Lecture 1
Recap, basic ensembles

Vladislav Goncharenko

Moscow, 2021

Outline

1. Linear models
2. Classification metrics
3. Decision trees
4. Bootstrap
5. Bias-Variance tradeoff
6. Bagging

Linear models recap

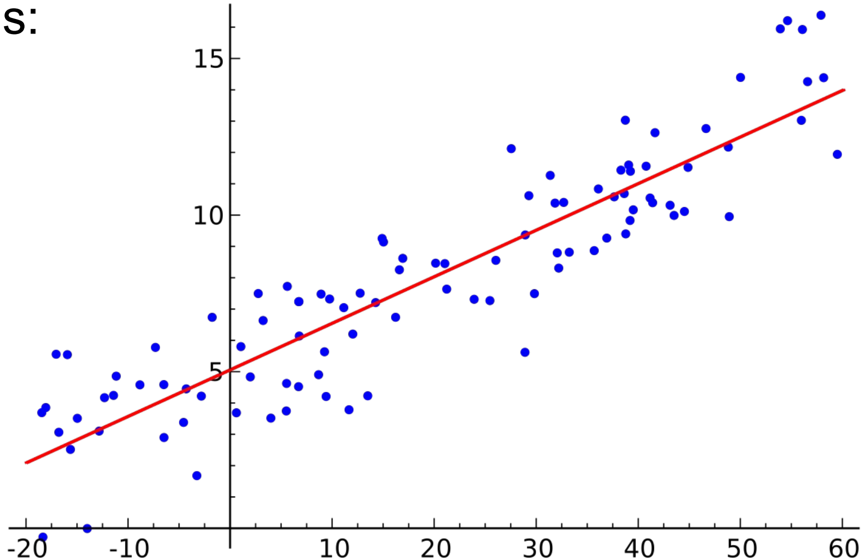
Linear models

$$Y = X_1 + X_2 + X_3$$

Dependent Variable	Independent Variable
Outcome Variable	Predictor Variable
Response Variable	Explanatory Variable

Linear models

- Predictive models:



Estimated (or predicted) Y value for observation i

Estimate of the regression intercept

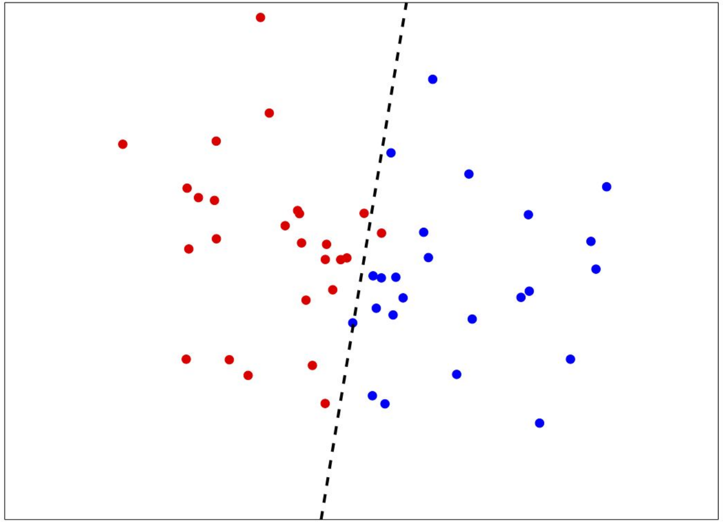
Estimate of the regression slope

Value of X for observation i

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

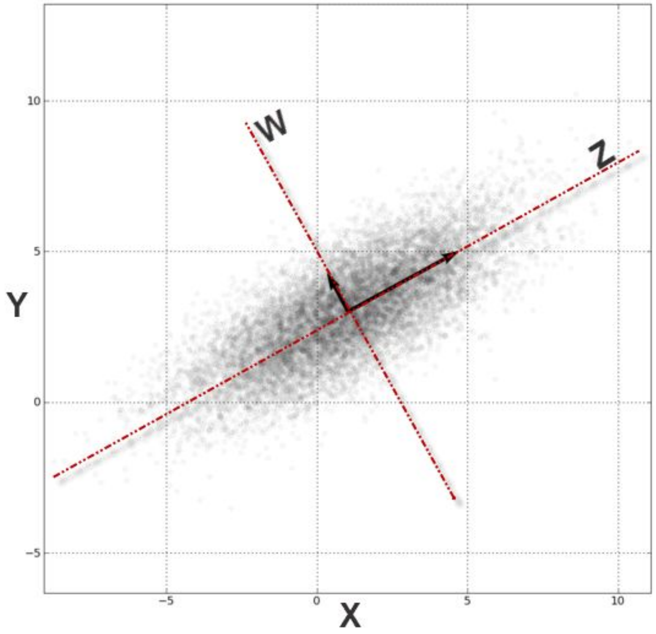
Linear models

- Predictive models:
- Classification models:



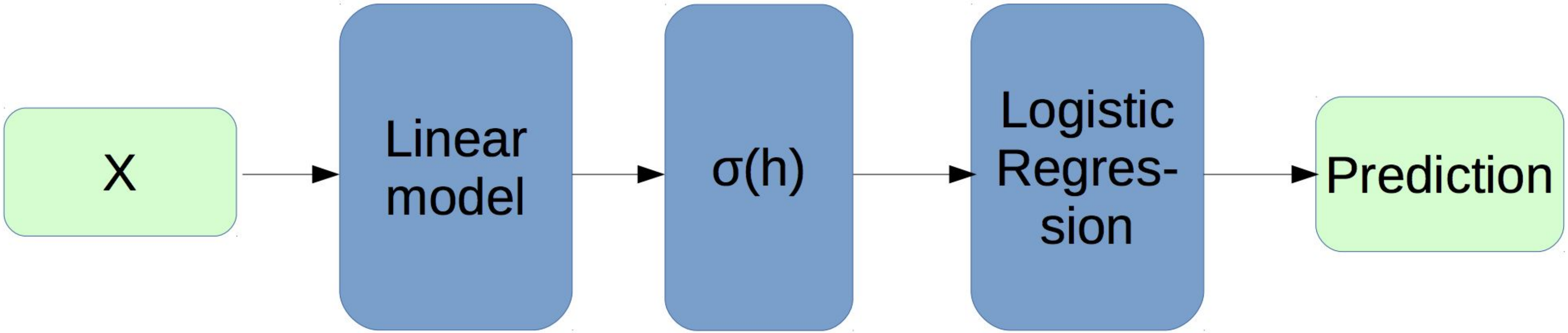
Linear models

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



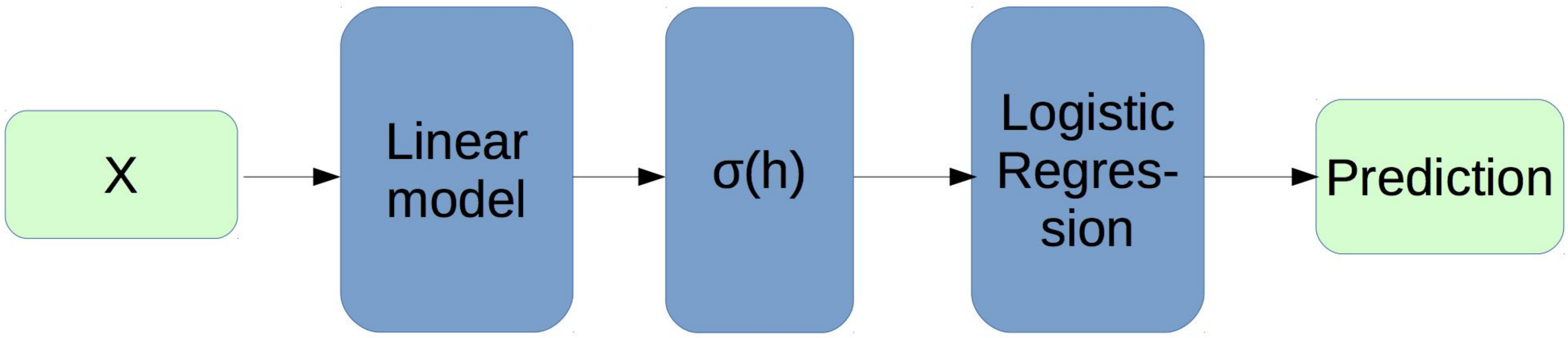
Linear models

- 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

Accuracy

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

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

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

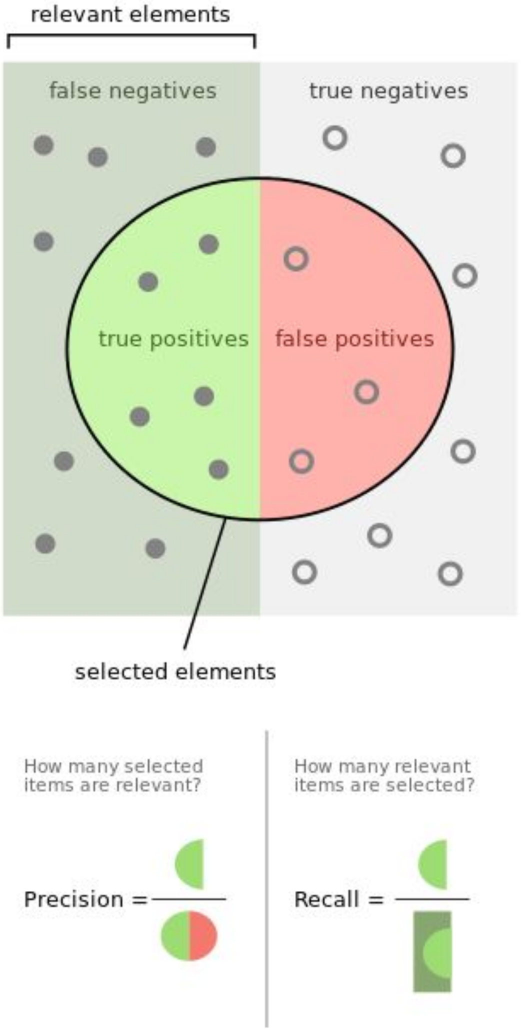
Precision and recall

		Actual Class	
		Yes	No
Predicted Class	Yes	**T**rue **P**ositive	**F**alse **P**ositive
	No	**F**alse **N**egative	**T**rue **N**egative

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

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

Precision and recall



		Actual Class	
		Yes	No
Predicted Class	Yes	**T**rue **P**ositive	**F**alse **P**ositive
	No	**F**alse **N**egative	**T**rue **N**egative

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

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

F-score

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

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

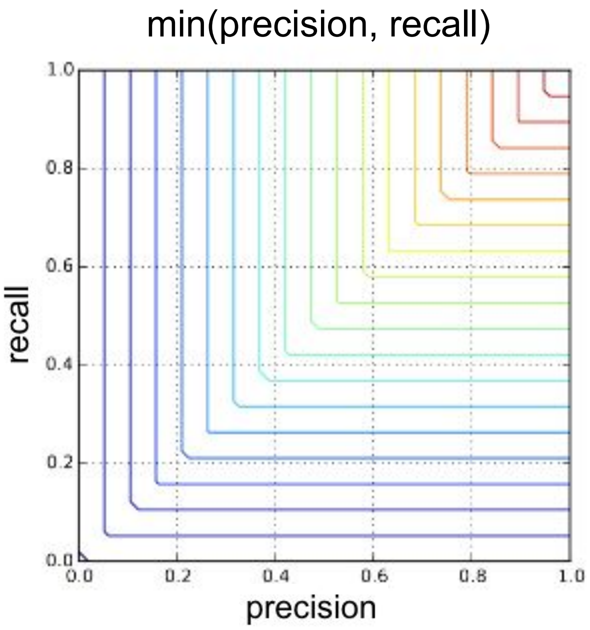
F-score

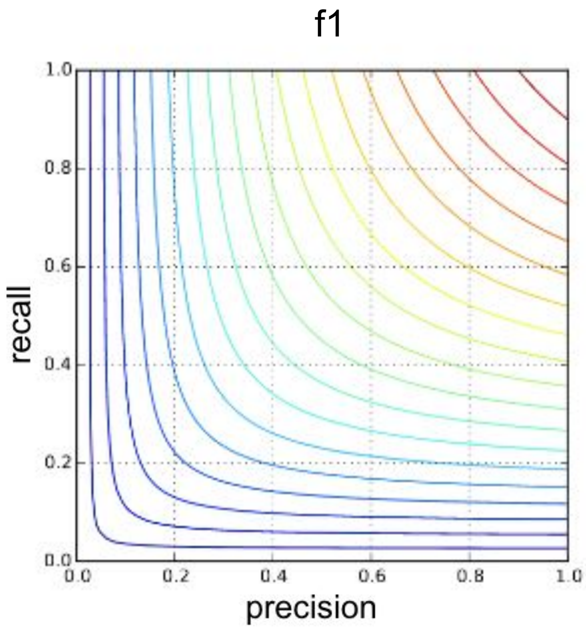
Harmonic mean of precision and recall.
Closer to the smallest one.

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

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

F-score


min(precision, recall)
recall
precision
0.0
0.2
0.4
0.6
0.8
1.0


f1
recall
precision
0.0
0.2
0.4
0.6
0.8
1.0

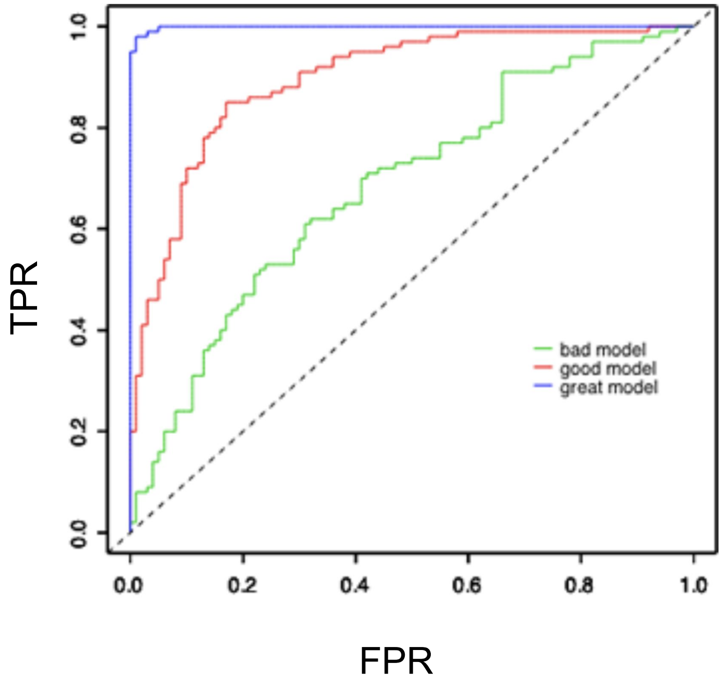
ROC - receiver operating characteristic

		Actual Class	
		Yes	No
Predicted Class	Yes	**T**rue **P**ositive	**F**alse **P**ositive
	No	**F**alse **N**egative	**T**rue **N**egative

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

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

ROC - receiver operating characteristic

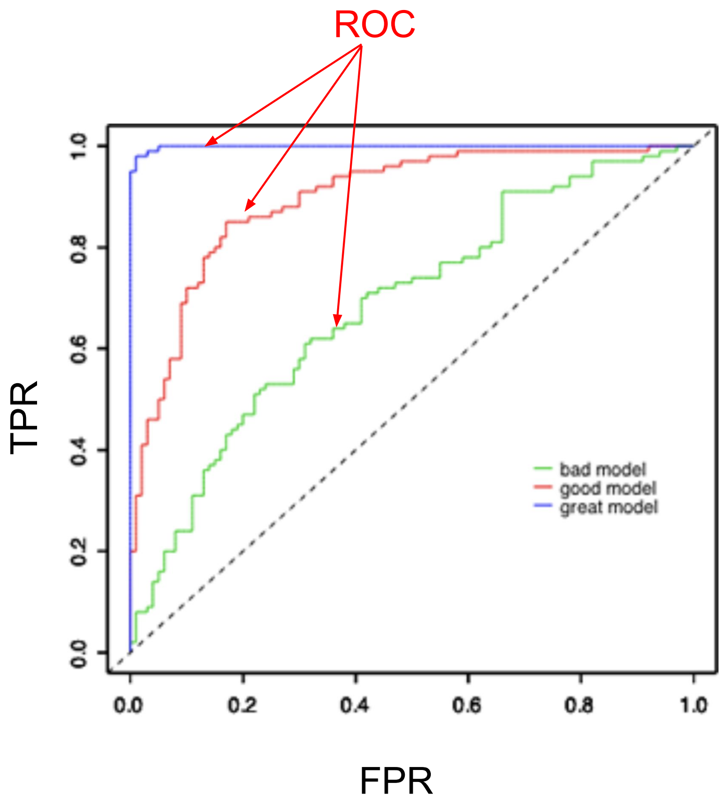


		Actual Class	
		Yes	No
Predicted Class	Yes	**T**rue **P**ositive	**F**alse **P**ositive
	No	**F**alse **N**egative	**T**rue **N**egative

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

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

ROC

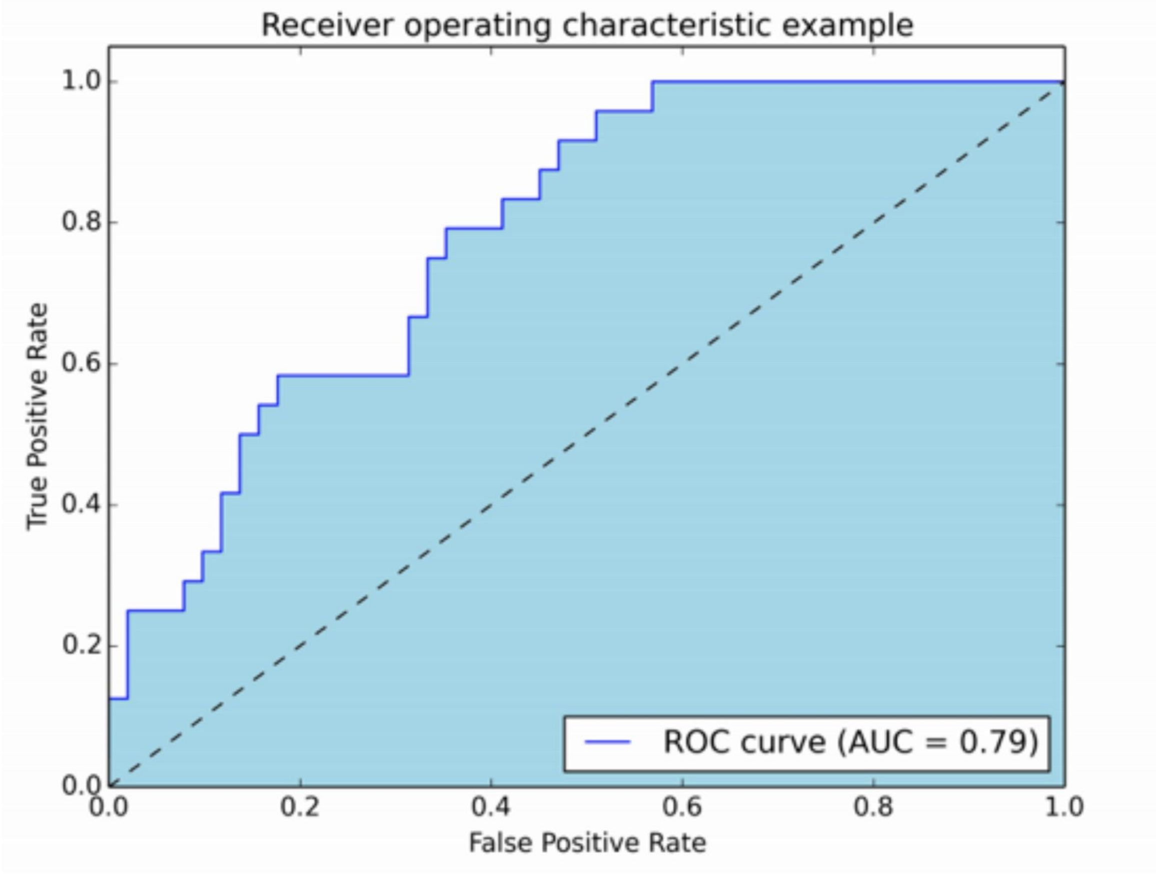


		Actual Class	
		Yes	No
Predicted Class	Yes	**T**rue **P**ositive	**F**alse **P**ositive
	No	**F**alse **N**egative	**T**rue **N**egative

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

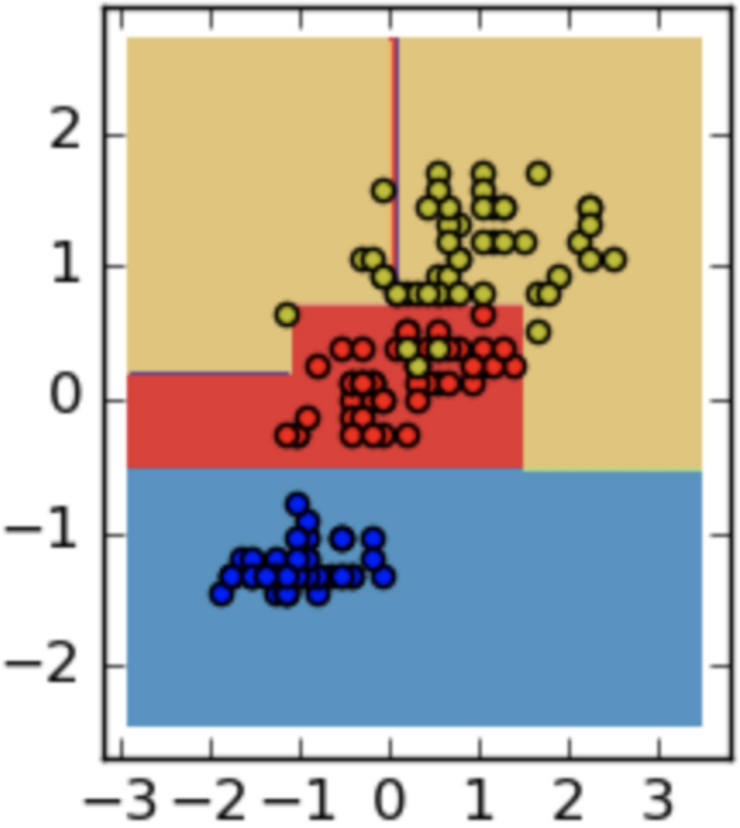
$$FPR = \frac{False\ positives}{False\ positives + True\ negatives}.$$

ROC-AUC - area under curve



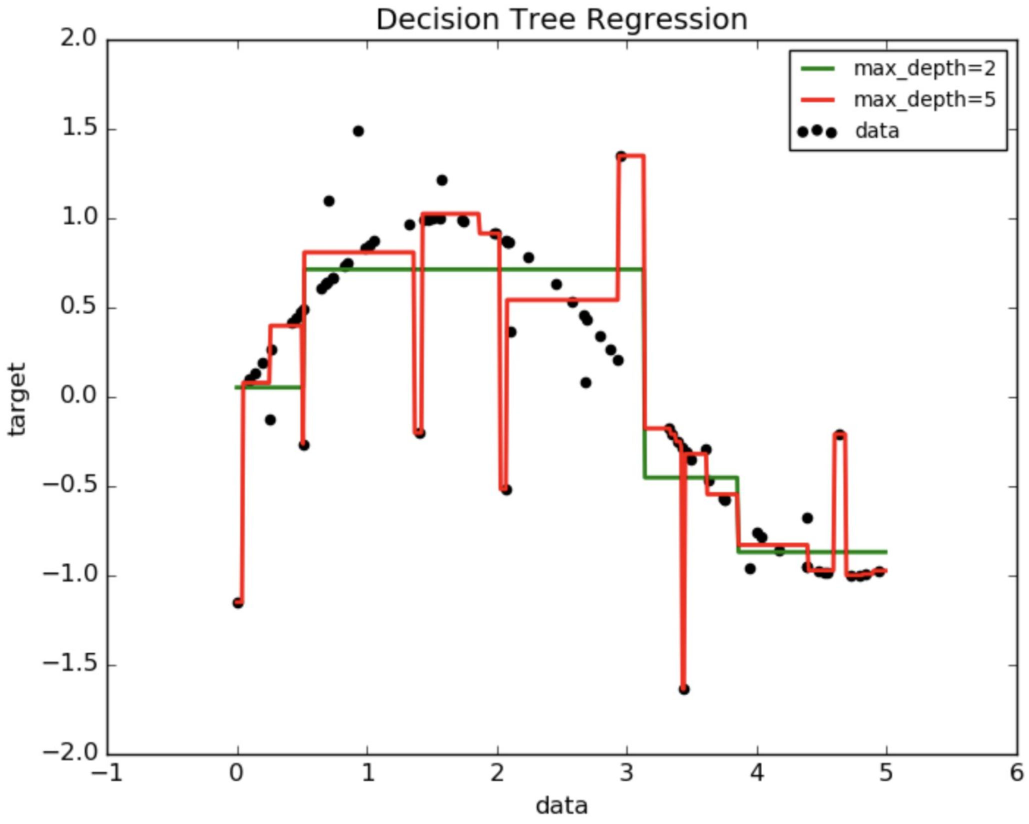
Decision trees

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

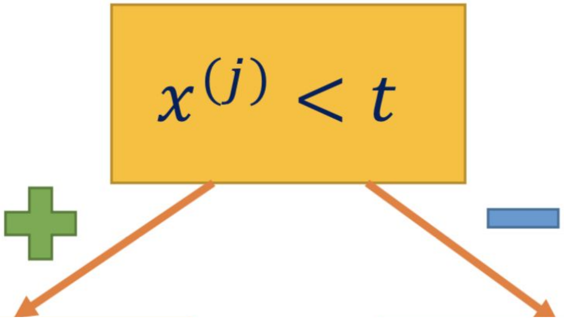
Every leaf corresponds to some constant.

Constructing decision trees

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

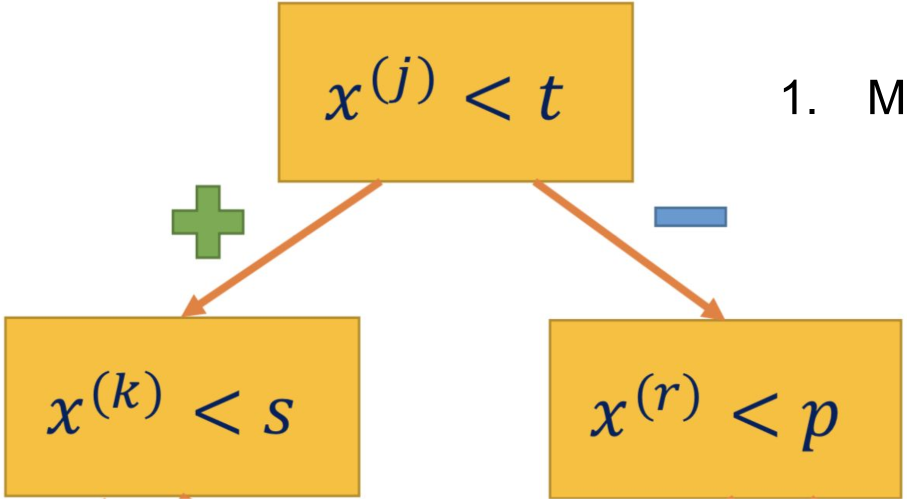
1. Make a split

Constructing decision trees



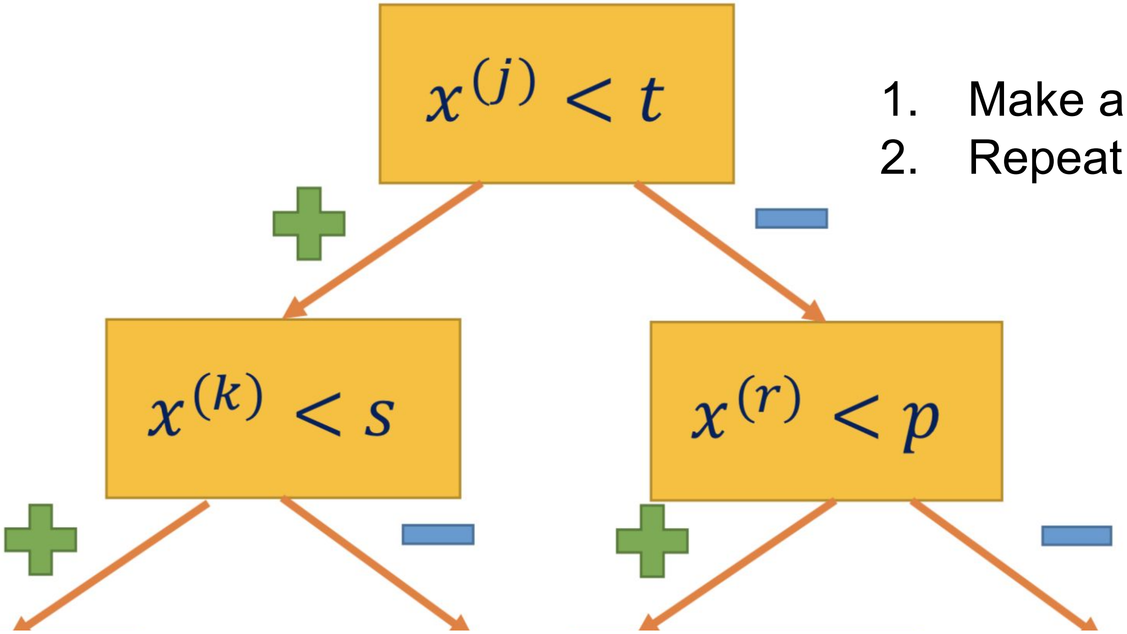
1. Make a split

Constructing decision trees



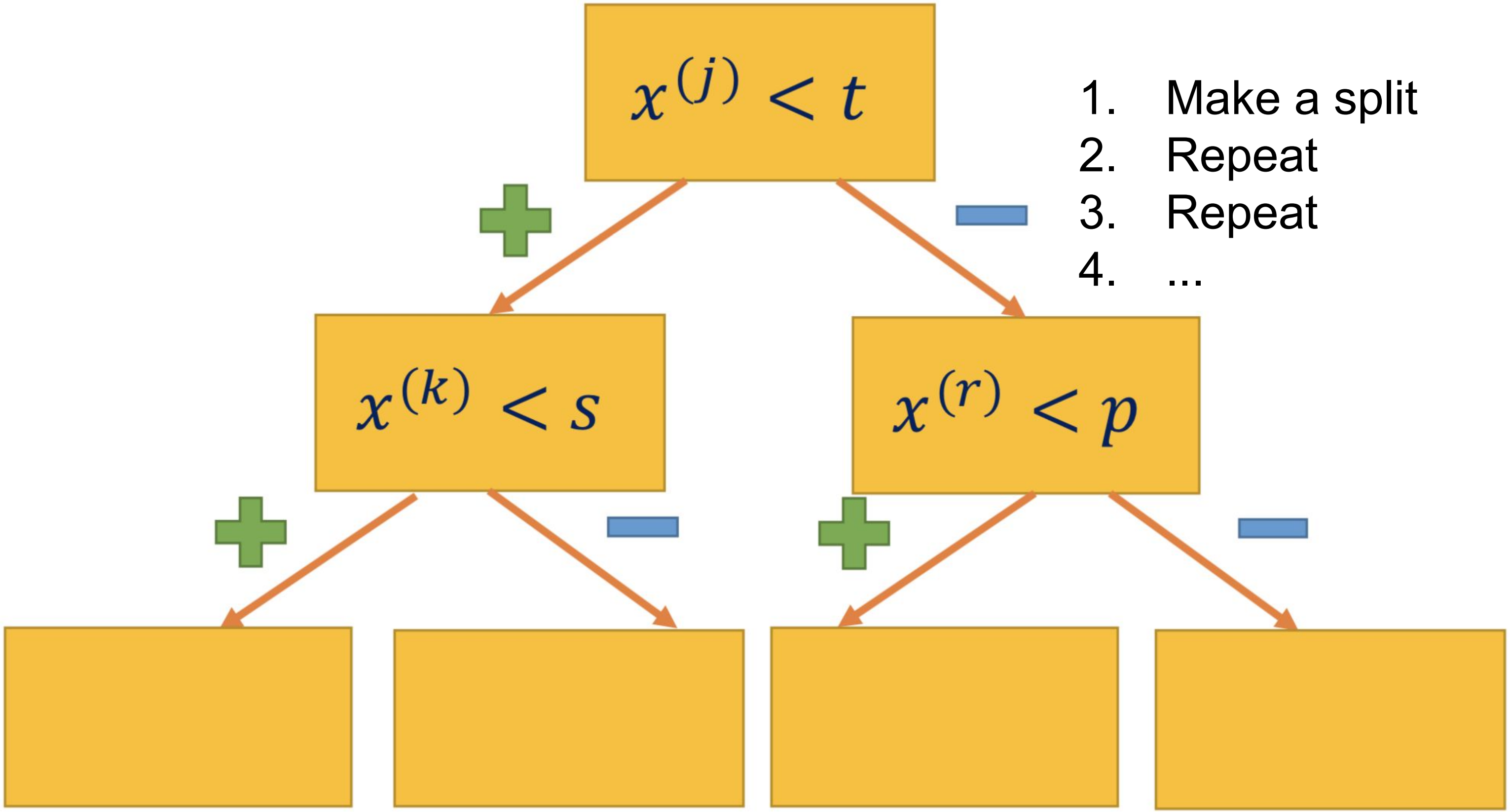
1. Make a split

Constructing decision trees



1. Make a split
2. Repeat

Constructing decision trees



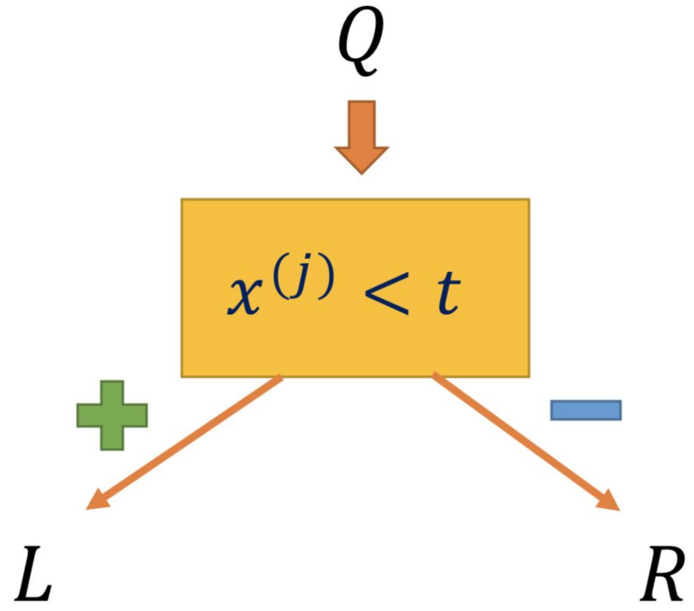
1. Make a split
2. Repeat
3. Repeat
4. ...

How to split data properly?

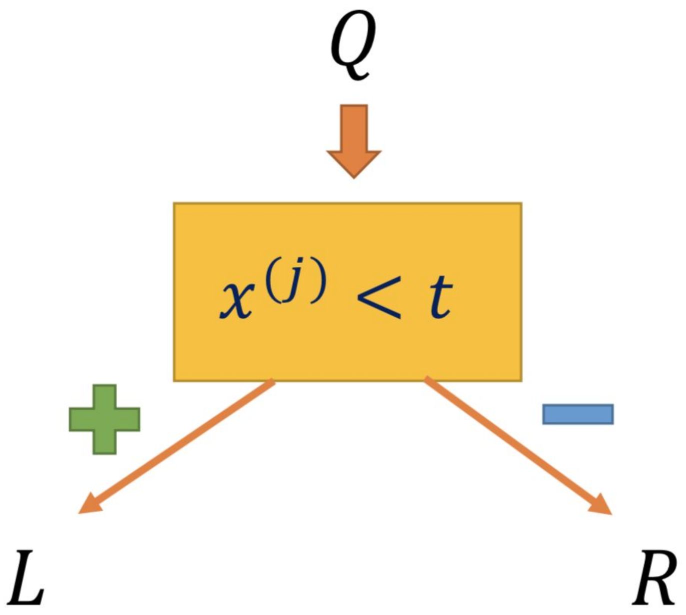
Q

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

How to split data properly?

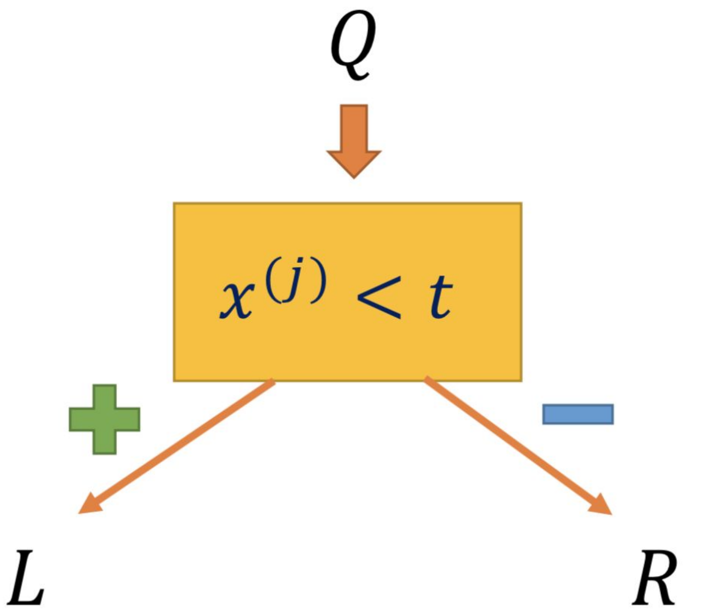


How to split data properly?



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

How to split data properly?



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

What is H()?

Information criteria

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

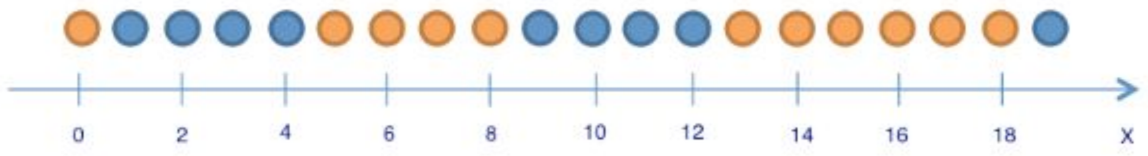
1. Misclassification criteria: $$H(R) = 1 - \max\{p_0, p_1\}$$

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

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

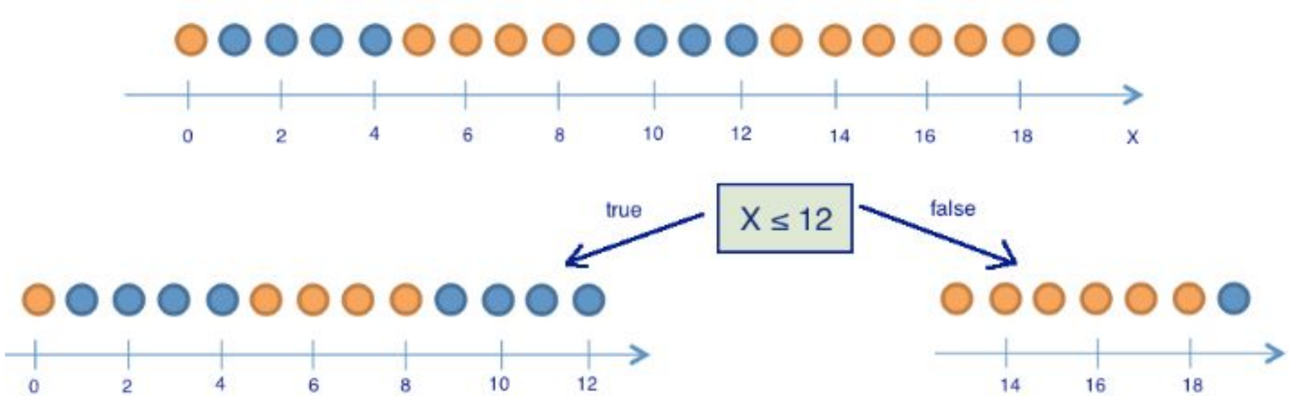
Information criteria

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

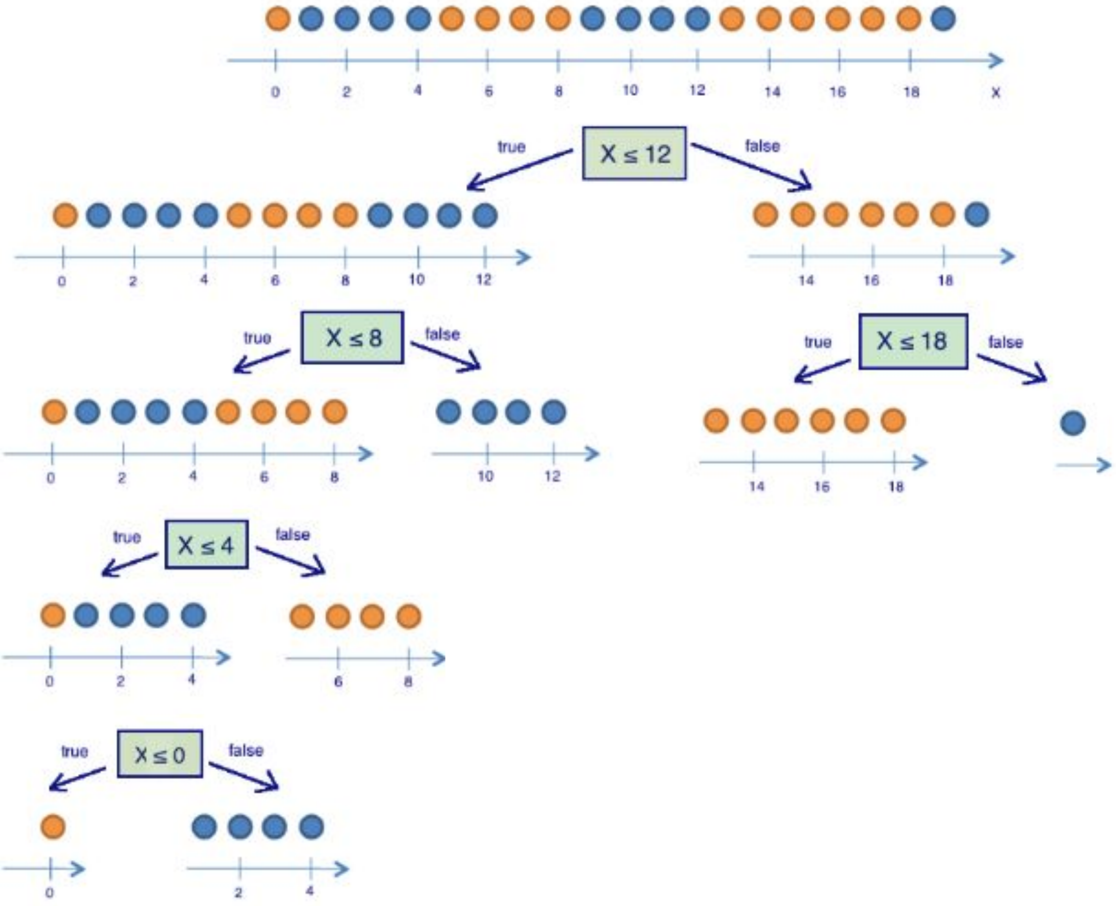


Information criteria

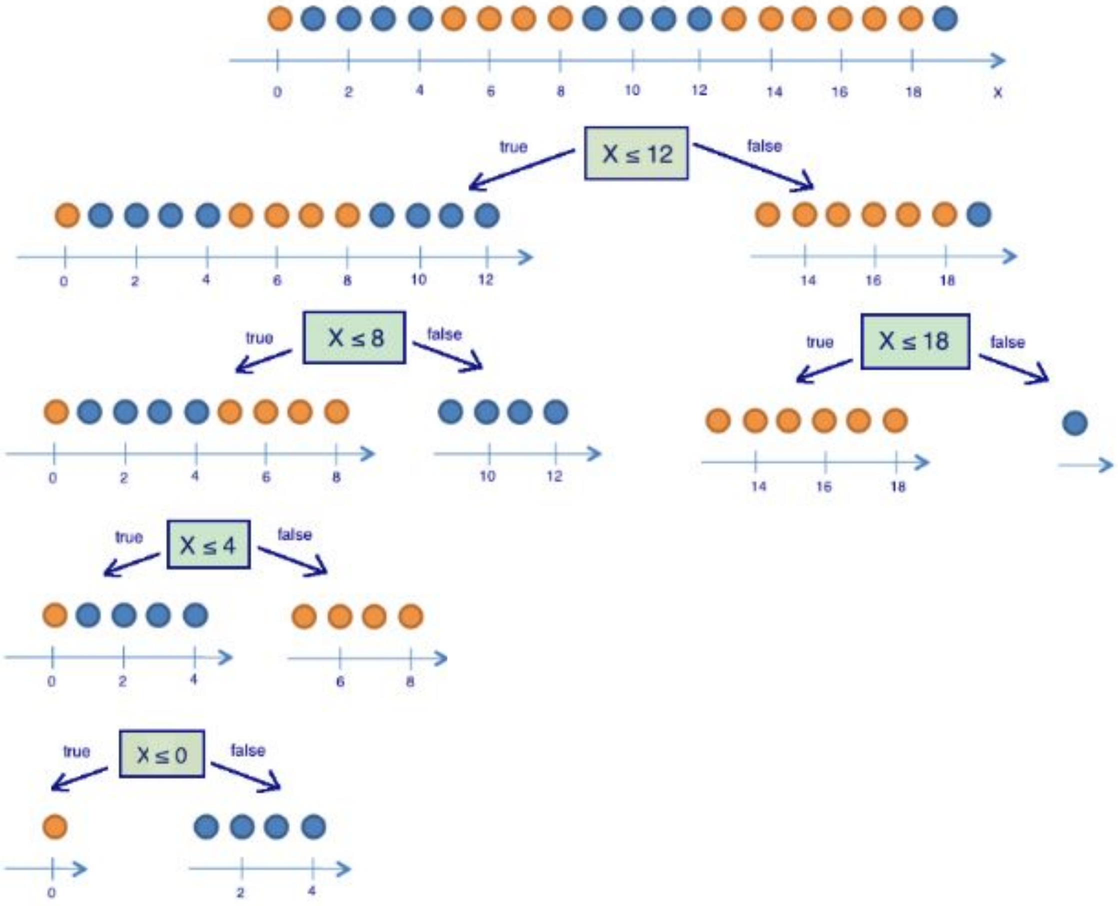
H(R) is measure of “heterogeneity” of our data.
Consider binary classification problem:



Information criteria: Entropy



source: https://habr.com/ru/company/ods/blog/322534/

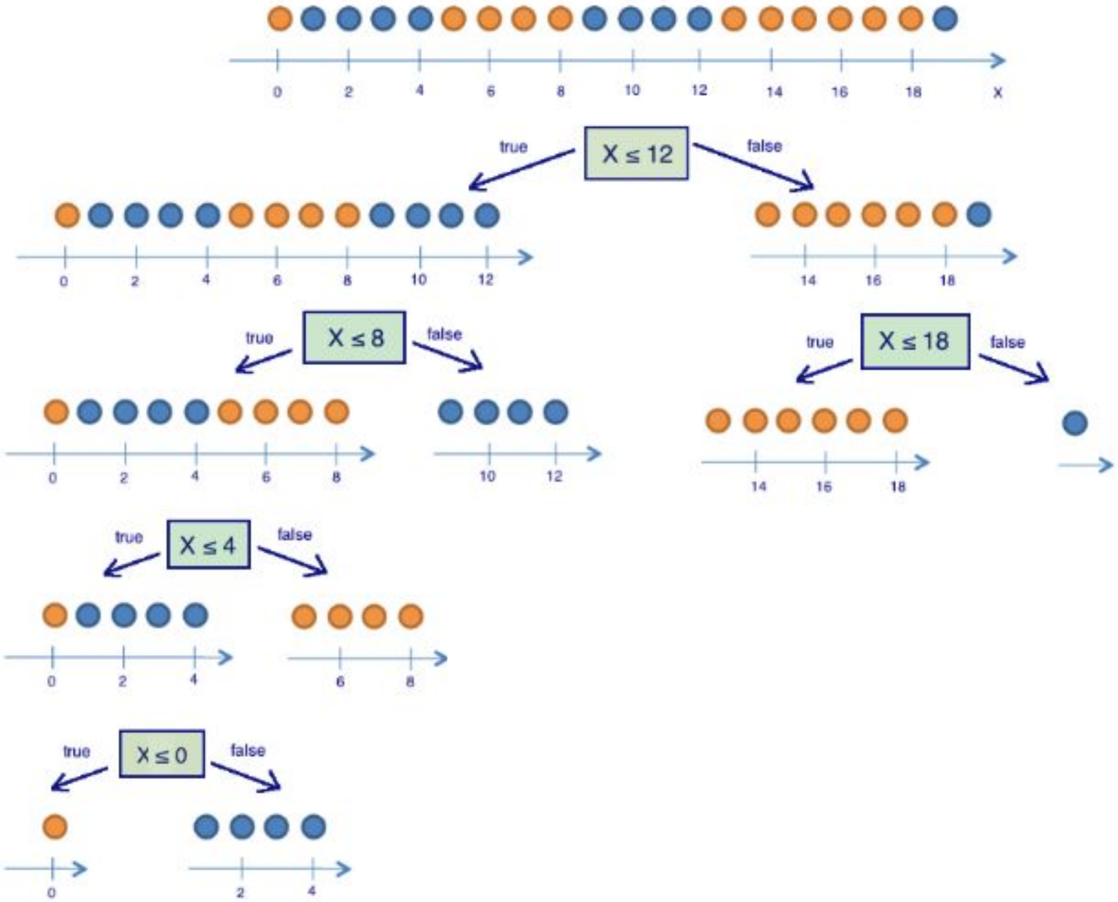


Information criteria: Entropy

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

source: https://habr.com/ru/company/ods/blog/322534/

Information criteria: Entropy



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

In binary case N = 2

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

source: https://habr.com/ru/company/ods/blog/322534/

Information criteria

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: $$H(R) = 1 - \sum_k (p_k)^2$$

Information criteria

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

1. Mean squared error

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

Information criteria

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?

Information criteria

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

Bootstrap

Consider dataset X containing N objects.

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

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

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

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

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:

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

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:

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

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

Error decreased by N times!

Bootstrap

Consider the errors ~~unbiased and uncorrelated~~:

Because this is a lie

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

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

The final model averages all predictions:

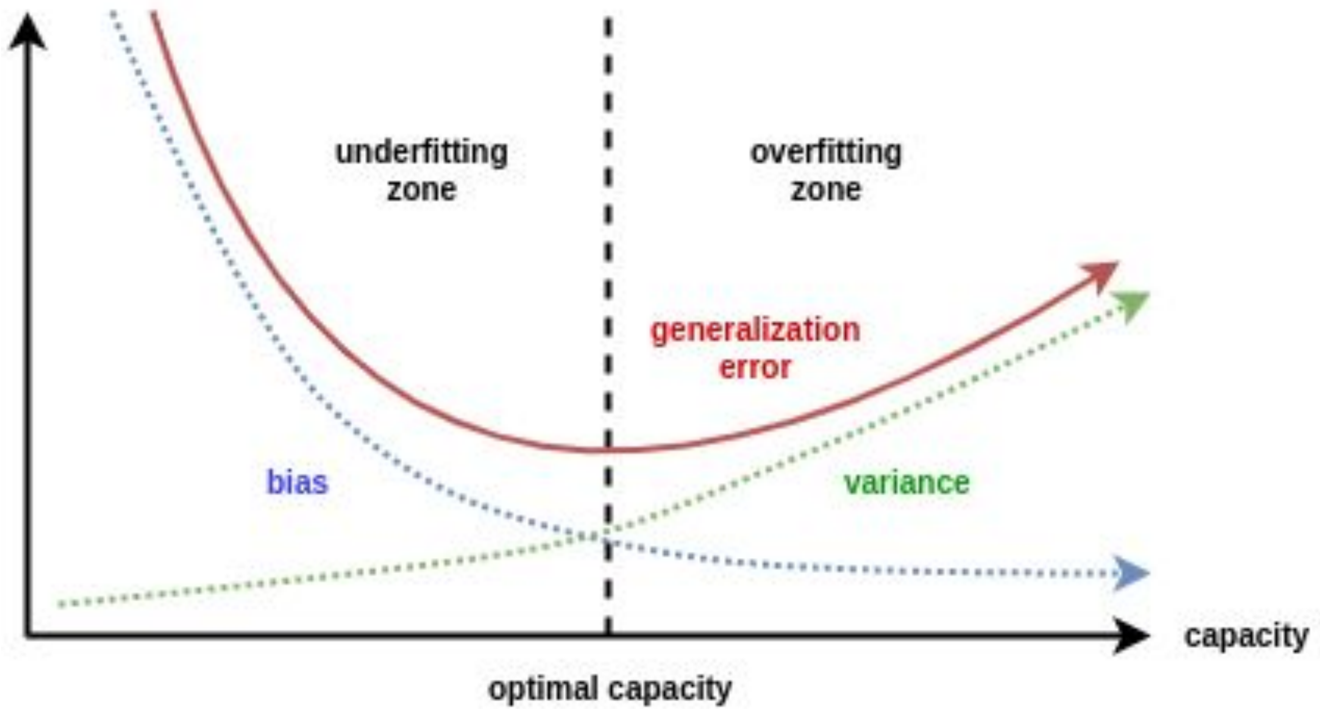
$$a(x) = \frac{1}{N} \sum_{j=1}^{N} b_j(x).$$

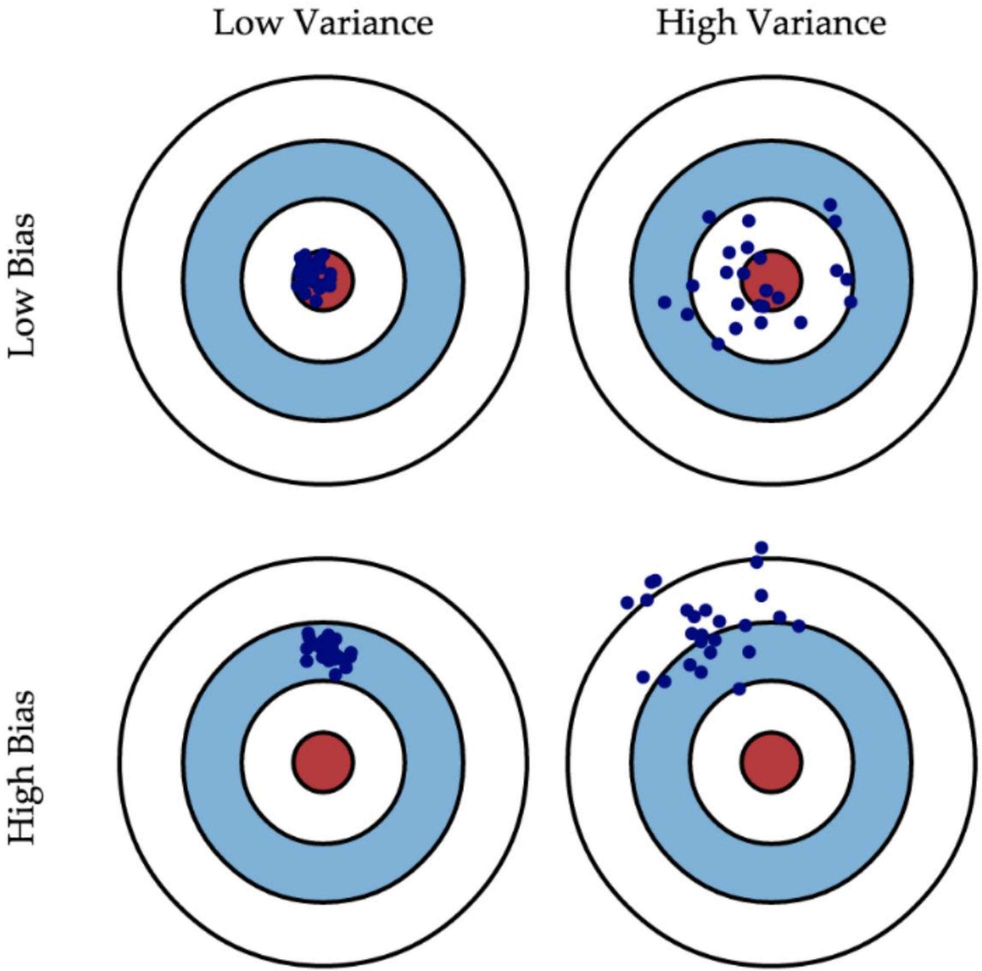
Error decreased by N times!


Original Dataset
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10}
Bootstrap 1
x_8 x_6 x_2 x_9 x_5 x_8 x_1 x_4 x_8 x_2
x_3 x_7 x_{10}
Bootstrap 2
x_{10} x_1 x_3 x_5 x_1 x_7 x_4 x_2 x_1 x_8
x_6 x_9
Bootstrap 3
x_6 x_5 x_4 x_1 x_2 x_4 x_2 x_6 x_9 x_2
x_3 x_7 x_8 x_{10}
Training Sets
Test Sets

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 $L(y, a) = \big(y - a(x)\big)^2$

The corresponding risk estimation is

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

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

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

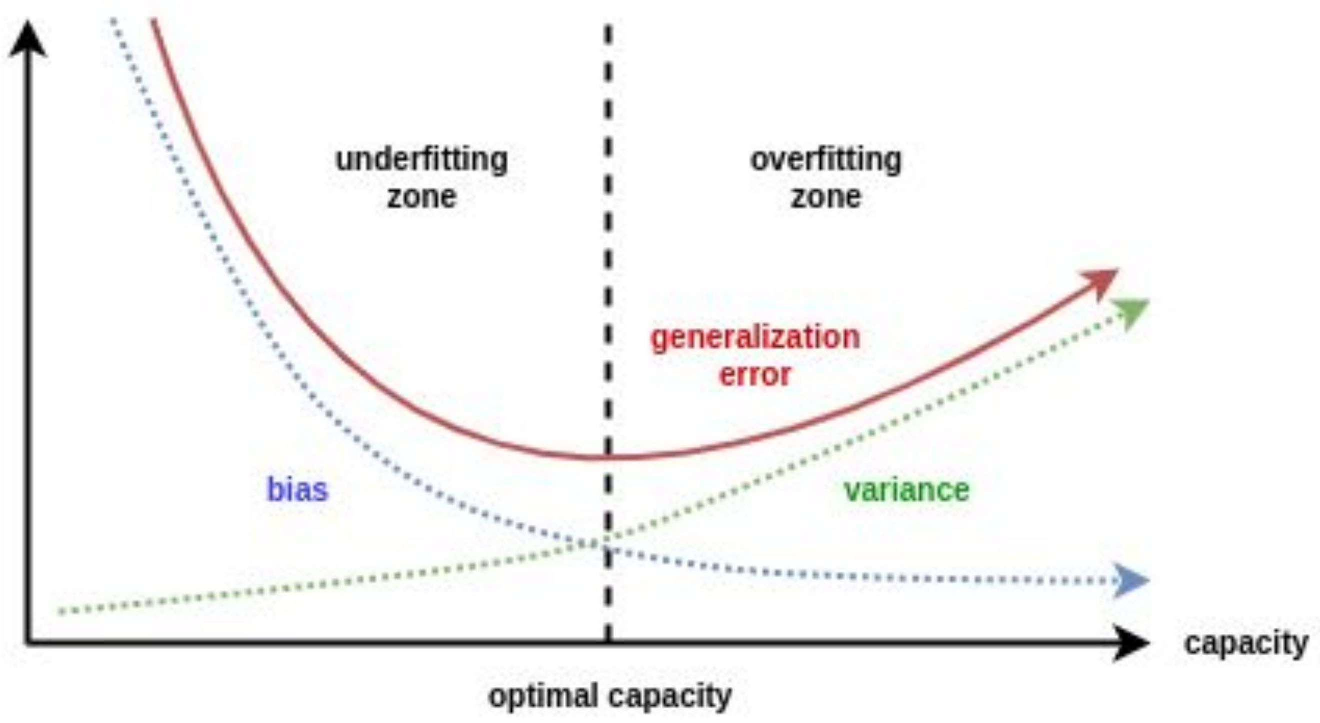
$$L(\mu) = \underbrace{\mathbb{E}_{x,y}\left[\left(y - \mathbb{E}[y \mid x]\right)^2\right]}_{\text{noise}} +$$

$$+ \underbrace{\mathbb{E}_x\left[\left(\mathbb{E}_X\left[\mu(X)\right] - \mathbb{E}[y \mid x]\right)^2\right]}_{\text{bias}} + \underbrace{\mathbb{E}_x\left[\mathbb{E}_X\left[\left(\mu(X) - \mathbb{E}_X\left[\mu(X)\right]\right)^2\right]\right]}_{\text{variance}}.$$

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 μ: $\tilde{\mu}(X) = \mu(\tilde{X})$. 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:

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

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

$$=\boxed{\mathbb{E}_{x,y}\left[\left(\mathbb{E}_X\left[\tilde{\mu}(X)(x)\right]-\mathbb{E}[y\mid x]\right)^2\right]}.$$

One algorithm bias

The variance:

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

$$\left.\left. + \frac{1}{N^2}\sum_{n_1 \neq n_2}\left(\tilde{\mu}(X)(x) - \mathbb{E}_X\left[\tilde{\mu}(X)(x)\right]\right)\left(\tilde{\mu}(X)(x) - \mathbb{E}_X\left[\tilde{\mu}(X)(x)\right]\right)\right]\right] =$$

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

$$+ \frac{1}{N^2}\mathbb{E}_{x,y}\left[\mathbb{E}_X\left[\sum_{n_1 \neq n_2}\left(\tilde{\mu}(X)(x) - \mathbb{E}_X\left[\tilde{\mu}(X)(x)\right]\right) \times \right.\right.$$

$$\left.\left. \times \left(\tilde{\mu}(X)(x) - \mathbb{E}_X\left[\tilde{\mu}(X)(x)\right]\right)\right]\right] =$$

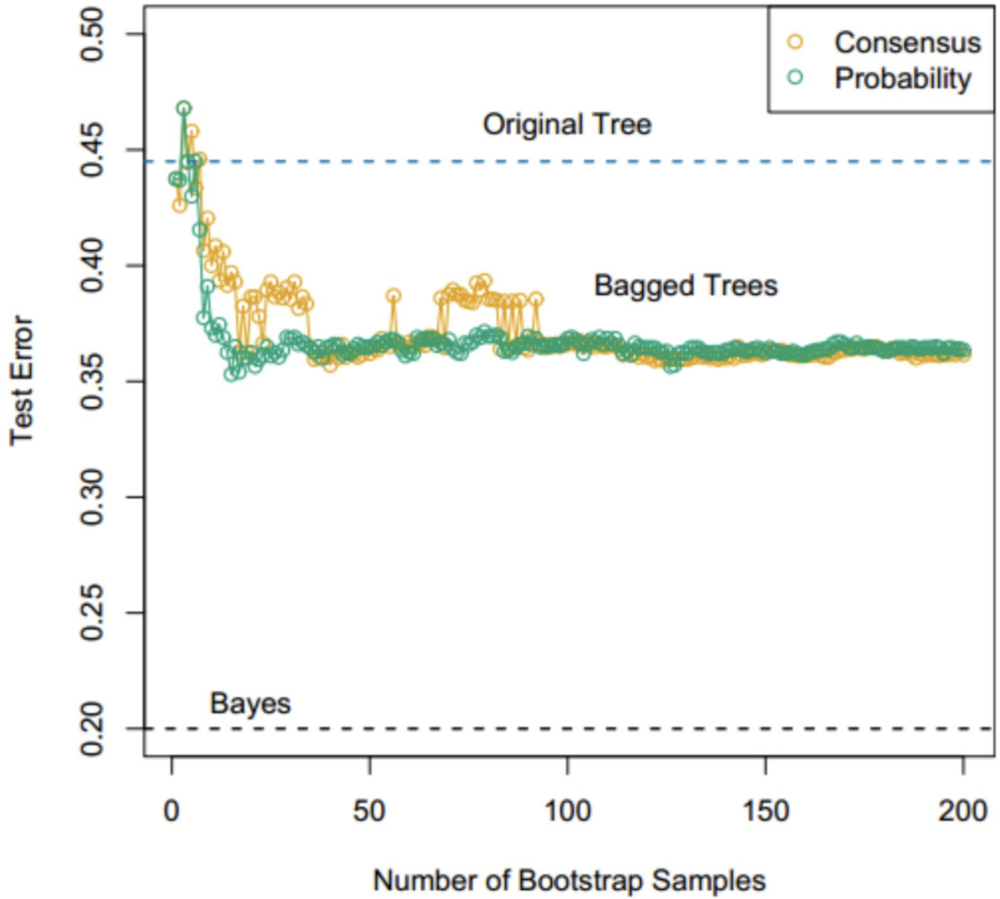
One algorithm variance * 1/N

$$= \boxed{\frac{1}{N}\mathbb{E}_{x,y}\left[\mathbb{E}_X\left[\left(\tilde{\mu}(X)(x) - \mathbb{E}_X\left[\tilde{\mu}(X)(x)\right]\right)^2\right]\right]} +$$

$$\boxed{+ \frac{N(N-1)}{N^2}\mathbb{E}_{x,y}\left[\mathbb{E}_X\left[\left(\tilde{\mu}(X)(x) - \mathbb{E}_X\left[\tilde{\mu}(X)(x)\right]\right) \times \left(\tilde{\mu}(X)(x) - \mathbb{E}_X\left[\tilde{\mu}(X)(x)\right]\right)\right]\right]}$$

Basic algorithms covariance

Bagging = Bootstrap aggregating



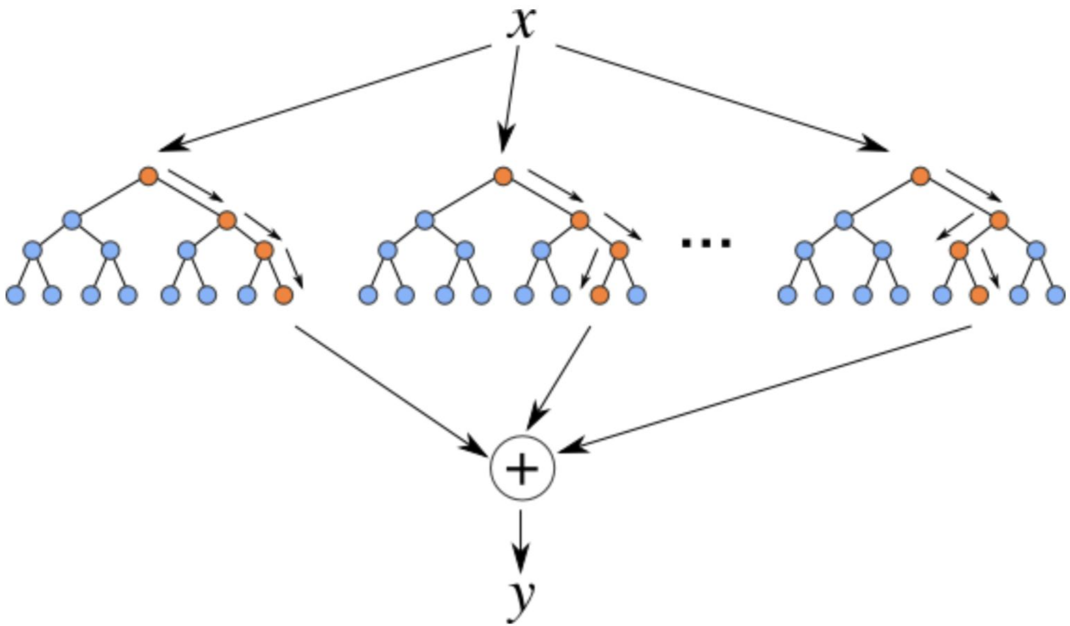
Bagging = Bootstrap aggregating

Decreases the variance if the basic algorithms are not correlated.

RSM - Random Subspace Method

Same approach, but with features.

Bagging + RSM = Random Forest



Random Forest

- One of the greatest “universal” models.

Random Forest

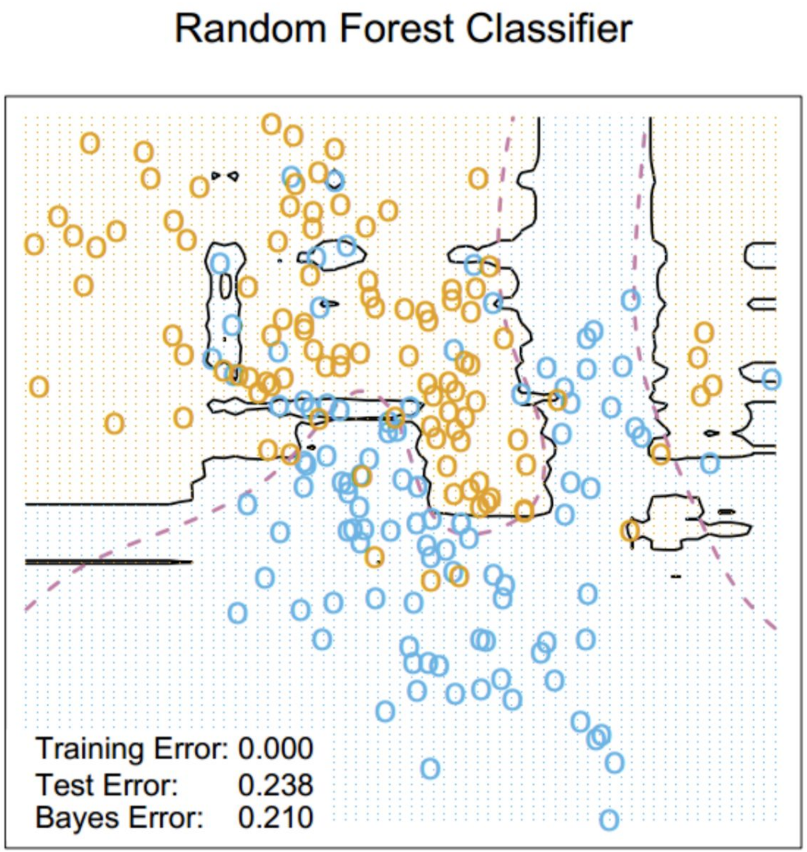
- One of the greatest “universal” models.
- There are some modifications: Extremely Randomized Trees, Isolation Forest, etc.
-

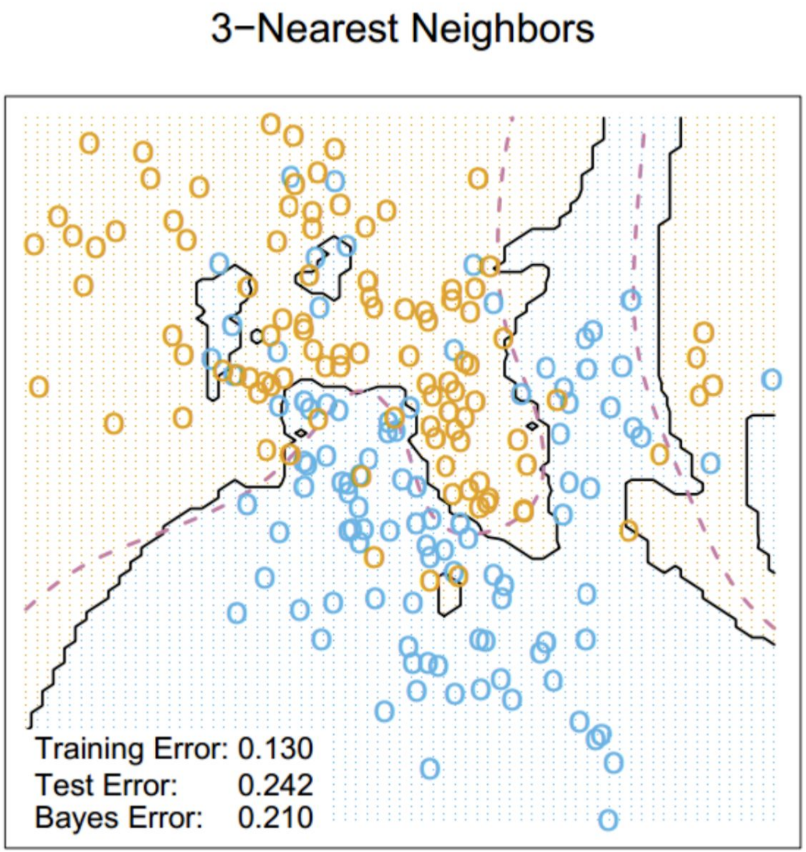
Random Forest

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

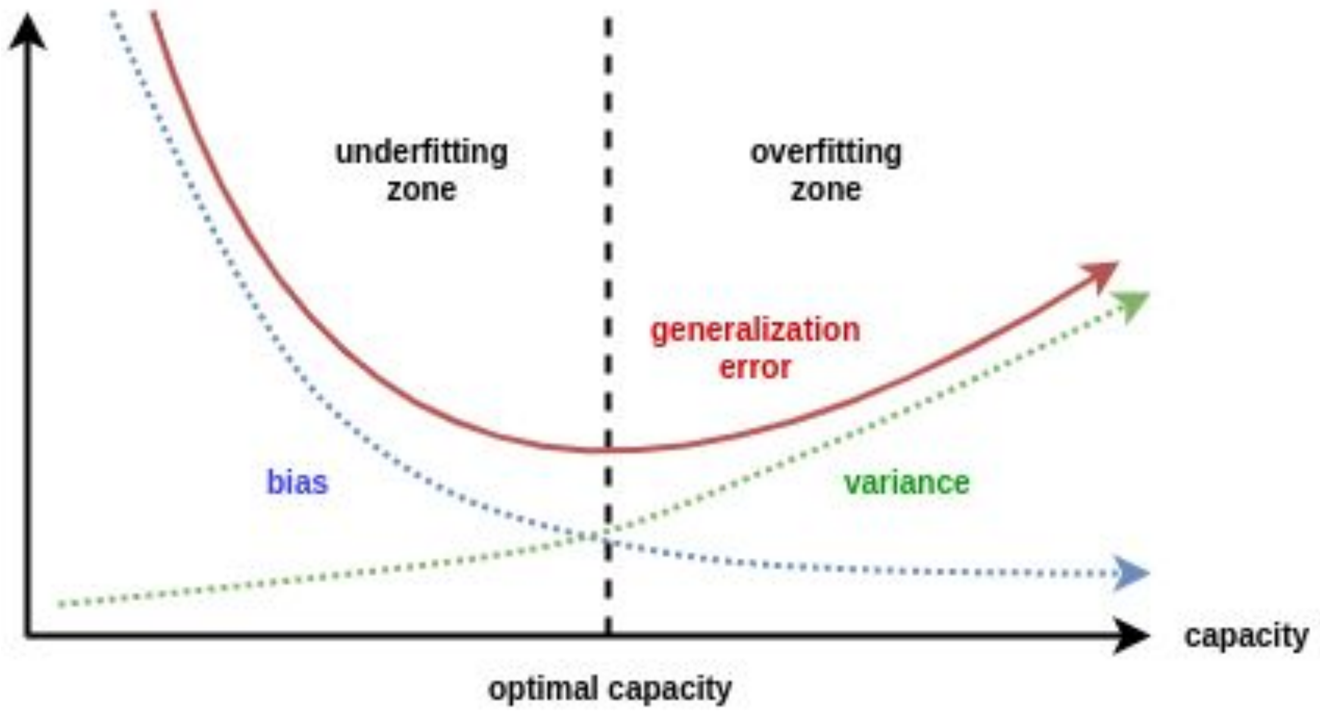
- One of the greatest “universal” models.
- There are some modifications: Extremely Randomized Trees, Isolation Forest, etc.
- Allows to use train data for validation: OOB

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


Random Forest Classifier
Training Error: 0.000
Test Error: 0.238
Bayes Error: 0.210

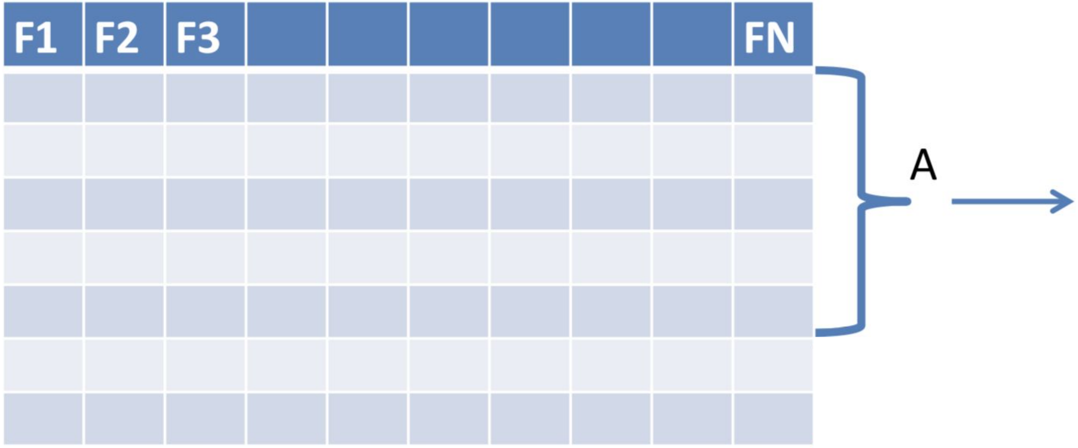

3−Nearest Neighbors
Training Error: 0.130
Test Error: 0.242
Bayes Error: 0.210

Bias-variance tradeoff



Stacking

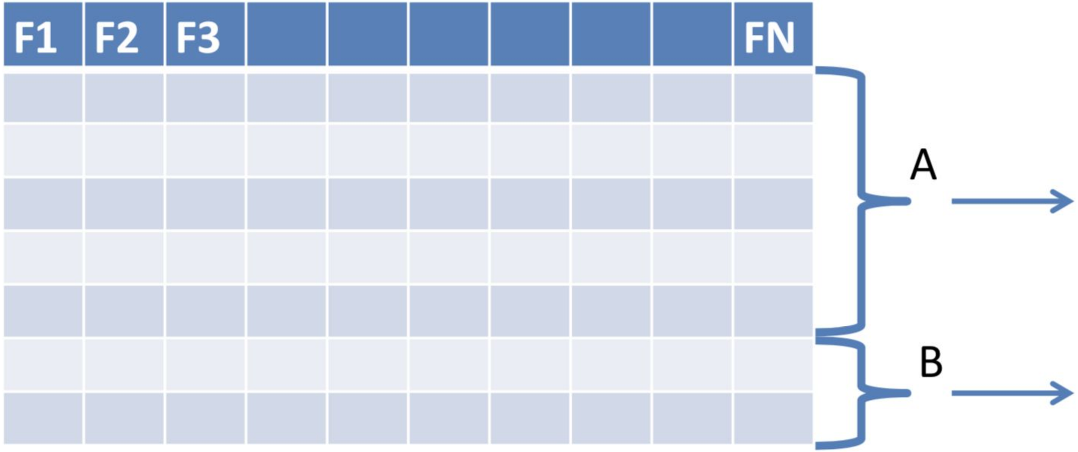
How to build an ensemble from *different* models?



Tune M basic algorithms

Stacking

How to build an ensemble from *different* models?

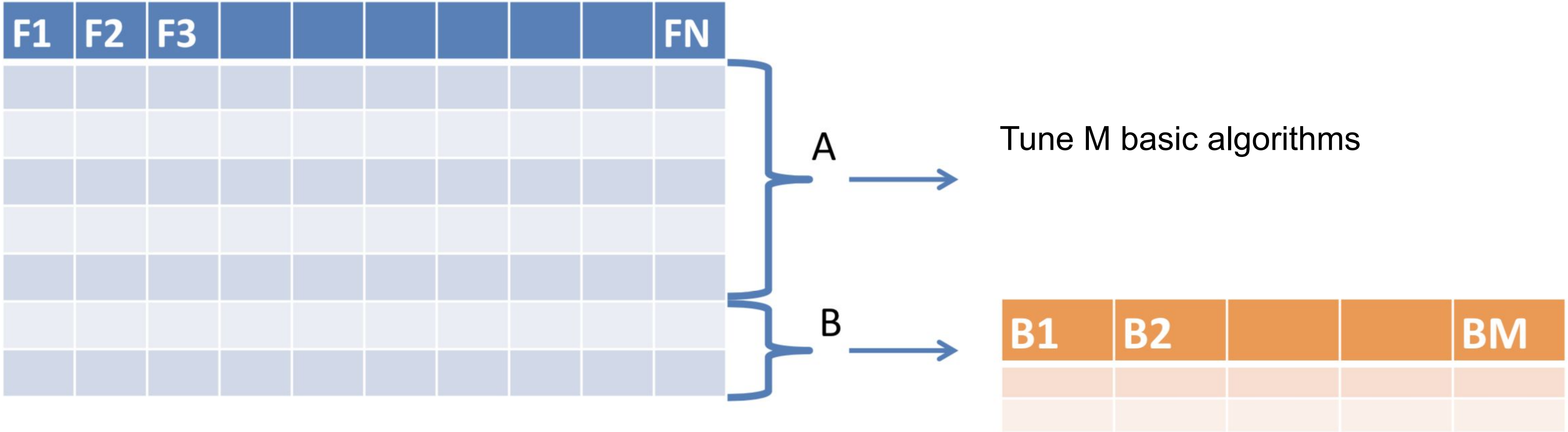


Tune M basic algorithms

Tune new model on the outputs of the basic algorithms

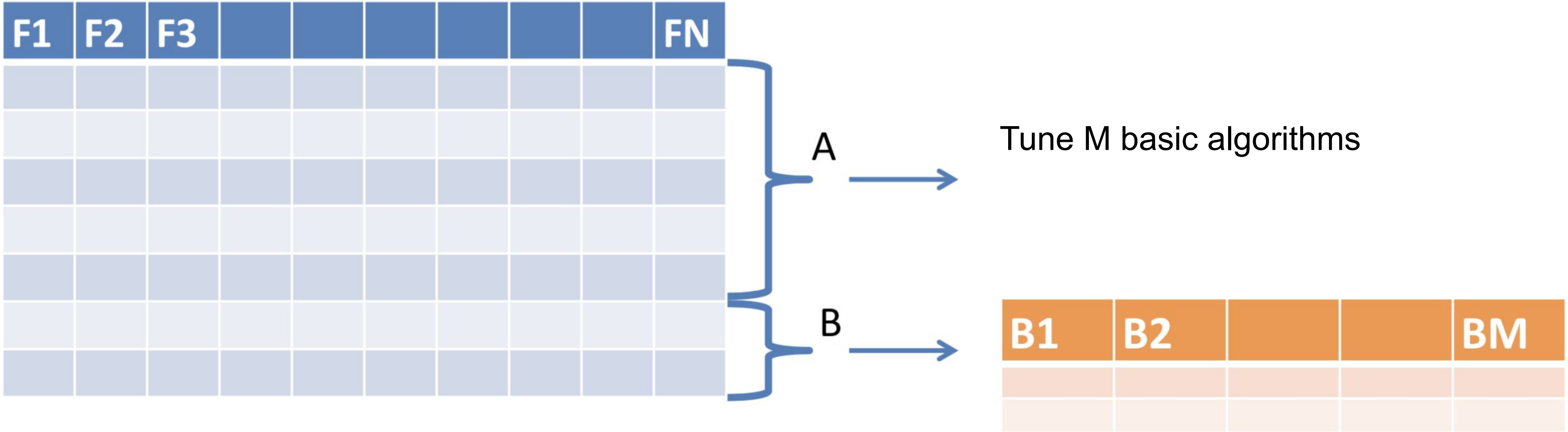
Stacking

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.

Stacking

How to build an ensemble from *different* models?

Stacking

How to build an ensemble from *different* models?

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

Stacking

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.)

Stacking

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 :)

Blending

Just combine several *strong/complex* models.

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

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

Blending

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

Blending

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.

Blending

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