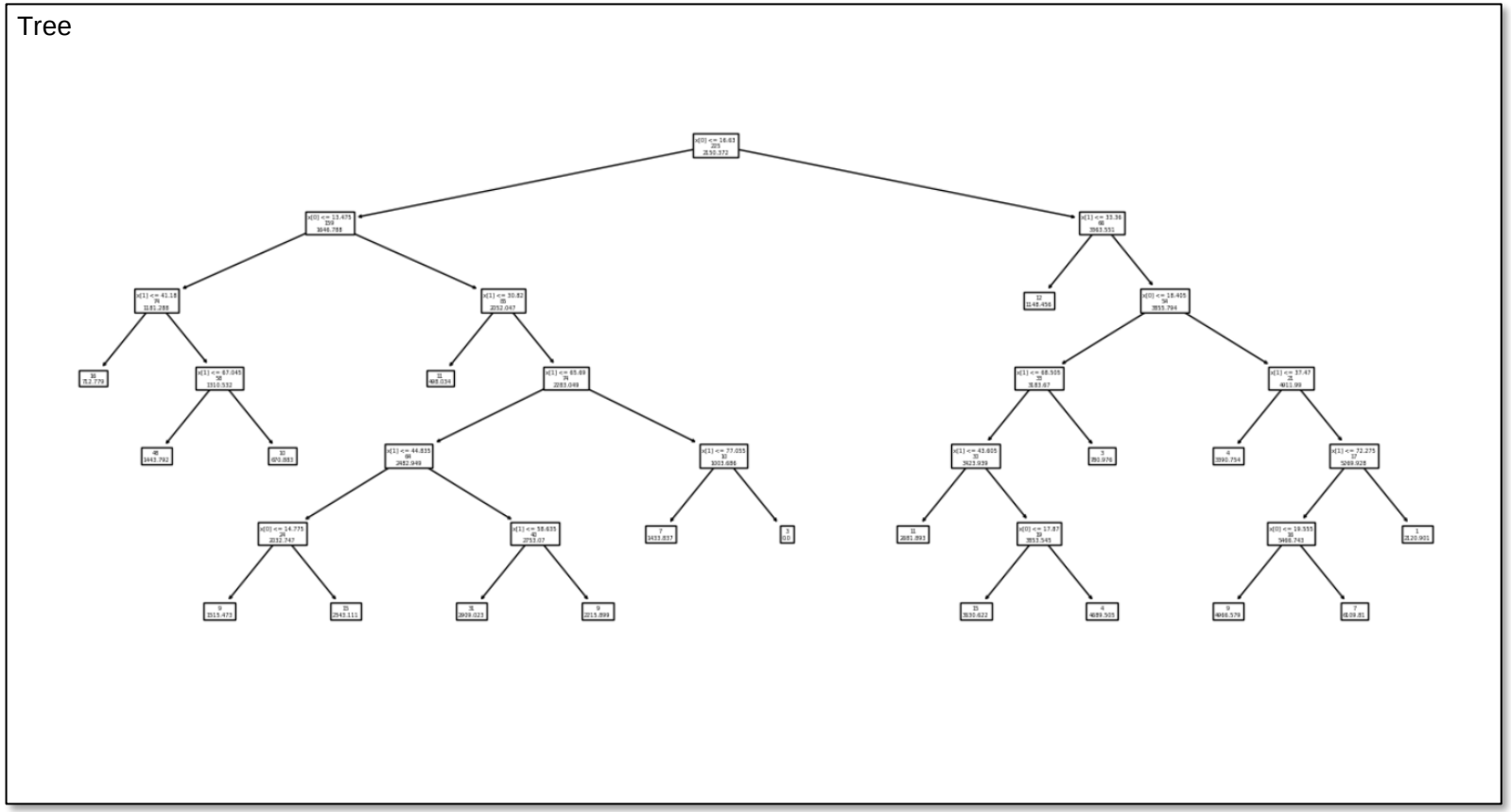
Decision Tree Tuning

Visualizing Decision Tree Complexity

I made a interactive Python dashboard to visualize,

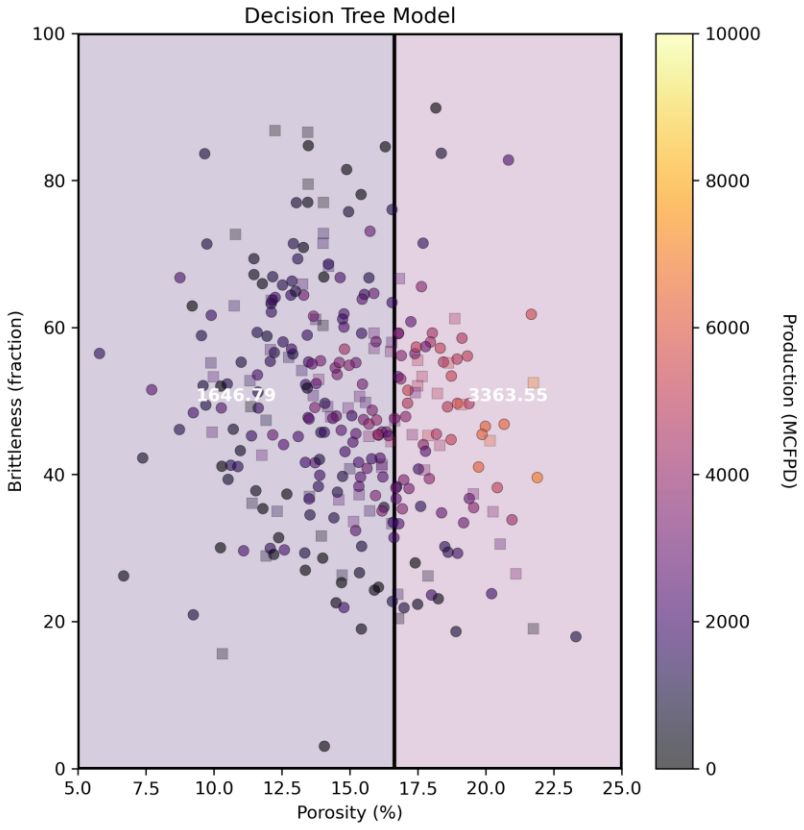
- tree models
- prediction vs. truth plots
- train and test vs. complexity plots

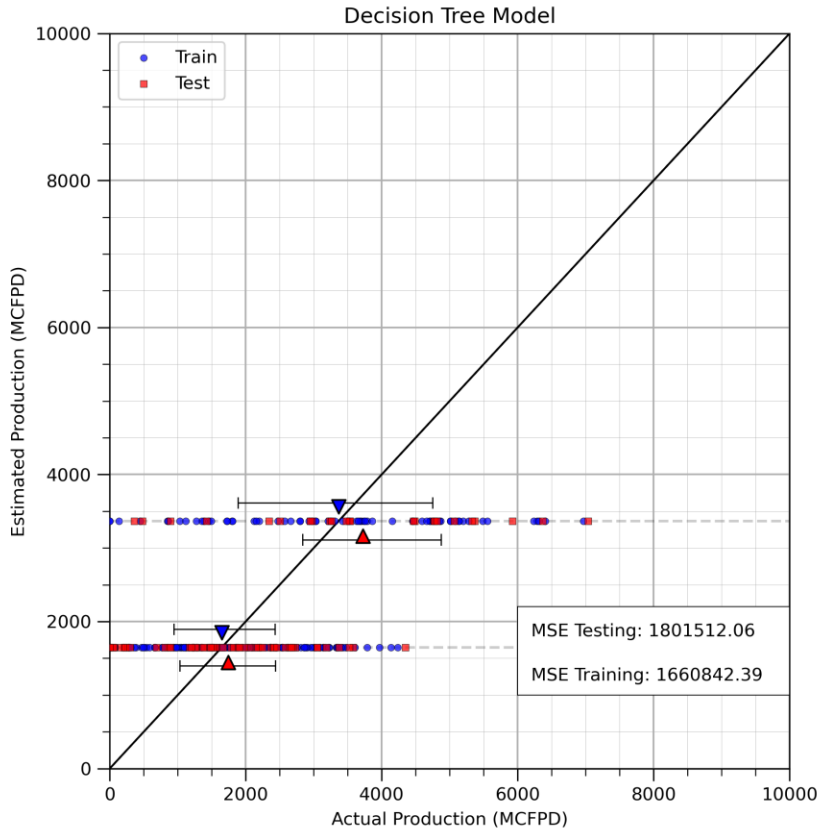
Let's look at some simple to complicated tree models.

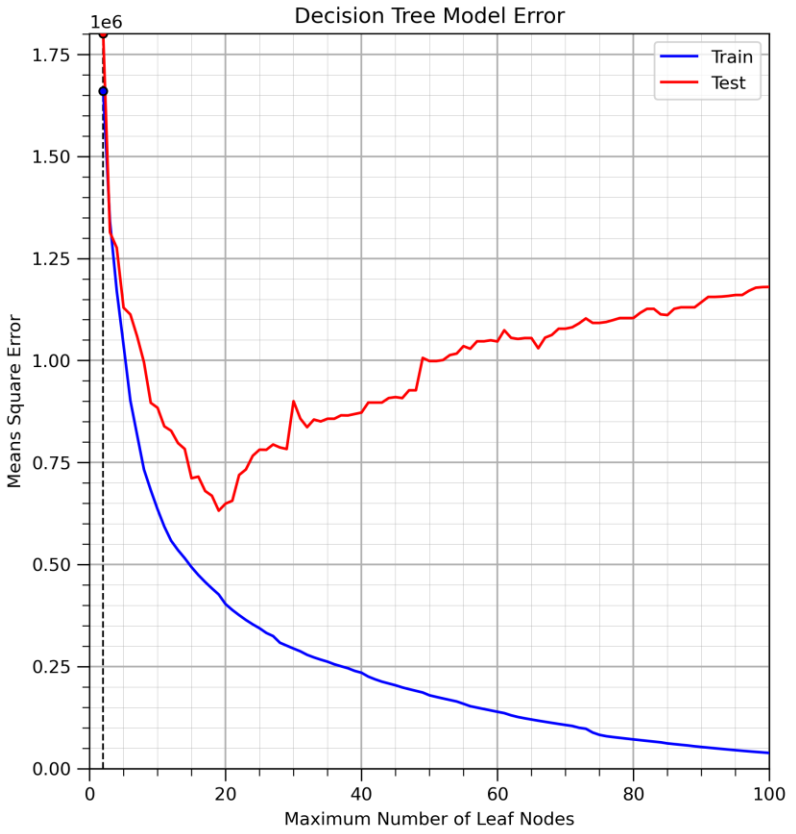


Our tuned 20 region decision tree model, from Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Decision Tree Training and Tuning



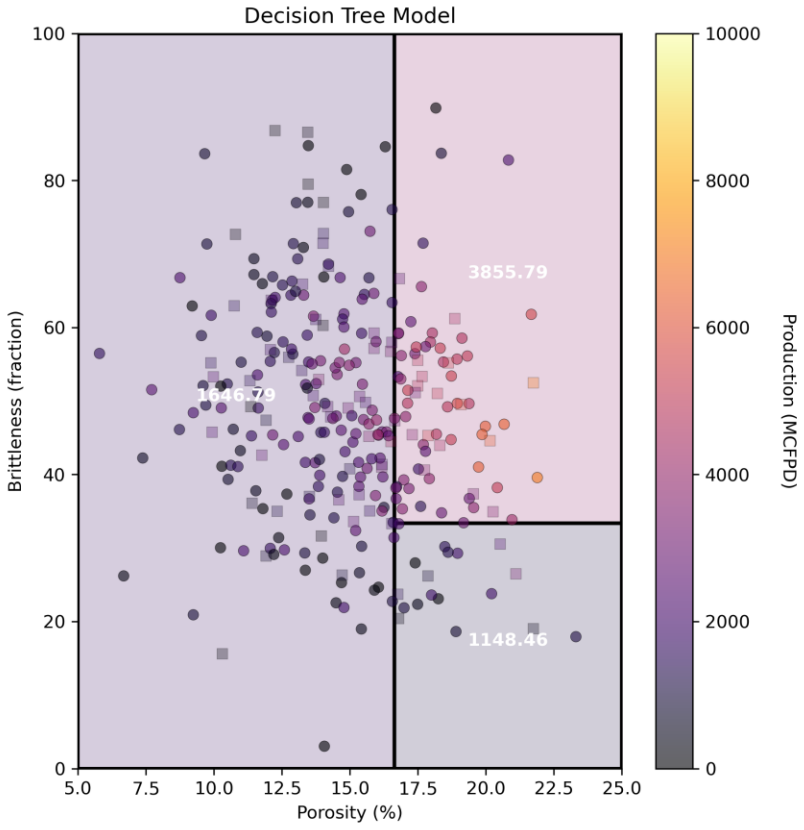


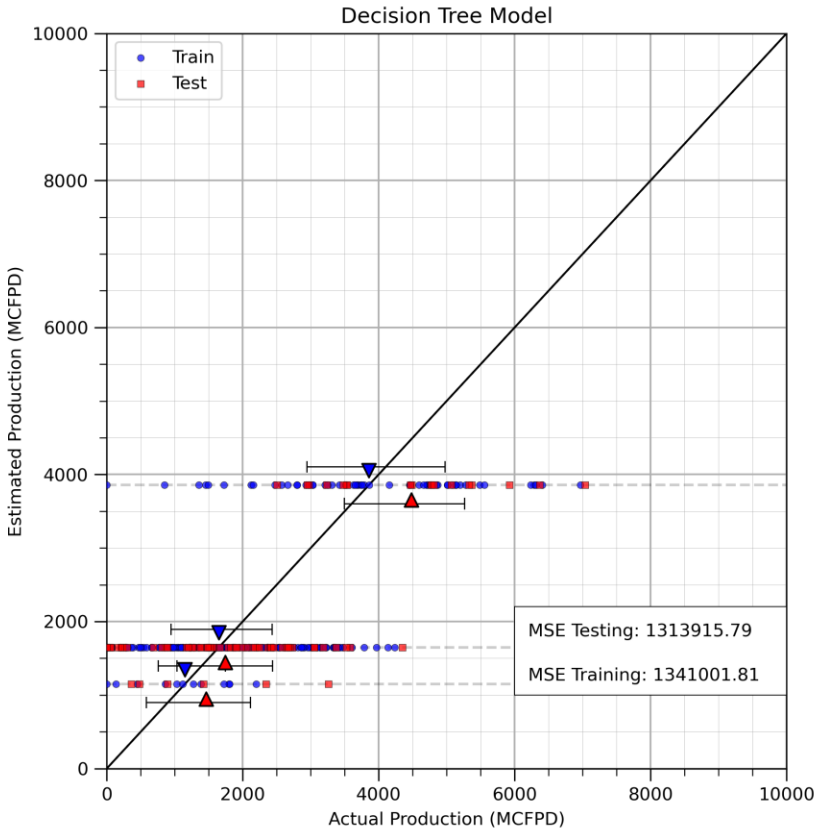


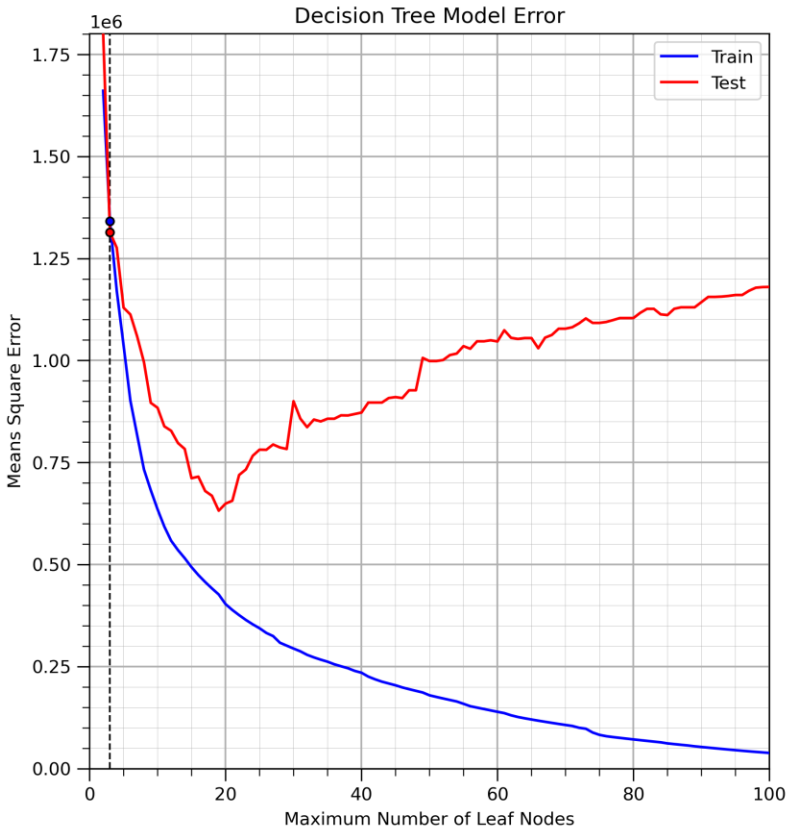
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 2 regions (leaf nodes) – very, very underfit model

Decision Tree Training and Tuning



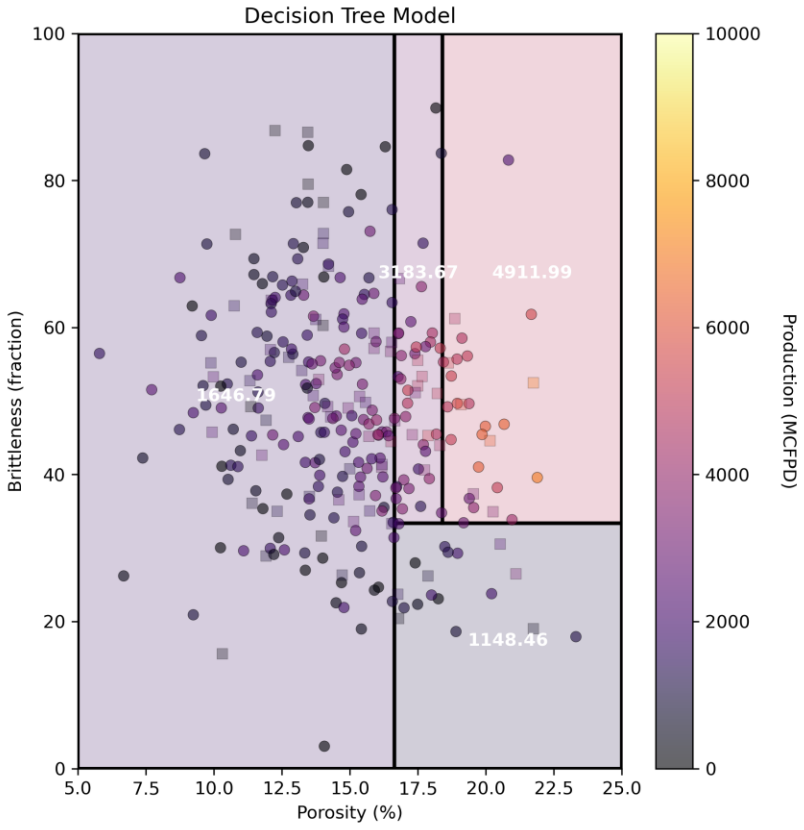


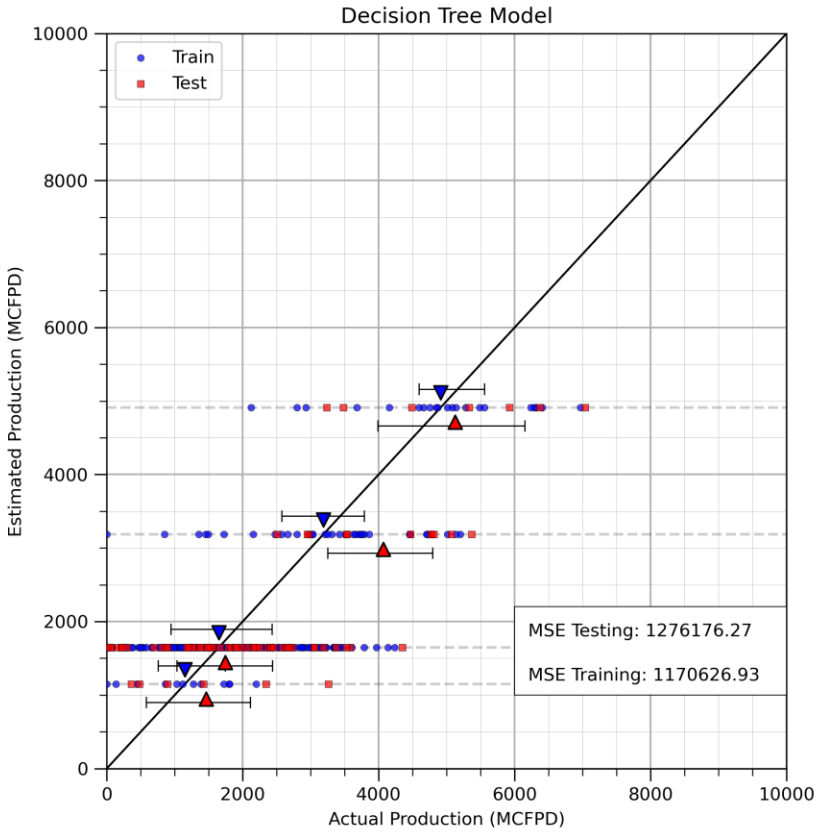


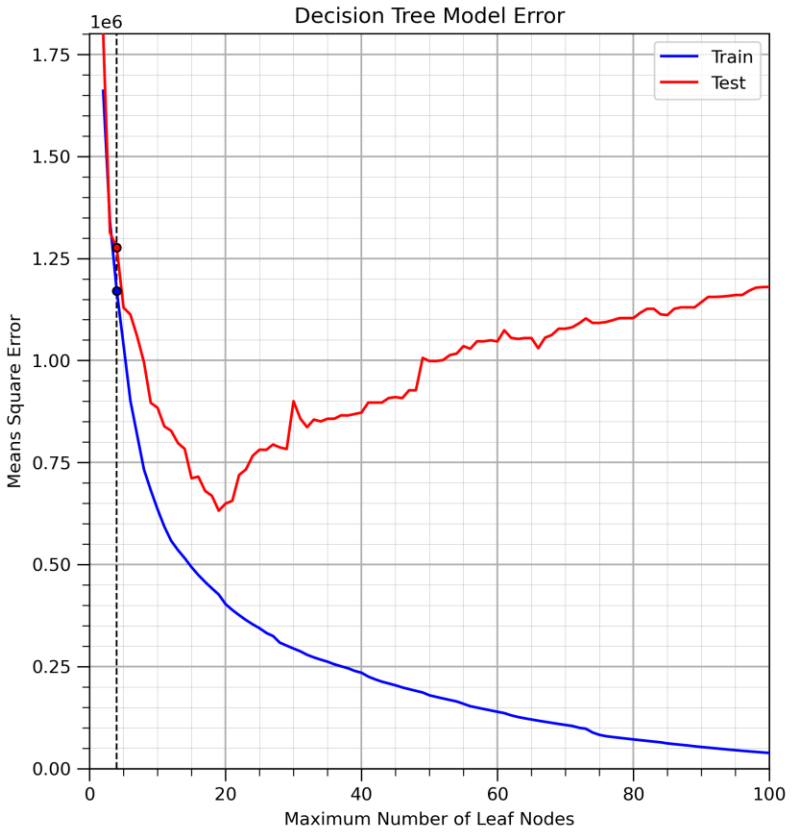
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 3 regions (leaf nodes) – very, very underfit model

Decision Tree Training and Tuning



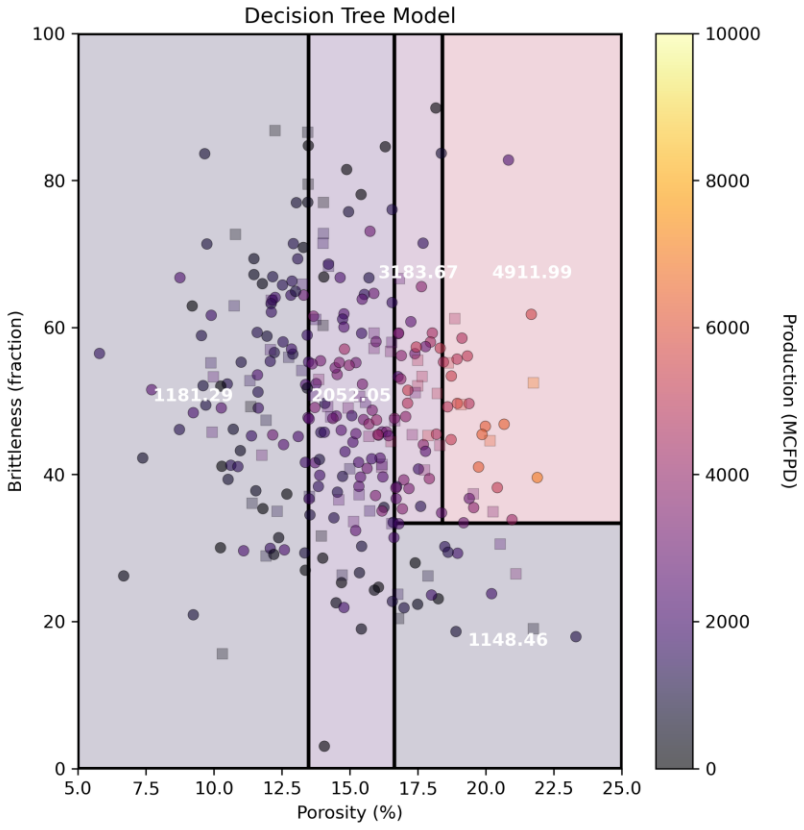


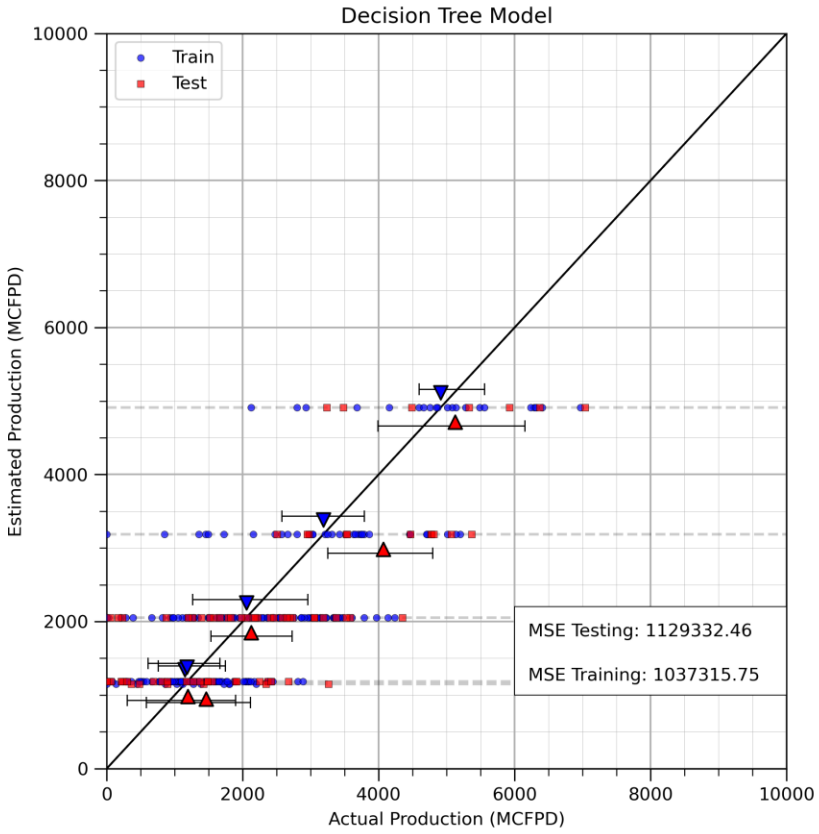


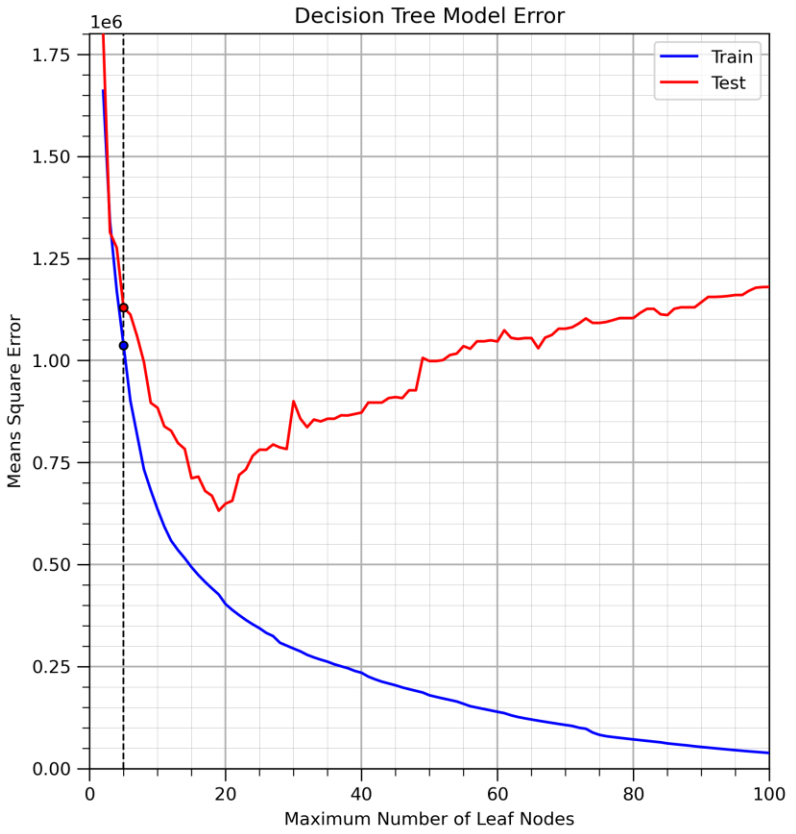
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 4 regions (leaf nodes) – very underfit model

Decision Tree Training and Tuning



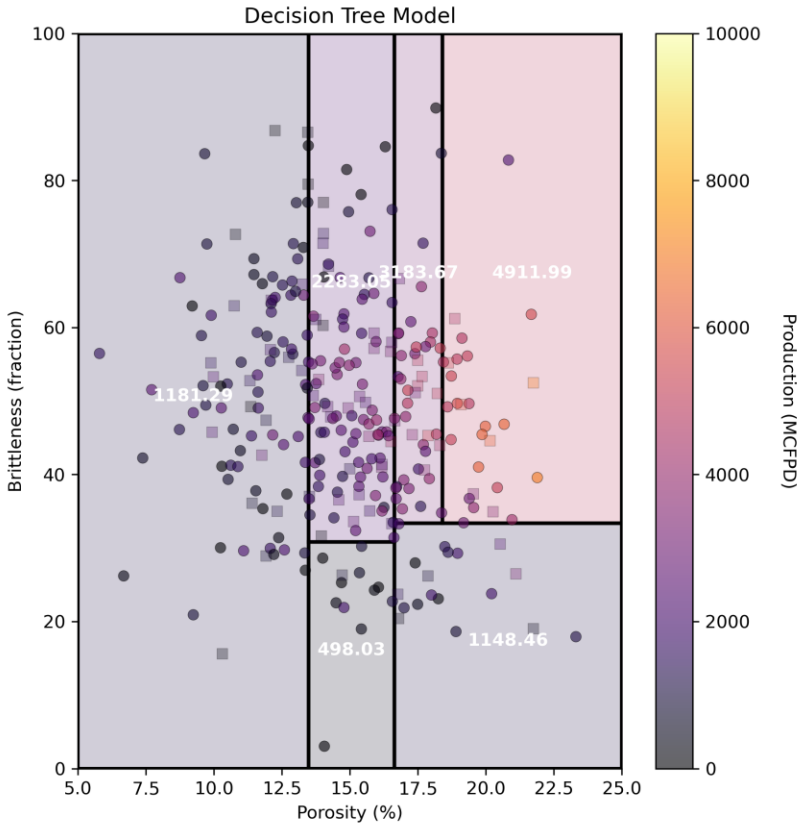


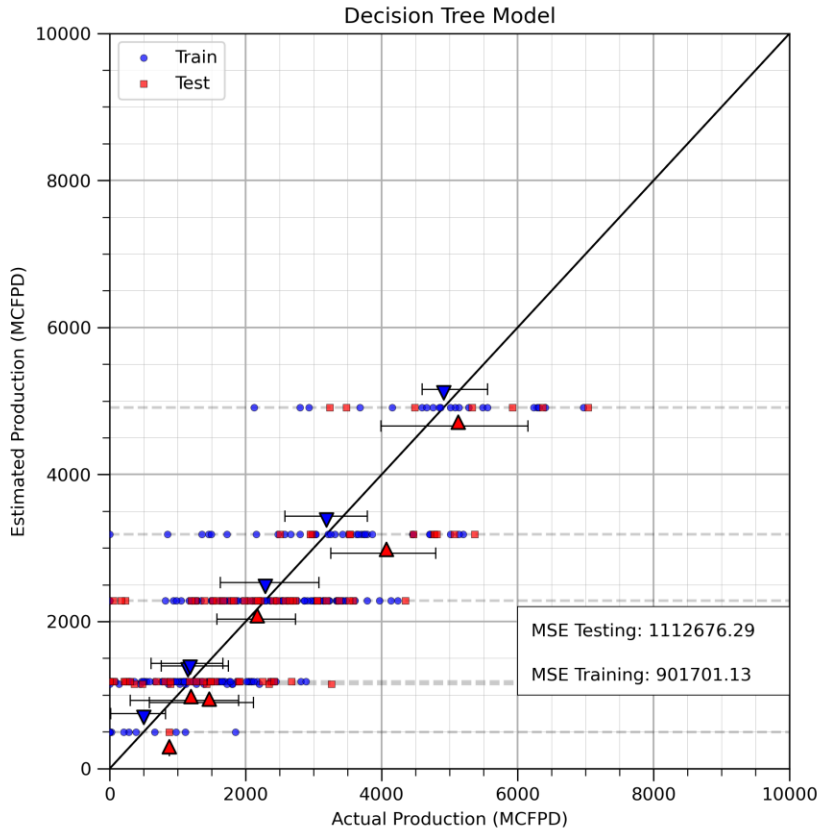


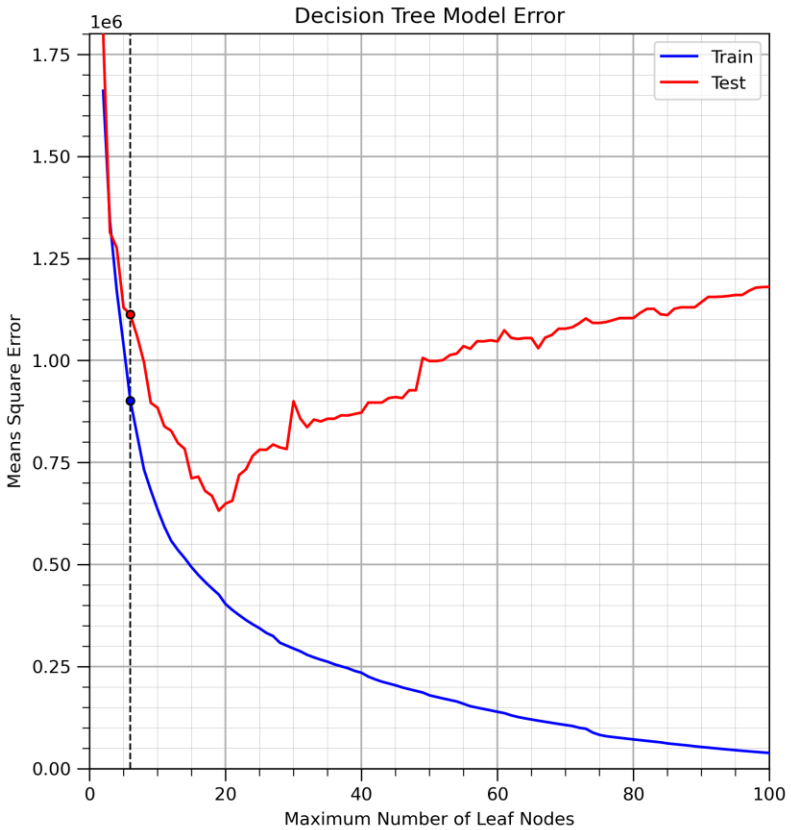
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 5 regions (leaf nodes) – very underfit model

Decision Tree Training and Tuning



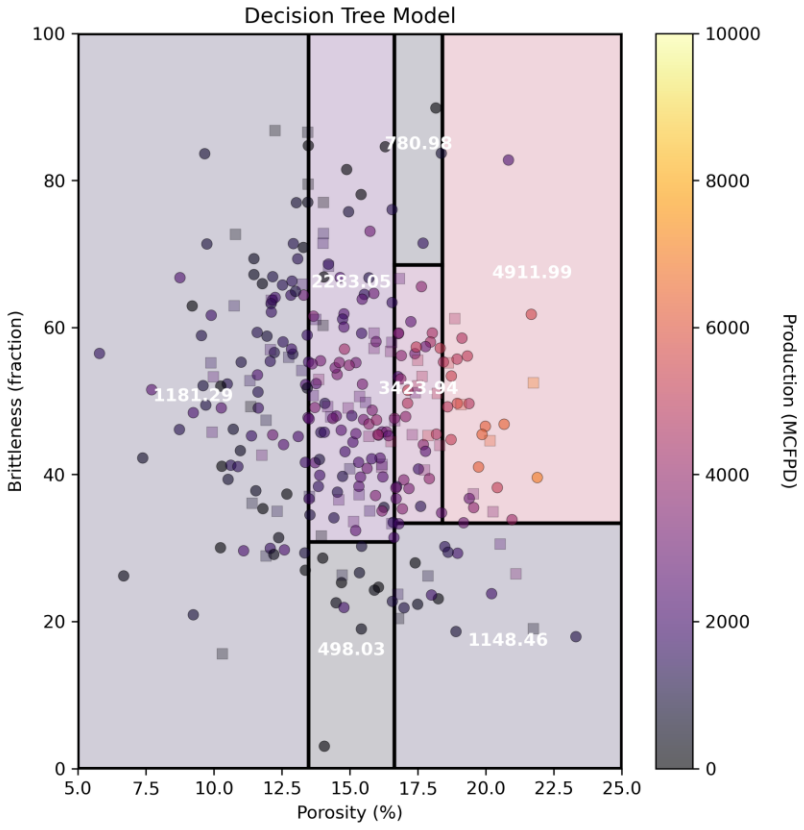


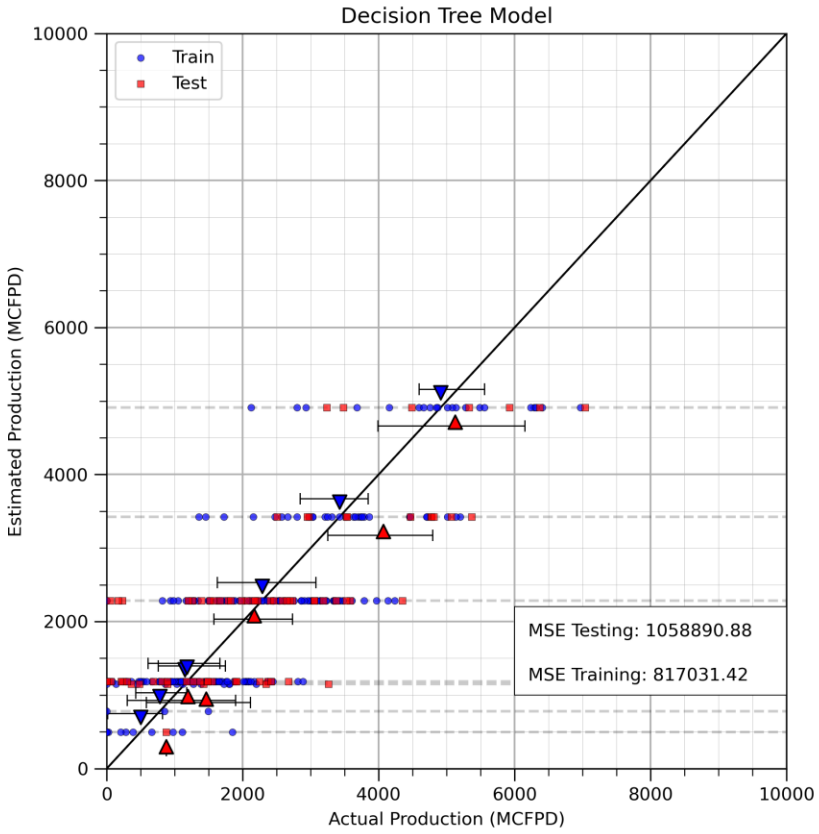


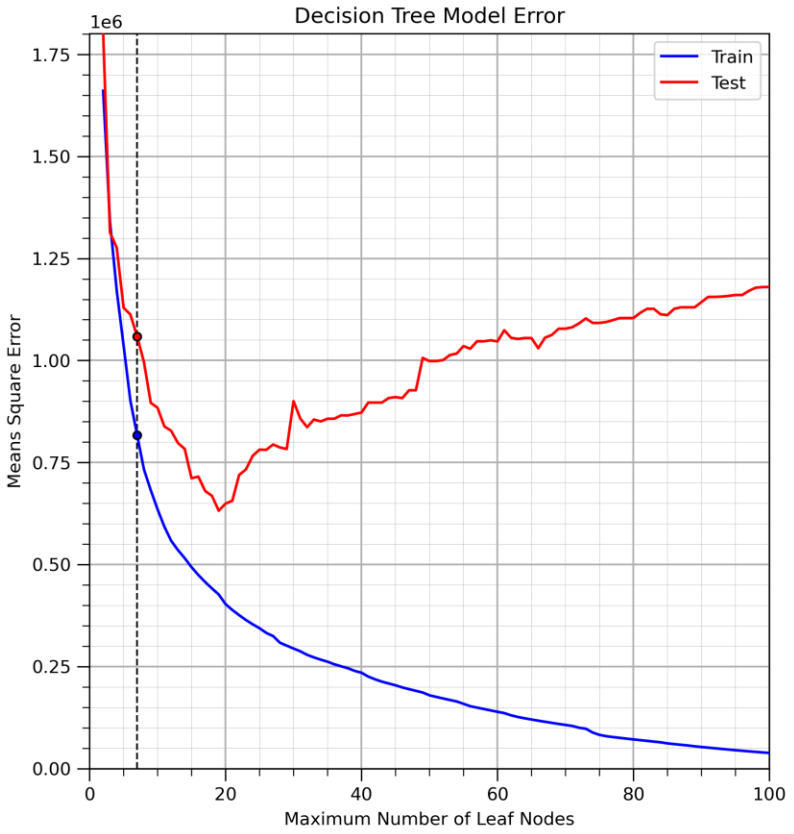
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 6 regions (leaf nodes) – underfit model

Decision Tree Training and Tuning



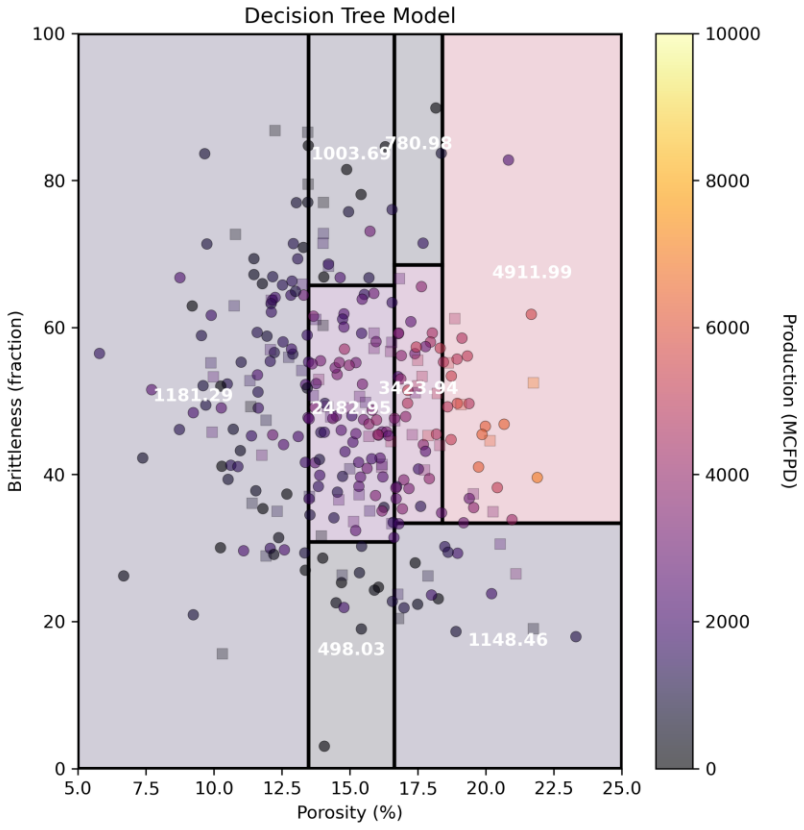


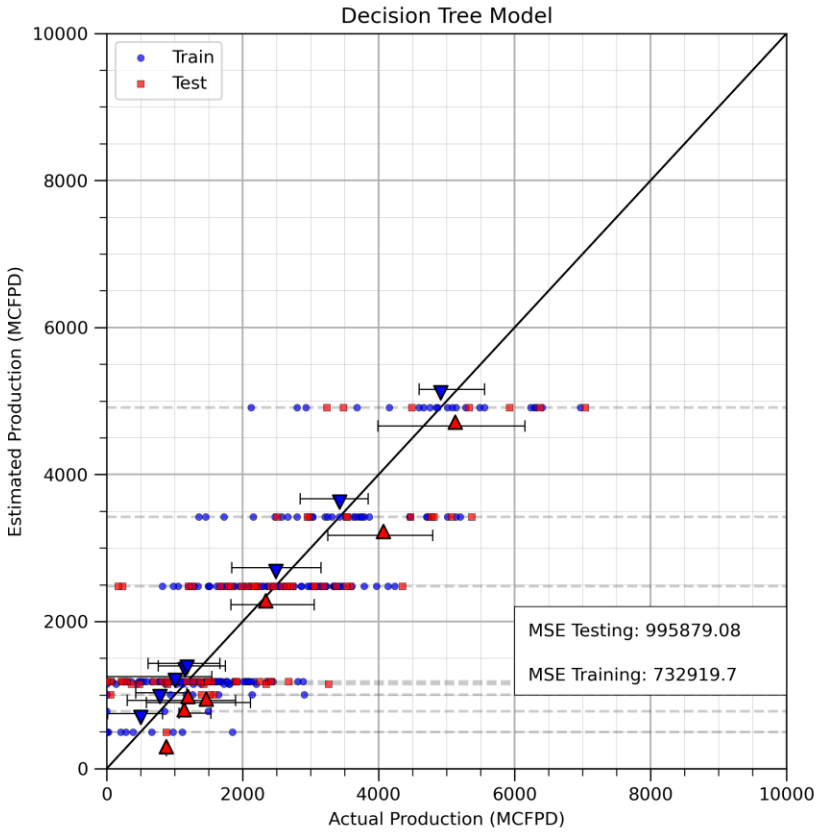


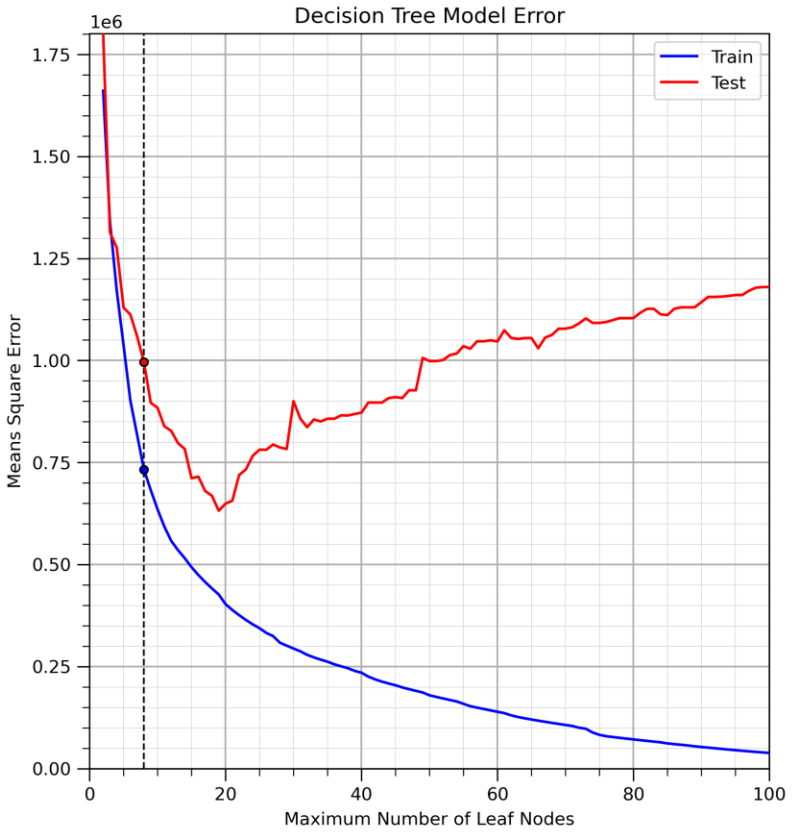
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 7 regions (leaf nodes) – underfit model

Decision Tree Training and Tuning



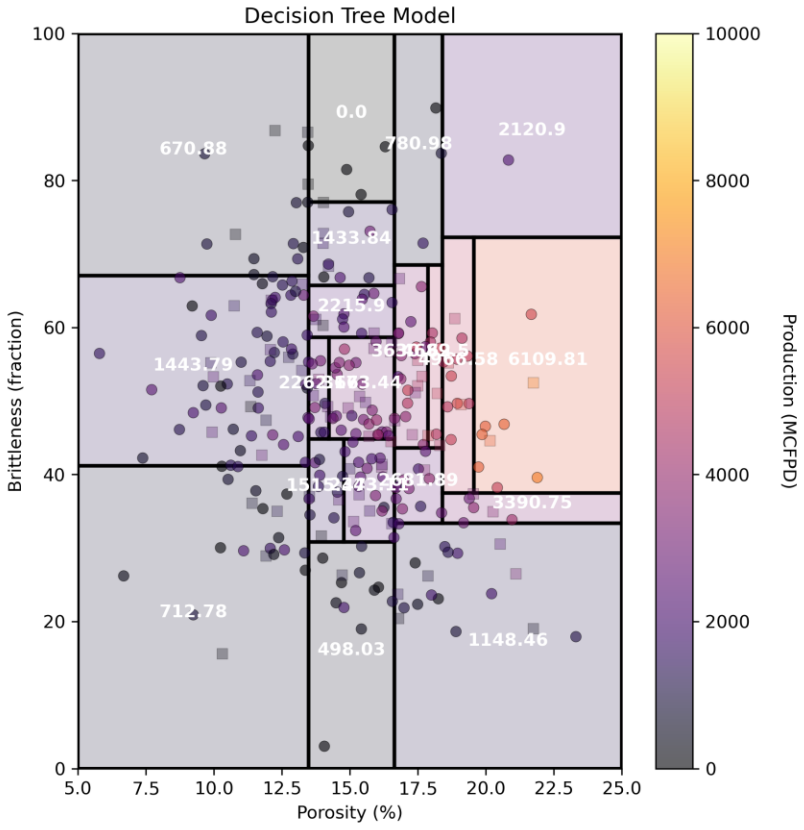


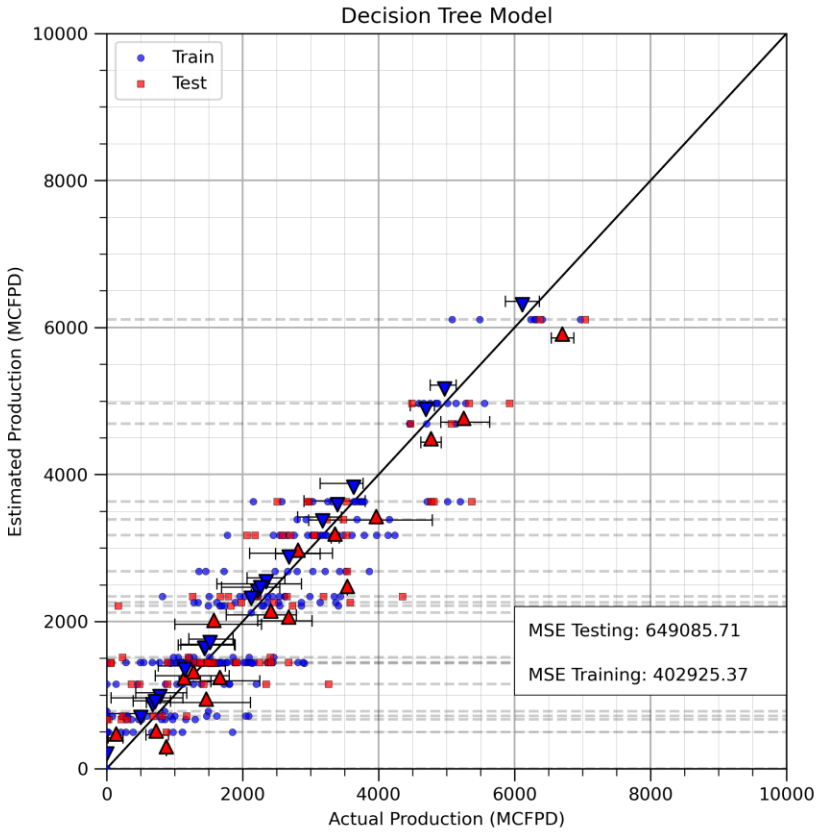


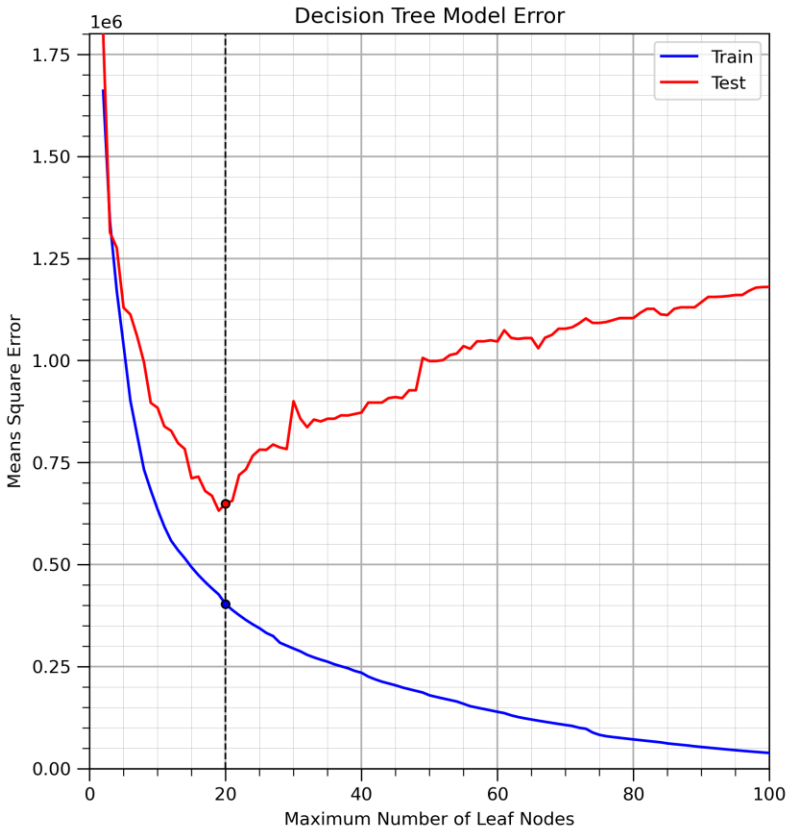
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 8 regions (leaf nodes) – underfit model

Decision Tree Training and Tuning



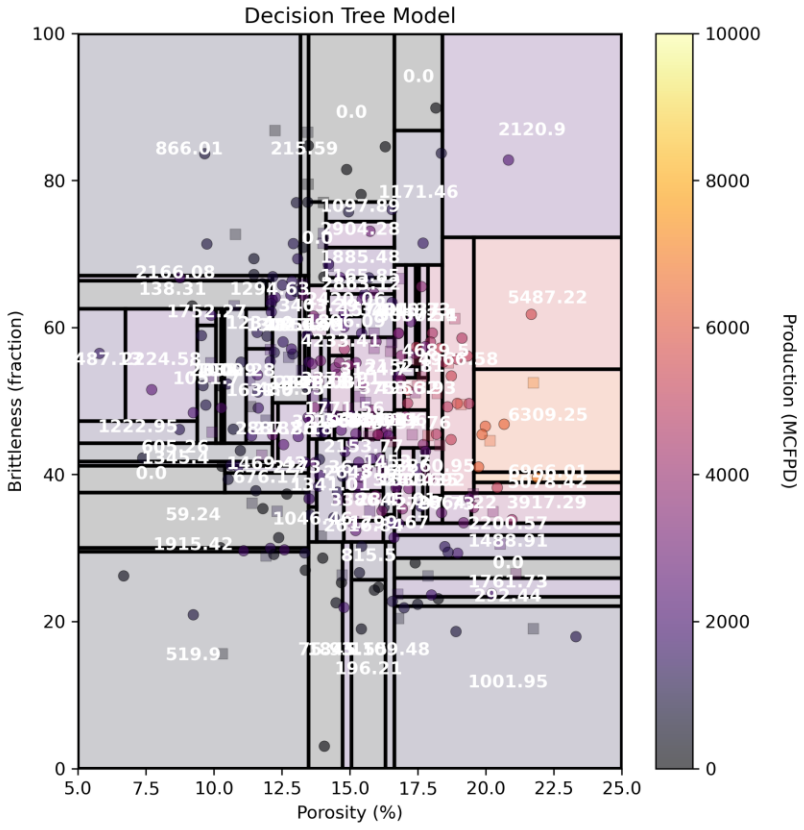


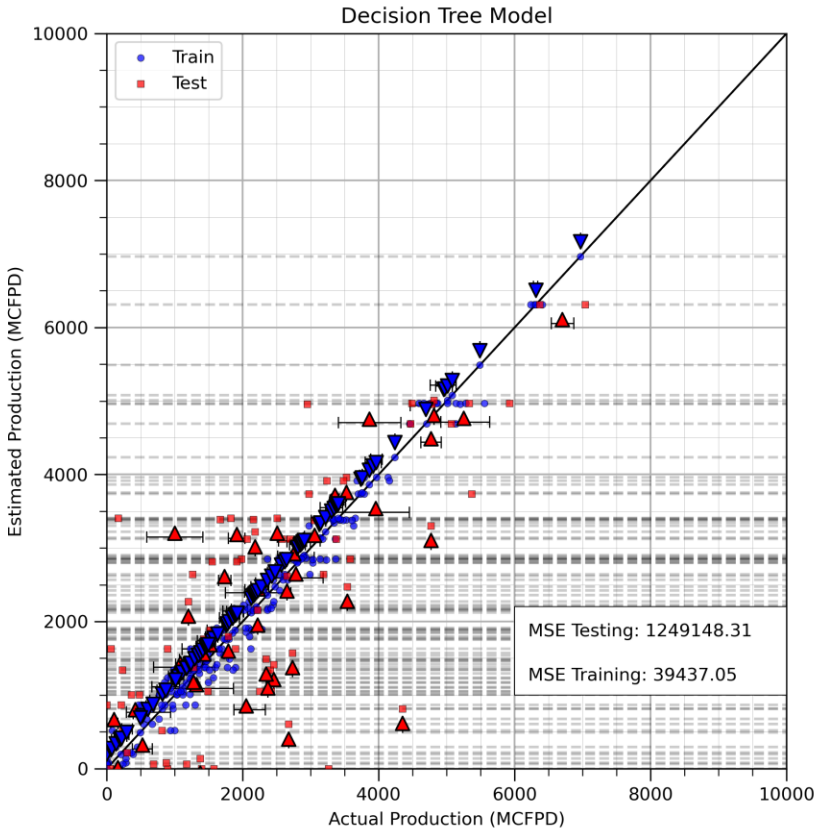


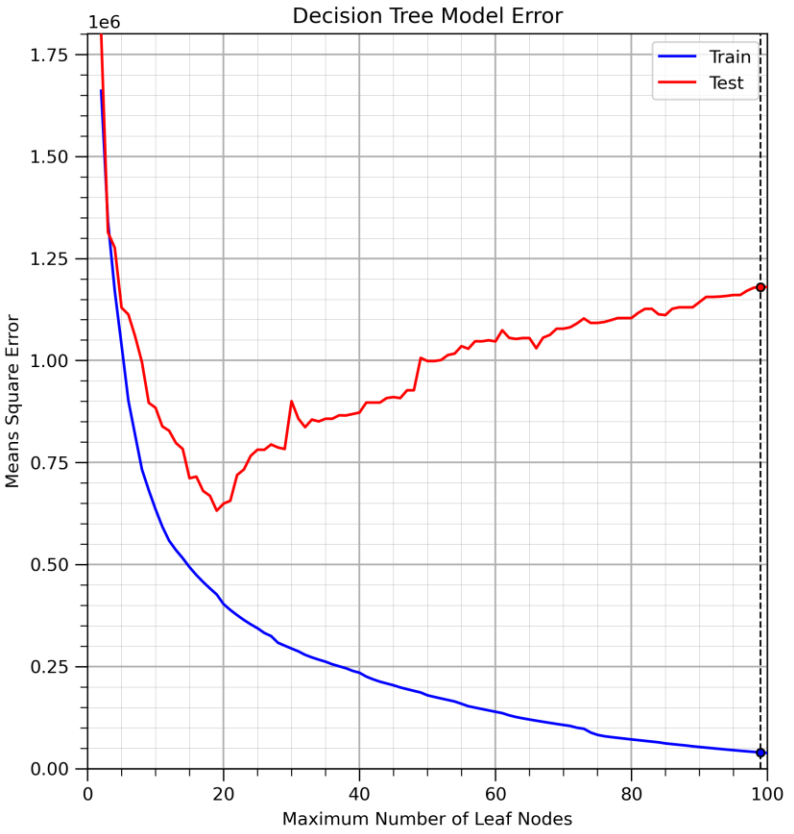
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 15 regions (leaf nodes) – tuned model

Decision Tree Training and Tuning







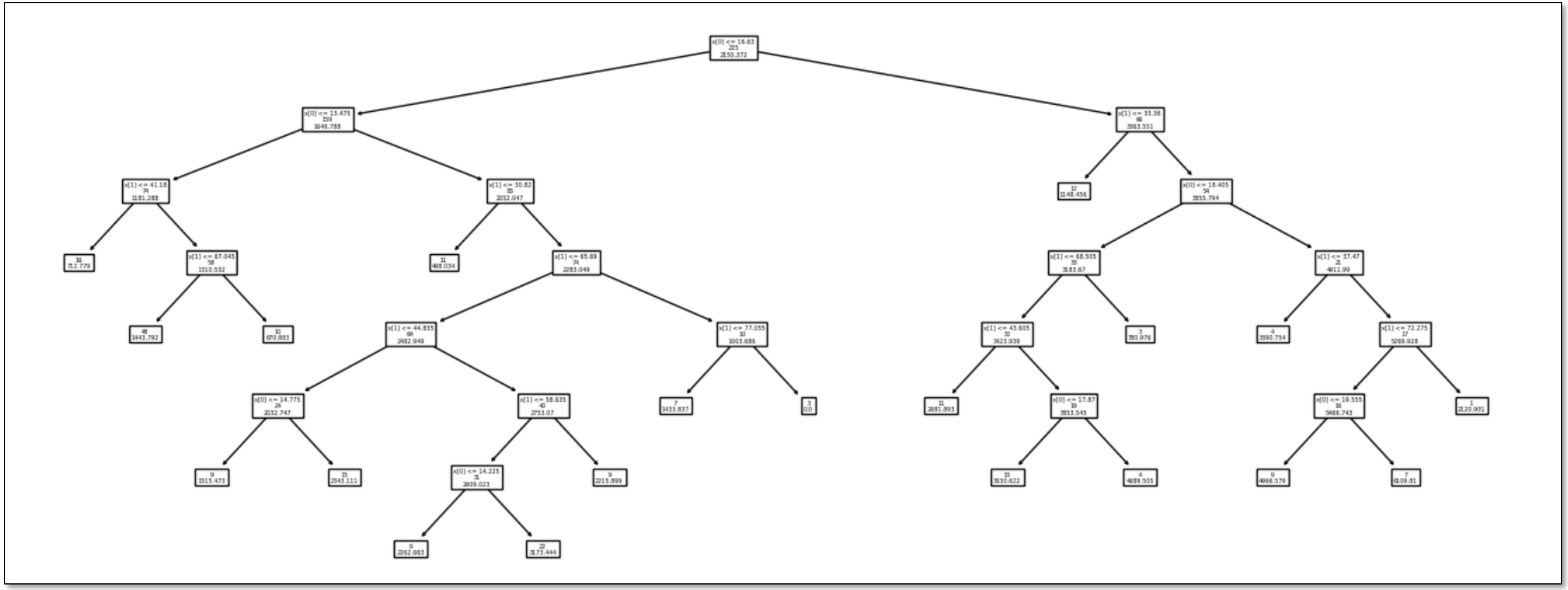
Interactive Decision Tree dashboard, file is Interactive_Decision_Tree.ipynb, also see Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter = 3 regions (leaf nodes) – very, very overfit model!

Decision Tree Training and Tuning

Now retrain the model with our tuned hyperparameter = 19 regions (leaf nodes)

- this is the model that we plot to predict production from porosity and brittleness.



Tuned decision tree model, from Decision Tree chapter of Applied Machine Learning e-book.

Decision Tree Hyperparameters

Decision Tree Hyperparameters

- **Number of regions** – very easy to understand, you know what the model will be

Other Hyperparameters,

- **Minimum reduction in RSS** – could stop early, e.g., a low reduction in RSS split could lead to a subsequent split with a larger reduction in RSS

- **Minimum number of training data in each region** – related to the concept of accuracy of the region mean prediction, i.e., we need at least n data for a reliable mean

- **Maximum number of levels** – forces symmetric trees, similar number of splits to get to each region

PGE 383 Subsurface Machine Learning

Lecture 14: Decision Tree

Lecture outline:

- **Decision Tree Tuning with K-fold Cross Validation**

Decision Trees Workflow

Training and Tuning a Decision Tree

1. Grow a very complicate true, overfit the data
2. Sequential removal of branches to access a set of simple to complicated decision trees
3. Evaluate testing error for each simple to complicate decision tree
4. Select hyperparameter that minimizes testing error, the tuned hyperparameters
5. Train the decision tree with the tuned hyperparameters with all data

Comments

- for K-fold cross validation train K trees on K-folds and aggregate testing error

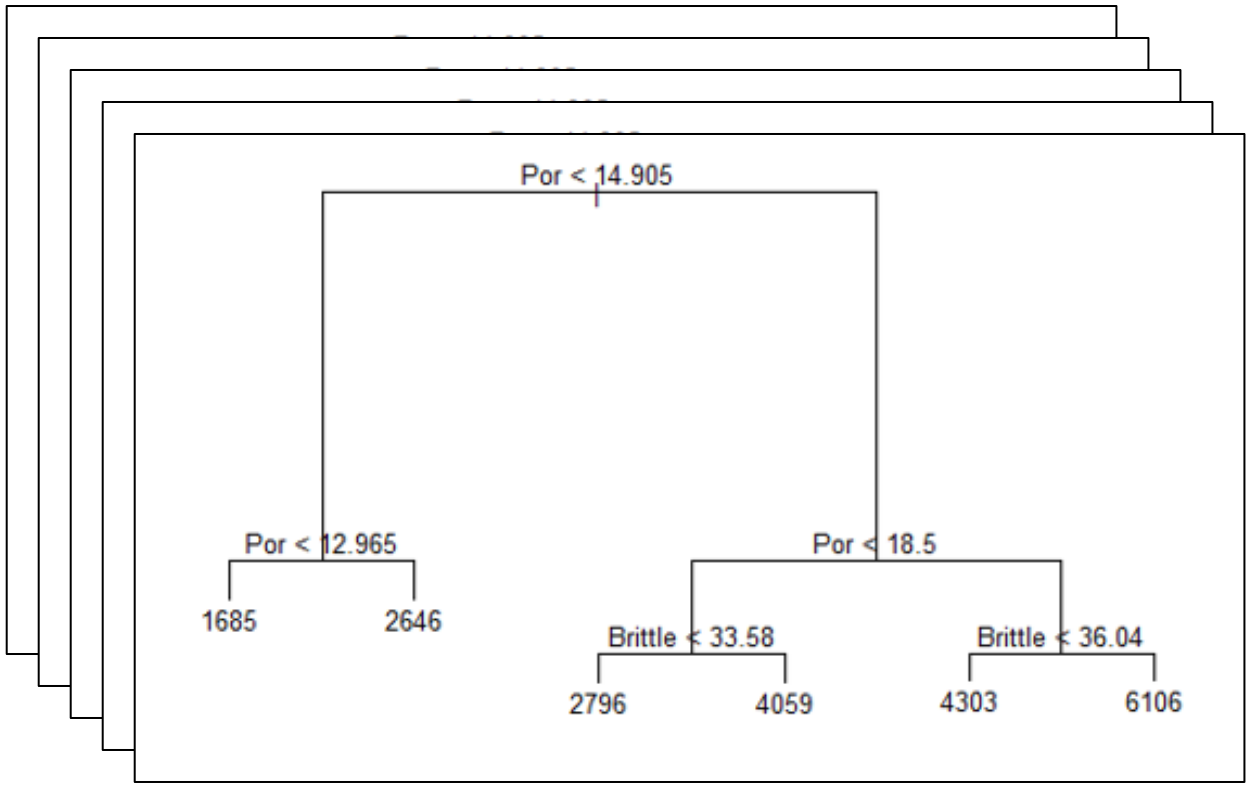
Decision Trees Workflow

Building a Regression Tree

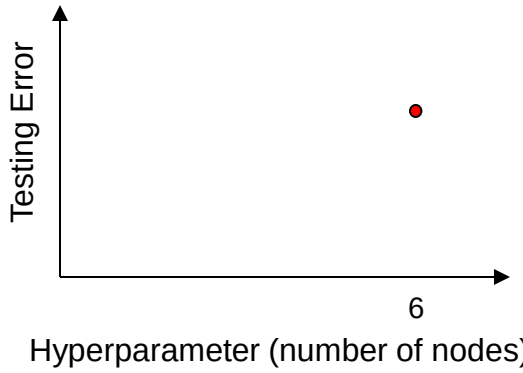
Over K folds.

- Build complicated decision tree
- Sequentially prune the tree
- Calculate error with testing data

Average K testing errors



k decision trees with hyperparameter max number of terminal nodes = 6.



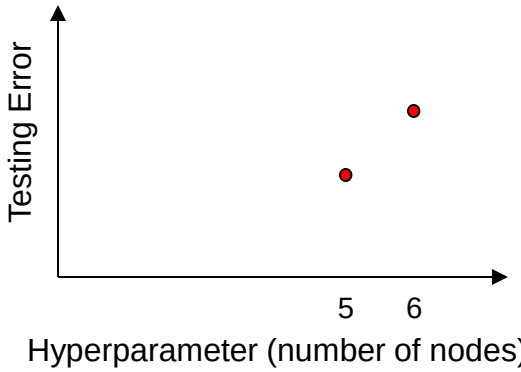
Decision Trees Workflow

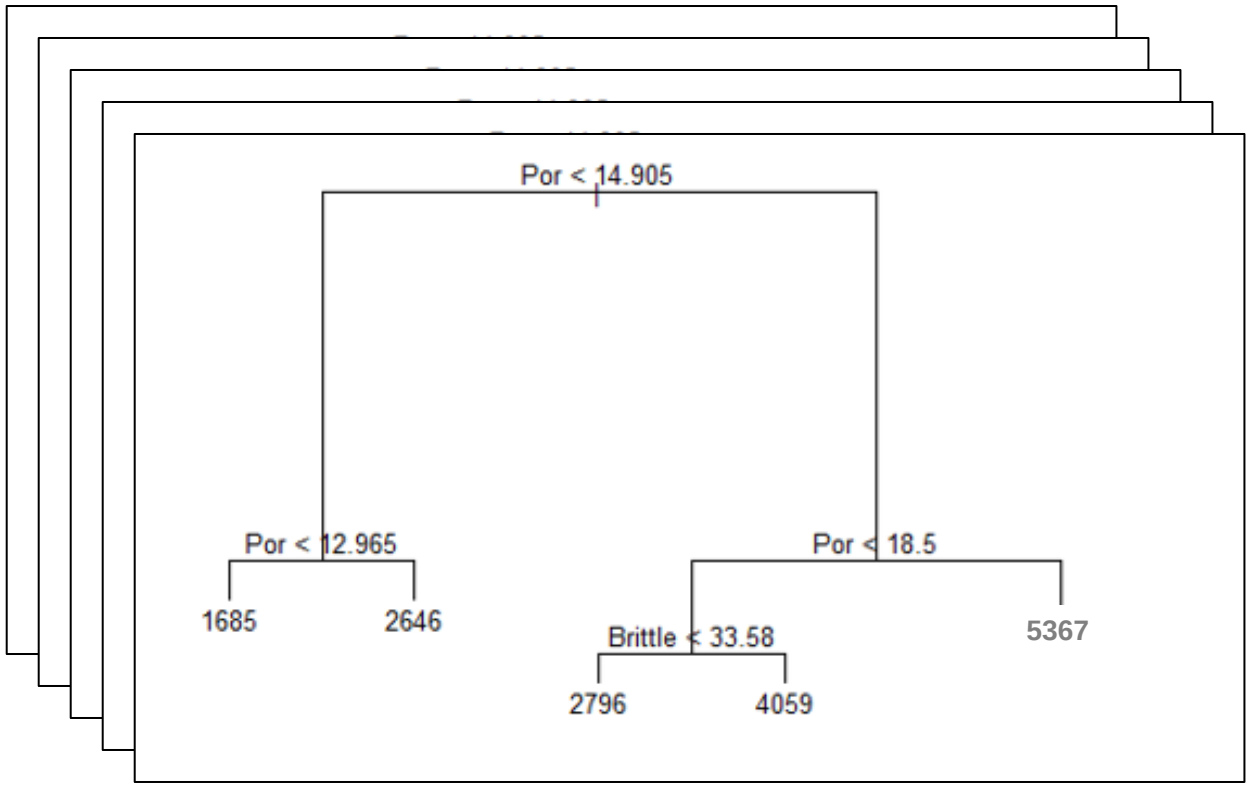
Building a Regression Tree

Over K folds.

- Build complicated decision tree
- Sequentially prune the tree
- Calculate error with testing data

Average K testing errors





k decision trees with hyperparameter max number of terminal nodes = 5.

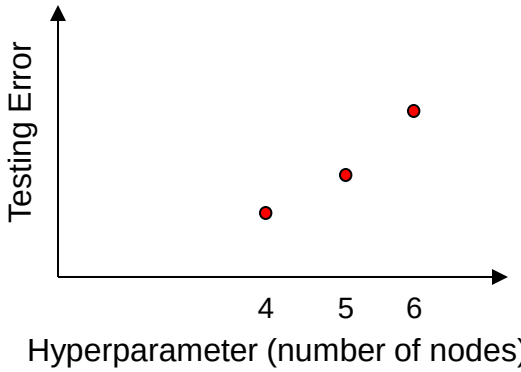
Decision Trees Workflow

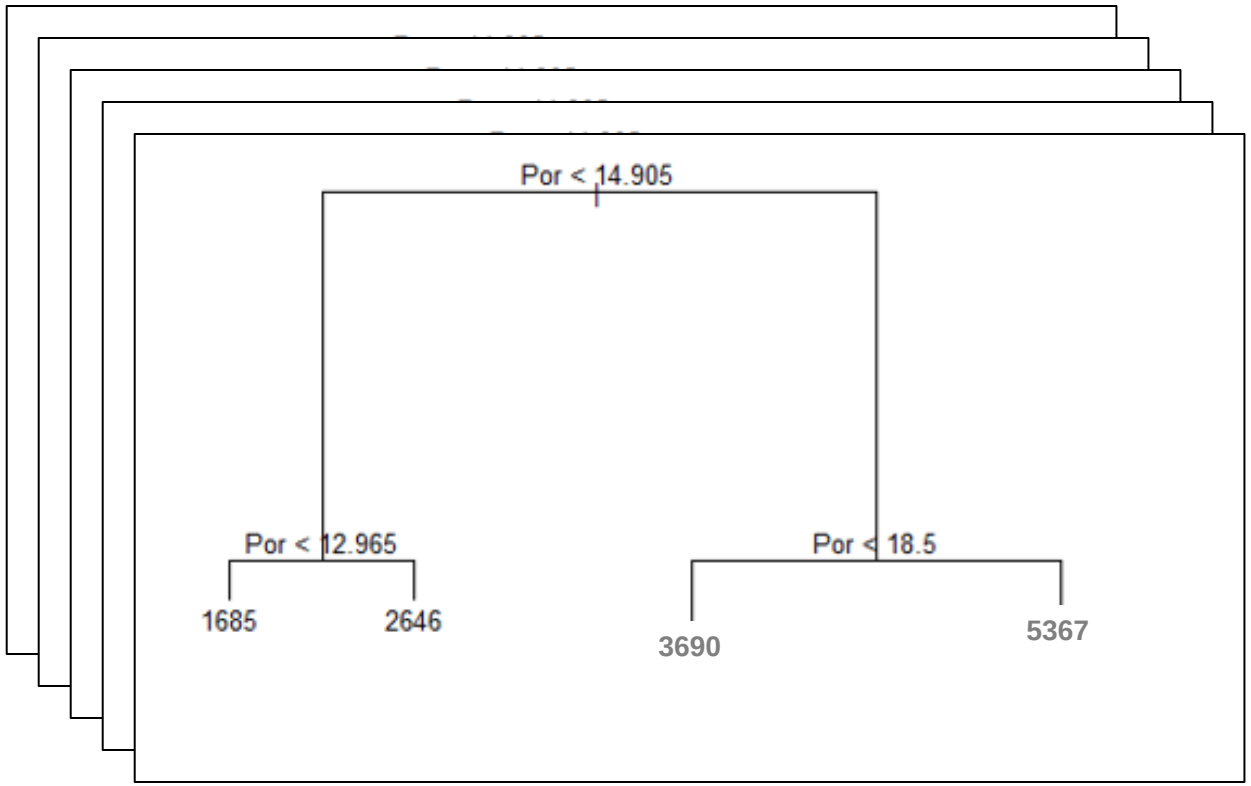
Building a Regression Tree

Over K folds.

- Build complicated decision tree
- Sequentially prune the tree
- Calculate error with testing data

Average K testing errors





k decision trees with hyperparameter max number of terminal nodes = 4.

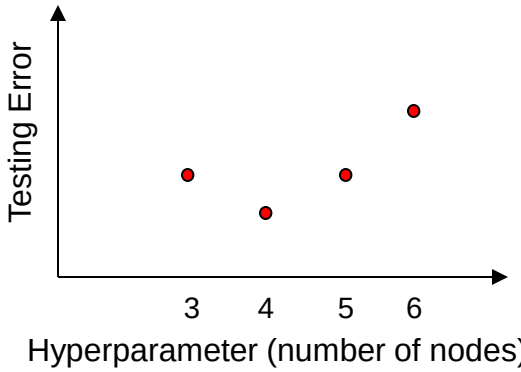
Decision Trees Workflow

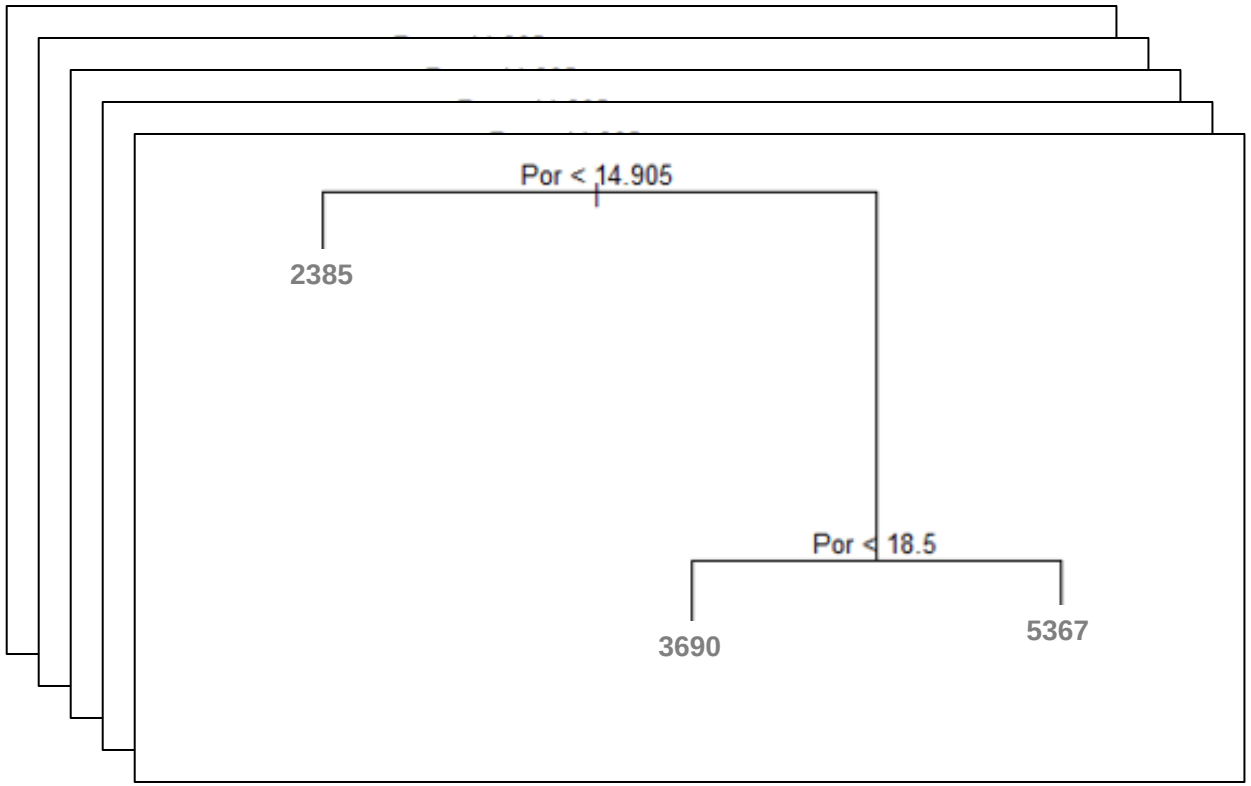
Building a Regression Tree

Over K folds.

- Build complicated decision tree
- Sequentially prune the tree
- Calculate error with testing data

Average K testing errors





k decision trees with hyperparameter max number of terminal nodes = 3.

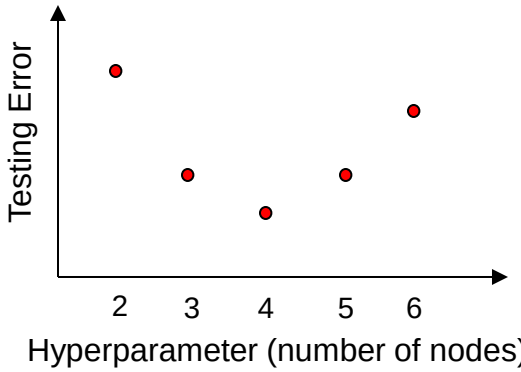
Decision Trees Workflow

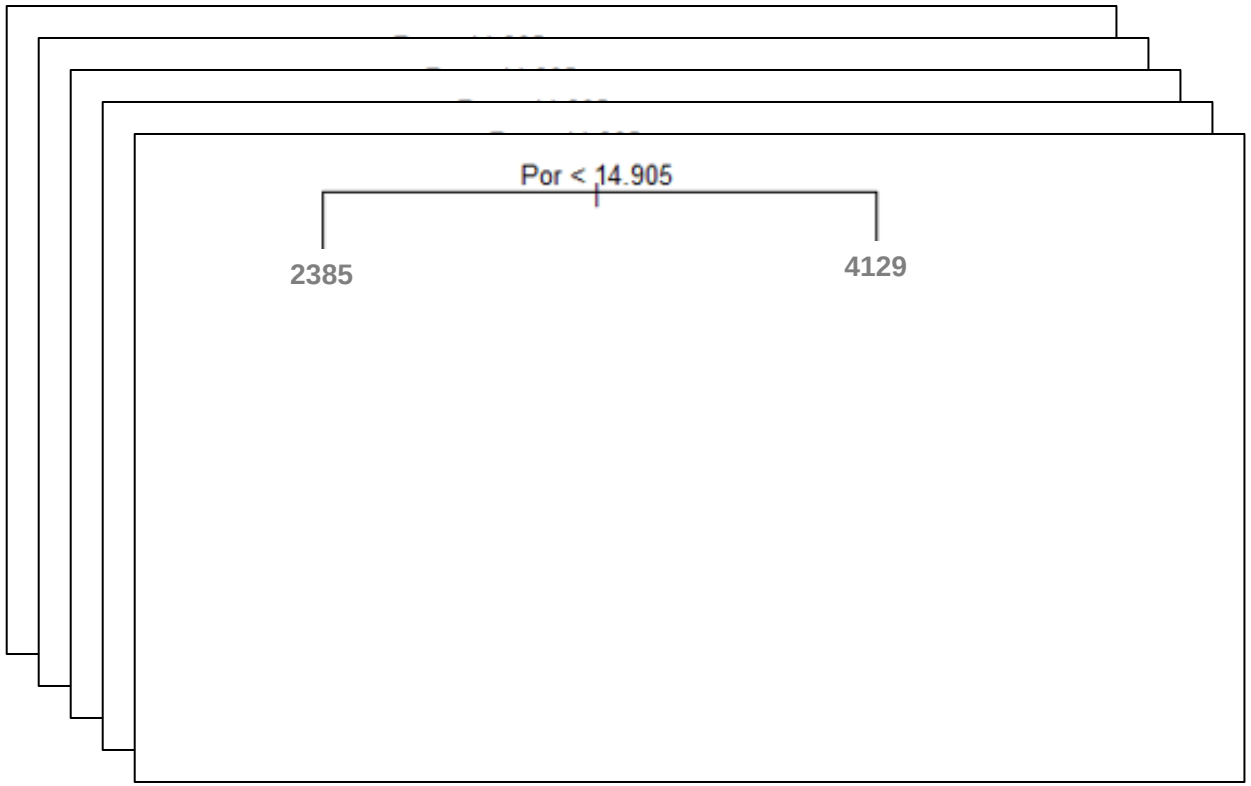
Building a Regression Tree

Over K folds.

- Build complicated decision tree
- Sequentially prune the tree
- Calculate error with testing data

Average K testing errors





k decision trees with hyperparameter max number of terminal nodes = 2.

Decision Trees Workflow

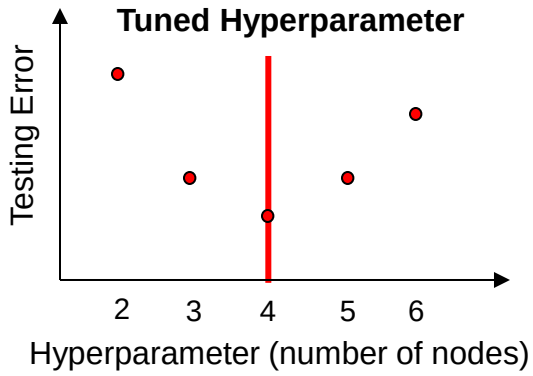
Building a Regression Tree

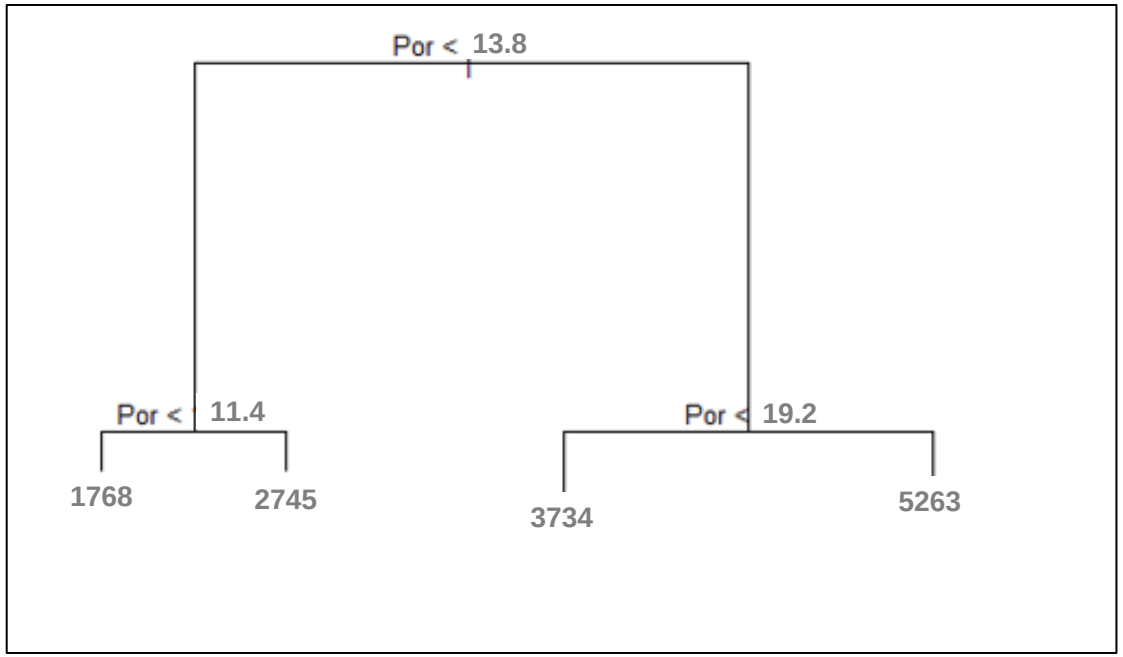
Over K folds.

- Build complicated decision tree
- Sequentially prune the tree
- Calculate error with testing data

Average K testing errors

Select hyperparameter that minimized testing error and retrain on all data.





Tuned decision tree, retrained with all the data for a final model.

PGE 383 Subsurface Machine Learning

Lecture 14: Decision Tree

Lecture outline:

- **Decision Tree Shapley Values**

Recall Shapley Values

We need to take a single model, $f(x_1, x_2, x_3, x_4)$, and make an estimate for all possible combinations feature subsets!

Note: the **naïve approach** is to train the full combinatorial of models. We don't want to do that if our goal is feature importance to diagnose our model, f. We want to support model explain-ability.

The variety of approaches are similar to imputation methods:

$$f(x_1, x_2, x_3) = f(x_1, x_2, x_3, x_4 = E[x_4])$$

$$f(x_1, x_2, x_3) = f(x_1, x_2, x_3, x_4 = P50_{x_4})$$

There is a unique method with tree-based models:

- Remove x_4 branches, and then the model does not use x_4 to make prediction

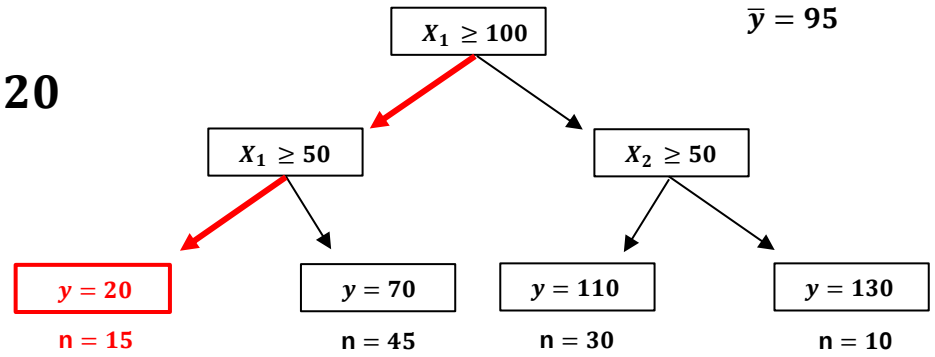
Recall Shapley Values

How do we remove features from a decision tree for Shapley value marginal contributions?

Here's a simple tree example to demonstrate for the case of $X_1 = 25$, $X_2 = 60$.

1. for cases where we don't encounter the removed feature, use use the tree as usual.

$$f(X_1 = 25) = 20$$



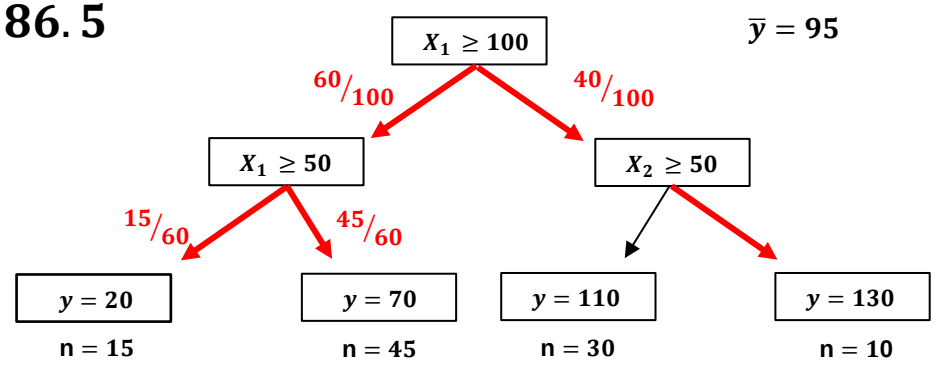
Marginal contribution, X_1:

$$f(X_1 = 25) - \bar{y} = 20 - 95 = -75$$

2. for cases where we encounter the removed feature, we weight (by number of training data) over both paths.

$$f(X_2 = 60) = 86.5$$

$$^{60}/_{100}\left[^{15}/_{60} \times 20 + ^{45}/_{60} \times 70\right] + ^{40}/_{100}[130] = 86.5$$



Marginal contribution, X_2:

$$f(X_2 = 60) - \bar{y} = 86.5 - 95 = 8.5$$

PGE 383 Subsurface Machine Learning

Lecture 14: Decision Tree

Lecture outline:

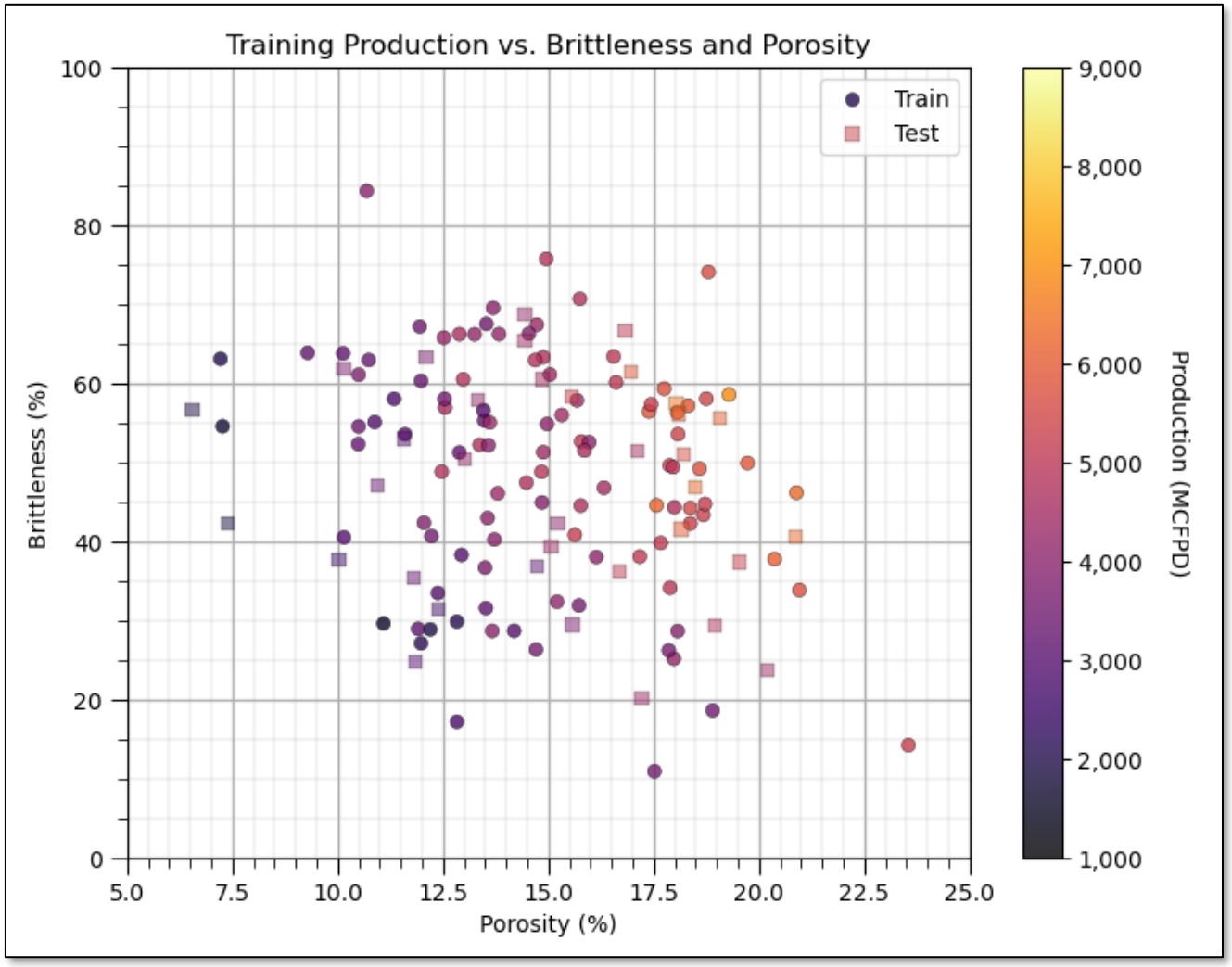
- **Decision Tree Example**

Decision Tree Example

Unconventional Multivariate Data

- both porosity and brittleness have interesting relationships with production

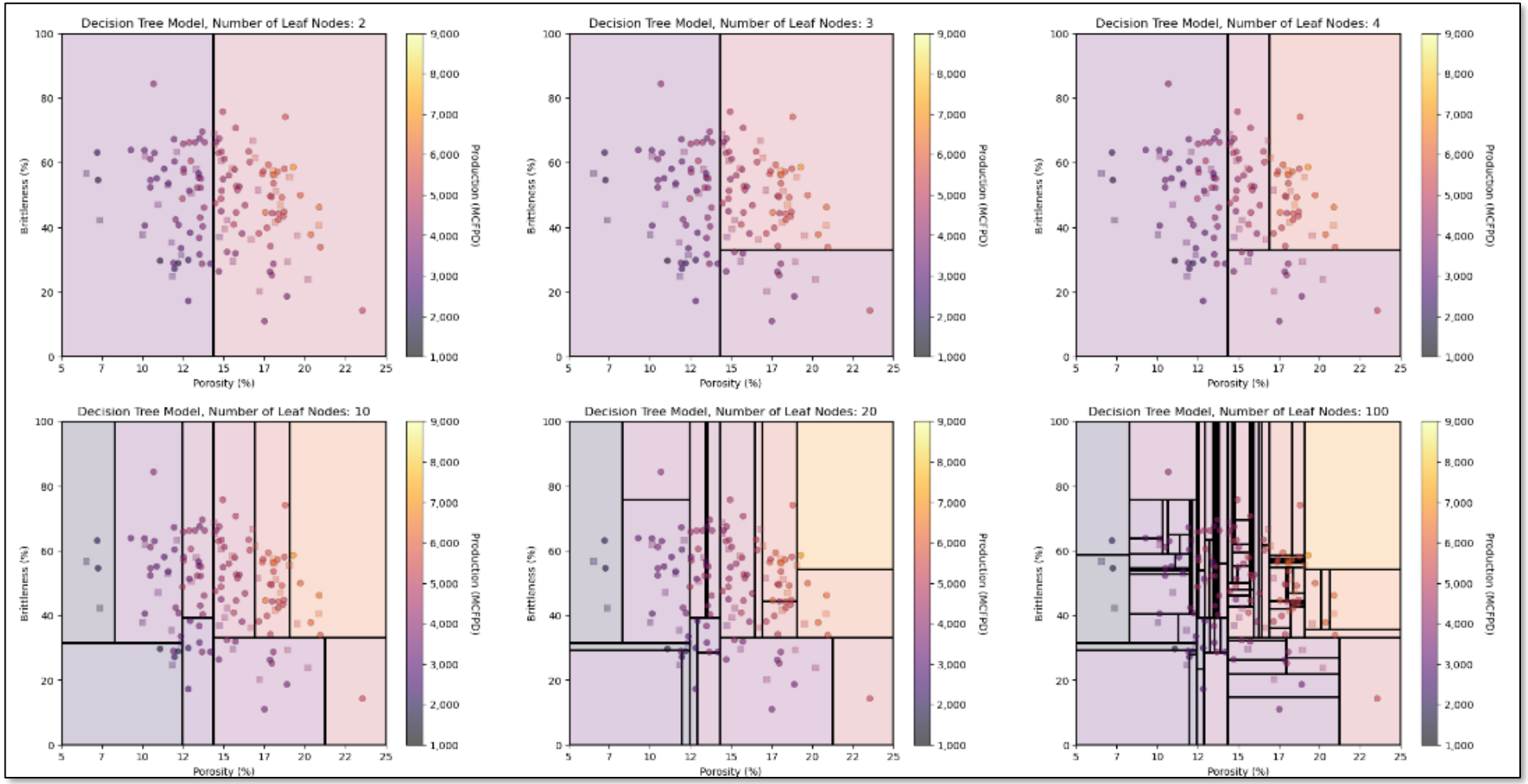
- for ease of visualization, we work with 2 predictor features.



Dataset, from Decision Tree chapter of Applied Machine Learning e-book.

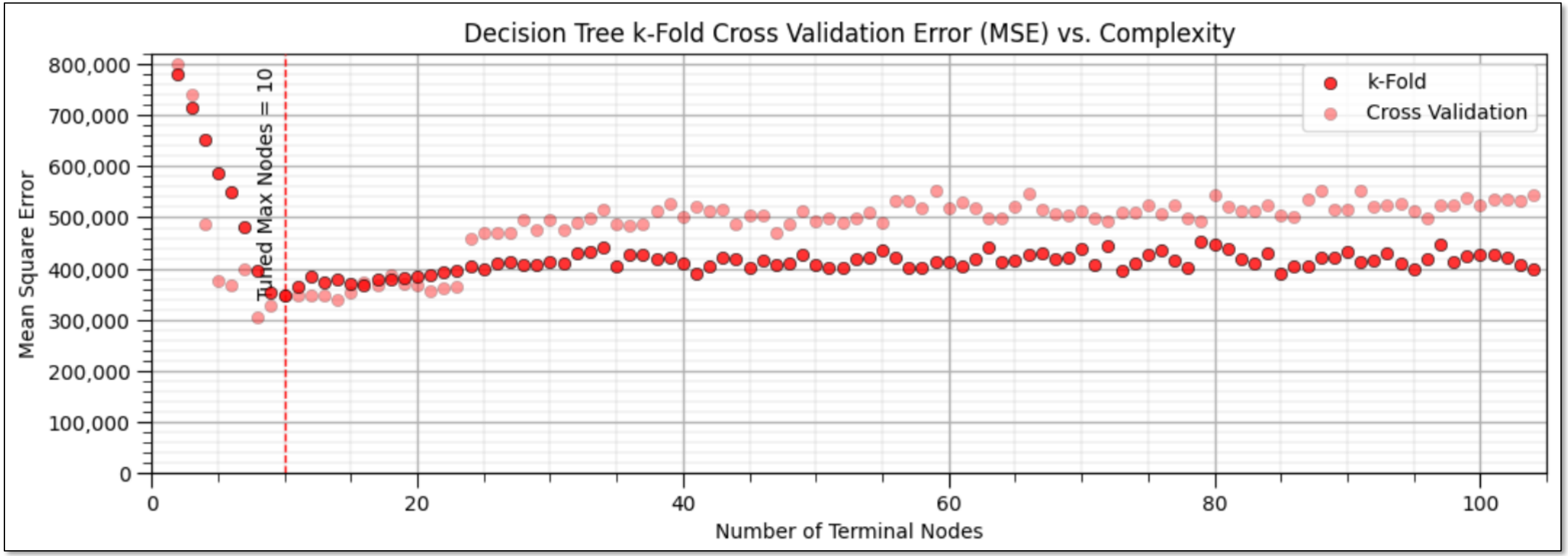
Decision Tree Example

Simple to Complicated Decision Trees



Dataset and decision trees, simple to complicated with number of region hyperparameter, from Decision Tree chapter of Applied Machine Learning e-book.

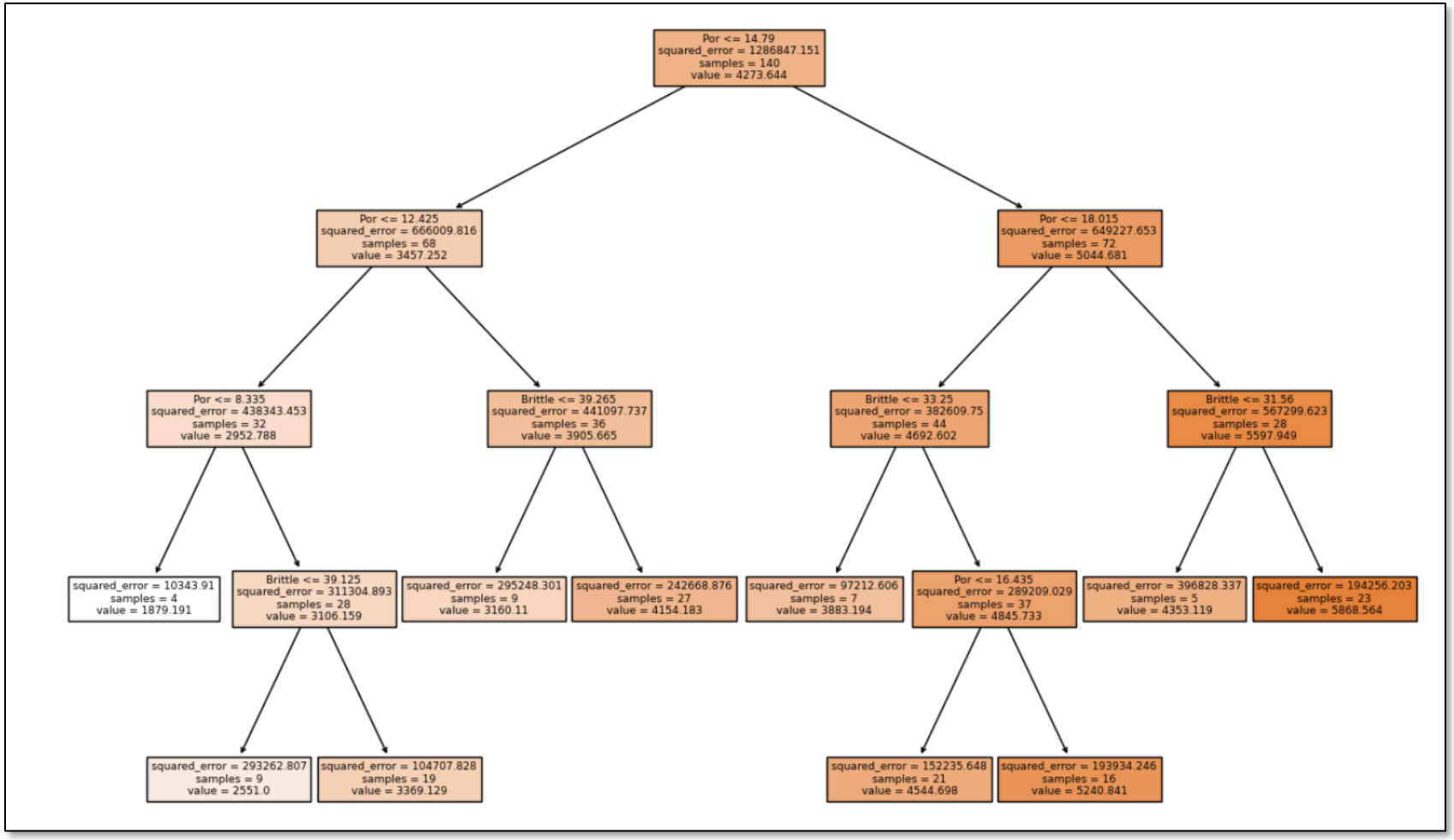
Hyperparameter Tuning



Decision tree hyperparameter tuning, simple to complicated with number of region hyperparameter, from Decision Tree chapter of Applied Machine Learning e-book.

Hyperparameter Tuning

- tuned decision tree



Tuned decision tree model, from Decision Tree chapter of Applied Machine Learning e-book.

PGE 383 Subsurface Machine Learning

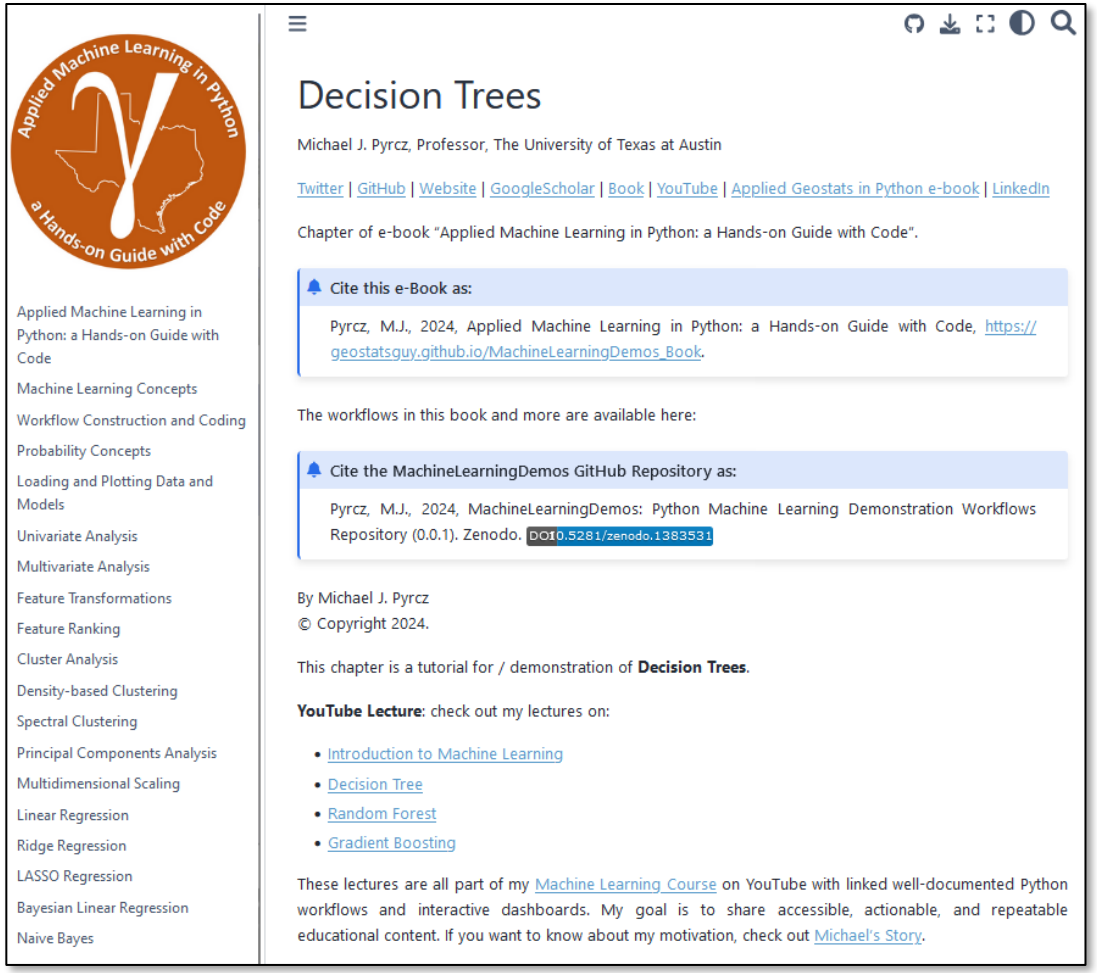
Lecture 14: Decision Tree

Lecture outline:

- **Decision Tree Hands-on**

Decision Tree Demonstration in Python

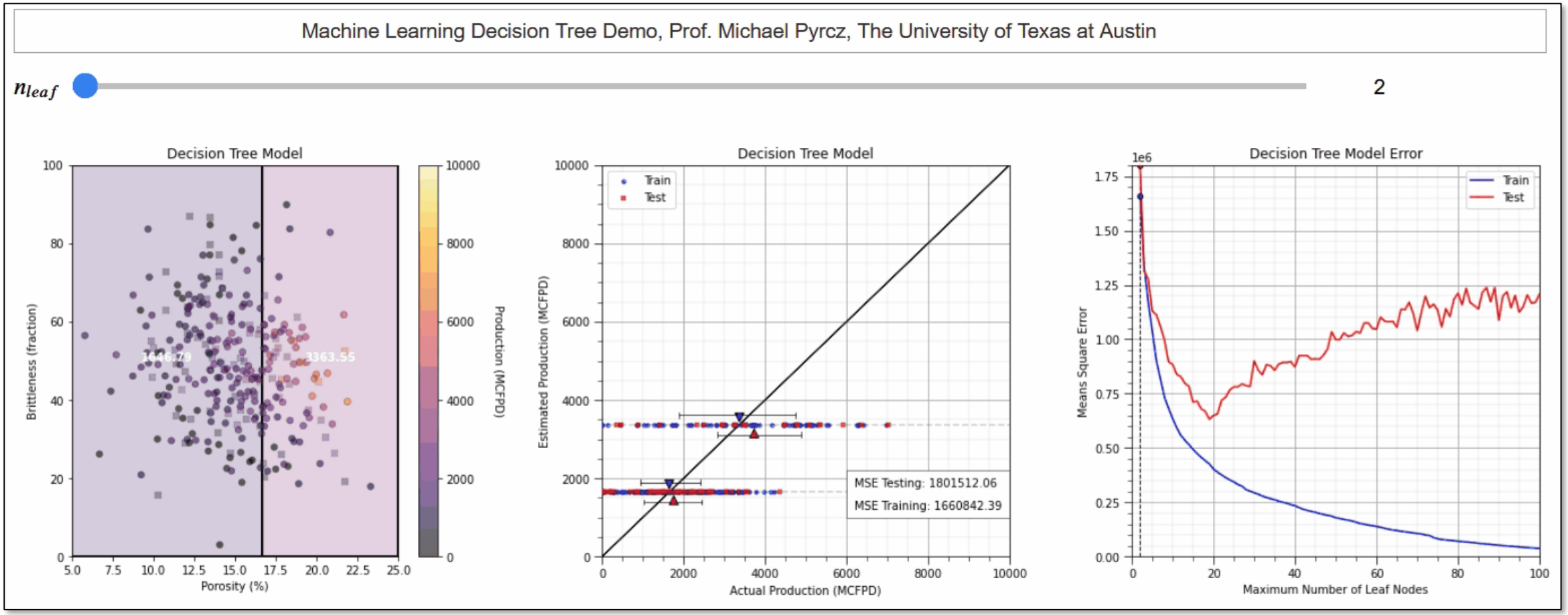
Demonstration of decision tree with a well-documented workflow.



Decision tree chapter of Applied Machine Learning in Python e-book.

Decision Tree Demonstration in Python

Demonstration of decision tree with a well-documented workflow.



Interactive decision tree Python dashboard, file is Interactive_Decision_Tree.ipynb.

PGE 383 Subsurface Machine Learning

Lecture 14: Decision Tree

Lecture outline:

- **Decision Tree**
- **Predictive Machine Learning Concepts**
- **Decision Tree Training**
- **Decision Tree Tuning**
- **Decision Tree Tuning with K-fold Cross Validation**
- **Decision Tree Shapley Values**
- **Decision Tree Example**
- **Decision Tree Hands-on**