

Лекция 11:

Задача классификации, линейные модели классификации, методы Байеса

Вероятностная постановка задачи классификации

- X – объекты, Y – классы, $X \times Y$ – в.п. с плотностью $p(x, y)$.
- Дано: $X^l = \{(x_i, y_i)\}_{i=1}^l \sim p(x, y)$ – простая выборка (i.i.d.).
- Найти: $a: X \rightarrow Y$ с минимальной вероятностью ошибки.

- Пусть известна совместная плотность

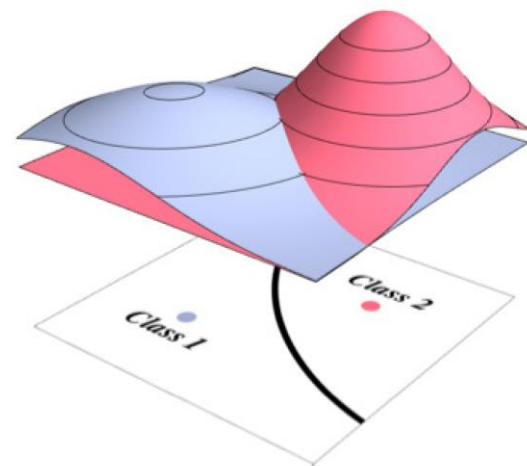
$$p(x, y) = p(x)P(y|x) = P(y)p(x|y)$$

- $P(y)$ – априорная вероятность класса y
 - $p(x|y)$ – функция правдоподобия класса y
 - $P(y|x)$ – апостериорная вероятность класса y
- По формуле Байеса:

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

- Байесовский классификатор:

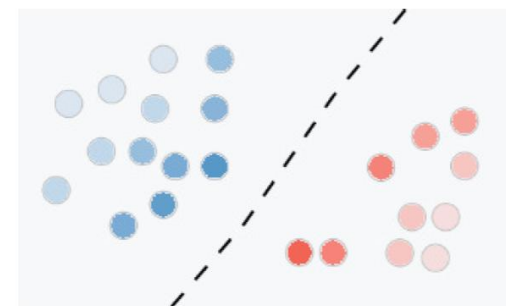
$$a(x) = \arg \max_{y \in Y} P(y|x) = \arg \max_{y \in Y} P(y)p(x|y)$$



Два подхода к обучению классификации

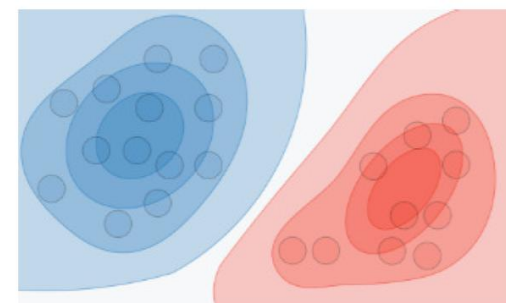
- Дискриминантный (discriminative):

- x – неслучайные векторы
- $P(y|x, w)$ – модель классификации
- Примеры: LR, GLM, SVM



- Генеративный (generative):

- $x \sim p(x|y)$ – случайные векторы
- $p(x|y, \theta)$ – модель генерации данных
- Примеры: NB, RBF, PW и др.



- Байесовские модели – генеративные:

- Моделируют **форму классов не только вдоль границы**, но и на всем пространстве, что избыточно для классификации
- Требуют больше данных для обучения
- Более устойчивы к шумовым выбросам

Оптимальный байесовский классификатор

■ Теорема:

- Пусть $P(y)$ и $p(x|y)$ известны, $\lambda_y \geq 0$ – потеря от ошибки на объекте класса $y \in Y$. Тогда минимум среднего риска

$$R(a) = \sum_{y \in Y} \lambda_y \int [a(x) \neq y] p(x, y) dx$$

- достигается **оптимальным** байесовским классификатором

$$a(x) = \arg \max_{y \in Y} \lambda_y P(y) p(x|y)$$

■ Замечание 1:

- после подстановки эмпирических оценок $\hat{P}(y)$ и $\hat{p}(x, y)$ байесовский классификатор уже **не оптимален**.

■ Замечание 2:

- задача оценивания плотности распределения – **более сложная**, чем задача классификации.

Задачи эмпирического оценивания

- Частотная оценка априорной вероятности (смещение?):

$$\hat{P}(y) = \frac{l_y}{l}, \quad l_y = |X_y|, \quad X_y = \{x_i \in X: y_i = y\}$$

- Оценки плотности $\hat{p}(x|y)$ по i.i.d. выборкам $X_y, y \in Y$:

- Параметрическая оценка плотности:

$$\hat{p}(x|y) = \varphi(x, \theta_y); \quad \theta_y = \arg \max_{\theta} \sum_{x_i \in X_y} \log \varphi(x_i, \theta)$$

- Непараметрическая оценка плотности:

$$\hat{p}(x|y) = \sum_{x_i \in X_y} \frac{1}{lV_h} K\left(\frac{\rho(x, x_i)}{h}\right)$$

- Восстановление смеси распределений:

$$\hat{p}(x|y) = \sum_{j=1}^k w_{yj} \varphi(x_i; \theta_{yj}); \quad (w_y, \theta_y) = \arg \max_{w, \theta} \sum_{x_i \in X_y} \log \hat{p}(x|y)$$

Наивный байесовский классификатор (Naïve Bayes)

- Наивное предположение: признаки $f_j: X \rightarrow D_j$ – **независимые** случайные величины с плотностями распределения

$$p_j(\xi|y), \quad y \in Y, \quad j = 1, \dots, n$$

- Тогда функции правдоподобия классов представимы в виде произведения одномерных плотностей по признакам, $x^j = f_j(x)$:

$$p(x|y) = p_1(x^1|y) \cdot \dots \cdot p_n(x^n|y), \quad x = (x^1, \dots, x^n), \quad y \in Y$$

- **Прологарифмировав** под $\arg\max$, получим классификатор

$$a(x) = \arg \max_{y \in Y} (\ln \lambda_y \hat{P}(y) + \sum_{j=1}^n \ln \hat{p}_j(x^j|y))$$

- Восстановление n одномерных плотностей – намного более простая задача, чем одной n -мерной.

Признаки с плотностями экспоненциального вида

- Предположение: одномерные плотности **экспоненциальны**:

$$p(x^j|y; \theta_{yj}, \varphi_{yj}) = \exp\left(\frac{x^j \theta_{yj} - c(\theta_{yj})}{\varphi_{yj}} + h(x^j, \varphi_{yj})\right)$$

□ где $\theta_{yj}, \varphi_{yj}$ – параметры, $c(\theta), h(x, \varphi)$ – параметры-функции.

- Задача максимизации log-правдоподобия:

$$L(\theta, \varphi) = \sum_{j=1}^n \sum_{y \in Y} \left(\sum_{x_i \in X_y} \ln p(x_i^j|y; \theta_{yj}, \varphi_{yj}) \right) \rightarrow \max_{\theta, \varphi}$$

распадается на независимые подзадачи для каждой пары (y, j) :

$$\sum_{x_i \in X_y} \left(\frac{x^j \theta_{yj} - c(\theta_{yj})}{\varphi_{yj}} + h(x^j, \varphi_{yj}) \right) \rightarrow \max_{\theta_{yj}, \varphi_{yj}}$$

- По θ_{yj} задача решается аналитически, по φ_{yj} не всегда.

Напоминание. Примеры экспоненциальных распределений

- μ – параметр математического ожидания, $\theta = g(\mu)$ – функции связи:

$$\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) = \exp\left(\frac{x\mu - \frac{1}{2}\mu^2}{\sigma^2} - \frac{x^2}{2\sigma^2} - \frac{1}{2}\ln(2\pi\sigma^2)\right)$$

$$\mu^x(1-\mu)^{1-x} = \exp\left(x \ln \frac{\mu}{1-\mu} + \ln(1-\mu)\right)$$

$$C_k^x \left(\frac{\mu}{k}\right)^x \left(1 - \frac{\mu}{k}\right)^{k-x} = \exp\left(x \ln \frac{\mu}{k-\mu} + k \ln(k-\mu) + \ln C_k^x - k \ln k\right)$$

$$\frac{1}{x!} e^{-\mu} \mu^x = \exp(x \ln(\mu) - \mu - \ln x!)$$

Распределение	Значения	$c(\theta)$	$c'(\theta)$	$[c']^{-1}(\mu)$	φ	$h(x, \varphi)$
Нормальное	\mathbb{R}	$\frac{1}{2}\theta^2$	θ	μ	σ^2	$-\frac{x^2}{2\varphi} - \frac{\ln(2\pi\varphi)}{2}$
Бернулли	$\{0,1\}$	$\ln(1 + e^\theta)$	$\frac{1}{1 + e^{-\theta}}$	$\ln \frac{\mu}{1-\mu}$	1	0
Биномиальное	$\{0, \dots, k\}$	$k \ln \frac{1 + e^\theta}{k}$	$\frac{k}{1 + e^{-\theta}}$	$\ln \frac{\mu}{k-\mu}$	1	$\ln C_k^x - k \ln k$
Пуассона	$\{0, 1, \dots\}$	e^θ	e^θ	$\ln \mu$	1	$-\ln x!$

Линейный наивный байесовский классификатор

- Решение θ_{yj} через **среднее значение признака j в классе y** :

$$\frac{\partial L}{\partial \theta_{yj}} = 0 \Rightarrow c'(\theta_{yj}) = \sum_{x_i \in X_y} \frac{x_i^j}{|X_y|} \equiv \bar{x}_{yj} \Rightarrow \theta_{yj} = [c']^{-1}(\bar{x}_{yj})$$

- Решение φ_{yj} **не всегда выражается** из уравнения $\frac{\partial L}{\partial \varphi_{yj}} = 0$, но для Пуассона, Бернулли, биномиального $\varphi_{yj} = 1$;
для гауссовского распределения (и если φ_{yj} не зависит от y):

$$\frac{\partial L}{\partial \varphi_{yj}} = 0 \Rightarrow \varphi_{yj} = \frac{1}{l} \sum_{i=1}^l (x_i^j - \bar{x}_{yij})^2$$

- В итоге Naïve Bayes оказывается **линейным** классификатором:

$$a(x) = \arg \max_{y \in Y} \left(\sum_{j=1}^n x^j \underbrace{\frac{\theta_{yj}}{\varphi_{yj}}}_{w_{yj}} + \underbrace{\ln(\lambda_y P(y)) - \sum_{j=1}^n \frac{c(\theta_{yj})}{\varphi_{yj}}}_{b_y} + \underbrace{h(x^j, \varphi_{yj})}_{\text{если от } y \text{ не зависит}} \right)$$

Пример

```
from sklearn.datasets import load_iris
from sklearn import naive_bayes
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.inspection import DecisionBoundaryDisplay
```

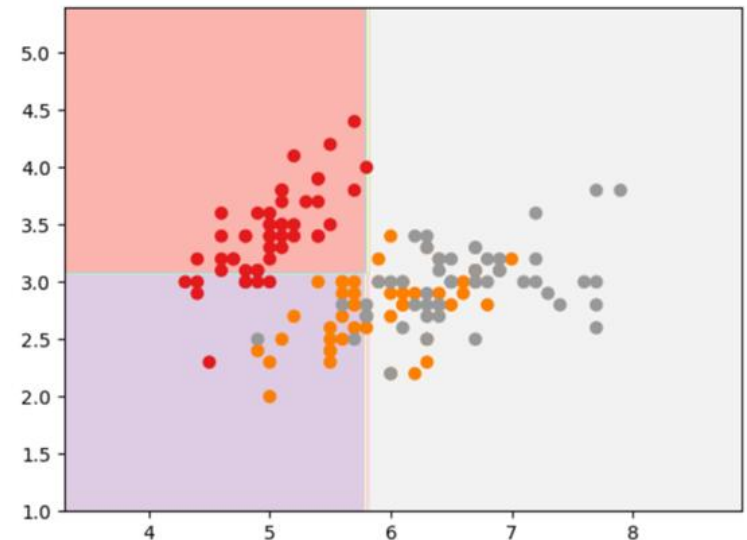
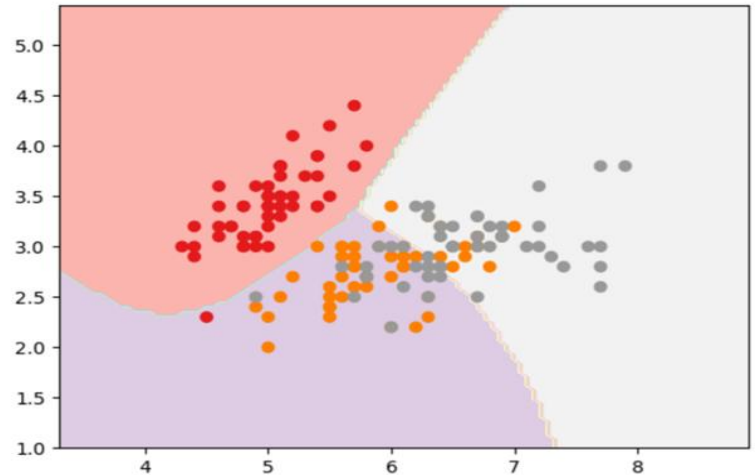
```
X, y = load_iris(return_X_y=True)
X = X[:, :2]
```

```
model = Pipeline([
    ("scaler", StandardScaler()),
    ("bayes", naive_bayes.GaussianNB())
])
```

```
model.fit(X, y)
```

```
DecisionBoundaryDisplay.from_estimator(model, X, cmap="Pastel1")
plt.scatter(*X.T, c=y, cmap="Set1")
```

```
DecisionBoundaryDisplay.from_estimator(Pipeline([
    ("1", StandardScaler()),
    ("2", naive_bayes.BernoulliNB())]).fit(X, y), X, cmap="Pastel1")
plt.scatter(*X.T, c=y, cmap="Set1")
```



Пример: построить модель прогноза ВОЗМОЖНОСТИ поиграть в ТЕННИС

outlook	
$P(\text{sunny} p) = 2/9$	$P(\text{sunny} n) = 3/5$
$P(\text{overcast} p) = 4/9$	$P(\text{overcast} n) = 0$
$P(\text{rain} p) = 3/9$	$P(\text{rain} n) = 2/5$
temperature	
$P(\text{hot} p) = 2/9$	$P(\text{hot} n) = 2/5$
$P(\text{mild} p) = 4/9$	$P(\text{mild} n) = 2/5$
$P(\text{cool} p) = 3/9$	$P(\text{cool} n) = 1/5$
humidity	
$P(\text{high} p) = 3/9$	$P(\text{high} n) = 4/5$
$P(\text{normal} p) = 6/9$	$P(\text{normal} n) = 2/5$
windy	
$P(\text{true} p) = 3/9$	$P(\text{true} n) = 3/5$
$P(\text{false} p) = 6/9$	$P(\text{false} n) = 2/5$

$$P(p) = 9/14$$

$$P(n) = 5/14$$

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

- Новый пример: <rain, hot, high, false>?
 - $P(x|p) \cdot P(p) = P(\text{rain}|p) \cdot P(\text{hot}|p) \cdot P(\text{high}|p) \cdot P(\text{false}|p) \cdot P(p) = 3/9 \cdot 2/9 \cdot 3/9 \cdot 6/9 \cdot 9/14 = 0.010582$
 - $P(x|n) \cdot P(n) = P(\text{rain}|n) \cdot P(\text{hot}|n) \cdot P(\text{high}|n) \cdot P(\text{false}|n) \cdot P(n) = 2/5 \cdot 2/5 \cdot 4/5 \cdot 2/5 \cdot 5/14 = 0.018286$
- Вывод - скорее всего не поиграть, т.к.

$$P(x|n) \cdot P(n) > P(x|p) \cdot P(p)$$

Задачи классификации (категоризации) текстов

■ Дано:

- x – текстовый документ (последовательность слов)
- $y \in Y$ – класс (тематическая категория или рубрика)
- $j \in \{1, \dots, n\}$ – слова, n – число слов в словаре
- $f_j(x_i) = x_i^j$ – частота (число вхождений) слова j в документе x_i
- $p_j(x^j|y)$ – распределение Пуассона, экспоненциального вида
- $\theta_{yj} = \ln \bar{x}_{yj}$ – оценка максимума правдоподобия, $\varphi_{yj} = 1$

■ Наивный байесовский классификатор – линейный, с весами θ_{yj} :

$$a(x) = \arg \max_{y \in Y} \left(\sum_{j=1}^n \theta_{yj} x^j + \underbrace{\ln(\lambda_y P(y))}_{b_y} - \bar{N}_y \right),$$

- $\bar{N}_y = \sum_{j=1}^n c(\theta_{yj}) = \sum_{j=1}^n \bar{x}_{yj}$ – средняя длина документов в классе y

■ Замечание: если \bar{x}_{yj} не зависит от y , то слово j не влияет на $a(x)$

Мультиномиальный наивный байесовский классификатор

- $x = (j_1, \dots, j_{N_x})$ – текстовый документ, длиной N_x слов

$$a(x) = \arg \max_{y \in Y} (\ln p(x|y) + \ln \lambda_y P(y))$$

$\pi_{yj} = p(j|y)$ – вероятность слова j в текстах класса y

$$\ln p(x|y) = \ln \prod_{t=1}^{N_x} p(j_t|y) = \sum_{j=1}^n \ln(\pi_{yj})^{x^j} = \sum_{j=1}^n x^j \ln \pi_{yj}$$

- Частотная оценка (оценка максимума правдоподобия):

$$\pi_{yj} = \frac{\#count(y, j)}{\#count(y)} = \frac{\sum_{i \in X_y} x_i^j}{\sum_{j=1}^n \sum_{i \in X_y} x_i^j} = \frac{\bar{x}_{yj}}{\sum_{j=1}^n \bar{x}_{yj}} = \frac{\bar{x}_{yj}}{\bar{N}_y}$$

- Тот же линейный NB, но с другой поправкой на длину текста:

$$a(x) = \arg \max_{y \in Y} \left(\sum_{j=1}^n x^j \ln \bar{x}_{yj} + \ln (\lambda_y P(y)) - N_x \ln \bar{N}_y \right)$$

Выводы про наивный байесовский классификатор

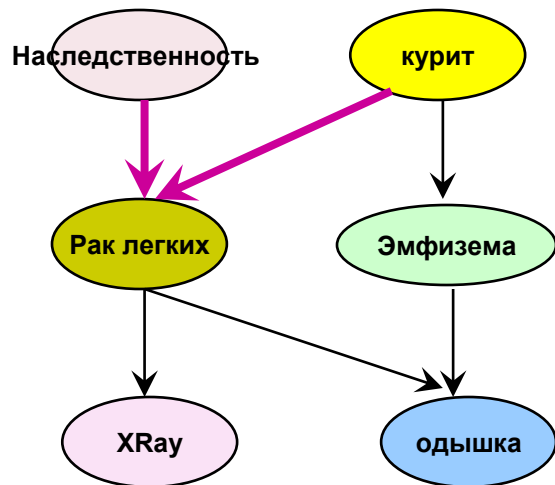
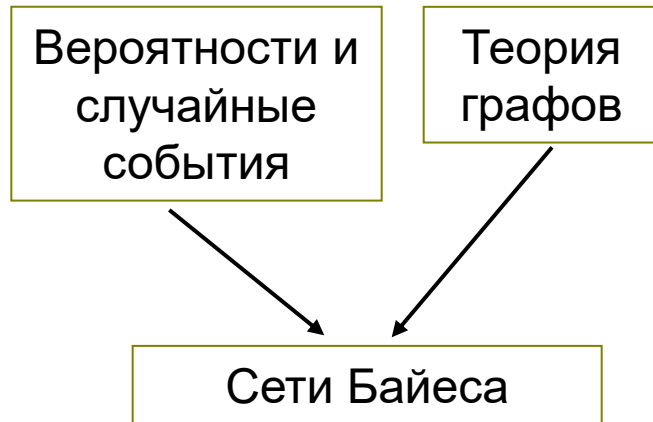
■ Достоинства:

- Очень **быстрое обучение** за $O(l n)$ – вычисление $\bar{x}_{yj}, \varphi_{yj}$
- Почти **нет переобучения**, даже на коротких выборках
- **Единообразная обработка** разнотипных признаков
- **Хорошее начальное приближение** для других методов
- **Оценка полезности** признаков: $\max_y p(y|j)$
- **Базовый уровень качества** при классификации текстов
- При классификации текстов отбор признаков по полезности **удаляет** стоп-слова, общую и нерелевантную лексику

■ Ограничения и недостатки:

- Гипотеза о **независимости** признаков
- Низкий уровень качества в большинстве приложений

Если зависимые признаки?



■ Особенности:

- ☐ Случайные переменные (входные или **скрытые**) – **вершины** ориентированного графа
- ☐ Отношения прямой зависимости – **ребра** графа
- ☐ Каждая переменная может зависеть только от **некоторого множества своих соседей**
- ☐ Плотность совместной вероятности значений всех переменных **редуцируется** до произведения условных плотностей

	(FH, S)	(FH, ~S)	(~FH, S)	(~FH, ~S)
Рак	0.8	0.5	0.7	0.1
~Рак	0.2	0.5	0.3	0.9

Байесовская сеть

■ Сеть Байеса:

- Состоит из множества случайных переменных и направленных связей между переменными;
- Каждая переменная (входная или скрытая) может принимать одно из конечного множества взаимоисключающих значений;
- Переменные вместе со связями образуют ориентированный граф без циклов;
- К каждой переменной-потомку A с переменными-предками B_1, \dots, B_n приписывается таблица условных вероятностей $P(A | B_1, \dots, B_n)$
- Переменные, не имеющие предков, описываются безусловными вероятностями, а их потомки на графе – условными

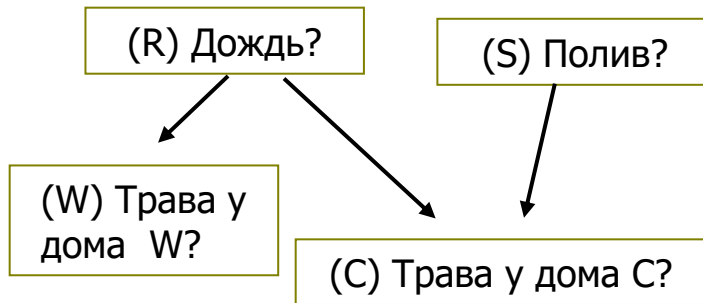
■ Редукция совместной вероятности (цепное правило полной вероятности):

$$P(A_1, \dots, A_n) = \prod_j P(A_j | pa(A_j)),$$

где $pa(A_j)$ - состояния всех переменных – предков для переменной A_j .

Пример

Граф Байесовской сети



Таблицы вероятностей

R	P(R)	S	P(S)	R	P(W=t R)	P(W=f R)
t	0.3	t	0.2	t	0.8	0.2
f	0.7	f	0.8	f	0.1	0.9

R	S	P(C=t R,S)	P(C=f R,S)
t	t	0.9	0.1
t	f	0.8	0.2
f	t	0.7	0.3
f	f	0.1	0.9

Пример редукции вероятности (с учетом независимости):

$$\begin{array}{c}
 4D \\
 \Downarrow \\
 2D
 \end{array}
 \quad
 P(R, S, C, W) = P(R) \cdot P(S|R) \cdot P(C|R, S) \cdot P(W|R, S, C)$$

$$P(R, S, C, W) = P(R) \cdot P(S) \cdot P(C|R, S) \cdot P(W|R)$$

$$\begin{aligned}
 P(C=t) &= \sum_{R=\{t,f\}, S=\{t,f\}} P(R) \cdot P(S) \cdot P(C=t|R, S) = \\
 &= 0.3 \cdot 0.2 \cdot 0.9 + 0.3 \cdot 0.8 \cdot 0.8 + 0.7 \cdot 0.2 \cdot 0.7 + 0.7 \cdot 0.8 \cdot 0.1 = 0.4
 \end{aligned}$$

$$P(W=t) = \sum_{R=\{t,f\}} P(R) \cdot P(W=t|R) = 0.31$$

Точные и приближенные вычисления вероятностей

- Применение уже обученной сети (расчет вероятностей):
 - Сложность точных вычислений вероятностей растет комбинаторно
 - На практике широко используются приближенные алгоритмы: метод Монте-Карло, Expectation-Maximization, Belief propagation
- «Обучение» - расчет вероятностей при заданной структуре сети:
 - Множество обучающих примеров $\{x_i\}_{i=1}^l = \{x_i^1, \dots, x_i^J\}_{i=1}^l$, каждый элемент множества – вектор значений для всех переменных x^j (j-й признак), x_i^j - j-й признак i-го примера.
 - Классическая схема – поиск максимума правдоподобия:
$$L = \frac{1}{nl} \sum_{j=1}^n \sum_{i=1}^l \log \left(P(x^j | pa(x^j), x_i) \right)$$
 - Условные вероятности переменных сети могут обучаться на отдельных наборах, учитывающих значения только тех переменных, которые влияют на данную, а условные вероятности $pa(x^j)$ могут быть представлены аппроксимациями плотности вероятности.

Синтез Байесовой сети на основе выборок

■ Для построения «вручную» Байесовой сети необходимо:

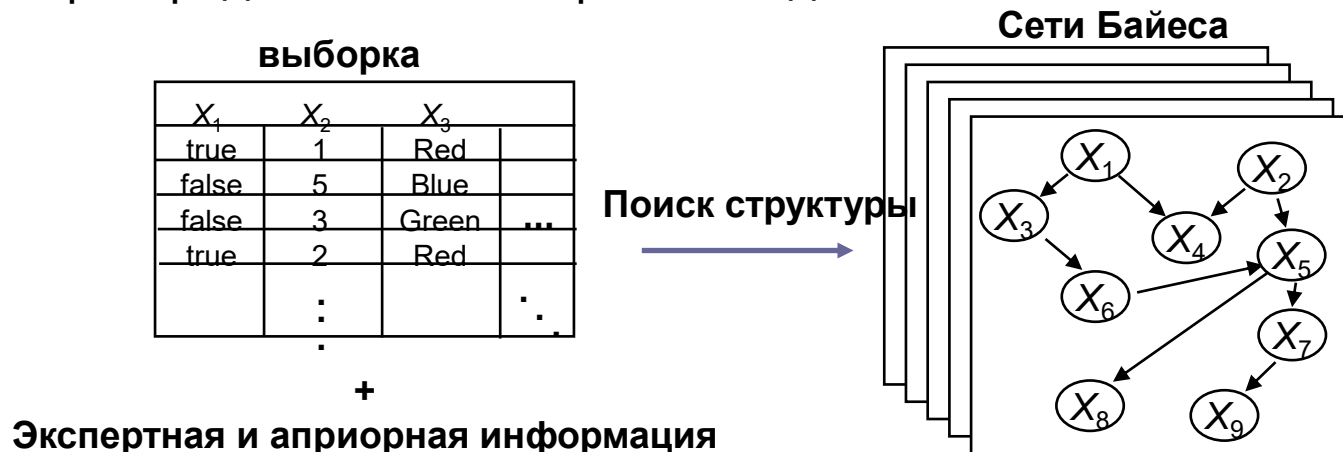
1. Сформулировать проблему в терминах вероятностей значений целевых переменных;
2. Выбрать понятийное пространство задачи, определить переменные, имеющие отношение к целевым переменным, описать возможные значения этих переменных;
3. Выбрать на основе опыта и имеющейся информации априорные вероятности значений переменных;
4. Описать отношения "причина-следствие" (как косвенные, так и прямые) в виде ориентированных ребер графа, разместив в вершинах переменные;
5. Для каждой вершины графа, имеющего входные ребра, указать оценки вероятностей различных значений переменной для комбинаций значений переменных-предков на графе.

■ Машинное обучение сетей Байеса:

- ☐ Шаги 2-5 желательно автоматизировать
- ☐ Две задачи: поиск структуры сети и расчет условных вероятностей

Методы синтеза структуры

- Задача NP-трудная – есть два приближенных подхода (часто используют их комбинацию)
- На основе ограничений:
 - Поиск сети Байеса, где набор ограничений на комбинации независимых атрибутов «соответствует» таким комбинациям в эмпирических данных
- Скоринговые методы (информационные критерии типа BIC):
 - Поиск сети Байеса, наилучшим образом приближающей распределение в эмпирических данных



Методы синтеза сети

- Основные свойства :

- Ограничивается пространство поиска с точностью до классов эквивалентности сети (графа)
- Оценка качества, например, $BIC = -2\log lik + \log(l) \cdot \dim(edges)$
- «Жадный» поиск наилучшего класса эквивалентности

- **Скоринговый** подход «от простого к сложному»:

Инициализация **пустым графом**

Стадия 1: Последовательно добавлять по 1 дуге, приводящей к максимальному улучшению оценки пока улучшается BIC

Стадия 2: Последовательно удалять по 1 дуге, приводящей к максимальному улучшению оценки пока улучшается BIC

- **Подход на основе ограничений** «от сложного к простому»:

Инициализация: **полносвязный неориентированный граф**

Шаг 1: Рекурсивно проверка условной независимости каждой дуги и удаление дуг (статистическим тестом, например, по χ^2)

Шаг 2: Получение ориентированного графа-для каждой дуги (A, B) статистический тест $A \Rightarrow B$ или $B \Rightarrow A$

Метод парзеновского окна тоже применим в «не наивном» случае (Parzen Window, PW)

- Непараметрическая оценка плотности Парзена-Розенблатта с функцией расстояния $\rho(x, x')$, для каждого класса $y \in Y$:

$$\hat{\rho}_h(x|y) = \frac{1}{l_y V_h} \sum_{x_i \in X_y} K\left(\frac{\rho(x, x_i)}{h}\right),$$

- Метод окна Парзена – это метрический классификатор:

$$a(x) = \arg \max_{y \in Y} \lambda_y \frac{P(y)}{l_y} \sum_{x_i \in X_y} K\left(\frac{\rho(x, x_i)}{h}\right)$$

- Замечание 1:

- нормирующий множитель $V_h = \int K\left(\frac{\rho(x, x_i)}{h}\right) dx$ сокращается под $\arg \max$, если он не зависит от x_i и y_i .

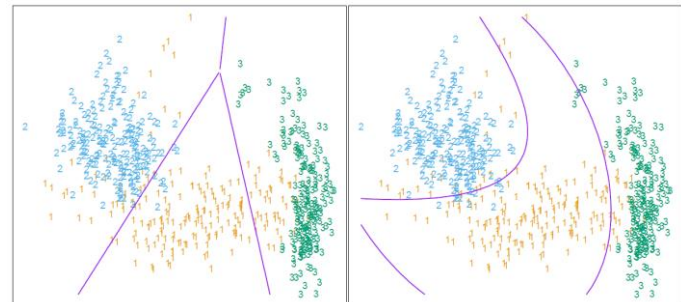
- Замечание 2 (напоминание):

- имеем проблемы выбора ядра $K(r)$, ширины окна h , функции расстояния $\rho(x, x')$.

Дискриминантный анализ

- Моделируем:
 - распределение в каждом из классов по отдельности
- Если используем **нормальные** распределения, это приводит к:
 - **линейному** (если считаем ковариационную матрицу одинаковой для всех классов) дискриминантному анализу
 - **квадратичному** (если матрицы разные) дискриминантному анализу
- Дискриминантная функция:
 - получаем логарифмируя оценку условной вероятности отклика и отбрасывая слагаемые, не зависящие от класса
 - для нескольких классов **softmax**

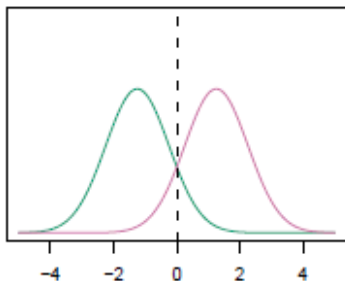
$$P(y = k|x) = \frac{e^{g_k(x)}}{\sum_{j=1}^K e^{g_j(x)}}$$



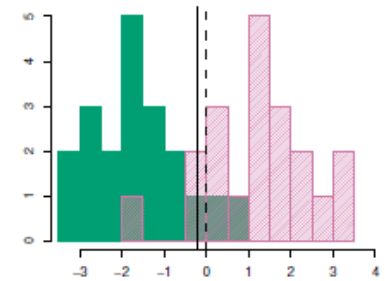
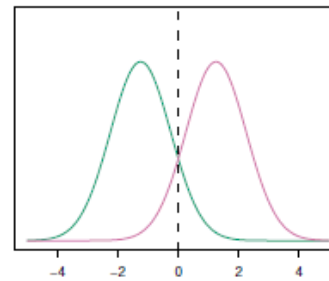
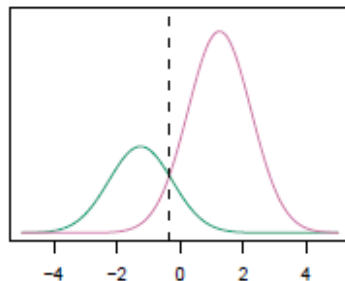
Дискриминантный анализ

- Используем *теорему Байеса*:
 - для получения **условных вероятностей откликов**
 - когда априорные вероятности различны, учитываем их
- Для классификации:
 - используем **правило Байеса** для апостериорных вероятностей
 - или для максимума дискриминантной функции
- Параметры распределений:
 - оцениваем по выборке

$\pi_1 = .5, \pi_2 = .5$



$\pi_1 = .3, \pi_2 = .7$



Линейный дискриминант Фишера (Fisher Linear Discriminant)

- Проблема: для малочисленных классов возможно $\det \hat{\Sigma}_y = 0$
- Пусть ковариационные матрицы классов равны: $\Sigma_y = \Sigma, y \in Y$
- Оценка максимума правдоподобия для Σ :

$$\hat{\Sigma} = \frac{1}{l} \sum_{i=1}^l (x_i - \hat{\mu}_{y_i})(x_i - \hat{\mu}_{y_i})^T$$

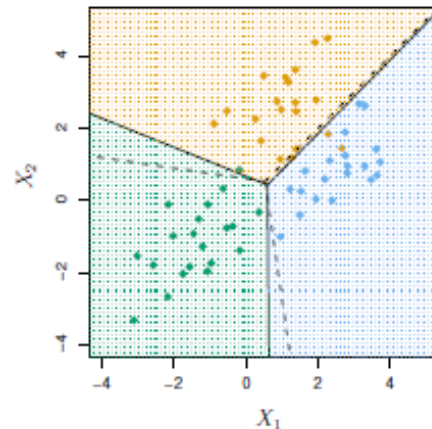
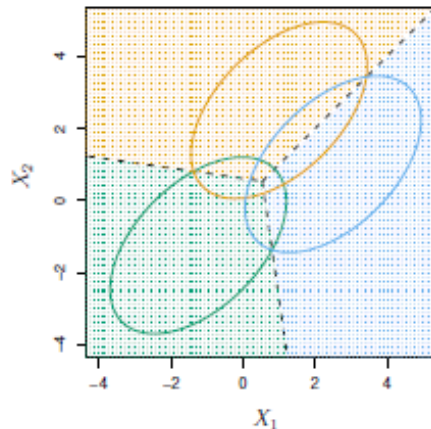
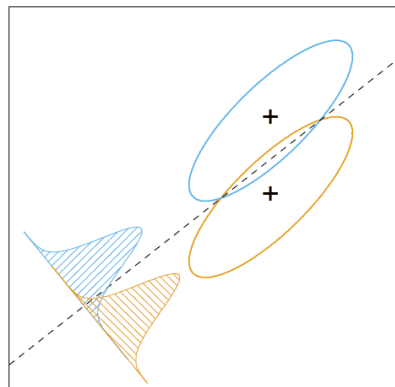
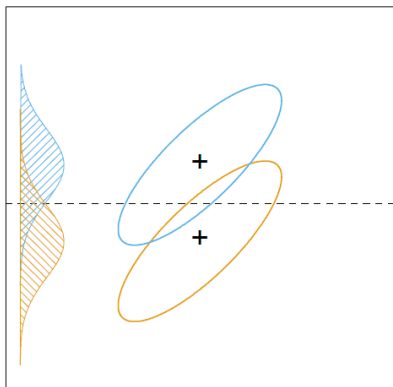
- Линейный дискриминант – подстановочный алгоритм:

$$\begin{aligned} a(x) &= \arg \max_{y \in Y} \lambda_y \hat{P}(y) \hat{p}(x|y) = \\ &= \arg \max_{y \in Y} \left(\underbrace{\ln(\lambda_y \hat{P}(y)) - \frac{1}{2} \hat{\mu}_y^T \hat{\Sigma}^{-1} \hat{\mu}_y}_{\beta_y} + x^T \underbrace{\hat{\Sigma}^{-1} \hat{\mu}_y}_{\alpha_y} \right) \\ a(x) &= \arg \max_{y \in Y} (x^T \alpha_y + \beta_y) \end{aligned}$$

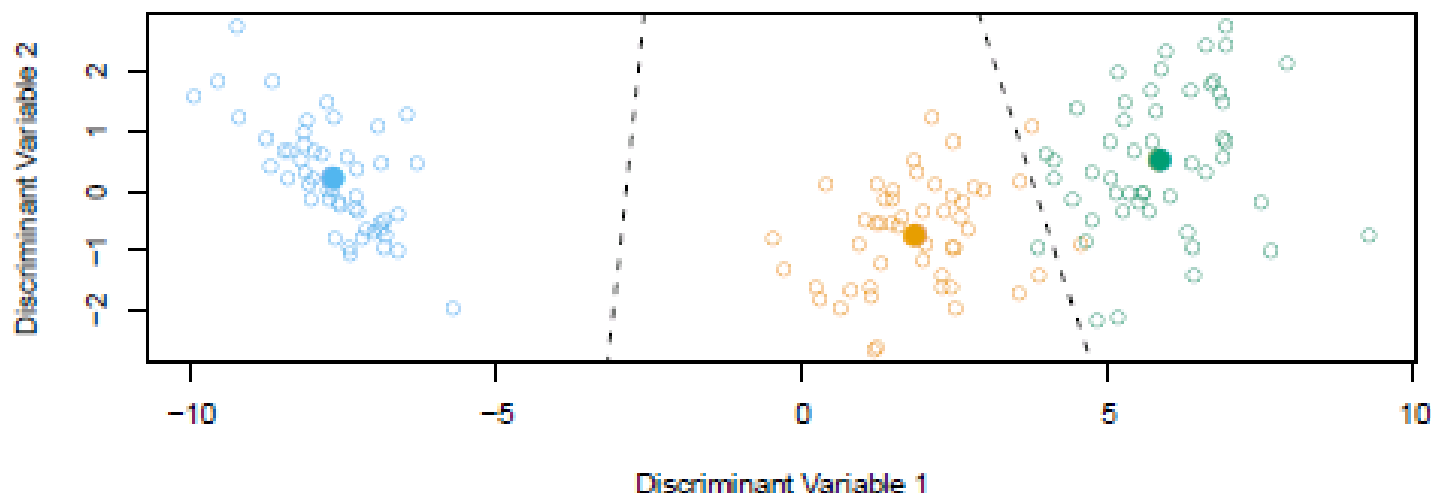
- В случае мультиколлинеарности – обращать матрицу $\hat{\Sigma} + \tau I_n$.

Геометрическая интерпретация линейного дискриминанта

- В одномерной проекции на направляющий вектор разделяющей гиперплоскости классы разделяются наилучшим образом, то есть с минимальной вероятностью ошибки.
- Пунктирные линии - границы решений Байеса. Если бы они были известны, они бы дали наименьшее число ошибочных классификаций среди всех возможных классификаторов.



Дискриминантная диаграмма



- Как в PCA можно найти главные компоненты (они независимы) по направлениям, наилучшим образом отделяющие классы.
- Когда есть K классов, линейный дискриминантный анализ можно рассматривать в $K-1$ -мерной проекции.
- Даже при $K > 3$ мы можем определить «лучшую» двумерную плоскость для визуализации дискриминантного правила.

Квадратичный дискриминант (Quadratic Discriminant Analysis)

- Гипотеза: **каждый класс** $y \in Y$ имеет n -мерную гауссовскую плотность с центром μ_y и **ковариационной матрицей** Σ_y :

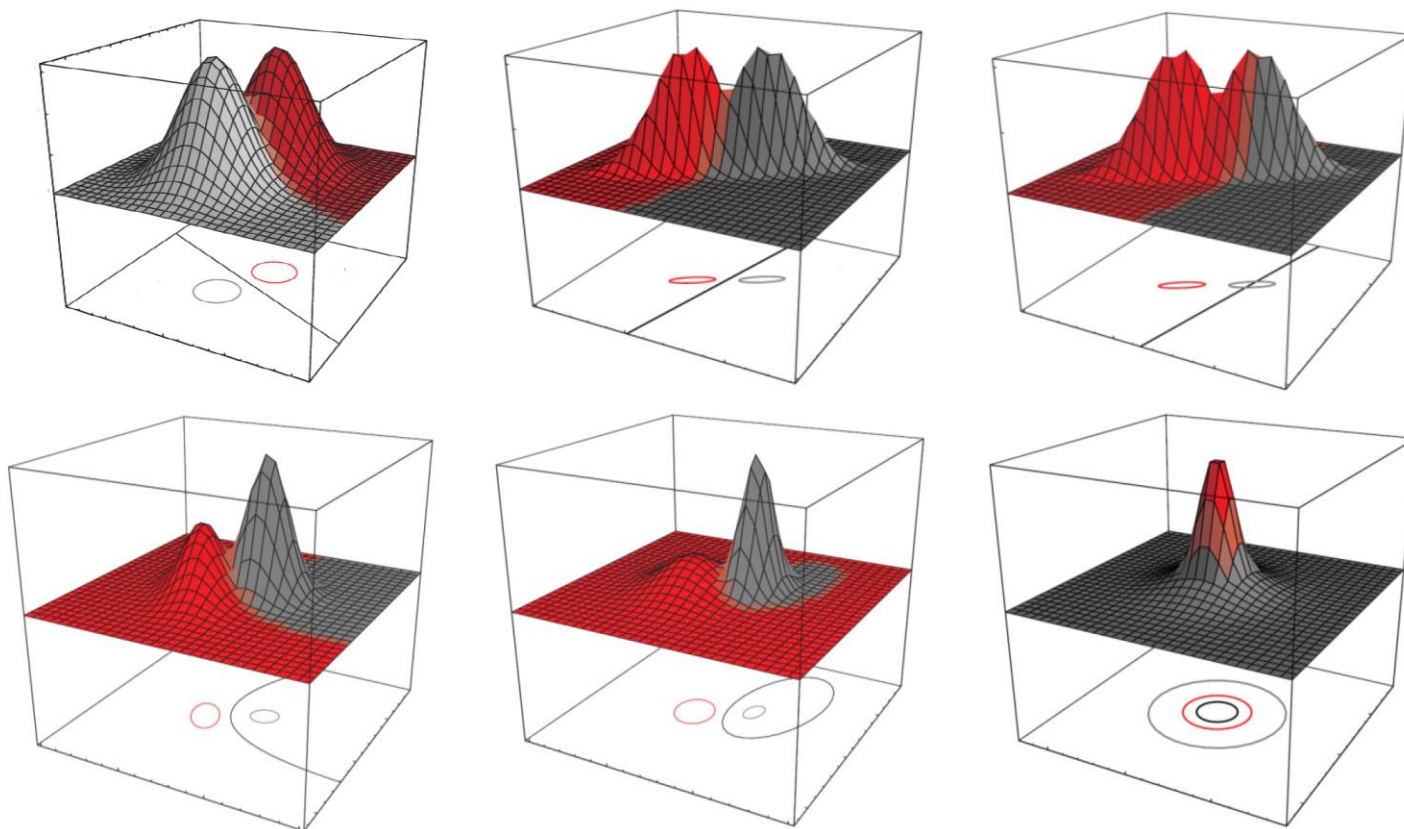
$$p(x|y) = N(x; \mu_y, \Sigma_y) = \frac{\exp\left(-\frac{1}{2}(x - \mu_y)^T \Sigma_y^{-1}(x - \mu_y)\right)}{\sqrt{(2\pi)^n \det \Sigma_y}}$$

- Теорема

- Разделяющая поверхность, определяемая уравнением $\lambda_y P(y)p(x|y) = \lambda_s P(s)p(x|s)$, **квадратична** для всех пар $y, s \in Y$.
- Если $\Sigma_y = \Sigma_s$, то поверхность **вырождается в линейную**.

- Квадратичный дискриминант – подстановочный алгоритм:

$$a(x) = \arg \max_{y \in Y} \left(\ln \lambda_y P(y) - \frac{1}{2} (x - \hat{\mu}_y)^T \hat{\Sigma}_y^{-1} (x - \hat{\mu}_y) - \frac{1}{2} \ln \det \hat{\Sigma}_y \right)$$



Логистическая регрессия по сравнению с нормальным дискриминантным анализом

- Для задачи двух классов можно показать, что модели LDA и логистическая регрессия имеют одну и ту же **линейную форму**(лу)
- Разница заключается в том, как оцениваются параметры:
 - Логистическая регрессия использует условное правдоподобие, на основе условной вероятности отклика (*дискриминационное обучение*).
 - LDA использует правдоподобие, основанное на совместном распределении отклика и признаков (*генеративное обучение*).
 - Несмотря на эти различия, на практике результаты часто похожи.
- Замечание:
 - логистическая регрессия может также быть построена с квадратичными границами, такими как у QDA, путем явного включения квадратичных членов в модель.

Почему используется дискриминантный анализ?

- Когда классы **хорошо отделимы**:
 - оценки параметров модели логистической регрессии очень **неустойчивы**, а оценки линейного дискриминанта **устойчивы**
- Если число наблюдений **мало** и распределение предикторов **близко к нормальному** в каждом из классов:
 - линейная дискриминантная модель снова более устойчива, чем модель логистической регрессии.
- Дискриминантный анализ хорош для нескольких классов
 - Можно использовать функцию **softmax** от дискриминантной функции как и в логистической регрессии

LDA – Пример

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.datasets import fetch_covtype
```

```
X, y = load_iris(return_X_y=True)
```

```
X = X[:, :2]
X.shape, y.shape, np.unique(y)

((150, 2), (150,), array([0, 1, 2]))
```

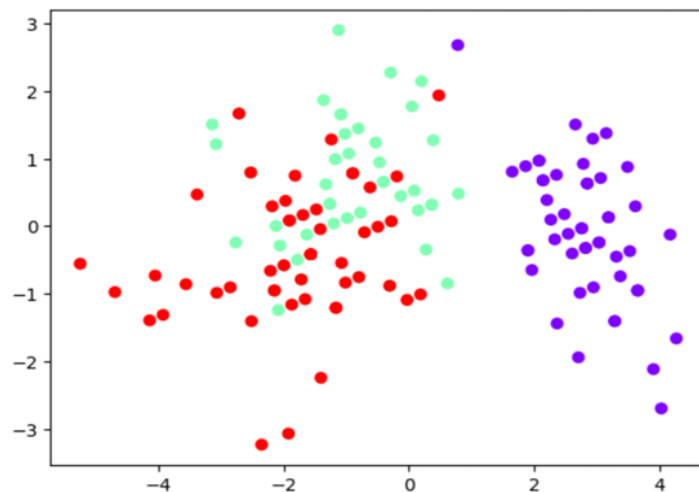
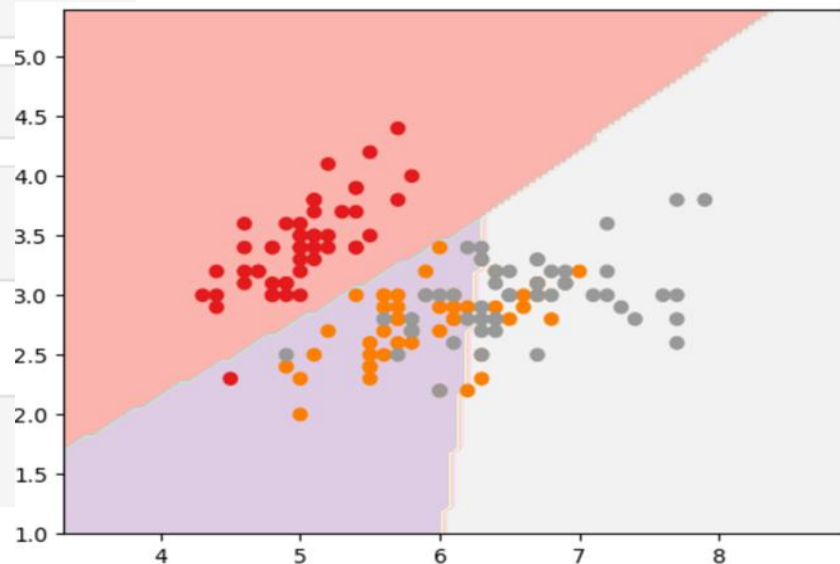
```
lda = LinearDiscriminantAnalysis(n_components=2)
lda.fit(X, y)
```

```
DecisionBoundaryDisplay.from_estimator(lda, X, cmap="Pastel1")
plt.scatter(*X.T, c=y, cmap="Set1")
```

```
transform = lda.transform(X)
transform.shape

(150, 2)
```

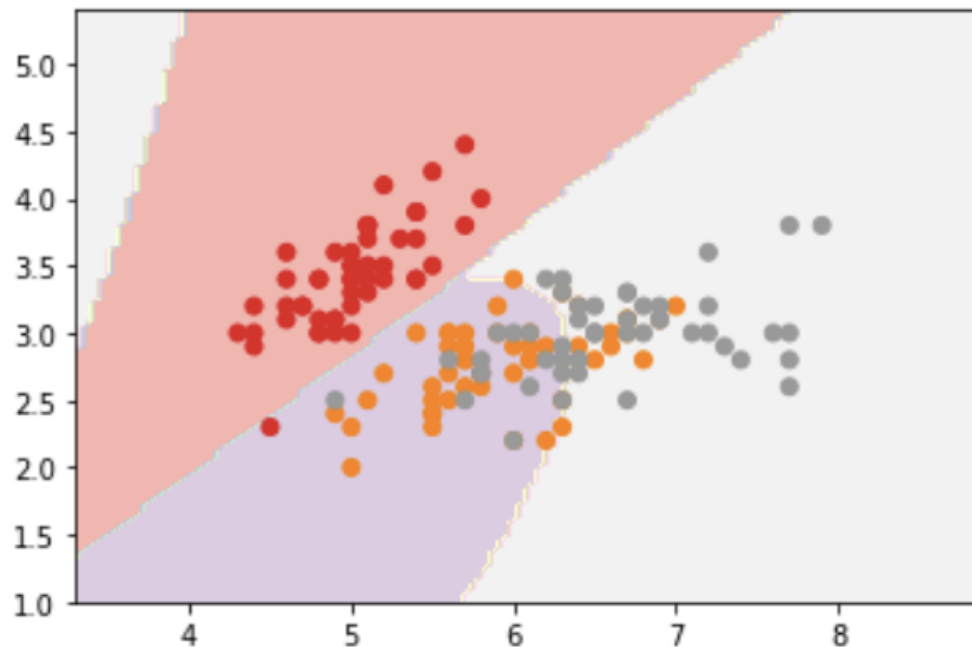
```
plt.scatter(*transform.T, c=y, cmap="rainbow")
```



QDA - Пример

```
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
qda = QuadraticDiscriminantAnalysis()
qda.fit(X, y)
DecisionBoundaryDisplay.from_estimator(qda, X, cmap="Pastel1")
plt.scatter(*X.T, c=y, cmap="Set1")
```

<matplotlib.collections.PathCollection at 0x167af6902e0>

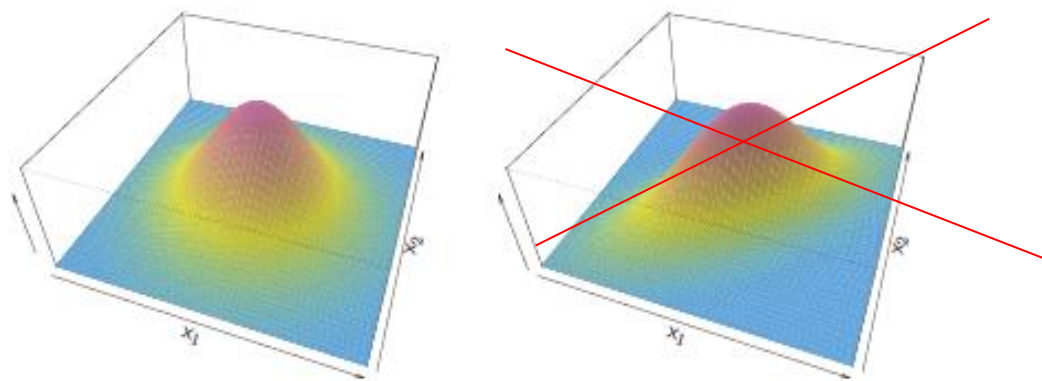


Гауссовская смесь с диагональными матрицами ковариации

- Гауссовская смесь GMM – Gaussian Mixture Model
- Допущения:
 - Функции правдоподобия классов $p(x|y)$ представимы в виде **смесей** k_y компонент, $y \in Y$.
 - Компоненты $j = 1, \dots, k_y$ имеют **n -мерные гауссовские** плотности с **некоррелированными** признаками:

$$\mu_{yj} = (\mu_{yj1}, \dots, \mu_{yjn}), \quad \Sigma_{yj} = \text{diag}(\sigma_{yj1}^2, \dots, \sigma_{yjn}^2):$$

$$p(x|y) = \sum_{j=1}^{k_y} w_{yj} p_{yj}(x), \quad p_{yj}(x) = N(x; \mu_{yj}, \Sigma_{yj}), \quad \sum_{j=1}^{k_y} w_{yj} = 1, \quad w_{yj} \geq 0$$



ЕМ-алгоритм. Эмпирические оценки средних и дисперсий

- Числовые признаки: $f_d: X \rightarrow \mathbb{R}, d = 1, \dots, n$.

- Е-шаг: для всех $y \in Y, j = 1, \dots, k_y, d = 1, \dots, n$:

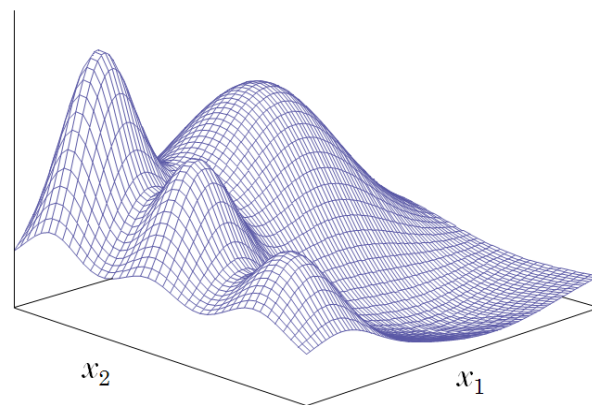
$$g_{yij} = \frac{w_{yj} N(x_i; \mu_{yj}, \Sigma_{yj})}{p(x_i|y)} \equiv P(j|x_i, y_i = y)$$

- М-шаг: для всех $y \in Y, j = 1, \dots, k_y, d = 1, \dots, n$

$$w_{yj} = \frac{1}{l_y} \sum_{i: y_i=y} g_{yij}$$

$$\hat{\mu}_{yjd} = \frac{1}{l_y w_{yj}} \sum_{i: y_i=y} g_{yij} f_d(x_i)$$

$$\hat{\sigma}_{yjd}^2 = \frac{1}{l_y w_{yj}} \sum_{i: y_i=y} g_{yij} (f_d(x_i) - \hat{\mu}_{yjd})^2$$



- Замечание: компоненты «наивны», но смесь не «наивна».

Байесовский классификатор

- Подставим гауссовскую смесь в байесовский классификатор:

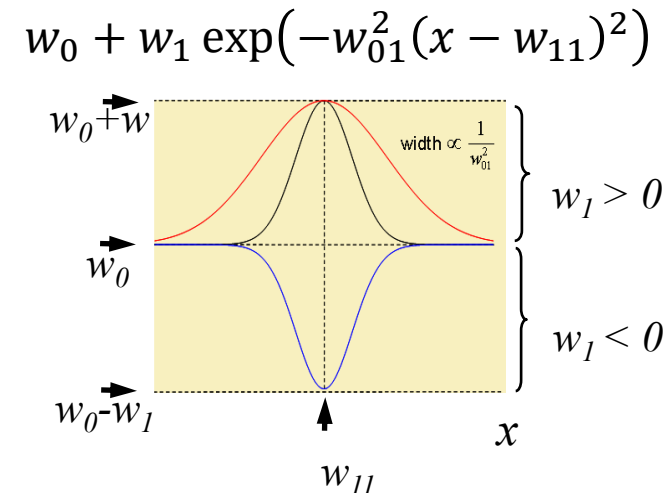
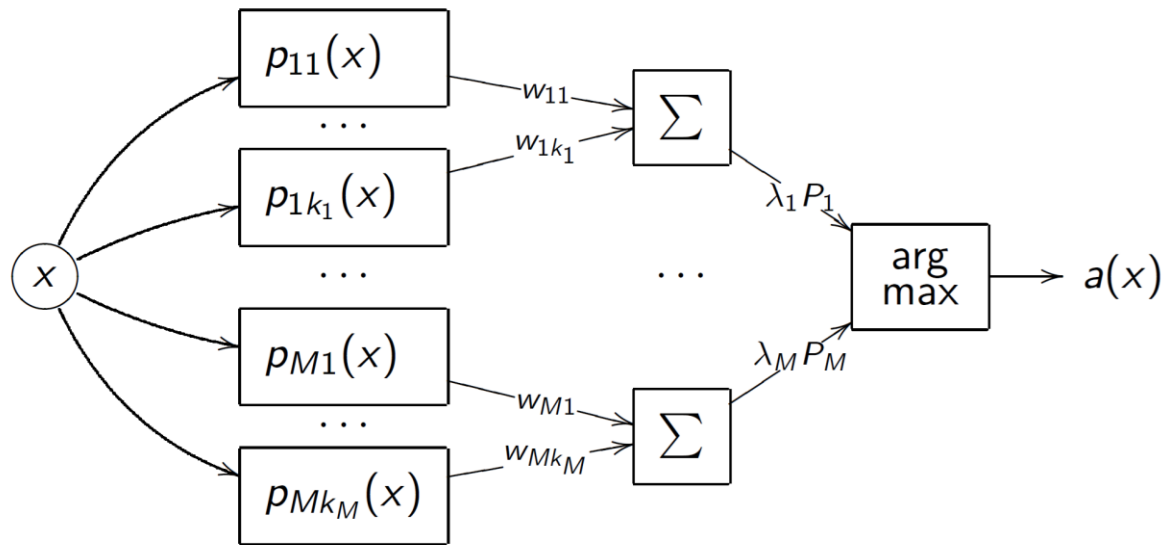
$$a(x) = \arg \max_{y \in Y} \left(\underbrace{\lambda_y P_y \sum_{j=1}^{k_y} w_{yj} \underbrace{N_{yj} \exp \left(-\frac{1}{2} \rho_{yj}^2(x, \mu_{yj}) \right)}_{p_{yj}(x)}}_{\Gamma_y(x)} \right)$$

- $N_{yj} = (2\pi)^{-\frac{n}{2}} (\sigma_{yj1}, \dots, \sigma_{ijn})^{-1}$ – нормировочные множители;
 - $\rho_{yj}(x, \mu_{yj})$ – взвешенная евклидова метрика в $X = \mathbb{R}^n$:
 - $\rho_{yj}^2(x, \mu_{yj}) = \sum_{d=1}^n \frac{1}{\sigma_{yjd}^2} (f_d(x) - \mu_{yjd})^2$.
- Интерпретация – как у метрического классификатора:
 - $p_{yj}(x)$ – близость объекта x к j -ой компоненте класса y ;
 - $\Gamma_y(x)$ – близость объекта x к классу y .

Сеть радиальных базисных функций (RBF)

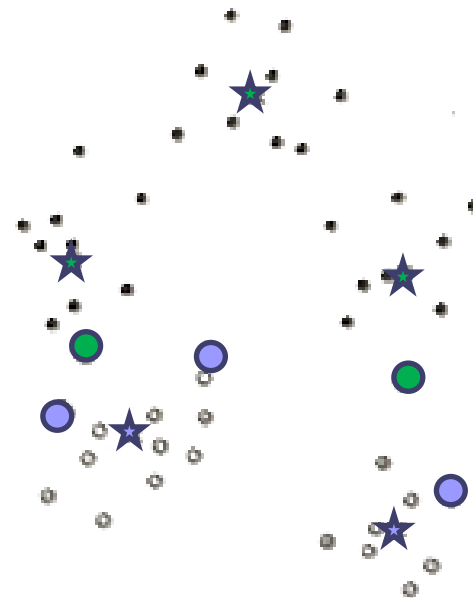
- Трехслойная сеть RBF (Radial Basis Functions):

$$a(x) = \arg \max_{y \in Y} \lambda_y P_y \sum_{j=1}^{k_y} w_{yj} p_{yj}(x)$$



ЕМ-алгоритм как метод обучения радиальных сетей

- Отличия **генеративного** RBF-ЕМ от **дискриминативного** RBF-SVM:
 - Векторы μ_{yj} – это не пограничные объекты выборки, а центры локальных сгущений классов.
 - Автоматически строится структурное описание каждого класса в виде совокупности компонент – кластеров.
- **Преимущества** ЕМ-алгоритма:
 - легко сделать устойчивым к шуму.
 - довольно быстро сходится.
- **Недостатки** ЕМ-алгоритма:
 - чувствителен к начальному приближению.
 - определение числа компонент



Резюме по байесовской теории классификации

- Основная формула:

$$a(x) = \arg \max_{y \in Y} \lambda_y P(y) p(x|y)$$

- Байесовские модели классификации – **генеративные**:

- ☐ моделируют форму классов на всем пространстве;
- ☐ требуют большего объема данных для обучения;
- ☐ менее чувствительны к шумовым выбросам.

- Наивный байесовский классификатор:

- ☐ основан на предположении о **независимости признаков**, для зависимых – **сети Байеса**;
- ☐ неплохо работает в задачах **категоризации текстов**.

- Три подхода к восстановлению плотности $p(x|y)$ по выборке:

- ☐ **Параметрический** подход: гауссовские классы => нормальный **дискриминантный** анализ
- ☐ **Непараметрический** подход: задана функция расстояния => метод **парзеновского окна**
- ☐ Разделение **смеси** распределений: классы описываются смесями гауссиан => **сеть RBF**