

**МЕТОДИЧЕСКАЯ РАЗРАБОТКА**  
к лабораторным работам по курсу ИИ  
(34 часа, из них 4 часа УСП)

**Автор:** доцент кафедры информационных систем управления  
факультета прикладной математики и информатики БГУ Образцов В.А.

## График и характер проведения лабораторных работ

| Тематика              | № лабораторной | Смысл работы                                      | № занятия | Характер работы     | Ссылка на описание работы |
|-----------------------|----------------|---|-----------|---------------------|---------------------------|
| Логический вывод      | -              | Ознакомление с задачей, постановка                | 1.        | УСР                 | Стр.2                     |
|                       | <b>1</b>       | Прямой вывод в БЗ продукционного типа             | 2.        | Работа над заданием | Стр. 3-4                  |
|                       |                |   | 3.        |                     |                           |
|                       |                |   | 4.        | Отчет               |                           |
|                       | <b>2</b>       | Табличный вывод в БЗ продукционного типа          | 5.        | Работа над заданием | Стр. 5-7                  |
|                       |                |   | 6.        |                     |                           |
|                       |                |   | 7.        | Отчет               |                           |
|                       | <b>3</b>       | Обратный вывод в БЗ продукционного типа           | 8.        | Работа над заданием | Стр. 8                    |
|                       |                |   | 9.        |                     |                           |
|                       |                |   | 10.       | Отчет               |                           |
| Распознавание образов | -              | Ознакомление с задачей, постановка                | 11.       | УСР                 | Стр. 9-11                 |
|                       | <b>4</b>       | Решение задачи распознавания образов без обучения | 12.       | Работа над заданием | Стр. 12-13                |
|                       |                |   | 13.       |                     |                           |
|                       |                |   | 14.       | Отчет               |                           |
|                       | <b>5</b>       | Решение задачи распознавания образов с обучением  | 15.       | Работа над заданием | Стр. 14-17                |
|                       |                |   | 16.       |                     |                           |
|                       |                |   | 17.       | Отчет               |                           |

# ПРИМЕР БАЗЫ ЗНАНИЙ (БЗ) "ПРОДУКЦИОННОГО" ТИПА

(см. статью: **Работа с Микроэкспертом**, в сб. **Вычислительная техника и ее применение**, №10, 1990 г., Стр. 33-45)

- 1 Если класс - голосеменные и структура листа – чешуеобразная  
То семейство - кипарисовые
- 2 Если класс - голосеменные и структура листа - иглоподобная и конфигурация - хаотическая  
То семейство - сосновые
- 3 Если класс - голосеменные и структура листа – иглоподобная и конфигурация - 2 ровных ряда  
То семейство - еловые
- 4 Если класс - голосеменные и структура листа - иглоподобная и конфигурация - 2 ровных ряда и серебристая полоса - нет  
То семейство - болотный кипарис
- 5 Если тип - деревья и форма листа - широкая и плоская  
То класс - покрытосеменные
- 6 Если тип - деревья и форма листа – не (широкая и плоская)  
То класс - голосеменные
- 7 Если стебель - зеленый  
То тип - травянистые
- 8 Если стебель - древесный и положение - стелющееся  
То тип - лианы
- 9 Если стебель - древесный и положение - прямостоящее и один основной ствол - да  
То тип - деревья
- 10 Если стебель - древесный и положение - прямостоящее и один основной ствол - нет  
То тип - кустарниковые
- 11 Если голова - болит и кости - ломит и глаза - слезятся  
То заболевание - грипп

**Структура правил:** **Если** (название признака – значение признака и ... *признаков может быть несколько*), **То** (название признака – значение признака). Признаки (точнее их названия), которые перечислены после **Если** будем называть **посылочными**. Признаки после **То** будем называть **заключительными**.

## Требования к практической задаче:

1. Число правил должно быть не менее 10.
2. Между посылочными и заключительными признаками должна быть транзитивная зависимость – по крайней мере 1 заключительный признак должен быть посылочным в другом правиле.

# АЛГОРИТМ ДЕДУКТИВНОГО ВЫВОДА ДЛЯ БЗ "ПРОДУКЦИОННОГО" ТИПА

**Begin** Поместить конечную цель в стек целей. **Логический признак**:= false

**While** not **логический признак** do

**if** можно найти правило для анализа **then**

**case** (значение правила) **of**

**истина:**

**Begin** поместить целевой признак, его значение и номер анализируемого правила в стек контекста, и принять соответствующее правило.

**if** стек целей пуст **then** **логический признак**: = true

**else** рассматривать следующую цель из стека целей

**end**

**ложь:**

                поместить правило в колоду сброса.

**неизвестно:**

                поместить первый неизвестный признак и номер соответствующего правила в стек целей

**end {case}**

**else**

**if** имеется вопрос, связанный с текущей целью **then**

**begin** задать вопрос. Поместить полученное значение в стек контекста вместе с признаком, удалив его из стека целей.

**if** в стеке целей имеется номер правила **then** анализировать это правило (**case**).

**end**

**else** **логический признак**: = true

**end**

**if** конечная цель находится в стеке контекста **then** ответ получен.

**else** ответ не может быть найден.

**end**

Приведенный алгоритм реализует т.н. **прямой** вывод. Смысл: для произвольного заключительного признака (по заданному названию) требуется определить его значение в результате определения значений посылочных признаков, полученных в диалоге с пользователем на базе имеющихся логических зависимостей.

## ПРИМЕР РАБОТЫ АЛГОРИТМА (при выводе значения признака семейство)

| Шаг | Анализируемое правило |            | Ответ на вопрос | Стек целей:     |   | Контекстный стек: |               | № принимаемого правила | № отбрасываемого правила |
|-----|-----------------------|------------|-----------------|-----------------|---|-------------------|---------------|------------------------|--------------------------|
|     | №                     | значение   |                 | признак         | № | признак           | значение      |                        |                          |
| 1   |                       |            |                 | семейство       |   |                   |               |                        |                          |
| 2   | 1                     | неизвестно |                 | класс           | 1 |                   |               |                        |                          |
| 3   | 5                     | неизвестно |                 | тип             | 5 |                   |               |                        |                          |
| 4   | 7                     | неизвестно |                 | стебель         | 7 |                   |               |                        |                          |
| 5   | нет                   |            | древесный       |                 |   | стебель           | древесный     |                        |                          |
| 6   | 7                     | ложь       |                 |                 |   |                   |               |                        | 7                        |
| 7   | 8                     | неизвестно |                 | положение       | 8 |                   |               |                        |                          |
| 8   | нет                   |            | прямо стоящее   |                 |   | положен.          | прямо стоящее |                        |                          |
| 9   | 8                     | ложь       |                 |                 |   |                   |               |                        | 8                        |
| 10  | 9                     | неизвестно |                 | 1основной ствол | 9 |                   |               |                        |                          |
| 11  | нет                   |            | да              |                 |   | 1осн.ств.         | да            |                        |                          |
| 12  | 9                     | истина     |                 |                 |   | тип               | деревья       | 9                      |                          |
| 13  | 5                     | неизвестно |                 | форма листа     |   |                   |               |                        |                          |
| 14  | нет                   |            | л(шир+пл)       |                 |   | форма листа       | л(ш+пл)       |                        |                          |
| 15  | 5                     | ложь       |                 |                 |   |                   |               |                        | 5                        |
| 16  | 6                     | истина     |                 |                 |   | класс             | голосеменные  | 6                      |                          |
| 17  | 1                     | неизвестно |                 | структура листа | 1 |                   |               |                        |                          |
| 18  | нет                   |            | чешуеобразная   |                 |   | структура листа   | чешуеобразная |                        |                          |
| 19  | 1                     | истина     |                 |                 |   | Семейство         | кипарис       | 1                      |                          |

### Требования к программному продукту:

1. Любой язык программирования, кроме **PROLOG** и **LISP**;
2. Графический интерфейс **обязателен**;
3. Вывод на экран по запросу log-файла, содержащего протокол вывода (пример см. на стр. 4. Можно попроще).

# АЛГОРИТМ ТАБЛИЧНОГО ВЫВОДА ДЛЯ БЗ "ПРОДУКЦИОННОГО" ТИПА

## Предварительные обозначения.

Пусть заданы  $n$  признаков, которые имеют наименования  $A_1, \dots, A_n$ . Каждому такому признаку  $A_i$  можно поставить в соответствие набор значений  $(a_{i1}, \dots, a_{ik_i})$ .

## Пример (по БЗ на стр. 2):

$A_1$ =семейство, ( $a_{11}$  = кипарисовые,  $a_{12}$  = сосновые,  $a_{13}$  = еловые,  $a_{14}$  = болотный кипарис);

$A_2$ =класс, ( $a_{21}$  = голосеменные);

$A_3$ =структура листа, ( $a_{31}$  = чешуеобразная,  $a_{32}$  = иглоподобная).

В общем случае правило продукции с помощью признаков  $A_i$  и их значений  $(a_{i1}, \dots, a_{ik_i})$  может быть представлено с использованием пар  $(A_i, a_{ij}), j \in \{1, 2, \dots, k_i\}$  в следующем виде:

$$P = (A_u, a_{uk}) \wedge \dots \wedge (A_v, a_{vl}) \Rightarrow (A_s, a_{sm}),$$

$u, v, s \in \{1, 2, \dots, n\}, k \in \{1, 2, \dots, k_u\}, l \in \{1, 2, \dots, k_l\}, m \in \{1, 2, \dots, k_m\}$ .

После приведения к КНФ получим:

$$P = \neg(A_u, a_{uk}) \vee \dots \vee \neg(A_v, a_{vl}) \vee (A_s, a_{sm}),$$

## Пример:

$$P_1 = (A_2, a_{21}) \wedge (A_3, a_{31}) \Rightarrow (A_1, a_{11}) \Leftrightarrow P_1 = \neg(A_2, a_{21}) \vee \neg(A_3, a_{31}) \vee (A_1, a_{11})$$

Предположим теперь, что БЗ содержит  $t$  правил  $P_1, \dots, P_t$  ( $t \in \mathbb{N}$ ). Приведем их все к КНФ и построим **таблицу правил**:

|         | Посылочные признаки   |         |                       |                       |         |                       |         |                       |         |                       | Заключительные признаки |         |                       |                       |         |                       |         |                       |         |                       |
|---------|-----------------------|---------|-----------------------|-----------------------|---------|-----------------------|---------|-----------------------|---------|-----------------------|-------------------------|---------|-----------------------|-----------------------|---------|-----------------------|---------|-----------------------|---------|-----------------------|
|         | $A_1$                 |         |                       | $A_2$                 |         |                       | $\dots$ | $A_n$                 |         |                       | $A_1$                   |         |                       | $A_2$                 |         |                       | $\dots$ | $A_n$                 |         |                       |
|         | $a_{11}$              | $\dots$ | $a_{1k_1}$            | $a_{21}$              | $\dots$ | $a_{2k_2}$            |         | $a_{n1}$              | $\dots$ | $a_{nk_n}$            | $a_{11}$                | $\dots$ | $a_{1k_1}$            | $a_{21}$              | $\dots$ | $a_{2k_2}$            |         | $a_{n1}$              | $\dots$ | $a_{nk_n}$            |
| $P_1$   | <b><math>b</math></b> |         | <b><math>b</math></b> | <b><math>b</math></b> |         | <b><math>b</math></b> |         | <b><math>b</math></b> |         | <b><math>b</math></b> | <b><math>b</math></b>   |         | <b><math>b</math></b> | <b><math>b</math></b> |         | <b><math>b</math></b> |         | <b><math>b</math></b> |         | <b><math>b</math></b> |
| $\dots$ |                       |         |                       |                       |         |                       |         |                       |         |                       |                         |         |                       |                       |         |                       |         |                       |         |                       |
| $P_t$   | <b><math>b</math></b> |         | <b><math>b</math></b> | <b><math>b</math></b> |         | <b><math>b</math></b> |         | <b><math>b</math></b> |         | <b><math>b</math></b> | <b><math>b</math></b>   |         | <b><math>b</math></b> | <b><math>b</math></b> |         | <b><math>b</math></b> |         | <b><math>b</math></b> |         | <b><math>b</math></b> |

Таблица заполняется в соответствии со следующим правилом:

Символы **b** в ячейке с индексом  $(i, j)$  принимают значения из множества  $\{\square, +, -\}$  в соответствии со следующими правилами. Фиксируем  $P_u$  и в клетки соответствующей строки заносим:

**b**= $\square$  (ничего не ставится, пробел) если признак  $A_i$  (как среди посылочных, так и заключительных) со значением  $a_{ij}$  не используется в правиле;

$b=-$ , если признак  $A_i$  является посылочным и его значение  $a_{ij}$  используется в правиле;

$b=+$ , если признак  $A_i$  является заключительным и его значение  $a_{ij}$  используется в правиле.

## **АЛГОРИТМ.**

**Шаг 0.** Модифицируем таблицу правил, оставив в ней только те посылочные и заключительные признаки, которые имеют значащие значения ( $\{+, -\}$ ) хотя бы для одного правила из набора  $P_1, \dots, P_t$ . Обозначим полученную таблицу (матрицу) через  $T_{np}$ . В результате получим **два набора признаков** –  $A_{пос}$  (посылочные) и  $A_{закл}$  (заключительные). Сформируем также **стек целей**  $St_{target}$  и **список полученных результатов**  $L_{res}$ , которые будут применяться в процессе вывода. Состоять они будут:

- $St_{target}$  из последовательности признаков  $A_u \in A_{закл}$ ;
- $L_{res}$  из последовательности установленных фактов в виде  $(A_u, a_{uj}) : A_u \in A_{пос} \vee A_{закл}, j \in \{1, 2, \dots, k_u\}$ .

*Задаем пользователю вопрос – какой из признаков из набора  $A_{закл}$  он желает вывести. Предположим, что ответ был такой – признак  $A_i \in A_{закл}$ .*

Заносим признак  $A_i$  в стек  $St_{target}$ .

**Шаг 1.** Формируем таблицу  $T_{goals}$ , которую по определению полагаем пустой (т.е. шапка и нет строк).

**Шаг 1.1.** Для текущего признака  $A_{current} \in St_{target}$  заносим в таблицу  $T_{goals}$  строки, соответствующие правилам, у которых заключительный признак  $A_{current}$  для одного из значений помечен знаком  $' + '$ . Соответствующий набор правил обозначим  $P_{current}$ . Если  $P_{current} = \emptyset$  то переходим на шаг 1.2. В противном случае ( $P_{current} \neq \emptyset$ ) – на шаг 2.

### **Шаг 1.2.**

Сообщаем пользователю:

*Ответ не может быть получен т.к. отсутствуют правила, соответствующие признаку  $A_{current}$ .*

переходим на шаг 6.

**Шаг 2.** В таблице  $T_{goals}$  находим строку (пусть для определенности это будет строка, соответствующая правилу  $P_u \in P_{current}$ ), у которой число признаков, помеченных знаком  $' - '$  минимально.

Если это число больше 0, то переходим к шагу 3.

В противном случае (равно 0) – заносим  $A_{current}$  и соответствующее ему значение в правиле  $P_u$  в  $L_{res}$  (обозначим полученный результат для определенности через  $(A_v, a_{vs})$ ) и в случае  $|St_{target}| = 1$  переходим к шагу 5. В противном случае ( $|St_{target}| > 1$ ) – к шагу 4.

**Шаг 3.** Выбираем первый признак в правиле  $P_u$ , помеченный знаком  $' - '$ . Пусть это будет признак  $A_v$ . Если  $A_v \notin A_{закл}$ , то:

Задаем пользователю вопрос – какое значение имеет признак  $A_v$ .  
Предположим, что ответ был такой –  $a_{vs}$ .

Заносим полученный ответ  $(A_v, a_{vs})$  в  $L_{res}$  и переходим к шагу 4.

В противном случае ( $A_v \in A_{закл}$ ) заносим  $A_v$  в  $St_{target}$  делая его  $A_{current}$  и переходим на шаг 1.1.

#### Шаг 4.

Применяем к матрице  $T_{goals}$  следующие преобразования.

**Шаг 4.1.** Последовательно фиксируем все строки (правила  $P_u$ ) из  $T_{goals}$ . Если просмотрены все строки, то переходим к шагу 4.2.

**Шаг 4.1.** Если  $A_v \in A_{пос}$  в правиле  $P_u$ , то переходим к шагу 4.1.1 Если  $A_u \in A_{закл}$  в правиле  $P_u$ , то переходим к шагу 4.1.2.

##### Шаг 4.1.1.

- Если признак  $A_v$  является значащим (помечен знаком ' – '), причем в позиции, которая не совпадает с номером  $s$ , то правило  $P_u$  исключаем из  $T_{goals}$ . Переходим к шагу 4.1;
- Если  $A_v$  является значащим (помечен знаком ' – ') и пара  $(A_v, a_{vs})$  также помечена знаком ' – ', то правило  $P_u$  исключаем из  $T_{goals}$ . Переходим к шагу 4.1;
- Если  $A_v$  является значащим (помечен знаком ' – '), причем в позиции, которая совпадает с номером  $s$ , то знак ' – ' заменяется на '  $\sqcup$  '. Переходим к шагу 4.1;

##### Шаг 4.1.2.

- Если  $A_v$  является помечен знаком ' + ' в позиции, которая не совпадает с номером  $s$ , то соответствующее ему значение  $a_{us}$  в виде  $-(A_u, a_{us})$  заносим в  $L_{res}$ , а правило  $P_u$  исключаем из  $T_{goals}$ . Переходим к шагу 4.1;
- Если  $A_v$  является помечен знаком ' + ' в позиции, которая совпадает с номером  $s$ , то правило  $P_u$  исключаем из  $T_{goals}$ . Переходим к шагу 4.1;

**Шаг 4.2.** Обновляем таблицу  $T_{goals}$  (помечаем удаленные правила  $P_u$ , чтобы не просматривать их в дальнейших шагах алгоритма) и переходим к шагу 3.0.

**Шаг 4.3.** Обнуляем список  $L_{res}$ , исключая из него все пары  $(A_v, a_{vs})$ . Если значение  $A_{current}$  установлено, то исключаем из  $St_{target}$  признак  $A_{current}$ . Меняем значение  $P_{current}$  (т.к. в стеке будет новое значение  $A_{current}$ ) и переходим к шагу 2.

**Шаг 5.** Сообщаем пользователю ответ:

Признак  $A_v$  имеет значение  $a_{vs}$ . (здесь пара  $(A_v, a_{vs})$  – последняя из занесенных в  $L_{res}$ ).

**Шаг 6.** Конец работы алгоритма.

**Требования к программному продукту** аналогичны изложенным на стр. 4. Кроме того, необходимо в программе вывести на экран последовательность  $T_{goals}$



## АЛГОРИТМ ОБРАТНОГО ВЫВОДА ДЛЯ БЗ “ПРОДУКЦИОННОГО” ТИПА

*Обратная цепочка рассуждений* применяется в задачах, соответствующих процессу проверки гипотез при решении проблем человеком — для заданной ситуации необходимо определить условия к ней приводящие.

**Алгоритм обратного вывода** обычно основан на *стратегии поиска в глубину*. Этот процесс предусматривает следующие шаги:

**Шаг 1.** Определить цель для обратного вывода и занести ее в стек целей. (Цель == признак и его значение).

**Шаг 2.** Сделать последнюю цель в стеке текущей. В списке правил найти первое правило, у которого заключительный признак соответствует текущей цели. Если правило найдено, то перейти к шагу 3. Если правило не найдено и число целей в стеке равно 1, сообщить пользователю, что ответ найти невозможно и перейти к шагу 7. Во всех остальных случаях занести признак, найденный на шаге 3, вместе с его значением в контекстный стек и вернуться на шаг 3.1

**Шаг 3.** Рассмотреть условную часть найденного правила последовательно выбирая все посылочные признаки.

**Шаг 3.1.** Если выбранный посылочный признак является выводимым (принадлежит к числу заключительных), то занести данный признак и его значение в стек целей и перейти к шагу 2. В противном случае занести признак вместе с его значением в контекстный стек.

**Шаг 3.2.** После исчерпания всех посылочных признаков для текущей цели, делаем проверку. Если число целей в стеке равно 1, то переходим к шагу 4. В противном случае удаляем последнюю текущую цель из стека целей и возвращаемся шагу 3, назначив новой текущей целью последнюю в стеке.

**Шаг 4.** Показать пользователю ответ, что цель для обратного вывода может быть достигнута при следующих условиях: *показать контекстный стек*.

**Шаг 5.** Алгоритм заканчивает работу.

**Требования к программному продукту** аналогичны изложенным на стр. 4. Кроме того, необходимо в программе вывести на экран последовательность  $T_{goals}$

## Подборка датасетов для задачи распознавания образов:

- [Биткойн, исторические данные](#) — данные биткойнов с интервалом в 1 минуту с избранных бирж, январь 2012 г. — март 2019 г.
- [FIFA 19 полный набор данных игроков](#) — 18k + FIFA 19 игроков, ~ 90 атрибутов, извлеченных из последней базы данных FIFA.
- [Статистика видео YouTube](#) — ежедневная статистика трендовых видео на YouTube.
- [Huge Stock Market Dataset](#) — исторические дневные цены и объемы всех американских акций и ETF.
- [Индикаторы мирового развития](#) — показатели развития стран со всего мира.
- [Kaggle Machine Learning & Data Science Survey 2017](#) — Большое представление о
- [Рентгенография грудной клетки \(пневмония\)](#) — 5,863 изображения, 2 категории.
- [Распознавание пола по голосу](#) — эта база данных была создана, чтобы идентифицировать голос как мужской или женский, основываясь на акустических свойствах голоса и речи. Набор данных состоит из 3168 записанных голосовых сэмплов, собранных от мужчин и женщин.
- [Студенческое потребление алкоголя](#) — данные были получены в ходе опроса учащихся по математике и португальскому языку на курсах в средней школе. Он содержит много интересной социальной, гендерной и учебной информации о студентах.
- [Набор данных о клетках малярии](#) — сотовые изображения для выявления малярии.
- [Опросы молодых людей](#) — данные о предпочтениях, интересах, привычках, мнениях и страхах молодых людей.
- [Мировые рейтинги университетов](#) — лучшие университеты мира.
- [Обнаружение мошенничества с кредитными картами](#) — датасет по анонимным транзакциям кредитных карт, помеченные как мошеннические или подлинные.
- [Датасет болезней сердца](#) — эта база данных содержит 76 атрибутов, таких как возраст, пол, тип боли в груди, артериальное давление в покое и другие.
- [Европейская футбольная база](#) — 25 000+ матчей, атрибуты игроков и команд для европейского профессионального футбола.
- [Винные обзоры](#) — 130k винных обзоров с разнообразием, местоположением, винодельней, ценой и описанием.
- [Baidu Apolloscapes](#). Большой датасет для распознавания 26 семантически разных объектов вроде машин, велосипедов, пешеходов, зданий, уличных фонарей и т. д.
- [Comma.ai](#). Более семи часов езды по шоссе. Датасет включает информацию о скорости машины, ускорении, угле поворота руля и GPS-координатах.
- [Распознавание цветов](#) — этот набор данных содержит 4242 изображения цветов. Сбор данных основан на данных flickr, изображениях Google, изображениях Яндекса.
- [Ежедневная рыночная цена каждой криптовалюты](#) — исторические цены на криптовалюту для всех токенов.
- [Шоколадный рейтинг](#) — Экспертный рейтинг более 1700 шоколадных батончиков.
- [Рынок медицинского страхования](#) — данные о планах в области здравоохранения и стоматологии на рынке медицинского страхования США.
- [Звуки сердцебиения](#) — классификация аномалий сердцебиения по стетоскопу.
- [База данных аниме рекомендаций](#) — рекомендации от 76 000 пользователей на myanimelist.net
- [Изображения клеток крови](#) — 12 500 изображений: 4 разных типа клеток.
- [Рентгенография грудной клетки](#) — более 112 000 рентгенограмм грудной клетки от более чем 30 000 уникальных пациентов.

- [База данных поддержанных автомобилей](#) — более 370000 поддержанных автомобилей. Содержание данных на немецком языке, поэтому нужно сначала перевести их, если вы не говорите на немецком.
- [Дом открытых данных правительства США](#) — данные, инструменты и ресурсы для проведения исследований, разработки веб-приложений и мобильных приложений, разработки визуализаций данных.
- [Национальный центр](#) профилактики хронических заболеваний и укрепления здоровья (NCCDPHP). Центр работает над снижением факторов риска хронических заболеваний.
- [Крупнейший](#) в Великобритании сборник социальных, экономических и демографических ресурсов.
- [EconData](#) — несколько тысяч экономических временных рядов, подготовленных рядом правительственных учреждений США и распространенных в различных форматах и СМИ.
- [Центр исследования побережья](#) — интересные данные о море и его биологическом составе. Здесь можно найти датасеты начиная с анализа данных модели Красного моря до исследования температуры и течений над узким южным калифорнийским шельфом.
- [Набор данных цифр языка жестов](#) — Турция, Анкара, Айранджи, Анадолу. Набор данных о языке жестов средней школы.
- [Качество красного вина](#) — простой и понятный практический набор данных для регрессионного или классификационного моделирования.
- [Таблицы](#) английской футбольной премьер-лиги (1968-2019).
- [HotspotQA Dataset](#) — датасет с вопросами-ответами, позволяющий создавать системы для ответов на вопросы более понятным способом.
- [xView](#) — один из самых больших общедоступных наборов воздушных снимков земли. Он содержит изображения различных сцен со всего мира, аннотированных с помощью ограничительных рамок.
- [Labelme](#) — Большой датасет аннотированных изображений.
- [ImageNet](#) — Датасет изображений для новых алгоритмов, организованный в соответствии с иерархией WordNet, в которой сотни и тысячи изображений представляют каждый узел иерархии.
- [LSUN](#). — датасет изображений, разбитых по сценам и категориям с частичной разметкой данных.
- [MS COCO](#) — крупномасштабный датасет для обнаружения и сегментации объектов.
- [COIL100](#) — 100 разных объектов, изображённых под каждым углом в круговом обороте.
- [Visual Genome](#) — датасет с ~100 тыс. подробно аннотированных изображений.
- [Google's Open Images](#). — коллекция из 9 миллионов URL-адресов к изображениям, «которые были помечены метками, охватывающими более 6000 категорий» под лицензией Creative Commons.
- [Labelled Faces in the Wild](#) — набор из 13 000 размеченных изображений лиц людей для использования приложений, которые предполагают использование технологии распознавания лиц.
- [Stanford Dogs Dataset](#) — содержит 20 580 изображений из 120 пород собак.
- [Indoor Scene Recognition](#). — датасет для распознавания интерьера зданий. Содержит 15 620 изображений и 67 категорий.
- [Oxford's Robotic Car](#) — более 100 повторений одного маршрута по Оксфорду, заснятого в течение года. В датасет попали разные комбинации погодных условий,

трафика и пешеходов, а также более длительные изменения вроде дорожных работ.

- [Cityscape Dataset](#) — большой датасет, содержащий записи ста уличных сцен в 50 городах.
- [KUL Belgium Traffic Sign Dataset](#) — более 10 000 аннотаций тысяч разных светофоров в Бельгии.
- [LISA Laboratory for Intelligent & Safe Automobiles](#) — датасет с дорожными знаками, светофорами, распознанными средствами передвижения и траекториями движения.
- [WPI datasets](#) — датасет для распознавания светофоров, пешеходов и дорожной разметки.
- [Berkeley DeepDrive](#) — огромный датасет для автопилотов. Он содержит более 100 000 видео с более чем 1100 часами записей вождения в разное время дня и в различных погодных условиях.
- [MIMIC-III](#) — датасет с обезличенными данными о состоянии здоровья ~40 000 пациентов, находящихся на интенсивной терапии (демографическими данными, показатели жизнедеятельности, лабораторными анализами и лекарствами).
- [Amazon Reviews](#) — Содержит около 35 млн отзывов с Amazon за 18 лет. Данные включают информацию о продукте и пользователе, оценки и сам текст отзыва.

#### Полезные ссылки по поиску датасетов:

- [Kaggle](#) — место встречи всех любителей соревнований по машинному обучению.
- [Google Dataset Search](#) — поиск датасетов по всей сети интернет. Также, при необходимости можно добавить [свои наборы данных](#).
- [Machine Learning Repository](#) — набор баз данных, теорий предметной области и генераторов данных, которые используются сообществом машинного обучения для эмпирического анализа алгоритмов машинного обучения.
- [VisualData](#) — поиск датасетов для машинного зрения, с удобной классификацией по категориям.
- [DATA USA](#) — полный набор по общедоступным данным США с визуализацией, описанием и инфографикой.

---

В итоге выбранная для лабораторных работ 4 и 5 задача (обозначим ее для краткости через  $T_0(\text{ask})$ ) должна удовлетворять следующей постановке:

#### ЗАДАНЫ:

- некоторое множество объектов  $X$ , разбитое на подмножества (классы)  $X_1, \dots, X_l$  ( $l \in \mathbb{N}$ ). Причем, классы не пересекаются  $X_i \cap X_j = \emptyset \forall i \neq j$  ( $i, j \in \{1, \dots, l\}$ ).
- выборка объектов  $X^0 \subset X$ , которая удовлетворяет условиям:  $|X^0| < +\infty$  и  $X^0 \cap X_i \neq \emptyset$  ( $\forall i \in \{1, \dots, l\}$ ). Кроме того, для каждого объекта  $x \in X^0$  известна (определена) информация о принадлежности к классам  $X_1, \dots, X_l$ . Эта информация задается в виде **информационного вектора**  $P(x) = (P_1(x), \dots, P_l(x))$ , компоненты которого определяются следующим образом

$$P_i(x) = \begin{cases} 1, & \text{если } x \in X_i, \\ 0, & \text{иначе.} \end{cases}$$

---

## ЗАДАЧА РАСПОЗНАВАНИЯ БЕЗ ОБУЧЕНИЯ

В этой задаче (обозначим ее через  $T_1$ ) требуется некоторое множество объектов  $X^0$  разбить на конечное число подмножеств (классов, таксонов, кластеров). В идеальном случае полученные классы при разбиении  $X^0$  должны соответствовать разбиению множества  $X$  на классы.

**Задана** задача  $T_0$  (см. стр. 11). Для данной задачи информационный вектор в процессе решения задачи не участвует, хотя для задачи  $T_0$  он известен. Для задачи  $T_0$  известно также число классов  $l$ .

Опишем простейший алгоритм, известный как алгоритм **иерархической кластеризации**. Он может использоваться для решения задачи  $T_1$ .

**Шаг 0** (предварительный).

Формируем первоначальный набор классов. Для этого каждый объект из  $X^0$  отождествляем с некоторым классом  $X'_i$ . Всего таких классов на предварительном шаге может быть сформировано по числу объектов, входящих в выборку  $|X^0| = k$ . Обозначим полученный набор классов через  $X' = (X'_1, \dots, X'_k)$ .

Далее алгоритм реализуется как последовательность шагов 1-4.

**Шаг 1.** Для каждого класса из  $X'_i \in X'$  определяем точку  $x_i$  по формуле

$$x_i = \left( \sum_{x_u \in X'_i} x_u \right) \times (|X'_i|)^{-1}$$

**Шаг 2.** С помощью следующей функции попарного сравнения объектов из  $X$

$$s: X \times X \rightarrow \mathbb{R}$$

для множества классов из  $X'$  по соответствующим точкам  $x_i$  находим близость между классами, выбираем ближайшие  $X'_s$  и  $X'_t$ . Объединяем их в новый класс  $X'_{st}$ .

В качестве  $s$  можно использовать любую метрику, включая описанные ниже метрики Евклида, Минковского или Хэмминга

**Шаг 3.** После этого модифицируем набор классов  $X'$  исключая из него  $X'_s$ ,  $X'_t$  и добавляя  $X'_{st}$ .

**Шаг 4.** Если  $|X'| = l$ , то алгоритм заканчивает работу. В противном случае переходим к шагу 1.

**Полученные результаты.**

При решении задачи  $T_1$  мы получим набор классов  $X' = (X'_1, \dots, X'_l)$ . При этом каждый объект из  $X^0$  будет занесен в один из классов набора  $X'$ . Можно этому результату сопоставить матрицу из **кластеризационных векторов**  $C(x) = (C_1(x), \dots, C_l(x))$ , компоненты которого могут быть определены следующим образом:

$$C_i(x) = \begin{cases} 1, & \text{если } x \in X'_i, \\ 0, & \text{иначе.} \end{cases}$$

Далее, имея кластеризационную матрицу и матрицу, состоящую из информационных векторов, можно определить **меру несоответствия** информации для задач  $T_0$  и  $T_1$ .

Эти задачи сравнимы, т.к. в них совпадает число классов. Для вычисления данной меры можно воспользоваться, к примеру, следующим алгоритмом.

**Шаг 0** (предварительный).

Каждому объекту  $x \in X^0$  сопоставляем пару  $(a_0(x), a_1(x))$ , где  $a_0(x), a_1(x)$  – номера классов для объекта  $x$  по информационному и кластеризационному векторам соответственно. Далее составляем  $l$  наборов из таких пар таким образом, чтобы в каждом из этих наборов первые компоненты пар совпадали. Обозначим эти наборы через  $A_1, \dots, A_l$ . Для определенности будем считать, что в  $A_i$  входят все пары, соответствующие классу с первым индексом равным  $i$ .

**Шаг 1.** Для каждого  $A_i$  по второму индексу  $a_1(x)$  вычисляем максимальное число вхождений числа из  $L = \{1, 2, \dots, l\}$ . Обозначим полученные максимумы через  $b_1, \dots, b_l$ , а соответствующие им номера классов через  $c_1, \dots, c_l$ .

Введем следующие множества:  $A = \{A_1, \dots, A_l\}$ ,  $B = \{b_1, \dots, b_l\}$ ,  $C = \{c_1, \dots, c_l\}$ ,  $B_0 = \emptyset$ ,  $C_0 = \emptyset$ .

**Шаг 2.** Находим максимальный элемент в  $B \setminus B_0$ . Пусть это будет  $b_u$ , а соответствующий номер класса  $c_u$ . Если  $c_u \in C_0$ , то переходим на шаг 2.1. В противном случае на шаг 2.2.

**Шаг 2.1.** Для набора  $A_u$  находим новый максимум  $b_u$  по номерам классов, не содержащихся в  $C_0$ . Если максимум не может быть найден (отсутствуют среди  $a_1(x)$  номера, не принадлежащие  $C_0$ ), то меняем в векторах  $b_1, \dots, b_l$  и  $c_1, \dots, c_l$  соответствующие компоненты  $b_u$  и  $c_u$  на 0,  $A_0 \stackrel{\text{def}}{=} A_0 \cup A_u$  и возвращаемся на шаг 2. В противном случае меняем  $b_u$  и  $c_u$  на новые значения и возвращаемся на шаг 2.

**Шаг 2.2.**  $B_0 \stackrel{\text{def}}{=} B_0 \cup b_u$ ,  $C_0 \stackrel{\text{def}}{=} C_0 \cup c_u$ ,  $A_0 \stackrel{\text{def}}{=} A_0 \cup A_u$ . Если  $|A_0| = l$  то алгоритм заканчивает работу (переходим на шаг 3). В противном случае возвращаемся на шаг 2.

**Шаг 3.** Фиксируем последовательность номеров классов в  $A_0$ . Пусть это будет такая последовательность  $A'_1, \dots, A'_l$ . Перенумеруем также номера классов  $X^0$  через  $X'_1, \dots, X'_l$  и соответствующие им максимумы через  $b'_1, \dots, b'_l$ . Подсчитываем **меру несоответствия** информации для задач **T<sub>0</sub>** и **T<sub>1</sub>** следующим образом:

$$\mu(T_0, T_1) = \left( \sum_{i=1}^l (|X'_i| - b'_i) \right) \times |X^0|^{-1}$$

**Требования к программному продукту:**

1. Любой язык программирования
2. Графический интерфейс **обязателен**;
3. Вывод на экран по запросу всей информации, которая описывает полученное решение. В данной задаче это разбиение  $X' = (X'_1, \dots, X'_l)$  и **мера несоответствия**  $\mu(T_0, T_1)$ .

# ЗАДАЧА РАСПОЗНАВАНИЯ С ОБУЧЕНИЕМ

## Шаг 0 (предварительный)

На входе имеем две задачи  $T_0$  и  $T_1$ . В каждой из этих задач имеется своя выборка объектов  $X^0$ , разбитая на  $l$  классов и соответствующие им информационные вектора. К каждой из этих выборок применяем одинаковую схему решения, описанную ниже.

### Разбиение выборки $X^0$ на две части

Шаг может повторяться некоторое число раз. Каждый раз выборка  $X^0$  разбивается на две части  $X_{обуч}^0$  и  $X_{контр}^0$  (называются соответственно **обучающей** и **контрольной** выборками). Выборки должны удовлетворять следующим условиям:

$$\begin{aligned} X_{i(o)}^0 &\stackrel{\text{def}}{=} X_{обуч}^0 \cap X_i^0 \neq \emptyset \quad \forall i \in \{1, \dots, l\}, \quad \bigcup_{i=1}^l X_{i(o)}^0 = X_{обуч}^0; \\ X_{i(к)}^0 &\stackrel{\text{def}}{=} X_{контр}^0 \cap X_i^0 \neq \emptyset \quad \forall i \in \{1, \dots, l\}, \quad \bigcup_{i=1}^l X_{i(к)}^0 = X_{контр}^0. \end{aligned}$$

Обозначим:  $|X_{i(o)}^0| = m_i$ ,  $|X_{i(к)}^0| = t_i$ ,  $\sum_{i=1}^l m_i = |X_{обуч}^0| = m$ ,  $\sum_{i=1}^l t_i = |X_{контр}^0| = t$ .

В результате шага 0 может быть получено два разбиения, которые мы обозначим через

$$\{X_{обуч}^0, X_{контр}^0\}^1, \{X_{обуч}^0, X_{контр}^0\}^2.$$

При этом первое разбиение сопоставляется задаче  $T_0$ , а второе – задаче  $T_1$ .

### Ограничение:

Все выборки  $\{X_{обуч}^0, X_{контр}^0\}^j$  должны удовлетворять следующему дополнительному условию:  $t_i / m_i \geq 0.2$ . После построения всех выборок  $\{X_{обуч}^0, X_{контр}^0\}^j$ , они фиксируются для последующего применения на каждой из них всего набора алгоритмов  $A$

Далее алгоритм  $A$  реализуется как последовательность шагов 1-3.

**Шаг 1.** Определение функции попарного сравнения объектов из  $X$ :

$$s: X \times X \rightarrow \mathbb{R} \tag{1}$$

Каждый объект  $x \in X$  можно сравнить с объектами из выборки  $X_{обуч}^0$ . В результате объекту  $x$  можно сопоставить вектор  $(s(x, x_1), \dots, s(x, x_m))$   $m_1$  первых компонент являются результатом сравнения  $x$  с объектами из  $X_{1(o)}^0$ ,  $m_2$  следующих являются результатом сравнения  $x$  с объектами из  $X_{2(o)}^0$  и т.д.

**Шаг 2.** Определение функции сравнения объектов из  $x \in X$  с объектами из обучающей выборки  $X_{i(o)}^0$ :

$$f_i: \mathbb{R}^m \rightarrow \mathbb{R}, \quad i \in \{1, \dots, l\} \tag{2}$$

С помощью функций (2) каждому вектору  $(s(x, x_1), \dots, s(x, x_m)) \in \mathbb{R}^m$  можно сопоставить вектор  $(f_1(x), \dots, f_l(x)) \in \mathbb{R}^l$ .

**Шаг 3.** Определение решающего правила  $P^A$  в виде:

$$P^A: \mathbb{R}^l \rightarrow \mathbb{B}_2^l, \mathbb{B}_2 = \{0, 1\} \quad (3)$$

Вектор  $P^A(x) = (P_1^A(x), \dots, P_l^A(x))$  в отличие от вектора  $P(x)$  называют обычно **классификационным**, тк его значения интерпретируются следующим образом. Считается, что объект  $x \in X$  заносится (или не заносится) алгоритмом  $A$  в класс  $X_i$ , если  $P_i^A(x) = 1$  ( $= 0$ ).

**Шаг 4.** Тестирование определенного на шагах 1-3 алгоритма распознавания  $A$  на фиксированной выборке  $\{X_{\text{обуч}}^0, X_{\text{контр}}^0\}^j, j = 1, 2, \dots, k$ .

Инициализация: полагаем  $t^0(X_{\text{контр}}^0) = 0$ .

**Шаг 4.1.** Последовательно перебираем все объекты  $x \in X_{\text{контр}}^0$  и для каждого из них вычисляем:

- вектор  $(s(x, x_1), \dots, s(x, x_m))$  для всех  $x \in X_{\text{обуч}}^0$ ;
- вектор  $(f_1(x), \dots, f_l(x))$  для всех  $i \in \{1, \dots, l\}$ ;
- вектор  $P^A(x) = (P_1^A(x), \dots, P_l^A(x))$
- если  $P^A(x) = P(x)$ , то  $t^0(X_{\text{контр}}^0) = t^0(X_{\text{контр}}^0) + 1$  и переходим к пункту е). В противном случае  $t^0(X_{\text{контр}}^0)$  не меняется и переходим к пункту е).
- если не все объекты  $x \in X_{\text{контр}}^0$  исчерпаны, то выполняем шаг 4.1 для следующего объекта контрольной выборки. В противном случае переходим к шагу 4.2.

**Шаг 4.2.** Вычисляем

$$\Phi_A(X_{\text{контр}}^0) = \frac{t^0(X_{\text{контр}}^0)}{t}$$

величину **функционала качества**  $\Phi_A(X_{\text{контр}}^0) \in [0, 1]$  и заносим ее в таблицу:

|           |   |   |
|-----------|---|---|
| Разбиение |   |   |
| Алгоритм  | $\{X_{\text{обуч}}^0, X_{\text{контр}}^0\}^1$ | $\{X_{\text{обуч}}^0, X_{\text{контр}}^0\}^2$ |
| $A$       |   |   |

Если все выборки  $\{X_{\text{обуч}}^0, X_{\text{контр}}^0\}^j$  исчерпаны, то задача решена в полном объеме. В противном случае выбираем новую выборку  $\{X_{\text{обуч}}^0, X_{\text{контр}}^0\}^j$  и возвращаемся на шаг 4.

## ВАРИАНТЫ ВЫБОРА ФУНКЦИЙ ДЛЯ ШАГОВ (1)-(3) СХЕМЫ АЛГОРИТМОВ

**ВАРИАНТ I.** (для пространства  $X \subseteq \mathbb{R}^n$ )

Выбор функции (1).

- метрика Евклида



$$s(x_1, x_2) = \left( \sum_{i=1}^n (x_{1i} - x_{2i})^2 \right)^{1/2}$$

- метрика Минковского ( $p \in \mathbb{N}$ )

$$s(x_1, x_2) = \left( \sum_{i=1}^n (x_{1i} - x_{2i})^p \right)^{1/p}$$

Выбор функции (2).

- среднее (м.б. взвешенное) расстояние до класса  $X_i$

$$f_i(x) = (m_i)^{-1} \sum_{x_j \in X_{i(0)}^0} s(x, x_j)$$

- $k$  ближайших соседей

пусть для класса  $X_i$  получен набор  $s(X_{i(0)}^0) = \{s(x, x_1), \dots, s(x, x_{m_i})\}$ ;

переупорядочим набор  $s(X_{i(0)}^0)$  по **возрастанию** элементов и поставим ему в соответствие новый набор  $\bar{X}_{i(0)}^0$ , в котором содержится  $k$  первых элементов  $x_j \in X_{i(0)}^0$  из полученного в результате переупорядочения набора;

посчитаем среднее расстояние до класса по новому набору  $\bar{X}_{i(0)}^0$

$$f_i(x) = (k)^{-1} \sum_{x_j \in \bar{X}_{i(0)}^0} s(x, x_j)$$

- по минимальному расстоянию до объектов класса  $X_i$

$$f_i(x) = \min_{x_j \in X_{i(0)}^0} s(x, x_j)$$

Выбор решающего правила (3).

- по минимуму оценки до класса  $X_i$

$$P_i^A(x) = \begin{cases} 1, & \text{если } f_i(x) = \min_{i \in \{1, \dots, l\}} \{f_1(x), \dots, f_l(x)\}; \\ 0, & \text{в противном случае} \end{cases}$$

## ВАРИАНТ II. (для пространства $X \subseteq \mathbb{B}^n$ )

Выбор функции (1).

- метрика Хэмминга

$$s(x_1, x_2) = \sum_{i=1}^n |x_{1i} - x_{2i}|$$

Выбор функции (2).

- среднее (м.б. взвешенное) расстояние до класса  $X_i$

$$f_i(x) = (m_i)^{-1} \sum_{x_j \in X_{i(0)}^0} s(x, x_j)$$

- $k$  ближайших соседей

пусть для класса  $X_i$  получен набор  $s(X_{i(0)}^0) = \{s(x, x_1), \dots, s(x, x_{m_i})\}$ ;

переупорядочим набор  $s(X_{i(0)}^0)$  по **возрастанию** элементов и поставим ему в соответствие новый набор  $\bar{X}_{i(0)}^0$ , в котором содержится  $k$  первых элементов  $x_j \in X_{i(0)}^0$  из полученного в результате переупорядочения набора;

посчитаем среднее расстояние до класса по новому набору  $\bar{X}_{i(0)}^0$

$$f_i(x) = (k)^{-1} \sum_{x_j \in \bar{X}_{i(o)}^0} s(x, x_j)$$

- по минимальному расстоянию до объектов класса  $X_i$

$$f_i(x) = \min_{x_j \in \bar{X}_{i(o)}^0} s(x, x_j)$$

Выбор решающего правила (3).

- по минимуму оценки до класса  $X_i$

$$P_i^A(x) = \begin{cases} 1, & \text{если } f_i(x) = \min_{i \in \{1, \dots, l\}} \{f_1(x), \dots, f_l(x)\}; \\ 0, & \text{в противном случае} \end{cases}$$

### Требования к программному продукту:

1. Любой язык программирования
2. Графический интерфейс **обязателен**;
3. Вывод на экран по запросу всей информации, которая описывает полученное решение.