

A hand wearing a pink glove is using metal tweezers to place a small green seedling into a clear plastic pot filled with dark soil. In the background, a microscope is visible on a laboratory bench, and several other similar pots with seedlings are lined up on a white surface. The scene is brightly lit, suggesting a laboratory or greenhouse environment.

курс «Машинное обучение»

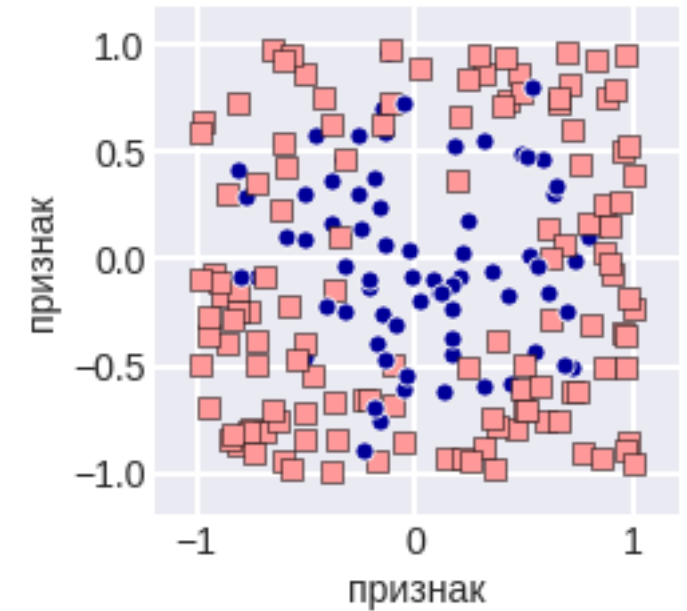
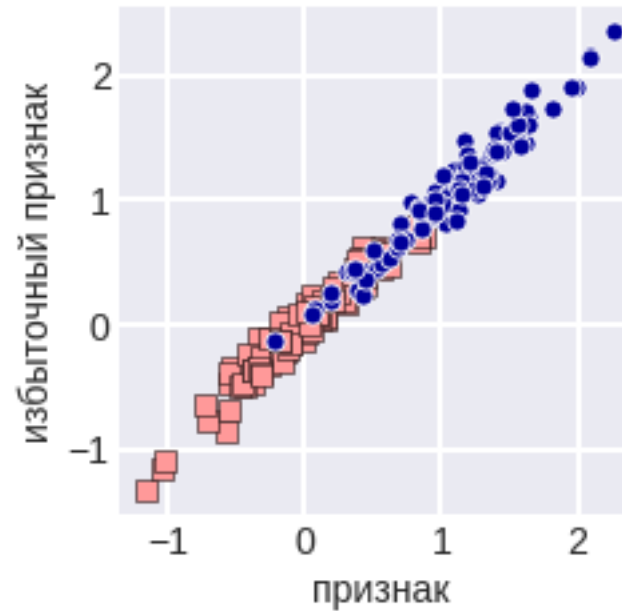
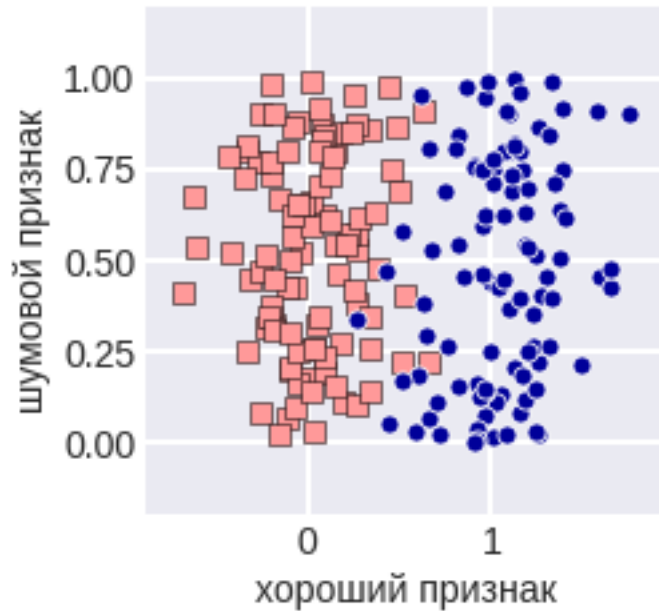
Отбор (селекция) признаков Feature Selection

Александр Дьяконов

07 декабря 2021 года

Отбор признаков (Feature Selection)

- **нахождение оптимального подмножества признаков, в соответствии с некоторым критерием**
- процесс удаления избыточных и нерелевантных признаков



Отбор признаков (Feature Selection)

Причины

- **интерпретация** (от чего и как зависит ответ)
- **скорость работы алгоритмов** (часто $O(n)$)
- **борьба с переобучением** (л/з для линейных методов)
- **повышения качества** (если много шума)
- **удешевление решения** (если добыча признаков что-то стоит)
- **моделирование** (хотим, чтобы решение зависело от определённых признаков)

Не путать с ...

- **извлечение признаков (feature extraction)**

создание новых признаков

- **сокращения размерности (dimensionality reduction)**

уменьшение n (вообще говоря, переход к новому признаковому пространству)

Отбор признаков (Feature Selection)

типичные ошибки в селекции

- **надеяться на селекцию**
идеальной процедуры отбора не существует!
- **попытки автоматизировать селекцию признаков**
может зависеть от модели / признаков / предобработки данных и т.п.
- **не обращать внимание на особенности данных**
надо совмещать и с генерацией
- **выбрасывание информации**

часть раньше было на http://frnsys.com/ai_notes/

Классификация методов

Фильтры (filter methods)

**по отдельному признаку и, возможно, целевому признаку
получают оценку его качества**
не ориентированы на конкретные модели алгоритмов ML
обычно считают статистики признаков

Обёртки (wrapper methods)

**получают оценки качества признаков с помощью анализа работы алгоритмов машинного
обучения на подмножествах признакового пространства**
ориентированы на конкретные модели алгоритмов ML

Встроенные (embedded / intrinsic methods)

являются частью методов ML
качество признаков получается автоматически
параллельно настройке модели

Фильтры: дискретные признаки
(принимают конечное число значений)

Энтропия:

$$H(X) = - \sum_{x_i \in X} p(x_i) \log p(x_i)$$

Условная энтропия (conditional entropy)

Энтропии, вычисленные
для фиксированных значений признаков:

$$\begin{aligned} H(Y | X) &= \sum_{x_i \in X} p(x_i) H(Y | X = \{x_i\}) = \\ &= \sum_{x_i \in X} p(x_i) \sum_{y_j \in Y} p(y_j | x_i) \log(p(y_j | x_i)) \end{aligned}$$

Свойства

$$H(Y) \geq H(Y | X) \geq 0$$

если X и Y независимы

$$H(Y | X) = H(Y)$$

$$H(X | X) = 0$$

$$H(X, Y) = H(Y | X) + H(X) = H(X | Y) + H(Y)$$

Фильтры: дискретные признаки

1) Взаимная информация (Information Gain, Mutual Information)

Насколько более чётко определена Y , если знаем X

$$MI(Y, X) = H(Y) - H(Y | X) = \sum_{y_j \in Y} \sum_{x_i \in X} p(x_i, y_j) \log \frac{p(x_i, y_j)}{p(x_i) p(y_j)}$$

Свойства:

$$MI(Y, X) = MI(X, Y)$$

$$0 \leq MI(Y, X) \leq \min[H(X), H(Y)]$$

левая граница – когда Y, X независимы,

правая – когда одна с.в. детерминированная функцией другой

Для независимых признаков = 0

Предпочитает выбирать признаки с большим числом значений

Фильтры: дискретные признаки

2) хи-квадрат

	Y=0	Y=1	
X=0	6	4	$\Sigma=10$
X=1	14	16	$\Sigma=30$
	$\Sigma=20$	$\Sigma=20$	$\Sigma=40$

**Ожидаемая вероятность в предположении
независимости:**

$$P(A \cap B) = P(A) \cdot (B)$$

$$\text{expected} = \frac{10}{40} \cdot \frac{20}{40} \cdot 40 = 5$$

$$\chi^2 = \sum \frac{(\text{observed} - \text{expected})^2}{\text{expected}}$$

в общем случае

$$\chi^2 = \sum_i \sum_j \frac{(m_{ij} - \mu_{ij})^2}{\mu_{ij}}, \quad \mu_{ij} = \frac{\sum_t m_{it} \sum_t m_{tj}}{m}$$

Фильтры: статистики признаков

3) низкая оценка дисперсии – почти константный признак

```
sel = VarianceThreshold(threshold=(.8 * (1 - .8)))  
sel.fit_transform(X)
```

4) t-оценка (для задачи с 2 классами) / t-критерий Стьюдента в предположениях о нормальности распределений

$$\frac{|\mu_1 - \mu_2|}{\sqrt{\frac{\sigma_1^2}{m_1} + \frac{\sigma_2^2}{m_2}}}$$

в общем случае – ANOVA (ANalysis Of VAriance)

Фильтры: статистики признаков**ANOVA F-test**

$$\frac{m_1 \sum_{i \in K_1} (\mu - \mu_1)^2 + m_2 \sum_{i \in K_2} (\mu - \mu_2)^2}{\frac{1}{m-2} \left(\sum_{i \in K_1} (x_i - \mu_1)^2 + \sum_{i \in K_2} (x_i - \mu_2)^2 \right)}$$

«explained variance» / «between-group variability»

Фильтры: статистики признаков

5) Корреляция между признаками

корреляция с целевым \Rightarrow хороший

корреляция с другим \Rightarrow один можно удалить

Оценка линейной зависимости – корреляционный коэффициент Пирсона (Pearson)

Оценка монотонной зависимости – корреляционный коэффициент Спирмена (Spearman)

коэффициент Кендалла (Kendall rank correlation coefficient)

для выборки с двумя признаками $\{(x_i, y_i)\}_{i=1}^m$

$$\frac{|\{(i, j) \mid \text{sgn}(x_i - x_j) = \text{sgn}(y_i - y_j)\}| - |\{(i, j) \mid \text{sgn}(x_i - x_j) \neq \text{sgn}(y_i - y_j)\}|}{C_m^2}$$

6) Использование других мер качества

ex: AUC-ROC-признака

или строим простой алгоритм на этом признаке
(но это уже похоже на обёртку)

Минутка кода: фильтры

```
from sklearn.datasets import load_breast_cancer
from sklearn.feature_selection import GenericUnivariateSelect, chi2
from sklearn.feature_selection import SelectKBest

X, y = load_breast_cancer(return_X_y=True)
X.shape
(569, 30)

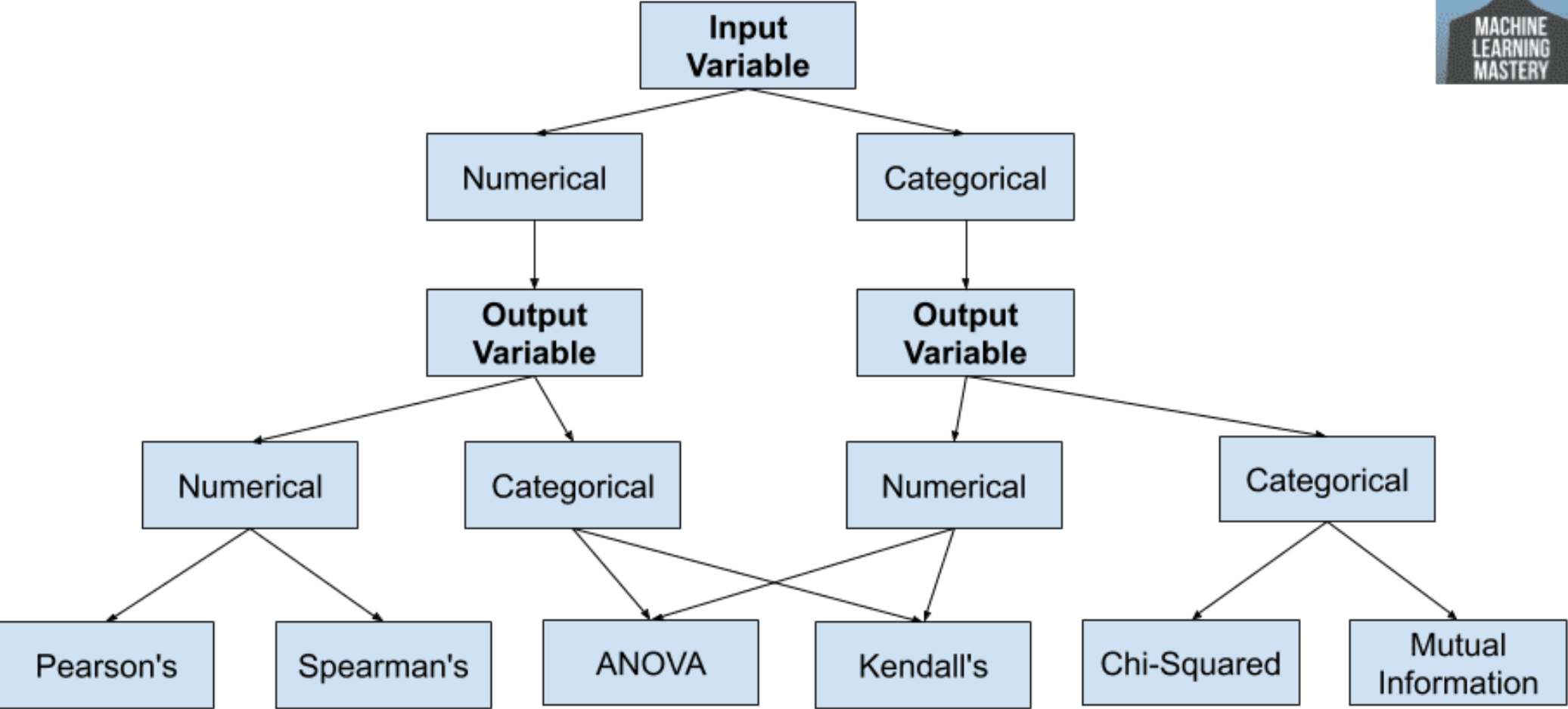
transformer = GenericUnivariateSelect(chi2, # как оценивать качество f(x[:,j], y)
                                     mode='k_best', # стратегия отбора
                                     param=20) # здесь - сколько признаков

X_new = transformer.fit_transform(X, y)
X_new.shape
(569, 20)

X_new = SelectKBest(chi2, k=20).fit_transform(X, y) # или можно было так...
```

Фильтры

How to Choose a Feature Selection Method



Copyright © MachineLearningMastery.com

<https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/>

Минутка кода: фильтры

sklearn.feature_selection: Feature Selection

The `sklearn.feature_selection` module implements feature selection algorithms. It currently includes univariate filter selection methods and the recursive feature elimination algorithm.

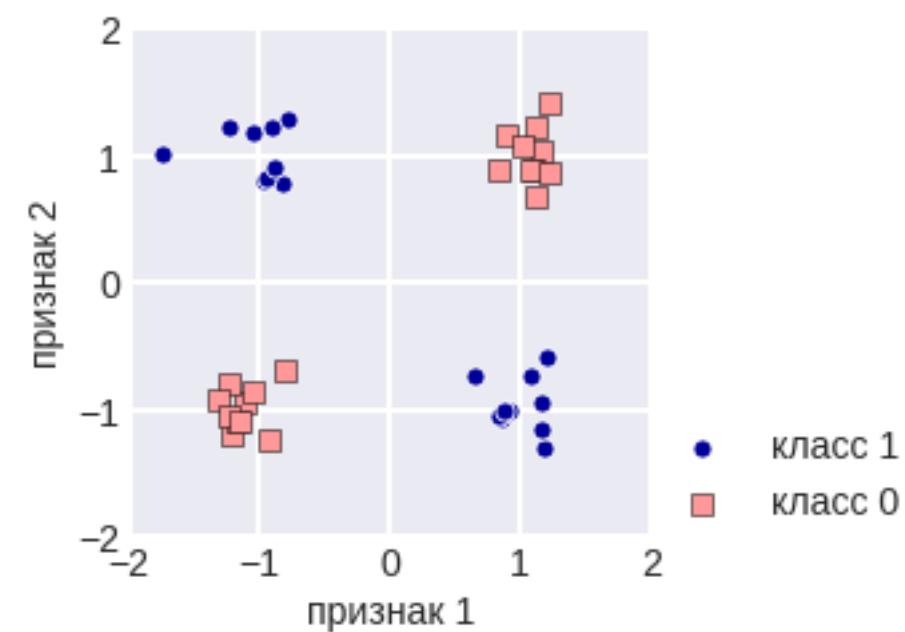
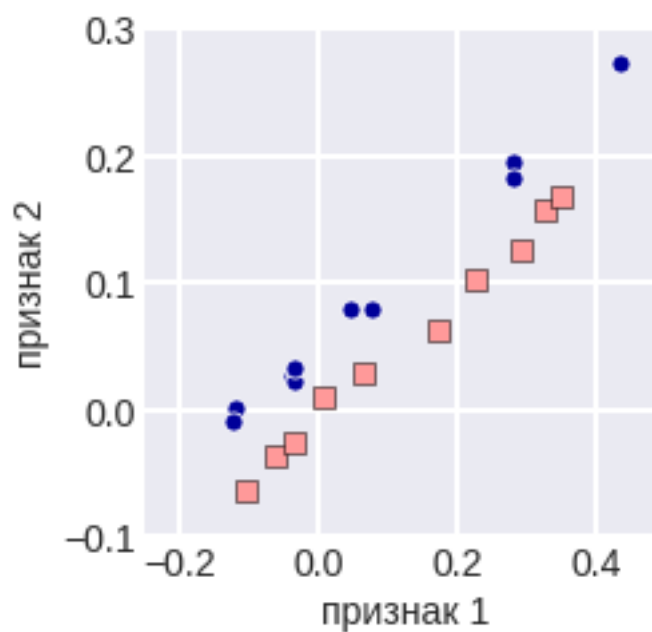
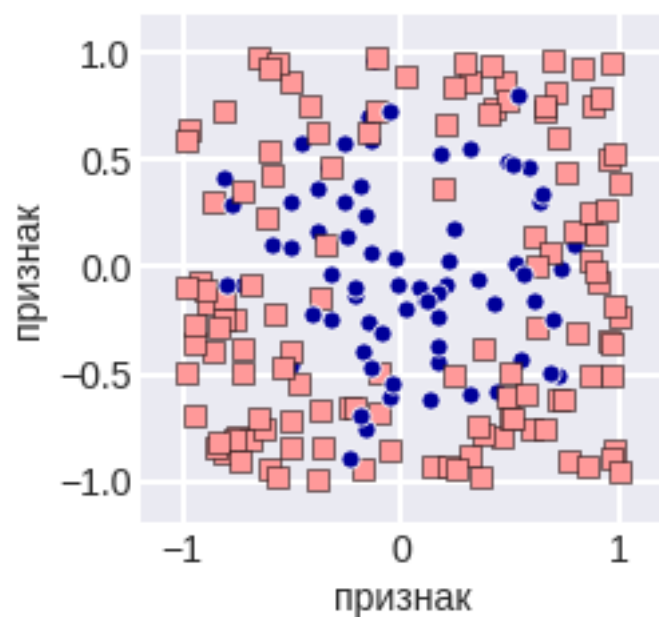
User guide: See the [Feature selection](#) section for further details.

<code>feature_selection.GenericUnivariateSelect([...])</code>	Univariate feature selector with configurable strategy.
<code>feature_selection.SelectPercentile([...])</code>	Select features according to a percentile of the highest scores.
<code>feature_selection.SelectKBest([score_func, k])</code>	Select features according to the k highest scores.
<code>feature_selection.SelectFpr([score_func, alpha])</code>	Filter: Select the p-values below alpha based on a FPR test.
<code>feature_selection.SelectFdr([score_func, alpha])</code>	Filter: Select the p-values for an estimated false discovery rate.
<code>feature_selection.SelectFromModel(estimator, *)</code>	Meta-transformer for selecting features based on importance weights.
<code>feature_selection.SelectFwe([score_func, alpha])</code>	Filter: Select the p-values corresponding to Family-wise error rate.
<code>feature_selection.SequentialFeatureSelector(...)</code>	Transformer that performs Sequential Feature Selection.
<code>feature_selection.RFE(estimator, *[, ...])</code>	Feature ranking with recursive feature elimination.
<code>feature_selection.RFECV(estimator, *[, ...])</code>	Recursive feature elimination with cross-validation to select the number of features.
<code>feature_selection.VarianceThreshold([threshold])</code>	Feature selector that removes all low-variance features.
<hr/>	
<code>feature_selection.chi2(X, y)</code>	Compute chi-squared stats between each non-negative feature and class.
<code>feature_selection.f_classif(X, y)</code>	Compute the ANOVA F-value for the provided sample.
<code>feature_selection.f_regression(X, y, *[, center])</code>	Univariate linear regression tests returning F-statistic and p-values.
<code>feature_selection.r_regression(X, y, *[, center])</code>	Compute Pearson's r for each features and the target.
<code>feature_selection.mutual_info_classif(X, y, *)</code>	Estimate mutual information for a discrete target variable.
<code>feature_selection.mutual_info_regression(X, y, *)</code>	Estimate mutual information for a continuous target variable.

Фильтры

- + сложность линейно зависит от числа признаков
- не учитываем алгоритм
- оцениваем отдельные признаки

ещё называют «одномерными методами»



Jundong Li, et al. «Feature Selection: A Data Perspective»

Обёртки: умный перебор признаков подпространств

1	1	1	1	0	1	1	1
3	1	1	1	0	2	1	1
2	1	2	2	3	1	1	2
1	1	2	2	2	0	2	2
0	2	1	3	1	1	1	1
1	2	1	3	1	1	1	1
1	2	2	1	1	0	1	3

1. Выбираем модель алгоритмов

2. Выбираем способ оценки качества модели на признаковом подпространстве (например, скользящий контроль + MSE)

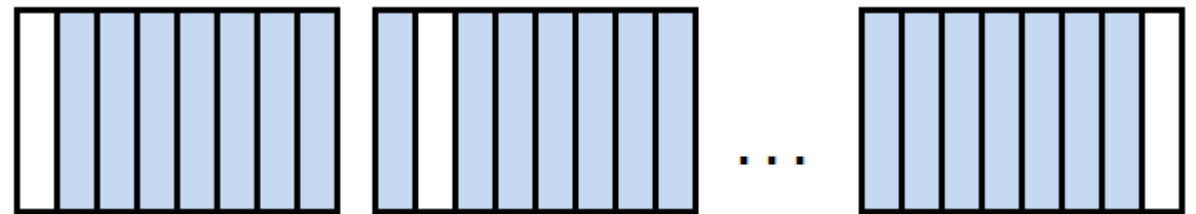
3. Запускаем алгоритм на различных (основной момент – как их перебирать) признаковых подпространствах, оцениваем качество

$$Q(\{f_1, \dots, f_k\}) = Q(A(X[:, [f_1, \dots, f_k]], y))$$

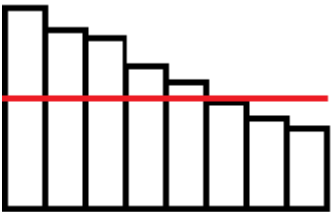
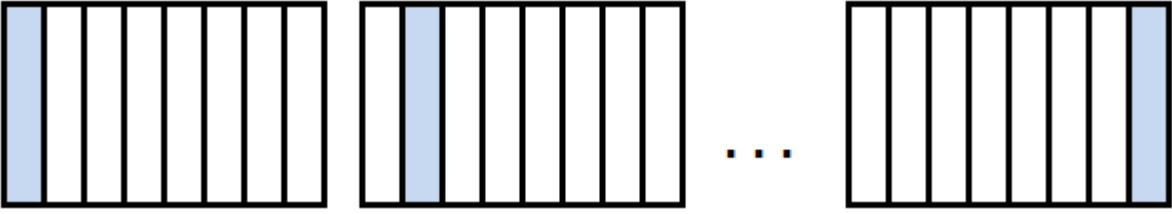
4. Выбираем подпространство с max качеством

Обёртки на практике

Исключение (перестановка) по одному



Качество по отдельным признакам

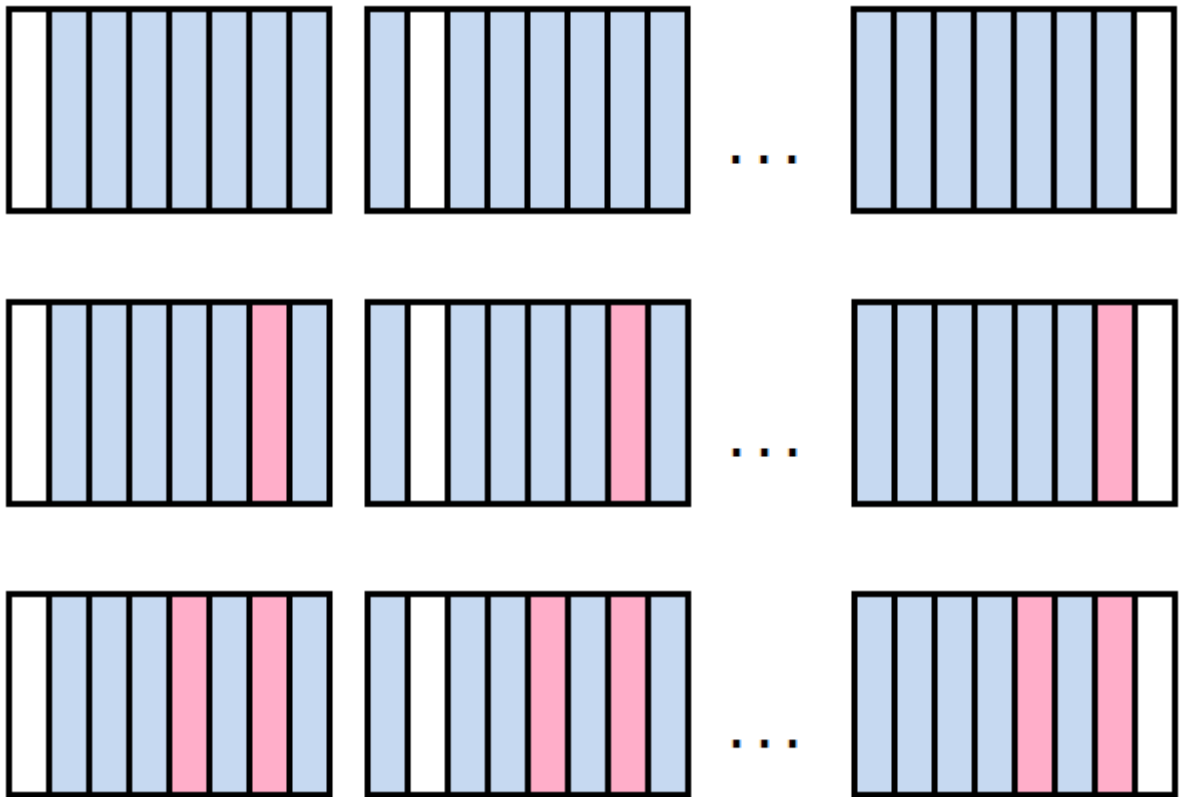


Например, если решаем задачу с одним признаком неглубокими лесами
можем выявить нелинейные закономерности

Обёртки на практике

Backward Stepwise Selection / Del

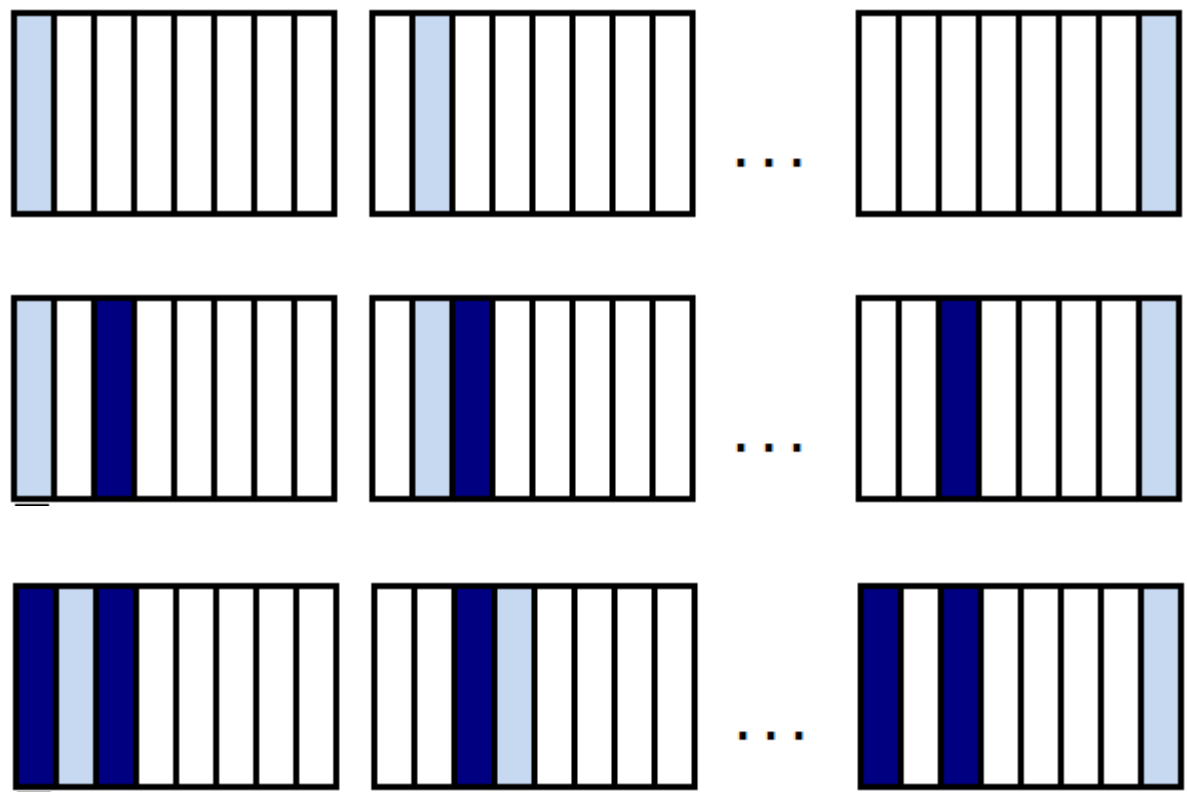
включить все и по одному исключать – см. качество
каждый раз делается «самое выгодное исключение»



Обёртки на практике

Forward Stepwise Selection / Add

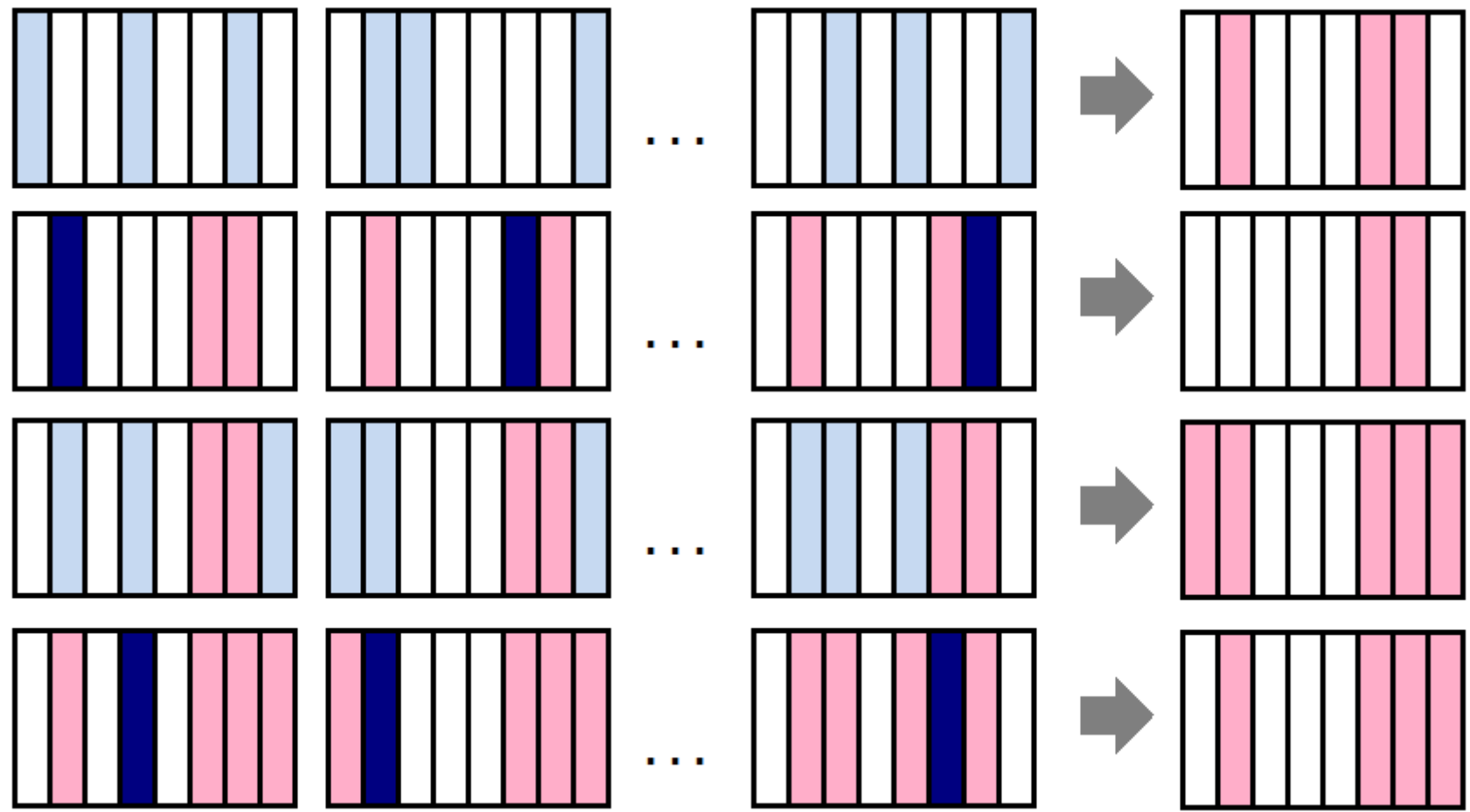
Начать с модели без признаков (null model) и по одному добавлять
делается самое выгодное добавление



stepwise selection / stepwise regression

Обёртки на практике

Add-Del – комбинация предыдущих стратегий

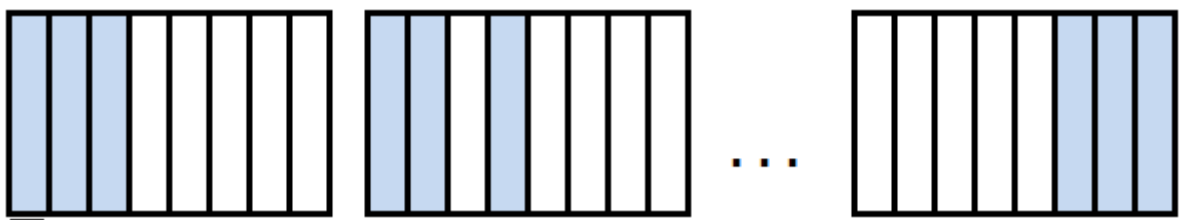


совет из практики $k \times [\text{add } t \text{ random}] + r \times [\text{del } s \text{ random}]$

Обёртки на практике

Subset Selection

перебрать подмножества признаков мощности $\leq k$
выбрать модель с лучшим качеством



Минутка кода: Sequential Feature Selection

реализует стратегии Add / Del

```
from sklearn.feature_selection import SequentialFeatureSelector
from sklearn.neighbors import KNeighborsClassifier
from sklearn.datasets import load_iris
X, y = load_iris(return_X_y=True)
knn = KNeighborsClassifier(n_neighbors=3)
sfs = SequentialFeatureSelector(knn, n_features_to_select=3)
sfs.fit(X, y)

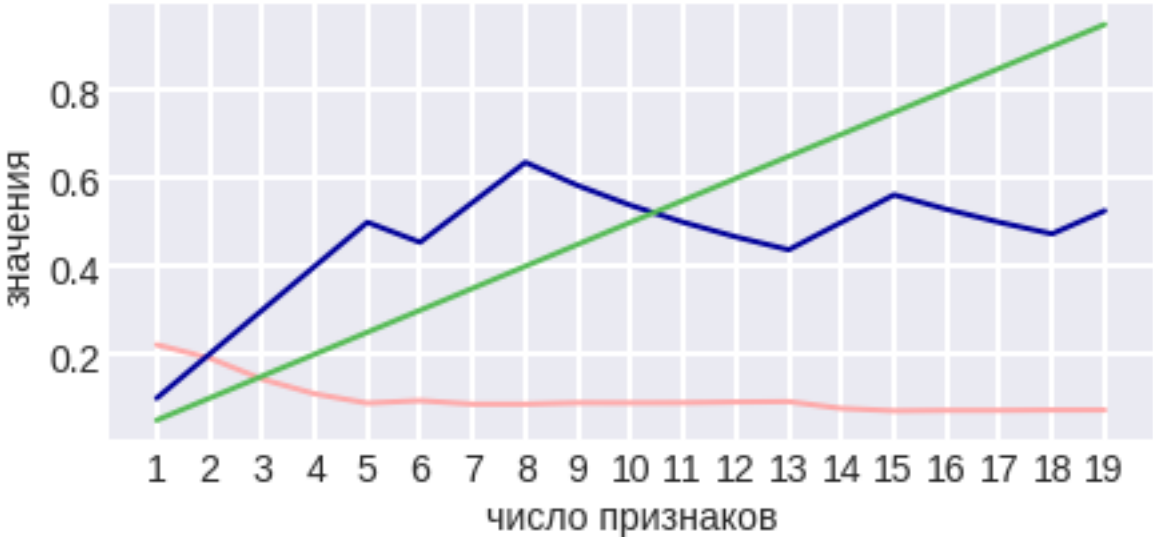
SequentialFeatureSelector(estimator=KNeighborsClassifier(n_neighbors=3),
                          direction='forward', # как выбирать, есть ещё
                          cv=5,
                          n_features_to_select=3) # сколько признаков выбрать

sfs.get_support()
array([ True, False,  True,  True])

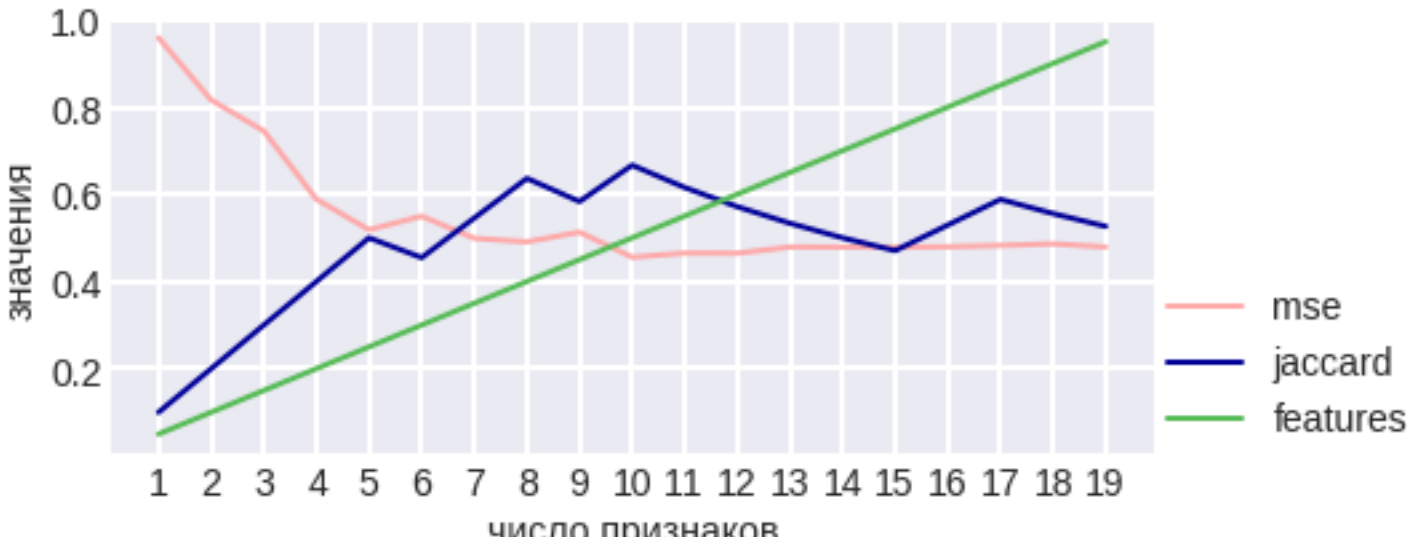
sfs.transform(X).shape
(150, 3)
```

sklearn.feature_selection.RFECV – Recursive feature elimination + CV
(для моделей с коэффициентами или важностями)

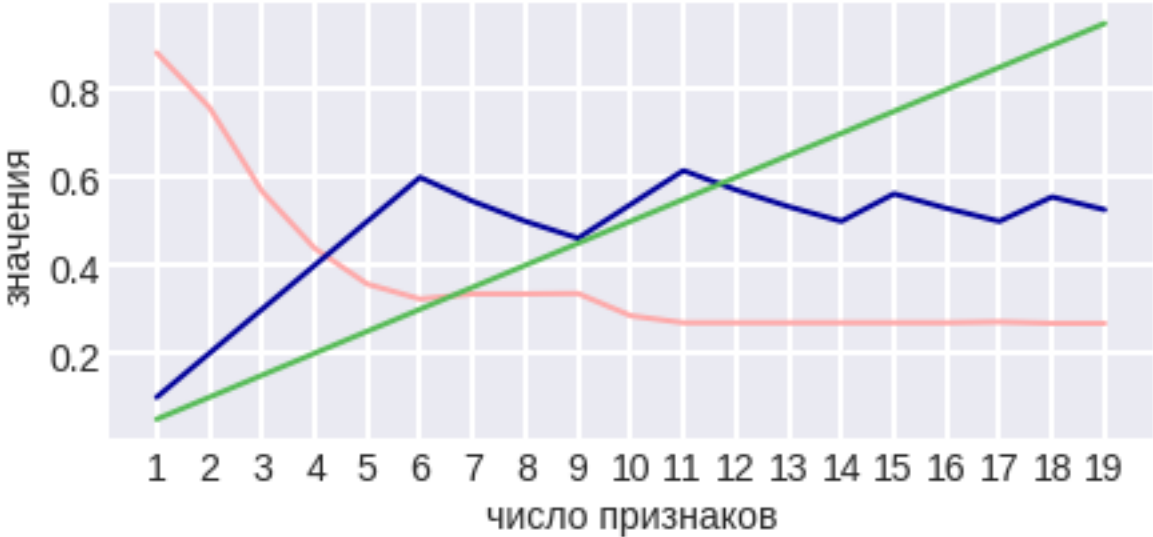
Эксперименты с линейной закономерностью



Ridge



LGBM



Lasso



RF

На что смотрят при селекции

1. Качество решения задачи

2. Штраф за сложность
(CV, Cp – AIC, BIC и т.п.)

3. Стабильность признакового пространства

4. Зависимости признаков

Штраф за сложность

Информационный критерий Акаике
(для моделей, которые максимизируют правдоподобие)

$$\text{AIC} = -2\log L + 2n$$

$$C_p = \frac{1}{m}(\text{RSS} + 2n\hat{\sigma}^2)$$

Байесовский информационный критерий

$$\text{BIC} = n\log m - 2\log L$$

$$C_p = \frac{1}{m}(\text{RSS} + \log(m)n\hat{\sigma}^2)$$

некоторые критерии могут использоваться и для обучения без учителя

Штраф за сложность ■

Adjusted R^2

$$R_{\text{adjusted}}^2 = 1 - \frac{\text{RSS}/(m - n - 1)}{\text{TSS}/(n - 1)}$$

TSS = the total sum of squares
тут нужно большое значение

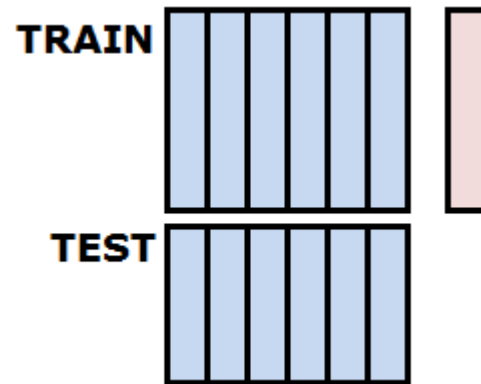
MDL (Minimum Description Length)

$$\text{length}(A) = -\log P(Y | X, w, A) - \log P(w | A)$$

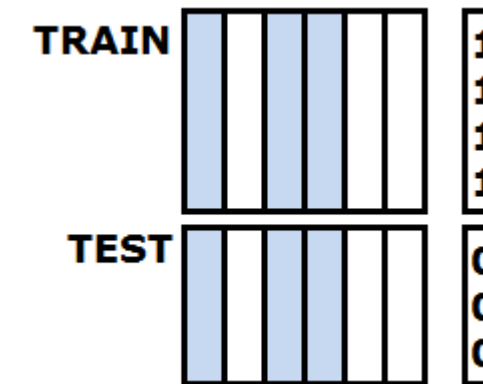
Нестабильность признакового пространства

– качество решения задачи разделения выборки на обучение и тест

Исходная задача



Новая задача



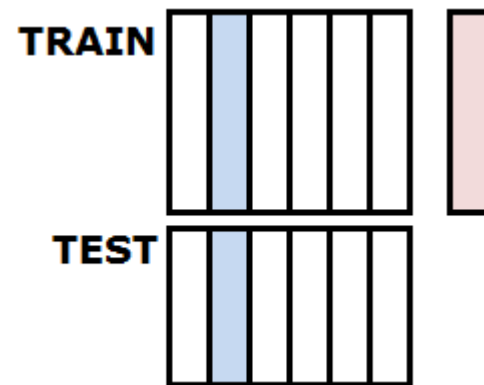
AUC ROC

- получаем оценку схожести обучения и контроля
- важности признаков – оценки их нестабильности
- отбор признаков – поиск стабильного признакового пространства

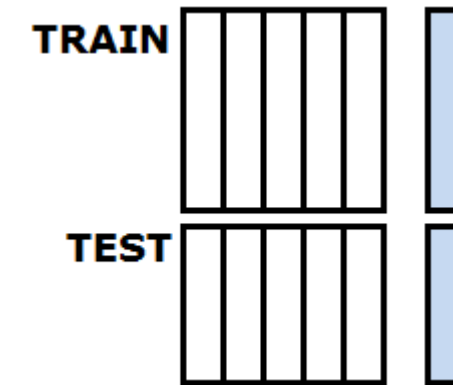
Зависимость (выводимость) признака

– качество решения задачи вывода признака

Исходная задача



Новая задача



Функционал?

- получаем оценку зависимости признака от остальных
- можно анализировать специальные зависимости (ex: линейные)
 - можем последовательно удалять лишние признаки
 - можно добавлять признаки к базовым

Отбор признаков как задача глобальной оптимизации

1	1	1	1	0	1	1
3	1	1	1	0	2	1
2	1	2	2	3	1	1
1	1	2	2	2	0	2
0	2	1	3	1	1	1
1	2	1	3	1	1	1
1	2	2	1	1	0	1

1
1
2
2
1
1
3

0	1	1	1	0	1	0
---	---	---	---	---	---	---

Максимизация функции

$f : \{0,1\}^n \rightarrow \mathbb{R}$

Заведомо нет лучшего алгоритма
пример функции с точечным носителем

Три группы методов:

- 1. Перебор
- 2. Направленный поиск
- 3. Стохастическая оптимизация

Полный / квазиполный перебор

– может не завершиться

+ можно грамотно организовать

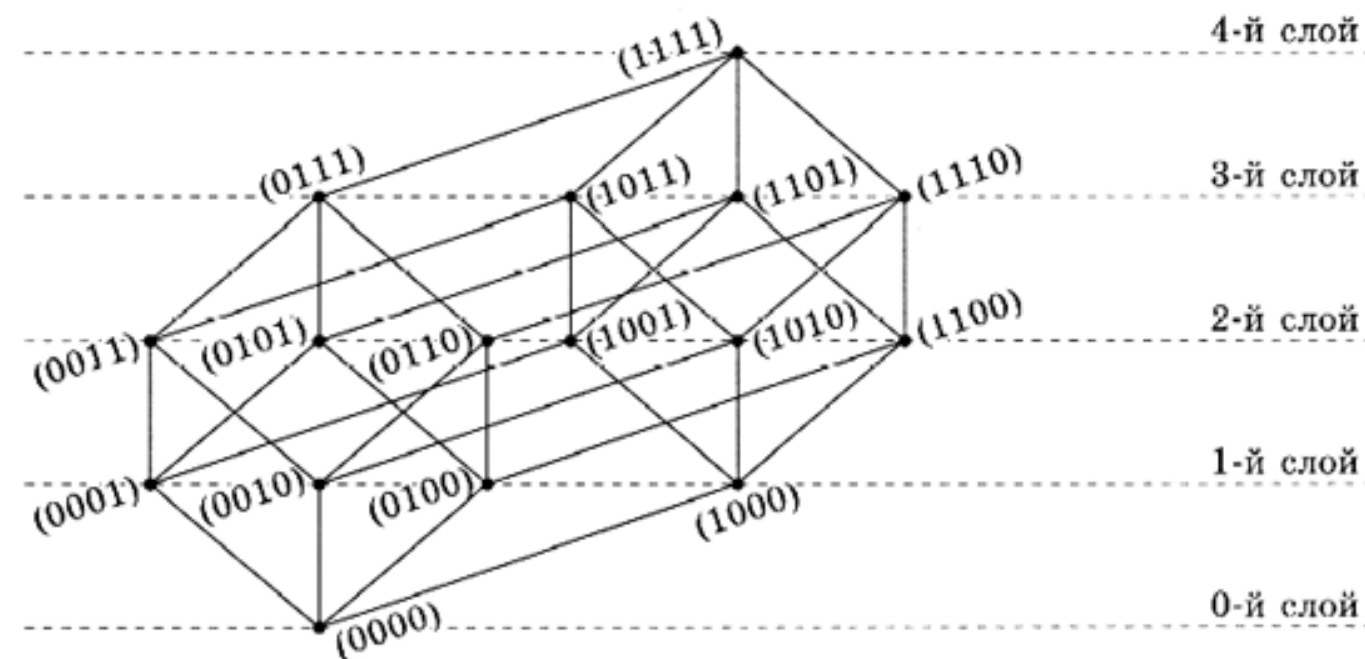
- сначала потенциально лучшие точки
- устанавливать свойства функции

(монотонность, несущественность, эквивалентность переменных и т.п.)

Пример

- перебор всех троек признаков
- удалить те, которые не попали в хорошие подпространства

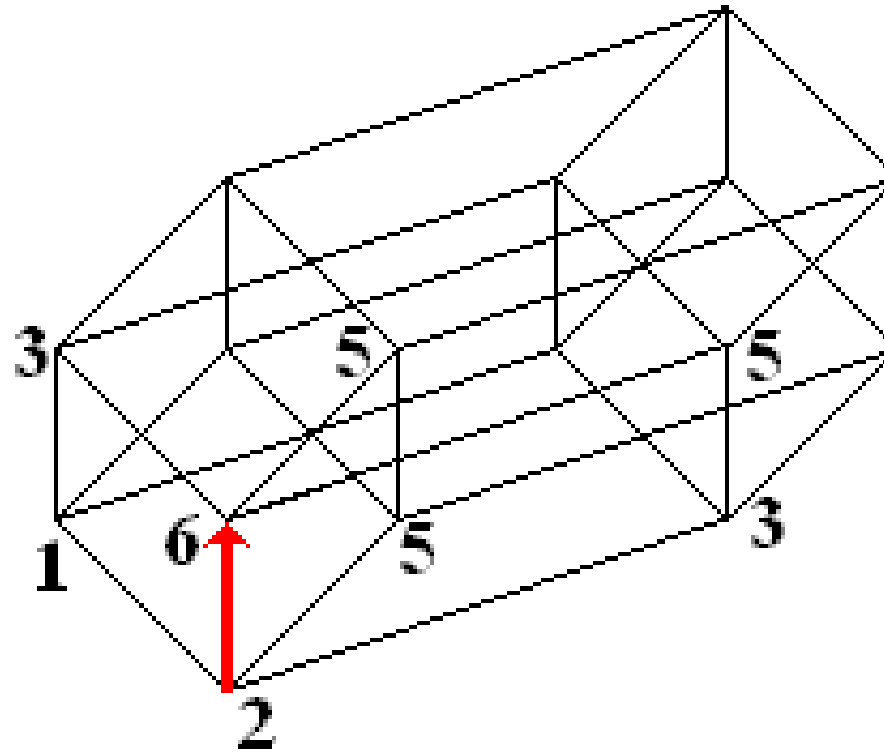
Направленный поиск



точки для перебора выбираем из окрестности уже исследованных точек

- **градиентный (жадный) алгоритм**
 - **симуляция отжига**
- **метод луча (beam search)**
 - **локальный поиск**

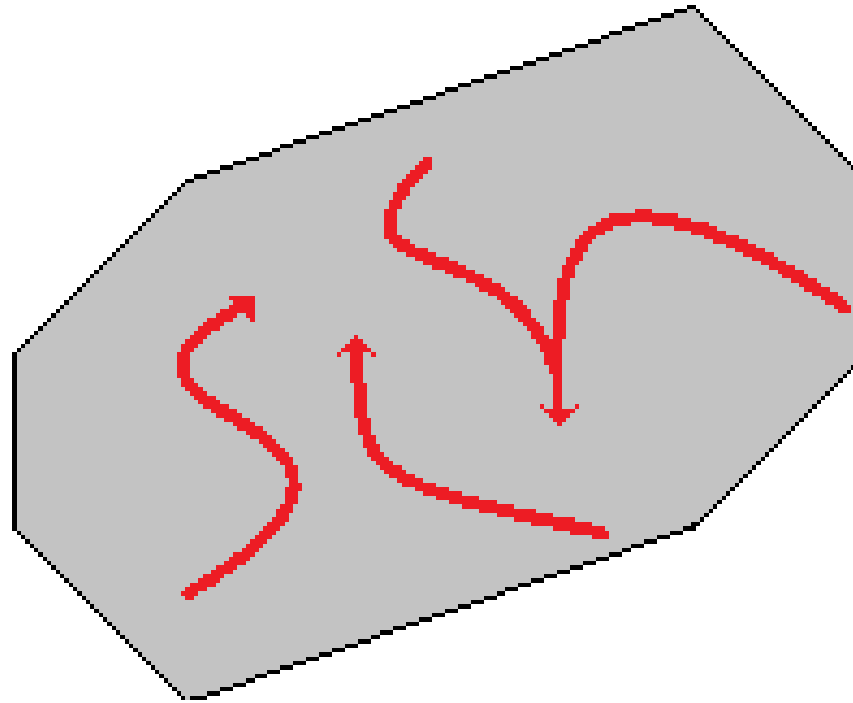
Направленный поиск: градиентный (жадный) алгоритм



1. Начинаем со случайной точки
2. Ищем в окрестности текущей точки наибольшее значение
3. Переходим в соответствующую точку

Останавливаемся в локальном максимуме

Направленный поиск: улучшения градиентного алгоритма

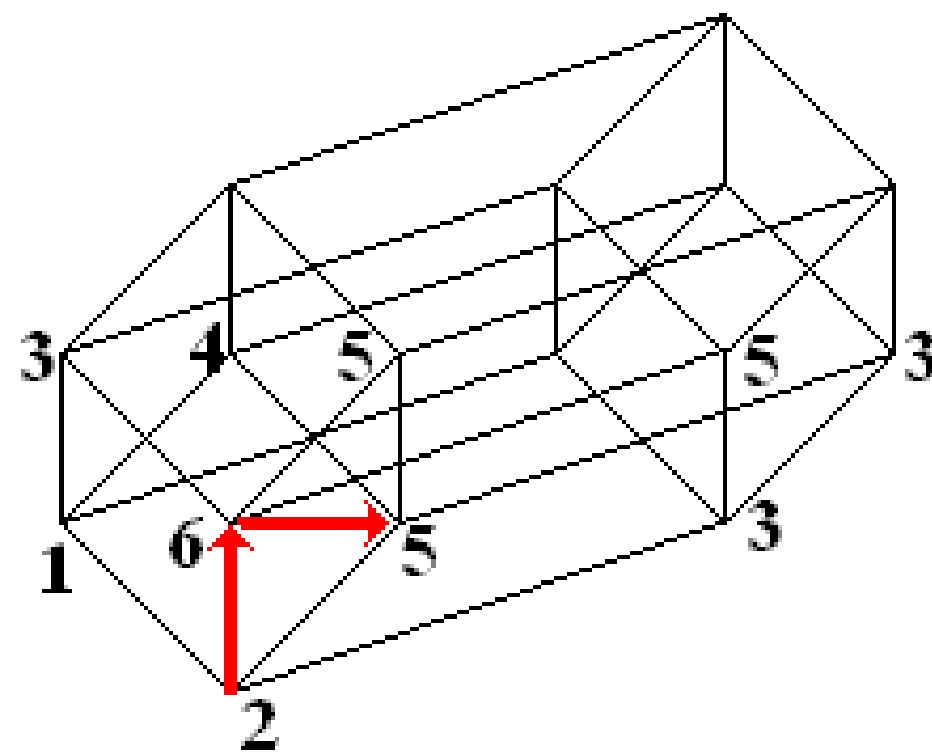


1. Перезапуски
2. Параллельные запуски с переключением на перспективные ветки
3. Продолжать движение в локальных максимумах – **симуляция отжига**

$$\exp([f(\tilde{z}^t) - f(\tilde{z})]/T)$$

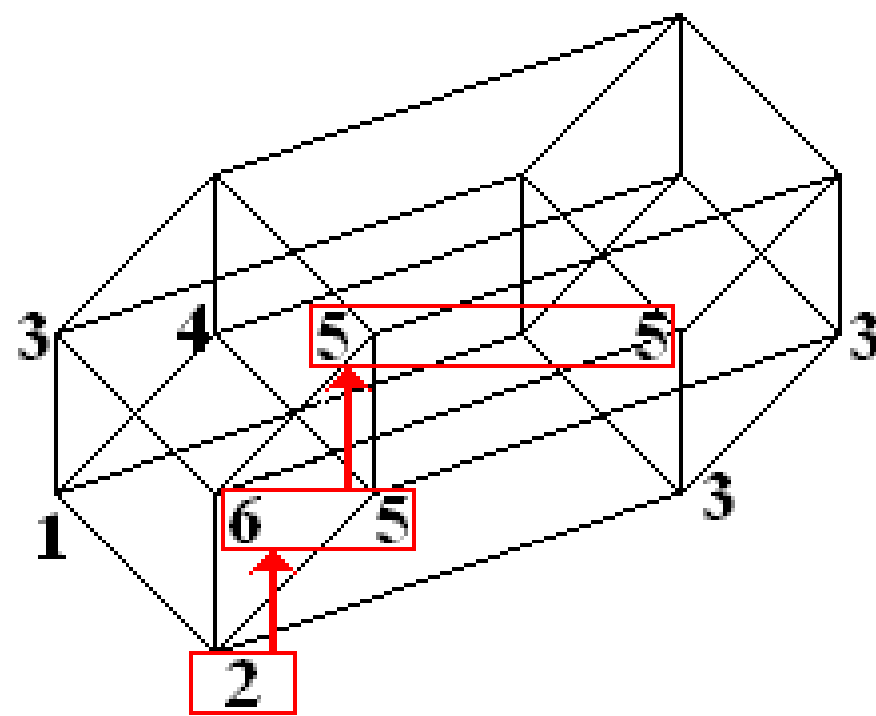
4. Идти в сторону к лучшим (запоминать, что посетили)
5. Также собирать информацию о функции

Направленный поиск: метод луча (beam search)



Храним k лучших точек

Направленный поиск: локальный поиск

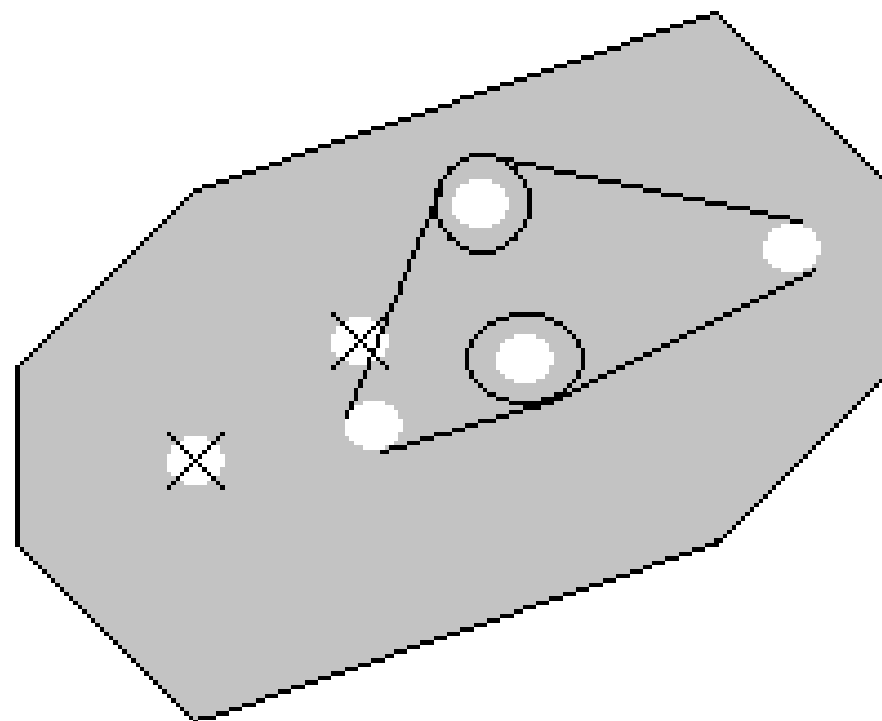


1. Стартуем с $\{(0,0,...,0)\}$

2. Среди соседей текущего множества выбираем k лучших соседей верхнего уровня

Стохастическая оптимизация: генетический алгоритм

1. Инициализация.
2. Селекция.
3. Скрещивание (размножение).
4. Мутации.
5. Переход к п. 2.



1010101110010

0110100101011

1010101101011

Стохастическая оптимизация: улучшения генетического алгоритма

Селекция

- **смерть от старения**
 - **отбор по вероятности (оценка ~ вероятность смерти)**
 - **смерть в боях (турниры)**
 - **приход чужаков**
 - **параллельно живущие популяции**
-

Скращивание

- **разные схемы кроссовера**
- **разный выбор для скрещивания (все по парам, с вероятностями)**
- **алгоритм с постоянным числом индивидов (дети вместо родителей)**
- **конвейерная версия (0.1 – выживает, 0.9 – случайная пара переносит потомков)**

Стохастическая оптимизация: улучшения генетического алгоритма

Мутация

- лучшие не мутируют (элитаризм)
- вероятность мутации выше, если нет улучшений
- генетика + градиент

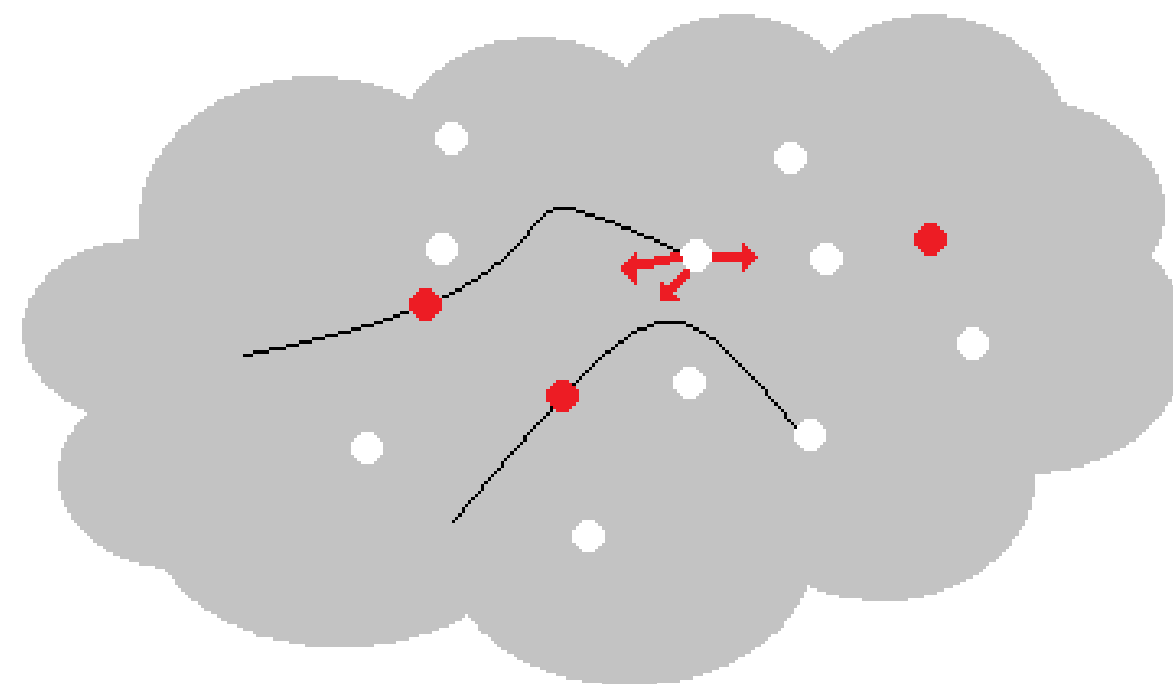
Кодирование особей

Число	Стандартный код	Код Грея
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Стохастическая оптимизация: роевой алгоритм

$$f : \mathbb{R}^n \rightarrow \mathbb{R}$$

- **К своему максимуму**
- **К максимуму роя**
- **К максимуму подруги**



$$x_{t+1}^i = x_t^i + \alpha(m^i - x_t^i) + \beta(m - x_t^i) + \gamma(m^j - x_t^i)$$

Стохастическая оптимизация

Полный перебор	высокая точность при экспоненциальном времени работы
Направленный перебор	+ приемлемая точность + простая реализация + быстрая работа – не полный перебор пространства
Стохастический перебор	– подбор гиперпараметров важен + простая реализация + простые модификации + нет проблемы локальных минимумов

Sean Luke Essentials of Metaheuristics. — Lulu, 2009. — 235 p.

Дилемма исследования–использования «Exploration vs. Exploitation»

задача исследования

просмотреть как можно больше (новых) точек из всего пространства поиска

задача использования

**не пропустить хорошее решение и по максимуму использовать
уже полученную информацию**

часто регулируется параметрами:

- **радиус окрестности в градиентном алгоритме**
 - **вероятность мутации**
 - **...**

Встроенные методы: коэффициенты в регрессии

Линейная регрессия

$$Xw = y$$

Решение линейной регрессии

$$\|Xw - y\|^2 \rightarrow \min$$

Регуляризация по Тихонову

$$\|Xw - y\|^2 + \lambda \|w\|^2 \rightarrow \min$$

LASSO

$$\|Xw - y\|^2 + \lambda_2 \|w\|^2 + \lambda_1 \|w\| \rightarrow \min$$

```
from sklearn.linear_model import Ridge
clf = Ridge(alpha=1.0)
clf.fit(X, y)
```

Нормализация признаков!

Встроенные методы: признаки в деревьях

Решающие деревья автоматически выбирают признаки.

Оценка важности в случайном лесе

1) Насколько уменьшает ошибку леса

$$Q(R, \theta) = \frac{|R|}{m} \left(H(R) - \frac{|R_{\text{left}}|}{|R|} H(R_{\text{left}}) - \frac{|R_{\text{right}}|}{|R|} H(R_{\text{right}}) \right)$$

2) Ухудшение на OOB при перемешивании значений конкретного признака

Минутка кода

```
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectFromModel

X, y = load_iris(return_X_y=True)
X.shape
(150, 4)

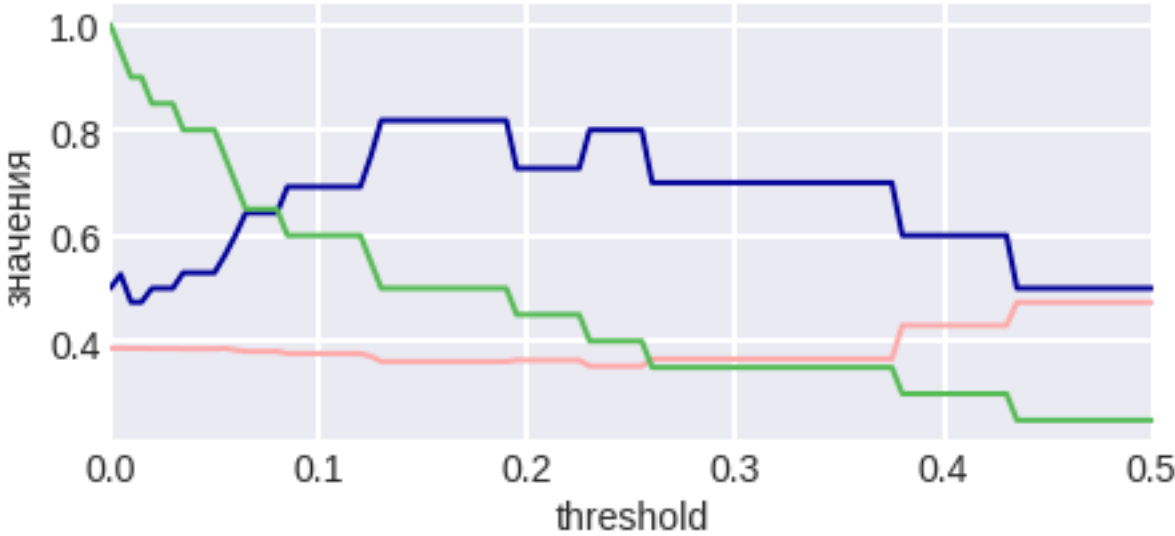
clf = ExtraTreesClassifier(n_estimators=50)
clf = clf.fit(X, y)
clf.feature_importances_
array([ 0.04...,  0.05...,  0.4...,  0.4...])

model = SelectFromModel(clf, prefit=True) # тут есть параметр threshold
                                           # и max_features

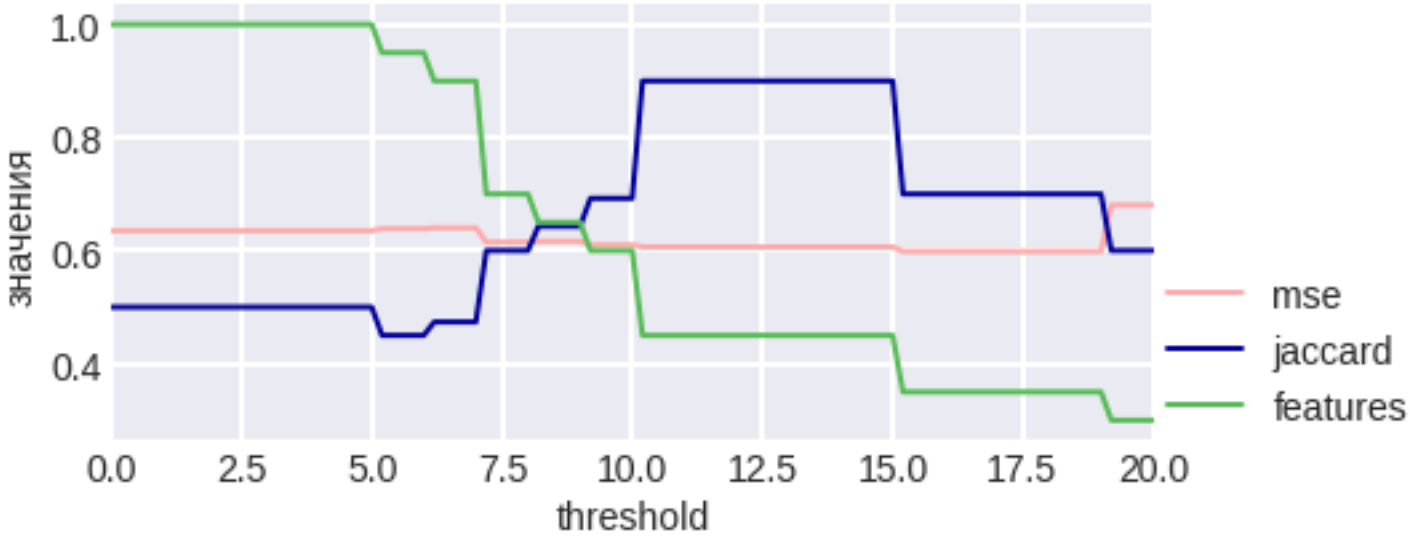
X_new = model.transform(X)
X_new.shape
(150, 2)
```

SelectFromModel – выбираем по порогу на основе весов или важностей

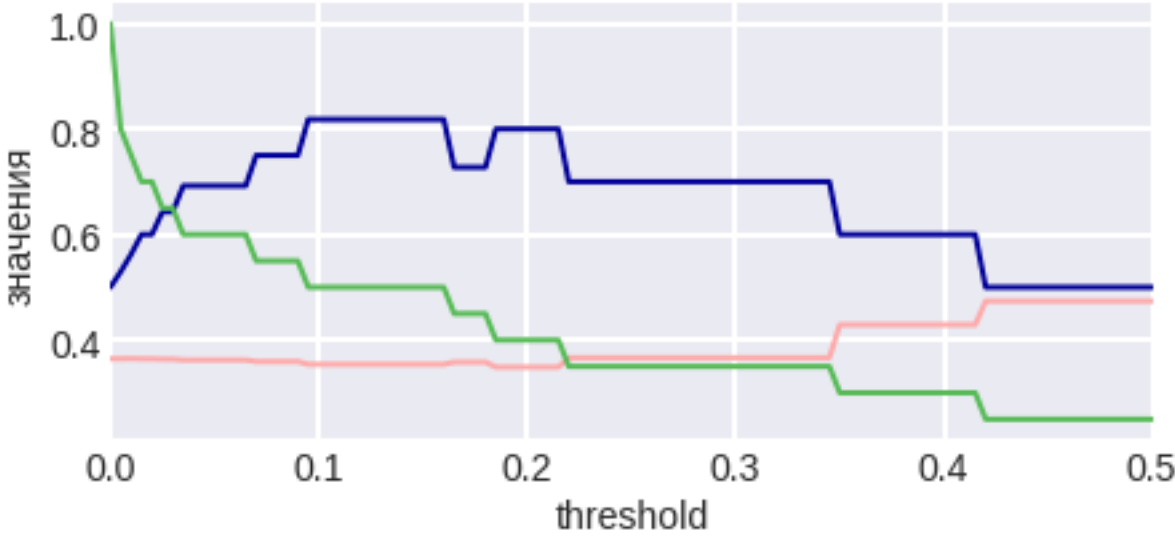
Эксперименты с линейной закономерностью



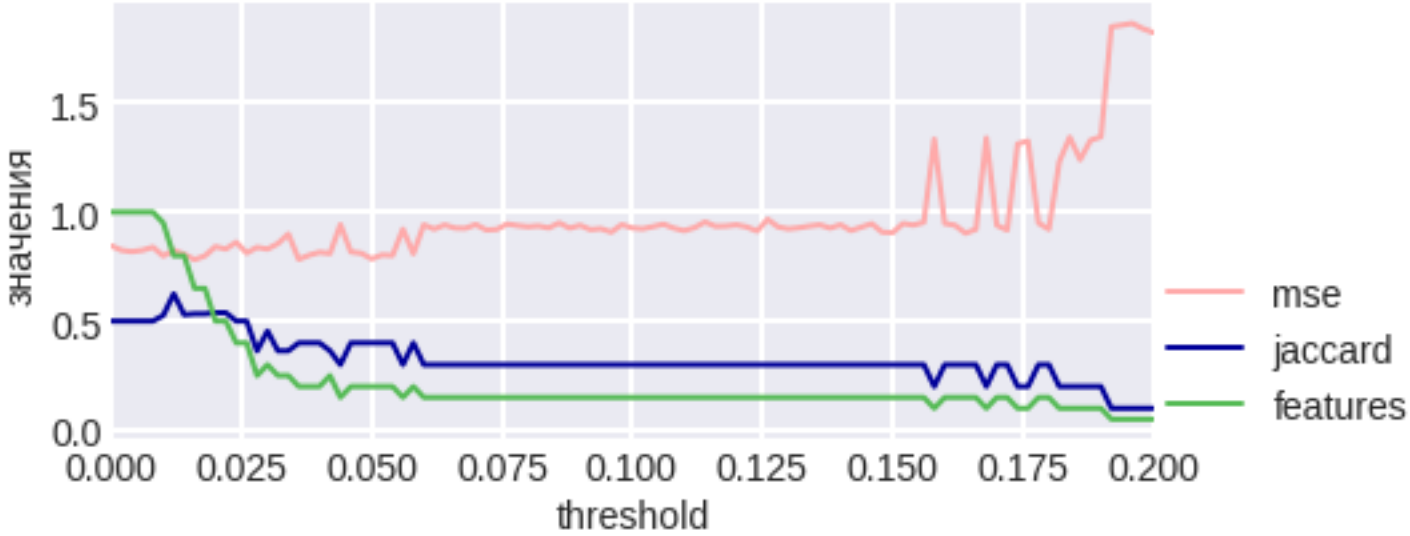
Ridge



LGBM

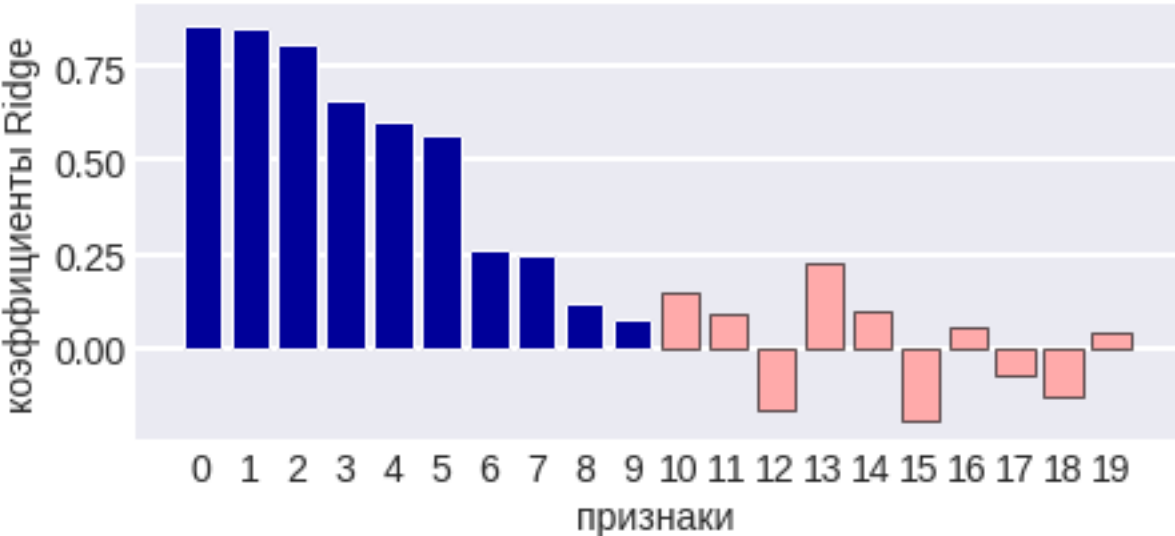


Lasso



RF

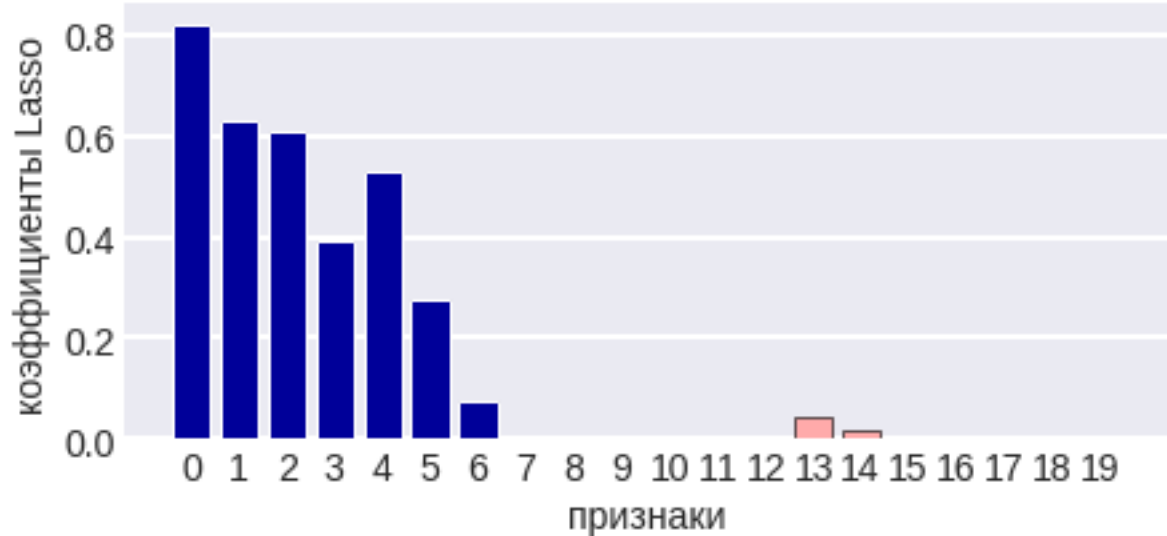
Эксперименты с линейной закономерностью



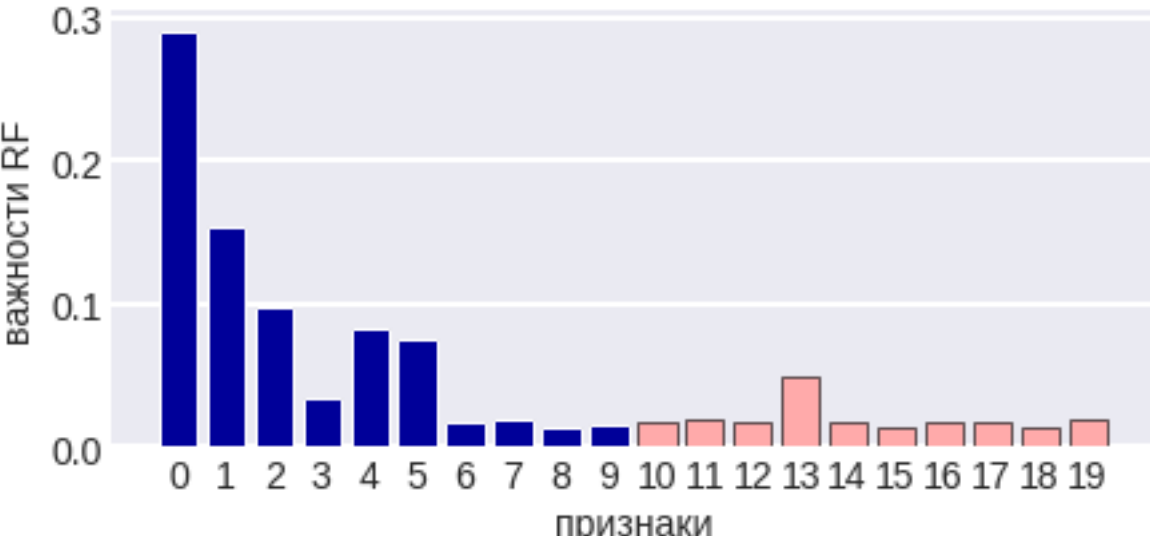
Ridge



LGBM



Lasso



RF

Проблемы селекции

1 проблема – как оценивать качество (было)

Важное замечание:

2 проблема – селекция – тоже подгонка под выборку!

```
clf = Pipeline([
    ('feature_selection', SelectFromModel(LinearSVC(penalty="l1"))),
    ('classification', RandomForestClassifier())
])
clf.fit(X, y)
```

Отдельная тема – «важности признаков»

Литература

Дэн Саймон «Алгоритмы эволюционной оптимизации», 2020.

Sean Luke «Essentials of Metaheuristics» — Lulu, 2009. — 235 p.
<https://cs.gmu.edu/~sean/book/metaheuristics/Essentials.pdf>

Дьяконов, А. Г. «Анализ данных, обучение по прецедентам, логические игры, системы WEKA, RapidMiner и MatLab (практикум на эвм кафедры математических методов прогнозирования)» — МАКСПресс, 2010. — 278 с.
<http://www.machinelearning.ru/wiki/images/7/7e/dj2010up.pdf>

