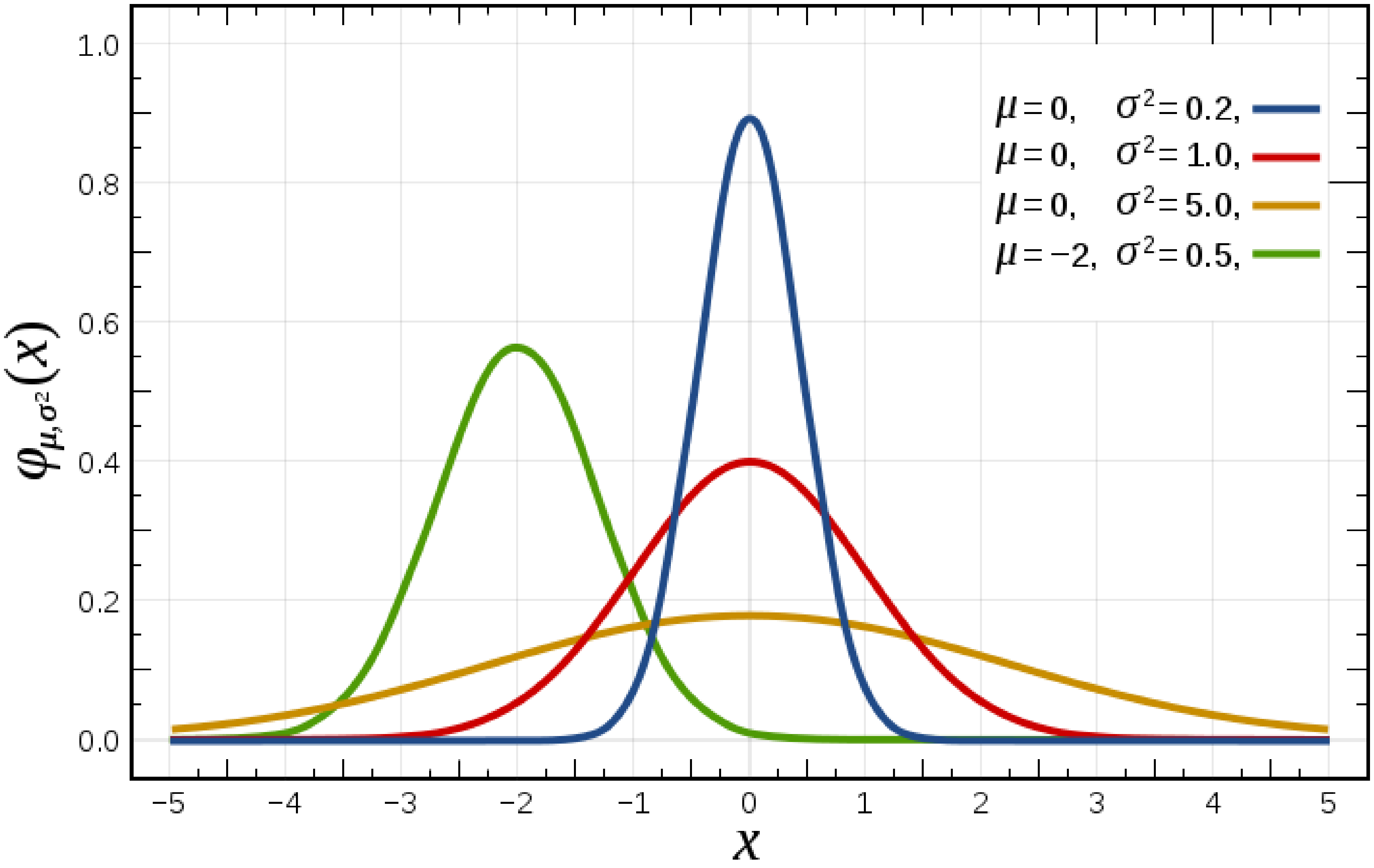
Machine Learning / Tree based methods

Decision Trees

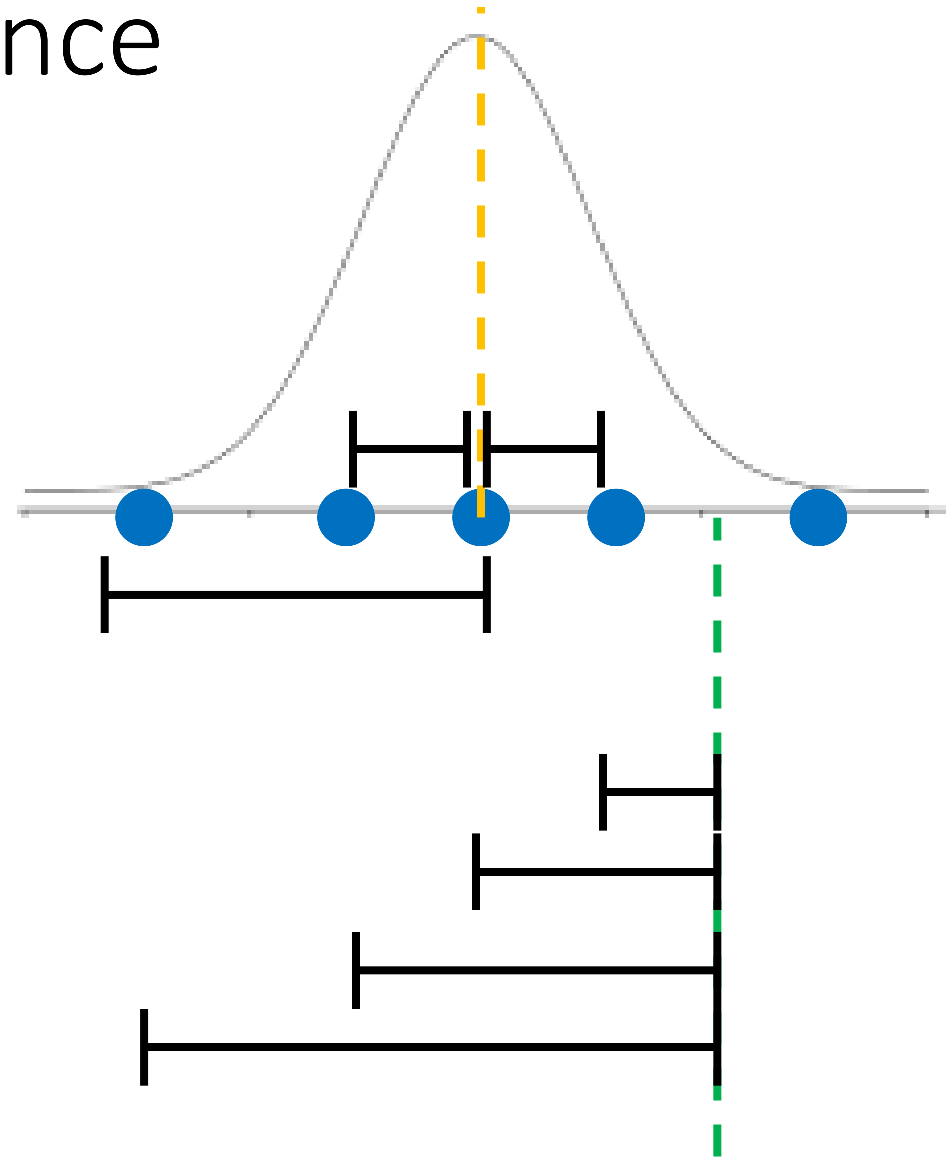
Regression

Variance



$\mu=0,\quad \sigma^2=0.2,$
$\mu=0,\quad \sigma^2=1.0,$
$\mu=0,\quad \sigma^2=5.0,$
$\mu=-2,\quad \sigma^2=0.5,$
1.0
0.8
0.6
0.4
0.2
0.0
$\varphi_{\mu,\sigma^2}(x)$
−5
−4
−3
−2
−1
0
1
2
3
4
5
x

Variance

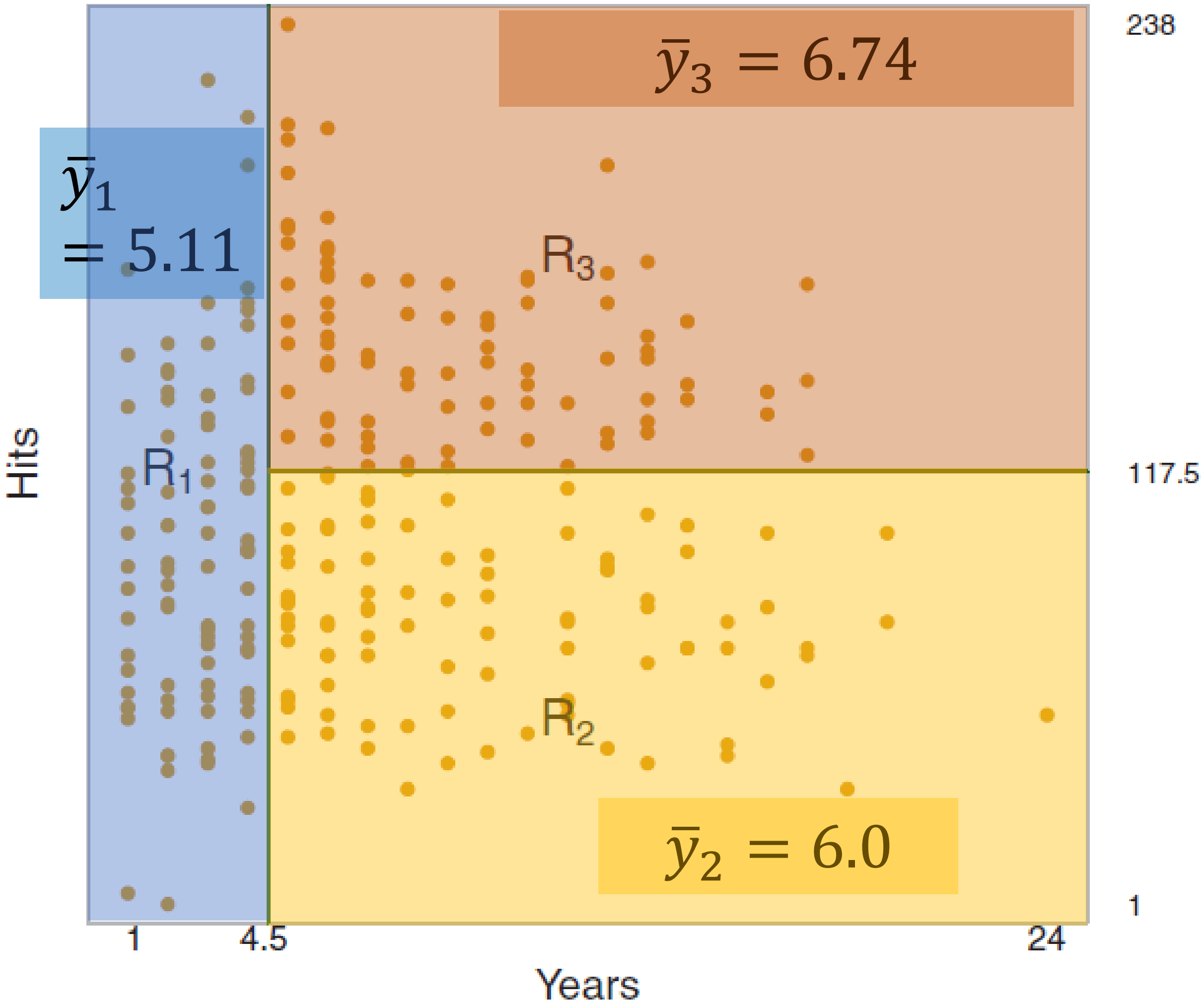


$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{n}(x_i - c)^2$$

$c = \bar{x}$ produce the lowest value of σ^2

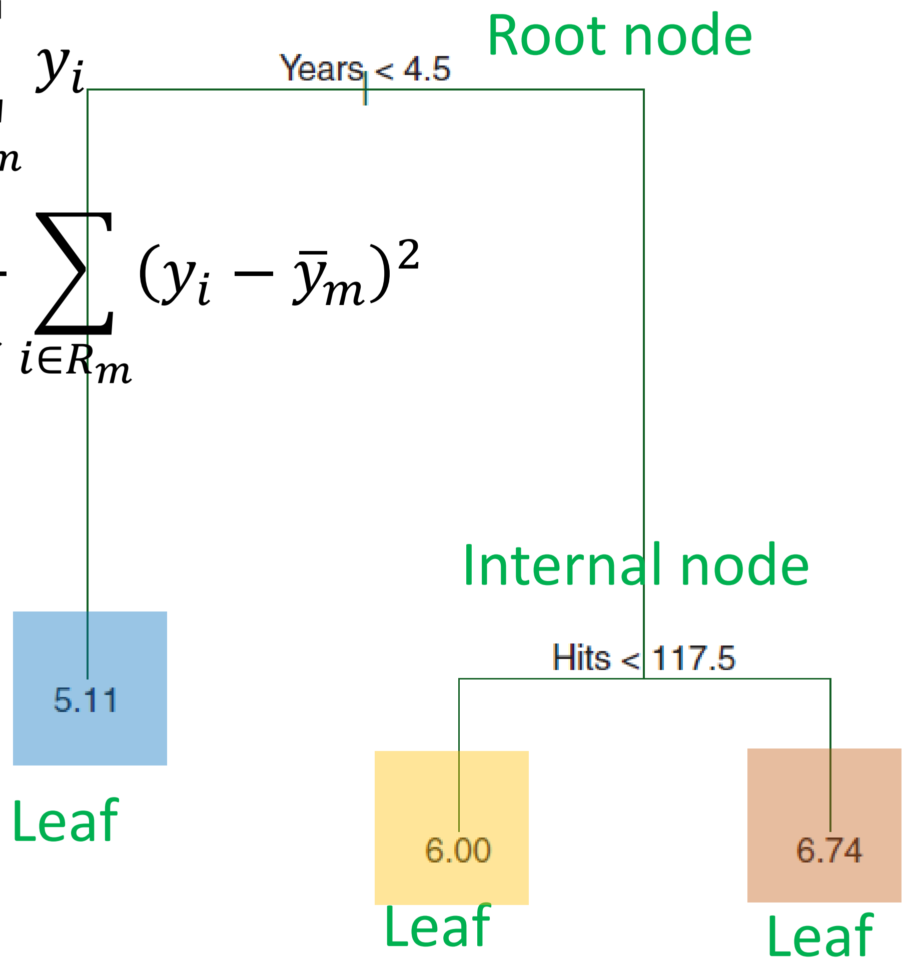
Domain partitioning

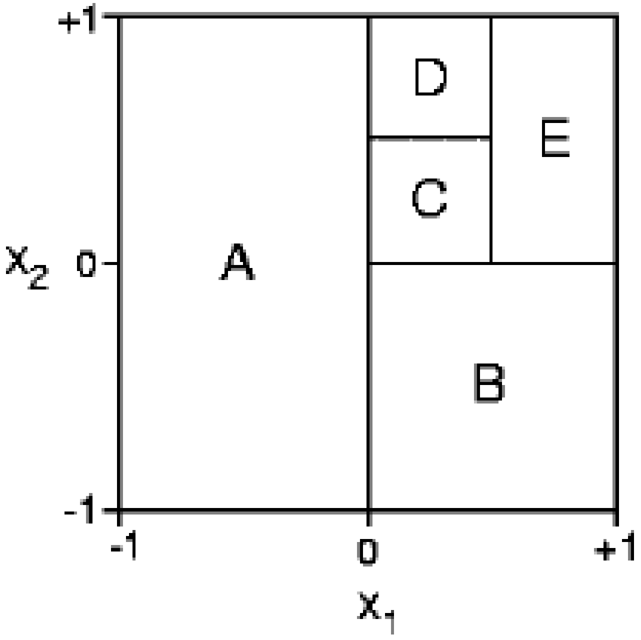
$$f(x_1, x_2) = y$$

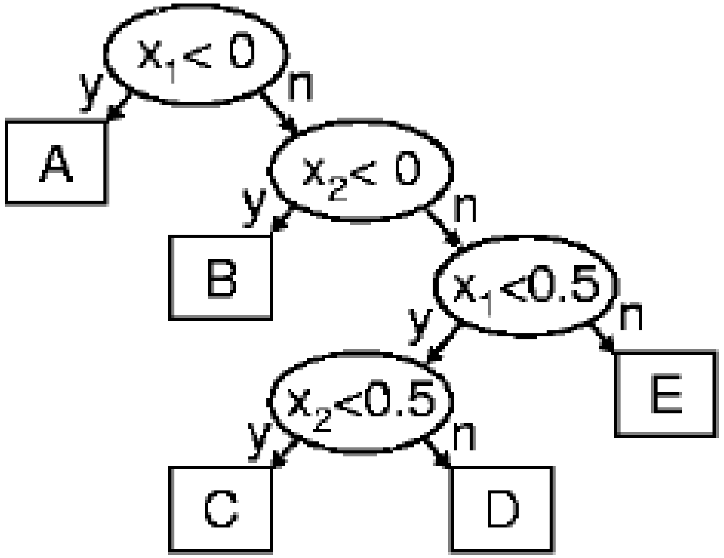


$$\bar{y}_m = \frac{1}{N_m} \sum_{i \in R_m} y_i$$

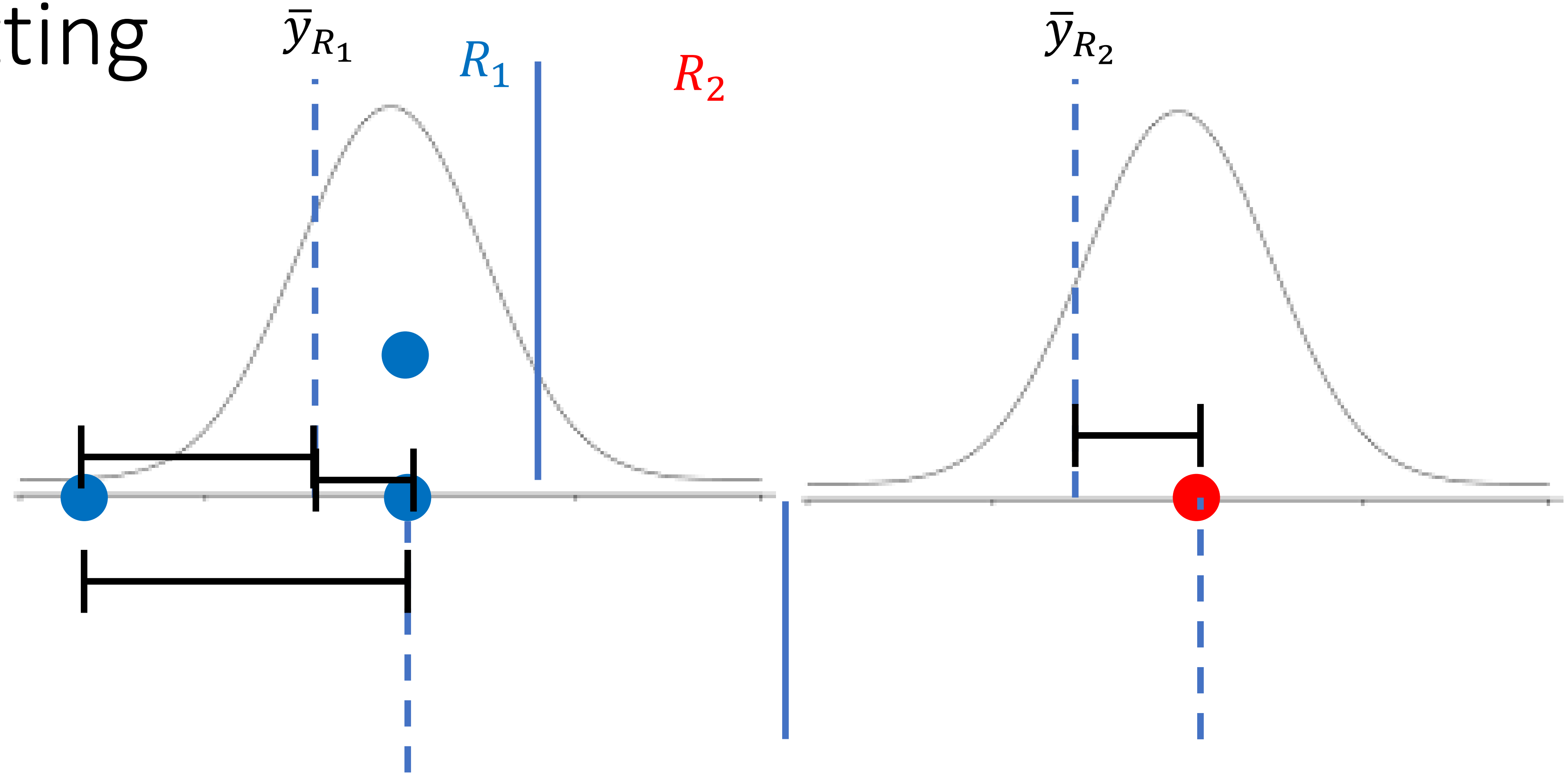
$$H(X_m) = \frac{1}{N_m} \sum_{i \in R_m} (y_i - \bar{y}_m)^2$$




+1
0
-1
x2
A
B
C
D
E
-1
0
+1
x1


x1< 0
y
n
A
x2< 0
y
n
B
x1<0.5
y
n
E
x2<0.5
y
n
C
D

Fitting

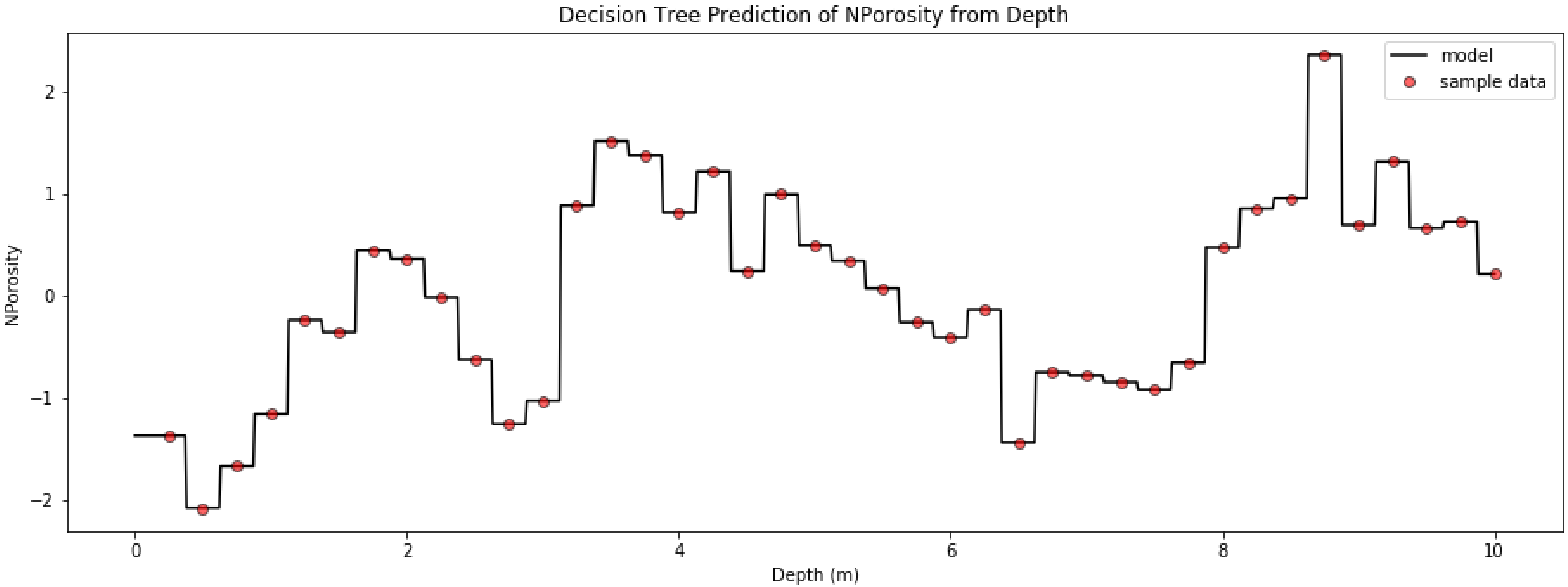


Fit

Step 1,find j and s that Minimize:

$$\sum_{i:\boldsymbol{x}_i \in R_1(j,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{i:\boldsymbol{x}_i \in R_2(j,s)} (y_i - \bar{y}_{R_2})^2$$

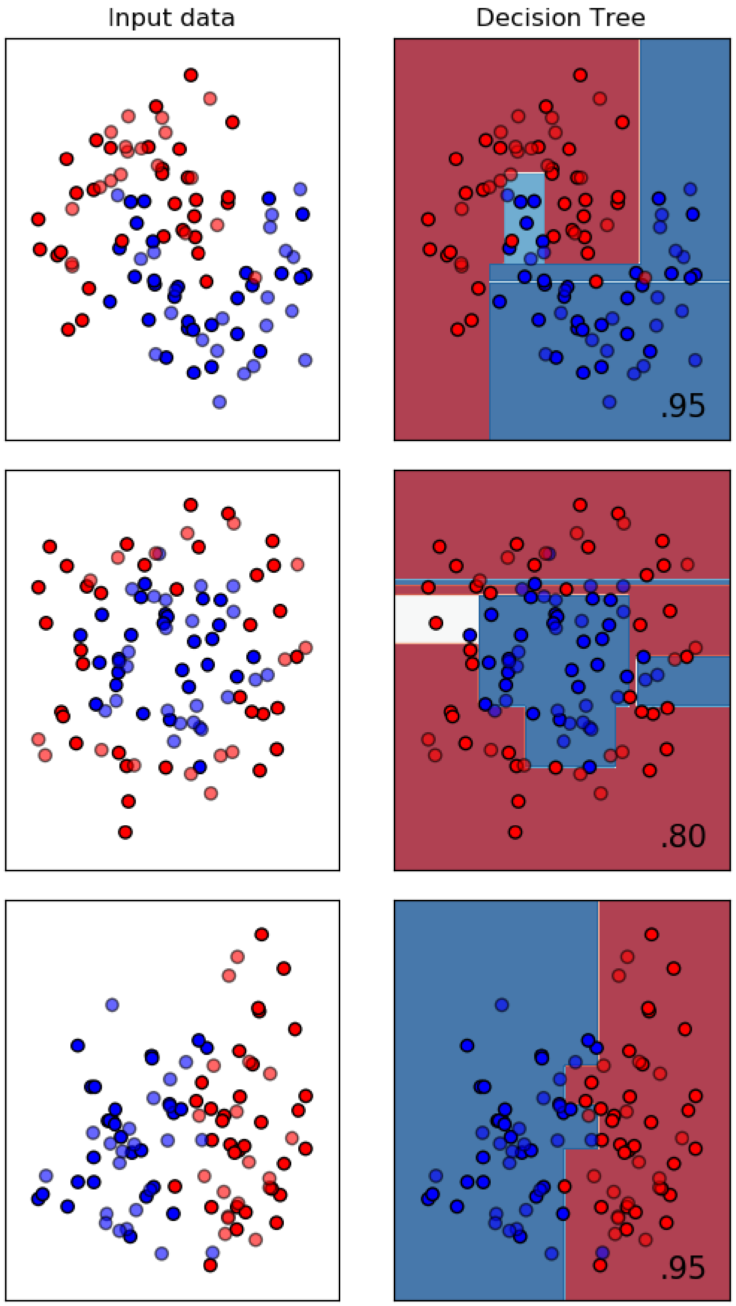
$j = 1, \ldots, n$
$s =$ cuttoff



https://github.com/GeostatsGuy/PythonNumericalDemos/blob/master/SubsurfaceDataAnalytics_PolygonalRegression.ipynb

Decision Trees

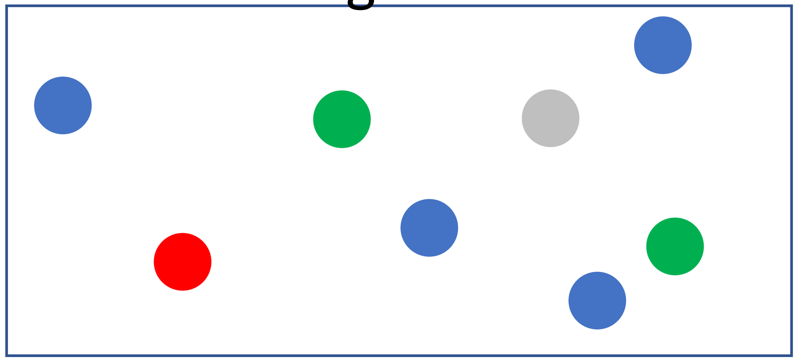
Classification



Proportions

$$p_{mk} := \frac{1}{N_m} \sum_{x_i \in R_m} \mathbb{I}(y_i = k)$$

Region m



$$N = 7$$

$$p_{m1} = \frac{1}{7}$$

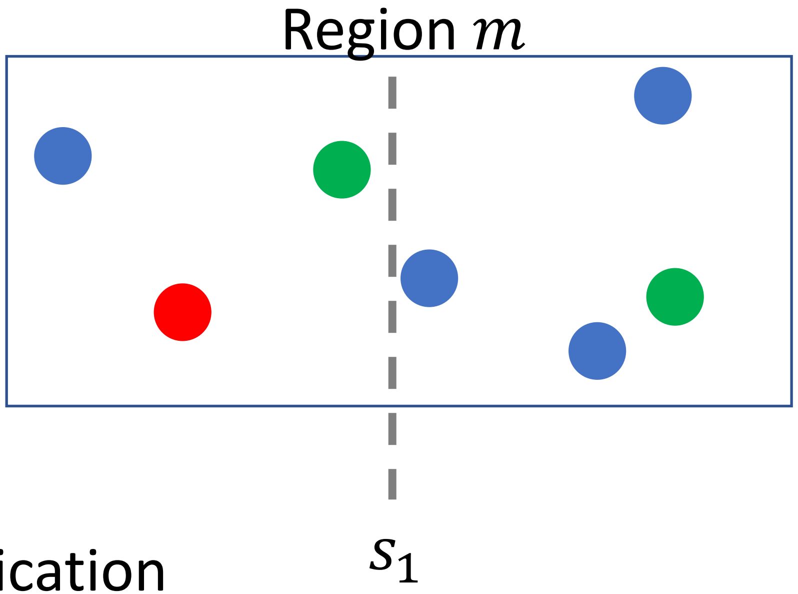
$$p_{m2} = \frac{2}{7}$$

$$p_{m3} = \frac{4}{7}$$

https://scikit-learn.org/stable/modules/tree.html#classification-criteria

p. 311-313, Hastie, 2014

Misclassification measure



$$H(X_m) = 1 - \max_k(p_{mk})$$

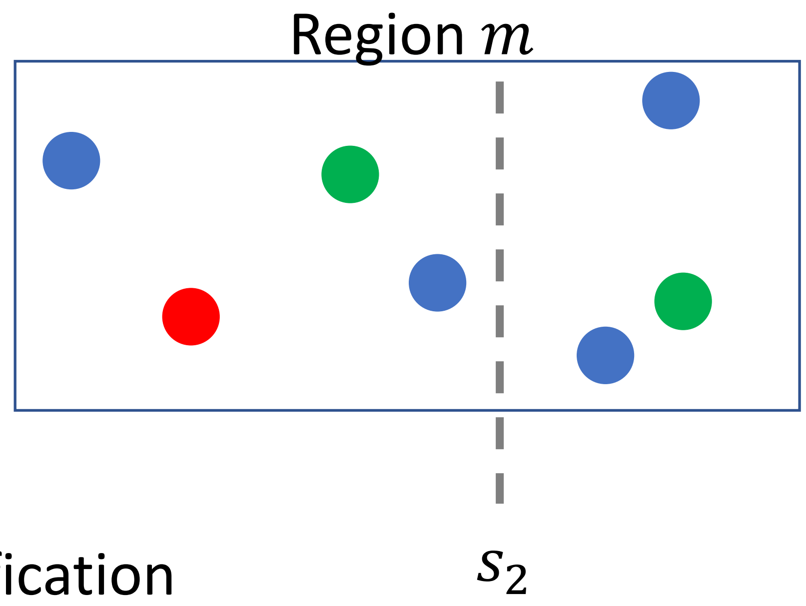
Misclassification

$$\frac{2}{3} + \frac{1}{4} = \frac{11}{12}$$

https://scikit-learn.org/stable/modules/tree.html#classification-criteria

p. 311-313, Hastie, 2014

Misclassification measure



$$H(X_m) = 1 - \max_k(p_{mk})$$

Misclassification

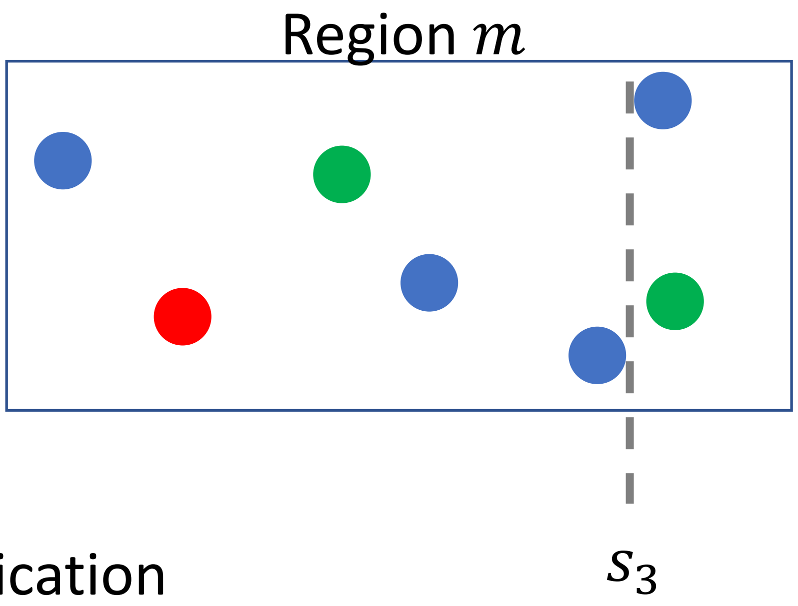
$$\frac{2}{3} + \frac{1}{4} = \frac{11}{12}$$

$$\frac{1}{2} + \frac{1}{3} = \frac{5}{6}$$

https://scikit-learn.org/stable/modules/tree.html#classification-criteria

p. 311-313, Hastie, 2014

Misclassification measure



$$H(X_m) = 1 - \max_k(p_{mk})$$

Misclassification

$$\frac{2}{3} + \frac{1}{4} = \frac{11}{12}$$

$$\frac{1}{2} + \frac{1}{3} = \frac{5}{6}$$

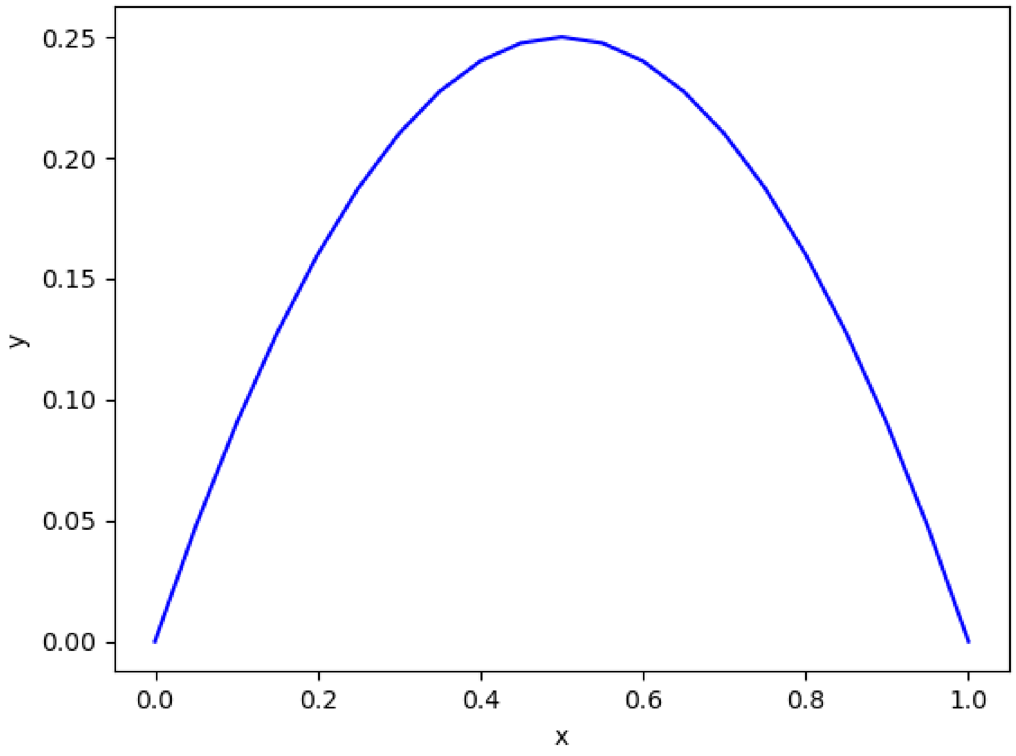
https://scikit-learn.org/stable/modules/tree.html#classification-criteria

p. 311-313, Hastie, 2014

Impurity functions. Gini Index

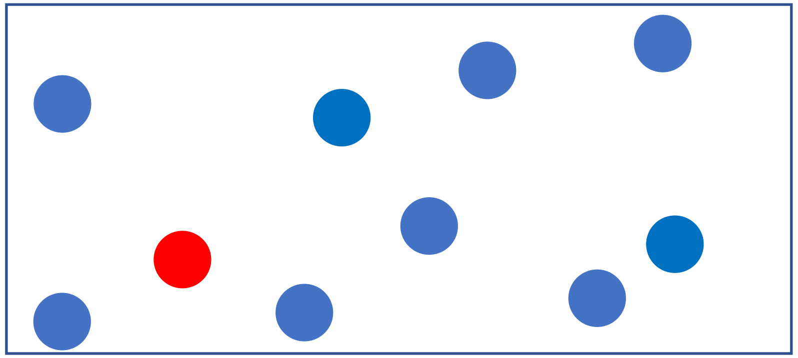
$$H(X_m) = \sum_k p_{mk}(1 - p_{mk})$$

- Find the maximum of $y = x(1 - x)$
- Plot y

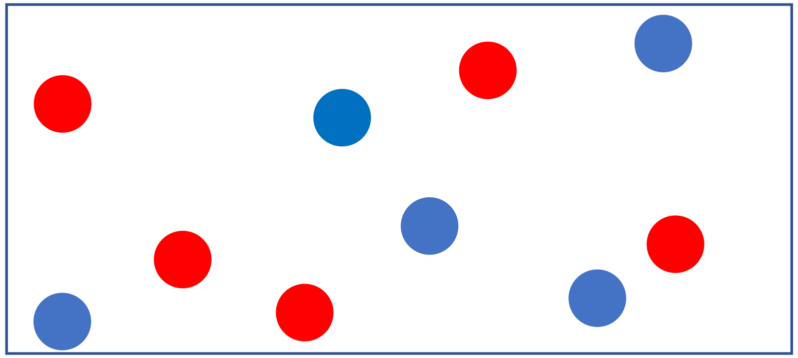


If only 2 classes, consider the following 2 cases

a) $\boldsymbol{p_1} \gg \boldsymbol{p_2}$

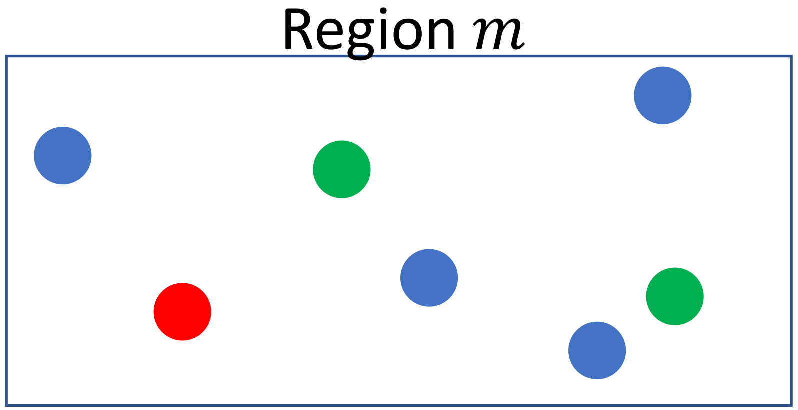


b) $\boldsymbol{p_1} \approx \boldsymbol{p_2}$



https://scikit-learn.org/stable/modules/tree.html#classification-criteria p. 311-313, Hastie, 2014

Impurity functions



$$p_{mk} := \frac{1}{N} \sum_{x_i \in R_m} \mathbb{I}(y_i = k)$$

$$H(X_m) = 1 - \max_k(p_{mk})$$ Misclassification

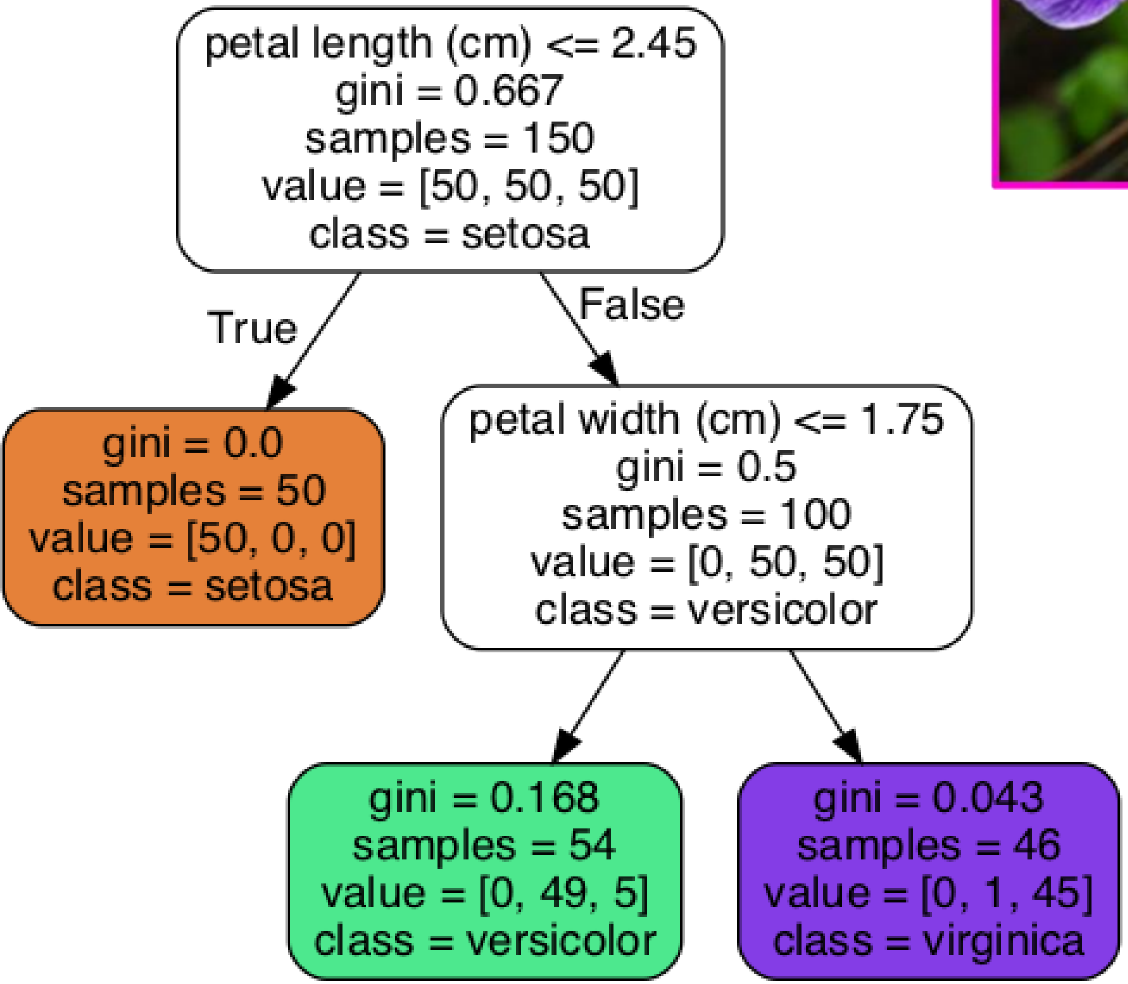
$$H(X_m) = \sum_k p_{mk}(1 - p_{mk})$$ Gini Index

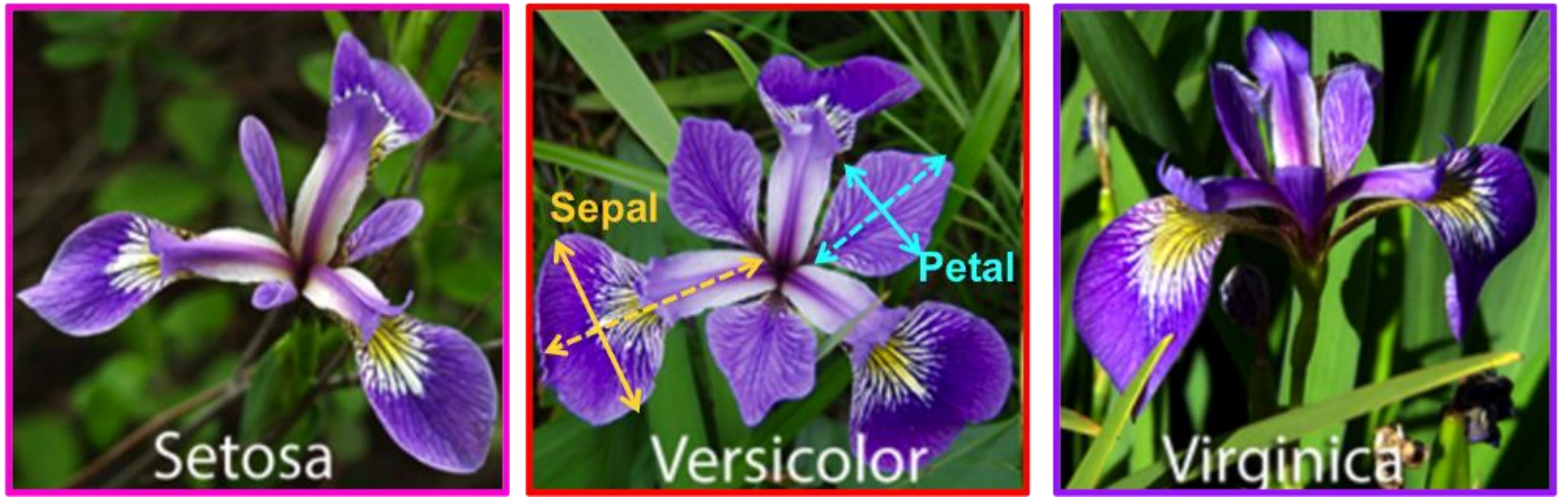
$$H(X_m) = -\sum_k p_{mk} \log(p_{mk})$$ Entropy

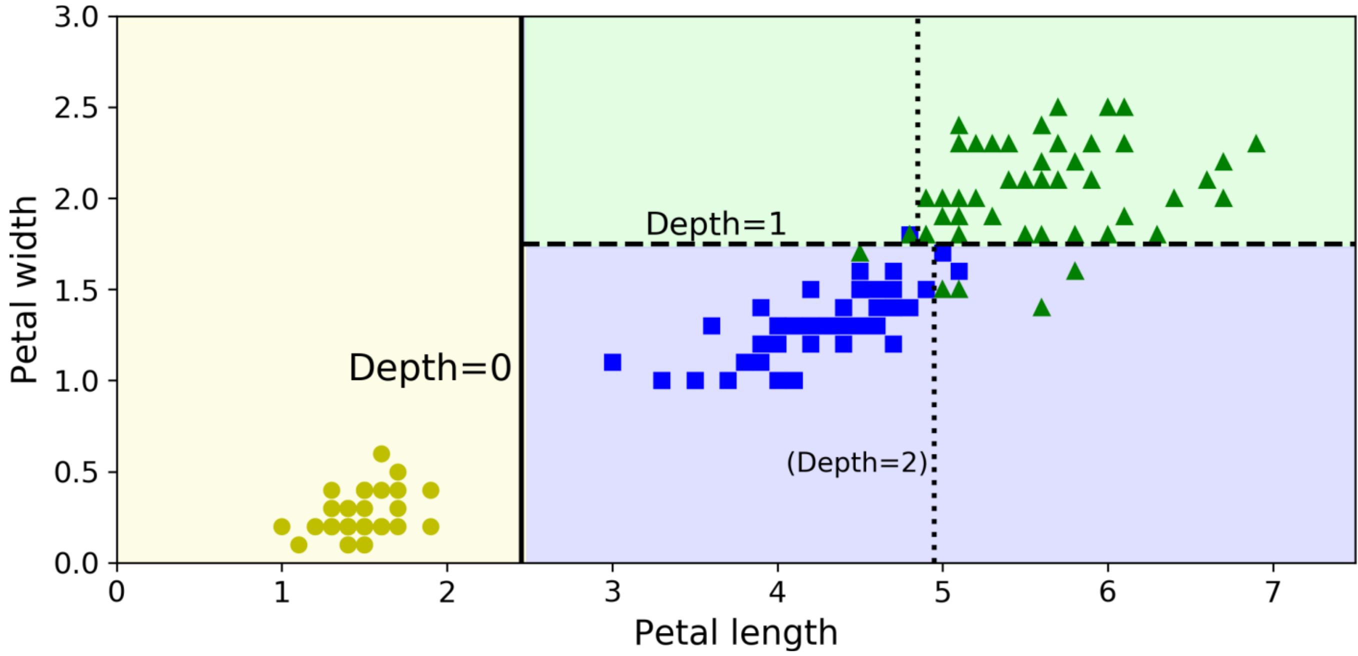
https://scikit-learn.org/stable/modules/tree.html#classification-criteria

p. 311-313, Hastie, 2014

Iris dataset







Homework assigment

Train and fine-tune a Decision Tree for the [moons dataset].

a. Generate a moons dataset using `make_moons(n_samples=10000, noise=0.4)`.

b. Split it into a training set and a test set using `train_test_split()`.

c. Use grid search with cross-validation (with the help of the `GridSearchCV` class) to find good hyperparameter values for a `DecisionTreeClassifier`.

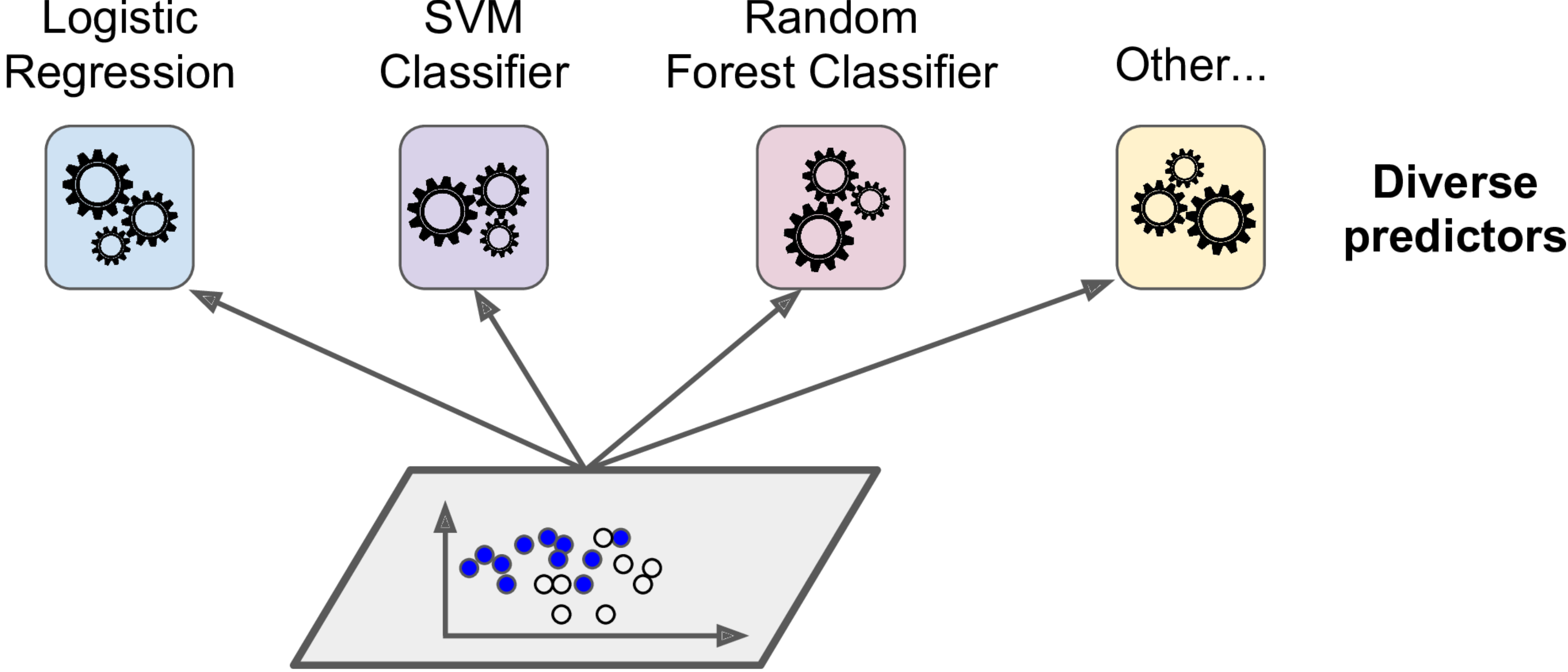
Hint: try various values for `max_leaf_nodes`.

d. Train it on the full training set using these hyperparameters, and measure your model's performance on the test set. You should get roughly 85% to 87% accuracy.

Decision Trees advantages and disadvantages

- Are simple to understand and interpret
- Have value even with little hard data
- A decision tree does not require normalization or scaling of data
- Help determine worst, best and expected values for different scenarios

- High variance
- They are often relatively inaccurate
- Rectangular domains



Ensemble methods

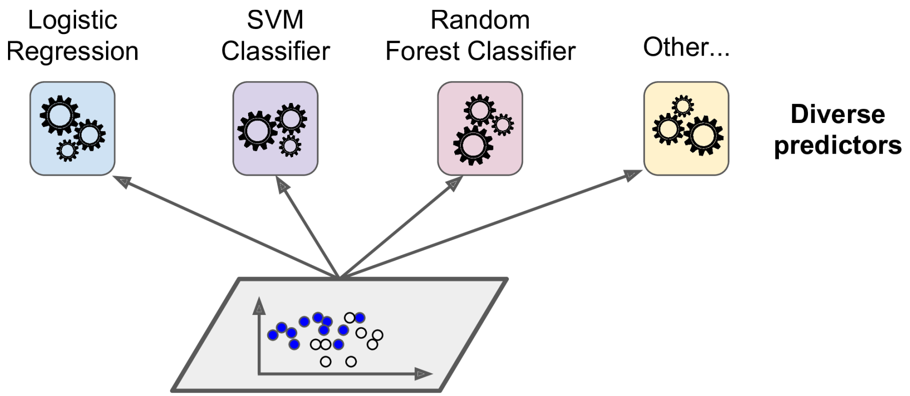
$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{b}(x)$$

Theoretical origin

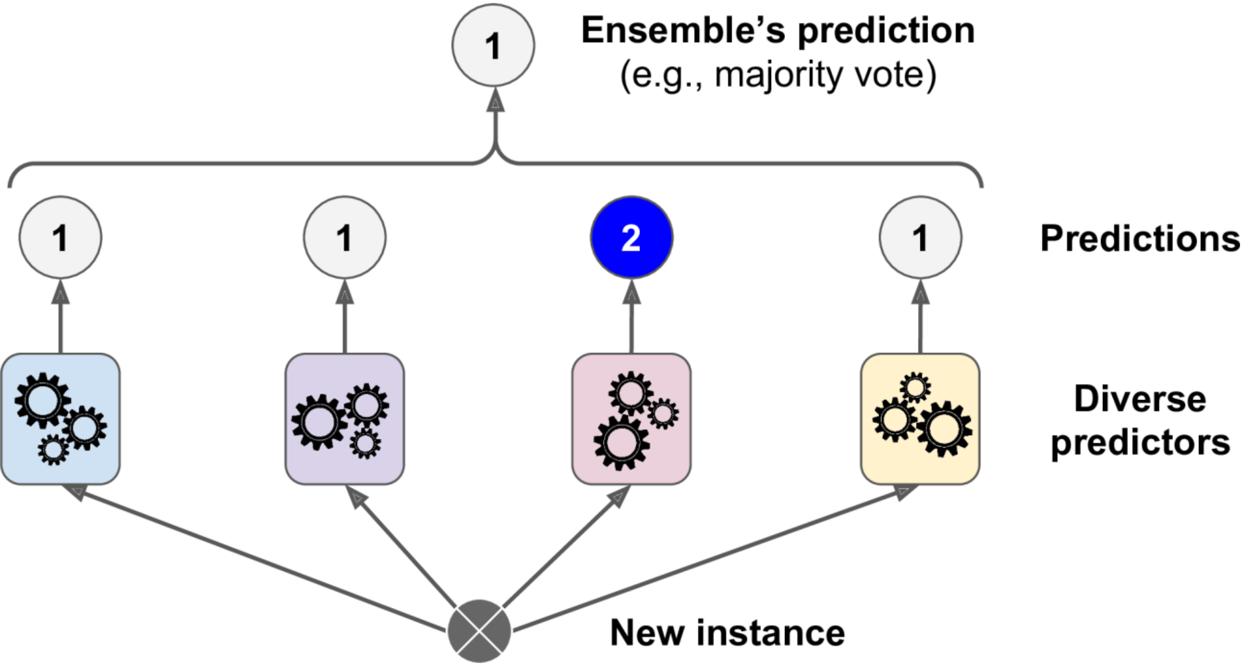
Variance of Sample mean:

Let $Z_1, \ldots, Z_n$ be n random variables i.i.d., each with variance σ^2. The variance of the mean $\bar{Z}$ of the observations is given by σ^2/n.

Classification (Voting)



Hard (majority) voting, i.e., the mode



Soft voting (argmax of probabilities)

	Class A	Class B
Classifier 1	99%	1%
Classifier 2	49%	51%
Classifier 3	49%	51%
Ensemble	**(99 + 49 + 49) / 3 =65.7%**	

Bagging (Bootstrap Aggregation. Parallel-wise model fitting)

Bootstraping. Sampling instances with replacement

Out Of Bag Error (oob 63+37).

Predictor 0	Predictor 1	Predictor 2	Predictor 3	Predictor 4	Label

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingClassifier.html

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingRegressor.html

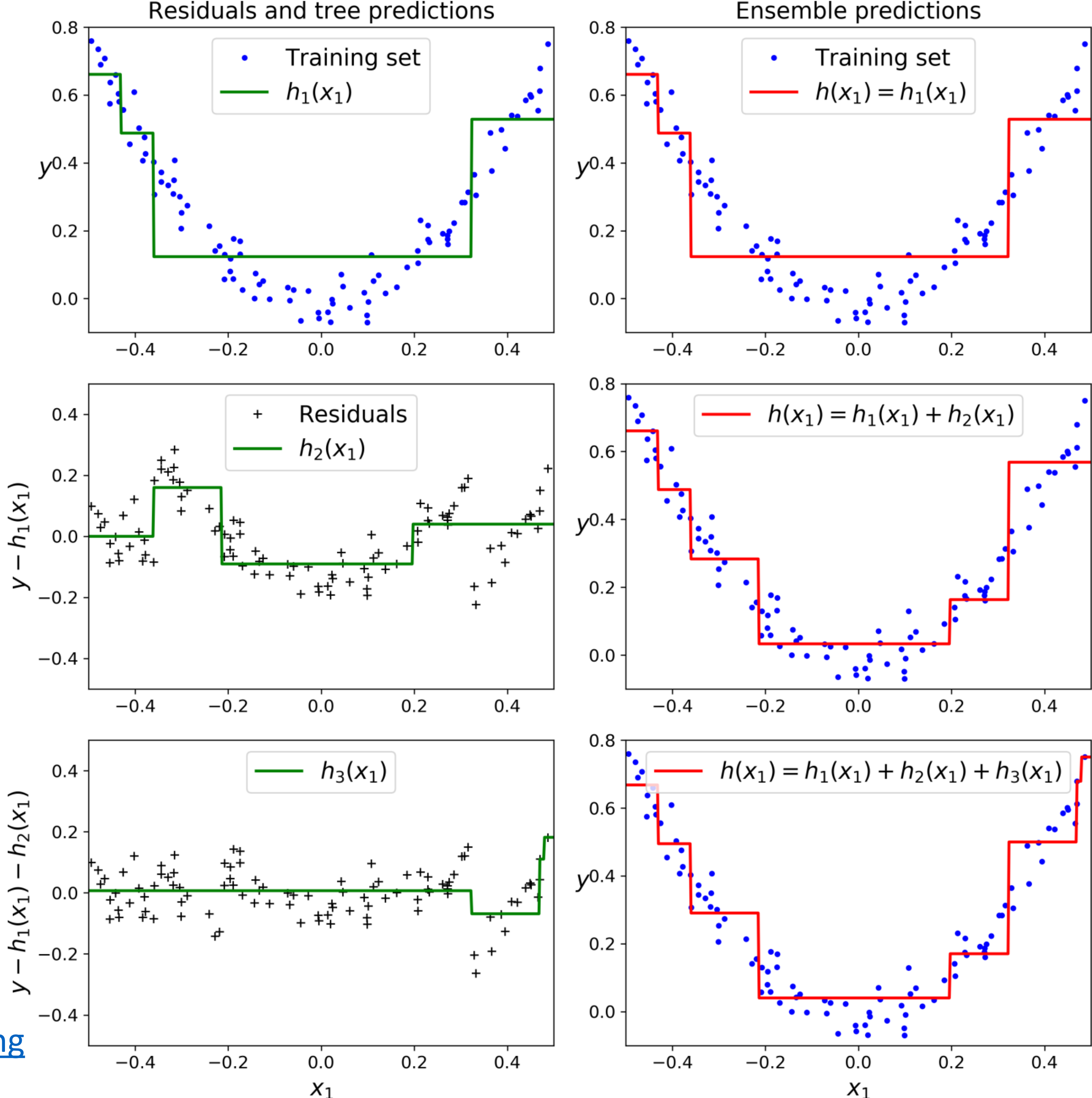
Random Forests

Multiple Decision Trees, each grown by sampling predictor variables

Predictor 0	Predictor 1	Predictor 2	Predictor 3	Predictor 4	Label

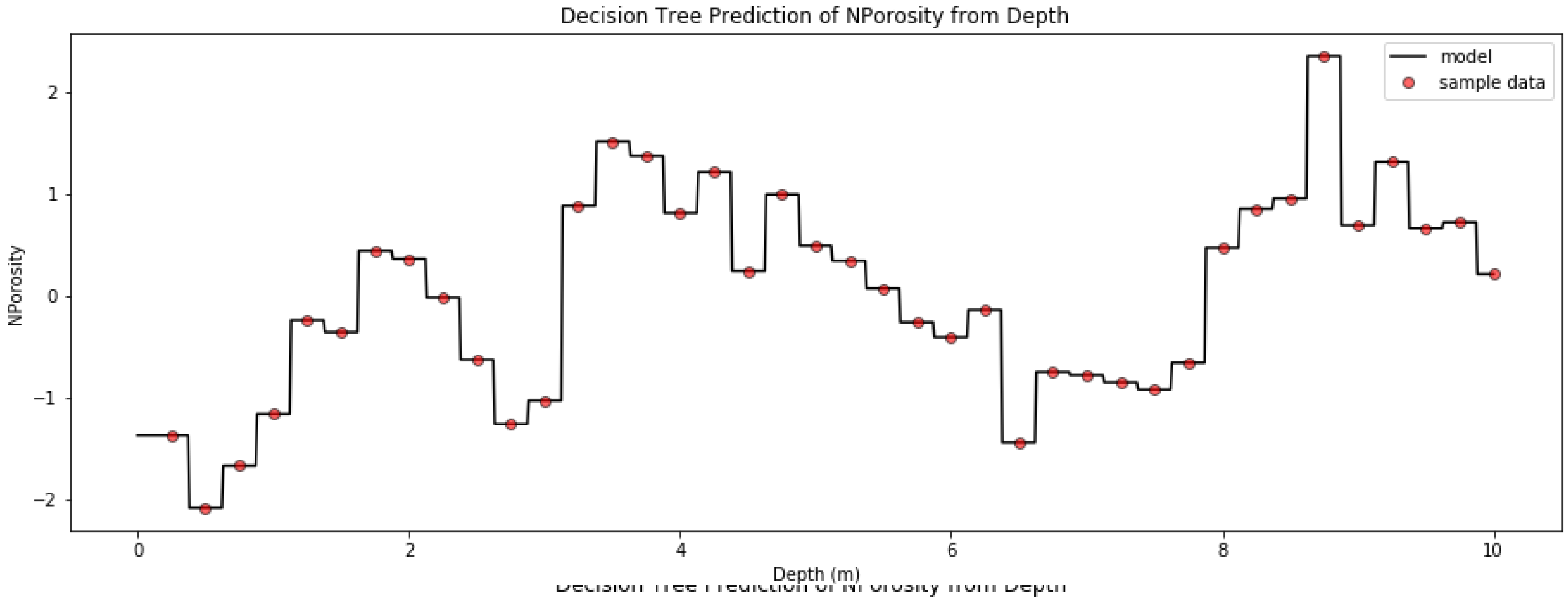
Boosting

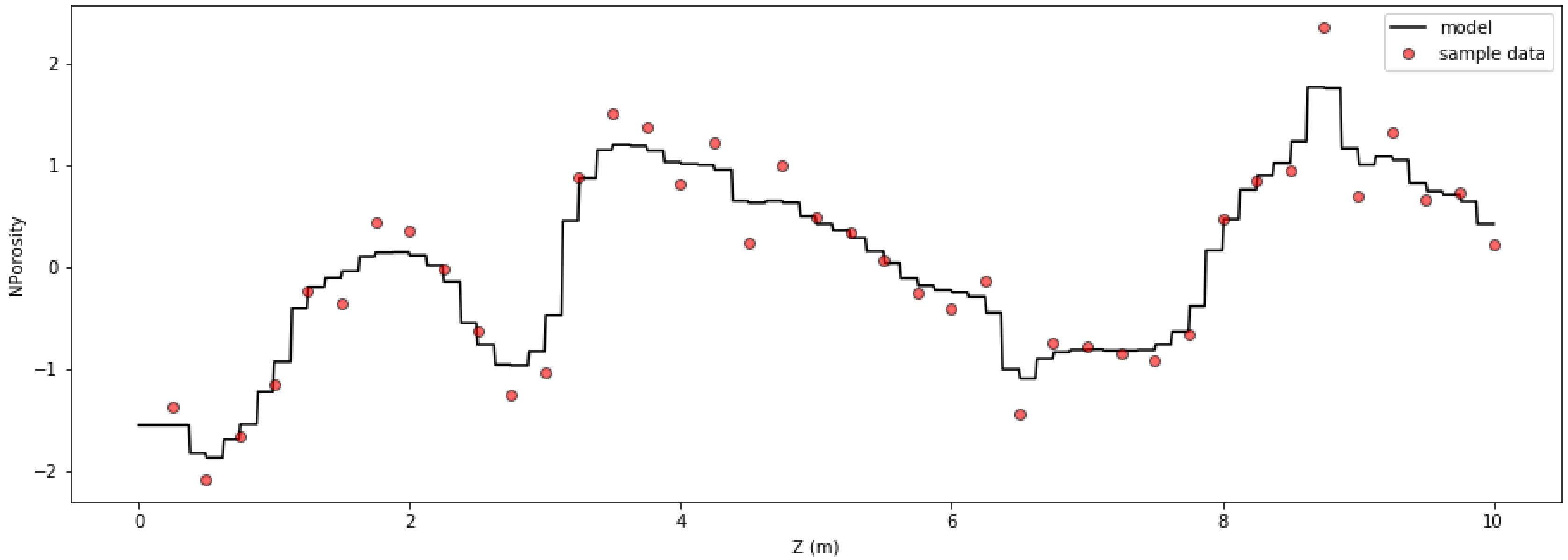
Sequentially



Aurelien, 2019. Page 207

[Video_ Trevor Hastie - Gradient Boosting Machine Learning]

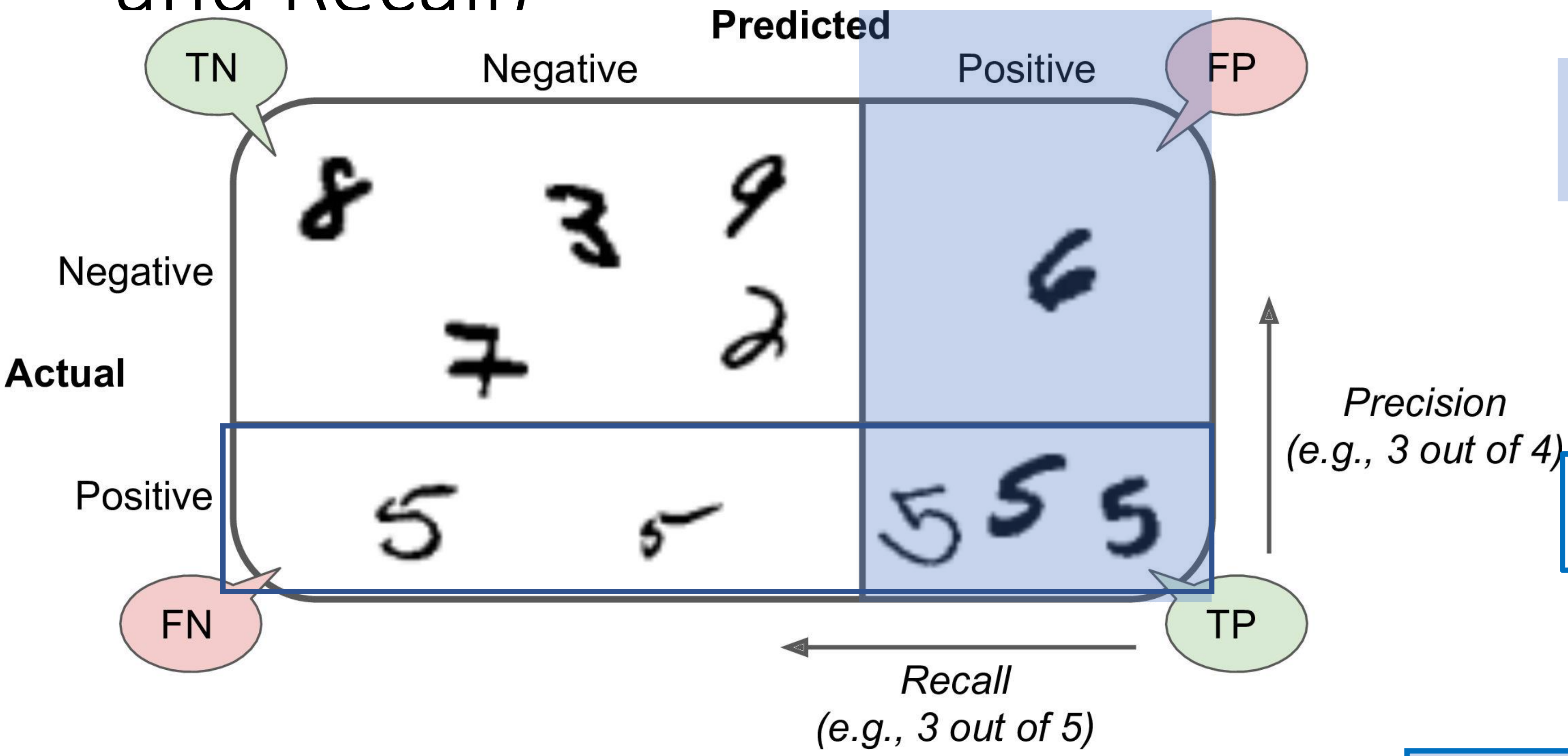




Random Forest

https://github.com/GeostatsGuy/PythonNumericalDemos/blob/master/SubsurfaceDataAnalytics_PolygonalRegression.ipynb

Accuracy assessment (Confusion Matrix, Precision and Recall)



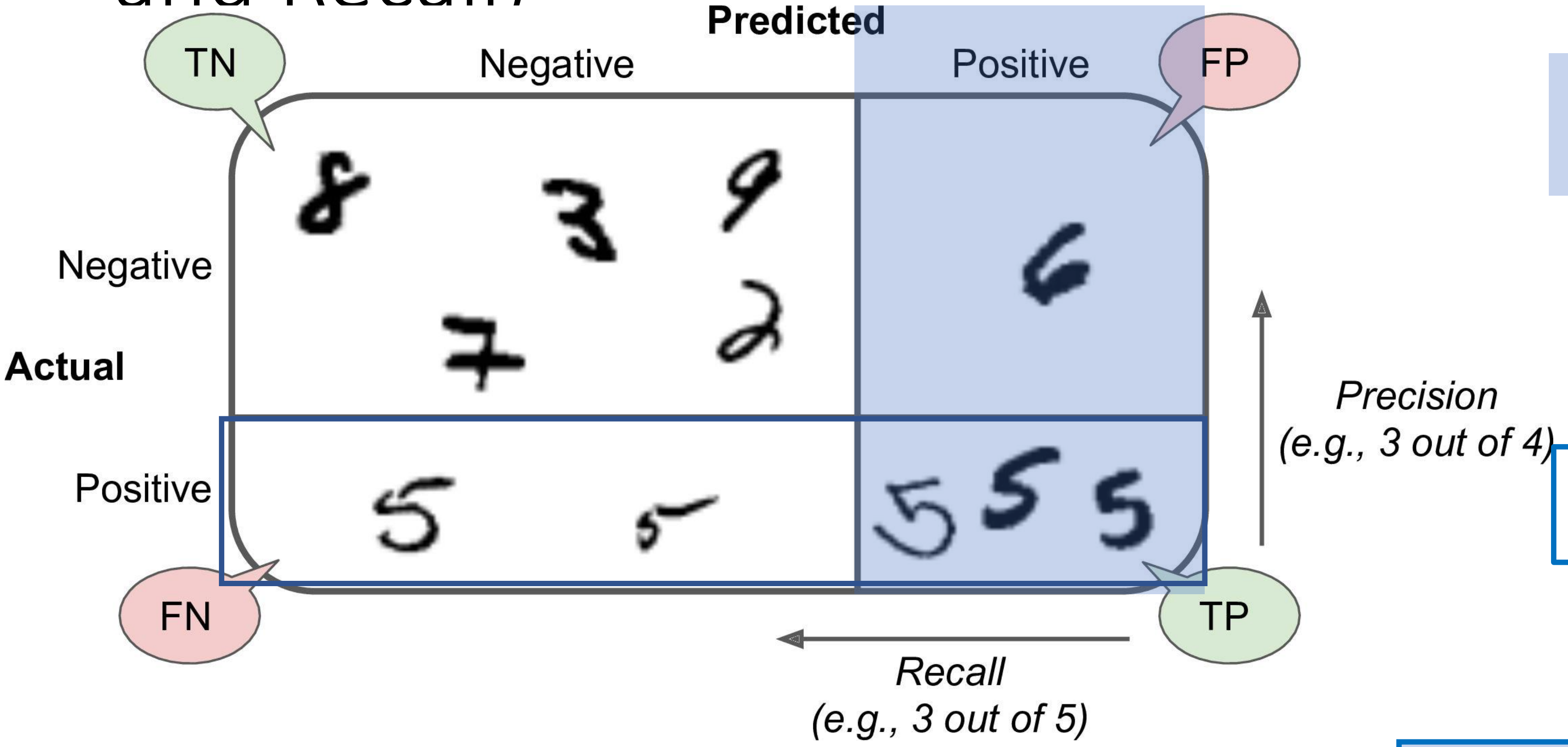
$$\text{Precision} = P := \frac{TP}{TP+FP}$$

$$\text{Recall} = R := \frac{TP}{TP+FN}$$

$$F_1 \text{ score} = \frac{2}{\frac{1}{P}+\frac{1}{R}}$$

```
>>> from sklearn.metrics import confusion_matrix
>>> confusion_matrix(y_train_5, y_train_pred)
array([[53057,  1522],
       [ 1325,  4096]])
```

Accuracy assessment (Confusion Matrix, Precision and Recall)



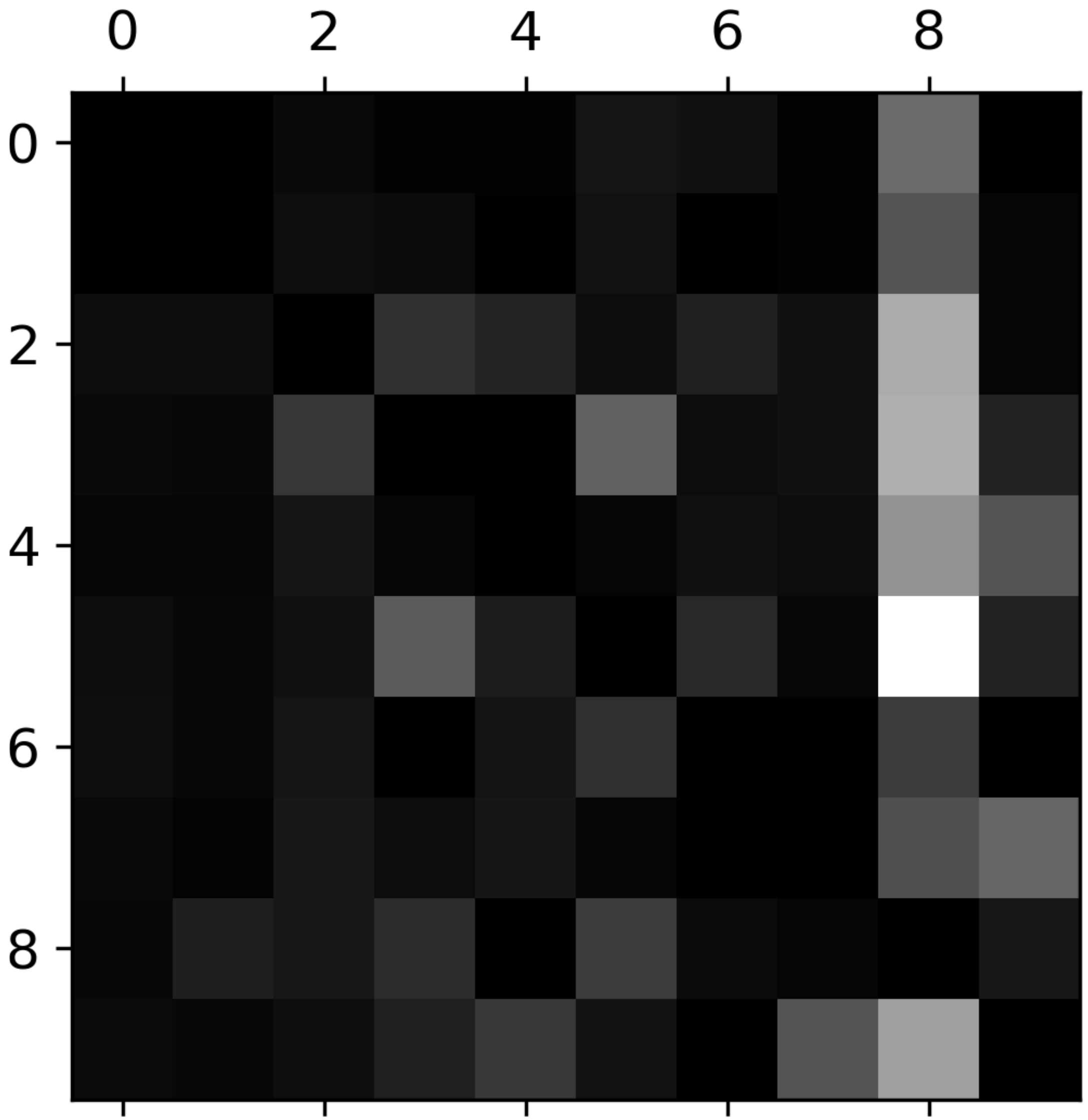
$$\text{Precision} = \mathrm{P} := \frac{TP}{TP+FP}$$

$$\text{Recall} = R := \frac{TP}{TP+FN}$$

$$F_1 \text{ score} = \frac{2}{\frac{1}{P}+\frac{1}{R}}$$

```
>>> from sklearn.metrics import precision_score, recall_score
>>> precision_score(y_train_5, y_train_pred) # == 4096 / (4096 + 1522)
0.7290850836596654
>>> recall_score(y_train_5, y_train_pred) # == 4096 / (4096 + 1325)
0.7555801512636044
```

# of All digits



Homework assigment

Train and fine-tune a SVM, Random Forest, ANN, Extra-Trees, AdaBoost for the moons dataset.

a. Generate a moons dataset using `make_moons(n_samples=10000, noise=0.4)`.

b. Split it into a training set and a test set using `train_test_split()`.

c. Measure your model's performance on the test set.