

## Домашняя работа по УМВАД по анализу данных

Выполнила: Комиссарова Дарья, ИССА

### Описание данных

Для анализа было выбрано 2 dataseta:

1. Данные по **пассажирам Титаника** (689 объектов)

Целевой признак – **выжил** ли **пассажир**.

- из всех признаков было выбрано **3 входных признака**: пол пассажира, его возраст, и класс обслуживания (1,2 или 3).
- эти признаки были **разделены на несколько** признаков:
  - возраст пассажира был разделен на несколько интервалов  $\geq 18$  лет,  $\geq 50$  лет,  $< 50$  лет,  $< 18$  лет
  - класс обслуживания (1 класс – самый лучший) также разделили на интервалы:  $\geq 2$  – го класса,  $\geq 3$  – го класса,  $< 3$  – го и  $< 2$  – го.
  - Пол пассажира представили 2-мя признаками: мужской и женский пол.

Итого мы получили **10 входных признаков** и 1 целевой.

```
1 class>=2,class>=3,class<=2,class<=1,age>=18,age>=50,age<50,age<18,male,female,survived
2 1,1,0,0,1,0,1,0,1,0,0
3 0,0,1,1,1,0,1,0,0,1,1
4 1,1,0,0,1,0,1,0,0,1,1
5 0,0,1,1,1,0,1,0,0,1,1
6 1,1,0,0,1,0,1,0,1,0,0
7 0,0,1,1,1,1,0,0,1,0,0
8 1,1,0,0,0,0,0,1,1,1,0,0
9 1,1,0,0,1,0,1,0,0,1,1
10 1,0,1,0,0,0,1,1,0,1,1
11 1,1,0,0,0,0,1,1,0,1,1
```

2. Данные по крестикам-ноликам (3x3, свои, не предложенные для анализа данные) (958 объектов – примеров)

Целевой признак – **выиграл** ли **1-й игрок** (который играет за x)

- признак заключается в том, содержится ли в **некоторой из 9 ячеек поля крестик** (перебор идет с левой верхней ячейки направо)
- обязательное преобразование признаков в другой вид не требуется

Итого мы получили **9 входных признаков** и 1 целевой.

```
1 x,x,x,x,o,o,x,o,o,1
2 x,x,x,x,o,o,o,o,o,1
3 x,x,x,x,o,o,o,o,o,1
4 x,x,x,x,o,o,o,o,o,1
5 x,x,x,x,o,o,o,o,o,1
6 x,x,x,x,o,o,o,o,o,1
7 x,x,x,x,o,o,o,o,o,1
8 x,x,x,x,o,o,o,o,o,1
9 x,x,x,x,o,o,o,o,o,1
10 x,x,x,x,o,x,o,o,o,1
11 x,x,x,x,o,o,o,x,o,1
12 x,x,x,o,x,o,o,o,x,1
13 x,x,x,o,x,o,o,o,o,1
14 x,x,x,o,x,o,o,o,o,1
15 x,x,x,o,x,o,o,o,o,1
```

## Первичная работа с данными

- Нам требовалось проверить качество классификации на имеющихся данных. Для этого из всего набора данных было выбрано **случайным образом** 80% и 20% объектов, составивших **обучающую** и **тестовую** выборки соответственно.
- **обучающая** выборка была разделена на **2 класса**:  $K_{positive}$  и  $K_{negative}$  в соответствии со значением целевого признака.

## Нахождение гипотез

Для **каждой** выборки  $K_{positive}$  и  $K_{negative}$  была построена **решетка формальных понятий**.

Нахождение формальных контекстов было реализовано с помощью **алгоритма Гантера**. Далее по формальным понятиям мы выбирали те содержания, которые являются **положительными** или **отрицательными гипотезами** соответственно.

Алгоритм действует по следующей схеме (по определению гипотез):

Для каждого содержания из решетки  $positive$  проверяется выполнение условия

$$h_+ : \forall g- \in G^-, h_+ \notin \{g-\}^-),$$

после чего находится число противоречий к данному условию.

Если это число не превышает количества объектов в формальном понятии, умноженного на **заданное пороговое значение  $\eta$** , то данное содержание принимаем как положительную гипотезу, в противном случае не включаем его во множество гипотез.

Тогда

$$num_{contradictions_i} < \eta * |concept[i]|$$

$\forall i \in |K_+|$ , где  $|concept[i]|$  – количество объектов в формал. понят.

Абсолютно симметрично рассматриваются отрицательные гипотезы.

Получили, что для dataset Titanic количество **формальных понятий** для **положительного** набора  $K_+$  равно **95**, для **отрицательного** – **75**. А при уровне  $\eta = 0.1$  число гипотез оказалось равным **18** и **5** гипотез соответственно.

```

classifier.py
example.py
test.py
tic-tac.data
titanic.data
two_usefull_functions.txt
External Libraries

i_h_n += 1
#print(concept[i])
print('Lattices sizes ',len(lattice_positive), len(lattice_negative))
print('Hypotesis sizes ',len(h_p),len(h_n))
return h_p, h_n

def scheme1_a(test_data, h_p, h_n, percent_aggr):
    num_attributes = len(test_data[list(test_data.keys())[0]])

    future_test_posit = []
    future_test_negat = []

Run test
C:\Users\1\AppData\Local\Programs\Python\Python35-32\python.exe "C:/Python проекты/Большой Андан/test.py"
689 689
Lattices sizes 95 75
Hypotesis sizes 18 5
scheme1
53 58 138

```

Таким образом, мы нашли множества положительных и отрицательных гипотез для наших данных.

## Классификация объектов из тестовой выборки путем выбора агрегирующего правила

Для классификации объектов на основе построенных гипотез были использованы 2 схемы:

### 1. По количеству объектов

Во-первых, среди всех положительных (отрицательных) гипотез мы для агрегации **не рассматриваем** более «бедные» по набору атрибутов гипотезы, у которых есть обобщающие их (по атрибутам) гипотезы.

Во-вторых, мы рассматриваем **объединение** всех **объектов** из подошедших гипотез.

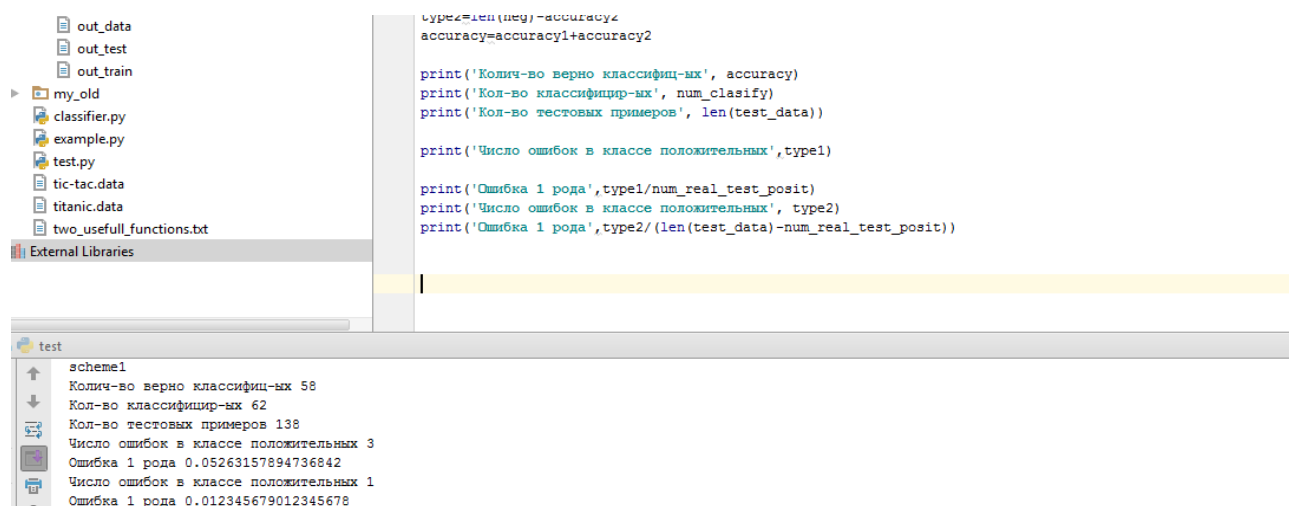
Агрегирующее правило состоит в следующем:

- если найденное **количество объектов** для **положительных** гипотез **превосходит** аналогичное число объектов для **отрицательных** с некоторым **пороговым значением**, то **относим** данный исследуемый пример к **классу положительных** объектов, в симметричном случае – к отрицательным, иначе – не классифицируем.

$$\bigcup_{m_i \in K_+'} m_i < \eta_2 * \bigcup_{m_i \in K_-'} m_i$$

### 2. По количеству гипотез

Здесь сравнивается лишь **число положительных** и **отрицательных** гипотез, удовлетворяющих текущему примеру. Если число одних с **некоторым пороговым значением** превышает число других, то отправляем в подходящий класс (если нет, то не классифицируем).



Результаты работы алгоритмов:

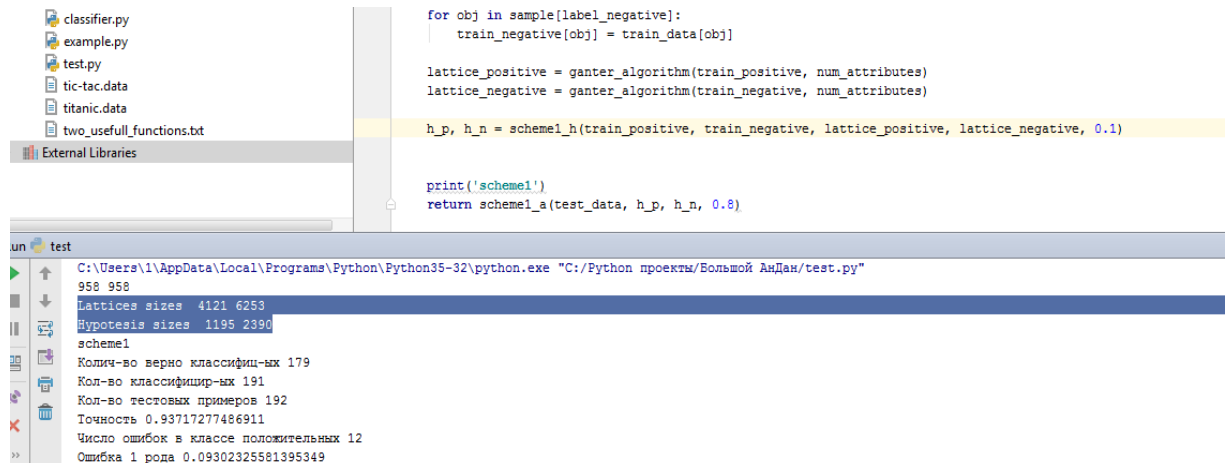
Данные	метод	Варьируемый множитель $\eta_1$	Варьируемый множитель $\eta_2$	Точность	Ошибка 1 рода	Ошибка 2 рода
titanic	1 метод	0.1	0.2	0.85	0.08	0.03
		0.1	0.8	0.91	0.06	0.02
		0.4	0.2	0.83	0.01	0.25
		0.4	0.8	0.84	0.06	0.24
	2 метод	0.1	0.2	0.82	0.04	0.04
		0.1	0.8	0.93	0.007	0.02
		0.4	0.2	0.77	0.09	0.17
		0.4	0.8	0.81	0.07	0.21
Tic_tac	1 метод	0.1	0.2	0.88	0.11	0.04
		0.1	0.8	0.95	0.06	0.03
		0.4	0.2	0.84	0.2	0.13
		0.4	0.8	0.87	0.14	0.19
	2 метод	0.1	0.2	0.79	0.09	0.11
		0.1	0.8	0.89	0.02	0.05
		0.4	0.2	0.8	0.07	0.16
		0.4	0.8	0.83	0.1	0.2

## Вывод

По результатам можно подвести итог:

- 1) Оптимальным для обоих алгоритмов стало низкое значение параметра  $\eta_1$  — доли противоречий при определении гипотез (0.1); а также довольно высокий порог для доли положительных и отрицательных объектов (0.8).

- 2) Для второго датасета про крестики-нолики программа находила гораздо большее число формальных содержаний и гипотез даже при  $\eta_1$ , а именно



```
classifier.py
example.py
test.py
tic-tac.data
titanic.data
two_usefull_functions.txt
External Libraries

for obj in sample[label_negative]:
    train_negative[obj] = train_data[obj]

lattice_positive = ganter_algorithm(train_positive, num_attributes)
lattice_negative = ganter_algorithm(train_negative, num_attributes)

h_p, h_n = schemel_h(train_positive, train_negative, lattice_positive, lattice_negative, 0.1)

print('schemel')
return schemel_a(test_data, h_p, h_n, 0.8)

run test
C:\Users\1\AppData\Local\Programs\Python\Python35-32\python.exe "C:/Python проекты/Большой Андан/test.py"
958 958
Lattices sizes 4121 6253
Hypotesis sizes 1195 2390
schemel
Колич-во верно классифицир-ых 179
Кол-во классифицир-ых 191
Кол-во тестовых примеров 192
Точность 0.93717277486911
Число ошибок в классе положительных 12
Ошибка 1 рода 0.09302325581395349
```

но в большинстве случаев точность классификации для него была выше.

- 3) Оба способа агрегации дают достаточно высокие показатели точности классификации ( в среднем 0.8-0.92) и низкие – ошибок 1 и 2 рода, колеблющиеся в зависимости от параметров.