Простые методы решения сложных задач

Александр Дьяконов

(ВМК МГУ имени М.В. Ломоносова)

04 декабря 2020 года

План лекции

- волшебные признаки
- понимание, как меняются признаки
- использование контекста
- утечки
- простые алгоритмы

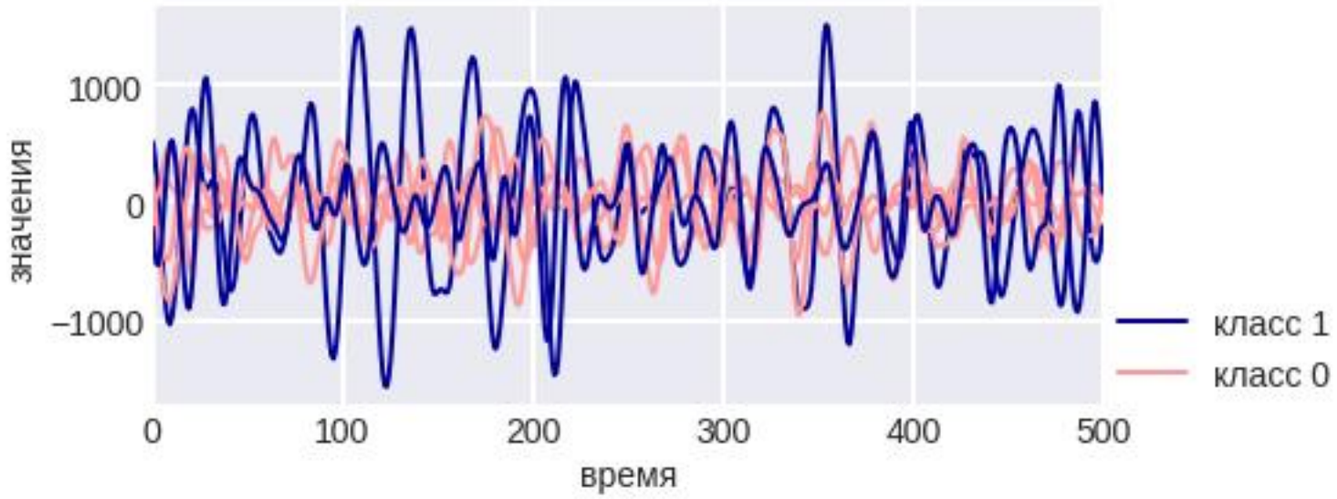
«Ford Classification Challenge» (2008)

Диагностика двигателя по сигналам датчиков

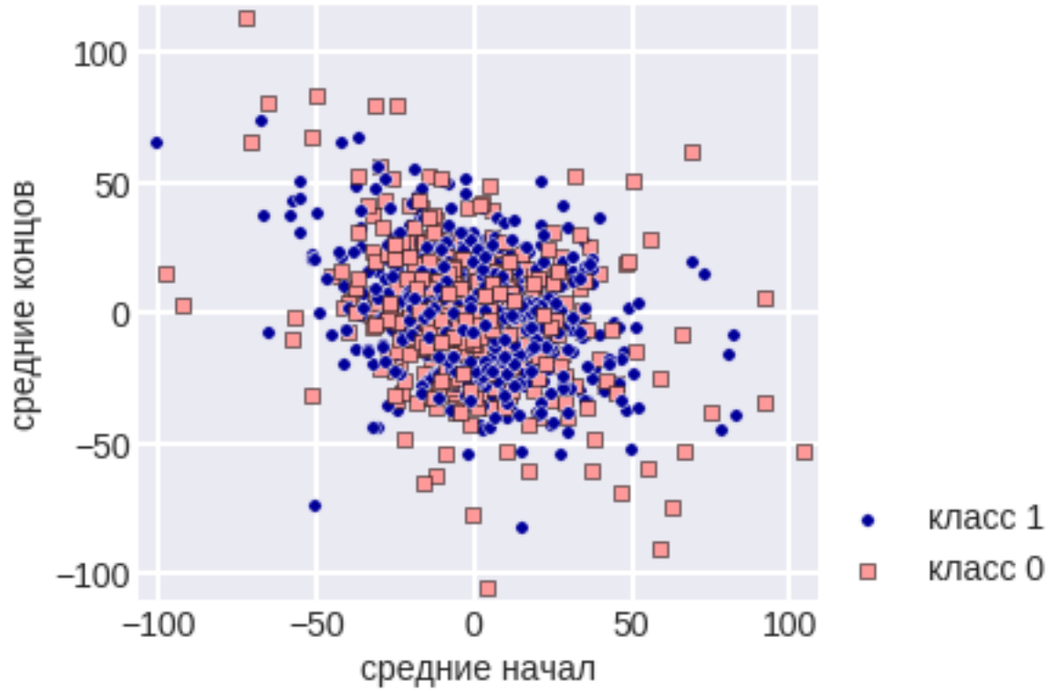
Задача классификации сигналов

Размеры данных: 3271×500

2 класса

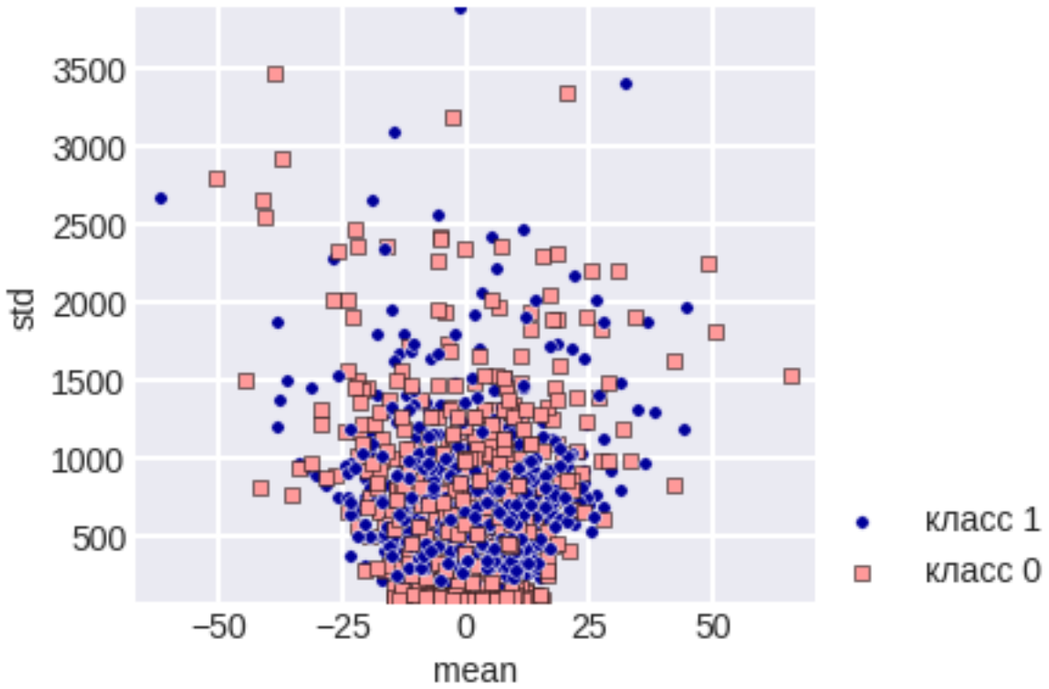


Особенности данных

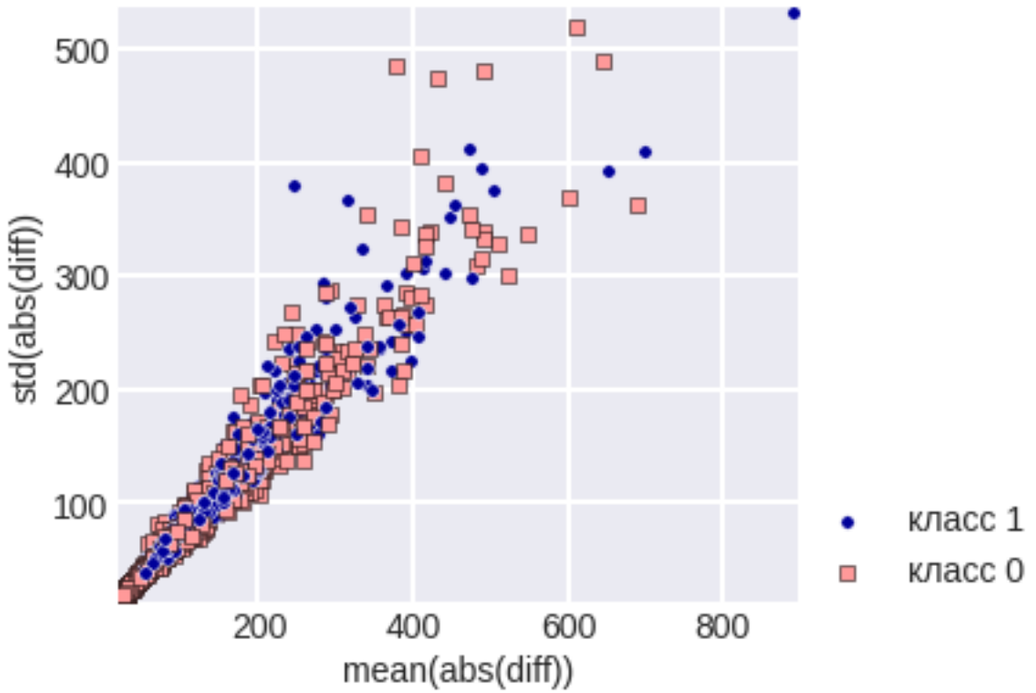


- **Неоднородность**
  (средние значения в начале сигнала не коррелируют по средними в конце)
- **Непериодичность**
- **Отсутствие заметных паттернов**

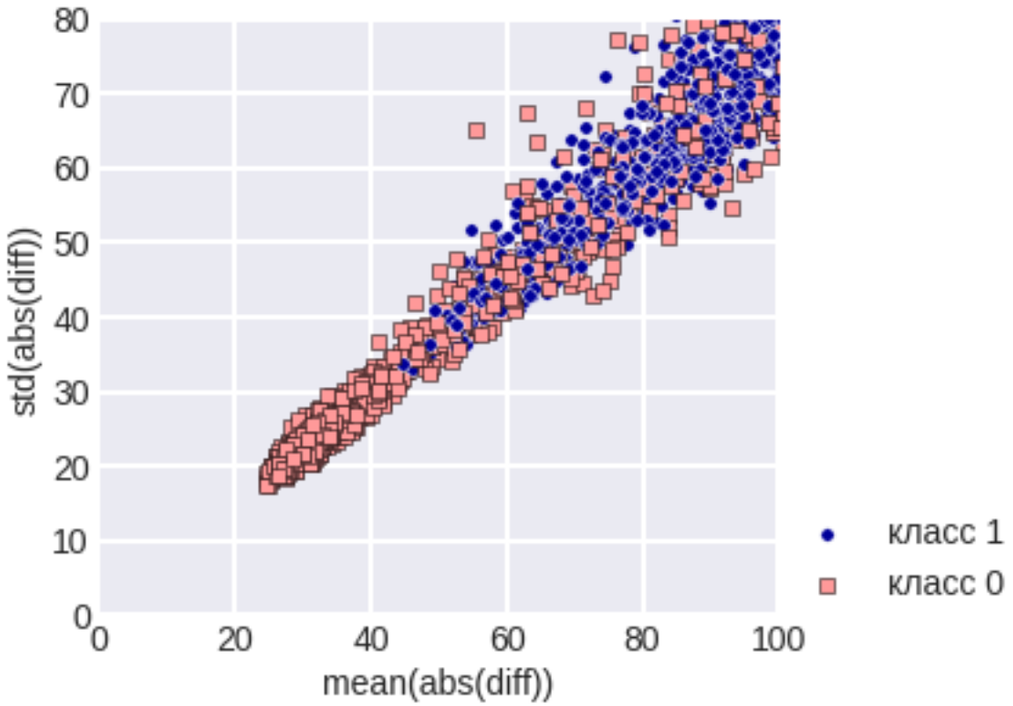
Пытаемся найти хороший признак



Пытаемся найти хороший признак



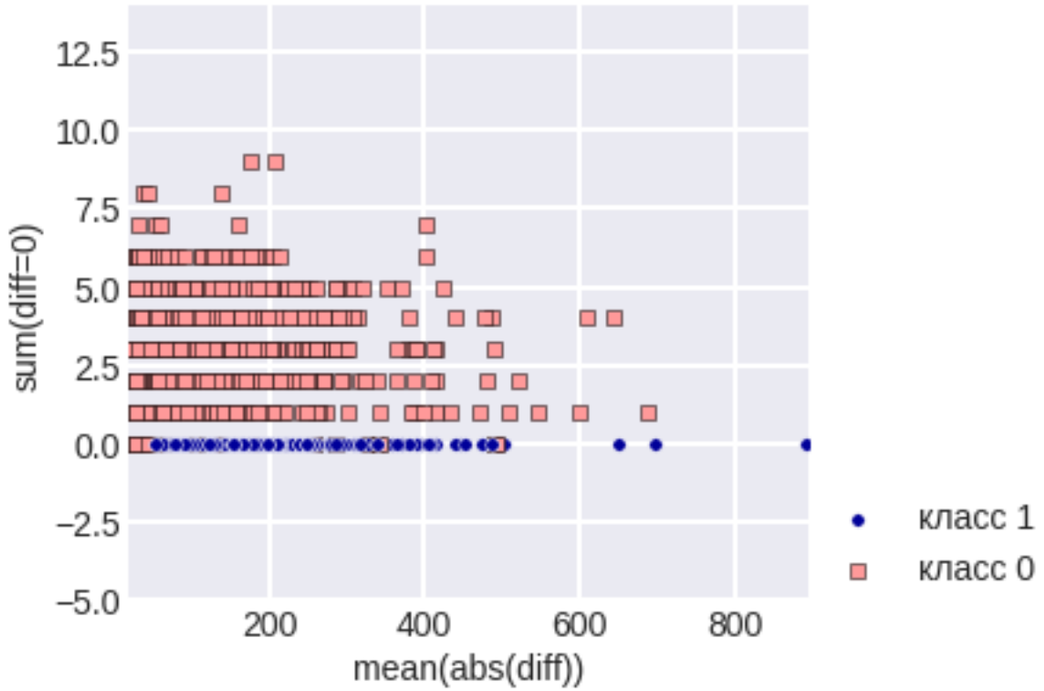
Пытаемся найти хороший признак



Увеличили изображение…

класс 0 – маленькие разности

Пытаемся найти хороший признак



Почти идеальный признак!

$$\frac{1}{n-1}\sum_{i=1}^{n-1} I[u_{i+1}-u_i=0] = \frac{1}{n-1}\sum_{i=1}^{n-1} I[u_{i+1}=u_i]$$

Считаем, сколько раз значения в сигнале повторяются…

Попробуем обобщить этот признак!

Пытаемся найти хороший признак

$$\frac{1}{n-1}\sum_{i=1}^{n-1} I[u_{i+1}-u_i=0] = \frac{1}{n-1}\sum_{i=1}^{n-1} I[u_{i+1}=u_i]$$

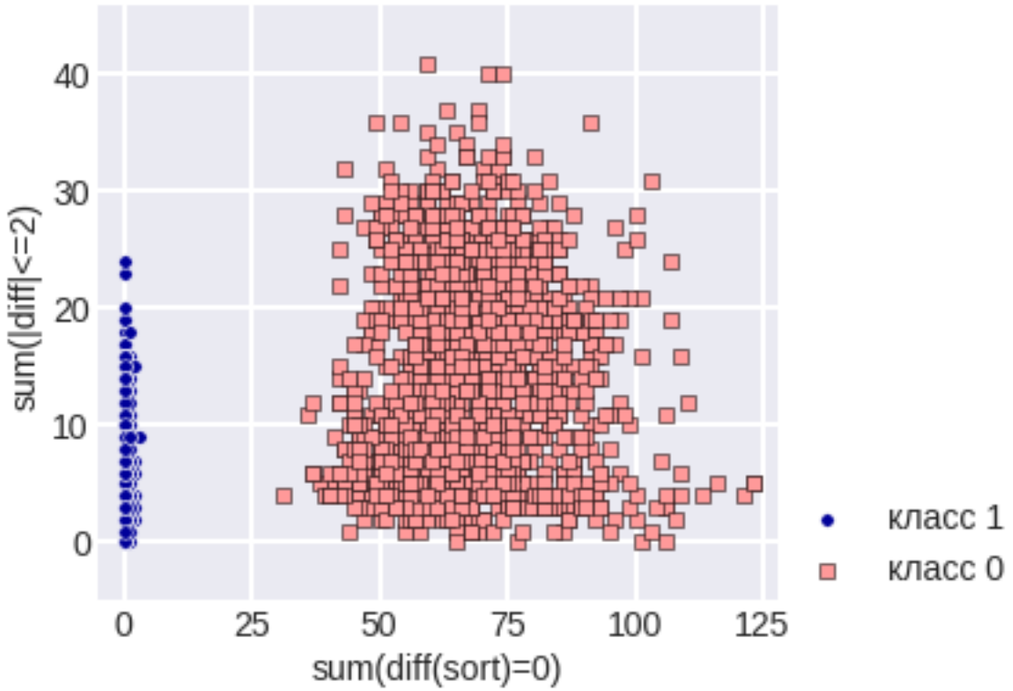
обобщение №1 – сколько раз значения СЛАБО ОТЛИЧАЮТСЯ

$$\frac{1}{n-1}\sum_{i=1}^{n-1} I[|u_{i+1}-u_i|<\varepsilon]$$

обощение №2 – сколько НЕуникальных значений сигнала

$$\frac{1}{n-1}\sum_{i=1}^{n-1} I[u_{i+1}^{\text{sorted}} = u_{i+1}^{\text{sorted}}]$$

Пытаемся найти хороший признак



Получили идеальный признак!

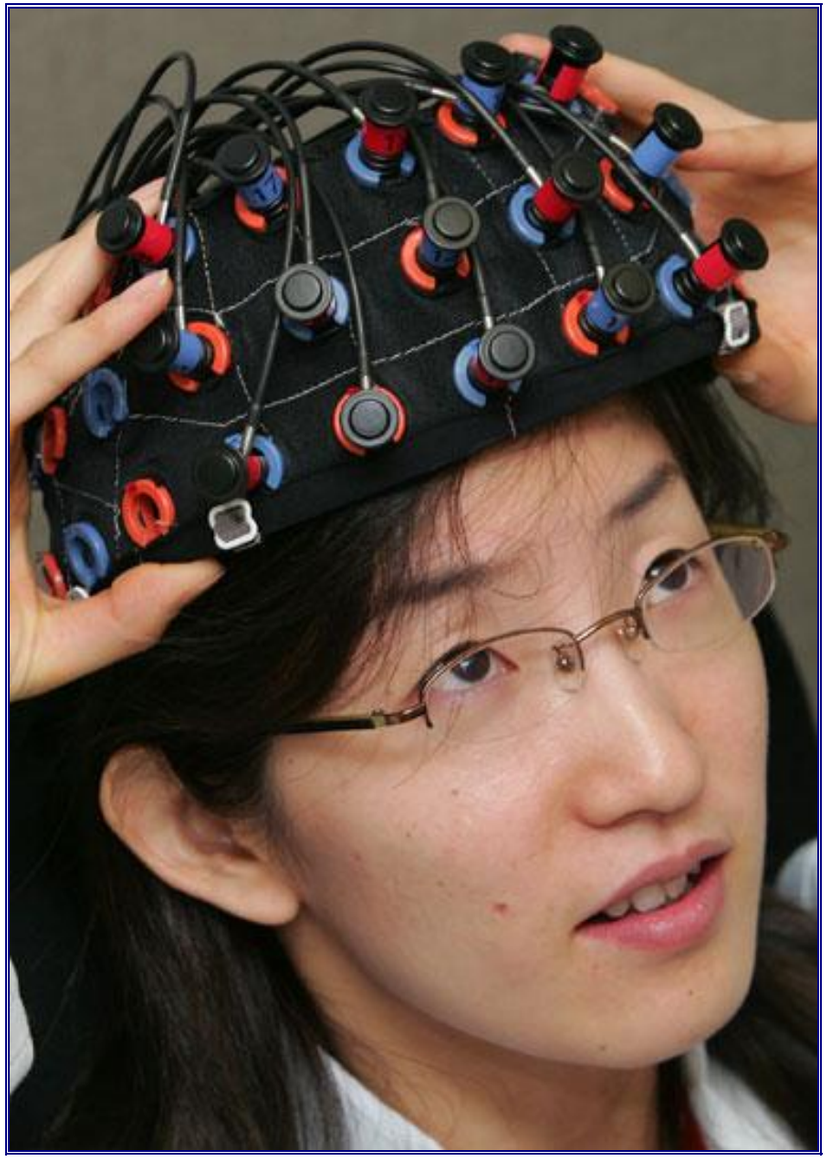
Решение задачи

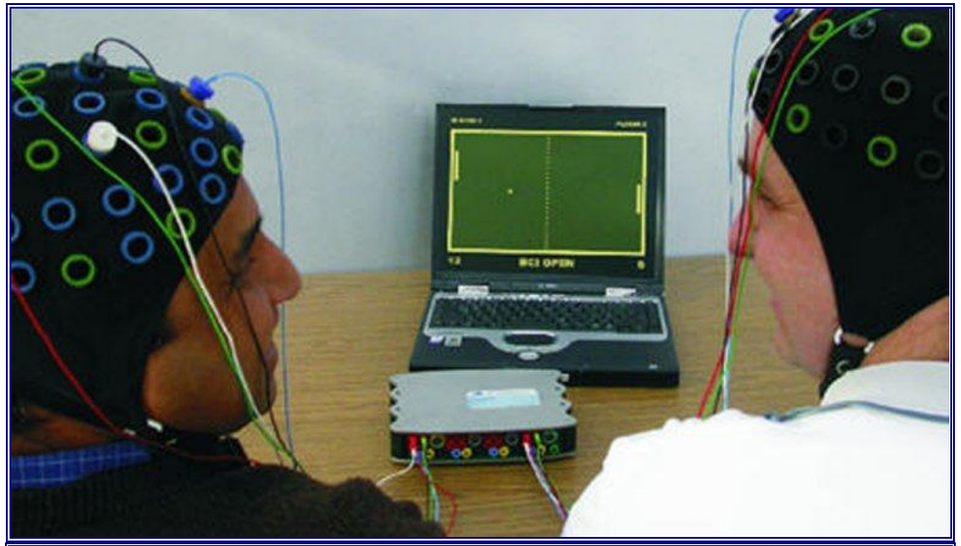
```
2 * (np.sum(np.diff(np.sort(train.values, axis=1), axis=1) == 0, axis=1) < 20) - 1
```

34 символа

```
2*((sum(diff(sort(X'))==0)<20)'-1)
```

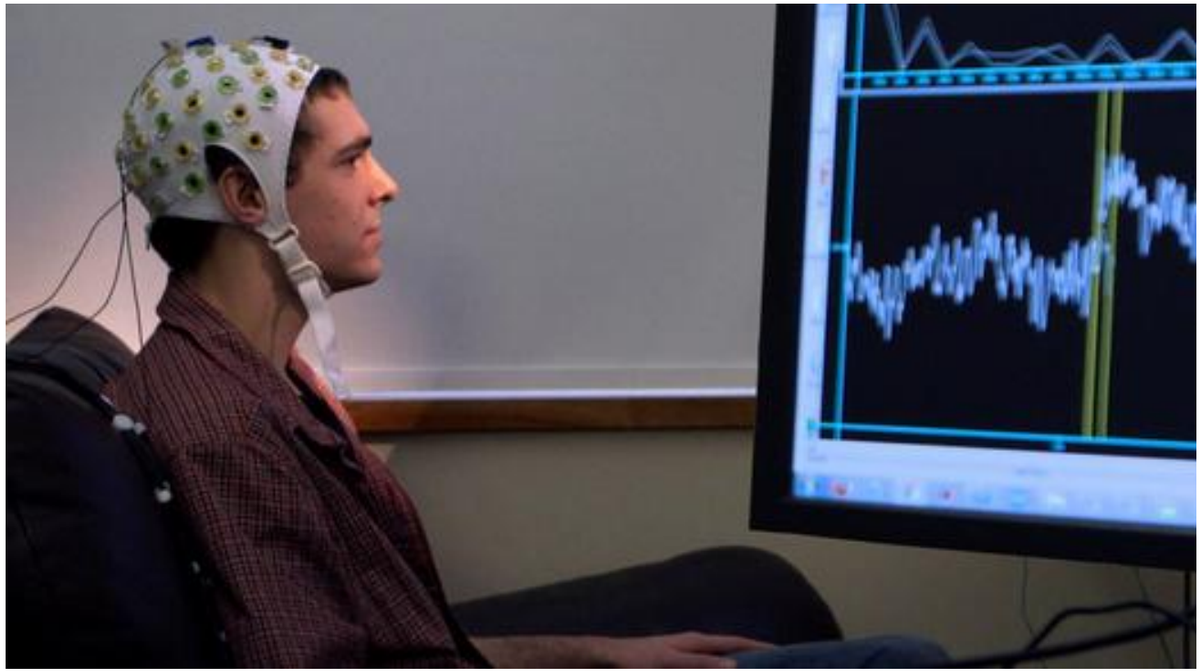
Brain Computer Interface





Brain Computer Interface

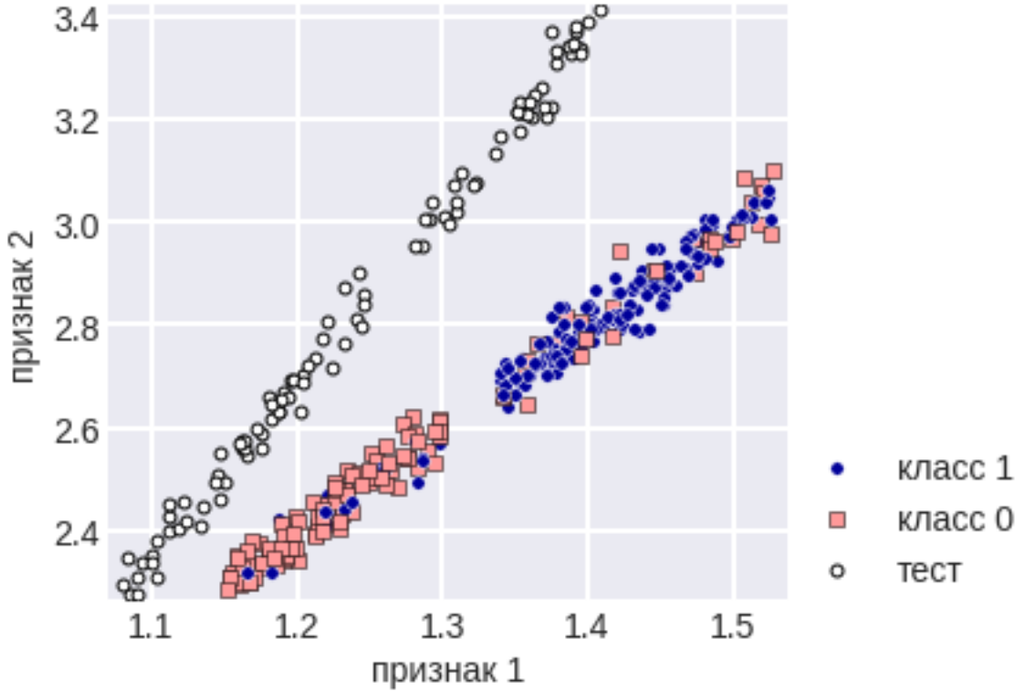
http://www.bbci.de/competition/iii/



278 сигналов × 64 электрода × 3000 замеров
(3 секунды с частотой 1000Гц)

- **сигналы только с одного электрода**
- **хороши признаки ~ скорость изменения сигнала**

Особенность – нестабильность признаков



два похожих признака

$$\frac{1}{n-1}\sum_{i=1}^{n-1}|u_{i+1}-u_i|$$

$$\frac{1}{n-1}\sum_{i=1}^{n-1}(u_{i+1}-u_i)^2$$

тоже 2 облака с зазором

Особенность – нестабильность признаков

в признаковых пространствах тест «смещается»

но можно понять КАК

Итоговое решение:

- **сглаживание сигнала**
- **усреднение скачков по окрестности**

решение

$$\frac{1}{n-k}\sum_{i=1}^{n-k}(\max[u_{i:i+k}]-\min[u_{i:i+k}])\geq c$$

$$u_{i:i+k}=\{u_i,\ldots,u_{i+k}\}$$

Задача определения реакции пользователя на рассылку

https://www.crowdanalytix.com/

клиент			услуга			статистика		
пол	id	регион	цена	скидка	категория	сколько предложений	сколько успешных	y
M	113	12	1100	0.2	17	5	2	0
Ж	078	13	1100	0.2	17	5	1	0
M	112	09	1200	0.0	16	7	0	0
M	111	07	1200	0.0	16	9	2	1

Изучаем признаки

идентифицируем клиента (регион, id)
сортируем по числу предложений

пол	id	регион	цена	скидка	категория	сколько предложений	сколько успешных	y
	113	09				0	0	0
	113	09				1	0	0
	113	09				2	0	1
	113	09				3	1	1
	113	09				4	2	0
	113	09				5	2	?

Для одного клиента – все значения целевого признака (кроме последнего) определяются со 100% точностью!

Изучаем признаки

80% меток известно

55 участников

Замечена утечка – 3 на первой стадии
Замечена утечка – 5 всего

Олимпиада ВикиМарта

http://olymp.wikimart.ru/ (сейчас недоступен)

Покинет ли человек сессию!

	id		сколько страниц в сессии	Номер текущей страницы					y
	76		3	1					0
	76		3	2					0
	76		3	3					1
	77		4	1					0
	77		4	2					0
	77		4	3					0
	77		4	4					1

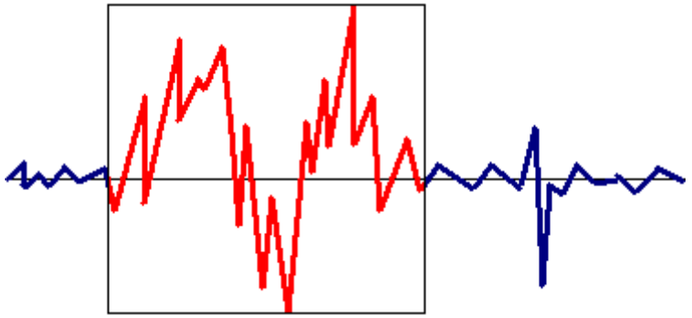
качество около 0.99 AUROC

Поиск хороших признаков в виде:

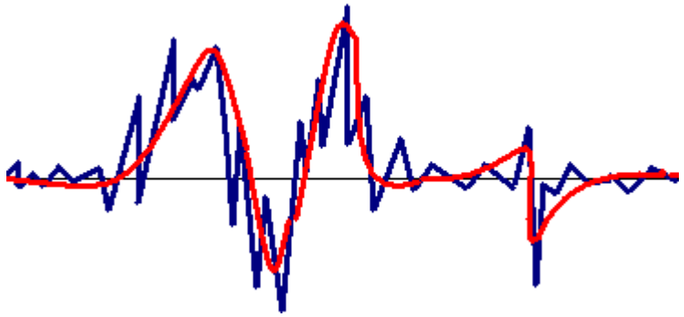
$$f = B(A_k(\ldots A_2(A_1(s))))$$

Операторы первого типа

Подсигнал

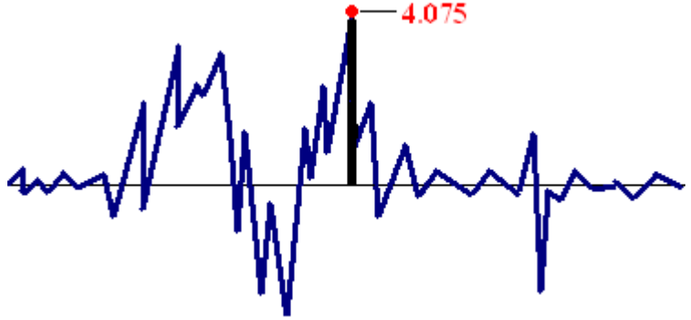


Сглаживание

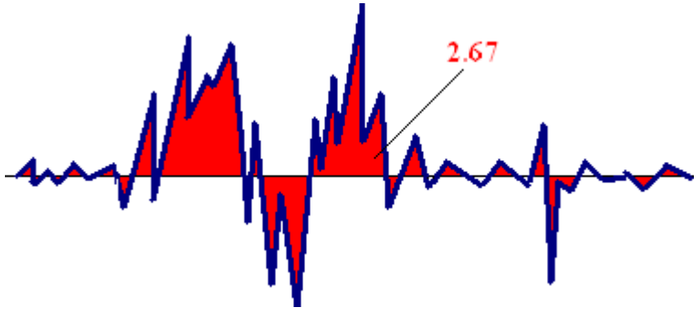


Операторы второго типа

Максимум



Интеграл



Задача

Дано: m, n, k, характеристики ЭВМ

Целевой признак: Время перемножения матриц размеров m×k и k×n

В контроле: ~5000 Записей, ~950 признаков

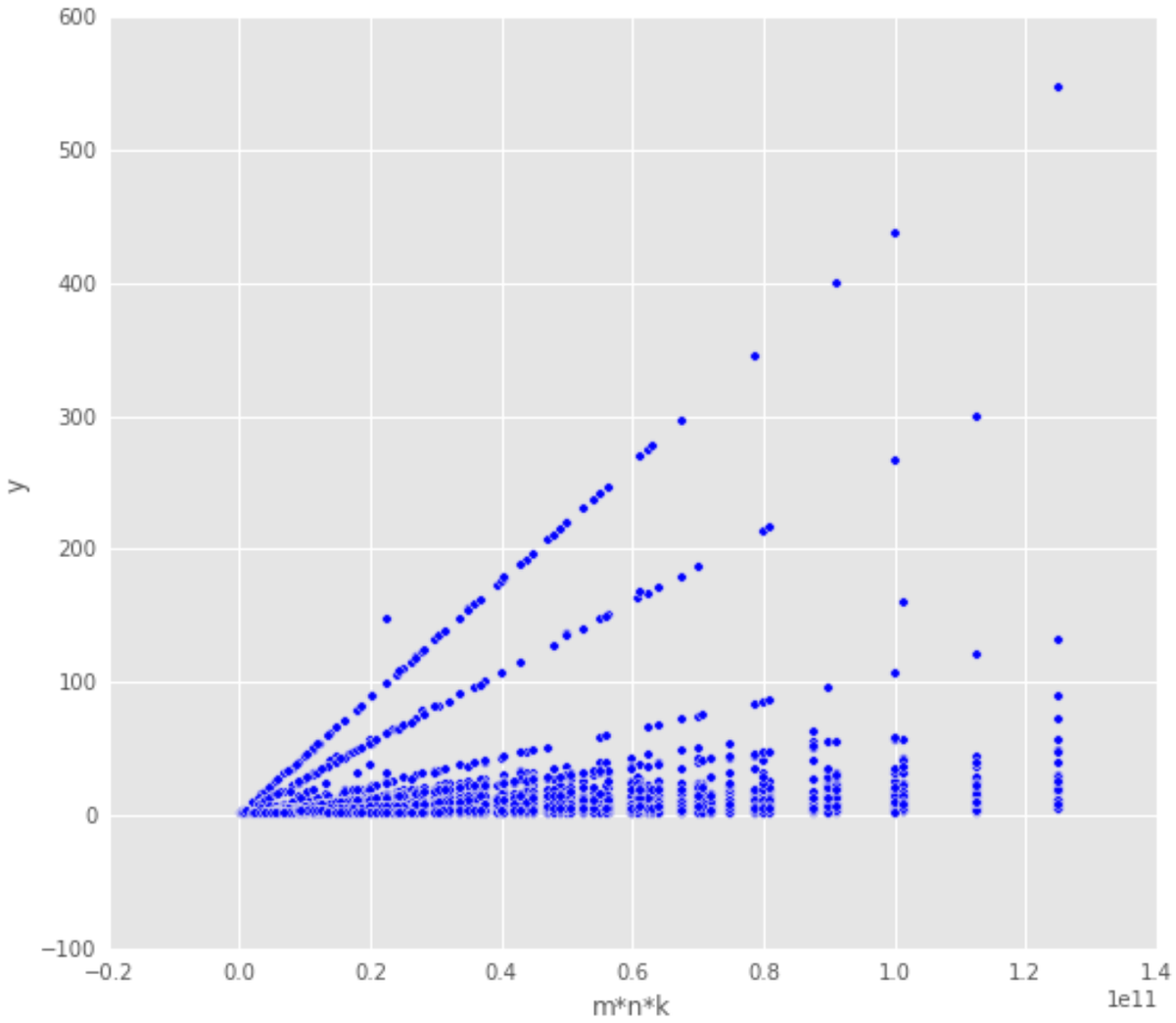
В тесте: =*=

Прагматика: прогнозирование времени вычислений

Простая логика: результат произведения – mn элементов, вычисление каждого – k умножений, общее число умножений = m×n×k

Функционал качества: MAPE

Загружаем данные – смотрим



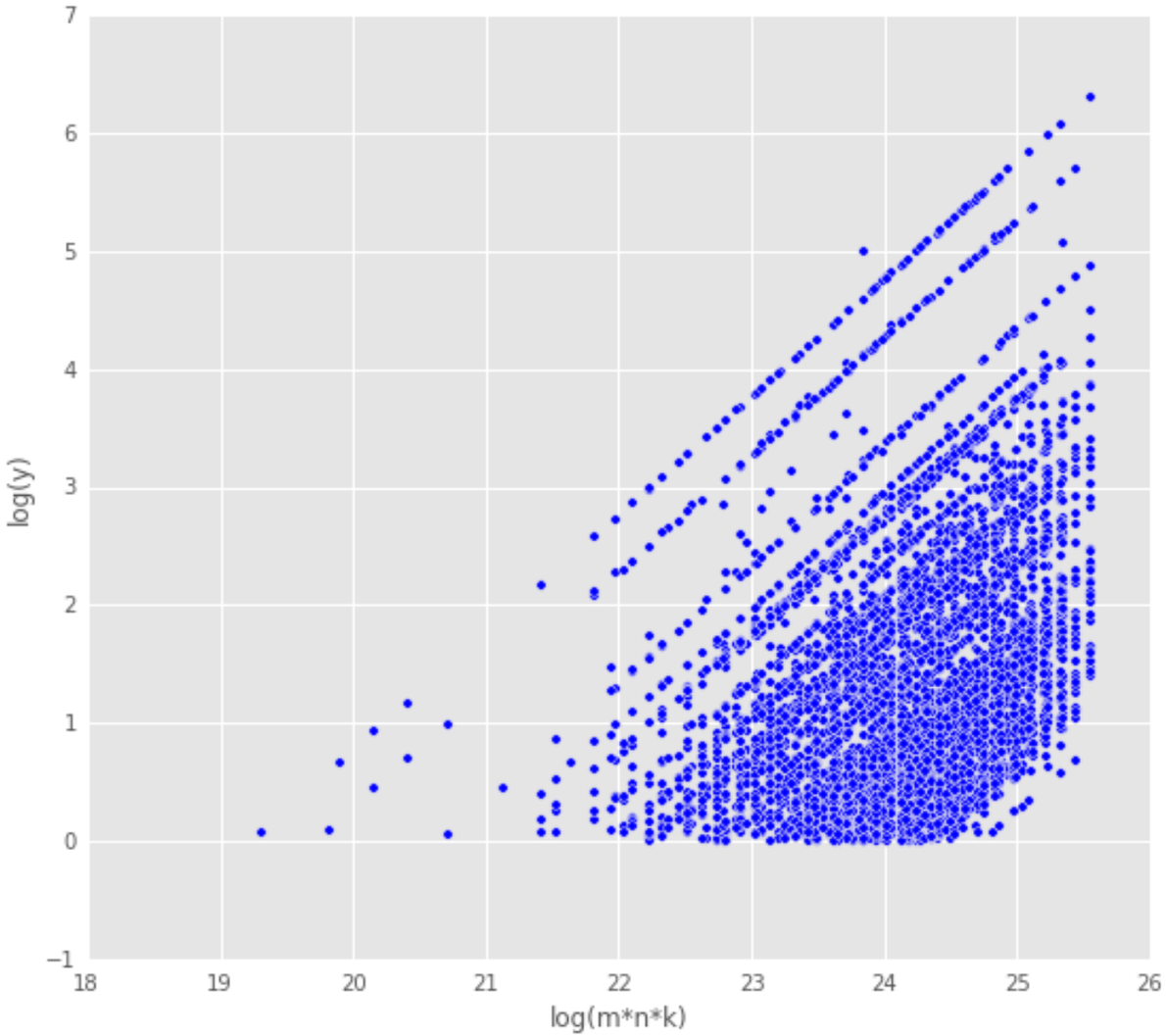

```
test = pd.read_csv('x_test.csv') #
train = pd.read_csv('x_train.csv') #
y = pd.read_csv('y_train.csv') #

y2 = y.values[:,0]


# основной признак
train['mnk'] = train.m * train.n * train.k

f = train.mnk.values

plt.scatter(f,y2)
```

Основная гипотеза: время перемножения ~ mnk

Загружаем данные – смотрим

	m	k	n	cacheL1IsShared	cacheL1Size	cacheL2IsShared	cacheL2Size	memType	memtRFC	memtCL	memtRCD	memtRP	memtRAS	os	cpuFull	cpuArch
0	2500	3500	5000	0	32	0	256	DDR3-SDRAM PC3-12800	1023	31	31	15	63	Windows 7 Professional Professional Media Cent...	Intel(R) Core (TM) i7-3820 CPU @ 3.60GHz	Sandy Bridge-E
1	2500	4000	4500	0	32	0	256	DDR3-SDRAM PC3-12800	1023	31	31	15	63	Windows 7 Professional Professional Media Cent...	Intel(R) Core (TM) i7-3820 CPU @ 3.60GHz	Sandy Bridge-E
2	2500	4000	5000	0	32	0	256	DDR3-SDRAM PC3-12800	1023	31	31	15	63	Windows 7 Professional Professional Media Cent...	Intel(R) Core (TM) i7-3820 CPU @ 3.60GHz	Sandy Bridge-E

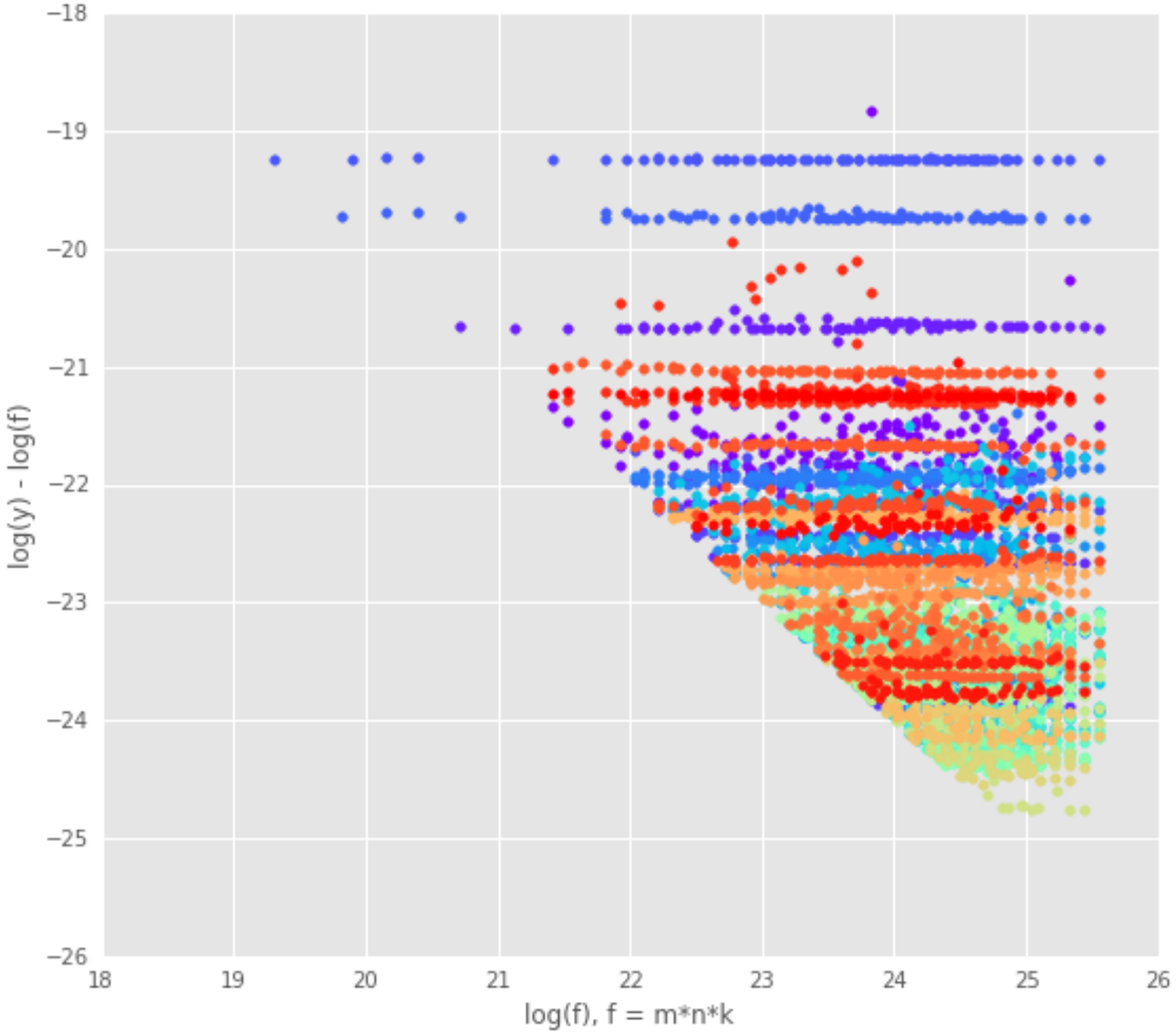
Логарифмируем основной и целевой признаки




```
plt.scatter(np.log(f),np.lo
g(y2))
plt.xlabel('log(m*n*k)')
plt.ylabel('log(y)')
```

Теперь чётко видны линии… что на них?!

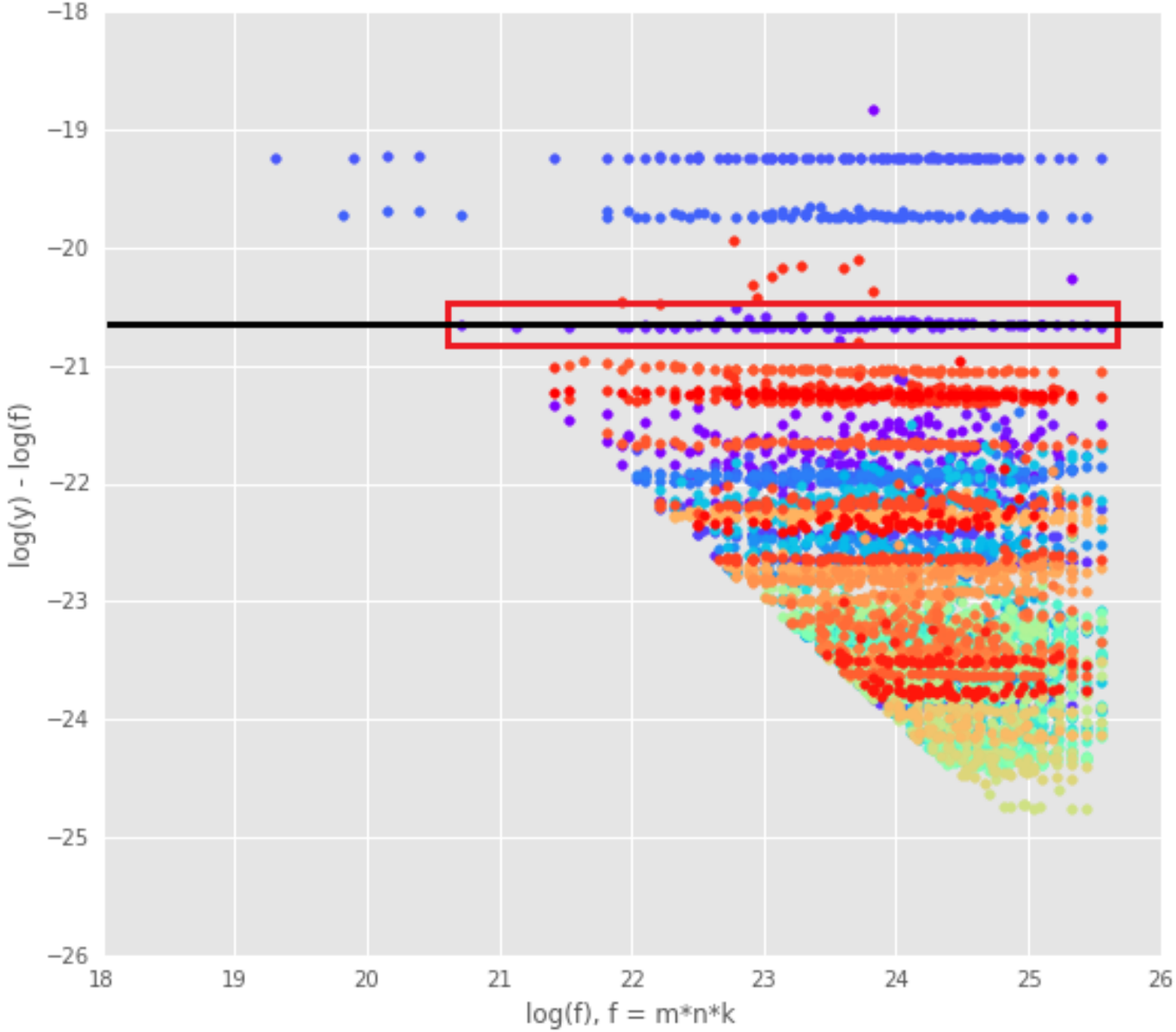
Гипотеза: разные процессоры




```
from sklearn.preprocessing import LabelEncoder
import matplotlib.cm as cm

tmp = train.cpuFull # строковый признак
encoder = LabelEncoder()
encoder.fit(tmp)
# нумеруем признак
tmpcodes = encoder.transform(tmp)

ncolors = np.max(tmpcodes) + 1 # число разных
значений

colors = cm.rainbow(np.linspace(0, 1,
ncolors))

# разным цветом - разные номера
for t, c in enumerate(colors):
        plt.scatter(np.log(f[tmpcodes==t]),
           np.log(y2[tmpcodes==t]) -
        np.log(f[tmpcodes==t]), color=c)
```

Вроде, так оно и есть…

Идея бенчмарка



Для каждого процессора оценить средний уровень
– его и предсказывать!

Такое тестирование нечестное (см. ниже).

Почему?

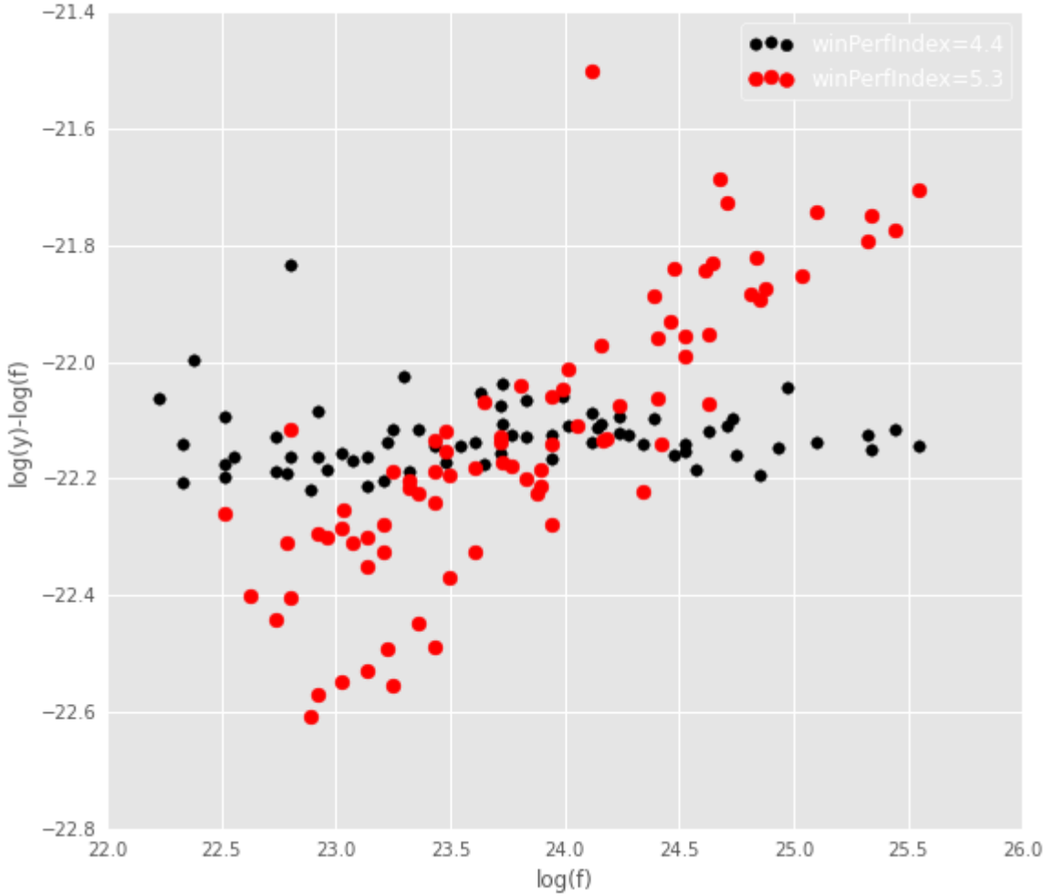
Честный бенчмарк

```
from sklearn.base import BaseEstimator, ClassifierMixin

# наш регрессор
class mybench(BaseEstimator, ClassifierMixin):
    def __init__(self):
        self.lst = []
        pass

    def fit(self, X, y):
        self.lst = []
        k = np.max(X.tmp.values) + 1
        for t in range(k):
            self.lst.append(np.median(np.log(y[X.tmp.values==t])-np.log(X.mnk.values[X.tmp.values==t])))
        return self

    def predict(self, X):
        a = np.zeros(X.shape[0])
        k = np.max(X.tmp.values) + 1
        for t in range(k):
            a[X.tmp.values==t] = np.exp(self.lst[t] + np.log(X.mnk.values[X.tmp.values==t]) )
        return a

    def coef(self):
        return (self.clf.coef_)
```

Честный бенчмарк

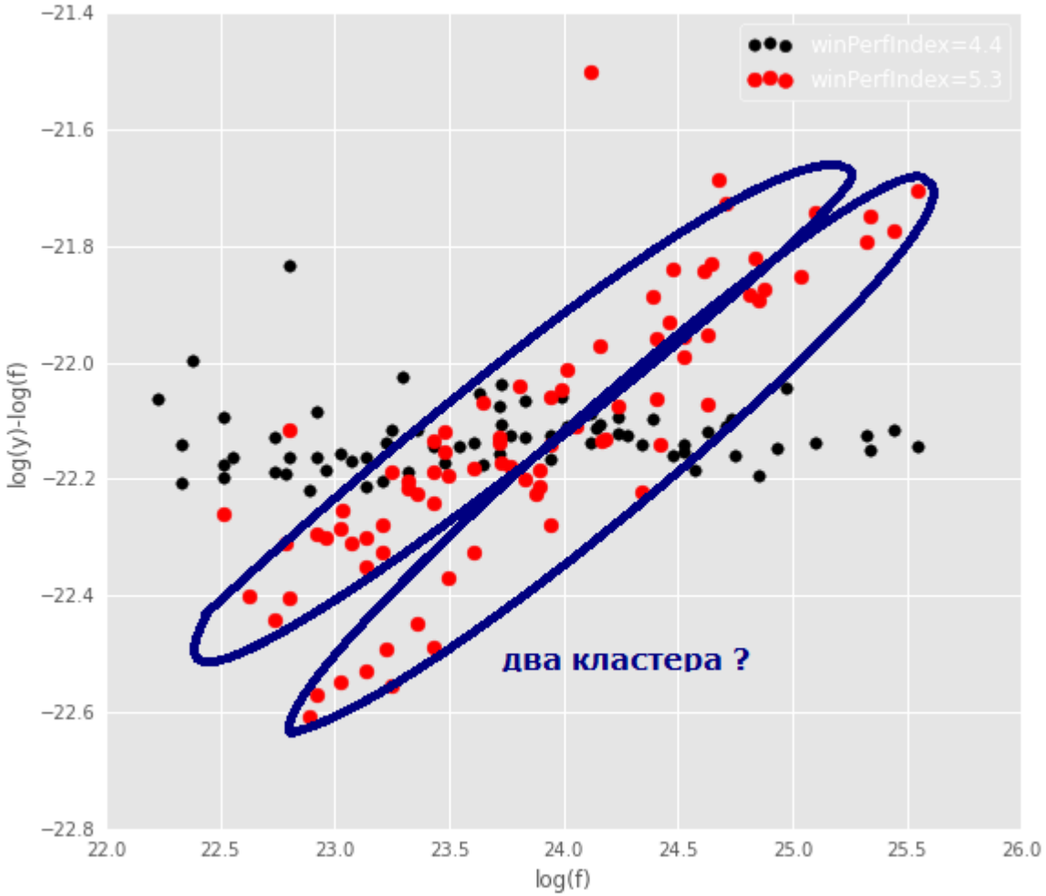
```
# тут в tmp-признак заносим номера...
train['tmp'] = train.cpuFull # строковый признак
encoder = LabelEncoder()
encoder.fit(train.tmp)
train['tmp'] = encoder.transform(train.tmp) # нумеруем признак

from sklearn.cross_validation import KFold
from sklearn.cross_validation import cross_val_score
import time

# пишем свой скорер
def my_accuracy_scoring(est, X, y):
    return np.mean((np.abs(est.predict(X) - y)/y))

# схема тестирования
cv = KFold(n=len(y), n_folds=20, shuffle=True, random_state=1)



# тестирование
tm = time.time()
model = mybench()
cvresult = cross_val_score(model, train, y2, scoring=my_accuracy_scoring, cv=cv)
print (cvresult, np.mean(cvresult))
print (time.time() - tm)
```

Продолжение идеи...



разные winPerfIndex (индекс производительности) на одинаковом процессоре

Ещё продолжение идеи…



Вроде, больше групп точек,

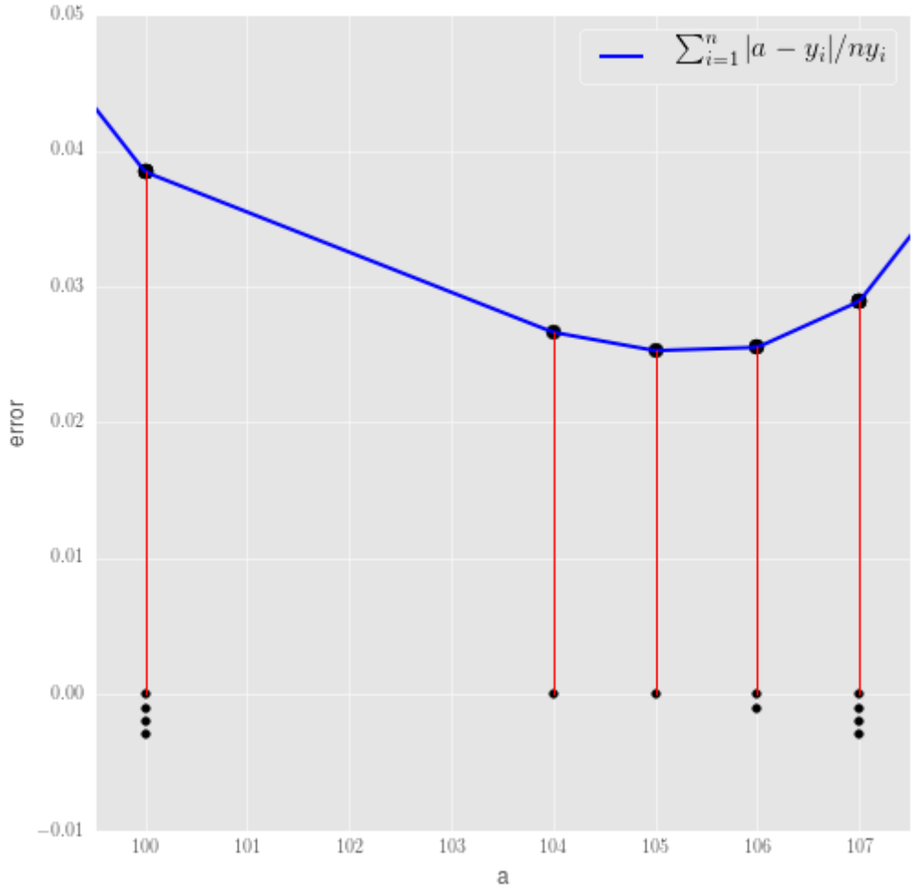
но эти две группы не отличаются по другим признакам…

Тестирование

Усредняем по …	mean	median	mymean минимизируем напрямую
по процессору	0.089	0.076	0.073
по процессору + winPerfIndex	0.075	0.063	0.061
по процессору + os	0.068	0.061	0.059
по процессору + os + RAM			0.059 0,055 LB (+trunc)

самописная регрессия = 0.055, 0.0525 LB (+trunc)

Разберёмся с функционалом



Некоторое новое среднее (больше похоже на медиану)

[100, 100, 100, 100, 104, 105, 106, 106, 107, 107, 107, 107]

Как действовать на практике

Итак, если надо

$$f(a) = \frac{1}{n}\sum_{i=1}^{n}\frac{|a - y_i|}{y_i} \to \min_a$$

Можно просто посмотреть значения

$f(y_1),\ldots,f(y_n)$ **и выбрать наименьшее**

Как действовать на практике

Если приближаем не константой, а линейной функцией

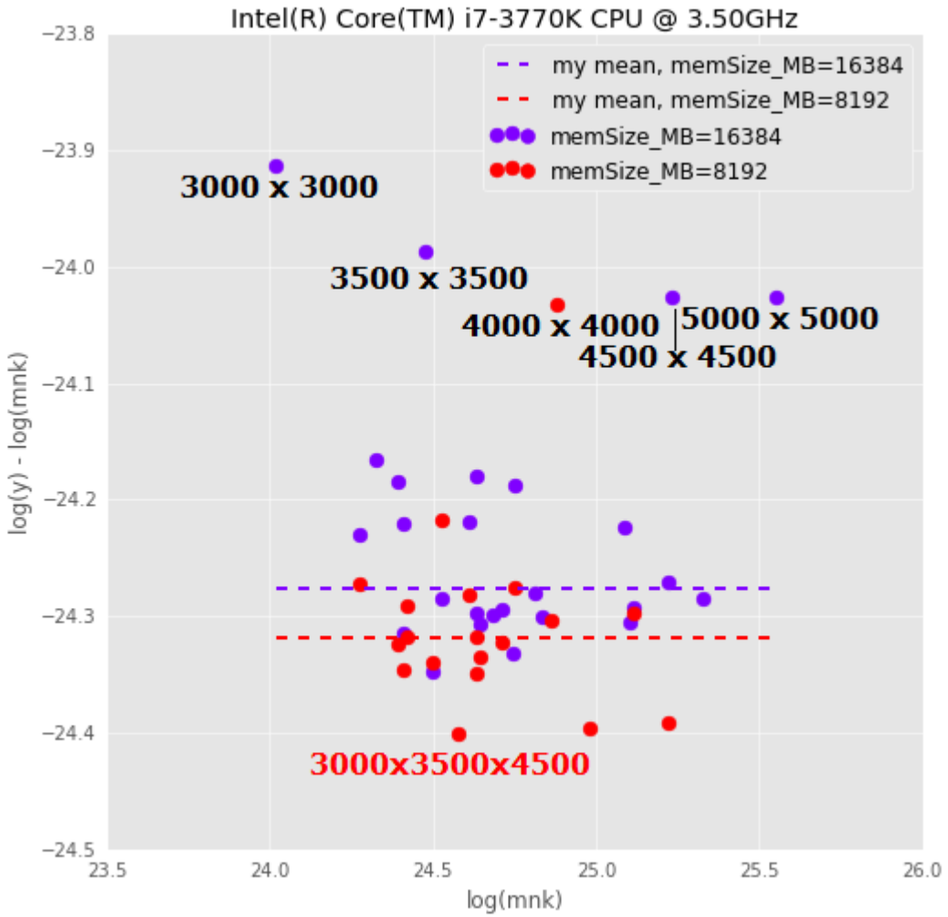
(см. дальше картинки)

$$f(a,b) = \frac{1}{n}\sum_{i=1}^{n}\frac{|\,ax_i + b - y_i\,|}{y_i} \to \min_{a,b}$$

Можно устроить интеллектуальный перебор (a, b) **по сетке от константного решения** $(0, b_{\text{opt}})$ **до решения обычной регрессией** $(a_{\text{ridge}}, b_{\text{ridge}})$

Что интересно

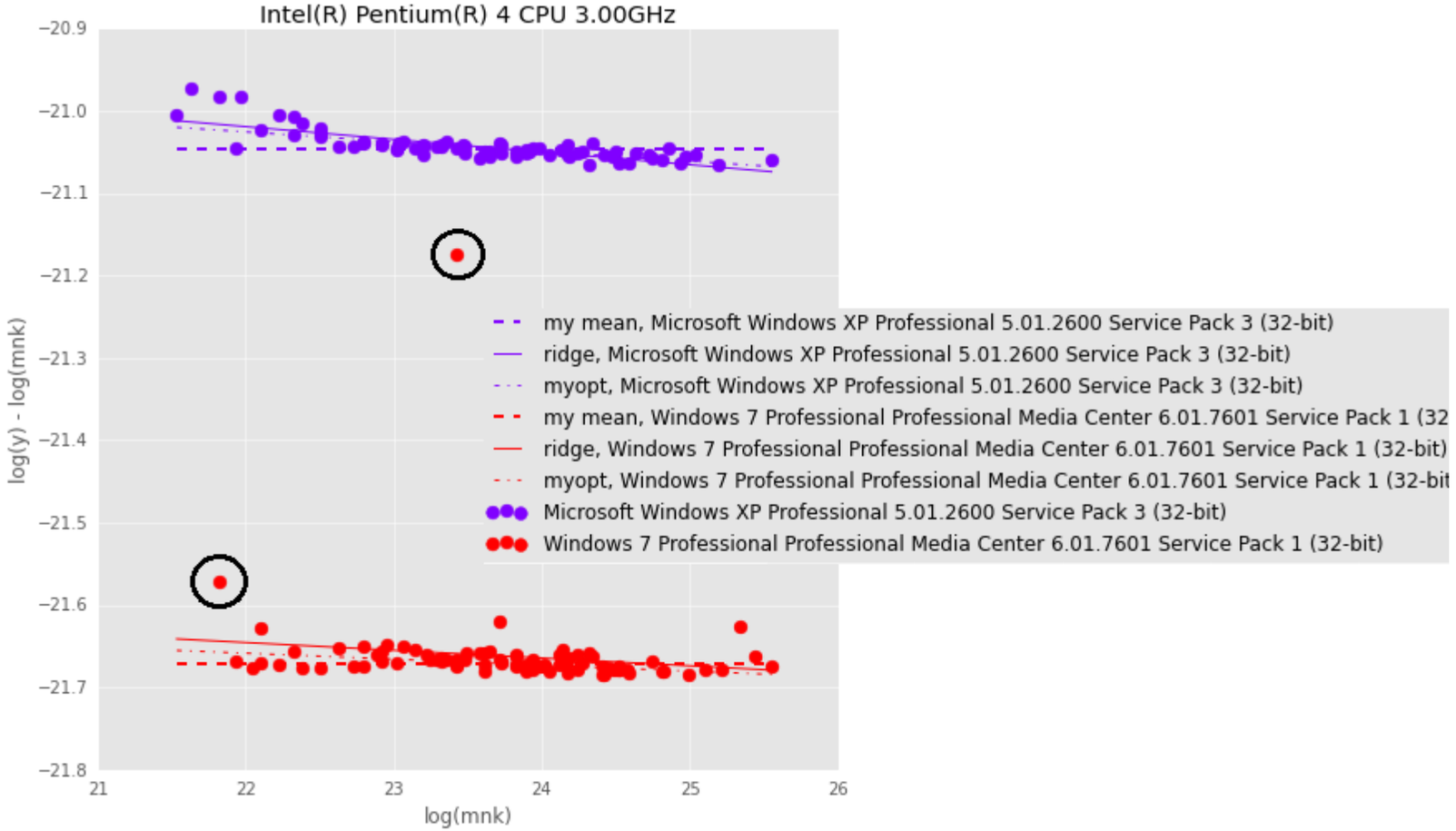
n	900	1000	1500	2000	2500	3000	3500	4000	4500	5000
m										
300	1									
500	1									
700	3									
900	2									
1000		2	6	16	29	62	70	125	155	222
1500			7	23	38	81	108	154	217	273
2000				20	35	84	113	194	211	294
2500					35	76	104	163	215	270
3000						44	83	120	176	243
3500							50	90	137	178
4000								43	86	149
4500									53	91
5000										41

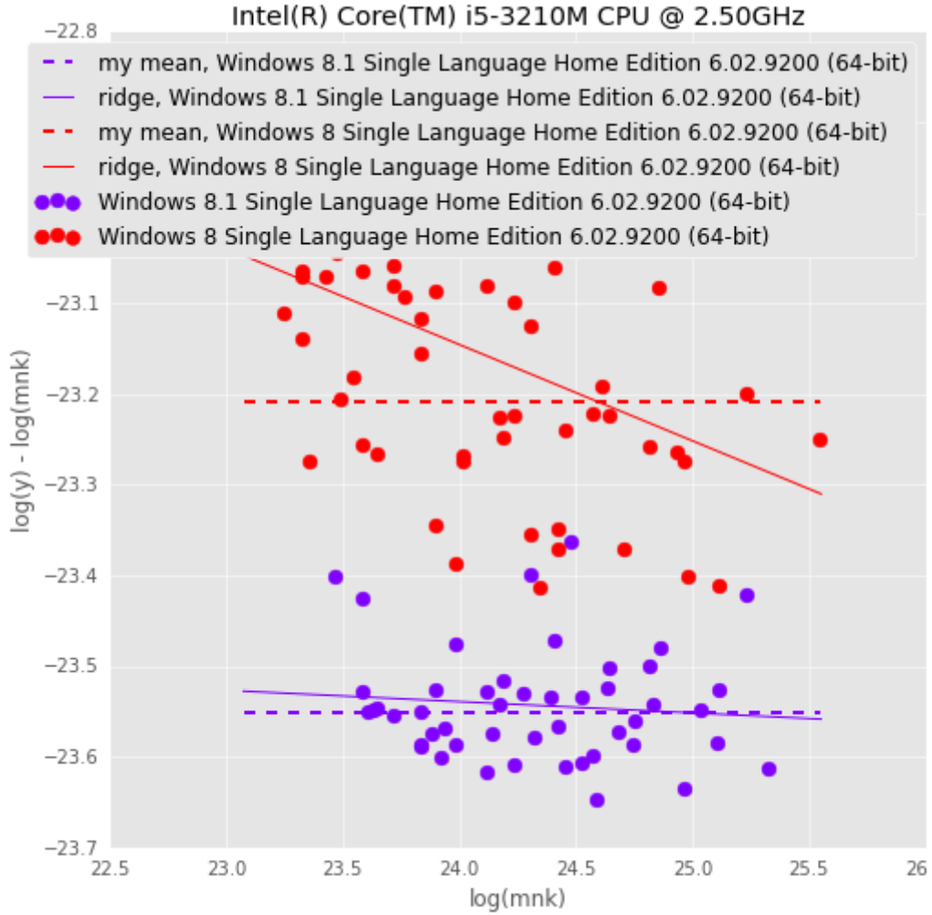


как часто встречаются комбинации размеров

время максимально, когда матрицы квадратные!!!

Что интересно





выделенные объекты ничем не отличаются (кроме размеров перемножаемых матриц)

насколько разные ОС: win8 и win 8.1

Что означает наш регрессионный метод

$$\log(y) \approx w_1 \log(mnk) + w_0$$

Ответ алгоритма:

$$a = e^{w_0} \cdot (mnk)^{w_1}$$

(деформированное произведение размеров)

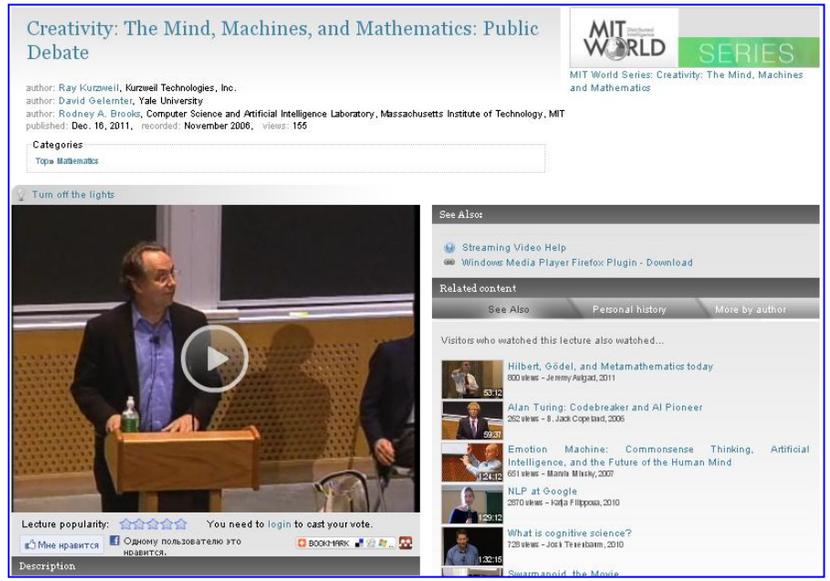
Если решаем не регрессией, а константой, то какое решение?

Разработка рекомендательной системы

Международное соревнование
«VideoLectures.Net Recommender System Challenge (ECML/PKDD Discovery Challenge 2011)»

http://tunedit.org/challenge/VLNetChallenge?m=summary

Опишем лучший алгоритм из 62

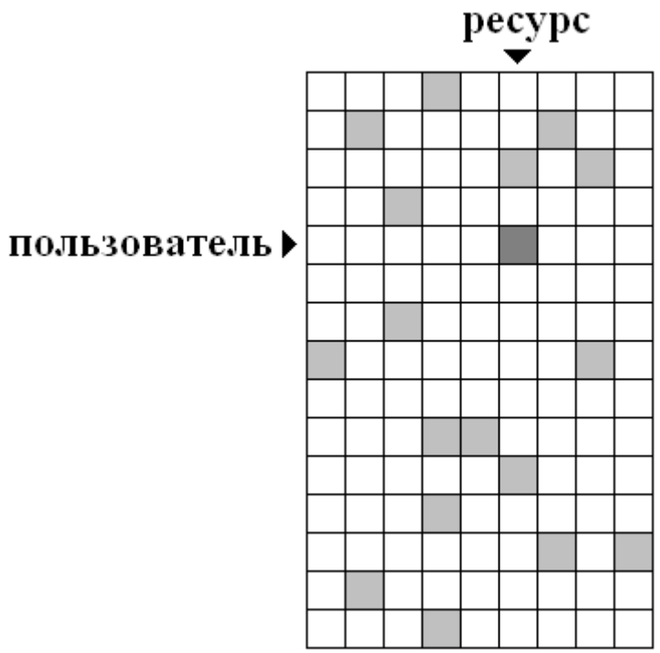


Дано: статистика популярности (+ описания лекций)

Надо: дать рекомендацию пользователю

– предложить лекции для просмотра

Обычно: матрица «пользователи – ресурсы»



Методы коллаборативной фильтрации

~ похожие пользователи – похожие ресурсы

Новое направление в анализе данных –

правильное обезличивание и усреднение

Pooled sequences

Формирование пост-троечных последовательностей

$$102 \to 33 \to 2 \to 34 \to 35 \to 2 \to 102 \to 17 \to 36,$$

удаляем из неё повторы:

$$102 \to 33 \to 2 \to 34 \to 35 \to 17 \to 36$$

после тройки $\{2,33,35\}$ **смотрел** $\{17,36\}$

7×{2,33,35}: 2×9, 5×13, 3×17, 1×30, 1×36

```
102 → 33 → 2 → 34 → 35 → 17 → 36
35 → 33 → 100 → 2 → 9 → 13 → 17
2 → 7 → 103
2 → 35 → 33 → 13 → 9 → 17
2 → 100 → 35 → 33 → 13 → 30
100 → 2 → 35 → 7 → 33
35 → 10 → 33 → 13
33 → 107 → 2 → 35 → 13
98 → 2 → 99 → 35 → 33 → 13
```

Дано: некоторые пост-троечные последовательности (109044 шт.)

Найти: другие пост-троечные последовательности

(точнее: 10 первых членов в нужном порядке)

```
7x {2, 33, 35}:   5×13, 3×17, 2×9,  1×30, 1×36
5x {2, 20, 21}:   3×1,  2×13, 2×30, 2×33, 2×40
8x {33, 20, 35}:  4×9,  4×13, 4×30, 2×7,  2×8
2x {1, 3, 35}:    2×7,  1×8,  1×13
...

?x {3, 20, 8} ?, ?, ?, ?, ?, …
```

рекомендации!

Качество

$$\frac{1}{|Z|}\sum_{z\in Z}\frac{|\{r_1,\ldots,r_{\min(S,R,z)}\}\cap\{s_1,\ldots,s_{\min(S,R,z)}\}|}{\min(S,R,z)}$$

$r_1,\ldots,r_R$ – **рекомендации**

$s_1,\ldots,s_S$ – **правильные ответы**

$Z=\{5,10\}$

Обозначения

Пост-троечная последовательность – вектор

$$v(\{a,b,c\}) = (v_1(\{a,b,c\}),\ldots,v_L(\{a,b,c\})),$$

L **– число лекций,**

$v_j(\{a,b,c\})$ **– сколько раз была просмотрена** j**-я лекция после тройки**

Как решать?

Объединение и пересечение множеств

(отдельная лекция по Fuzzy Sets)

мультимножества и нечёткие множества

$$\{1,2,2,3\} \cup \{2,3,4,4\} = \{1,2,2,2,3,3,4,4\}$$

	(1,2,1,0, …)	сложение характеристических векторов
+	(0,1,2,2, …)	
=	(1,3,3,2, …)	

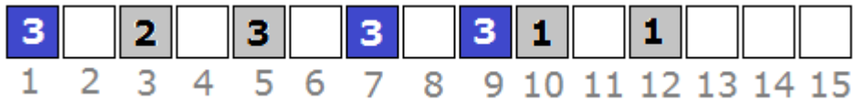
или

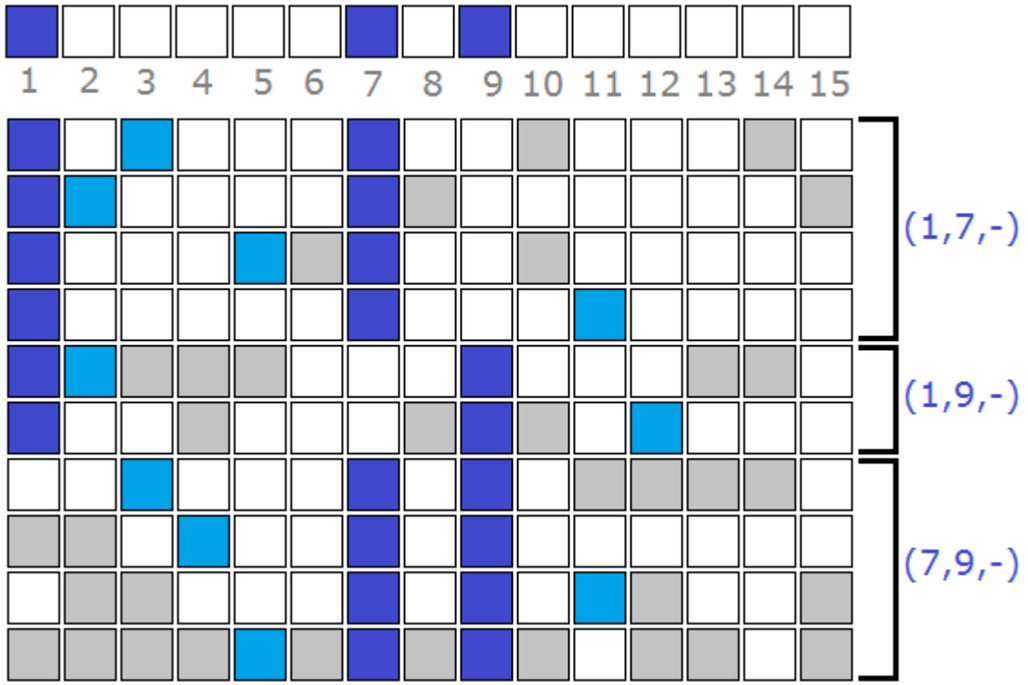
$$\{1,2,2,3\} \cup \{2,3,4,4\} = \{1,2,2,3,3,4,4\}$$

	(1,2,1,0, …)	если у элементов есть цвета, то можем гарантировать, что в объединение войдёт максимум элементов…
max	(0,1,2,2, …)	
=	(1,2,2,2,…)	

«Объединение информации»

3×{1,7,9}: 3×5, 2×3, 1×10, 1×12





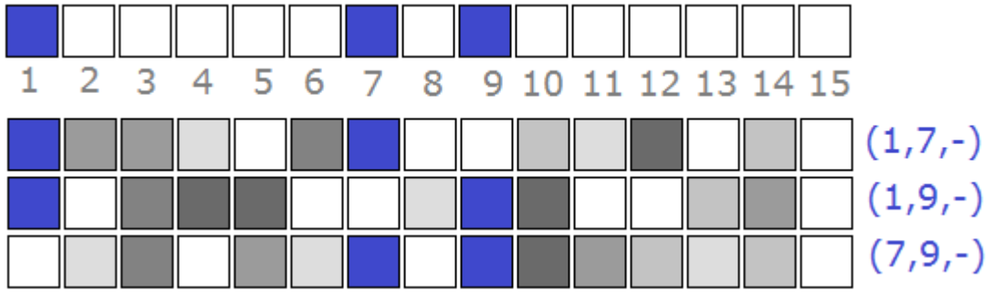
«Объединение информации»

Объединяем с помощью суммирования:

$$s(\{a,b\}) = \sum_d v(\{a,b,d\}),$$
$$s(\{a,c\}) = \sum_d v(\{a,c,d\}),$$
$$s(\{b,c\}) = \sum_d v(\{b,c,d\}).$$

Получили информацию по парам

«Пересечение информации»



$$s(\{a,b\}) \cdot s(\{b,c\}) \cdot s(\{a,c\})$$

$$(s(\{a,b\}) + \varepsilon) \cdot (s(\{b,c\}) + \varepsilon) \cdot (s(\{a,c\}) + \varepsilon)$$

но предварительно использовались нормировки…

Пример нормировки

Аналог IDF

$$v'(\{a,b,c\}) = \left(\frac{v_1(\{a,b,c\})}{\log(|\{\tilde{t} \in T \,|\, v_1(\tilde{t}) > 0\}| + 2)} \quad \cdots \quad \frac{v_L(\{a,b,c\})}{\log(|\{\tilde{t} \in T \,|\, v_L(\tilde{t}) > 0\}| + 2)} \right)$$

$|\{\tilde{t} \in T \,|\, v_j(\tilde{t}) > 0\}|$ – **число троек из обучения,**

в пост-троечные последовательности которых входит j**-я лекция.**

Как формировались итоговые оценки

$$\gamma = \log(s(\{a,b\}) \cdot s(\{b,c\}) \cdot s(\{a,c\}))$$

$$\gamma = \log(s(\{a,b\})) + \log(s(\{b,c\})) + \log(s(\{a,c\}))$$

$$(s(\{a,b\}) + \varepsilon) \cdot (s(\{b,c\}) + \varepsilon) \cdot (s(\{a,c\}) + \varepsilon)$$

не происходит зануления большинства элементов вектора (и потери информации)

$$\gamma = \frac{\log(s(\{a,b\}) + 0.02)}{\text{std}(\omega(s(\{a,b\}))) + 0.5} + \frac{\log(s(\{b,c\}) + 0.02)}{\text{std}(\omega(s(\{b,c\}))) + 0.5} + \frac{\log(s(\{a,c\}) + 0.02)}{\text{std}(\omega(s(\{a,c\}))) + 0.5}$$

ω ~ множество ненулевых элементов вектора

Как формировались итоговые оценки

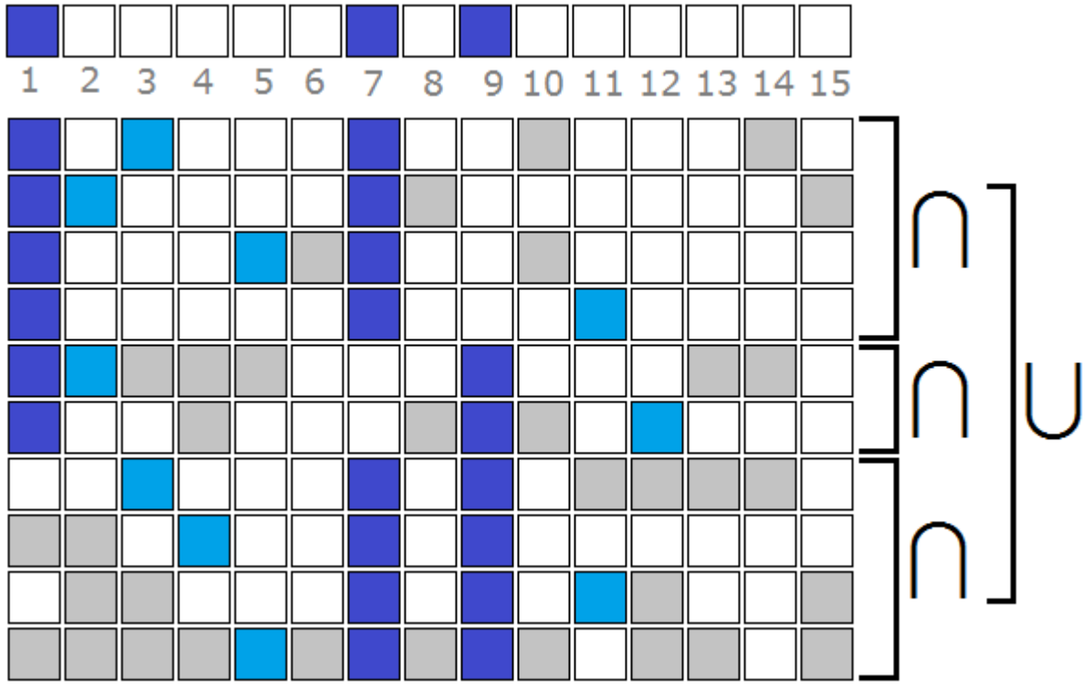
γ (вид выражения)	качество
$(s(\{a,b\})+\varepsilon)\cdot(s(\{b,c\})+\varepsilon)\cdot(s(\{a,c\})+\varepsilon),\ \varepsilon=0$	**57.27%**
$(s(\{a,b\})+\varepsilon)\cdot(s(\{b,c\})+\varepsilon)\cdot(s(\{a,c\})+\varepsilon),\ \varepsilon=0.01$	**62.11%**
$(s(\{a,b\})+\varepsilon)\cdot(s(\{b,c\})+\varepsilon)\cdot(s(\{a,c\})+\varepsilon),\ \varepsilon=0.1$	**61.60%**
$(s(\{a,b\})+\varepsilon)\cdot(s(\{b,c\})+\varepsilon)\cdot(s(\{a,c\})+\varepsilon),\ \varepsilon=1$	**58.84%**
$(s(\{a,b\})+s(\{b,c\})+\varepsilon)\cdot(s(\{b,c\})+s(\{a,c\})+\varepsilon)\cdot$ $\cdot(s(\{a,c\})+s(\{a,b\})+\varepsilon),\ \varepsilon=0$	**58.63%**
$(s(\{a,b\})+s(\{b,c\})+\varepsilon)\cdot(s(\{b,c\})+s(\{a,c\})+\varepsilon)\cdot$ $\cdot(s(\{a,c\})+s(\{a,b\})+\varepsilon),\ \varepsilon=0.001$	**59.87%**

РП второго места:

$$\gamma = s(\{a,b\})\log(s(\{a,b\}))+s(\{b,c\})\log(s(\{b,c\}))+s(\{a,c\})\log(s(\{a,c\}))$$

Идея алгоритма

А так… всё просто



Rank	Team		Preliminary Result	Final Result
1	+ D'yakonov Alexander		0.62102	0.62415
2	meridion		0.60791	0.61172
3	UniQ		0.58727	0.59063
4	+ Haibin Liu		0.47384	0.47507
5	+ barney		0.47060	0.47243
6	vyatka		0.45149	0.45553
7	+ Saul Delabrida		0.44343	0.44571
8	+ Inner Peace		0.28096	0.28282
9	+ DMIR		0.27137	0.27439
10	dddnnn		0.25921	0.26185

Если известно хорошее статистическое описание объекта, то признаковое описание бесполезно

Автор, область, кратность (в п-т посл-ти)	Название
Anastasia Krithara Text Mining	Active, Semi-Supervised Learning for Textual Information Access
Isabelle Guyon Machine Learning	Introduction to Machine Learning
Mikaela Keller Statistics	Basics of probability and statistics
Ulrike von Luxburg, Clustering, 5×	Lectures on Clustering
William Cohen, Text Mining, 4×	Text Classification
John Shawe-Taylor, Statistical Learning, 3×	Statistical Learning Theory
Cynthia Rudin, Boosting, 3×	The Dynamics of AdaBoost

Итог

Как и раньше, визуализация помогает

Сначала делайте простой метод!

Можно решать задачи с агрегированной информацией (например, с пост-троечными последовательносьями)

Литература

Лекция «Шаманство в анализе данных» http://alexanderdyakonov.narod.ru/lpotdyakonov.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

Дьяконов А.Г. Алгоритмы для рекомендательной системы: технология LENKOR // Бизнес-информатика, 2012, 1 (19) https://bijournal.hse.ru/2012--1(19)/53535879.html