

Машинное обучение и классификация (РСА)

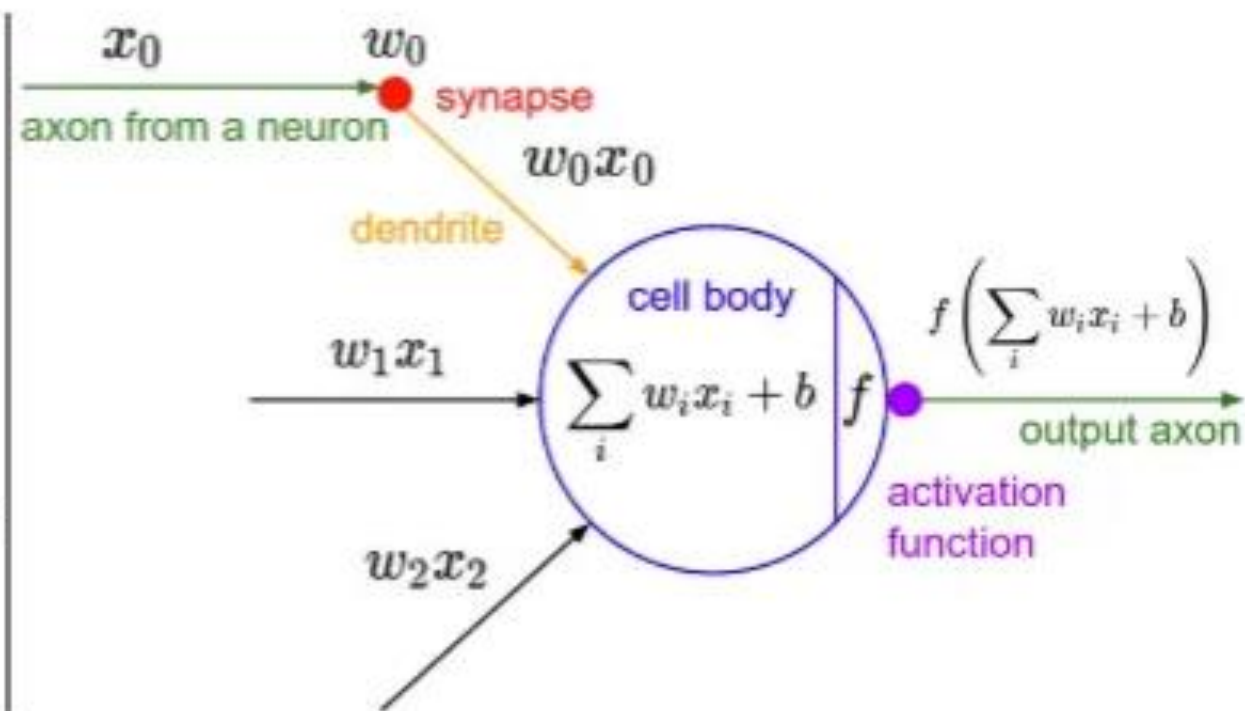
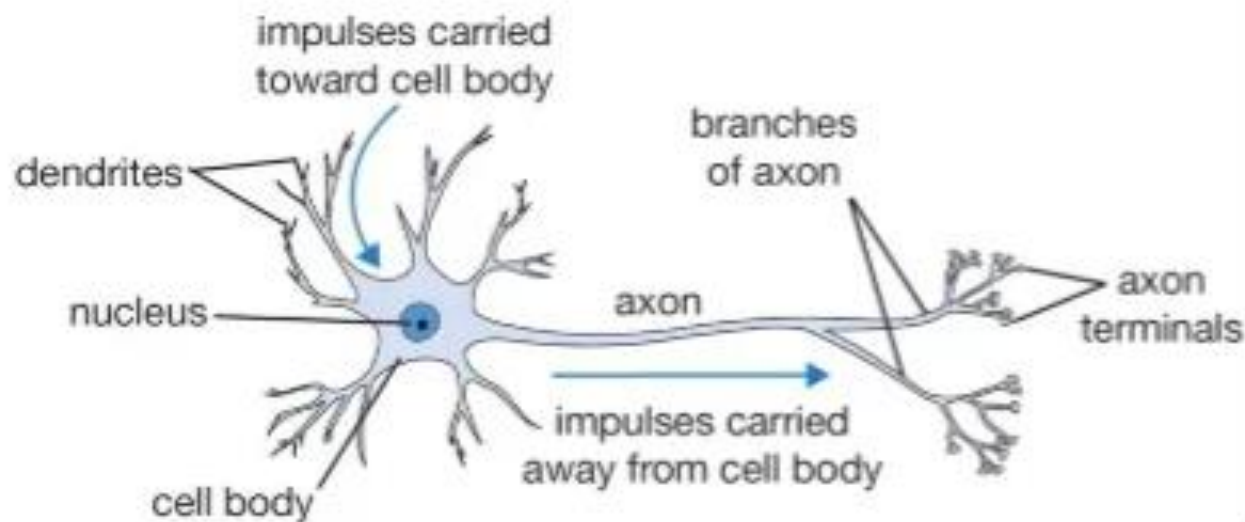
Парфенов П.С.

https://github.com/qrspeter/pca_pbs

Машинное обучение и классификация

- Единичный нейрон
 - Математическая модель, функция активации
 - Обучение на одномерных массивах (от 2x1 до спектра КТ PbS)
- Метод главных компонент (классификация)
 - Спектры молока с разной концентрацией лактозы
 - Спектры PbS двух типов
- Анализ данных с сенсора
 - Распознавание спиртов
 - МО тут лишнее? (нет)

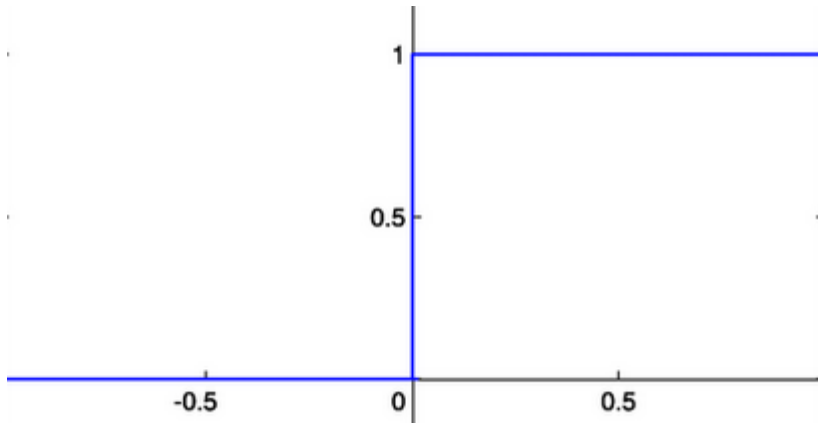
Нейрон и его математическая модель



Функция активации

- Пороговая передаточная функция

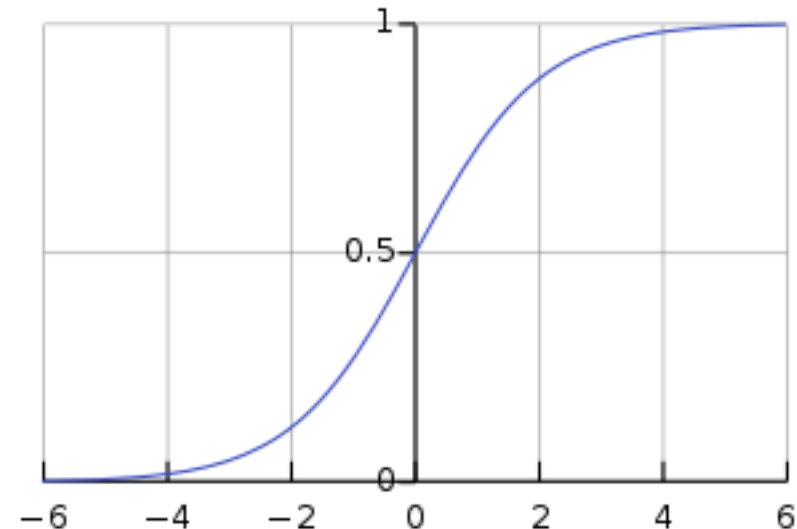
$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{else} \end{cases}$$



$$x = \sum w_i x_i + b$$

- Сигмоидальная передаточная функция

$$\sigma(x) = \frac{1}{(1 + \exp(-tx))}$$



Обучение на массиве 3x1

	Input			Output
Example 1	0	0	1	0
Example 2	1	1	1	1
Example 3	1	0	1	1
Example 4	0	1	1	0

New situation	1	0	0	?
---------------	---	---	---	---

$$\begin{cases} w_1 \cdot 0 + w_2 \cdot 0 + w_3 \cdot 1 = 0 \\ w_1 \cdot 1 + w_2 \cdot 1 + w_3 \cdot 1 = 1 \\ w_1 \cdot 1 + w_2 \cdot 0 + w_3 \cdot 1 = 1 \\ w_1 \cdot 0 + w_2 \cdot 1 + w_3 \cdot 1 = 0 \end{cases}$$

$$w_1 = 1, w_2 = 0, w_3 = 0$$

$$1 \cdot 1 + 0 \cdot 0 + 0 \cdot 0 = 1$$

- Решение уравнений матричным способом

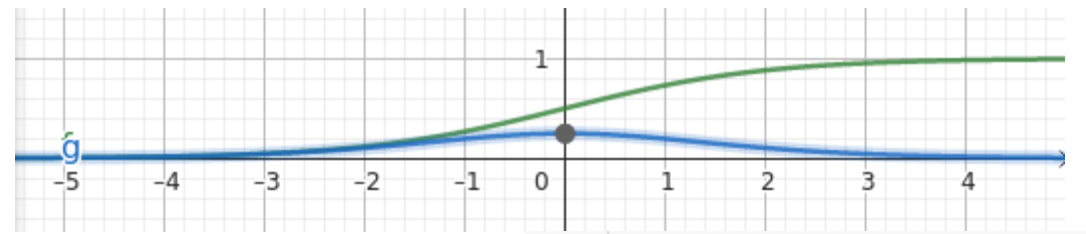
$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3 \end{cases} \quad \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \bullet \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \quad A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, \quad X = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \quad B = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$A \bullet X = B$$

$$A^{-1} \bullet A \bullet X = A^{-1} \bullet B$$

$$X = A^{-1} \bullet B$$

Алгоритм обучения



●	$f(x) = \frac{1}{1 + e^{-x}}$	⋮
●	$g(x) = \frac{1}{1 + e^{-x}} \left(1 - \frac{1}{1 + e^{-x}} \right)$	

- 1. Тренировочные наборы input и output
- 2. Случайный выбор весовых коэффициентов w_1 , w_2 и w_3
- 3. Расчет суммарного сигнала от входов

$$\sum w_i \cdot x_i = w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3$$

- 4. Расчет выходного значения нейрона

$$f = \frac{1}{1 + e^{-\sum w_i \cdot x_i}}$$

- 5. Расчет поправки к весовым коэффициентам

$$\Delta = error \cdot input \cdot \frac{\partial f}{\partial x} = (output - f) \cdot input \cdot f \cdot (1 - f)$$

- 6. $w + \Delta \Rightarrow w$, повторить с п. 3.

(neural.py)

```
from numpy import exp, array, random, dot
```

```
training_set_inputs = array([[0, 0, 1], [1, 1, 1], [1, 0, 1], [0, 1, 1]])
```

```
training_set_outputs = array([[0, 1, 1, 0]]).T
```

```
random.seed(1)
```

```
synaptic_weights = 2 * random.random((3, 1)) - 1
```

```
for iteration in xrange(10000):
```

```
    output = 1 / (1 + exp(-(dot(training_set_inputs, synaptic_weights))))
```

```
    synaptic_weights += dot(training_set_inputs.T, (training_set_outputs - output) * output * (1 - output))
```

1-й:

```
[[0.2689864 ]  
 [0.3262757 ]  
 [0.23762817]  
 [0.36375058]]
```

2-й:

```
[[0.29929909]  
 [0.44378327]  
 [0.32511054]  
 [0.41433436]]
```

```
[[0.32550793]  
 [0.55437725]  
 [0.41995875]  
 [0.45332188]]
```

```
[[0.34051508]  
 [0.63066567]  
 [0.49900324]  
 [0.46955296]] ....
```

1000-й:

```
[[0.03178421]  
 [0.97414645]  
 [0.97906682]  
 [0.02576499]]
```

```
print (1 / (1 + exp(-(dot(array([1, 0, 0]), synaptic_weights)))))
```

🔍 [0.99929937]

training_set_inputs:

```
[[0 0 1]  
 [1 1 1]  
 [1 0 1]  
 [0 1 1]]
```

training_set_outputs:

```
[[0]  
 [1]  
 [1]  
 [0]]
```

synaptic_weights (0-й):

```
[[ -0.16595599],  
 [ 0.44064899],  
 [-0.99977125]]
```

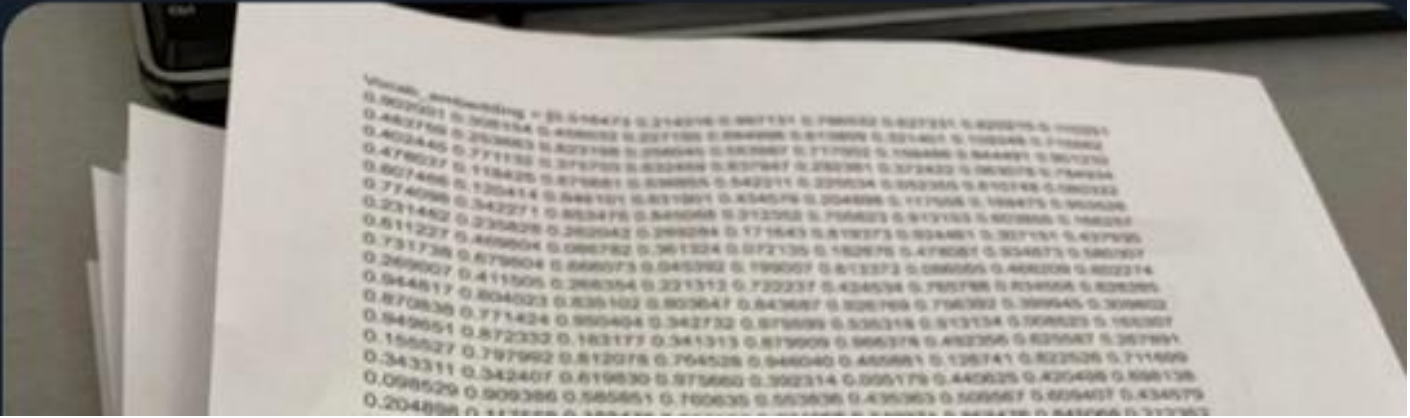



Owen

@O42nl

Printed the chatgpt weights and will be multiplying matrices for each question (hope each question isn't too many tokens)

Prof said we can bring whatever to the open book exam as long as it is on printer paper



Vocab_embedding = [0.546475 0.214316 0.987151 0.788022 0.827351 0.820216 0.110251
0.502201 0.305154 0.408012 0.227155 0.884998 0.813895 0.321401 0.102246 0.718802
0.482758 0.253883 0.507198 0.258045 0.583887 0.717052 0.708486 0.844891 0.501370
0.402445 0.771132 0.371703 0.832484 0.837947 0.250161 0.372432 0.063078 0.784934
0.478027 0.118425 0.871881 0.838854 0.542311 0.325534 0.552355 0.810748 0.080332
0.507486 0.120414 0.944101 0.831981 0.434576 0.204898 0.117508 0.188475 0.500526
0.774098 0.342271 0.853475 0.845098 0.312352 0.758823 0.813153 0.803885 0.788257
0.231462 0.235828 0.282542 0.289284 0.171643 0.810373 0.804481 0.307151 0.437950
0.811227 0.235828 0.282542 0.289284 0.171643 0.810373 0.804481 0.307151 0.437950
0.731738 0.879804 0.888782 0.361324 0.072135 0.182878 0.478081 0.834873 0.580907
0.269907 0.411505 0.288354 0.221312 0.722237 0.434534 0.780788 0.834854 0.828385
0.544817 0.804023 0.838102 0.803647 0.843687 0.828788 0.758392 0.389845 0.309802
0.870835 0.771424 0.885404 0.342732 0.878899 0.538318 0.813154 0.508823 0.180307
0.849851 0.872332 0.183177 0.341313 0.879809 0.888378 0.482356 0.825887 0.387881
0.155527 0.787982 0.812078 0.764528 0.848040 0.485881 0.138741 0.822526 0.711689
0.343311 0.342407 0.819830 0.875880 0.392314 0.055178 0.440325 0.420488 0.888138
0.098529 0.909386 0.585851 0.788635 0.553836 0.435363 0.509567 0.805407 0.434575
0.204898 0.411505 0.288354 0.221312 0.722237 0.434534 0.780788 0.834854 0.828385

Массив 2x1 (neural_single_eq.py)

```
from numpy import exp, array, random, dot
training_set_inputs = array([[0, 1]])
training_set_outputs = array([[0]]).T
```

[[0, 1]]
[[0]]

$$w_1 \cdot 0 + w_2 \cdot 1 = 0$$

```
iterations = 1000
random.seed(1)
synaptic_weights = 2 * random.random((2, 1)) - 1
```

[[-0.16595599],
[0.44064899]]

```
for iteration in range(iterations):
```

```
    output = 1 / (1 + exp(-(dot(training_set_inputs, synaptic_weights))))
    synaptic_weights += dot(training_set_inputs.T, (training_set_outputs - output) * output * (1 - output))
```

[[0.60841366]]

[[-0.16595599],
[0.29569657]]

```
print( 1 / (1 + exp(-(dot(array([1, 1]), synaptic_weights)))))
print( 1 / (1 + exp(-(dot(array([1, 0]), synaptic_weights)))))
print( 1 / (1 + exp(-(dot(array([0, 0]), synaptic_weights)))))
```

[0.01994744]
[0.45860596]
[0.5]

w1 останется таким, каким был назначен изначально

Массив 10x1 (neural_class_10.py)

```
training_set_inputs = array([[0.5, 1, 0.5, 0, 0, 0, 0, 0, 0, 0], [0, 0.5, 1, 0.5, 0, 0, 0, 0, 0, 0],  
                             [0, 0, 0.5, 1, 0.5, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0.5, 1, 0.5, 0, 0],  
                             [0, 0, 0, 0, 0, 0, 0.5, 1, 0.5, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0.5, 1, 0.5]])  
training_set_outputs = array([[0, 0, 0, 1, 1, 1]]).T
```

```
# [0.5, 1, 0.5, 0, 0, 0, 0, 0, 0, 0] -> 0
```

```
# [0, 0.5, 1, 0.5, 0, 0, 0, 0, 0, 0] -> 0
```

```
# [0, 0, 0.5, 1, 0.5, 0, 0, 0, 0, 0] -> 0
```

```
# [0, 0, 0, 0, 0, 0.5, 1, 0.5, 0, 0] -> 1
```

```
# [0, 0, 0, 0, 0, 0, 0.5, 1, 0.5, 0] -> 1
```

```
# [0, 0, 0, 0, 0, 0, 0, 0.5, 1, 0.5] -> 1
```

Considering new situation [0 0 0 0 0 0 0 0 0 0] -> ?:

[0.5]

Considering new situation [0 0 0 0 0 0 0 0 0 1] -> ?:

[0.7899541]

Interviewer: What's your biggest strength?

Me: I'm an expert in machine learning.

Interviewer: What's $9 + 10$?

Me: Its 3.

Interviewer: Not even close. It's 19.

Me: It's 16.

Interviewer: Wrong. Its still 19.

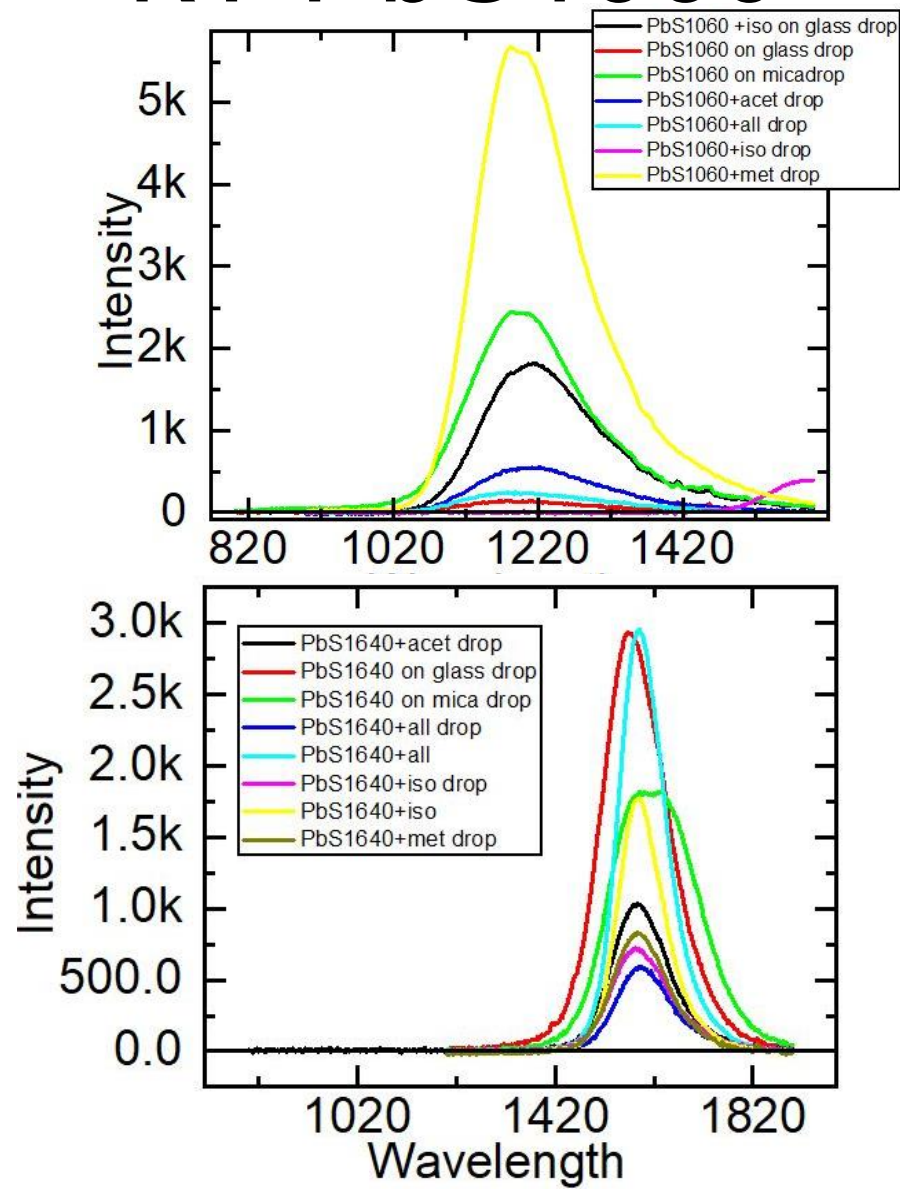
Me: It's 18.

Interviewer: No, it's 19.

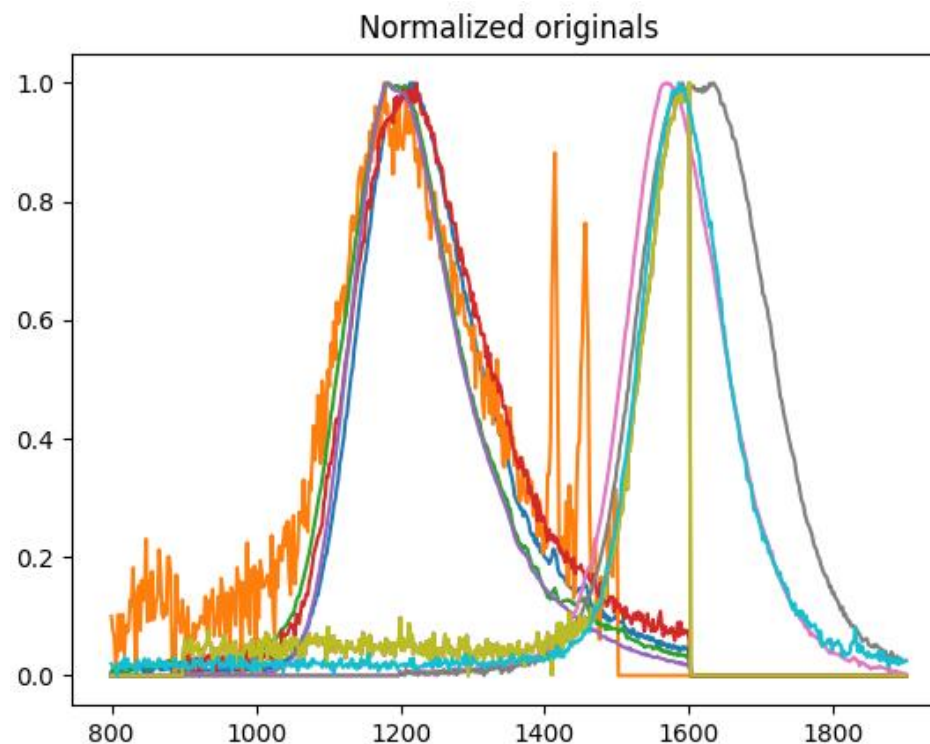
Me: it's 19.

Interviewer: You're hired

KT PbS1060 + PbS1640 (neuron_pbs.py)



1060:	1060:	1640:
900.0000	800.0000	1200.0000
902.0000	802.0000	1202.0000
...
1600.0000	1600.0000	1900.0000

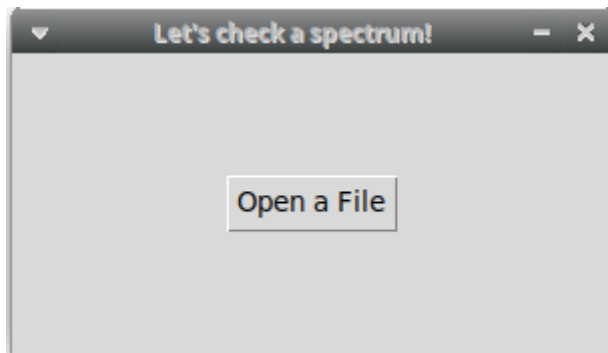


- 0 – PbS1060
- 1 – PbS1640
- Порог - 0,5

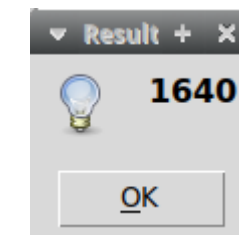
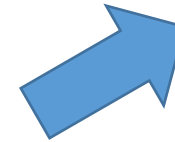
```
training_set_outputs = array([[0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1]]).T
```

```
inputs = 551
```

```
trainings = 3 #10000
```



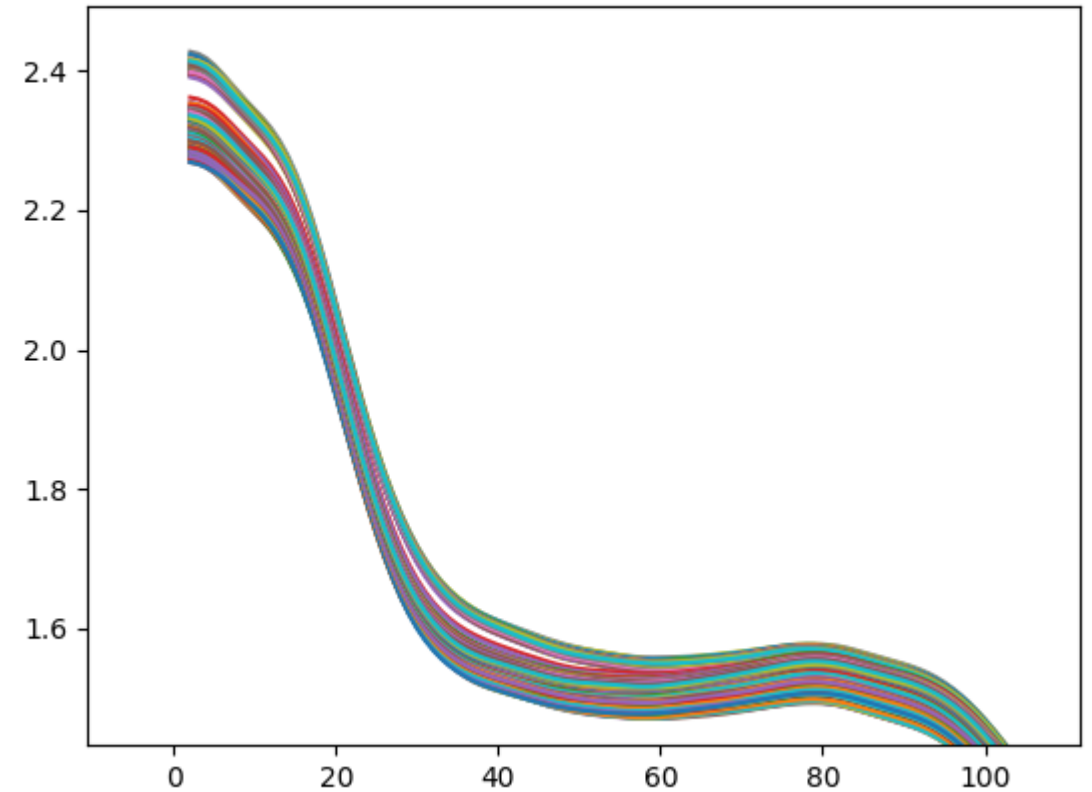
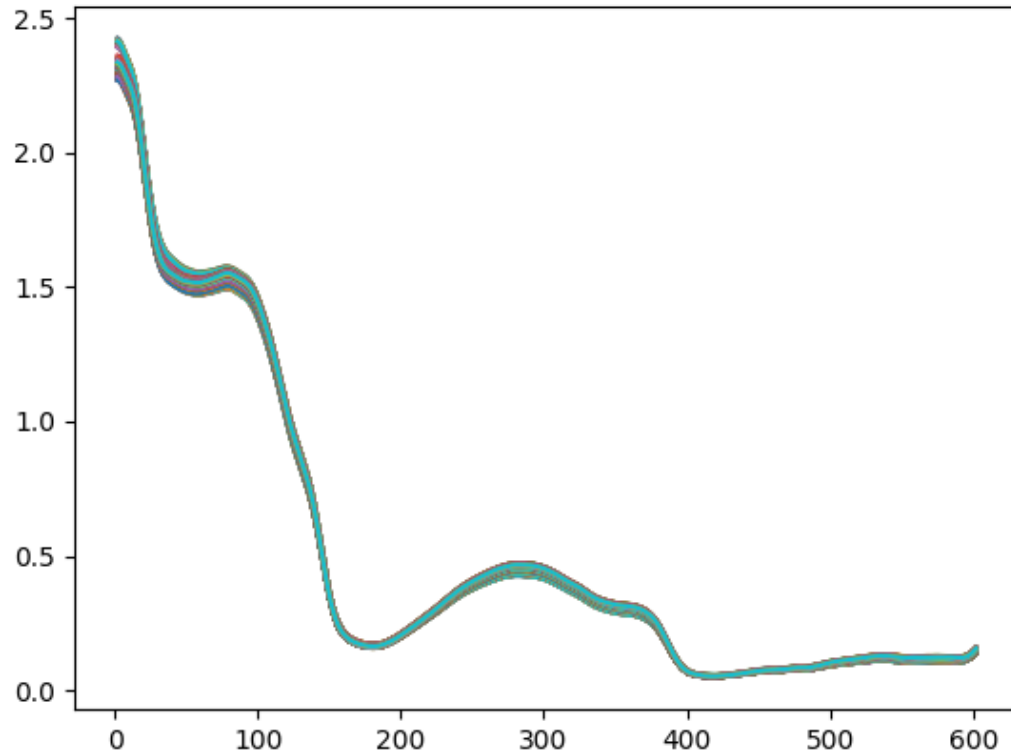
```
if result < 0.5:  
    mess = '1060'  
else:  
    mess = '1640'
```



РСА - “Изобретён Карлом Пирсоном в 1901 году.”

https://en.wikipedia.org/wiki/Principal_component_analysis

(pca_milk.py)

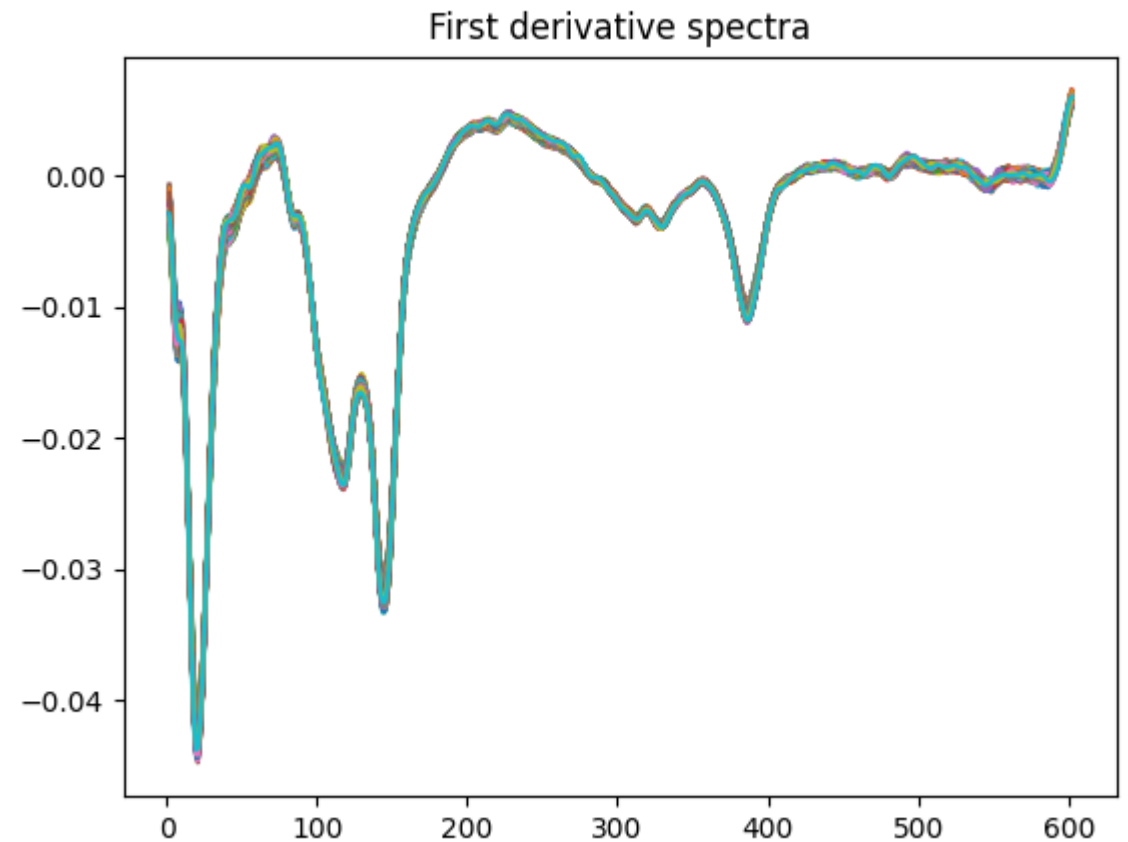
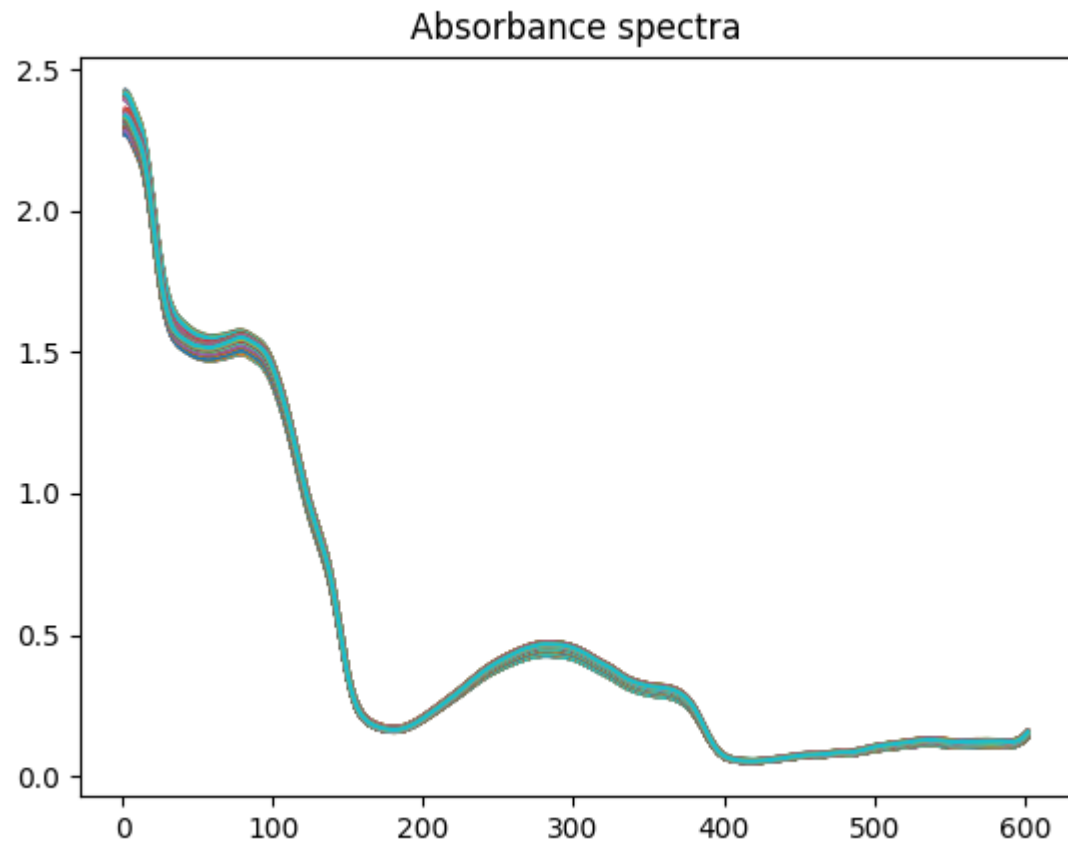


NIR spectra from milk samples with a varying concentration of lactose.

<https://nirpyresearch.com/classification-nir-spectra-principal-component-analysis-python/>

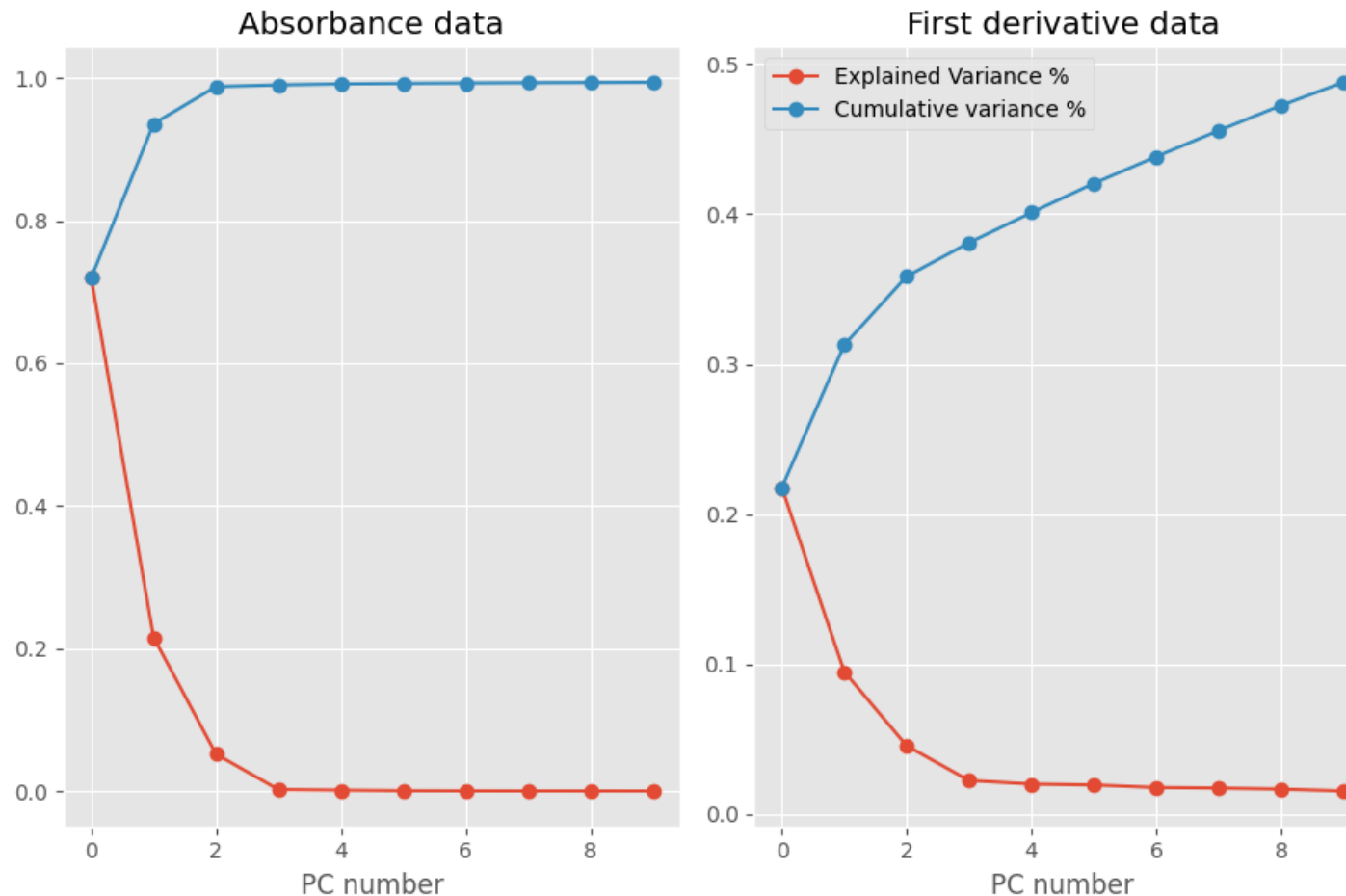
Классификация спектров

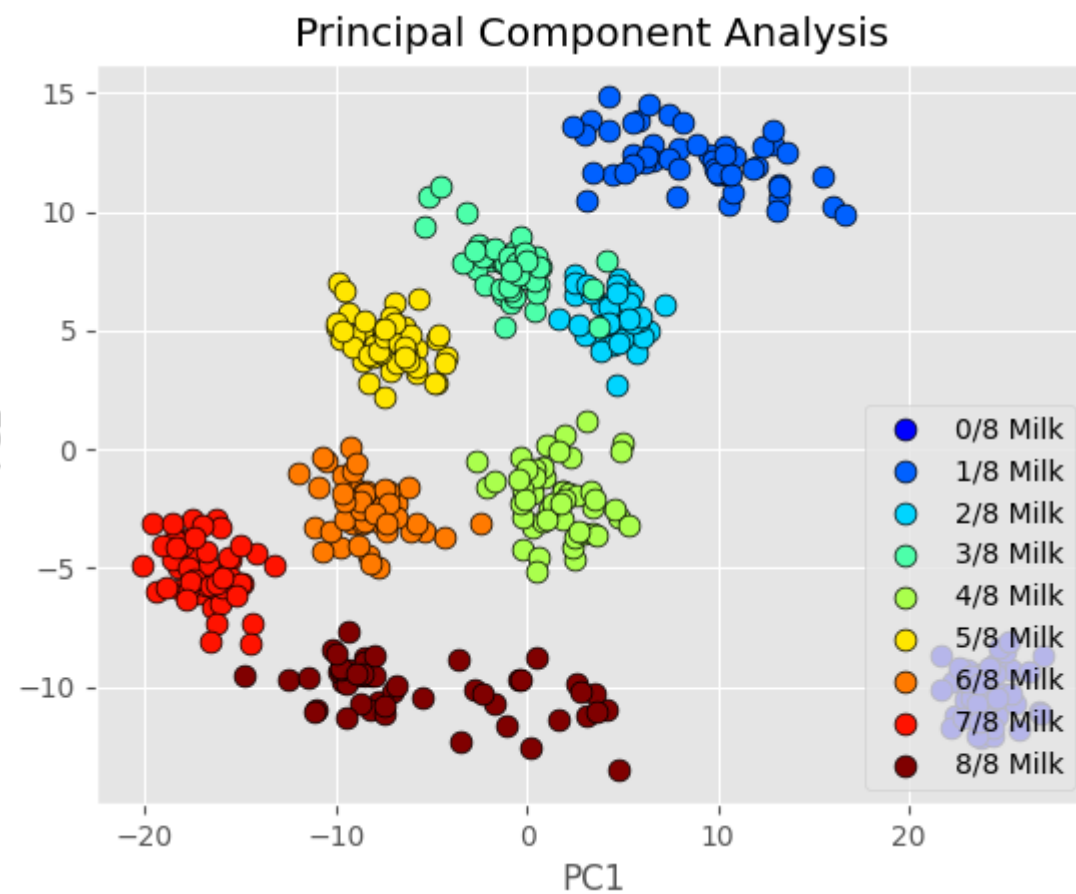
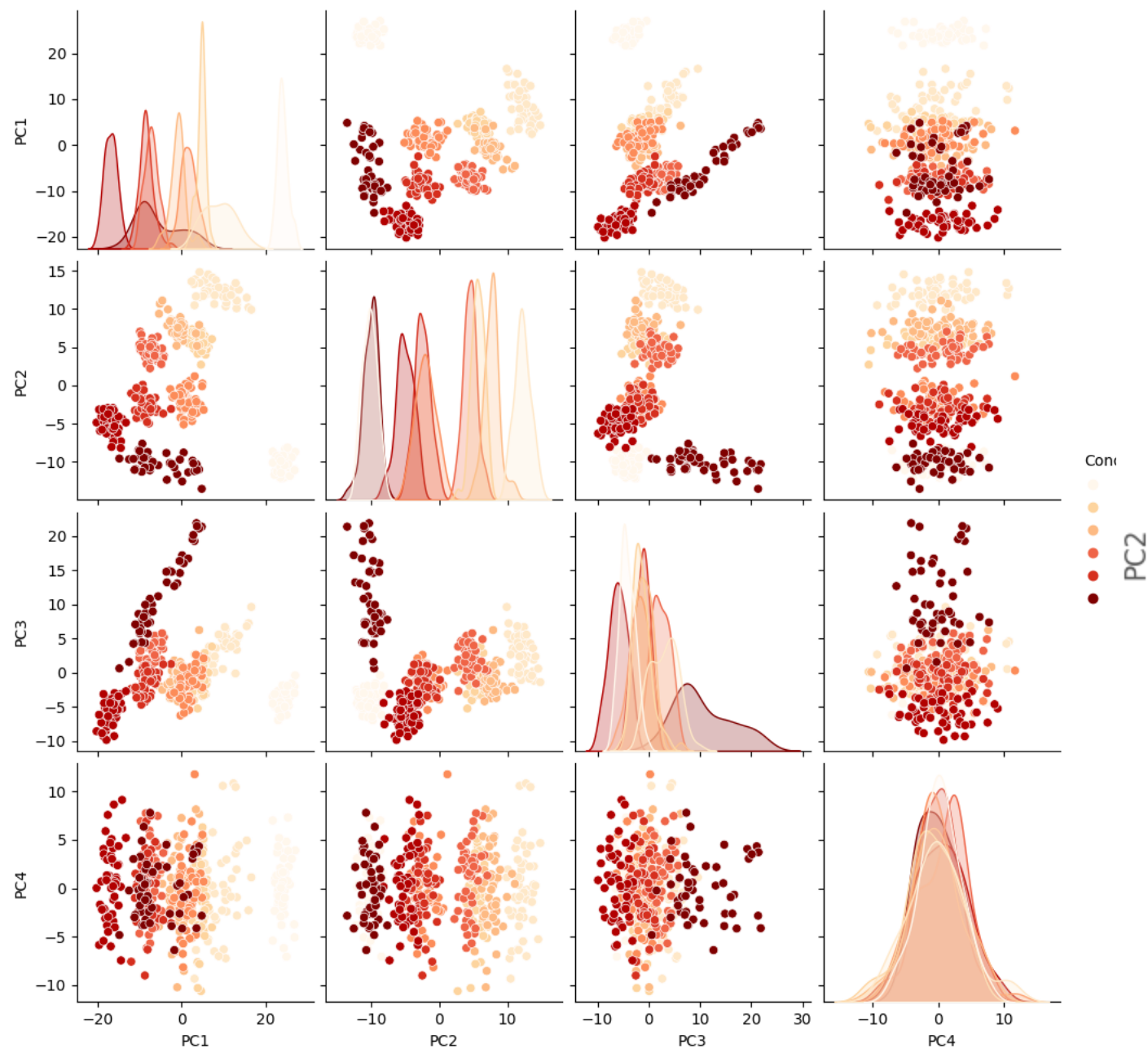
- – информативнее производная
- – желательно сглаживание



Классификация спектров

- – информативнее производная
- – желательно сглаживание
- – ОПТИМАЛЬНОЕ КОЛИЧЕСТВО КОМПОНЕНТ





ALWAYS HAS BEEN

WAIT, IT'S ALL STATISTICS?

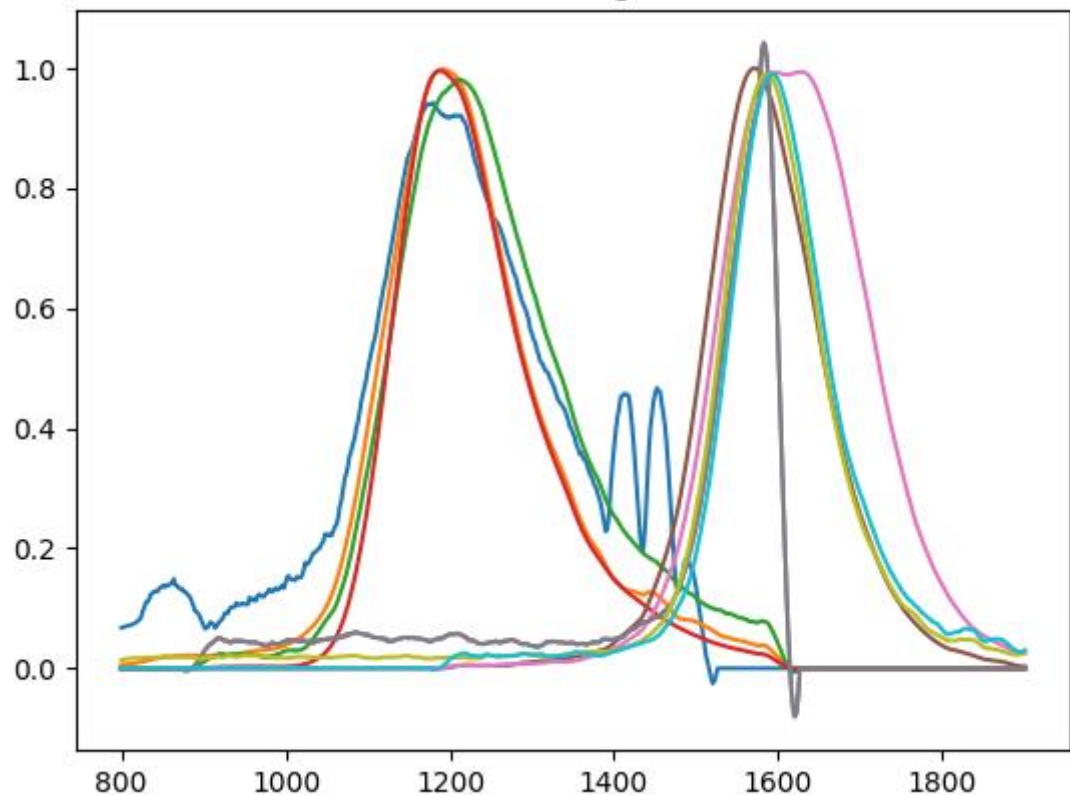
*MACHINE LEARNING
DEEP LEARNING
ARTIFICIAL INTELLIGENCE*

PCA & PbS

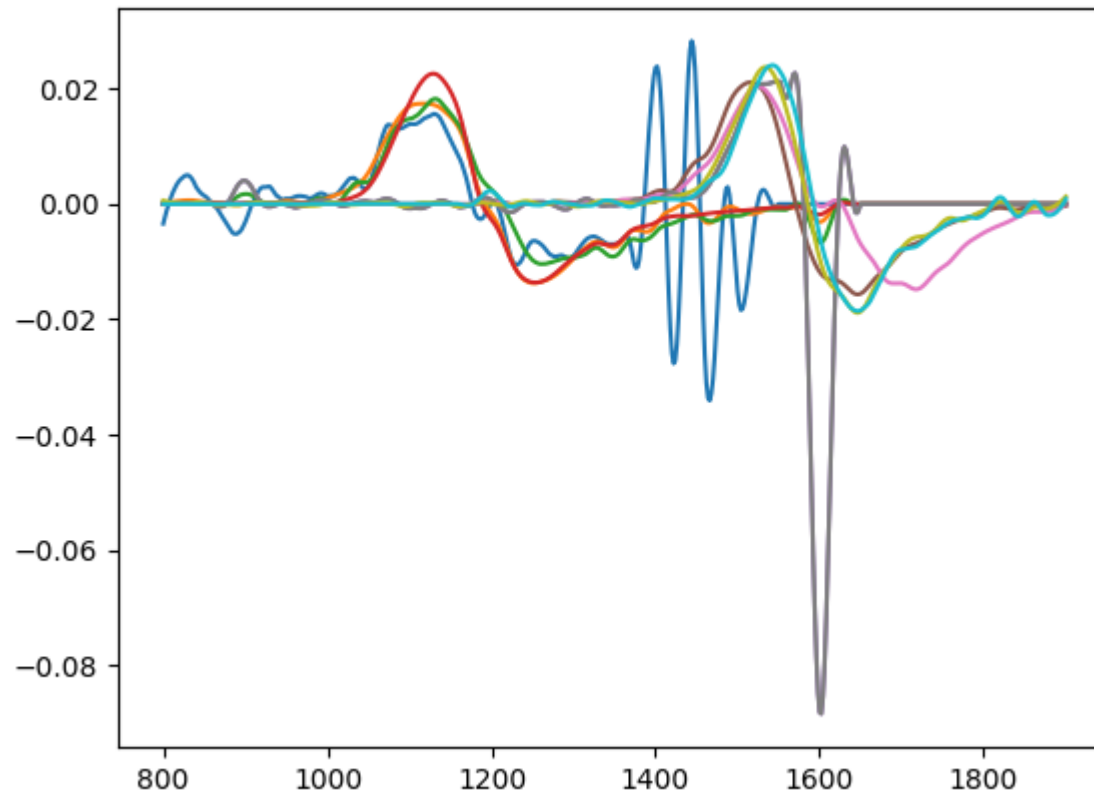
pca_pbs.py

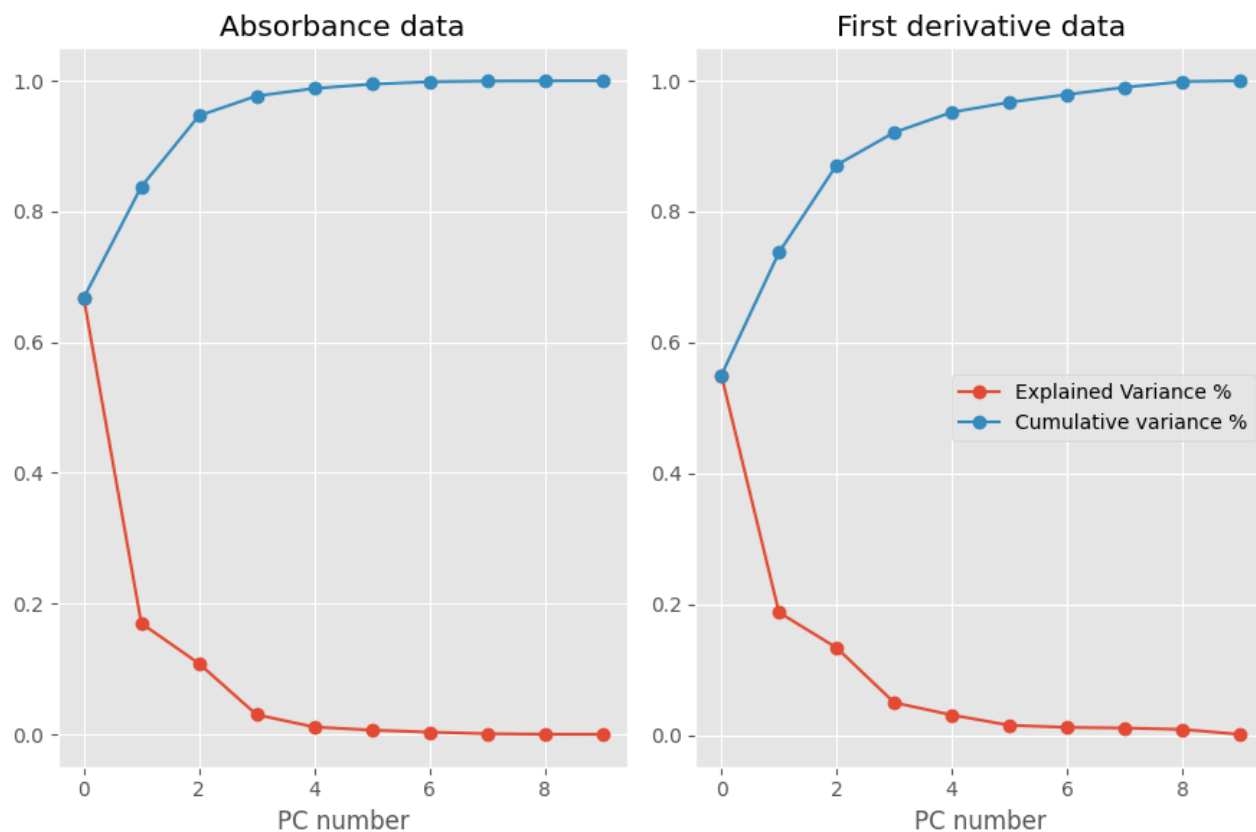
- Filtered (для производной фильтрация и шумы критичны)

Filtered originals

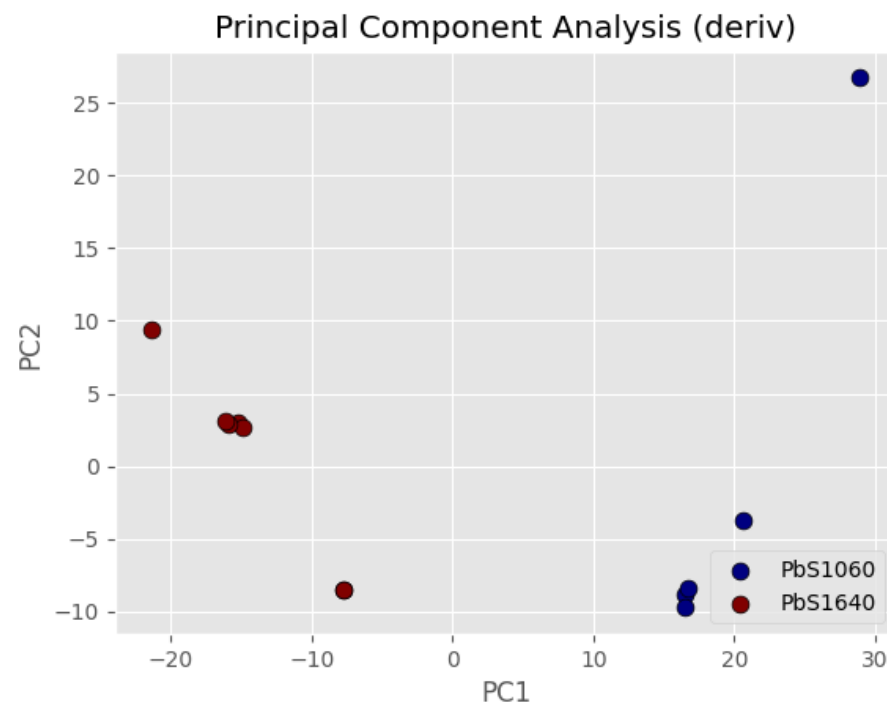
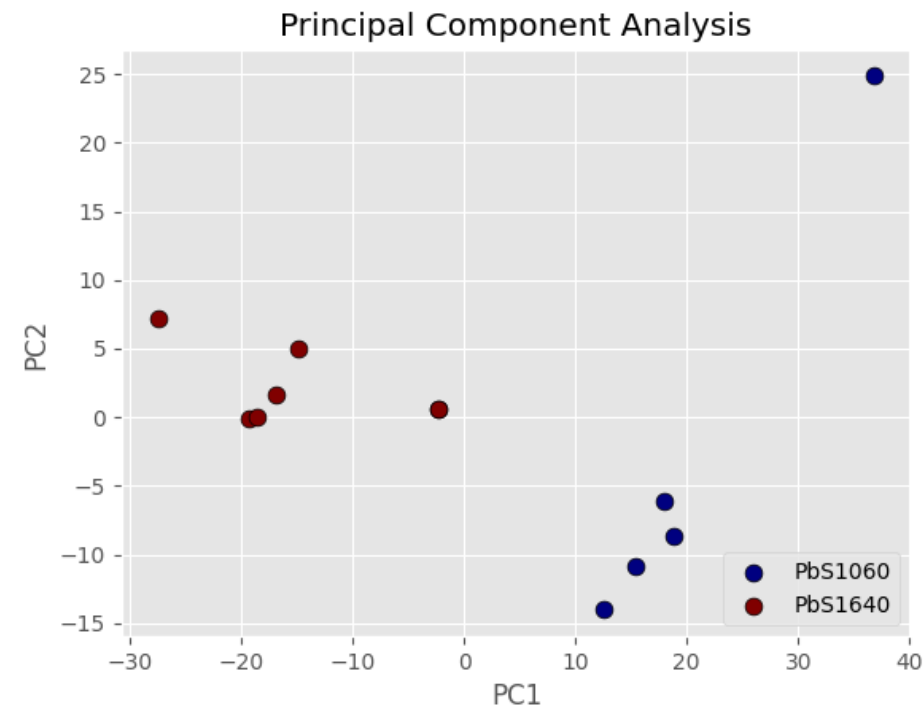


Filtered derivatives

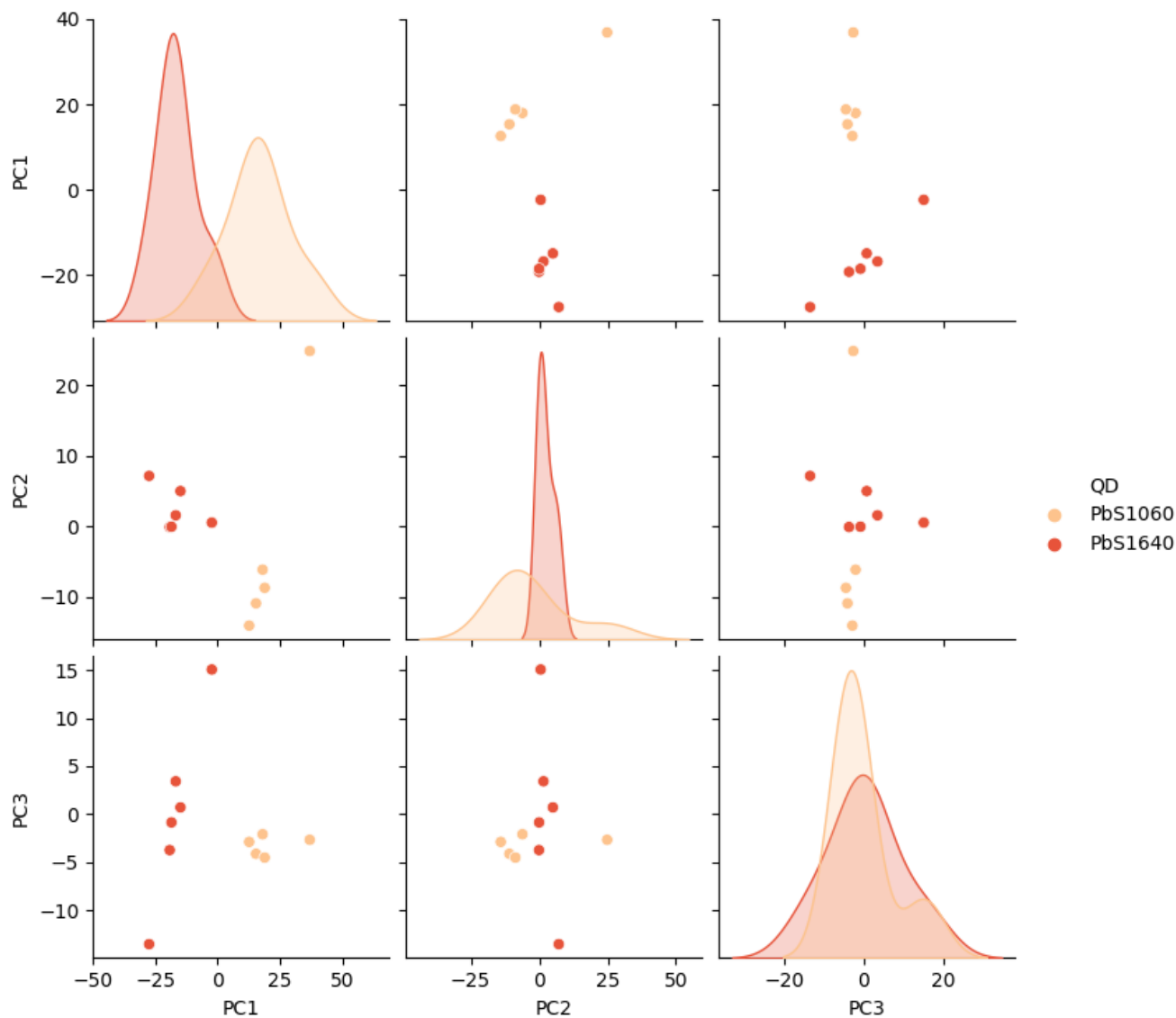


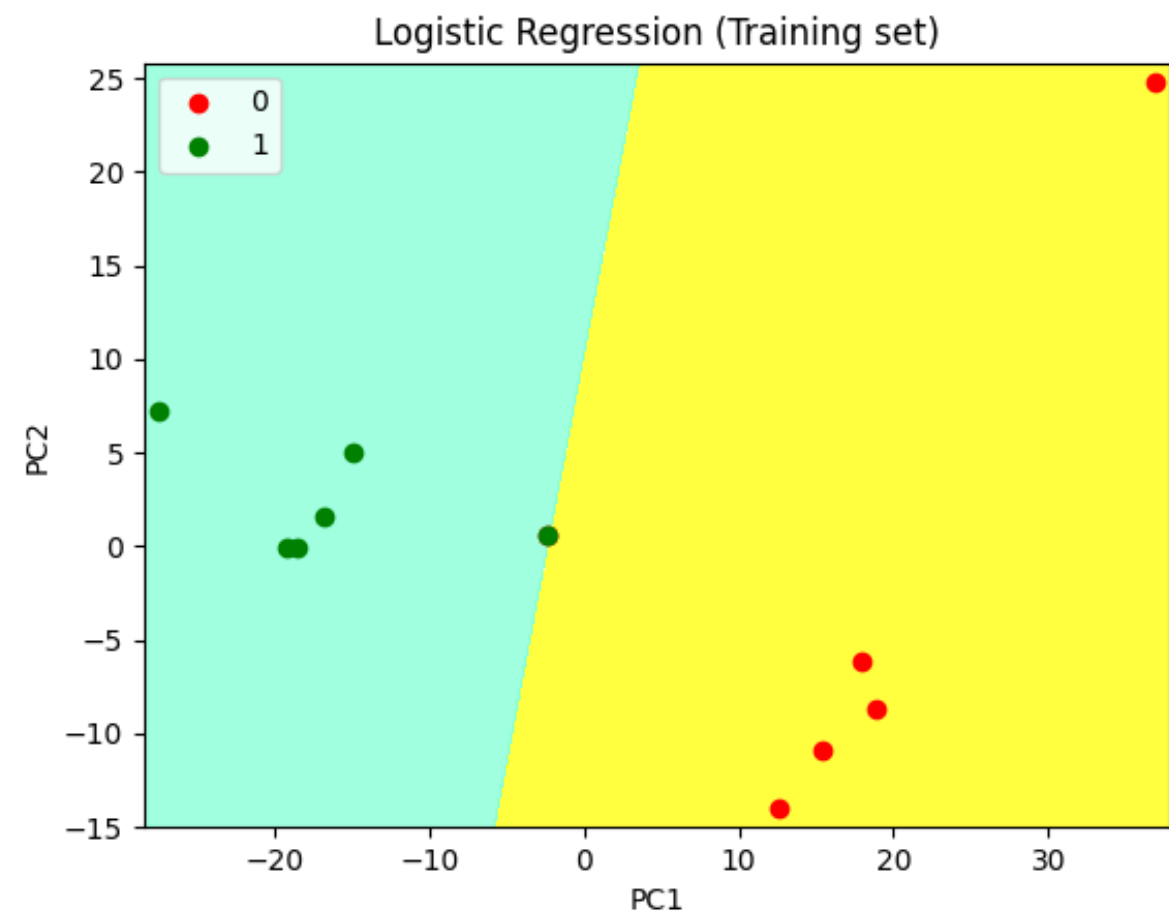
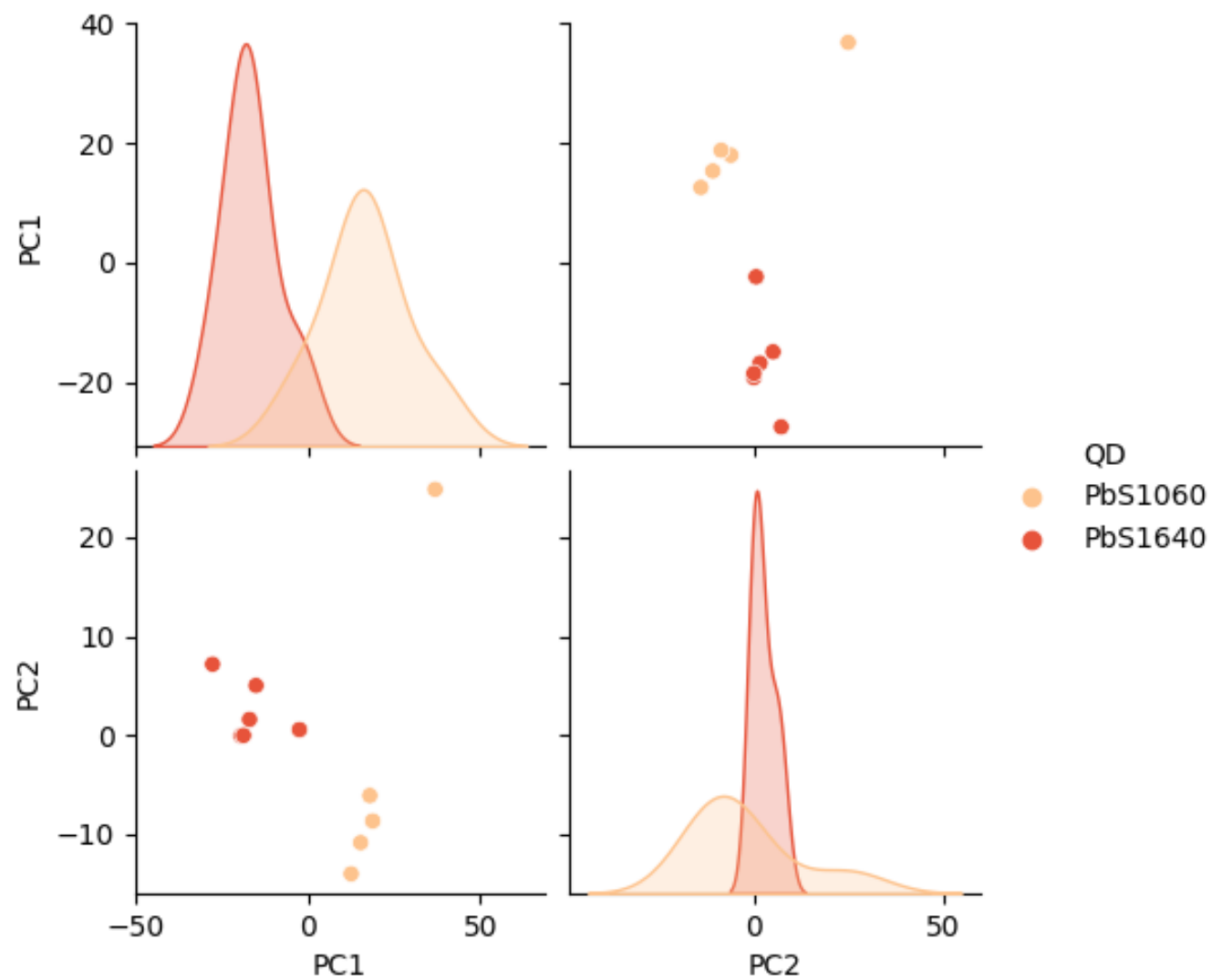


- Из-за шумов производная (nfeat2) не факт что лучше исходной (nfeat1)

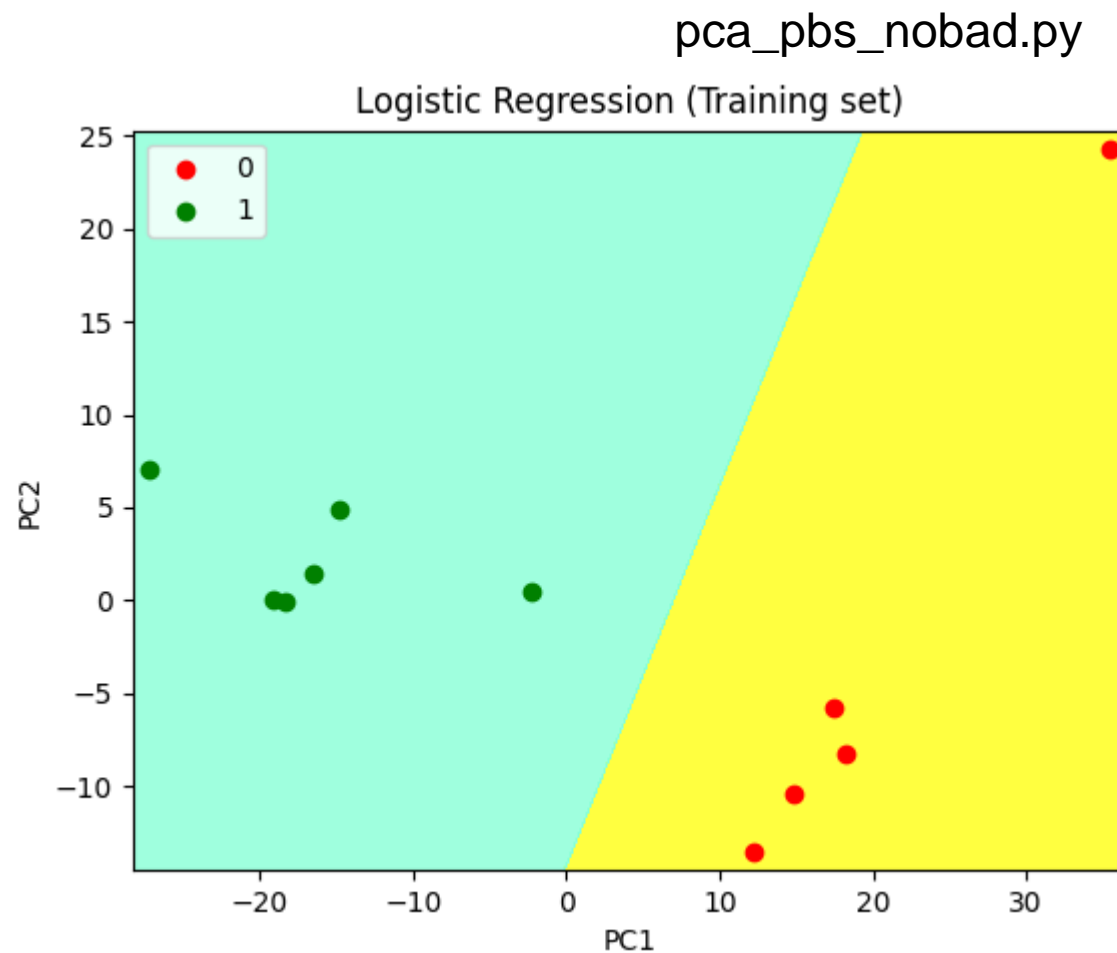
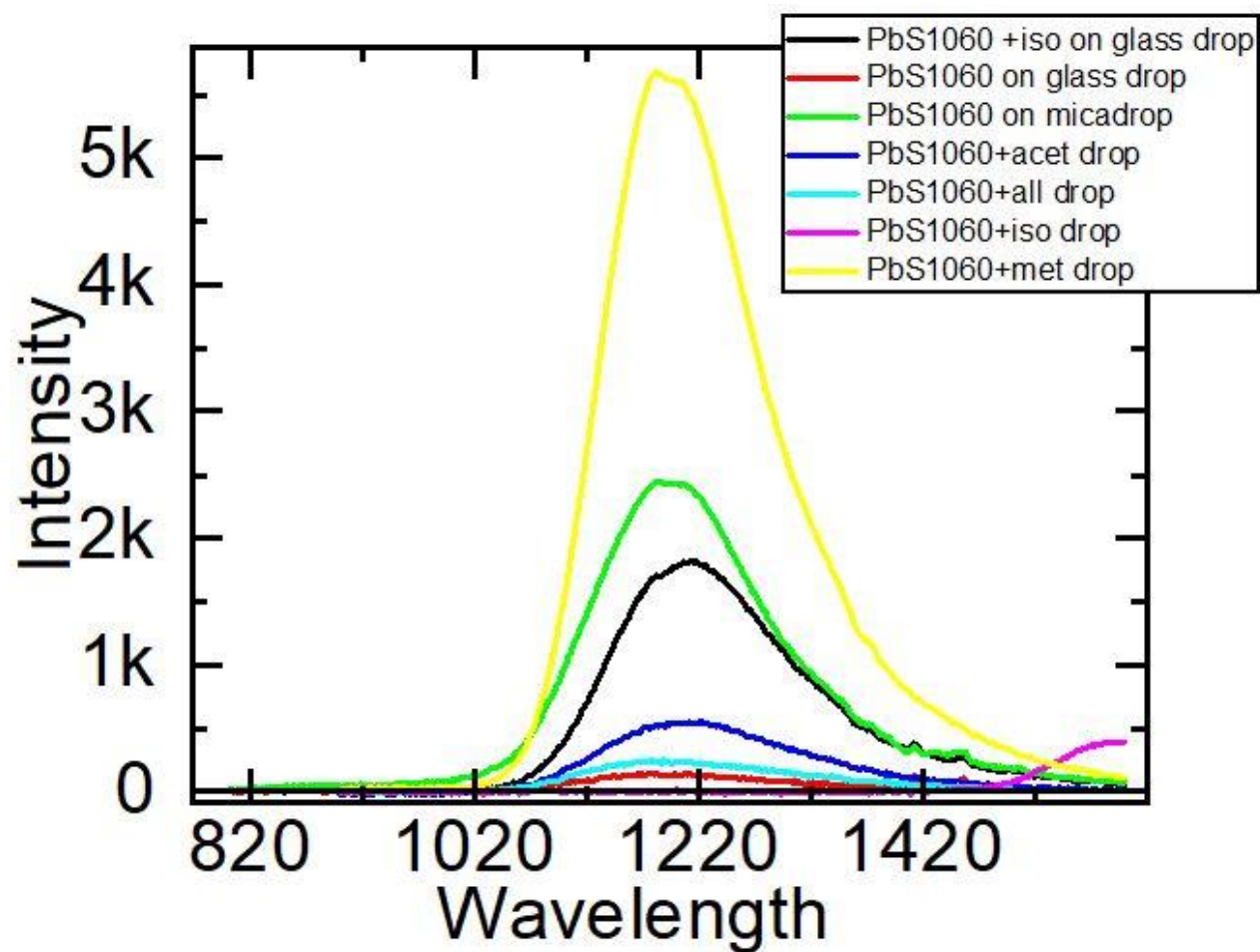


Layered kernel density estimate (KDE)





NB! Один спектр из другого набора



Это все хорошо, но где
распознавание? Где обучение?)

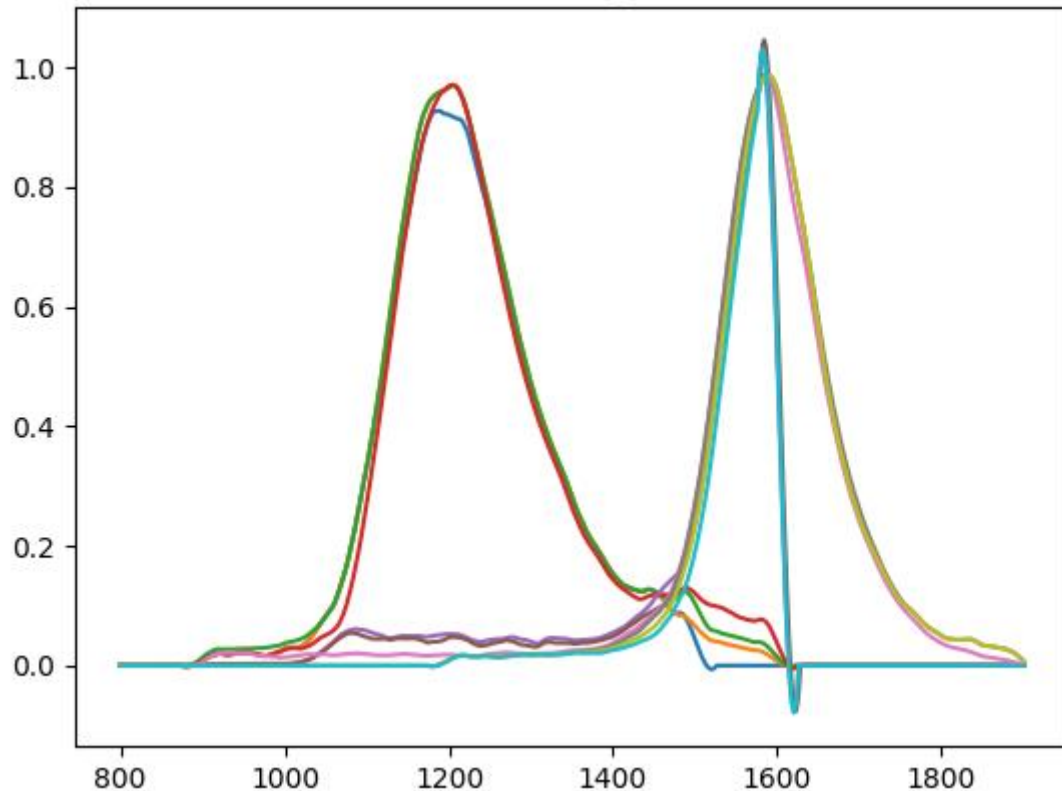
- Логистическая регрессия
- 2 компоненты PCA
- 10 – обучение, 2 – проверка.
- Добавил медианную фильтрацию

(pca_pbs_logreg_realtest.py)

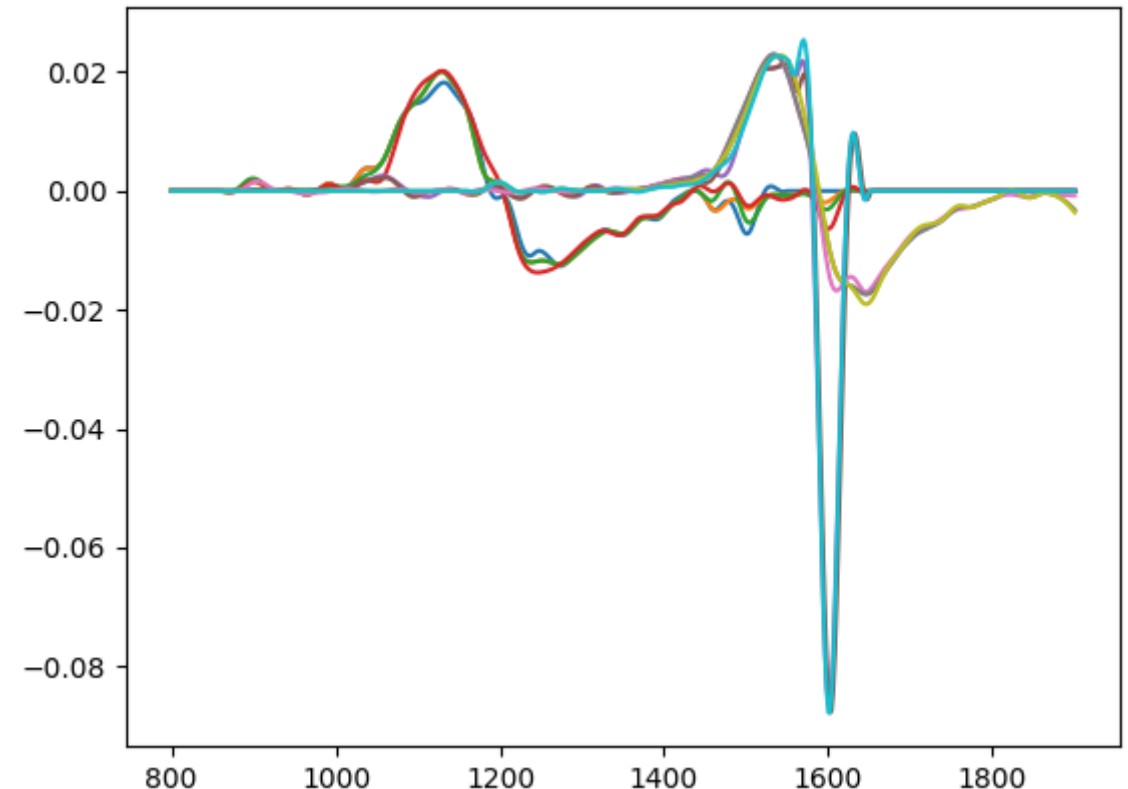
nfeat = nfeat1
test = test1

nfeat = nfeat2
test = test2

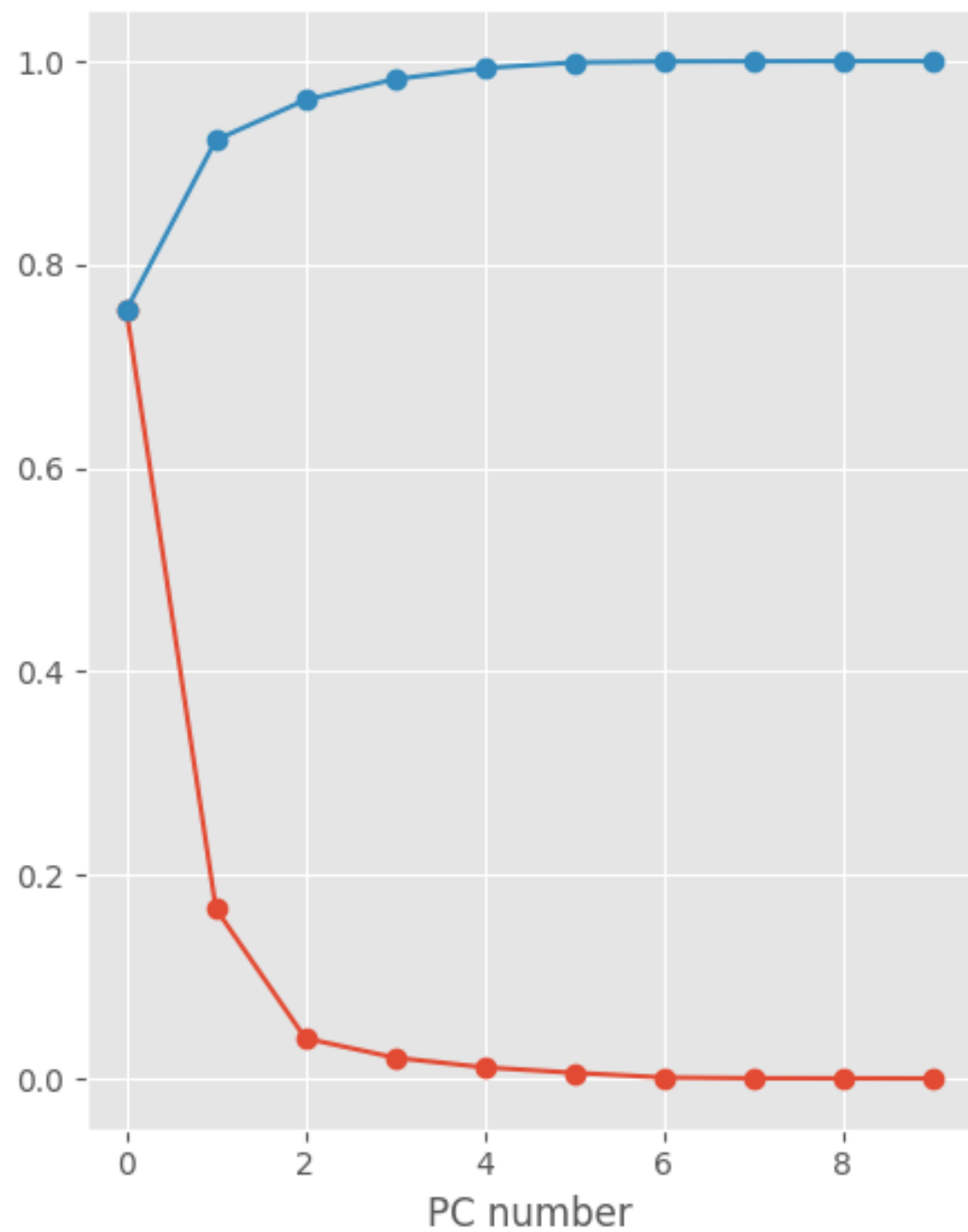
Filtered originals



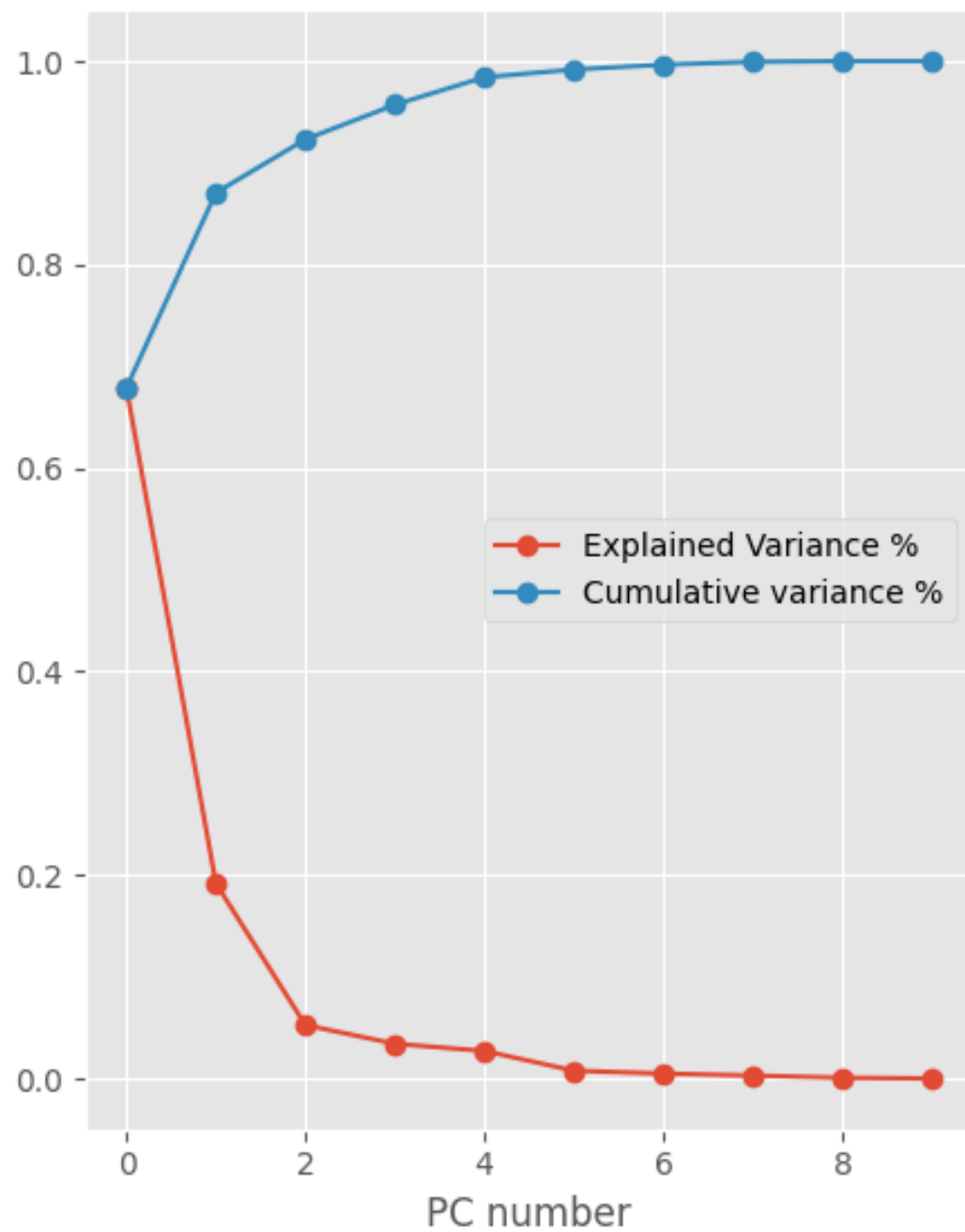
Filtered derivatives



Luminescence data

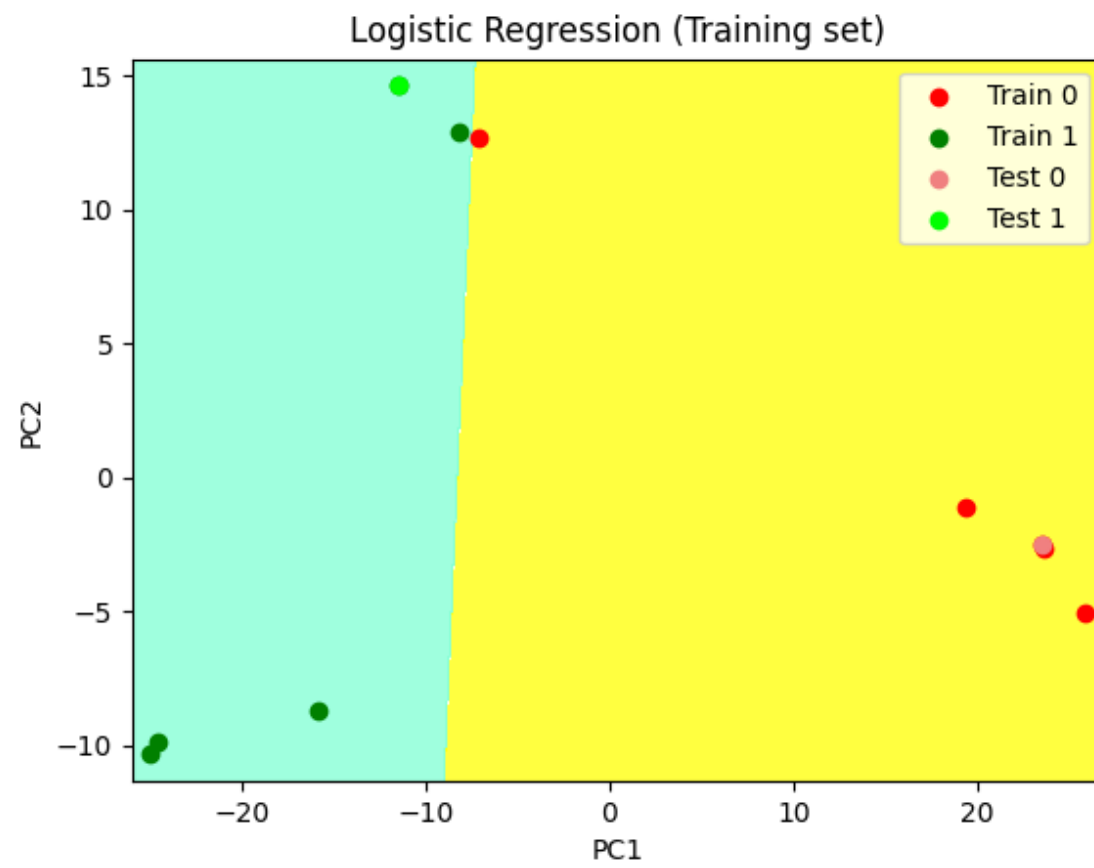
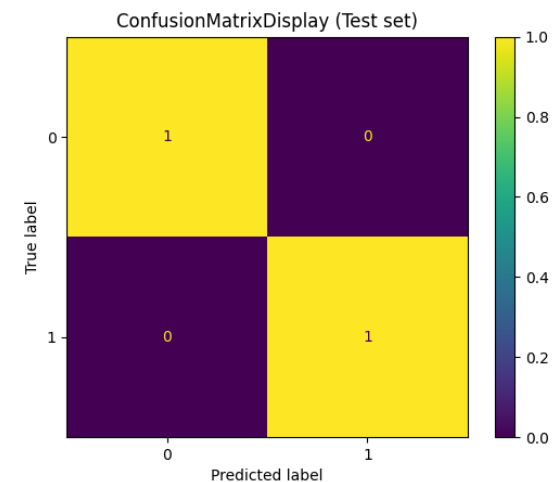
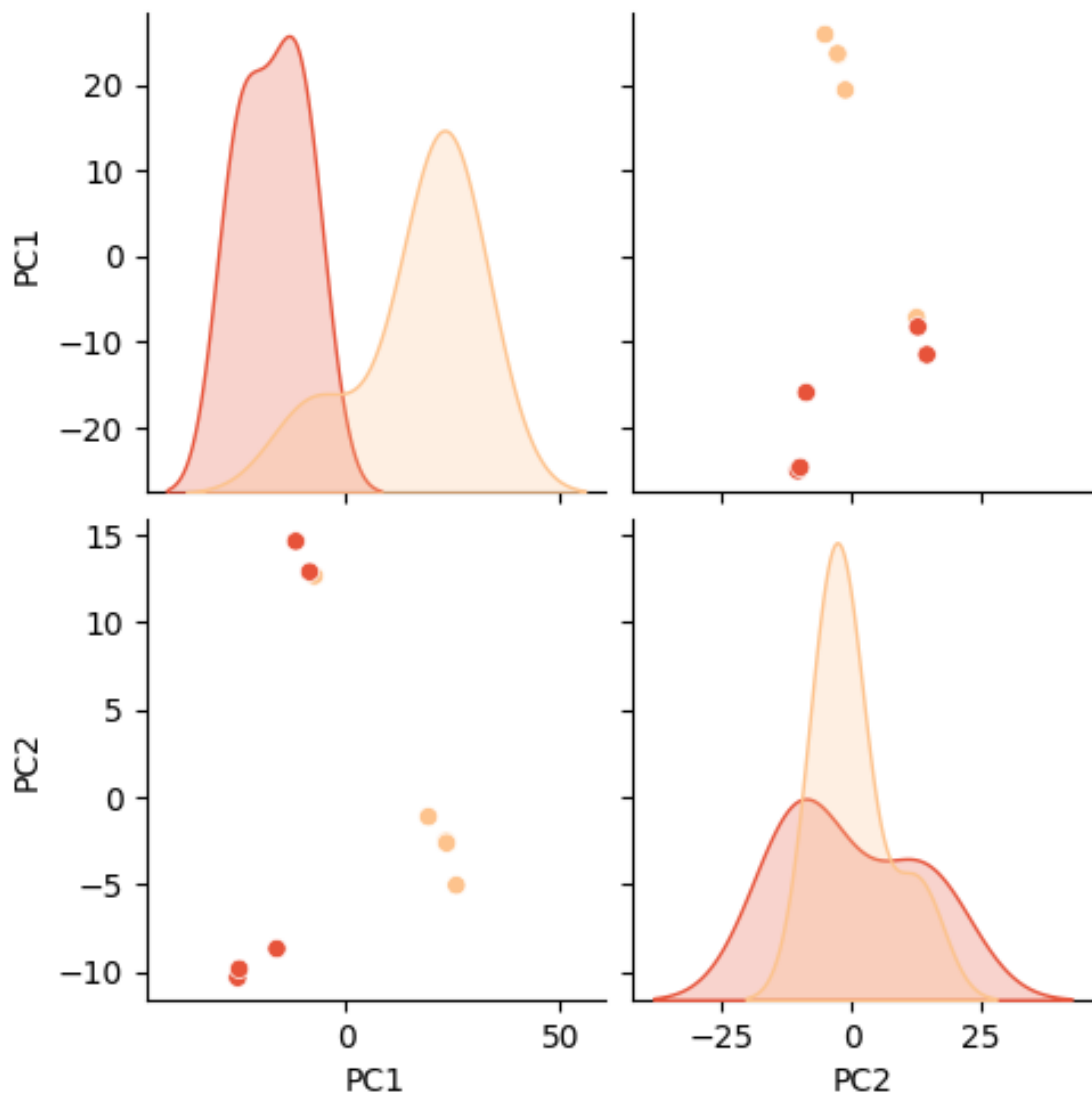


First derivative data



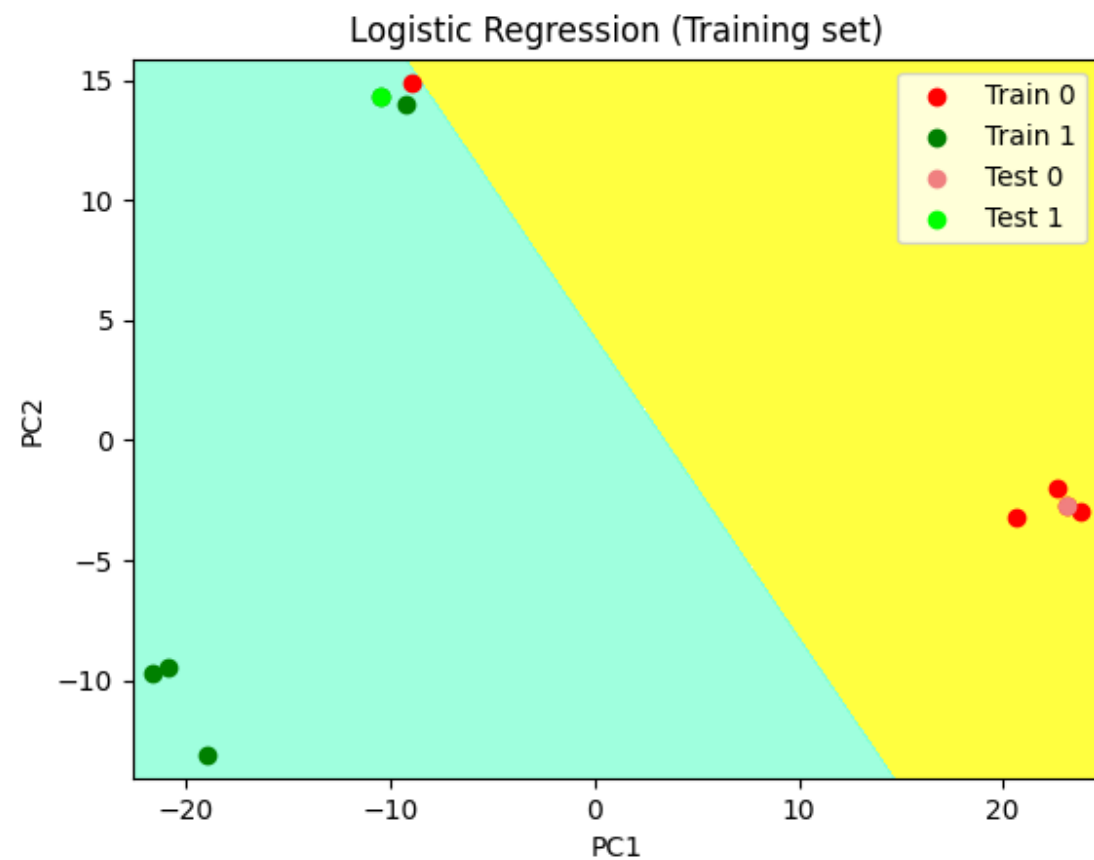
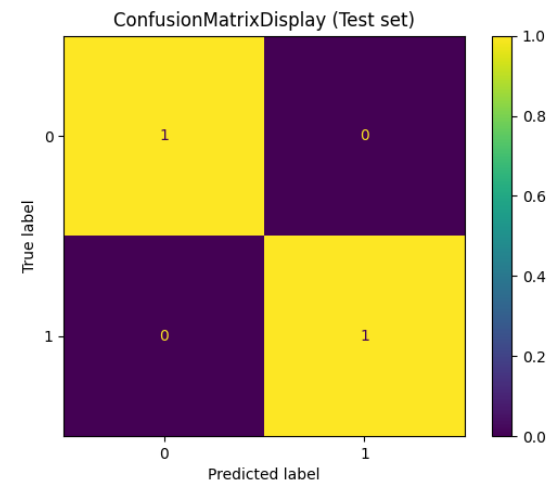
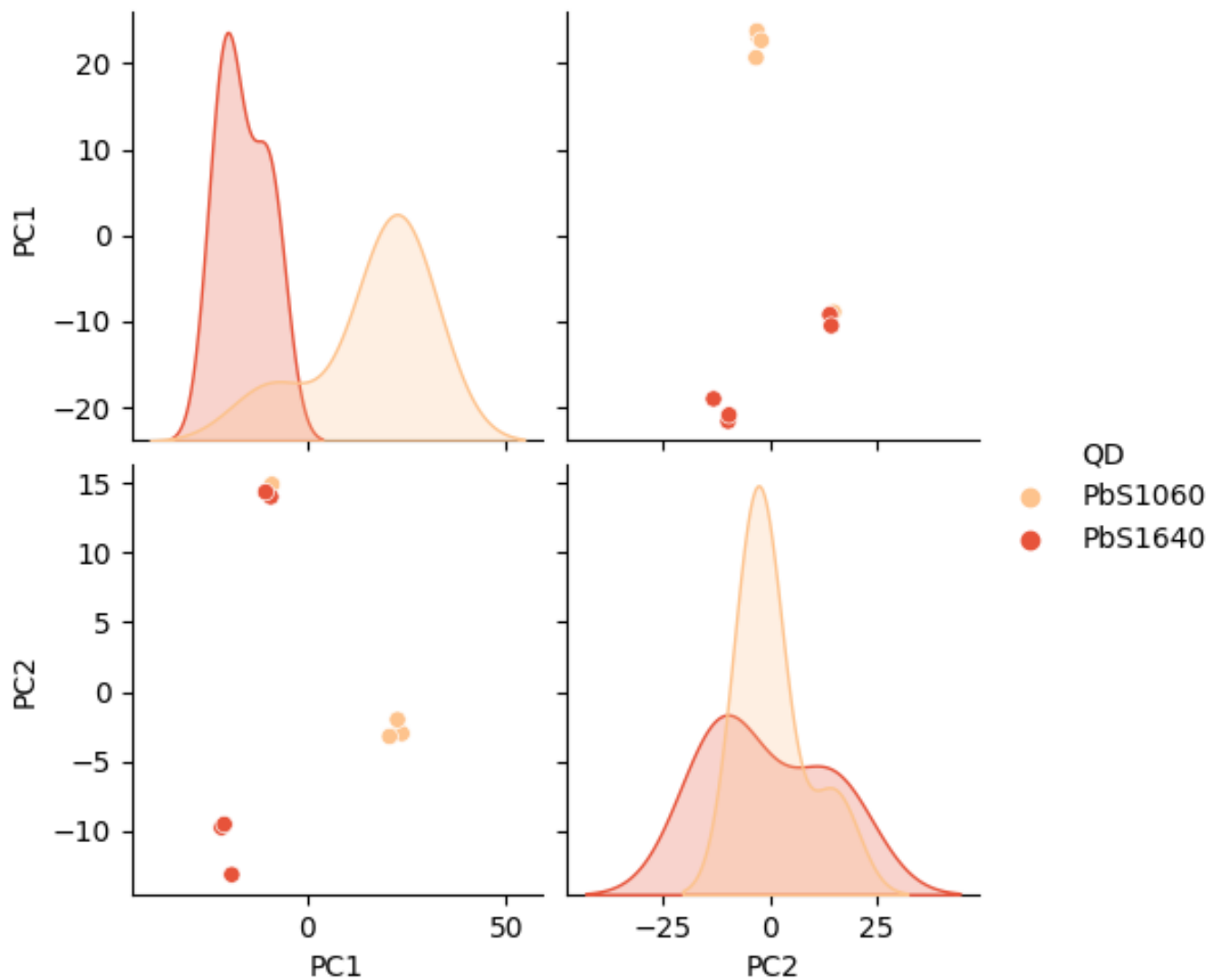
• Оригинал спектра

nfeat = nfeat1
test = test1



nfeat = nfeat2
test = test2

- Производная спектра



Сенсоры вместо спектрометров?

M. Fatih Adak, Peter Lieberzeit, Purim Jarujamrus, Nejat Yumusak, Classification of alcohols obtained by QCM sensors with different characteristics using ABC based neural network, Engineering Science and Technology, an International Journal, 2019, 23(3), 463-469.
<https://doi.org/10.1016/j.jestch.2019.06.011>

Задача

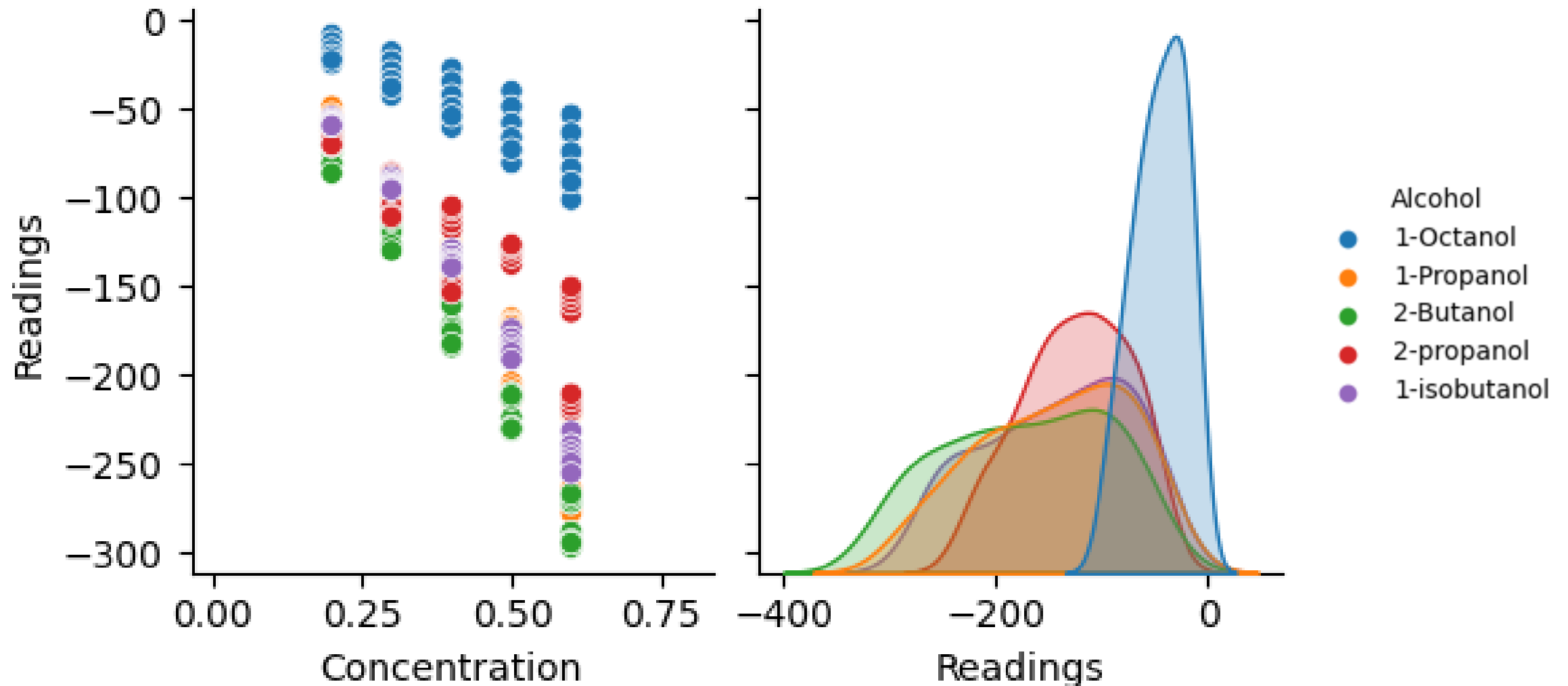
https://github.com/qrspeter/ml_alcohol

- распознавание известных химических веществ по показаниям сенсоров.

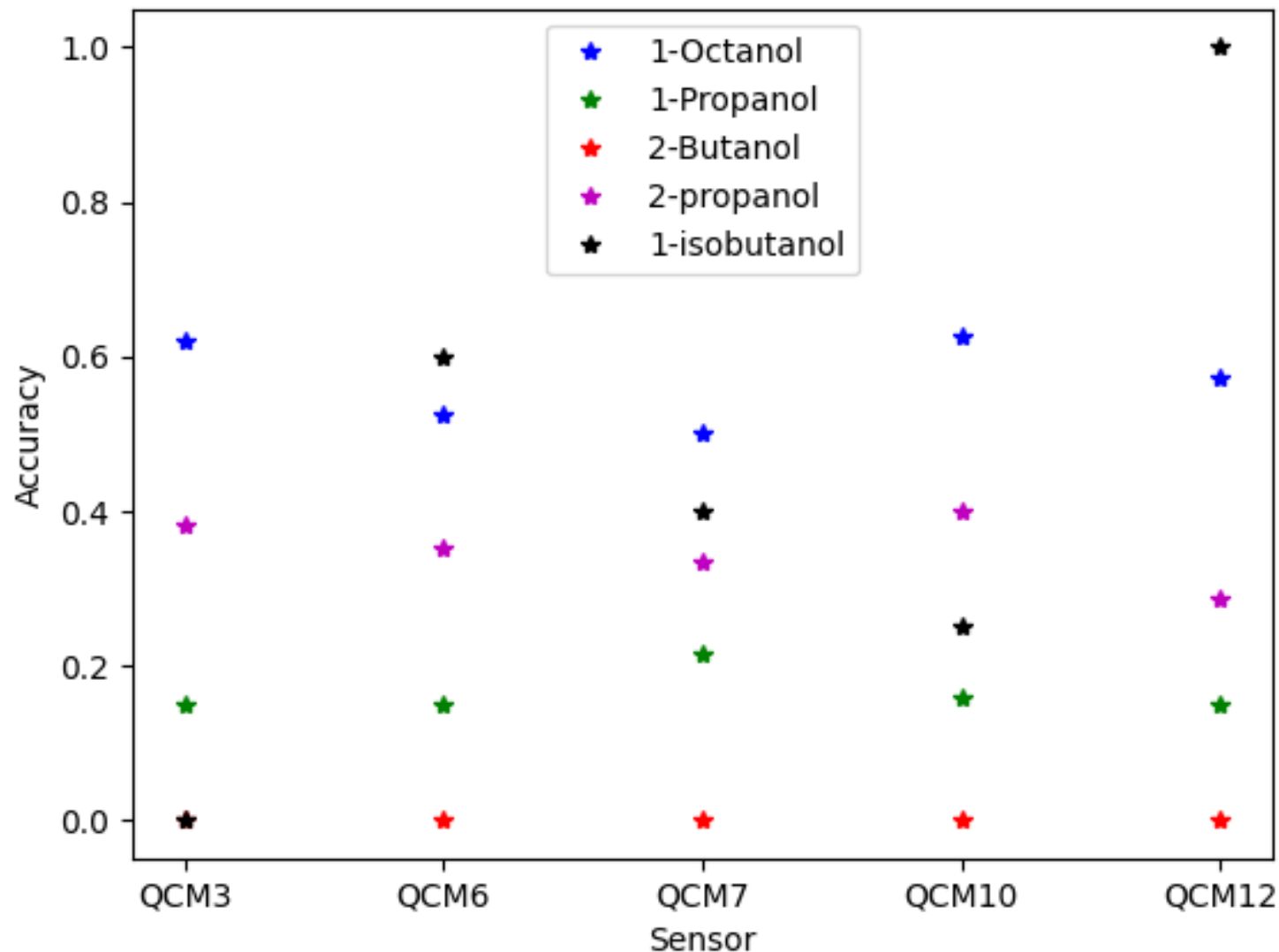
Исходные данные:

- Данные с пяти датчиков (5 файлов), различающихся соотношением двух каналов: QCM3, QCM6, QCM7, QCM10, QCM12.
- Пять типов спиртов: 1-октанол, 1-пропанол, 2-бутанол, 2-пропанол, 1-изобутанол
- Проба газа проходит через датчик в пяти различных концентрациях
- 10 измерений с каждым спиртом/концентрацией = 250 измерений/датчик

QCM12 – наилучшие результаты



Точность предсказания



"All the five of the QCM sensors gave successful results, but QCM12-constructed using only NP-was the most successful... The results of 300 different scenarios showed that different alcohols can be classified successfully by using ANN-ABC on the sensor data from QCM12.

ANN-ABC is able to classify the 5 gasses with a success rate of over 99%. "

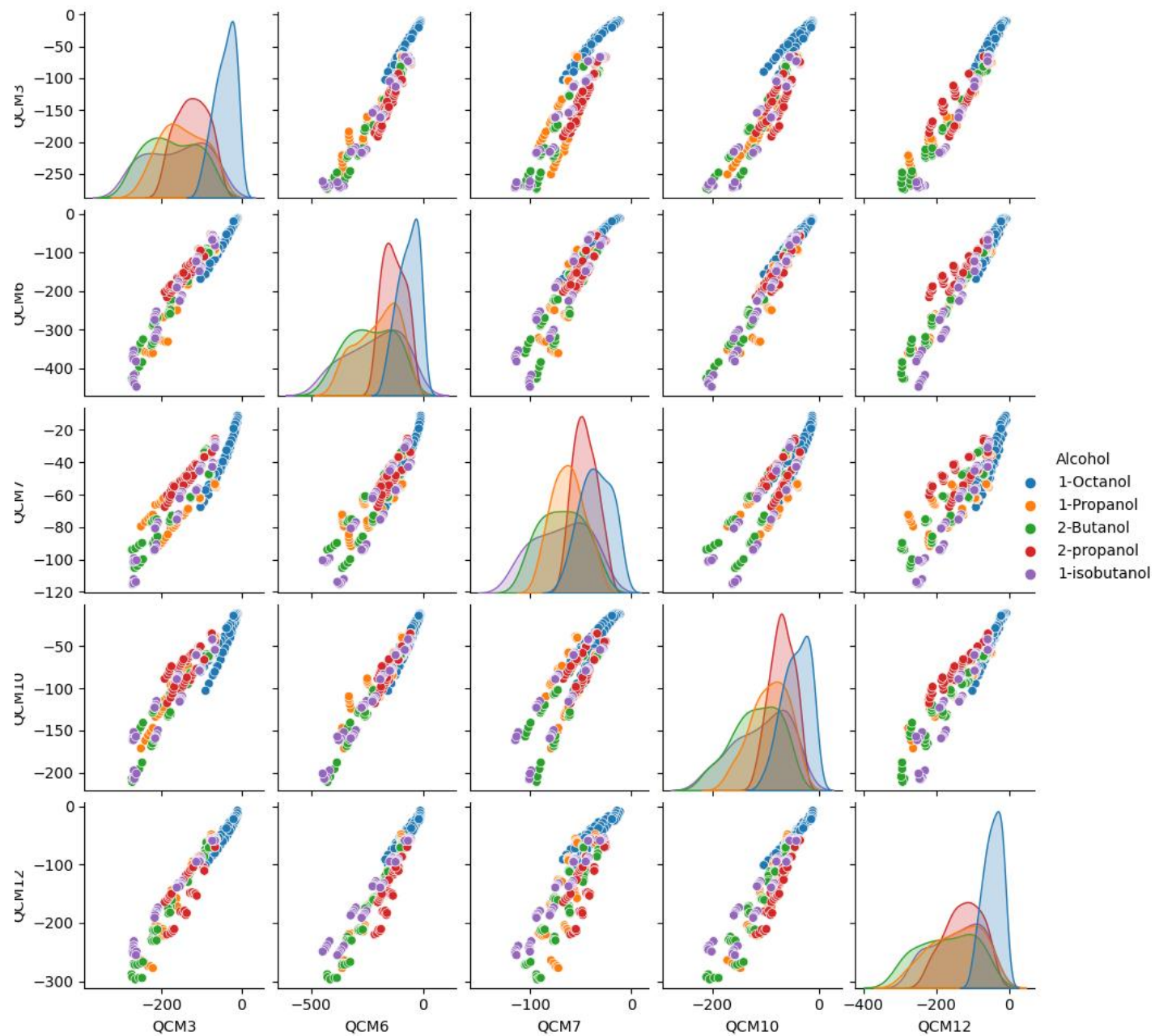
(<https://www.sciencedirect.com/science/article/pii/S2215098619303337>)

Какая пара сенсоров точнее предскажет?

(ml_alcohol_pairs.py)

- Объединить датафреймы датчиков в один
- Тогда на двух датчиках одновременно будут сравниваться равные концентрации, а не смесь вообще.

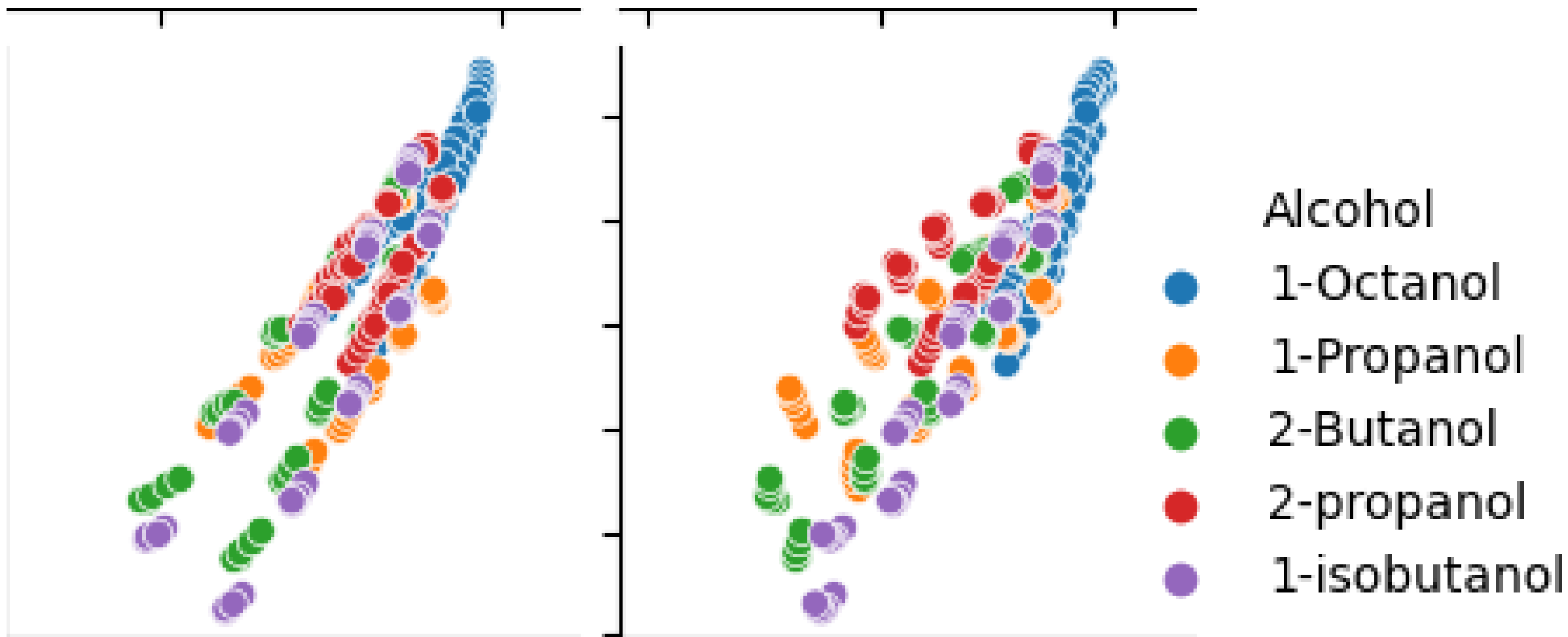
	Concentration	Alcohol	QCM3	QCM6	QCM7	QCM10	QCM12
0	0.2	1-Octanol	-10.06	-11.82	-11.23	-11.98	-9.40
1	0.2	1-Octanol	-10.62	-13.29	-14.21	-10.99	-7.95
2	0.3	1-Octanol	-14.43	-19.32	-18.71	-19.12	-21.44
3	0.3	1-Octanol	-18.31	-26.28	-22.65	-17.28	-17.46
4	0.4	1-Octanol	-24.64	-38.14	-27.32	-33.13	-34.39
..
245	0.4	1-isobutanol	-161.48	-191.28	-75.36	-89.26	-139.36
246	0.5	1-isobutanol	-215.52	-322.75	-80.85	-159.23	-186.98
247	0.5	1-isobutanol	-216.68	-274.53	-93.97	-122.95	-191.19
248	0.6	1-isobutanol	-262.01	-447.74	-100.50	-200.98	-248.98
249	0.6	1-isobutanol	-263.28	-381.94	-113.82	-157.18	-255.23



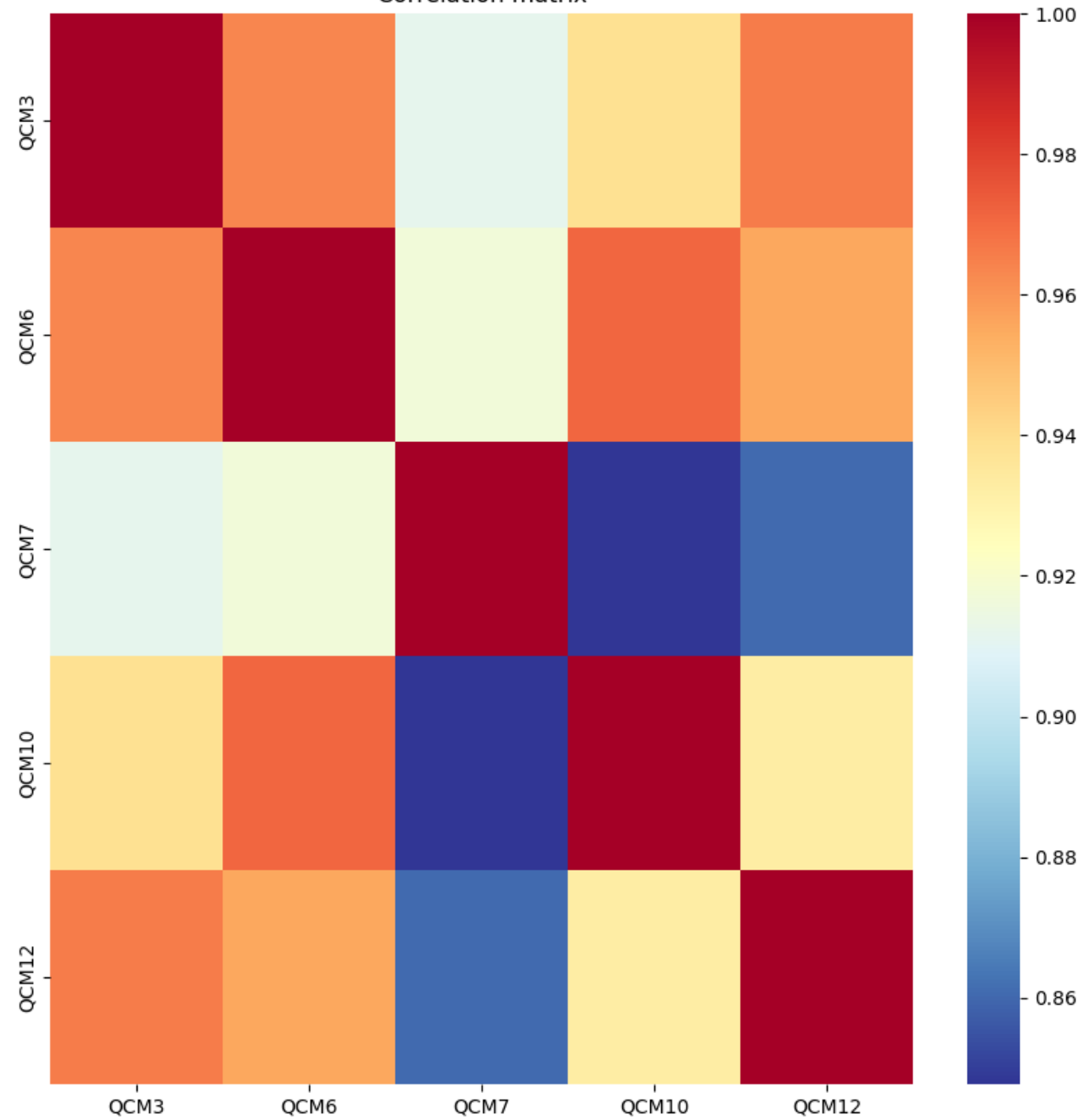
— Какая пара сенсоров точнее предскажет?

— Никакая

- Сильная корреляция датчиков



Correlation matrix



Пара датчиков 7 и 12

	QCM7	QCM12
0	-11.23	-9.40
1	-14.21	-7.95
2	-18.71	-21.44
3	-22.65	-17.46
4	-27.32	-34.39
..
245	-75.36	-139.36
246	-80.85	-186.98
247	-93.97	-191.19
248	-100.50	-248.98
249	-113.82	-255.23

	precision	recall	f1-score	support
1-Octanol	0.63	0.92	0.75	13
1-Propanol	0.00	0.00	0.00	7
1-isobutanol	0.57	0.44	0.50	9
2-Butanol	0.67	0.40	0.50	10
2-propanol	0.50	0.45	0.48	11
accuracy			0.50	50
macro avg	0.47	0.44	0.45	50
weighted avg	0.51	0.50	0.49	50

Developing of pH sensors based on carbon dots from o-phenylenediamine

M.D. Miruschenko^{1,*}, A.A. Vedernikova¹, E.V. Ushakova¹

¹ITMO University, 197101 Saint Petersburg, Russia

*Contacts:

ofussr@itmo.ru

m@miruschenko.ru

