

Линейные модели классификации

на основе CSC2515 University of Toronto

Владимир Анатольевич Судаков
2025

Что такое «линейная» классификация?

- Классификация по своей сути нелинейна
 - Он помещает неидентичные вещи в один и тот же класс, разница во входном векторе иногда приводит к нулевому изменению ответа (к чему это приводит?)
- «Линейная классификация» означает, что часть, которая адаптируется, является линейной.
 - За адаптивной частью следует фиксированная нелинейность.
 - Ему также может предшествовать фиксированная нелинейность (например, нелинейные базисные функции).

$$y(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0, \quad \text{Decision} = f(y(\mathbf{x}))$$

адаптивная
линейная функция

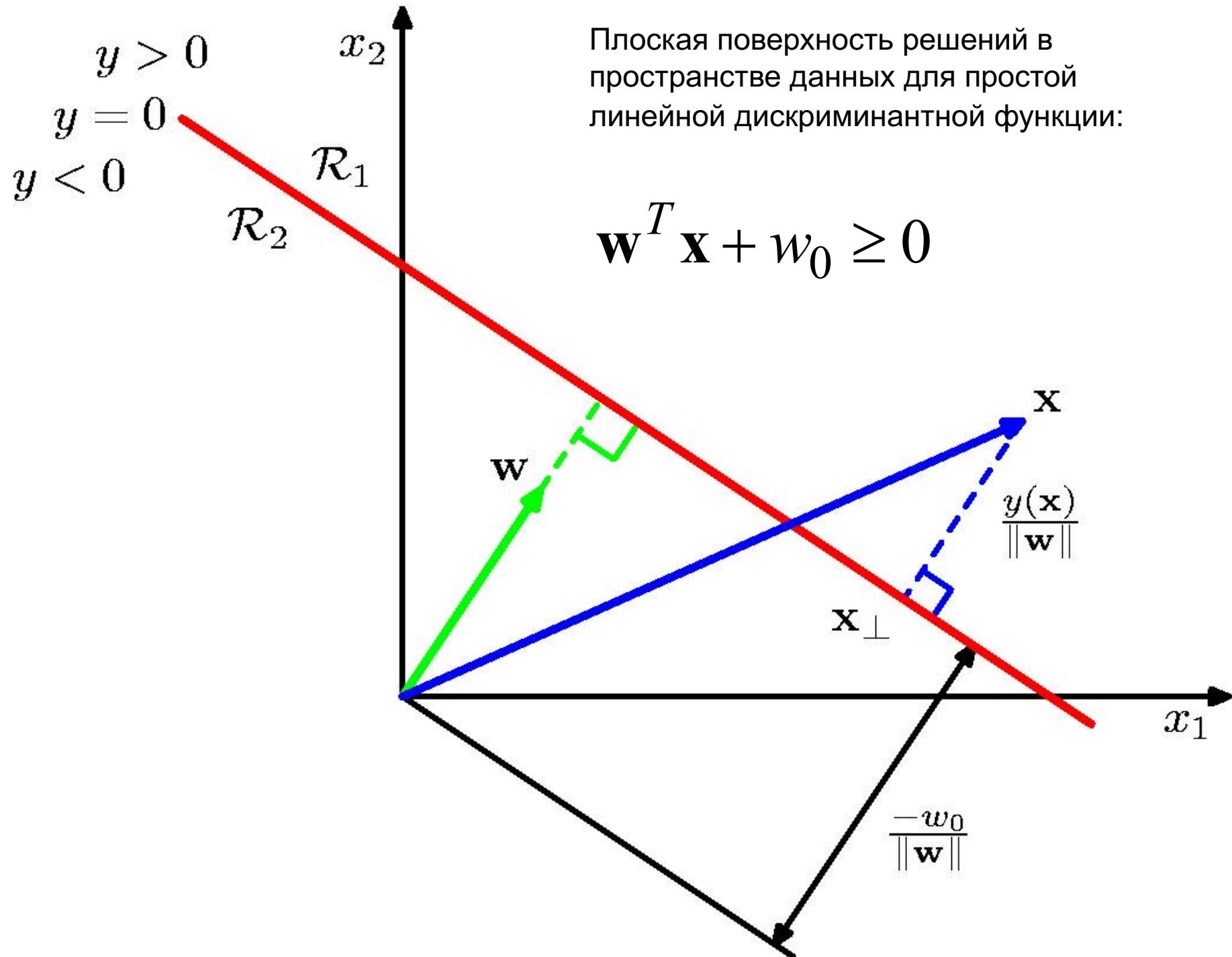
фиксированная
нелинейная
функция

Представление целевых значений для классификации

- Если есть только два класса, мы обычно используем один действительный выходной сигнал, который имеет целевые значения 1 для «положительного» класса и 0 (или иногда -1) для другого класса.
 - Для вероятностных меток классов целевым значением может быть вероятность положительного класса, а выход модели представляет собой вероятность, которую модель назначает положительному классу.
- Если имеется N классов, мы часто используем вектор из N целевых значений, содержащий одну 1 для правильного класса и нули в остальных местах.
 - Для вероятностных меток мы можем затем использовать вектор вероятностей классов в качестве целевого вектора.

Три подхода к классификации

- Используйте дискриминантные функции напрямую, без вероятностей:
 - Преобразовать входной вектор в одно или несколько действительных значений, чтобы можно было применить простую операцию (например, определение порога) для получения класса.
 - Действительные значения следует выбирать так, чтобы максимально использовать полезную информацию о метке класса, содержащуюся в действительном значении.
- Вывести условные вероятности классов: $p(class = C_k | \mathbf{x})$
 - Вычислите условную вероятность каждого класса.
 - Затем примите решение, которое минимизирует некоторую функцию потерь.
- Сравните вероятность ввода в отдельных, специфичных для класса, генеративных моделях.
 - Например, обучите многомерную гауссову функцию по входным векторам каждого класса и посмотрите, какая гауссова функция сделает вектор тестовых данных наиболее вероятным. (Это лучший вариант?)



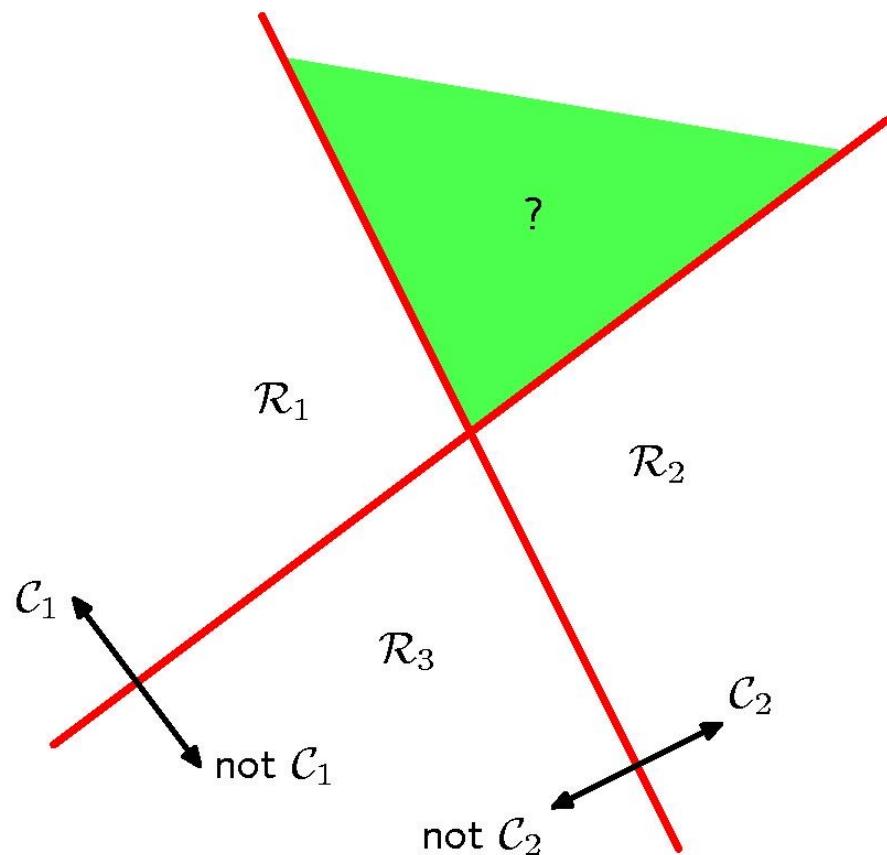
Напоминание: три разных пространства, которые легко спутать

- Пространство весов
 - Каждая ось соответствует весу
 - Точка — это вектор весов
 - Размерность = #веса +1 дополнительное измерение для потери
- Пространство данных
 - Каждая ось соответствует входному значению
 - Точка — это вектор данных. Поверхность решения — это плоскость.
 - Размерность = размерность вектора данных
- «Пространство случая»
 - Каждая ось соответствует учебному случаю
 - Точка присваивает скалярное значение каждому обучающему случаю.
 - Таким образом, он может представлять одномерные цели или может представлять значение одного входного компонента по всем обучающим данным.
 - Размерность = #кейсы обучения

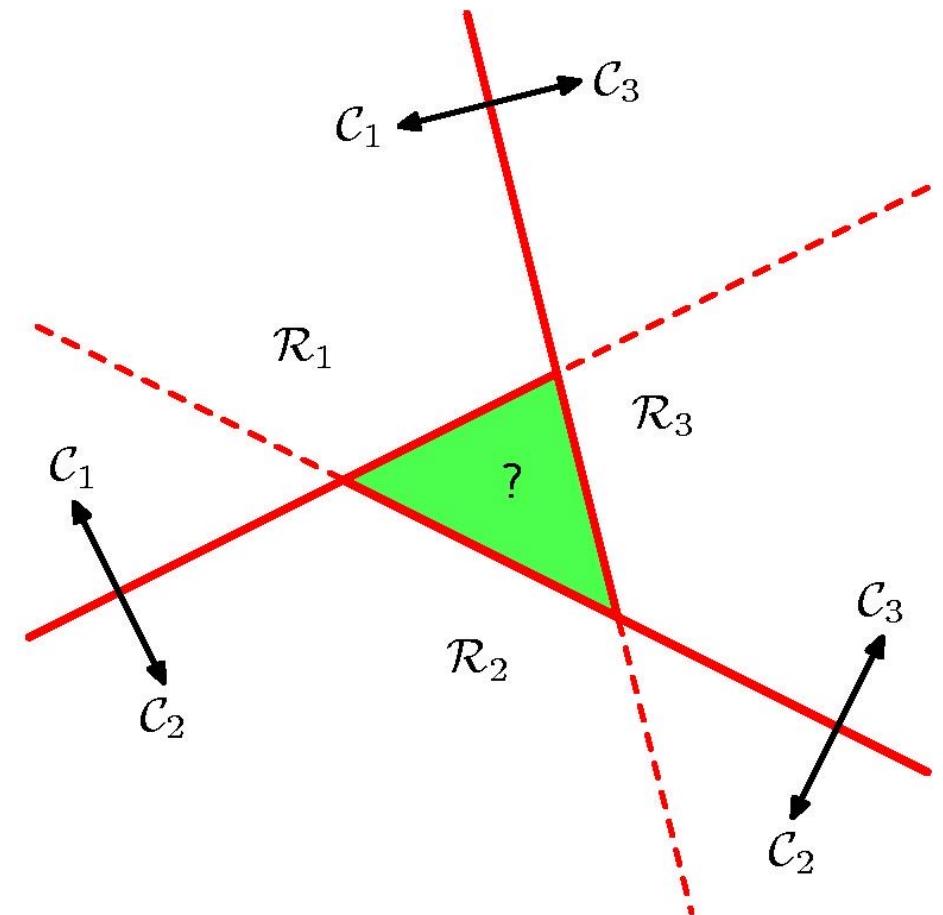
Дискриминантные функции для $N > 2$ классов

- Одной из возможностей является использование N двусторонних дискриминантных функций.
 - Каждая функция отличает один класс от остальных.
- Другая возможность — использовать $N(N-1)/2$ двусторонних дискриминантных функций.
 - Каждая функция различает два конкретных класса.
- Оба эти метода имеют проблемы

Проблемы с многоклассовыми дискриминантными функциями



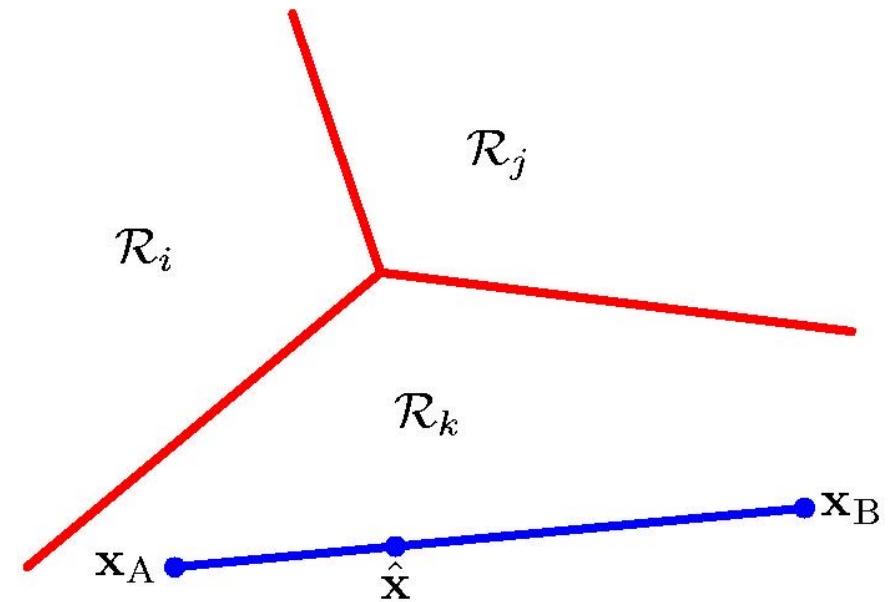
Более одного
хорошего ответа



Двусторонние предпочтения
не обязательно должны быть
транзитивными!

Простое решение

- Используйте N $y_i, y_j, y_k \dots$ дискриминантных функций и выберите максимальную.
 - Это гарантированно даст последовательные и выпуклые области решений, если у линейна.



$$y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A) \text{ and } y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B)$$

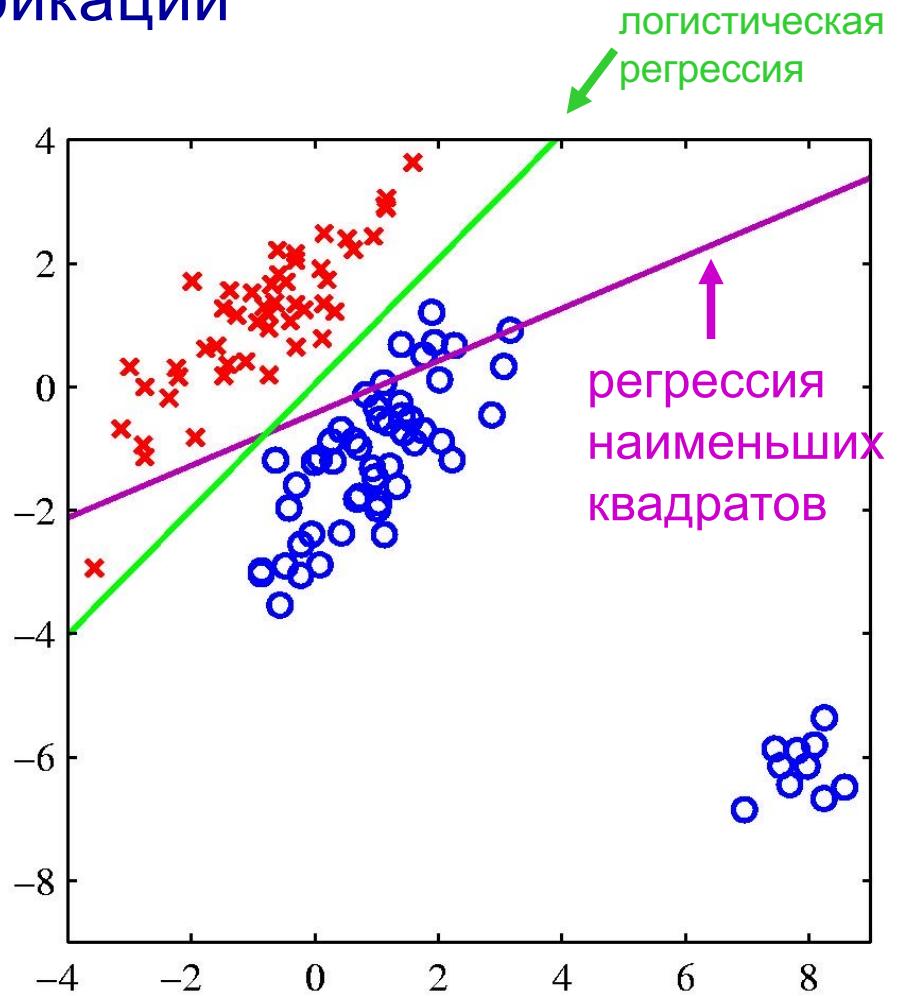
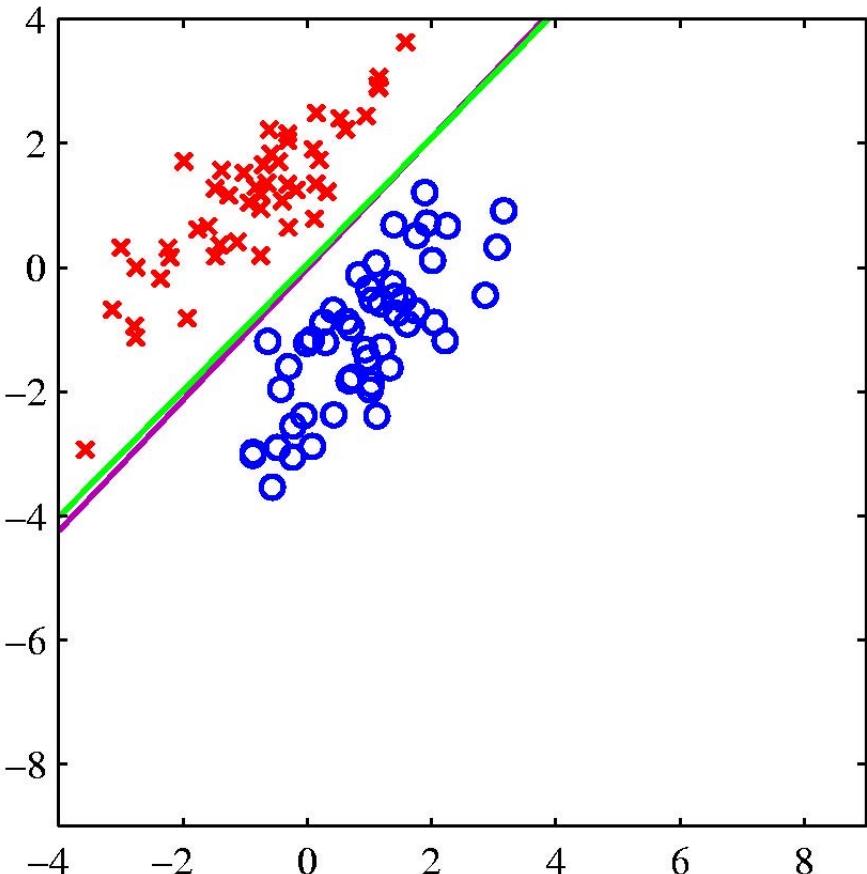
implies (for positive α) that

$$y_k(\alpha \mathbf{x}_A + (1 - \alpha) \mathbf{x}_B) > y_j(\alpha \mathbf{x}_A + (1 - \alpha) \mathbf{x}_B)$$

Использование метода наименьших квадратов для классификации

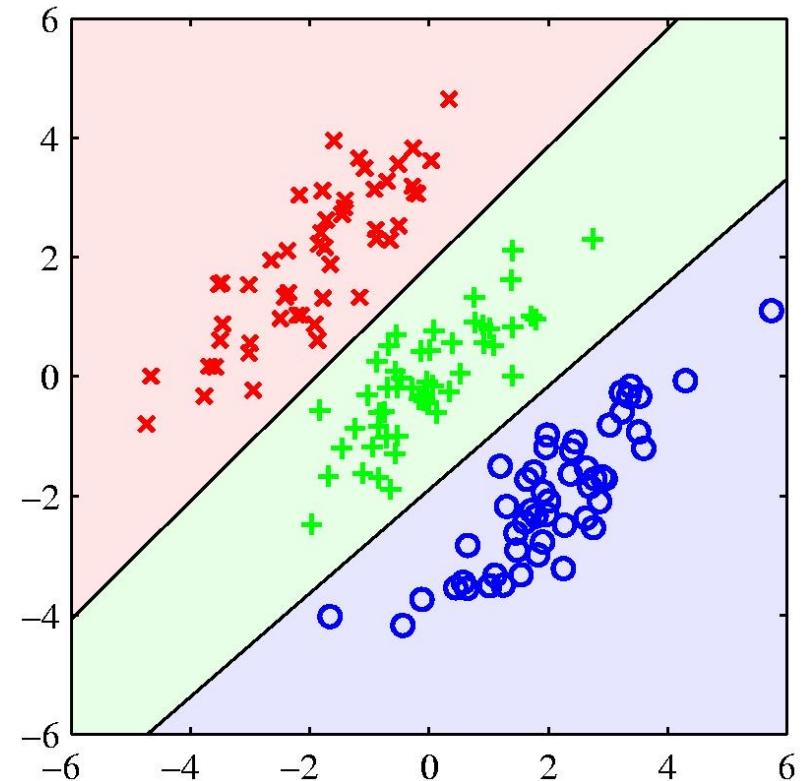
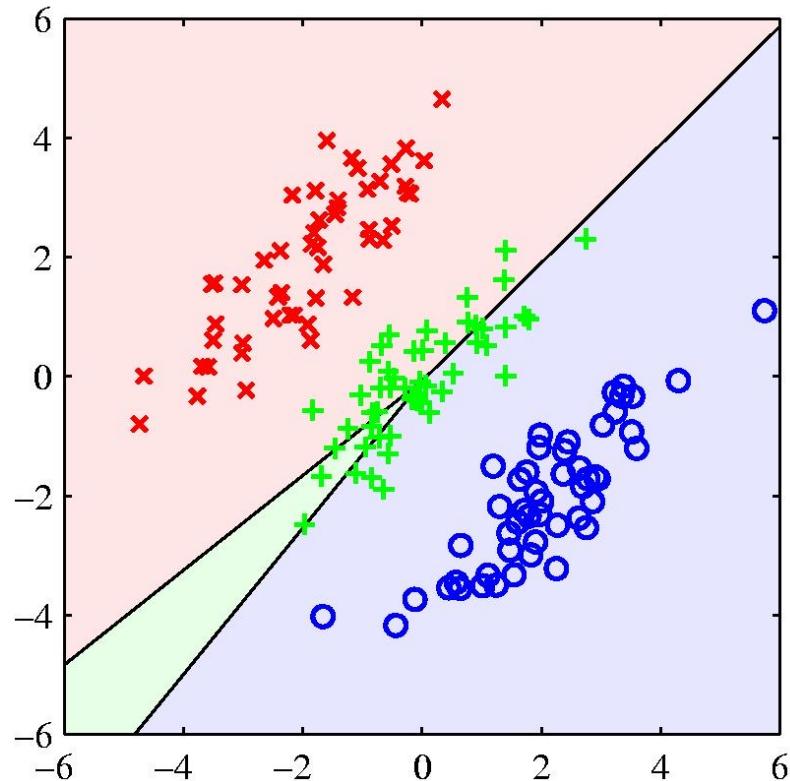
- Не работает так хорошо, как лучшие методы, но он прост:
 - Он сводит классификацию к регрессии наименьших квадратов.
 - Мы уже умеем строить регрессию. Остаётся только найти оптимальные веса, используя матричную алгебру (см. ML 2).
- Используем цели, которые равны условной вероятности класса с учетом входных данных.
 - Когда имеется более двух классов, мы рассматриваем каждый класс как отдельную задачу (мы не можем избежать этого, если используем функцию принятия решения “*max*”).

Проблемы с использованием наименьших квадратов для классификации



Если правильный ответ — 1, а модель говорит 1,5, она проигрывает, поэтому она меняет границу, чтобы избежать «слишком правильной» оценки.

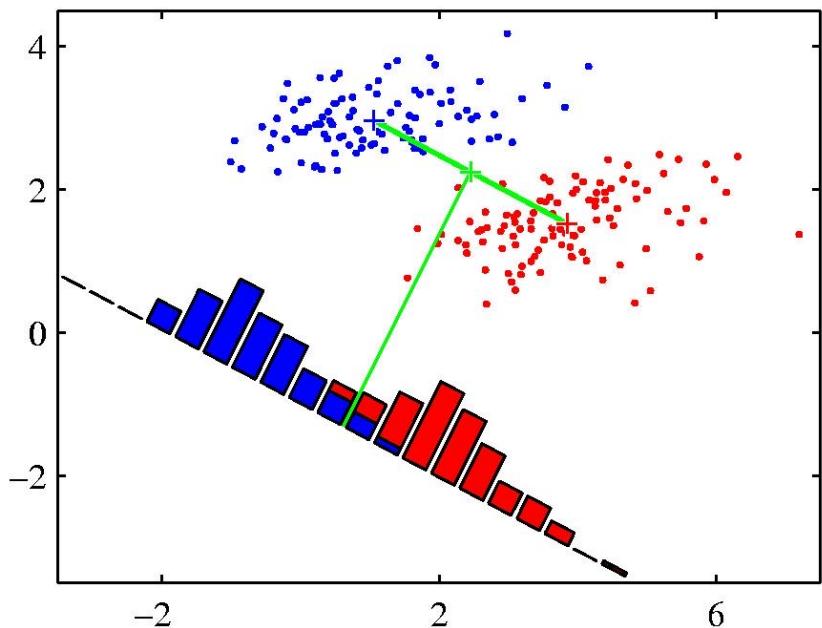
Другой пример, когда регрессия наименьших квадратов дает плохие поверхности решений



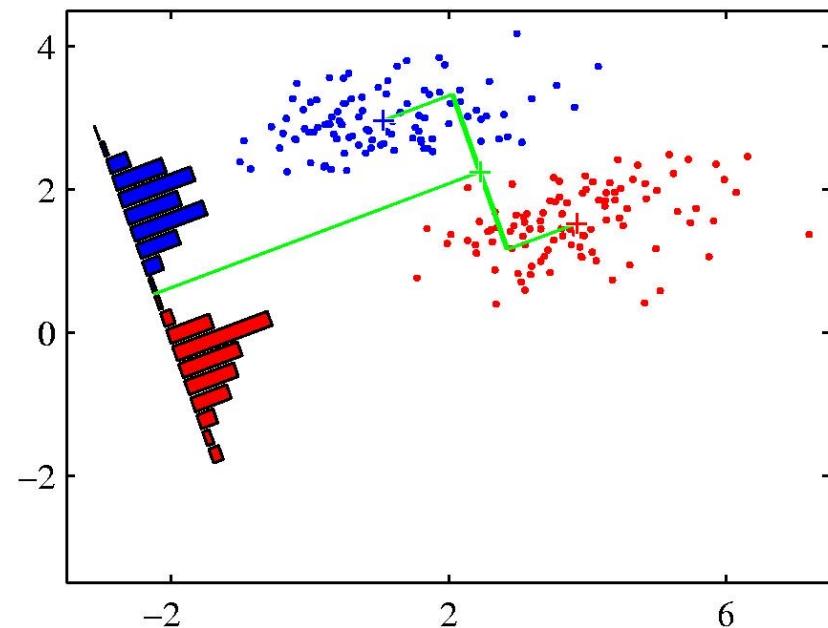
Линейный дискриминант Фишера

- Простая линейная дискриминантная функция представляет собой проекцию данных на одномерный уровень.
 - Итак, выберите проекцию, которая обеспечивает наилучшее разделение классов. Что мы подразумеваем под «наилучшим разделением»?
- Очевидным направлением выбора является направление линии, соединяющей средние значения класса.
 - Однако если основное направление дисперсии в каждом классе не ортогонально этой линии, то это не даст хорошего разделения (см. следующий рисунок).
- Метод Фишера выбирает направление, которое максимизирует отношение межклассовой дисперсии к внутриклассовой дисперсии.
 - Это направление, в котором спроектированные точки содержат наибольшую информацию о принадлежности к классу (в соответствии с гауссовыми предположениями).

Рисунок, демонстрирующий преимущество линейного дискриминанта Фишера



При проецировании на линию, соединяющую классы, классы не очень хорошо разделены.



Фишер выбирает направление, которое делает прогнозируемые классы гораздо плотнее, даже если их прогнозируемые средние значения не так далеко друг от друга.

Математика линейных дискриминантов Фишера

- Какое линейное преобразование лучше всего подходит для дискриминации?
- Проекция на вектор, разделяющий класс - кажется разумным

$$y = \mathbf{w}^T \mathbf{x}$$

$$\mathbf{w} \propto \mathbf{m}_2 - \mathbf{m}_1$$

- Нам также нужна небольшая дисперсия внутри каждого класса:

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in C_1} \mathbf{x}_n, \quad \mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in C_2} \mathbf{x}_n$$

$$s_1^2 = \sum_{n \in C_1} (y_n - m_1)^2$$

$$s_2^2 = \sum_{n \in C_2} (y_n - m_2)^2$$

- Целевая функция Фишера:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

← между
← в пределах

Больше математики линейных дискриминантов Фишера

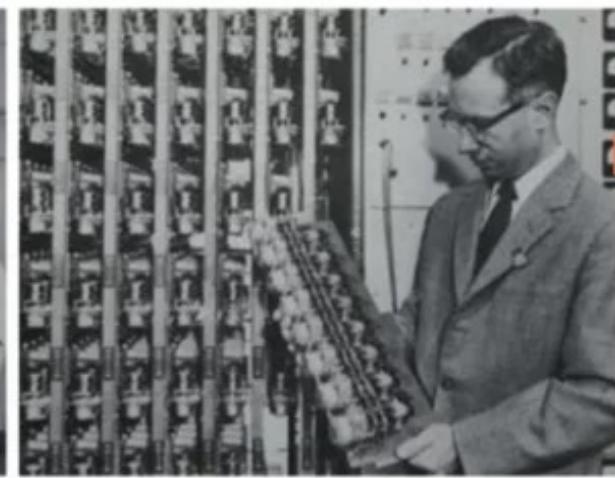
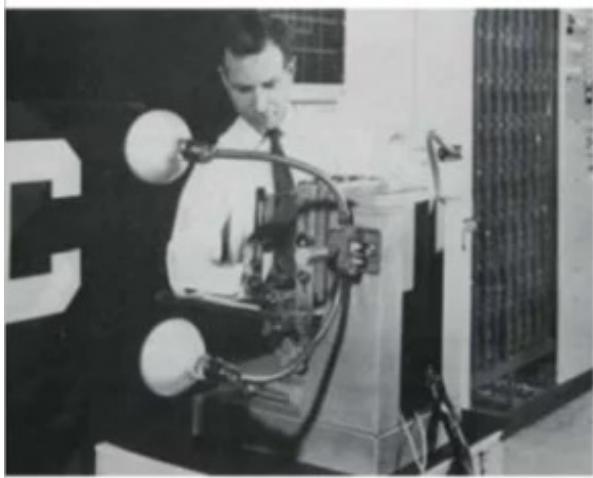
$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1) (\mathbf{m}_2 - \mathbf{m}_1)^T$$

$$\mathbf{S}_W = \sum_{n \in C_1} (\mathbf{x}_n - \mathbf{m}_1) (\mathbf{x}_n - \mathbf{m}_1)^T + \sum_{n \in C_2} (\mathbf{x}_n - \mathbf{m}_2) (\mathbf{x}_n - \mathbf{m}_2)^T$$

Optimal solution: $\mathbf{w} \propto \mathbf{S}_W^{-1} (\mathbf{m}_2 - \mathbf{m}_1)$

Персепtron



- «Персептроны» представляют собой целое семейство обучающихся машин, но стандартный тип состоит из слоя фиксированных нелинейных базисных функций, за которым следует простая линейная дискриминантная функция.
 - Они были введены в конце 1950-х годов и имели простую процедуру онлайн-обучения.
 - Об их способностях были сделаны громкие заявления. Это вызвало множество споров.
 - Исследователи символического ИИ подчеркивали его ограничения (в рамках идеологической кампании против действительных чисел, вероятностей и обучения)
- Машины опорных векторов (Support Vector Machines) — это просто персептроны с умным способом выбора неадаптивных, нелинейных базисных функций и лучшей процедурой обучения.
 - Они имеют те же ограничения, что и персептроны, в отношении того, какие типы функций они могут изучить.

Персепtron

представляет собой модель бинарной классификации, в которой входной вектор \mathbf{x} сначала преобразуется с помощью нелинейного преобразования для получения вектора признаков $\Phi(\mathbf{x})$, а затем используется для построения обобщенной линейной модели вида

$$y(\mathbf{x}) = f(\mathbf{w}^T \Phi(\mathbf{x}))$$

Предполагаем, что вектор $\Phi(\mathbf{x})$ включает компонент смещения ϕ_0 . Нелинейная функция активации $f(\cdot)$ задаётся ступенчатой функцией вида

$$f(\alpha) = \begin{cases} +1, & \alpha \geq 0 \\ -1, & \alpha < 0 \end{cases}$$

Это связано с тем, что для персептрана удобнее использовать целевые значения

$t = +1$ для класса \mathcal{C}_1

$t = -1$ для класса \mathcal{C}_2 , вместо $t \in \{0,1\}$.

ФУНКЦИЯ ОШИБКИ

рассматриваем функцию ошибки, называемую *критерием персептрана*.

Ищется весовой вектор \mathbf{w} , такой, что входные данные \mathbf{x}_n , принадлежащие классу \mathcal{C}_1 , имеют $\mathbf{w}^T \Phi(\mathbf{x}_n) > 0$, тогда как входные данные, принадлежащие классу \mathcal{C}_2 , имеют $\mathbf{w}^T \Phi(\mathbf{x}_n) < 0$.

Учитывая схему кодирования $t \in \{-1, +1\}$, следует, что все входные данные должны удовлетворять:

$$\mathbf{w}^T \Phi(\mathbf{x}_n) t_n > 0$$

Таким образом, критерий персептрана пытается минимизировать величину $-\mathbf{w}^T \Phi(\mathbf{x}_n) t_n$, для всех неправильно классифицированных входных данных.

$$E_P(\mathbf{w}) = - \sum_{n \in \mathcal{M}} \mathbf{w}^T \Phi(\mathbf{x}_n) t_n$$

где \mathcal{M} обозначает набор неправильно классифицированных шаблонов.

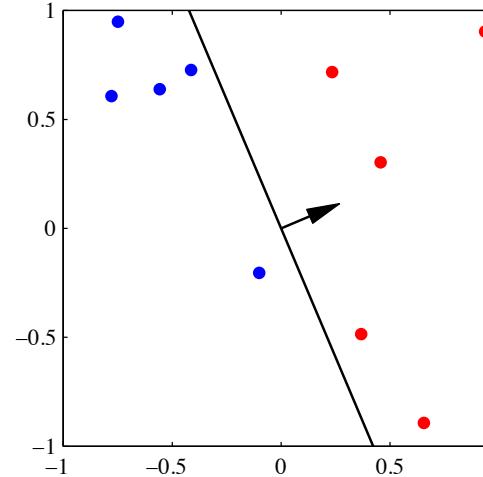
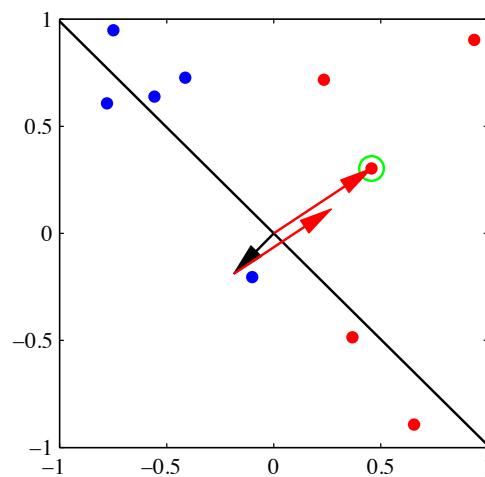
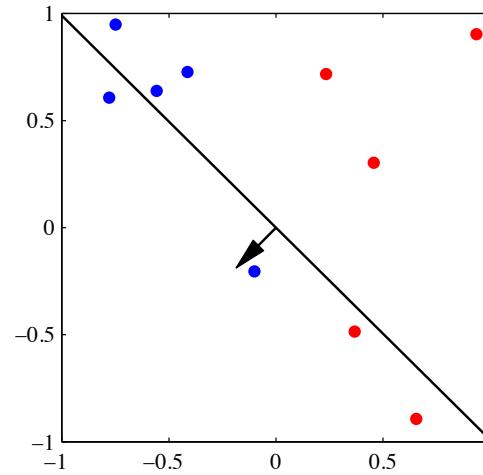
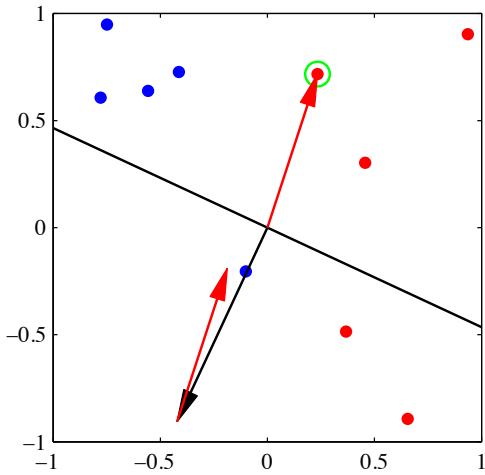
Процедура сходимости персептрана

- Добавьте к каждому вектору признаков дополнительный компонент со значением 1. Вес «смещения» этого компонента равен минус пороговому значению. Теперь о пороговом значении можно забыть.
- Выбирайте учебные случаи, используя любую политику, которая гарантирует, что каждый учебный случай будет выбран.
 - Если вывод правильный, оставьте его веса в покое.
 - Если выход равен 0, а должен быть 1, добавьте вектор признаков к вектору весов.
 - Если выход равен 1, а должен быть 0, вычтите вектор признаков из вектора веса.
- Это гарантирует нахождение набора весов, который даст правильный ответ на всем обучающем наборе, **если такой набор существует.**
- Нет необходимости выбирать скорость обучения.

Естественный способ попытаться доказать сходимость

- Очевидный подход — записать функцию ошибки и попытаться показать, что каждый шаг процедуры обучения уменьшает ошибку.
 - Для стохастического онлайн-обучения мы хотели бы показать, что каждый шаг уменьшает ожидаемую ошибку, где ожидание касается выбора обучающих случаев.
 - Это не может быть квадратичная ошибка, поскольку размер обновления не зависит от размера ошибки.
- В учебнике в качестве меры погрешности пытаются использовать сумму расстояний на неправильной стороне поверхности принятия решений.
 - В результате был сделан вывод о том, что процедура сходимости персептрона не гарантирует снижения общей ошибки на каждом шаге.
 - Это справедливо для данной функции ошибки, даже если существует набор весов, дающий правильный ответ для каждого случая обучения.

Вес и пространство данных



Сходимость алгоритма обучения персептрана, на которой показаны точки из двух классов {красного и синего} в двухмерном пространстве признаков (ϕ_1, ϕ_2). Слева вверху показан начальный вектор параметров представленный в виде черной стрелки вместе с соответствующей границей решения (черная линия), где стрелка указывает на область принятия решения, которая классифицируется как принадлежащая к красному классу. Точка, обведенная зеленым кружком, классифицирована ошибочно, поэтому ее вектор-функция добавляется к текущему вектору весов, что дает новую границу решения, показанную справа вверху. Слева внизу показана следующая ошибочная точка, обозначенная зеленым кружком, которую следует учесть, а ее вектор-функция снова добавляется к весовому вектору, давая границу решения, показанную справа внизу, где все точки классифицированы правильно

Лучший способ доказать сходимость (используя выпуклость решений в весовом пространстве)

- Очевидный тип функции ошибок измеряет расхождение между целевыми значениями и выходными данными модели.
- Другой тип функции стоимости использует квадрат расстояния между текущими весами и допустимым набором весов.
 - Используя эту функцию стоимости, можно показать, что каждый шаг процедуры уменьшает ошибку.
 - При условии, что существует набор допустимых весов.
- Используя этот тип функции стоимости, процедуру можно легко обобщить на более чем два класса, используя правило принятия решений MAX.

Почему процедура обучения работает

- Рассмотрим квадрат расстояния между любым удовлетворительным весовым вектором и текущим весовым вектором.
 - Каждый раз, когда персепtron совершает ошибку, алгоритм обучения уменьшает квадрат расстояния между текущим вектором веса и любым удовлетворительным вектором веса (если только он не пересекает плоскость принятия решений).
- Поэтому рассмотрим «вполне удовлетворительные» векторы веса, которые лежат в допустимой области с запасом, по крайней мере таким же большим, как и наибольшее обновление.
 - Каждый раз, когда персепtron совершает ошибку, квадрат расстояния до всех этих весовых векторов всегда уменьшается по крайней мере на квадрат длины наименьшего вектора обновления.

Чему персептроны не могут научиться

- Адаптивная часть персептрана не может даже определить, имеют ли два однобитовых признака одинаковое значение!

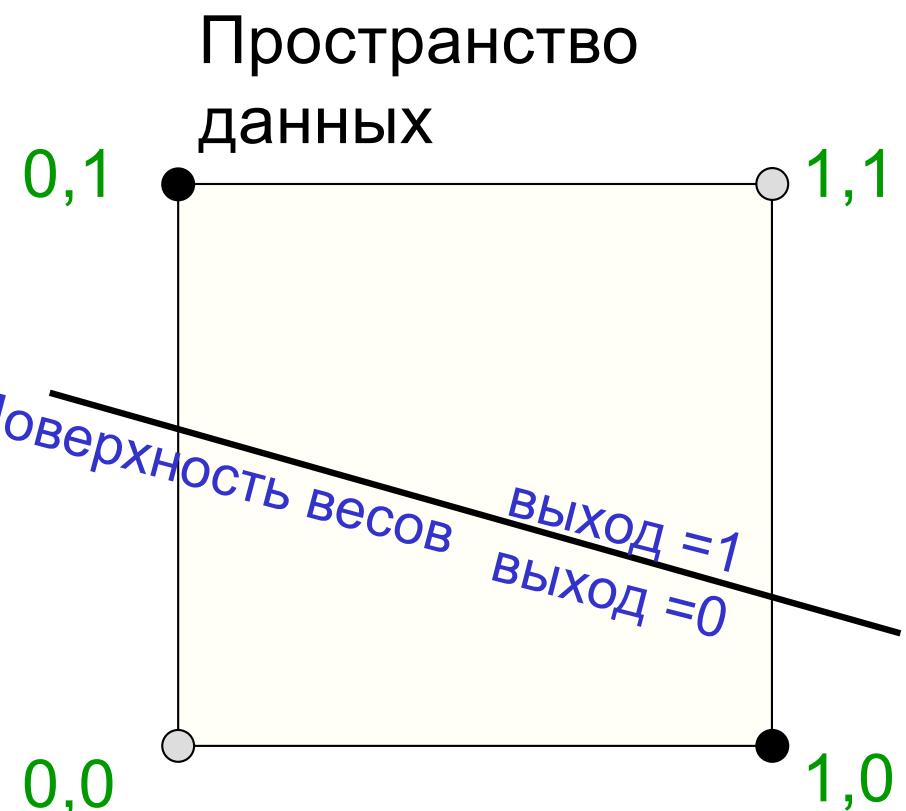
Однаковые : $(1,1) \rightarrow 1; (0,0) \rightarrow 1$

Разные : $(1,0) \rightarrow 0; (0,1) \rightarrow 0$

- Четыре пары «признак-выход» дают четыре неравенства, которые невозможно удовлетворить:

$$w_1 + w_2 \geq \theta, \quad 0 \geq \theta$$

$$w_1 < \theta, \quad w_2 < \theta$$



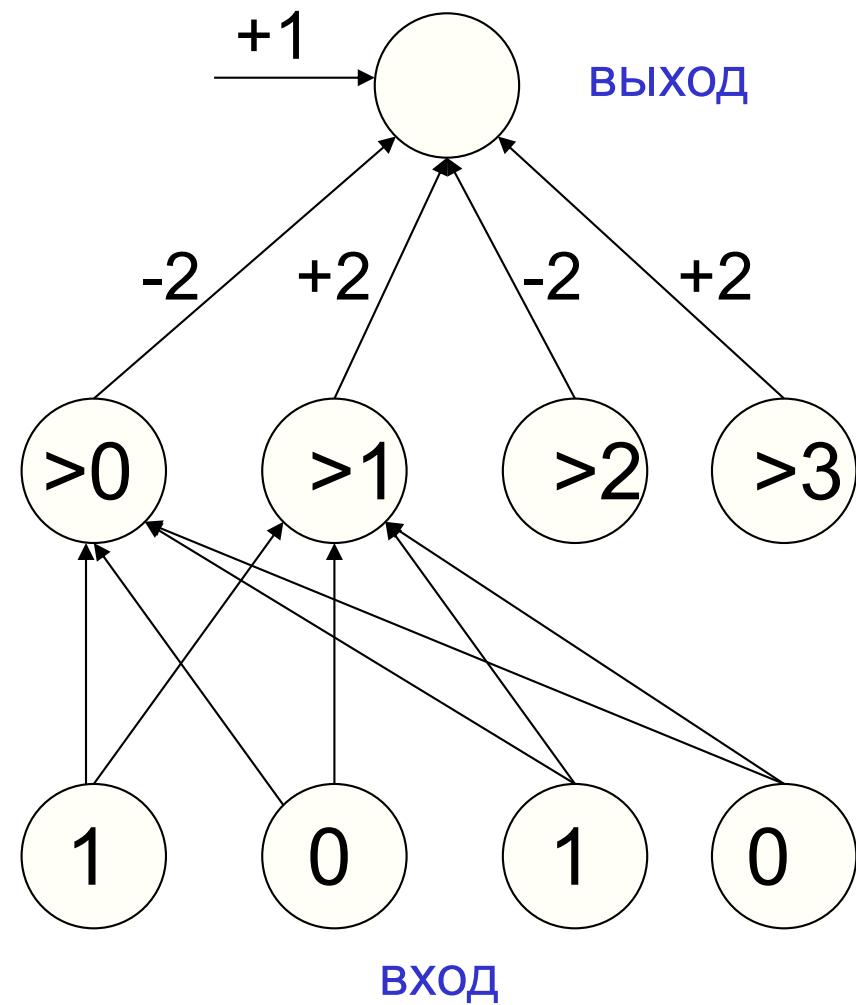
Положительные и отрицательные случаи
не могут быть разделены плоскостью

Что умеют персептроны?

- Они могут решать задачи только в том случае, если закодированные вручную признаки преобразуют исходную задачу в линейно разделимую. Насколько это сложно?
- Задача проверки четности N-бит:
 - Требуется N признаков вида: Включены ли по крайней мере m бит?
 - Каждая функция должна учитывать **все компоненты входных данных.**
- Задача на двумерную связность
 - требуется экспоненциальное количество функций!

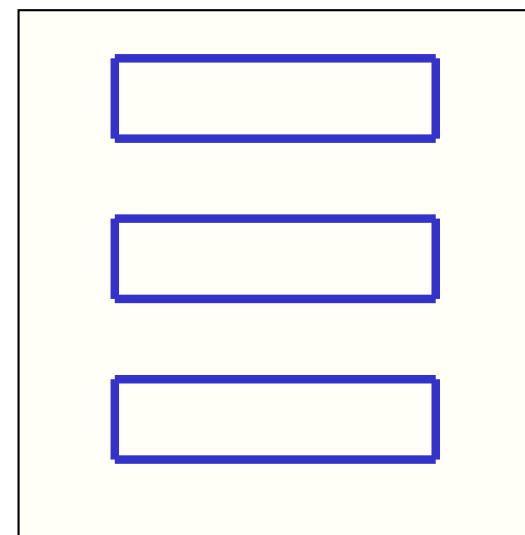
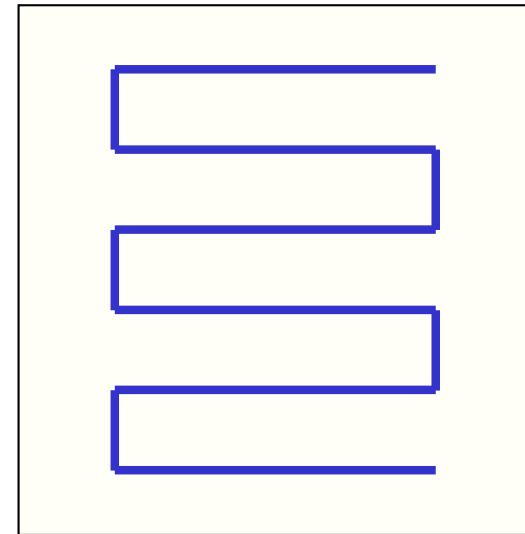
Задача проверки четности N-бит

- Существует простое решение, требующее N скрытых единиц.
 - Каждый скрытый блок вычисляет, включено ли более M входов.
 - Это линейно разделимая задача.
- Существует множество вариантов этого решения.
 - Этому можно научиться.
 - Он хорошо обобщает, если: $2^N \gg N^2$



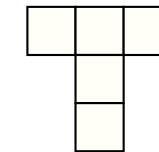
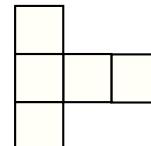
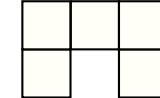
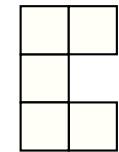
Почему связность трудно вычислить

- Даже для простых линейных рисунков существует экспоненциальное количество случаев.
- Удаление одного сегмента может нарушить связность
 - Но это зависит от точного расположения остальных частей.
 - В отличие от паритета, здесь нет простых обобщений других частей, которые бы говорили нам, что произойдет.
- Связность легко вычисляется с помощью итеративного алгоритма.
 - Начните с любого места в чернилах
 - Распространить маркер
 - Посмотрите, все ли чернила испачкаются.



Различение Т и С в любой ориентации и положении

- Какие признаки необходимы для различения двух разных шаблонов из 5 пикселей независимо от положения и ориентации?
 - Нужно ли нам реплицировать шаблоны Т и С во всех позициях и ориентациях?
 - Просмотр пар пикселей не будет работать.
 - Рассмотрение троек будет работать, если предположить, что каждое входное изображение содержит только один объект.



Повторите следующие два детектора признаков во всех позициях.



Если хотя бы один из показателей равен пороговому значению 2, то это С. Если нет, то это Т.

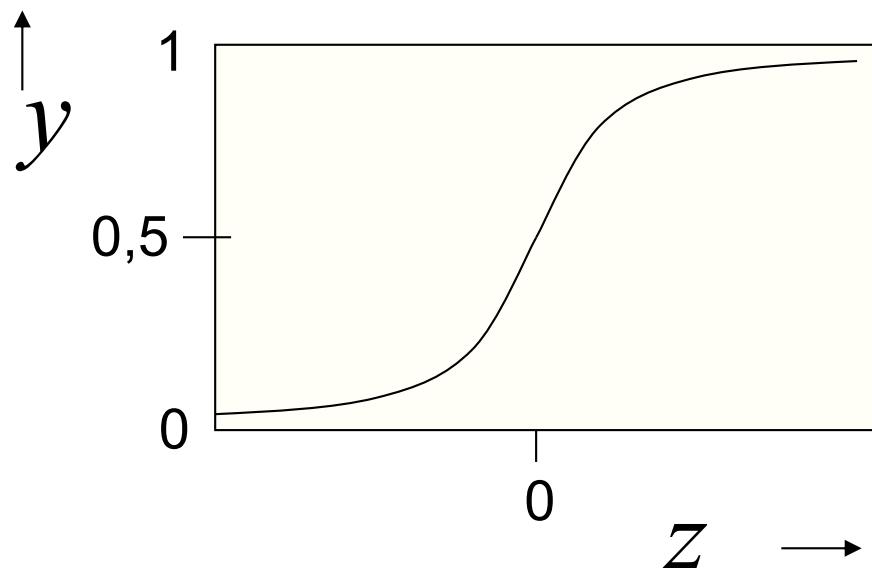
Логистическая регрессия

(перейти к странице
205)

- Когда есть только два класса, мы можем смоделировать условную вероятность положительного класса как
$$p(C_1 | \mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x} + w_0) \quad \text{where} \quad \sigma(z) = \frac{1}{1 + \exp(-z)}$$
- Если мы используем правильную функцию ошибки, происходит нечто интересное: градиент логистической функции и градиент функции ошибки компенсируют друг друга:
$$\nabla E(\mathbf{w}) = -\ln p(t|\mathbf{w}), \quad \nabla E(\mathbf{w}) = \sum_{n=1}^N (y_n - t_n) \mathbf{x}_n$$

Логистическая функция

- Выходные данные представляют собой плавную функцию входных данных и весов.



$$z = \mathbf{w}^T \mathbf{x} + w_0$$

$$y = \sigma(z) = \frac{1}{1 + e^{-z}}$$

$$\frac{\partial z}{\partial w_i} = x_i$$

$$\frac{\partial z}{\partial x_i}$$

$$\frac{dy}{dz} = y(1-y)$$



Странно
с точки зрения y .

Естественная функция ошибок для логистической

- Чтобы построить логистическую модель с использованием максимального правдоподобия, нам необходимо минимизировать отрицательную логарифмическую вероятность правильного ответа, просуммированного по обучающему набору.

$$\begin{aligned} E &= -\sum_{n=1}^N \ln p(t_n | y_n) \\ &= -\sum_{n=1}^N t_n \ln y_n + (1-t_n) \ln (1-y_n) \\ &\quad \uparrow \qquad \qquad \uparrow \\ &\quad \text{если } t = 1 \qquad \qquad \text{если } t = 0 \\ \frac{\partial E_n}{\partial y_n} &= -\frac{t_n}{y_n} + \frac{1-t_n}{1-y_n} \\ &= \frac{y_n - t_n}{y_n (1-y_n)} \end{aligned}$$

↗ производная ошибки в обучающем случае n

Использование цепного правила для получения производных ошибок

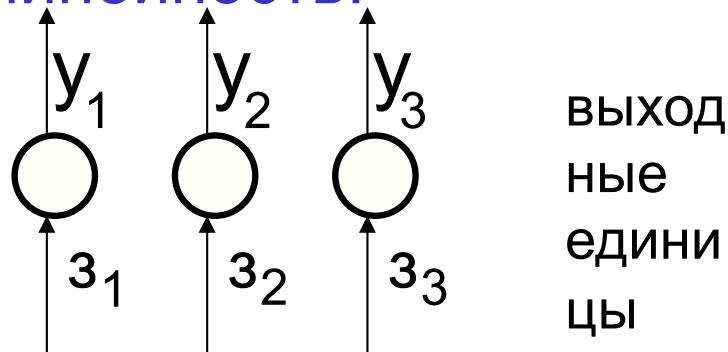
$$z_n = \mathbf{w}^T \mathbf{x}_n + w_0, \quad \frac{\partial z_n}{\partial \mathbf{w}} = \mathbf{x}_n$$

$$\frac{\partial E_n}{\partial y_n} = \frac{y_n - t_n}{y_n(1-y_n)}, \quad \frac{dy_n}{dz_n} = y_n(1-y_n)$$

$$\frac{\partial E_n}{\partial \mathbf{w}} = \frac{\partial E_n}{\partial y_n} \frac{dy_n}{dz_n} \frac{\partial z_n}{\partial \mathbf{w}} = (y_n - t_n) \mathbf{x}_n$$

ФУНКЦИЯ ОШИБКИ «SOFTMAX» ИЛИ КРОСС-ЭНТРОПИЯ ДЛЯ МНОГОКЛАССОВОЙ КЛАССИФИКАЦИИ

Выходные блоки используют нелокальную нелинейность:



Функция естественной стоимости — это отрицательный логарифм вероятности правильного ответа.

Крутизна Е точно уравновешивает пологость softmax.

$$y_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$\frac{\partial y_i}{\partial z_i} = y_i (1 - y_i)$$

целевое значение

$$E = - \sum_j t_j \ln y_j$$

$$\frac{\partial E}{\partial z_i} = \sum_j \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial z_i} = y_i - t_i$$

Частный случай softmax для двух классов

$$y_1 = \frac{e^{z_1}}{e^{z_1} + e^{z_0}} = \frac{1}{1 + e^{-(z_1 - z_0)}}$$

- Таким образом, логистика — это просто частный случай, в котором избегается использование избыточных параметров:
 - Добавление одной и той же константы к z_1 и z_0 не дает никакого эффекта.
 - Излишняя параметризация softmax обусловлена тем, что вероятности должны в сумме давать 1.

Вероятностные генеративные модели дискриминации

- Используйте отдельную генеративную модель входных векторов для каждого класса и посмотрите, какая модель делает тестовый входной вектор наиболее вероятным.
 - Апостериорная вероятность класса 1 определяется по формуле:
- $$\frac{p(C_1)p(\mathbf{x} | C_1)}{p(C_1)p(\mathbf{x} | C_1) + p(C_0)p(\mathbf{x} | C_0)} = \frac{1}{1 + e^{-z}}$$

where $z = \ln \frac{p(C_1)p(\mathbf{x} | C_1)}{p(C_0)p(\mathbf{x} | C_0)} = \boxed{\ln \frac{p(C_1 | \mathbf{x})}{1 - p(C_1 | \mathbf{x})}}$



z называется логарифмом
и задается логарифмом
шансов

Простой пример для непрерывных входных данных

- Предположим, что входные векторы для каждого класса имеют гауссовское распределение и все классы имеют одинаковую ковариационную матрицу.

$$p(\mathbf{x} | C_k) = \frac{1}{\text{нормализу ющая константа}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_k)\right\}$$

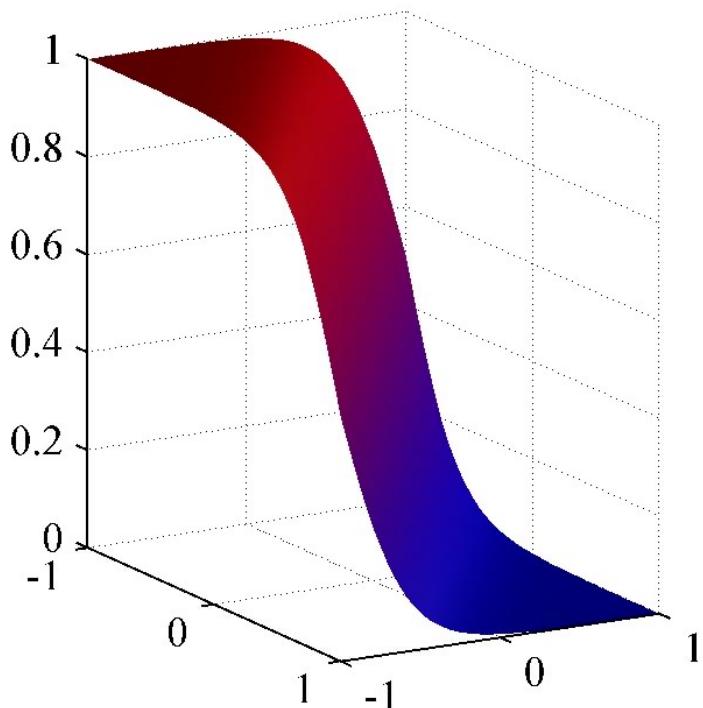
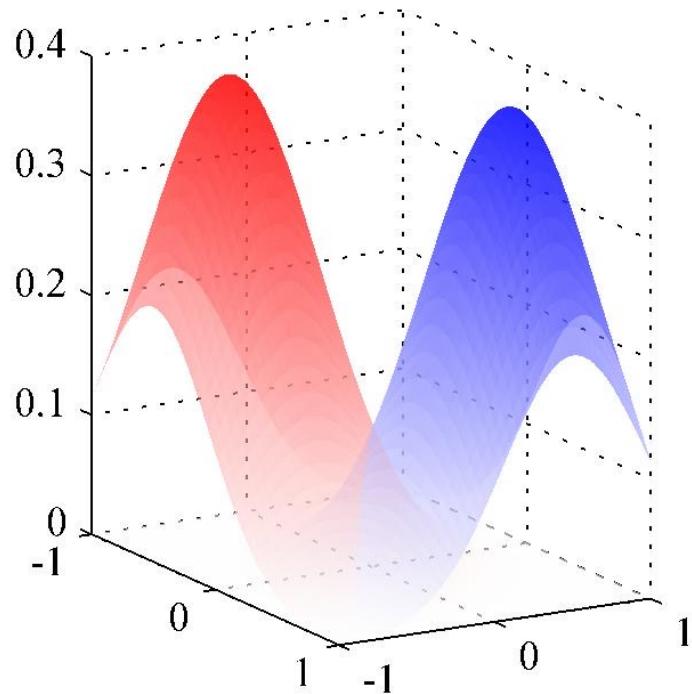
- Для двух классов, C1 и C0, апостериорная функция представляет собой логистику:

$$p(C_1 | \mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x} + w_0)$$

$$\mathbf{w} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

$$w_0 = -\frac{1}{2}\boldsymbol{\mu}_1^T \Sigma^{-1} \boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_0^T \Sigma^{-1} \boldsymbol{\mu}_0 + \ln \frac{p(C_1)}{p(C_0)}$$

Изображение двух гауссовых моделей и полученная апостериорная функция для красного класса



Способ размышления о роли обратной ковариационной матрицы

- Если гауссиана сферическая, нам не нужно беспокоиться о ковариационной матрице.
- Итак, мы могли бы начать с преобразования пространства данных, чтобы сделать гауссово сферическое
 - Это называется «отбеливанием» данных.
 - Он предварительно умножается на квадратный корень матрицы обратной ковариации.

$$\mathbf{w} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

gives the same value

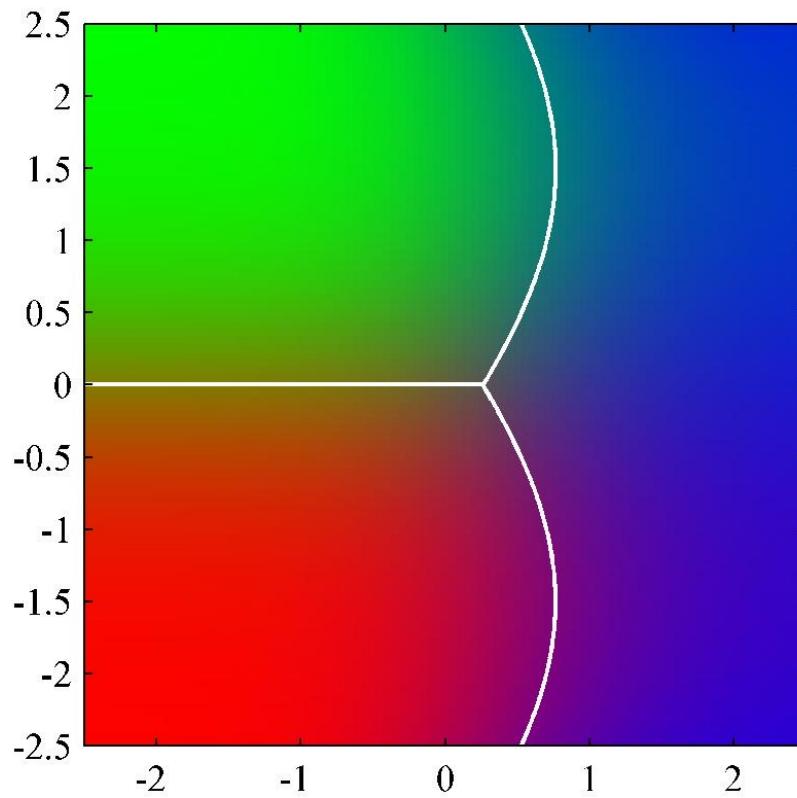
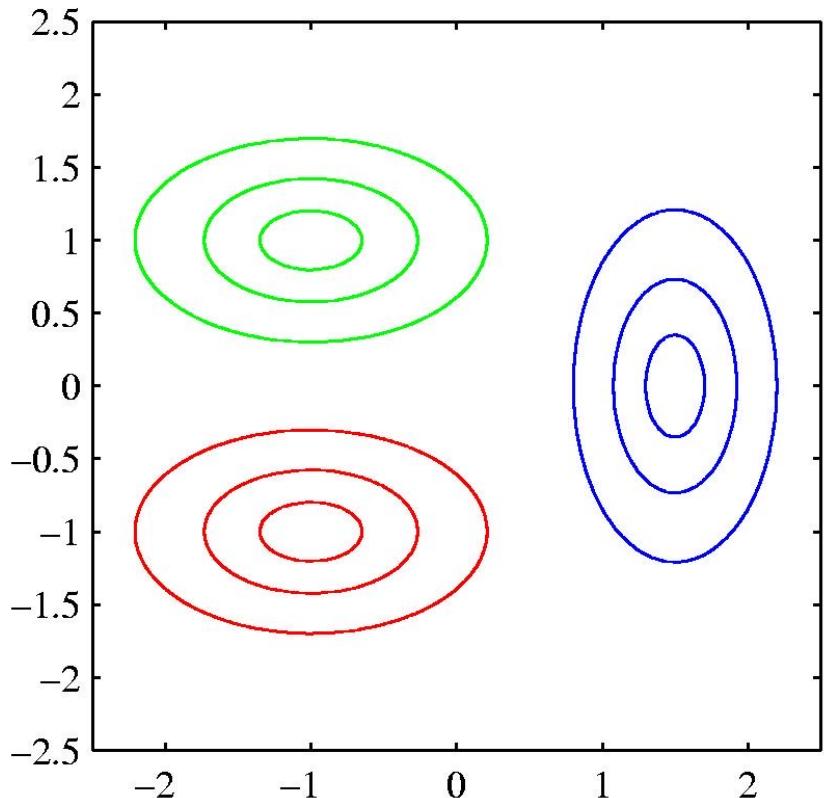
for $\mathbf{w}^T \mathbf{x}$ as :

$$\mathbf{w}_{aff} = \Sigma^{-\frac{1}{2}} \boldsymbol{\mu}_1 - \Sigma^{-\frac{1}{2}} \boldsymbol{\mu}_0$$

$$and \quad \mathbf{x}_{aff} = \Sigma^{-\frac{1}{2}} \mathbf{x}$$

gives for $\mathbf{w}_{aff}^T \mathbf{x}_{aff}$

Апостериорная вероятность, когда матрицы ковариации различны для разных классов

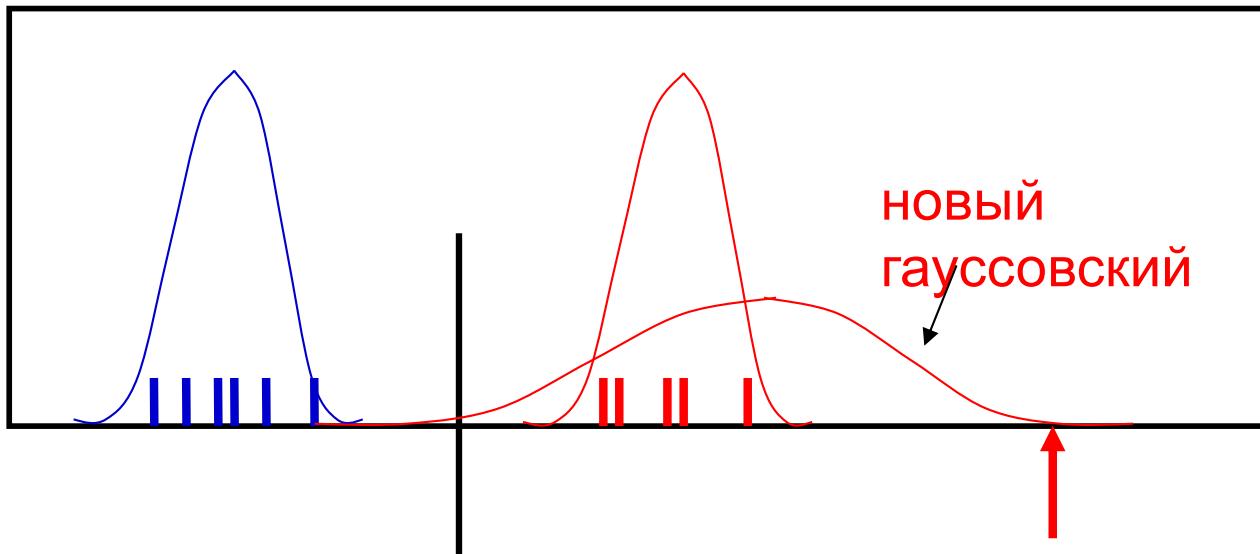


Поверхность принятия решений является плоской, если ковариационные матрицы одинаковы, и квадратной, если они не одинаковы.

Два способа обучения набора генеративных моделей, специфичных для класса

- Генеративный подход Обучайте каждую модель отдельно, чтобы она соответствовала входным векторам соответствующего класса.
 - Разные модели можно обучать на разных ядрах.
 - Легко добавить новый класс без переобучения всех остальных классов.
- Это существенные преимущества, когда модели сложнее обучать, чем простые линейные модели, рассматриваемые здесь.
- Дискриминационный подход. Обучите все параметры обеих моделей, чтобы максимально увеличить вероятность правильного присвоения меток.

Пример, когда два типа обучения ведут себя совершенно по-разному



граница
принятия
решений

Что произойдет с границей
принятия решений, если мы
добавим сюда новую красную
точку?

При генеративной подгонке красное среднее смещается вправо, а граница принятия решения — влево! Если вы действительно верите в гауссовские данные, это разумно.

Оценка качества классификации

Пусть есть два класса $Y=\{0,1\}$. Пусть банк использует систему классификации заёмщиков на кредитоспособных и некредитоспособных. Обнаружение некредитоспособного заёмщика ($y=1$) можно рассматривать как "сигнал тревоги", сообщающий о возможных рисках.

Возможны следующие исходы классификации:

- Некредитоспособный заёмщик классифицирован как некредитоспособный, т.е. положительный класс распознан как положительный (True Positive — TP).
- Кредитоспособный заёмщик классифицирован как кредитоспособный, т.е. отрицательный класс распознан как отрицательный. (True Negative — TN).
- Кредитоспособный заёмщик классифицирован как некредитоспособный, т.е. имела место ошибка, в результате которой отрицательный класс был распознан как положительный (False Positive — FP) – это ошибка I рода (ложная тревога).
- Некредитоспособный заёмщик распознан как кредитоспособный, т.е. имела место ошибка, в результате которой положительный класс был распознан как отрицательный (False Negative — FN) – это ошибка II рода (пропуск цели).

Вопрос для самоконтроля

Где ошибка первого рода и где ошибка второго рода?



Метрики качества классификации

Аккуратность (англ. Accuracy) – доля правильных ответов. Бесполезна в задачах с неравными классами.

Точность (англ. Precision) - доля правильных ответов модели в пределах класса:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Полнота (англ. Recall) - это доля истинно положительных классификаций:

$$\text{Recall} = \frac{TP}{TP + FN}$$

F-мера (англ. F-score) – гармоническое среднее между точностью и полнотой.

Сравнение метрик

Модель 1

F1=0.34, accuracy=0.48, precision=0.21, recall=0.95

[[51 77]

[1 20]]

Модель 2

F1=0.54, accuracy=0.82, precision= 0.42, recall=0.76

[[106 22]

[5 16]]