Лекция 1. Различные задачи машинного обучения

Д. П. Ветров[1]

[1]МГУ, ВМиК, каф. ММП

Курс «Математические основы теории прогнозирования»

План лекции

Концепция машинного обучения

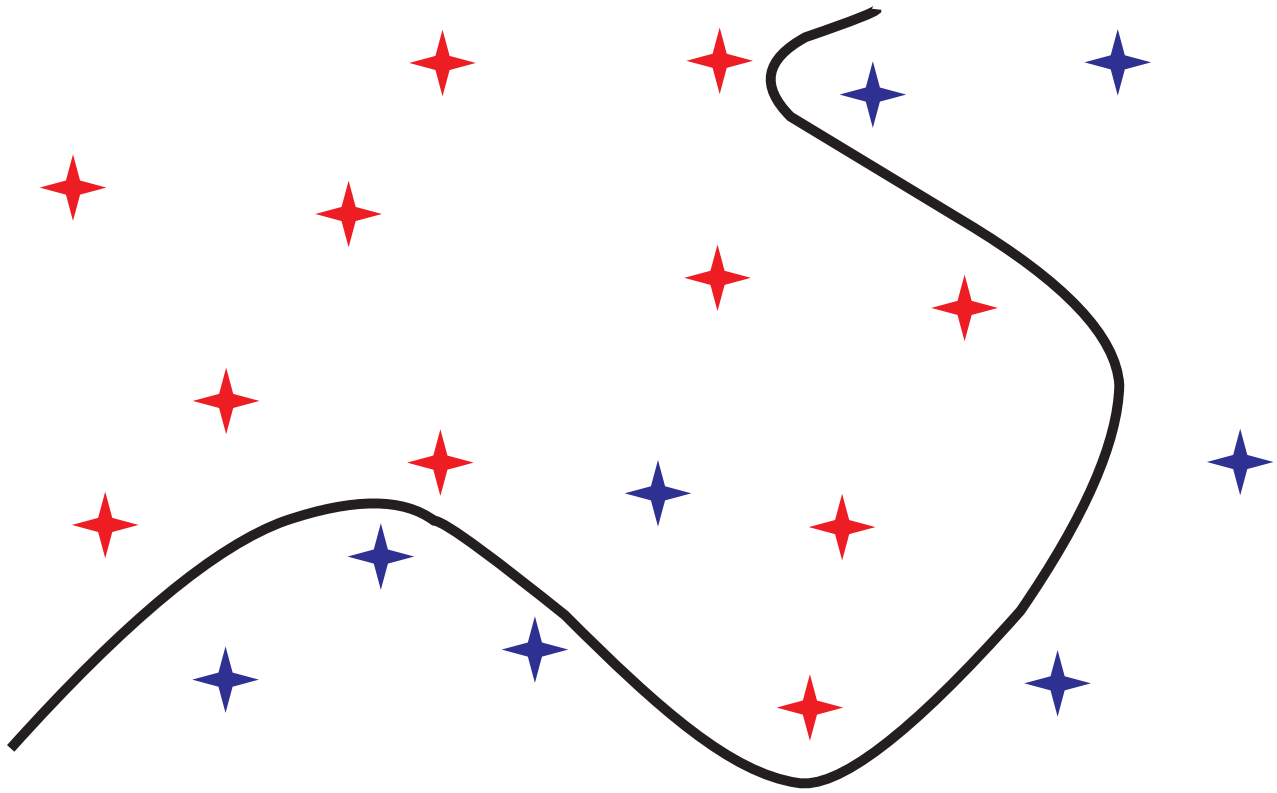
- Решение задач путем обработки прошлого опыта (case-based reasoning)
- Альтернатива построению математических моделей (model-based reasoning)
- Основное требование – наличие обучающей информации
- Как правило в качестве таковой выступает выборка прецедентов – ситуационных примеров из прошлого с известным исходом
- Требуется построить алгоритм, который позволял бы обобщить опыт прошлых наблюдений/ситуаций для обработки новых, не встречавшихся ранее случаев, исход которых неизвестен.

План лекции

Классификация

- Исторически возникла из задачи машинного зрения, поэтому часто употребляемый синоним – распознавание образов
- В классической задаче классификации обучающая выборка представляет собой набор отдельных объектов $X = \{\boldsymbol{x}_i\}_{i=1}^{n}$, характеризующихся вектором вещественнозначных признаков $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,d})$
- В качестве исхода объекта $\boldsymbol{x}$ фигурирует переменная t, принимающая конечное число значений, обычно из множества $\mathcal{T} = \{1, \ldots, l\}$
- Требуется постросить алгоритм (классификатор), который по вектору признаков $\boldsymbol{x}$ вернул бы метку класса $\hat{t}$ или вектор оценок принадлежности (апотериорных вероятностей) к каждому из классов $\{p(s|\boldsymbol{x})\}_{s=1}^{l}$

Классификация



Примеры задач классификации

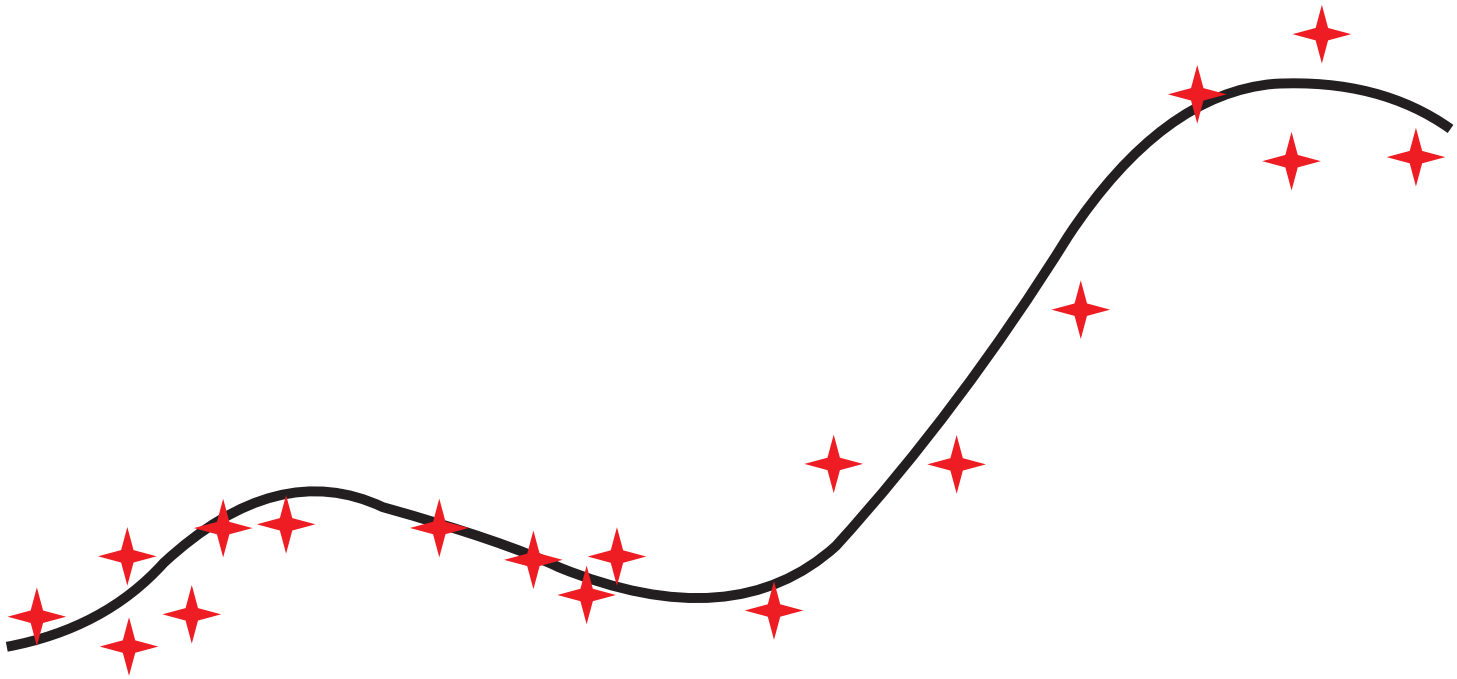
- Медицинская диагностика: по набору медицинских характеристик требуется поставить диагноз
- Геологоразведка: по данным зондирования почв определить наличие полезных ископаемых
- Оптическое распознавание текстов: по отсканированному изображению текста определить цепочку символов, его формирующих
- Кредитный скоринг: по анкете заемщика принять решение о выдаче/отказе кредита
- Синтез химических соединений: по параметрам химических элементов спрогнозировать свойства получаемого соединения

План лекции

Регрессия

- Исторически возникла при исследовании влияния одной группы непрерывных случайных величин на другую группу непрерывных случайных величин
- В классической задаче восстановления регрессии обучающая выборка представляет собой набор отдельных объектов $X = \{\boldsymbol{x}_i\}_{i=1}^{n}$, характеризующихся вектором вещественнозначных признаков $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,d})$
- В качестве исхода объекта $\boldsymbol{x}$ фигурирует непрерывная вещественнозначная переменная t
- Требуется постросить алгоритм (регрессор), который по вектору признаков $\boldsymbol{x}$ вернул бы точечную оценку значения регрессии $\hat{t}$, доверительный интервал (t_-, t_+) или апостериорное распределение на множестве значений регрессионной переменной $p(t|\boldsymbol{x})$

Регрессия



Примеры задач восстановления регрессии

- Оценка стоимости недвижимости: по характеристике района, экологической обстановке, транспортной связности оценить стоимость жилья
- Прогноз свойств соединений: по параметрам химических элементов спрогнозировать температуру плавления, электропроводность, теплоемкость получаемого соединения
- Медицина: по постоперационным показателям оценить время заживления органа
- Кредитный скоринг: по анкете заемщика оценить величину кредитного лимита
- Инженерное дело: по техническим характеристикам автомобиля и режиму езды спрогнозировать расход топлива

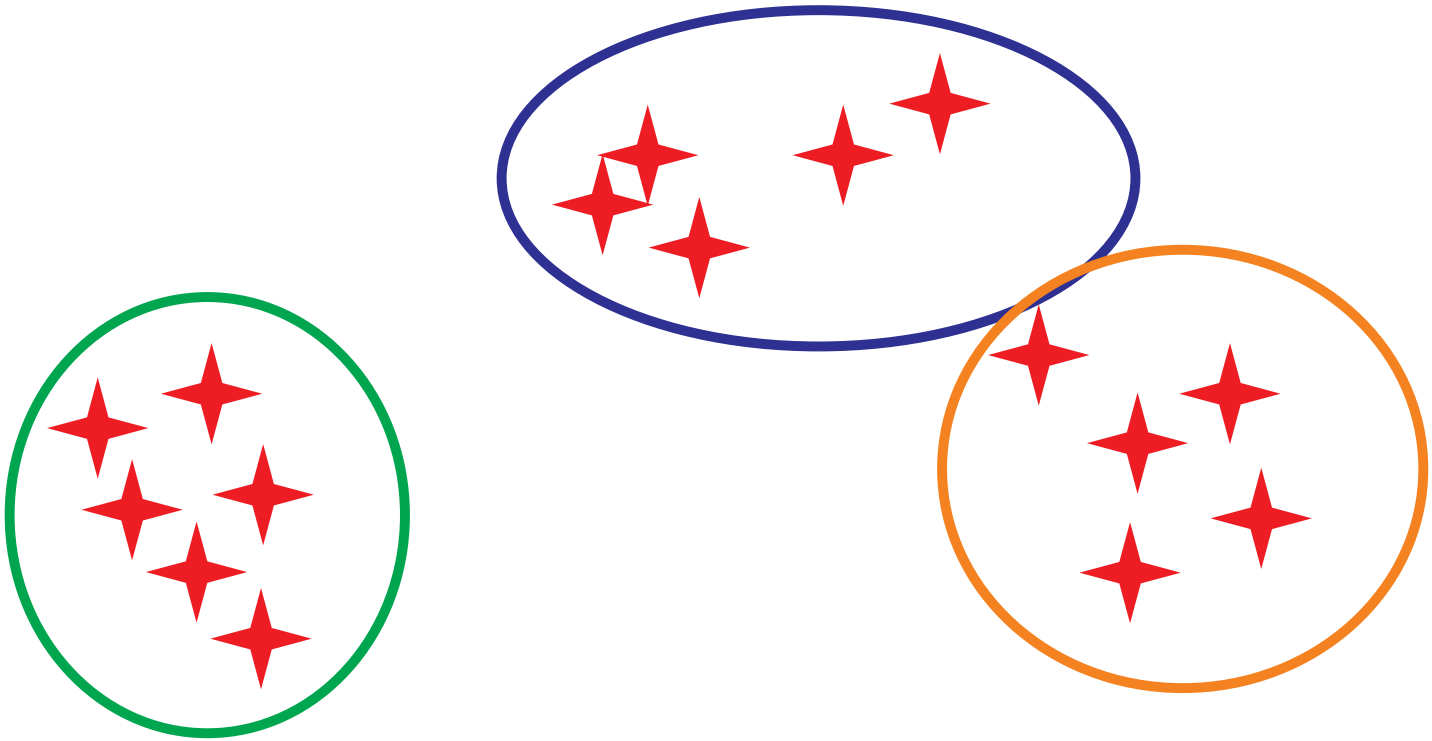
План лекции

1. Некоторые задачи машинного обучения
   - Задача классификации
   - Задача восстановления регрессии
   - Задача кластеризации (обучения без учителя)
   - Задача идентификации
   - Задача прогнозирования
   - Задача извлечения знаний
2. Основные проблемы машинного обучения
   - Малый объем обучающей выборки
   - Некорректность входных данных
   - Переобучение
3. Напоминание
   - Полезные сведения из линейной алгебры и теории оптими
   - Основные понятия мат. статистики

Кластеризация

- Исторически возникла из задачи группировки схожих объектов в единую структуру (кластер) с последующим выявлением общих черт
- В классической задаче кластеризации обучающая выборка представляет собой набор отдельных объектов $X = \{\boldsymbol{x}_i\}_{i=1}^{n}$, характеризующихся вектором вещественнозначных признаков $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,d})$
- Требуется постросить алгоритм (кластеризатор), который разбил бы выборку на непересекающиеся группы (кластеры) $X = \bigcup_{j=1}^{k} C_k$, $C_j \subset \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$, $C_i \cap C_j = \emptyset$
- В каждый класс должны попасть объекты в некотором смысле похожие друг на друга

Кластеризация



Примеры задач кластерного анализа

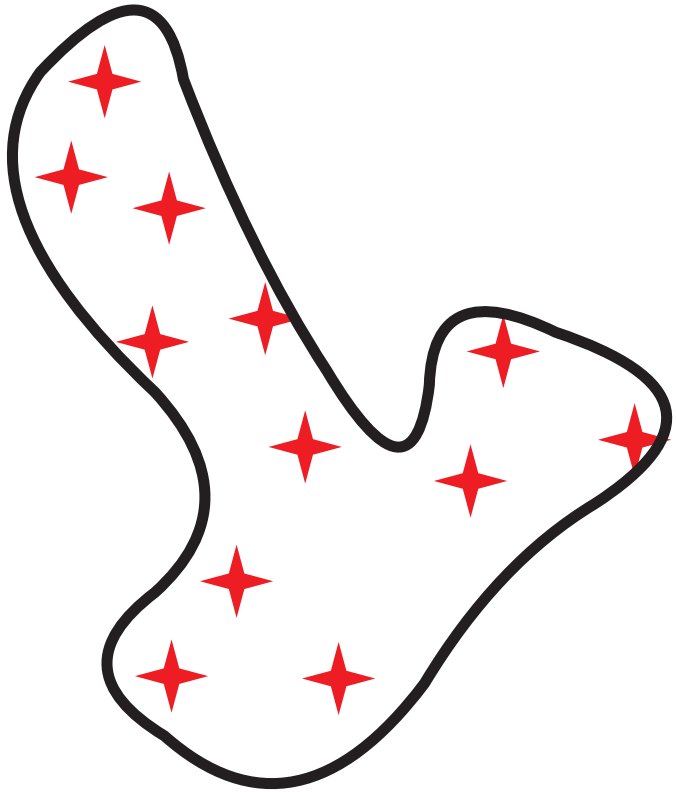
- Экономическая география: по физико-географическим и экономическим показателям разбить страны мира на группы схожих по экономическому положению государств
- Финансовая сфера: по сводкам банковских операций выявить группы «подозрительных», нетипичных банков, сгуппировать остальные по степени близости проводимой стратегии
- Маркетинг: по результатам маркетинговых исследований среди множества потребителей выделить характерные группы по степени интереса к продвигаемому продукту
- Социология: по результатам социологических опросов выявить группы общественных проблем, вызывающих схожую реакцию у общества, а также характерные фокус-группы населения

План лекции

Идентификация

- Исторически возникла из классификации, необходимости отделить объекты, обладающие определенным свойством, от «всего остального»
- В классической задаче идентификации обучающая выборка представляет собой набор отдельных объектов $X = \{\boldsymbol{x}_i\}_{i=1}^{n}$, характеризующихся вектором вещественнозначных признаков $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,d})$, обладающих некоторым свойством $\chi_A(\boldsymbol{x}) = 1$
- Особенностью задачи является то, что все объекты принадлежат одному классу, причем не существует возможности сделать репрезентативную выборку из класса «все остальное»
- Требуется построить алгоритм (идентификатор), который по вектору признаков $\boldsymbol{x}$ определил бы наличие свойства A у объекта $\boldsymbol{x}$, либо вернул оценку степени его выраженности $p(\chi_A(\boldsymbol{x}) = 1|\boldsymbol{x})$

Идентификация



Примеры задач идентификации

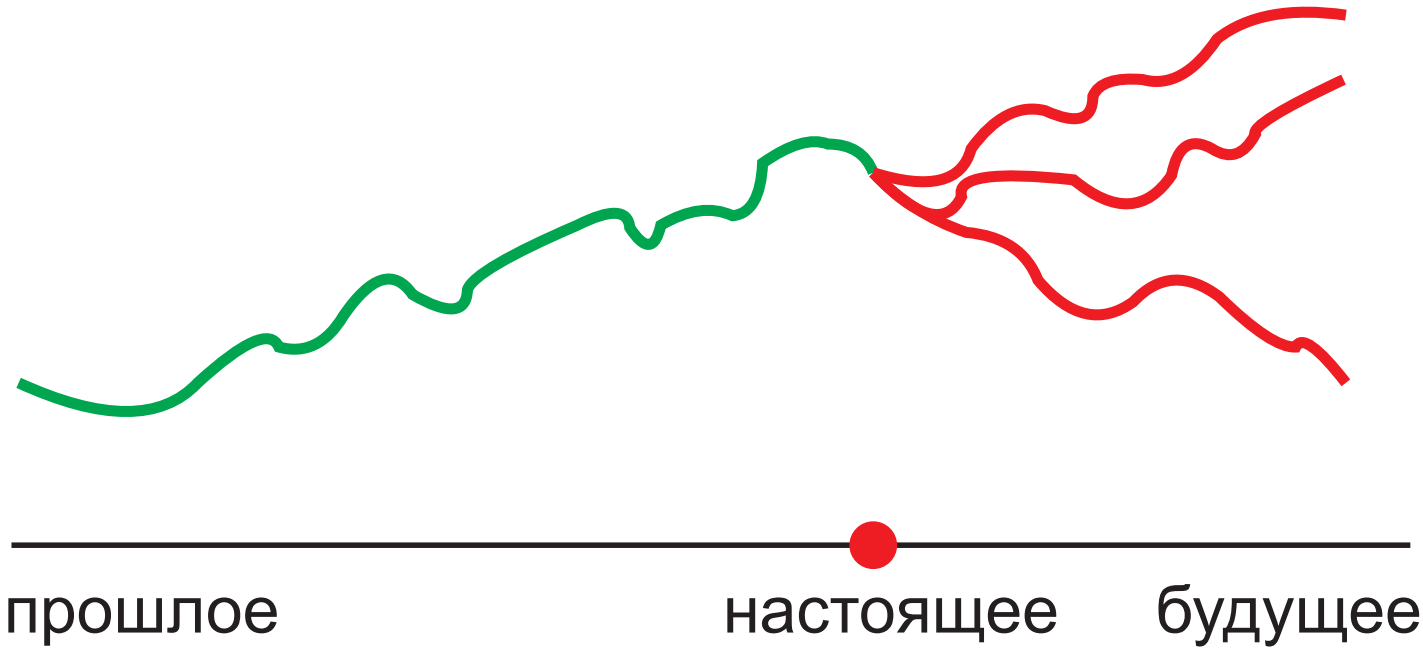
- Медицинская диагностика: по набору медицинских характеристик требуется установить наличие/отсутствие конкретного заболевания
- Системы безопасности: по камерам наблюдения в подъезде идентифицировать жильца дома
- Банковское дело: определить подлинность подписи на чеке
- Обработка изображений: выделить участки с изображениями лиц на фотографии
- Искусствоведение: по характеристикам произведения (картины, музыки, текста) определить, является ли его автором тот или иной автор

План лекции

Прогнозирование

- Исторически возникла при исследовании временных рядов и попытке предсказания их значений через какой-то промежуток времени
- В классической задаче прогнозирования обучающая выборка представляет собой набор измерений $X = \{\boldsymbol{x}[i]\}_{i=1}^{n}$, представляющих собой вектор веществennозначных величин $\boldsymbol{x}[i] = (x_1[i], \ldots, x_d[i])$, сделанных в определенные моменты времени
- Требуется постросить алгоритм (предиктор), который вернул бы точечную оценку $\{\hat{\boldsymbol{x}}[i]\}_{i=n+1}^{n+q}$, доверительный интервал $\{(\boldsymbol{x}_-[i], \boldsymbol{x}_+[i])\}_{i=n+1}^{n+q}$ или апостериорное распределение $p(\boldsymbol{x}[n+1], \ldots, \boldsymbol{x}[n+q] | \boldsymbol{x}[1], \ldots, \boldsymbol{x}[n])$ прогноза на заданную глубину q
- В отличие от задачи восстановления регрессии, здесь осуществляется прогноз по времени, а не по признакам

Прогнозирование



Примеры задач прогнозирования

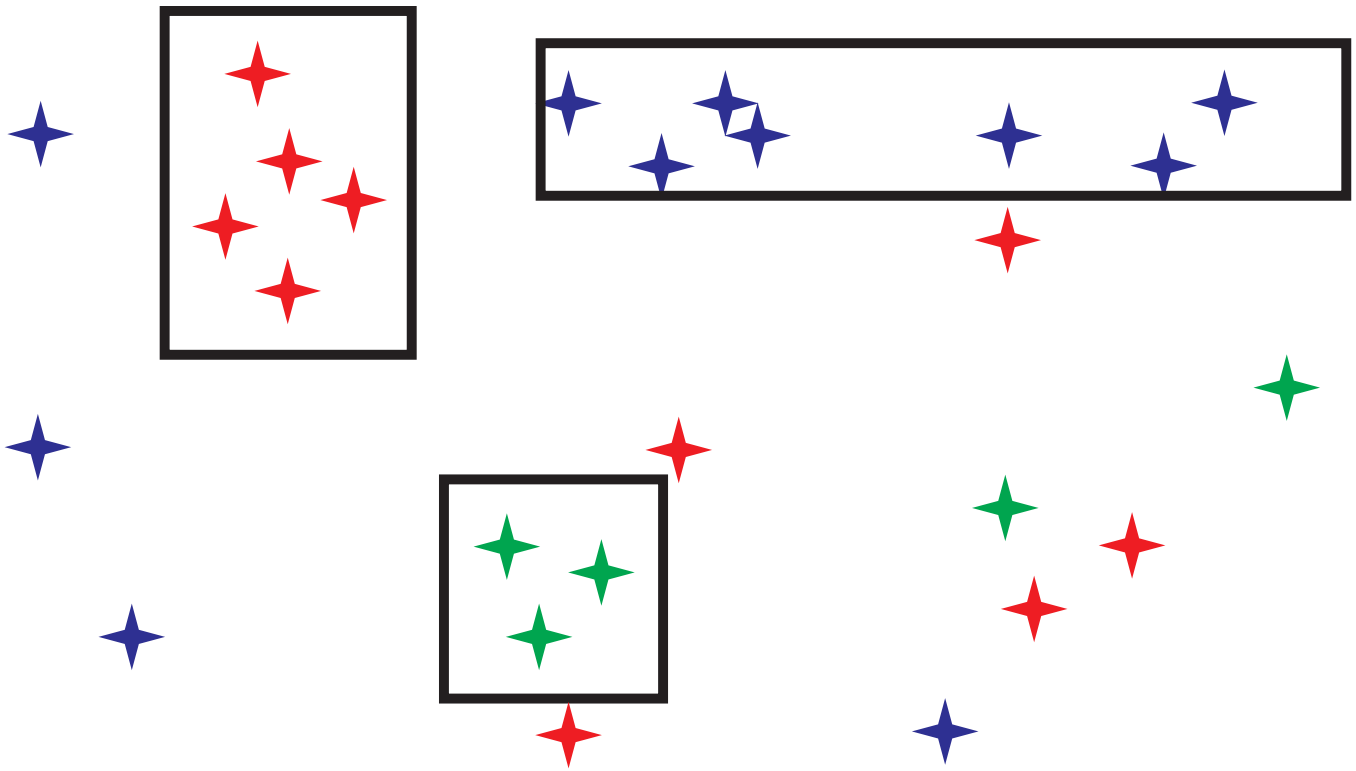
- Биржевое дело: прогнозирование биржевых индексов и котировок
- Системы управления: прогноз показателей работы реактора по данным телеметрии
- Экономика: прогноз цен на недвижимость
- Демография: прогноз изменения численности различных социальных групп в конкретном ареале
- Гидрометеорология: прогноз геомагнитной активности

План лекции

Извлечение знаний

- Исторически возникла при исследовании взаимозависимостей между косвенными показателями одного и того же явления
- В классической задаче извлечения знаний обучающая выборка представляет собой набор отдельных объектов $X = \{\boldsymbol{x}_i\}_{i=1}^{n}$, характеризующихся вектором вещественнозначных признаков $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,d})$
- Требуется постросить алгоритм, генерирующий набор объективных закономерностей между признаками, имеющих место в генеральной совокупности
- Закономерности обычно имеют форму предикатов «ЕСЛИ ... ТО ...» и могут выражаться как в цифровых терминах $((0.45 \leq x_4 \leq 32.1)\&(-6.98 \leq x_7 \leq -6.59) \Rightarrow (3.21 \leq x_2 \leq 3.345))$, так и в текстовых («ЕСЛИ Давление – низкое И (Реакция – слабая ИЛИ Реакция – отсутствует) ТО Пульс – нитевидный»)

Извлечение знаний



Примеры задач извлечения знаний

- Медицина: поиск взаимосвязей (синдромов) между различными показателями при фиксированной болезни
- Социология: определение факторов, влияющих на победу на выборах
- Генная инженерия: выявление связанных участков генома
- Научные исследования: получение новых знаний об исследуемом процессе
- Биржевое дело: определение закономерностей между различными биржевыми показателями

План лекции

Объем выборки I

- Основным объектом работы любого метода машинного обучения служит обучающая выборка
- Большой объем выборки позволяет
  - Получить более надежные результаты
  - Использовать более сложные модели алгоритмов
  - Оценить точность обучения
  - НО: Время обучения быстро растет
- При малых выборках
  - Можно использовать только простые модели алгоритмов
  - Скорость обучения максимальна – можно использовать методы, требующие много времени на обучение
  - Высока вероятность переобучения при ошибке в выборе модели

Объем выборки II

- Одна и та же выборка может являться большой для простых моделей алгоритмов и малой для сложных моделей.
- С ростом числа признаков увеличивается количество объектов, необходимое для корректного анализа данных
- Часто рассматривается т.н. эффективная размерность выборки $\frac{n}{d}$
- При объемах данных порядка десятков и сотен тысяч встает проблема уменьшения выборки с сохранением ее репрезентативности (active learning)

План лекции

Неполнота признакового описания

- Отдельные признаки могут отсутствовать у некоторых объектов. Это может быть связано с отсутствием данных об измерении данного признака для данного объекта, а может быть связано с принципиальным отсутствием данного свойства у данного объекта
- Такое часто встречается в медицинских и химических данных
- Необходимы специальные процедуры, позволяющие корректно обрабатывать пропуски в данных
- Одним из возможных способов такой обработки является замена пропусков на среднее по выборке значение данного признака
- По возможности, пропуски следует игнорировать и исключать из рассмотрения при анализе соответствующего объекта

Противоречивость данных

- Объекты с одним и тем же признаковым описанием могут иметь разные исходы (принадлежать к разным классам, иметь отличные значения регрессионной переменной и т.п.)
- Многие методы машинного обучения не могут работать с такими наборами данных
- Необходимо заранее исключать или корректировать противоречащие объекты
- Использование вероятностных методов обучения позволяет корректно обрабатывать противоречивые данные
- При таком подходе предполагается, что исход t для каждого признакового описания $\boldsymbol{x}$ есть случайная величина, имеющая некоторое условное распределение $p(t|\boldsymbol{x})$

Разнородность признаков

- Хотя формально предполагается, что признаки являются вещественнозначными, они могут быть дискретными и номинальными
- Номинальные признаки отличаются особенностями метрики между значениями
- Стандартная практика состоит в замене номинальных признаков на набор бинарных переменных по числу значений номинального признака
- Текстовые признаки, признаки-изображения, даты и пр. необходимо заменить на соответствующие номинальные либо числовые значения

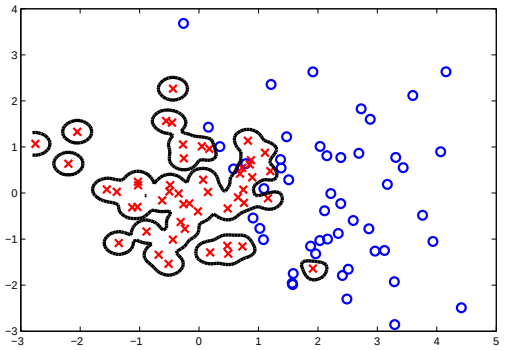
План лекции

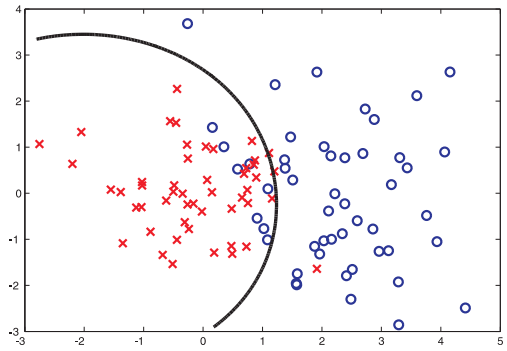
Идея машинного обучения

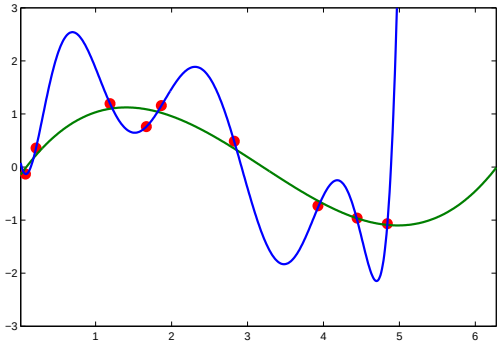
- Задача машинного обучения заключается в восстановлении зависимостей по конечным выборкам данных (прецедентов)
- Пусть $(X, \boldsymbol{t}) = (\boldsymbol{x}_i, t_i)_{i=1}^n$ – обучающая выборка, где $\boldsymbol{x}_i \in \mathbb{R}^d$ – признаковое описание объекта, а $t \in \mathcal{T}$ – значение скрытой компоненты (классовая принадлежность (не по Марксу!), значение прогноза, номер кластера и т.д.)
- При статистическом подходе к решению задачи МО предполагается, что обучающая выборка является выборкой из некоторой генеральной совокупности с плотностью $p(\boldsymbol{x}, t)$
- Требуется восстановить $p(t|\boldsymbol{x})$, т.е. знание о скрытой компоненте объекта по измеренным признакам

Проблема переобучения

Прямая минимизация невязки на обучающей выборке ведет к получению решающих правил, способных объяснить все что угодно и найти закономерности даже там, где их нет.







Способы оценки и увеличения обобщающей способности

- На сегодняшний день единственным универсальным способом оценивания обобщающей способности является скользящий контроль
- Все попытки предложить что-нибудь отличное от метода проб и ошибок пока не привели к общепризнанному решению. Наиболее известны из них следующие:
  - Структурная минимизация риска (В. Вапник, А. Червоненкис, 1974)
  - Минимизация длины описания (Дж. Риссанен, 1978)
  - Информационные критерии Акаике и Байеса-Шварца (Х. Акаике, 1974, Г. Шварц, 1978)
  - Максимизация обоснованности (Д. МакКай, 1992)
- Последний принцип позволяет надеяться на конструктивное решение задачи выбора модели

Примеры задач выбора модели

- Определение числа кластеров в данных
- Выбор коэффициента регуляризации в задаче машинного обучения
- Установка степени полинома при интерполяции сплайнами
- Выбор наилучшей базисной функции в обощенных линейных моделях
- Определение количества ветвей в решающем дереве
- и многое другое...

План лекции

Матричная нотация

- При работе с многомерными величинами очень удобна матричная нотация, т.е. представление многих операций над векторами и числами в виде операций над матрицами
- Скалярное произведение двух векторов $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ принимает вид

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{d} x_i y_i = \boldsymbol{x}^T \boldsymbol{y},$$

  т.е. вектора трактуются как частные случаи матриц
- Квадратичная форма

$$\langle A\boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{d} \sum_{j=1}^{d} x_i a_{ij} y_j = \boldsymbol{x}^T A \boldsymbol{y}$$

- Матричная нотация облегчает математические выкладки и позволяет реализовать вычисления на ЭВМ более эффективно

Пример использования

- Предположим нам надо решить несовместную систему линейных уравнений $A\boldsymbol{x} \approx \boldsymbol{b}$, $A \in \mathbb{R}^{m \times n}$
- Для этого будем минимизировать квадрат нормы невязки (система-то нерешаемая) $\|A\boldsymbol{x} - \boldsymbol{b}\|^2 \to \min_{\boldsymbol{x}}$
- Представляя норму в матричной виде, дифференцируя по вектору и приавнивая производную к нулю получаем известную формулу для псевдорешения СЛАУ

$$\|A\boldsymbol{x} - \boldsymbol{b}\|^2 = \langle A\boldsymbol{x} - \boldsymbol{b}, A\boldsymbol{x} - \boldsymbol{b} \rangle = (A\boldsymbol{x} - \boldsymbol{b})^T (A\boldsymbol{x} - \boldsymbol{b}) =$$

$$(A\boldsymbol{x})^T A\boldsymbol{x} - \boldsymbol{b}^T A\boldsymbol{x} - (A\boldsymbol{x})^T \boldsymbol{b} + \boldsymbol{b}^T \boldsymbol{b} =$$

$$\boldsymbol{x}^T A^T A\boldsymbol{x} - 2\boldsymbol{x}^T A^T \boldsymbol{b} + \boldsymbol{b}^T \boldsymbol{b}$$

$$\frac{\partial}{\partial \boldsymbol{x}} \left(\boldsymbol{x}^T A^T A\boldsymbol{x} - 2\boldsymbol{x}^T A^T \boldsymbol{b} + \boldsymbol{b}^T \boldsymbol{b} \right) = 2A^T A\boldsymbol{x} - 2A^T \boldsymbol{b} = 0$$

$$\boldsymbol{x} = \left(A^T A \right)^{-1} A^T \boldsymbol{b}$$

- Заметим, что если матрица A квадратная (число уравнений равно числу неизвестных) и невырожденная, то последняя формула переходит в формулу обычного решения СЛАУ $\boldsymbol{x} = A^{-1}\boldsymbol{b}$

Задача условной оптимизации

Пусть $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ — гладкая функция. Предположим, что нам необходимо найти ее экстремум:

$$f(\boldsymbol{x}) \to \underset{\boldsymbol{x}}{\text{extr}}$$

Для того, чтобы найти экстремум (решить задачу безусловной оптимизации), достаточно проверить условие стационарности:

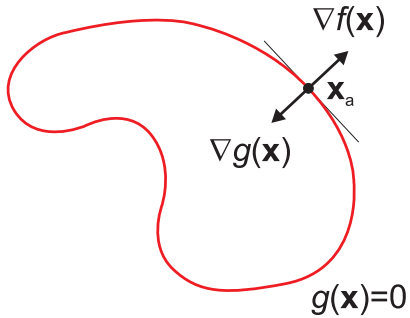
$$\nabla f(\boldsymbol{x}) = 0$$

Предположим, что нам необходимо найти экстремум функции при ограничениях:

$$f(\boldsymbol{x}) \to \underset{\boldsymbol{x}}{\text{extr}}$$

$$g(\boldsymbol{x}) = 0$$

Поверхность ограничения



Заметим, что $\nabla g(\boldsymbol{x})$ ортогонален поверхности ограничения $g(\boldsymbol{x}) = 0$. Пусть $\boldsymbol{x}$ и $\boldsymbol{x} + \boldsymbol{\varepsilon}$ — две близкие точки поверхности. Тогда

$$g(\boldsymbol{x} + \boldsymbol{\varepsilon}) \simeq g(\boldsymbol{x}) + \boldsymbol{\varepsilon}^T \nabla g(\boldsymbol{x})$$

Т.к. $g(\boldsymbol{x} + \boldsymbol{\varepsilon}) = g(\boldsymbol{x})$, то $\boldsymbol{\varepsilon}^T \nabla g(\boldsymbol{x}) \simeq 0$. При стремлении $\|\boldsymbol{\varepsilon}\| \to 0$ получаем $\boldsymbol{\varepsilon}^T \nabla g(\boldsymbol{x}) = 0$. Т.к. $\boldsymbol{\varepsilon}$ параллелен поверхности $g(\boldsymbol{x}) = 0$, то $\nabla g(\boldsymbol{x})$ является нормалью к этой поверхности.

Функция Лагранжа

Необходимым условием оптимальности является ортогональность $\nabla f(\boldsymbol{x})$ поверхности ограничения, т.е.:

$$\nabla f + \lambda \nabla g = 0 \tag{1}$$

Здесь $\lambda \neq 0$ — коэффициент Лагранжа. Он может быть любого знака.

Функция Лагранжа

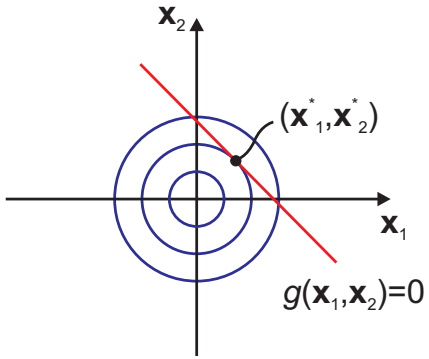
$$L(\boldsymbol{x}, \lambda) \triangleq f(\boldsymbol{x}) + \lambda g(\boldsymbol{x})$$

Тогда

$$\nabla_{\boldsymbol{x}} L = 0 \quad \Rightarrow \quad \text{условие } (1)$$

$$\frac{\partial}{\partial \lambda} L = 0 \quad \Rightarrow \quad g(\boldsymbol{x}) = 0$$

Функция Лагранжа. Пример.



$$f(x_1, x_2) = 1 - x_1^2 - x_2^2 \to \max_{x_1, x_2}$$

$$g(x_1, x_2) = x_1 + x_2 - 1 = 0$$

Функция Лагранжа:

$$L(\boldsymbol{x}, \lambda) = 1 - x_1^2 - x_2^2 + \lambda(x_1 + x_2 - 1)$$

Условия стационарности:

$$-2x_1 + \lambda = 0$$

$$-2x_2 + \lambda = 0$$

$$x_1 + x_2 - 1 = 0$$

Решение: $(x_1^*, x_2^*) = (\frac{1}{2}, \frac{1}{2}),\ \lambda = 1.$

План лекции

Краткое напоминание основных вероятностных понятий

- $X : \Omega \to \mathbb{R}$ – случайная величина
- Вероятность попадания величины в интервал (a, b) равна

$$P(a \leq X \leq b) = \int_a^b p(x)dx,$$

где $p(x)$ – плотность распределения X,

$$p(x) \geq 0, \quad \int_{-\infty}^{\infty} p(x)dx = 1$$

- Если поведение случайной величины определяется некоторым параметром, возникают условные плотности $p(x|\theta)$. Если рассматривать условную плотность как функцию от параметра

$$f(\theta) = p(x|\theta),$$

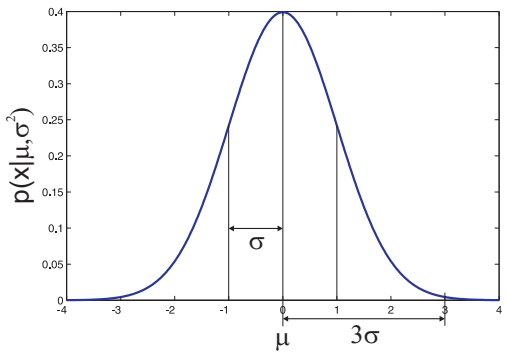
то принято говорить о т.н. функции правдоподобия

Нормальное распределение

- Нормальное распределение играет важнейшую роль в математической статистике

$$X \sim \mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

$$\mu = \mathbb{E}X, \quad \sigma^2 = \mathbb{D}X \triangleq \mathbb{E}(X-\mathbb{E}X)^2$$



- Из центральной предельной теоремы следует, что сумма независимых случайных величин с ограниченной дисперсией стремится к нормальному распределению
- На практике многие случайные величины можно считать приближенно нормальными

Многомерное нормальное распределение

- Многомерное нормальное распределение имеет вид

$$X \sim \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma) = \frac{1}{\sqrt{2\pi}^n \sqrt{\det \Sigma}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right),$$

  где $\boldsymbol{\mu} = \mathbb{E}X$, $\Sigma = \mathbb{E}(X-\boldsymbol{\mu})(X-\boldsymbol{\mu})^T$ — вектор математических ожиданий каждой из n компонент и матрица ковариаций соответственно
- Матрица ковариаций показывает, насколько сильно связаны (коррелируют) компоненты многомерного нормального распределения

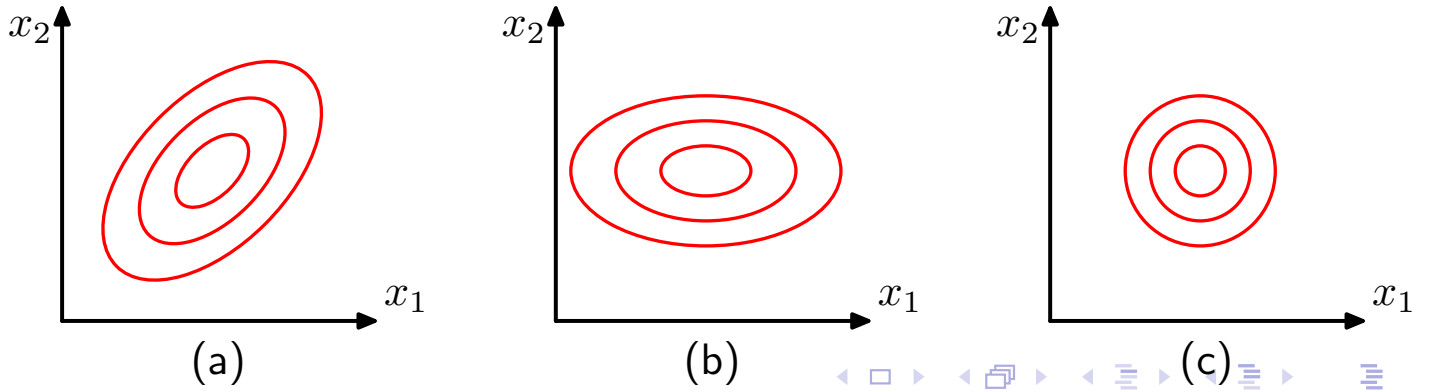
$$\Sigma_{ij} = \mathbb{E}(X_i - \mu_i)(X_j - \mu_j) = \mathrm{Cov}(X_i, X_j)$$

- Если мы поделим ковариацию на корень из произведений дисперсий, то получим коэффициент корреляции

$$\rho(X_i, X_j) \triangleq \frac{\mathrm{Cov}(X_i, X_j)}{\sqrt{\mathbb{D}X_i \mathbb{D}X_j}} \in [-1, 1]$$

Особенности нормального распределения

- Нормальное распределение полностью задается первыми двумя моментами (мат. ожидание и матрица ковариаций/дисперсия)
- Матрица ковариаций неотрицательно определена, причем на диагоналях стоят дисперсии соответствующих компонент
- Нормальное распределение имеет очень легкие хвосты: большие отклонения от мат. ожидания практически невозможны. Это обстоятельство нужно учитывать при приближении произвольных случайных величин нормальными



Основная задача мат. статистики

- Распределение случайной величины X известно с точностью до параметра θ
- Имеется выборка значений величины X, $\boldsymbol{x} = (x_1, \ldots, x_n)$
- Требуется оценить значение θ
- Метод максимального правдоподобия

$$\hat{\theta}_{ML} = \arg\max f(\theta) = \arg\max p(\boldsymbol{x}|\theta) = \arg\max \prod_{i=1}^{n} p(x_i|\theta)$$

- Можно показать, что ММП (не путать с одноименной кафедрой) является асимптотически оптимальным при $n \to \infty$
- Обычно максимизируют не само правдоподобие, а его логарифм, т.к. это вычислительно проще (произведение плотностей по всем объектам переходит в сумму логарифмов плотностей)

Пример использования

- Пусть имееется выборка из нормального распределения $\mathcal{N}(x|\mu, \sigma^2)$ с неизвестными мат. ожиданием и дисперсией
- Выписываем логарифм функции правдоподобия

$$L(X|\mu,\sigma) = -\sum_{i=1}^{n}\frac{(x_i-\mu)^2}{2\sigma^2} - n\log\sigma - \frac{n}{2}\log(2\pi) \to \max_{\mu,\sigma}$$

$$\frac{\partial L}{\partial \mu} = -\sum_{i=1}^{n}\frac{(x_i-\mu)}{\sigma^2} = 0 \qquad \frac{\partial L}{\partial \sigma} = \sum_{i=1}^{n}\frac{(x_i-\mu)^2}{\sigma^3} - \frac{n}{\sigma} = 0$$

$$\mu_{ML} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma^2_{ML} = \frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2 =$$

$$= \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \frac{1}{n}\sum_{j=1}^{n} x_j\right)^2$$

Выводы

- Не все параметры можно настраивать в ходе обучения
- Существуют специальные параметры (будем называть их структурными), которые должны быть зафиксированы до начала обучения
- В последнем примере величина m (количество компонент смеси) является структурным параметром
- Основной открытой проблемой машинного обучения является проблема выбора структурных параметров, позволяющих избегать переобучения

Обобщенные линейные модели

Ю. И. Журавлев[1], Д. П. Ветров[1]

[1]МГУ, ВМиК, каф. ММП

Курс «Математические основы теории прогнозирования»

План лекции

Напоминание

- Формула Байеса
- Решение нерешаемых систем уравнений

Линейная регрессия

- Классическая линейная регрессия
- Метод наименьших квадратов
- Вероятностная постановка задачи

Применение регрессионных методов для задачи классификаци

- Логистическая регрессия
- Метод IRLS

План лекции

Условная вероятность

- Пусть X и Y — случайные величины с плотностями $p(x)$ и $p(y)$ соответственно
- В общем случае их совместная плотность $p(x,y) \neq p(x)p(y)$. Если это равенство выполняется, величины называют независимыми
- Условной плотностью называется величина

$$p(x|y) = \frac{p(x,y)}{p(y)}$$

- Смысл: как факт $Y = y$ влияет на распределение X. Заметим, что $\int p(x|y)dx \equiv 1$, но $\int p(x|y)dy$ не обязан равняться единице, т.к. относительно y это не плотность, а функция правдоподобия
- Очевидная система тождеств $p(x|y)p(y) = p(x,y) = p(y|x)p(x)$ позволяет легко переходить от $p(x|y)$ к $p(y|x)$

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

Правило суммирования вероятностей

- Все операции над вероятностями базируются на применении всего двух правил
- Правило суммирования: Пусть $A_1, \ldots, A_k$ взаимоисключающие события, одно из которых всегда происходит. Тогда

$$P(A_i \cup A_j) = P(A_i) + P(A_j) \qquad \sum_{i=1}^{k} P(A_i) = 1$$

- Очевидное следствие (формула полной вероятности): $\forall B$ верно $\sum_{i=1}^{k} P(A_i|B) = 1$, откуда

$$\sum_{i=1}^{k} \frac{P(B|A_i)P(A_i)}{P(B)} = 1 \qquad P(B) = \sum_{i=1}^{k} P(B|A_i)P(A_i)$$

- В интегральной форме

$$p(b) = \int p(b,a)da = \int p(b|a)p(a)da$$

Правило произведения вероятностей

- Правило произведения гласит, что любую совместную плотность всегда можно разбить на множители

$$p(a,b) = p(a|b)p(b) \qquad P(A,B) = P(A|B)P(B)$$

- Аналогично для многомерных совместных распределений

$$p(a_1,\ldots,a_n) =$$

$$p(a_1|a_2,\ldots,a_n)p(a_2|a_3,\ldots,a_n)\ldots p(a_{n-1}|a_n)p(a_n)$$

- Можно показать (Jaynes, 1995), что правила суммирования и произведения вероятностей являются единственными возможными операциями, позволяющими рассматривать вероятности как промежуточную ступень между истиной и ложью

Априорные и апостериорные суждения

- Предположим, мы пытаемся изучить некоторое явление
- У нас имеются некоторые знания, полученные до (лат. a priori) наблюдений/эксперимента. Это может быть опыт прошлых наблюдений, какие-то модельные гипотезы, ожидания
- В процессе наблюдений эти знания подвергаются постепенному уточнению. После (лат. a posteriori) наблюдений/эксперимента у нас формируются новые знания о явлении
- Будем считать, что мы пытаемся оценить неизвестное значение величины θ посредством наблюдений некоторых ее косвенных характеристик $x|\theta$

Формула Байеса

- Знаменитая формула Байеса (1763 г.) устанавливает правила, по которым происходит преобразование знаний в процессе наблюдений
- Обозначим априорные знания о величине θ за $p(\theta)$
- В процессе наблюдений мы получаем серию значений $\boldsymbol{x} = (x_1, \ldots, x_n)$. При разных θ наблюдение выборки $\boldsymbol{x}$ более или менее вероятно и определяется значением правдоподобия $p(\boldsymbol{x}|\theta)$
- За счет наблюдений наши представления о значении θ меняются согласно формуле Байеса

$$p(\theta|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\theta)p(\theta)}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|\theta)p(\theta)}{\int p(\boldsymbol{x}|\theta)p(\theta)d\theta}$$

- Заметим, что знаменатель не зависит от θ и нужен исключительно для нормировки апостериорной плотности

План лекции

Псевдообращение матриц

- Предположим, нам необходимо решить СЛАУ вида $A\boldsymbol{x} = \boldsymbol{b}$
- Если бы матрица A была квадратной и невырожденной (число уравнений равно числу неизвестных и все уравнения линейно независимы), то решение задавалось бы формулой $\boldsymbol{x} = A^{-1}\boldsymbol{b}$
- Предположим, что число уравнений больше числа неизвестных, т.е. матрица A прямоугольная. Домножим обе части уравнения на A^T слева

$$A^T A\boldsymbol{x} = A^T\boldsymbol{b}$$

- В левой части теперь квадратная матрица и ее можно перенести в правую часть

$$\boldsymbol{x} = \left(A^T A\right)^{-1} A^T\boldsymbol{b}$$

- Операция $\left(A^T A\right)^{-1} A^T$ называется псевдообращением матрицы A, а $\boldsymbol{x}$ – псевдорешением

Нормальное псевдорешение

- Если матрица A^TA вырождена, псевдорешений бесконечно много, причем найти их на компьютере нетривиально
- Для решения этой проблемы используется ридж-регуляризация матрицы A^TA

$$A^TA + \lambda I,$$

где I – единичная матрица, а λ – коэффициент регуляризации. Такая матрица невырождена для любых $\lambda > 0$

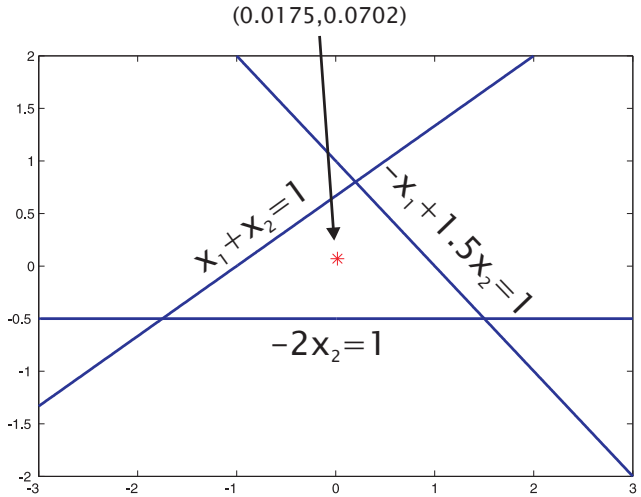
- Величина

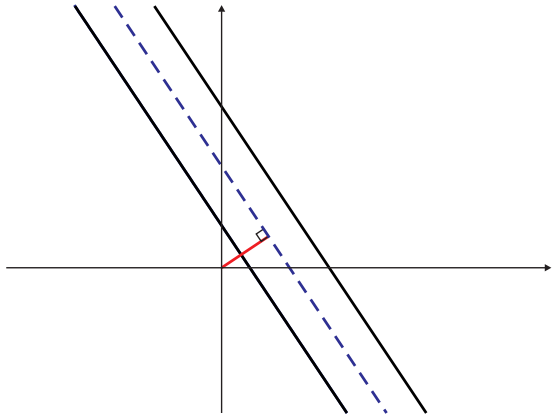
$$\boldsymbol{x} = \left(A^TA + \lambda I\right)^{-1} A^T\boldsymbol{b}$$

называется нормальным псевдорешением. Оно всегда единственно и при небольших положительных λ определяет псевдорешение с наименьшей нормой

Графическая иллюстрация

- Псевдорешение соответствует точке, минимизирующей невязку, а нормальное псевдорешение отвечает псевдорешению с наименьшей нормой





- Заметим, что псевдообратная матрица $\left(A^TA\right)^{-1}A^T$ совпадает с обратной матрицей A^{-1} в случае невырожденных квадратных матриц

План лекции

Задача восстановления регрессии

- Задача восстановления регрессии предполагает наличие связи между наблюдаемыми признаками $\boldsymbol{x}$ и непрерывной переменной t
- В отличие от задачи интерполяции допускаются отклонения решающего правила от правильных ответов на объектах обучающей выборки
- Уравнение регрессии $y(\boldsymbol{x}, \boldsymbol{w})$ ищется в некотором параметрическом виде путем нахождения наилучшего значения вектора весов

$$\boldsymbol{w}_* = \arg\max_{\boldsymbol{w}} F(X, \boldsymbol{t}, \boldsymbol{w})$$

Линейная регрессия

- Наиболее простой и изученной является линейная регрессия
- Главная особенность: настраиваемые параметры входят в решающее правило линейно
- Заметим, что линейная регрессия не обязана быть линейной по признакам
- Общее уравнение регрессии имеет вид

$$y(\boldsymbol{x}, \boldsymbol{w}) = \sum_{j=1}^{m} w_j \phi_j(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x})$$

Особенность выбора базисных функций

- Общего метода выбора базисных функций $\phi_j(\boldsymbol{x})$ — не существует
- Обычно они подбираются из априорных соображений (например, если мы пытаемся восстановить какой-то периодический сигнал, разумно взять функции тригонометрического ряда) или путем использования некоторых «универсальных» базисных функций
- Наиболее распространенными базисными функциями являются
  - $\phi(\boldsymbol{x}) = x_k$
  - $\phi(\boldsymbol{x}) = x_{k_1} x_{k_2} \dots x_{k_l}$
  - $\phi(\boldsymbol{x}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{x}_0\|^p), \ \gamma, p > 0.$
- Метод построения линейной регрессии (настройки весов $\boldsymbol{w}$) не зависит от выбора базисных функций

Формализация задачи

- Пусть $S(t, \hat{t})$ — функция потерь от ошибки в определении регрессионной переменной t
- Необходимо минимизировать потери от ошибок на генеральной совокупности

$$\mathbb{E}S(t, y(\boldsymbol{x}, \boldsymbol{w})) = \int\int S(t, y(\boldsymbol{x}, \boldsymbol{w}))p(\boldsymbol{x}, t)d\boldsymbol{x}dt \to \min_{\boldsymbol{w}}$$

- Дальнейшие рассуждения зависят от вида функции потерь
- Во многих случаях даже не нужно восстанавливать полностью условное распределение $p(t|\boldsymbol{x})$

Важная теорема

- Теорема. Пусть функция потерь имеет вид
  - $S(t, \hat{t}) = (t - \hat{t})^2$ — «Потери старушки»;
  - $S(t, \hat{t}) = |t - \hat{t}|$ — «Потери олигарха»;
  - $S(t, \hat{t}) = \delta^{-1}(t - \hat{t})$ — «Потери инвалида».

  Тогда величиной, минимизирующей функцию $\mathbb{E}S(t, y(\boldsymbol{x}, \boldsymbol{w}))$, является следующая
  - $y(\boldsymbol{x}) = \mathbb{E}p(t|\boldsymbol{x})$;
  - $y(\boldsymbol{x}) = \text{med}\ p(t|\boldsymbol{x})$;
  - $y(\boldsymbol{x}) = \text{mod}\ p(t|\boldsymbol{x}) = \arg\max_t p(t|\boldsymbol{x})$.
- В зависимости от выбранной системы предпочтений, мы будем пытаться оценивать тот или иной функционал от апостериорного распределения вместо того, чтобы оценивать его самого

План лекции

Минимизация невязки

- Наиболее часто используемой функцией потерь является квадратичная $S(t, \hat{t}) = (t - \hat{t})^2$
- Значение регрессионной функции на обучающей выборке в матричном виде может быть записано как $\boldsymbol{y} = \Phi \boldsymbol{w}$, где $\Phi = (\phi_{ij}) = (\phi_j(\boldsymbol{x}_i)) \in \mathbb{R}^{n \times m}$
- Таким образом, приходим к следующей задаче

$$\|\boldsymbol{y} - \boldsymbol{t}\|^2 = \|\Phi \boldsymbol{w} - \boldsymbol{t}\|^2 \to \min_{\boldsymbol{w}}$$

Взяв производную по $\boldsymbol{w}$ и приравняв ее к нулю, получаем

$$\frac{\partial \|\Phi \boldsymbol{w} - \boldsymbol{t}\|^2}{\partial \boldsymbol{w}} = \frac{\partial [\boldsymbol{w}^T \Phi^T \Phi \boldsymbol{w} - 2\boldsymbol{w}^T \Phi^T \boldsymbol{t} + \boldsymbol{t}^T \boldsymbol{t}]}{\partial \boldsymbol{w}} =$$

$$= 2\Phi^T \Phi \boldsymbol{w} - 2\Phi^T \boldsymbol{t} = 0$$

$$\boldsymbol{w} = (\Phi^T \Phi)^{-1} \Phi^T \boldsymbol{t}$$

Регуляризация задачи

- Заметим, что формула для весов линейной регрессии представляет собой псевдорешение уравнения $\Phi \boldsymbol{w} = \boldsymbol{t}$
- Матрица $\Phi^T \Phi \in \mathbb{R}^{m \times m}$ вырождена (Упр.)при $m > n$
- Регуляризуя вырожденную матрицу, получаем

$$\boldsymbol{w} = \left(\Phi^T \Phi + \lambda I\right)^{-1} \Phi^T \boldsymbol{t}$$

- Отсюда формула для прогноза объектов обучающей выборки по их правильным значениям

$$\hat{\boldsymbol{t}} = \boldsymbol{y} = \Phi \left(\Phi^T \Phi + \lambda I\right)^{-1} \Phi^T \boldsymbol{t} = H\boldsymbol{t}$$

С историческим обозначением прогноза — навешиванием шляпки связано неформальное название матрицы H, по-английски звучащее как hat-matrix

Особенности квадратичной функции потерь

- Достоинства
  - Квадратичная функция потерь гладкая (непрерывная и дифференцируемая)
  - Решение может быть получено в явном виде
  - Существует простая вероятностная интерпретация прогноза и функции потерь
- Недостатки
  - Решение неустойчиво (не робастно) относительно даже малого количества выбросов. Это связано с быстрым возрастанием квадратичной функции потерь при больших отклонениях от нуля
  - Квадратичная функция неприменима к задачам классификации

План лекции

Нормальное распределение ошибок

- Рассмотрим вероятностную постановку задачи восстановления регрессии. Регрессионная переменная t — случайная величина с плотностью распределения $p(t|\boldsymbol{x})$
- В большинстве случаев предполагается, что t распределена нормально относительно некоторого мат. ожидания $y(\boldsymbol{x})$, определяемого точкой $\boldsymbol{x}$

$$t = y(\boldsymbol{x}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(\varepsilon|0, \sigma^2)$$

- Необходимо найти функцию $y(\boldsymbol{x})$, которую мы можем отождествить с уравнением регрессии
- Предположение о нормальном распределении отклонений можно обосновать ссылкой на центральную предельную теорему

Метод максимального правдоподобия для регрессии

- Используем ММП (не путать с одноименной кафедрой) для поиска $y(\boldsymbol{x})$
- Правдоподобие задается следующей формулой

$$p(\boldsymbol{t}|\boldsymbol{y}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(t_i - y_i)^2}{2\sigma^2}\right) \to \max$$

- Взяв логарифм и отбросив члены, не влияющие на положение максимума, получим

$$\sum_{i=1}^{n}(t_i - y_i)^2 = \sum_{i=1}^{n}(t_i - \boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i))^2 \to \min_{\boldsymbol{w}}$$

- Таким образом, применение метода максимального правдоподобия в предположении о нормальности отклонений эквивалентно методу наименьших квадратов

Вероятностный смысл регуляризации

- Теперь будем максимизировать не правдоподобие, а апостериорную вероятность
- По формуле условной вероятности

$$p(\boldsymbol{w}|\boldsymbol{t}, X) = \frac{p(\boldsymbol{t}|X, \boldsymbol{w})p(\boldsymbol{w})}{p(\boldsymbol{t}, X)} \to \max_{\boldsymbol{w}},$$

знаменатель не зависит от $\boldsymbol{w}$, поэтому им можно пренебречь

- Пусть $p(\boldsymbol{w}) \sim \mathcal{N}\left(\boldsymbol{w} \middle| \boldsymbol{0}, \left(\frac{\sigma^2}{\lambda}\right) I\right)$. Тогда

$$p(\boldsymbol{w}|\boldsymbol{t}, X) \propto \frac{\lambda^{m/2}}{\left(\sqrt{2\pi}\sigma\right)^{m+n}} \exp\left(-\frac{1}{2}\left(\sigma^{-2}\|\Phi\boldsymbol{w} - \boldsymbol{t}\|^2 + \frac{\lambda}{\sigma^2}\|\boldsymbol{w}\|^2\right)\right)$$

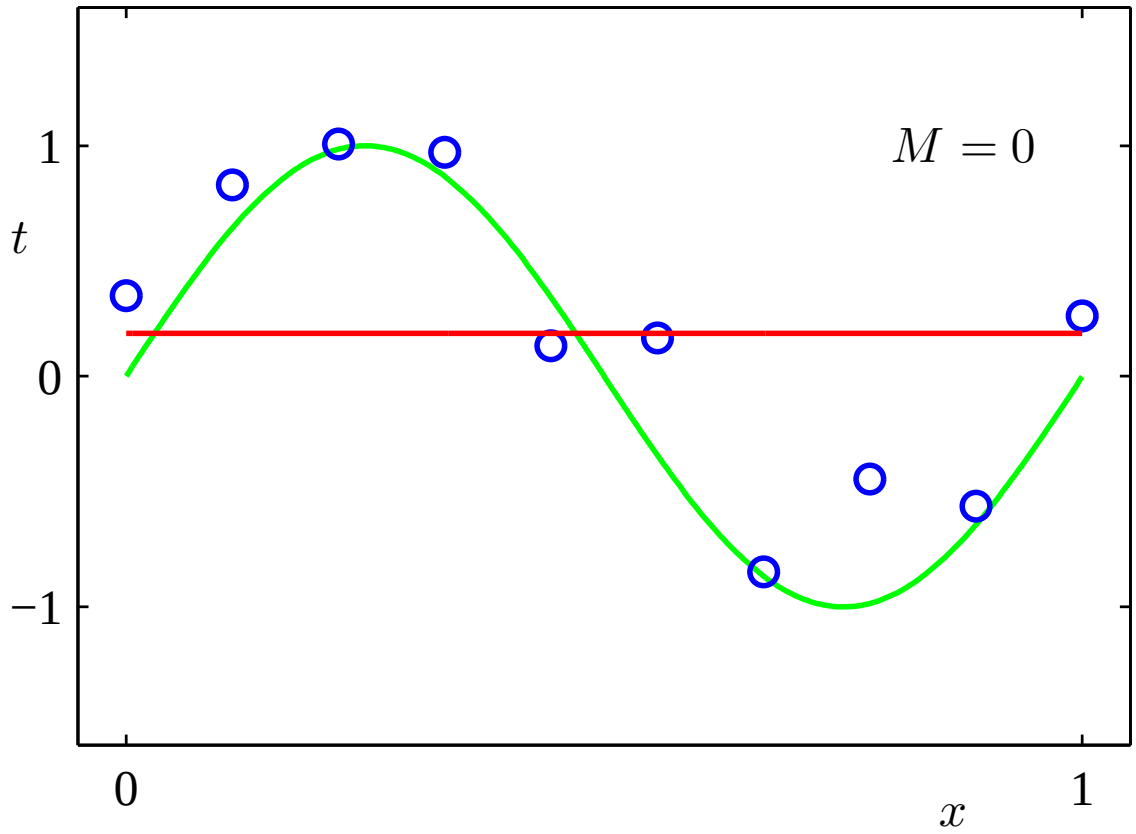
- Логарифмируя и приравнивая производную по $\boldsymbol{w}$ к нулю, получаем

$$\boldsymbol{w} = (\Phi^T\Phi + \lambda I)^{-1}\Phi^T\boldsymbol{t}$$

- Регуляризация эквивалентна введению априорного распределения, поощряющего небольшие веса

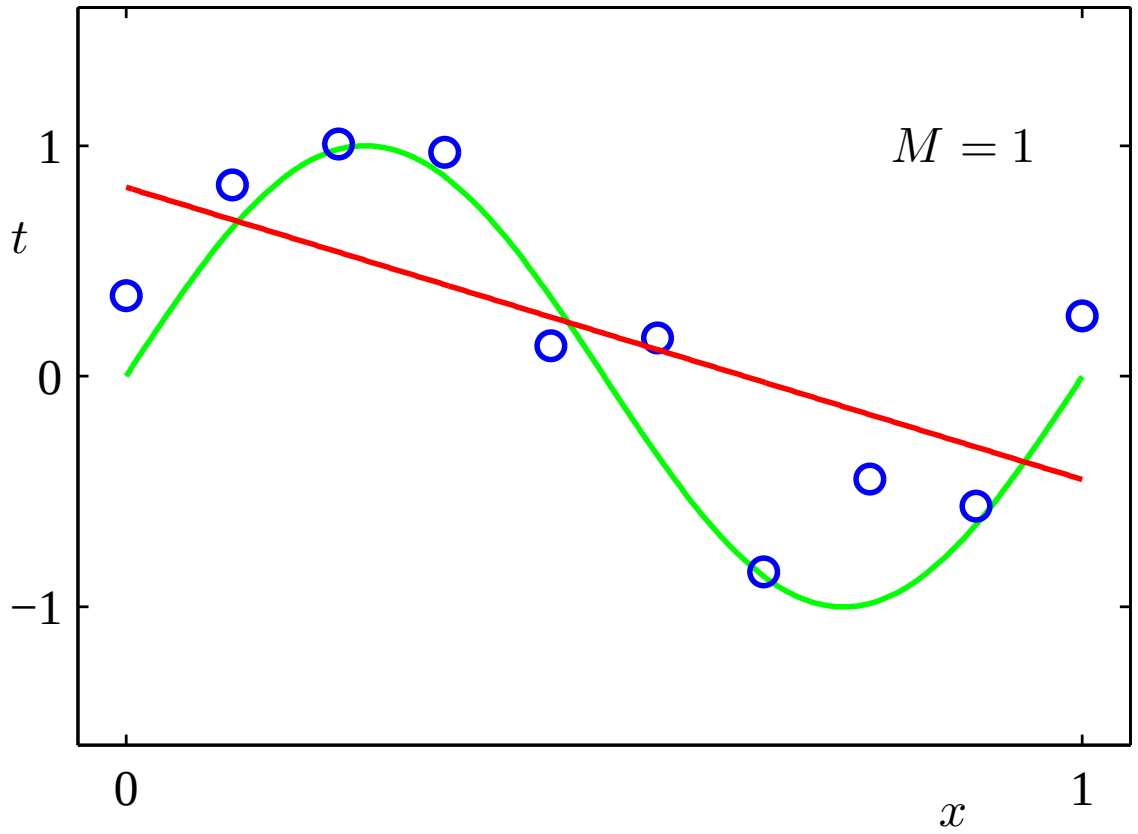
Зачем нужна регугляризация весов

Рассмотрим задачу восстановления регрессии с полиномиальными базисными функциями: $x \in \mathbb{R}$, $\phi_j(x) = x^j$, $j = 0, \ldots, M$



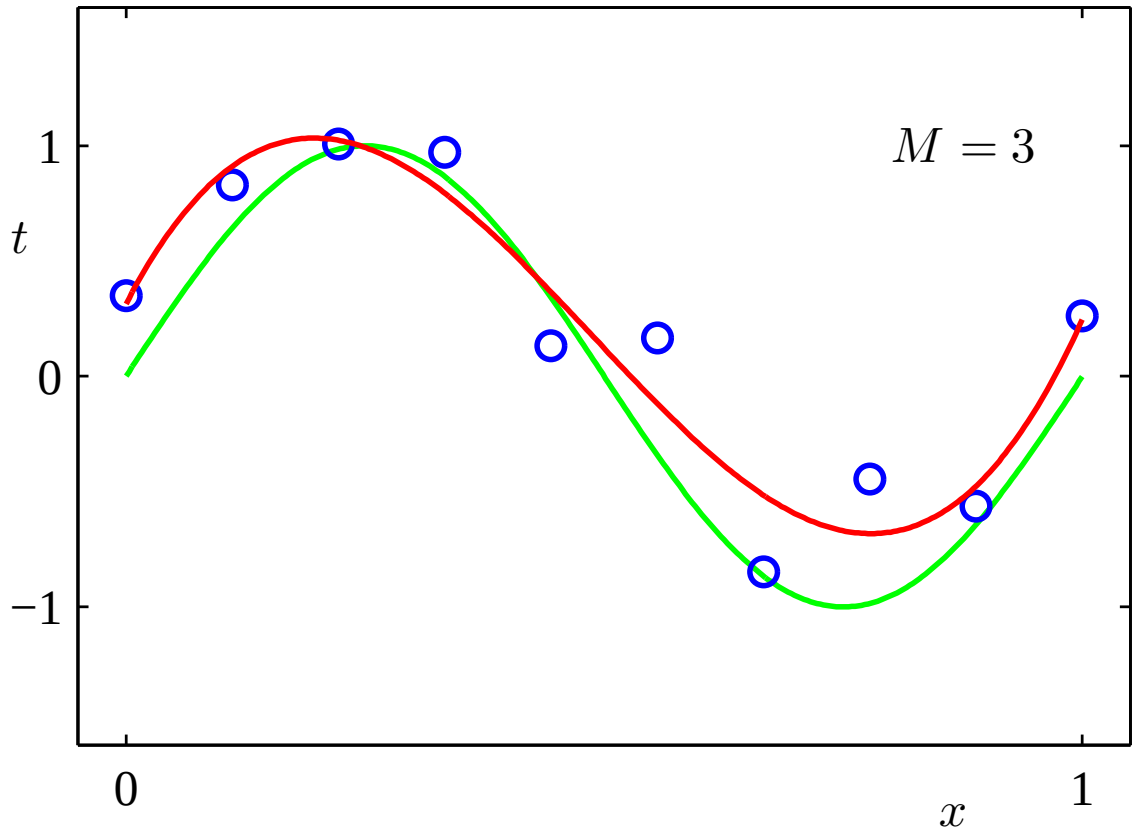
Зачем нужна реугляризация весов

Рассмотрим задачу восстановления регрессии с полиномиальными базисными функциями: $x \in \mathbb{R}$, $\phi_j(x) = x^j$, $j = 0, \ldots, M$



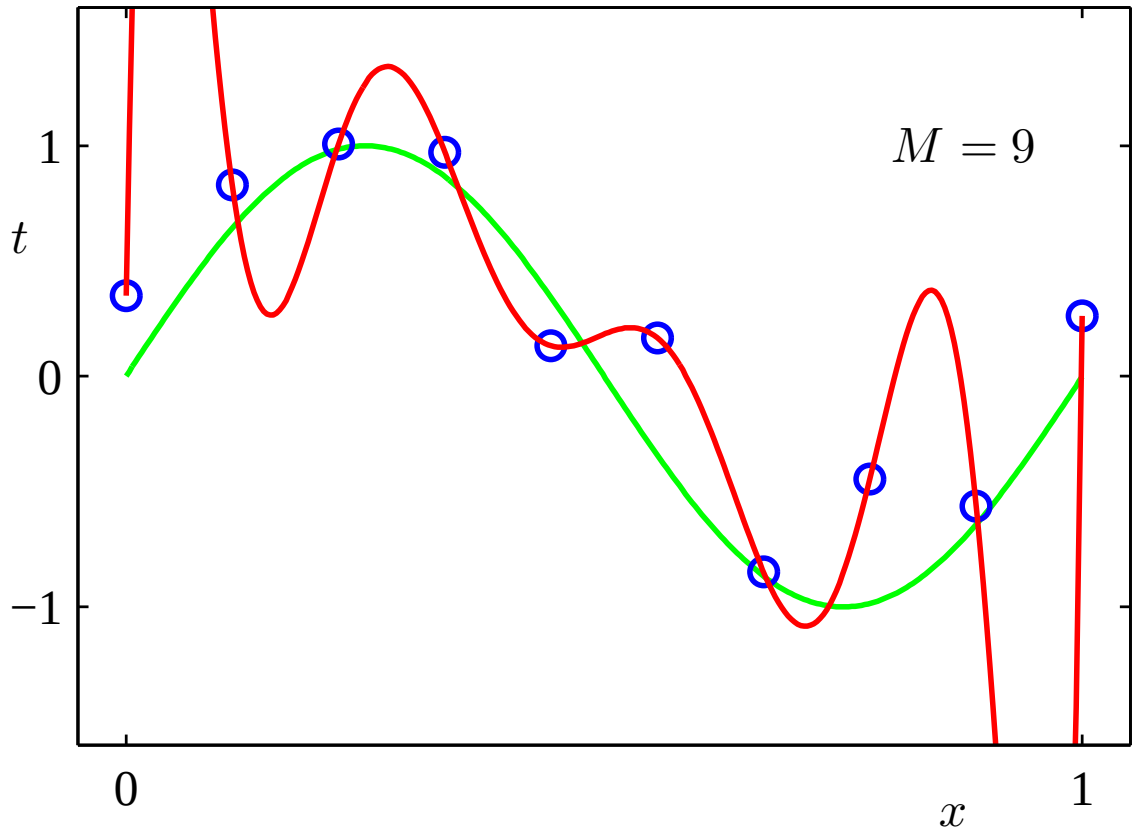
Зачем нужна регугляризация весов

Рассмотрим задачу восстановления регрессии с полиномиальными базисными функциями: $x \in \mathbb{R}$, $\phi_j(x) = x^j$, $j = 0, \ldots, M$



Зачем нужна регугляризация весов

Рассмотрим задачу восстановления регрессии с полиномиальными базисными функциями: $x \in \mathbb{R}$, $\phi_j(x) = x^j$, $j = 0, \ldots, M$



Значения наиболее правдоподобных весов

weight	$M = 0$	$M = 1$	$M = 3$	$M = 9$
w_0	0.19	0.82	0.31	0.35
w_1		-1.27	7.99	232.37
w_2			-25.43	-5321.83
w_3			17.37	48568.31
w_4				-231639.30
w_5				640042.26
w_6				-1061800.52
w_7				1042400.18
w_8				-557682.99
w_9				125201.43

Таблица: Значения наиболее правдоподобных весов в зависимости от степени полинома. С увеличением степени, абсолютные значения весов быстро растут

План лекции

Особенности задачи классификации

- Рассмотрим задачу классификации на два класса $t \in \{-1, +1\}$
- Ее можно свести к задаче регрессии, например, следующим образом

$$\hat{t}(\boldsymbol{x}) = \text{sign}(y(\boldsymbol{x})) = \text{sign} \sum_{j=1}^{m} w_j \phi_j(\boldsymbol{x})$$

- Возникает вопрос: что использовать в качестве значений регрессионной переменной на этапе обучения?
- Наиболее распространенный подход заключается в использовании значения $+\infty$ для $t = +1$ и $-\infty$ для $t = -1$
- Геометрический смысл: чем дальше от нуля значение $y(\boldsymbol{x})$, тем увереннее мы в классификации объекта $\boldsymbol{x}$

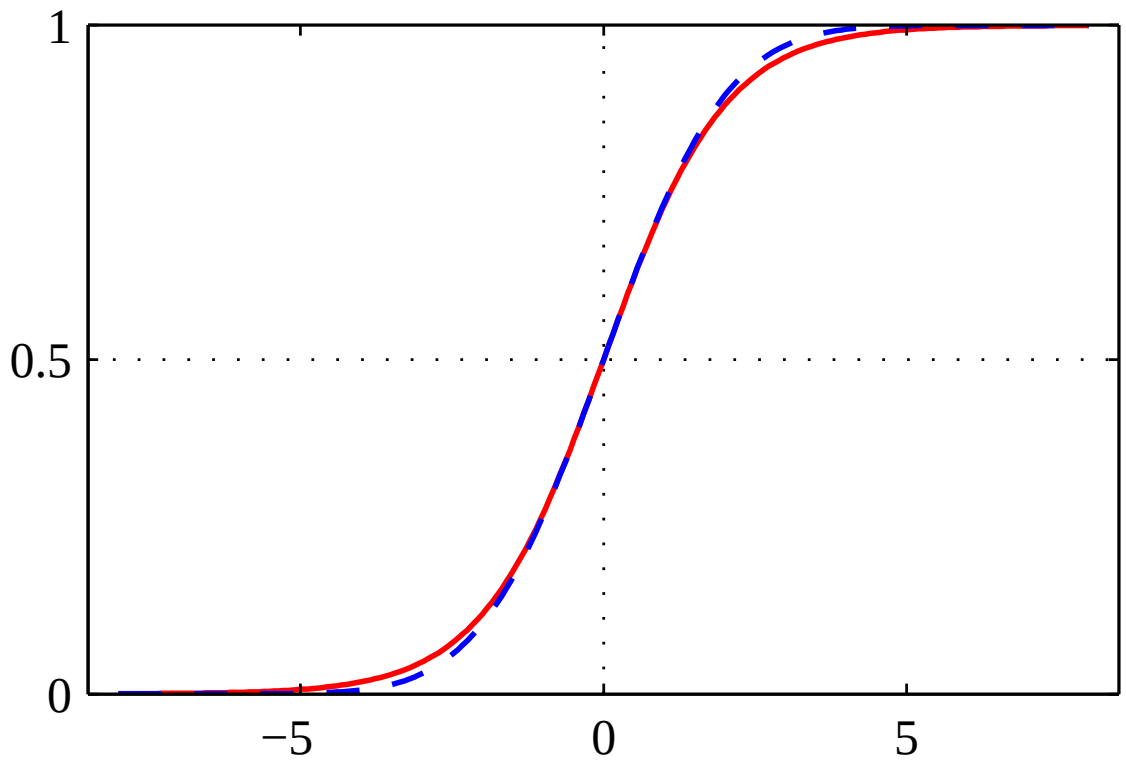
Правдоподобие правильной классификации

- Метод наименьших квадратов, очевидно, неприменим при таком подходе
- Воспользуемся вероятностной постановкой для выписывания функционала качества
- Определим правдоподобие классификации следующим образом

$$p(t|\boldsymbol{x}, \boldsymbol{w}) = \frac{1}{1 + \exp(-ty(\boldsymbol{x}))}$$

- Это логистическая функция. Легко показать, что $\sum_t p(t|\boldsymbol{x}, \boldsymbol{w}) = 1$ и $p(t|\boldsymbol{x}, \boldsymbol{w}) > 0$, а, значит, она является функцией правдоподобия

Функционал качества в логистической регрессии



- Правдоподобие правильной классификации всей выборки имеет вид

$$p(\boldsymbol{t}|X,\boldsymbol{w}) = \prod_{i=1}^{n} p(t_i|\boldsymbol{x}_i,\boldsymbol{w}) = \prod_{i=1}^{n} \frac{1}{1+\exp\left(-t_i \sum_{j=1}^{m} w_j \phi_j(\boldsymbol{x}_i)\right)}$$

План лекции

Особенности функции правдоподобия классификации

- Приравнивание градиента логарифма правдоподобия к нулю приводит к трансцендентным уравнениям, которые неразрешимы аналитически
- Легко показать (Упр.), что гессиан логарифма правдоподобия неположительно определен

$$\frac{\partial^2 \log p(t|\boldsymbol{x}, \boldsymbol{w})}{\partial \boldsymbol{w}^2} \leq 0$$

- Это означает, что логарифм функции правдоподобия является вогнутым.
- Логарифм правдоподобия обучающей выборки $L(\boldsymbol{w}) = \log p(\boldsymbol{t}|X, \boldsymbol{w})$, являющийся суммой вогнутых функций, также вогнут, а, значит, имеет единственный максимум

Метод оптимизации Ньютона

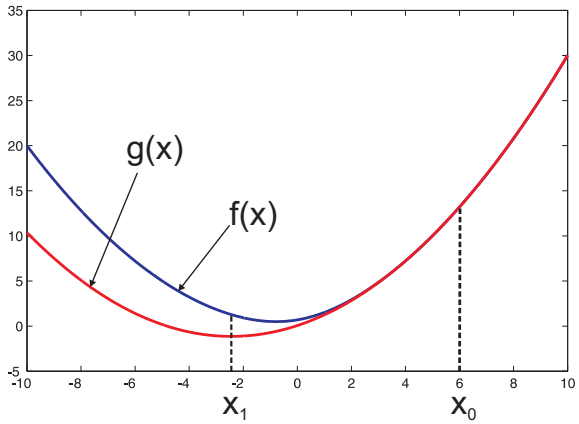
Основная идея метода Ньютона — это приближение в заданной точке оптимизируемой функции параболой и выбор минимума этой параболы в качестве следующей точки итерационного процесса:

$$f(\boldsymbol{x}) \to \min_{\boldsymbol{w}}$$

$$f(\boldsymbol{x}) \simeq g(\boldsymbol{x}) = f(\boldsymbol{x}_0) + (\nabla f(\boldsymbol{x}_0))^T(\boldsymbol{x} - \boldsymbol{x}_0) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_0)^T(\nabla\nabla f(\boldsymbol{x}_0))(\boldsymbol{x} - \boldsymbol{x}_0)$$

$$\nabla g(\boldsymbol{x}_*) = \nabla f(\boldsymbol{x}_0) + (\nabla\nabla f(\boldsymbol{x}_0))(\boldsymbol{x}_* - \boldsymbol{x}_0) = 0 \Rightarrow \boldsymbol{x}_* = \boldsymbol{x}_0 - (\nabla\nabla f(\boldsymbol{x}_0))^{-1}(\nabla f(\boldsymbol{x}_0))$$

Пример. Функция $f(x) = \log(1 + \exp(x)) + \frac{x^2}{5}$.
$x_0 = 6,\ x_1 = -2.4418.$



Итеративная минимизация логарифма правдоподобия

- Так как прямая минимизация правдоподобия невозможна, воспользуемся итерационным методом Ньютона
- Обоснованием корректности использования метода Ньютона является унимодальность оптимизируемой функции $L(\boldsymbol{w})$ и ее гладкость во всем пространстве весов
- Формула пересчета в методе Ньютона

$$\boldsymbol{w}^{new} = \boldsymbol{w}^{old} - H^{-1}\nabla L(\boldsymbol{w}),$$

где $H = \nabla\nabla L(\boldsymbol{w})$ — гессиан логарифма правдоподобия обучающей выборки

Формулы пересчета

Обозначим $s_i = \frac{1}{1+\exp(-t_i y_i)}$, тогда:

$$\nabla L(\boldsymbol{w}) = \Phi^T(\boldsymbol{s} - \boldsymbol{t}^*), \quad \nabla\nabla L(\boldsymbol{w}) = \Phi^T R \Phi$$

$$R = \begin{pmatrix} s_1(1-s_1) & 0 & \dots & 0 \\ 0 & s_2(1-s_2) & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & s_n(1-s_n) \end{pmatrix}$$

$$\boldsymbol{w}^{new} = \boldsymbol{w}^{old} - (\Phi^T R \Phi)^{-1} \Phi^T (\boldsymbol{s} - \boldsymbol{t}^*) =$$

$$(\Phi^T R \Phi)^{-1} \left(\Phi^T R \Phi \boldsymbol{w}^{old} - \Phi^T R R^{-1} (\boldsymbol{s} - \boldsymbol{t}^*) \right) = (\Phi^T R \Phi)^{-1} \Phi^T R \boldsymbol{z},$$

где $\boldsymbol{z} = \Phi \boldsymbol{w}^{old} - R^{-1}(\boldsymbol{s} - \boldsymbol{t}^*)$, $t_i^* = \mathcal{H}(t_i)$

Название метода (метод наименьших квадратов с итеративно пересчитываемыми весами) связано с тем, что последняя формула является формулой для взвешенного МНК (веса задаются диагональной матрицей R), причем на каждой итерации веса корректируются

Заключительные замечания

- На практике матрица $\Phi^T R \Phi$ часто бывает вырождена (всегда при $m > n$), поэтому обычно прибегают к регуляризации матрицы $(\Phi^T R \Phi + \lambda I)$
- Параметр регуляризации λ является структурным параметром
- Базисные функции $\phi_j(\boldsymbol{x})$, а значит и матрица Φ являются структурными параметрами
- С поиском методов автоматического выбора базисных функций связана одна из наиболее интригующих проблем современного машинного обучения

Лекция 2. Графические модели. Общее представление

Ю. И. Журавлев[1], Д. П. Ветров[1]

[1]МГУ, ВМиК, каф. ММП

Курс «Математические основы теории прогнозирования»

План

Ликбез

- Формула Байеса
- Условная независимость случайных величин

Графические модели

- Задачи со структурными ограничениями
- Основные проблемы в анализе графических моделей

Байесовские сети

- Факторизация байесовских сетей
- Три элементарных графа
- Пример использования

Марковские сети

- Потенциалы и энергия клик
- Пример использования
- Связь с байесовскими сетями

План

Условная вероятность

- Пусть X и Y — случайные величины с плотностями $p(x)$ и $p(y)$ соответственно
- В общем случае их совместная плотность $p(x,y) \neq p(x)p(y)$. Если это равенство выполняется, величины называют независимыми
- Условной плотностью называется величина

$$p(x|y) = \frac{p(x,y)}{p(y)}$$

- Смысл: как факт $Y = y$ влияет на распределение X. Заметим, что $\int p(x|y)dx \equiv 1$, но $\int p(x|y)dy$ не обязан равняться единице, т.к. относительно y это не плотность, а функция правдоподобия
- Очевидная система тождеств $p(x|y)p(y) = p(x,y) = p(y|x)p(x)$ позволяет легко переходить от $p(x|y)$ к $p(y|x)$

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

Правило суммирования вероятностей

- Все операции над вероятностями базируются на применении всего двух правил
- Правило суммирования: Пусть $A_1, \ldots, A_k$ взаимоисключающие события, одно из которых всегда происходит. Тогда

$$P(A_i \cup A_j) = P(A_i) + P(A_j) \qquad \sum_{i=1}^{k} P(A_i) = 1$$

- Очевидное следствие (формула полной вероятности): $\forall B$ верно $\sum_{i=1}^{k} P(A_i|B) = 1$, откуда

$$\sum_{i=1}^{k} \frac{P(B|A_i)P(A_i)}{P(B)} = 1 \qquad P(B) = \sum_{i=1}^{k} P(B|A_i)P(A_i)$$

- В интегральной форме

$$p(b) = \int p(b,a)da = \int p(b|a)p(a)da$$

Правило произведения вероятностей

- Правило произведения гласит, что любую совместную плотность всегда можно разбить на множители

$$p(a,b) = p(a|b)p(b) \qquad P(A,B) = P(A|B)P(B)$$

- Аналогично для многомерных совместных распределений

$$p(a_1,\ldots,a_n) =$$

$$p(a_1|a_2,\ldots,a_n)p(a_2|a_3,\ldots,a_n)\ldots p(a_{n-1}|a_n)p(a_n)$$

- Можно показать (Jaynes, 1995), что правила суммирования и произведения вероятностей являются единственными возможными операциями, позволяющими рассматривать вероятности как промежуточную ступень между истиной и ложью

Априорные и апостериорные суждения

- Предположим, мы пытаемся изучить некоторое явление
- У нас имеются некоторые знания, полученные до (лат. a priori) наблюдений/эксперимента. Это может быть опыт прошлых наблюдений, какие-то модельные гипотезы, ожидания
- В процессе наблюдений эти знания подвергаются постепенному уточнению. После (лат. a posteriori) наблюдений/эксперимента у нас формируются новые знания о явлении
- Будем считать, что мы пытаемся оценить неизвестное значение величины θ посредством наблюдений некоторых ее косвенных характеристик $x|\theta$

Формула Байеса

- Знаменитая формула Байеса (1763 г.) устанавливает правила, по которым происходит преобразование знаний в процессе наблюдений
- Обозначим априорные знания о величине θ за $p(\theta)$
- В процессе наблюдений мы получаем серию значений $\boldsymbol{x} = (x_1, \ldots, x_n)$. При разных θ наблюдение выборки $\boldsymbol{x}$ более или менее вероятно и определяется значением правдоподобия $p(\boldsymbol{x}|\theta)$
- За счет наблюдений наши представления о значении θ меняются согласно формуле Байеса

$$p(\theta|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\theta)p(\theta)}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|\theta)p(\theta)}{\int p(\boldsymbol{x}|\theta)p(\theta)d\theta}$$

- Заметим, что знаменатель не зависит от θ и нужен исключительно для нормировки апостериорной плотности

План

Условная независимость случайных величин

- Случайные величины x и y называются условно независимыми от z, если

$$p(x, y|z) = p(x|z)p(y|z)$$

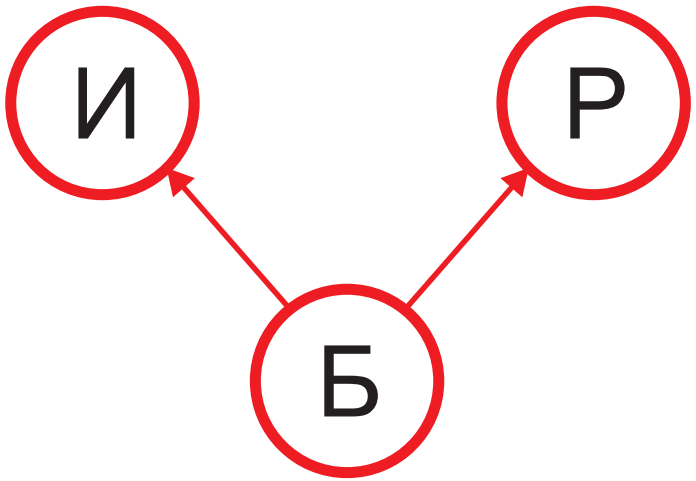
- Другими словами вся информация о взаимозависимостях между x и y содержится в z
- Заметим, что из безусловной независимости не следует условная и наоборот
- Основное свойство условно независимых случайных величин

$$p(z|x, y) = \frac{p(x, y|z)p(z)}{p(x, y)} = \frac{p(x|z)p(y|z)p(z)}{p(x, y)} =$$

$$\frac{p(x|z)p(z)p(y|z)p(z)}{p(x, y)p(z)} = \frac{p(z|x)p(z|y)}{p(z)p(x)p(y)p(x, y)} = \frac{1}{Z}\frac{p(z|x)p(z|y)}{p(z)}$$

Пример

- Рассмотрим следующую гипотетическую ситуацию: римские легионы во главе с императором атакуют вторгшихся варваров
- События «гибель императора» и «уничтожение Рима» не являются независимыми
- Однако, если нам дополнительно известен исход битвы с варварами, эти два события становятся независимыми
- В самом деле, если легионы битву проиграли, то судьба Рима мало зависит от того, был ли император убит в сражении



План

Классическая задача машинного обучения

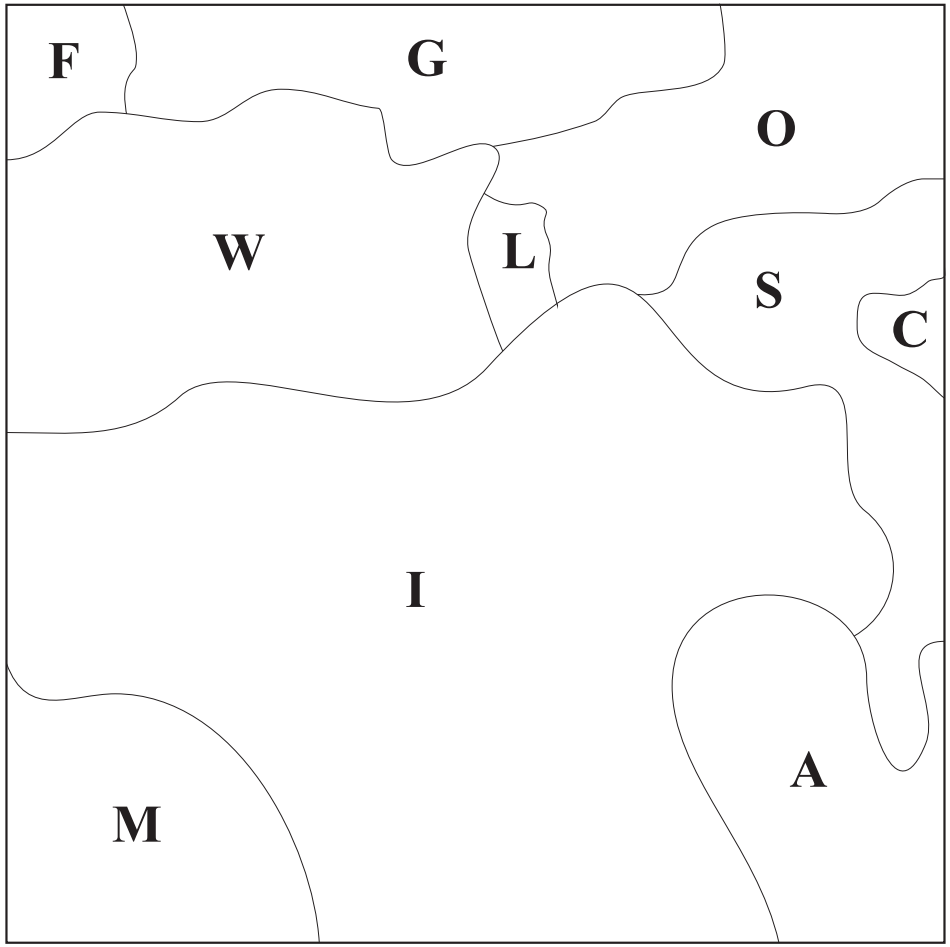
- Задачу машинного обучения можно трактовать как восстановление неизвестных зависимостей между наблюдаемыми переменными X и скрытыми (латентными) переменными T. В случае обучения с учителем такое восстановление производится по обучающей выборке Y
- В классических задачах машинного обучения предполагается, что обучающая выборка сформирована из однородных и независимых объектов $Y = \{(\boldsymbol{x}_i, \boldsymbol{t}_i)\}_{i=1}^n$
- До недавнего времени вероятностные методы обработки данных ограничивались только таким простейшим случаем, а изложение каждого метода начиналось со слов «Предположим, что нам дана выборка из независимых одинаково распределенных случайных величин...»

Задачи со структурными ограничениями

- Во многих задачах взаимосвязи между наблюдаемыми и скрытыми переменными носят сложный характер
- В частности, между отдельными переменными существуют вероятностные зависимости
- Факт зависимости переменных друг от друга удобно отображать с помощью неориентированного графа (марковской сети)
- Если связи между переменными причинно-следственные, то их удобно отображать в виде ориентированных графов (байесовских сетей)
- Основным средством работы с графическими моделями служит аппарат теории вероятностей, в ее байесовской интерпретации

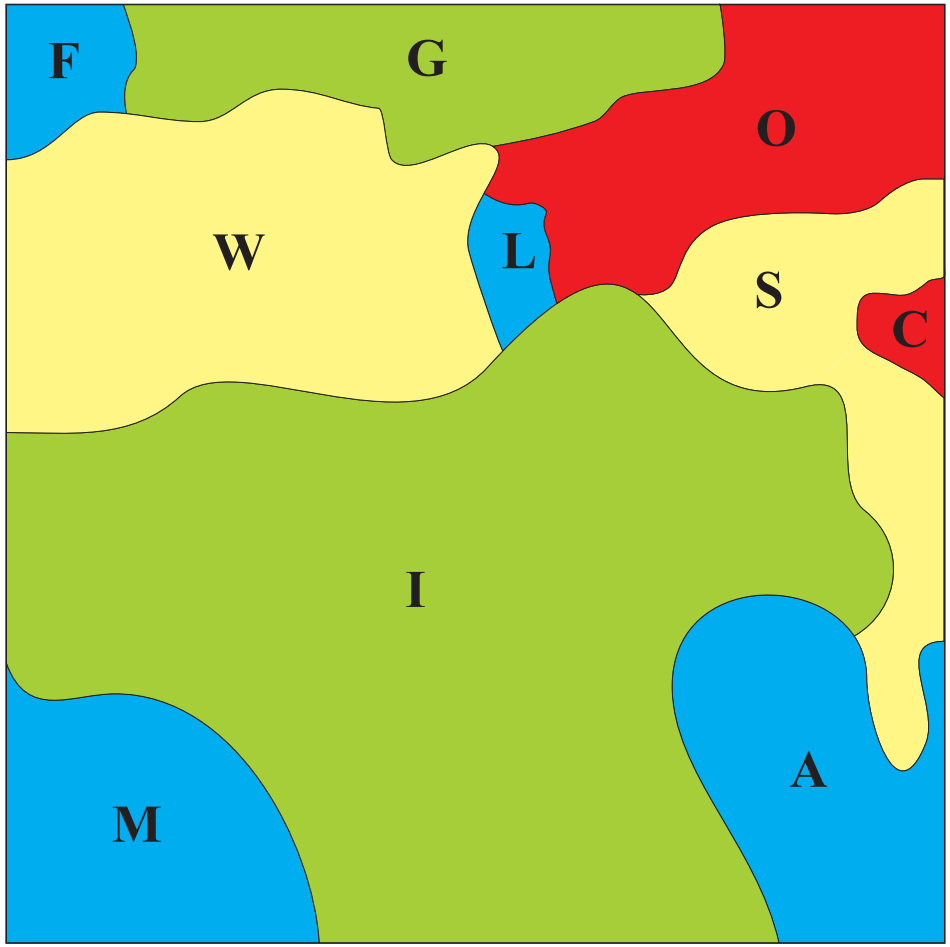
Простой пример

Задача о раскраске областей на плоскости так, чтобы никакие соседние не были окрашены в одинаковый цвет



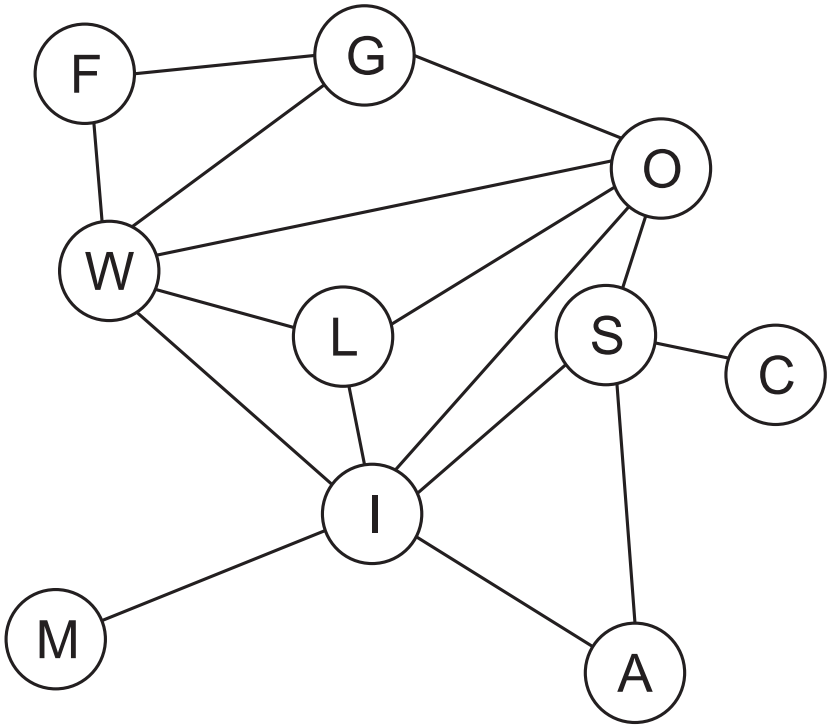
Простой пример

Задача о раскраске областей на плоскости так, чтобы никакие соседние не были окрашены в одинаковый цвет



Простой пример

Такая задача легко формулируется в терминах графической модели, в которой каждая вершина графа может находиться в одном из четырех состояний



Вопрос залу: почему четырех?

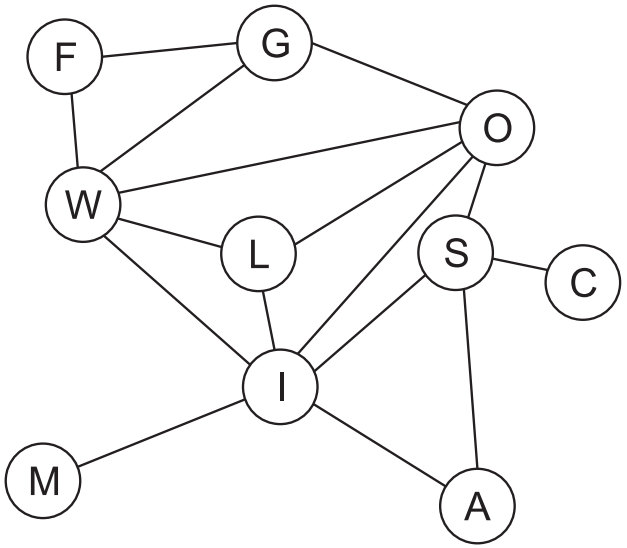
Примеры задач со структурными связями

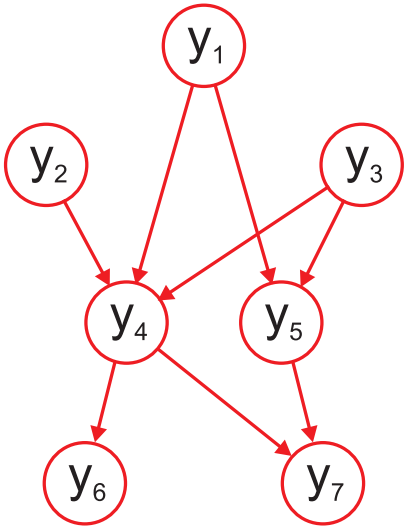
- Обработка изображений, сигналов
- Анализ социальных сетей
- Поиск залежей полезных ископаемых
- Анализ естественных языков
- Биомедицина и биоинформатика
- Веб-поиск
- и др.

План

Графические модели

- Графическая модель представляет собой ориентированный или неориентированный граф
- Вершины графа соответствуют переменным
- Ребра графа соответствуют вероятностным отношениям, определяющим непосредственные зависимости





Главные задачи в анализе графических моделей

Обозначим совокупность наблюдаемых переменных X, а ненаблюдаемых переменных T. Основными задачами в анализе графических моделей являются

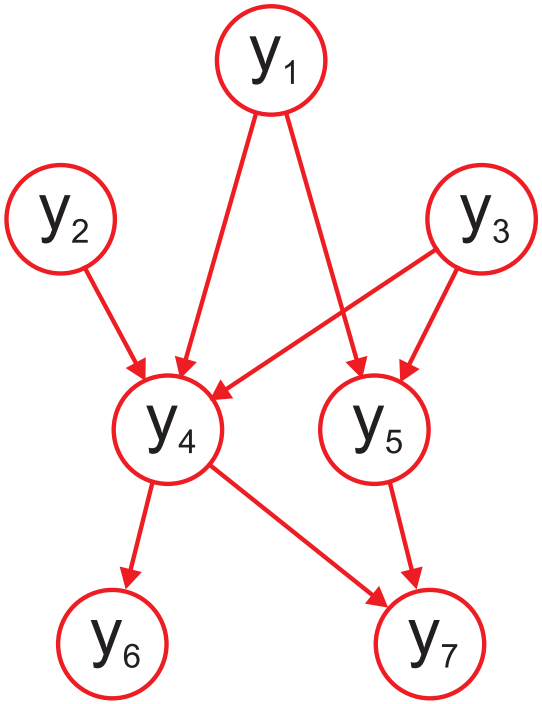
- Подсчет условного распределения на значения отдельной скрытой переменной $p(t_i|X)-?$
- Нахождение наиболее вероятной конфигурации скрытых переменных $p(T|X) \to \max_T$
- Оценка адекватности выбранной графической модели данным $p(X)-?$

Трудности, возникающие при использовании графических моделей

- Не во всех случаях существуют строгие алгоритмы вывода и обучения графических моделей
- Даже там, где они существуют, их применение может оказаться невозможно из-за высоких вычислительных требований и требований к памяти
- В настоящее время в мире активно разрабатываются приближенные эффективные методы обучения и принятия решения в графических моделях (Monte Carlo Markov chains, Variational bounds, Expectation propagation, Belief propagation, и др.)

План

Совместное распределение переменных



Совместное распределение системы переменных задается выражением

$$p(Y) = p(y_1, y_2, y_3, y_4, y_5, y_6, y_7) =$$
$$p(y_1)p(y_2)p(y_3)p(y_4|y_1, y_2, y_3)p(y_5|y_1, y_3)p(y_6|y_4)p(y_7|y_4, y_5).$$

Совместное и условные распределения

- В общем случае совместное распределение для ориентированного графа с n вершинами

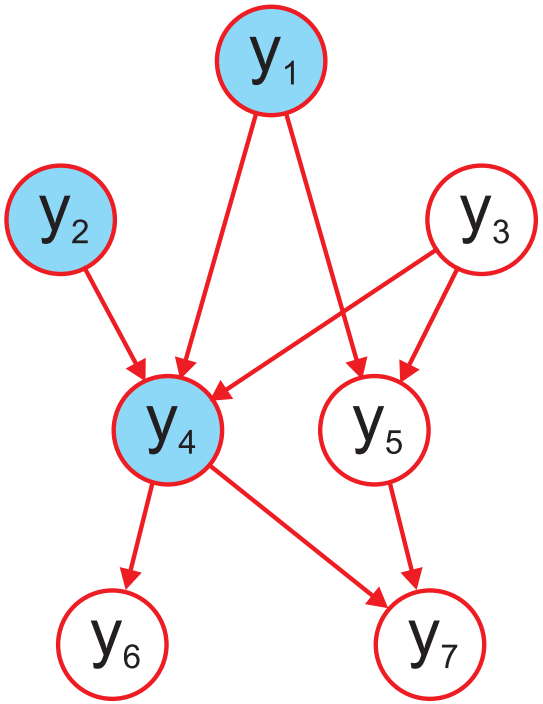
$$p(Y) = \prod_{i=1}^{n} p(y_i|\text{pa}_i),$$

где pa_i — множество вершин-родителей y_i

- Обычно предполагается, что атомарные условные распределения $p(y_i|\text{pa}_i)$ известны
- Зная атомарные распределения, мы можем рассчитать (хотя бы теоретически) любые условные вероятности одних подмножеств переменных по другим подмножествам переменных

Вычисление условных распределений I

- Вернемся к иллюстрации графической модели из семи переменных
- Пусть нам необходимо найти распределение (y_5, y_7) при заданных значениях y_1, y_2, y_4 и неизвестных y_3, y_6



Вычисление условных распределений II

- По определению условной вероятности

$$p(y_5, y_7|y_1, y_2, y_4) = \frac{p(y_1, y_2, y_4, y_5, y_7)}{p(y_1, y_2, y_4)}$$

- Расписываем знаменатель

$$p(y_1, y_2, y_4) = p(y_1)p(y_2)p(y_4|y_1, y_2) = \{Sum\ rule\}$$

$$p(y_1)p(y_2)\int p(y_4|y_1, y_2, y_3)p(y_3)dy_3$$

- Аналогично числитель

$$p(y_1, y_2, y_4, y_5, y_7) = p(y_1)p(y_2)p(y_4|y_1, y_2)p(y_5|y_1)p(y_7|y_5, y_4) = p(y_1)\times$$

$$p(y_2)\left(\int p(y_4|y_1, y_2, y_3)p(y_3)dy_3\right)\left(\int p(y_5|y_1, y_3)p(y_3)dy_3\right)p(y_7|y_5, y_4)$$

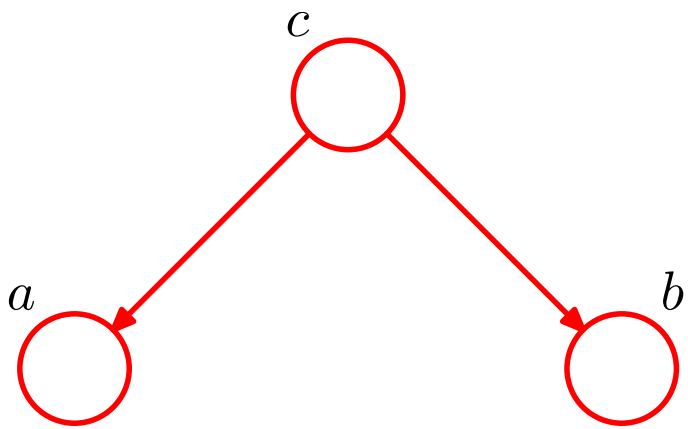
- Для взятия возникающих интегралов обычно пользуются различными аппроксимационными методами
- Таким образом, условное распределение выражено через известные атомарные распределения вида $p(y_i|\mathrm{pa}_i)$

План

Особенности использования байесовских сетей

- По смыслу построения байесовские сети не могут содержать ориентированные циклы, т.к. это будет нарушать правило умножения вероятностей
- Главным достоинством графических моделей является относительно простое выделение условно-независимых величин, которое облегчает дальнейший анализ, позволяя значительно уменьшить количество факторов, влияющих на данную переменную
- В байесовских сетях сделать это несколько сложнее, чем в марковских

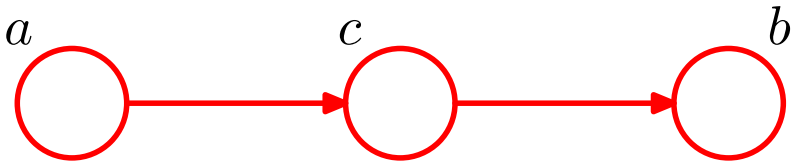
Граф 1



- Аналогия: Рим (a), император (b) и варвары (c)
- Переменные a и b независимы при заданном c
- Возможна маргинализация (исключение переменной)

$$p(a,b) = \int p(a|c)p(b|c)p(c)dc \neq p(a)p(b)$$

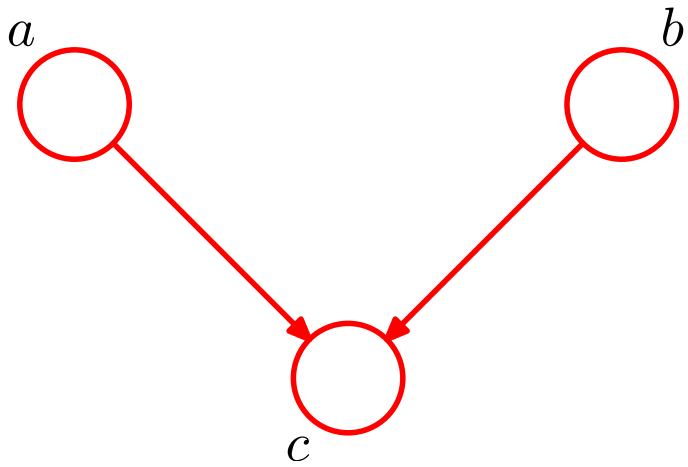
Граф 2



- Аналогия: хорошая работа (a), премия (c), яхта (b)
- Переменные a и b независимы при заданном c
- Возможна маргинализация (исключение переменной)

$$p(a,b) = p(a)\int p(b|c)p(c|a)dc \neq p(a)p(b)$$

Граф 3

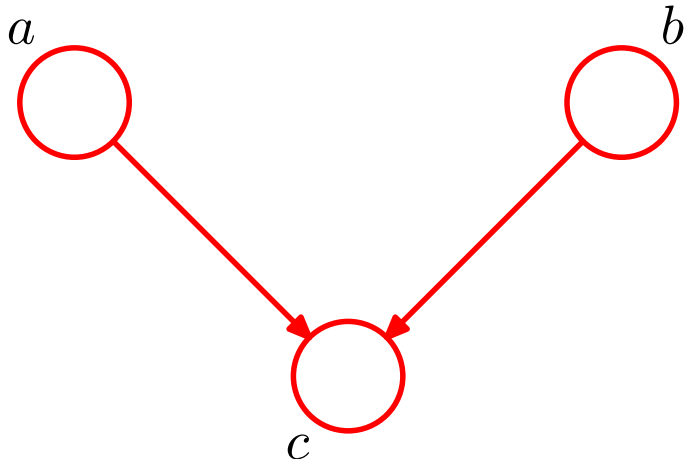


- Аналогия: вор (a), землетрясение (b) и сигнализация (c)
- Переменные a и b независимы, т.е. $p(a,b) = p(a)p(b)$, но не условно независимы!
- Зависимость $p(c|a,b)$ не может быть выражена через $p(c|a)$ и $p(c|b)$, хотя обратное верно

$$p(c|a) = \int p(c|a,b)p(b)db$$

План

Пример нестрогих вероятностных рассуждений I



- Рассмотрим последний граф подробнее. Введем обозначения событий: «сигнализация сработала/не сработала» $(s/\neg s)$, «вор есть/вора нет» $(v/\neg v)$ и «землетрясение произошло/не произошло» $(z/\neg z)$
- Пусть $p(s|v,\neg z) = p(s|v,z) = 1$, $p(s|\neg v,z) = 0.1$, $p(s|\neg v,\neg z) = 0$, $p(v) = 2 \times 10^{-4}$, $p(z) = 10^{-2}$. Графическая модель полностью определена

Пример нестрогих вероятностных рассуждений II

Допустим, мы получили сигнал тревоги. Необходимо оценить вероятность того, что в квартире вор $p(v|s)$

$$p(v|s) = \frac{1}{Z}p(s|v)p(v) = \frac{p(s|v)p(v)}{p(s|v)p(v) + p(s|\neg v)p(\neg v)}$$

$$p(s|\neg v) = p(s|\neg v, z)p(z) + p(s|\neg v, \neg z)p(\neg z) = 10^{-3}$$

$$p(s|v) = 1$$

$$p(v|s) \approx \frac{1}{6}, \quad p(\neg v|s) \approx \frac{5}{6}, \quad Z \approx 1.2 \times 10^{-3}$$

Пример нестрогих вероятностных рассуждений III

- Пусть теперь дополнительно стало известно, что произошло землетрясение. Как изменится вероятность того, что в квартире вор $p(v|s,z)$?

$$p(v|s,z) = \frac{1}{Z} p(s|v,z)p(v|z), \quad p(v|z) = p(v)$$

$$Z = p(s|v,z)p(v|z) + p(s|\neg v,z)p(\neg v|z) =$$

$$1 \times 2 \times 10^{-4} + 0.1 \times (1 - 2 \times 10^{-4}) = 0.1002$$

$$p(v|s,z) = 0.002, \quad p(\neg v|s,z) = 0.998$$

- Заметим, что события z и v перестали быть независимыми, и добавление сведений о значении z меняет знания о значении v. Это называется эффектом оправдания (explaining away)

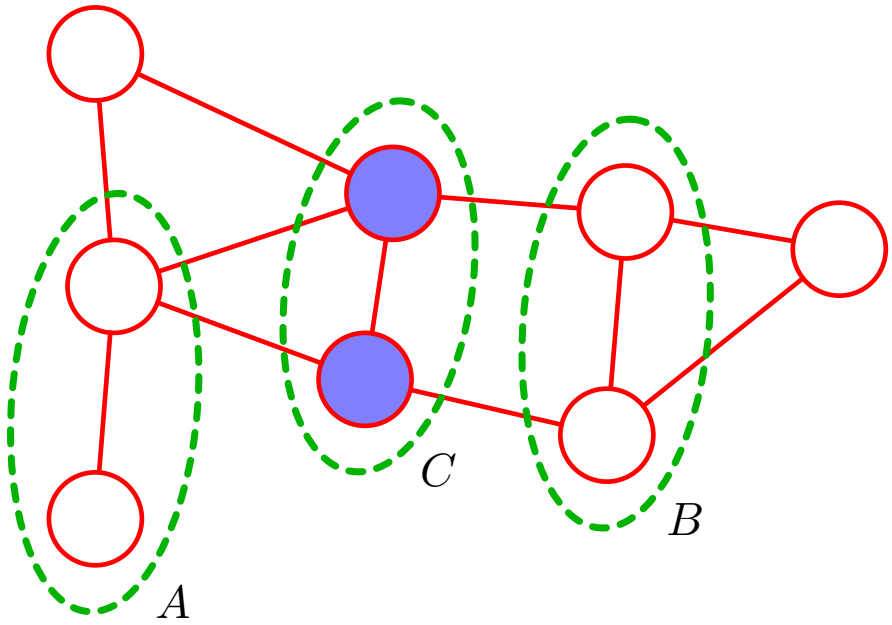
Примеры байесовских сетей

- Скрытые марковские модели
- Фильтр Калмана
- Экспертные системы
- Вероятностный PCA
- Смеси экспертов
- Факторный анализ
- и др.

План

Неориентированные графические модели

- При использовании ориентированных графов определение условной независимости не очень просто
- В марковских сетях это проще. На рисунке A и B независимы при условии C
- Ребра графа связывают переменные, между которыми существуют непосредственные (а не опосредованные) зависимости



Факторизация в марковских сетях

- Пусть y_i и y_j независимы при условии, что все остальные переменные нам известны, т.е. $p(y_i, y_j|Y_{\{i,j\}}) = p(y_i|Y_{\{i,j\}})p(y_j|Y_{\{i,j\}})$
- Это означает, что y_i и y_j не соединены ребром (иначе не было бы условной независимости)
- Запишем совместное распределение и применим правило умножения вероятностей

$$p(Y) = p(y_i, y_j|Y_{\{i,j\}})p(Y_{\{i,j\}}) = p(y_i|Y_{\{i,j\}})p(y_j|Y_{\{i,j\}})p(Y_{\{i,j\}})$$

- Таким образом, переменные, не соединенные ребрами, входят в разные множители совместного распределения

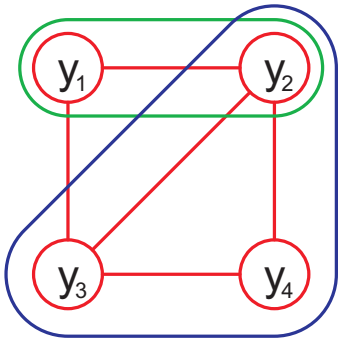
Потенциалы марковской сети

- В общем виде совместное распределение значений элементов сети записывается с помощью неотрицательных потенциальных функций, определенных на максимальных кликах

$$p(Y) = \frac{1}{Z}\prod_C \psi_C(Y_C), \quad Z = \sum_Y \prod_C \psi_C(Y_C), \quad \psi_C(Y_C) \geq 0$$

- На рисунке синяя клика является максимальной, а зеленая — нет. Совместное распределение имеет вид

$$p(Y) = \frac{1}{Z}\psi_1(y_1, y_2, y_3)\psi_2(y_2, y_3, y_4)$$



Потенциальная функция не обязана иметь вероятностную природу, но чем она больше, тем более вероятны соответствующие значения переменных. Обычно потенциальные функции задаются пользователем исходя из априорных предпочтений тех или иных конфигураций переменных. Реже — настраиваются по данным

Энергетическая нотация

- Иногда удобно ввести обозначение $\psi_C(Y_C) = \exp(-E_C(Y_C))$, где $E_C(Y_C)$ имеет смысл энергии
- Тогда задача нахождения наиболее вероятного состояния системы сводится к задаче минимизации полной энергии системы

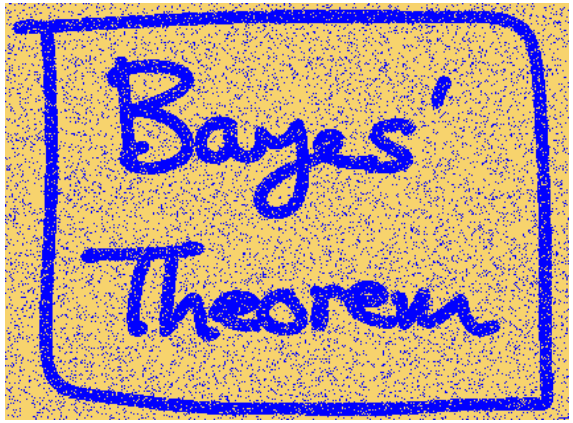
$$\arg\max p(Y) = \arg\max \frac{1}{Z}\prod_C \psi_C(Y_C) =$$

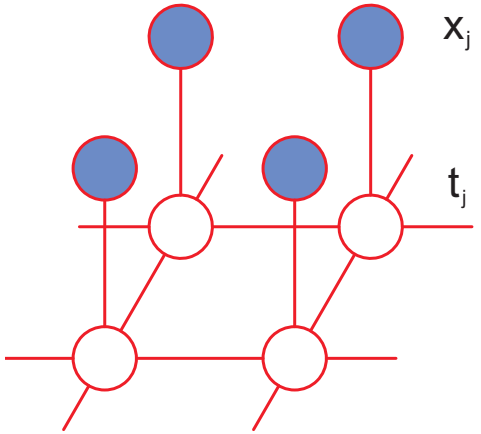
$$\arg\max \exp\left(-\sum_C E_C(Y_C)\right) = \arg\min \sum_C E_C(Y_C)$$

- Заметим, что в отличие от байесовских сетей для полного задания графической модели необходимо знать (или конструктивно уметь подсчитывать) нормировочную константу Z

План

Фильтрация изображений





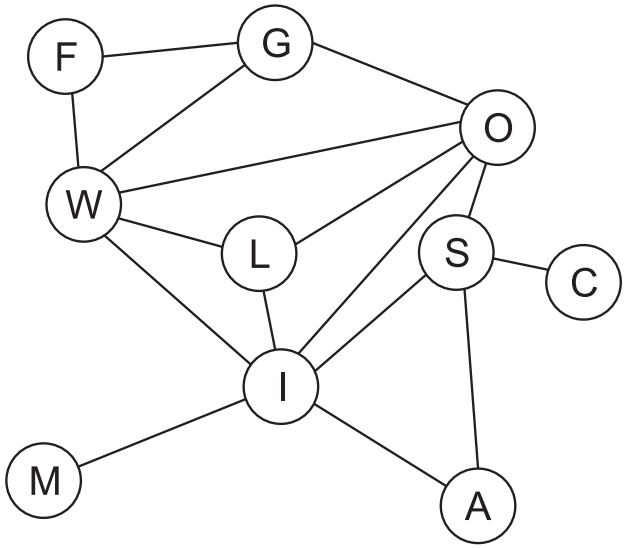
- Рассмотрим задачу фильтрации изображения. Пусть $x_i \in \{-1, 1\}$ — наблюдаемые пиксели бинарного изображения, а $t_i \in \{-1, 1\}$ — истинные значения пикселей
- Введем энергию системы

$$E(X, T) = h \sum_i t_i - \beta \sum_{i,j} t_i t_j - \eta \sum_i t_i x_i,$$

где $h \in \mathbb{R}$ позволяет отразить априорные предпочтения в пользу того или иного цвета (например, указать, что желтый цвет встречается чаще, чем синий), $\beta > 0$ выражает степень зависимости между соседними пикселями, а $\eta > 0$ показывает интенсивность шума

Разметка областей I

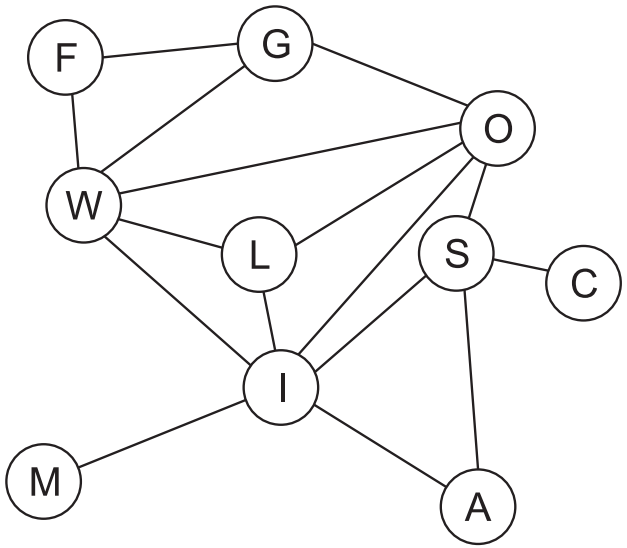
Вернемся к примеру со странами



Совместная плотность задается формулой

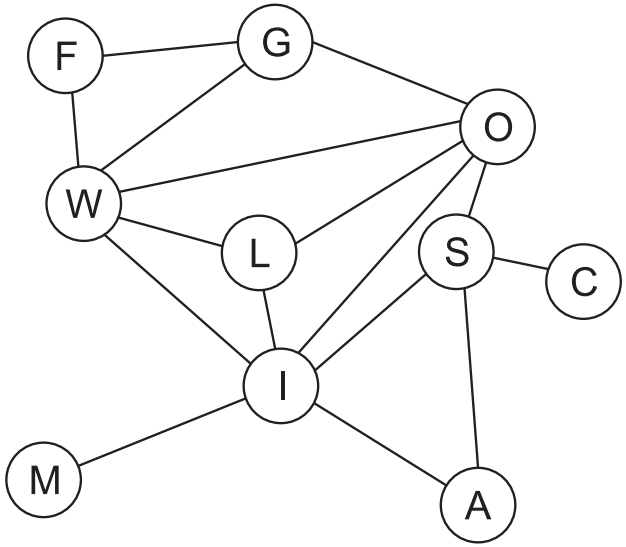
$$p(X) = \frac{1}{Z}\psi_1(F,G,W)\psi_2(G,O,W)\psi_3(W,O,L,I)\psi_4(I,S,O)\times$$
$$\times\,\psi_5(S,I,A)\psi_6(S,C)\psi_7(M,I)$$

Разметка областей II



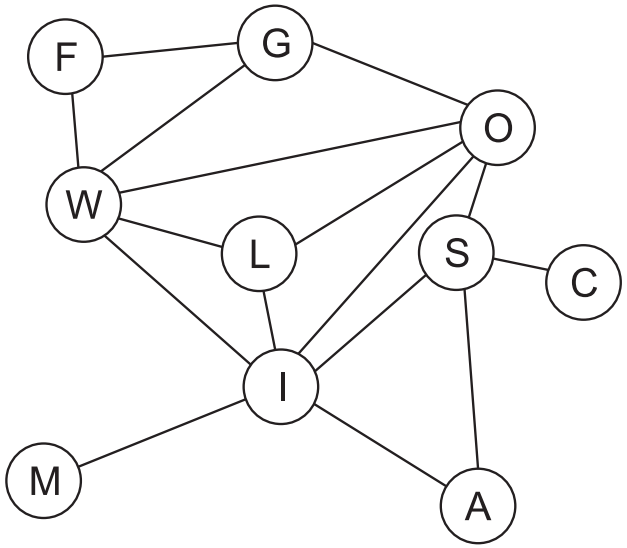
- Предположим, что переменные могут принимать одно из четырех значений $\{red, yellow, blue, green\}$
- Требование несовпадающих цветов областей эквивалентно условию равенства нулю потенциала, если хотя бы два его аргумента имеют одинаковое значение, например $\psi_5(red, blue, red) = 0$

Разметка областей III



Мы можем снизить число нежелаемых цветовых переходов (например, из желтого в красный) снизив соответствующие значения потенциалов $\psi_7(red, yellow)$, $\psi_7(yellow, red)$, $\psi_6(red, yellow)$ и $\psi_6(yellow, red)$

Разметка областей IV



Мы можем искусственно способствовать окраске отдельных регионов в выбранные цвета, вводя индивидуальные множители, например

$$\psi_1(F, G, W) = \phi_1(F, G, W)\phi_2(F)\phi_3(W).$$

Теперь можно увеличить значение $\phi_2(blue)$ и $\phi_3(green)$, чтобы получить политическую карту, привычную российскому глазу

Примеры марковских сетей

- Изображения
- Социальные сети
- Случайные поля
- Карты сайтов
- Условные случайные поля
- и др.

План

Марковские vs. Байесовские сети

Сходства и различия двух типов графических моделей

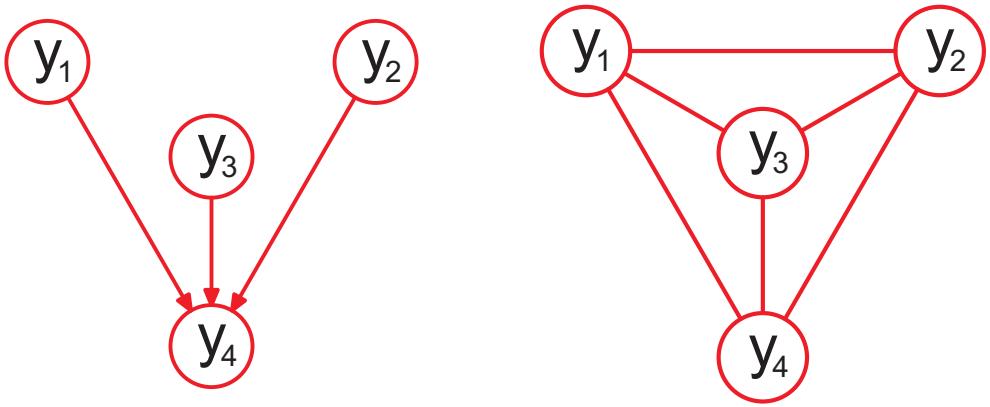
Свойство	Марковские сети	Байесовские сети
Форма	Произв. потенциалов	Произв. потенциалов
Потенциалы	Произвольные	Усл. вероятности
Циклы	Разрешены	Запрещены
Нормировка	$Z = ?$	$Z = 1$
Условная нез-ть	Легко проверяема	Сложнее
Полнота	нет	нет
Анализ	MCMC, BP, и т.д..	Сводить к марковским

Существующее положение дел

- На сегодняшний день существуют эффективные алгоритмы (sum-product, max-product) анализа ациклических графов (деревьев), решающие все три основные задачи анализа графических моделей
- Частным случаем деревьев являются графы-цепочки, характеризующие, например, сигналы во времени
- В случае наличия циклов сложность точных алгоритмов резко возрастает
- Для анализа графов с циклами в основном используются приближенные методы (loopy BP, EP, MCMC)
- В некоторых частных случаях для анализа циклических сетей существуют эффективные точные алгоритмы, например, разрезы графов

Сведение байесовских сетей к марковским

- Наиболее разработаны в настоящее время методы анализа марковских сетей
- Байесовскую сеть можно легко свести к марковской с потерей информации, переженив всех родителей (морализация)
- Заметим, что в приведенном примере все полезные свойства оказались потеряны, и мы получили банальную клику, в которой все зависят от всех



Лекция 3. Скрытые марковские модели. Часть 1

Ю. И. Журавлев[1], Д. П. Ветров[1]

[1]МГУ, ВМиК, каф. ММП

Курс «Математические основы теории прогнозирования»

План

Ликбез

Метод динамического программирования

Основы применения CMM

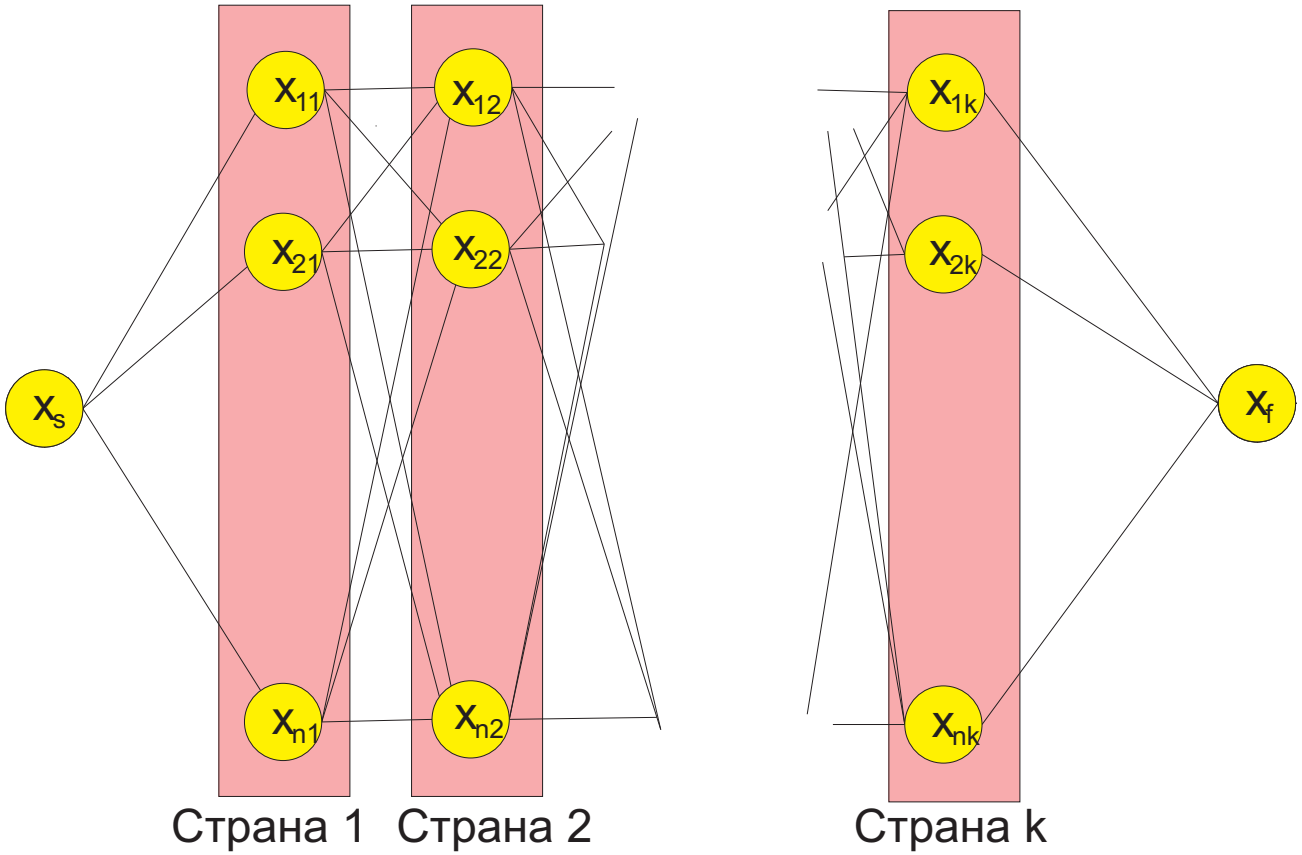
Определение CMM
Обучение CMM с учителем
Алгоритм Витерби

EM-алгоритм

Графические модели с неполными данными
Разделение гауссовской смеси

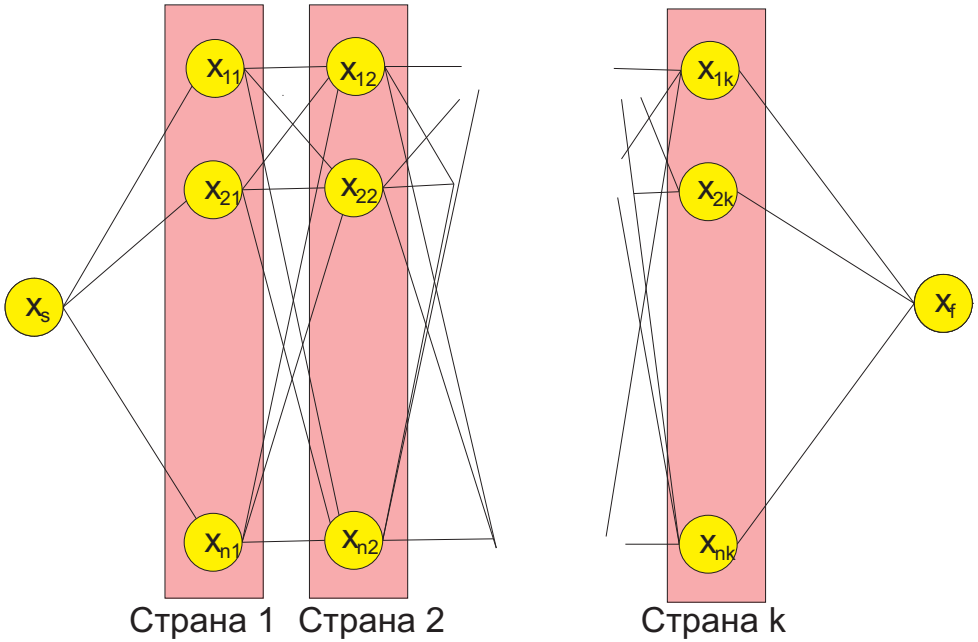
План

Задача объезда стран



- Рассмотрим такую задачу: необходимо в определенной последовательности объехать k стран, в каждой из которых провести одну ночь в отеле, потратив минимум денег
- Если в каждой стране имеется n городов, то задача сводится к перебору n^k вариантов

Функция Беллмана

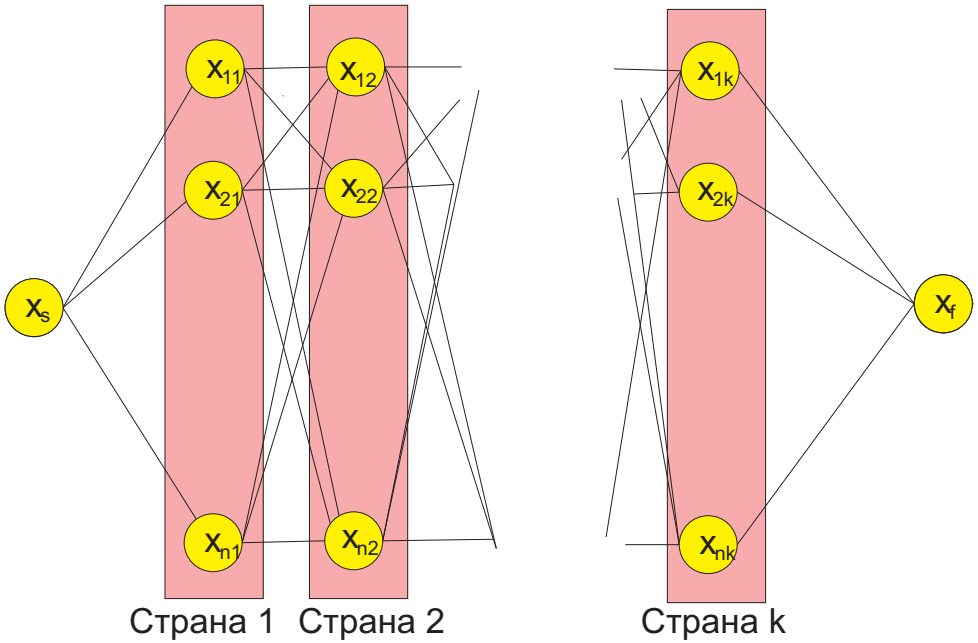


- Пусть цена билета между городами x_{ij} и $x_{i+1,l}$ задается функцией $f(x_{ij}, x_{i+1,l})$, а цена ночлега в городе — функцией $h(x_{ij})$
- Подсчитаем функцию Беллмана $V(x)$, определяемую рекуррентно: $V(x_s) = 0$

$$V(x_{i+1,l}) = \min_j \left[V(x_{ij}) + f(x_{ij}, x_{i+1,l}) + h(x_{i+1,l})\right]$$

- Физический смысл функции Беллмана это наименьшая сумма денег, которую нужно потратить, чтобы добраться из города x_s в город $x_{i+1,l}$ (легко показать по индукции)

Динамическое программирование



- Определим также функцию $S(x)$, возвращающую город, откуда мы приехали в город x: $S(x_s) = \emptyset$

$$S(x_{i+1,l}) = \arg\min_j \left[V(x_{ij}) + f(x_{ij}, x_{i+1,l}) + h(x_{i+1,l})\right]$$

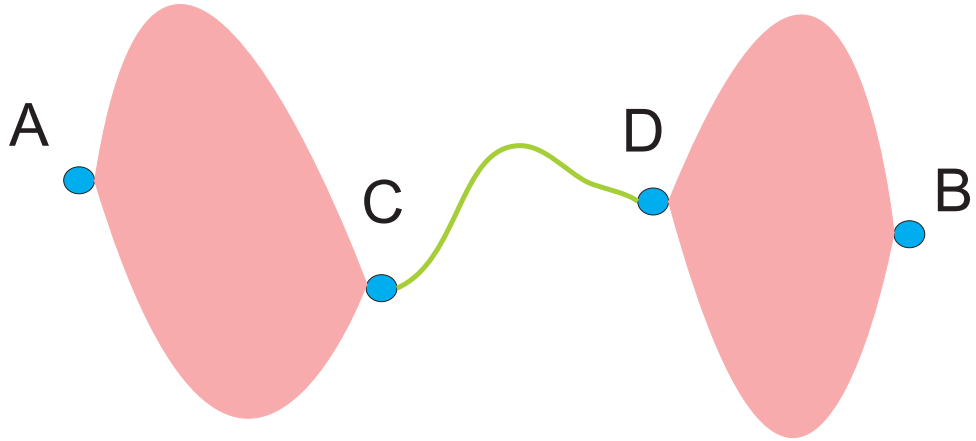
- Тогда оптимальный путь $(x_1^*, x_2^*, \ldots, x_k^*)$ может быть получен путем рекуррентного вызова функции $S(x)$: $x_n^* = S(x_f)$

$$x_m^* = S(x_{m+1}^*)$$

Особенности динамического программирования

Метод динамического программирования эффективен, если имеет место

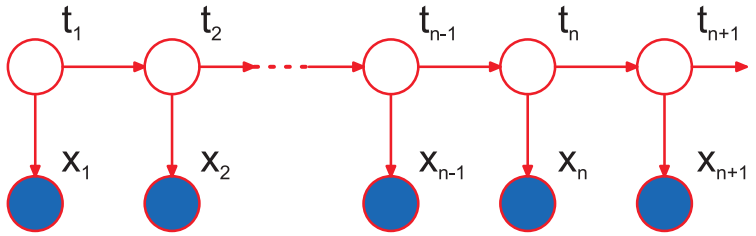
- Перекрывающиеся подзадачи, которые необходимо решить, чтобы решить исходную задачу
- Оптимальная подструктура (принцип кусочной оптимальности)
- Возможность запомнить решения подзадач



Если известно, что оптимальный путь проходит через точки C и D и известен оптимальный путь между ними, то этот путь станет частью оптимального пути между A и B

План

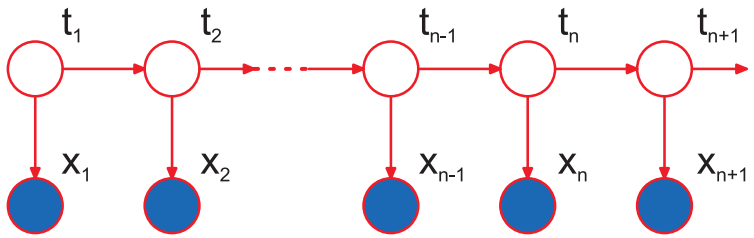
Скрытая Марковская модель (СММ)



Скрытая Марковская модель [первого порядка] — это вероятностная модель последовательности, которая

- Состоит из набора наблюдаемых переменных $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^d$ и латентных (скрытых) переменных $T = \{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N\}$, $\boldsymbol{t}_n \in \{0, 1\}^K$, $\sum_{j=1}^{K} t_{nj} = 1$
- Латентные переменные T являются бинарными и кодируют K состояний, поэтому их иногда называют переменными состояния
- Значение наблюдаемого вектора $\boldsymbol{x}_n$, взятого в момент времени n, зависит только от скрытого состояния $\boldsymbol{t}_n$, которое в свою очередь зависит только от скрытого состояния в предыдущий момент времени $\boldsymbol{t}_{n-1}$

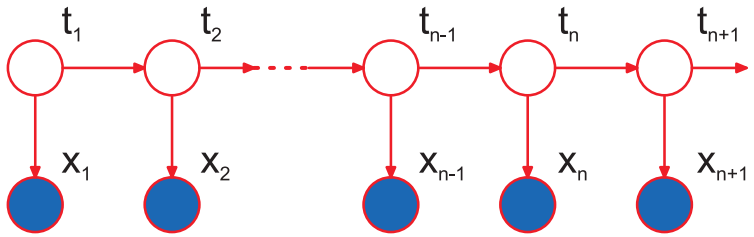
Примеры использования СММ



Что можно анализировать с помощью СММ

- Речь
- Видео
- Поведение
- Фондовые рынки
- Естественный язык
- ДНК
- и др.

Скрытая Марковская модель (СММ)



- Скрытая марковская модель является частным случаем байесовской сети (графической модели, задаваемой ориентированным графом)
- Граф, задающий СММ, является ациклическим, поэтому для СММ существуют эффективные алгоритмы вывода
- Для полного задания модели достаточно задать все условные распределения вида $p(\boldsymbol{x}_n|\boldsymbol{t}_n)$, $p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1})$ и априорное распределение $p(\boldsymbol{t}_1)$

Спецификация вероятностной модели

Пусть имеется K возможных состояний. Закодируем состояние в каждый момент времени n бинарным вектором $\boldsymbol{t}_n = (t_{n1}, \ldots, t_{nK})$, где

$$t_{nj} = \begin{cases} 1, & \text{если в момент } n \text{ модель находится в состоянии } j \\ 0, & \text{иначе} \end{cases}$$

Тогда распределение $p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1})$ можно задать матрицей перехода A размера $K \times K$, где $A_{ij} = p(t_{nj} = 1|t_{n-1,i} = 1)$, $\sum_j A_{ij} = 1$, т.е.

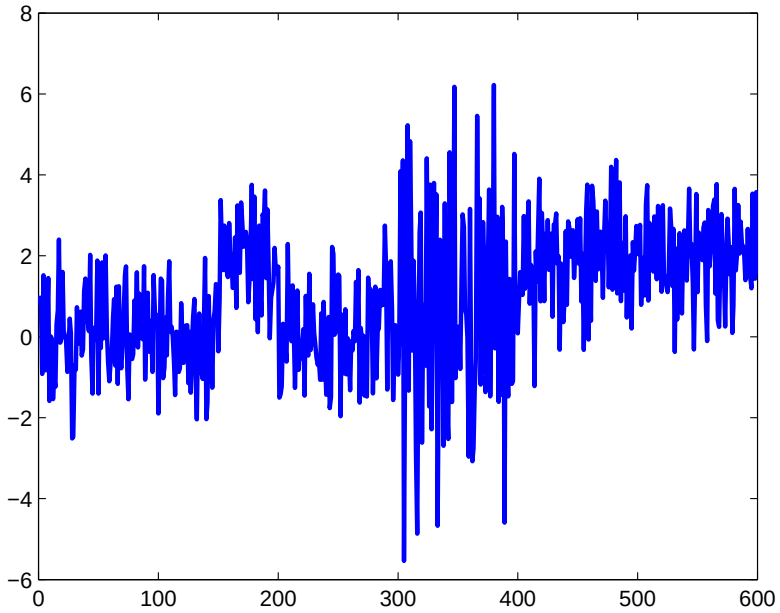
$$p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}) = \prod_{i=1}^{K}\prod_{j=1}^{K} A_{ij}^{t_{n-1,i}t_{nj}}$$

Пусть в первый момент времени $p(t_{1j} = 1) = \pi_j$. Тогда

$$p(\boldsymbol{t}_1) = \prod_{j=1}^{K} \pi_j^{t_{1j}}$$

Замечание

- Хотя матрица A может быть произвольного вида с учетом ограничений на неотрицательность и сумму элементов строки, с точки зрения СММ представляет интерес диагональное преобладание матрицы перехода
- В этом случае можно ожидать, что процесс находится в некотором состоянии на протяжении какого-то отрезка времени
- Появляется простая физическая интерпретация СММ: имеется процесс, который иногда (относительно редко) скачкообразно меняет свои характеристики



Спецификация вероятностной модели

- Условное распределение $p(\boldsymbol{x}_n|\boldsymbol{t}_n)$ определяется текущим состоянием $\boldsymbol{t}_n$
- Обычно предполагают, что оно нам известно с точностью до параметров $\boldsymbol{\phi}_k$, $k \in \{1, \ldots, K\}$, т.е. если $t_{n1} = 1$, то $\boldsymbol{x}_n$ взят из распределения $p(\boldsymbol{x}_n|\boldsymbol{\phi}_1)$, если $t_{n2} = 1$, то $\boldsymbol{x}_n$ взят из распределения $p(\boldsymbol{x}_n|\boldsymbol{\phi}_2)$, и т.д.
- Таким образом

$$p(\boldsymbol{x}_n|\boldsymbol{t}_n) = \prod_{j=1}^{K} (p(\boldsymbol{x}_n|\boldsymbol{\phi}_k))^{t_{nk}}$$

Задачи в СММ

Обозначим полный набор параметров $\Theta = \{\boldsymbol{\pi}, A, \boldsymbol{\phi}\}$. Тогда основные задачи, возникающие в СММ, можно сформулировать следующим образом:

- Обучение с учителем. Известна некоторая последовательность X, для которой заданы T. Задача состоит в оценке по обучающей выборке набора параметров Θ.
- Сегментация. Известна некоторая последовательность X и набор параметров Θ. Задача состоит в получении наиболее правдоподобной последовательности состояний T как $\arg\max_T p(T|X, \Theta)$ (алгоритм Витерби).
- Обучение без учителя. Известна некоторая последовательность X и число состояний K. Задача состоит в оценке параметров Θ (ЕМ-алгоритм).
  - Нахождение маргинального распределения $p(\boldsymbol{t}_n|X, \Theta)$ компоненты $\boldsymbol{t}_n$ по заданым X и Θ
- Прогнозирование. Известна некоторая последовательность X. Задача состоит в оценке наблюдаемого вектора в следующий момент времени $N + 1$ — $p(\boldsymbol{x}_{N+1}|X)$.

План

Совместное распределение переменных СММ

- Предположим, нам задана обучающая выборка (X, T), представляющая собой одну или несколько последовательностей, в которых известны значения скрытых компонент
- Требуется оценить вектор параметров Θ
- По построению байесовской сети совместное распределение переменных задается формулой

$$p(X, T|\Theta) = p_{\boldsymbol{\pi}}(\boldsymbol{t}_1) \prod_{n=1}^{N} p_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{t}_n) \prod_{n=2}^{N} p_A(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}) =$$

$$= \prod_{j=1}^{K} \pi_j^{t_{1j}} \left(\prod_{n=2}^{N} \prod_{i=1}^{K} \prod_{j=1}^{K} A_{ij}^{t_{n-1,i} t_{nj}} \right) \left(\prod_{n=1}^{N} \prod_{k=1}^{K} (p(\boldsymbol{x}_n|\boldsymbol{\phi}_k))^{t_{nk}} \right)$$

Метод максимального правдоподобия

- Для оценки параметров Θ воспользуемся методом максимального правдоподобия

$$\Theta_{ML} = \arg\max p(X, T|\Theta) = \arg\max \log p(X, T|\Theta)$$

- Для удобства перейдем к логарифму функции правдоподобия (положение максимума, очевидно, не изменится)

$$\log p(X, T|\Theta) = \left(\sum_{j=1}^{K} t_{1j} \log \pi_j \right) +$$

$$+ \left(\sum_{n=2}^{N} \sum_{i=1}^{K} \sum_{j=1}^{K} t_{n-1,i} t_{nj} \log A_{ij} \right) + \left(\sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k) \right)$$

Функция Лагранжа

- Параметры, входящие в Θ, не могут принимать произвольные значения, следовательно необходима оптимизация при ограничениях

$$\sum_{j=1}^{K} \pi_j = 1, \quad \sum_{j=1}^{K} A_{ij} = 1, \quad \forall i = \overline{1, K}$$

- Воспользуемся правилом множителей Лагранжа и выпишем лагранжиан

$$\mathcal{L}(\Theta, \lambda, \boldsymbol{\mu}) = \log p(X, T|\Theta) + \lambda \left(\sum_{j=1}^{K} \pi_j - 1 \right) + \sum_{i=1}^{K} \mu_i \left(\sum_{j=1}^{K} A_{ij} - 1 \right) \rightarrow \text{extr}$$

Оценка $\boldsymbol{\pi}$

$$\log p(X, T|\Theta) = \left(\sum_{j=1}^{K} t_{1j} \log \pi_j\right) +$$

$$+ \left(\sum_{n=2}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K} t_{n-1,i} t_{nj} \log A_{ij}\right) + \left(\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)\right)$$

$$\mathcal{L}(\Theta, \lambda, \boldsymbol{\mu}) = \log p(X, T|\Theta) + \lambda\left(\sum_{j=1}^{K} \pi_j - 1\right) + \sum_{i=1}^{K} \mu_i \left(\sum_{j=1}^{K} A_{ij} - 1\right) \to \text{extr}$$

$$\frac{\partial \mathcal{L}(\Theta, \lambda, \boldsymbol{\mu})}{\partial \pi_j} = \frac{t_{1j}}{\pi_j} + \lambda = 0 \;\Rightarrow\; \pi_j = -\frac{t_{1j}}{\lambda}$$

$$\sum_{j=1}^{K} \pi_j = 1 \;\Rightarrow\; \lambda = -\sum_{j=1}^{K} t_{1j} = -1$$

$$\pi_j = t_{1j}$$

Оценка матрицы A

$$\log p(X, T|\Theta) = \left(\sum_{j=1}^{K} t_{1j} \log \pi_j\right) +$$

$$+ \left(\sum_{n=2}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K} t_{n-1,i} t_{nj} \log A_{ij}\right) + \left(\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)\right)$$

$$\mathcal{L}(\Theta, \lambda, \boldsymbol{\mu}) = \log p(X, T|\Theta) + \lambda\left(\sum_{j=1}^{K} \pi_j - 1\right) + \sum_{i=1}^{K} \mu_i \left(\sum_{j=1}^{K} A_{ij} - 1\right) \to \text{extr}$$

$$\frac{\partial \mathcal{L}(\Theta, \lambda, \boldsymbol{\mu})}{\partial A_{ij}} = \sum_{n=2}^{N} \frac{t_{n-1,i} t_{nj}}{A_{ij}} + \mu_i = 0 \;\Rightarrow\; A_{ij} = -\sum_{n=2}^{N} \frac{t_{n-1,i} t_{1j}}{\mu_i}$$

$$\sum_{j=1}^{K} A_{ij} = 1 \;\Rightarrow\; \mu_i = -\sum_{n=2}^{N}\sum_{j=1}^{K} t_{n-1,i} t_{nj} = -\sum_{n=2}^{N} t_{n-1,i}$$

$$A_{ij} = \frac{\sum_{n=2}^{N} t_{n-1,i} t_{nj}}{\sum_{n=2}^{N} t_{n-1,i}}$$

Оценка условной плотности $p(\boldsymbol{x}|\boldsymbol{t})$

$$\log p(X,T|\Theta) = \left(\sum_{j=1}^{K} t_{1j}\log\pi_j\right) +$$

$$+\left(\sum_{n=2}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K} t_{n-1,i}t_{nj}\log A_{ij}\right) + \left(\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)\right)$$

$$\mathcal{L}(\Theta,\lambda,\boldsymbol{\mu}) = \log p(X,T|\Theta) + \lambda\left(\sum_{j=1}^{K}\pi_j - 1\right) + \sum_{i=1}^{K}\mu_i\left(\sum_{j=1}^{K} A_{ij} - 1\right) \to \text{extr}$$

$$\frac{\partial\mathcal{L}(\Theta,\lambda,\boldsymbol{\mu})}{\partial\boldsymbol{\phi}_k} = \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\frac{\partial\log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)}{\boldsymbol{\phi}_k} = \sum_{\{n:\ t_{nk}=1\}}\frac{\partial\log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)}{\boldsymbol{\phi}_k} = 0$$

Получили стандартную задачу максимизации правдоподобия по выборке независимых одинаково-распределенных объектов

$$\boldsymbol{\phi}_k = \arg\max\sum_{\{n:\ t_{nk}=1\}}\log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)$$

Для оценки параметров $\boldsymbol{\phi}_k$ можно воспользоваться методами восстановления плотностей, например, EM-алгоритмом

План

Сегментация сигнала

- Пусть известна некоторая последовательность наблюдений X и набор параметров СММ Θ. Требуется определить наиболее вероятную последовательность состояний T, т.е. найти $\arg\max_T p(T|X,\Theta)$
- Заметим, что $p(X|\Theta)$ не зависит от T, поэтому

$$\arg\max_T p(T|X,\Theta) = \arg\max_T \frac{p(X,T|\Theta)}{p(X|\Theta)} =$$
$$= \arg\max_T p(X,T|\Theta) = \arg\max_T \log p(X,T|\Theta)$$

- Но это же классическая задача динамического программирования!

Аналогия с задачей объезда стран

- В самом деле логарифм совместной плотности по определению

$$\log p(X, T|\Theta) = \left(\sum_{j=1}^{K} t_{1j} \log \pi_j\right) +$$

$$+ \left(\sum_{n=2}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K} t_{n-1,i} t_{nj} \log A_{ij}\right) + \left(\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)\right)$$

- Первое слагаемое определяет «пункт отбытия», второе слагаемое — стоимость перезда из города страны $n-1$ в город в стране n, а третье слагаемое отражает «стоимость ночлега» в выбранном городе страны n

Алгоритм Витерби

$$\log p(X, T|\Theta) = \left(\sum_{j=1}^{K} t_{1j} \log \pi_j\right) +$$

$$+ \left(\sum_{n=2}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K} t_{n-1,i} t_{nj} \log A_{ij}\right) + \left(\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)\right)$$

Функция Беллмана $V_{1j} = \log \pi_j$

$$V_{nj} = \max_i \left[V_{n-1,i} + \log A_{ij} + \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_j)\right]$$

Функция S_{nj} определяется аналогично: $S_{1j} = \emptyset$

$$S_{nj} = \arg\max_i \left[V_{n-1,i} + \log A_{ij} + \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_j)\right]$$

Алгоритм Витерби

$$V_{nj} = \max_i \left[V_{n-1,i} + \log A_{ij} + \log p(\boldsymbol{x}_n | \boldsymbol{\phi}_j) \right]$$

$$S_{nj} = \arg\max_i \left[V_{n-1,i} + \log A_{ij} + \log p(\boldsymbol{x}_n | \boldsymbol{\phi}_j) \right]$$

Выполнив прямой проход по сигналу, мы оцениваем V_{nj} и S_{nj}, а выполнив обратный проход, мы получаем оптимальные номера оптимальных состояний $(i^*(1), \ldots, i^*(N))$: $i^*(N) = \arg\max_i V_{Ni}$

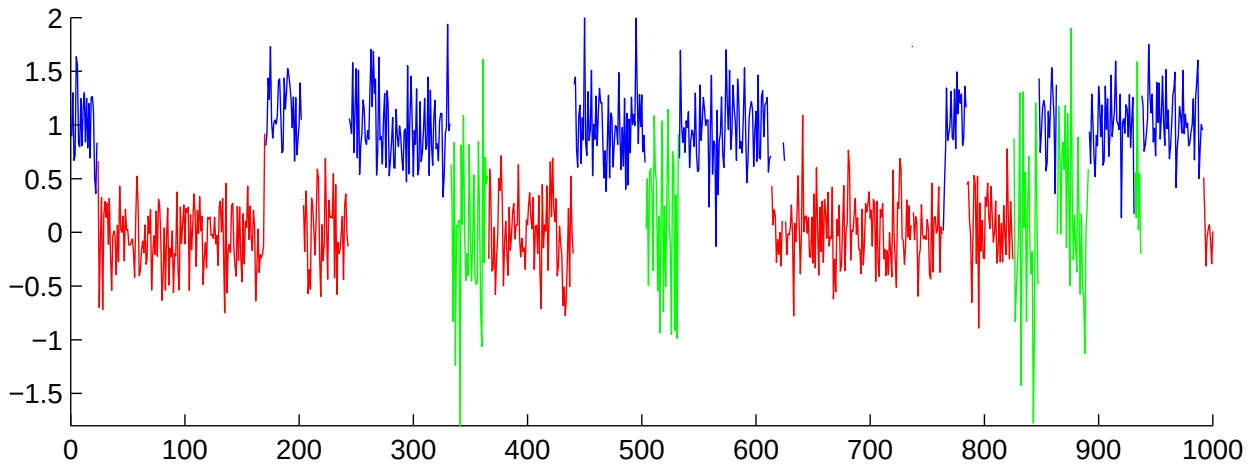
$$i^*(n) = S_{n+1,i^*(n+1)}$$

Легко видеть, что значения переменных $\boldsymbol{t}_n$ определяются так: $t_{n,i^*(n)} = 1$, $t_{ni} = 0$, $\forall i \neq i^*(n)$

- Алгорим Витерби позволяет быстро проводить сегментацию очень длинных сигналов
- Существует версия алгоритма Витерби, позволяющая осуществлять сегментацию в реальном времени (точнее, с небольшой задержкой)

Пример использования

Пример разметки сигнала на три состояния с помощью алгоритма Витерби

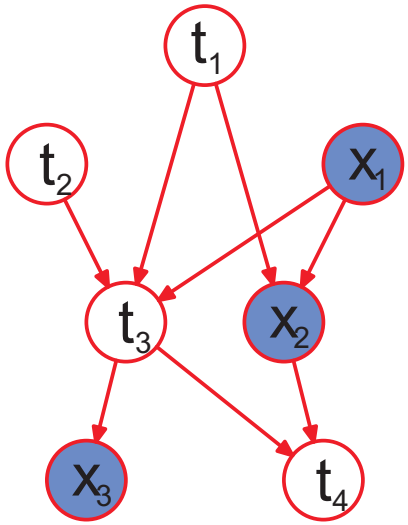


План

Максимум неполного правдоподобия

- Предположим имеется графическая модель, в которой известна только часть значений переменных
- Атомарные распределения известны с точностью до вектора параметров $\boldsymbol{\theta}$
- Требуется оценить параметры по наблюдаемым величинам с помощью метода максимального правдоподобия, т.е. найти

$$\boldsymbol{\theta}_{ML} = \arg\max p(X|\boldsymbol{\theta})$$



Трудности оптимизации неполного правдоподобия

- По правилу суммирования вероятностей неполное правдоподобие может быть получено в виде суммирования по скрытым переменным полного правдоподобия

$$p(X|\boldsymbol{\theta}) = \sum_T p(X, T|\boldsymbol{\theta})$$

- Во многих случаях (в частности, в байесовских сетях) подсчет полного правдоподобия тривиален
- При оптимизации правдоподобия удобно переходить к логарифму, в частности выше мы получили явные формулы для $\arg\max_{\boldsymbol{\theta}} p(X, T|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \log p(X, T|\boldsymbol{\theta})$ в СММ
- Прямая оптимизация логарифма неполного правдоподобия очень затруднительна даже в итерационной форме, т.к. функционал имеет вид «логарифм суммы», в то время как удобно оптимизировать «сумму логарифмов»

$$\log p(X|\boldsymbol{\theta}) = \log \sum_T p(X, T|\boldsymbol{\theta})$$

Схема EM-алгоритма

- На входе: выборка X, зависящая от набора параметров $\boldsymbol{\theta}$
- Инициализируем $\boldsymbol{\theta}$ некоторыми начальным приближением
- E-шаг: Оцениваем распределение скрытой компоненты при фиксированном значении параметров $\boldsymbol{\theta}_{old}$

$$p(T|X,\boldsymbol{\theta}_{old}) = \frac{p(X,T|\boldsymbol{\theta}_{old})}{\sum_T p(X,T|\boldsymbol{\theta}_{old})}$$

- M-шаг: Оптимизируем

$$\mathbb{E}_{T|X,\boldsymbol{\theta}_{old}} \log p(X,T|\boldsymbol{\theta}) = \sum_T p(T|X,\boldsymbol{\theta}_{old}) \log p(X,T|\boldsymbol{\theta}) \to \max_{\boldsymbol{\theta}}$$

Если бы мы точно знали значение $T = T_0$, то вместо мат. ожидания по всевозможным (с учетом наблюдаемых данных) $T|X,\boldsymbol{\theta}_{old}$, мы бы оптимизировали $\log p(X,T_0|\boldsymbol{\theta})$

- Переход к E-шагу, пока процесс не сойдется

Замечания

- Оптимизация проводится итерационно методом покоординатного спуска: на каждой итерации последовательно уточняются возможные значения T (E-шаг), а потом пересчитываются значения $\boldsymbol{\theta}$ (M-шаг)
- Во многих случаях на M-шаге можно получить явные формулы, т.к. там происходит оптимизация выпуклой комбинации логарифмов полных правдоподобий

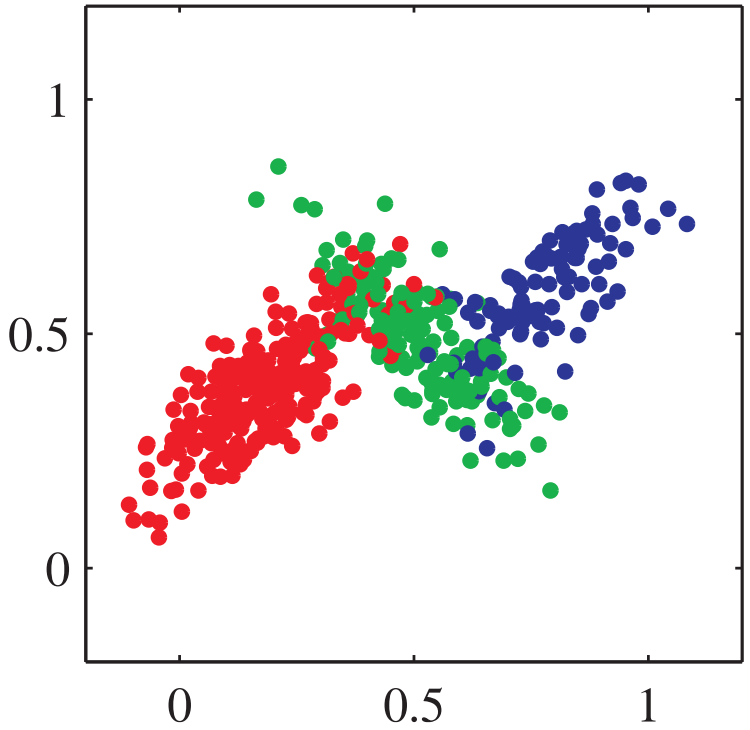
$$\sum_T p(T|X, \boldsymbol{\theta}_{old}) \log p(X, T|\boldsymbol{\theta}) \to \max_{\boldsymbol{\theta}},$$

имеющей вид взвешенной «суммы логарифмов»

План

Смесь гауссовских распределений

- Имеется выборка $X \sim \sum_{j=1}^{l} w_j \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \Sigma_j)) \in \mathbb{R}^d$, $w_j \geq 0$, $\sum_{j=1}^{l} w_j = 1$
- Требуется восстановить плотность генеральной совокупности



EM-алгоритм

- Выбираем начальное приближение $\boldsymbol{\mu}_j$, w_j, Σ_j
- E-шаг: Вычисляем распределение скрытых переменных $\boldsymbol{z}_i \in \{0,1\}$, $\sum_j z_{ij} = 1$, которые определяют, к какой компоненте смеси принадлежит объект $\boldsymbol{x}_i$

$$\gamma(z_{ij}) = \frac{w_j \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j, \Sigma_j))}{\sum_{k=1}^{l} w_k \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k, \Sigma_k))}$$

- M-шаг: С учетом новых вероятностей на $\boldsymbol{z_i}$, пересчитываем параметры смеси

$$\boldsymbol{\mu}_j^{new} = \frac{1}{N_j} \sum_{i=1}^{n} \gamma(z_{ij}) \boldsymbol{x}_i \qquad w_j^{new} = \frac{N_j}{n} \qquad N_j = \sum_{i=1}^{n} \gamma(z_{ij})$$

$$\Sigma_j^{new} = \frac{1}{N_j} \sum_{i=1}^{n} \gamma(z_{ij}) (\boldsymbol{x}_i - \boldsymbol{\mu}_j^{new}) (\boldsymbol{x}_i - \boldsymbol{\mu}_j^{new})^T$$

- Переход к E-шагу, пока не будет достигнута сходимость

Вывод формул на M-шаге

Выражение для мат. ожидания логарифма правдоподобия по апостериорному распределению имеет вид

$$\mathbb{E}_{Z|X,\boldsymbol{\theta}} \log p(X, Z|\boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{j=1}^{l} \gamma(z_{ij}) \left(\log w_j + \log \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_j, \Sigma_j)\right)$$

Дифференцирование по $\boldsymbol{\mu}_j$ и Σ_j (точнее по Σ_j^{-1}) выполняется аналогично случаю одной многомерной гауссианы, только объекты теперь берутся с весом $\gamma(z_{ij})$.

Оптимизация по w_j проводится с учетом органичения $\sum_{j=1}^{l} w_j = 1$ по правилу множителей Лагранжа

$$\sum_{i=1}^{n} \sum_{j=1}^{l} \gamma(z_{ij}) \log w_j + \lambda \left(\sum_{j=1}^{l} w_j - 1 \right)$$

Дифференцируя лагранжиан по w_j получаем уравнения

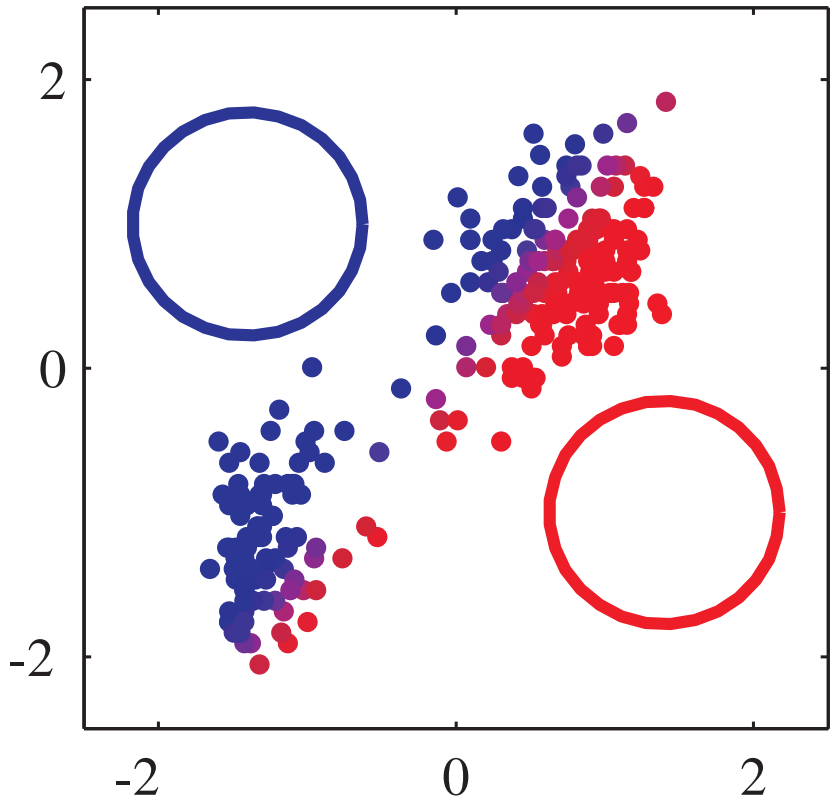
$$\frac{\sum_{i=1}^{n} \gamma(z_{ij})}{w_j} + \lambda = 0, \quad w_j = -\frac{\sum_{i=1}^{n} \gamma(z_{ij})}{\lambda}$$

Учитывая, что $\sum_{j=1}^{l} w_j = 1$ окончательно получаем

$$\lambda = -n, \quad w_j = \frac{\sum_{i=1}^{n} \gamma(z_{ij})}{n} = \frac{N_j}{n}$$

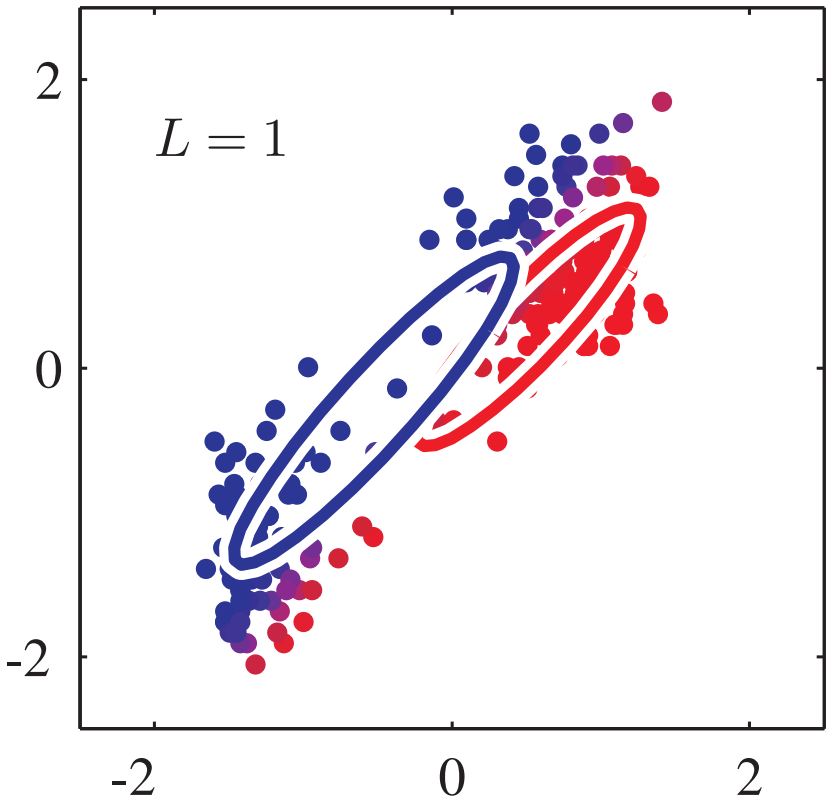
Пример работы EM-алгоритма

Применение EM-алгоритма для разделения смеси двух гауссиан. Итерация 0.



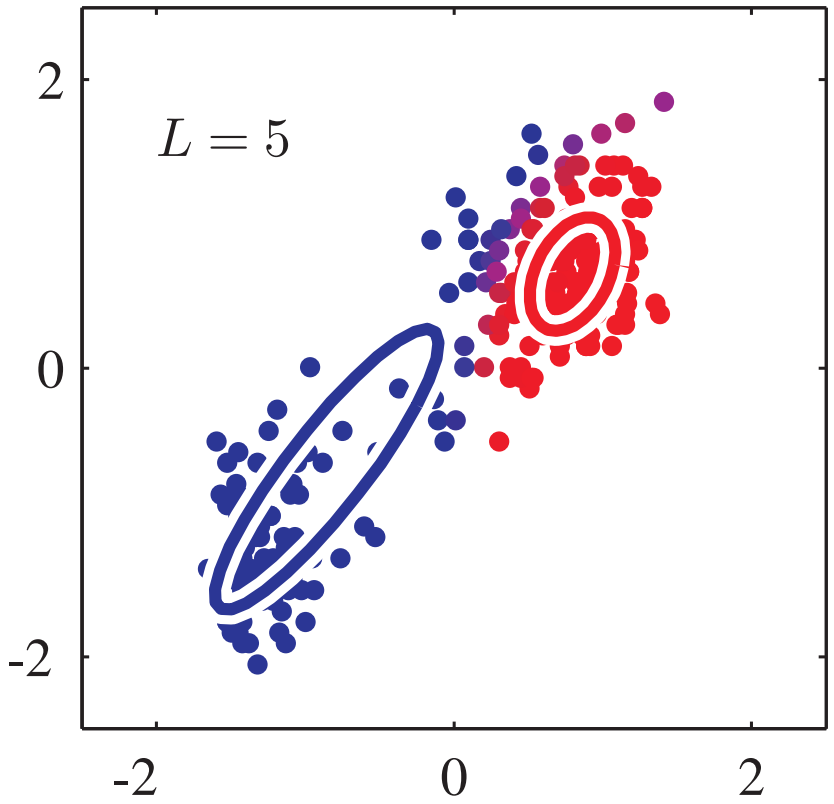
Пример работы EM-алгоритма

Применение EM-алгоритма для разделения смеси двух гауссиан. Итерация 1.



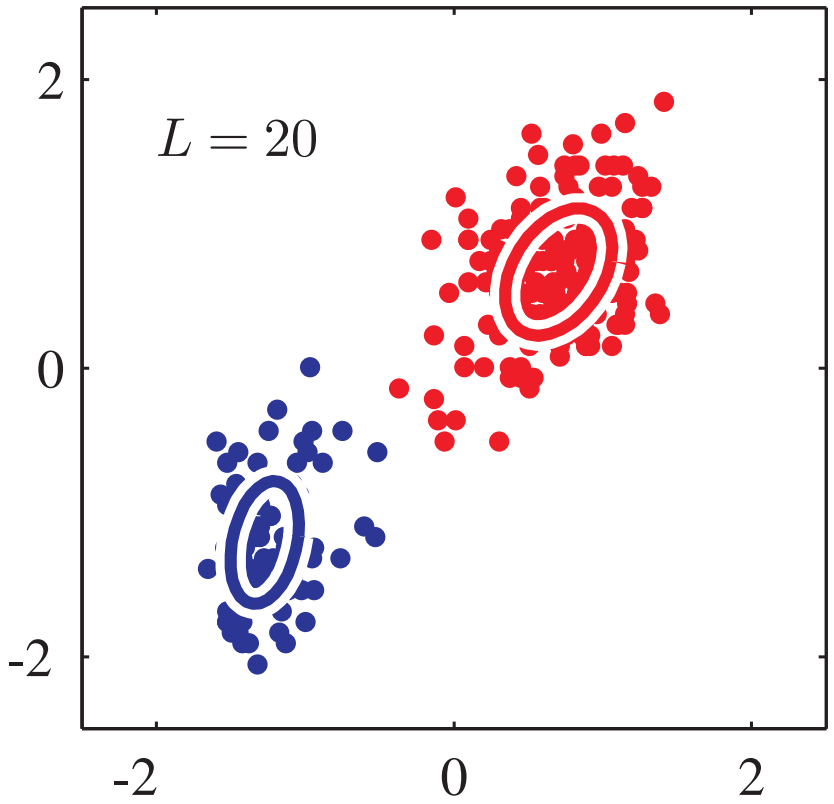
Пример работы EM-алгоритма

Применение EM-алгоритма для разделения смеси двух гауссиан. Итерация 5.



Пример работы EM-алгоритма

Применение EM-алгоритма для разделения смеси двух гауссиан. Итерация 20.

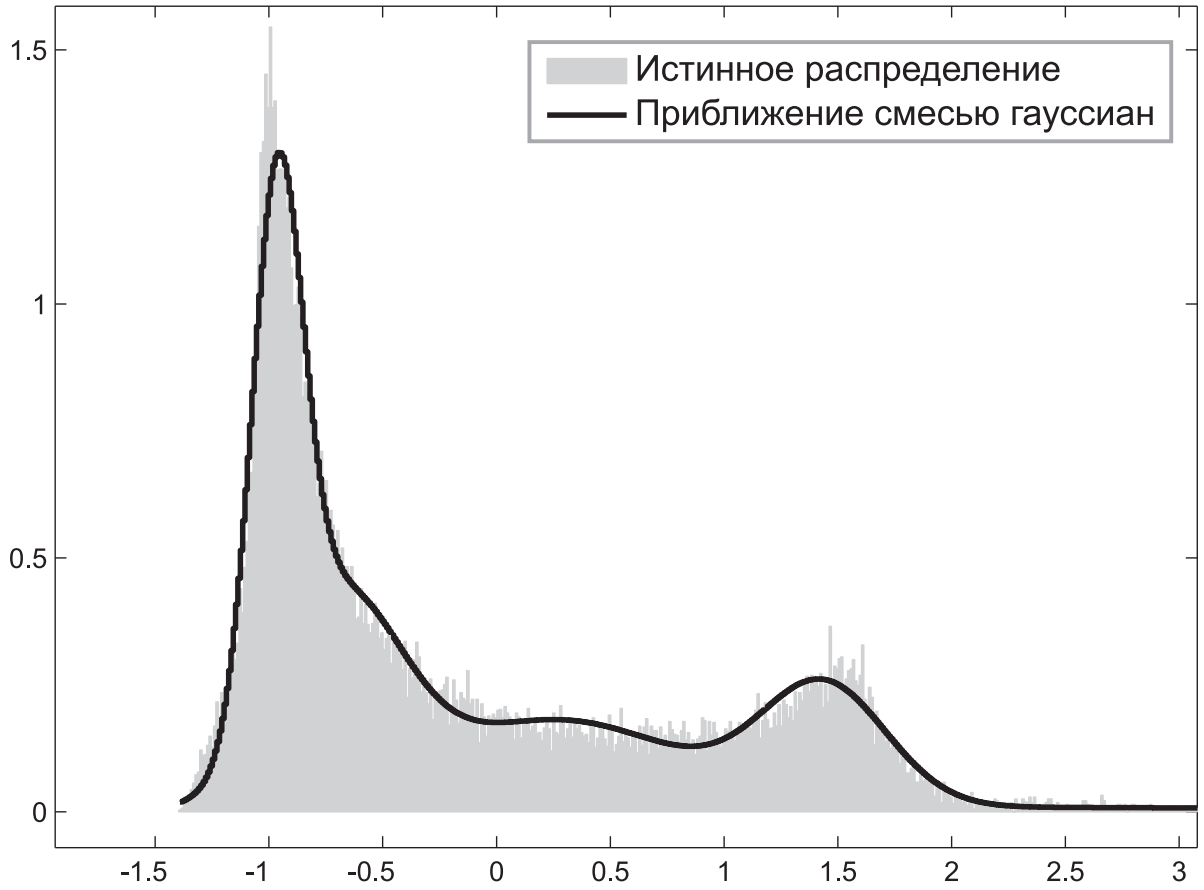


Недостатки EM-алгоритма

- В зависимости от выбора начального приближения может сходиться к разным точкам
- EM-алгоритм находит локальный экстремум, в котором значение правдоподобия может оказаться намного ниже, чем в глобальном максимуме
- EM-алгоритм не позволяет определить количество компонентов смеси l
- !! Величина l является структурным параметром!!

Применение EM-алгоритма для описания сложных плотностей

Смесью гауссиан можно эффективно приближать сложные, плохо-параметризуемые плотности. При этом получаются аналитически заданные приближения



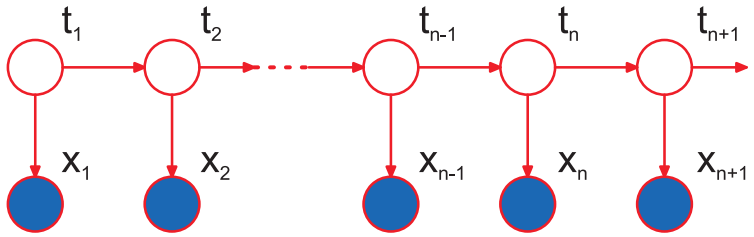
Лекция 4. Скрытые марковские модели. Часть 2.

Ю. И. Журавлев[1], Д. П. Ветров[1]

[1]МГУ, ВМиК, каф. ММП

Курс «Математические основы теории прогнозирования»

Скрытая Марковская модель (СММ)



Скрытая Марковская модель [первого порядка] — это вероятностная модель последовательности, которая

- Состоит из набора наблюдаемых переменных $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ и латентных (скрытых) переменных $T = \{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N\}$.
- Латентные переменные T являются дискретными, поэтому их иногда называют переменными состояния.
- Значение наблюдаемого вектора в момент времени n $\boldsymbol{x}_n$ зависит только от скрытого состояния $\boldsymbol{t}_n$, которое в свою очередь зависит только от скрытого состояния в предыдущий момент времени $\boldsymbol{t}_{n-1}$.

Спецификация вероятностной модели I

Пусть имеется K возможных состояний. Закодируем состояние в каждый момент времени n бинарным вектором $\boldsymbol{t}_n = (t_{n1}, \ldots, t_{nK})$, где

$$t_{nj} = \begin{cases} 1, & \text{если в момент } n \text{ модель находится в состоянии } j \\ 0, & \text{иначе} \end{cases}$$

Тогда распределение $p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1})$ можно задать матрицей перехода A размера $K \times K$, где $A_{ij} = p(t_{nj} = 1|t_{n-1,i} = 1)$, $\sum_j A_{ij} = 1$, т.е.

$$p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}) = \prod_{i=1}^{K}\prod_{j=1}^{K} A_{ij}^{t_{n-1,i}t_{nj}}$$

Пусть в первый момент времени $p(t_{1j} = 1) = \pi_j$. Тогда

$$p(\boldsymbol{t}_1) = \prod_{j=1}^{K} \pi_j^{t_{1j}}$$

Спецификация вероятностной модели II

Пусть для каждого состояния $k \in \{1, \ldots, K\}$ в момент времени n известна модель генерации наблюдаемых данных $\boldsymbol{x}_n$ $p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)$, задаваемая с помощью набора параметров $\boldsymbol{\phi}_k$. Тогда

$$p(\boldsymbol{x}_n|\boldsymbol{t}_n) = \prod_{j=1}^{K} (p(\boldsymbol{x}_n|\boldsymbol{\phi}_k))^{t_{nk}}$$

Обозначим полный набор параметров СММ через $\Theta = \{\boldsymbol{\pi}, A, \boldsymbol{\phi}\}$. Тогда правдоподобие в СММ вычисляется как

$$p(X, T|\Theta) = p(\boldsymbol{t}_1, \boldsymbol{\pi}) \prod_{n=2}^{N} p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}, A) \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{t}_n, \boldsymbol{\phi}) =$$

$$= \prod_{j=1}^{K} \pi_j^{t_{1j}} \left(\prod_{n=2}^{N} \prod_{i=1}^{K} \prod_{j=1}^{K} A_{ij}^{t_{n-1,i} t_{nj}} \right) \left(\prod_{n=1}^{N} \prod_{k=1}^{K} (p(\boldsymbol{x}_n|\boldsymbol{\phi}_k))^{t_{nk}} \right)$$

Задачи в СММ

- Распознавание (предыдущая лекция). Известна некоторая последовательность X и набор параметров Θ. Задача состоит в получении наиболее правдоподобной последовательности состояний T как $\arg\max_T p(T|X,\Theta)$ (алгоритм Витерби).
- Обучение с учителем (предыдущая лекция). Известна некоторая последовательность X, для которой заданы T. Задача состоит в оценке по обучающей выборке набора параметров Θ.
- Обучение без учителя. Известна некоторая последовательность X и число состояний K. Задача состоит в оценке параметров Θ (EM-алгоритм).
  - Нахождение маргинального распределения $p(\boldsymbol{t}_n|X,\Theta)$ компоненты $\boldsymbol{t}_n$ по заданым X и Θ
- Прогнозирование. Известна некоторая последовательность X. Задача состоит в оценке наблюдаемого вектора в следующий момент времени $N+1$ — $p(\boldsymbol{x}_{N+1}|X)$.

План лекции

План лекции

Метод максимального правдоподобия для СММ

Алгоритм «вперед-назад»

Устойчивые формулы для алгоритма «вперед-назад»

Модельный пример

Метод максимального правдоподобия

Для оценки параметров СММ Θ воспользуемся методом максимального правдоподобия:

$$\Theta_* = \arg\max_{\Theta} p(X|\Theta) = \arg\max_{\Theta} \sum_{T} p(X, T|\Theta)$$

Прямая максимизация правдоподобия затруднительна, т.к. оптимизируемая функция не является выпуклой и, кроме того, для вычисления функции требуется суммирование N^K слагаемых. Можно воспользоваться итерационным EM-алгоритмом.

EM-алгоритм в общем виде

Требуется найти максимум правдоподобия в вероятностной модели со скрытыми переменными:

$$p(X|\Theta) = \sum_T p(X,T|\Theta) \to \max_\Theta \Leftrightarrow \log\left(\sum_T p(X,T|\Theta)\right) \to \max_\Theta$$

- E-шаг. Фиксируется значение параметров Θ_{old}. Оценивается апостериорное распределение на скрытые переменные $p(T|X,\Theta_{old})$, и полное правдоподобие усредняется по полученному распределению:

$$\mathbb{E}_{T|X,\Theta_{old}} \log p(X,T|\Theta) = \sum_T \log p(X,T|\Theta) p(T|X,\Theta_{old})$$

- M-шаг. Фиксируется апостериорное распределение $p(T|X,\Theta_{old})$, и производится поиск новых значений параметров Θ_{new}:

$$\Theta_{new} = \arg\max_\Theta \mathbb{E}_{T|X,\Theta_{old}} \log p(X,T|\Theta)$$

- Шаги E и M повторяются до сходимости.

E-шаг

Обозначим

$$\gamma(t_{nj}) = \mathbb{E}_{T|X,\Theta} t_{nj} = p(t_{nj} = 1|X, \Theta_{old}),$$
$$\xi(t_{n-1,i}, t_{nj}) = \mathbb{E}_{T|X,\Theta}(t_{n-1,i} t_{nj}) = p(t_{n-1,i} = 1, t_{nj} = 1|X, \Theta_{old}).$$

Тогда

$$\log p(X, T|\Theta) = \sum_{j=1}^{K} t_{1j} \log \pi_j + \sum_{n=2}^{N} \sum_{i=1}^{K} \sum_{j=1}^{K} t_{n-1,i} t_{nj} \log A_{ij} + \sum_{n=1}^{N} \sum_{j=1}^{K} t_{nj} \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)$$

$$\mathbb{E}_{T|X,\Theta_{old}} \log p(X, T|\Theta) = \sum_{j=1}^{K} \gamma(t_{1j}) \log \pi_j + \sum_{n=2}^{N} \sum_{i=1}^{K} \sum_{j=1}^{K} \xi(t_{n-1,i}, t_{nj}) \log A_{ij} + \sum_{n=1}^{N} \sum_{j=1}^{K} \gamma(t_{nj}) \log p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)$$

M-шаг

$$\boldsymbol{\pi}: \quad \sum_{j=1}^{K} \gamma(t_{1j}) \log \pi_j + \lambda \left(\sum_{j=1}^{K} \pi_j - 1 \right) \to \underset{\boldsymbol{\pi}, \lambda}{\text{extr}}$$

$$\frac{\gamma(t_{1j})}{\pi_j} + \lambda = 0 \Rightarrow \pi_j = -\frac{\gamma(t_{1j})}{\lambda}$$

$$\sum_{j=1}^{K} \pi_j = 1 \Rightarrow \lambda = -\sum_{i=1}^{K} \gamma(t_{1i})$$

$$\pi_j^{new} = \frac{\gamma(t_{1j})}{\sum_{i=1}^{K} \gamma(t_{1i})}$$

Действуя аналогично для A, принимая во внимание, что $\sum_{j=1}^{K} A_{ij} = 1 \ \forall \ i$, получаем:

$$A_{ij}^{new} = \frac{\sum_{n=2}^{N} \xi(t_{n-1,i} t_{nj})}{\sum_{k=1}^{K} \sum_{n=2}^{N} \xi(t_{n-1,i} t_{nk})}$$

M-шаг для компонент ϕ

M-шаг для компонент генерации данных $p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)$ абсолютно аналогичен M-шагу для оценки параметров при восстановлении смесей распределений. В частности, если в качестве компонент выступают многомерные нормальные распределения

$$p(\boldsymbol{x}_n|\boldsymbol{\phi}_k) = \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \Sigma_k),$$

то задача оптимизации для параметров $\boldsymbol{\mu}_k, \Sigma_k$ может быть решена в явном виде:

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma(t_{nk})\boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma(t_{nk})}$$

$$\Sigma_k = \frac{\sum_{n=1}^{N} \gamma(t_{nk})(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T}{\sum_{n=1}^{N} \gamma(t_{nk})}$$

Инициализация параметров Θ

Для начала работы EM-алгоритма необходимо задать начальные значения параметров $\Theta = (\boldsymbol{\pi}, A, \boldsymbol{\phi})$. Заметим, что если какой-нибудь параметр инициализирован нулем, то в процессе итераций EM его значение не изменится.

- Значения параметров $\boldsymbol{\pi}$ и A обычно выбираются случайными при соблюдении ограничений $\sum_j \pi_j = 1$ и $\sum_j A_{ij} = 1 \ \forall i$.
- Инициализация $\boldsymbol{\phi}$ зависит от формы распределений $p(\boldsymbol{x}|\boldsymbol{\phi})$. В случае нормальных распределений можно провести кластеризацию данных на K кластеров и выбрать в качестве $\boldsymbol{\mu}_k$ и Σ_k центр и разброс соответствующего кластера.

План лекции

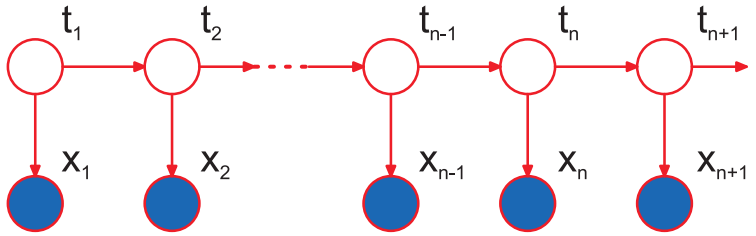
Вычисление апостериорных распределений на скрытые компоненты

На E-шаге алгоритма обучения СММ требуется вычисление апостериорных распределений на скрытые компоненты

$$\gamma(t_{nj}) = p(t_{nj} = 1|X, \Theta), \quad \xi(t_{n-1,i}, t_{nj}) = p(t_{n-1,i} = 1, t_{nj} = 1|X, \Theta)$$

Алгоритм «вперед-назад» (Баума-Уэлша) позволяет эффективно вычислять эти величины для всех n, i, j за линейное по N время.
В дальнейшем для удобства будем опускать Θ во всех формулах, считая набор параметров фиксированным.

Свойства условной независимости для СММ



$$
\begin{aligned}
p(X|\boldsymbol{t}_n) &= p(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n|\boldsymbol{t}_n)p(\boldsymbol{x}_{n+1},\ldots,\boldsymbol{x}_N|\boldsymbol{t}_n)\\
p(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{n-1}|\boldsymbol{x}_n,\boldsymbol{t}_n) &= p(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{n-1}|\boldsymbol{t}_n)\\
p(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{n-1}|\boldsymbol{t}_{n-1},\boldsymbol{t}_n) &= p(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{n-1}|\boldsymbol{t}_{n-1})\\
p(\boldsymbol{x}_{n+1},\ldots,\boldsymbol{x}_N|\boldsymbol{t}_n,\boldsymbol{t}_{n+1}) &= p(\boldsymbol{x}_{n+1},\ldots,\boldsymbol{x}_N|\boldsymbol{t}_{n+1})\\
p(\boldsymbol{x}_{n+2},\ldots,\boldsymbol{x}_N|\boldsymbol{t}_{n+1},\boldsymbol{x}_{n+1}) &= p(\boldsymbol{x}_{n+2},\ldots,\boldsymbol{x}_N|\boldsymbol{t}_{n+1})\\
p(X|\boldsymbol{t}_{n-1},\boldsymbol{t}_n) &= p(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{n-1}|\boldsymbol{t}_{n-1})p(\boldsymbol{x}_n|\boldsymbol{t}_n)\times\\
&\quad\times p(\boldsymbol{x}_{n+1},\ldots,\boldsymbol{x}_N|\boldsymbol{t}_n)\\
p(\boldsymbol{x}_{N+1}|X,\boldsymbol{t}_{N+1}) &= p(\boldsymbol{x}_{N+1}|\boldsymbol{t}_{N+1})\\
p(\boldsymbol{t}_{N+1}|\boldsymbol{t}_N,X) &= p(\boldsymbol{t}_{N+1}|\boldsymbol{t}_N)
\end{aligned}
$$

Вычисление $\gamma(\boldsymbol{t}_{nj})$

По формуле Байеса

$$\gamma(\boldsymbol{t}_n) = p(\boldsymbol{t}_n|X) = \frac{p(X|\boldsymbol{t}_n)p(\boldsymbol{t}_n)}{p(X)}$$

Пользуясь свойствами условной независимости для СММ, получаем

$$\gamma(\boldsymbol{t}_n) = \frac{p(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n,\boldsymbol{t}_n)p(\boldsymbol{x}_{n+1},\ldots,\boldsymbol{x}_N|\boldsymbol{t}_n)}{p(X)} = \frac{\boldsymbol{\alpha}(\boldsymbol{t}_n)\boldsymbol{\beta}(\boldsymbol{t}_n)}{p(X)}$$

Здесь

$$\boldsymbol{\alpha}(\boldsymbol{t}_n) = p(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n,\boldsymbol{t}_n)$$
$$\boldsymbol{\beta}(\boldsymbol{t}_n) = p(\boldsymbol{x}_{n+1},\ldots,\boldsymbol{x}_N|\boldsymbol{t}_n)$$

Алгоритм «вперед-назад» позволяет быстро рекуррентно вычислять значение $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ через $\boldsymbol{\alpha}(\boldsymbol{t}_{n-1})$ (проход вперед) и $\boldsymbol{\beta}(\boldsymbol{t}_n)$ через $\boldsymbol{\beta}(\boldsymbol{t}_{n+1})$ (проход назад).

Рекуррентная формула для $\alpha(t_n)$

$$
\begin{aligned}
\alpha(t_n) =& p(x_1, \ldots, x_n, t_n) = \\
=& p(x_1, \ldots, x_n | t_n) p(t_n) = \\
=& p(x_n | t_n) p(x_1, \ldots, x_{n-1} | t_n) p(t_n) = \\
=& p(x_n | t_n) p(x_1, \ldots, x_{n-1}, t_n) = \\
=& p(x_n | t_n) \sum_{t_{n-1}} p(x_1, \ldots, x_{n-1}, t_{n-1}, t_n) = \\
=& p(x_n | t_n) \sum_{t_{n-1}} p(x_1, \ldots, x_{n-1}, t_n | t_{n-1}) p(t_{n-1}) = \\
=& p(x_n | t_n) \sum_{t_{n-1}} p(x_1, \ldots, x_{n-1} | t_{n-1}) p(t_n | t_{n-1}) p(t_{n-1}) = \\
=& p(x_n | t_n) \sum_{t_{n-1}} p(x_1, \ldots, x_{n-1}, t_{n-1}) p(t_n | t_{n-1}) = \\
=& p(x_n | t_n) \sum_{t_{n-1}} \alpha(t_{n-1}) p(t_n | t_{n-1})
\end{aligned}
$$

Вычисление $\boldsymbol{\alpha}(\boldsymbol{t}_n)$

Для запуска рекуррентного процесса необходимо вычислить

$$\boldsymbol{\alpha}(\boldsymbol{t}_1) = p(\boldsymbol{x}_1, \boldsymbol{t}_1) = p(\boldsymbol{t}_1)p(\boldsymbol{x}_1|\boldsymbol{t}_1) = \prod_{j=1}^{K}(\pi_j p(\boldsymbol{x}_1|\boldsymbol{\phi}_j))^{t_{1j}}$$

На каждом шаге рекурсии вектор $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ длины K умножается на матрицу $p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1})$ размера $K \times K$. Поэтому сложность всего рекуррентного процесса составляет $O(NK^2)$.

Рекуррентная формула для $\boldsymbol{\beta}(\boldsymbol{t}_n)$

$$\begin{aligned}
\boldsymbol{\beta}(\boldsymbol{t}_n) =& p(\boldsymbol{x}_{n+1}, \ldots, \boldsymbol{x}_N | \boldsymbol{t}_n) = \\
=& \sum_{\boldsymbol{t}_{n+1}} p(\boldsymbol{x}_{n+1}, \ldots, \boldsymbol{x}_N, \boldsymbol{t}_{n+1} | \boldsymbol{t}_n) = \\
=& \sum_{\boldsymbol{t}_{n+1}} p(\boldsymbol{x}_{n+1}, \ldots, \boldsymbol{x}_N | \boldsymbol{t}_{n+1}) p(\boldsymbol{t}_{n+1} | \boldsymbol{t}_n) = \\
=& \sum_{\boldsymbol{t}_{n+1}} p(\boldsymbol{x}_{n+2}, \ldots, \boldsymbol{x}_N | \boldsymbol{t}_{n+1}) p(\boldsymbol{x}_{n+1} | \boldsymbol{t}_{n+1}) p(\boldsymbol{t}_{n+1} | \boldsymbol{t}_n) = \\
=& \sum_{\boldsymbol{t}_{n+1}} \boldsymbol{\beta}(\boldsymbol{t}_{n+1}) p(\boldsymbol{x}_{n+1} | \boldsymbol{t}_{n+1}) p(\boldsymbol{t}_{n+1} | \boldsymbol{t}_n)
\end{aligned}$$

Инициализация рекуррентного процесса для $\boldsymbol{\beta}(\boldsymbol{t}_n)$

Вспомним формулу, связывающую значение $\boldsymbol{\gamma}(\boldsymbol{t}_n)$ с $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ и $\boldsymbol{\beta}(\boldsymbol{t}_n)$:

$$\boldsymbol{\gamma}(\boldsymbol{t}_n) = \frac{p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{t}_n) p(\boldsymbol{x}_{n+1}, \ldots, \boldsymbol{x}_N | \boldsymbol{t}_n)}{p(X)} = \frac{\boldsymbol{\alpha}(\boldsymbol{t}_n)\boldsymbol{\beta}(\boldsymbol{t}_n)}{p(X)}$$

Подставляя в формулу значение $n = N$, получаем

$$\boldsymbol{\gamma}(\boldsymbol{t}_N) = p(\boldsymbol{t}_N | X) = \frac{p(X, \boldsymbol{t}_N)\boldsymbol{\beta}(\boldsymbol{t}_N)}{p(X)}$$

Таким образом, $\boldsymbol{\beta}(\boldsymbol{t}_N) = 1$.

Нормировочная константа $p(X)$

Значение $\boldsymbol{\gamma}(\boldsymbol{t}_n)$ определено с точностью до нормировочной константы $p(X)$. Однако, на M-шаге значение $\boldsymbol{\gamma}(\boldsymbol{t}_n)$, как правило, входит в числитель и в знаменатель. Таким образом, нормировочная константа сокращается. Например, при вычислении центра нормального распределения:

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma(t_{nk})\boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma(t_{nk})} = \frac{\sum_{n=1}^{N} \alpha(t_{nk})\beta(t_{nk})\boldsymbol{x}_k}{\sum_{n=1}^{N} \alpha(t_{nk})\beta(t_{nk})}$$

Тем не менее, сама нормировочная константа $p(X)$ — это значение правдоподобия, которое может представлять отдельный интерес (например, можно отслеживать возрастание правдоподобия при итерациях EM-алгоритма). Зная $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ и $\boldsymbol{\beta}(\boldsymbol{t}_n)$, правдоподобие может быть легко вычислено как

$$p(X) = \sum_{\boldsymbol{t}_n} \boldsymbol{\alpha}(\boldsymbol{t}_n)\boldsymbol{\beta}(\boldsymbol{t}_n), \;\; \forall n \;\; \Rightarrow \;\; p(X) = \sum_{\boldsymbol{t}_N} \boldsymbol{\alpha}(\boldsymbol{t}_N)$$

Формула для $\xi(\boldsymbol{t}_{n-1}, \boldsymbol{t}_n)$

$$\xi(\boldsymbol{t}_{n-1}, \boldsymbol{t}_n) = p(\boldsymbol{t}_{n-1}, \boldsymbol{t}_n | X) =$$

$$= \frac{p(X|\boldsymbol{t}_{n-1}, \boldsymbol{t}_n) p(\boldsymbol{t}_{n-1}, \boldsymbol{t}_n)}{p(X)} =$$

$$= \frac{p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-1}|\boldsymbol{t}_{n-1}) p(\boldsymbol{x}_n|\boldsymbol{t}_n) p(\boldsymbol{x}_{n+1}, \ldots, \boldsymbol{x}_N|\boldsymbol{t}_n) p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}) p(\boldsymbol{t}_{n-1})}{p(X)} =$$

$$= \frac{\boldsymbol{\alpha}(\boldsymbol{t}_{n-1}) p(\boldsymbol{x}_n|\boldsymbol{t}_n) p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}) \boldsymbol{\beta}(\boldsymbol{t}_n)}{p(X)}$$

Итоговый EM-алгоритм для случая, когда $p(\boldsymbol{x}_n|\boldsymbol{t}_n)$ — нормальные распределения

- Начальная инициализация параметров $\Theta = (\boldsymbol{\pi}, A, \boldsymbol{\phi})$. Параметры $\boldsymbol{\pi}$ и A можно инициализировать случайно, а $\boldsymbol{\phi}$ с помощью кластеризации данных.
- E-шаг. Вычисление $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ и $\boldsymbol{\beta}(\boldsymbol{t}_n)$ с помощью рекуррентного алгоритма «вперед-назад». Вычисление величин $\gamma(t_{nj})$ и $\xi(t_{n-1,i}, t_{nj})$ и, возможно, правдоподобия $p(X)$.
- M-шаг. Вычисление новых значений параметров:

$$\pi_j^{new} = \frac{\gamma(t_{1j})}{\sum_{i=1}^{K}\gamma(t_{1i})}, \qquad A_{ij}^{new} = \frac{\sum_{n=2}^{N}\xi(t_{n-1,i}t_{nj})}{\sum_{k=1}^{K}\sum_{n=2}^{N}\xi(t_{n-1,i}t_{nk})}$$

$$\boldsymbol{\mu}_k^{new} = \frac{\sum_{n=1}^{N}\gamma(t_{nk})\boldsymbol{x}_n}{\sum_{n=1}^{N}\gamma(t_{nk})}, \quad \Sigma_k^{new} = \frac{\sum_{n=1}^{N}\gamma(t_{nk})(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T}{\sum_{n=1}^{N}\gamma(t_{nk})}$$

- Повторять шаги E и M до сходимости (пока значение $p(X)$ или Θ не стабилизируется).

Задача прогнозирования

$$p(\boldsymbol{x}_{N+1}|X) = \sum_{\boldsymbol{t}_{N+1}} p(\boldsymbol{x}_{N+1}, \boldsymbol{t}_{N+1}|X) = \sum_{\boldsymbol{t}_{N+1}} p(\boldsymbol{x}_{N+1}|\boldsymbol{t}_{N+1}) p(\boldsymbol{t}_{N+1}|X) =$$

$$= \sum_{\boldsymbol{t}_{N+1}} p(\boldsymbol{x}_{N+1}|\boldsymbol{t}_{N+1}) \left(\sum_{\boldsymbol{t}_N} p(\boldsymbol{t}_{N+1}, \boldsymbol{t}_N|X) \right) =$$

$$= \sum_{\boldsymbol{t}_{N+1}} p(\boldsymbol{x}_{N+1}|\boldsymbol{t}_{N+1}) \left(\sum_{\boldsymbol{t}_N} p(\boldsymbol{t}_N|X) p(\boldsymbol{t}_{N+1}|\boldsymbol{t}_N) \right) =$$

$$= \sum_{\boldsymbol{t}_{N+1}} p(\boldsymbol{x}_{N+1}|\boldsymbol{t}_{N+1}) \left(\sum_{\boldsymbol{t}_N} p(\boldsymbol{t}_{N+1}|\boldsymbol{t}_N) \frac{p(\boldsymbol{t}_N, X)}{p(X)} \right) =$$

$$= \frac{1}{p(X)} \sum_{\boldsymbol{t}_{N+1}} p(\boldsymbol{x}_{N+1}|\boldsymbol{t}_{N+1}) \left(\sum_{\boldsymbol{t}_N} p(\boldsymbol{t}_{N+1}|\boldsymbol{t}_N) \boldsymbol{\alpha}(\boldsymbol{t}_N) \right)$$

Это фактически смесь распределений с компонентами $p(\boldsymbol{x}_{N+1}|\boldsymbol{t}_{N+1})$ и весами $w_k = \frac{1}{p(X)} \sum_{\boldsymbol{t}_N} p(\boldsymbol{t}_{N+1}|\boldsymbol{t}_N)\boldsymbol{\alpha}(\boldsymbol{t}_N)$. Для получения точечного прогноза $\boldsymbol{x}^*_{N+1}$ можно воспользоваться MCMC.

План лекции

Метод максимального правдоподобия для СММ

Алгоритм «вперед-назад»

Устойчивые формулы для алгоритма «вперед-назад»

Модельный пример

Необходимость устойчивых вычислений для $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ и $\boldsymbol{\beta}(\boldsymbol{t}_n)$

Формулы пересчета для $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ и $\boldsymbol{\beta}(\boldsymbol{t}_n)$:

$$\boldsymbol{\alpha}(\boldsymbol{t}_n) = p(\boldsymbol{x}_n|\boldsymbol{t}_n) \sum_{\boldsymbol{t}_{n-1}} \boldsymbol{\alpha}(\boldsymbol{t}_{n-1}) p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1})$$

$$\boldsymbol{\beta}(\boldsymbol{t}_n) = \sum_{\boldsymbol{t}_{n+1}} \boldsymbol{\beta}(\boldsymbol{t}_{n+1}) p(\boldsymbol{t}_{n+1}|\boldsymbol{t}_n) p(\boldsymbol{x}_{n+1}|\boldsymbol{t}_{n+1})$$

На практике значения вероятностей $p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1})$ и $p(\boldsymbol{x}_n|\boldsymbol{t}_n)$ могут быть существенно меньше единицы. В процессе пересчета эти вероятности умножаются друг на друга, и получающиеся значения перестают укладываться в машинную точность.

Устойчивые формулы для $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ I

Предлагается вместо $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ рассмотреть следующую величину:

$$\hat{\boldsymbol{\alpha}}(\boldsymbol{t}_n) = p(\boldsymbol{t}_n|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \frac{\boldsymbol{\alpha}(\boldsymbol{t}_n)}{p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)}$$

Можно надеяться, что значения $\hat{\boldsymbol{\alpha}}(\boldsymbol{t}_n)$ будут существенно отличны от нуля, т.к. $\sum_{\boldsymbol{t}_n} \hat{\boldsymbol{\alpha}}(\boldsymbol{t}_n) = 1$.
Рассмотрим также величины

$$c_n = p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}, \ldots, \boldsymbol{x}_1).$$

Вычисление этих величин также будет устойчивым, т.к. они имеют смысл одномерных вероятностных распределений.

Устойчивые формулы для $\boldsymbol{\alpha}(\boldsymbol{t}_n)$ II

Очевидно, что

$$p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \prod_{i=1}^{n} c_i$$

$$\boldsymbol{\alpha}(\boldsymbol{t}_n) = p(\boldsymbol{t}_n|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \hat{\boldsymbol{\alpha}}(\boldsymbol{t}_n)\left(\prod_{i=1}^{n} c_i\right)$$

Подставляя это выражение в формулу пересчета для $\boldsymbol{\alpha}(\boldsymbol{t}_n)$

$$\boldsymbol{\alpha}(\boldsymbol{t}_n) = p(\boldsymbol{x}_n|\boldsymbol{t}_n)\sum_{\boldsymbol{t}_{n-1}} \boldsymbol{\alpha}(\boldsymbol{t}_{n-1})p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}),$$

получаем

$$c_n\hat{\boldsymbol{\alpha}}(\boldsymbol{t}_n) = p(\boldsymbol{x}_n|\boldsymbol{t}_n)\sum_{\boldsymbol{t}_{n-1}} \hat{\boldsymbol{\alpha}}(\boldsymbol{t}_{n-1})p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1})$$

Значение c_n определяется из условия нормировки для $\hat{\boldsymbol{\alpha}}(\boldsymbol{t}_n)$.

Устойчивые формулы для $\boldsymbol{\beta}(\boldsymbol{t}_n)$

Рассмотрим следующую величину

$$\hat{\boldsymbol{\beta}}(\boldsymbol{t}_n) = \frac{p(\boldsymbol{x}_{n+1}, \ldots, \boldsymbol{x}_N | \boldsymbol{t}_n)}{p(\boldsymbol{x}_{n+1}, \ldots, \boldsymbol{x}_N | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)} = \frac{\boldsymbol{\beta}(\boldsymbol{t}_n)}{\prod_{i=n+1}^{N} c_i}$$

Вычисление данной величины будет устойчивым, т.к. $\hat{\boldsymbol{\beta}}(\boldsymbol{t}_n)$ является отношением двух распределений на $(\boldsymbol{x}_{n+1}, \ldots, \boldsymbol{x}_N)$. Подставляя выражение для $\hat{\boldsymbol{\beta}}(\boldsymbol{t}_n)$ в формулу пересчета

$$\boldsymbol{\beta}(\boldsymbol{t}_n) = \sum_{\boldsymbol{t}_{n+1}} \boldsymbol{\beta}(\boldsymbol{t}_{n+1}) p(\boldsymbol{t}_{n+1} | \boldsymbol{t}_n) p(\boldsymbol{x}_{n+1} | \boldsymbol{t}_{n+1}),$$

получаем

$$c_{n+1} \hat{\boldsymbol{\beta}}(\boldsymbol{t}_n) = \sum_{\boldsymbol{t}_{n+1}} \hat{\boldsymbol{\beta}}(\boldsymbol{t}_{n+1}) p(\boldsymbol{t}_{n+1} | \boldsymbol{t}_n) p(\boldsymbol{x}_{n+1} | \boldsymbol{t}_{n+1})$$

Значения c_n определяются из формул пересчета для $\hat{\boldsymbol{\alpha}}(\boldsymbol{t}_n)$.

Окончательные выражения для E-шага

Правдоподобие $p(X)$ вычисляется как

$$p(X) = \prod_{n=1}^{N} c_n$$

Другие необходимые величины вычисляются как

$$\gamma(\boldsymbol{t}_n) = \hat{\boldsymbol{\alpha}}(\boldsymbol{t}_n)\hat{\boldsymbol{\beta}}(\boldsymbol{t}_n)$$

$$\xi(\boldsymbol{t}_{n-1}, \boldsymbol{t}_n) = \frac{1}{c_n}\hat{\boldsymbol{\alpha}}(\boldsymbol{t}_{n-1})p(\boldsymbol{x}_n|\boldsymbol{t}_n)p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1})\hat{\boldsymbol{\beta}}(\boldsymbol{t}_n)$$

План лекции

Метод максимального правдоподобия для СММ

Алгоритм «вперед-назад»

Устойчивые формулы для алгоритма «вперед-назад»

Модельный пример

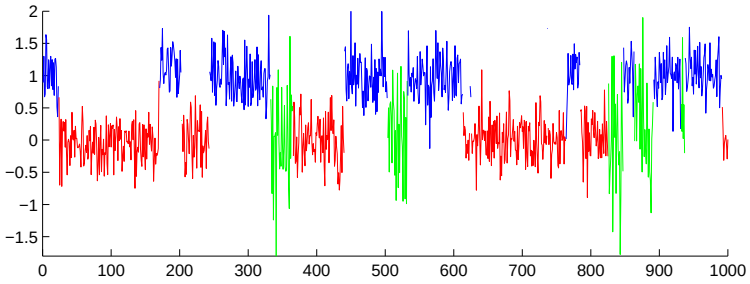
Модельный пример I

В качестве иллюстрации работы EM-алгоритма обучения СММ рассмотрим простой модельный пример. $X \in \mathbb{R}$, $K = 3$, данные сгенерированы из нормальных распределений со следующими параметрами:

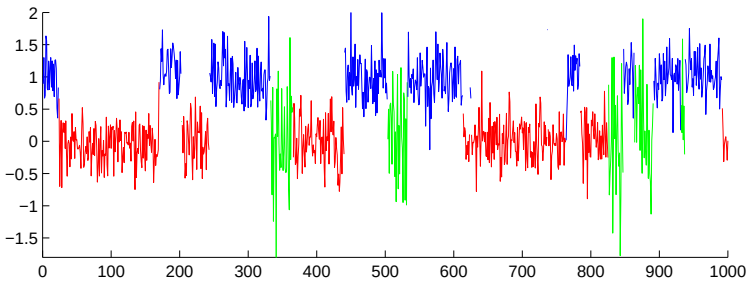
$$\mu_1 = 0, \quad \mu_2 = 0, \quad \mu_3 = 1$$
$$\sigma_1^2 = 0.1, \quad \sigma_2^2 = 0.5, \quad \sigma_3^2 = 0.1$$

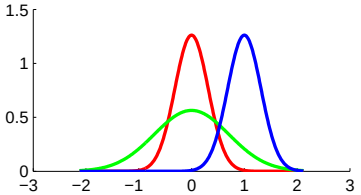
Априорные вероятности и матрица перехода выбраны следующим образом:

$$\boldsymbol{\pi} = [0.3,\ 0.2,\ 0.5], \qquad A = \begin{bmatrix} 0.98 & 0.01 & 0.01 \\ 0.01 & 0.97 & 0.02 \\ 0.01 & 0.01 & 0.98 \end{bmatrix}$$

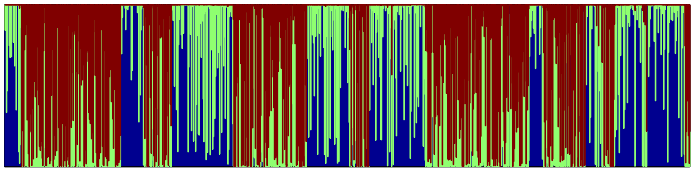


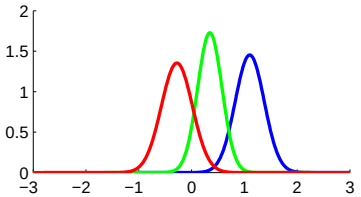
Модельный пример II



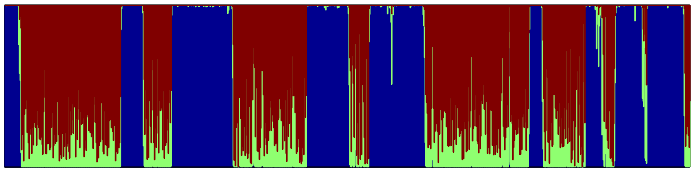


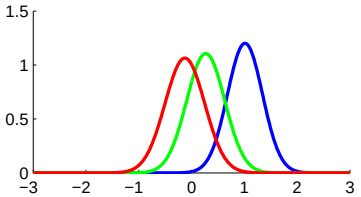
Ит. 1:



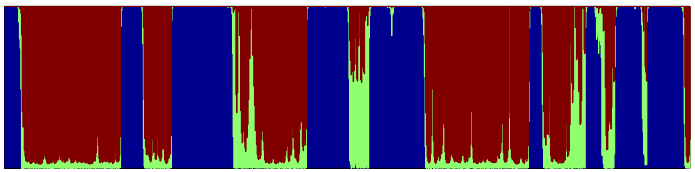


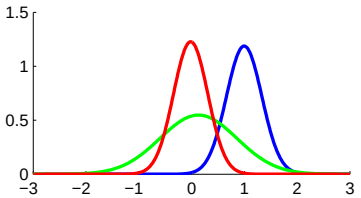
Ит. 5:



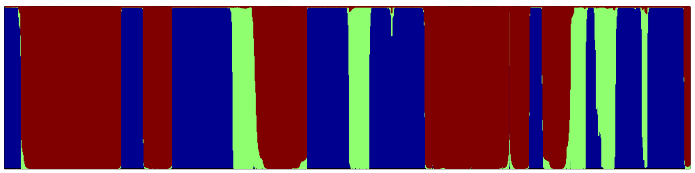


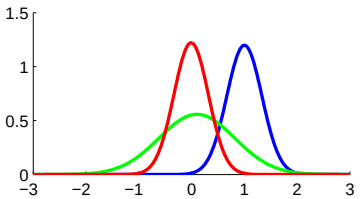
Ит. 20:





Ит. 54:





Модельный пример III

После 54-ой итерации EM-алгоритма значения параметров были следующие:

$$\boldsymbol{\pi} = [10^{-190},\ 10^{-125},\ 1], \qquad A = \begin{bmatrix} 0.984 & 0.004 & 0.012 \\ 0.013 & 0.949 & 0.038 \\ 0.011 & 0.011 & 0.978 \end{bmatrix}$$

$$\mu_1 = -0.01,\ \sigma_1^2 = 0.11$$
$$\mu_2 = 0.1,\ \sigma_2^2 = 0.51$$
$$\mu_3 = 1,\ \sigma_3^2 = 0.11$$

Лекция 6. Снижение размерности в данных. Метод главных компонент.

Курс «Математические основы теории прогнозирования»

План лекции

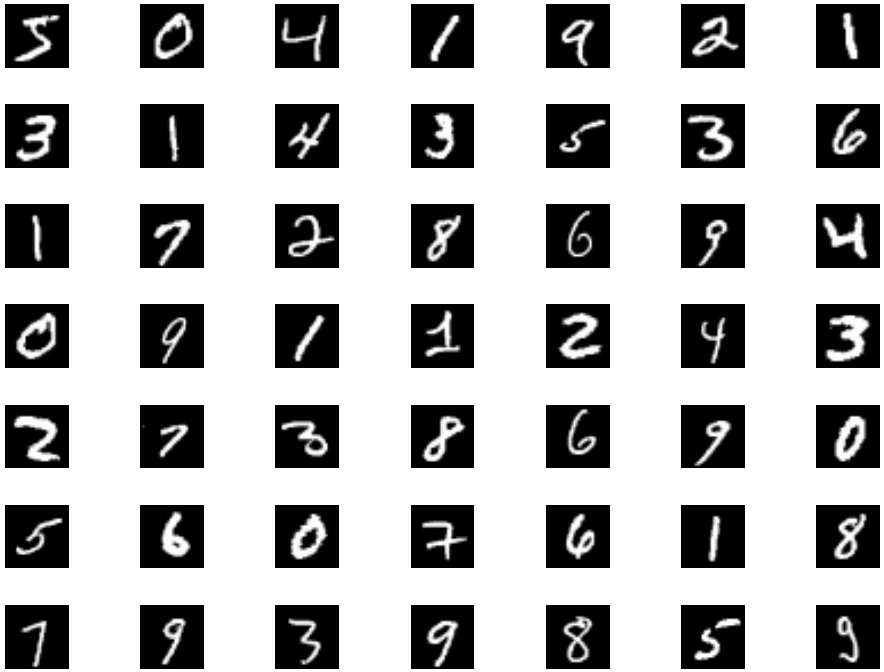
Метод главных компонент (PCA)

Вероятностный метод главных компонент

- Вероятностная формулировка метода
- EM-алгоритм для PCA
- Выбор числа главных компонент с помощью байесовского п

Другие модели

Практический пример: распознавание рукописных цифр



Задача: выбор адекватного признакового описания изображения $\boldsymbol{x} = (x_1, \ldots, x_D)$ для дальнейшего использования алгоритмов распознавания.

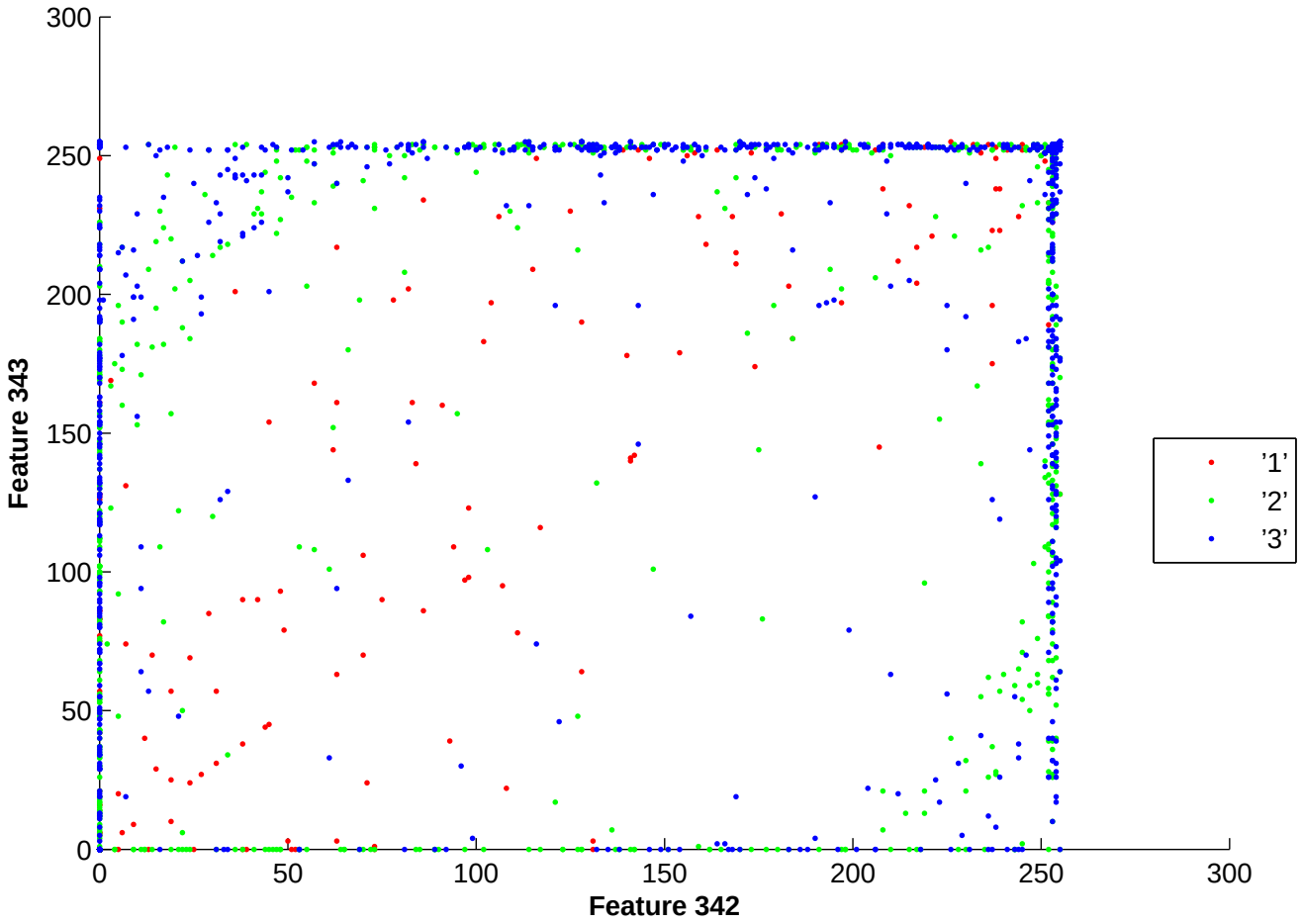
Выбор признакового пространства

Прямой способ: вытянуть матрицу интенсивностей в вектор.
Такой способ описания данных не является адекватным в силу ряда причин:

- Слишком большое число признаков (для картинки 28×28 получается 784 признака). Как следствие, большое время обработки, проблемы переобучения и т.д.
- Близкие в признаковом пространстве объекты не соответствуют одному классу. Гипотеза компактности является одним из основных предположений большинства методов распознавания.

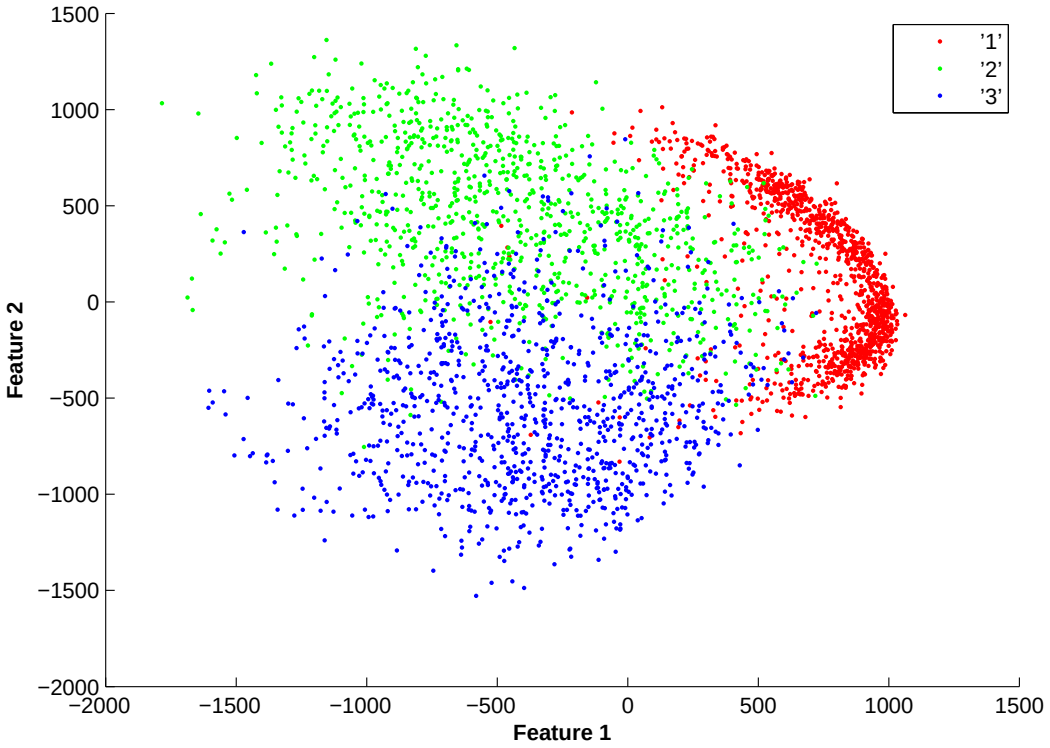
Иллюстрация

Близкие в признаковом пространстве объекты не соответствуют одному классу.



Преобразование признакового пространства

После преобразования признакового описания (с помощью метода главных компонент) количество признаков уменьшается (с 784 до нескольких десятков), причем объекты одного класса образуют компактные области в признаковом пространстве.



Сокращение размерности в данных

Пусть имеется некоторая выборка $X = \{\boldsymbol{x}_n\}_{n=1}^N$, $\boldsymbol{x}_n \in \mathbb{R}^D$.
Цель — представить выборку в пространстве меньшей размерности $d < D$, причем в новом пространстве «схожие» объекты должны образовывать компактные области.
Причины сокращения размерности:

- уменьшение вычислительных затрат при обработке данных
- борьба с переобучением
- сжатие данных для более эффективного хранения информации
- визуализация данных
- извлечение признаков
- интерпретация данных
- ...

План лекции

Метод главных компонент (PCA)

Вероятностный метод главных компонент

- Вероятностная формулировка метода
- EM-алгоритм для PCA
- Выбор числа главных компонент с помощью байесовского п

Другие модели

Идея метода

Идея метода главных компонент (разложение Карунена-Лоева, Principal Component Analysis, PCA) — проекция данных на гиперплоскость с наименьшей ошибкой проектирования. Эквивалентная формулировка: поиск проекции на гиперплоскость с сохранением большей части дисперсии в данных.

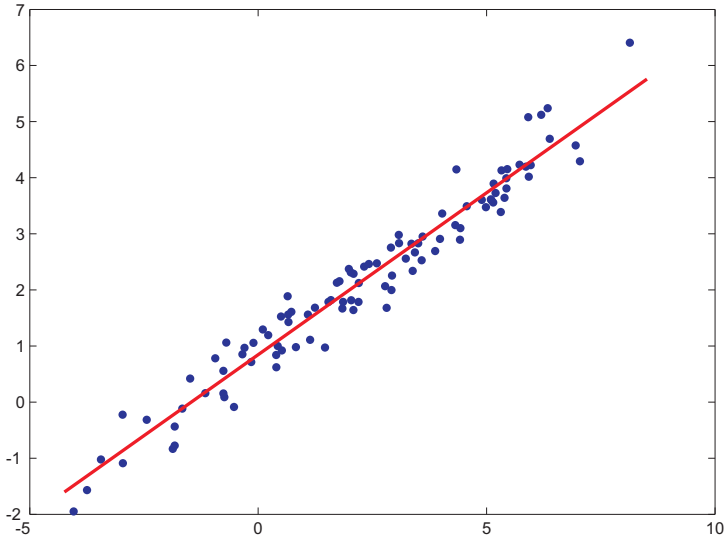
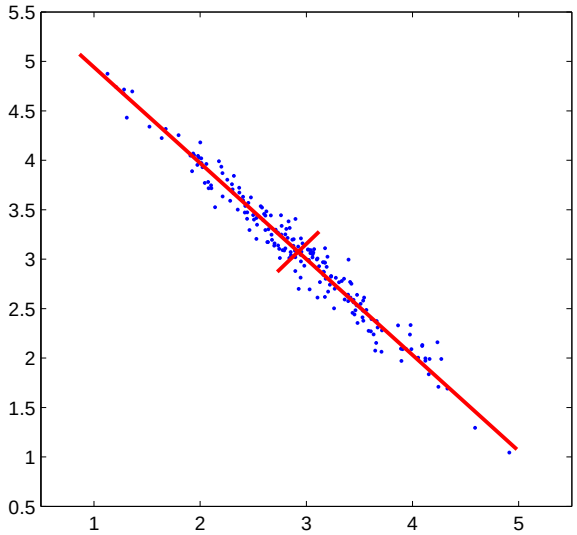


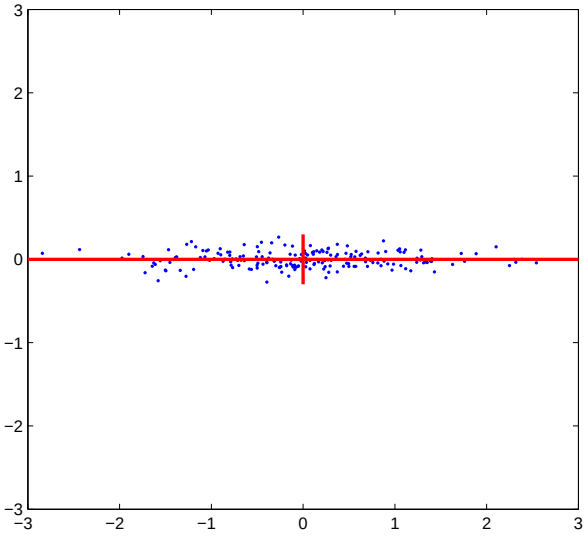
Иллюстрация решения PCA

Выборка в исходном пространстве



$\longrightarrow$

Выборка в пространстве декоррелированных признаков



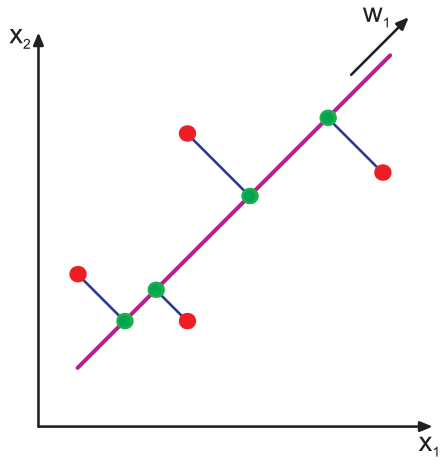
- Переносим начало координат в центр выборки
- Поворачиваем оси координат так, чтобы признаки стали декоррелированными
- Отбрасываем направления с низкой дисперсией

Линейное преобразование

Пусть D — размерность исходного пространства, d — размерность искомого пространства. Будем искать преобразование данных в семействе линейных функций:

$$\boldsymbol{x} = \boldsymbol{\mu} + t_1\boldsymbol{w}_1 + \cdots + t_d\boldsymbol{w}_d = W\boldsymbol{t} + \boldsymbol{\mu}$$

Здесь $\boldsymbol{x} \in \mathbb{R}^D$, $\boldsymbol{t} \in \mathbb{R}^d$ — новые координаты объекта, $W = (\boldsymbol{w}_1|\ldots|\boldsymbol{w}_d) \in \mathbb{R}^{D\times d}$, $\boldsymbol{\mu} \in \mathbb{R}^D$.
Графическая интерпретация — проекция данных на гиперплоскость, $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_d$ — базис гиперплоскости.



Критерий поиска гиперплоскости

Критерий выбора гиперплоскости — минимальная ошибка проектирования в смысле суммы квадратов отклонений исходных точек и их проекций.

Пусть $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_D$ — ортонормированный базис в пространстве $\mathbb{R}^D$, первые d компонент которого — базис искомой гиперлоскости.

Тогда

$$\boldsymbol{x}_n = \sum_{i=1}^{D} (\boldsymbol{x}_n^T \boldsymbol{w}_i) \boldsymbol{w}_i \text{ — исходные точки}$$

$$\hat{\boldsymbol{x}}_n = \sum_{i=1}^{d} t_{ni} \boldsymbol{w}_i + \sum_{i=d+1}^{D} \mu_i \boldsymbol{w}_i \text{ — приближение}$$

Критерий

$$J = \frac{1}{N} \sum_{n=1}^{N} \|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2 \to \min_{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_d, \boldsymbol{t}_1, \ldots, \boldsymbol{t}_N, \boldsymbol{\mu}}$$

Решение задачи

$$J = \frac{1}{N}\sum_{n=1}^{N}\|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2 \to \min_{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_d,\boldsymbol{t}_1,\ldots,\boldsymbol{t}_N,\boldsymbol{\mu}}$$

Можно показать, что решением задачи будет следующее:

$\bar{\boldsymbol{x}} = \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}_n$ — выборочное среднее

$S = \frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T$ — выборочная матрица ковариации

$\boldsymbol{w}_1, \ldots, \boldsymbol{w}_d$ — ортонормированный базис из собственных векторов матрицы S, отвечающих d наибольшим собственным значениям $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$.

$$t_{ni} = \boldsymbol{x}_n^T\boldsymbol{w}_i$$

$$\mu_i = \bar{\boldsymbol{x}}^T\boldsymbol{w}_i$$

Преобразование критерия I

$$J = \frac{1}{N}\sum_{n=1}^{N}\|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{i=1}^{D}(\boldsymbol{x}_n^T\boldsymbol{w}_i)\boldsymbol{w}_i - \sum_{i=1}^{d} t_{ni}\boldsymbol{w}_i - \right.$$

$$\left. - \sum_{i=d+1}^{D}\mu_i\boldsymbol{w}_i\right)^T\left(\sum_{i=1}^{D}(\boldsymbol{x}_n^T\boldsymbol{w}_i)\boldsymbol{w}_i - \sum_{i=1}^{d} t_{ni}\boldsymbol{w}_i - \sum_{i=d+1}^{D}\mu_i\boldsymbol{w}_i\right) =$$

$$\frac{1}{N}\sum_{n=1}^{N}\left(\sum_{i=1}^{D}(\boldsymbol{x}_n^T\boldsymbol{w}_i)^2 + \sum_{i=1}^{d} t_{ni}^2 + \sum_{i=d+1}^{D}\mu_i^2 - \right.$$

$$\left. 2\sum_{i=1}^{d}(\boldsymbol{x}_n^T\boldsymbol{w}_i)t_{ni} - 2\sum_{i=d+1}^{D}\mu_i(\boldsymbol{x}_n^T\boldsymbol{w}_i)\right)$$

$$\frac{\partial}{\partial t_{ni}} : -2(\boldsymbol{x}_n^T\boldsymbol{w}_i) + 2t_{ni} = 0 \Rightarrow t_{ni} = \boldsymbol{x}_n^T\boldsymbol{w}_i$$

$$\frac{\partial}{\partial \mu_i} : \sum\nolimits_{n=1}^{N}(-2(\boldsymbol{x}_n^T\boldsymbol{w}_i) + 2\mu_i) = 0 \Rightarrow \mu_i = \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}_n^T\boldsymbol{w}_i = \bar{\boldsymbol{x}}^T\boldsymbol{w}_i$$

Преобразование критерия II

$$t_{ni} = \boldsymbol{x}_n^T \boldsymbol{w}_i, \quad \mu_i = \bar{\boldsymbol{x}}^T \boldsymbol{w}_i$$

$$\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n = \sum_{i=d+1}^{D} \left\{ (\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T \boldsymbol{w}_i \right\} \boldsymbol{w}_i$$

$$J = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=d+1}^{D} (\boldsymbol{x}_n^T \boldsymbol{w}_i - \bar{\boldsymbol{x}}^T \boldsymbol{w}_i)^2 = \sum_{i=d+1}^{D} \boldsymbol{w}_i^T S \boldsymbol{w}_i \to \min_{\boldsymbol{w}_{d+1}, \ldots, \boldsymbol{w}_D}$$

Здесь $S = \frac{1}{N} \sum_{n=1}^{N} (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T$ — выборочная матрица ковариации данных.

Решение I

$$J = \sum_{i=d+1}^{D} \boldsymbol{w}_i^T S \boldsymbol{w}_i \to \min_{\boldsymbol{w}_{d+1},\ldots,\boldsymbol{w}_D}$$

$$\boldsymbol{w}_i^T \boldsymbol{w}_j = \delta_{ij}$$

Можно показать, что оптимум достигается, когда в качестве $\boldsymbol{w}_{d+1}, \ldots, \boldsymbol{w}_D$ выбираются собственные вектора матрицы ковариации S, отвечающие наименьшим собственным значениям $\lambda_{d+1} \geq \cdots \geq \lambda_D$. Тогда

$$J = \sum_{i=d+1}^{D} \lambda_i$$

Решение II

Рассмотрим оптимизацию по $\boldsymbol{w}_i$:

$$\tilde{J} = \boldsymbol{w}_i^T S \boldsymbol{w}_i \to \min_{\boldsymbol{w}_i}$$

$$\boldsymbol{w}_i^T \boldsymbol{w}_i = 1$$

Функция Лагранжа:

$$L = \boldsymbol{w}_i^T S \boldsymbol{w}_i + \lambda_i (1 - \boldsymbol{w}_i^T \boldsymbol{w}_i) \to \text{extr}$$

Приравнивая производную к нулю, получаем:

$$S \boldsymbol{w}_i = \lambda_i \boldsymbol{w}_i$$

Подставляя обратно в критерий, получаем:

$$\tilde{J} = \boldsymbol{w}_i S \boldsymbol{w}_i = \lambda_i$$

Альтернативная интерпретация PCA

Рассмотрим в качестве критерия проектирования максимизацию дисперсии спроектированных данных. Пусть $d = 1$, т.е. необходимо найти некоторое направление $\boldsymbol{w}_1$: $\boldsymbol{w}_1^T \boldsymbol{w}_1 = 1$. Каждый объект выборки $\boldsymbol{x}_n$ проектируется в $\boldsymbol{w}_1^T \boldsymbol{x}_n$. Тогда дисперсия проекций вычисляется как

$$\frac{1}{N} \sum_{n=1}^{N} (\boldsymbol{w}_1^T \boldsymbol{x}_n - \boldsymbol{w}_1^T \bar{\boldsymbol{x}})^2 = \boldsymbol{w}_1^T S \boldsymbol{w}_1$$

Здесь

$$\bar{\boldsymbol{x}} = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_i, \quad S = \frac{1}{N} \sum_{n=1}^{N} (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T$$

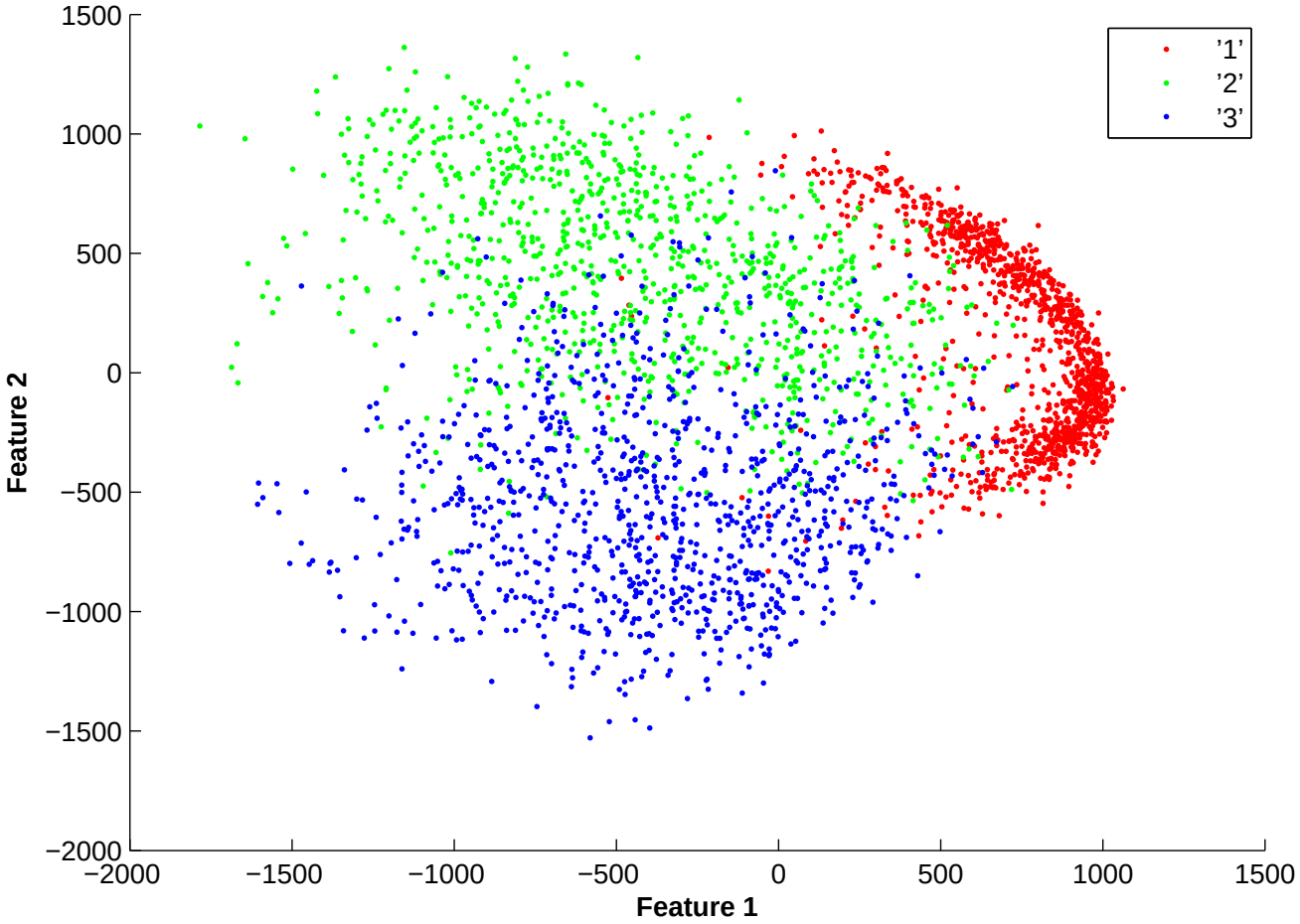
Альтернативная интерпретация II

Теперь необходимо максимизировать дисперсию проекций $\boldsymbol{w}_1^T \boldsymbol{S} \boldsymbol{w}_1$ при ограничении $\boldsymbol{w}_1^T \boldsymbol{w}_1 = 1$. Таким образом, мы приходим к точно такому же критерию, который возникал в случае минимизации невязки. Максимальная дисперсия достигается для собственного вектора выборочной матрицы ковариации, отвечающего максимальному собственному значению.

Рассматривая аналогично проектирование на новые направления, ортогональные уже найденным, получим, что наилучшее d-мерное линейное подпространство определяется собственными векторами выборочной матрицы ковариации, отвечающие d максимальным собственным значениям.

Пример с распознаванием рукописных цифр

Проекция данных на первые два признака (два собственных вектора матрицы ковариации, отвечающих наибольшим собственным значениям).



Интерпретация новых признаков

По построению каждой точке $\boldsymbol{x} \in \mathbb{R}^D$ соответствует некоторая картинка. Новые признаки $\boldsymbol{t} \in \mathbb{R}^d$ — проекции на выбранные направления (собственные вектора). Изменение одного признака в новом пространстве $\mathbb{R}^d$ соответствует движению вдоль собственного вектора в исходном пространстве $\mathbb{R}^D$.

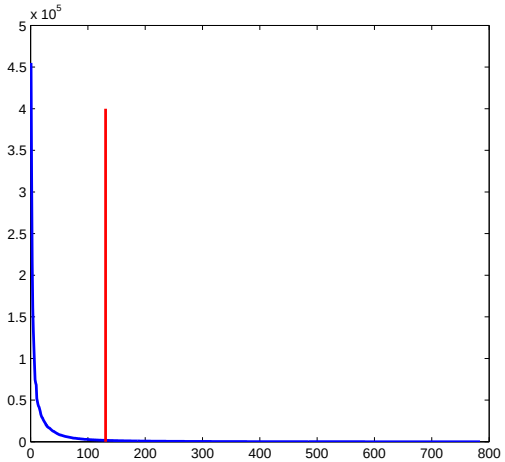
Движение вдоль первого признака:

Интерпретация: изменение ширины цифры

Движение вдоль второго признака:

Интерпретация: изменение характерного наклона цифры

Выбор размерности подпространства d в PCA



- Рассматриваем собственные значения в порядке убывания $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_D$
- Выбираем d так, что $\sum_{i=d+1}^{D} \lambda_i \ll \sum_{i=1}^{d} \lambda_i$ либо, чтобы

$$\sum_{i=1}^{d} \lambda_i \geq \gamma \sum_{i=1}^{D} \lambda_i$$

Здесь γ — доля объясняемой дисперсии, типичные значения 0.95, 0.99.

Неоднозначность решения PCA

- Если все собственные значения выборочной матрицы ковариации различны, т.е. $\lambda_1 > \lambda_2 > \cdots > \lambda_D$, то ортонормированный базис из собственных векторов определен однозначно, т.е. гиперплоскость проекции определена однозначно. Тем не менее, в полученной гиперплоскости базис может быть выбран произвольным образом.
- Если существуют одинаковые собственные вектора, то тогда гиперплоскость проекции определена не однозначно.

В PCA существует произвол в выборе координат объектов в новом пространстве

План лекции

Метод главных компонент (PCA)

Вероятностный метод главных компонент

- Вероятностная формулировка метода
- EM-алгоритм для PCA
- Выбор числа главных компонент с помощью байесовского п

Другие модели

Мотивация

PCA может быть сформулирован как вероятностная модель с латентными переменными, для оптимизации которой используется метод максимального правдоподобия.
Преимущества вероятностной постановки:

- Вычисление правдоподобия на тестовой выборке позволяет непосредственно сравнивать различные вероятностные модели
- EM-алгоритм для PCA позволяет быстро находить решения в ситуациях, когда требуется небольшое число лидирующих главных компонент, а также позволяет избежать вычисление выборочной матрицы ковариации в качестве промежуточного шага
- Возможность применения байесовского подхода для автоматического определения количества главных компонент (по аналогии с методом релевантных векторов)
- Можно работать со смесью PCA и использовать вариант EM-алгоритма для обучения в такой модели

Вероятностная модель PCA

Пусть имеется выборка $\{\boldsymbol{x}_n\}_{n=1}^N$, $\boldsymbol{x}_n \in \mathbb{R}^d$. Рассмотрим для каждого объекта $\boldsymbol{x}_n$ латентную переменную $\boldsymbol{t}_n \in \mathbb{R}^d$ как координаты объекта $\boldsymbol{x}_n$ в подпространстве, вообще говоря, меньшей размерности $d < D$. Определим априорное распределение в пространстве латентных переменных как

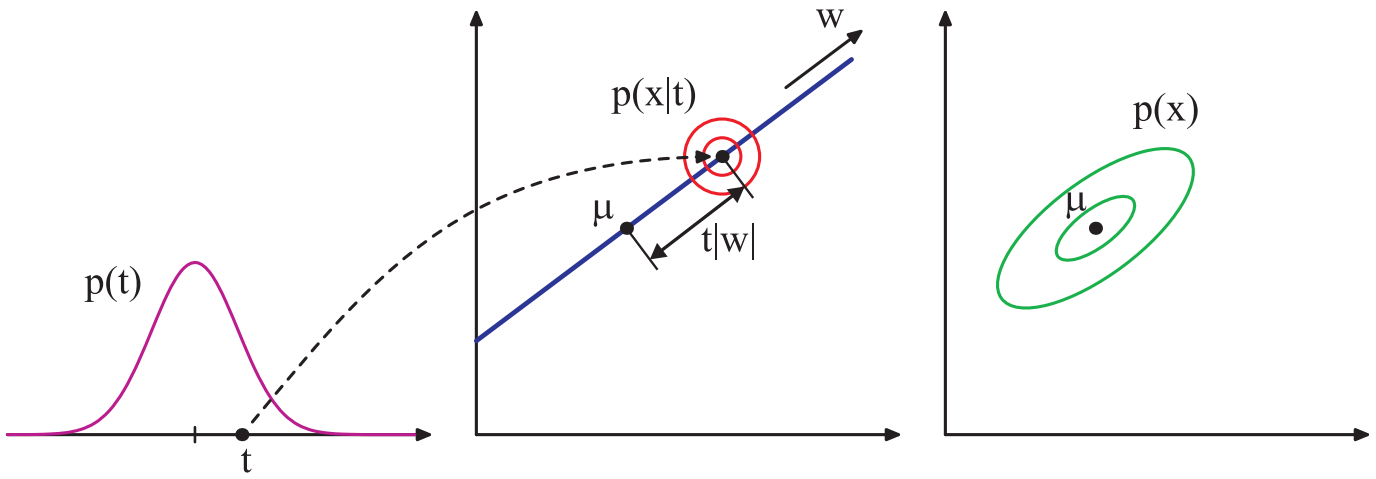
$$p(\boldsymbol{t}) = \mathcal{N}(\boldsymbol{t}|\boldsymbol{0}, I)$$

Модель наблюдаемых переменных $\boldsymbol{x}$ представляет собой линейное преобразование латентной переменной с добавлением гауссовского шума с единой дисперсией по всем направлениям:

$$p(\boldsymbol{x}|\boldsymbol{t}) = \mathcal{N}(\boldsymbol{x}|W\boldsymbol{t} + \boldsymbol{\mu}, \sigma^2 I)$$

Здесь $W \in \mathbb{R}^{D\times d}$, $\boldsymbol{\mu} \in \mathbb{R}^D$.

Иллюстрация вероятностной модели PCA



Функция правдоподобия

Функция правдоподобия может быть вычислена как

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{t})p(\boldsymbol{t})d\boldsymbol{t}$$

Этот интеграл является сверткой двух нормальных распределений и может быть вычислен аналитически. В результате получается снова нормальное распределение

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, C), \quad C = WW^T + \sigma^2 I$$

Действительно,

$$\begin{aligned} \mathbb{E}\boldsymbol{x} &= \mathbb{E}(W\boldsymbol{t} + \boldsymbol{\mu} + \boldsymbol{\varepsilon}) = \boldsymbol{\mu} \\ \mathbb{E}(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T &= \mathbb{E}(W\boldsymbol{t} + \boldsymbol{\varepsilon})(W\boldsymbol{t} + \boldsymbol{\varepsilon})^T = \\ &= W\mathbb{E}\boldsymbol{t}\boldsymbol{t}^T W^T + \mathbb{E}\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T = WW^T + \sigma^2 I \end{aligned}$$

Инвариантность относительно поворота координат в пространстве латентных переменных

Пусть R — некоторая ортогональная матрица. Рассмотрим матрицу базисных векторов $\tilde{W} = WR$. В этом случае вероятностная модель PCA приводит к тому же распределению, что и в случае матрицы W. Действительно, т.к. $RR^T = I$, то

$$C = \tilde{W}\tilde{W}^T + \sigma^2 I = WRR^T W^T + \sigma^2 I = WW^T + \sigma^2 I$$

Решение

Логарифм правдоподобия для рассматриваемой модели выглядит как

$$\log p(X|\boldsymbol{\mu}, W, \sigma^2) = \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\boldsymbol{\mu}, W, \sigma^2) =$$

$$= -\frac{ND}{2}\log 2\pi - \frac{N}{2}\log\det C - \frac{1}{2}\sum_{n=1}^{N}(\boldsymbol{x}_n - \boldsymbol{\mu})^T C^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu})$$

Дифференцируя правдоподобие и приравнивая производную к нулю, получаем:

$$\boldsymbol{\mu}_{ML} = \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}_n, \qquad \boldsymbol{\sigma}^2_{ML} = \frac{1}{D-d}\sum_{i=d+1}^{D}\lambda_i$$

$$W_{ML} = U_d(L_d - \sigma^2 I)^{1/2}R$$

Здесь $U_d \in \mathbb{R}^{D\times d}$ — матрица, состоящая из d собственных векторов выборочной матрицы ковариации, отвечающие d наибольшим собственным значениям $\lambda_1, \ldots, \lambda_d$, $L_d = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$, R — произвольная ортогональная матрица.

Апостериорное распределение латентной переменной

Можно показать, что

$$p(\boldsymbol{t}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{t}|M^{-1}W^T(\boldsymbol{x}-\boldsymbol{\mu}), \sigma^2 M^{-1})$$

Здесь $M = W^T W + \sigma^2 I$. Для решения задачи визуализации данных в пространстве латентных переменных требуется мат. ожидание латентной переменной по своему апостериорному распределению:

$$\mathbb{E}(\boldsymbol{t}) = M^{-1}W_{ML}(\boldsymbol{x}-\bar{\boldsymbol{x}})$$

EM-алгоритм в общем виде

Пусть имеется вероятностная модель, в которой:

- часть переменных X — известна
- часть переменных T — не известна
- имеется некоторый набор параметров Θ

Требуется оценить набор параметров Θ с помощью метода максимального правдоподобия:

$$p(X|\Theta) = \int p(X,T|\Theta)dT \to \max_{\Theta}$$

Схема EM-алгоритма

Требуется найти максимум правдоподобия в вероятностной модели со скрытыми переменными:

$$p(X|\Theta) = \int p(X,T|\Theta)dT \to \max_{\Theta} \Leftrightarrow \log\left(\int p(X,T|\Theta)dT\right) \to \max_{\Theta}$$

- E-шаг. Фиксируется значение параметров Θ_{old}. Оценивается апостериорное распределение на скрытые переменные $p(T|X,\Theta_{old})$, и полное правдоподобие усредняется по полученному распределению:

$$\mathbb{E}_{T|X,\Theta_{old}} \log p(X,T|\Theta) = \int \log p(X,T|\Theta) p(T|X,\Theta_{old})dT$$

- M-шаг. Фиксируется апостериорное распределение $p(T|X,\Theta_{old})$, и производится поиск новых значений параметров Θ_{new}:

$$\Theta_{new} = \arg\max_{\Theta} \mathbb{E}_{T|X,\Theta_{old}} \log p(X,T|\Theta)$$

- Шаги E и M повторяются до сходимости.

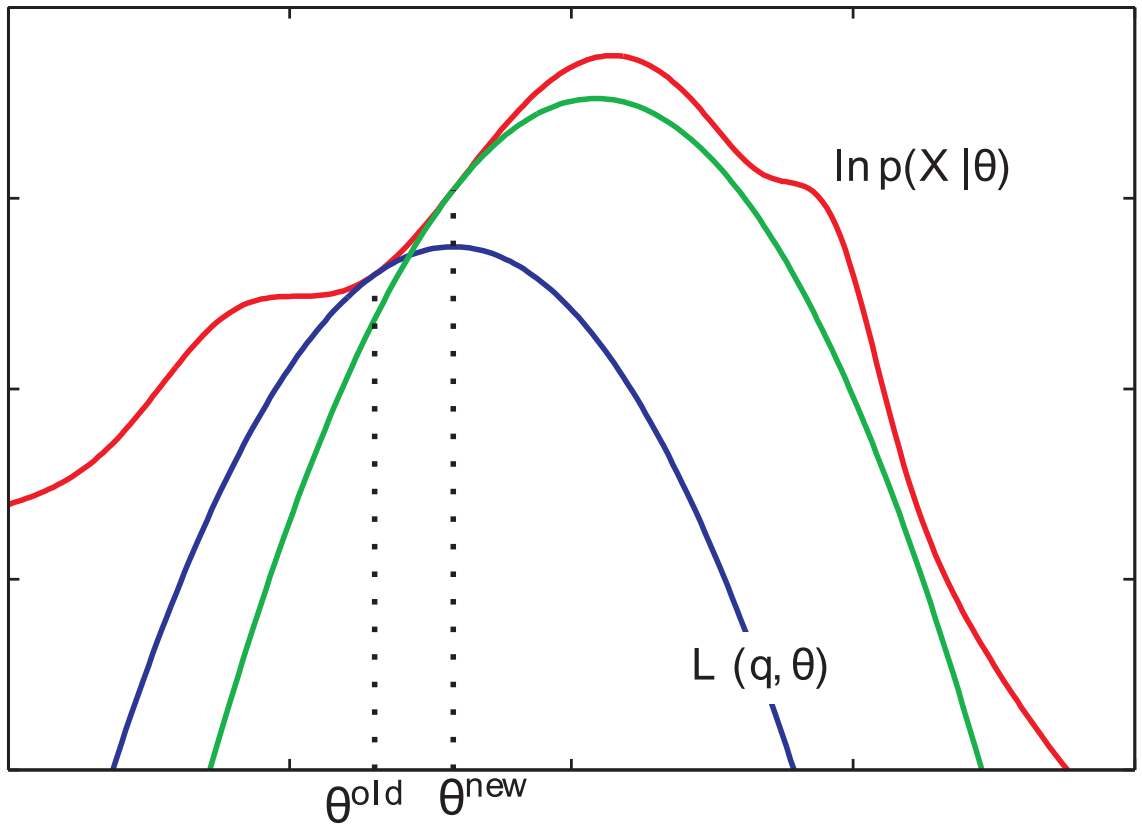
Нижняя оценка для логарифма правдоподобия

Можно показать, что

$$\log p(X|\Theta) \geq \mathbb{E}_{T|X,\Theta_{old}} \log p(X,T|\Theta) + Const \text{ и } =\Leftrightarrow \Theta = \Theta_{old}$$

Здесь *Const* - некоторая константа, не зависящая от Θ.

Иллюстрация EM-алгоритма



EM-алгоритм для PCA

$$\mathbb{E}\boldsymbol{t}_n = M^{-1}W^T(\boldsymbol{x} - \overline{\boldsymbol{x}})$$

$$\mathbb{E}\boldsymbol{t}_n\boldsymbol{t}_n^T = \sigma^2 M^{-1} + \mathbb{E}\boldsymbol{t}_n\mathbb{E}\boldsymbol{t}_n^T$$

$$W_{new} = \left[\sum_{n=1}^{N}(\boldsymbol{x}_n - \overline{\boldsymbol{x}})\mathbb{E}\boldsymbol{t}_n^T\right]\left[\sum_{n=1}^{N}\mathbb{E}\boldsymbol{t}_n\boldsymbol{t}_n^T\right]^{-1}$$

$$\sigma^2_{new} = \frac{1}{ND}\sum_{n=1}^{N}[\|\boldsymbol{x}_n - \overline{\boldsymbol{x}}\|^2 - 2\mathbb{E}\boldsymbol{t}_n^T W_{new}^T(\boldsymbol{x}_n - \overline{\boldsymbol{x}}) + \mathrm{tr}(\mathbb{E}\boldsymbol{t}_n\boldsymbol{t}_n^T W_{new}^T W_{new})]$$

Мотивация EM-алгоритма для PCA

Несмотря на существование решения для W и σ^2 в явном виде, использование EM-алгоритма может быть предпочтительным в ряде случаев:

- EM-алгоритм избегает вычисления выборочной матрицы ковариации (сложность $O(ND^2)$) и поиска ее собственных значений (сложность $O(D^3)$). Самые сложные операции в EM-алгоритме требуют $O(NDd)$ и $O(d^3)$, что может дать существенный выигрыш в скорости для данных больших размерностей.
- EM-алгоритм может быть применен для модели факторного анализа, для которой не существует решения в явном виде. Эта модель полностью повторяет вероятностную модель для PCA, однако различные факторы могут иметь разную дисперсию:

$$p(\boldsymbol{x}|\boldsymbol{t}) = \mathcal{N}(\boldsymbol{x}|W\boldsymbol{t} + \boldsymbol{\mu}, \Psi)$$

Здесь Ψ — диагональная матрица.

Проблема выбора количества главных компонент d

- Параметр d является типичным структурным параметром и должен задаваться до старта EM-алгоритма
- Одним из возможных способов выбора d является визуальное оценивание распределения собственных значений выборочной матрицы ковариации с выделением области, в которой собственные значения не отличаются существенно от нуля. К сожалению, на практике такой порог часто провести не удается.
- Разумной альтернативой является байесовский подход

Априорное распределение на базисные вектора

Выберем в качестве априорного распределения на веса нормальное распределение с отдельными коэффициентами регуляризации для каждого базисного вектора:

$$p(W|\boldsymbol{\alpha}) = \prod_{i=1}^{D} \left(\frac{\alpha_i}{2\pi}\right)^{D/2} \exp\left(-\frac{1}{2}\alpha_i \boldsymbol{w}_i^T \boldsymbol{w}_i\right)$$

Здесь $\boldsymbol{w}_i$ — i-ая колонка матрицы W. Для подбора коэффициентов $\boldsymbol{\alpha}$ можно использовать максимизацию обоснованности:

$$p(X|\boldsymbol{\alpha}, \boldsymbol{\mu}, \sigma^2) = \int p(X|W, \boldsymbol{\mu}, \sigma^2) p(W|\boldsymbol{\alpha}) dW$$

Данный интеграл не берется аналитически, поэтому возможными путями являются приближение Лапласа и вариационный подход.

Процедура обучения

Использование приближения Лапласа приводит к следующей формуле пересчета коэффициентов регуляризации:

$$\alpha_i^{new} = \frac{D}{\boldsymbol{w}_i^T \boldsymbol{w}_i}$$

Таким образом, процедура обучения является итерационной. На каждой итерации для текущих значений $\boldsymbol{\alpha}$ с помощью EM-алгоритма оцениваются $W, \boldsymbol{\mu}, \sigma^2$, а затем коэффициенты $\boldsymbol{\alpha}$ пересчитываются. В EM-алгоритме формула для пересчета W выглядит следующим образом:

$$W_{new} = \left[\sum_{n=1}^{N} (\boldsymbol{x}_n - \bar{\boldsymbol{x}}) \mathbb{E}\boldsymbol{t}_n^T\right] \left[\sum_{n=1}^{N} \mathbb{E}\boldsymbol{t}_n \boldsymbol{t}_n^T + \sigma^2 A\right]^{-1}$$

Здесь $A = \text{diag}(\alpha_1, \ldots, \alpha_D)$.

План лекции

Недостатки PCA

- Метод главных компонент способен находить только линейные подпространства исходного пространства, которые «объясняют» данные с высокой точностью. На практике поверхность, вдоль которой располагаются данные, может быть существенно нелинейной
- PCA является инвариантным относительно поворота координат в пространстве латентных переменных. Это означает, что восстановление значений латентных переменных является неоднозначным. В ряде случаев такая ситуация может быть неадекватной, например, в задаче разделения независимых источников, представленных линейной смесью с неизвестными коэффициентами.

Метод независимых компонент, ICA

Пусть наблюдаемые данные $\boldsymbol{x} = (x_1, \ldots, x_d)$ являются линейной комбинацией независимых источников $\boldsymbol{t} = (t_1, \ldots, t_d)$:

$$\boldsymbol{x} = W\boldsymbol{t},$$

причем размерности $\boldsymbol{x}$ и $\boldsymbol{t}$ совпадают, а матрица W неизвестна.

Если преобразование W является невырожденным, то $\boldsymbol{t} = W^T\boldsymbol{x}$. Рассмотрим задачу поиска одного источника t_1. Будем искать t_1 как линейную комбинацию наблюдаемых данных $t_1 = \boldsymbol{a}^T\boldsymbol{x}$. Задача – определить вектор $\boldsymbol{a}$.

Если $\boldsymbol{a}$ совпадает с колонкой матрицы W, то t_1 будет одним из искомых источников.

$$t_1 = \boldsymbol{a}^T\boldsymbol{x} = \boldsymbol{a}^T W\boldsymbol{t} = \{\boldsymbol{z} \triangleq W^T\boldsymbol{a}\} = \boldsymbol{z}^T\boldsymbol{t} = z_1t_1 + \cdots + z_dt_d$$

По центральной предельной теореме сумма случайных величин приближается к нормальному распределению. Следовательно, чем больше слагаемых в сумме $z_1t_1 + \cdots + z_dt_d$, тем более она похожа на гауссиану. Отсюда

$\boldsymbol{a}$ должен быть таким, чтобы $\boldsymbol{a}^T\boldsymbol{x}$ было как можно меньше похоже на гауссиану.

Критерии негауссовости

- Эксцесс.

$$\text{kurt} = \mathbb{E}t^4 - 3(\mathbb{E}t^2)^2$$

В предположении, что случайная величина t нормализована, $\text{kurt} = \mathbb{E}t^4$. Для гауссианы значение эксцесса равно 0, для остальных распределений может быть как положительным, так и отрицательным. Поэтому в качестве меры негауссовости выбирают модуль или квадрат эксцесса.

- Негэнтропия.

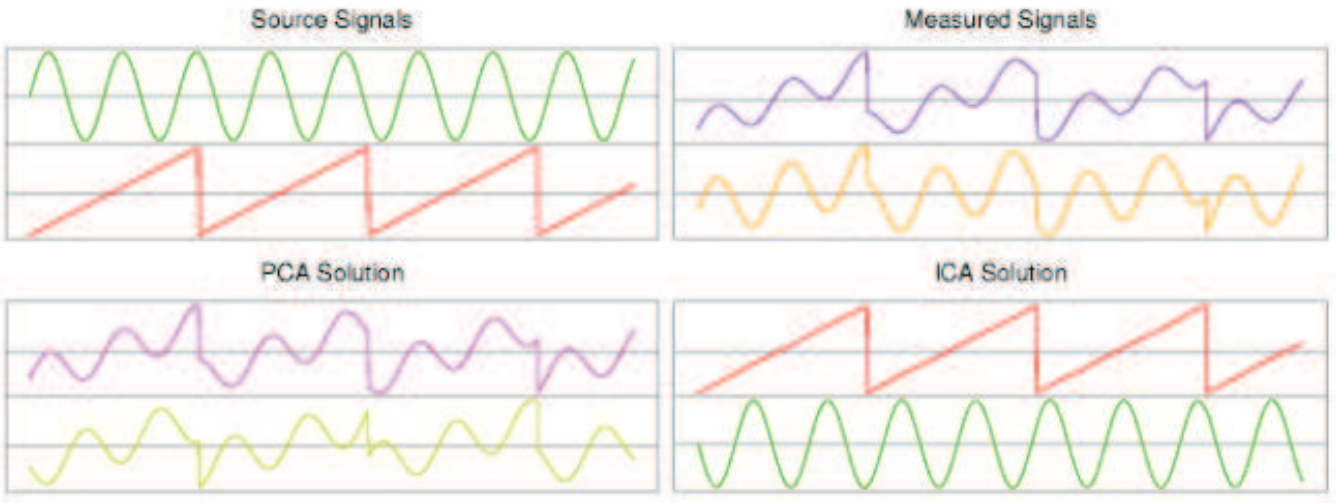
$$H(t) = -\int \log p(t)p(t)dt$$

У гауссовского распределения энтропия $H(t)$ является максимальной среди всех распределений с одинаковой матрицей ковариации. Поэтому в качестве меры негауссовости используется негэнтропия

$$H(t_{gauss}) - H(t)$$

Здесь t_{gauss} — гауссиана с той же матрицей ковариации, что и t. Негэнтропия является неотрицательной, и ее нужно максимизировать для поиска независимых источников.

ICA vs. PCA. Пример.



Анализ независимых факторов, Independent Factor Analysis, IFA

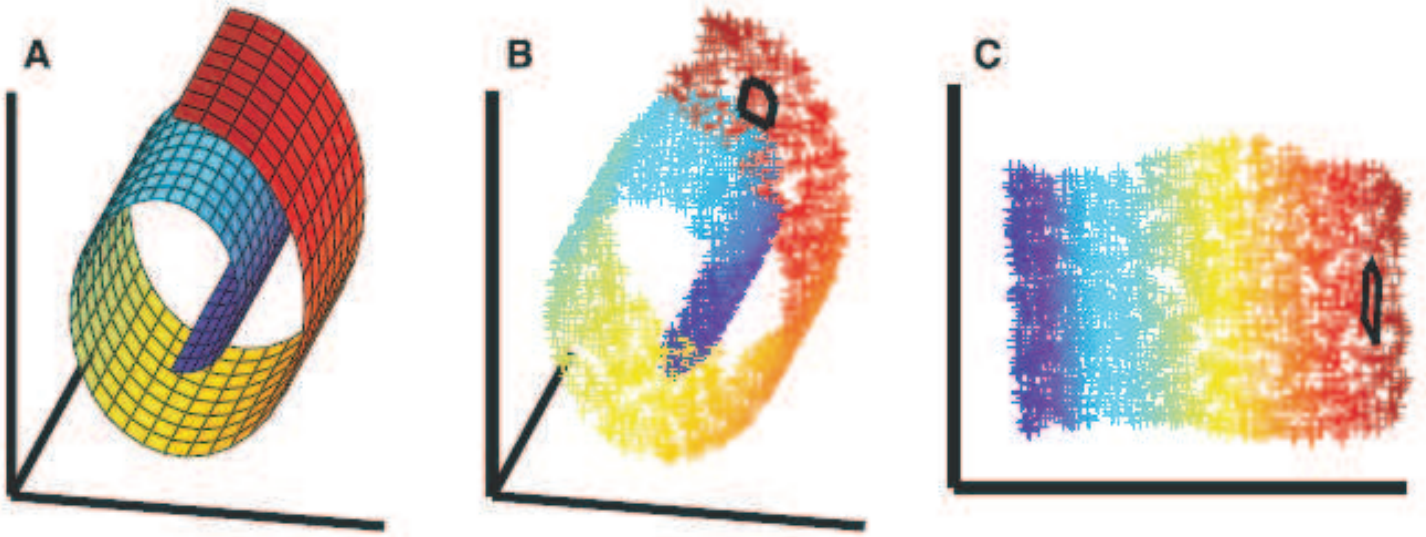
В методе PCA и факторном анализе в качестве априорного распределения латентных переменных предполагается выбирать нормальное распределение с центром в нуле и некоторой дисперсией, возможно, различной для разных направлений. В анализе независимых факторов в качестве априорного распределения предлагается выбирать независимое:

$$p(\boldsymbol{t}) = \prod_{i=1}^{d} p(\boldsymbol{t}_i), \qquad p(\boldsymbol{x}|\boldsymbol{t}) = \mathcal{N}(\boldsymbol{x}|W\boldsymbol{t} + \boldsymbol{\mu}, \Psi)$$

С одной стороны, данная модель является существенным обобщением PCA и позволяет находить, вообще говоря, нелинейные зависимости, с другой стороны, модель является достаточно простой, чтобы можно было предложить варианты EM-алгоритма для ее оптимизации.

Одним из возможных вариантов реализации данного метода является моделирование распределений отдельных факторов с помощью смеси гауссиан с разным числом компонентов для разных факторов.

Локальное линейное погружение, Local Linear Embedding, LLE



Если данные располагаются вдоль некоторой поверхности, то разумно искать такое преобразование признакового пространства, при котором близкие объекты на этой поверхности были бы близки в новом пространстве. При этом близкие в евклидовом смысле объекты в результате такого преобразования могут оказаться очень далекими.

PCA не позволяет находить преобразования с подобными свойствами!

Схема LLE

Основная идея — искать преобразование с сохранением отношения соседства между объектами

- Восстановление структуры соседства на обучающей выборке:

$$\varepsilon(W) = \sum_{n=1}^{N} \Big\| \boldsymbol{x}_n - \sum_{j \in Neighbours(\boldsymbol{x}_n)} W_{nj} \boldsymbol{x}_j \Big\|^2 \to \min_W$$

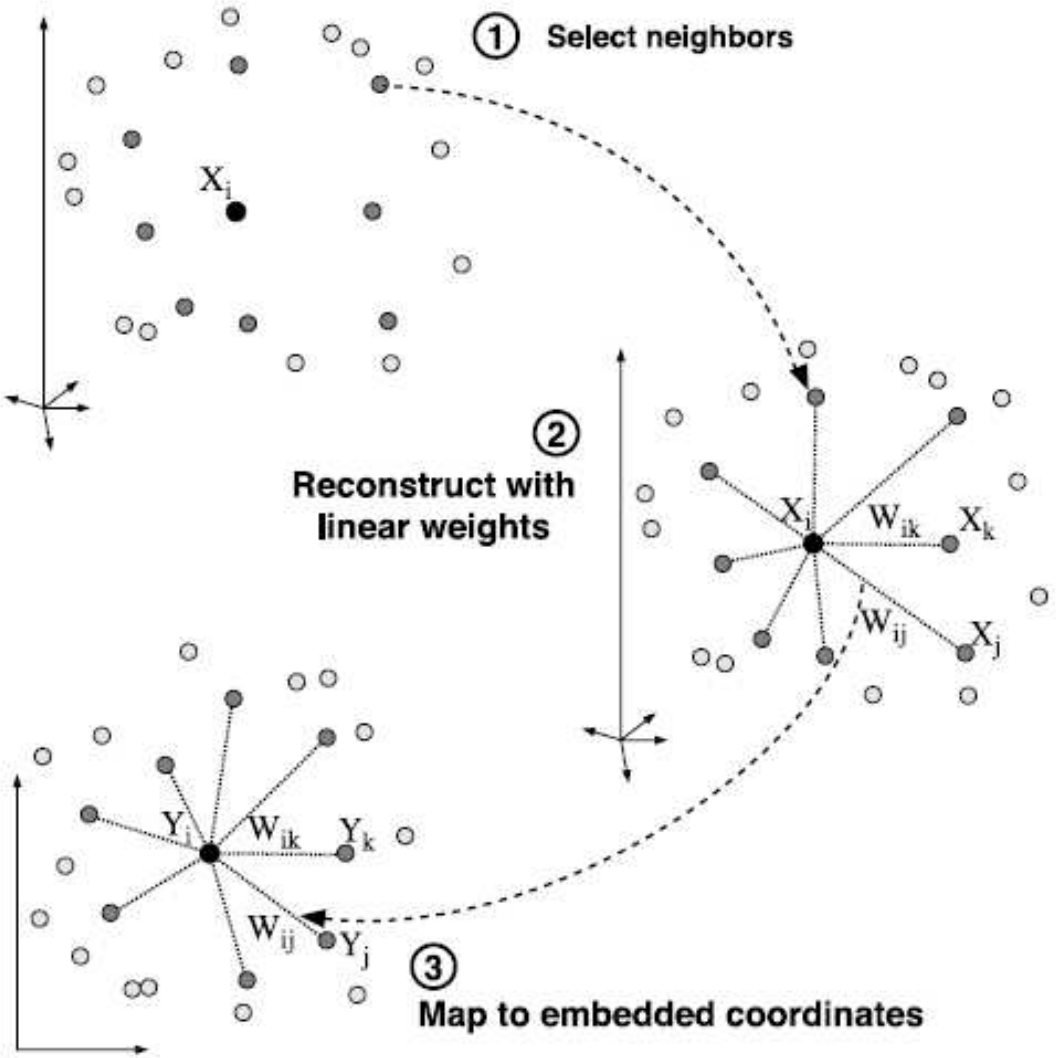
$$\sum_j W_{nj} = 0$$

- Поиск координат в новом пространстве со схожей структурой соседства:

$$\Phi(T) = \sum_{n=1}^{N} \Big\| \boldsymbol{t}_n - \sum_{j \in Neighbours(\boldsymbol{x}_n)} W_{nj} \boldsymbol{t}_j \Big\|^2 \to \min_T$$

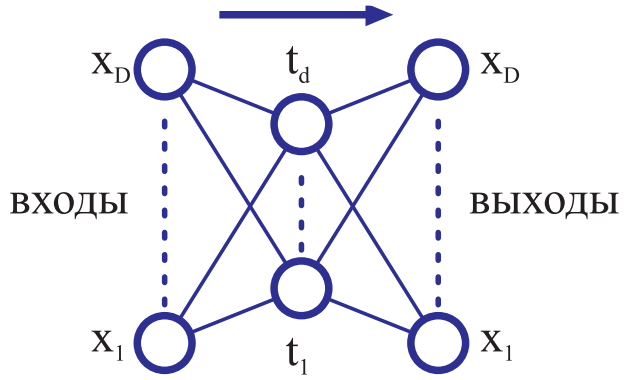
$$\sum_{n=1}^{N} \boldsymbol{t}_n = \boldsymbol{0}, \quad \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{t}_n \boldsymbol{t}_n^T = I$$

Иллюстрация LLE



Автоассоциативные нейронные сети

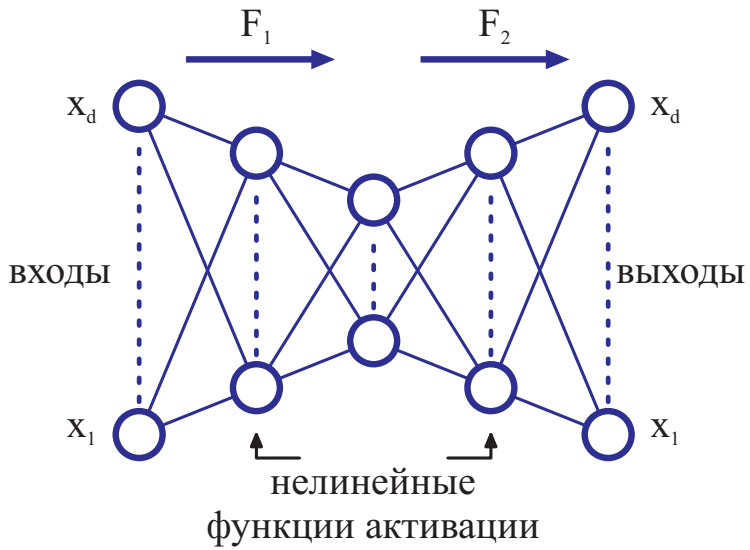
Рассмотрим многослойный персептрон следующего вида:

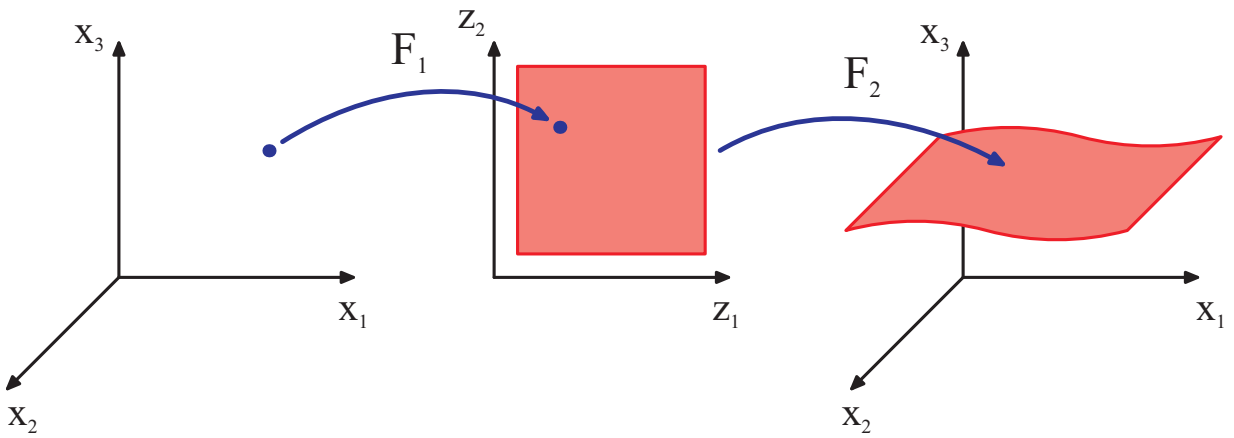


Здесь $d < D$. Задача нейронной сети — объяснить как можно точнее входные данные с помощью выходных. Таким образом, функционалом качества обучения является следующий:

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N} \|y(\boldsymbol{x}_n, \boldsymbol{w}) - \boldsymbol{x}_n\|^2$$

Если в сети только один скрытый слой, то даже с использованием нелинейной сигмоидной функцией активации результат обучения сети совпадает с результатом PCA.


F1
F2
xd
xd
ВХОДЫ
ВЫХОДЫ
x1
x1
нелинейные
функции активации


x3
x1
x2
F1
z2
z1
F2
x3
x1
x2

Generative Topographic Mapping, GTM

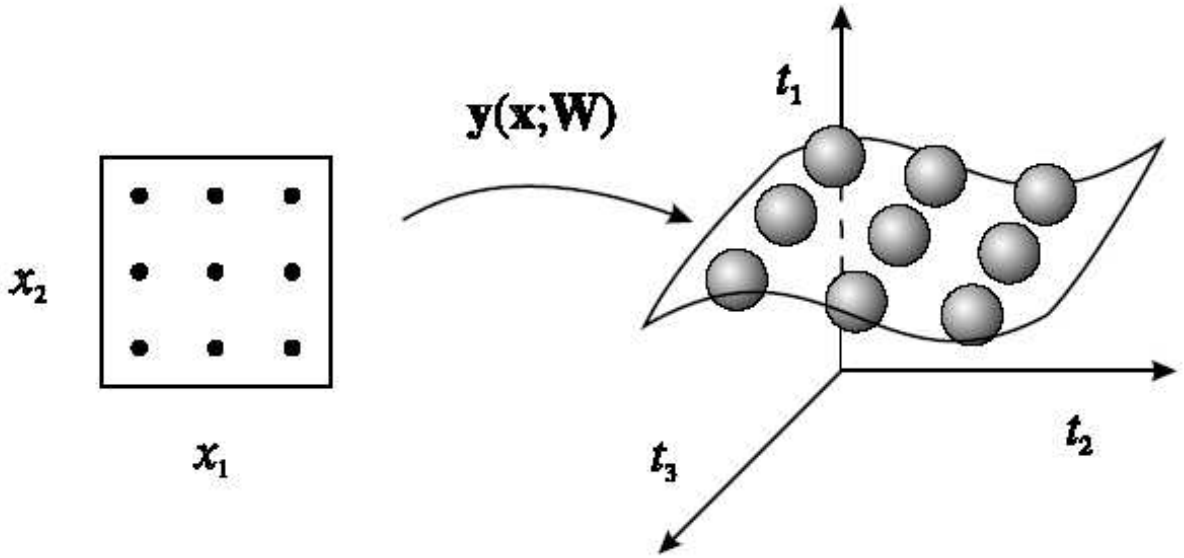
Метод GTM используется, в основном, для визуализации данных и является вероятностным обобщением самоорганизующихся сетей Кохонена.
В модели GTM предполагается, что данные образуются в результате нелинейного преобразования латентных переменных с добавлением нормального шума:

$$p(\boldsymbol{x}|\boldsymbol{t}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{y}(\boldsymbol{t}, W), \sigma^2 I)$$

Пространство латентных переменных является двухмерным, а распределение латентных переменных представляет собой сумму дельта-функций с центрами в некоторой регулярной сетке:

$$p(\boldsymbol{t}) = \frac{1}{l} \sum_{j=1}^{l} \delta(\boldsymbol{t} - \boldsymbol{t}_j)$$

Иллюстрация GTM



Функция правдоподобия выглядит следующим образом:

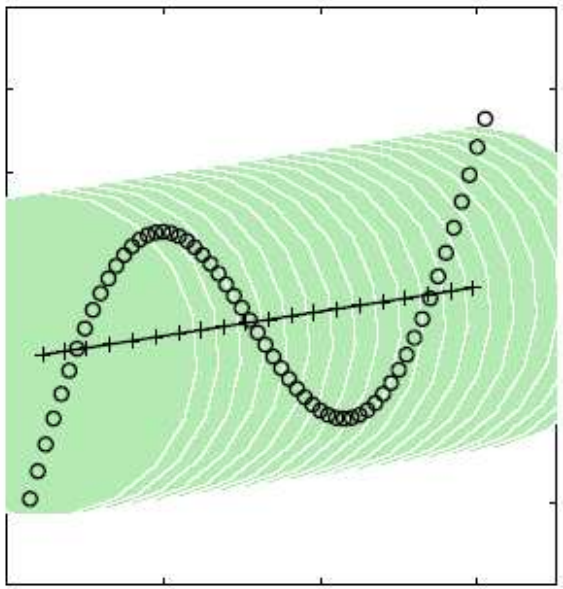
$$\log p(X|W,\sigma^2) = \sum_{n=1}^{N} \log \left[\frac{1}{l} \sum_{j=1}^{l} p(\boldsymbol{x}_n|\boldsymbol{t}_j, W, \sigma^2) \right]$$

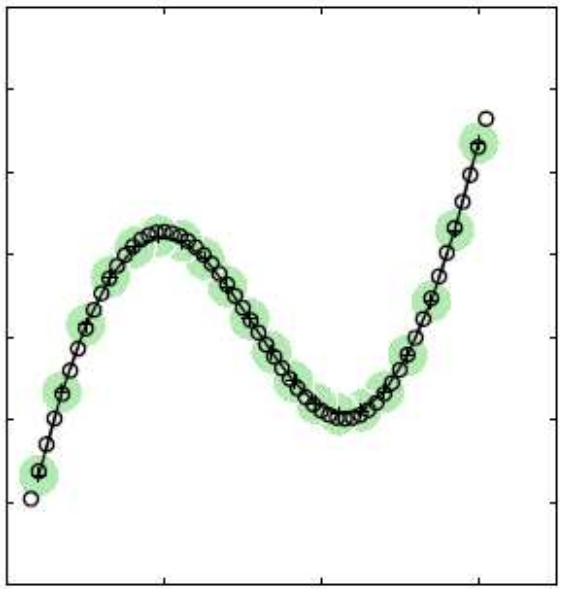
Иллюстрация GTM

В качестве функции регрессии предлагается взять обобщенную линейную:

$$\boldsymbol{y}(\boldsymbol{t}, W) = W\boldsymbol{\phi}(\boldsymbol{t})$$

Здесь $\{\phi_j(\boldsymbol{t})\}_{j=1}^{d}$ — набор фиксированных базисных функций, в качестве которых могут выступать, например, гауссианы с центрами в регулярной сетке. Тогда можно предложить эффективный EM-алгоритм для оптимизации такой модели.





Разрезы графов

Ю. И. Журавлев[1], Д. П. Ветров[1]

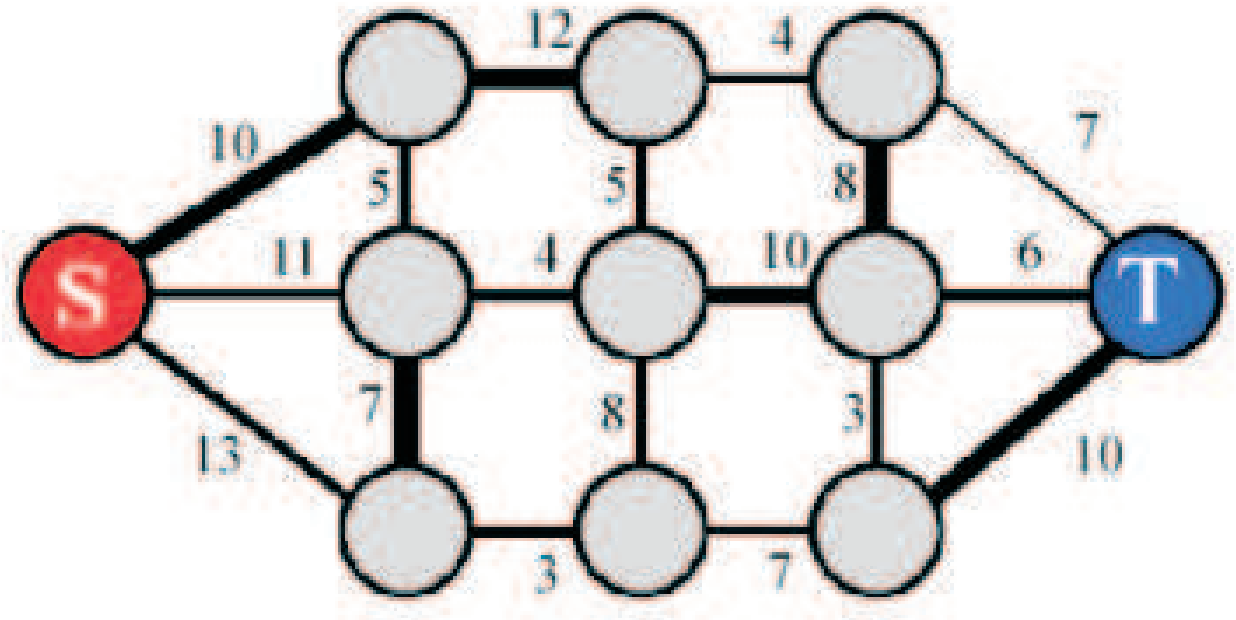
[1]МГУ, ВМиК, каф. ММП

Курс «Математические основы теории прогнозирования»

План

Потоки в сетях

- Рассмотрим неориентированный граф с двумя выделенными вершинами (стоком t и истоком s)
- Пусть с каждым ребром $(u, v) \in E$ ассоциировано некоторое неотрицательное число $c(u, v) \geq 0$ — пропускная способность



Потоки в сетях

- Назовем потоком неотрицательную функцию $f(u, v)$, определенную на ребрах графа, такую что

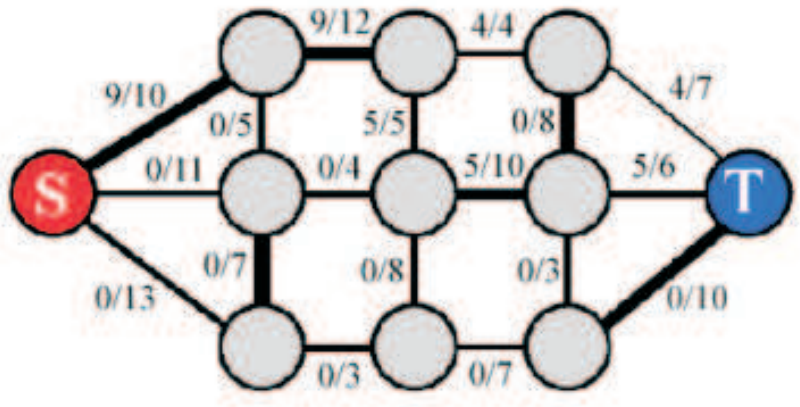
$$f(u, v) \leq c(u, v)$$

$$\sum_{v:(u,v)\in E} f(u, v) - \sum_{v:(v,u)\in E} f(v, u) = 0, \quad \forall u \neq \{s, t\}$$

- Первое условие ограничивает поток через ребро его пропускной способностью, а второе гарантирует отсутствие источников и стоков вне выделенной пары вершин
- Задача поиска максимального потока состоит в максимизации величины

$$M(f) = \sum_{v:(s,v)\in E} f(s, v) \to \max_f$$

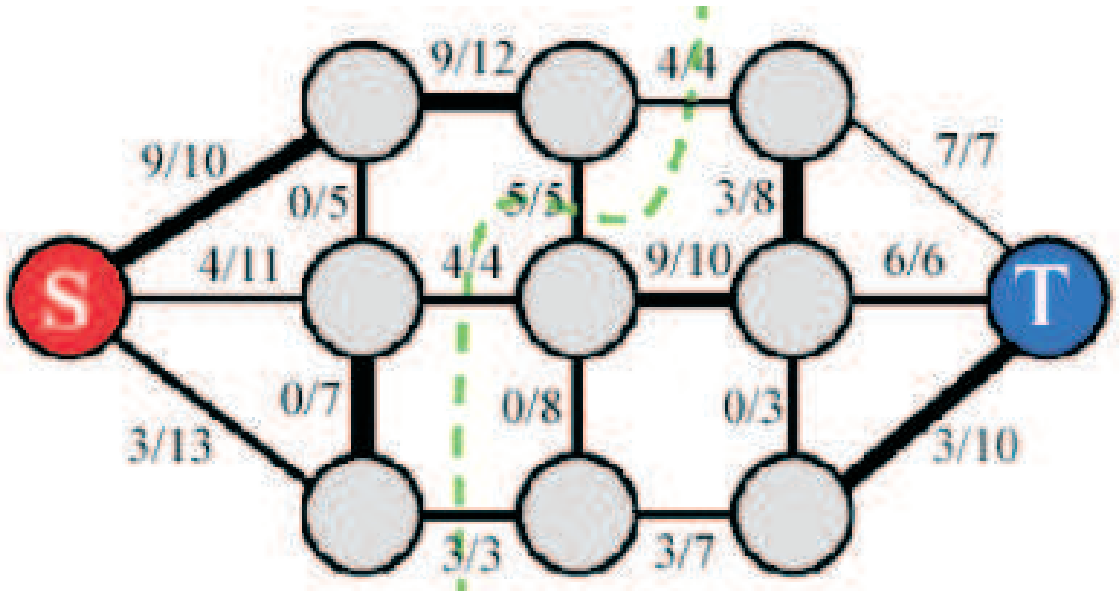
по всем допустимым потокам



Разрезы графов

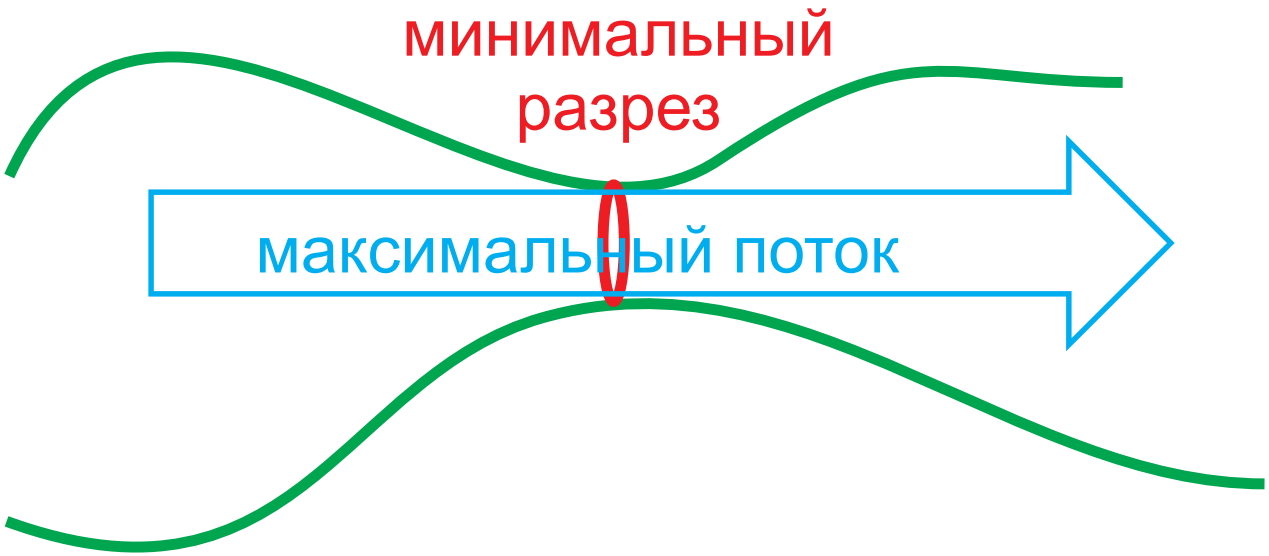
- $(s-t)$-разрезом графа называется разбиение вершин графа на два непересекающихся множества S и T, такие что $s \in S$, $t \in T$
- Величиной разреза называется сумма пропускных способностей всех ребер, один конец которых находится в множестве S, а другой — в множестве T

$$c(S,T) = \sum_{(u,v)\in E, u\in S, v\in T} c(u,v)$$



Теорема Форда-Фалкерсона

- Известная теорема Форда-Фалкерсона гласит, что максимальный поток в сети равен ее минимальному разрезу
- Существует эффективный (полиномиальной сложности) алгоритм решения задачи поиска минимального разреза в графе
- Задачи поиска максимального потока и минимального разреза являются двойственными

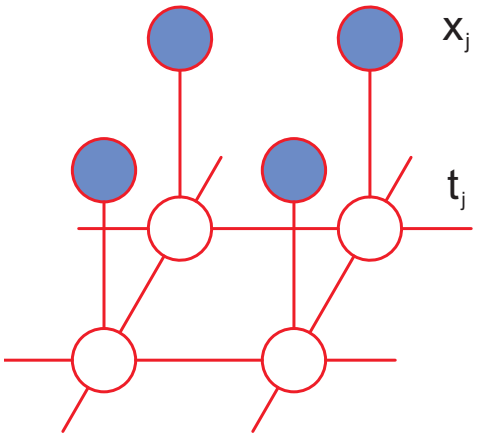


Марковские решетки

- В дальнейшем будем рассматривать т.н. марковские решетки, в которых размер максимальной клики не превосходит двух
- В этом случае совместное распределение переменных марковской сети выражается формулой

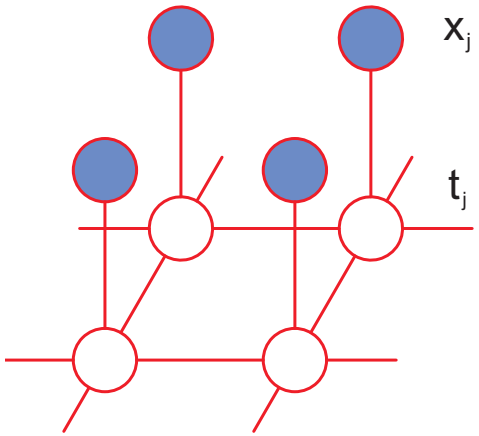
$$p(Y) = \frac{1}{Z} \prod_{(y_i, y_j) \in E} \psi_{ij}(y_i, y_j)$$

- Наиболее типичным примером таких сетей являются изображения



Марковские решетки с бинарными переменными

- В дальнейшем будем рассматривать марковские решетки такого вида

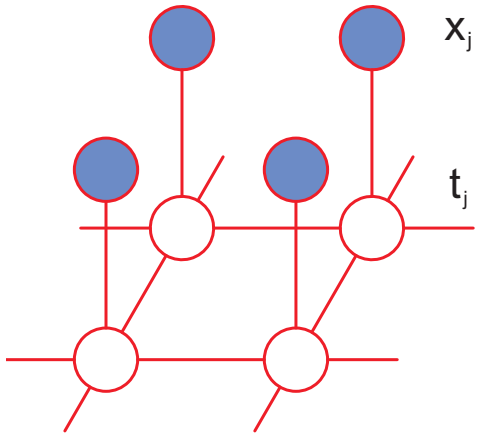


- Необходимо по наблюдаемым переменным X восстановить наиболее вероятные значения скрытых переменных T

$$T_{MP} = \arg\max_{T} P(T|X)$$

- Остановимся на важном частном случае, когда скрытые переменные бинарные $t \in \{0, 1\}$

Марковские решетки с бинарными переменными



- Распределение скрытых переменных марковской сети в этом случае выглядит так

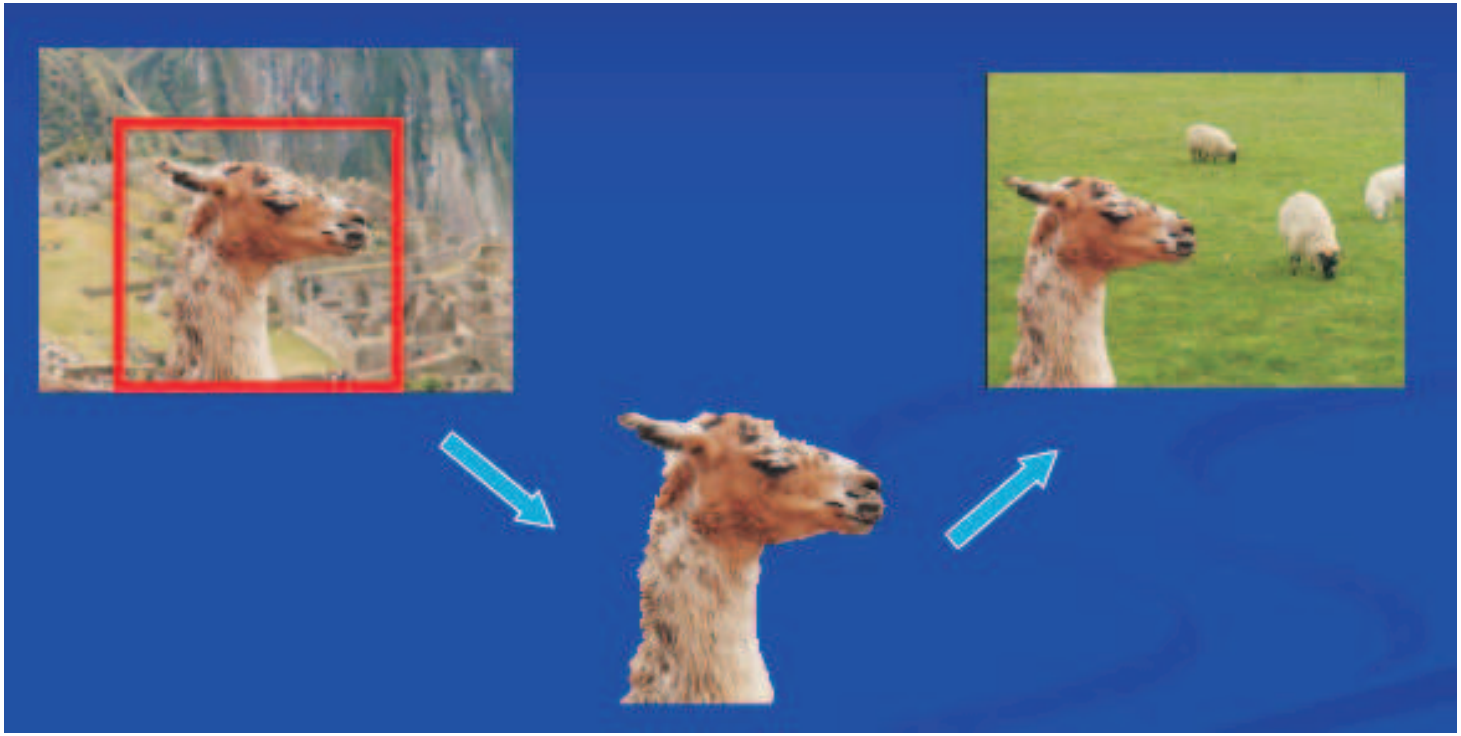
$$p(T|X) = \frac{p(X,T)}{p(X)} = \prod_{(i,j)\in E} \psi_{ij}(t_i, t_j) \prod_i \psi_i(x_i, t_i) \times \text{Const}$$

- В энергетической нотации задача максимизации этого распределения принимает вид минимизации энергии, что более удобно с вычислительной точки зрения

$$E(T|X) = \sum_{(i,j)\in E} E_{ij}(t_i, t_j) + \sum_i E_i(x_i, t_i) =$$

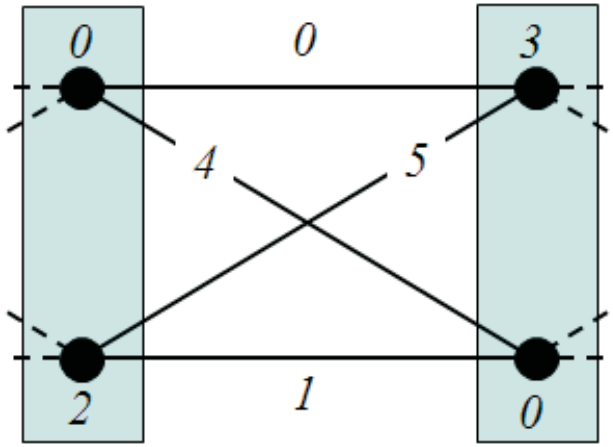
$$- \sum_{(i,j)\in E} \log \psi_{ij}(t_i, t_j) - \sum_i \log \psi_i(x_i, t_i) \to \min_T$$

Пример задачи сегментации



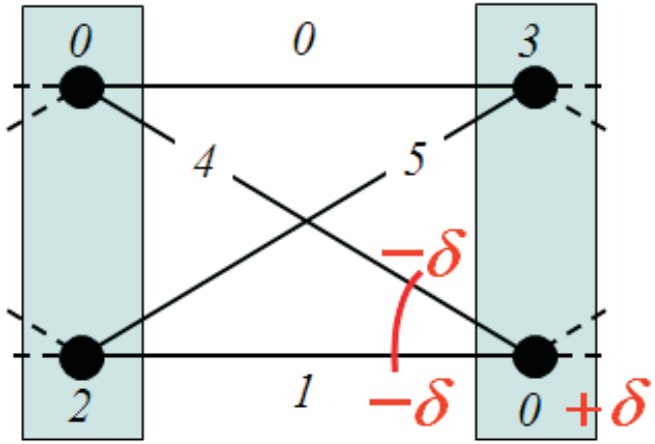
Значения скрытых переменных кодируют принадлежность каждого пикселя к объекту либо к фону. Использование графических моделей позволяет учесть, что соседние пиксели чаще всего относятся к одному классу

Репараметризация



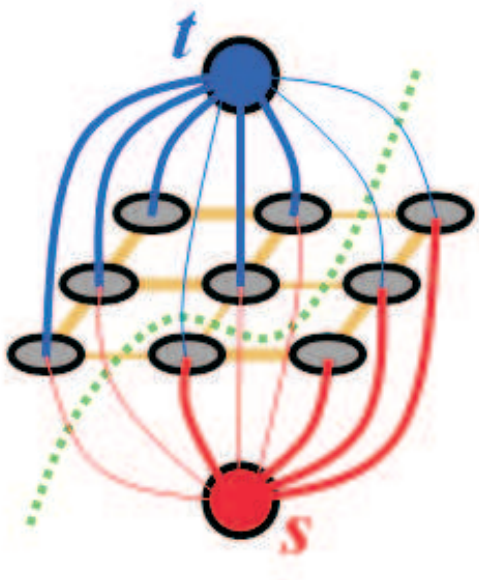
- Рассмотрим две соседние вершины t_i и t_j, каждая из которых может принимать два значения
- Тогда функция $E_i(x_i, t_i)$ задается двумя значениями (при известном x_i), а функция $E_{ij}(t_i, t_j)$ — четырьмя
- Для сведения к задаче о поиске минимального разреза нам понадобится выполнить т.н. репараметризацию, сделав «веса» горизонтальных ребер нулевыми

Репараметризация



- Вычитая одинаковые значения из двух ребер, сходящихся в одну вершину и прибавляя это же значение к весу самой вершины, мы получаем эквивалентный функционал энергии
- Значения такой энергии в каждой точке совпадает со значением исходной энергии
- Применение этой процедуры позволяет путем изменения функции E_i получить эквивалентный энергетический функционал, в котором $E_{ij}(0,0) = E_{ij}(1,1) = 0$ и $E_{ij}(0,1) = E_{ij}(1,0)$ для всех $(i,j) \in E$

Сведение к разрезу графов



- Определим на следующем графе пропускную способность таким образом:

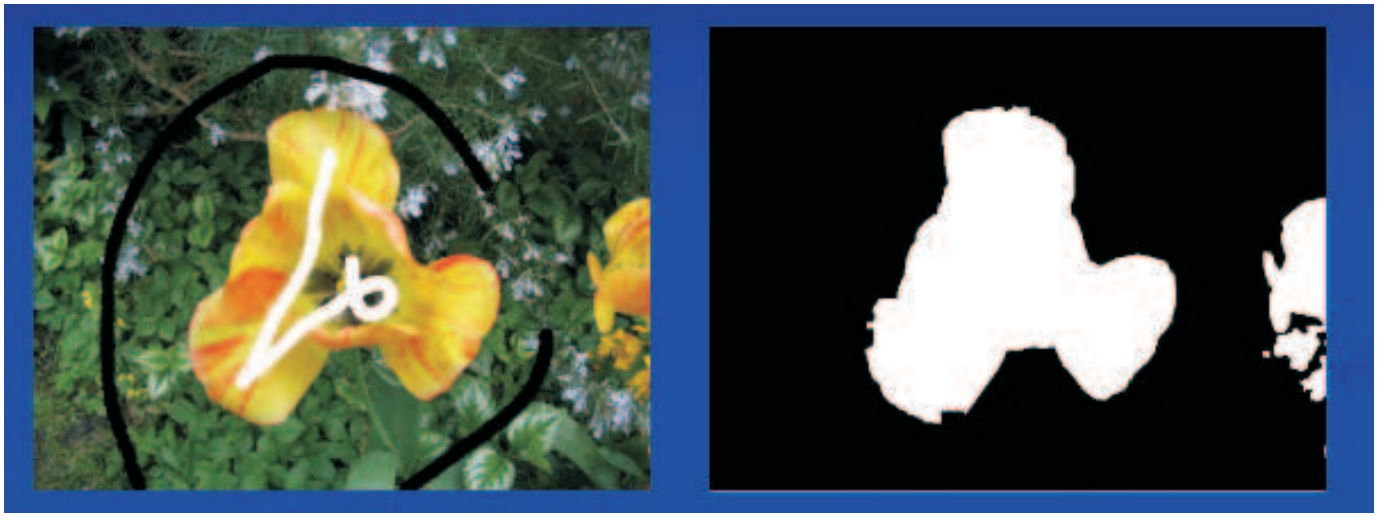
$$c(S,t_i) = E_i(x_i,0), \quad c(T,t_i) = E_i(x_i,1), \quad c(t_i,t_j) = E_{ij}(0,1),\ \forall (i,j) \in E$$

- Поиск минимального разреза в таком графе отвечает минимизации энергии

$$E(T|X) = \sum_{(i,j)\in E} E_{ij}(t_i,t_j) + \sum_i E_i(x_i,t_i)$$

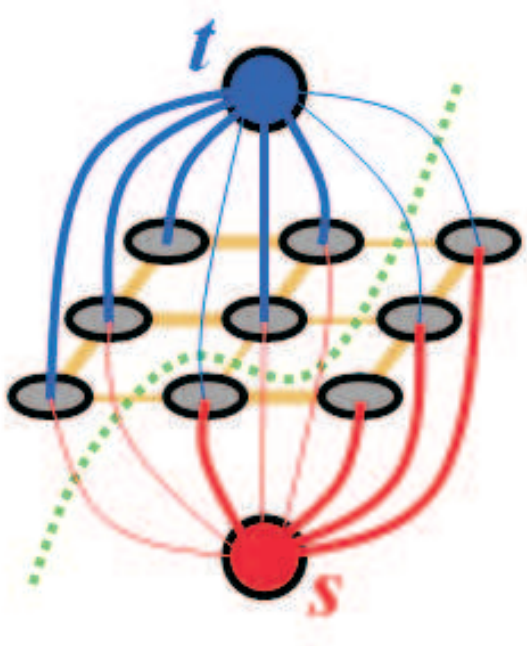
т.е. поиску наиболее вероятных значений T

Сегментация с семенами



- Часто для некоторых пикселей известно заранее, к какому классу они принадлежат
- Например, пользователь может задать фрагменты изображения и фона (семена)

Сведение к разрезу графов



- Пусть O — семена объекта ($t_i = 1$), а B — семена фона ($t_i = 0$)
- Тогда достаточно задать $c(S, t_i) = +\infty$, $\forall t_i \in O$ и $c(T, t_i) = +\infty$, $\forall t_i \in B$
- Этим мы запретим соответствующие разрезы, сделав невозможным отнесение семян объекта к фону и наоборот

Способы введения унарного слагаемого

По смыслу унарное (относительно скрытых переменных) слагаемое $E_i(x_i, t_i)$ задает насколько данный пиксель соответствует тому или иному классу. Оно может отражать следующую информацию:

- Цветовая модель — показывает насколько появление тех или иных цветов более вероятно в данном классе
- Позиционная модель — показывает априорные предположения о положении данного класса на изображении
- Текстурная модель — показывает насколько текстура окрестности пикселя вероятна для данного пикселя

Способы введения парного слагаемого

Парное слагаемое $E_{ij}(t_i, t_j)$ отражает степень взаимозависимостей классов соседних пикселей. Наиболее распространенными примерами являются

- Модель Поттса: $E_{ij}(t_i, t_j) = 1 - \delta(t_i, t_j)$ — штраф за несовпадение классов соседних пикселей
- Штраф за несовпадение классов с учетом контраста

$$E_{ij}(t_i, t_j) = \exp\left(-\frac{(x_i - x_j)^2}{2\sigma^2}\right)(1 - \delta(t_i, t_j))$$

  Чем сильнее различаются цвета (интенсивности) пикселей x_i, тем меньше штраф за несовпадение классов
- Заметим, что во втором случае парное слагаемое зависит от наблюдаемых переменных x_i и x_j - такая конструкция называется условным случайным полем

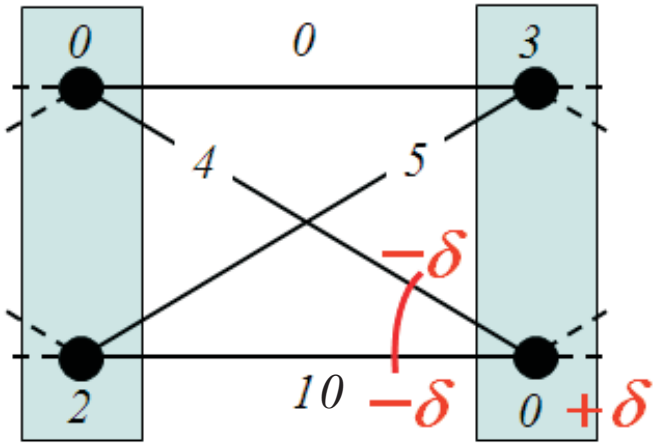
Субмодулярность

- Назовем энергию субмодулярной, если для всех ее парных слагаемых верно

$$E_{ij}(0,0) + E_{ij}(1,1) \leq E_{ij}(0,1) + E_{ij}(1,0)$$

- Условие субмодулярности является в некотором смысле аналогом выпуклости для функций бинарного переменного
- Унарное слагаемое при этом может быть произвольным
- Легко показать, что с помощью разрезов графов можно оптимизировать именно субмодулярную энергию

Доказательство необходимости

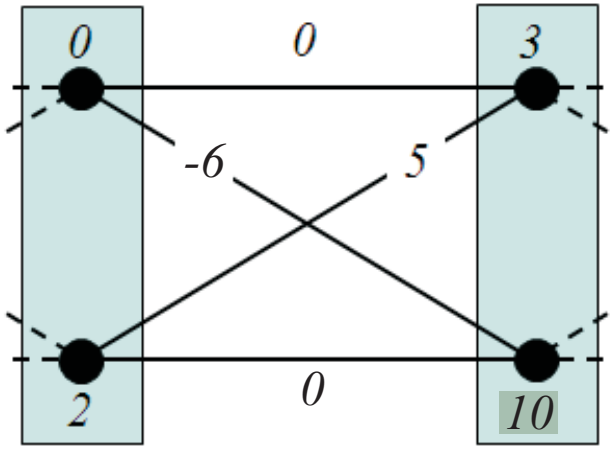


- Пусть имеется парное слагаемое, не являющееся субмодулярной функцией

$$E_{ij}(0,0) + E_{ij}(1,1) > E_{ij}(0,1) + E_{ij}(1,0)$$

- Тогда и только тогда в результате репараметризации нулевые веса горизонтальных связей приведут к возникновению отрицательных диагональных связей

Доказательство необходимости

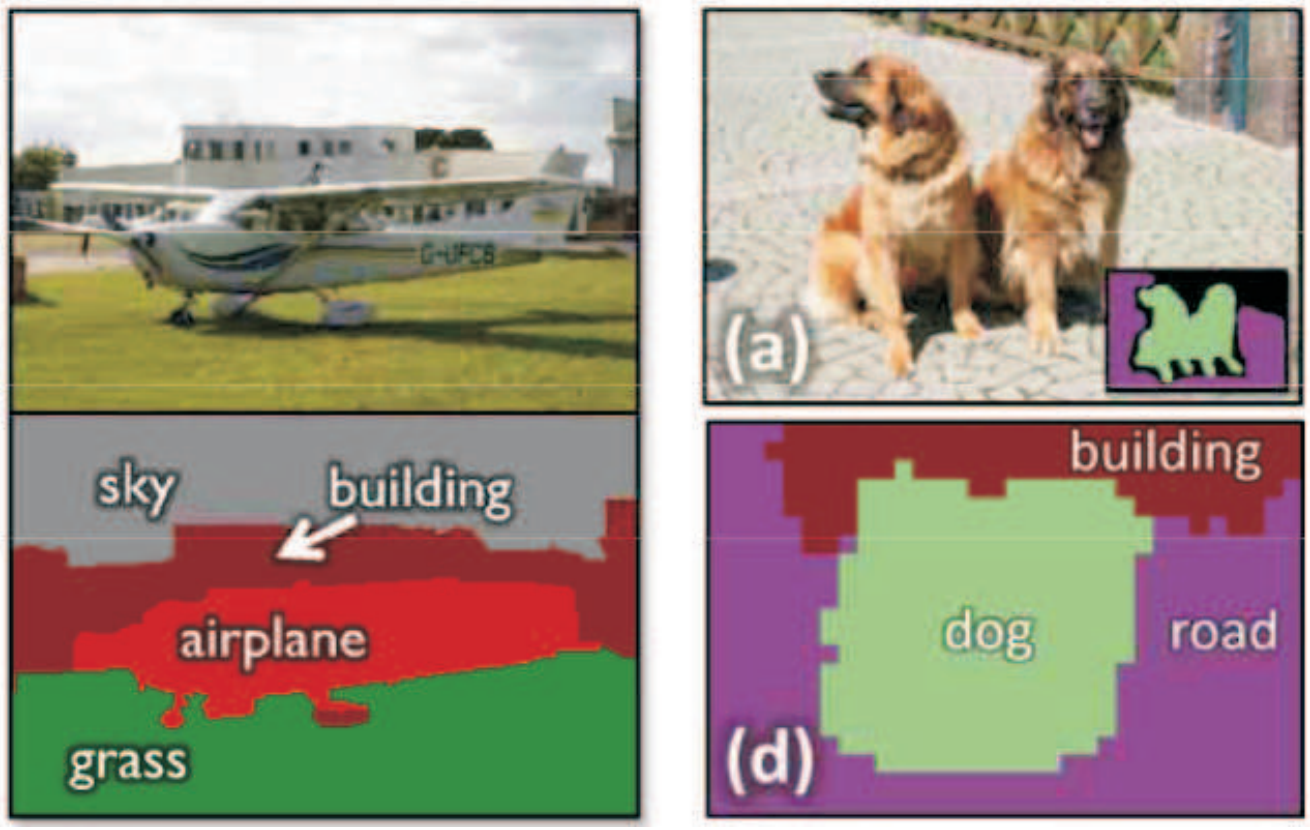


- Но это означает, что пропускная способность некоторых ребер в графе, разрез которого мы будем минимизировать, станет отрицательной!
- В этой ситуации классические полиномиальные алгоритмы поиска минимального разреза в графе неприменимы
- Задача оптимизации энергии стала NP-трудной
- Существуют некоторые обобщения полиномиального алгоритма на случаи, когда энергия может быть сведена в субмодулярной путем замены части переменных на свои отрицания: $t_i \to (1 - t_i)$

Случай небинарных переменных

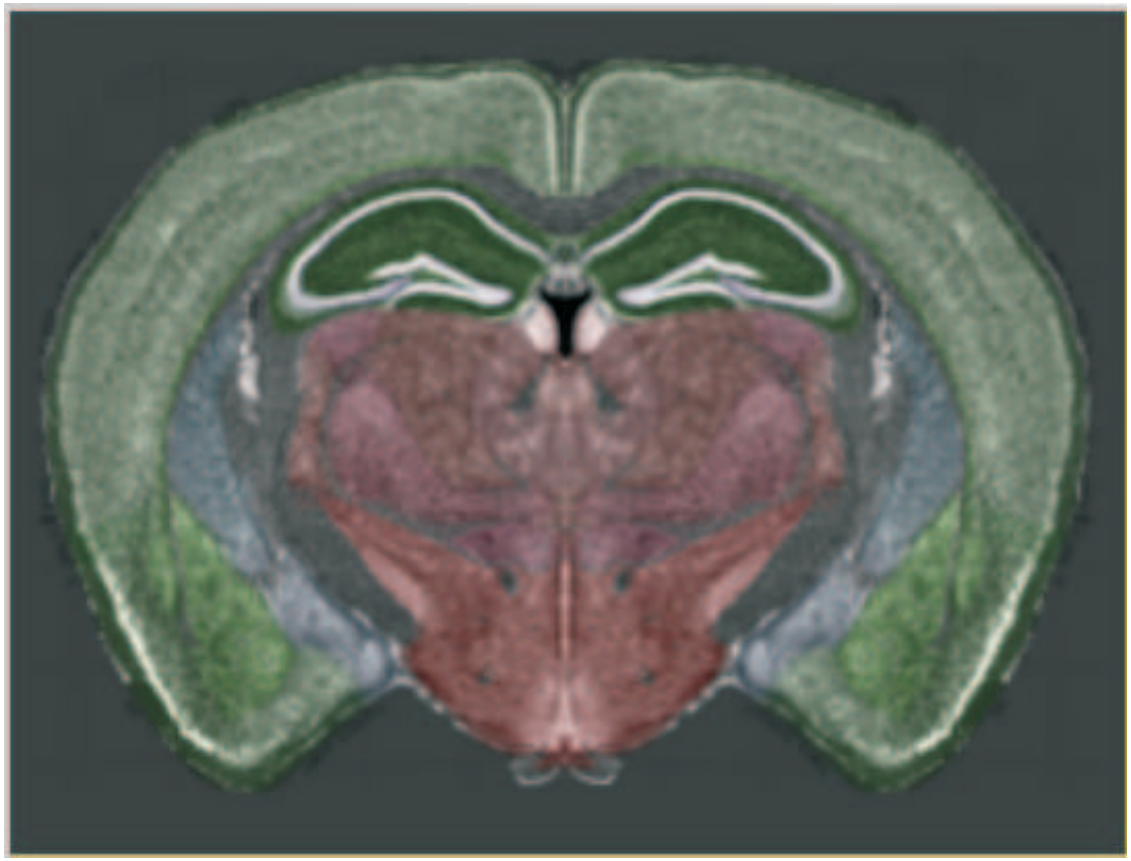
- До сих пор рассматривался случай, когда скрытые переменные бинарные
- Теперь рассмотрим ситуацию, когда скрытые переменные t_i могут принимать одно из K значений
- Физически это соответствует делению изображения на K областей
- Такие задачи возникают при построении карт диспаритетов, коллажах, семантической сегментации и пр.

Семантическая сегментация



Пример задачи семантической сегментации

Анатомическая разметка



Пример задачи автоматического выделения анатомических зон головного мозга мыши

Итерационная схема

$$E(T|X) = \sum_{(i,j)\in E} E_{ij}(t_i, t_j) + \sum_i E_i(x_i, t_i) \to \min_T, \quad t_i \in \{0, 1, \ldots, K-1\}$$

- Задача оптимизации энергии по K-значным скрытым переменным $(K > 2)$ является NP-трудной
- Тем не менее, в ряде случаев можно построить итерационную процедуру, сходящуюся к близкому к глобальному оптимуму ответу
- Наибольшее распространение получил алгоритм т.н. α-расширения

Итерационная схема

- Начинаем с произвольного начального приближения
- В цикле для каждой метки $\alpha \in \{0, \ldots, K-1\}$ заменяем часть других меток на данную, так чтобы минимизировать энергию (выполняем α-расширение)
- Если хотя бы для одной метки энергию удалось уменьшить, то переходим в предыдущему шагу, иначе выход

Для сходимости такого алгоритма необходимо, чтобы каждое парное слагаемое было метрикой в пространстве $\{0, \ldots, K-1\}$, т.е. удовлетворяло следующим условиям $\forall \alpha, \beta, \gamma \in \{0, \ldots, K-1\}$

- $E_{ij}(\alpha, \beta) = E_{ij}(\beta, \alpha)$ (симметричность)
- $E_{ij}(\alpha, \beta) = 0 \Leftrightarrow \alpha = \beta$ (аксиома тождества)
- $E_{ij}(\alpha, \gamma) \leq E_{ij}(\alpha, \beta) + E_{ij}(\beta, \gamma)$ (неравенство треугольника)

α-расширение

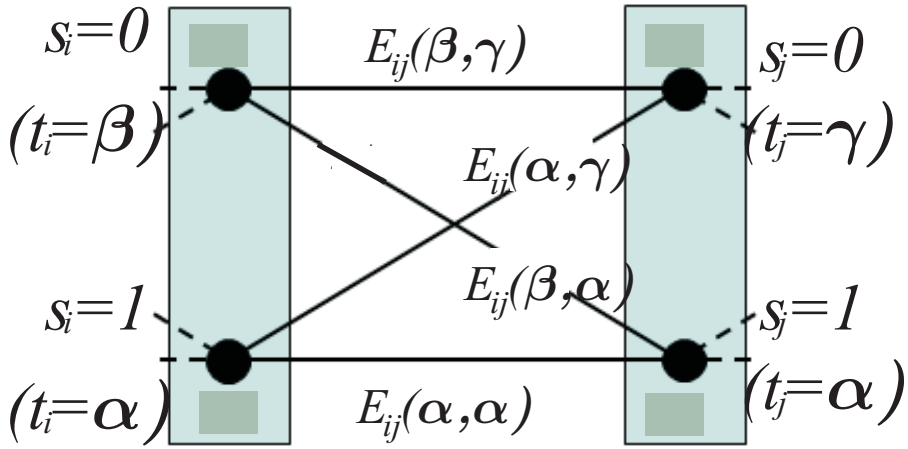
- В ходе альфа-расширения часть других меток принимает значение α, стремясь минимизировать энергию
- Введем вспомогательную марковскую решетку с бинарными переменными s_j, в которой значение 0 будет соответствовать тому, что исходные скрытые переменные t_j не изменились, а значение 1 будет означать, что исходные переменные приняли значение α

$$\forall j : t_j^{old} = \alpha \Rightarrow s_j \equiv 0$$

$$\forall j : t_j^{old} \neq \alpha \Rightarrow \begin{cases} s_j = 0 \Rightarrow t_j^{new} = t_j^{old} \\ s_j = 1 \Rightarrow t_j^{new} = \alpha \end{cases}$$

- Теперь относительно новых переменных можно построить минимальный разрез графа, минимизирующий энергию по всевозможным α-расширениям

α-расширение

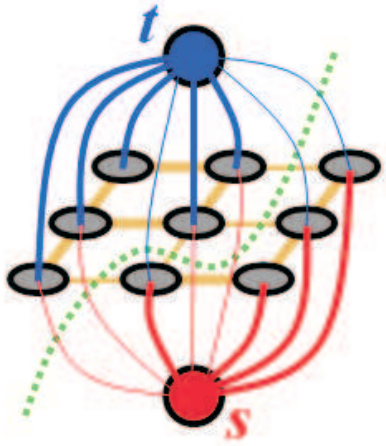


- Рассмотрим некоторую пару скрытых переменных (t_i, t_j), соединенную ребром
- Предположим, что старые значения переменных равнялись β и γ соответственно
- Для корректной репараметризации необходимо выполнение неравенства треугольника

$$E_{ij}(\beta, \gamma) \leq E_{ij}(\alpha, \gamma) + E_{ij}(\beta, \alpha)$$

- Теперь относительно новых переменных можно построить минимальный разрез графа, минимизирующий энергию по всевозможным α-расширениям

α-расширение



- Запретим изменять метки класса α

$$c(S, t_i) = E_i(x_i, \alpha), \quad c(T, t_i) = +\infty$$
$$c(t_i, t_j) = E_{ij}(\alpha, \alpha) = 0, \ \forall (i,j) \in E: \ t_i = t_j = \alpha$$

- Для остальных вершин ($t_i^{old} \neq \alpha$) определим пропускную способность следующим образом:

$$c(S, t_i) = E_i(x_i, \alpha), \quad c(T, t_i) = E_i(x_i, t_i^{old})$$
$$c(t_i, t_j) = E(\alpha, t_i^{old}), \quad c(t_j, t_i) = E(\alpha, t_j^{old}), \quad \forall (i,j) \in E$$

- Вершинам, попавшим в тот же подграф, что и исток S, будет присвоена метка α, энергия при этом уменьшится

Точность получающегося решения

- Алгоритм α-расширения является итерационным, полиномиальным, поэтому не гарантирует достижение глобального оптимума (NP-трудная задача)
- Можно показать, что значение энергии, получившейся в результате альфа-расширения, лежит в интервале

$$E(T^*) \leq E(T) \leq 2kE(T^*),$$

где T^* — оптимальное (наиболее вероятное) значение скрытых переменных, а

$$k = \frac{\max E_{ij}(\beta, \gamma)}{\min_{\beta \neq \gamma} E_{ij}(\beta, \gamma)}$$

степень контрастности парной энергии

Московский государственный университет
имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики

Математические основы теории прогнозирования

(курс лекций)

лектор — академик РАН Ю. И. Журавлев

2008

Оглавление

Курс лекций, прочитанный для 3 потока IV курса факультета ВМиК, набран в системе LaTeX студентами:
1 лекция — П. Клеменков
2 лекция — А. Гудков
3 лекция — К. Симонян
4 и 5 лекции — А. Фокин

Лекция 1

1.1 Стандартная задача распознавания

Пусть дано множество M, являющееся суммой подмножеств $K_1, \ldots, K_l$, называемых обычно классами.

$$M = \bigcup_{j=1}^{l} K_j$$

Различают случаи а) $K_u \cap K_v = \varnothing$, б) $K_u \cap K_v$, вообще говоря, не пусто. В случае а) говорят о задаче с пересекающимися, в случае б) — непересекающимися классами (множества $K_j, j = 1 \ldots l$ принято называть классами).

В дальнейшем рассматриваются только M специального вида: элементы M являются наборами длины n: $\tilde{a} \in M, \tilde{a} = (a_1, a_2, \ldots, a_i, \ldots, a_n)$. При этом для каждого номера $i, i = i \ldots n$, определено множество допустимых значений M_i, являющееся метрическим пространством с метрикой ρ_i, т.е. выполнены аксиомы: $\rho_i(c,d) \geq 0, \rho_i(c,c) = 0, \rho_i(c,d) = = \rho_i(d,c), \rho_i(c,e) + \rho_i(e,d) \geq \rho_i(c,d), c,d,e \in M_i$. В некоторых случаях выполнения последней аксиомы (аксиомы треугольника) не требуют. Тогда говорят, что в M_i введена полуметрика.

В качестве исходной информации задаются некоторые сведения о множестве M и классах $K_1, \ldots, K_l$.

В дальнейшем в качестве исходной информации рассматривается так называемая стандартная обучающая информация I: выделяется конечное множество $S_1, \ldots, S_m$ элементов из M : $S_i = (a_{i1}, a_{i2}, \ldots, a_{it}, \ldots, a_{in}), i = 1 \ldots m, a_{it} \in M_t$, для которых известно, в какие из $K_1, \ldots, K_l$ они входят. Последнее оформляется заданием информационного вектора $\tilde{\alpha}(S_i) = (\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{ij}, \ldots, \alpha_{in}), (\alpha_{ij} = 1) \rightarrow S_i \in K_j, (\alpha_{ij} = 0) \rightarrow S_i \notin K_j,\ i = 1 \ldots m,$ $j = 1 \ldots l$.

Для удобства данные об элементах S_i и их информационных векторах представляют в виде таблиц:

	1	2	…	t	…	n
S_1	a_{11}	a_{12}	…	a_{1t}	…	a_{1n}
S_2	a_{21}	a_{22}	…	a_{2t}	…	a_{2n}
…	…	…	…	…	…	…
S_i	a_{i1}	a_{i2}	…	a_{it}	…	a_{in}
…	…	…	…	…	…	…
S_m	a_{m1}	a_{m2}	…	a_{mt}	…	a_{mn}

T_1

K_1	K_2	$\dots$	K_j	$\dots$	K_l	
α_{11}	α_{12}	$\dots$	α_{1j}	$\dots$	α_{1l}	$\tilde{\alpha}(S_1)$
α_{21}	α_{22}	$\dots$	α_{2j}	$\dots$	α_{2l}	$\tilde{\alpha}(S_2)$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
α_{i1}	α_{i2}	$\dots$	α_{ij}	$\dots$	α_{il}	$\tilde{\alpha}(S_i)$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
α_{m1}	α_{m2}	$\dots$	α_{mj}	$\dots$	α_{ml}	$\tilde{\alpha}(S_m)$

T_2

Совокупность таблиц T_1, T_2 называется стандартной обучающей информацией I, таблица T_2 называется информационной матрицей.

Стандартная задача распознавания: пусть задан элемент $S \in M, S \notin \{S_1, S_2, \dots, S_i, \dots, S_m\}$. Найти алгоритм A, который, используя только I и представление S строит информационный вектор $\tilde{\alpha}(S) = (\tilde{\alpha_1}(S), \dots, \tilde{\alpha_j}(S), \dots, \tilde{\alpha_l}(S))$.

$$A(I, S) = \tilde{\alpha}(S).$$

В задачах распознавания часто обучающая информация I оказывается недостаточной для построения "правильного" вектора $\tilde{\alpha}(S)$. Поэтому допускаются и широко используются эвристические алгоритмы, допускающие ошибки и отказы при вычислении координат информационных векторов.

Такие алгоритмы $\tilde{A}(I, S)$ строят квази-информационные векторы $\tilde{\beta}(S) = (\beta_1(S), \dots, \beta_j(S), \dots, \beta_l(S))$. При этом возможно, что: $\beta_j(S) \neq \tilde{\alpha_j}(S)$ (ошибка в распознавании), $\beta_j(S) = \Delta$ — так кодируется отказ от вычисления j-й координаты информационного вектора.

В литературе описано значительное число таких эвристических алгоритмов, допускающих небольшое (допустимое при практическом применении) число ошибок и отказов при решении достаточно узких классов реальных прикладных задач. Мы опишем два таких алгоритма, получивших большое распространение при прогнозировании в геологии, медицине, технике и т.п.

В дальнейшем координаты $1, 2, \dots, n$, задающие n-мерные объекты в M, будем называть признаками.

1.2 Алгоритм "Кора" (Вайнцвайг, Бонгарт)

Применяется для M, элементами которых являются бинарные признаки: $M_i = \{0, 1\}$, $i = i \dots n$, в основном для задач с двумя непересекающимися классами: $M = K_1 \cup K_2$, $K_1 \cap K_2 = \oslash$.

В таблице $\|a_{ij}\|_{m \times n}$, задающей объекты с известной классовой принадлежностью, пусть $S_1, \dots, S_q$ принадлежат K_1, $S_{q+1}, \dots, S_m$ принадлежат K_2. Просматриваем все тройки признаков r, u, v (число таких троек, очевидно, равно $\binom{n}{3}$) и анализируем часть таблицы T_1,

составленной только из столбцов r, u, v:

$$\begin{array}{ccc}
a_{1r} & a_{1u} & a_{1v} \\
a_{2r} & a_{2u} & a_{2v} \\
\cdots & \cdots & \cdots \\
a_{ir} & a_{iu} & a_{iv} \\
\cdots & \cdots & \cdots \\
a_{qr} & a_{qu} & a_{qv} \\
\hline
a_{q+1r} & a_{q+1u} & a_{q+1v} \\
a_{q+2r} & a_{q+2u} & a_{q+2v} \\
\cdots & \cdots & \cdots \\
a_{jr} & a_{ju} & a_{jv} \\
\cdots & \cdots & \cdots \\
a_{mr} & a_{mu} & a_{mv}
\end{array}$$

Среди первых q строк выделяем и фиксируем все тройки, не совпадающие ни с одной из троек в строках $q+1, \ldots, m$. Формируем множество таких троек $\{(a_{ir}, a_{iu}, a_{iv})\}$. Аналогично выделяем все тройки (a_{jr}, a_{ju}, a_{jv}), не совпадающие ни с одной из первых q троек. Множества $\{(a_{ir}, a_{iu}, a_{iv})\}, \{(a_{jr}, a_{ju}, a_{jv})\}$ назовем, соответственно, характеристиками классов K_1, K_2. Такие характеристики формируем для всех троек (r, u, v).

Пусть задан для распознавания объект $S = (b_1 \ldots b_r \ldots b_u \ldots b_v \ldots b_n)$. Сравниваем все характеристики всех троек для K_1 с соответствующими тройками в распознаваемом объекте S. Число совпадений $(a_{ir}, a_{iu}, a_{iv}) = (b_r, b_u, b_v)$ обозначаем $\Gamma(S, K_1)$ — число голосов, поданных для S за класс K_1. Аналогично формируем величину $\Gamma(S, K_2)$: число совпадений $(a_{jr}, a_{ju}, a_{jv}) = (b_r, b_u, b_v)$. Вводим пороговый параметр ν.

Если $\Gamma(S, K_1) - \nu > \Gamma(S, K_2)$, относим S классу K_1, при $\Gamma(S, K_2) - \nu > \Gamma(S, K_1)$ — в класс K_2. В остальных случаях алгоритм отказывается от классификации. На практике часто полагают $\nu = 0$.

Пример 1

Дана таблица T_1

	1	2	3	4	5
S_1	1	0	1	0	0
S_2	0	1	0	1	0
S_3	0	0	1	0	1
S_4	1	0	0	1	0
S_5	1	0	0	0	1
S_6	0	1	0	0	1

Имеем $\binom{5}{3} = 10$ троек признаков. Перечислим характеристики для K_1 и K_2.
Характеристики для K_1:

$$\begin{array}{ll}
1.(1,2,3) : (101), (001) & 2.(1,2,4) : (011), (000) \\
3.(1,2,5) : (010), (001) & 4.(1,3,4) : (110), (001), (010) \\
5.(1,3,5) : (100), (000), (011) & 6.(1,4,5) : (100), (010) \\
7.(2,3,4) : (010), (101) & 8.(2,3,5) : (010), (100), (011) \\
9.(2,4,5) : (000), (110) & 10.(3,4,5) : (100), (101)
\end{array}$$

Характеристики для K_2

$1.(1,2,3):(100)$	$2.(1,2,4):(101),(010)$
$3.(1,2,5):(101),(011)$	$4.(1,3,4):(101),(100),(000)$
$5.(1,3,5):(100),(101),(001)$	$6.(1,4,5):(110),(101)$
$7.(2,3,4):(001),(000),(100)$	$8.(2,3,5):(000),(001),(101)$
$9.(2,4,5):(010),(101)$	$10.(3,4,5):(011)$

Очевидно, что с увеличением n (числа признаков) число троек в характеристиках растет весьма быстро. Поэтому при реальных решениях обязательно использование компьютеров.

Заметим, что для объекта $S = (00000) : \Gamma(S, K_1) = \Gamma(S, K_2) = 3$. Поэтому алгоритм не классифицирует этот объект.

При распознавании объекта $S = (10101)$ имеют место совпадения с элементами характеристики K_1: $(1,2,3),(101);(1,3,4),(110);(2,3,4),(010);(2,3,5),(011);(3,4,5),(101)$. Следовательно, $\Gamma(S, K_1) = 5$. Легко проверить, что $\Gamma(S, K_2) = 2$. Алгоритм "Кора" заносит S в класс K_1.

1.3 Тестовый алгоритм (Ю. И. Журавлев)

Пусть задана бинарная таблица $\|a_{ij}\|_{m\times n}$, строки которой $S_1, \ldots, S_m$ разделены на два класса, причем $S_1, \ldots, S_q$ — строки первого класса K_1, $S_{q+1}, \ldots, S_m$ — строки второго класса K_2.

Набор столбцов с номерами $k_1, \ldots, k_l$ образует тест, если после удаления из таблицы всех столбцов за исключением вышеозначенных, ни одна из строк из K_1 не совпадет ни с одной из строк класса K_2. Тест называется тупиковым, если при удалении из него хотя бы одного столбца, хотя бы одна из строк K_1 совпадет хотя бы с одной из строк K_2.

Предположим, что построены все тупиковые тесты $T_1, \ldots, T_r$ бинарной таблицы, $T_i = \{n_{i1}, \ldots, n_{ip(i)}\}$, $i = 1 \ldots r$, n_{uv} — номера столбцов, входящих в тест.

Распознаваемый объект $S = (a_1 \ldots a_n)$ последовательно совмещается с тупиковыми тестами. При работе с тестом T_i набор $a_{n_{i1}}, \ldots, a_{n_{ip(i)}}$ сравнивается по столбцам теста со всеми строками таблицы $\|a_{ij}\|_{m\times n}$. При этом возможно совпадение со строкой не более чем в одном из классов K_1, K_2 (это следует из определения теста). Число совпадений суммируется отдельно для классов K_1, K_2. Полученные суммы $\Gamma(S, K_1)$, $\Gamma(S, K_2)$ используются для классификации объекта S так же, как в алгоритме "Кора".

Понятия "тест" и "тупиковый тест" нетрудно распространить для таблиц, составленных из элементов произвольной природы. Необходимо только, чтобы элементы каждого столбца содержались в метрическом пространстве. Обозначим метрику этого пространства через ρ_t. Два элемента a_{it}, a_{jt} назовем различимыми, если $\rho_t(a_{it}, a_{jt}) > \epsilon_t$; в противном случае a_{it}, a_{jt} неразличимы. В определении теста достаточно заменить слова "равны" и "не равны" на "неразличимы" и "различимы". Значение ϵ_t задается из "содержательных" соображений или определяется при решении другой задачи.

Построение совокупности тупиковых тестов связано с решением системы базовых уравнений.

Пусть дана система

$$f_i(x_1 \ldots x_n) = 1, i = 1 \ldots k. \tag{1.1}$$

Система (1.1) эквивалентна одному уравнению

$$\prod_{i=1}^{k} f_i(x_1 \ldots x_n) = 1. \tag{1.2}$$

Представим f_i в виде дизъюнктивной нормальной формы (д.н.ф.)

$$f_i = \mathfrak{D}_i = \bigvee_i \mathcal{K}_{it(i)},$$

где $\mathcal{K}_{it(i)}$ — элементарные конъюнкции, т.е. произведения вида $x_{j_1}^{\sigma_1} \cdot \ldots \cdot x_{j_p}^{\sigma_p}, x^\sigma = x$ при $\sigma = 1, \bar{x}$ при $\sigma = 0$.

Выполним в (1.2) операции логического умножения и получим финальную д.н.ф.

$$\bigvee_{i=1}^{r} Q_i = 1. \qquad Q_i = x_{i_1}^{\delta_1} \cdot \ldots \cdot x_{i_p}^{\delta_p}.$$

Последовательно решаем уравнения $Q_i = 1$; $x_{i_1} = \delta_1, \ldots, x_{i_p} = \delta_p$, остальные $x_j = 0, 1$. Совокупность всех решений и есть совокупность всех решений системы (1.1).

Выведем систему уравнений для построения всех тупиковых тестов таблицы $\|a_{ij}\|_{m\times n}$, в которой строки $S_1, \ldots, S_q$ принадлежат K_1, а строки $S_{q+1}, \ldots, S_m$ — классу K_2.

Сопоставим столбцам $1, 2, \ldots, n$ булевские переменные $x_1, \ldots, x_n$. Напишем систему из $q \cdot (m - q)$ булевых уравнений. Паре $S_i = (a_{i1}, \ldots, a_{in}) \in K_1$, $S_j = (a_{j1}, \ldots, a_{jn}) \in K_2$ сопоставим уравнение

$$f_{ij} = \bigvee_{a_{it} \neq a_{jt}} x_t, \qquad i = 1, \ldots, q;\ j = q+1, \ldots, m$$

$$\prod_{\substack{i=1,\ldots,q \\ j=q+1,\ldots,m}} f_{ij} = 1.$$

Выполняем умножение и приходим к д.н.ф. $\bigvee x_{u_1} \cdot \ldots \cdot x_{u_p}$, в которой проводим все упрощения $\mathcal{K} \vee \mathcal{K} \cdot \tilde{\mathcal{K}} = \mathcal{K}$.

Финальное уравнение

$$\bigvee x_{l_1} \cdot \ldots \cdot x_{l_v}$$

определяет все тупиковые тесты $(l_1, \ldots, l_v)$.

Обоснование алгоритма см. в лекции 4.

Процесс умножения при переходе к финальному уравнению и реализация функций в классе д.н.ф. весьма трудоемки. В тестовом алгоритме последнее отсутствует, т.к. уравнения сразу задаются в виде д.н.ф. Процесс умножения можно существенно упростить, используя специфику булевой алгебры. Укажем несколько упрощающих приемов.

а) из двух уравнений $f = 1$, $f \cdot \tilde{f} = 1$. Второе можно удалить, т.к. оно является следствием первого.

б) пусть даны уравнения $f_0 \vee f_i = 1$, $i = 1 \ldots k$; тогда $\prod_{i=1}^{k}(f_0 \vee f_i) = f_0 \vee f_1 \cdot \ldots \cdot f_k$. Действительно: $f_0 \cdot f_0 = f_0$, $f_0 \vee f_0 \cdot f_i = f_0$. (правило поглощения)

в) $\mathcal{K} \cdot (\mathcal{K} \cdot \mathcal{K}') = K \cdot K'$, $\qquad Q \vee Q = Q$.

Существует большое число других упрощающих правил. Эффективность трех приведенных выше продемонстрируем на примере, рассмотренном в алгоритме “Кора”.

	1	2	3	4	5
S_1	1	0	1	0	0
S_2	0	1	0	1	0
S_3	0	0	1	0	1
S_4	1	0	0	1	0
S_5	1	0	0	0	1
S_6	0	1	0	0	1

$S_1, S_2, S_3 \in K_1$; $S_4, S_5, S_6 \in K_2$

Имеем 9 уравнений, получаемых при сравнении строк $S_i, i = 1, 2, 3$ со строками S_j, $j = 4, 5, 6$.

$$(S_1, S_4) : x_3 \vee x_4 = 1; \quad (S_1, S_5) : x_3 \vee x_5 = 1; \qquad (S_1, S_6) : x_1 \vee x_2 \vee x_3 \vee x_5 = 1$$
$$(S_2, S_4) : x_1 \vee x_2 = 1; \quad (S_2, S_6) : x_4 \vee x_5 = 1; \qquad (S_2, S_5) : x_1 \vee x_2 \vee x_4 \vee x_5 = 1$$
$$(S_3, S_5) : x_1 \vee x_3 = 1; \quad (S_3, S_6) : x_2 \vee x_3 = 1; \qquad (S_3, S_4) : x_1 \vee x_3 \vee x_4 \vee x_5 = 1$$

По правилу а) удаляются уравнения (S_1, S_6), (S_2, S_5), (S_3, S_4). Среди оставшихся выделим уравнения $x_3 \vee x_4 = 1$, $x_3 \vee x_5 = 1$, $x_1 \vee x_3 = 1$, $x_2 \vee x_3 = 1$. По правилу б) произведение левых частей даст $x_3 \vee x_1 \cdot x_2 \cdot x_4 \cdot x_5 = 1$. Перемножаем оставшиеся два уравнения

$$(x_1 \vee x_2) \cdot (x_4 \vee x_5) = x_1x_4 \vee x_1x_5 \vee x_2x_4 \vee x_2x_5$$

Перемножая левые части двух полученных уравнений и, используя в), имеем:

$$x_1x_3x_4 \vee x_1x_3x_5 \vee x_2x_3x_4 \vee x_2x_3x_5 \vee x_1x_2x_4x_5 = 1.$$

К последней д.н.ф. правило поглощения неприменимо, поэтому наборы

$$(1, 3, 4), (1, 3, 5), (2, 3, 4), (2, 3, 5), (1, 2, 4, 5) \quad \text{образуют все тупиковые тесты.}$$

Классифицируем, как в алгоритме "Кора" набор $S = (10101)$. По тесту $(1, 3, 4)$ имеем совпадение с $S_1 \in K_1$; по $(1, 3, 5)$ нет совпадений ни с одной строкой; по $(2, 3, 4)$ — совпадение с $S_1, S_3 \in K_1$; по $(2, 3, 5)$ — с $S_3 \in K_1$; по $(1, 2, 4, 5)$ — с $S_5 \in K_2$. Следовательно

$$\Gamma(S, K_1) = 3, \quad \Gamma(S, K_2) = 1.$$

Вывод: при $\nu < 2 : S \in K_1$, при $\nu \geq 2$ алгоритм откажется от распознавания.

Лекция 2

2.1 Логические алгоритмы распознавания

Для избежания громоздких выкладок и привлечения теории функций k-значной логики ограничимся задачей распознавания с двумя непересекающимися классами K_1, K_2, причем признаки будут принимать только значения 0,1.

В дальнейшем объекты исходной информации I задаются бинарными наборами $S_1, \ldots, S_r, S_{r+1}, \ldots, S_m$, где $S_i = (\alpha_{i1} \ldots \alpha_{ik} \ldots \alpha_{in})$, $i = 1 \ldots n$. Объекты S_i, $i = 1 \ldots r$ принадлежат K_1, объекты S_i, $i = r+1, \ldots, m$ — классу K_2.

Напомним некоторые сведения из теории булевых функций (функций алгебры логики).

Каждая $f(x_1, \ldots, x_n)$, вообще говоря, неоднозначно представима дизъюнктивной нормальной формой д.н.ф. $\bigvee\limits_i \mathcal{K}_i$, где $\mathcal{K}_i$ — элементарные конъюнкции. Если $\mathcal{K}_i = x_1^{\sigma_1}, \ldots, x_k^{\sigma_k}$, то k — ранг конъюнкции,

$$x^\sigma = \begin{cases} x, & \text{если } \sigma = 1; \\ \overline{x}, & \text{если } \sigma = 0. \end{cases}$$

Если $\mathcal{N}_f$ — множество единиц f, $\mathcal{N}_\mathcal{K}$ — интервал, K - множество единиц конъюнкции $\mathcal{K}$, то $f = \bigvee\limits_{i=1}^{t} \mathcal{K}_i \Leftrightarrow \mathcal{N}_{\mathcal{K}_i} = \bigcup\limits_{i=1}^{t} \mathcal{N}_{\mathcal{K}_i}$. Интервал называется максимальным, а соответствующая ему элементарная конъюнкция — простой импликантой, если не существует $\mathcal{N}_{\mathcal{K}'} : \mathcal{N}_\mathcal{K} \subset \subset \mathcal{N}_{\mathcal{K}'} \subset \mathcal{N}_f$.

Пусть $\mathcal{N}_{\mathcal{K}_i}, \ldots, \mathcal{N}_{\mathcal{K}_c}$ — совокупность всех максимальных интервалов функции f. Д.н.ф. $D_c(f)$ называют сокращенной д.н.ф. функции f. Каждая д.н.ф. минимальной сложности получается удалением из $D_c(f)$ некоторых э.к. ($D_c(f) = \bigvee\limits_{i=1}^{l} \mathcal{K}_i$).

Напомним, что сложностью д.н.ф называется сумма рангов входящих в нее э.к.

Построение минимальных д.н.ф подразделяется на следующие этапы:

I. строится произвольная д.н.ф. D_f, реализующая f

II. к D_f применяются преобразования $x_i\mathcal{K}_u \vee \overline{x}_i\mathcal{K}_v \to x_i\mathcal{K}_u \vee \overline{x}_i\mathcal{K}_v \vee \mathcal{K}_u\mathcal{K}_v$; $\mathcal{K} \vee \mathcal{K}\mathcal{K}' \to \mathcal{K}$ до тех пор, пока это возможно. Построенная д.н.ф. называется сокращенной.

III. в д.н.ф. $D_c(f) = \bigvee\limits_{i=1}^{l} \mathcal{K}_i$ выбирается произвольный интервал $\mathcal{N}_{\mathcal{K}_j}, 1 \le j \le l$, такой, что $\mathcal{N}_{K_j} \subset \bigcup\limits_{i \ne j} \mathcal{N}_{K_i}$. Э.к. $\mathcal{K}_j$ удаляются из $D_c(f)$, оставшиеся интервалы образуют покрытие $\mathcal{N}_f$; поэтому $D_c(f) \setminus \mathcal{K}_j$ реализует f. Процесс повторяется до тех пор, пока в покрытии не остаются только интервалы, не покрывающиеся суммой осталь-

ных $\mathcal{N}_{\mathcal{K}_{U_1}}, \ldots, \mathcal{N}_{\mathcal{K}_{U_p}}$. Соответствующая д.н.ф называется тупиковой для $f : D_1(f) = \bigvee_{i=1}^{p} K_{U_i}$.

Процесс удаления конъюнкций (их интервалов из покрытия) не однозначен, поэтому число тупиковых д.н.ф. может быть велико. Очевидно, что среди тупиковых содержатся и все минимальные д.н.ф.

Для дальнейшего нам понадобятся несколько утверждений и алгоритмов:

I. Как относительно нетрудоемко построить д.н.ф. приемлемой сложности, реализующую f

II. Найти аналитический критерий, позволяющий легко проверить:

$$\mathcal{N}_{\mathcal{K}} \subseteq \bigcup_{i=1}^{q} \mathcal{N}_{\mathcal{K}_i},$$

что необходимо для построения тупиковых д.н.ф.

III. Как влияют на соотношения $\mathcal{N}_{\mathcal{K}} \subseteq \bigcup \mathcal{N}_{\mathcal{K}_i}$, $\mathcal{N}_{\mathcal{K}} \not\subseteq \bigcup \mathcal{N}_{\mathcal{K}_i}$ преобразования $x_i \to x_j^{\sigma_{ij}}$, $\binom{i}{j}$ — подстановка (преобразование взаимно однозначно), $\sigma_{ij} \in \{0, 1\}$

Проверка соотношения $\mathcal{N}_{\mathcal{K}} \subseteq \bigcup_{i=1}^{l} \mathcal{N}_{\mathcal{K}_i}$. Не ограничивая общности считаем, что в $\mathcal{K}$ и $\mathcal{K}_i,\ i = 1 \ldots l$ нет переменных x_t в различных степенях. Т.е. если $x_t^{\sigma} \in \mathcal{K}$, то в $\mathcal{K}_i,\ i = 1 \ldots l$ нет сомножителей $x_t^{\overline{\sigma}}$. Действительно, если бы $x_i^{\overline{\sigma}}$ находится в $\mathcal{K}_i$, то $\mathcal{K}\mathcal{K}_i \equiv 0, (\mathcal{N}_{\mathcal{K}} \cap \mathcal{N}_{\mathcal{K}_i} = \varnothing)$ и $\mathcal{N}_{\mathcal{K}_i}$ не влиял бы на выполнимость проверяемого соотношения.

Представим $\mathcal{K}_i$ в виде $\mathcal{K}_i^1 \cdot \mathcal{K}_i^2$; в $\mathcal{K}_i^1$ входят все сомножители, общие с $\mathcal{K}$, в $\mathcal{K}_i^2$ — оставшиеся. Если оставшихся нет, полагаем $\mathcal{K}_i^2 = 1$.

Теорема 1 (Критерий поглощения) $\mathcal{N}_{\mathcal{K}} \subseteq \bigcup_{i=1}^{l} \mathcal{N}_{\mathcal{K}_i}$ *тогда и только тогда, когда* $\bigvee_{i=1}^{l} \mathcal{K}_i^2 \equiv 1$.

Доказательство. *Достаточность.* Пусть $\tilde{\alpha} \in \mathcal{N}_{\mathcal{K}} : \mathcal{K}(\tilde{\alpha}) = 1$. Но тогда очевидно $\mathcal{K}_i^1 = 1,\ i = 1 \ldots l$. Выделим в $\tilde{\alpha}$ поднабор из координат, соответствующих переменным из $\mathcal{K}_i^2,\ i = 1 \ldots l$. Так как $\bigvee_{i=1}^{l} \mathcal{K}_i^2 = 1$, то найдется $\mathcal{K}_u^2,\ 1 \le u \le l$, равное 1 на этом наборе. Но в этом случае $\mathcal{K}_u^1 \cdot \mathcal{K}_u^2(\tilde{\alpha}) = 1$, $\mathcal{K}_u(\tilde{\alpha}) = 1$. Следовательно $\tilde{\alpha} \in \mathcal{N}_{\mathcal{K}_u} \subseteq \bigcup_{i=1}^{l} \mathcal{N}_{\mathcal{K}_i}$. Достаточность доказана.

Необходимость. Пусть $\bigvee_{i=1}^{l} \mathcal{K}_i^2 \not\equiv 1$. Тогда найдется поднабор $\tilde{\beta}$ координат переменных, не входящих в $\mathcal{K}$, такой, что $\mathcal{K}_i^2(\tilde{\beta}) = 0,\ i = 1 \ldots l$.

Пусть $\mathcal{K} = X_{t_1}^{\sigma_1}, \ldots, X_{t_v}^{\sigma_v}$. Сформулируем набор $\tilde{\gamma}$, положив в нем координаты $t_1, \ldots, t_v$, равные, соответственно $\sigma_1, \ldots, \sigma_v$, добавим значения координат из набора $\tilde{\beta}$; остальные координаты зададим произвольно. Тогда $\mathcal{K}(\tilde{\gamma}) = 1, \mathcal{K}_i^1(\tilde{\gamma}) \cdot \mathcal{K}_i^2(\tilde{\gamma}) = 0,\ i = 1 \ldots l$, так $\mathcal{K}_i^2(\tilde{\gamma}) = 0,\ i = 1 \ldots l\ \ (\mathcal{K}_i^2(\tilde{\beta}) = 0$, а $\tilde{\beta}$ — часть набора $\tilde{\gamma})$. Имеем: $\tilde{\gamma} \in \mathcal{N}_{\mathcal{K}}$, $\tilde{\gamma} \overline{\in} \bigcup_{i=1}^{l} \mathcal{N}_{\mathcal{K}_i}$. Необходимость доказана.

Пусть π — преобразование $x_i \to y_j^{\sigma_{ij}}$, $\binom{i}{j}$ — подстановка, $\pi(\mathcal{K})$ — результат преобразования $\mathcal{K}$ с помощью π.

Теорема 2 $\mathcal{N}_\mathcal{K} \subseteq \bigcup\limits_{i=1}^{l} \mathcal{N}_{\mathcal{K}_i} \leftrightarrow \mathcal{N}_{\pi(\mathcal{K})} \subseteq \bigcup\limits_{i=1}^{l} \mathcal{N}_{\pi(\mathcal{K}_i)}$

Доказательство. Пусть $\mathcal{N}_\mathcal{K} \subseteq \bigcup\limits_{i=1}^{l} \mathcal{N}_{\mathcal{K}_i}$. Тогда $\bigvee\limits_{i=1}^{l} \mathcal{K}_i^2 \equiv 1$ (по т. 1). Но любая подстановка в функцию $f(z_1, \ldots, z_m),\ z_i = \varphi_i(y_{i_1}, \ldots, y_{i_{k_i}})$ приводит к $f(\varphi_1, \ldots, \varphi_m) \equiv 1$, если $f(z_1, \ldots, z_m) \equiv 1$. Проверка последнего тривиальна. Если $\mathcal{N}_\mathcal{K} \not\subseteq \bigcup\limits_{i=1}^{l} \mathcal{N}_{\mathcal{K}_i}$, то $\bigvee\limits_{i=1}^{l} \mathcal{K}_i^2 \not\equiv 1$ (т. 1), и существует набор $\tilde{\beta}$: $\mathcal{K}_i^2(\tilde{\beta}) = 0,\ i = 1 \ldots l$. Это значит, что в каждой $\mathcal{K}_i^2$ есть сомножитель x_r^σ, а r-я координата в $\tilde{\beta}$ равна $\overline{\sigma}$. Если при $\pi : x_r \to y_t$, то в новом наборе t-я координата равна $\overline{\sigma}$, а соответствующий сомножитель: y_t^σ. Очевидно, $\pi(\mathcal{K}_i^2(\tilde{\beta})) = 0$. Случай $x_r \to \overline{y}_t$ разбирается аналогично.

Сказанное выше применимо ко всем $\mathcal{K}_i^2$, имеющим сомножитель x_r^σ. Остальные $\mathcal{K}_u^2$ либо не имеют сомножителя от переменной x_r, либо имеют сомножитель $x_r^{\overline{\sigma}}$. Следовательно, в каждой из этих $\mathcal{K}_u^2$ найдется сомножитель от $x_q,\ q \neq r$, для которого проходят предыдущие выкладки. Таким образом $\pi(\bigvee\limits_{i=1}^{l} \mathcal{K}_i^2) \not\equiv 1$ и $\mathcal{N}_{\pi(\mathcal{K})} \not\subseteq \bigcup\limits_{i=1}^{l} \mathcal{N}_{\pi(\mathcal{K}_i)}$.
Переходим к реализации I. Докажем сначала:

$$(x_1 \vee \ldots \vee x_n) \cdot (\overline{x}_1 \vee \ldots \vee \overline{x}_n) =$$
$$= x_1 \cdot x_2 \vee x_2 \cdot x_3 \vee \ldots \vee x_i \cdot \overline{x}_{i+1} \vee x_{i+1} \cdot \overline{x}_{i+2} \vee \ldots \vee x_{n-1} \cdot \overline{x} \vee x_n \cdot \overline{x}_1)$$

Легко видеть, что левая часть реализует функцию, равную на наборах $(0 \ldots 0 \ldots 0)$, $(1 \ldots 1 \ldots 1)$. Действительно, на любом другом наборе либо найдется пара координат $i,\ i+1$, таких, что $\alpha_i = 1,\ \alpha_{i+1} = 0$ и тогда $x_i \cdot \overline{x}_{i+1} = 1$, либо $\alpha_n = 1,\ \alpha_1 = 0$. И тогда $x_n \cdot \overline{x}_1 = 1$ (рассматривается набор $\tilde{\alpha} = (\alpha_1 \ldots \alpha_n)$.

Заметим, что умножение двух конъюнкций $(x_1^{\sigma_1} \vee \ldots \vee x_n^{\sigma_n}) \cdot (x_1^{\delta_1} \vee \ldots \vee x_n^{\delta_n})$ — произведение реализует функцию, равную 0 только на наборах $(\overline{\sigma}_1 \ldots \overline{\sigma}_n)$, $(\overline{\delta}_1, \ldots, \overline{\delta}_n)$ — с помощью преобразования π можно привести к умножению двух конъюнкций

$$(y_1 \vee \ldots \vee y_k \vee \overline{y}_{k+1} \vee \ldots \vee \overline{y}_l \vee y_{l+1} \vee \ldots \vee y_n) \cdot$$
$$\cdot (y_1 \vee \ldots \vee y_k \vee \overline{y}_{k+1} \vee \ldots \vee \overline{y}_l \vee \overline{y}_{l+1} \vee \ldots \vee \overline{y}_n) =$$
$$= y_1 \vee \ldots \vee y_k \vee \overline{y}_{k+1} \vee \ldots \vee \overline{y}_l \vee y_{l+1} \cdot \overline{y}_{l+2} \vee \ldots \vee y_{n+1} \cdot \overline{y}_n \vee y_n \cdot \overline{y}_{l+1})$$

Таким образом, д.н.ф., реализующую функцию с двумя нулями можно построить, использовав только n конъюнкций. Оказывается, что, обобщив приведенные выше построения, можно легко получить д.н.ф. относительно невысокой сложности, если число нулей функции невелико.

Пусть таблица нулей булевой функции имеет вид

$$(\alpha_{11} \ldots \alpha_{1i} \ldots \alpha_{1n}) = \tilde{\alpha}_1$$
$$(\alpha_{21} \ldots \alpha_{2i} \ldots \alpha_{2n}) = \tilde{\alpha}_2$$
$$\ldots$$
$$(\alpha_{k1} \ldots \alpha_{ki} \ldots \alpha_{kn}) = \tilde{\alpha}_k$$

Формулы, реализующая функцию, нуля которой суть наборы $\tilde{\alpha}_1, \tilde{\alpha}_2, \cdots \tilde{\alpha}_k$ имеет вид

$$\prod_{i=1}^{k} (x_1^{\alpha_{i1}} \vee x_2^{\alpha_{i2}} \vee \ldots \vee x_n^{\alpha_{in}}) \tag{2.1}$$

Приведем произведение (2.1) к д.н.ф. Сразу исключим из таблицы нулевой и единичный столбцы, так соответствующие им переменные в (2.1) можно сразу вынести за скобки и написать $x_{u_1} \vee \ldots \vee x_{u_p} \vee \overline{x}_{t_1} \vee \ldots \vee \overline{x}_{t_v}$, соответственно, для нулевых и единичных столбцов.

Выполним преобразование $x_i \to x_j^{\sigma_{ij}},\ \sigma_{ij} \in \{0,1\},\ \binom{i}{j}$ — подстановка, таким образом, чтобы строка $\tilde{\alpha}_1$ перешла в строку $(0\ 0\ \ldots\ 0\ \ldots\ 0\ 0) = \tilde{0}$, одинаковые столбцы таблицы $\|\alpha_{ij}\|_{k\times n}$ получили последовательные номера, объединившись в блоки, число которых не может превосходить $2^{k-1}-1$ (столбцы образует $k-1$ строка и нет нулевого столбца).

Пример 2 *Исходная таблица* T_1*:*

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
0	1	0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0	1	0
1	1	1	1	1	1	0	0	0	0

Проведем сначала преобразование $x_1 \to x_1,\ x_2 \to \overline{x_2},\ x_3 \to x_3,\ x_4 \to \overline{x_4},\ x_5 \to x_5,\ x_6 \to \overline{x_6},\ x_7 \to x_7,\ x_8 \to \overline{x_8},\ x_9 \to x_9,\ x_{10} \to \overline{x_{10}}$, а затем номера переменных преобразуем подстановкой

$$\begin{pmatrix} 1 & 3 & 5 & 8 & 10 & 2 & 4 & 6 & 7 & 9 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{pmatrix}$$

Получим таблицу:

y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	0	0	0	0	0

Если рассматривать блок отдельно, то имеем функцию, принимающую значения 0 на нулевом и единичном наборах. Напишем формулу:

$$\varphi = y_1\overline{y}_2 \vee y_2\overline{y}_3 \vee y_3\overline{y}_4 \vee y_4\overline{y}_5 \vee y_5\overline{y}_1 \vee y_6\overline{y}_7 \vee y_7\overline{y}_8 \vee y_8\overline{y}_9 \vee y_9\overline{y}_{10} \vee y_{10}\overline{y}_6$$

Функция φ равно 0 на всех трех строках таблицы, но также имеются и "лишние"нули, например:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$\varphi = 0$ на любом наборе, имеющем одинаковые первые 5 и вторые 5 координат. Чтобы отбросить лишние нули, построим функцию ψ: она зависит от первых переменных каждого из блоков (в нашем случае от y_1 и y_6). Образуем таблицу нулей функции ψ из столбцов значений этих переменных. На остальных наборах $\psi = 1$ (в нашем случае это набор (0 1). Реализуем ψ с помощью д.н.ф.: $\psi = \overline{y}_1 \cdot y_6$.

y_1	y_6	ψ
0	1	0
1	1	0
1	0	0
0	1	1

Формула $\varphi \vee \psi$ реализует функцию, нули которой заданы таблицей T_1: φ равна 0 на любых наборах, которые в каждом из блоков имеют одинаковые значения координат, ψ "отбрасывает"лишние нули, оставляя те, которые реально задаются таблицей.

Теперь достаточно выполнить обратное преобразование, и мы получим формулу для функции, нули которой заданы таблицей T_1. Заметим, что для функции от 10 переменных мы получили д.н.ф. из $10+1$ конъюнкции.

В "наихудшем случае"в преобразованной таблице получится $2^{3-1}-1=3$ блока, и функция ψ примет вид:

y_1	y_{i_1}	y_{i_2}	ψ
0	0	0	1
0	1	1	0
1	0	1	0

На остальных наборах $\psi=1$. Легко написать д.н.ф. для ψ:

$$\psi=\overline{y}_1\overline{y}_{i_1}y_{i_2}\vee y_1\overline{y}_{i_2}\vee\overline{y}_{i_2}y_{i_1}\vee y_1\overline{y}_{i_2}$$

Таким образом для функций с тремя нулями д.н.ф. для ψ не может состоять более, чем из 4 элементарных конъюнкций.

В общем случае, для функций с k нулевыми наборами $\tilde{\alpha}_1,\ldots,\tilde{\alpha}_k$ процесс построения д.н.ф. полностью повторяет действия примера.

Находим преобразование, которое приводит таблицу нулей к виду: 1) первая строка становится нулевой, 2) таблица состоит из блоков одинаковых столбцов (различные блоки состоят из разных столбцов). В каждом блоке присутствуют только строки $\tilde{0}=(0\ \ldots\ 0)$ и $\tilde{1}=(1\ \ldots\ 1)$. Пусть таблица разбита на блоки $B_1,B_2,\ldots,B_i,\ldots,B_q,\ q\le 2^{k-1}-1$. Пусть блок B_i состоит из столбцов значений переменных

$$x_{t_i},x_{t_i+1},\ldots,x_{t_i+v_i}$$

Запишем формулу $\mathcal{X}(B_i)=x_{t_i}\cdot\overline{x}_{t_i+1}\vee x_{t_i+1}\cdot\overline{x}_{t_i+2}\vee\ldots\vee x_{t_i+v_i-1}\cdot\overline{x}_{t_i+v_i}\vee x_{t_i+v_i}\cdot\overline{x}_{t_i}$. Д.н.ф. $D=\bigvee\limits_{i=1}^{q}\mathcal{X}(B_i)$ реализует функцию, равную 0 на (и только на) наборах, таких, что значения координат в каждом блоке одинаковы.

Для исключения "лишних"нулей образуем (как в примере) функцию, зависящую от переменных, взятых по одному из каждого блока (например, от первых переменных блока). Соответствующие столбцы образуют множество нулей функции ψ (столбцы значений выбранных переменных). Остальные наборы образуют множество единиц функции ψ, что исключает "лишние"нули и позволяет написать для функции с нулями $\tilde{\alpha}_1,\ldots,\tilde{\alpha}_k$ формулу

$$\bigvee_{i=1}^{q}\mathcal{X}(B_i)\vee\psi$$

Написав д.н.ф. для ψ (ψ зависит не более, чем от $2^{k-1}-1$ переменной), получим искомую д.н.ф. Применив к ней преобразования

$$x_i\mathcal{K}\vee\overline{x}_i\mathcal{K}'\to x_i\mathcal{K}\vee\overline{x}_i\mathcal{K}'\vee\mathcal{K}\mathcal{K}',\ \ \mathcal{K}\vee\mathcal{K}\mathcal{K}'\to\mathcal{K}$$

построим сокращенную д.н.ф. с нулями $\tilde{\alpha}^1,\ldots,\tilde{\alpha}^k$.

Замечание. Умножение "тестовых"уравнений приводит к д.н.ф., в которой нет переменных с отрицаниями . Поэтому сокращенная д.н.ф. (совокупность всех простых импликант) получается применением только преобразования

$$\mathcal{K}\vee\mathcal{K}\mathcal{K}'\Rightarrow\mathcal{K}$$

Но $\mathcal{K}\cdot\mathcal{K}'$ не может соответствовать тупиковой тест, так как тест, соответствующих $\mathcal{K}$, получается из него удалением некоторых столбцов. Заметим также, что полученная д.н.ф. является единственной тупиковой (для доказательства применить критерий поглощения и заметить, что д.н.ф., не содержащая отрицаний переменных, не может быть равной 1 на всех наборах значений переменных).

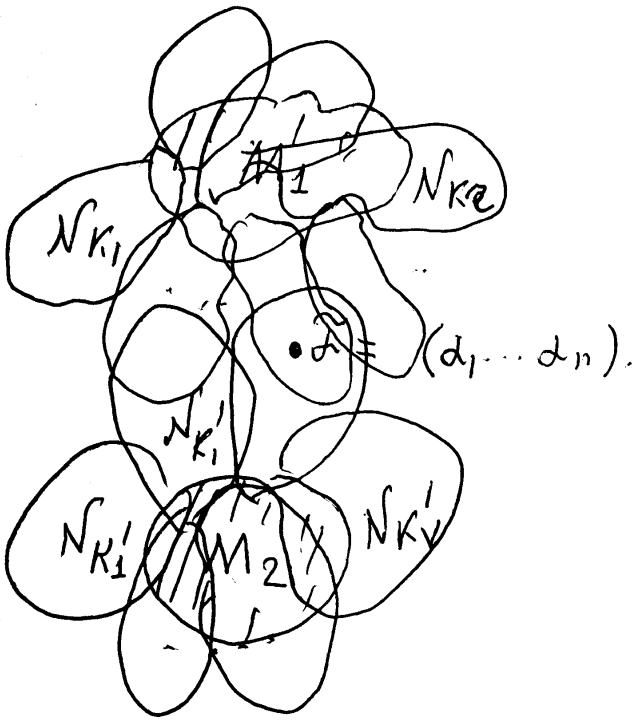
Применение к построению алгоритмов распознавания. Пусть в исходной бинарной таблице наборы $\tilde{\alpha}^i = (\alpha_{i1}, \ldots, \alpha_{in})$, $i = 1 \ldots k$ принадлежат K_1, наборы $\tilde{\beta}^j = (\beta_{j1}, \ldots, \beta_{jn})$, $j = 1 \ldots l$ принадлежат K_2, $K_1 \cap K_2 = \oslash$. Построим семейство логических алгоритмов распознавания. Введем две не всюду определенные булевские функции:

$$F_1(x_1, \ldots, x_n) = \begin{cases} 1 \text{ на наборах } \tilde{\alpha}^i,\ i = 1 \ldots k \\ 0 \text{ на наборах } \tilde{\beta}^j,\ j = 1 \ldots l \end{cases}$$

$$F_2(x_1, \ldots, x_n) = \begin{cases} 1 \text{ на наборах } \tilde{\beta}^j,\ j = 1 \ldots l \\ 0 \text{ на наборах } \tilde{\alpha}^i,\ i = 1 \ldots k \end{cases}$$

С помощью описанной выше процедуры по нулям F_1 и F_2 построим д.н.ф. для функций, равных 0 только на этих наборах, и с помощью преобразований Блейка построим сокращенные д.н.ф этих функций (на практике число добавлеяемых конъюнкций ограничивается). Из сокращенной д.н.ф. удалим все конъюнкции, интервалы которых имеют пустое пересечение с множеством единиц соответствующей функции.

Пусть после этих процедур для F_1 остались интервалы $\mathcal{N}_{\mathcal{K}_1}, \ldots, \mathcal{N}_{\mathcal{K}_r}$, имеющие непустое пересечение с $M_1 = \{\tilde{\alpha}^1 \ldots \tilde{\alpha}^k\}$, и для F_2 — $\mathcal{N}_{\mathcal{K}'_1}, \ldots, \mathcal{N}_{\mathcal{K}'_r}$ с непустым пересечением с $M_2 = \{\tilde{\beta}^1, \ldots, \tilde{\beta}^l\}$. Схематически это можно изобразить так:



Здесь $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$ — распознаваемый объект. Построим совокупность тупиковых д.н.ф $D_{11}, \ldots, D_{1p}$ для F_1 и $D_{21}, \ldots, D_{2q}$ — для F_2. При этом можно пользоваться критерием поглощения. Но можно проверить условия

$$M_1 \cap \mathcal{N}_{\mathcal{K}_i} \subseteq \bigcup_{j \neq i} \mathcal{N}_{\mathcal{K}_j} \cap M_1$$

$$M_2 \cap \mathcal{N}'_{\mathcal{K}_i} \subseteq \bigcup_{j \neq i} \mathcal{N}'_{\mathcal{K}_j} \cap M_2$$

и удалить интервалы, пересечение которых с M_1 (M_2) покрывается остающимися интервалами. Для такой проверки существуют более сложный критерий поглощения (см. Ю. И.

Журавлев, "Об отделимости подмножеств вершин единичного n-мерного куба [1]). Пусть построены тупиковые д.н.ф. $T_{11}, \ldots, T_{1K_1}$ для F_1 и $T_{21}, \ldots, T_{2K_2}$ для F_2.

Пусть число выполненных равенств $T_{1i}(\tilde{\alpha}) = 1,\ i = 1 \ldots K_1$ равно Q_1, а выполненных равенств $T_{2i}(\tilde{\alpha}) = 1, i = 1, \ldots K_2$ равно Q_2.

Простейшее решающее правило:

$$Q_1 > Q_2 \to \tilde{\alpha} \in K_1,$$
$$Q_1 < Q_2 \to \tilde{\alpha} \in K_2$$

при $Q_1 = Q_2$ алгоритм отказывается от распознавания.

Возможно усложнение введением порога c

$$Q_1 - c > Q_2 \to \tilde{\alpha} \in \mathcal{K}_1,$$
$$Q_2 - c > Q_1 \to \tilde{\alpha} \in \mathcal{K}_2$$

в остальных случаях алгоритм отказывается от распознавания.

На базе описанного выше алгоритма может быть построено параметрическое семейство — тупиковым д.н.ф. приписаны веса w, и суммируется не число вхождений $\tilde{\alpha}$, а сумма весов тупиковых д.н.ф., интервалы которых содержат точку $\tilde{\alpha}$.

Возможна оптимизация весов по текущему контролю или независимому контрольному материалу (см. подбор значений параметров алгоритмов вычисления оценок).

Лекция 3

3.1 Алгоритмы вычисления оценок

В эвристическом алгоритме "Тест" было проведено сравнение объектов из таблицы обучения и распознаваемого объекта по подмножествам признаков, образующих тупиковые тесты. Очевидно, такие сравнения могут выполняться и по другим подмножествам, например, выбираемым экспертами. При формировании величин $\Gamma(S, K_1), \Gamma(S, K_2)$ добавлялась 1 при выявлении любого совпадения. Однако в реальности признаки и объекты из таблицы обучения не равноценны, и при формировании $\Gamma(S, K_i)$ следует учитывать — с какими объектами по каким признакам произошло совпадение. Это можно делать, задавая различные веса признакам и объектам из таблицы обучения — эталонным объектам. При самом определении близости, если признаки не бинарные, можно также использовать различные возможности, определенные, например, параметрами $\epsilon_1, \ldots, \epsilon_n$. Наконец, "поощрения" при формировании $\Gamma(S, K_i)$ возможны также за отсутствие близости к объектам другого класса, а штраф — за наличие близости к объектам класса, для которого не формируется оценка и отсутствие близости к объектам, для которых оценка формируется. Указанные обстоятельства могут реализовываться многими различными способами. Формальное описание таких способов приводит к формированию класса алгоритмов, получившего название "алгоритмы вычисления оценок", или "алгоритмы голосования".

Как и ранее, будем рассматривать исходную (или обучающую) информацию, заданную в виде двух таблиц:
$\|a_{ij}\|_{m\times n}$ — совокупность m объектов, заданных наборами n признаков,
$\|\alpha_{ij}\|_{m\times l}$ — информационная матрица (таблица), где строка $(\alpha_{i1} \ldots \alpha_{ij} \ldots \alpha_{il}) = \tilde{\alpha}(S_i)$ указывает — каким из классов $K_1, \ldots, K_l$ принадлежит или не принадлежит объект S_i, определенный строкой $(a_{i1} \ldots a_{in})$ значений признаков $1, 2, \ldots, n$ из таблицы обучения. Как и ранее, полагаем, что области определения признаков — это метрические пространства M_t с метриками $\rho_t,\ t = 1, \ldots, n$.

Алгоритмы вычисления оценок определяются:

I. Заданием системы опорных множеств признаков.
Это могут быть любые множества, элементами которых являются непустые подмножества множества признаков $1, 2, \ldots, n$. Такими могут быть:

   a) совокупность всех непустых подмножеств множества $\{1, 2, \ldots, n\}$,

   b) совокупность всех подмножеств из k элементов и т. д.

   В случаях a) и b) имеем, соответственно, $2^n - 1$ и $\binom{n}{k},\ k = 1, 2, \ldots, n-1$ подмножеств признаков, по которым происходит сравнение эталонных и распознаваемого объекта.

   Вообще говоря, может быть задана любая совокупность $\{\Omega\}_A$ опорных множеств, задающих распознающий алгоритм A. Для удобства, в некоторых случаях, вместо

опорного множества $\Omega = \{u_1, \dots u_k\}$ будем рассматривать характеристический вектор $\tilde{\omega} = \{\sigma_1, \dots \sigma_n\}$, где $\sigma_{u_1} = \dots = \sigma_{u_k} = 1$, и остальные координаты равны 0. Очевидно: $\Omega \leftrightarrow \tilde{\omega}, \{\Omega\}_A \leftrightarrow \{\tilde{\omega}\}_A$.

Введем понятия $\tilde{\omega}$-части объекта S и таблицы $\|a_{ij}\|_{m\times n}$:
$\tilde{\omega}$-часть строки $S = (a_1, \dots a_n)$ — обозначение $\tilde{\omega}S$ — это набор

$$\tilde{\omega}S = (a_{u_1}, \dots, a_{u_k}),\ \tilde{\omega}(\|a_{ij}\|_{m\times n}) = \begin{pmatrix} \tilde{\omega}S_1 \\ \vdots \\ \tilde{\omega}S_m \end{pmatrix} = \begin{pmatrix} a_{1u_1} & \dots & a_{1u_k} \\ a_{2u_1} & \dots & a_{2u_k} \\ \dots & \dots & \dots \\ a_{mu_1} & \dots & a_{mu_k} \end{pmatrix}.$$

Будем также использовать обозначение $\tilde{\omega}T_1$, T_1 — таблица обучения.

II. Заданием функции близости $\mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i)$ между $\tilde{\omega}$-частями распознаваемого объекта S_i и эталонного объекта S. В дальнейшем рассматриваются только функции близости, принимающие значения $0, 1$. Тогда корректно введение функции $\overline{\mathcal{N}}(\tilde{\omega}S, \tilde{\omega}S_i)$, указывающей на отсутствие близости между $\tilde{\omega}S$ и $\tilde{\omega}S_i$.

Обычно рассматриваются три вида функций близости:

1) введем неотрицательные параметры $\epsilon_1, \dots \epsilon_n$. Пусть $\tilde{\omega}S = (a_{u_1}, \dots, a_{u_k})$, $\tilde{\omega}S_i = (a_{iu_1}, \dots, a_{iu_k})$. Тогда

$$\mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = \begin{cases} & \rho(a_{u_1}, a_{iu_1}) \le \epsilon_{u_1} \\ 1, \quad \text{если} & \dots \\ & \rho(a_{u_k}, a_{iu_k}) \le \epsilon_{u_k} \\ 0, \quad \text{если хотя бы одно из этих неравенств не выполнено.} \end{cases}$$

2) кроме параметров $\epsilon_1, \dots \epsilon_n$ введем неотрицательный целочисленный параметр ν: $\mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = 1$, если среди приведенных выше неравенств не более ν не выполнены, и равна 0 в остальных случаях.
Пусть $|\Omega|$ — число элементов в опорном множестве $\Omega \in \{\Omega\}_A$; пусть также

$$\min_{\Omega\in\{\Omega\}_A} |\Omega| = q.$$

Легко видеть, что следует рассматривать только значения ν, удовлетворяющие неравенству

$$0 \le \nu \le \left[\frac{q}{2}\right] - 1.$$

3) вместо параметра ν можно рассмотреть параметр ν^* и определить функцию близости следующим образом: $\mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = 1$ тогда и только тогда, когда из k неравенств

$$\rho(a_{u_1}, a_{iu_1}) \le \epsilon_{u_1}$$
$$\dots$$
$$\rho(a_{u_k}, a_{iu_k}) \le \epsilon_{u_k}$$

не выполнены r неравенств, и $\frac{r}{k} < \nu^*$.
В практических системах распознавания, в основном, применяется функция близости, зависящая только от параметров $\epsilon_1, \dots \epsilon_n$.

III. Признакам $1, 2, \ldots, n$, описывающим объекты, присваиваются веса $w_1, w_2, \ldots, w_n$. Как правило, $w_i \geq 0,\ i = 1 \ldots\ n$. Но последнее ограничение не является обязательным. Объектам из исходной таблицы $T_1 : S_1, \ldots, S_m$ приписываются веса $w(S_1) = = w^1, \ldots, w(S_m) = w^m$. Здесь $w^i \geq 0,\ i = 1 \ldots\ m$. Множеству $\tilde{\omega}S_i = (a_{iu_1}, \ldots, a_{iu_k})$ приписывается вес (число голосов) $\Gamma(\tilde{\omega}S_i) = w^i \cdot (w_{u_1} + \ldots + w_{u_k})$.

IV. При сравнении $\tilde{\omega}S$ и $\tilde{\omega}S_i$ возможны следующие случаи, сведенные в таблицу и оцененные параметрами $x_{ij},\ i = 0, 1,\ j = 0, 1$.

$S_i\ K_j$ \ $\mathcal{N}$	$\mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = 1$	$\overline{\mathcal{N}}(\tilde{\omega}S, \tilde{\omega}S_i) = 0$
$S_i \in K_j$	x_{11}	x_{10}
$S_i \notin K_j$	x_{01}	x_{00}

При формировании оценки принадлежности S классу $K_j,\ 1 \leq j \leq l$ (величины $\Gamma_j(\tilde{\omega}S, \tilde{\omega}S_i)$) учитываются оценка $\Gamma(\tilde{\omega}S, \tilde{\omega}S_i)$ и то, какой из четырех указанных случаев имеет место: оценка $\Gamma(\tilde{\omega}S, \tilde{\omega}S_i)$ умножается на соответствующий параметр. Так, если $S_i \in K_j,\ \mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = 1$, то

$$\Gamma_j(\tilde{\omega}S, \tilde{\omega}S_i) = x_{11} \cdot \Gamma(\tilde{\omega}S, \tilde{\omega}S_i).$$

Аналогично формируются оценки в остальных трех случаях. Заметим, что естественно полагать

$$x_{11} \geq 0, \quad x_{00} \geq 0, \quad x_{01} \leq 0, \quad x_{10} \leq 0, \quad x_{00} < x_{11}.$$

Действительно, близость к объекту из K_j или отсутствие близости с объектом, не принадлежащим K_j, благоприятны для оценки вхождения S в K_j, причем второй случай не более благоприятен, чем первый. Два других случая: $S_i \notin K_j,\ \mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = = 1$ и $S_i \in K_j,\ \overline{\mathcal{N}}(\tilde{\omega}S, \tilde{\omega}S_i) = 1$ не должны повышать оценку вхождения S в K_j.

Заметим, что приведенные здесь объяснения не являются строгими доказательствами, но лишь эвристическими правдоподобными рассуждениями, оправдывающими (в некоторой степени) даваемое нами определение класса алгоритмов вычисления оценок.

V. Оценка $\Gamma_j(S)$ вхождения объекта S в класс K_j задается следующей формулой

$$\Gamma_j(S) = \frac{1}{Q} \sum_{i=1}^{m} \sum_{\tilde{\omega} \in \{\tilde{\omega}\}_A} \Gamma_j(\tilde{\omega}S, \tilde{\omega}S_i), \tag{3.1}$$

Q — нормирующий множитель. Его величина не влияет на дальнейшие преобразования $\Gamma_j(S)$. Поэтому достаточно рассмотреть случай $Q = 1$.

VI. Решающее правило определяется числовыми параметрами $c_1, c_2,\ 0 < c_1 < c_2$.
Алгоритм A формирует для S информационный (квази-информационный) вектор $(\beta_1(S), \ldots, \beta_j(S), \ldots, \beta_l(S))$ следующим образом:
$\beta_j(S) = 1 \to S \in K_j$, если $\Gamma_j(S) > c_2$;
$\beta_j(S) = 0 \to S \notin K_j$, если $\Gamma_j(S) < c_1$;
в остальных случаях $\beta_j(S) = \Delta$, что означает: алгоритм A отказался распознавать вхождение S в класс K_j.

Отметим, что алгоритм вычисления оценок (АВО) подразделяется на две части. После выполнения этапа V формируется числовой вектор оценок $(\Gamma_1(S), \ldots, \Gamma_j(S), \ldots, \Gamma_l(S)) = = \overrightarrow{\Gamma_l}(S)$. Эту часть алгоритма принято называть распознающим оператором B:

$$B(I, S) = (\Gamma_1(S), \ldots, \Gamma_j(S), \ldots, \Gamma_l(S)) = \overrightarrow{\Gamma_l}(S).$$

Вторая часть алгоритма (этап VI) переводит $\overrightarrow{\Gamma_l}(S)$ в квази-информационный вектор. Это — решающее правило C:

$$C(\overrightarrow{\Gamma_l}(S)) = C(\Gamma_1(S), \ldots, \Gamma_j(S), \ldots, \Gamma_l(S)) = (C(\Gamma_1(S)), \ldots, C(\Gamma_j(S)), \ldots, C(\Gamma_l(S))) = \\ = (\beta_1, \ldots, \beta_j, \ldots, \beta_l).$$

Резюме. Алгоритм A определяется заданием системы $\{\Omega\}_A$ опорных множеств, параметров $\epsilon_1, \ldots \epsilon_n$ (возможно и ν или ν^*), определяющих функцию близости, весов признаков $w_1, \ldots, w_n$, весов эталонных объектов $w^1, \ldots, w^m$, параметров x_{11}, x_{00}, "поощряющих" благоприятные ситуации, x_{01}, x_{10}, "штрафующих" за неблагоприятные ситуации, параметров c_1, c_2 порогового решающего правила, с помощью которых принимается окончательное решение о вхождении, невхождении распознаваемого объекта S в классы $K_1, \ldots K_l$ или, для некоторых классов (может быть, для всех), — об отказе от распознавания.

Пример 3

	1	2	3	4	5	6	K_1	K_2
S_1	0	0	3	2	0.7	0.8	1	0
S_2	0	1	1	4	0.6	0.7	0	1
S_3	1	1	4	1	0.5	0.6	1	1

I

S	1	1	2	3	0.6	0.8

Алгоритм определяется следующими множествами и параметрами:

$$\{\Omega\}_A = \{(1,2),(3,4),(5,6)\};$$
$$\epsilon_1 = \epsilon_2 = 0, \quad \epsilon_3 = \epsilon_4 = 1, \quad \epsilon_5 = \epsilon_6 = 0.1;$$
$$w^1 = w^2 = 2, \quad w^3 = 1;$$
$$w_1 = w_2 = 1, \quad w_3 = w_4 = 2, \quad w_5 = w_6 = 3;$$
$$x_{11} = 3, \quad x_{00} = 1, \quad x_{01} = x_{10} = -1.$$

Таблица значений функции близости

	Ω_1	Ω_2	Ω_3
S_1	0	1	1
S_2	1	0	1
S_3	0	0	0

Таблица использования параметров $x_{ij},\ i = 0,1;\ j = 0,1$

x_{10}	x_{11}	x_{11}
x_{01}	x_{00}	x_{01}
x_{01}	x_{01}	x_{01}

Таблица T_Ω оценок $\Gamma_1(\tilde{\omega}S, \tilde{\omega}S_i)$

	Ω_1	Ω_2	Ω_3
S_1	4	8	12
S_2	4	8	12
S_3	2	4	6

Таблица $T_{x_{ij}}$ значений параметров x_{ij}

-1	3	3
-1	1	-1
-1	-1	-1

Умножим поэлементно матрицу T_Ω на матрицу $T_{x_{ij}}$ и сложим все элементы матрицы $T_\Omega \cdot T_{x_{ij}}$. Получим число 36. При $c_2 < 36$ алгоритм зачислит объект S в класс K_1.
Аналогично вычисляется оценка $\Gamma_2(S)$. Вычисление предоставляется выполнить читателю.

3.2 Эффективные формулы вычисления оценок

Прямое использование формулы (3.1) для вычисления оценок $\Gamma_j(S),\ j = 1, \dots l$ затруднительно при большом числе множеств. Так, если $\{\Omega\}_A$ состоит из всех непустых подмножеств множества $\{1, 2, \dots, n\}$, то потребовалось бы вычислить $m \cdot (2^n - 1)$ слагаемых. Поэтому рассмотрим пути сокращения вычислений. Рассмотрим вычисления при фиксированной строке S_i:

$$\sum_{\tilde{\omega} \in \{\tilde{\omega}\}_A} \Gamma_j(\tilde{\omega}S, \tilde{\omega}S_i).$$

Разберем два различных случая: $1^\circ \quad S_i \in K_j$ и $2^\circ \quad S_i \notin K_j$. В случае 1° имеем два подслучая $1^\circ 1$ и $1^\circ 0$: $\mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = 1$ и $\mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = 0$.

$$1^\circ 1: \quad x_{11} \cdot w^j \cdot \sum_{\substack{\Omega:\, \mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = 1 \\ \Omega \in \{\Omega\}_A}} P(\Omega), \quad P(\Omega) = w_{i_1} + \ldots + w_{i_k}, \text{ если } \Omega = \{i_1, \dots, i_k\}.$$

Обозначим через $R(\mathcal{N} = 1, t)$ число опорных множеств, содержащих t и участвующих в суммировании. Очевидно, величина w_t встретится при суммировании $R(\mathcal{N} = 1, t)$ раз, $t = 1, \dots, n$. Поэтому формулу для $1^\circ 1$ можно переписать:

$$x_{11} \cdot w^j \cdot \sum_{t=1}^{n} w_t \cdot R(\mathcal{N} = 1, t).$$

В подслучае $1^\circ 0$ вместо множителя x_{11} появится x_{10}, а величина $R(\mathcal{N} = 1, t)$ заменится на $R(\mathcal{N} = 0, t)$, где $R(\mathcal{N} = 0, t)$ — число опорных подмножеств Ω, содержащих t и таких, что $\mathcal{N}(\tilde{\omega}S, \tilde{\omega}S_i) = 0$. Окончательно при $S_i \in K_j$ получаем:

$$w^j(x_{11} \cdot \sum_{t=1}^{n} w_t \cdot R(\mathcal{N} = 1, t) + x_{10} \cdot \sum_{t=1}^{n} w_t \cdot R(\mathcal{N} = 0, t)) = \Gamma_j^1(s_i \in K_j). \tag{3.2}$$

При 2°, действуя в точности как в 1°, выводим:

$$w^j(x_{00} \cdot \sum_{t=1}^{n} w_t \cdot R(\mathcal{N} = 0, t) + x_{10} \cdot \sum_{t=1}^{n} w_t \cdot R(\mathcal{N} = 1, t)) = \Gamma_j^0(s_i \notin K_j). \tag{3.3}$$

Окончательно в формуле

$$\sum_{\Omega \in \{\Omega\}_A} \Gamma_j(\tilde{\omega}S, \tilde{\omega}S_i)$$

суммирование по опорным множествам заменяется на два суммирования n слагаемых и вычисление значений $R(\mathcal{N} = 1, t),\ R(\mathcal{N} = 0, t)$.
Нами доказана

Теорема 3

$$\Gamma_j(S) = \frac{1}{Q}\left(\sum_{S_i \in K_j} \Gamma_j^1(S_i \in K_j) + \sum_{S_i \notin K_j} \Gamma_j^0(S_i \notin \mathcal{K}_j)\right),$$

где величины $\Gamma_j^1,\ \Gamma_j^0$ *определяются по формулам* (3.2),(3.3).

Вычисление $R(\mathcal{N}=1,t),\ R(\mathcal{N}=0,t)$ для некоторых семейств опорных множеств.

1. Пусть система $\{\Omega\}_A$ состоит из всех k-элементных подмножеств множества $\{1,2,\ldots,n\}$. Функция близости определяется только параметрами $\epsilon_1,\epsilon_2,\ldots,\epsilon_n$. Пусть также $S_i=(a_{i_1},\ldots,a_{i_k},\ldots,a_{i_n})$, $S=(a_1,\ldots,a_k,\ldots,a_n)$. Выпишем неравенства

$$\begin{aligned}&\rho_1(a_1,a_{i1})\le\epsilon_1\\&\ldots\\&\rho_k(a_k,a_{ik})\le\epsilon_k\\&\ldots\\&\rho_n(a_n,a_{in})\le\epsilon_n\end{aligned}$$

   Совокупность номеров признаков, для которых выполнены или, соответственно, не выполнены неравенства, обозначим через M^+, M^-, а мощности соответствующих множеств через $|M^+|$, $|M^-|$.

   Легко видеть, что $R(\mathcal{N}=1,t)$ равно $\binom{|M^+|-1}{k-1}$ для $t\in M^+$, и равно 0 для $t\in M^-$. Так как число подмножеств, содержаших t в $\{\Omega\}_A$, равно $\binom{n-1}{k-1}$, то

$$R(\mathcal{N}=0,t)=\begin{cases}\binom{n-1}{k-1}-\binom{|M^+|-1}{k-1}, & \text{для } t\in M^+\\ \binom{n-1}{k-1}, & \text{для } t\in M^-.\end{cases}$$

   Из последнего видно, что $\sum\limits_{t=1}^{n} w_t\cdot R(\mathcal{N}=1,t)$, $\sum\limits_{t=1}^{n} w_t\cdot R(\mathcal{N}=0,t)$ в рассматриваемом случае заменится более "простыми" суммами, так как величины $R(\mathcal{N}=1,t)$, $R(\mathcal{N}=0,t)$ для каждой строки S_i принимают только два различных значения. Так,

$$\sum_{t=1}^{n} w_t\cdot R(\mathcal{N}=1,t)=\left(\sum_{t\in M^+}P_t\right)\cdot\binom{|M^+|-1}{k-1}.$$

   Аналогично упрощаются и другие введенные ранее формулы.

2. Рассмотрим ту же, что и в 1, систему опорных множеств и функцию близости с параметрами $\epsilon_1,\ldots,\epsilon_n$, ν, $\nu<[\frac{k}{2}]-1$. Аналогично предыдущему пункту, при сравнении строк S и S_i образуем подмножества признаков-координат M^+ и M^-.

   Пусть $t\in M^+$. Тогда функция близости равна 1, если она содержит $0,1,2,\ldots,\min(\nu,|M^-|)$ признаков из M^- и, соответственно, $k-1,k-2,\ldots,k-\min(\nu,|M^-|)-$ -1 признаков из M^+ (для упрощения выкладок мы рассматриваем только случай $k\ge\min(\nu,|M^-|)+1$). Тогда число опорных подмножеств, содержащих t и таких, что $\mathcal{N}(\tilde{\omega}S,\tilde{\omega}S_i)=1$, очевидно, равно

$$\binom{|M^+|-1}{k-1}\cdot\binom{|M^-|}{0}+\binom{|M^+|-1}{k-2}\cdot\binom{|M^-|}{1}+\ldots+\binom{|M^+|-r}{k-1-r}\cdot\binom{|M^-|}{r}+\ldots+$$
$$+\binom{|M^+|-1}{k-1-\nu}\cdot\binom{|M^-|}{\nu}.$$

   Все t из M^+ имеют одинаковый только что выписанный коэффициент при P_t. Остальные случаи вычисляются так же просто.

3. В качестве опорных рассматриваются все непустые подмножества множества $\{1, 2, \ldots, n\}$. Тогда

$$R(\mathcal{N}=1,t)=\begin{cases}2^{|M^+|-1}, & \text{для } t\in M^+\\ 0, & \text{для } t\in M^-.\end{cases}$$

$$R(\mathcal{N}=0,t)=\begin{cases}2^{|M^+|-1}\cdot(2^{|M^-|}-1), & \text{для } t\in M^+\\ 2^{n-1}, & \text{для } t\in M^-.\end{cases}$$

Подставляя полученные значения в (3.2), получаем

$$w^i\Bigg(x_{11}\cdot\Bigg(\sum_{t\in M^+}w_t\Bigg)\cdot 2^{|M^+|-1}+x_{10}\Bigg(\Bigg(\sum_{t\in M^+}w_t\Bigg)\cdot(2^{|M^+|-1})\cdot(2^{|M^-|}-1)+\\ +\Bigg(\sum_{t\in M^-}w_t\Bigg)\cdot 2^{n-1}\Bigg)\Bigg)$$

Аналогичный вид принимает после соответствующих подстановок формула (3.3).

4. Совокупность характеристических векторов опорных множеств образует интервал в E^n, т.е. удовлетворяет условию: $\mathcal{K}=1$, $\mathcal{K}$ — элементарная конъюнкция. Не ограничивая общности, можно полагать

$$\mathcal{K}=\overline{x_1}\cdot\ldots\cdot\overline{x_r}\cdot x_{r+1}\cdot\ldots\cdot x_{r+k}.$$

Тогда в систему опорных множеств не войдут признаки $1,\ldots,r$; в каждое из опорных множеств войдут признаки $r+1,\ldots,r+k$, и к ним последовательно присоединятся все подмножества (включая пустое, если $\mathcal{K}\neq 0$) или все непустые подмножества множества $r+k+1,\ldots,n$, если $\mathcal{K}=0$. Будем рассматривать $\mathcal{K}\neq 0$ и рассмотрим функцию близости с параметрами $\epsilon_1,\ldots,\epsilon_n$.

Если не выполнено включение $\{r+1,\ldots,r+k\}\subseteq M^+$, то $R(\mathcal{N}=1,t)=0$, $R(\mathcal{N}=0,t)=2^{n-(r+k+1)}$. Последнее следует из того, что $|\{\Omega\}_A|=2^{n-(r+k)}$, и хотя бы один из признаков любого опорного подмножества принадлежит M^-.

Пусть имеет место: $\{r+1,\ldots,r+k\}\subseteq M^+$. Тогда для $t\in\{r+1,\ldots,r+k\}$:

$$R(\mathcal{N}=1,t)=2^{|M^+|-k},$$
$$R(\mathcal{N}=0,t)=2^{|M^-|}-1.$$

Для $t\in M^+\backslash\{r+1,\ldots,r+k\}$:

$$R(\mathcal{N}=1,t)=2^{|M^+|-k-1},$$
$$R(\mathcal{N}=0,t)=2^{|M^+|-k-1}\cdot(2^{n-(k+r)}-1).$$

Наконец, для $t\in M^-$: $R(\mathcal{N}=1,t)=0$, $R(\mathcal{N}=0,t)=2^{|M^-|-1}$.

После соответствующих подстановок, получаем эффективные формулы вычисления $\Gamma_j(S),\ j=1,\ldots,l$.

Последний случай (4) можно использовать при решении прикладных задач. Рассмотрим булевскую функцию, равную 1 на элементах $\{\tilde{\omega}\}_A$. Если реализовать ее дизъюнктивной нормальной формой $\bigvee\limits_{i=1}^{r}\mathcal{K}_i$, где $\mathcal{K}_i$ — элементарные конъюнкции, и

$\mathcal{N}_{\mathcal{K}_u} \bigcap \mathcal{N}_{\mathcal{K}_v} = \varnothing$, то, написав по 4 формулы для каждой $\mathcal{K}_i$ и сложив их, получим формулу для вычисления $\Gamma_j(S),\ j = 1, \ldots, l$. Достаточно и реализации в классе д.н.ф., в которых интервалы некоторых пар конъюнкций пересекаются (интервалы конъюнкций из разных пар не пересекаются). Пусть оценка по системе опорных множеств, задаваемых конъюнкцией $\mathcal{K}$, есть $\Gamma_j(\mathcal{K}, S)$. Пусть, также, $\{\tilde{\omega}\}_A = \mathcal{K}_1 \bigcup \mathcal{K}_2,\ \mathcal{K}_1 \cdot \mathcal{K}_2 \not\equiv 0$. Тогда, очевидно, $\Gamma_j(S) = \Gamma_j(S, \mathcal{K}_1) + \Gamma_j(S, \mathcal{K}_2) - \Gamma_j(S, \mathcal{K}_1 \cdot \mathcal{K}_2)$.

Лекция 4

4.1 Вычисление характеристик, определяющих алгоритм вычисления оценок

Как было показано в лекции 3, алгоритм распознавания определяется заданием системы опорных множеств $\{\Omega\}_A$ и числовых параметров $\epsilon_1, \ldots, \epsilon_n$, $w_1, \ldots, w_n$, $w^1, \ldots, w^m$, x_{11}, x_{00}, x_{01}, x_{10}, c_1, c_2.

В своей работе алгоритм использует исходную (обучающую) информацию, состоящую из таблицы обучения $\|a_{ij}\|_{m\times n}$ и ее информационной матрицы $\|\alpha_{ij}\|_{m\times l}$. Рассматривается задача с l, вообще говоря, пересекающимися классами $K_1, \ldots, K_l$.

Параметры подбираются таким образом, чтобы обеспечить максимальную точность распознавания на определенном заранее множестве объектов.

I. Текущий контроль. Из исходной матрицы последовательно изымаются строки S_i, $i = 1 \ldots m$, вместе с информационным вектором $\tilde{\alpha}(S_i)$, и для строки S_i по оставшейся исходной информации строится квази-информационный вектор $\tilde{\beta}(S_i) = (\beta_{i1} \ldots \beta_{il})$, $i = 1 \ldots m$

В матрице $\|\alpha_{ij} \dot{-} \beta_{ij}\|_{m\times l}$ определяется число единиц. Операция $\alpha_{ij} \dot{-} \beta_{ij}$ определяется следующим образом:

α_{ij} \ β_{ij}	0	1	Δ
0	0	1	1
1	1	0	1

Алгоритм A подбирается таким образом, чтобы число единиц (т.е. сумма ошибок и отказов) была бы минимальной.

II. Независимый контроль. формируется контрольное множество $S^1, \ldots, S^q$, $S_i = (b_{i1} \ldots b_{in})$, $i = 1 \ldots q$ и таблицы информационных векторов $\|\beta_{ij}\|_{q\times l}$. С использованием алгоритма A и всей исходной информации формируется совокупность $\tilde{\gamma} = (\gamma_{i1} \ldots \gamma_{il})$ квази-информационных векторов и минимизируется число единиц в матрице $\|\beta_{ij} \dot{-} \gamma_{ij}\|_{q\times l}$.

1) Система опорных множеств определяется (задается) экспертами и последовательно рассматриваются алгоритмы в набором всех k-элементарных подмножеств: $k = 2, 3, \ldots, r$. Как правило, достаточно ограничиться $r \le 3\sqrt{\pi}$.

2) Параметры x_{ij} определяются перебором вариантов. Как правило, рассматриваются целочисленные значения $x_{11} = 1, 2, 3, 4, 5, 6, 7$; $x_{00} \le [\frac{1}{2}x_{11}]$, x_{01}, x_{10} принимают отрицательные или нулевые значения. В большинстве действующих систем полагают: $x_{11} = 1$, $x_{00} = x_{01} = x_{10} = 0$.

3) Пусть определения все характеристика за исключением $\epsilon_1, \ldots, \epsilon_n$ (о них позднее).

Рассмотрим задачу с независимым контролем и напишем систему неравенств

$$\begin{aligned} &\Gamma_j(S^i) > c_2, \forall \beta_{ij} = 1 \\ &\Gamma_j(S^i) < c_1, \forall \beta_{ij} = 0 \\ &i = 1, 2, \ldots, q, j = 1, \ldots, l \end{aligned} \tag{4.1}$$

В левой части системы неравенств (4.1) находятся билинейные формы $\Sigma w_u \cdot w^v$. Полагаем $w_1 = \ldots = w_n = 1$. Получаем систему линейных относительно $w^1 \ldots w^q$ неравенств. Находим максимальную совместную подсистему и ее решение $w_1^1, \ldots, w_1^q$. Подставляем эти значения в левую часть (4.1) и получаем линейную относительно $w_1, \ldots, w_n$ систему. Находим совместную максимальную подсистему и ее решение $w_{11}, \ldots, w_{n1}$. Подставляем эти значения левую часть (4.1) и т.д. Процесс заканчивается либо когда удастся получить наборы параметров, удовлетворяющие всем неравенствам (4.1), либо когда после очередной итерации число неравенств в совместной подсистеме уменьшится.

4) Для определения $\epsilon_1, \ldots, \epsilon_n$ существует большое число эвристических методов. Приведем один из них (может быть, не лучший). Оставим в таблице обучения и контроля только k-е столбцы:

$$\begin{pmatrix} a_{1k} \\ a_{2k} \\ \vdots \\ a_{mk} \end{pmatrix}, \begin{pmatrix} b_{1k} \\ b_{2k} \\ \vdots \\ b_{lk} \end{pmatrix}$$

Для каждой пары $(a_{ik} b_{jk})$ вычислим $\|\tilde{\alpha}(S_i) + \tilde{\beta}(S^k)_{mod2}\|$, то есть число различных координат в этом векторе.

Если $l - \|\tilde{\alpha}(S_i) + \tilde{\beta}(S^k)\| > \|\tilde{\alpha}(S_i) + \tilde{\beta}(S^k)\|$, то формируем неравенство

$$|a_{ik} - b_{jk}| < \epsilon_k$$

При изменении знака:

$$|a_{ik} - b_{jk}| > \epsilon_k.$$

При $l - \|\tilde{\alpha}(S_i) + \tilde{\beta}(S^k)\| = \|\tilde{\alpha}(S_i) + \tilde{\beta}(S^k)\|$ выписываем одно из неравенств

$$|a_{ik} - b_{jk}| \leq \epsilon_k, \qquad |a_{ik} - b_{jk}| \geq \epsilon_k.$$

Значение ϵ_k находится из условия: данное значение удовлетворяет наибольшему числу построенных неравенств.

4.2 Алгебры над алгоритмами

Ранее мы видели, что алгоритм вычисления оценок делится на две части: распознающий оператор B и решающее правило C: $A = B \cdot C$. $A(I,S) = (\Gamma_1(S), \ldots, \Gamma_l(S)) = \vec{\Gamma_l}(S)$, $C(\vec{\Gamma_l}(S)) = (C(\Gamma_1(S)), \ldots, C(\Gamma_l(S)))$

$$C(\Gamma_j(S)) = \begin{cases} 1, & \Gamma_j(S) > c_2 \\ 0, & \Gamma_j(S) < c_1 \qquad c_1 < c_2 \\ \Delta, & c_1 \le \Gamma_j(S) \le c_2 \end{cases}$$

Оказывается, что подобное представление имеет место для большого класса алгоритмов.

Пусть A — алгоритм, работающий с исходной информацией $I \in \{I\}$, $A \in \{A\}$, и каждый из A по любой $I \in \{I\}$ должен получить ответ на фиксированные вопросы $Q_1, \ldots Q_l$, причем число возможных ответов равно трем: 1, 0, Δ. Тогда

Теорема 4 *Каждый A может быть представлен в виде $A = B \cdot C$, $B(I) = (a_1 \ldots a_j \ldots a_l)$ — числовой вектор $\vec{a}$, $C(\vec{a}) = (C(a_1), \ldots, C(a_l))$, причем*

$$C(a_i) = \begin{cases} 1, & a_i > c_2 \\ 0, & a_i < c_1 \\ \Delta, & c_1 \le a_i \le c_2 \end{cases}$$

где c_1 и c_2 — константы, фиксированные для всех алгоритмов. Заметим, что $A(I) = (C(a_1), \ldots, C(a_l)) = (\delta_1 \ldots \delta_l) = \vec{\delta}$

Доказательство. Введем вспомогательный оператор $C^{-1}(\vec{\delta}) = (C^{-1}(\beta_1) \ldots C^{-1}(\beta_l)) = = (a_1, \ldots, a_l) = \tilde{a}$. Тогда $B = A \cdot C^{-1}$, $A = (A \cdot C^{-1}) \cdot C$. Теорема доказана.

Класс алгоритмов $\{A\}$ порождает класс операторов $\{B\}$, которые можно складывать, умножать, умножать на число.

Действительно, если $B_1(I) = (a_{11} \ldots a_{il})$, $B_2(I) = (a_{21} \ldots a_{2l})$, то $(B_1 + B_2)(I) = (a_{11} + + a_{21}, \ldots, a_{1l} + a_{2l})$, $B_1 \cdot B_2(I) = (a_{11} \cdot a_{21}, \ldots, a_{1l} \cdot a_{2l})$, $(d \cdot B_1)(I) = (d \cdot a_{11}, \ldots, d \cdot a_{1l})$.

Нетрудно видеть, что используя операторы из $\{B\}$, $A = B \cdot C$, можно построить полиномы

$$\tilde{B} = \Sigma c_{i_1 \ldots i_k} \cdot B_{i_1}^{r_1} \cdot \ldots \cdot B_{i_k}^{r_k},$$

где роль переменных играют операторы; $c_{i_1 \ldots i_k}$ — константы.

Совокупность $\{\tilde{B}\}$ называют алгебраическим замыканием семейства $\{B\}$, а $\{\tilde{B}\} \cdot C$ — алгебраическим замыканием класса алгоритмов $\{A\}$ — обозначения $\mathfrak{U}\{B\}$, $\mathfrak{U}\{A\}$.

Оказывается (Ю. И. Журавлев), что в $\mathfrak{U}\{A\}$ при выполнении простых легко проверяемых условий можно построить алгоритм, не делающий ошибок на контрольной совокупности.

Алгоритм имеет вид:

$$(d \cdot \Sigma(c_i B_i)^{k_i}) \cdot C.$$

B_i представимы линейными формами от распознающих операторов вычисления оценок.

Условия:

Пусть $I = \{\|a_{ij}\|_{m \times n} \|\alpha_{ij}\|_{m \times l}\}$.

Контрольный материал: $\|b_{uv}\|_{q \times n}$, $\|\beta_{ij}\|_{q \times l}$.

1. в матрице $\|\alpha_{ij}\|_{m\times l}$ нет одинаковых столбцов.

2. для каждой пары S^u, S^v контрольных объектов, $S^u = (b_{u1} \dots b_{un})$, $S^v = (b_{v1} \dots b_{vn})$ найдется $S_r \in I$, $S_r = (a_{r1} \dots a_{rl})$ и признак k, $1 \le k \le n$, $r = r(u,v)$, $k = k(u,v)$ такие, что

$$\rho_k(a_{rk}, b_{uk}) \neq \rho_k(a_{rk}, b_{vk})$$

Лекция 5

5.1 Построение алгоритмов распознавания, корректных для заданной контрольной выборки

Рассматривается задача распознавания (или прогноза) со стандартной обучающей информацией:

$$
\begin{aligned}
&I_0 = \{S_1, \ldots, S_m, \tilde{\alpha}(S_1), \ldots, \tilde{\alpha}(S_m)\},\\
&S_i = (a_{i1}, \ldots, a_{in}), \qquad i = 1, 2, \ldots, m,\\
&a_{ij} \in M_j, \qquad j = 1, 2, \ldots, n.
\end{aligned}
$$

Здесь $S_1, \ldots, S_m$ — описания объектов, составляющих обучающий материал, $\tilde{\alpha}(S_i)$, $i = 1, 2, \ldots, m$, — информационные векторы объектов S_i по свойствам $P_j \equiv S_i \in K_j$. Другими словами, если $\alpha(S_i) = (\alpha_{i1}, \ldots, \alpha_{ij}, \alpha_{il})$, $j = 1, 2, \ldots, l$, то

$$
\alpha_{ij} = \begin{cases} 1, & S_i \in K_j, \\ 0, & S_i \bar{\in} K_j. \end{cases}
$$

Задача распознавания Z определяется начальной информацией I_0 и конечной выборкой $\tilde{S}^q = (S^1, \ldots, S^q)$, $S^i = (b_{i1}, \ldots, b_{in})$, $i = 1, 2, \ldots, q$, т.е. $Z = \{I_0, \tilde{S}^q\}$.

Требуется для каждого объекта S^i из $\tilde{S}^q$ вычислить его информационный вектор $\tilde{\beta}(S^i)$ или, что то же самое, значение свойства $P_j(S^i) \equiv S^i \in K_j$, $j = 1, 2, \ldots, l$.

Далее будем считать, что информационные векторы для $\tilde{S}^q$ известны, и, основываясь на этом, строить алгоритм, который правильно вычисляет эти свойства.

I. Пусть дано множество $\{A\}$, вообще говоря, некорректных алгоритмов для решения задач распознавания, представленных в виде $A = B \cdot C$, где B — распознающий оператор, C - решающее правило. Напомним, что $B(I_0, S^q) = \|\Gamma_{ij}\|_{q \times l}$. Здесь Γ_{ij} — действительные числа, $C(\|\Gamma_{ij}\|)_{q \times l} = \|\beta_{ij}\|_{q \times l}$; $\beta_{ij} \in \{0, 1, \Delta\}$, $\beta_{ij} = \Delta$ означает, что алгоритм A отказался от вычисления свойства $P_j(S^i)$; $\beta_{ij} = \beta$, $\beta \in \{0, 1\}$, означает, что алгоритм A вычислил свойство $P_j(S^i)$ равным β. При этом мы допускаем возможность ошибки.

   Известно, что каждый алгоритм A может быть представлен в виде $B \cdot C$ и что по исходному семейству $\{A\}$ с помощью операций сложения, умножения и умножения на скаляр можно построить алгебраическое расширение

$$
\mathfrak{U}\{A\} = \mathfrak{U}\{B\} \cdot \{C\}
$$

   класса алгоритмов $\{A\}$ и алгебраические расширения конечных степеней

$$\mathfrak{U}^k\{A\} = \mathfrak{U}^k\{B\} \cdot \{C\}.$$

При выполнении некоторых условий для задачи Z и исходного семейства $\{A\}$ в расширении $\mathfrak{U}^k\{A\}$ можно построить алгоритм A^*, правильно вычисляющий все значения $P_j(S^i)$, $i = 1, 2, \ldots, q$, $j = 1, 2, \ldots, l$. Если в качестве исходного семейства $\{A\}$ рассмотреть класс алгоритмов вычисления оценок, то искомый алгоритм A^* представим в виде

$$A^* = \left[(c_1 + c_2) \sum_{i=1}^{q} \sum_{j=1}^{l} \beta_{ij} (B_{ij})^k \right] C(c_1, c_2),$$

где c_1, c_2 — параметры решающего правила C,

$$\begin{aligned}
&\beta_{ij} P_j(S^i), \qquad i = 1, 2, \ldots, q, \qquad j = 1, 2, \ldots, l, \\
&B_{ij} = B_j + B_i^j, \\
&B_j = B_{j1} + \ldots + B_{j,j-1} + B_{j,j+1} + \ldots + B_{jl}, \\
&B_i^j = B_{i1}^j + \ldots + B_{i,i-1}^j + B_{i,i+1}^j + \ldots + B_{iq}^j;
\end{aligned}$$

здесь каждый оператор B_{ij} является оператором вычисления оценок, каждый оператор B_{iv}^j является либо оператором вычисления оценок, либо разностью двух операторов вычисления оценок. Для величины k также имеется формула, приводить которую здесь нет необходимости.

Каждый оператор вычисления оценок кодируется значениями $2n + 3m + 3$ параметров, где n — число признаков, описывающих объекты, m — число объектов в I_0. Нетрудно также видеть, что приведенная выше формула для алгоритма A^* включает в себя по крайней мере $l(l-1) + lq(q-1)$ операторов вычисления оценок. Следовательно, для полной записи кода алгоритма A^* требуется по крайней мере $(2n + m + 3)l[(l-1) + q(q-1)]$ чисел. Указанная величина может быть несколько уменьшена с помощью специальных приемов, однако она все-таки остается большой, и это неудобно при машинной реализации алгоритма, если величины n, m, l, q велики. Поэтому при реальном синтезе корректного алгоритма A^* будет использоваться только его принципиальная запись, данная выше, а реализация операторов типа B_{ij} будет проводиться другими методами. В дальнейшем будут использоваться только пороговые решающие правила $C(c_1, c_2)$: $C(\|\Gamma_{ij}\|_{q \times l}) = \|C(\Gamma_{ij})\|_{q \times l}$,

$$C(\Gamma_{ij}) = \begin{cases} 1, & \Gamma_{ij} > c_2, \\ 0, & \Gamma ij < c_1, \qquad 0 < c_1 < c_2. \\ \Delta, & c_1 \le \Gamma_{ij} \le c_2 \end{cases}$$

II. Рассмотрим информационную матрицу $\|\beta_{ij}\|_{q \times l}$ выборки $\tilde{S}^q$ в задаче Z. Положим

$$\begin{aligned}
&M = \{(i,j)\}, \qquad i = 1, 2, \ldots, q, \qquad j = 1, 2, \ldots, l, \\
&M_\alpha = \{(i,j)\} : \beta_{ij} = \alpha, \qquad \alpha = 0, 1.
\end{aligned}$$

Очевидно, $M = M_0 \cup M_1$.

Пусть B — распознающий оператор и

$$B(Z) = \|\Gamma_{rt}(B)\|_{q\times l},$$

где Γ_{rt} — действительные числа.

Определение 1 *Оператор B называется допустимым для задачи Z, если существует хотя бы одна пара (u, v) из M_1 такая, что для всех (i, j) из M_0*

$$\Gamma_{uv}(B) > |\Gamma_{ij}(B)|.$$

Пара (u, v) называется в этом случае отмеченной в B. Совокупность всех пар, отмеченных в B, обозначим через $M(B)$.

Пусть

$$\Gamma^0_{max}(B) = \max_{(i,j)\in M_0} |\Gamma_{ij}(B)|,$$
$$\Gamma^1_{min}(B) = \min_{(i,j)\in M(B)} \Gamma_{ij}(B).$$

Положим

$$\Gamma(B) = [\Gamma^1_{min}(B)]^{-1}. \tag{5.1}$$

По оператору B построим оператор B':

$$B' = \Gamma(B) \cdot (B). \tag{5.2}$$

Пусть $B'(Z) = \|\Gamma'_{ij}\|_{q\times l}$. Тогда из (5.2) легко следует, что

$$\Gamma'_{ij} = \Gamma(B) \cdot \Gamma_{ij}(B). \tag{5.3}$$

Лемма 1 *Если (u, v) отмечена в B, то $\Gamma'_{uv}(B) \geq 1$; если (u, v) не отмечена в B, то*

$$\Gamma'_{uv}(B) \leq \frac{\Gamma^0_{max}(B)}{\Gamma^1_{min}(B)} = \Gamma(B) < 1.$$

Доказательство. Если (u, v) отмечена в B, то для $\Gamma_{uv}(B)$ выполнено неравенство

$$\Gamma_{uv}(B) \geq \min_{(i,j)\in M(B)} \Gamma_{ij}(B) = \Gamma^1_{min}(B).$$

Из этого неравенства и соотношений (5.1)-(5.3) легко следует первое утверждение леммы.

Если пара (u, v) не является отмеченной в B, то

$$|\Gamma_{uv}(B)| \le \max_{(i,j)\in M_0} \Gamma_{ij}(B) = \Gamma^0_{max} < \min_{(i,j)\in M(B)} \Gamma_{ij}(B) = \Gamma^1_{min}(B).$$

Из последних неравенств и соотношений (5.1)-(5.3) легко следует второе утверждение леммы.

В дальнейшем положим

$$\Gamma^0_{max}(B)/\Gamma^1_{min}(B) = Q(B).$$

Пусть $\{B\}$ — произвольная конечная система распознающих операторов.

Определение 2 *Система $\{B\}$ называется базисной для Z, если*

$$M_1 = \bigcup_{B\in\{B\}} M(B).$$

По базисной для Z системе $\{B\}$ построим алгоритм A^*, корректный для задачи Z.

Введем для операторов B', $B' = \Gamma(B)\cdot B$, $B \in \{B\}$ целые числа $k(B')$ так, чтобы выполнялось неравенство

$$[Q(B)]^{k(B')} < \frac{c_1}{(c_1+c_2)|\{B\}|}. \tag{5.4}$$

Для этого достаточно положить

$$k(B') = \frac{\ln(c_1+c_2) + \ln|\{B\}| - \ln c_1}{|\ln Q(B)|} + 1.$$

Очевидно, что в этом случае неравенство (5.4) выполнено. Напомним, что величины c_1, c_2 суть пороги решающего правила $C(c_1, c_2)$.

Теорема 5 *Алгоритм*

$$A = \left\{ (c_1+c_2) \sum_{B\in\{B\}} (B')^{k(B')} \right\} \cdot C(c_1, c_2)$$

является корректным для Z.

Доказательство. Распознающим оператором B^* в алгоритме A^* является оператор

$$(c_1+c_2) \sum_{B\in\{B\}} (B')^{k(B')}.$$

Пусть $B^*(Z) = \|\Gamma^*_{ij}\|_{q\times l}$. Тогда

$$\Gamma^*_{ij} = (c_1+c_2) \sum_{B\in\{B\}} (\Gamma'_{ij}(B))^{k(B')}.$$

Случай 1. $P_j(S^i) = \beta_{ij} = 1$, $1 \le i \le q$, $1 \le j \le l$,

$$\Gamma^*_{ij} = (c_1 + c_2) \left[\sum_{B \in B(i,j)} (\Gamma'_{ij}(B))^{k(B')} + \sum_{B \bar{\in} B(i,j)} (\Gamma'_{ij}(B))^{k(B')} \right].$$

Здесь $B(i,j)$ — совокупность всех операторов из $\{B\}$, в которых пара (i,j) является отмеченной. Так как $\{B\}$ — базисная система для Z, то множество $B(i,j)$ непусто. Поэтому

$$\sum_{B \in B(i,j)} (\Gamma'_{ij}(B))^{k(B')} \ge 1, \tag{5.5}$$

$$\left| \sum_{B \bar{\in} B(i,j)} (\Gamma'_{ij}(B))^{k(B')} \right| < |\{B\}| \frac{c_1}{(c_1 + c_2)|\{B\}|}. \tag{5.6}$$

Неравенство (5.5) следует из определения отмеченной пары, неравенство (5.6) — из (5.4). Окончательно получаем

$$\Gamma^*_{ij} > (c_1 + c_2) \left(1 - \frac{c_1}{c_1 + c_2} \right) = c_2.$$

Но тогда из определения порогового решающего правила следует $C(\Gamma^*_{ij}) = 1 = \beta_{ij} = = p_j(S^j)$.

Случай 2. $p_j(S^i) = \beta_{ij} = 0$, $0 \le i \le q$, $1 \le j \le l$. В этом случае пара (i,j) не является отмеченной ни в одном операторе B. Поэтому

$$\Gamma^*_{ij} = (c_1 + c_2) \sum_{B \bar{\in} B(i,j)} (\Gamma'_{ij}(B))^{k(B')} < (c_1 + c_2)|\{B\}| \times \frac{c_1}{(c_1 + c_2)|\{B\}|} = c_1.$$

Из определения $C(c_1, c_2)$ следует

$$C(\Gamma^*_{ij}) = 0 = \beta_{ij} = P_j(S^i).$$

Теорема доказана.

Определение 3 *Базисная система $\{B\}$ называется неприводимой для Z, если никакая собственная часть $\{B\}$ не является базисной для Z.*

Очевидно,

$$|\{B\}| \le |M_1| \le q \times l.$$

Поэтому неравенство (5.4) для неприводимых систем записывается в виде

$$Q(B)^{k(B')} < \frac{c_1}{(c_1 + c_2)|M_1|}.$$

Этому неравенству удовлетворяет

$$k(B') = \frac{\ln |M_1| + \ln (c_1 + c_2) - \ln c_1}{\ln Q(B)} + 1.$$

Формулировка теоремы 1 при новых $k(B')$ сохранится без изменений.

Из теоремы 1 следует, что для построения эффективного корректного алгоритма достаточно построить систему из небольшого числа просто выполняемых распознающих операторов B, базисную для Z. Из базисной системы затем нетрудно получить неприводимую систему. Для построения базисной системы подходит любой оператор, отмечающий непустое множество пар (i, j) таких, что $\beta_{ij} = 1$. В [X] для любой задачи $Z = \{I_0, \tilde{S}^q\}$, $I_0 = \{S_1, \ldots, S_m, \tilde{\alpha}(S_1), \ldots, \tilde{\alpha}(S_m)\}$, в которой объекты из $\tilde{S}^q$ попарно неизоморфны относительно I_0 и информационная матрица $\|\alpha_{ij}\|_{m \times l}$ в I_0 состоит из попарно различны столбцов, указана базисная система из $|M_1|$ операторов. Каждый оператор этой базисной системы гарантирует отметку, вообще говоря, ровно одной пары (i, j), $\beta_{ij} = 1$. Можно указать случаи, когда один оператор гарантирует отметку существенно большего числа пар. Это возможно и для простых распознающих операторов.

Литература

[1] Труды Мат. ин-та им. В.А.Стеклова, том 51, 1958 г.

[2] Ю.И. Журавлев. Избранные научные труды. Москва, из-во Магистр, 1996 г., стр. 378–384

[3] Ю.И. Журавлев, И.В. Исаев. Построение алгоритмов распознавания, корректных для заданной контрольной выборки. Журнал вычислительной математики и математической физики, том 19. №3, 1979 г.