Системы искусственного интеллекта

02.03.03 - Математическое обеспечение и администрирование информационных систем, направленность (профиль) - разработка и администрирование информационных систем
09.03.03 - Прикладная информатика, направленность (профиль) - прикладная информатика в экономике

http//vikchas.ru

Тема 1.Введение в искусственный интеллект и машинное обучение

Лекция 6 «Машинное обучение»

Часовских Виктор Петрович
д-р техн. наук, профессор кафедры ШИиКМ

ФГБОУ ВО «Уральский государственный экономический университет»

Екатеринбург 2023

Машинное обучение — это класс методов искусственного интеллекта, которые позволяют улучшить результаты работы компьютеров путем обучения на известных данных.

Существует множество моделей для машинного обучения, но они, как правило, относятся к одному из трех типов:

- **обучение с учителем** (supervised learning);
- **обучение без учителя**, или самообучение (unsupervised learning);
- **обучение с подкреплением** (reinforcement learning)

Обучение с учителем

В общем виде машинное обучение можно описать как построение алгоритмов, выявляющих или «изучающих» присутствующие в данных ***закономерности***; обучение с учителем представляет собой подраздел машинного обучения, специализирующийся на поиске взаимосвязей между ***характеристиками уже подготовленных данных***.

Будем рассматривать типичную для обучения с учителем задачу: поиск связи между характеристиками дома и его стоимостью. Очевидно, что цена дома зависит от таких характеристик, как количество комнат, их метраж, близость к школам и другой социальной инфраструктуре, а также от того, хотят ли люди приобрести дом в собственность или предпочитают аренду. Все эти данные ***уже измерены***, и цель обучения с учителем состоит в выявлении корреляции между ними.

Под «измеренными» данными понимается представление каждой характеристики в виде числа.

После преобразования всех «характеристик» в числа нужно выбрать структуру для их представления.

В машинном обучении принята почти универсальная структура, которая, как оказалось, облегчает процесс вычислений.

Это представление одного наблюдения, например одного дома, в виде строки данных.

Такие строки складываются друг на друга, формируя «пакеты» данных, которые и передаются моделям в виде двумерных массивов ndarray.

Свои прогнозы модели выдают также в виде двумерных объектов ndarray, по одному прогнозу на наблюдение.

При этом длина каждой строки в массиве определяется количеством признаков данных.

В общем случае одна характеристика может давать целый набор признаков. Тогда мы говорим, что точка данных относится к одной или нескольким ***категориям***.

Например, дом может быть облицован красным кирпичом, коричневым кирпичом или плиткой из сланца.

Процесс извлечения признаков из того, что мы воспринимаем как характеристики наблюдений, называется ***проектированием признаков***.

Обучение с учителем в конечном счете сводится к выявлению взаимосвязей между характеристиками данных.

На практике мы выбираем одну характеристику и пытаемся предсказать ее поведение. Такая характеристика называется целевой.

Выбор целевой характеристики зависит от поставленной задачи.

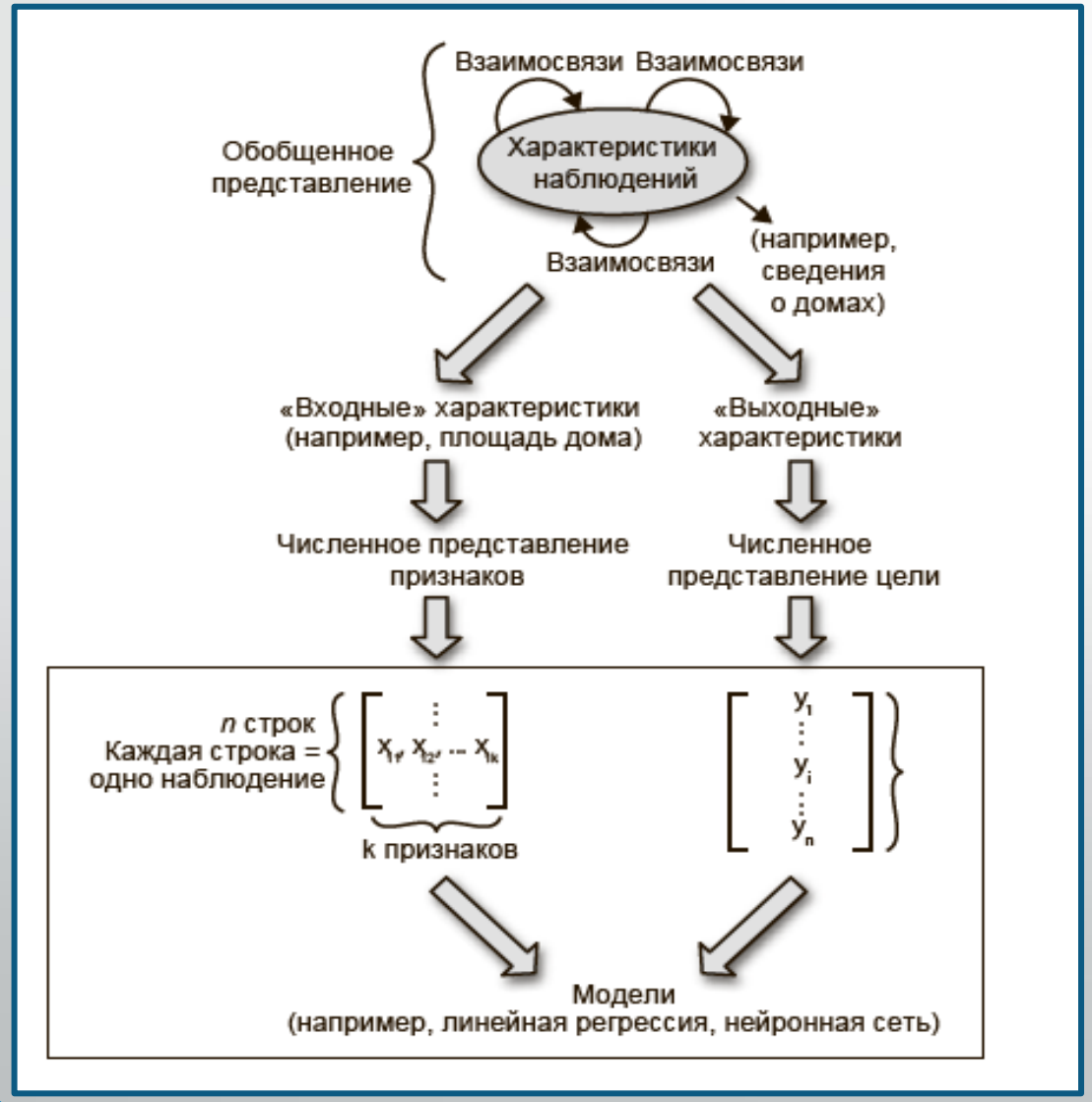
Например, если нужно описать взаимосвязь между ценами домов и количеством комнат в них, в качестве целевого признака можно выбрать как цену, так и количество комнат.

В обоих случаях обученная модель опишет взаимосвязь между этими двумя характеристиками.

Если же наша задача научить модель оценивать ранее не выставлявшиеся на продажу дома, целевым признаком должна стать цена, потому что именно ее будет предсказывать обученная модель исходя из остальных признаков.

Схематично принцип обучения с учителем показан на рисунке.

Показан как самый высокий уровень поиска взаимосвязей между данными, так и самый низкий уровень классификации этих связей между целевыми и исходными признаками после обучения модели.



На практике мы будем заниматься тем, что показано в нижней части рисунка.

При этом зачастую собрать правильный набор данных, корректно определить задачу и выполнить проектирование признаков намного сложнее, чем осуществить моделирование.

Алгоритмы обучения с учителем

Теперь, когда вы получили общее представление о назначении алгоритмов для обучения с учителем, напомню, что эти алгоритмы представляют собой просто вложенные функции.

С этой точки зрения цель обучения с учителем состоит в ***поиске*** функции, которая принимает в качестве входных данных массивы **ndarray** и их же дает на выходе. Эта функция должна преобразовывать массивы предоставленных ей признаков в массив, значения которого как можно точнее аппроксимируют целевые данные.

Мы будем представлять данные в виде матрицы ***X***, состоящей из ***n*** строк, каждая из которых содержит наблюдение с ***k*** признаками, причем все эти признаки числовые. Фактически каждое наблюдение представляет собой вектор

$$x_i = [\, x_{i1}, x_{i2}, \ldots\ldots\ldots, x_{ik}] \,,$$

и совокупно все эти наблюдения составляются в пакет.

Вот как будет выглядеть пакет из трех наблюдений:

$$X_{batch} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1k} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2k} \\ x_{31} & x_{32} & x_{33} & \cdots & x_{3k} \end{bmatrix}$$

batch - партия

Каждому пакету наблюдений будет сопоставлен набор целей.

Элемент такого пакета представляет собой номер цели для соответствующего наблюдения. Его можно представить как вектор-столбец:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Наша задача в рамках обучения с учителем будет состоять в построении функции, принимающей пакеты наблюдений, представленные в виде структуры $\boldsymbol{X}_{batch}$*, и продуцирующей векторы* p_i*(которые мы интерпретируем как "предсказания"), достаточно близкие к целевым значениям* y_i*.*

Можно приступить к построению первой модели на базе реальных данных. Начнем мы с такой распространенной модели, как линейная регрессия.

Линейная регрессия

Примеры применение регрессионного анализ

1. Моделирование числа поступивших в университет для лучшего понимания факторов, удерживающих детей в том же учебном заведении.
2. Моделирование потоков миграции в зависимости от таких факторов как средний уровень зарплат, наличие медицинских, школьных учреждений, географическое положение…
3. Моделирование дорожных аварий как функции скорости, дорожных условий, погоды и т.д.,
4. Моделирование потерь от пожаров как функции от таких переменных как количество пожарных станций, время обработки вызова, или цена собственности.
5. Предсказание цены акции,
6. оценка цены объекта недвижимости

Линейная регрессия позволяет ответить на следующие вопросы:

- Есть ли связь между 2 переменными?
- Насколько прочны отношения?
- Какая переменная вносит наибольший вклад?
- Насколько точно мы можем оценить влияние каждой переменной?
- Насколько точно мы можем предсказать цель?
- Являются ли отношения линейными?
- Есть ли эффект взаимодействия?

Предположим, у нас есть только одна переменная **x** и одна цель – **y**.

Тогда линейная регрессия выражается как:

$$y = a_0 + a_1 x_1$$

Это уравнение для линейной модели с 1 переменной и 1 целью.

В приведенном выше уравнении a_0 и a_1 являются коэффициентами.

Эти коэффициенты - то, что нам нужно, чтобы делать прогнозы с нашей моделью.

Итак, как мы можем найти эти параметры?

Чтобы найти параметры, нам нужно минимизировать **сумму квадратов ошибок.**

Конечно, линейная модель не идеальна, и она не будет точно предсказывать все данные, а это означает, что существует разница между фактическим значением и прогнозом. Ошибка легко вычисляется с помощью:

$$e_i = y_i - \hat{y}_i$$

т.е. из истинного значения y_i вычитается прогноз $\hat{y}_i$

Но почему ошибки возводятся в квадрат?

Принято ошибку возводить в квадрат, потому что прогноз может быть выше или ниже истинного значения, что приводит к отрицательной или положительной разнице соответственно. Если бы мы не возводили в квадрат ошибки, сумма ошибок могла бы уменьшиться из-за отрицательных различий, а не потому, что модель хорошо подходит.

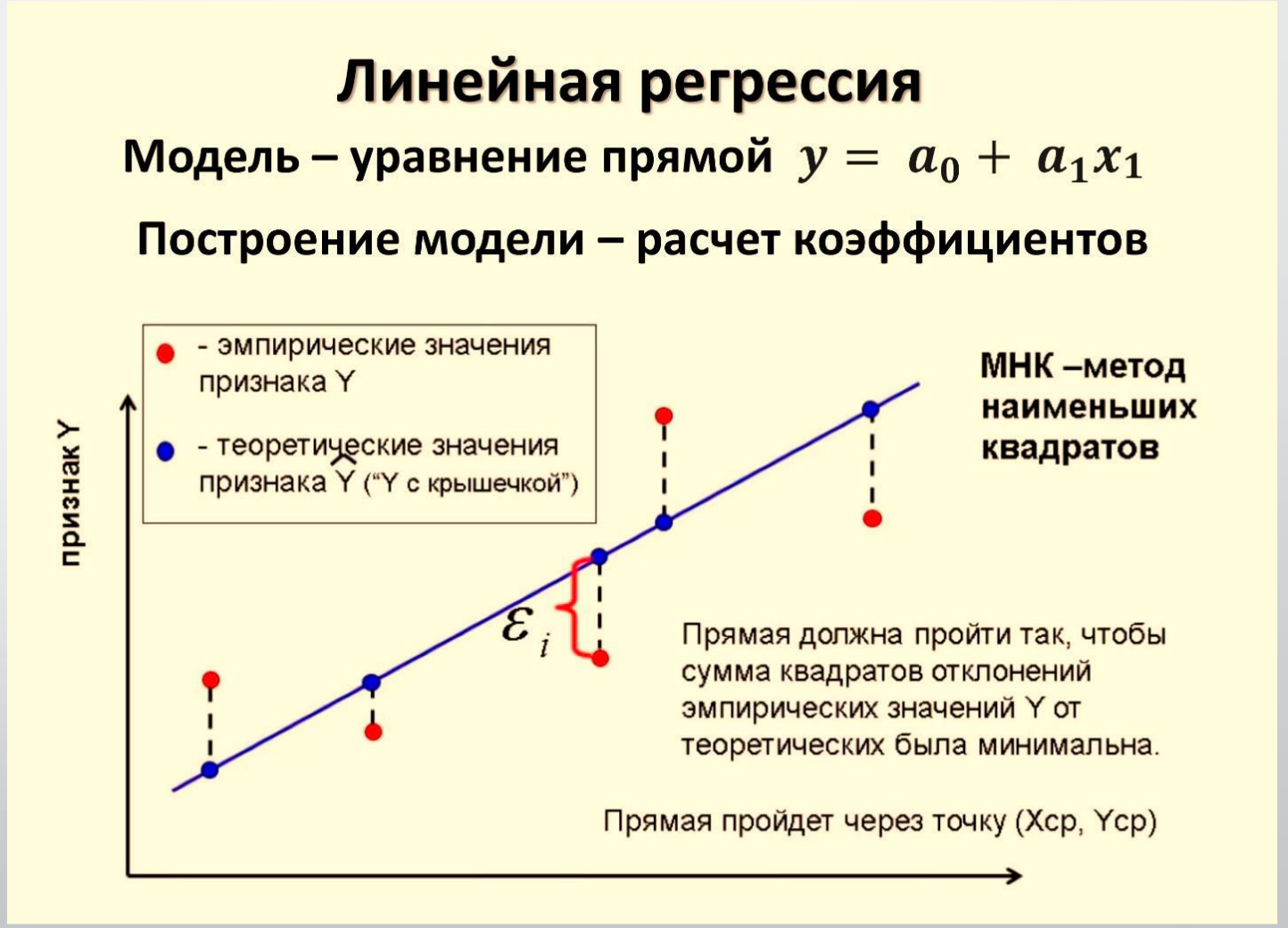
Кроме того, возведение в квадрат ошибок учитывает большие различия, поэтому минимизация квадратов ошибок «гарантирует» лучшую модель.

Рассмотрим график линейной регрессии чтобы лучше понять.

На приведенном графике красные точки — это истинные данные, а синяя линия - линейная модель.

Прерывистые черные линии иллюстрируют ошибки между предсказанными и истинными значениями.

Таким образом, синяя линия - это та, которая минимизирует сумму квадратов длины прерывистых черных линий.



Математический вывод коэффициентов определяет следующие выражения:

$$a_i = \frac{\sum_{i=1}^{n}(x_i - \hat{x})(y_i - \hat{y})}{\sum_{i=1}^{n}(x_i - \hat{x})^2}$$

$$a_0 = \hat{y} - a_1\hat{x}$$

Где $\hat{x}$ и $\hat{y}$ представляют собой среднее.

Как определить, что коэффициенты актуальны для нашего прогнозирования? Для этого необходимо оценить корреляцию.

Коэффициент детерминации

Предположим, что есть ряд наблюдений y_i, рассчитали среднее значение $\bar{y}$. Реальные значения отклоняются от этой средней, можно измерить отклонение суммированием всех возведенных в квадрат погрешностей:

Вся SSE (сумма всех возведенных в квадрат погрешностей, или **СКП) =** $\sum(y_i - \bar{y})^2$

Полученное выражение суммы квадратических погрешностей (СКП), можно разделить на различные компоненты

Модель регрессии объясняет *некоторые* отклонения реальных наблюдений от средней:

Объяснимая часть СКП = $\sum(\hat{y}_i - \bar{y})^2$

Но есть еще то, что модель регрессии не объясняет всех отклонений, и кое-какие *остатки* так и останутся необъясненными:

Необъяснимая часть СКП = $\sum(y_i - \bar{y})^2$**.**

Найдем, что:

Вся СКП = Объяснимые СКП + Необъяснимые СКП.

Чем больше отклонение, объяснимое регрессией, тем точнее прямая наилучшего соответствия. Отсюда показатель наилучшего соответствия прямой данным — это отношение суммарных **СКП,** которое *объясняется* моделью регрессии. Это и есть **коэффициент детерминации:**

$$\textbf{коэффициент детерминации} = \frac{\text{Объяснимые СКП}}{\text{Вся СКП}}$$

Этот показатель принимает значения от 0 до 1. Если он близок к 1, тогда большая часть отклонений объяснена регрессией, необъяснимое отклонение невелико, и прямая идеально подходит к данным. Если значение около 0, то, наоборот, большая часть отклонений необъяснима, и прямая, хотя и лучший из вариантов, но тем не менее она все равно «низкого качества».

Математическое выражение для **коэффициента детерминации** следующее:

$$\textbf{коэффициента детерминации} = \left[\frac{n\sum xy - \sum x \sum y}{\sqrt{[n\sum x^2 - (\sum x)^2] * [n\sum y^2 - (\sum y)^2]}}\right]^2$$

Коэффициент детерминации обозначается r^2 или R^2, и лучше всего доверить расчет компьютеру.

Обычно любое значение коэффициента детерминации свыше 0,5 считается хорошим. При более низком коэффициенте, скажем близком к 0,2 , большая часть отклонения необъяснима с помощью регрессии, и зависимость несильная. Однако не следует забывать о единичных крайних, далеко отстоящих данных наблюдений. Даже случайное значение такого рода может сказаться на регрессии и снизить коэффициент детерминации, так что всегда есть искушение допустить, что они — ошибки, и проигнорировать их. Этого делать ни в коем случае нельзя! Такую точку можно отбросить, только если есть настоящая причина - ошибка или потому, что точка абсолютно не сопоставима с другими данными. Решение произвольно отбросить некоторые наблюдения, потому что они портят рисунок, приводят к потере самом цели анализа — оценить, есть ли некая линия и степень зависимости.

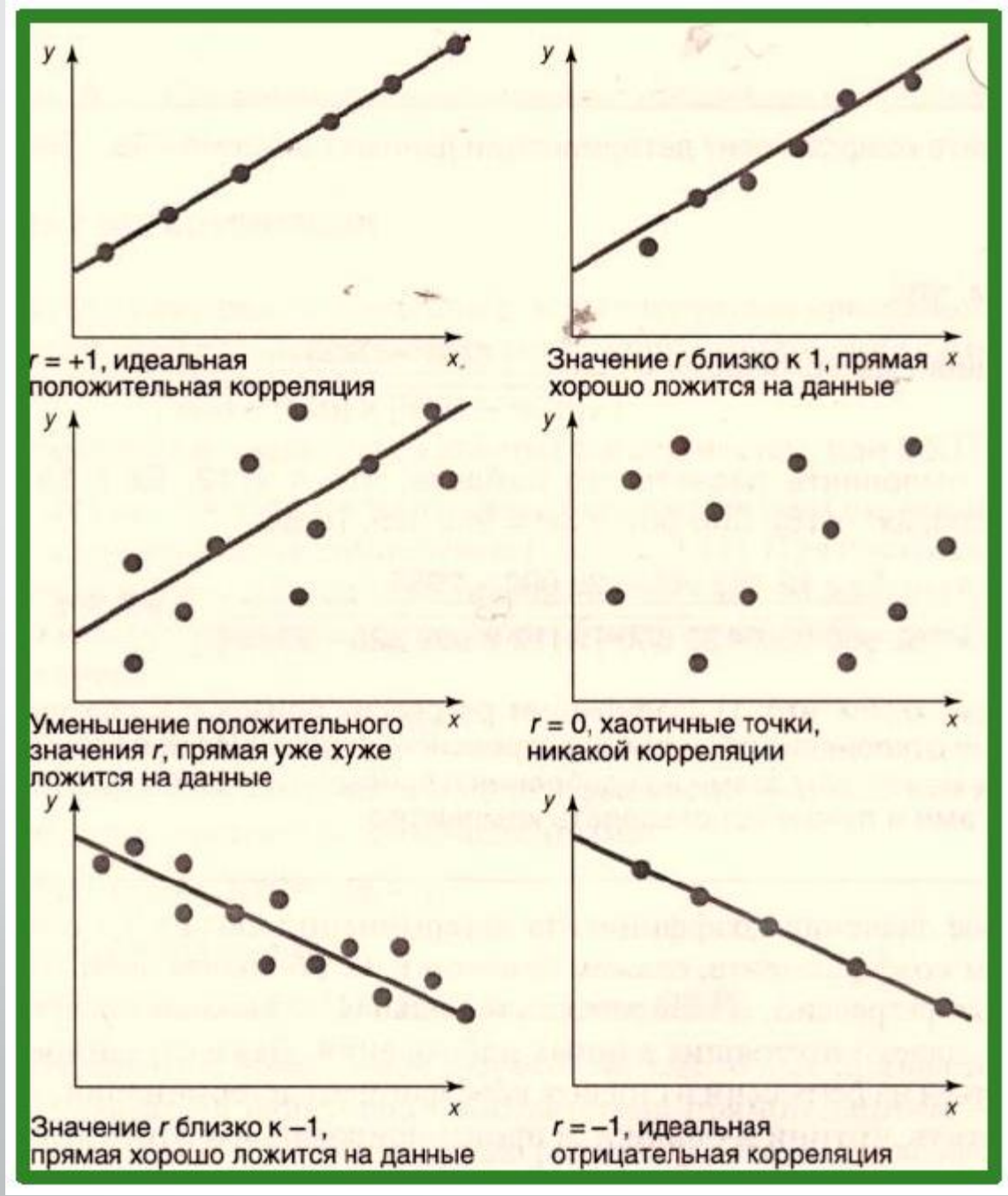
Коэффициент корреляции

Вторым показателем регрессии выступает **коэффициент корреляции,** который отвечает на основной вопрос: связаны ли x и y линейно? Коэффициенты корреляции и детерминации, очевидно, отвечают на очень схожие вопросы, и типичный результат показывает, что:

$$\text{Коэффициент корреляции} = \sqrt{\text{Коэффициент детерминации}}$$

Теперь понятно, почему мы обозначили коэффициент детерминации как r^2 - чтобы можно было обозначить коэффициент корреляции как r. Его называют еще коэффициентом Пирсона, и он принимает значения между +1 и - 1.

Значение $r = 1$ показывает, что две переменные идеально связаны линейной зависимостью при полном отсутствии помех: когда одна растет, то же самое происходит и с другой. Интерпретация коэффициента корреляции показана на следующем рисунке:



При коэффициенте корреляции, близком к +1 или -1, между двумя переменными будет сильная зависимость.

Однако, когда r падает до 0,7 или -0,7, коэффициент детерминации равен 0,49, т. е. становится очень низким.

Можно заключить, что при значениях *r* от 0,7 до -0,7 зависимость достаточно слабая.

Линейная регрессия в машинном обучение

Линейную регрессию в машинном обучение часто представляют следующей формулой:

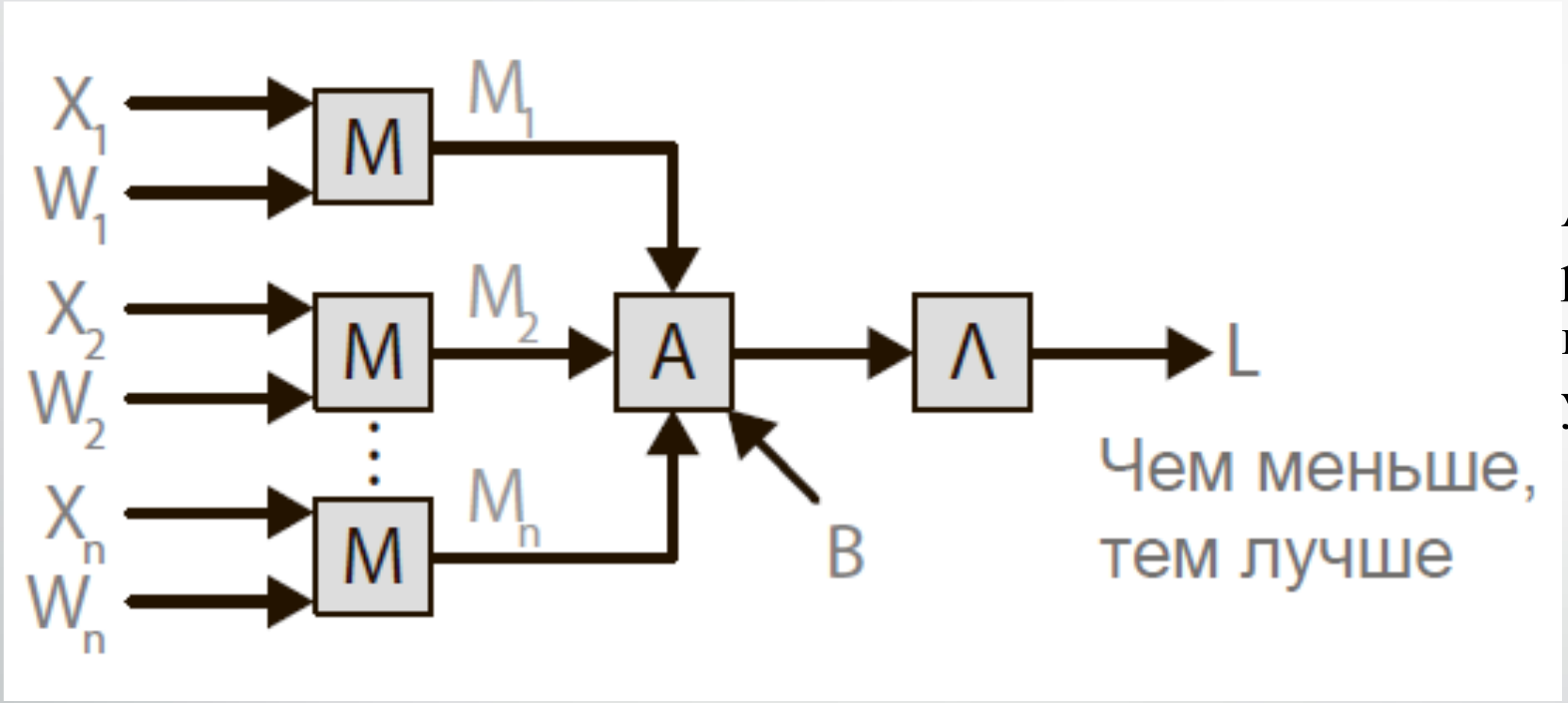
$$y_i = \beta_0 + \beta_1 \times x_1 + \ldots + \beta_n \times x_k + \epsilon$$

Фактически мы предполагаем, что каждую целевую переменную можно представить как линейную комбинацию k признаков и константы β_0, которая корректирует ≪базовое≫ значение прогноза (в частности, это прогноз при нулевом значении всех признаков).

Из определения непонятно, каким образом мы будем писать код и обучать эту модель. Поэтому нужно выразить модель в виде функций, с которыми в предыдущей лекции.

И проще всего начать со схемы.

Визуализация



Алгоритм линейной регрессии, представленный в виде отдельных операций умножения и сложения.

На уровне отдельных элементов модель выглядит так:
Вычисляем произведение каждой пары x_i и w_i , а затем суммируем полученные результаты.

Рассмотрим модель, в котором отсутствует свободный член (он обозначен β_0).

Результат работы модели линейной регрессии можно представить как скалярное произведение каждого вектора наблюдений $x_i = [x_1, x_2, \ldots\ldots, x_k]$ и вектора параметров (весов), который мы назовем *W*:

$$W = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \ldots \\ w_k \end{bmatrix}$$

В этом случае предсказание выражается формулой:

$$p_i = x_i \times W = w_1 \times x_{i1} + w_2 \times x_{i2} + \ldots + w_k \times x_{ik}$$

Очевидно, что для нашей модели линейной регрессии ≪генерацию предсказания≫ можно представить с помощью всего одной операции: скалярного произведения.

Для пакета наблюдений будет использоваться другая операция: ***умножение матриц***.

Например, если есть пакет из трех наблюдений, записанный в виде матрицы:

$$P_{batch} = X_{batch} \times W = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1k} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2k} \\ x_{31} & x_{32} & x_{33} & \cdots & x_{3k} \end{bmatrix} \times \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \cdots \\ w_k \end{bmatrix} =$$

его умножение на вектор W даст набор предсказаний:

$$P_{batch} = X_{batch} \times W = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1k} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2k} \\ x_{31} & x_{32} & x_{33} & \cdots & x_{3k} \end{bmatrix} \times \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \cdots \\ w_k \end{bmatrix} =$$

$$\begin{bmatrix} x_{11} \times w_1 & x_{12} \times w_2 & x_{13} \times w_3 & \cdots & x_{1k} \times w_k \\ x_{21} \times w_1 & x_{22} \times w_2 & x_{23} \times w_3 & \cdots & x_{2k} \times w_k \\ x_{31} \times w_1 & x_{32} \times w_2 & x_{33} \times w_3 & \cdots & x_{3k} \times w_k \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

Обучение модели

Что означает «обучение» модели? Говоря простыми словами, модели берут данные, каким-то образом комбинируют их с ***параметрами*** и дают предсказания.

Например, модель линейной регрессии берет данные в виде матрицы *X* и набор параметров (весов) в виде вектора *W* и, перемножив их, дает вектор предсказаний:

$$P_{batch} = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

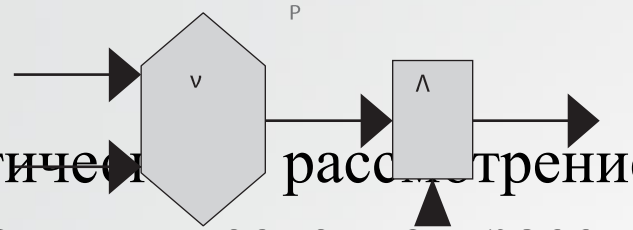
Однако для обучения модели требуется информация о **точности полученных прогнозов** (погрешность). Чтобы ее получить, добавим вектор целевых значений y_{batch}, связанный с подаваемым на вход модели набором наблюдений X_{batch}, и посчитаем величину отклонения сделанных прогнозов от ожидаемых.

Такая мера количества ошибок называется функцией потерь. Часто для этой цели используют **среднеквадратическую ошибку** (mean squared error, **MSE**):

$$MSE\left(P_{batch}, y_{batch}\right) = MSE\left(\begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}\right) = \frac{\left(y_1 - p_1\right)^2 + \left(y_2 - p_2\right)^2 + \left(y_3 - p_3\right)^2}{3}$$

Обозначим полученное число *L*. Можно посчитать его *градиент* по каждому элементу вектора *W*. Этими данными мы воспользуемся для ***обновления всех элементов вектора W в направлении, уменьшающем значение L***.

Многократное повторение этой процедуры и называется «**обучением**» модели.



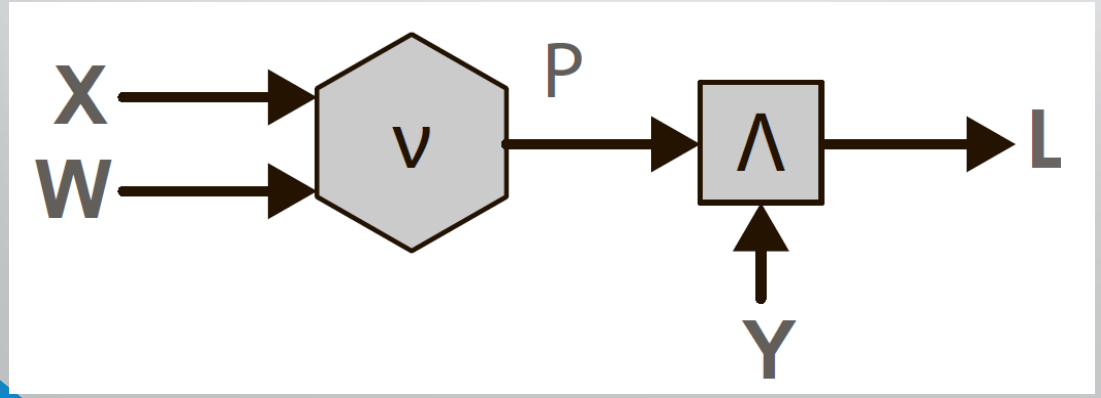
В математическом рассмотрение производных функций нескольких векторных переменных позволяет рассчитать частную производную каждого элемента вектора X по скалярному произведению N.

Этот показатель называется **градиентом** X и можно обозначить его dNdX. Дело в том, что каждому элементу вектора X, например x_3, в массиве dNdx соответствует элемент (например это dNdX [2]), представляющий собой частную производную скалярного произведения N по x_3.

Далее термин «градиент» будет обозначать , массив частных производных функции по каждому из ее аргументов.

Изменим предыдущий рисунок линейной регрессии.

Линейная регрессия: еще одна визуализация и математическая модель



Представим функцию потерь как набор вложенных функций:

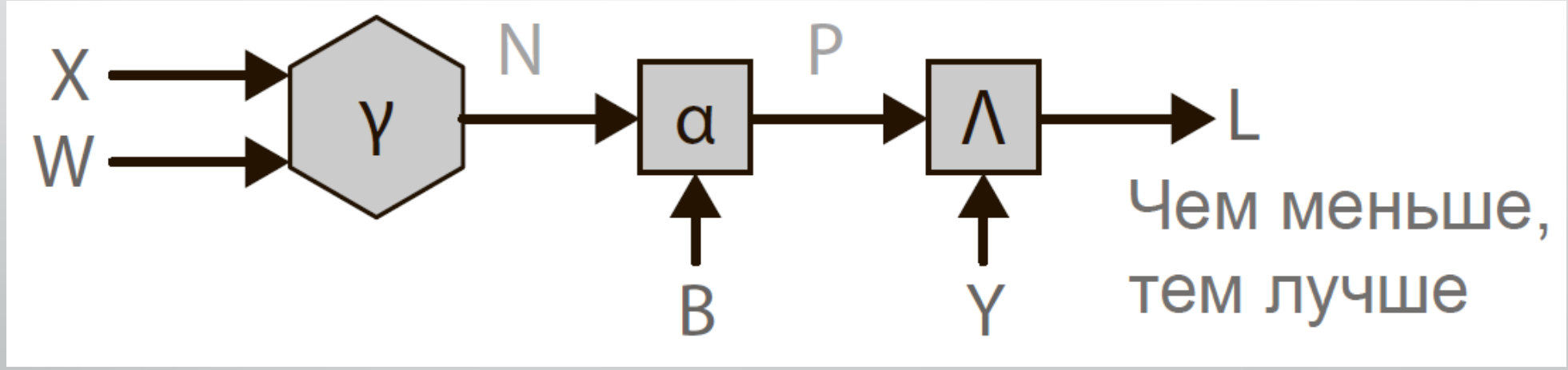
$$L = \Lambda\left(\nu\left(X, W\right), Y\right)$$

Свободный элемент

Добавим в нашу модель свободный элемент.

Это еще один элемент диаграммы, отвечающий за «смещение» (bias).

Вычислительный граф для функции линейной регрессии с добавленным элемента смещения следующий:



Изменения в математическом представлении после добавления смещения.

К скалярному произведению в каждом элементе p_i вектора предсказания будет прибавляться константа b:

$$P_{batch\_with\_bias} = x_i \times W + b = \begin{bmatrix} x_{11} \times w_1 & x_{12} \times w_2 & x_{13} \times w_3 & \cdots & x_{1k} \times w_k \\ x_{21} \times w_1 & x_{22} \times w_2 & x_{23} \times w_3 & \cdots & x_{2k} \times w_k \\ x_{31} \times w_1 & x_{32} \times w_2 & x_{33} \times w_3 & \cdots & x_{3k} \times w_k \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

Поскольку в линейной регрессии всего одно пересечение линии оценки, к каждому наблюдению добавляется ***одно и то же*** значение смещения.

Код

Напишем функцию, которая принимает данные наших наблюдений X_{batch}, целевые значения y_{batch} и дает на выходе прогноз и функцию потерь.

В случае вложенных функций вычисление производных происходит в два этапа.

Сначала во время «прямого прохода» входные данные последовательно пропускаются через набор операций с сохранением полученного результата.

Затем следует «обратный проход», во время которого сохраненные результаты применяются для вычисления соответствующих производных.

Результаты прямого прохода будем сохранять в словарь. Чтобы отделить их от параметров (которые также потребуются во время обратного прохода), поместим параметры в отдельный словарь:

```
def forward_linear_regression(X_batch: ndarray,
                              y_batch: ndarray,
                              weights: Dict[str, ndarray])
                          -> Tuple[float, Dict[str, ndarray]]:
'''
Прямой проход для линейной регрессии.
'''
# проверяем совпадение размеров X и y
assert X_batch.shape[0] == y_batch.shape[0]

# проверяем допустимость умножения матриц
assert X_batch.shape[1] == weights['W'].shape[0]

# проверяем, что B это объект ndarray размером 1x1
assert weights['B'].shape[0] == weights['B'].shape[1] == 1
```

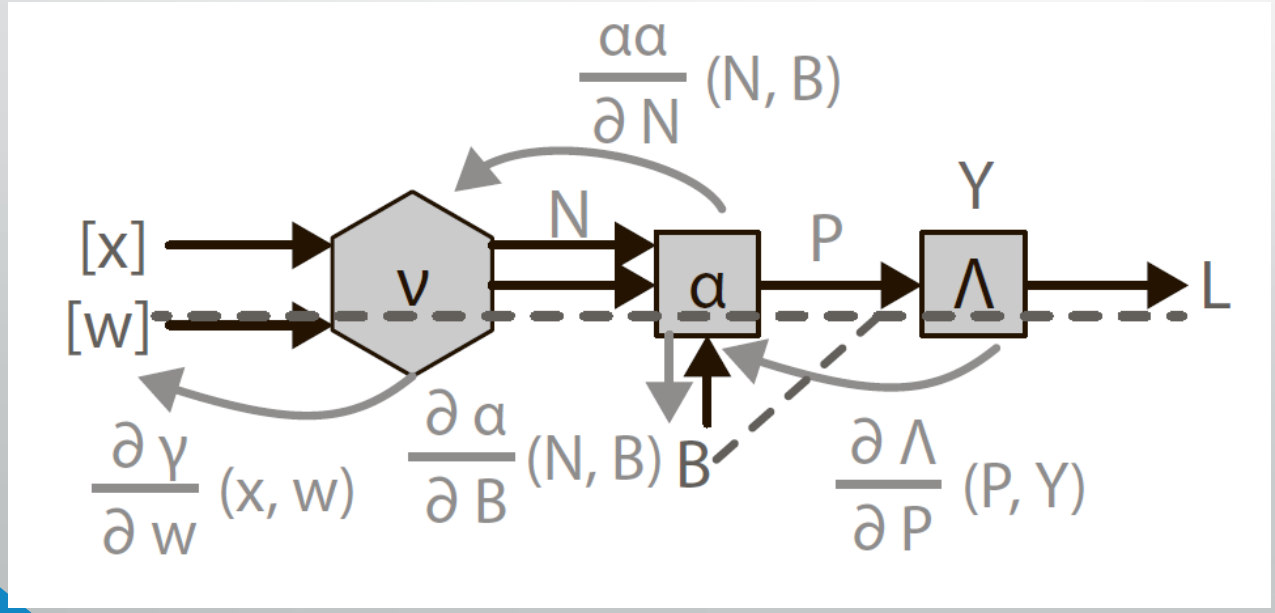
```
# вычисления
N = np.dot(X_batch, weights['W’])

P = N + weights['B’]

loss = np.mean(np.power(y_batch – P, 2))

# сохранение информации, полученной во время прямого прохода
forward_info: Dict[str, ndarray] = {}
forward_info['X'] = X_batch
forward_info['N'] = N
forward_info['P'] = P
forward_info['y'] = y_batch

return loss, forward_info
```

Обучение модели

Воспользуемся следующими инструментами и методами, вычислим $\frac{\partial L}{\partial w_i}$ для всех элементов w_i вектора W, а также $\frac{\partial L}{\partial b}$. Для этого выполним обратный проход, во время которого оценим частные производные вложенных функций при заданных входных значениях, а затем перемножим полученные результаты.

Диаграмма для операции вычисления градиентов

Концепция того, что мы хотим получить, представлена на диаграмме



Обратный проход по вычислительному графу линейной регрессии.

Будем вычислять частные производные каждой из вложенных функций, двигаясь изнутри наружу, а затем оценивать их для значений, полученных во время прямого прохода. После чего останется получить произведение результатов.

Обучение модели

Обучение представляет цикл из следующих операций:

1. Выбор набора данных.

2. Прямой проход модели.

3. Обратный проход модели с применением данных, полученных во

время прямого прохода.

4. Применение вычисленных градиентов для обновления весов.

Оценка точности модели

Важным является понимание насколько корректно построенная модель раскрывает взаимосвязи в данных?

Обучающая выборка представляет собой лишь часть от общей совокупности данных. А перед нами строит задача построить модель, выявляющую взаимосвязи во всей совокупности, несмотря на ограниченность тренировочных данных.

Всегда существует опасность, что модель начнет выбирать взаимосвязи, существующие в обучающей выборке, но отсутствующие в совокупности данных. Представьте, что в выборку случайно попали дома с тремя ванными комнатами, облицованные желтым сланцем, которые предлагаются по относительно низкой цене. Нейронная сеть обнаружит эту закономерность, хотя в общей совокупности данных она не наблюдается. Это явление называется переобучением (overfitting). Как понять, что у модели может быть подобный недостаток?

Чтобы избежать такой ситуации, данные, предназначенные для обучения модели, разбивают на ***обучающий набор*** (training set) и ***тестовый набор*** (testing set).

Первый используется для обучения модели (то есть для итеративного обновления весов), после чего точность работы модели оценивается на тестовых данных.

В основе этого подхода лежит простая логика. Если модель смогла обнаружить взаимосвязи, которые работают и на ***остальной части обучающей выборки*** (то есть на всем наборе данных), велика вероятность, что они присутствуют и в ***общей совокупности данных***.

Код

Попробуем оценить нашу модель на тестовом наборе. Первым делом напишем функцию, генерирующую предсказания, обрезав функцию forward_loss:

```
def predict(X: ndarray,
            weights: Dict[str, ndarray]):
'''
Генерация предсказаний для модели линейной регрессии.
'''
N = np.dot(X, weights['W’])
return N + weights['B’]
```

Теперь возьмем веса, которые возвращает обучающая функция, и напишем:

```
preds = predict(X_test, weights) # weights = train_info[0]
```

Насколько хороши эти предсказания? Пока ответа на этот вопрос нет.

Ведь мы еще не знаем, работает ли выбранный подход — определение модели как набора операций и ее обучение путем итеративной корректировки параметров, для которой мы вычисляем частные производные функции потерь по различным параметрам. Будет хорошо, если окажется, что все это хоть как-то работает. Для проверки результатов построим график зависимости предсказанных значений от фактических. В идеальном случае все точки должны оказаться на прямой линии с наклоном в 45 градусов.

Сравнение предсказанных и действительных значений для модели линейной регрессии

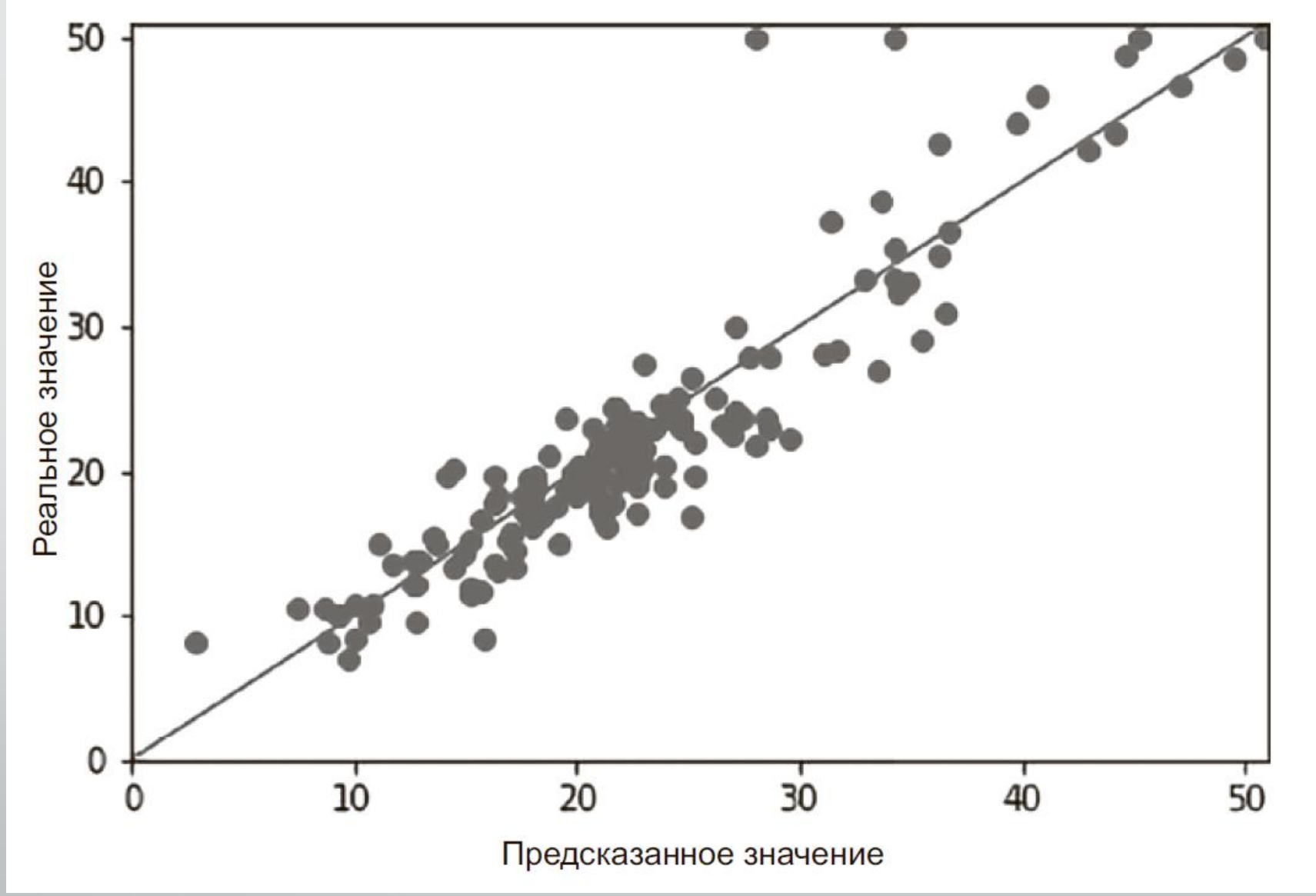


График выглядит вполне приемлемо, поэтому количественная оценка точности работы модели имеет смысл. Это можно сделать двумя способами:

- Вычислить абсолютное значение среднего расстояния между предсказаниями модели и фактическими значениями. Эту метрику называют ***средним модулем отклонения*** (mean absolute error, MAE):

```
def mae(preds: ndarray, actuals: ndarray):
    ''’
    Вычисление среднего линейного отклонения.
     ''’
     return np.mean(np.abs(preds — actuals))
```

- Вычислить средний квадрат расстояния между предсказаниями модели и фактическими значениями. Эту метрику называют *корнем из среднего квадрата отклонения* (root mean squared error, RMSE):

```
def rmse(preds: ndarray, actuals: ndarray):
    '‘’
    Вычисление корня из среднего квадрата отклонения.
    '‘’
    return np.sqrt(np.mean(np.power(preds — actuals, 2)))
```

Для рассматриваемой модели были получены значения:

Mean absolute error: 3.5643 - *средняя абсолютная ошибка*
Root mean squared error: 5.0508 - среднеквадратическая ошибка
Среднее(y_test) от целевого показателя (из примера) 22.0776

Соответственно прогнозы цен на недвижимость в этой модели в среднем отклоняются от фактических на 5.0508/22.0776 = 22.9%.

Такой подход вполне применим для построения моделей машинного обучения.

Благодарю за внимание!