MADMO
Introduction to …

Boostings and ensembles

Taras Khakhulin
Deep Learning Engineer Samsung AI Center
Skoltech & MIPT alumnus

t.khakhulin@gmail.com
https://github.com/khakhulin/
https://twitter.com/t_khakhulin
https://www.linkedin.com/in/taras-khakhulin/

Организационные моменты*

*чтобы не было потом каких-то недопониманий

Оценка за курс*

- 4 дз (50% оценки)
- итоговый проект (30% оценки) (ОЧЕНЬ ВАЖНО)
- Итоговый тест (20% оценки)
- Бонусы (?)

* предварительная версия

Итоговые проекты

1. “Рабочий” проект
2. Соревнование на kaggle
3. “Улучшенное” домашнее задание

1. “Рабочий проект”

1) Описание задачи: данные и их объём, метрики
2) Ограничения: по памяти, по времени, …
3) Проведённые эксперименты
4) Итоговый алгоритм
5) Идеи для улучшения

https://www.picbon.org/tag/ulkovarasto

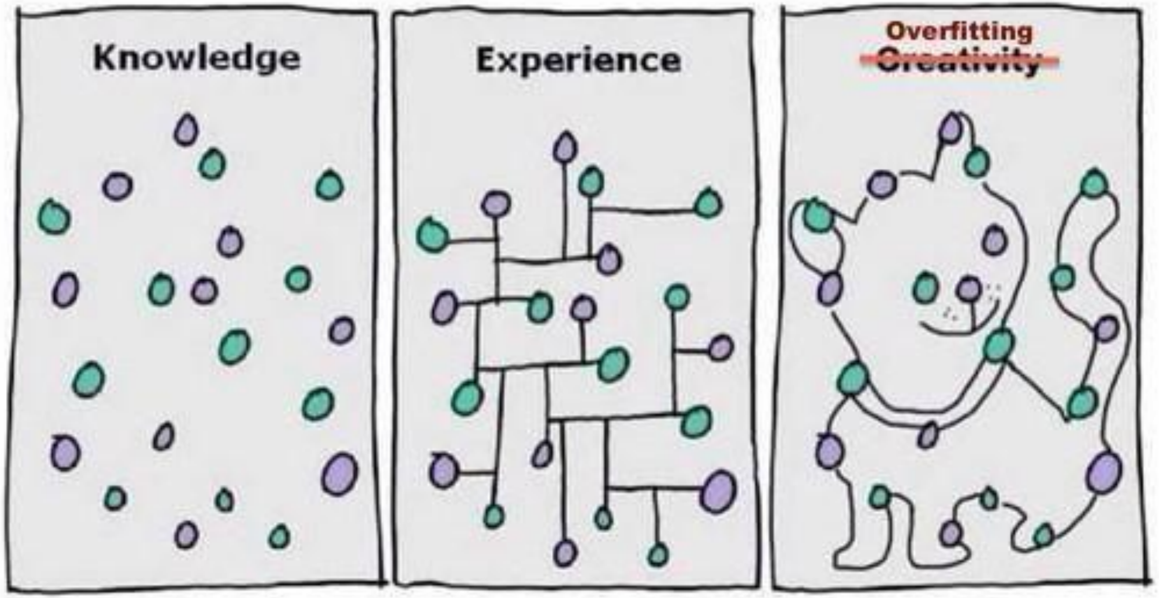
2. Соревнование на kaggle

1. Описание данных
2. .ipynb с экспериментами
   a. exploratory data analysis
   b. генерация признаков
   c. разбиение train-dev
   d. эксперименты с моделями
   e. финальный сабмит
   f. идеи для улучшения

http://www.shivambansal.com/blog/kaggle-bot/

3. “Улучшенное” (доделанное) домашнее задание

1. Идея улучшения
2. Эксперименты
3. Результаты



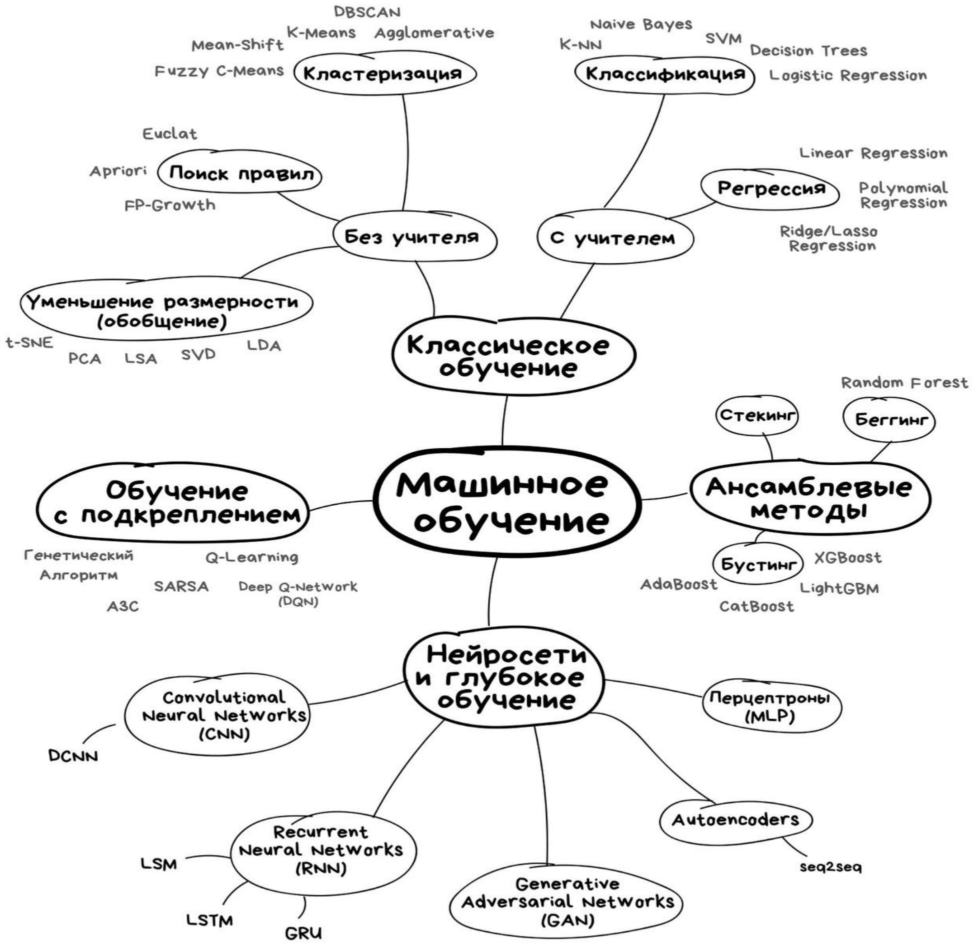
https://twitter.com/gagan_s

А что хотелось бы лично Вам?

https://forms.gle/tADLg7ZY8A44gcey5

Зоопарк моделей

https://vas3k.ru/blog/machine_learning/



https://vas3k.ru/blog/machine_learning/

https://vas3k.ru/blog/machine_learning/

Let’s start our journey

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

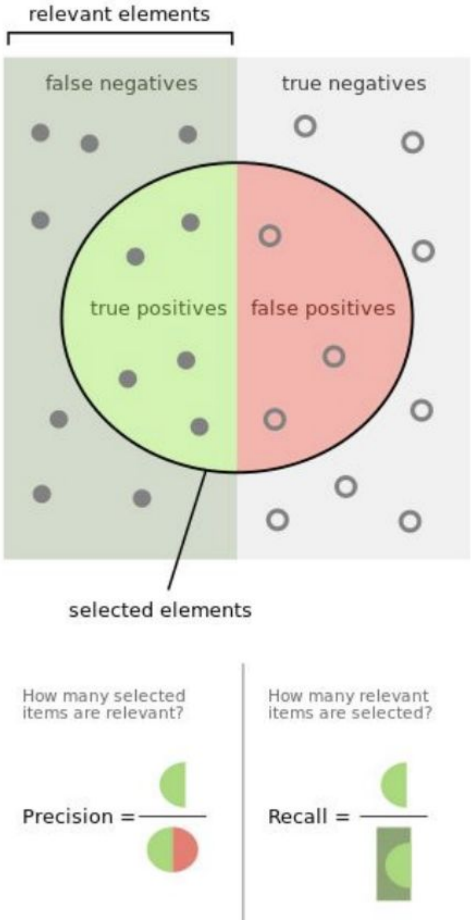
Precision and recall

		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.

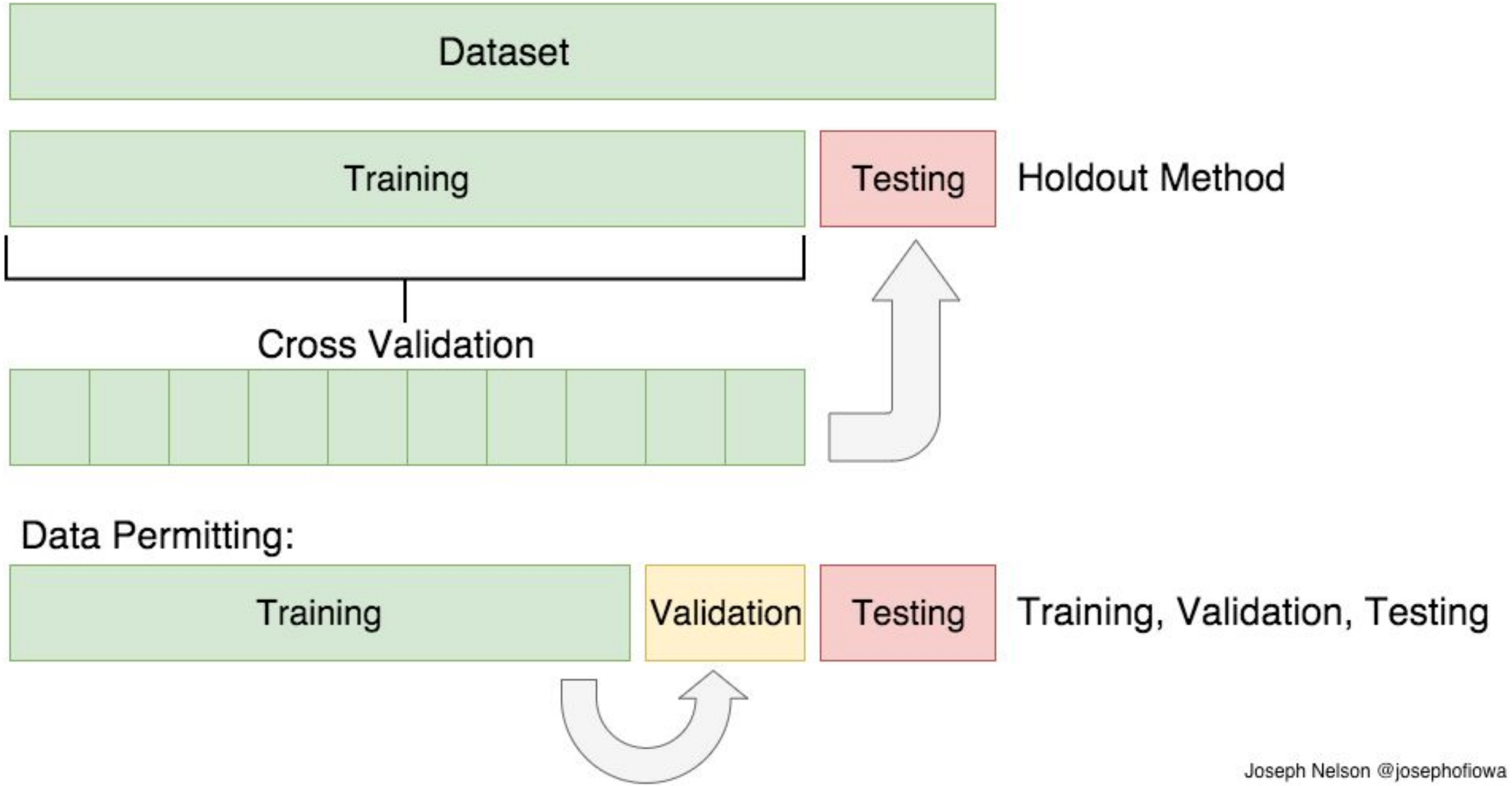
$$F_1 = \left(\frac{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

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

Warning about your training

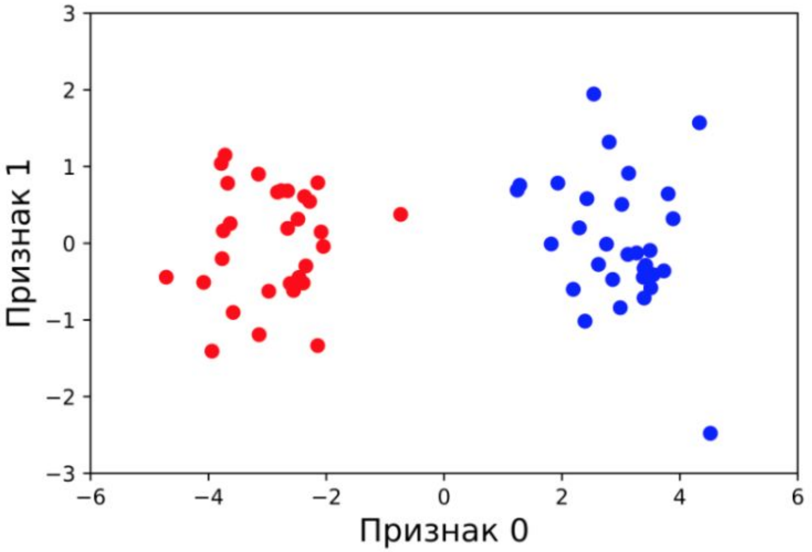
- Size of dev and test datasets
- The homogeneity of the train, dev, test
- The choice of algorithm
- Metrics
- Error analysis

Pipeline



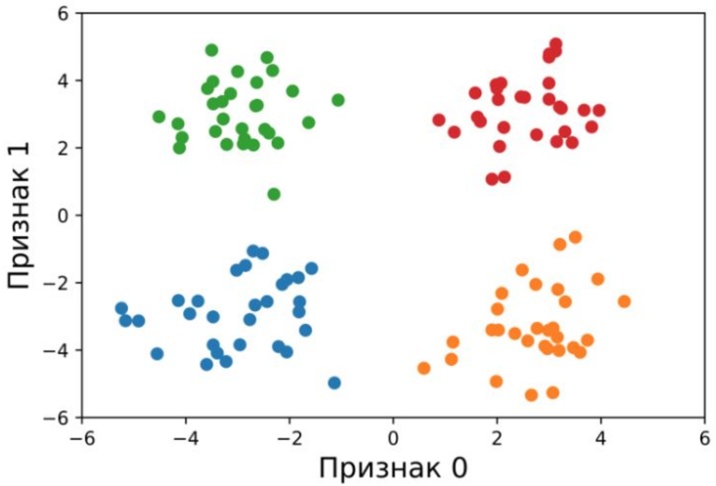
https://discuss.pytorch.org

Decision Trees



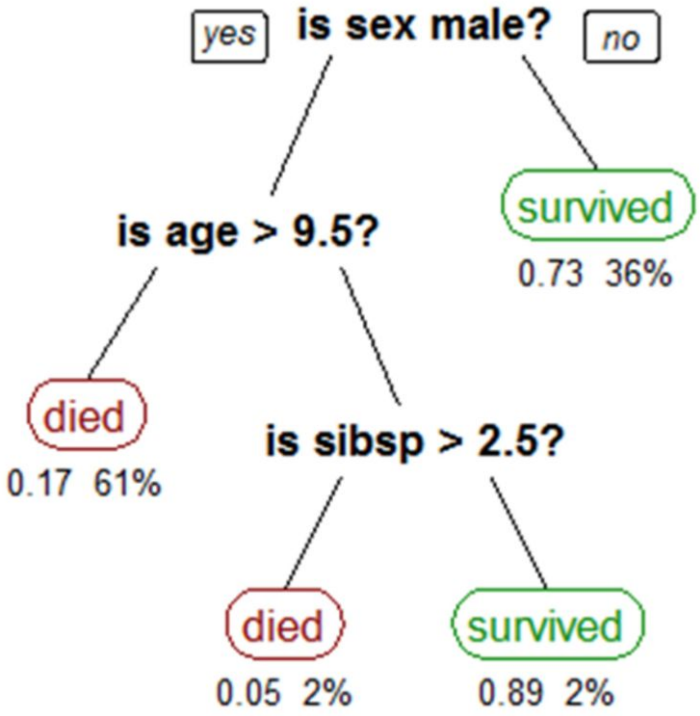

```python
def classify(X):
    if X[0] < 0:
        return "red"
    else:
        return "blue"
```

Decision Trees

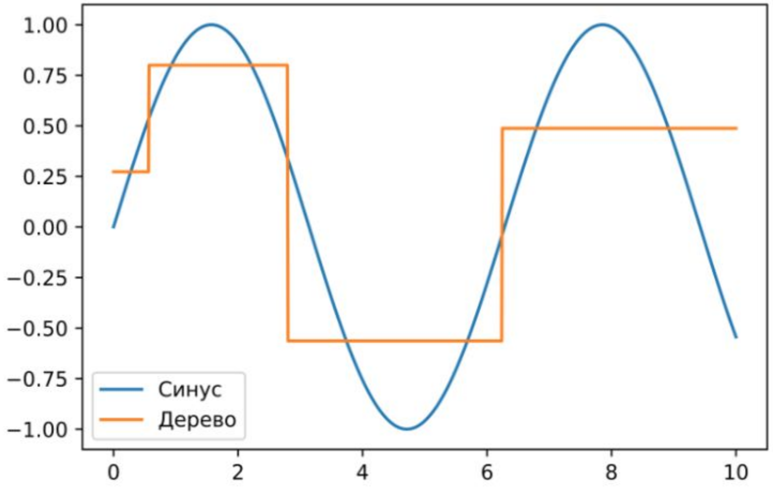


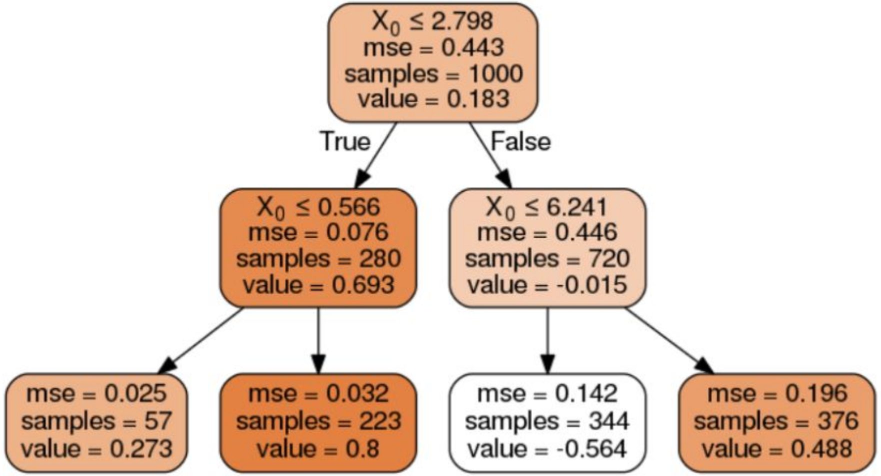

```python
def classify(X):
    if X[0] < 0:
        if X[1] < 0:
            return "blue"
        else:
            return "green"
    else:
        if X[1] > 0:
            return "red"
        else:
            return "orange"
```

Decision Trees

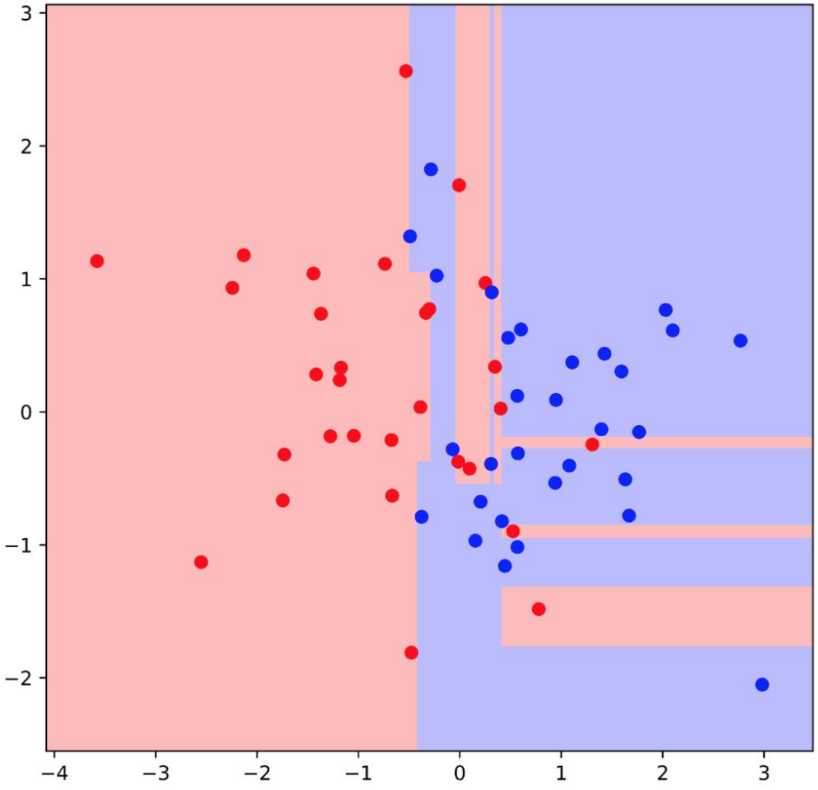


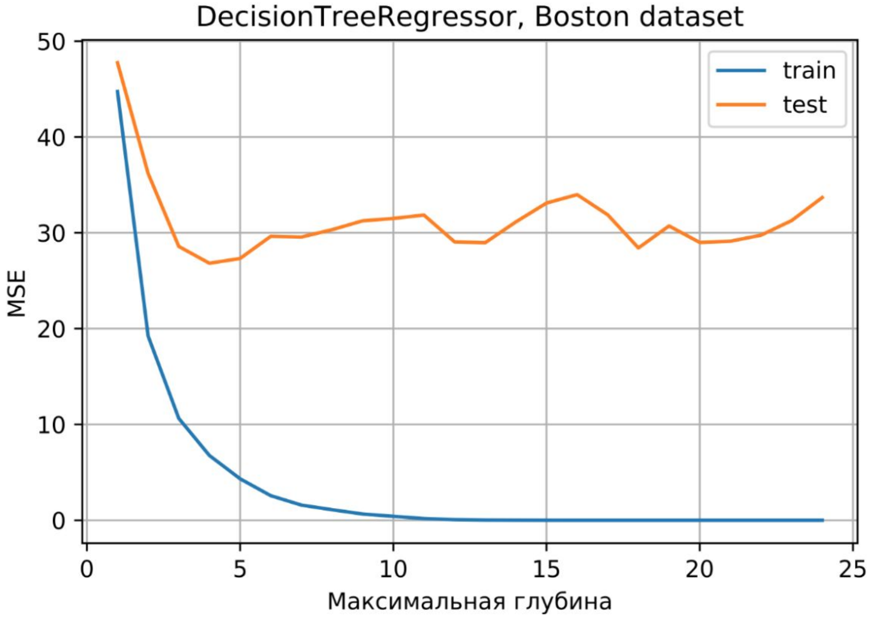
Decision Trees





Decision Trees





Information criteria

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

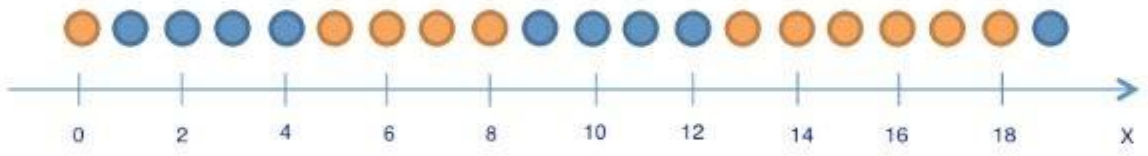
1. Misclassification criteria: $$H(R) = 1 - \max\{p_0, p_1\}$$

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

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

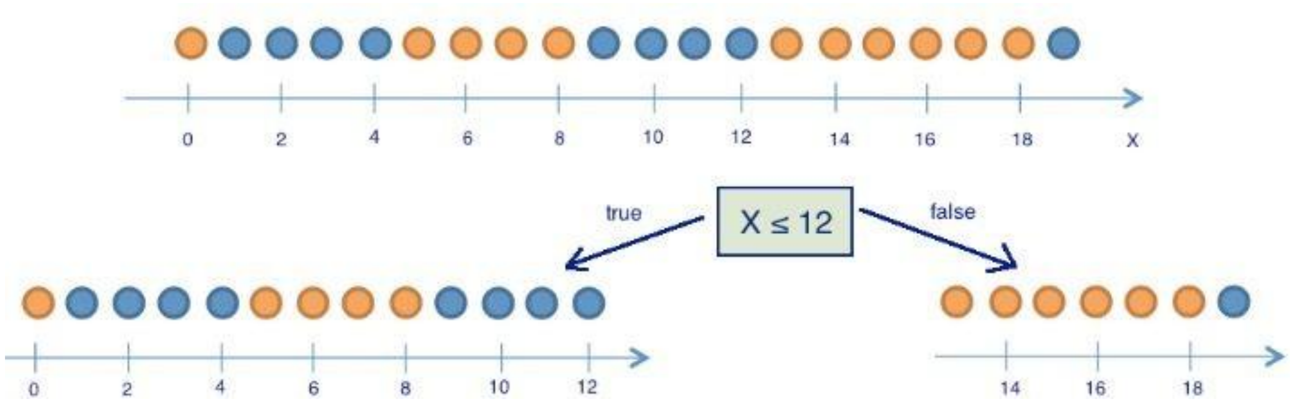
Information criteria

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

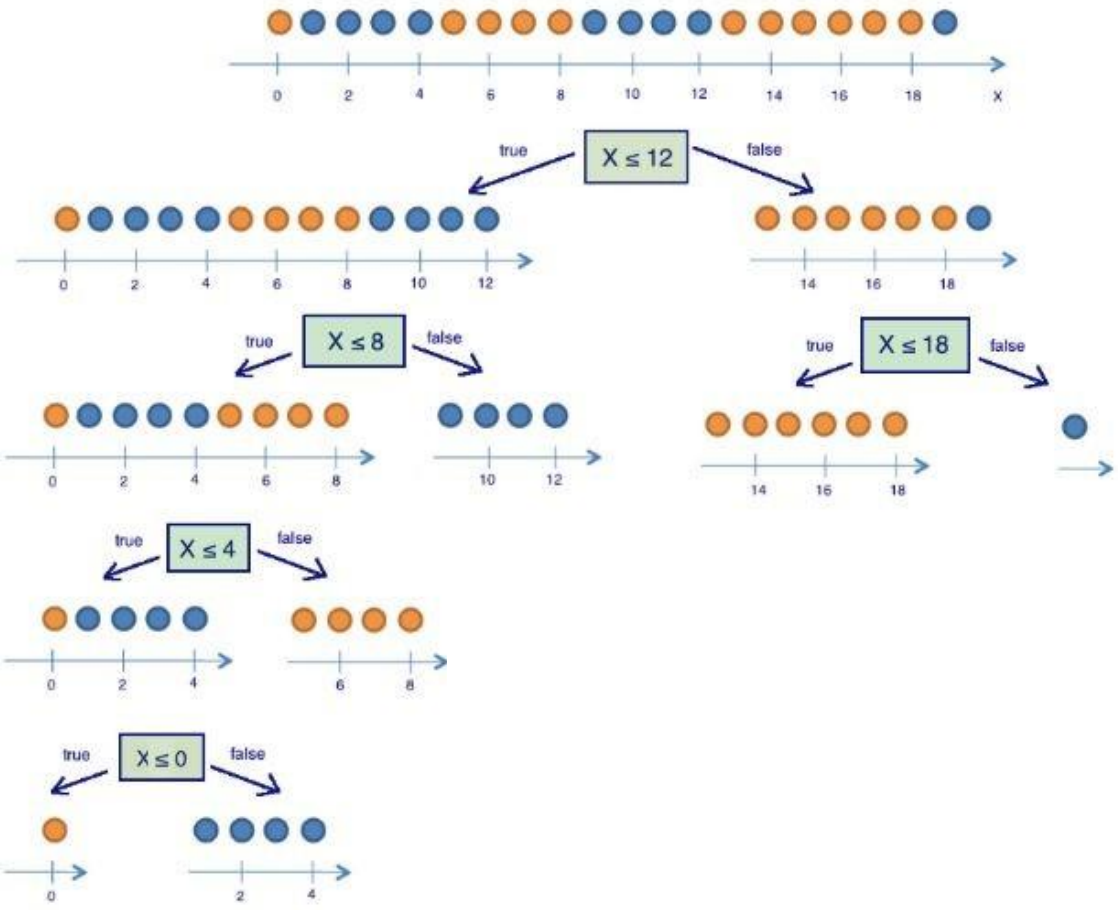


Information criteria

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

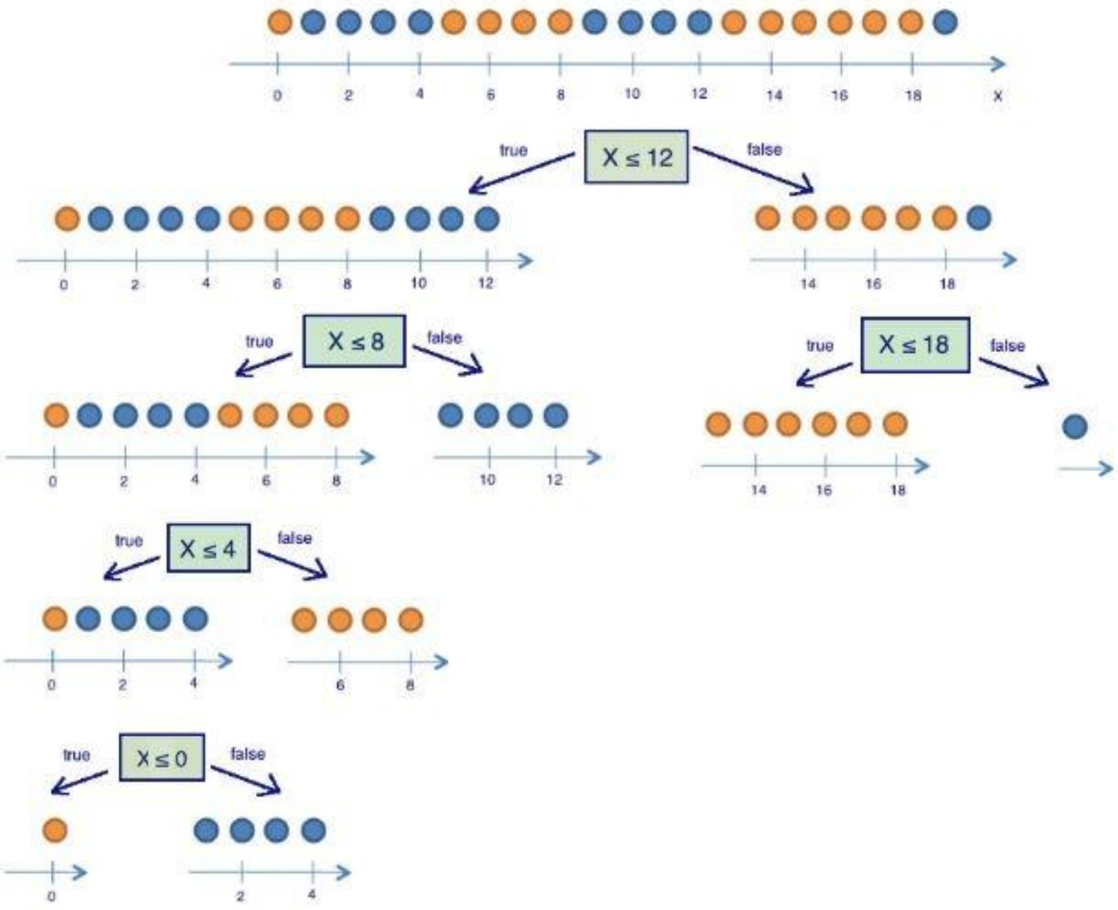


Entropy



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

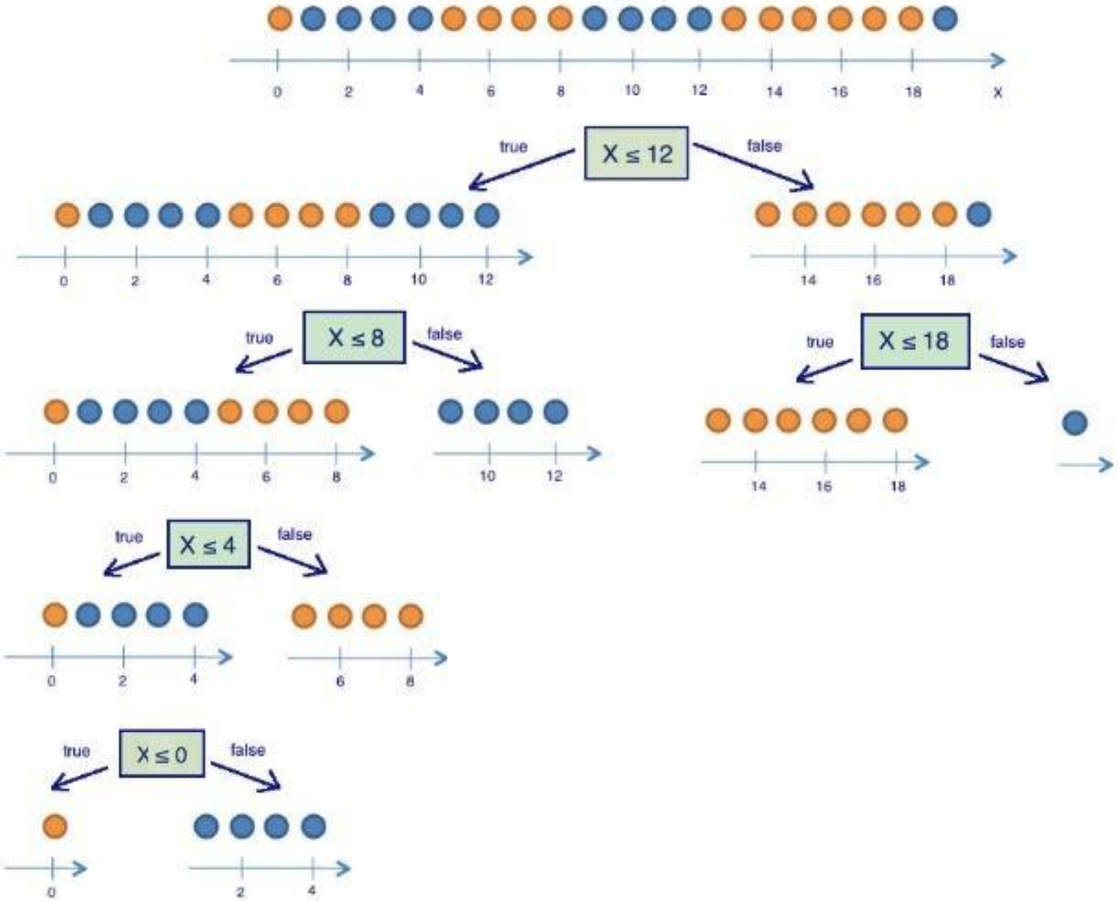
Entropy



$$S = -\sum_{k} p_k \log_2 p_k$$

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

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: Gini impurity

$$G = 1 - \sum_k (p_k)^2$$

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

Gini impurity

$$G = 1 - \sum_k (p_k)^2$$

In binary case N = 2

$$G = 1 - p_+^2 - p_-^2 = 1 - p_+^2 - (1 - p_+)^2 = 2p_+(1 - p_+)$$

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

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

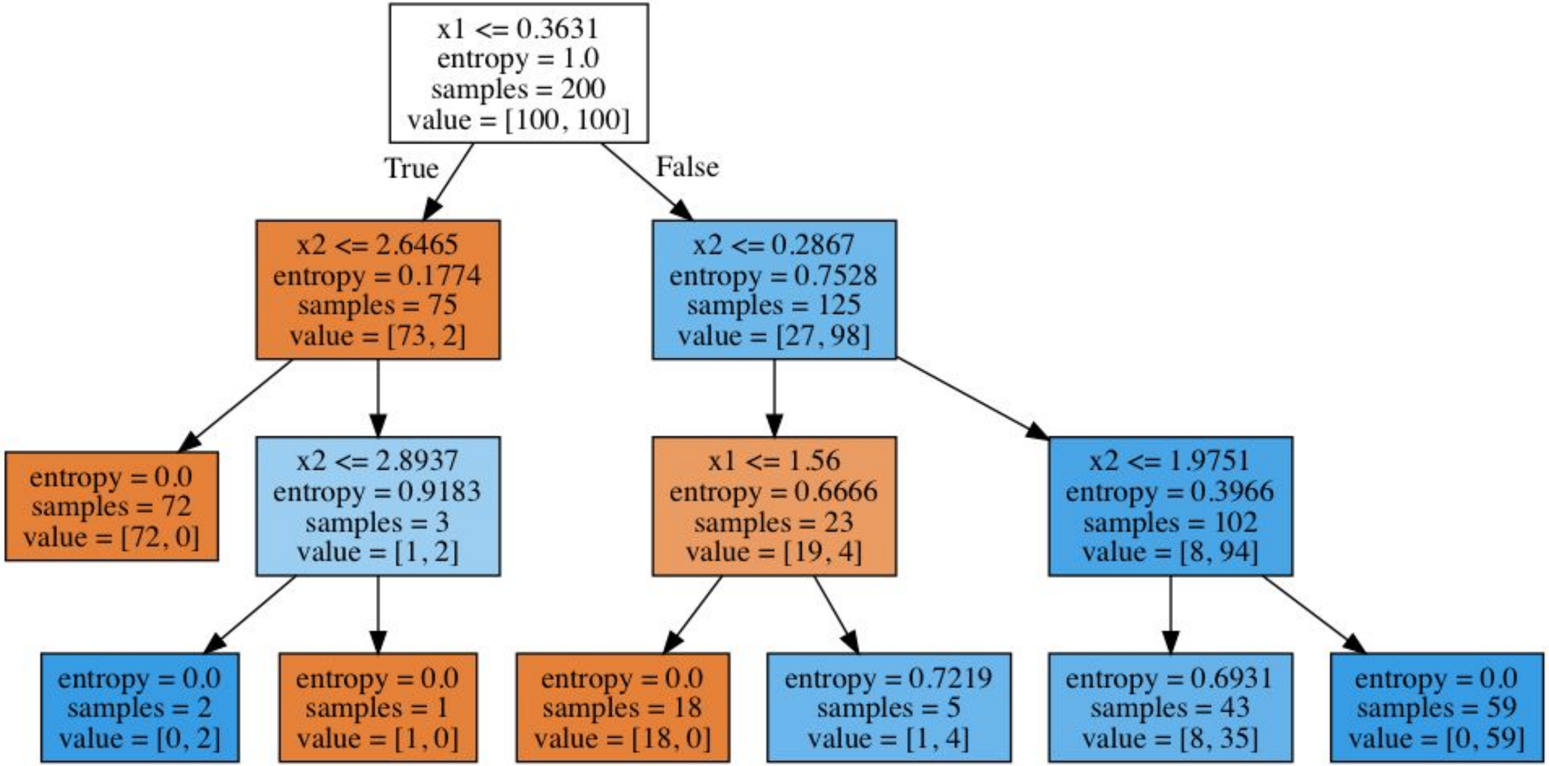
$$c^* = \frac{1}{|R|} \sum_{y_i \in R} y_i$$

How the trees are actually constructed

- ID-3
- C4.5
- C5.0
- CART
- etc.

How the trees are actually constructed

- ID-3
- C4.5
- C5.0
- CART
- etc.



Ensembles

Bootstrap
aggregating

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:

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

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

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

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

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!

Great News!

Regression Ensemble

Example:

Regression Ensemble

Example: $$a(x) = \frac{1}{n}\left(b_1(x) + \ldots + b_n(x)\right)$$

Classification Ensemble

Example:

Classification Ensemble

Example: $a(x) = \text{mode}\left(b_1(x), \ldots, b_n(x)\right)$

Real-world Ensemble

b - meta-algorithm

$$a(x) = b\big(b_1(x), \ldots, b_n(x)\big)$$

(every b_i is a weak learner)

Why do we use ensembles in classification?

0 - correct
1 - incorrect

b1 = b2 = b3, the probability of the error p

(0, 0, 0) (1-p)(1-p)(1-p)
(1, 0, 0) p(1-p)(1-p)
(0, 1, 0) p(1-p)(1-p)
(0, 0, 1) p(1-p)(1-p)

The error of all algos: p^3

Why do we use ensembles in classification?

0 - correct
1 - incorrect

b1 = b2 = b3, the probability of the error p

(0, 0, 0)   (1-p)(1-p)(1-p)
(1, 0, 0)   p(1-p)(1-p)
(0, 1, 0)   p(1-p)(1-p)
(0, 0, 1)   p(1-p)(1-p)

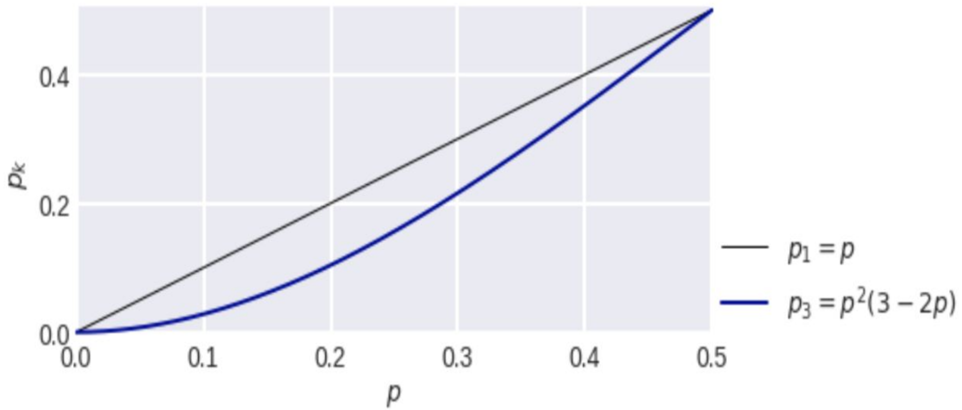
The error of all algos: p^3

The error of all ensembles:
p^3 + 3(1-p)p^2

Why do we use ensembles in classification?

0 - correct
1 - incorrect

b1 = b2 = b3, the probability of the error p



The error of all algos: p^3
But we see on single

The error of all ensembles:
p^3 + 3(1-p)p^2

Where is the problem of all methods?

Fixed target function

Similar dataset

Solve one task

Ensembles

Voting

Stacking

Boosting

Output Coding

Bagging

Heuristics

Ensembles

Voting
- averaging

Stacking

Boosting

Output Coding
- code target (squared)

Bagging

Heuristics
- Hand-crafted methods

Voting

$$a(x) = \mathrm{mode}\big(b_1(x), \ldots, b_n(x)\big)$$

$$a(x) = \frac{1}{n}\big(\mathrm{rank}(b_1(x)) + \ldots + \mathrm{rank}(b_n(x))\big)$$

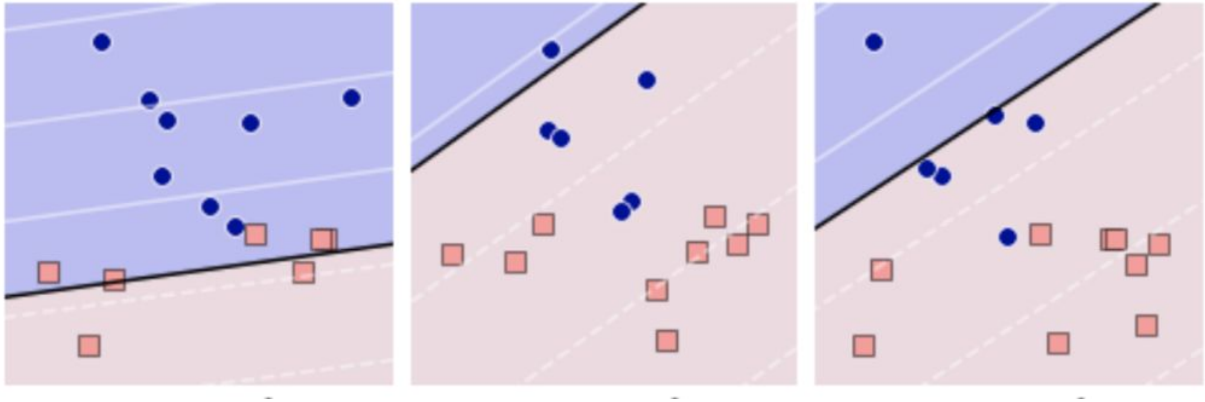
Ranking for roc-auc

$$a(x) = \frac{1}{w_1 + \ldots + w_n}\big(w_1 \cdot b_1(x) + \ldots + w_n \cdot b_n(x)\big)$$

Weighted averaged

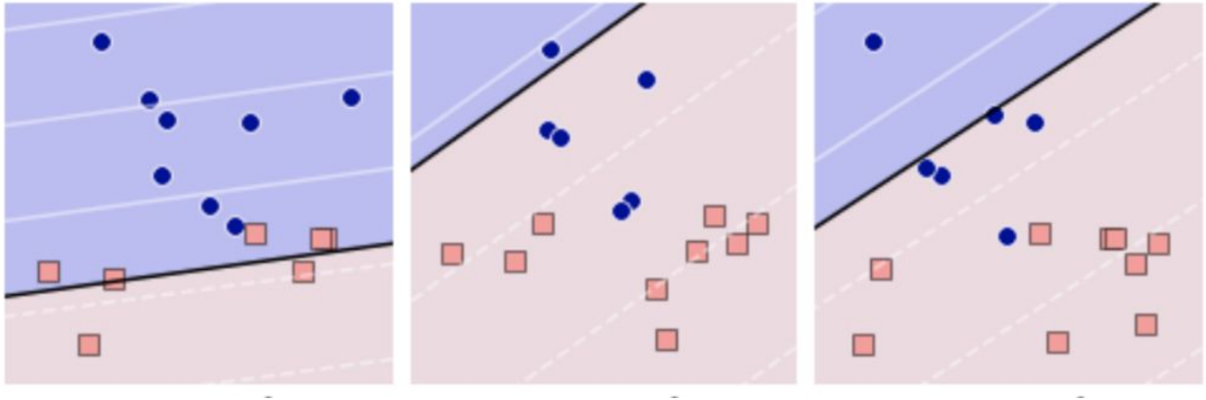
$$a(x) = w_1(x) \cdot b_1(x) + \ldots + w_n(x) \cdot b_n(x)$$

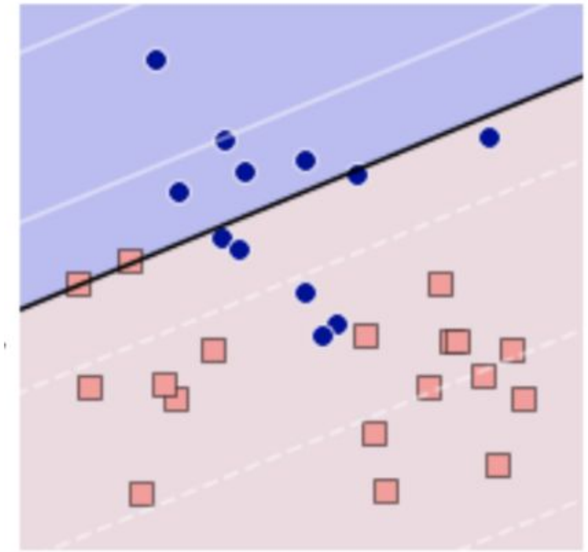
Featured-weighted ensembles



```
model =
BaggingClassifier(base_estimator=LogisticRegression(),
                                 n_estimators=100,
                                 max_samples=1.0,
                                 max_features=1.0,
                                 bootstrap=True,
                                 bootstrap_features=False,
                                 oob_score=False,
                                 warm_start=False,
                                 n_jobs=None,
                                 random_state=None,
                                 verbose=0)

model.fit(X, y)
```





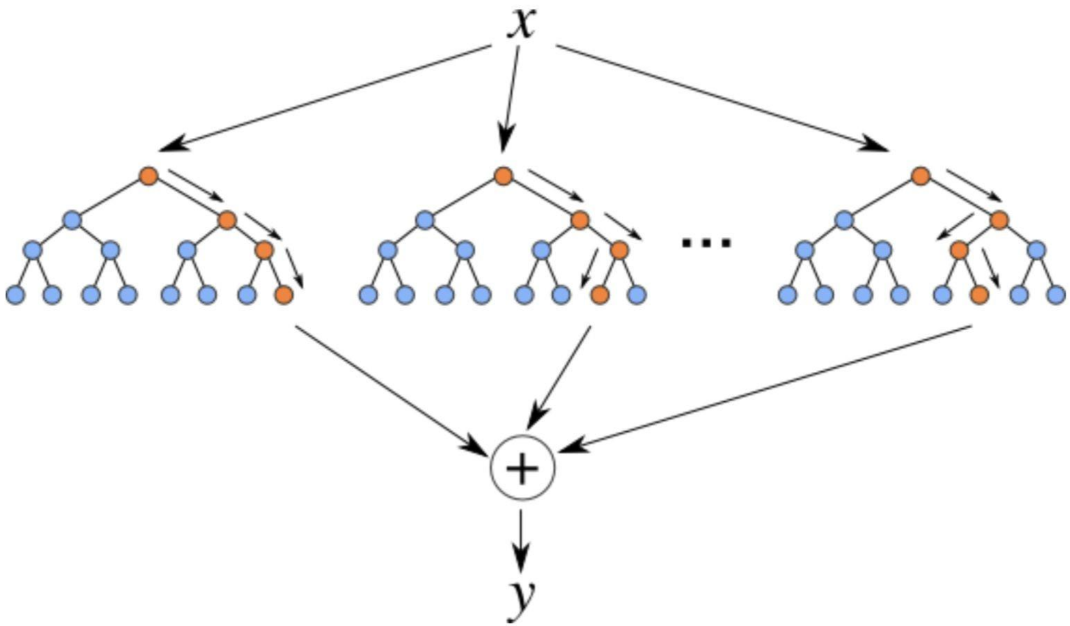
```
model =
BaggingClassifier(base_estimator=LogisticRegression(),
                  n_estimators=100,
                  max_samples=1.0,
                  max_features=1.0,
                  bootstrap=True,
                  bootstrap_features=False,
                  oob_score=False,
                  warm_start=False,
                  n_jobs=None,
                  random_state=None,
                  verbose=0)

model.fit(X, y)
```

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

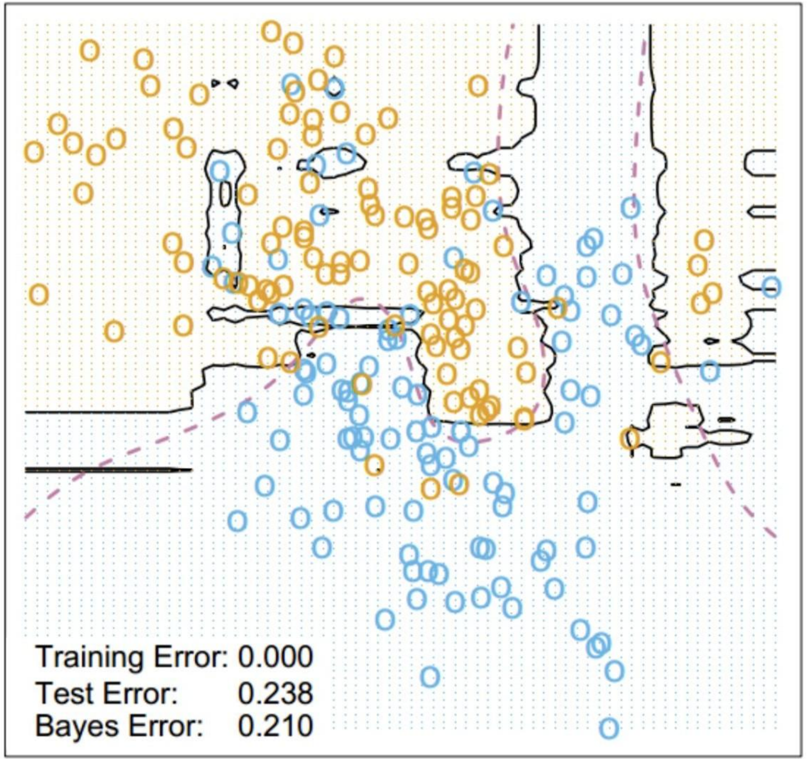
- One of the greatest “universal” models.
- There are some modifications: Extremely Randomized Trees, Isolation Forest, etc.
- Allows to use train data for validation: OOB

Random Forest

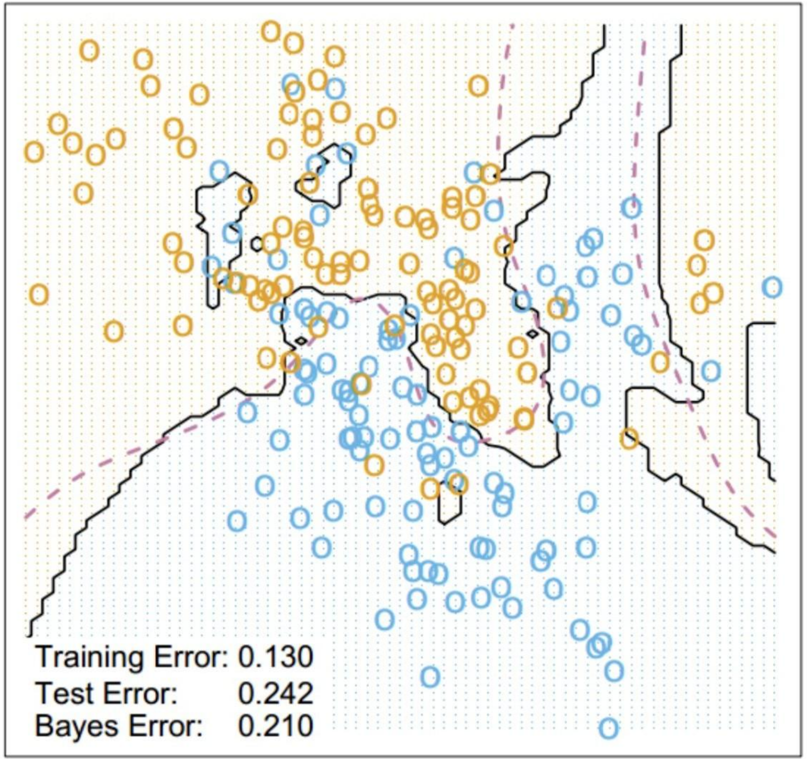
- One of the greatest “universal” models.
- There are some modifications: Extremely Randomized Trees, Isolation Forest, etc.
- Allows to use train data for validation: OOB

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

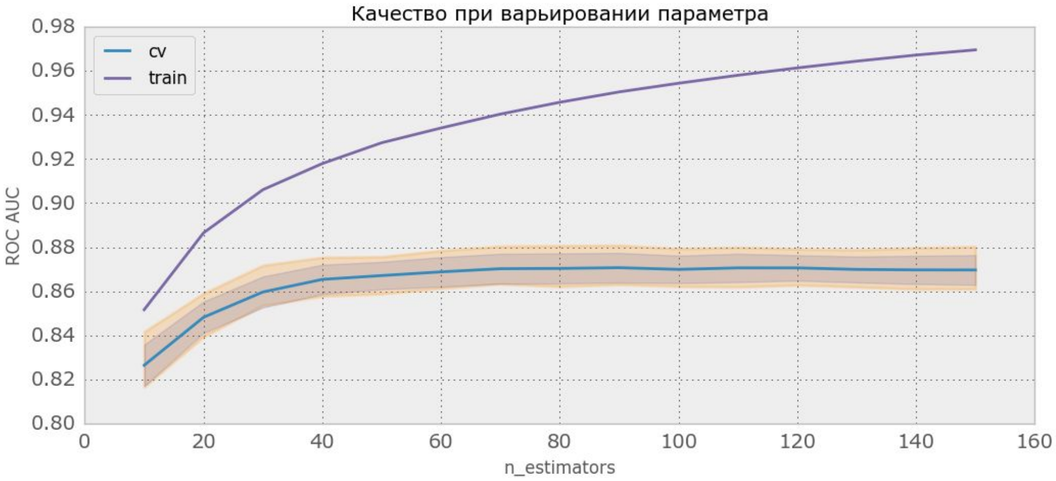
Random Forest Classifier

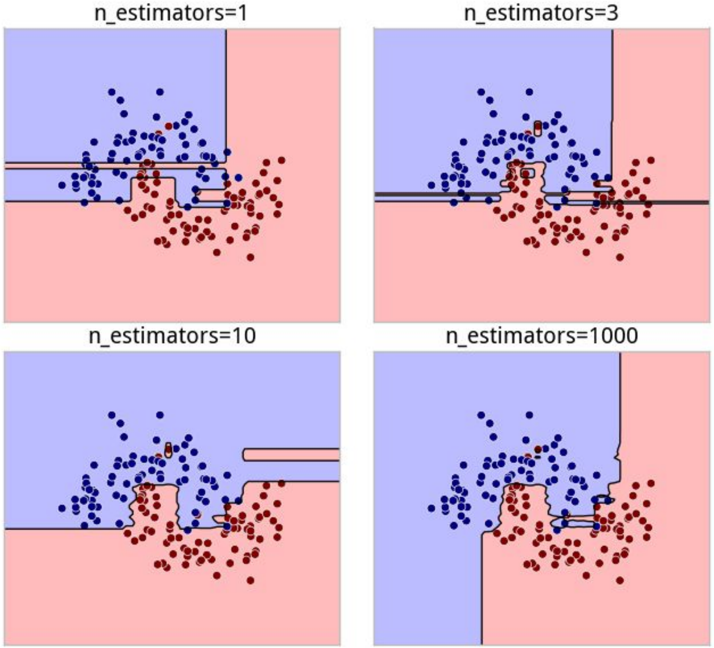


3−Nearest Neighbors



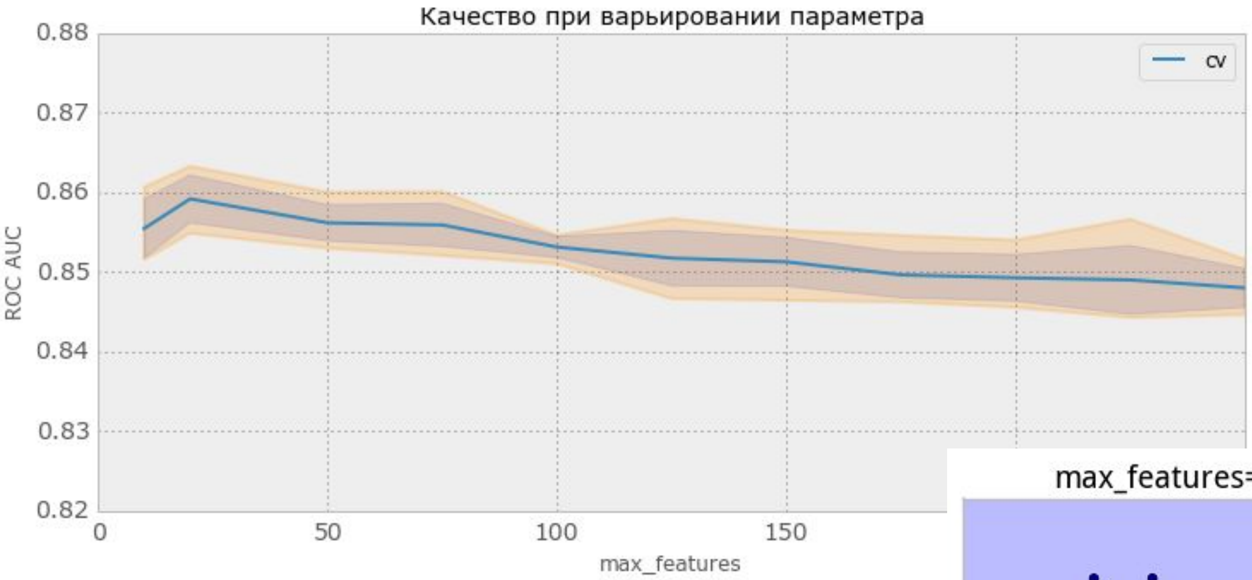
n_estimators



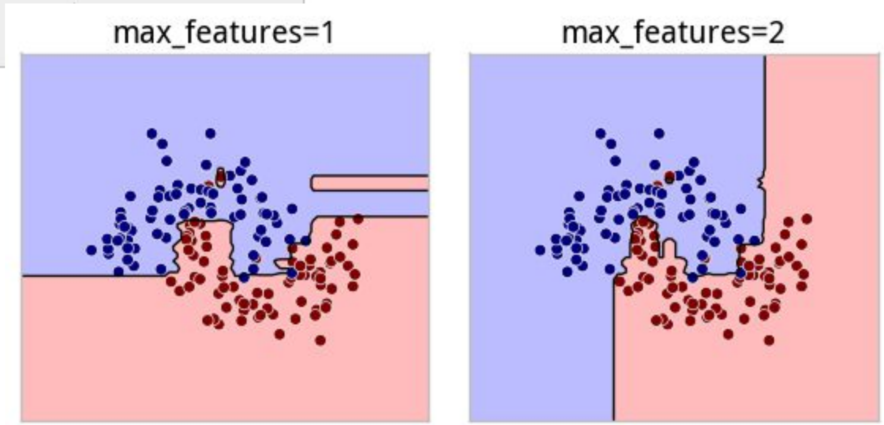


https://dyakonov.org

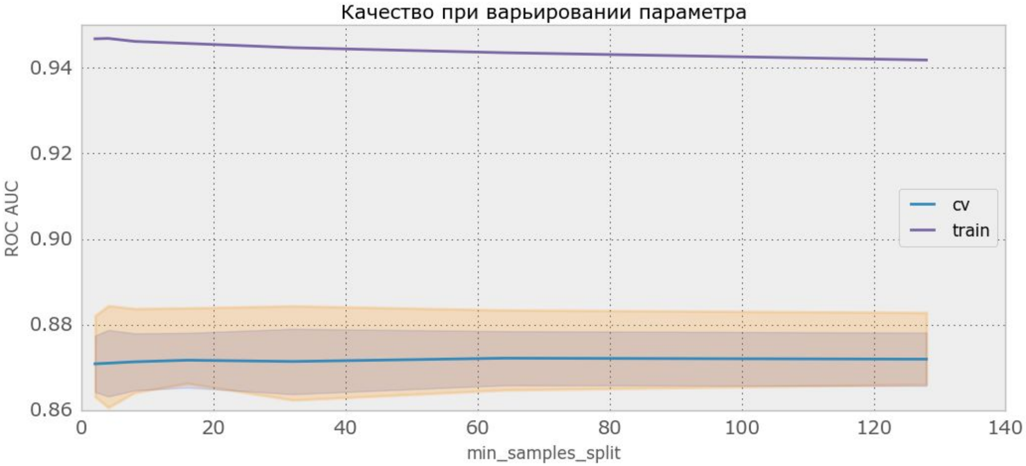
max_features

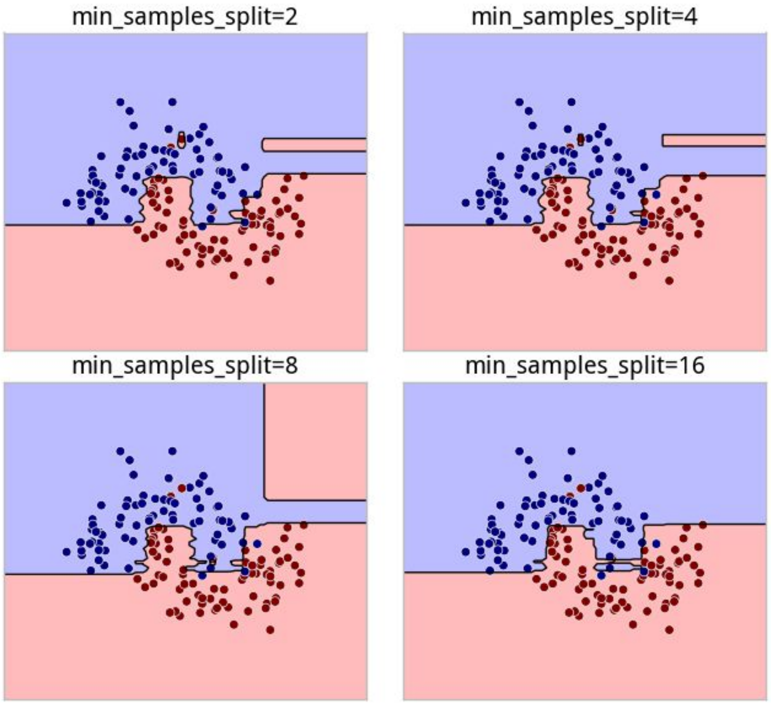


https://dyakonov.org



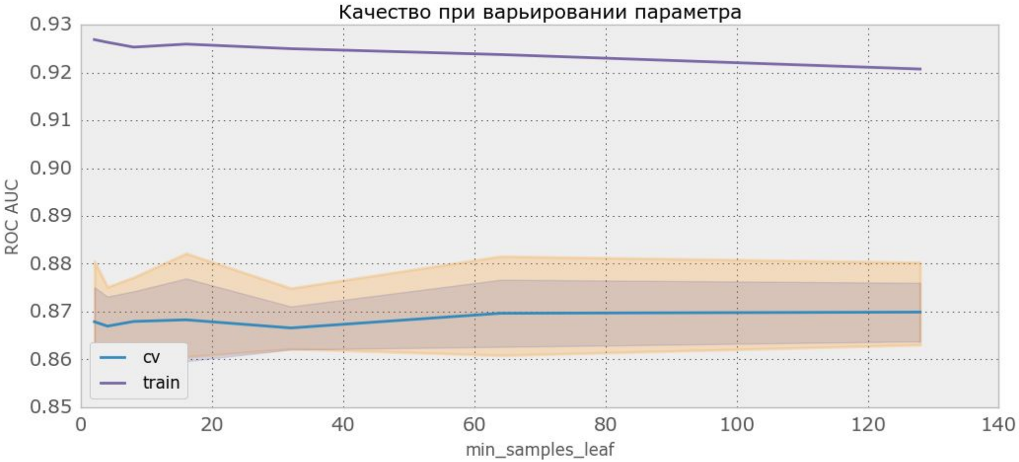
min_samples_split

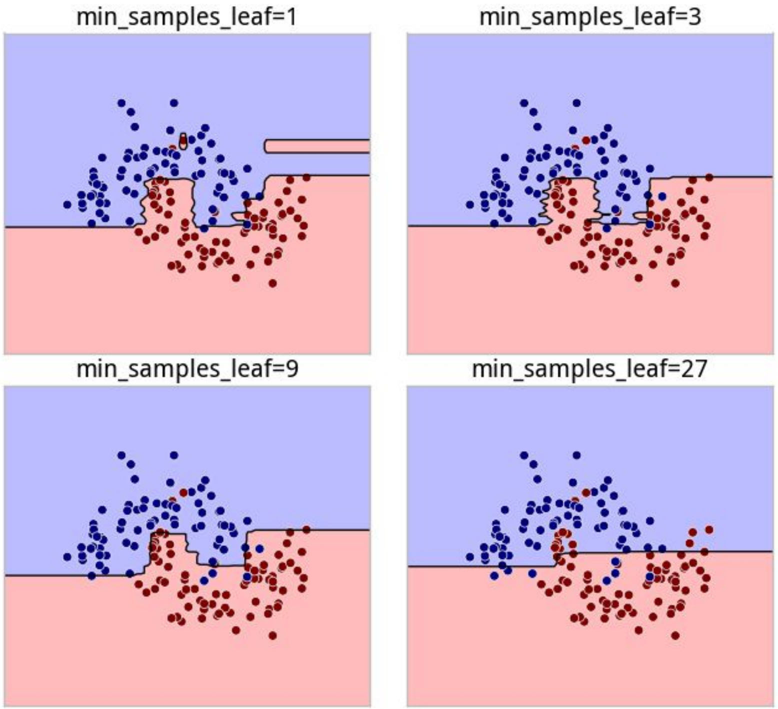




https://dyakonov.org

min_samples_leaf





https://dyakonov.org

Bias-variance decomposition

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