

# Методы машинного обучения

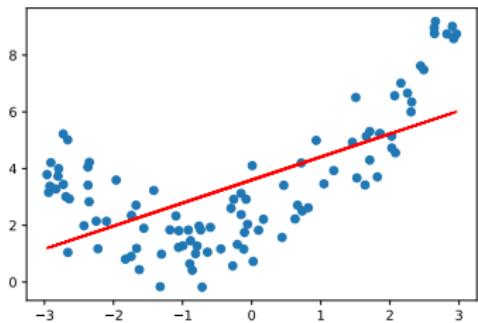
Шорохов С.Г.

кафедра математического моделирования и искусственного интеллекта

## Лекция 2. Нелинейная регрессия



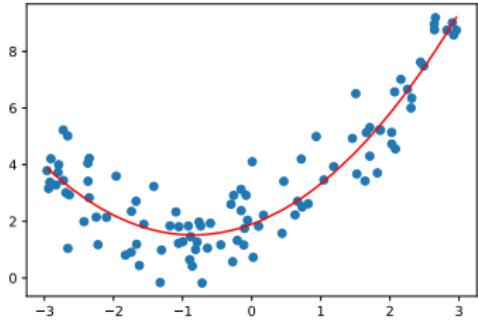
# Регрессия для нелинейных данных



Линейной регрессии вида

$$Y = \beta + \boldsymbol{\omega}^T \mathbf{X} + \varepsilon = \beta + \sum_{i=1}^d \omega_i X_i + \varepsilon$$

может быть недостаточно для выявления взаимосвязи между признаками  $X_1, X_2, \dots, X_d$  и откликом  $Y$  в случае, когда эта взаимосвязь является нелинейной.





# Полиномиальная регрессия

Полиномиальная регрессия – это форма регрессионного анализа, в которой взаимосвязь между независимой переменной  $X$  и зависимой переменной  $Y$  моделируется как полином  $m$ -й степени от  $X$

$$Y = f(X, \mathbf{w}) = w_0 + w_1 X + w_2 X^2 + \dots + w_m X^m + \varepsilon$$

Полиномиальная регрессия соответствует нелинейной зависимости между значением  $X$  и соответствующим условным средним значением  $Y$ , обозначаемым  $\mathbb{E}[Y | X]$ . Хотя полиномиальная регрессия подгоняет нелинейную модель к нелинейным данным, как задача статистической оценки она является линейной в том смысле, что функция регрессии  $\mathbb{E}[Y | X]$  является линейной по неизвестным параметрам  $\mathbf{w} = (w_0, w_1, \dots, w_m)^T$ , которые оцениваются на основе данных. По этой причине полиномиальная регрессия считается частным случаем множественной линейной регрессии.



# Уравнения полиномиальной регрессии

Пусть входные данные имеют вид  $\mathbf{D} = \left\{ \mathbf{X} = (x_1, \dots, x_n)^T, \mathbf{Y} = (y_1, \dots, y_n)^T \right\}$ , тогда модель полиномиальной регрессии принимает вид системы линейных уравнений относительно весов  $\mathbf{w} = (w_0, w_1, \dots, w_m)^T$ :

$$\mathbf{Y} = \mathbf{V} \mathbf{w} + \boldsymbol{\varepsilon},$$

где  $\mathbf{V}$  представляет собой  $n \times (m + 1)$ -матрицу Вандермонда вида

$$\mathbf{V} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^m \\ 1 & x_2 & x_2^2 & \dots & x_2^m \\ 1 & x_3 & x_3^2 & \dots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^m \end{bmatrix}, \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Можно доказать, что матрица Вандермонда  $\mathbf{V}$  при  $m < n$  имеет максимальный ранг  $m + 1$  тогда и только тогда, когда все точки  $x_i$  различны. Тогда  $(m + 1) \times (m + 1)$ -матрица  $\mathbf{V}^T \mathbf{V}$  будет невырожденной, т.е. будет существовать обратная матрица  $(\mathbf{V}^T \mathbf{V})^{-1}$ .



# Коэффициенты полиномиальной регрессии

Матрица  $\mathbf{V}$  нелинейным образом зависит от значений признака  $\mathbf{X}$ , но функция полиномиальной регрессии является линейной относительно коэффициентов регрессии (весов)  $\mathbf{w} = (w_0, w_1, \dots, w_m)^T$ :

$$\hat{\mathbf{Y}} = f(\mathbf{X}, \mathbf{w}) = \mathbf{V}(\mathbf{X}) \mathbf{w},$$

поэтому в случае максимального ранга матрицы  $\mathbf{V}$  коэффициенты регрессии  $\mathbf{w}$  могут быть определены методом наименьших квадратов:

$$\min_{\mathbf{w}} SSE = \min_{\mathbf{w}} \|\boldsymbol{\varepsilon}\|^2 = \min_{\mathbf{w}} \left\| \mathbf{Y} - \hat{\mathbf{Y}} \right\|^2 = \min_{\mathbf{w}} \left\| \mathbf{Y} - \mathbf{V}(\mathbf{X}) \mathbf{w} \right\|^2$$

Приравнивая градиент  $SSE$  по  $\mathbf{w}$  к нулю и решая полученные уравнения, получим формулу для коэффициентов полиномиальной регрессии

$$\mathbf{w} = (\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T \mathbf{Y}$$

Таким образом, задача полиномиальной регрессии решается с помощью методов множественной регрессии, если переменные  $X, X^2, X^3, \dots$  трактуются как отдельные независимые переменные в модели множественной регрессии.



# Линейная регрессия общего вида

Полиномиальную регрессию можно обобщить на случай [линейной регрессии общего вида](#), когда восстанавливается зависимость переменной  $\mathbf{Y}$  от другой или нескольких других переменных (признаков)  $\mathbf{X} \in \mathbb{R}^d$  с линейной зависимостью от неизвестных коэффициентов  $\mathbf{w} = (w_1, \dots, w_m) \in \mathbb{R}^m$  вида:

$$\mathbf{Y} = f(\mathbf{X}, \mathbf{w}) = \sum_{k=1}^m w_k f_k(\mathbf{X}),$$

где  $f_1(\mathbf{X}), \dots, f_m(\mathbf{X})$  – некоторые базисные функции. В качестве базисных функций могут рассматриваться различные полиномы, сплайны, радиальные базисные функции, вейвлеты и т.п.

Пусть даны значения независимых переменных  $\mathbf{x}_i \in \mathbb{R}^d, i = \overline{1, n}$  и соответствующие значения зависимой переменной  $y_i \in \mathbb{R}, i = \overline{1, n}$ . Введем матричные обозначения

$$\mathbf{F}(\mathbf{X}) = \begin{pmatrix} f_1(\mathbf{x}_1) & \dots & f_m(\mathbf{x}_1) \\ \dots & \dots & \dots \\ f_1(\mathbf{x}_n) & \dots & f_m(\mathbf{x}_n) \end{pmatrix}, \quad \mathbf{Y} = \begin{pmatrix} y_1 \\ \dots \\ y_n \end{pmatrix}.$$



# Коэффициенты линейной регрессии общего вида

Матрица  $\mathbf{F}(\mathbf{X})$  размерами  $n \times m$  имеет нелинейную зависимость общего вида от значений признаков  $\mathbf{X}$ , но функция регрессии  $f(\mathbf{X}, \mathbf{w})$  является линейной относительно коэффициентов регрессии (весов)  $\mathbf{w}$ :

$$\hat{\mathbf{Y}} = f(\mathbf{X}, \mathbf{w}) = \mathbf{F}(\mathbf{X}) \mathbf{w},$$

поэтому в случае максимальности ранга матрицы  $\mathbf{F}$  коэффициенты регрессии  $\mathbf{w}$  могут быть определены методом наименьших квадратов аналогично случаю полиномиальной регрессии:

$$\min_{\mathbf{w}} SSE = \min_{\mathbf{w}} \|\boldsymbol{\varepsilon}\|^2 = \min_{\mathbf{w}} \left\| \mathbf{Y} - \hat{\mathbf{Y}} \right\|^2 = \min_{\mathbf{w}} \left\| \mathbf{Y} - \mathbf{F}(\mathbf{X}) \mathbf{w} \right\|^2$$

Приравнивая градиент  $SSE$  по  $\mathbf{w}$  к нулю и решая полученные уравнения, получим формулу для коэффициентов линейной регрессии общего вида

$$\mathbf{w} = (\mathbf{F}^T \mathbf{F})^{-1} \mathbf{F}^T \mathbf{Y}$$

Таким образом, задача линейной регрессии общего вида также решается с помощью методов множественной регрессии, если переменные  $f_1(\mathbf{X}), \dots, f_m(\mathbf{X})$  трактуются как отдельные независимые переменные в модели множественной регрессии.



# Псевдообратная матрица

Матрица  $\mathbf{F}$  имеет размеры  $n \times m$ , где, вообще говоря,  $n > m$ , матрица  $\mathbf{F}^T$  имеет размеры  $m \times n$ , а матрицы  $\mathbf{F}^T\mathbf{F}$  и  $(\mathbf{F}^T\mathbf{F})^{-1}$  – размеры  $m \times m$ .

Матрица  $\mathbf{F}^+ = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T$  в формуле весов линейной регрессии общего вида называется **псевдообратной матрицей** (или матрицей Мура–Пенроуза):

$$\mathbf{F}^+\mathbf{F} = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{F} = \mathbf{1}_m,$$

где  $\mathbf{1}_m$  – единичная  $m \times m$ -матрица. Матрица  $\mathbf{F}\mathbf{F}^+ = \mathbf{F}(\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T$  (размерами  $n \times n$ ) называется **проекционной матрицей**.

Значение функции  $SSE$  для построенного вектора весов  $\mathbf{w}$  будет равно

$$SSE = \|\mathbf{Y} - \mathbf{F}\mathbf{F}^+\mathbf{Y}\|^2$$

Если столбцы матрицы  $\mathbf{F}$  линейно-зависимы (случай мультиколлинеарности), то матрица  $\mathbf{F}^T\mathbf{F}$  будет вырожденой и обратная матрица  $(\mathbf{F}^T\mathbf{F})^{-1}$  не существует.

Если же столбцы матрицы  $\mathbf{F}$  будут близки к линейной зависимости, то обращение матрицы  $\mathbf{F}^T\mathbf{F}$  будет сложной вычислительной задачей.



# Проблема мультиколлинеарности

Если определитель матрицы  $\mathbf{F}^T \mathbf{F}$  близок к нулю (имеются собственные числа близкие к нулю), то

- решение задачи линейной регрессии общего вида  $\mathbf{w}$  становится неустойчивым и неинтерпретируемым и может содержать слишком большие компоненты  $w_j$  различных знаков
- на обучающих данных функция ошибки  $SSE = \|\mathbf{Y} - \mathbf{F}\mathbf{w}\|^2$  может иметь малые значения, а на контрольных (или новых) данных функция  $SSE = \|\mathbf{Y}' - \mathbf{F}'\mathbf{w}\|^2$  может принимать значительно большие значения (т.е. будет иметь место [переобучение](#))

Для устранения мультиколлинеарности (и переобучения) можно провести:

- [отбор признаков](#), а именно, отбрасывание тех признаков, которые могут оказаться линейно-зависимыми с другими признаками
- [регуляризацию](#) (накладываем дополнительные ограничения на норму вектора весов  $\mathbf{w}$  вида  $\|\mathbf{w}\| \leq \alpha$ )
- [преобразование признаков](#) (отображение входных данных на новое пространство признаков)



Для некоторых сложных типов данных, таких, как текст, последовательности, изображения и т.п., обычно из данных извлекают или конструируют набор признаков в виде многомерных векторов, которые представляют экземпляры данных. Иными словами, используют отображение  $\phi$ , которое заданному экземпляру данных  $\mathbf{x}$  (например, последовательности) сопоставляет векторное представление  $\phi(\mathbf{x})$ . Если входные данные представляют собой числовую матрицу и требуется выявить нелинейные взаимосвязи между признаками, то для нелинейного отображения  $\phi$  вектор  $\phi(\mathbf{x})$  представляет собой вектор нелинейных признаков в соответствующем пространстве высокой размерности.

Будем использовать термин **входное пространство** для входных данных  $\mathbf{x}$  и термин **пространство признаков** для пространства отображенных векторов  $\phi(\mathbf{x})$ . Отображение в пространство признаков позволяет применять к данным различные численные методы, однако получающееся пространство признаков может иметь очень высокую размерность.



# Ядерная матрица

Ядерные методы избегают явного преобразования каждой точки  $\mathbf{x}$  входного пространства в отображаемую точку  $\phi(\mathbf{x})$  пространства признаков. Вместо этого входные объекты представляются через  $n \times n$  попарных значений сходства. Функция сходства, называемая **ядром** (kernel), выбирается таким образом, чтобы она представляла скалярное произведение в некотором пространстве признаков высокой размерности, однако она может быть вычислена без явного построения отображения  $\phi(\mathbf{x})$ .

Пусть  $\mathcal{I}$  обозначает входное пространство и  $\mathbf{D} \subset \mathcal{I}$  – набор данных из  $n$  объектов  $\mathbf{x}_i$ ,  $i = 1, \dots, n$  входного пространства. Значения сходства между парами точек в  $\mathbf{D}$  представляются в виде **ядерной матрицы** размеров  $n \times n$ :

$$K = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \dots & K(\mathbf{x}_1, \mathbf{x}_n) \\ \dots & \dots & \dots \\ K(\mathbf{x}_n, \mathbf{x}_1) & \dots & K(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix},$$

где  $K : \mathcal{I} \times \mathcal{I} \rightarrow \mathbb{R}$  – **ядерная функция** на любых двух точках входного пространства.



# Ядерные функции

Дополнительно требуется, чтобы функция  $K$  представляла собой скалярное произведение в некотором пространстве признаков. То есть для любой пары точек  $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{I}$  ядерная функция  $K$  должна удовлетворять условию

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j),$$

где  $\phi : \mathcal{I} \rightarrow \mathcal{F}$  – это отображение из входного пространства  $\mathcal{I}$  в пространство признаков  $\mathcal{F}$ . Это означает, что можно вычислить скалярное произведение в пространстве признаков, используя исходное представление  $\mathbf{x}$  без обращения к отображению  $\phi(\mathbf{x})$ . Поэтому не любая функция может быть использована в качестве ядерной функции.

В качестве ядерных функций, в частности, могут рассматриваться:

- однородные полиномиальные ядра  $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y})^q$
- неоднородные полиномиальные ядра  $K(\mathbf{x}, \mathbf{y}) = (c + \mathbf{x}^T \mathbf{y})^q, c \geq 0$
- гауссовые ядра (Gaussian radial basis function)  $K(\mathbf{x}, \mathbf{y}) = \exp\left\{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}\right\}$

# Линейное ядро для набора Ирисы

Рассмотрим линейную ядерную функцию

$$K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y}) = \mathbf{x}^T \mathbf{y},$$

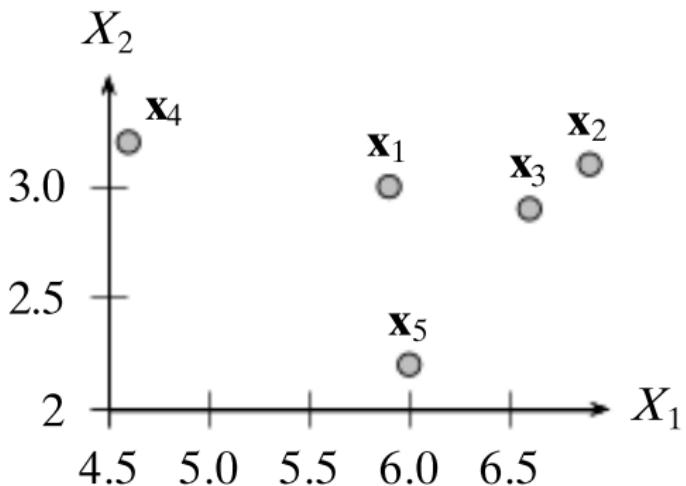
порождаемую тождественным отображением  $\phi : \mathbf{x} \rightarrow \mathbf{x}$ .

Рассмотрим первые 5 точек двумерного набора Ирисы:

$$\mathbf{x}_1 = \begin{bmatrix} 5.9 \\ 3.0 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 6.9 \\ 3.1 \end{bmatrix},$$

$$\mathbf{x}_3 = \begin{bmatrix} 6.6 \\ 2.9 \end{bmatrix}, \quad \mathbf{x}_4 = \begin{bmatrix} 4.6 \\ 3.2 \end{bmatrix},$$

$$\mathbf{x}_5 = \begin{bmatrix} 6.0 \\ 2.2 \end{bmatrix}$$





# Ядерная матрица для набора Ирисы

Ядерная матрица для этого набора точек и линейного ядра показана ниже:

<b>K</b>	<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>x<sub>4</sub></b>	<b>x<sub>5</sub></b>
<b>x<sub>1</sub></b>	43.81	50.01	47.64	36.74	42.00
<b>x<sub>2</sub></b>	50.01	57.22	54.53	41.66	48.22
<b>x<sub>3</sub></b>	47.64	54.53	51.97	39.64	45.98
<b>x<sub>4</sub></b>	36.74	41.66	39.64	31.40	34.64
<b>x<sub>5</sub></b>	42.00	48.22	45.98	34.64	40.84

Например, для элемента  $K(x_1, x_2)$  имеем

$$K(x_1, x_2) = \mathbf{x}_1^T \mathbf{x}_2 = 5.9 \cdot 6.9 + 3.0 \cdot 3.1 = 50.01$$



# Квадратичное ядро для набора Ирисы

Рассмотрим квадратичное отображение  $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ , которое отображает точку  $\mathbf{x} = (x_1, x_2)^T$  следующим образом

$$\phi : (x_1, x_2)^T \rightarrow \left( x_1^2, x_2^2, \sqrt{2}x_1x_2 \right)^T$$

Скалярное произведение между образами двух точек  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2$  равно

$$\phi(\mathbf{x})^T \phi(\mathbf{y}) = x_1^2 y_1^2 + x_2^2 y_2^2 + 2x_1 y_1 x_2 y_2 = (x_1 y_1 + x_2 y_2)^2 = (\mathbf{x}^T \mathbf{y})^2 = K(\mathbf{x}, \mathbf{y})$$

Скалярное произведение  $\phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2)$  точек  $\phi(\mathbf{x}_1)$  и  $\phi(\mathbf{x}_2)$  в пространстве признаков может быть вычислено непосредственно:

$$\phi(\mathbf{x}_1) = \left( 5.9^2, 3^2, \sqrt{2} \cdot 5.9 \cdot 3 \right)^T = (34.81, 9, 25.03)^T$$

$$\phi(\mathbf{x}_2) = \left( 6.9^2, 3.1^2, \sqrt{2} \cdot 6.9 \cdot 3.1 \right)^T = (47.61, 9.61, 30.25)^T$$

$$\phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2) = 34.81 \cdot 47.61 + 9 \cdot 9.61 + 25.03 \cdot 30.25 = 2501$$

Оценка ядерной функции  $K(\mathbf{x}_1, \mathbf{x}_2)$  позволяет получить то же значение:

$$K(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1^T \mathbf{x}_2)^2 = 50.01^2 = 2501$$



# Ядерная регрессия

Пусть  $\phi$  – это отображение из входного пространства в пространство признаков, то есть каждая точка  $\phi(\mathbf{x}_i)$  в пространстве признаков является отображением входной точки  $\mathbf{x}_i$ . Добавим фиксированное значение 1 в качестве первого элемента к  $\phi(\mathbf{x}_i)$  и получим дополненную (augmented) точку  $\tilde{\phi}(\mathbf{x}_i)^T = \begin{pmatrix} 1 & \phi(\mathbf{x}_i)^T \end{pmatrix}$ .

Обозначим через  $\tilde{\mathbf{D}}_\phi$  дополненный набор данных в признаковом пространстве, состоящий из точек  $\tilde{\phi}(\mathbf{x}_i)$  для  $i = 1, \dots, n$ . Дополненная ядерная функция в пространстве признаков равна

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = \tilde{\phi}(\mathbf{x}_i)^T \tilde{\phi}(\mathbf{x}_j) = 1 + \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = 1 + K(\mathbf{x}_i, \mathbf{x}_j),$$

где  $K(\mathbf{x}_i, \mathbf{x}_j)$  – обычная ядерная функция. Пусть  $\mathbf{Y}$  – наблюдаемый вектор откликов. Прогнозируемый ядерной регрессией вектор откликов  $\hat{\mathbf{Y}}$  равен

$$\hat{\mathbf{Y}} = \tilde{\mathbf{D}}_\phi \tilde{\mathbf{w}},$$

где  $\tilde{\mathbf{w}}$  – дополненный вектор весов в пространстве признаков.



# Задача минимизации для ядерной регрессии

Применение ядер (kernel) позволяет обобщить линейную регрессию на нелинейный случай, то есть найти нелинейную аппроксимацию данных для минимизации квадратичной ошибки наряду с регуляризацией. Функция  $\phi(\mathbf{x}_i)$  отображает входную точку  $\mathbf{x}_i$  в пространство признаков.

Для регуляризованной регрессии решается следующая задача минимизации в пространстве признаков:

$$\min_{\tilde{\mathbf{w}}} J(\tilde{\mathbf{w}}), \quad J(\tilde{\mathbf{w}}) = \left\| \mathbf{Y} - \hat{\mathbf{Y}} \right\|^2 + \alpha \left\| \tilde{\mathbf{w}} \right\|^2 = \left\| \mathbf{Y} - \tilde{\mathbf{D}}_\phi \tilde{\mathbf{w}} \right\|^2 + \alpha \left\| \tilde{\mathbf{w}} \right\|^2$$

Приравнивая градиент  $J(\tilde{\mathbf{w}})$  по  $\tilde{\mathbf{w}}$  к нулю, получим

$$\frac{\partial}{\partial \tilde{\mathbf{w}}} J(\tilde{\mathbf{w}}) = 0 \Rightarrow \tilde{\mathbf{w}} = \tilde{\mathbf{D}}_\phi^T \left( \frac{1}{\alpha} (\mathbf{Y} - \tilde{\mathbf{D}}_\phi \tilde{\mathbf{w}}) \right) \Rightarrow \tilde{\mathbf{w}} = \tilde{\mathbf{D}}_\phi^T \mathbf{c}, \quad \mathbf{c} = \frac{1}{\alpha} (\mathbf{Y} - \tilde{\mathbf{D}}_\phi \tilde{\mathbf{w}})$$

То есть вектор весов  $\tilde{\mathbf{w}}$  представляет собой линейную комбинацию точек в пространстве признаков, а вектор  $\mathbf{c}$  представляет собой вектор коэффициентов смещения точек.

# Задача минимизации для ядерной регрессии

Продолжим вычисления для вектора  $\mathbf{c}$ :

$$\begin{aligned}\mathbf{c} &= \frac{1}{\alpha} (\mathbf{Y} - \tilde{\mathbf{D}}_\phi \tilde{\mathbf{w}}) \Rightarrow \alpha \mathbf{c} = \mathbf{Y} - \tilde{\mathbf{D}}_\phi \tilde{\mathbf{w}} = \mathbf{Y} - \tilde{\mathbf{D}}_\phi \tilde{\mathbf{D}}_\phi^T \mathbf{c} \Rightarrow \\ &\Rightarrow (\tilde{\mathbf{D}}_\phi \tilde{\mathbf{D}}_\phi^T + \alpha \mathbf{I}_n) \mathbf{c} = \mathbf{Y} \Rightarrow \mathbf{c} = (\tilde{\mathbf{D}}_\phi \tilde{\mathbf{D}}_\phi^T + \alpha \mathbf{I}_n)^{-1} \mathbf{Y}\end{aligned}$$

Итак, получаем для вектора  $\mathbf{c}$  решение в виде

$$\mathbf{c} = (\tilde{\mathbf{K}} + \alpha \mathbf{I}_n)^{-1} \mathbf{Y},$$

где  $\mathbf{I}_n \in \mathbb{R}^{n \times n}$  – единичная матрица размера  $n \times n$  и  $\tilde{\mathbf{K}}$  – дополненная ядерная матрица, так как

$$\tilde{\mathbf{D}}_\phi \tilde{\mathbf{D}}_\phi^T = \left\{ \tilde{\phi}(\mathbf{x}_i)^T \tilde{\phi}(\mathbf{x}_j) \right\}_{i,j=1}^n = \left\{ \tilde{K}(\mathbf{x}_i, \mathbf{x}_j) \right\}_{i,j=1}^n = \tilde{\mathbf{K}}$$



# Прогнозируемый отклик в ядерной регрессии

Выражение для прогнозируемого отклика равно

$$\hat{\mathbf{Y}} = \tilde{\mathbf{D}}_\phi \tilde{\mathbf{w}} = \tilde{\mathbf{D}}_\phi \tilde{\mathbf{D}}_\phi^T \mathbf{c} = \tilde{\mathbf{D}}_\phi \tilde{\mathbf{D}}_\phi^T (\tilde{\mathbf{K}} + \alpha \mathbf{I})^{-1} \mathbf{Y} = \tilde{\mathbf{K}} (\tilde{\mathbf{K}} + \alpha \mathbf{I})^{-1} \mathbf{Y}$$

Условие  $\alpha > 0$  гарантирует, что обратная матрица существует, что является еще одним преимуществом использования ядерной гребневой регрессии в дополнение к регуляризации.

Для предсказания значения ядерной регрессии на новой точке  $\mathbf{z}$  вычисляется вектор  $\tilde{\mathbf{K}}_{\mathbf{z}}$ , включающий дополненные ядерные значения  $\mathbf{z}$  по всем точкам данных в наборе  $\mathbf{D}$ , и берется его скалярное произведение с вектором коэффициентов  $\mathbf{c}$ , чтобы получить прогнозируемый отклик  $\hat{y}$ :

$$\begin{aligned}\hat{y} &= \tilde{\phi}(\mathbf{z})^T \tilde{\mathbf{w}} = \tilde{\phi}(\mathbf{z})^T \tilde{\mathbf{D}}_\phi^T \mathbf{c} = \tilde{\phi}(\mathbf{z})^T \left( \sum_{i=1}^n c_i \tilde{\phi}(\mathbf{x}_i) \right) = \\ &= \sum_{i=1}^n c_i \tilde{\phi}(\mathbf{z})^T \tilde{\phi}(\mathbf{x}_i) = \sum_{i=1}^n c_i \tilde{K}(\mathbf{z}, \mathbf{x}_i) = \mathbf{c}^T \tilde{\mathbf{K}}_z\end{aligned}$$



# Алгоритм ядерной регрессии

Входными данными для алгоритма ядерной регрессии являются матрица входных данных  $\mathbf{D}$ , вектор откликов  $\mathbf{Y}$  для точек набора  $\mathbf{D}$ , ядерная функция  $K$ , коэффициент регуляризации  $\alpha > 0$ .

Kernel-Regression ( $\mathbf{D}, \mathbf{Y}, K, \alpha$ ):

- 1  $\mathbf{K} \leftarrow \{K(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j=1,\dots,n}$  // стандартная ядерная матрица
- 2  $\tilde{\mathbf{K}} \leftarrow \mathbf{K} + 1$  // дополненная ядерная матрица
- 3  $\mathbf{c} \leftarrow (\tilde{\mathbf{K}} + \alpha \cdot \mathbf{I})^{-1} \mathbf{Y}$  // коэффициенты смещения
- 4  $\hat{\mathbf{Y}} \leftarrow \tilde{\mathbf{K}} \mathbf{c}$

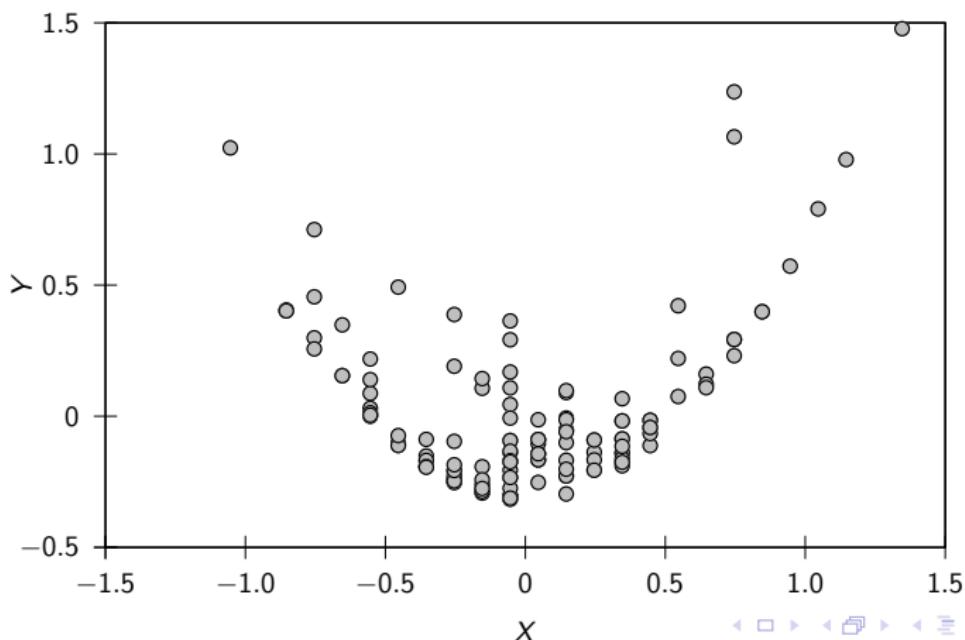
Kernel-Regression-Prediction ( $\mathbf{z}, \mathbf{D}, K, \mathbf{c}$ ):

- 5  $\tilde{\mathbf{K}}_{\mathbf{z}} \leftarrow \{1 + K(\mathbf{z}, \mathbf{x}_i)\}_{\forall \mathbf{x}_i \in \mathbf{D}}$
- 6  $\hat{y} \leftarrow \mathbf{c}^T \tilde{\mathbf{K}}_{\mathbf{z}}$

# Пример ядерной регрессии для набора Ирисы

Рассмотрим нелинейный набор данных Iris, полученный посредством нелинейного преобразования атрибутов длины чашелистика ( $A_1$ ) и ширины чашелистика ( $A_2$ ):

$$X = A_2, \quad Y = 0.2 A_1^2 + A_2^2 + 0.1 A_1 A_2$$



# Пример ядерной регрессии для набора Ирисы



Переменная  $Y$  рассматривается как отклик, а переменная  $X$  – как независимая. Точки на рисунке показывают четкую квадратичную (нелинейную) связь между ними.

Линейная аппроксимация дает

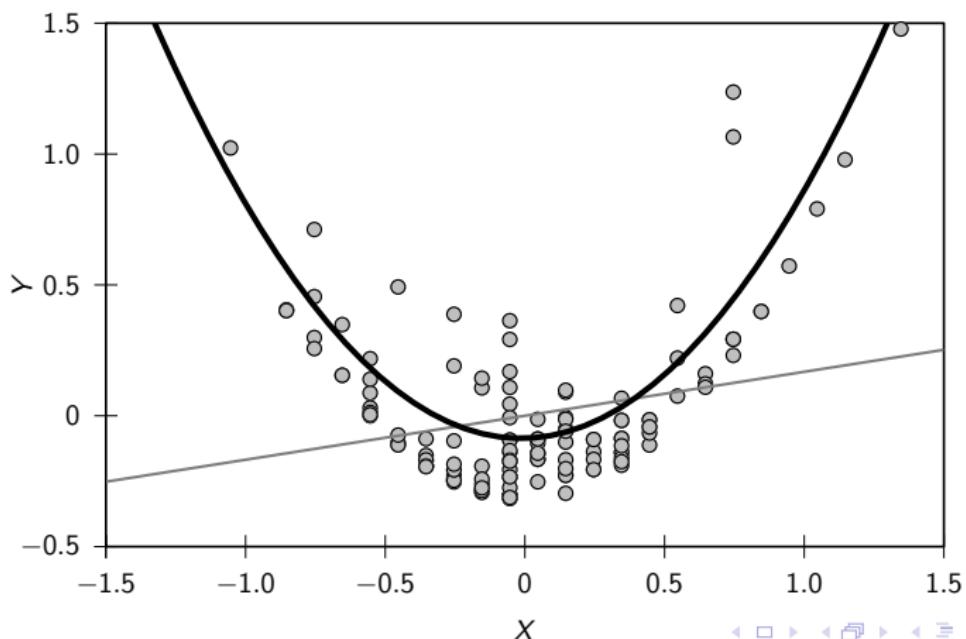
$$\hat{Y} = 0.168 X$$

Используя квадратичное (неоднородное) ядро над  $X$ , состоящее из постоянной (1), линейных членов ( $X$ ) и квадратичных членов ( $X^2$ ) с параметром  $\alpha = 0.1$ :

$$\hat{Y} = -0.086 + 0.168 X + 0.922 X^2$$

# Пример ядерной регрессии для набора Ирисы

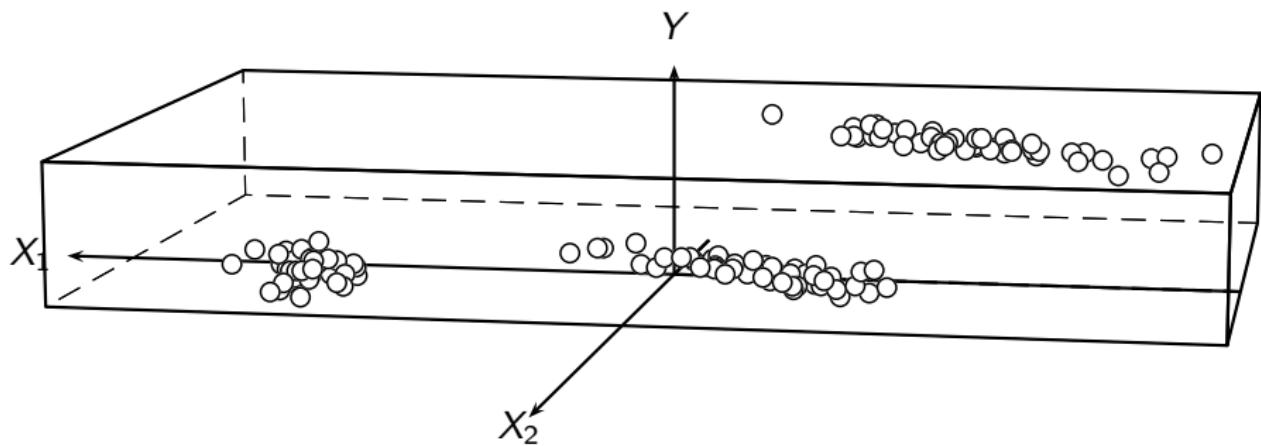
Серой линией показана линейная, а черной линией – квадратичная аппроксимация. Ошибка SSE составляет 13.82 для линейной аппроксимации и 4.33 для квадратичного ядра. Квадратичное ядро (как и ожидалось) дает гораздо лучшее соответствие данным.



# Пример ядерной гребневой регрессии (Ирисы)



Рассмотрим набор данных Ирисы для двух главных компонент  $X_1$  и  $X_2$ . Переменная отклика  $Y$  является бинарной со значением 1, соответствующим классу Iris-virginica (точки вверху справа со значением  $Y = 1$ ), и значением 0, соответствующим классам Iris-setosa и Iris-versicolor (две другие группы точек со значением  $Y = 0$ ).

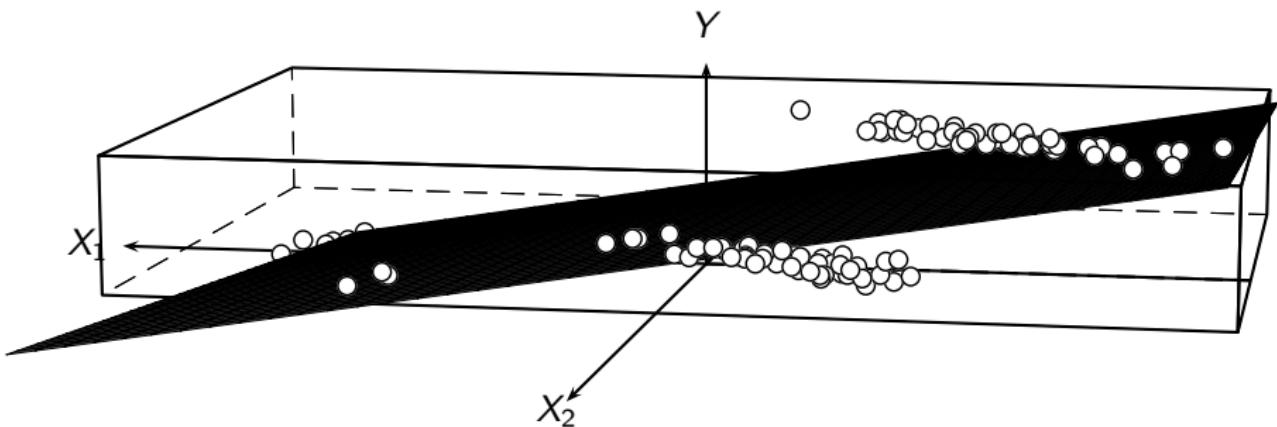


# Пример ядерной гребневой регрессии (Ирисы)



На рисунке показана подобранный плоскость регрессии с использованием линейного ядра со значением параметра регуляризации  $\alpha = 0.01$ :

$$\hat{Y} = 0.333 - 0.167 X_1 + 0.074 X_2$$



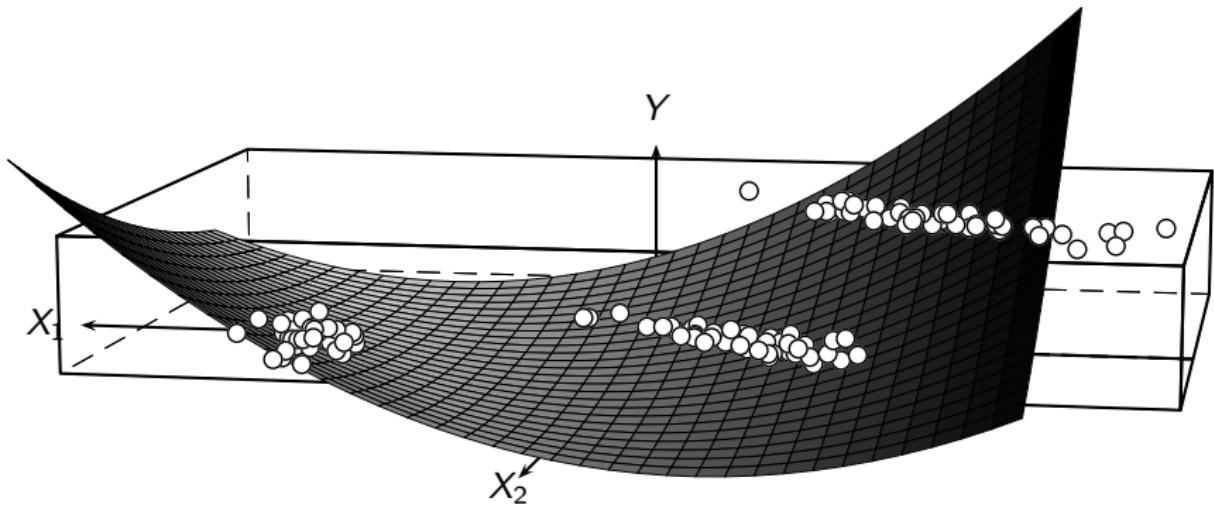
Ошибка SSE для линейной модели составляет 15.47

# Пример ядерной гребневой регрессии (Ирисы)



На рисунке ниже показана подобранная нелинейная модель, когда используется неоднородное квадратичное ядро с  $\alpha = 0.01$ :

$$\hat{Y} = -0.03 - 0.167 X_1 - 0.186 X_2 + 0.092 X_1^2 + 0.1 X_1 X_2 + 0.029 X_2^2$$



Ошибка SSE для квадратичного ядра равна 8.44, что указывает на лучшее соответствие обучающим данным.



# Обобщенная линейная модель

Линейная регрессия – не единственный способ оценивания зависимости между признаками. Например, если имеются взаимно-однозначные преобразования (например, монотонные функции) признаков  $x$  и  $y$   $f(x)$  и  $g(y)$ , то коэффициенты для модели вида

$$y(x) = g^{-1}(a f(x) + b + \varepsilon),$$

ищутся точно так же, как и для линейной модели. На практике просто ко всем признакам  $x$  и  $y$  из выборки следует заранее применить преобразования  $f(x)$  и  $g(y)$ , после чего искать коэффициенты уже для получившейся линейной модели. Например, популярной моделью является логистическая регрессия или логит-регрессия, которая используется для прогнозирования вероятности возникновения некоторого события путём подгонки данных к логистической кривой

$$f(z) = \frac{1}{1 + e^{-z}}.$$



# Логистическая регрессия: набор данных

Целью логистической регрессии является прогнозирование вероятностей значений зависимой переменной в зависимости от независимых переменных (признаков). На самом деле, логистическая регрессия является подходом к решению другой задачи машинного обучения – задачи классификации.

Рассмотрим набор данных из  $d$  (независимых) признаков  $X_1, X_2, \dots, X_d$  и бинарной переменной  $Y$ , принимающей только два значения, а именно, 0 и 1. Таким образом, имеется обучающий набор данных  $\mathbf{D}$ , состоящий из  $n$  точек  $\mathbf{x}_i \in \mathbb{R}^d$ , и соответствующих наблюдаемых значений  $y_i \in \{0, 1\}$ . Дополним матрицу данных  $\mathbf{D}$  добавлением нового признака  $X_0$ , принимающего фиксированное значение 1 в каждой точке. Вектор  $\tilde{\mathbf{x}} = (1, x_1, \dots, x_d)^T \in \mathbb{R}^{d+1}$  обозначает дополненную точку, а многомерный случайный вектор  $\tilde{\mathbf{X}}$ , включающий все независимые признаки задан как  $\tilde{\mathbf{X}} = (X_0, X_1, \dots, X_d)^T$ . Дополненный обучающий набор  $\tilde{\mathbf{D}}$  содержит  $n$  дополненных точек  $\tilde{\mathbf{x}}_i$  с метками классов  $y_i$  для  $i = \overline{1, n}$ .



# Логистическая функция

Так как зависимая переменная  $Y$  принимает только два значения, функция вероятности равна

$$\mathbb{P}[Y = 1 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = \pi(\tilde{\mathbf{x}}), \quad \mathbb{P}[Y = 0 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = 1 - \pi(\tilde{\mathbf{x}}),$$

где  $\pi(\tilde{\mathbf{x}})$  обозначает вероятность того, что  $Y = 1$  при условии, что  $\tilde{\mathbf{X}} = \tilde{\mathbf{x}}$ .  
Ожидаемое значение  $Y$  при условии  $\tilde{\mathbf{X}} = \tilde{\mathbf{x}}$  равно

$$\mathbb{E}[Y \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = 1 \cdot \mathbb{P}[Y = 1 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] + 0 \cdot \mathbb{P}[Y = 0 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = \pi(\tilde{\mathbf{x}})$$

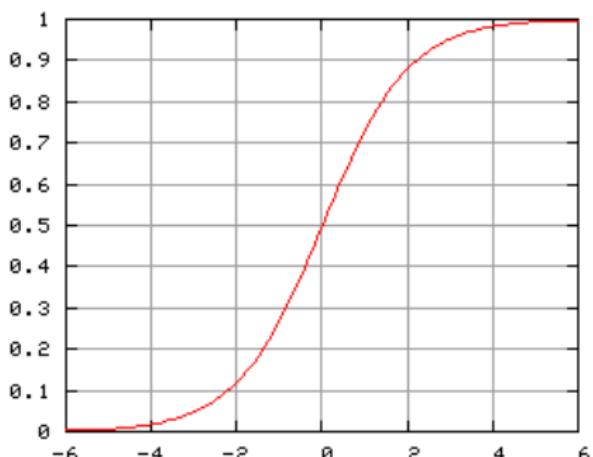
Следовательно, в логистической регрессии вместо прямого прогнозирования значения зависимой переменной  $Y$  цель состоит в прогнозировании вероятности  $\mathbb{P}[Y = 1 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}]$ . Так как  $\mathbb{P}[Y = 1 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}]$  – вероятность, нецелесообразно аппроксимировать вероятность при помощи линейной функции, принимающей произвольные значения от  $-\infty$  до  $+\infty$ . Поэтому необходима функция, принимающая значения на интервале  $[0, 1]$ .

В качестве такой функции используется **логистическая функция** (или **сигмоида**)

$$\theta(z) = \frac{1}{1 + \exp(-z)} = \frac{\exp(z)}{1 + \exp(z)}$$

# Свойства логистической функции

Логистическая функция  $\theta(z)$  принимает любые значения от 0 до 1:



Логистическая функция обладает следующими основными свойствами:

$$\theta(-\infty) = \frac{1}{1 + \exp(\infty)} = \frac{1}{\infty} = 0,$$

$$\theta(+\infty) = \frac{1}{1 + \exp(-\infty)} = \frac{1}{1} = 1,$$

$$\theta(0) = \frac{1}{1 + \exp(0)} = \frac{1}{2} = 0.5,$$

$$1 - \theta(z) = 1 - \frac{\exp(z)}{1 + \exp(z)} = \theta(-z)$$

Значение  $z = 0$  может использоваться как пороговое значение, т.к. для  $z > 0$   $\theta(z) > 0.5$  и для  $z < 0$   $\theta(z) < 0.5$ .



# Бинарная логистическая регрессия

С использованием логистической функции модель бинарной логистической регрессии определяется так:

$$\mathbb{P}[Y = 1 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = \pi(\tilde{\mathbf{x}}) = \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) = \frac{\exp(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})}{1 + \exp(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})},$$

где  $\tilde{\mathbf{w}} = (w_0, w_1, \dots, w_d)^T$ ,  $\tilde{\mathbf{w}}^T \tilde{\mathbf{x}} = w_0x_0 + w_1x_1 + \dots + w_dx_d$ . Таким образом, вероятность того, что  $Y = 1$ , является значением логистической функции от  $\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}$ . С другой стороны, вероятность того, что  $Y = 0$ , равна

$$\mathbb{P}[Y = 0 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = 1 - \mathbb{P}[Y = 1 \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = \theta(-\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) = \frac{1}{1 + \exp(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})}$$

Комбинируя эти два случая в полную модель логистической регрессии, получим

$$\mathbb{P}[Y \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})^Y \theta(-\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})^{1-Y},$$

так как  $Y$  является случайной величиной Бернулли, которая принимает значения 0 или 1.



# Оценка правдоподобия

Пусть  $\tilde{\mathbf{D}}$  – дополненный обучающий набор данных, содержащий  $n$  дополненных точек  $\tilde{\mathbf{x}}_i$  с метками классов  $y_i$ . Пусть  $\tilde{\mathbf{w}} = (w_0, w_1, \dots, w_d)^T$  – дополненный вектор весов, где  $w_0 = b$  обозначает ожидаемое смещение, а  $w_i$  – ожидаемый вес признака  $X_i$ .

**Правдоподобие** (likelihood) определяется как вероятность наблюдаемых данных для вектора параметров  $\tilde{\mathbf{w}}$ . Предполагается, что все бинарные переменные  $y_i$  все являются независимыми. Тогда правдоподобие наблюдаемых данных для вектора параметров  $\tilde{\mathbf{w}}$  определяется как

$$L(\tilde{\mathbf{w}}) = \mathbb{P}[Y \mid \tilde{\mathbf{w}}] = \prod_{i=1}^n \mathbb{P}[y_i \mid \tilde{\mathbf{x}}_i] = \prod_{i=1}^n \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)^{y_i} \theta(-\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)^{1-y_i}$$

Вместо максимизации правдоподобия удобнее максимизировать **логарифм правдоподобия**, чтобы перейти от произведения к сумме:

$$\ln L(\tilde{\mathbf{w}}) = \sum_{i=1}^n [y_i \ln \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i) + (1 - y_i) \ln \theta(-\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)]$$



# Кросс-энтропия

Кросс-энтропия (или перекрестная энтропия) для двух дискретных распределений  $p$  и  $q$  над одним и тем же вероятностным пространством определяется по формуле

$$H(p, q) = - \sum_x p(x) \ln q(x)$$

Логарифм правдоподобия, взятый с обратным знаком и деленный на количество точек в наборе данных, может рассматриваться как кросс-энтропия для распределения  $q$ , задаваемого логистической функцией с параметрами  $\tilde{\mathbf{w}}$ , и эмпирического распределения  $p$  в наборе данных:

$$H(p, q) = -\frac{1}{n} \ln L(\tilde{\mathbf{w}}) = \frac{1}{n} \sum_{i=1}^n \left[ y_i \ln \left( \frac{1}{\theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)} \right) + (1 - y_i) \ln \left( \frac{1}{1 - \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)} \right) \right]$$

Таким образом, задача максимизации логарифма правдоподобия эквивалентна задаче минимизации кросс-энтропии.

Логарифм правдоподобия является выпуклой функцией, имеющей единственный глобальный максимум, поэтому для определения оптимальных параметров могут использоваться различные численные методы.



# Максимизация логарифма правдоподобия

Для максимизации логарифма подобия  $\ln L(\tilde{\mathbf{w}})$  могут быть применены такие методы, как:

- метод Ньютона ( $H$  – матрица Гесса функции  $\ln L(\tilde{\mathbf{w}})$ )

$$\tilde{\mathbf{w}}_{k+1} = \tilde{\mathbf{w}}_k + \alpha H^{-1}(\tilde{\mathbf{w}}_k) \nabla \ln L(\tilde{\mathbf{w}}_k), \alpha > 0$$

- метод градиентного подъема (используются все точки набора данных)

$$\tilde{\mathbf{w}}_{k+1} = \tilde{\mathbf{w}}_k + \alpha \nabla \ln L(\tilde{\mathbf{w}}_k) = \tilde{\mathbf{w}}_k + \alpha \sum_{i=1}^n \left( y^{(i)} - \theta(\tilde{\mathbf{w}}_k^T \tilde{\mathbf{x}}_i) \right) \tilde{\mathbf{x}}_i, \alpha > 0$$

- метод стохастического градиентного подъема (используются случайные выборки из набора данных)

Логистическая регрессия применяется для решения задач бинарной классификации следующим образом: объект  $x$  можно отнести к классу  $y = 1$ , если предсказанная моделью вероятность  $\mathbb{P}[y = 1 | x] \geq 0.5$ , и к классу  $y = 0$  в противном случае. Получающиеся при этом правила классификации являются линейными классификаторами.



## Метод градиентного подъема

Метод градиентного подъема использует градиент логарифма правдоподобия, который может быть получен вычислением частных производных  $\ln L(\tilde{\mathbf{w}})$  по  $\tilde{\mathbf{w}}$ :

$$\nabla(\tilde{\mathbf{w}}) = \frac{\partial}{\partial \tilde{\mathbf{w}}} \ln L(\tilde{\mathbf{w}}) = \frac{\partial}{\partial \tilde{\mathbf{w}}} \left\{ \sum_{i=1}^n [y_i \ln \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i) + (1 - y_i) \ln \theta(-\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)] \right\}$$

После упрощения получим, что  $\nabla(\tilde{\mathbf{w}}) = \sum_{i=1}^n [y_i - \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)] \tilde{\mathbf{x}}_i$

Метод градиентного подъема стартует с начальных весов  $\tilde{\mathbf{w}}^0$ . На каждом шаге  $t$  веса изменяются в направлении наиболее крутого подъема, задаваемого вектором градиента логарифма правдоподобия.

Если текущие веса  $\tilde{\mathbf{w}}^t$ , то следующая оценка равна  $\tilde{\mathbf{w}}^{t+1} = \tilde{\mathbf{w}}^t + \eta \nabla(\tilde{\mathbf{w}}^t)$ , где  $\eta > 0$  – коэффициент скорости обучения (learning rate). Коэффициент не должен быть слишком большим, иначе оценки будут сильно различаться от одной итерации к другой, и он не должен быть слишком маленьким, иначе сходимость займет много времени.

В точке максимума  $\tilde{\mathbf{w}}$  градиент будет равен нулю, т.е.  $\nabla(\tilde{\mathbf{w}}) = 0$ , как и требовалось.

# Стохастический градиентный подъем

Метод градиентного подъема вычисляет градиент, рассматривая все точки данных, поэтому его называют пакетный (batch) градиентный подъем.

Для больших наборов данных обычно намного быстрее вычислить градиент, рассматривая только одну (случайным образом выбранную) точку на каждом шаге. Такой метод называется стохастический градиентный подъем (SGA):

$$\nabla (\tilde{\mathbf{w}}, \tilde{\mathbf{x}}_i) = [y_i - \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)] \tilde{\mathbf{x}}_i$$

Если классификатор на основе логистической регрессии был обучен, то прогноз класса для любой новой дополненной точки  $\tilde{\mathbf{x}}$  получается по формулам:

$$\hat{y} = \begin{cases} 1, & \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) \geq 0.5 \\ 0, & \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) < 0.5 \end{cases}$$

# Алгоритм логистической регрессии с SGA



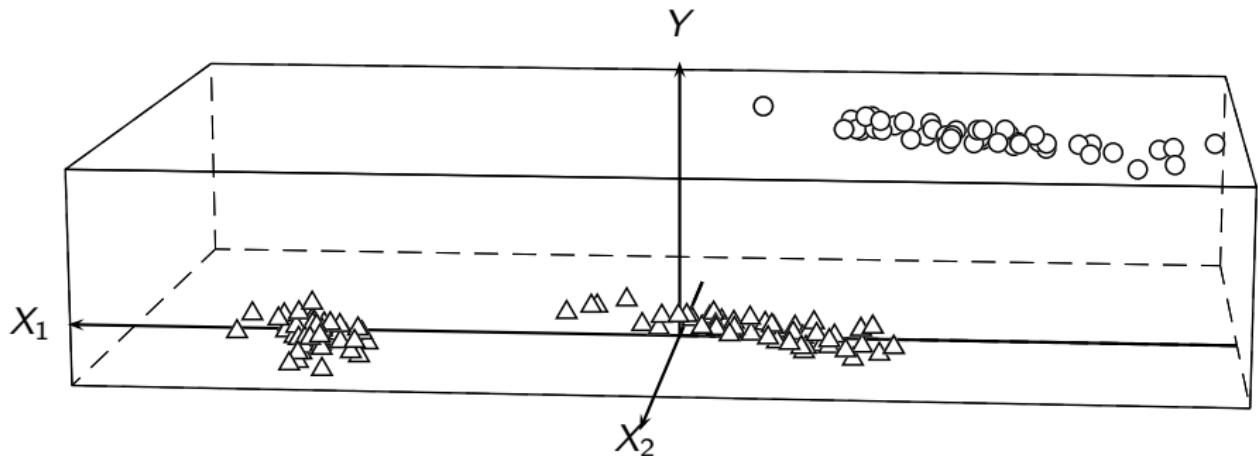
Входными данными для алгоритма логистической регрессии являются матрица входных данных  $\mathbf{D}$ , коэффициент скорости обучения  $\eta$ , точность  $\varepsilon$ .

LogisticRegression-SGA ( $\mathbf{D}, \eta, \varepsilon$ ):

```
1 foreach  $\mathbf{x}_i \in \mathbf{D}$  do  $\tilde{\mathbf{x}}_i^T \leftarrow (1 \ \mathbf{x}_i^T)$  // отображение в  $\mathbb{R}^{d+1}$ 
2  $t \leftarrow 0$  // счетчик шагов/итераций
3  $\tilde{\mathbf{w}}^0 \leftarrow (0, \dots, 0)^T \in \mathbb{R}^{d+1}$  // начальный вектор весов
4 repeat
5    $\tilde{\mathbf{w}} \leftarrow \tilde{\mathbf{w}}^t$  // сделать копию  $\tilde{\mathbf{w}}^t$ 
6   foreach  $\tilde{\mathbf{x}}_i \in \tilde{\mathbf{D}}$  in random order do
7      $\nabla(\tilde{\mathbf{w}}, \tilde{\mathbf{x}}_i) \leftarrow (y_i - \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i)) \cdot \tilde{\mathbf{x}}_i$  // градиент в точке  $\tilde{\mathbf{x}}_i$ 
8      $\tilde{\mathbf{w}} \leftarrow \tilde{\mathbf{w}} + \eta \cdot \nabla(\tilde{\mathbf{w}}, \tilde{\mathbf{x}}_i)$  // обновить оценку для  $\tilde{\mathbf{w}}$ 
9    $\tilde{\mathbf{w}}^{t+1} \leftarrow \tilde{\mathbf{w}}$  // обновить  $\tilde{\mathbf{w}}^{t+1}$ 
10   $t \leftarrow t + 1$ 
11 until  $\|\tilde{\mathbf{w}}^t - \tilde{\mathbf{w}}^{t-1}\| \leq \varepsilon$ 
```

# Классификация набора Ирисы

Рассмотрим в качестве признаков  $X_1, X_2$  две первые главные компоненты набора Ирисы, а в качестве зависимой бинарную переменную  $Y$ , представляющую тип ириса:  $Y = 1$  соответствует типу «iris-virginica»,  $Y = 0$  соответствует двум другим типам.



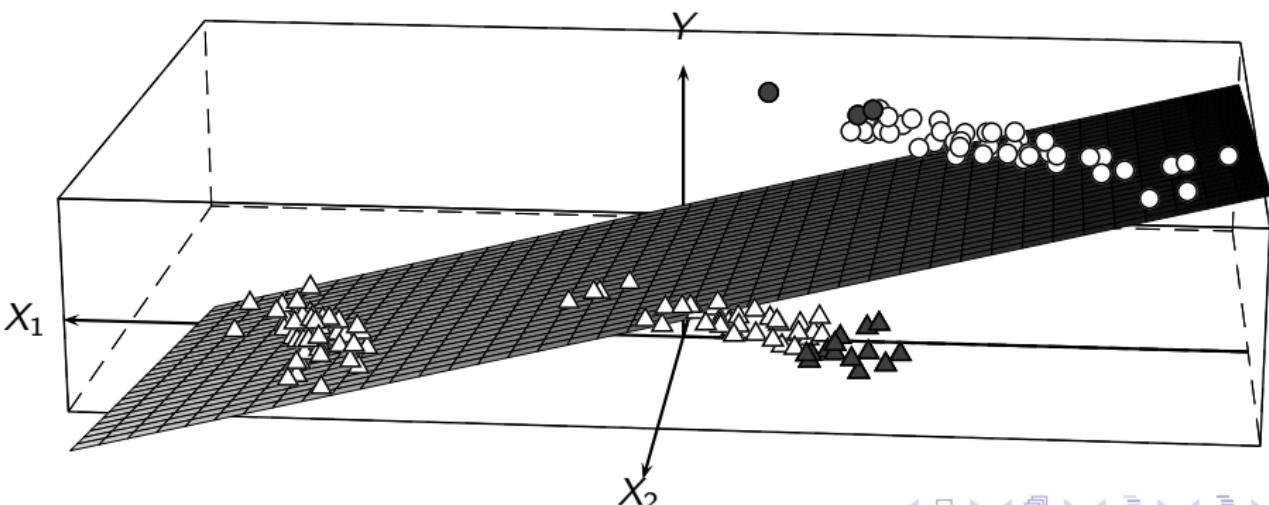
# Линейная регрессия для набора Ирисы

Плоскость наилучшего соответствия линейной регрессии имеет веса:

$$\tilde{\mathbf{w}} = (0.333, -0.167, 0.074)^T \Rightarrow \hat{y} = f(\tilde{\mathbf{x}}) = 0.333 - 0.167x_1 + 0.074x_2$$

Так как отклик  $Y$  бинарный, предсказываем класс  $y = 1$ , если  $f(\tilde{\mathbf{x}}) \geq 0.5$ , и класс  $y = 0$  в противном случае.

Линейная регрессия неверно классифицирует 17 точек, достигая точности (аккуратности) 88.7%.



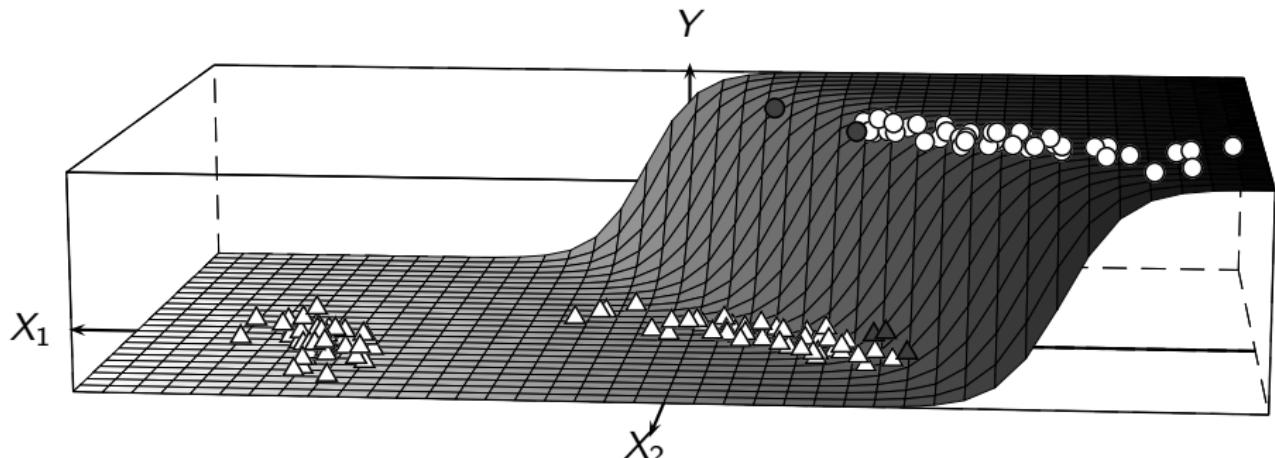
# Логистическая регрессия для набора Ирисы

Обученная логистическая модель характеризуется весами

$$\tilde{\mathbf{w}} = (w_0, w_1, w_2)^T = (-6.79, -5.07, -3.29),$$

$$\mathbb{P}[Y = 1 \mid \tilde{\mathbf{x}}] = \theta(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) = \frac{1}{1 + \exp(6.79 + 5.07x_1 + 3.29x_2)}$$

Для заданной точки  $\tilde{\mathbf{x}}$ , если  $\mathbb{P}[Y = 1 \mid \tilde{\mathbf{x}}] \geq 0.5$ , то прогнозируем  $\hat{y} = 1$ , в противном случае прогнозируем  $\hat{y} = 0$ . Логистическая регрессия ошибочно классифицирует только 5 точек, достигая точности (аккуратности) 96.7%.





# Многоклассовая логистическая регрессия

Обобщим логистическую регрессию на случай, когда отклик  $Y$  может принимать  $K$  различных номинальных категориальных (текстовых) значений, называемых классами, т.е.  $Y \in \{c_1, c_2, \dots, c_K\}$ .

Будем моделировать  $Y$  как  $K$ -мерную случайную величину Бернулли. Так как отклик  $Y$  может принимать только одно из  $K$  значений, используем однократное кодирование (one-hot encoding), чтобы отобразить каждое категориальное значение  $c_i$  в  $K$ -мерный бинарный вектор

$$\mathbf{e}_i = \left( \overbrace{0, \dots, 0}^{i-1}, 1, \overbrace{0, \dots, 0}^{K-i} \right)^T,$$

в котором  $i$ -й элемент равен единице  $e_{ii} = 1$ , а все остальные элементы равны нулю  $e_{ij} = 0$ ,  $i \neq j$ , то есть  $\sum_{j=1}^K e_{ij} = 1$ . Функция вероятности для  $\mathbf{Y}$  при условии  $\tilde{\mathbf{X}} = \tilde{\mathbf{x}}$  равна  $\mathbb{P} [\mathbf{Y} = \mathbf{e}_i | \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = \pi_i(\tilde{\mathbf{x}})$ ,  $i = 1, 2, \dots, K$ .

Имеется  $K$  неизвестных величин  $\pi_i(\tilde{\mathbf{x}})$ , таких, что:

$$\sum_{i=1}^K \pi_i(\tilde{\mathbf{x}}) = \sum_{i=1}^K \mathbb{P} [\mathbf{Y} = \mathbf{e}_i | \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = 1$$



# Многоклассовая логистическая регрессия

Так как только один элемент  $\mathbf{Y}$  равен единице, функция вероятности  $\mathbf{Y}$  равна

$$\mathbb{P} [\mathbf{Y} \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}] = \prod_{j=1}^K (\pi_j (\tilde{\mathbf{x}}))^{Y_j}$$

Выберем класс  $c_K$  в качестве базового класса и рассмотрим логарифмическое отношение шансов (log-odds ratio) других классов относительно  $c_K$ .

Допустим, что эти логарифмические отношения шансов линейны по  $\tilde{\mathbf{X}}$ , но величины  $\tilde{\mathbf{w}}_i$  различны для классов  $c_i$ :

$$\begin{aligned} \ln (\text{odds} (\mathbf{Y} = \mathbf{e}_i \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}})) &= \ln \frac{\mathbb{P} [\mathbf{Y} = \mathbf{e}_i \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}]}{\mathbb{P} [\mathbf{Y} = \mathbf{e}_K \mid \tilde{\mathbf{X}} = \tilde{\mathbf{x}}]} = \\ &= \ln \frac{\pi_i (\tilde{\mathbf{x}})}{\pi_K (\tilde{\mathbf{x}})} = \tilde{\mathbf{w}}_i^T \tilde{\mathbf{x}} = w_{i0} x_0 + w_{i1} x_1 + \dots + w_{id} x_d, \end{aligned}$$

где  $w_{i0} = \beta_i$  – это смещение для класса  $c_i$ .



# Многоклассовая логистическая регрессия

Приравнивая  $\tilde{\mathbf{w}}_K = 0$ , получаем  $\exp \left\{ \tilde{\mathbf{w}}_K^T \tilde{\mathbf{x}} \right\} = 1$ , тогда полная модель для многоклассовой логистической регрессии принимает следующий вид

$$\pi_i(\tilde{\mathbf{x}}) = \frac{\exp \left\{ \tilde{\mathbf{w}}_i^T \tilde{\mathbf{x}} \right\}}{\sum_{j=1}^K \exp \left\{ \tilde{\mathbf{w}}_j^T \tilde{\mathbf{x}} \right\}}, i = 1, 2, \dots, K$$

Функция, указанная выше, называется функцией softmax.

Когда  $K = 2$ , эта формулировка дает в точности модель бинарной логистической регрессии.

Выбор базисного класса несущественен, так как логарифмическое отношение шансов может быть выведено для любых двух классов  $c_i$  и  $c_j$ .



# Оценка максимального правдоподобия

Чтобы найти  $K$  векторов весов регрессии  $\tilde{\mathbf{w}}_i$ ,  $i = 1, 2, \dots, K$ , используем метод градиентного подъема для максимизации функции логарифма правдоподобия. Правдоподобие данных равно

$$L(\tilde{\mathbf{W}}) = \mathbb{P}[Y | \tilde{\mathbf{W}}] = \prod_{i=1}^n \mathbb{P}[\mathbf{y}_i | \tilde{\mathbf{X}} = \tilde{\mathbf{x}}_i] = \prod_{i=1}^n \prod_{j=1}^K (\pi_j(\tilde{\mathbf{x}}_i))^{y_{ij}},$$

где  $\tilde{\mathbf{W}} = \{\tilde{\mathbf{w}}_1, \tilde{\mathbf{w}}_2, \dots, \tilde{\mathbf{w}}_K\}$  – это множество  $K$  весовых векторов.

Тогда логарифм правдоподобия равен

$$\ln(L(\tilde{\mathbf{W}})) = \sum_{i=1}^n \sum_{j=1}^K y_{ij} \ln(\pi_j(\tilde{\mathbf{x}}_i)) = \sum_{i=1}^n \sum_{j=1}^K y_{ij} \ln \left( \frac{\exp\{\tilde{\mathbf{w}}_j^T \tilde{\mathbf{x}}_i\}}{\sum_{k=1}^K \exp\{\tilde{\mathbf{w}}_k^T \tilde{\mathbf{x}}_i\}} \right)$$

Заметим, что функция логарифма правдоподобия, взятая с обратным знаком, может рассматриваться как функция ошибки, называемая ошибкой кросс-энтропии.

Для стохастического градиентного подъема вектора весов обновляются по одной точке данных на каждом шаге.



# Оценка максимального правдоподобия с SGA

Градиент функции логарифма правдоподобия относительно  $\tilde{\mathbf{w}}_j$  в точке  $\tilde{\mathbf{x}}_i$  равен

$$\nabla (\tilde{\mathbf{w}}_j, \tilde{\mathbf{x}}_i) = [y_{ij} - \pi_j(\tilde{\mathbf{x}}_i)] \tilde{\mathbf{x}}_i,$$

что приводит к следующему правилу обновления для  $j$ -го вектора весов:

$$\tilde{\mathbf{w}}_j^{t+1} = \tilde{\mathbf{w}}_j^t + \eta \nabla (\tilde{\mathbf{w}}_j^t, \tilde{\mathbf{x}}_i),$$

где  $\tilde{\mathbf{w}}_j^t$  обозначает оценку  $\tilde{\mathbf{w}}_j$  на шаге  $t$  и  $\eta$  – это коэффициент скорости обучения.

Когда модель обучена, можно прогнозировать класс  $\hat{y}$  для любой новой точки  $\tilde{\mathbf{z}}$  так:

$$\hat{y} = \arg \max_{c_i} \{\pi_i(\tilde{\mathbf{z}})\} = \arg \max_{c_i} \left\{ \frac{\exp \{\tilde{\mathbf{w}}_i^T \tilde{\mathbf{z}}\}}{\sum_{j=1}^K \exp \{\tilde{\mathbf{w}}_j^T \tilde{\mathbf{z}}\}} \right\}$$

Здесь оценивается функция softmax и прогнозируемый класс  $\hat{y}$  имеет наибольшую вероятность.

**LogisticRegression-MultiClass ( $D, \eta, \epsilon$ ):**

- 1 **foreach**  $(x_i^T, y_i) \in D$  **do**
- 2    $\tilde{x}_i^T \leftarrow (1 \ x_i^T)$  // map to  $\mathbb{R}^{d+1}$
- 3    $y_i \leftarrow e_j$  if  $y_i = c_j$  // map  $y_i$  to  $K$ -dim Bernoulli vector
- 4    $t \leftarrow 0$  // step/iteration counter
- 5 **foreach**  $j = 1, 2, \dots, K$  **do**  $\tilde{w}_j^t \leftarrow (0, \dots, 0)^T \in \mathbb{R}^{d+1}$
- 6 **repeat**
- 7   **foreach**  $j = 1, 2, \dots, K - 1$  **do**  $\tilde{w}_j \leftarrow \tilde{w}_j^t$  // copy  $\tilde{w}_j^t$
- 8   **foreach**  $\tilde{x}_i \in \tilde{D}$  in random order **do**
- 9     **foreach**  $j = 1, 2, \dots, K - 1$  **do**
- 10        $\pi_j(\tilde{x}_i) \leftarrow \exp\{\tilde{w}_j^T \tilde{x}_i\} / \sum_{a=1}^K \exp\{\tilde{w}_a^T \tilde{x}_i\}$
- 11        $\nabla(\tilde{w}_j, \tilde{x}_i) \leftarrow (y_{ij} - \pi_j(\tilde{x}_i)) \cdot \tilde{x}_i$  // gradient at  $\tilde{w}_j$
- 12        $\tilde{w}_j \leftarrow \tilde{w}_j + \eta \cdot \nabla(\tilde{w}_j, \tilde{x}_i)$  // update estimate for  $\tilde{w}_j$
- 13   **foreach**  $j = 1, 2, \dots, K - 1$  **do**  $\tilde{w}_j^{t+1} \leftarrow \tilde{w}_j$  // update  $\tilde{w}_j^{t+1}$
- 14    $t \leftarrow t + 1$
- 15 **until**  $\sum_{j=1}^{K-1} \|\tilde{w}_j^t - \tilde{w}_j^{t-1}\| \leq \epsilon$

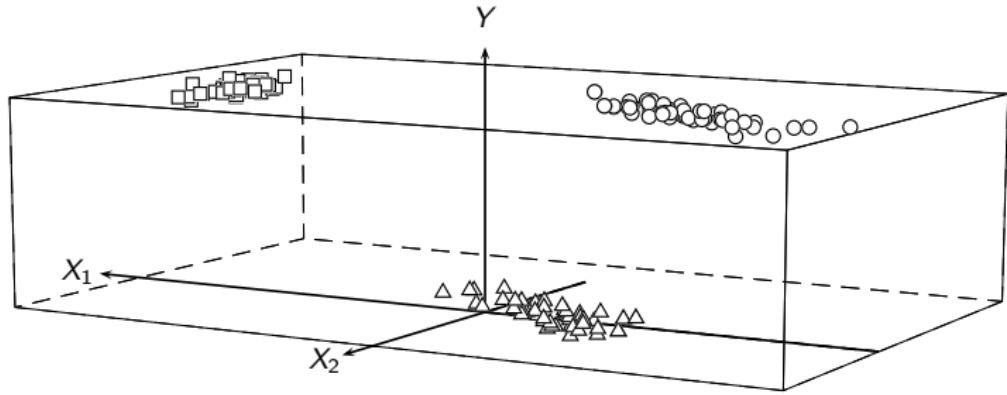


Рассмотрим двумерную версию набора Ирисы (после применения PCA),  $n = 150$ .

Отклик  $Y$  принимает 3 значения:  $Y = c_1$  – Iris-setosa ( $\square$ ),  $Y = c_2$  – Iris-versicolor ( $\circ$ ) и  $Y = c_3$  – Iris-virginica ( $\Delta$ ).

$$Y = c_1 \rightarrow \mathbf{e}_1 = (1, 0, 0)^T, Y = c_2 \rightarrow \mathbf{e}_2 = (0, 1, 0)^T, Y = c_3 \rightarrow \mathbf{e}_3 = (0, 0, 1)^T$$

Все точки фактически лежат на плоскости  $(X_1, X_2)$ , но классы  $c_1$  и  $c_2$  смещены на рисунке вдоль оси  $Y$  для наглядности.

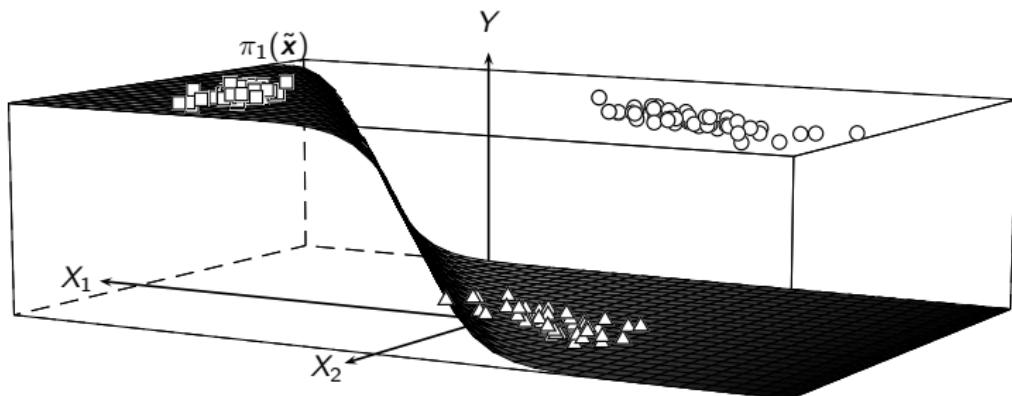


Используем  $Y = c_3$  как базовый класс. Обученная модель имеет веса:

$$\tilde{\mathbf{w}}_1 = (-3.52, 3.62, 2.61)^T, \tilde{\mathbf{w}}_2 = (-6.95, -5.18, -3.40)^T, \tilde{\mathbf{w}}_3 = (0, 0, 0)^T$$

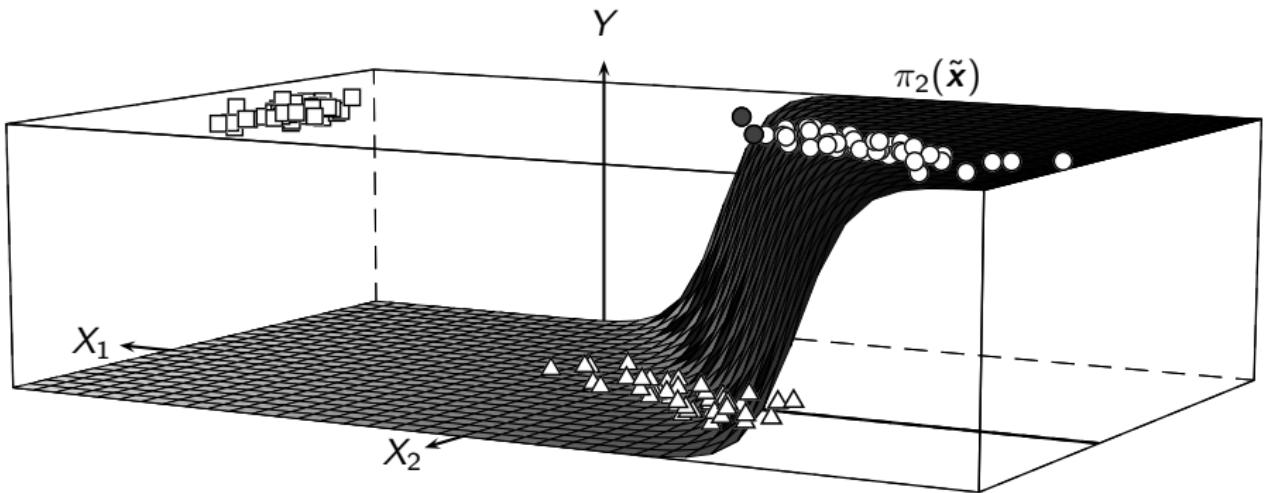
Поверхность принятия решения для класса  $c_1$  задается формулой

$$\pi_1(\tilde{\mathbf{x}}) = \frac{\exp\{\tilde{\omega}_1^T \tilde{\mathbf{x}}\}}{1 + \exp\{\tilde{\omega}_1^T \tilde{\mathbf{x}}\} + \exp\{\tilde{\omega}_2^T \tilde{\mathbf{x}}\}}$$



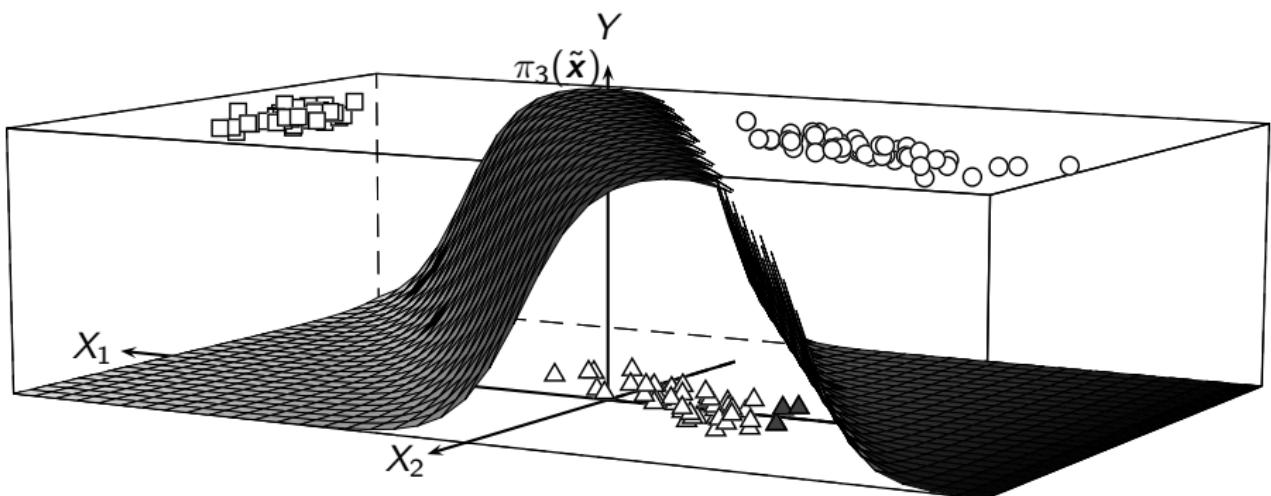
Поверхность принятия решения для класса  $c_2$  задается формулой

$$\pi_2(\tilde{\mathbf{x}}) = \frac{\exp\{\tilde{\omega}_2^T \tilde{\mathbf{x}}\}}{1 + \exp\{\tilde{\omega}_1^T \tilde{\mathbf{x}}\} + \exp\{\tilde{\omega}_2^T \tilde{\mathbf{x}}\}}$$



Поверхность принятия решения для класса  $c_3$  задается формулой

$$\pi_3(\tilde{\mathbf{x}}) = \frac{1}{1 + \exp\{\tilde{\omega}_1^T \tilde{\mathbf{x}}\} + \exp\{\tilde{\omega}_2^T \tilde{\mathbf{x}}\}}$$



На обучающих данных точность (аккуратность) классификации равна 96.7%, так как неверно классифицированы только 5 точек (показаны на визуализации серым). Например, для точки  $\tilde{\mathbf{x}} = (1, -0.52, -1.19)^T$  имеем

$$\pi_1(\tilde{\mathbf{x}}) = 0, \pi_2(\tilde{\mathbf{x}}) = 0.448, \pi_3(\tilde{\mathbf{x}}) = 0.552$$

Предсказанный класс для  $\tilde{\mathbf{x}}$  равен  $\hat{y} = \arg \max_{c_i} \{\pi_i(\tilde{\mathbf{x}})\} = c_3$ , в то время как истинный класс равен  $y = c_2$ .

