НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЯДЕРНЫЙ УНИВЕРСИТЕТ МИФИ

ЛЕКЦИЯ 8.
ЗАДАЧА КЛАССИФИКАЦИИ В OCR

Демидов Д.В.

Обработка аудиовизуальной информации
Бакалавры, 6 семестр. Магистры, 9 семестр

План лекции

- Постановка задачи классификации
- Алгоритмы интеллектуального анализа данных для классификации символов
- Алгоритмы коррекции распознанных текстов

Задача классификации

Классификация

Обучение классификатора

Анализ полноты и точности

Задача классификации

- Задано конечное множество объектов и конечное множество классов.
- Для каждого объекта известно к какому классу он относится.
- Требуется построить алгоритм, способный *классифицировать* (соотносить с классом) произвольный объект.
- Подходят методы обучения с учителем.
- Обычно объекты представляются точками в признаковом пространстве.

Разновидности задачи

- **Двухклассовая классификация.** Наиболее простой в техническом отношении случай, который служит основой для решения более сложных задач.
- **Многоклассовая классификация**. Число классов может достигать многих тысяч. Например, при распознавании иероглифов или слитной речи.
- **Непересекающиеся классы**. Объект относится строго к одному классу.
- **Пересекающиеся классы**. Объект может относиться одновременно к нескольким классам.
- **Нечёткие классы**. Объект относится к каждому классу с некоторой степенью принадлежности в интервале от 0 до 1.

Признаковое пространство

- *Признаком* называется отображение f: X→D_f, где D_f — множество допустимых значений признака.
- Если заданы признаки f_1, f_2,..., f_n, то
  - вектор ($f_1(x)$, ..., $f_n(x)$) называется признаковым описанием объекта x и таким образом может задавать объект.
  - Множество $X = D_{f_1} \times D_{f_2} \times ... \times D_{f_n}$ называют *признаковым пространством*.
- Признаки делятся на следующие типы:
  - *бинарный* признак: $D_f = \{0; 1\}$;
  - *номинальный* признак: D_f — конечное множество;
  - *порядковый* признак: D_f — конечное упорядоченное множество;
  - *количественный* признак: D_f — множество действительных чисел.

Задача кластеризации

- Задано конечное множество объектов.
- Множество классов не задано.
- Требуется *кластеризовать* объекты — сопоставить объекты с кластерами объектов.
- Методы обучения без учителя подходят, а методы обучения с учителем нет.

База образцов

- Каждый класс задаётся кодом Unicode и набором образцов.
- Например, класс «А» с кодом u0410 будет задан набором:

A: u0410	10			11			12			14			16		
Calibri	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*
Courier New	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*
Times new roman	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*
Arial	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*	A	**A**	*A*

- А класс «а» с кодом u0430 набором:

a: u0430	10			11			12			14			16		
Calibri	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*
Courier New	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*
Times new roman	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*
Arial	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*	a	**a**	*a*

- Для каждого образца рассчитывается признаковое описание и сохраняется в базе.

Простейший алгоритм классификации

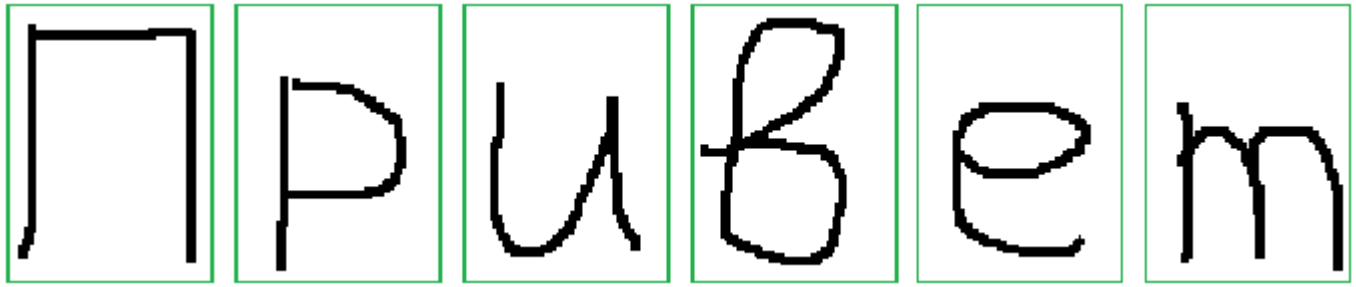
1. Для изображения неизвестного символа строится признаковое описание.
2. Рассчитывается мера близости неизвестного символа с каждым известным символом (образцом). Так формируется упорядоченное множество гипотез с оценками.
3. В методе ближайшего соседа выбирается класс ближайшего образца (лучшая гипотеза). В методе k ближайших соседей отбираются k гипотез с лучшими оценками. Каждая гипотеза голосует за свой класс, и классы ранжируются. Выбирается класс, набравший большинство голосов.

Пример результата классификации

- Результаты классификации отсортированы по убыванию степени принадлежности образца классам:
  1. П: 0.99, Л: 0.95, Д: 0.76, …
  2. р: 1.0, о: 0.68, ъ: 0.55, …
  3. и: 0.97, н: 0.82, п: 0.79, …
  4. в: 0.96, я: 0.77, б: 0.67, …
  5. е: 0.98, с: 0.96, о: 0.88, …
  6. т: 1.0, г: 0.92, п: 0.56, …
- В первом столбце читается распознанный текст

Колоночный текст

Сегментированная строка



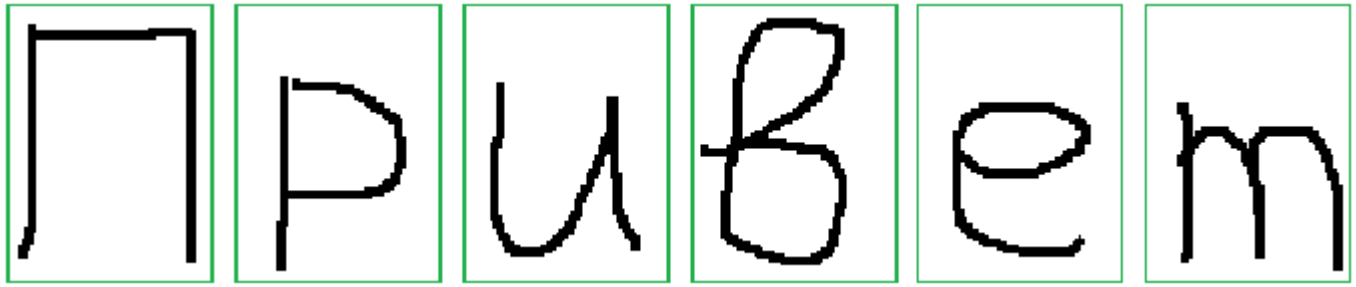
Все возможные гипотезы по убыванию меры близости

П 0.99	р 1.0	и 0.97	в 0.96	е 0.98	т 1.0
Л 0.95	о 0.68	н 0.82	я 0.77	с 0.96	г 0.92
Д 0.76	ъ 0.55	п 0.79	б 0.67	о 0.88	п 0.56
…	…	…	…	…	…

- Удобно для визуализации и оценки гипотез

Бывает и не так хорошо:

Сегментированная строка



Все возможные гипотезы по убыванию меры близости

Л 0.99	р 1.0	и 0.97	в 0.96	е 0.98	м 1.0
П 0.95	о 0.68	н 0.82	я 0.77	с 0.96	т 0.92
Д 0.76	ъ 0.55	п 0.79	б 0.67	о 0.88	п 0.56
…	…	…	…	…	…

- Правильные гипотезы не всегда самые первые

Генерация выходного текста

- Выходная строка инициализируется пустой строкой.
- Для каждого классифицированного образца:
  - Определяется лучшая гипотеза и извлекается код символа.
  - Выходная строка наращивается символом с этим кодом.
- Результат выводится пользователю.

Критерий уверенного распознавания

- Варианты:
  - Оценки гипотез для одного символа близки
  - Оценки гипотез для одного символа сильно различаются
- Что значит «близки»?
- Что значит «сильно различны»?

Лингвистическая коррекция текста

- Проверка получившихся слов по словарю
- Выбор лучшей цепочки гипотез из колоночного текста
  - Алгоритм Витерби + корпус n-грамм + хранилище MARISA-Trie.
  - N-граммы — цепочки из n символов. Например, для цепочки «Привет, мир!»:
    - Биграммы: «Пр», «ри», «ив», «ве», «ет», «т,», «, », « м», «ми», «ир», «р!»
    - Триграммы: «При», «рив», «иве», «вет», «ет,», «т, », «, м», « ми», «мир», «ир!»
    - Квадрограммы: «Прив», «риве», «ивет», «вет,», «ет, », «т, м», «, ми», « мир», «мир!»
    - и т.д.
  - Учитывается встречаемость n-грамм в текстах и оценки полученных гипотез. Среди возможных цепочек выбирается статистически наиболее правдоподобная

Оценка качества классификатора

- При классификации примеров из обучающей выборки нам всегда известен верный ответ.
- Для каждого примера по отношению к каждому классу имеется 4 варианта:

	Принадлежность объекта классу, предсказанная классификатором	
Фактическая принадлежность объекта классу	Верно отнесён	Неверно отброшен
	Неверно отнесён	Верно отброшен

Пропуск цели
Ошибка II рода

Ложная тревога
Ошибка I рода

Точность и полнота

- **Точность —** число примеров, верно отнесённых классификатором к данному классу, по отношению к общему числу примеров, отнесённых классификатором к этому классу:

$$Precision = P = \frac{ВернoОтнесённые}{ВернoОтнесённые + ЗряОтнесённые}$$

- **Полнота —** число примеров, верно отнесённых классификатором к данному классу, по отношению к общему числу примеров, принадлежащих этому классу:

$$Recall = R = \frac{ВернoОтнесённые}{ВернoОтнесённые + ЗряОтброшенные}$$

F-мера Ван Ризбергена

- Мера Ван Ризбергена (F-мера) — обобщение среднего гармонического на основе взвешенных точности и полноты:

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{P \cdot R}{\alpha R + (1-\alpha)P}.$$

- Или

$$F = (\beta^2 + 1)\frac{Precision \times Recall}{\beta^2 Precision + Recall} = \frac{P \cdot R}{\frac{\beta^2}{(\beta^2+1)}P + \frac{1}{(\beta^2+1)}R}$$

- При β=1 полнота и точность вносят равный вклад, а F-мера вырождается в среднее гармоническое:

$$F_1 = 2\frac{P \cdot R}{P + R}$$

Матрица неточностей

- *Матрица неточностей* — это матрица размера N на N, где N — количество классов. Фактические классы описываются строками.
- Для фактического класса *i*:
  - значение элемента (*i, i*) соответствует числу *верно отнесённых* примеров;
  - ненулевые элементы строки *i*, кроме *(i, i)*, соответствуют *зря отнесённым* к классу *i* примерам (ложная тревога);
  - ненулевые элементы столбца *i*, кроме *(i, i)*, соответствуют *зря отброшенным, не узнанным* примерам класса *i* (пропуск цели);
  - остальные ненулевые элементы матрицы соответствуют верно отброшенным примерам по отношению к классу *i*.
- Матрица неточностей позволяет определить наиболее проблемные классы.

Пример матрицы неточностей

Точность Полнота

Предсказанные данные

Ложноположительные, зря отнесли к А

Фактические данные

	0.91	0.96	0.94	0.75	1.00	0.83	0.85	0.97	1.00	0.86	1.00	0.79	1.00	0.75	1.00	1.00	0.96	0.90	0.81	0.89	0.94	0.98	0.86	0.89	0.94	0.92	0.96
0.80		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
0.95	1	94	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
1.00	2	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.29	3	0	0	6	0	0	3	2	0	1	0	0	0	0	0	0	1	1	0	0	1	0	1	3	0	2	0
1.00	4	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.50	5	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	0	1	1
0.92	6	1	0	0	0	0	152	0	0	1	0	0	0	0	0	0	0	1	4	2	3	0	0	0	0	2	0
0.97	7	1	0	1	0	0	0	256	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	2	0
0.33	8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
0.97	9	0	0	0	0	0	0	0	0	69	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
0.82	10	0	0	0	0	0	2	0	0	0	18	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
0.87	11	0	0	0	0	0	0	0	0	0	0	34	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0
1.00	12	0	0	0	0	0	0	0	0	0	0	0	37	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.57	13	0	0	0	0	0	0	0	0	0	0	9	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0
0.63	14	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	3	0	0	0	0	0	0	0	0	0
0.50	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1	1	0	0	0	0	0	0
0.77	16	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	47	0	1	3	4	0	0	2	0	1	0
0.87	17	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	69	1	2	5	0	0	0	0	0	0
0.97	18	0	0	0	0	1	4	0	0	1	0	0	0	0	0	0	0	0	197	1	0	0	0	0	0	0	0
0.78	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	35	183	13	0	0	2	0	1	0
0.97	20	0	0	0	0	0	10	3	0	1	0	0	0	0	0	0	0	0	0	4	702	0	0	0	0	6	0
0.93	21	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	56	0	2	0	0	0
0.29	22	0	0	1	0	0	2	0	0	6	0	0	0	0	0	0	0	0	1	1	1	0	6	2	0	1	0
0.91	23	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	3	6	0	0	115	0	0	0
1.00	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0
0.93	25	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2	4	5	0	0	0	1	196	0
0.98	26	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	78

Ложноотрицательные, не узнали класс А

Всего классов: 26
Точность: 0.8
Полнота: 0.91

Улучшение классификатора

- С чем бороться в первую очередь?
  - С большим количеством ошибок по классу;
  - С большим количеством ошибок в одной ячейке;
  - С остальными ошибками.
- Как бороться?
  - Добавлять признаки, которые потенциально могут разделить часто путаемые символы;
  - Добавить признаки пачками наудачу, оценивая их влияние на матрицу неточностей.

Построение классификаторов методом обучения с учителем

Обучение с учителем

- *Размеченные данные* — входные данные, для которых указаны выходные данные.
- При обучении с учителем набор размеченных данных разбивается на две выборки:
  - *Обучающая выборка* (training set) используется для обучения (конструирования) модели.
  - *Тестовая выборка* (test set) используется для проверки работы построенной модели.

Конструирование модели

- На основе сопоставленных входных и выходных данных с помощью некоторого алгоритма строится модель.
- Модель, как правило, обобщает имеющиеся в виде обучающей выборки сведения и может быть представлена:
  - Классифицирующими правилами,
  - Деревом (деревьями) решений,
  - Математической формулой.
- Полученная модель должна классифицировать примеры из обучающей выборки максимально точно и полно.

Оценка модели

- С помощью тестовой выборки можно предсказать поведение модели на неизвестных данных.
- Благодаря тому, что тестовая выборка также размечена, получают количественные оценки качества модели:
  - Интегральные оценки: точность, полнота, F-мера
  - Количество ошибок I и II рода по каждому классу.

Некоторые алгоритмы построения классификаторов

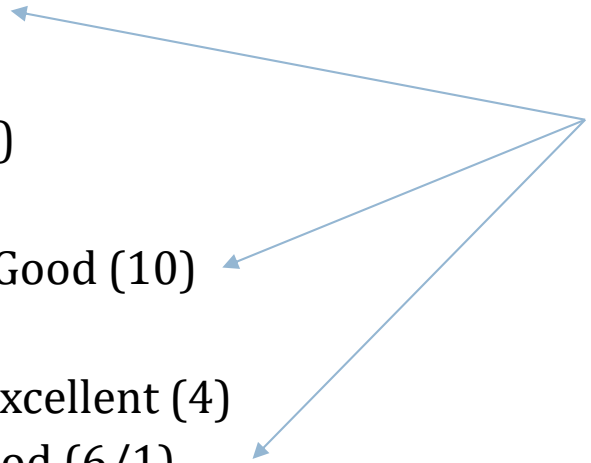
- ID3 — алгоритм построения дерева принятия решений, основанный на оценке энтропии признаков.
- C4.5 — усовершенствованный ID3 с отсечением ветвей, возможностью работы с числовыми атрибутами, возможностью построения дерева из неполной обучающей выборки, в которой отсутствуют значения некоторых атрибутов.
- C5 — усовершенствованный C4.5, детали реализации которого не раскрываются.

Джон Росс Куинлан

Пример модели, построенной C5

Дерево решений:

```
noise3 <= 9e-005: Excellent (8)
noise3 > 9e-005:
:...noise2 > 0.05643: Satisfactory (4/2)
    noise2 <= 0.05643:
    :...crosses whites <= 0: Good (14/2)
        crosses whites > 0:
        :...noise3 <= 0.00048: Excellent (4)
            noise3 > 0.00048:
            :...isolated blacks <= 0.000544: Good (10)
                isolated blacks > 0.000544:
                :...crosses blacks <= 6e-006: Excellent (4)
                    crosses blacks > 6e-006: Good (6/1)
```



Образцов отнесено/ число ошибок

Пример оценки классификатора

- 4 класса
- 50 образцов в тестовой выборке
- Количество ошибок: 5/50
- Построена матрица неточностей 4x4, где видны все ошибки
- Посчитан процент использования каждого признака при классификации

```
Evaluation on training data (50 cases):
        Decision Tree
      ----------------
      Size      Errors   Cost
         7    5(10.0%)   0.10   <<
       (a)   (b)   (c)   (d)    <-classified as
      ----  ----  ----  ----
        16     1                (a): class Excellent
              27     1          (b): class Good
               2     2          (c): class Satisfactory
                     1          (d): class Poor
Attribute usage:
    100%  noise3
     84%  noise2
     76%  crosses whites
     40%  isolated blacks
     20%  crosses blacks
```

Что почитать

- **Маннинг К.Д., Рагхаван П., Шютце Х**. Введение в информационный поиск. — Пер. с англ. — М.: ООО «И.Д. Вильямс», 2011. — 528 с.
- Алгоритм Витерби
  - **Viterbi, A. J.** “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” IEEE Trans. Inform. Theory, vol. IT–13, pp. 260–269, April 1967.
  - https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%92%D0%B8%D1%82%D0%B5%D1%80%D0%B1%D0%B8
- Алгоритмы ID3 (Iterative Dichotomiser 3) , C4.5, C5
  - **Quinlan, J. R.** 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81–106
  - **Quinlan, J. R.** 1993. C4.5: Programs for machine learning San Mateo, Calif. : Morgan Kaufmann Publishers. Pp. 302.
  - **Quinlan, J. R.** Improved Use of Continuous Attributes in C4.5 // Journal of Artificial Intelligence Research. 1996. Vol. 4. P. 77–90. ISSN 1076-9757
  - https://en.wikipedia.org/wiki/ID3_algorithm
  - Алгоритм C5 https://www.rulequest.com/see5-info.html
- Matching Algorithm with Recursively Implemented StorAge (MARISA)
  - https://www.s-yata.jp/marisa-trie/docs/readme.en.html
  - https://github.com/s-yata/marisa-trie