

# Lecture 1: Intro, Trees, ensembles

**Neychev Radoslav**

ML Instructor (MIPT, HSE, Harbour.Space, BigData Team)

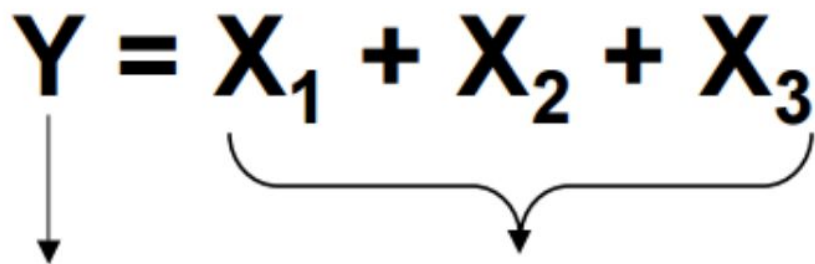
Research Scientist, MIPT

02.10.2019, Moscow, Russia

# Outline

1. Linear models and metrics recap
2. Decision trees in classification and regression.
3. Information criteria.
4. Pruning.
5. Ensembling methods
6. Optional: advanced ensembling methods

# Linear models recap

$$Y = X_1 + X_2 + X_3$$


Dependent Variable

Outcome Variable

Response Variable

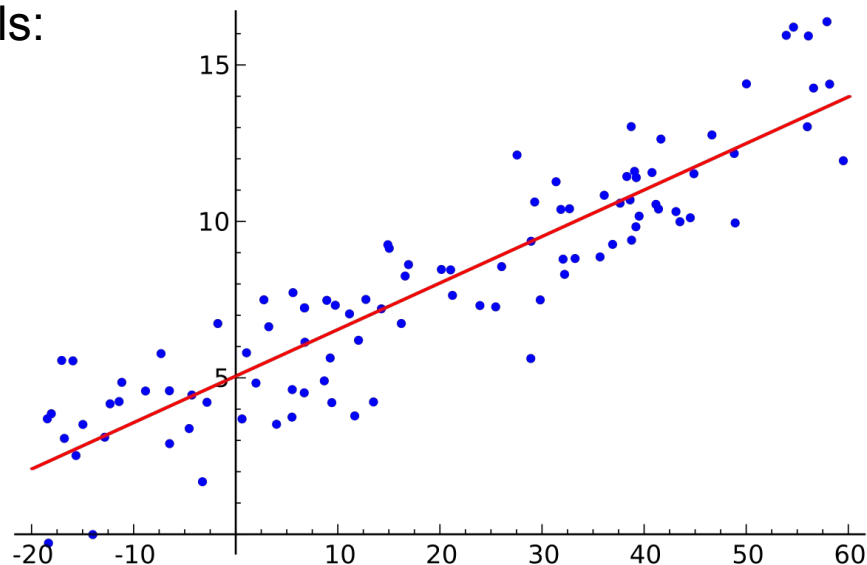
Independent Variable

Predictor Variable

Explanatory Variable

# Linear models

- Predictive models:



Estimated  
(or predicted)  
Y value for  
observation  $i$

Estimate of  
the regression  
intercept

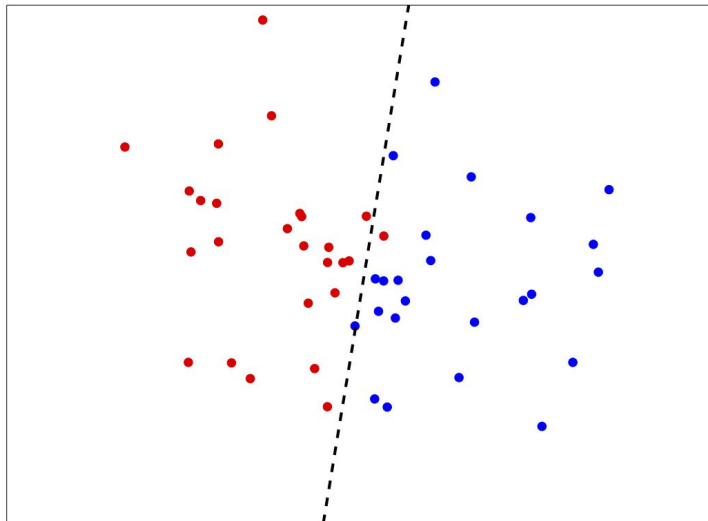
Estimate of the  
regression slope

Value of X for  
observation  $i$

$$\hat{Y}_i = b_0 + b_1 X_i$$

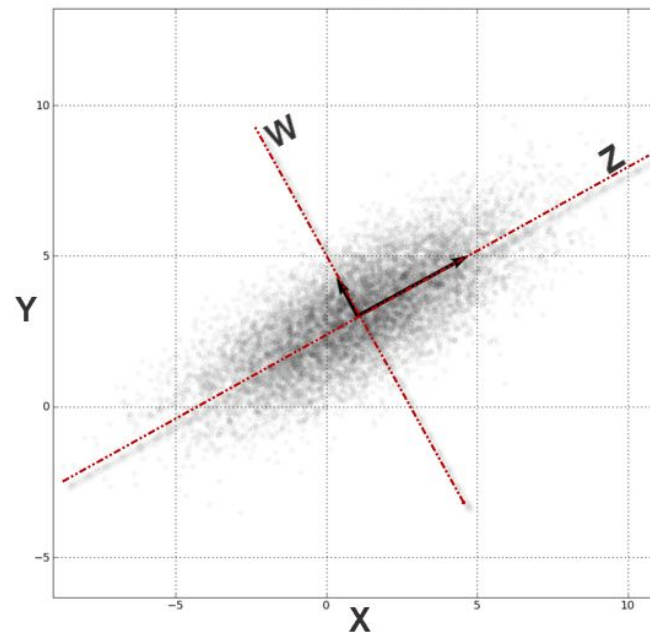
# Linear models

- Predictive models:
- Classification models:



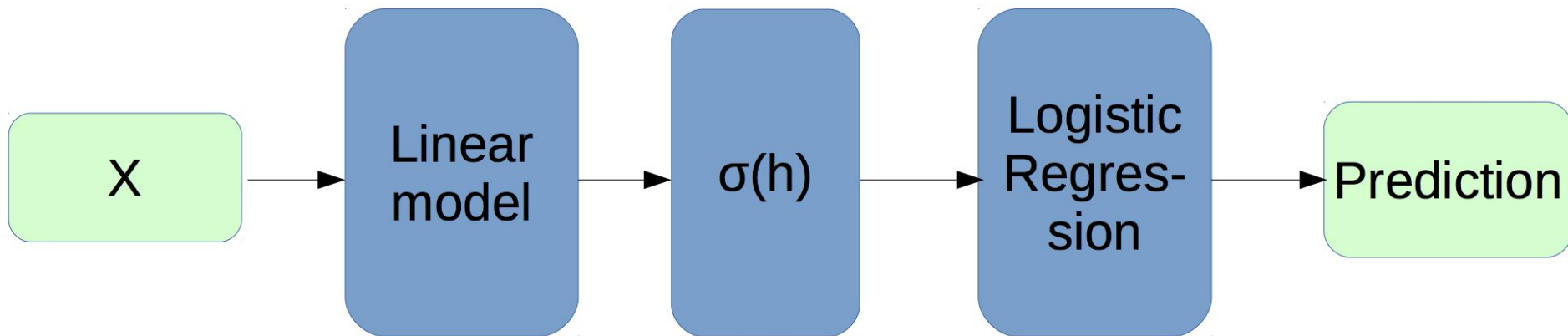
# Linear models

- Predictive models:
- Classification models:
- Unsupervised models (e.g. PCA analysis)



# Linear models

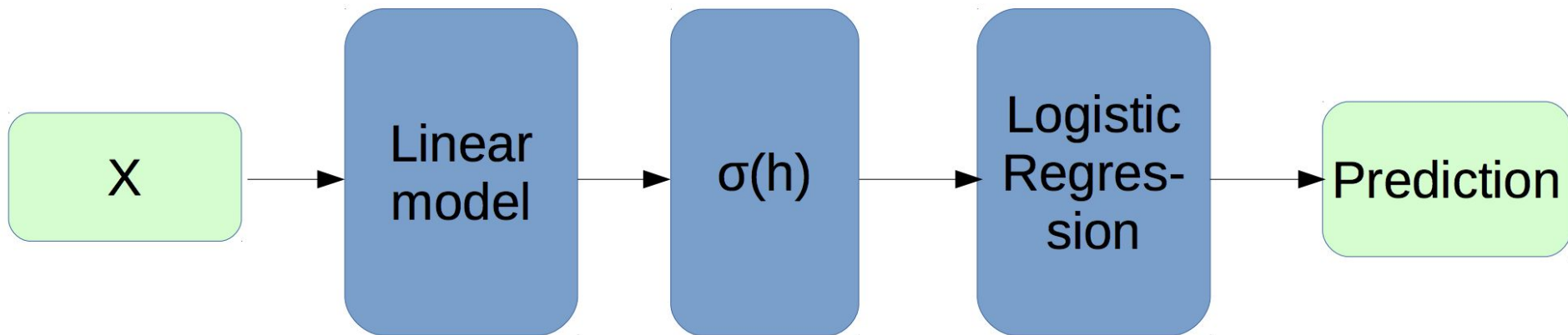
- Predictive models:
- Classification models:
- Unsupervised models (e.g. PCA analysis)
- Building block of other models (ensembles, NNs, etc.)





# Linear models

- Predictive models:
- Classification models:
- Unsupervised models (e.g. PCA analysis)
- Building block of other models (ensembles, NNs, etc.)



*Actually, it's a neural network. We will meet it later.*

# Quality functions in classification

# Quality functions in classification

- Accuracy
- Precision
- Recall
- F-score
- ROC-curve, ROC-AUC
- PR-curve

# Accuracy

Number of right classifications

target: 1 0 1 0 0 0 0 1 0 0

Number of right classifications

target: 1 0 1 0 0 0 0 1 0 0

predicted: 0 0 1 0 0 0 0 1 1 0

Number of right classifications

target: 1 0 1 0 0 0 0 1 0 0

predicted: 0 0 1 0 0 0 0 1 1 0

Number of right classifications

target: 1 0 1 0 0 0 0 1 0 0

predicted: 0 0 1 0 0 0 0 1 1 0

accuracy = 8/10 = 0.8

# Precision and recall

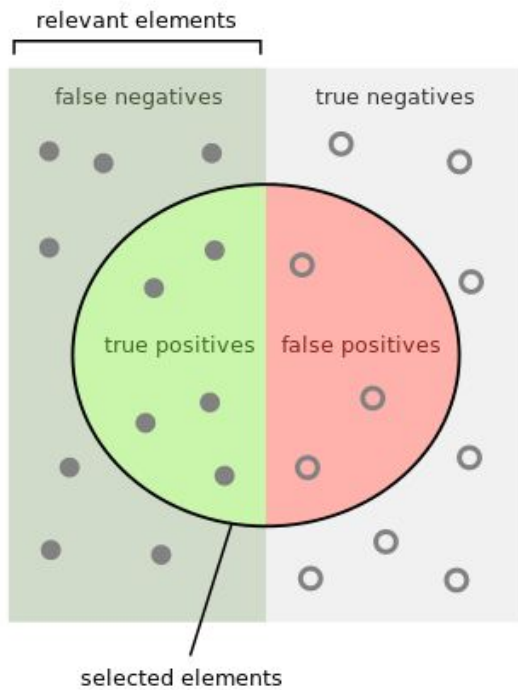
		Actual Class	
		Yes	No
Predicted Class	Yes	<b>T</b> True <b>P</b> Positive	<b>F</b> False <b>P</b> Positive
	No	<b>F</b> False <b>N</b> Negative	<b>T</b> True <b>N</b> Negative

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$



# Precision and recall



		Actual Class	
		Yes	No
Predicted Class	Yes	<b>T</b> True <b>P</b> Positive	<b>F</b> False <b>P</b> Positive
	No	<b>F</b> False <b>N</b> Negative	<b>T</b> True <b>N</b> Negative

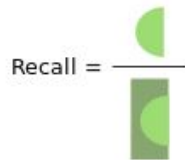
$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

How many selected items are relevant?



How many relevant items are selected?



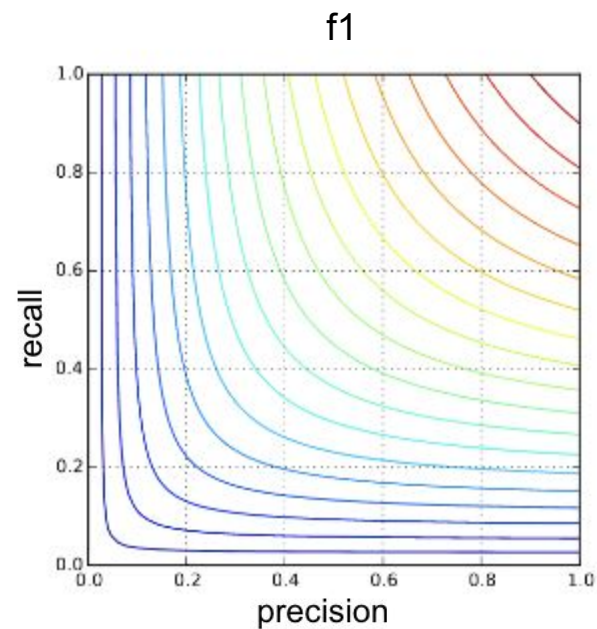
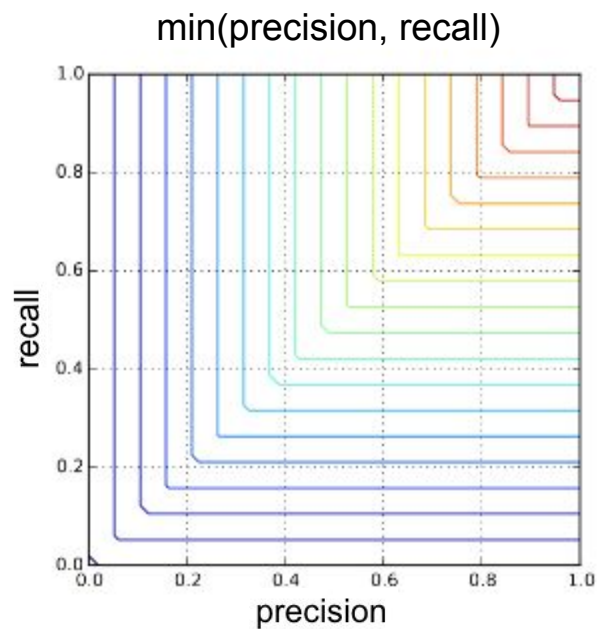
Harmonic mean of precision and recall.  
Closer to the smallest one.

$$F_1 = \left( \frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Harmonic mean of precision and recall.  
Closer to the smallest one.

$$F_1 = \left( \frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$



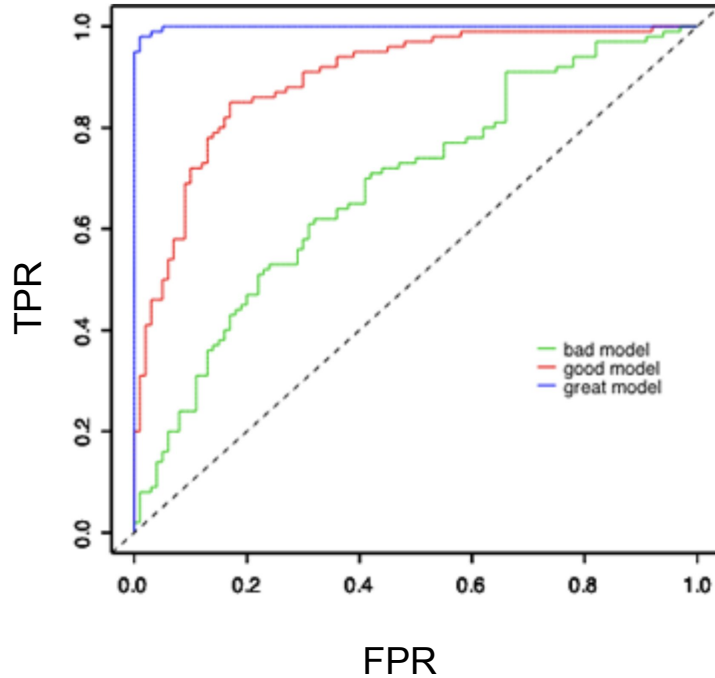
# ROC - receiver operating characteristic

		Actual Class	
		Yes	No
Predicted Class	Yes	<b>True Positive</b>	<b>False Positive</b>
	No	<b>False Negative</b>	<b>True Negative</b>

$$TPR = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}}$$

$$FPR = \frac{\text{False positives}}{\text{False positives} + \text{True negatives}}.$$

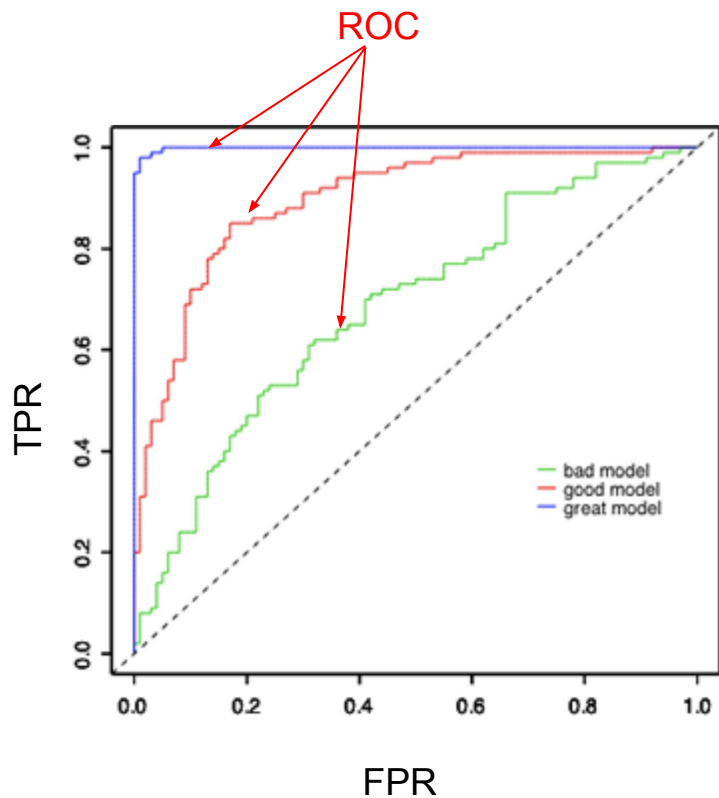
# ROC - receiver operating characteristic



		Actual Class	
		Yes	No
Predicted Class	Yes	<b>True Positive</b>	<b>False Positive</b>
	No	<b>False Negative</b>	<b>True Negative</b>

$$TPR = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}}$$

$$FPR = \frac{\text{False positives}}{\text{False positives} + \text{True negatives}}$$

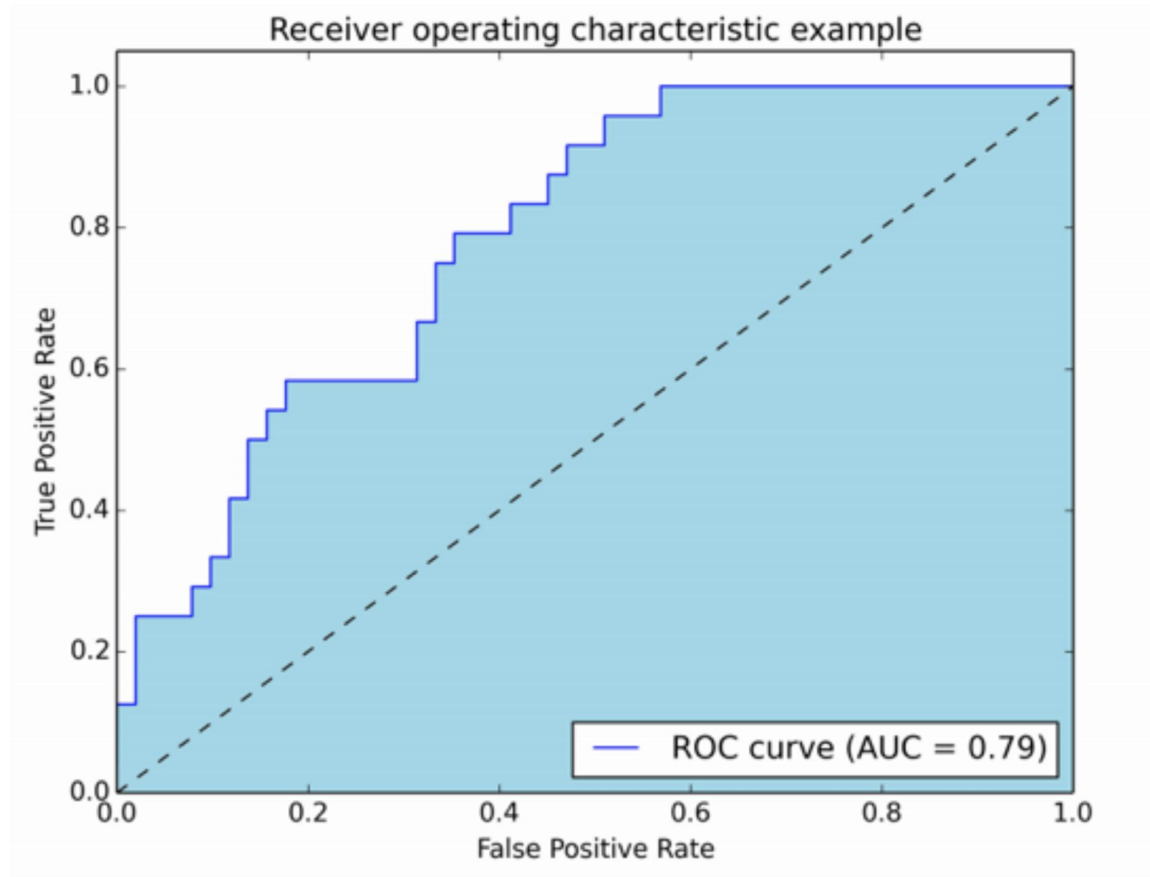


		Actual Class	
		Yes	No
Predicted Class	Yes	<b>True Positive</b>	<b>False Positive</b>
	No	<b>False Negative</b>	<b>True Negative</b>

$$TPR = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}}$$

$$FPR = \frac{\text{False positives}}{\text{False positives} + \text{True negatives}}$$

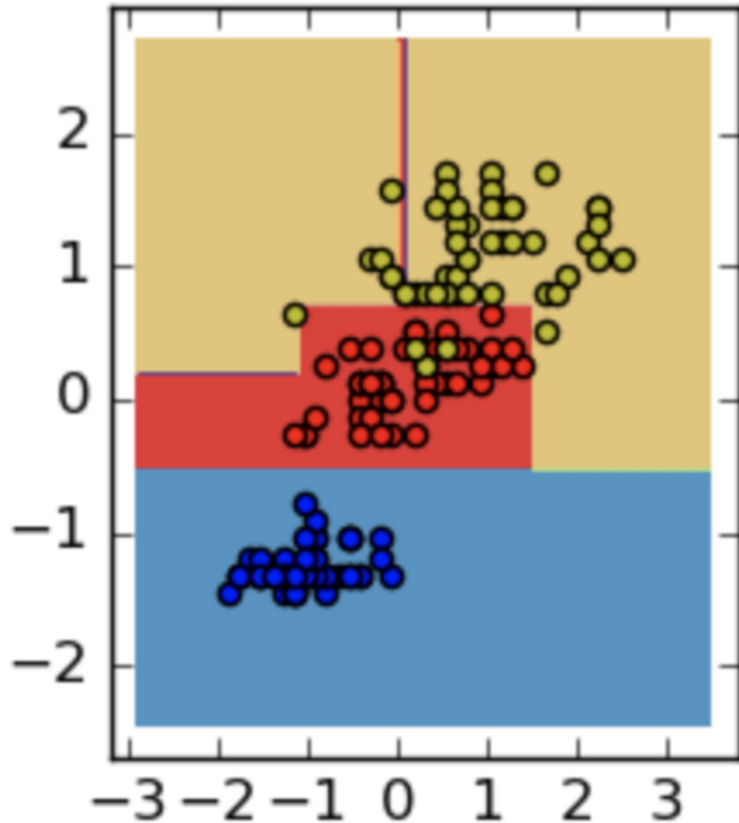
# ROC-AUC - area under curve





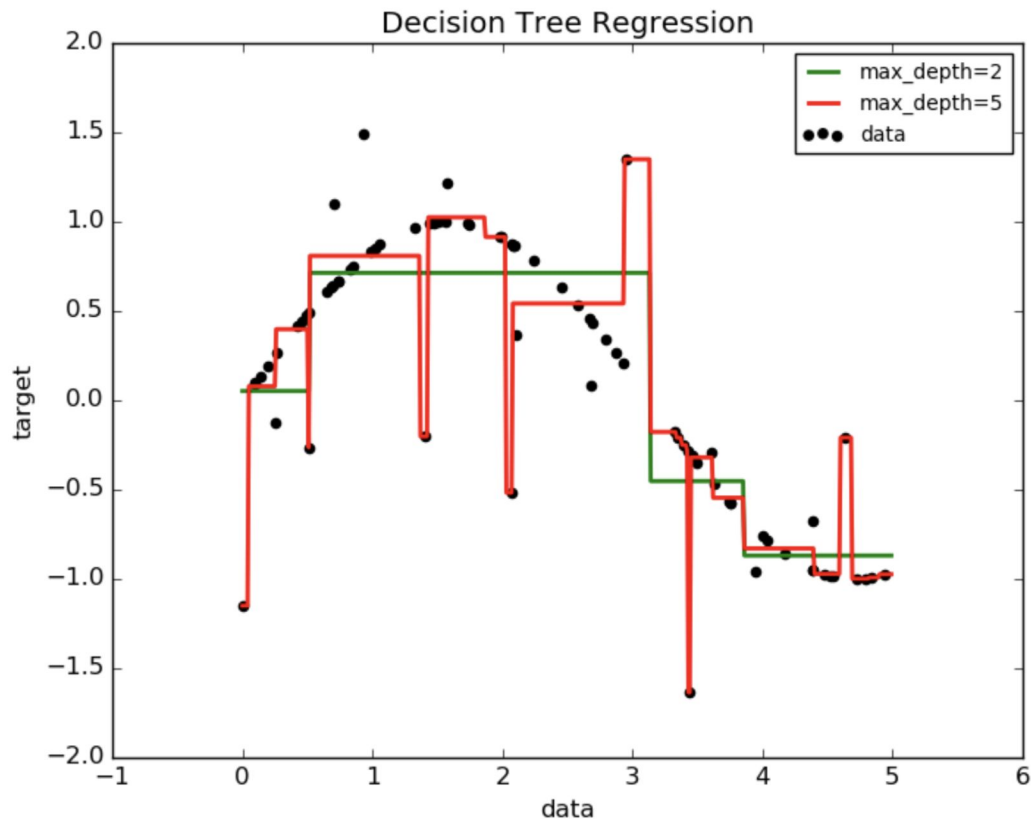
# Decision trees

# Decision tree in classification



Classification problem with 3 classes and 2 features.

# Decision tree in regression



Green - decision tree of depth 2

Red - decision tree of depth 5

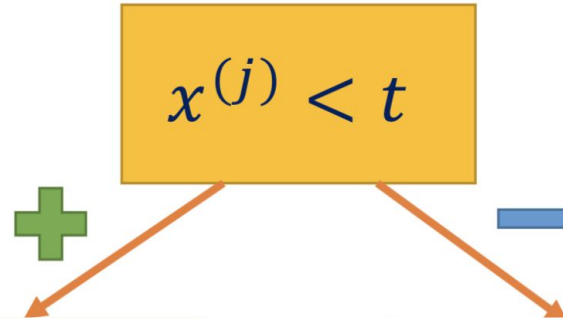
Every leaf corresponds to some constant.

# Constructing decision trees

$$x^{(j)} < t$$

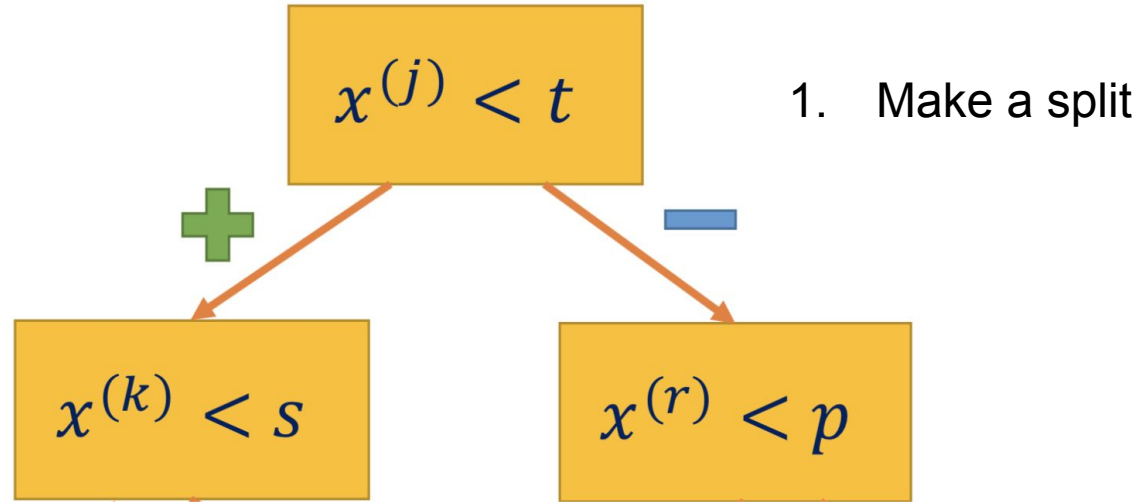
1. Make a split

# Constructing decision trees

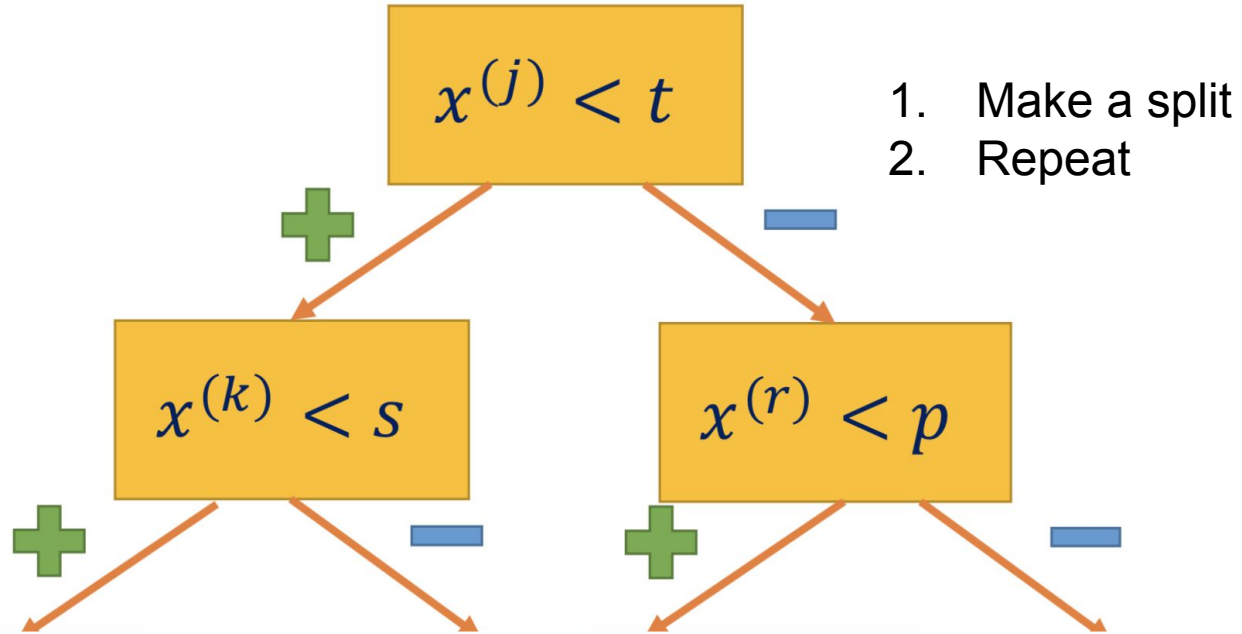


1. Make a split

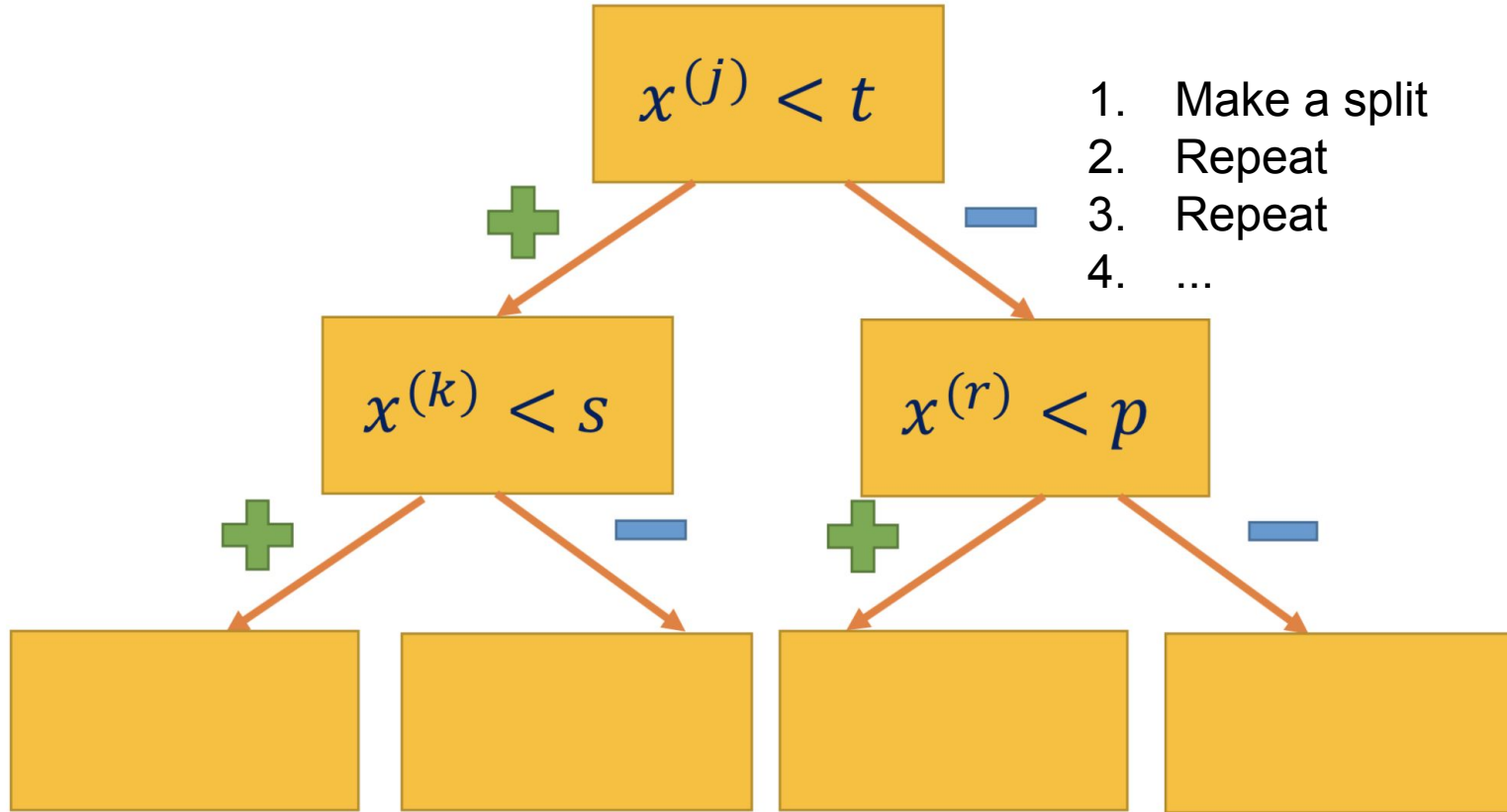
# Constructing decision trees



# Constructing decision trees



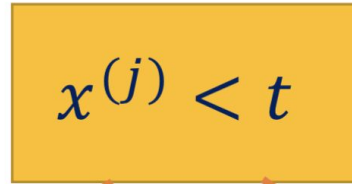
# Constructing decision trees



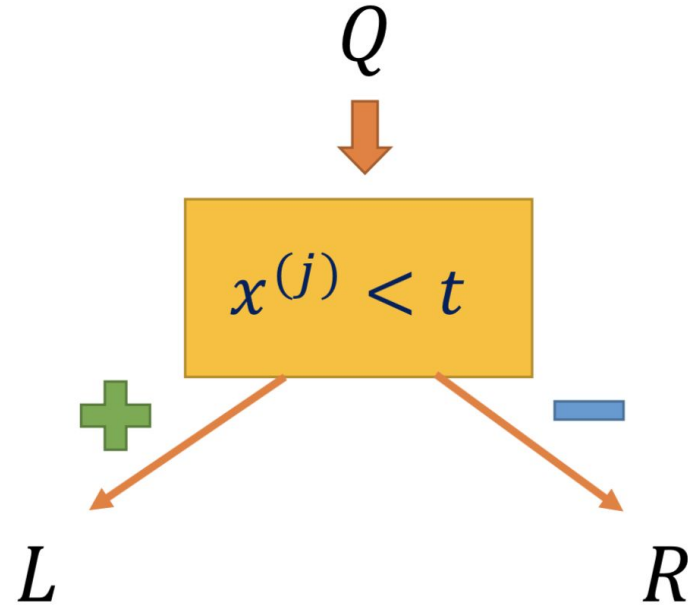


# How to split data properly?

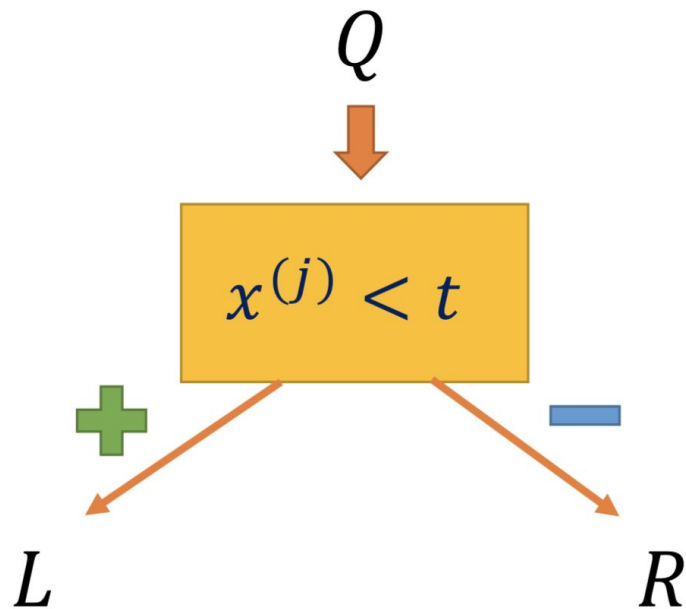
$Q$



# How to split data properly?

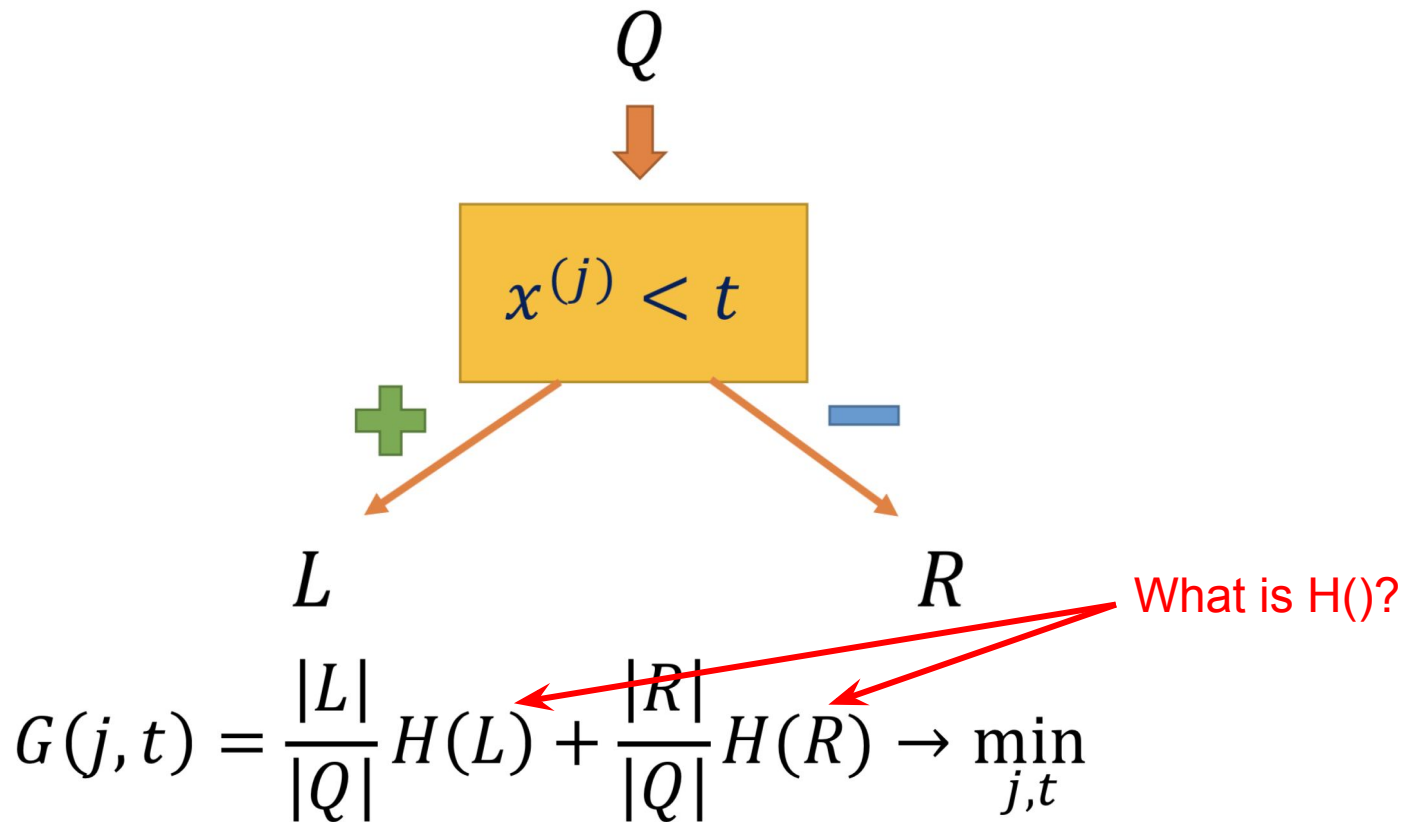


# How to split data properly?



$$G(j, t) = \frac{|L|}{|Q|} H(L) + \frac{|R|}{|Q|} H(R)$$

# How to split data properly?



# Information criteria

$H(R)$  is measure of “heterogeneity” of our data.

Consider **binary classification** problem:

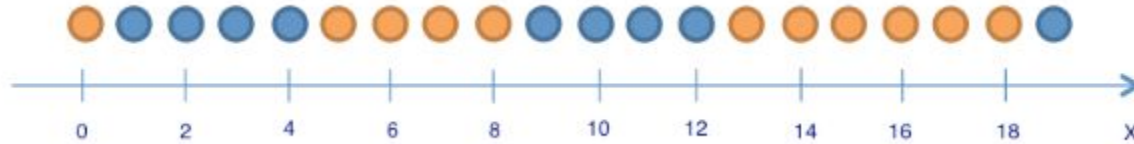
1. Misclassification criteria:  $H(R) = 1 - \max\{p_0, p_1\}$

2. Entropy criteria:  $H(R) = -p_0 \log_2 p_0 - p_1 \log_2 p_1$

3. Gini impurity:  $H(R) = 1 - p_0^2 - p_1^2 = 1 - 2p_0p_1$

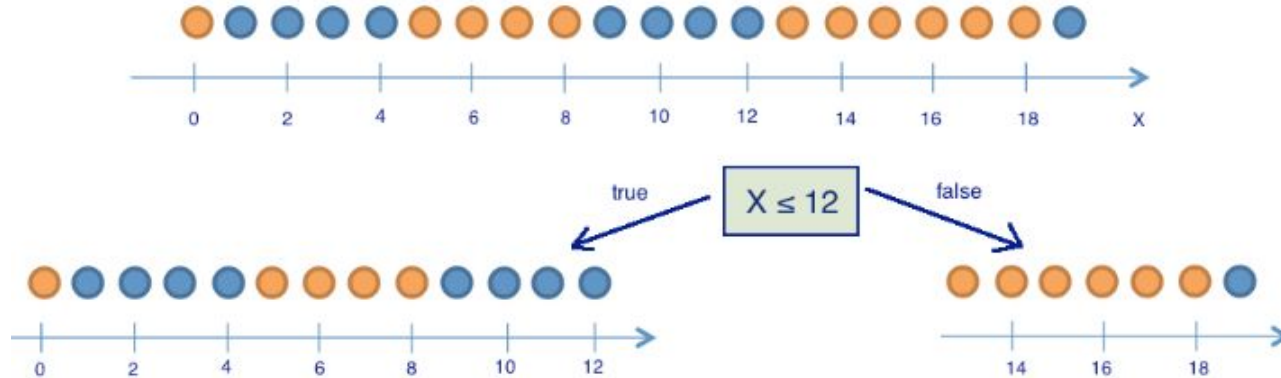
# Information criteria

$H(R)$  is measure of “heterogeneity” of our data.  
Consider binary classification problem:

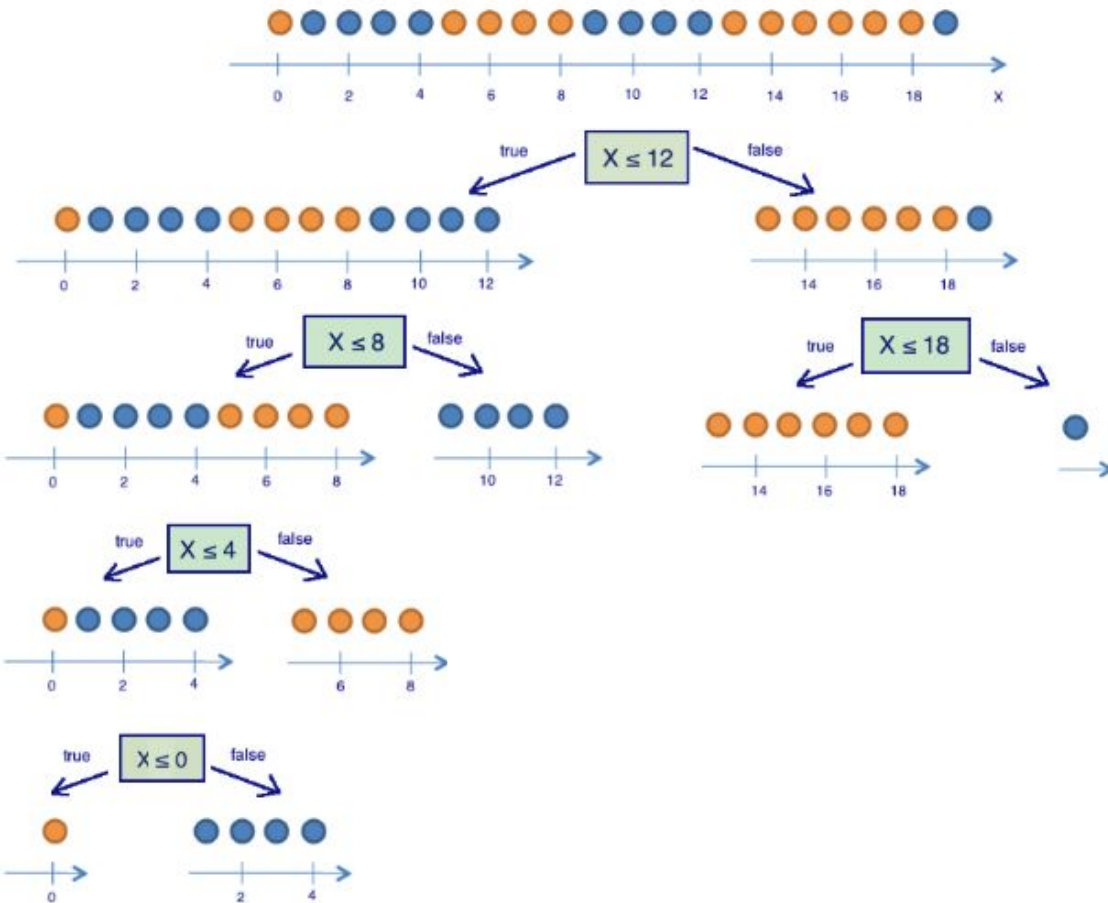


# Information criteria

$H(R)$  is measure of “heterogeneity” of our data.  
Consider binary classification problem:



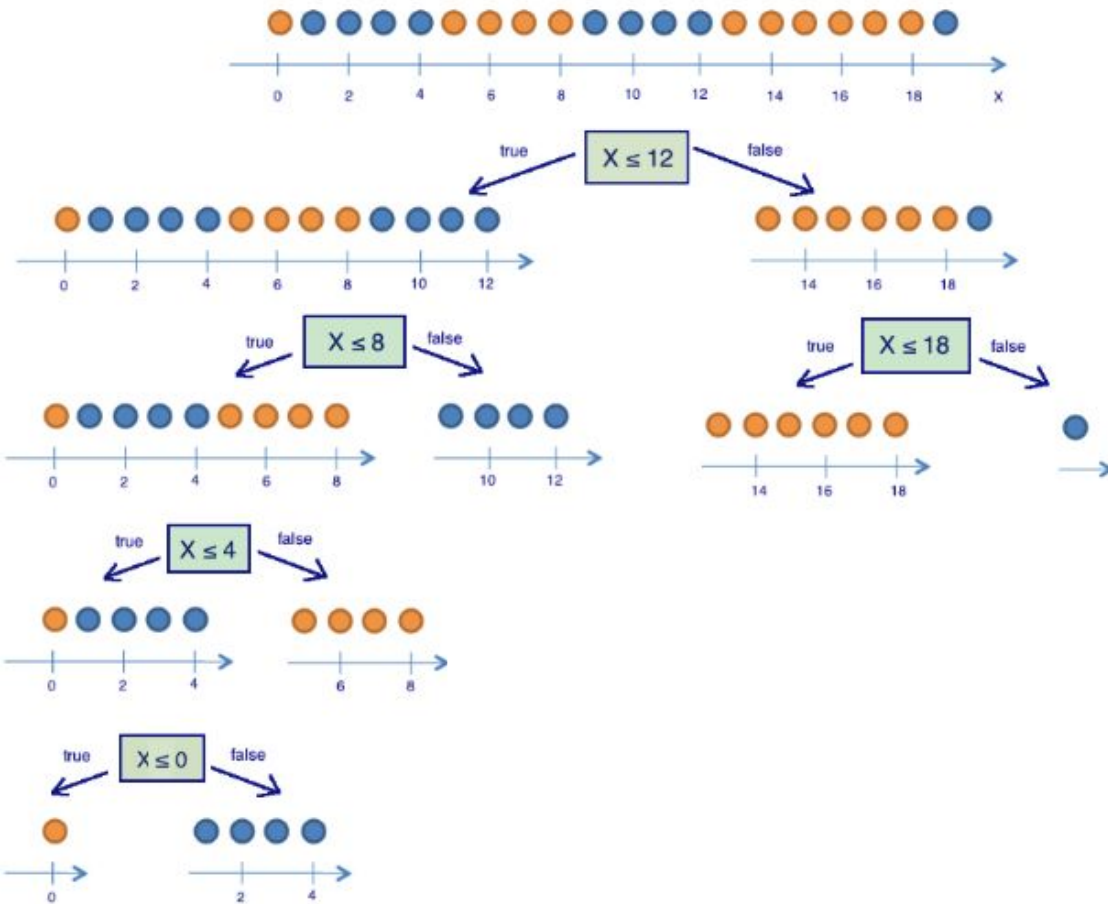
# Information criteria: Entropy





# Information criteria: Entropy

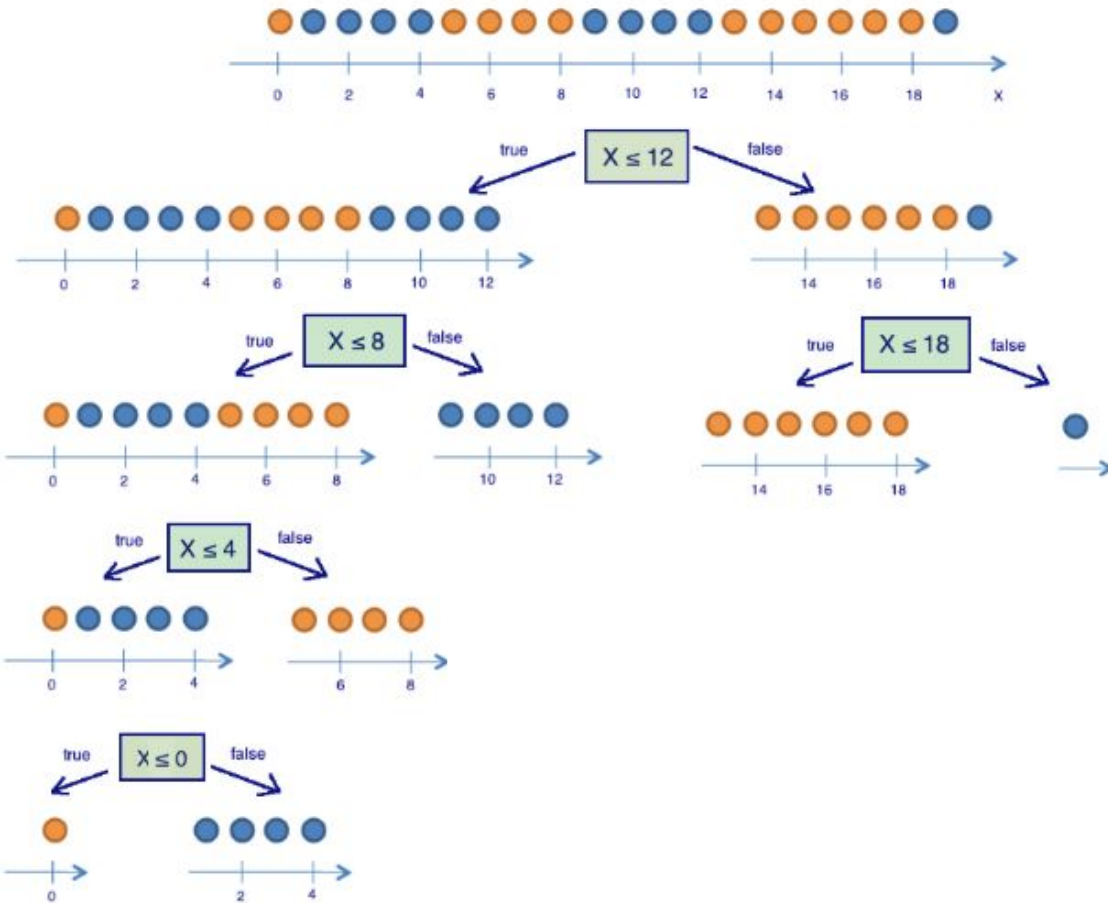
$$S = - \sum_k p_k \log_2 p_k$$



# Information criteria: Entropy

$$S = - \sum_k p_k \log_2 p_k$$

In binary case  $N = 2$



$$S = -p_+ \log_2 p_+ - p_- \log_2 p_- = -p_+ \log_2 p_+ - (1 - p_+) \log_2 (1 - p_+)$$

$H(R)$  is measure of “heterogeneity” of our data.

Consider **multiclass classification** problem:

1. Misclassification criteria: 
$$H(R) = 1 - \max_k \{p_k\}$$

2. Entropy criteria: 
$$H(R) = - \sum_k p_k \log_2 p_k$$

3. Gini impurity: 
$$H(R) = 1 - \sum_k (p_k)^2$$

$H(R)$  is measure of “heterogeneity” of our data.

Consider **regression** problem:

1. Mean squared error

$$H(R) = \min_c \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - c)^2$$

$H(R)$  is measure of “heterogeneity” of our data.

Consider **regression** problem:

1. Mean squared error

$$H(R) = \min_c \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - c)^2$$

What is the constant?

$H(R)$  is measure of “heterogeneity” of our data.

Consider **regression** problem:

1. Mean squared error

$$H(R) = \min_c \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - c)^2$$

$$c^* = \frac{1}{|R|} \sum_{y_i \in R} y_i$$

# Bootstrap

Consider dataset  $X$  containing  $N$  objects.

Pick  $I$  objects with return from  $X$  and repeat in  $N$  times to get  $N$  datasets.

Error of model trained on  $X_j$ :  $\varepsilon_j(x) = b_j(x) - y(x), \quad j = 1, \dots, N,$

Then  $\mathbb{E}_x(b_j(x) - y(x))^2 = \mathbb{E}_x \varepsilon_j^2(x).$

The mean error of  $N$  models:  $E_1 = \frac{1}{N} \sum_{j=1}^N \mathbb{E}_x \varepsilon_j^2(x).$



Consider the errors unbiased and uncorrelated:

$$\mathbb{E}_x \varepsilon_j(x) = 0;$$

$$\mathbb{E}_x \varepsilon_i(x) \varepsilon_j(x) = 0, \quad i \neq j.$$

The final model averages all predictions:

$$a(x) = \frac{1}{N} \sum_{j=1}^N b_j(x).$$

# Bootstrap

Consider the errors unbiased and uncorrelated:

$$\mathbb{E}_x \varepsilon_j(x) = 0;$$

$$\mathbb{E}_x \varepsilon_i(x) \varepsilon_j(x) = 0, \quad i \neq j.$$

The final model averages all predictions:

$$a(x) = \frac{1}{N} \sum_{j=1}^N b_j(x).$$

$$\begin{aligned} E_N &= \mathbb{E}_x \left( \frac{1}{N} \sum_{j=1}^n b_j(x) - y(x) \right)^2 = \\ &= \mathbb{E}_x \left( \frac{1}{N} \sum_{j=1}^N \varepsilon_j(x) \right)^2 = \\ &= \frac{1}{N^2} \mathbb{E}_x \left( \sum_{j=1}^N \varepsilon_j^2(x) + \underbrace{\sum_{i \neq j} \varepsilon_i(x) \varepsilon_j(x)}_{=0} \right) = \\ &= \frac{1}{N} E_1. \end{aligned}$$

# Bootstrap

Consider the errors unbiased and uncorrelated:

$$\mathbb{E}_x \varepsilon_j(x) = 0;$$

$$\mathbb{E}_x \varepsilon_i(x) \varepsilon_j(x) = 0, \quad i \neq j.$$

The final model averages all predictions:

$$a(x) = \frac{1}{N} \sum_{j=1}^N b_j(x).$$

Error decreased by N times!

$$\begin{aligned} E_N &= \mathbb{E}_x \left( \frac{1}{N} \sum_{j=1}^n b_j(x) - y(x) \right)^2 = \\ &= \mathbb{E}_x \left( \frac{1}{N} \sum_{j=1}^N \varepsilon_j(x) \right)^2 = \\ &= \frac{1}{N^2} \mathbb{E}_x \left( \sum_{j=1}^N \varepsilon_j^2(x) + \underbrace{\sum_{i \neq j} \varepsilon_i(x) \varepsilon_j(x)}_{=0} \right) = \\ &= \frac{1}{N} E_1. \end{aligned}$$

# Bootstrap

Consider the errors ~~unbiased and uncorrelated~~:

Because this is a lie

$$\mathbb{E}_x \varepsilon_j(x) = 0;$$

$$\mathbb{E}_x \varepsilon_i(x) \varepsilon_j(x) = 0, \quad i \neq j.$$

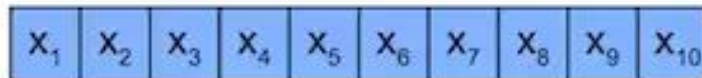
The final model averages all predictions:

$$a(x) = \frac{1}{N} \sum_{j=1}^N b_j(x).$$

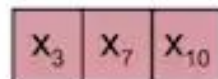
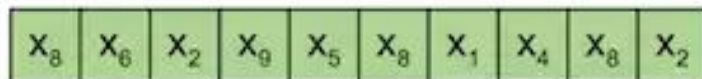
Error decreased by N times!

$$\begin{aligned} E_N &= \mathbb{E}_x \left( \frac{1}{N} \sum_{j=1}^n b_j(x) - y(x) \right)^2 = \\ &= \mathbb{E}_x \left( \frac{1}{N} \sum_{j=1}^N \varepsilon_j(x) \right)^2 = \\ &= \frac{1}{N^2} \mathbb{E}_x \left( \sum_{j=1}^N \varepsilon_j^2(x) + \underbrace{\sum_{i \neq j} \varepsilon_i(x) \varepsilon_j(x)}_{=0} \right) = \\ &= \frac{1}{N} E_1. \end{aligned}$$

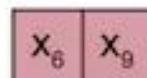
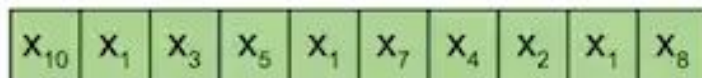
Original Dataset



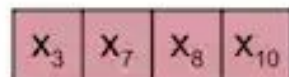
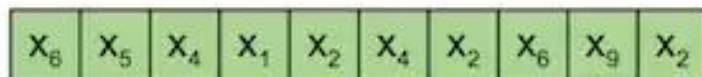
Bootstrap 1



Bootstrap 2



Bootstrap 3



Training Sets

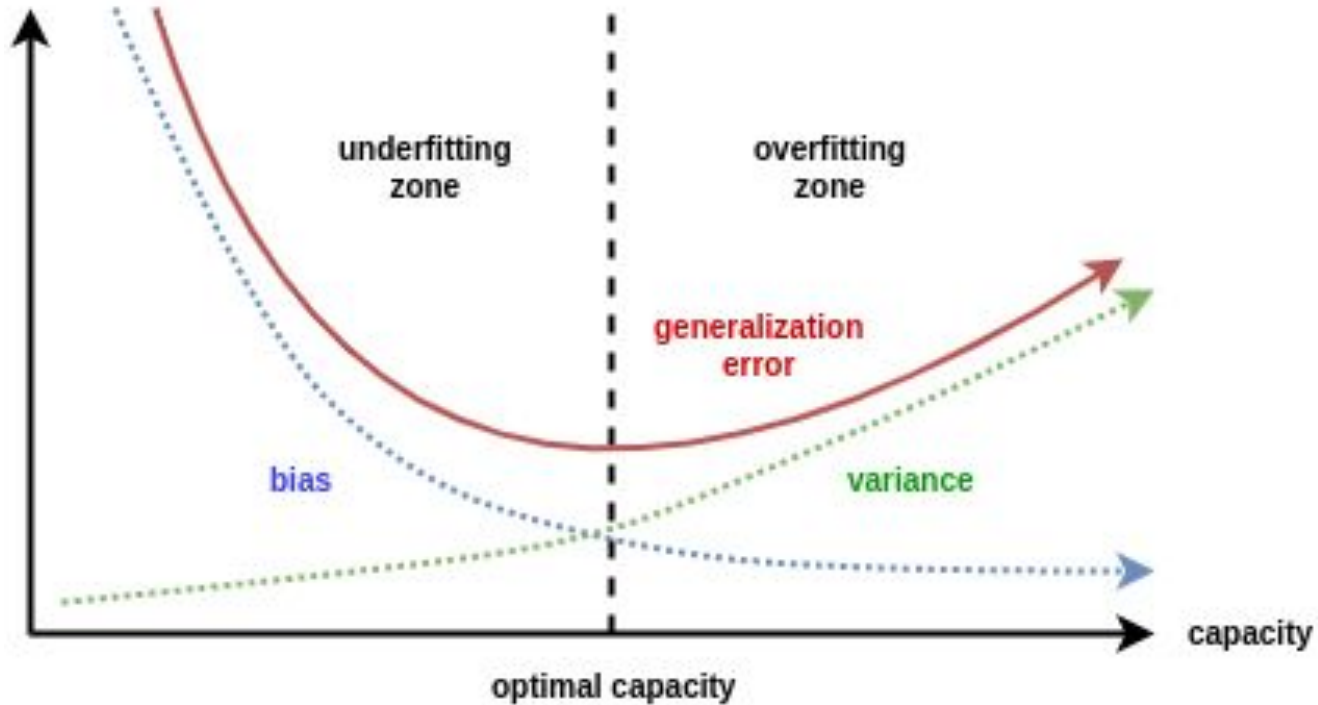
Test Sets



This work by Sebastian Raschka is licensed under a  
Creative Commons Attribution 4.0 International License.

# Bias-variance decomposition

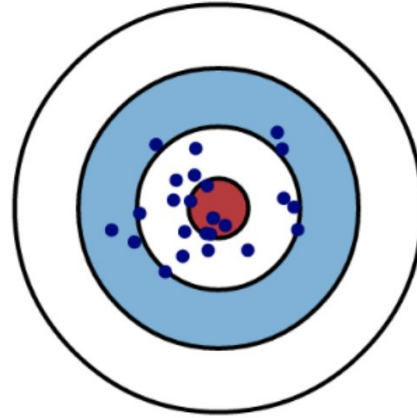
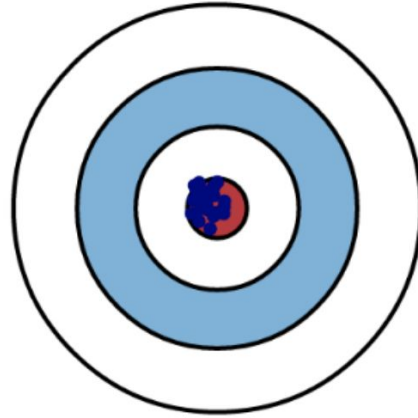
# Bias-variance tradeoff



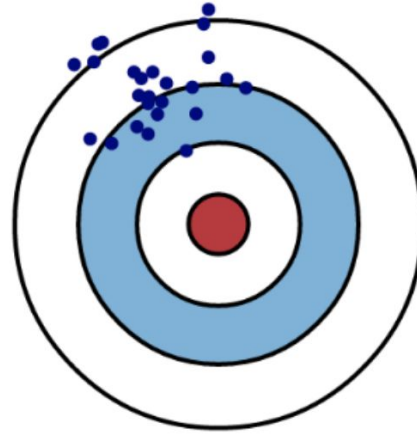
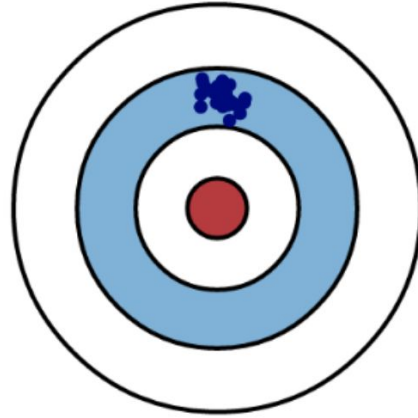
Low Variance

High Variance

Low Bias



High Bias





# Bias-variance decomposition

The dataset  $X = (x_i, y_i)_{i=1}^{\ell}$  with  $y_i \in \mathbb{R}$  for regression problem.

Denote loss function  $L(y, a) = (y - a(x))^2$

The corresponding risk estimation is

$$R(a) = \mathbb{E}_{x,y} \left[ (y - a(x))^2 \right] = \int_{\mathbb{X}} \int_{\mathbb{Y}} p(x, y) (y - a(x))^2 dx dy.$$

Denote  $\mu : (\mathbb{X} \times \mathbb{Y})^\ell \rightarrow \mathcal{A}$ , where  $\mathcal{A}$  is some family of algorithms.

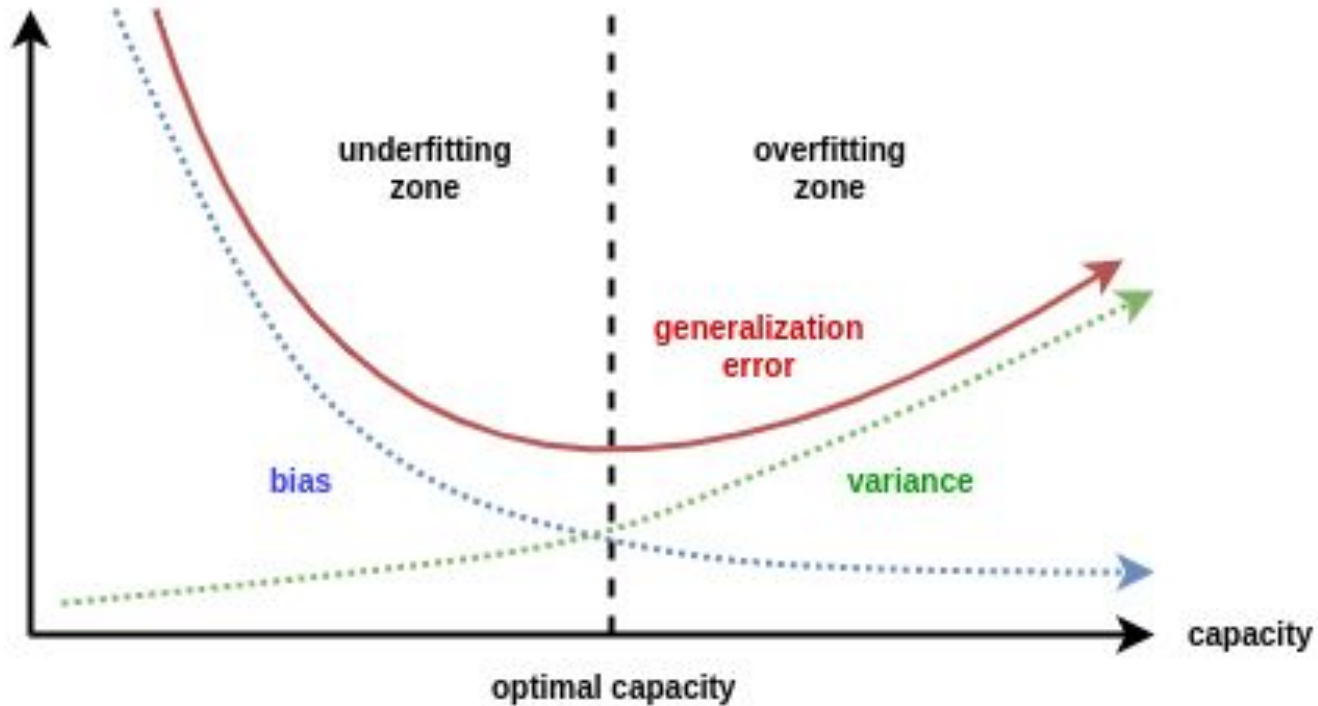
So  $L(\mu) = \mathbb{E}_X \left[ \mathbb{E}_{x,y} \left[ (y - \mu(X)(x))^2 \right] \right]$ , where  $X$  dataset.

$$\begin{aligned}
 L(\mu) = & \underbrace{\mathbb{E}_{x,y} \left[ (y - \mathbb{E}[y | x])^2 \right]}_{\text{noise}} + \\
 & \underbrace{\mathbb{E}_x \left[ (\mathbb{E}_X [\mu(X)] - \mathbb{E}[y | x])^2 \right]}_{\text{bias}} + \underbrace{\mathbb{E}_x \left[ \mathbb{E}_X \left[ (\mu(X) - \mathbb{E}_X [\mu(X)])^2 \right] \right]}_{\text{variance}}.
 \end{aligned}$$

This exact form of bias-variance decomposition is correct for square loss in regression.

However, it is much more general. See extra materials for more exotic cases.

# Bias-variance tradeoff



# Bagging = Bootstrap aggregating

Denote dataset  $\tilde{X}$  bootstrapped from  $X$ .

Denote  $\mu: \tilde{\mu}(X) = \mu(\tilde{X})$ . Let  $b_n(x)$  be basic algorithm.

Denote the ensemble:

$$a_N(x) = \frac{1}{N} \sum_{n=1}^N b_n(x) = \frac{1}{N} \sum_{n=1}^N \tilde{\mu}(X)(x).$$

The **bias** term takes the following form:

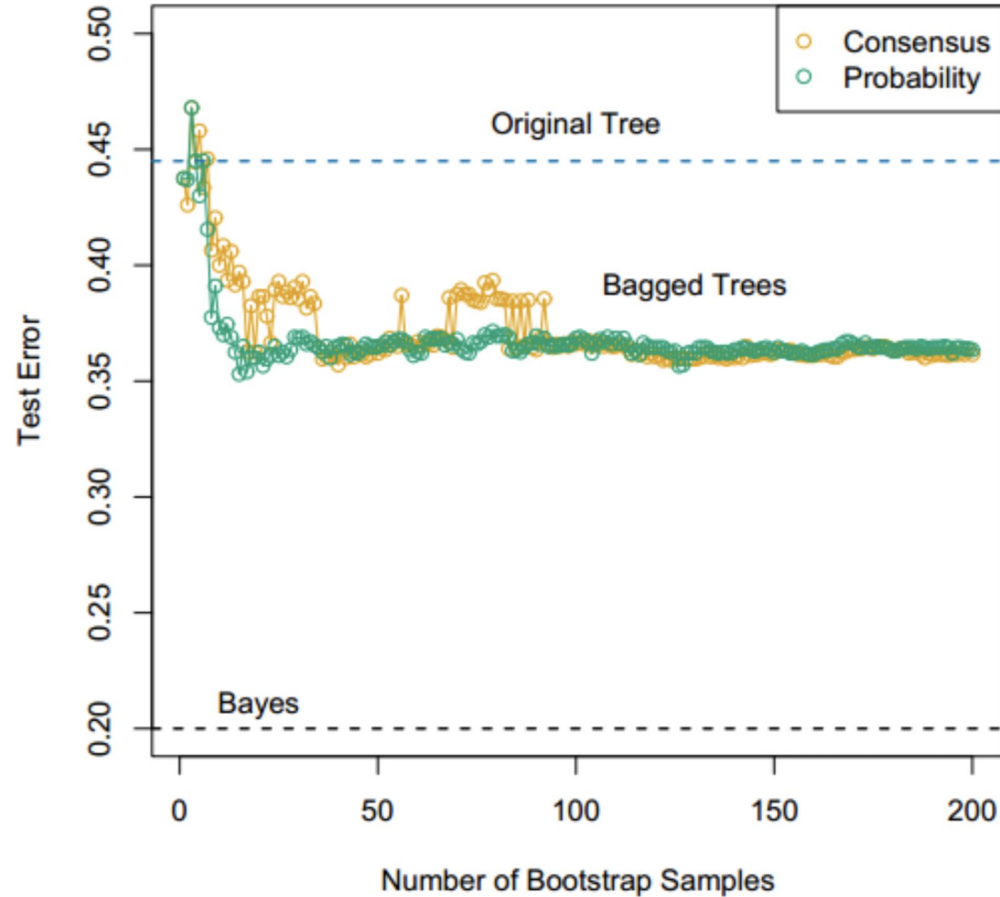
$$\begin{aligned} \mathbb{E}_{x,y} \left[ \left( \mathbb{E}_X \left[ \frac{1}{N} \sum_{n=1}^N \tilde{\mu}(X)(x) \right] - \mathbb{E}[y | x] \right)^2 \right] &= \\ &= \mathbb{E}_{x,y} \left[ \left( \frac{1}{N} \sum_{n=1}^N \mathbb{E}_X [\tilde{\mu}(X)(x)] - \mathbb{E}[y | x] \right)^2 \right] = \\ &= \mathbb{E}_{x,y} \left[ \left( \mathbb{E}_X [\tilde{\mu}(X)(x)] - \mathbb{E}[y | x] \right)^2 \right]. \end{aligned}$$

One algorithm bias

The **variance**:

$$\begin{aligned} & \mathbb{E}_{x,y} \left[ \mathbb{E}_X \left[ \frac{1}{N^2} \sum_{n=1}^N \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right)^2 + \right. \right. \\ & \quad \left. \left. + \frac{1}{N^2} \sum_{n_1 \neq n_2} \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right) \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right) \right] \right] = \\ &= \frac{1}{N^2} \mathbb{E}_{x,y} \left[ \mathbb{E}_X \left[ \sum_{n=1}^N \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right)^2 \right] \right] + \\ & \quad + \frac{1}{N^2} \mathbb{E}_{x,y} \left[ \mathbb{E}_X \left[ \sum_{n_1 \neq n_2} \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right) \times \right. \right. \\ & \quad \left. \left. \times \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right) \right] \right] = \text{One algorithm} \\ &= \frac{1}{N} \mathbb{E}_{x,y} \left[ \mathbb{E}_X \left[ \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right)^2 \right] \right] + \text{variance} * 1/N \\ & \quad + \frac{N(N-1)}{N^2} \mathbb{E}_{x,y} \left[ \mathbb{E}_X \left[ \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right) \times \right. \right. \\ & \quad \left. \left. \times \left( \tilde{\mu}(X)(x) - \mathbb{E}_X [\tilde{\mu}(X)(x)] \right) \right] \right] \end{aligned}$$

# Bagging = Bootstrap aggregating





Bagging = Bootstrap aggregating

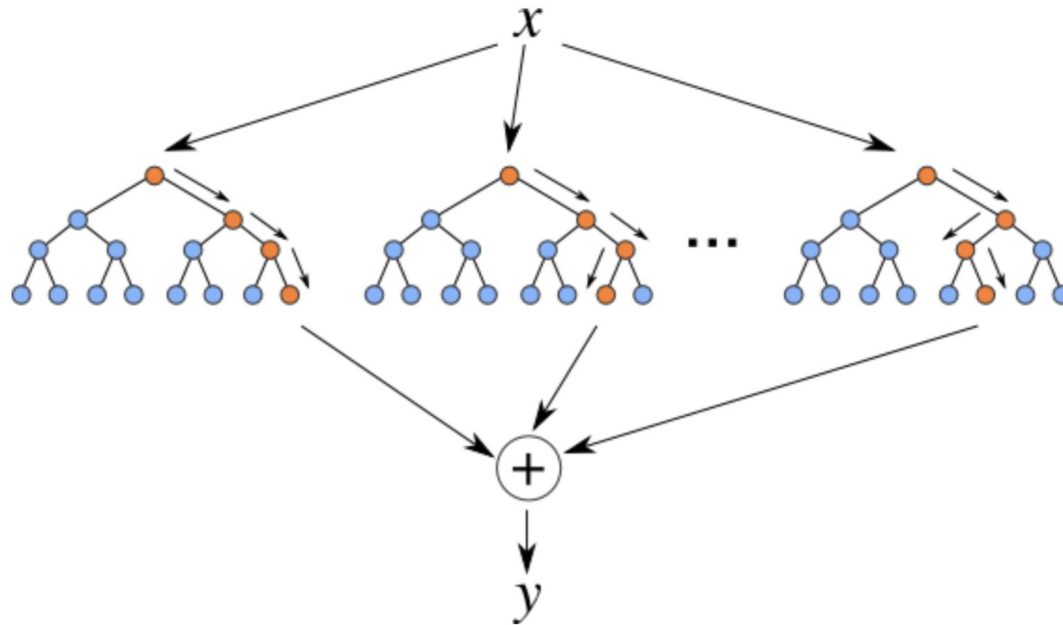
Decreases the variance if the basic algorithms are not correlated.

# RSM - Random Subspace Method

Same approach, but with features.

# Random Forest

Bagging + RSM = Random Forest



- One of the greatest “universal” models.

# Random Forest

- One of the greatest “universal” models.
- There are some modifications: Extremely Randomized Trees, Isolation Forest, etc.
-

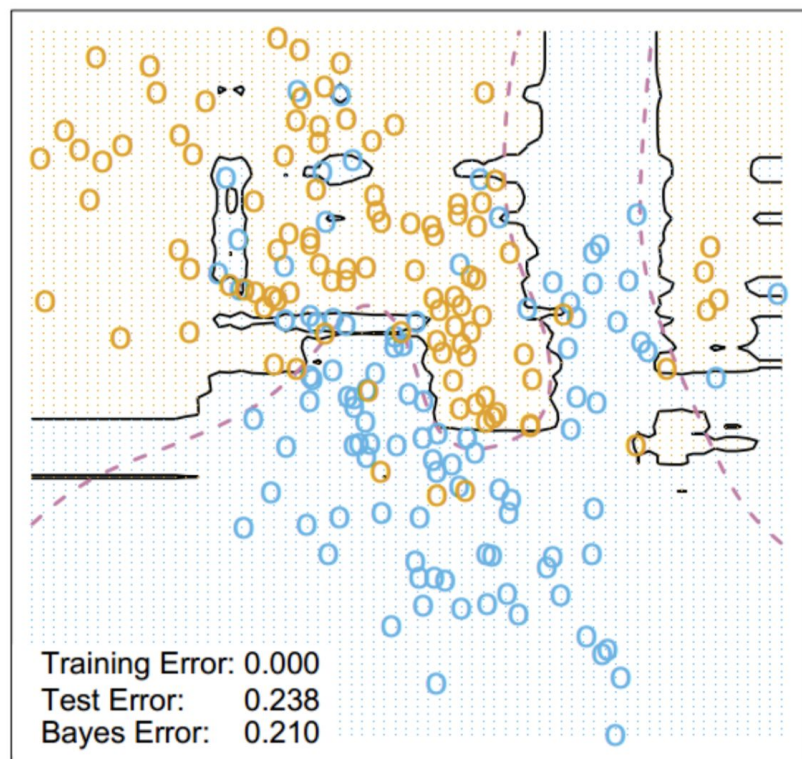
# Random Forest

- One of the greatest “universal” models.
- There are some modifications: Extremely Randomized Trees, Isolation Forest, etc.
- Allows to use train data for validation: OOB

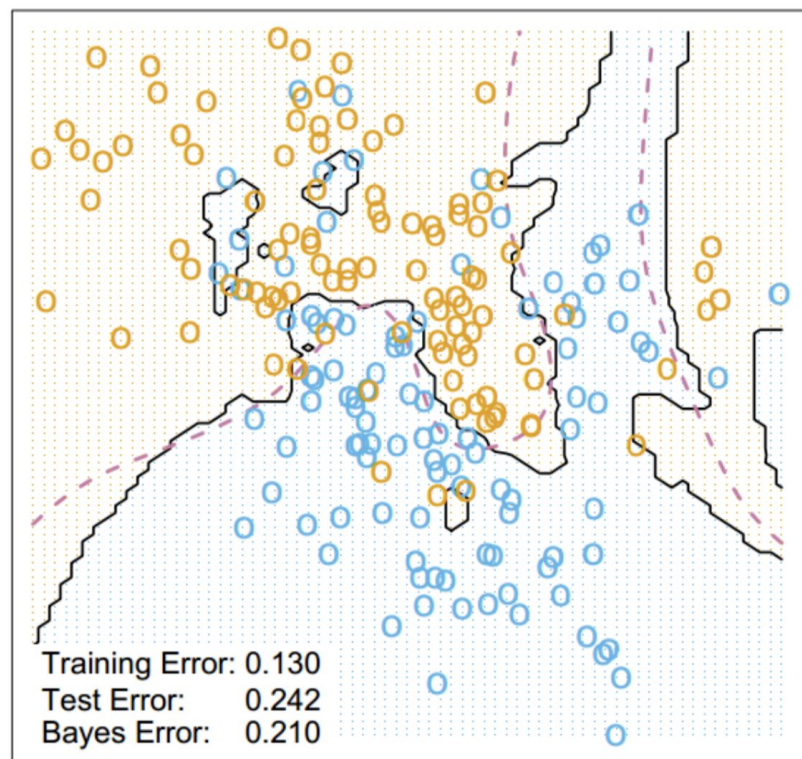
- One of the greatest “universal” models.
- There are some modifications: Extremely Randomized Trees, Isolation Forest, etc.
- Allows to use train data for validation: OOB

$$\text{OOB} = \sum_{i=1}^{\ell} L \left( y_i, \frac{1}{\sum_{n=1}^N [x_i \notin X_n]} \sum_{n=1}^N [x_i \notin X_n] b_n(x_i) \right)$$

Random Forest Classifier

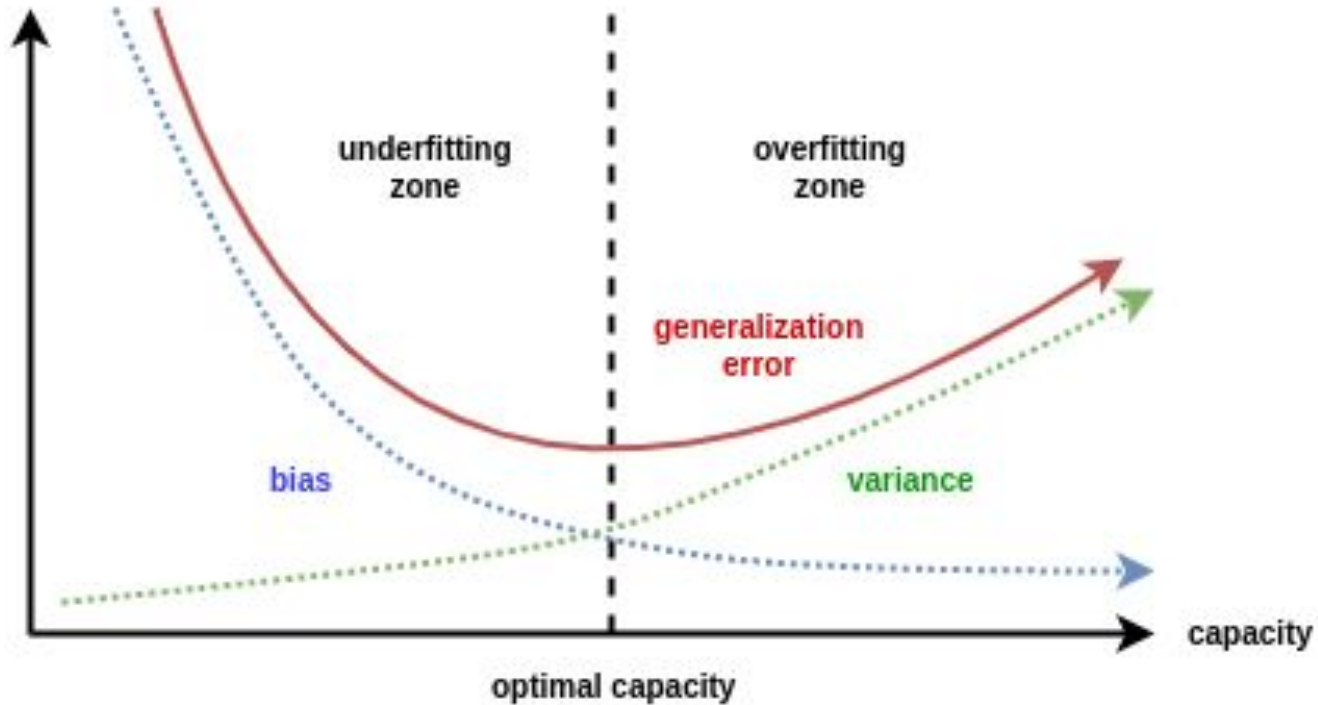


3-Nearest Neighbors



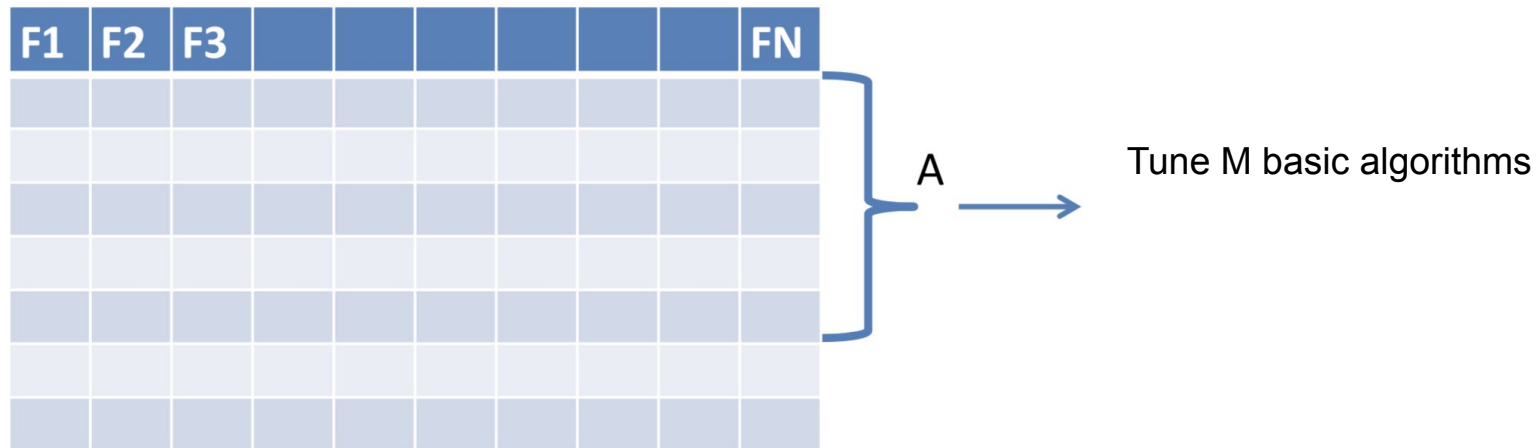


# Bias-variance tradeoff



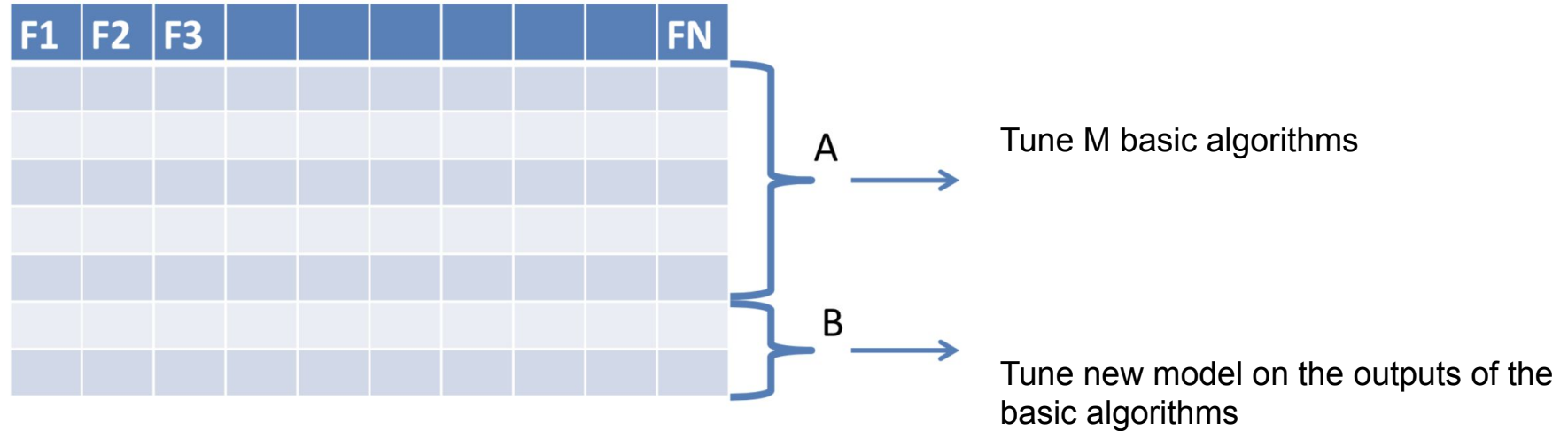
# Stacking

How to build an ensemble from *different* models?



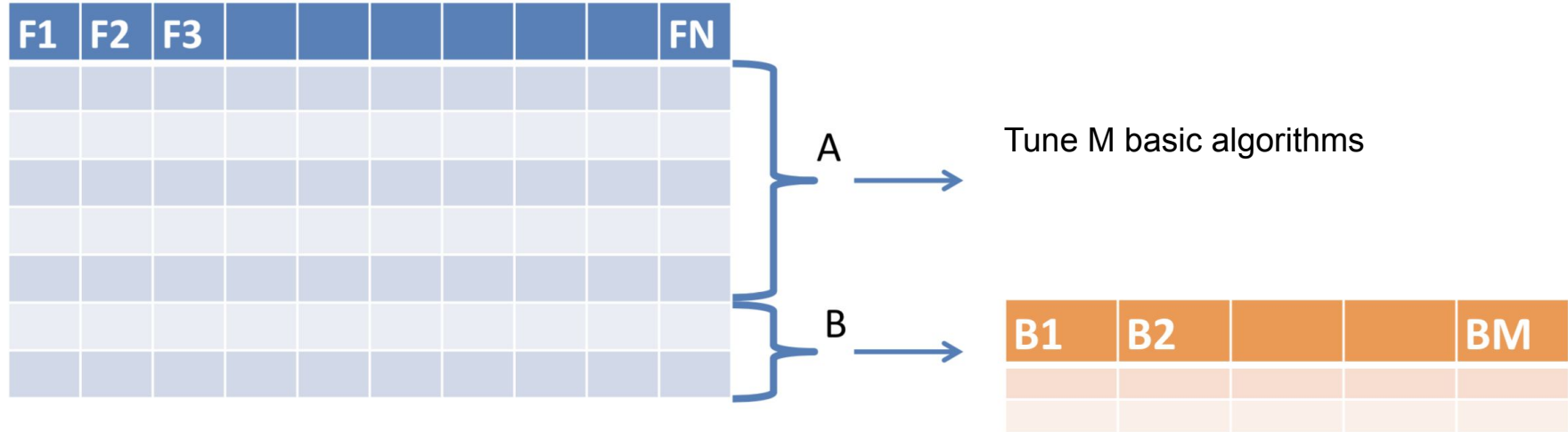
# Stacking

How to build an ensemble from *different* models?



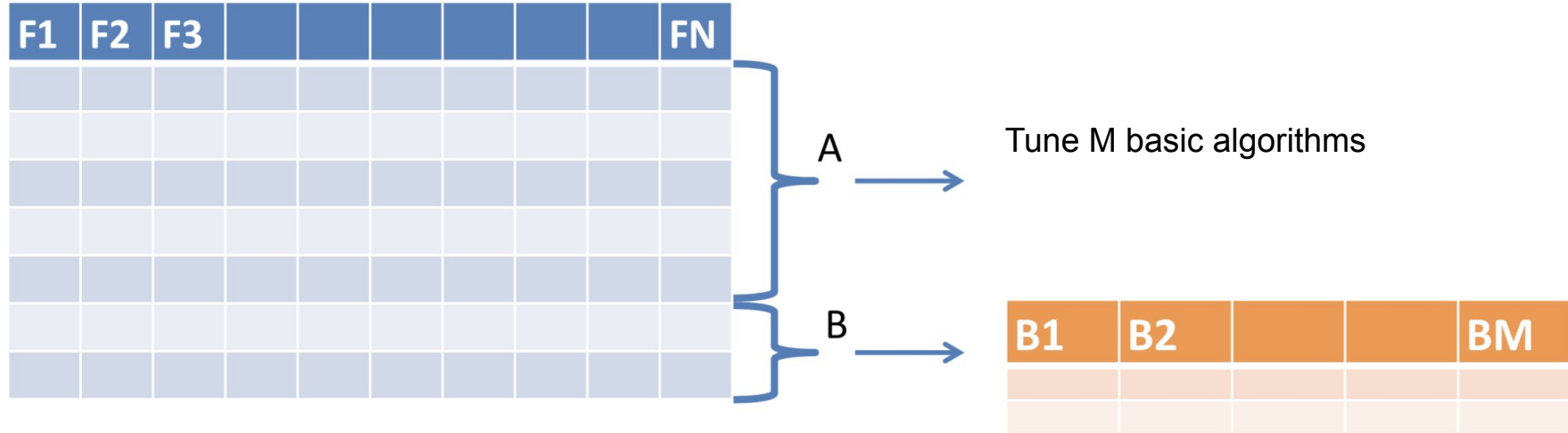
# Stacking

How to build an ensemble from *different* models?



# Stacking

How to build an ensemble from *different* models?



$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

e.g.

How to build an ensemble from *different* models?

How to build an ensemble from *different* models?

- Use different datasets (or datasets parts) for different level models.

How to build an ensemble from *different* models?

- Use different datasets (or datasets parts) for different level models.
- Experiment with different models (linear, trees ensembles, simple networks, etc.)



How to build an ensemble from *different* models?

- Use different datasets (or datasets parts) for different level models.
- Experiment with different models (linear, trees ensembles, simple networks, etc.)
- Or just different GBT ensembles (hola, kaggle :)

Just combine several *strong/complex* models.

Weights should sum up to 1  
and come from [0; 1]

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

Just combine several *strong/complex* models.

Weights should sum up to 1  
and come from [0; 1]

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

- Simple and intuitive ensembling method

Just combine several *strong/complex* models.

Weights should sum up to 1  
and come from [0; 1]

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

- Simple and intuitive ensembling method.
- Finding optimal weights could be tricky.

Just combine several *strong/complex* models.

Weights should sum up to 1  
and come from  $[0; 1]$

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

- Simple and intuitive ensembling method.
- Finding optimal weights could be tricky.
- Linear composition is not always enough.

**Boosting** is coming next time.  
Stay tuned.