Классификация данных. Основные понятия

Курс «Интеллектуальные информационные системы»

Кафедра управления и интеллектуальных технологий

НИУ «МЭИ»

Предобработка данных

- Выбор информативных признаков и их предобработка – полпути к успеху. Garbage in – garbage out.
- Что делать с пропущенными значениями?
- Как обработать категориальные признаки?
- Что делать с текстовыми данными – рассмотрим в следующих лекциях.
- Масштабирование данных
- Преобразование признаков к оптимальному типу данных

Пропущенные значения

1) Не использовать (удалить) признак с большим количеством пропусков

2) Заменить пропущенные данные:
   - нулями
   - средним\медианой по выборке
   - средним\медианой по кластеру
   - экспертным значением

```
dataset = pd.read_csv('dataset.csv')
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 907813 entries, 0 to 907812
Data columns (total 37 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   timestamp               907813 non-null  object
 1   src_ip                  907813 non-null  object
 2   src_port                907813 non-null  float64
 3   dst_ip                  907813 non-null  object
 4   dst_port                907813 non-null  float64
 5   request_http_method     907813 non-null  object
 6   request_http_request    907813 non-null  object
 7   request_http_protocol   907813 non-null  object
 8   request_user_agent      903981 non-null  object
 9   request_referer         291394 non-null  object
 10  request_host            907781 non-null  object
 11  request_origin          1609 non-null    object
 12  request_cookie          445262 non-null  object
 13  request_content_type    28544 non-null   object
 14  request_accept          381783 non-null  object
 15  request_accept_language 15860 non-null   object
 16  request_accept_encoding 366300 non-null  object
 17  request_do_not_track    907813 non-null  float64
 18  request_connection      482860 non-null  object
 19  request_body            26743 non-null   object
 20  response_http_protocol  907813 non-null  object
```

Категориальные признаки

Нужно как-то закодировать. Обычно:

А) унитарное кодирование (one-hot encoding, OHE)
Б) Label encoding
В) Binary encoding
Г) Экспертные подходы

Порядковые признаки – среднее между категориальным и количественным. Нужно перевести эти признаки либо к одним, либо к другим.

Масштабирование признаков

Цель – уравнять влияние признаков, изменяющихся в разных диапазонах.
Особо важно:

- при использовании Евклидова расстояния! (см. далее)
- для метрических методов построения моделей.

Нормализация: $$x_{i,norm} = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

Стандартизация: $$x_{i,stand} = \frac{x_i - x_{mean}}{x_{std}}$$

Меры близости и расстояния

Евклидово расстояние $$d(\vec{X}_j, \vec{X}_l) = \sqrt{\sum_{i=1}^{M} (x_j^{(i)} - x_l^{(i)})^2}$$

Расстояние городских кварталов $$d(\vec{X}_j, \vec{X}_l) = \sum_{i=1}^{M} | x_j^{(i)} - x_l^{(i)} |$$

Косинусная мера близости

$$d(\vec{X}_j, \vec{X}_l) = \cos(\vec{X}_j, \vec{X}_l) = \frac{\sum_{i=1}^{M} x_j^{(i)} x_l^{(i)}}{\sqrt{\sum_{i=1}^{M} (x_j^{(i)})^2 \sum_{i=1}^{M} (x_j^{(i)})^2}}$$

```
import sklearn.metrics.pairwise as pw

D = [[6,0,0,3,3]]
E = [[3,0,0,2,2]]
F = [[1,1,1,1,1]]

print('Euclidean E-D: \t',pw.euclidean_distances(D, E))
print('Euclidean E-F: \t',pw.euclidean_distances(E, F))

print('\nCosine: E-D \t',pw.cosine_similarity(D, E))
print('Cosine E-F: \t',pw.cosine_similarity(E, F))

print('\nManhattan: E-D \t',pw.manhattan_distances(D, E))
print('Manhattan E-F: \t',pw.manhattan_distances(E, F))
```

```
Euclidean E-D:   [[3.31662479]]
Euclidean E-F:   [[2.82842712]]

Cosine: E-D      [[0.99014754]]
Cosine E-F:      [[0.7592566]]

Manhattan: E-D   [[5.]]
Manhattan E-F:   [[6.]]
```

Пример обработки выборки

```
import pandas as pd
dataset = pd.DataFrame({'A': [1 , 2, None, 2],
                        'B': ['red', 'red', 'yellow', 'green'],
                        'C': [3300, 1250, 4600, 4500],
                        'D': ['MSK', 'SPB', 'EKB', 'MSK']})
dataset
```

]:

	A	B	C	D
0	1.0	red	3300	MSK
1	2.0	red	1250	SPB
2	NaN	yellow	4600	EKB
3	2.0	green	4500	MSK

```
# OHE encoding
dataset = pd.get_dummies(dataset, columns = ['B'])
dataset
```

]:

	A	C	D	B_green	B_red	B_yellow
0	1.0	3300	MSK	0	1	0
1	2.0	1250	SPB	0	1	0
2	NaN	4600	EKB	0	0	1
3	2.0	4500	MSK	1	0	0

```
# Label encoding
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
le.fit(dataset['D'])
```

]: LabelEncoder()

```
dataset['D'] = le.transform(dataset['D'])
dataset
```

]:

	A	C	D	B_green	B_red	B_yellow
0	1.0	3300	1	0	1	0
1	2.0	1250	2	0	1	0
2	NaN	4600	0	0	0	1
3	2.0	4500	1	1	0	0

Пример обработки выборки

```
# Заполняем пропущенные данные
dataset['A'] = dataset['A'].fillna(np.mean(dataset['A']))
dataset
```

]:

	A	C	D	B_green	B_red	B_yellow
0	1.000000	3300	1	0	1	0
1	2.000000	1250	2	0	1	0
2	1.666667	4600	0	0	0	1
3	2.000000	4500	1	1	0	0

```
dataset['C_normalized'] = (dataset['C'] - dataset['C'].min()) / (dataset['C'].max() - dataset['C'].min())
dataset['C_standardized'] = (dataset['C'] - dataset['C'].mean()) / dataset['C'].std()
dataset
```

]:

	A	C	D	B_green	B_red	B_yellow	C_normalized	C_standardized
0	1.000000	3300	1	0	1	0	0.611940	-0.072209
1	2.000000	1250	2	0	1	0	0.000000	-1.388018
2	1.666667	4600	0	0	0	1	1.000000	0.762206
3	2.000000	4500	1	1	0	0	0.970149	0.698021

Задача классификации

Задача классификации – отнести новый объект к одному из заранее определенных классов на основе некоторой функции (алгоритма, решающего правила, классификатора)

Виды классификации:

- Бинарная классификация (классификация на 2 класса, k=2)
- На k непересекающихся классов (k>2)
- На k классов, которые могут пересекаться
- Одноклассовая

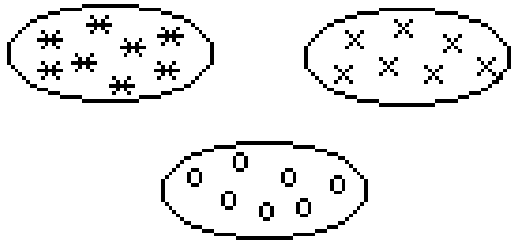
Формирование выборок

- Эффективность методов Machine Learning сильно зависит от того, как были сформированы выборки.
- Выборки должны быть:
  - Независимо извлеченными из генеральной совокупности
  - Представительными (репрезентативными)
  - Содержать минимум нетипичных объектов
- Не так важно, как выглядит генеральная совокупность во всем пространстве признаков. Гораздо важнее, как она выглядит в районе границы между двумя классами

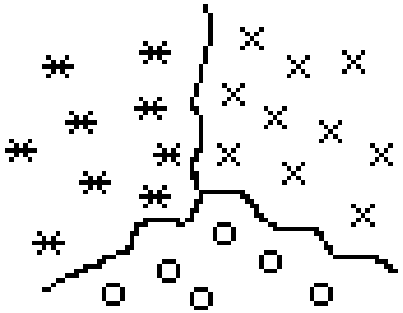
Неидеальность разметки объектов – разные эксперты могут отнести один объект к разным классам. Как поступать?

Как оценить выборку?

Ядерная (центроидная) модель



Модель рассеяния



Модель засорения

Средняя сумма внутриклассовой дисперсии:

$$Q_1 = \frac{1}{N_k}\sum_{j=1}^{N_k} d^2(\vec{X}_j, \vec{X}_k) \quad \rightarrow \quad Q_1^* = \frac{1}{M}\sum_{k=1}^{M}\frac{1}{N_k}\sum_{j=1}^{N_k} d^2(\vec{X}_j, \vec{\bar{X}}_k),$$

Средняя сумма квадратов внутриклассовых попарных расстояний

$$Q_2 = \frac{1}{N_k}\sum_{l=1}^{N_k}\sum_{j=1, j\neq l}^{N_k} d^2(\vec{X}_l, \vec{X}_j) \quad \rightarrow \quad Q_2^* = \frac{1}{M}\sum_{k=1}^{M}\frac{1}{N_k}\sum_{l=1}^{N_k}\sum_{j=1, j\neq l}^{N_k} d^2(\vec{X}_l, \vec{X}_j)$$

Как оценить выборку? (2)

Средняя сумма квадратов межклассовых попарных расстояний

$$Q_3 = \frac{1}{N_k N_s} \sum_{l=1}^{N_k} \sum_{j=1, j\neq l}^{N_s} d^2(\vec{X}_l, \vec{X}_j) \quad \text{или} \quad Q_3 = \frac{1}{M} \sum_{k=1}^{M} \sum_{s=1, s\neq k}^{M} \frac{1}{N_k N_s} \sum_{l=1}^{N_k} \sum_{j=1, j\neq l}^{N_s} d^2(\vec{X}_l, \vec{X}_j)$$

Обобщенный функционал

$$Q_4 = \frac{Q_3}{Q_2}$$

На основе такого анализа исследователь может: 1) объединить несколько близких небольших классов в один; 2) удалить “нехарактерные” шумовые элементы, расположенные вдалеке от центра классов (модель засорения); 3) заново сформировать выборку, увеличив (уменьшив) количество классов или количество элементов.

Свойства сформированных выборок

- любая обучающая выборка конечного размера не является полной, т.е. не содержит необходимого количества элементов для проведения безошибочной классификации;
- элементы обучающей выборки обычно имеют произвольное распределение в пространстве признаков и, как следствие, получаемые решающие правила могут обладать неодинаковой достоверностью в различных областях изменения параметров;
- выборки, как правило, содержат шумовые (нерелевантные, не относящиеся к указанным классам) элементы, другую противоречивую или ошибочную информацию, которая так или иначе попадает в обучающую выборку.

Виды выборок

Часть размеченных документов оставляют для обучения, часть – для оценки точности метода.

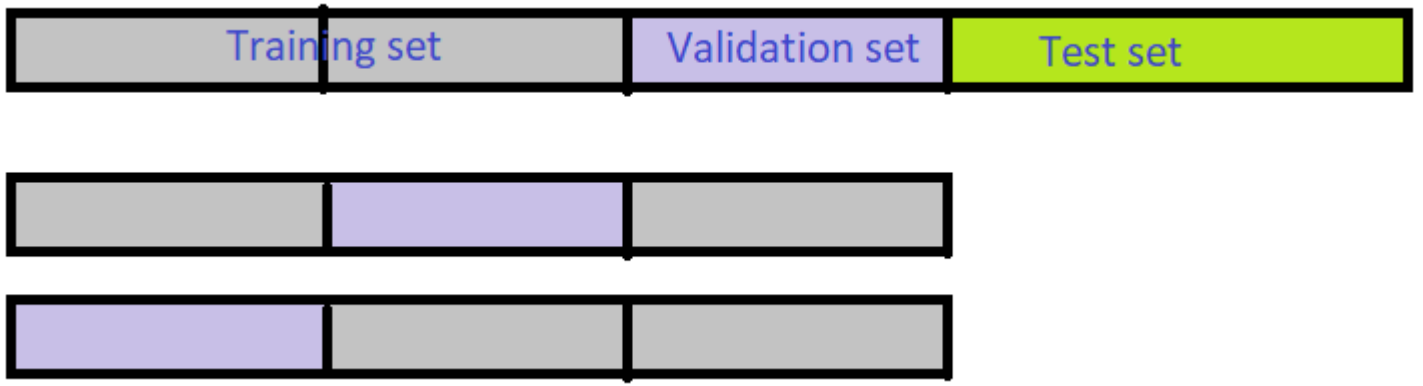
Выборка (dataset) делится на две части:

- Обучающая (тренировочная, training set)
- Тестовая (экзаменационная, test set)

Nобуч > Nтест(обычно 70/30)

Оценка качества классификации в задачах Data Mining

- *Оценка качества с помощью k–кратной перекрестной проверки (k–fold cross validation)*



- *Оценка качества с помощью скользящего контроля (или «метод складного ножа», «Jackknife») – для небольших выборок*
- Bootstrap – имитация статистического выбора. Суть метода заключается в формировании множества выборок на основе случайного выбора с повторениями.

Оценка качества классификации в задачах Text Mining (2)

Ошибка классификации – несовпадение метки, назначенной классификатором с меткой, назначенной экспертом (учителем).

Точность (правильность, аккуратность)

$$\text{Accuracy} = \frac{P}{N}$$

P- количество объектов, по которым классификатор принял правильное решение

$$\text{Точность } Precision = \frac{TP}{TP+FP}$$

$$\text{Полнота Recall} = \frac{TP}{TP+FN}$$

$$\text{F}-\text{measure} = \frac{2(Precision*Recall)}{Precision+Recall}$$

Матрица неточностей (Confusion matrix)

	Оценка эксперта	
Оценка классификатора	Положительная	Отрицательная
Положительная	TP	FP
Отрицательная	FN	TN

Матрица неточностей

$\text{Precision}_k = \frac{A_{kk}}{\sum_{i=1}^{n} A_{ki}}$

$\text{Recall}_k = \frac{A_{kk}}{\sum_{i=1}^{n} A_{ik}}$

Macro average Precision: $= \sum_k \text{Precision}_k$

Micro average Precision: $\frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FN_i}$

Для мультиклассовой классификации:

Micro-avg (F1) = Micro-avg(Pr) = Micro-avg(Rec) = accuracy

Будут отличия для multilabel!

Weighted average Precision:

$= \sum_k (\text{Precision}_k * support\ proportion)$

	0.91	0.96	0.94	0.75	1.00	0.83	0.85	0.97	1.00	0.86	1.00	0.79	1.00	0.75	1.00	1.00	0.96	0.90	0.81	0.89	0.94	0.98	0.86	0.89	0.94	0.92	0.96
0.80		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
0.95	1	94					3														1					1	
1.00	2		32																								
0.29	3			6			3	2		1							1	1			1		1	3		2	
1.00	4				2																						
0.50	5					5															1			2		1	1
0.92	6	1					152			1								1	4	2	3					2	
0.97	7	1		1				256												1	2					2	
0.33	8								1										1							1	
0.97	9									69																	2
0.82	10						2				18										1	1					
0.87	11											34		4										1			
1.00	12												37														
0.57	13											9		12													
0.63	14														5			3									
0.50	15															2				1	1						
0.77	16						2	1									47		1	3	4			2		1	
0.87	17									1							1	69	1	2	5						
0.97	18					1	4			1									197	1							
0.78	19																	2	35	183	13			2		1	
0.97	20						10	3		1										4	702					6	
0.93	21		2																			56		2			
0.29	22			1			2			6									1	1	1		6	2		1	
0.91	23						1											1		3	6			115			
1.00	24																								16		
0.93	25	1						1											2	4	5				1	196	
0.98	26	1																		1							78

Несбалансированная выборка

Пример sklearn

```
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score, roc_auc_score

print('Оценки классфикатора')
print ('Точность: ', accuracy_score(prediction41_2, twenty_test.target))
print(classification_report(prediction41_2, twenty_test.target))

print('Матрица неточностей')
print(confusion_matrix(prediction41_2, twenty_test.target))
```

```
Оценки классфикатора
Точность:  0.8754295532646048
              precision    recall  f1-score   support

           0       0.91      0.88      0.90       407
           1       0.85      0.82      0.83       412
           2       0.86      0.94      0.90       345

    accuracy                           0.88      1164
   macro avg       0.88      0.88      0.88      1164
weighted avg       0.88      0.88      0.87      1164

Матрица неточностей
[[358  41   8]
 [ 31 337  44]
 [  3  18 324]]
```

Несбалансированные выборки

Несбалансированная выборка (Unbalances sample) – количество объектов каждого класса очень сильно разнится.

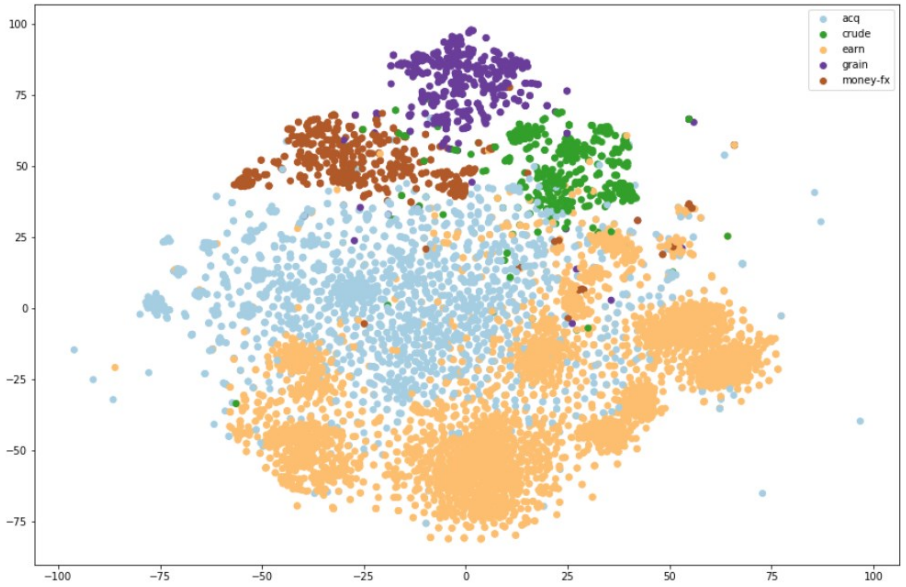
- Низкая точность классификации у малых классов
- Часто классификатору бывает выгодно объекты всех малых классов приписать к самому большому и не обучаться на малых.

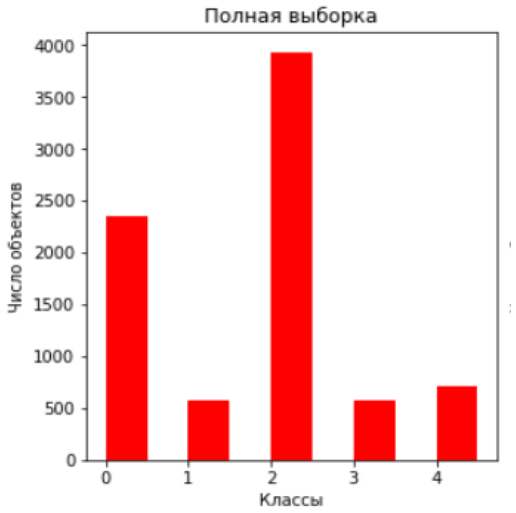
Нужно приводить к сбалансированному виду –

- Oversampling – дублирование объектов малых классов
- Undersampling – удаление объектов из больших классов
- SMOTE (Synthetic Minority Over-sampling Technique) – создание искусственных объектов малых классов

Библиотека для работы с несбалансированными данными: https://imbalanced-learn.org/stable/index.html

Несбалансированные выборки





Несбалансированные выборки (Imbalanced Data) – выборки, в которых количество объектов в классах очень сильно разнится.

Классы, бОльшие по объему – мажоритарные, классы малых объемов – миноритарные.

Следствия:

- Неравномерное обучение на разных классах
- Плохое распознавание объектов малых классов
- Нельзя использовать Accuracy. Используем **weighted-avg.**
- Выгоднее отнести все объекты к мажоритарному классу

- K1 = 10 объектов
- K2 = 100 объектов

$$\text{Accuracy} = \frac{TP+TN}{N}$$

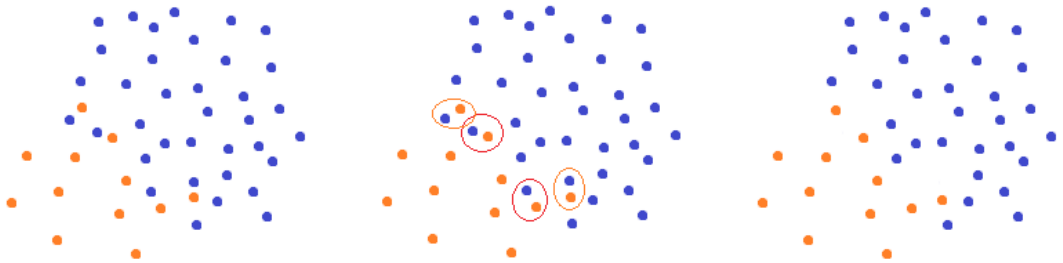
Эксперт: / Модель:	K1	K2
K1	6	16
K2	4	84

Oversampling

- Random Over Samper
- SMOTE – Находит n ближайших соседей, для каждого миноритарного объекта. «Соединяет» их линиями и случайным образом создает новые объекты на этих линиях.
- ADASYN - вместо линейной генерации объектов, добавляется некоторый шум, позволяющий создать нелинейную зависимость между родителем и синтезированным объектом

Undersampling

- Random Under Sampler
- Tomek Links - Пара (E_i,E_j) называется связью Томека, если
- $\begin{cases} d(E_i, E_l) < d(E_i, E_j) \\ d(E_j, E_l) < d(E_i, E_j) \end{cases}$



Объекты мажоритарного класса, входящие в связи Томека удаляются

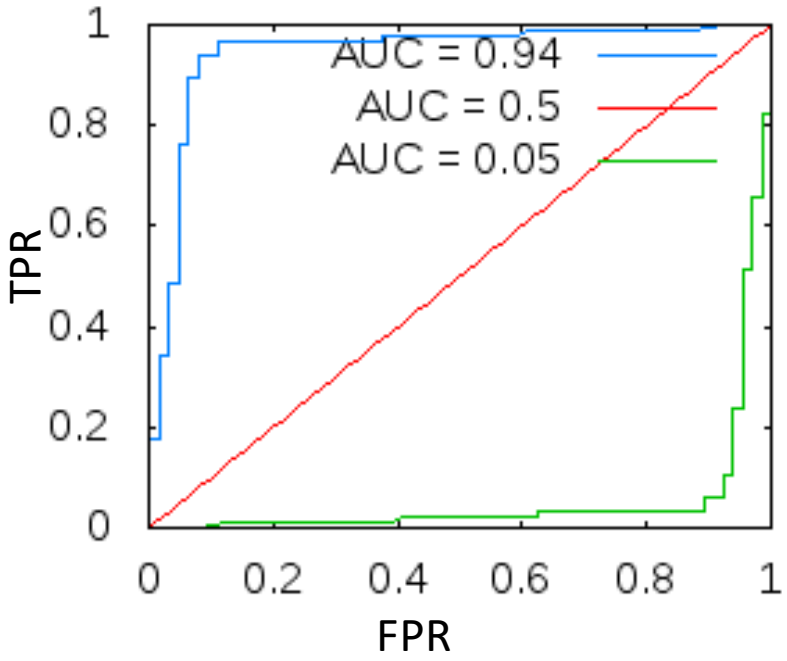
- EditedNearestNeighbors - Объект мажоритарного класса удаляется при условии, если не все его k ближайших соседей имеют метку мажоритарного класса.

AUC ROC

ROC - receiver operating characteristic, кривая ошибок
AUC ROC - площадь под кривой ошибок, Area Under ROC Curve – Зависимость доли верных положительных классификаций от доли ложных положительных классификаций при варьировании порога решающего правила.
Используется для оценки качества бинарной классификации, не чувствителен к несбалансированности.

AUC ROC — эквивалентна вероятности, что классификатор присвоит большее значение случайно выбранному позитивному объекту, чем случайно выбранному негативному объекту.
Когда **AUC = 0.5**, то данный классификатор равен случайному.
Если **AUC < 0.5**, то можно просто перевернуть выдаваемые значения классификатором.

Визуально - чем больше график прижимается к верхнему левому углу, тем больше значение AUC



$$TPR = \frac{TP}{TP+FN} = Recall$$

$$FPR = \frac{FP}{TN + FP}$$

Переобучение

Качество на обучающей выборке

Accuracy a = 0.85
Accuracy a’ = 0,99
} Какой алгоритм лучше?

Проверяем качество на тестовой выборке, которая не участвовала в процессе обучения и настройки параметров:

Accuracy a = 0.84
Accuracy a’ = 0,79

Переобучение (overfitting, overtraining) – нежелательное явление, когда качество классификации на тестовой выборке (или на реальных данных) существенно ниже, чем на обучающей выборке.