Работа призера заключительного этапа

командной инженерной олимпиады школьников

Олимпиада Национальной технологической инициативы

Профиль «БОЛЬШИЕ ДАННЫЕ И МАШИННОЕ ОБУЧЕНИЕ»

Крапива Артемий Львович

Класс: 10

Город: Томск

Школа: МБОУ РКГ №2 г.Томска

Регион: Томская область

Уникальный номер участника: 201569

Команда на заключительном этапе: pyGeeks

Результаты заключительного этапа:

Индивидуальный тур				Командный тур			Итого:
Математ. нормиров	Коэф. 40%	Информат. нормиров..	Коэф. 40%	Задача 1 норм.	Задача 2 норм.	Коэф. команд. задач 60%	
0.9	36.6	0.2	6.7	0.6	0.8	80,7	123.9

Расшифровка индивидуальной части:

Индивидуальный тур													
Математика							Информатика						
2.2.1	2.2.2	2.2.3	2.2.4	2.2.5	Итого	Итого нормиров.	2.3.1	2.3.2	2.3.3	2.3.4	2.3.5	Итого	Итого норм.
0	2	10	20	0	32	0.9	0	5	0	0	0	5	0,2

Индивидуальная часть

Персональный лист участника с номером 201569:

Олимпиада НТИ

ФИО Кропива Артемий Львович

Город Томск

Школа № МБОУ РКГ №2 г. Томска

Математика

Командная инженерная олимпиада «Олимпиада НТИ»

Направление БД и МО

Предмет Математика

Номер участника 201569

32 MG

Baseline:

I) при $k = 10$ рассмотрим возможности "выигрыша" учеников:

для I: при любом наблюдении он говорит, что человек в норме.

Значит лучший результат будет, если в выборке будут все "нормальные". Но II ученик также правильно классифицирует нормальных $\Rightarrow$ у II и I будут одинаковые результаты при отсутствии неадекватных, т.к. если появляются неадекватные, повышается точность I ученика.

Ответ: нет.

для III: слабой стороной III ученика являются нормально мыслящие люди $\Rightarrow$ составим выборку только из неадекватных и ненормальных:

к-р, 9 ненормальных и 1 неадекватных.

В таком случае III ученик отгадает 100%, II – 10%, а I – 0%.

Ответ: да

для II: составив выборку из нормальных и неадекватных мы дадим II ученику выиграть, т.к. его точность будет 100%, а III будет ошибаться на норм., I – на неадекватных.

2) при $k = 30$ условия выигрыша I и II ученика не меняются: I никогда не выиграет, II – при выборке, состоящей из нормальных и неадекватных.

А вот III ученик теперь не сможет выиграть. Возьмём минимальное число норм. людей и посмотрим на точность учеников на пар-ре выборках:

Зачем.	I	II	III
10 неадек.	0%	100%	100%
10 ненорм.	0%	0%	100%
10 норм.	100%	100%	0%

, т.е. при лучших условиях для него, III ученик не сможет суммарно превзойти I ученика.

Ответ: нет.

Командная инженерная олимпиада «Олимпиада НТИ»

Направление 5А и МО

Предмет Математика

Номер участника 201569

Профиль компании: и МБС

1. Можно: т.е. квадрат

+

2. Можно:

Так, чтобы C – точка середина окружностей

+

3. Нет, нельзя. Мы не можем расположить вокруг одной точки равноудалённо точки других цветов, т.к. центральная точка не будет взаимно ближайшей.

–

+

Если вокруг ● мы хотим рисовать ○ настолько близко, чтобы периферия соседних ●, нам придётся это делать бесконечно.

Разрешающая поверхность и ГМТ

Ближайшая точка окружности B_K заданной точки A будет лежать на отрезке между A и центром окружности C. $\Rightarrow AC = AB + BC$

Окружность задана $(x+2)^2 + (y-1)^2 = 1 \Rightarrow C(-2;1)$, $R = 1 = BC$

$AB = AC - BC = AC - R$

$AB = \sqrt{(5-(-2))^2 + (8-1)^2} = 7\sqrt{2} - 1$

+

Информатика

Задача 2.3.1 Кусочно постоянные функции (подзадачи 1, 2, 3)

Решение не предоставлено

Задача 2.3.2 Метод ближайшего соседа (подзадачи 1, 2, 3)

Подзадача 1:
```
print(int((int(input())//2)))
```

Результат: 5 баллов

Подзадача 2:
Решение не предоставлено

Подзадача 3:
Решение не предоставлено

Задача 2.3.3 Монотонные классификаторы (подзадачи 1, 2, 3)

Решение не предоставлено

Задача 2.3.4 Различные методы (подзадачи 1, 2, 3)

Решение не предоставлено

Задача 2.3.5 Классификация и кластеризация (подзадачи 1, 2, 3)

Решение не предоставлено

Командная часть

Результаты были получены в рамках выступления команды: pyGeeks

Личный состав команды:

- Бакулин Вячеслав Викторович
- Крапива Артемий Львович
- Борисов Евгений Юрьевич

Фото команды:

Рис.1 – Фото участников команды

Решения командного этапа:

Код программы на языке Pyton, обеспечивающий решение задачи №1:

```
import pandas as pd
import numpy as np

X_train = pd.read_csv('C:/Users/slava/Py_Proj/New/in/X_train.csv',
encoding='windows-1251').drop(['HID','Encoded_FIO'],axis = 1)
X_test = pd.read_csv('C:/Users/slava/Py_Proj/New/in/X_test.csv',
encoding='windows-1251').drop(['HID','Encoded_FIO'],axis = 1)
y_train = pd.read_csv('C:/Users/slava/Py_Proj/New/in/y_train.csv', encoding='windows-1251',
names=['days'])
X_train = X_train.drop(list(set(X_train.columns)-set(X_test.columns)),axis = 1)
X_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144 entries, 0 to 143
Data columns (total 22 columns):
Год                                                                143 non-null float64
Пол                                                                144 non-null object
Дата рождения                                                      144 non-null object
Онкологический диагноз                                             144 non-null object
Дата постановки онкологического диагноза/ начала первичного лечения 132
non-null object
Первичный очаг (дата удаления)                                     92 non-null object
Дата развития МГМ                                                  128 non-null object
Активирующие мутации                                               47 non-null object
Дата проведения ОВГМ                                               126 non-null object
Дата операции на ГМ                                                132 non-null object
Число радиохирургий (РХ) на аппарате Гамма Ноже                    144 non-null
float64
Дата 1й РХ на ГН                                                   144 non-null object
Индекс Карновского на момент 1 РХ                                  144 non-null float64
Число очагов в ГМ подвергнутых 1й радиохирургии                    144 non-null
float64
Суммарный объем очагов, подвергнутых 1й радиохирургии              144
non-null float64
Объем макс очага, подвергнутого 1й радиохирургии                   144 non-null
object
Экстракраниальные метастазы на момент проведения радиохирургии /
гиплфракционирования    92 non-null object
Лекарственное лечение на момент проведения радиохирургии/гтпофракционирования
97 non-null object
Локальный рецидив после радиохирургии / гипофракционирования       104
non-null object
Дистантные метастазы после радиохирургии / гипофракционирования    104
non-null object
Интракраниальная прогрессия (локальные рецидивы +дистантные метастазы)
104 non-null object
Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования
68 non-null object
dtypes: float64(5), object(17)
memory usage: 15.2+ KB

# Функции для нормализации даты

def func1(x):
    try:
        x = x.split('.')
        x[2] = '20' + x[2]
        x = '.'.join(x)
        return pd.to_datetime(x)
    except:
        return np.nan
def func2(x):
    x = x.split('.')
```

```
    x[2] = '19' + x[2]
    x = '.'.join(x)
    return pd.to_datetime(x)

def aa(n):
    try:
        return int(n.days)
    except:
        return 0

# Пол
def pol(x):
    if x == 'М':
        x = '1'
    else:
        x = '0'
    return x

# Мутации и метастазы
def yesno(x):
    if x == 'есть':
        return 1
    else:
        return 0

#Переход от символьного диагноза к цифровому по схеме: чем больше число
#тем легче поддаётся лечению болезнь

def dischange(x):
    if x == 'КРР' or x == 'НМРЛ' or x == 'Меланома':
        x = 0
    if x == 'РМЖ':
        x = 4
    if x == 'РП':
        x = 5
    return x

#Для индекса карновского
def check(x, i):
    x = int(x)
    if x <= 30:
        x = 4
    elif x <= 50 and x > 30:
        x = 3
    elif x <=70 and x > 50:
        x = 2
    elif x > 70 and x <= 90:
        x = 1
    elif x > 90:
        x = 0
    if x == i:
```

```
        return 1
    else:
        return 0



def b(x):
    if x == 'химиотерапия' or x == 'Химиотерапия':
        return 2
    elif x == 'таргетная терапия' or x == 'Таргетная терапия':
        return 4
    else:
        return 0

def c(x):
    if x == 'ОВГМ':
        return 2
    elif x == 'РХ' or x == 'гипофракционирование':
        return 4
    else:
        return 0

def d(x):
    if (x == 'Оп'):
        return 1
    else:
        return 0

def e(x, states):
    if x in states:
        return 1
    else:
        return 0

def a(x):
    if x == 'есть':
        return 1
    else:
        return 0

# все прменяем
X_train['Пол'] = X_train['Пол'].apply(pol)
#переводим в datetime
X_train['Дата рождения'] = X_train['Дата рождения'].apply(func2)
columns1 = ['Дата постановки онкологического диагноза/ начала первичного
лечения','Первичный очаг (дата удаления)', 'Дата проведения ОВГМ', 'Дистантные
метастазы после радиохирургии / гипофракционирования','Дата операции на ГМ', 'Дата 1й
РХ на ГН', 'Дата развития МГМ ', 'Локальный рецидив после радиохирургии /
гипофракционирования']
for col in columns1:
    X_train[col] = X_train[col].apply(func1)
```

```
#отсчитываем дни
columns2 = ['Дата рождения', 'Дата постановки онкологического диагноза/ начала первичного лечения','Первичный очаг (дата удаления)', 'Дата проведения ОВГМ', 'Дистантные метастазы после радиохирургии / гипофракционирования',  'Дата операции на ГМ', 'Дата развития МГМ ',  'Локальный рецидив после радиохирургии / гипофракционирования']
for col in columns2:
    X_train[col] = X_train[col] - X_train['Дата 1й РХ на ГН']
    X_train[col] = X_train[col].apply(aa)
X_train = X_train.drop(['Дата 1й РХ на ГН'], axis=1)
#Исправляем кривой объем
X_train['Объем макс очага, подвергнутого 1й радиохирургии'] = X_train['Объем макс очага, подвергнутого 1й радиохирургии'].str.replace(',','.').astype(float)
          #Индекс Карновского
X_train['Онкологический диагноз'] = X_train['Онкологический диагноз'].apply(dischange)
for i in [0, 1, 2, 3, 4]:
    X_train[str('ИндКар ' + str(i))] = X_train['Индекс Карновского на момент 1 РХ'].apply(lambda x: check(x, i))
X_train['Химиотерапия после ИК'] = X_train['Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования'].apply(b)
X_train['Радиотерапия после ИК'] = X_train['Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования'].apply(c)
X_train['Операция после ИК'] = X_train['Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования'].apply(d)
X_train['Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования'] = X_train['Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования'].apply(lambda x: e(x, ['ОВГМ', 'Оп', 'РХ', 'таргетная терапия', 'химиотерапия']))
X_train['Интракраниальная прогрессия (локальные рецидивы +дистантные метастазы)'] = X_train['Интракраниальная прогрессия (локальные рецидивы +дистантные метастазы)'].apply(lambda x: e(x, ['ДМ', 'ЛР', 'ЛР+ДМ']))
X_train['Активирующие мутации'] = X_train['Активирующие мутации'].apply(a)
X_train['Экстракраниальные метастазы на момент проведения радиохирургии / гиплфракционирования'] = X_train['Экстракраниальные метастазы на момент проведения радиохирургии / гиплфракционирования'].apply(a)
X_train['Лекарственное лечение на момент проведения радиохирургии/гтпофракционирования'] = X_train['Лекарственное лечение на момент проведения радиохирургии/гтпофракционирования'].apply(b)
for i in range(len(X_train)):
    if X_train['Дата проведения ОВГМ'].iloc[i] > 0.0:
        X_train['Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования'].iloc[i] = 1
        if X_train['Радиотерапия после ИК'].iloc[i] == 0:
            X_train['Радиотерапия после ИК'].iloc[i] = 2
    elif X_train['Дата операции на ГМ'].iloc[i] > 0.0:
        X_train['Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования'].iloc[i] = 1
        X_train['Операция после ИК'].iloc[i] = 1

# все прменяем
X_test['Пол'] = X_test['Пол'].apply(pol)
```

```
#переводим в datetime
X_test['Дата рождения'] = X_test['Дата рождения'].apply(func2)
columns1 = ['Дата постановки онкологического диагноза/ начала первичного
лечения','Первичный очаг (дата удаления)', 'Дата проведения ОВГМ', 'Дистантные
метастазы после радиохирургии / гипофракционирования','Дата операции на ГМ', 'Дата 1й
РХ на ГН', 'Дата развития МГМ ', 'Локальный рецидив после радиохирургии /
гипофракционирования']
for col in columns1:
    X_test[col] = X_test[col].apply(func1)
#отсчитываем дни
columns2 = ['Дата рождения', 'Дата постановки онкологического диагноза/ начала
первичного лечения','Первичный очаг (дата удаления)', 'Дата проведения ОВГМ',
'Дистантные метастазы после радиохирургии / гипофракционирования',  'Дата операции на
ГМ', 'Дата развития МГМ ',  'Локальный рецидив после радиохирургии /
гипофракционирования']
for col in columns2:
    X_test[col] = X_test[col] - X_test['Дата 1й РХ на ГН']
    X_test[col] = X_test[col].apply(aa)
X_test = X_test.drop(['Дата 1й РХ на ГН'], axis=1)
#Исправляем кривой объем
X_test['Объем макс очага, подвергнутого 1й радиохирургии'] = X_test['Объем макс очага,
подвергнутого 1й радиохирургии'].str.replace(',','.').astype(float)
          #Индекс Карновского
X_test['Онкологический диагноз'] = X_test['Онкологический диагноз'].apply(dischange)
for i in [0, 1, 2, 3, 4]:
    X_test[str('ИндКар ' + str(i))] = X_test['Индекс Карновского на момент 1
РХ'].apply(lambda x: check(x, i))
X_test['Химиотерапия после ИК'] = X_test['Лечение ИК рецидива после после 1-ой
радиохирургии / гипофракционирования'].apply(b)
X_test['Радиотерапия после ИК'] = X_test['Лечение ИК рецидива после после 1-ой
радиохирургии / гипофракционирования'].apply(c)
X_test['Операция после ИК'] = X_test['Лечение ИК рецидива после после 1-ой
радиохирургии / гипофракционирования'].apply(d)
X_test['Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования']
= X_test['Лечение ИК рецидива после после 1-ой радиохирургии /
гипофракционирования'].apply(lambda x: e(x, ['ОВГМ', 'Оп', 'РХ', 'таргетная терапия',
'химиотерапия']))
X_test['Интракраниальная прогрессия (локальные рецидивы +дистантные метастазы)'] =
X_test['Интракраниальная прогрессия (локальные рецидивы +дистантные
метастазы)'].apply(lambda x: e(x, ['ДМ', 'ЛР', 'ЛР+ДМ']))
X_test['Активирующие мутации'] = X_test['Активирующие мутации'].apply(a)
X_test['Экстракраниальные метастазы на момент проведения радиохирургии /
гиплфракционирования'] = X_test['Экстракраниальные метастазы на момент проведения
радиохирургии / гиплфракционирования'].apply(a)
X_test['Лекарственное лечение на момент проведения
радиохирургии/гтпофракционирования'] = X_test['Лекарственное лечение на момент
проведения радиохирургии/гтпофракционирования'].apply(b)
for i in range(len(X_test)):
    if X_test['Дата проведения ОВГМ'].iloc[i] > 0.0:
        X_test['Лечение ИК рецидива после после 1-ой радиохирургии /
гипофракционирования'].iloc[i] = 1
```

```
        if X_test['Радиотерапия после ИК'].iloc[i] == 0:
            X_test['Радиотерапия после ИК'].iloc[i] = 2
    elif X_test['Дата операции на ГМ'].iloc[i] > 0.0:
        X_test['Лечение ИК рецидива после после 1-ой радиохирургии /
гипофракционирования'].iloc[i] = 1
        X_test['Операция после ИК'].iloc[i] = 1
```

```
C:\Users\slava\Anaconda3\lib\site-packages\pandas\core\indexing.py:194:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:
http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
```

```
set(X_test['Лечение ИК рецидива после после 1-ой радиохирургии /
гипофракционирования']) - set(X_train['Лечение ИК рецидива после после 1-ой
радиохирургии / гипофракционирования'])
```

```
{'гипофракционирование'}
X_train['Радиотерапия после ИК'] = X_train['Лечение ИК рецидива после после 1-ой
радиохирургии / гипофракционирования'].apply(c)
```

```
X_train['Год'] = X_train['Год'].fillna(X_train['Год'].median())
```

X_train

	Год	Пол	Дата рождения	Онкологический диагноз	Дата постановки онкологического диагноза/ начала первичного леч	Первичный очаг (дата удаления)	Дата развития МГМ	Активирующие мутации	Дата проведения ОВГМ	Дата операции на ГМ	...	Интракраниальная прогрессия (локальные рецидивы +дистантные метастазы)	Лечение ИК рецидива после после 1-ой радиохирургии / гипофракционирования	ИндКар0	ИндКар1	ИндКар2	ИндКар3	ИндКар4	Химиотерапия после ИК	Радиотерапия после ИК	Операция после ИК

					ения															
0	10.0	1	-223663	0	-415	0	-409	0	0	0	...	0	0	0	0	1	0	0	0	0	0
1	13.0	0	-227735	4	-2010	-2009	-197	1	0	1	...	0	1	0	1	0	0	0	0	0	1
2	13.0	0	-238851	0	-614	0	-9	0	0	0	...	1	0	0	1	0	0	0	0	0	0
3	12.0	1	-281332	0	-117	0	-116	0	0	0	...	1	0	0	1	0	0	0	0	0	0
4	15.0	0	-242273	0	-439	-439	-83	1	0	0	...	0	0	0	1	0	0	0	0	0	0
5	16.0	0	-229770	0	-269	-269	-277	1	-276	0	...	0	0	0	0	1	0	0	0	0	0

6	11.0	1	-195144	0	0	0	0	0	0	0	...	1	1	0	0	1	0	0	0	0	0
7	16.0	1	-158873	0	-4375	-4375	-1	1	0	0	...	1	0	0	1	0	0	0	0	0	0
8	9.0	1	-257774	0	-2281	-2543	-360	0	0	0	...	0	0	0	0	1	0	0	0	0	0
9	8.0	0	-141771	0	-1010	-1010	-31	0	0	0	...	1	0	0	1	0	0	0	0	0	0
10	9.0	0	-264403	0	121	0	-213	0	0	0	...	0	0	0	0	1	0	0	0	0	0
11	10.0	1	-199559	0	208	208	-1	0	0	0	...	1	1	0	1	0	0	0	0	0	0
12	13	0	-27	4	-1042	-1042	-33	0	0	0	...	0	0	0	1	0	0	0	0	0	0

	.0		975				8													
13	12.0	1	-296664	0	-752	-752	-202	1	0	0	...	1	0	0	1	0	0	0	0	0
14	10.0	1	-191331	0	-162	0	-162	0	0	0	...	1	1	0	1	0	0	0	0	0
15	10.0	1	-255515	0	-2994	-2994	2483	0	0	0	...	0	0	0	1	0	0	0	0	0
16	11.0	0	-193380	4	-3699	-3688	227	0	0	0	...	0	0	0	0	1	0	0	0	0
17	15.0	1	-186613	5	-559	-559	-13	0	0	0	...	1	0	0	0	1	0	0	0	0
18	8.0	1	-204444	0	-221	0	-222	0	0	0	...	1	1	0	1	0	0	0	0	0

19	13.0	1	-196441	0	-771	-771	-56	0	0	0	...	1	1	1	0	0	0	0	0	0	0
20	11.0	0	-200041	0	-1166	-1166	-70	0	0	0	...	1	1	0	0	1	0	0	0	0	0
21	14.0	0	-197447	4	-1309	0	-128	0	0	0	...	0	0	0	0	0	1	0	0	0	0
22	12.0	0	-190555	0	-1338	-1338	-245	0	0	0	...	1	0	0	0	1	0	0	0	0	0
23	9.0	1	-191168	5	-736	0	-376	0	0	0	...	0	0	0	0	1	0	0	0	0	0
24	12.0	1	-180003	0	-946	-946	-214	0	0	0	...	1	0	0	1	0	0	0	0	0	0
25	9.0	0	-18	0	0	0	0	0	0	0	-39	...	1	1	0	0	1	0	0	0	0	0

			872																		
26	110.0	1	-150087	0	-235	-235	-189	0	0	171	...	1	1	0	0	1	0	0	0	0	1
27	120.0	1	-229991	0	-300	0	-2995	0	0	0	...	0	0	0	1	0	0	0	0	0	0
28	130.0	0	-275568	0	-2049	-2049	-599	1	0	0	...	0	0	0	0	1	0	0	0	0	0
29	120.0	1	-199971	0	0	0	0	0	0	0	-492	...	1	0	0	1	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
305	140.0	1	-2210109	0	-251	-251	-250	0	0	0	...	0	0	0	0	1	0	0	0	0	0
306	11	0	-1	4	-1533	0	-8	1	-140	29	..	0	1	0	0	1	0	0	0	0	1

	.0		8486				02			2	.										
307	140.0	1	-157701	0	-686	-686	-3111	1	0	0	...	1	1	0	1	0	0	0	0	0	0
308	140.0	0	-223314	0	-1898	0	-90	0	0	-90	...	0	0	0	1	0	0	0	0	0	0
309	110.0	0	-2111192	4	-814	-814	-68	0	0	0	...	0	0	0	0	1	0	0	0	0	0
310	7.0	1	-2803035	5	-874	-874	0	0	0	0	...	1	1	0	0	1	0	0	4	0	0
311	150.0	1	-198224	0	-1858	0	-60	0	0	-16	...	0	0	0	0	1	0	0	0	0	0
312	130.0	1	-21881	0	0	0	0	0	-286	-272	...	1	0	0	1	0	0	0	0	0	0

			0																	
313	15.0	0	-239924	4	-2348	-2258	-165	0	0	0	...	0	0	0	1	0	0	0	0	0	0
314	80.0	1	-2311118	5	-600	-600	-1400	0	0	0	...	0	0	0	1	0	0	0	0	0	0
315	140.0	1	-1767671	0	-1190	0	-91	0	0	0	...	0	0	0	0	1	0	0	0	0	0
316	70.0	0	-1895858	0	-1001	-1001	-273	0	-259	0	...	0	0	0	1	0	0	0	0	0	0
317	130.0	1	-2662213	0	-1791	-1791	-334	1	0	0	...	1	0	0	0	1	0	0	0	0	0
318	150.0	1	-1632828	0	-474	-474	-72	0	0	0	...	0	0	0	1	0	0	0	0	0	0

319	50.0	0	-165115	0	0	0	0	0	0	0	...	0	0	0	1	0	0	0	0	0
320	100.0	0	-245333	4	-1070	0	-157	1	4	0	...	0	1	0	0	1	0	0	0	0
321	110.0	1	-2002251	5	-376	0	-376	0	0	0	...	0	0	0	1	0	0	0	0	0
322	120.0	0	-208889	4	-2826	0	-100000	0	-1000	0	...	0	0	0	1	0	0	0	0	0
323	100.0	1	-2228778	5	75	75	-21	0	0	0	...	0	0	0	0	1	0	0	0	0
324	130.0	1	-209334	0	-1100	0	-3700	0	0	-35	...	0	0	0	0	1	0	0	0	0
325	14	0	-22	4	-724	0	-16	0	0	0	...	0	0	0	0	1	0	0	0	0

	.0		337																		
326	160.0	0	-185007	4	-4991	-4985	-969	1	-494	0	...	1	1	0	1	0	0	0	0	0	0
327	70.0	1	-2002226	5	-1194	-1194	-92	0	0	0	...	0	0	0	0	1	0	0	0	0	0
328	150.0	0	-1927272	0	-2169	-2169	-152	0	0	-146	...	0	0	0	1	0	0	0	0	0	0
329	150.0	0	-2193935	0	-1371	-1371	-278	0	0	-277	...	0	0	0	1	0	0	0	0	0	0
330	150.0	0	-2404042	0	-387	-387	-47	0	0	0	...	0	0	0	1	0	0	0	0	0	0
331	150.0	1	-2655353	0	-2299	-2299	-12	0	0	0	...	1	1	0	1	0	0	0	0	0	0

332	140.0	1	-174000	0	-218	0	-67	1	0	0	...	1	1	0	1	0	0	0	0	0
333	150.0	1	-2140409	0	-63	0	-60	0	0	0	...	0	0	0	0	1	0	0	0	0
334	160.0	0	-2344477	0	-771	-771	-34	1	0	0	...	1	1	0	1	0	0	0	0	0

335 rows × 29 columns

```
for i in range(3000):
    X,x,Y,y = train_test_split(X_train_, y_train.days,test_size=0.10)

    from sklearn.linear_model import Lasso
    sgd = SGDRegressor(loss='huber', max_iter=300,verbose = 0)
    sgd.fit(X, Y)
    preds = model.predict(x)
    r2 = r2_score(preds,y)
    if r2>0.74:
        preds = model.predict(X_test_)
        pd.DataFrame(preds).to_csv('preds/preds_SGD_'+str(r2)+'.csv',index = False)
        print(r2)

from sklearn.linear_model import ElasticNet
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split

for i in range(10, 100):
    for j in range(0, 11):
        X,x,Y,y = train_test_split(X_train, y_train.days,test_size=0.1)
        sgd = ElasticNet(alpha=(i/100), l1_ratio = (j/10), random_state=0)
        sgd.fit(X, Y)
        preds = sgd.predict(x)
        r2 = r2_score(preds,y)
        if r2>0.8:
            print('! '+ str(r2) + str(i) + ' ' + str(j) + ' ' + str(p))
```

```
            preds = sgd.predict(X_test)
            pd.DataFrame(preds).apply(abs).to_csv('./preds_EN.csv',index = False)
        if r2>0.6:
            print(str(r2) + ' ' + str(i) + ' ' + str(j) + ' ' + str(p))
```

```
C:\Users\slava\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.py:491:
ConvergenceWarning: Objective did not converge. You might want to increase the number of
iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)

0.6906839041550573 20 1 148
0.6253892104605163 29 6 148
0.6203960662261474 37 10 148
0.7030577332293673 40 6 148
0.6664551566165801 46 1 148
0.6385402593573815 58 8 148
```

```
X,x,Y,y = train_test_split(X_train, y_train.days,test_size=0.1)

sgd = ElasticNet(alpha=(i/100), l1_ratio = (j/10), random_state=p)
sgd.fit(X, Y)
preds = sgd.predict(x)
r2 = r2_score(preds,y)
if r2>0.76:
    print('! '+ str(r2) + str(i) + ' ' + str(j) + ' ' + str(p))
    preds = sgd.predict(X_test)
    pd.DataFrame(preds).to_csv('./preds_EN.csv',index = False)
if r2>0.6:
    print(str(r2) + ' ' + str(i) + ' ' + str(j) + ' ' + str(p))
```

```
C:\Users\slava\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.py:491:
ConvergenceWarning: Objective did not converge. You might want to increase the number of
iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
```

Код программы на языке Pyton, обеспечивающий решение задачи №2:

```
import pandas as pd
import numpy as np
import matplotlib
%matplotlib inline
import seaborn as sns
import matplotlib.pyplot as plt
pd.options.display.max_columns

20
x_test = pd.read_csv('./X_test2.csv', encoding='windows-1251').drop('Encoded_FIO',axis = 1)
x_train = pd.read_csv('./x_train2.csv', encoding='windows-1251').drop('Encoded_FIO',axis = 1)
```

```
y_train = pd.read_csv('./y_train2.csv', encoding='windows-1251', names = ['Comeback'])
```

Work with DATA

```
isnull = x_train.isnull()


def a(x):
    if x == 'Ж':
        return 1
    elif x == 'М':
        return 0

def d(x, i):
    if x==i:
        return 1
    else:
        return 0

def r(x):
    x = str(x)
    if x != 'н/д' and x != 'NaN':
        x = x.replace(',', '.')
        return float(x)
    else:
        return 0.0

def rint(x):
    try:
        x = int(x)
        if x <= 24:
            return x
        else:
            return 0
    except:
        return 0

def do(x):
    x = str(x)
    x = x.replace(',', '.')
    x = float(x)
    if x > 98.0 or x < 40.0:
        return 0
    else:
        return x



col = ['MV12GY_1', 'MV10GY_1','NV12GY_1', 'NV10GY_1']
```

```
x_test['V1'] = x_test['V1'].apply(r)
x_train['V1'] = x_train['V1'].apply(r)
for x in x_test[col]:
    x_test[x] = x_test[x].apply(str)
    x_train[x] = x_train[x].apply(str)
    x_test[x] = x_test[x].apply(r)
    x_train[x] = x_train[x].apply(r)

x_test.PD1 = x_test.PD1.apply(rint)
x_train.PD1 = x_train.PD1.apply(rint)
#x_test['target'] = y_train.Comeback

x_test['PI1'] = x_test['PI1'].apply(do)
x_train['PI1'] = x_train['PI1'].apply(do)

# x_train['PD1/PI1'] = x_train['PD1']/x_train['PI1']
# x_test['PD1/PI1'] = x_test['PD1']/x_test['PI1']

# x_train['PD1/PI1'][x_train['PD1/PI1']>1]=0
# x_test['PD1/PI1'][x_test['PD1/PI1']>1]=0


x_train['SEX'] = x_train['SEX'].apply(a)
x_test['SEX'] = x_test['SEX'].apply(a)


# col = x_train.columns
# Core_col = ['V1', 'MV10GY_1', 'PI1', 'MV12GY_1']
# for i in col:
#     for j in col:
#         if x_train[i].dtype!='object' and x_train[j].dtype!='object':
#             x_train[i+'_'+j]= x_train[j]*x_train[i]
#             x_test[i+'_'+j]= x_test[j]*x_test[i]

x_train = x_train.drop(['Localization', 'CANCER_HISTOLOGY', 'RS1'], axis=1)
x_test = x_test.drop(['Localization', 'CANCER_HISTOLOGY', 'RS1'], axis=1)

for i in reversed(range(len(x_train))):
    if isnull.as_matrix()[i].sum()!=0:
        y_train = y_train.drop(i,axis = 0)
x_train = x_train.dropna()

from imblearn.over_sampling import ADASYN
from sklearn.model_selection import train_test_split

X,Y = x_train,y_train
# X,x,Y,y = train_test_split(x_train, y_train, train_size = 0.95)
pca = PCA(n_components=2)
X_vis = pca.fit_transform(X)
```

```
ada = ADASYN()
X_resampled, y_resampled = ada.fit_sample(X, Y)
X_res_vis = pca.transform(X_resampled)
X_resampled = pd.DataFrame(X_resampled,columns = x_test.columns)
```

```
C:\Program Files\Anaconda3\lib\site-packages\sklearn\utils\validation.py:578:
DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please
change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
# from collections import Counter
# from sklearn.datasets import make_classification
# from imblearn.over_sampling import ADASYN

# X,Y = x_train,y_train

# ada = ADASYN(random_state=42)
# X_resampled, y_resampled = ada.fit_sample(X, Y)

# cols =x_train.columns
# for i in cols:
#     for j in cols:
#         X_resampled[i+'_'+j]= X_resampled[j]*X_resampled[i]
#         x_test[i+'_'+j]= x_test[j]*x_test[i]


for col in x_train.columns:
    maximum = max(max(X_resampled[col]),max(x_test[col]))
    minimum = min(min(X_resampled[col]),min(x_test[col]))
    if maximum == 0:
        continue
    X_resampled[col] = ((X_resampled[col]-minimum)/(maximum-minimum))
    x_test[col] = ((x_test[col]-minimum)/(maximum-minimum))


x_test = x_test.fillna(1)

pd.DataFrame(model.predict(x_test.as_matrix()))[0].to_csv('preds.csv',index = False)

import numpy as np
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense, Dropout
from sklearn.cross_validation import train_test_split



count = len(X_resampled)
Xtrain = X_resampled
dummy = y_resampled
```

```
dumm_y=np.array(range(count*2)).reshape(count,2)
for i in range(count):
   dummy_y[i,0] = 1*(dummy[i]==0)
   dummy_y[i,1] = 1*(dummy[i]==1)


def baseline_model():
   # create model
   model = Sequential()
   model.add(Dense(12, input_dim=Xtrain.shape[1], init='normal', activation='relu'))
   model.add(Dropout(0.2))
   model.add(Dense(10, init='normal', activation='relu'))
   model.add(Dropout(0.2))
   model.add(Dense(8, init='normal', activation='relu'))
   model.add(Dropout(0.2))
   model.add(Dense(2, init='normal', activation='softmax'))

   model.compile(loss='categorical_crossentropy', optimizer='Adam', metrics=['accuracy'])
#logloss
   return model

model=baseline_model()

X_train, X_val, y_train, y_val = train_test_split(Xtrain, dummy_y, test_size=0.1,
random_state=42)
cl_weight = pd.Series(dummy).value_counts()

fit= model.fit(X_train,y_train,
                  nb_epoch=150,
                  verbose=True,
)

scores_val = model.predict(X_val)

preds = pd.DataFrame(model.predict(X_val.as_matrix()))
f1_score(preds[1].round().as_matrix(),y_val.reshape(2, len(y_val))[1])

C:\Program Files\Anaconda3\lib\site-packages\ipykernel\__main__.py:41: UserWarning: Update
your `Dense` call to the Keras 2 API: `Dense(12, kernel_initializer="normal", input_dim=8,
activation="relu")`
C:\Program Files\Anaconda3\lib\site-packages\ipykernel\__main__.py:43: UserWarning: Update
your `Dense` call to the Keras 2 API: `Dense(24, kernel_initializer="normal", activation="relu")`
C:\Program Files\Anaconda3\lib\site-packages\ipykernel\__main__.py:45: UserWarning: Update
your `Dense` call to the Keras 2 API: `Dense(24, kernel_initializer="normal", activation="relu")`
C:\Program Files\Anaconda3\lib\site-packages\ipykernel\__main__.py:47: UserWarning: Update
your `Dense` call to the Keras 2 API: `Dense(24, kernel_initializer="normal", activation="relu")`
C:\Program Files\Anaconda3\lib\site-packages\ipykernel\__main__.py:49: UserWarning: Update
your `Dense` call to the Keras 2 API: `Dense(2, kernel_initializer="normal",
activation="softmax")`
C:\Program Files\Anaconda3\lib\site-packages\keras\models.py:944: UserWarning: The
```

```
`nb_epoch` argument in `fit` has been renamed `epochs`.
  warnings.warn('The `nb_epoch` argument in `fit` '

Epoch 1/150
5977/5977 [==============================] - 1s 159us/step - loss: 0.6932 - acc:
0.5054
Epoch 2/150
5977/5977 [==============================] - 0s 39us/step - loss: 0.6835 - acc: 0.5535
Epoch 3/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.6475 - acc: 0.6468
Epoch 4/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.6290 - acc: 0.6637
Epoch 5/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.6226 - acc: 0.6786
Epoch 6/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.6140 - acc: 0.6789
Epoch 7/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.6097 - acc: 0.6906
Epoch 8/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.6085 - acc: 0.6938
Epoch 9/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.6017 - acc: 0.6955
Epoch 10/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.6084 - acc: 0.6856
Epoch 11/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.6019 - acc: 0.6960
Epoch 12/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.6012 - acc: 0.6927
Epoch 13/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5985 - acc: 0.6947
Epoch 14/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.6000 - acc: 0.6992
Epoch 15/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.6006 - acc: 0.6927
Epoch 16/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5981 - acc: 0.6955
Epoch 17/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5971 - acc: 0.6983
Epoch 18/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5941 - acc: 0.7025
Epoch 19/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5897 - acc: 0.7019
Epoch 20/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5926 - acc: 0.6975
Epoch 21/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5988 - acc: 0.6890
Epoch 22/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5873 - acc: 0.7025
Epoch 23/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5880 - acc: 0.7064
Epoch 24/150
```

5977/5977 [==============================] - 0s 39us/step - loss: 0.5922 - acc: 0.6997
Epoch 25/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5873 - acc: 0.7050
Epoch 26/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5806 - acc: 0.7082
Epoch 27/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5902 - acc: 0.6967
Epoch 28/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5882 - acc: 0.7007
Epoch 29/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5867 - acc: 0.7049
Epoch 30/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5916 - acc: 0.6988
Epoch 31/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5889 - acc: 0.7030
Epoch 32/150
5977/5977 [==============================] - 0s 39us/step - loss: 0.5894 - acc: 0.6977
Epoch 33/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5811 - acc: 0.7000
Epoch 34/150
5977/5977 [==============================] - 0s 39us/step - loss: 0.5851 - acc: 0.7037
Epoch 35/150
5977/5977 [==============================] - 0s 39us/step - loss: 0.5795 - acc: 0.7082
Epoch 36/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5781 - acc: 0.7050
Epoch 37/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5846 - acc: 0.7035
Epoch 38/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5800 - acc: 0.7040
Epoch 39/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5845 - acc: 0.7020
Epoch 40/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5783 - acc: 0.7060
Epoch 41/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5872 - acc: 0.7019
Epoch 42/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5771 - acc: 0.7101
Epoch 43/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5801 - acc: 0.7052
Epoch 44/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5844 - acc: 0.6992
Epoch 45/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5869 - acc: 0.6957
Epoch 46/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5795 - acc: 0.7126
Epoch 47/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5779 - acc: 0.7047
Epoch 48/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5852 - acc: 0.6985
Epoch 49/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5706 - acc: 0.7085

Epoch 50/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5804 - acc: 0.7080
Epoch 51/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5806 - acc: 0.7099
Epoch 52/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5728 - acc: 0.7116
Epoch 53/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5814 - acc: 0.7052
Epoch 54/150
5977/5977 [==============================] - 0s 31us/step - loss: 0.5761 - acc: 0.7069
Epoch 55/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5770 - acc: 0.7040
Epoch 56/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5778 - acc: 0.7074
Epoch 57/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5708 - acc: 0.7136
Epoch 58/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5793 - acc: 0.6998
Epoch 59/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5814 - acc: 0.7014
Epoch 60/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5776 - acc: 0.7101
Epoch 61/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5742 - acc: 0.7089
Epoch 62/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5731 - acc: 0.7117
Epoch 63/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5708 - acc: 0.7149
Epoch 64/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5760 - acc: 0.7060
Epoch 65/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5774 - acc: 0.7030
Epoch 66/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5730 - acc: 0.7146
Epoch 67/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5692 - acc: 0.7092
Epoch 68/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5741 - acc: 0.7080
Epoch 69/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5761 - acc: 0.7146
Epoch 70/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5729 - acc: 0.7122
Epoch 71/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5718 - acc: 0.7141
Epoch 72/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5687 - acc: 0.7087
Epoch 73/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5717 - acc: 0.7104
Epoch 74/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5718 - acc: 0.7161
Epoch 75/150

5977/5977 [==============================] - 0s 34us/step - loss: 0.5754 - acc: 0.7054
Epoch 76/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5708 - acc: 0.7064
Epoch 77/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5708 - acc: 0.7119
Epoch 78/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5727 - acc: 0.7126
Epoch 79/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5676 - acc: 0.7142
Epoch 80/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5670 - acc: 0.7161
Epoch 81/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5754 - acc: 0.7059
Epoch 82/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5707 - acc: 0.7119
Epoch 83/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5712 - acc: 0.7121
Epoch 84/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5754 - acc: 0.7159
Epoch 85/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5707 - acc: 0.7099
Epoch 86/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5698 - acc: 0.7106
Epoch 87/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5707 - acc: 0.7101
Epoch 88/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5732 - acc: 0.7092
Epoch 89/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5719 - acc: 0.7193
Epoch 90/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5617 - acc: 0.7161
Epoch 91/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5756 - acc: 0.7167
Epoch 92/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5612 - acc: 0.7206
Epoch 93/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5702 - acc: 0.7141
Epoch 94/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5685 - acc: 0.7109
Epoch 95/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5780 - acc: 0.6995
Epoch 96/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5670 - acc: 0.7096
Epoch 97/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5725 - acc: 0.7069
Epoch 98/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5679 - acc: 0.7141
Epoch 99/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5631 - acc: 0.7111
Epoch 100/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5696 - acc: 0.7141

Epoch 101/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5680 - acc: 0.7144
Epoch 102/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5709 - acc: 0.7161
Epoch 103/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5735 - acc: 0.7025
Epoch 104/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5623 - acc: 0.7139
Epoch 105/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5638 - acc: 0.7184
Epoch 106/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5665 - acc: 0.7144
Epoch 107/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5675 - acc: 0.7152
Epoch 108/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5717 - acc: 0.7096
Epoch 109/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5686 - acc: 0.7074
Epoch 110/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5714 - acc: 0.7069
Epoch 111/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5708 - acc: 0.7139
Epoch 112/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5695 - acc: 0.7074
Epoch 113/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5597 - acc: 0.7206
Epoch 114/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5698 - acc: 0.7040
Epoch 115/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5666 - acc: 0.7119
Epoch 116/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5622 - acc: 0.7152
Epoch 117/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5646 - acc: 0.7126
Epoch 118/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5611 - acc: 0.7137
Epoch 119/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5669 - acc: 0.7106
Epoch 120/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5698 - acc: 0.7126
Epoch 121/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5667 - acc: 0.7094
Epoch 122/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5685 - acc: 0.7102
Epoch 123/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5676 - acc: 0.7124
Epoch 124/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5673 - acc: 0.7146
Epoch 125/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5698 - acc: 0.7116
Epoch 126/150

5977/5977 [==============================] - 0s 37us/step - loss: 0.5650 - acc: 0.7154
Epoch 127/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5707 - acc: 0.7091
Epoch 128/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5559 - acc: 0.7270
Epoch 129/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5745 - acc: 0.7029
Epoch 130/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5695 - acc: 0.7099
Epoch 131/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5631 - acc: 0.7211
Epoch 132/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5686 - acc: 0.7166
Epoch 133/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5678 - acc: 0.7112
Epoch 134/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5668 - acc: 0.7112
Epoch 135/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5640 - acc: 0.7203
Epoch 136/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5668 - acc: 0.7152
Epoch 137/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5644 - acc: 0.7142
Epoch 138/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5667 - acc: 0.7097
Epoch 139/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5657 - acc: 0.7147
Epoch 140/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5739 - acc: 0.7106
Epoch 141/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5579 - acc: 0.7151
Epoch 142/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5627 - acc: 0.7144
Epoch 143/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5701 - acc: 0.7084
Epoch 144/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5616 - acc: 0.7176
Epoch 145/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5598 - acc: 0.7166
Epoch 146/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5615 - acc: 0.7114
Epoch 147/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5612 - acc: 0.7164
Epoch 148/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5578 - acc: 0.7141
Epoch 149/150
5977/5977 [==============================] - 0s 37us/step - loss: 0.5635 - acc: 0.7151
Epoch 150/150
5977/5977 [==============================] - 0s 34us/step - loss: 0.5602 - acc: 0.7157
roc_auc_score val 0.842679325457307
[0.51018554 0.48981449 0.68678313 ..., 0.28903675 0.4207736

```
 0.57922637]
[0 1 1 ..., 0 0 1]
roc_auc_score val 0.8285669059867715

0.46451612903225803
from catboost import CatBoostClassifier
model = CatBoostClassifier(iterations = 300, loss_function  = 'Logloss',depth=10,
                           eval_metric='F1',learning_rate=0.1,)#od_type='Iter')
# X,Y = X_res_vis, y_resampled
X,x,Y,y = train_test_split(X_resampled, y_resampled, test_size = 0.10)
# X,x,Y,y = train_test_split(x_train, y_train, test_size = 0.1)


model.fit(X,Y,eval_set=(x,y),use_best_model=True,plot=True,verbose=False)
preds = model.predict(x)
f1_score(preds,y)

---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-450-90e953ad3bf4> in <module>()
      7
      8
----> 9 model.fit(X,Y,eval_set=(x,y),use_best_model=True,plot=True,verbose=False)
     10 preds = model.predict(x)
     11 f1_score(preds,y)

C:\Program Files\Anaconda3\lib\site-packages\catboost\core.py in fit(self, X, y, cat_features,
sample_weight, baseline, use_best_model, eval_set, verbose, logging_level, plot)
   1472         model : CatBoost
   1473         """
-> 1474         self._fit(X, y, cat_features, None, sample_weight, None, None, baseline,
use_best_model, eval_set, verbose, logging_level, plot)
   1475         if y is not None:
   1476             setattr(self, "_classes", np.unique(y))

C:\Program Files\Anaconda3\lib\site-packages\catboost\core.py in _fit(self, X, y, cat_features,
pairs, sample_weight, query_id, pairs_weight, baseline, use_best_model, eval_set, verbose,
logging_level, plot)
    659                 raise ImportError(str(e))
    660         with log_fixup():
--> 661             self._train(X, eval_set, params)
    662         if calc_feature_importance:
    663             setattr(self, "_feature_importance", self.get_feature_importance(X))

_catboost.pyx in _catboost._CatBoostBase._train()

_catboost.pyx in _catboost._CatBoost._train()

_catboost.pyx in _catboost._CatBoost._train()

KeyboardInterrupt:
```

```
preds = pd.DataFrame(model.predict(x_test.as_matrix()))
preds[1].round().to_csv('preds.csv',index = False)

preds.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1836 entries, 0 to 1835
Data columns (total 2 columns):
0    1836 non-null float32
1    1836 non-null float32
dtypes: float32(2)
memory usage: 14.4 KB
```

Models

```
from sklearn.model_selection import KFold, cross_val_score
from sklearn import preprocessing
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model  import LogisticRegression
from sklearn.linear_model  import LogisticRegressionCV

from sklearn.model_selection import train_test_split

#######Сохранение данных


#x_train.to_csv('x_train.csv')
#x_test.to_csv('x_test.csv')

##### МОДЕЛИ #####
##### МОДЕЛИ #####
##### МОДЕЛИ #####
##### МОДЕЛИ #####
##### МОДЕЛИ #####

### Сохранение

#pd.DataFrame(model.predict(x_test))[0].to_csv('preds.csv',index = False)

from sklearn.model_selection import cross_val_score, StratifiedKFold, GridSearchCV
from sklearn.metrics import accuracy_score
from sklearn.ensemble  import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score

rfc = RandomForestClassifier(random_state=42, n_jobs=-1, oob_score=True)
X,x,Y,y = train_test_split(x_train, y_train, test_size = 0.1, random_state = 0)
#results = cross_val_score(rfc, X, Y, cv=skf)
rfc.fit(X, Y)
```

preds = rfc.predict(x)

f1_score(preds, y)

pd.DataFrame(rfc.predict(x_test))

	0
0	0
1	0
2	0
3	0
4	1
5	0
6	0
7	0
8	0
9	0
10	0
11	1
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0

21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	1
29	0
...	...
1596	0
1597	0
1598	0
1599	1
1600	1
1601	0
1602	0
1603	0
1604	0
1605	0
1606	0
1607	0
1608	0
1609	1
1610	0

1611	0
1612	1
1613	0
1614	0
1615	0
1616	1
1617	0
1618	0
1619	0
1620	0
1621	0
1622	0
1623	0
1624	0
1625	0

1626 rows × 1 columns