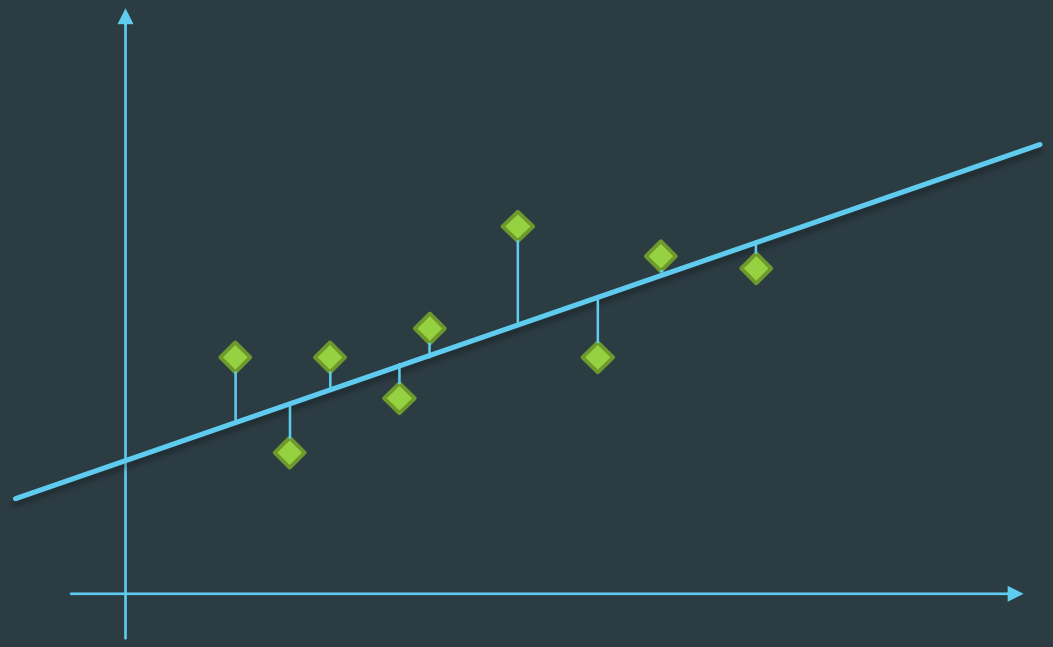
Задача регрессии



Объект, отклик

i-я строка матрицы – описание i-го объекта (конкретного экземпляра)

$$x^i = (x_1^i, x_2^i, x_3^i, x_4^i)$$

Каждому объекту сопоставлена непрерывная величина y (непрерывнозначный отклик, целевая переменная)

$$(x_1^i, x_2^i, x_3^i, x_4^i) \to y^i$$

$$y^i \in \mathbb{R}$$

Обучение с учителем. Регрессия

Размеченные данные

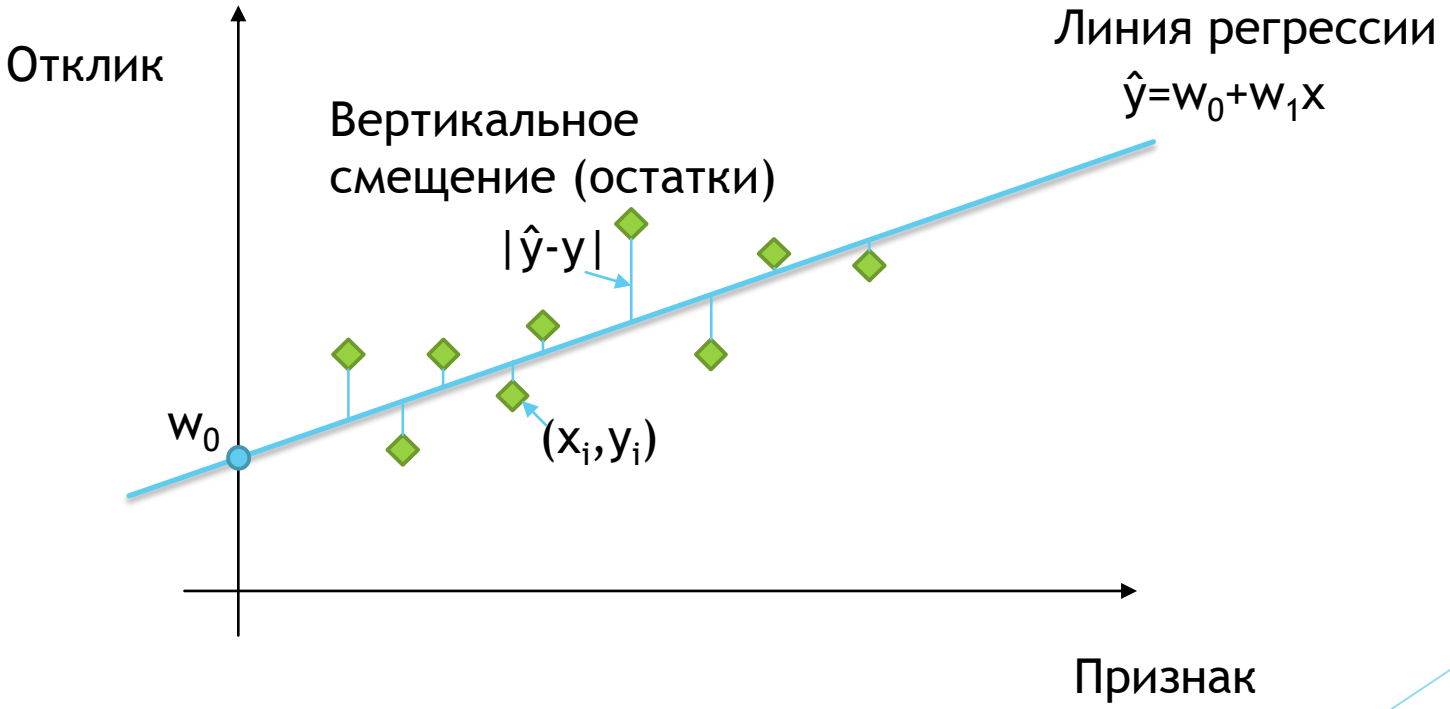
- Есть образцы
- Есть целевая переменная (непрерывная)

- Есть новые образцы

Требуется для новых образцов используя размеченные данные смоделировать связь между признаками и целевой переменной как можно более точно

Задача регрессии. Одномерный случай

- Смоделировать связь между признаком и откликом
- Уравнение линейной модели $y=w_0+w_1x$
- Задача – найти веса w_0 и w_1 чтобы смоделировать связь



Как решать задачу?

- Минимизировать сумму квадратичных вертикальных расстояний (остатков) до точек образцов
- Введем следующую функцию издержек – среднеквадратическая ошибка предсказания (Sum of Squared Error), ŷ – предсказанное значение

$$S(\mathbf{w}) = \frac{1}{2} \sum_i \left(y^{(i)} - \hat{y}^{(i)} \right)^2$$

По сути постановка задачи такая же как в методе наименьших квадратов

Нормальное уравнение

Задача: при обучении найти веса, по которым для новых экземпляров будем получать прогноз $\hat{y}=w_0+w_1x$

- Возможно решение в матричном виде
- X – матрица объектов
- y – целевая переменная для имеющихся объектов
- Связь объектов с целевой переменной:

$$Xw = y$$

- Решение в явном виде:

$$w = \left(X^T X\right)^{-1} X^T y$$

```
if y.ndim < 2:
    self.coef_, self._residues = optimize.nnls(X, y)
else:
    # scipy.optimize.nnls cannot handle y with shape (M, K)
    outs = Parallel(n_jobs=n_jobs_)(
        delayed(optimize.nnls)(X, y[:, j]) for j in range(y.shape[1])
    )
    self.coef_, self._residues = map(np.vstack, zip(*outs))
```

scipy

```
def nnls(A,b):
    """
    Solve ``argmin_x || Ax - b ||_2`` for ``x>=0``. This is a wrapper
    for a FORTAN non-negative least squares solver.
```

В sklearn этот метод реализован в классе sklearn.linear_model.LinearRegression, который по сути делает то же самое (просто обертка над данным решением)

lection_5_lin

Задача регрессии. Многомерный случай

- Смоделировать связь между признаками и откликом
- Уравнение модели (гиперплоскость)

$$y = w_0 x_0 + w_1 x_1 + \cdots + w_m x_m = \mathbf{w}^T \mathbf{x}$$

- Задача – найти веса $w_0, w_1, w_2..$ чтобы смоделировать связь

Что если зависимость нелинейная? Превращение линейной регрессии в полиномиальную

- Добавим в модель полиномиальные члены, где d – степень полинома

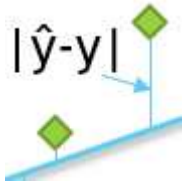
$$y = w_0 + w_1 x + w_2 x^2 + \ldots + w_d x^d$$

- Несмотря на нелинейные связи, модель по прежнему линейная (относительно коэффициентов w)

В sklearn есть специальный класс, который позволяет получить полиномиальные коэффициенты sklearn.preprocessing.PolynomialFeatures. Для его применения удобно использовать конвейер (Pipeline)

lection_5_nonlin

Анализ остатков



Остатки: $e_i = \hat{y}_i - y_i$

После того, как проверили модель на отложенных данных, от остатков хотим получить следующее:

- Гомоскедастичность – постоянство дисперсии
- Нормальное распределение остатков
- Независимость (актуально для временных рядов, там проверяется автокорреляция)

lection_5_residuals

Сравнение моделей между собой. Метрики

- MSE – mean square error, RMSE – корень из MSE

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

- MAE – mean absolute error

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n}|e_i|}{n}$$

Отличие: MSE придает дополнительный вес большим ошибкам

Лучшее значение: 0.0

Коэффициент детерминации

- это доля дисперсии зависимой переменной, объясняемая рассматриваемой моделью – соответствие модели данным

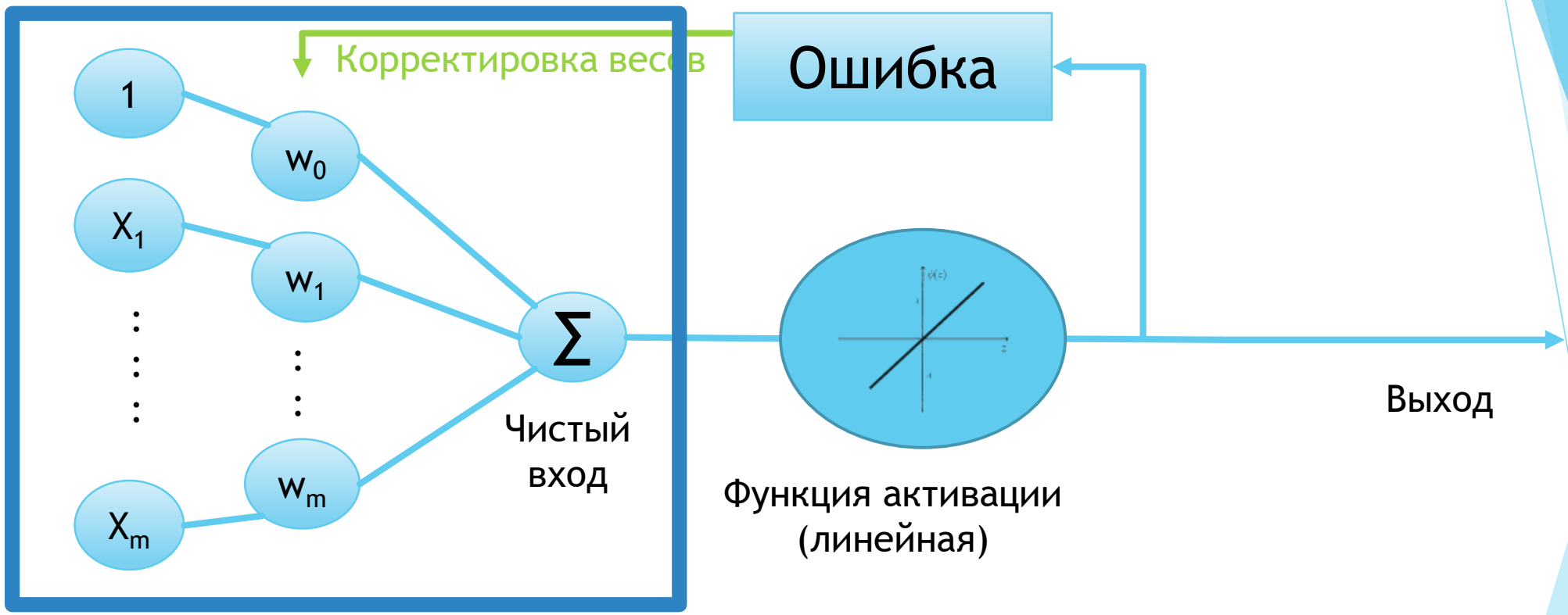
$$R^2 = 1 - \frac{D[y|x]}{D[y]} \quad \text{или} \quad R^2 = 1 - \frac{MSE}{D[y]}$$

В иностранной литературе Var(y)

- Изменяется от 0 до 1. Чем ближе к 1, тем сильнее зависимость
- Достаточно хорошие модели с $R^2 > 0.9$
- $R^2 = 1$ – функциональная зависимость между переменными

lection_5_compareModels

Процесс обучения (правило Видроу-Хоффа)



Минимизируем функцию $$S(\mathbf{w}) = \frac{1}{2}\sum_i \left(y^{(i)} - \phi(z^{(i)})\right)^2$$

Изменяем веса $$\Delta \mathbf{w} = -\eta \nabla S(\mathbf{w})$$

lection_5_lin_gd

Регрессор на базе дерева решений

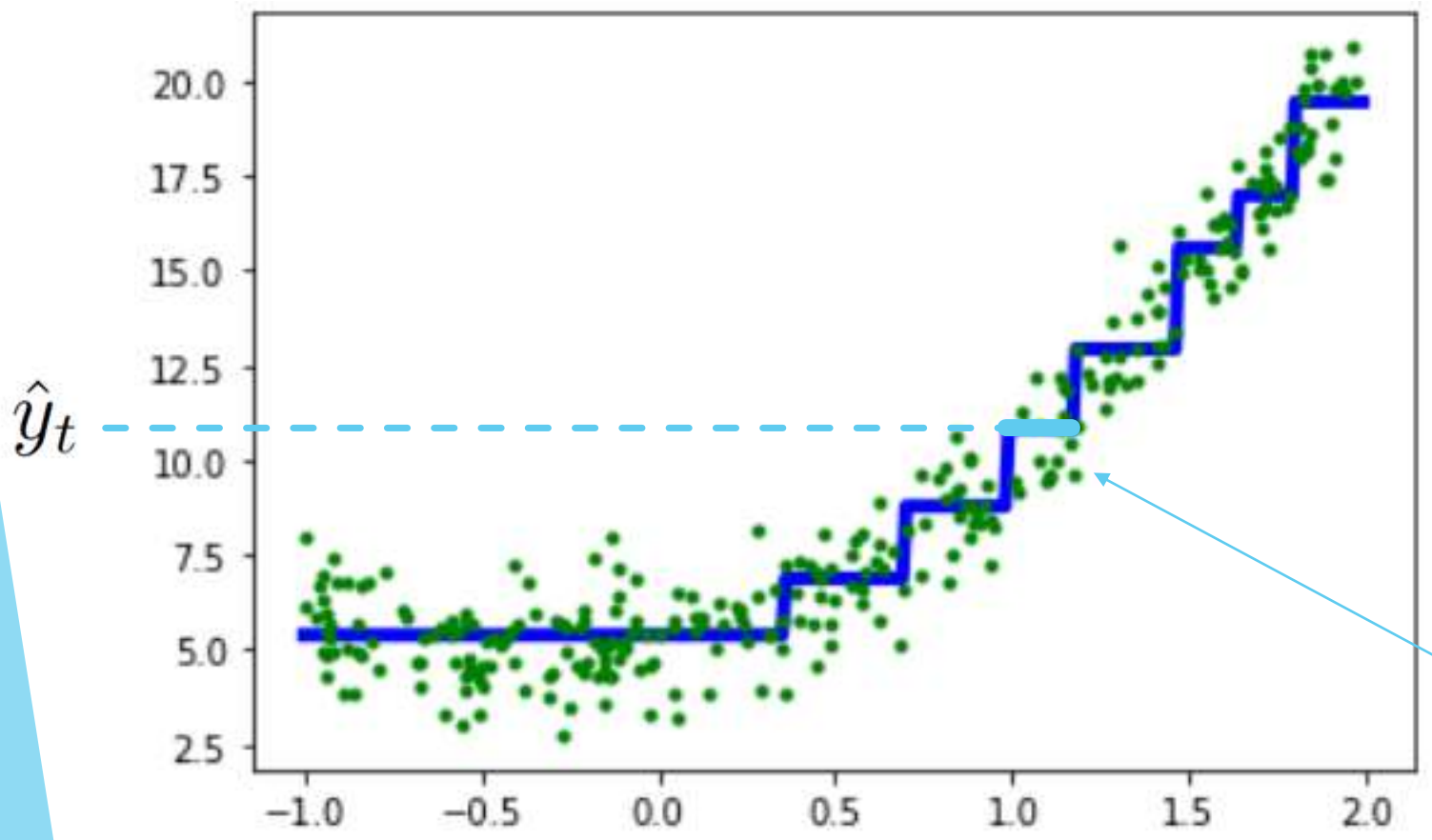
- Задача – найти расщепление признака, которое максимизирует прирост информации (сокращает неоднородность в дочерних узлах)
- В качестве меры вместо энтропии будем использовать MSE – внутриузловая дисперсия

$$I_t = MSE(t) = \frac{1}{N_t} \sum_{i \in D_t}^{n} \left(y^{(i)} - \hat{y}_t \right)^2$$

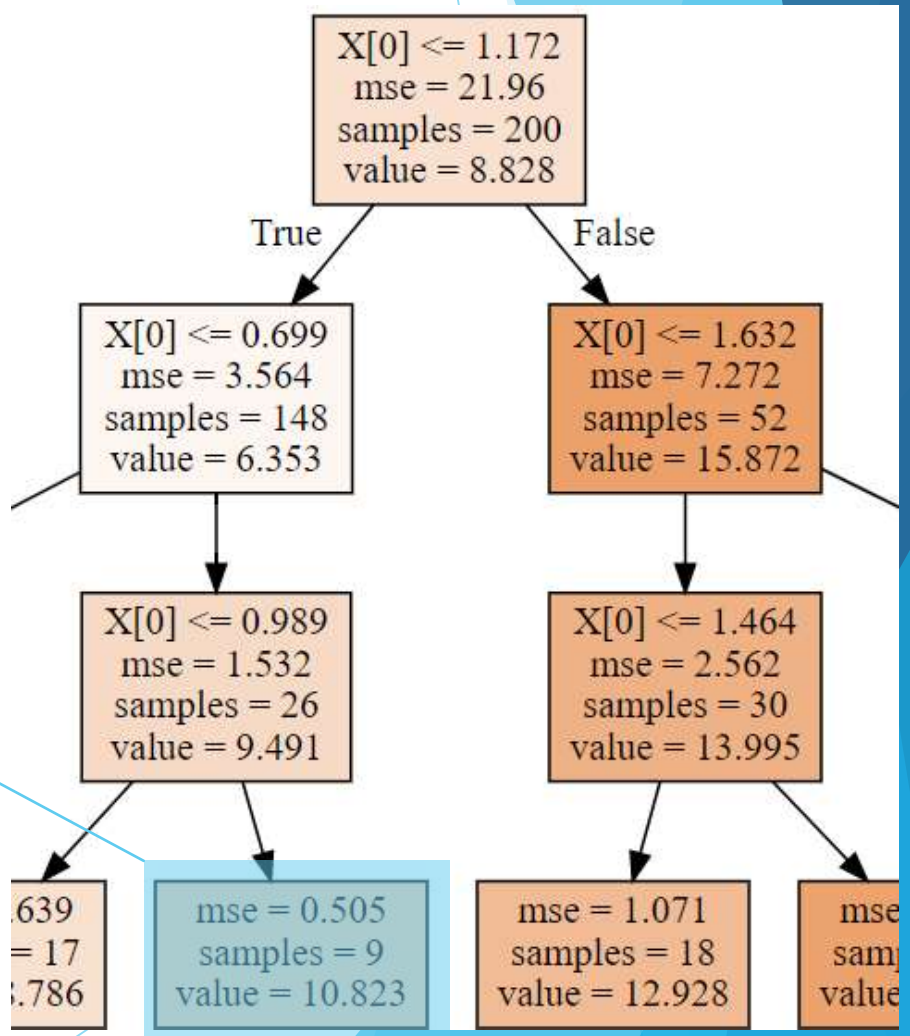
- Оценка внутри узла – среднее значение

$$\hat{y}_t = \frac{1}{N} \sum_{i \in D_t} y^{(i)}$$

Результат работы регрессора на базе дерева

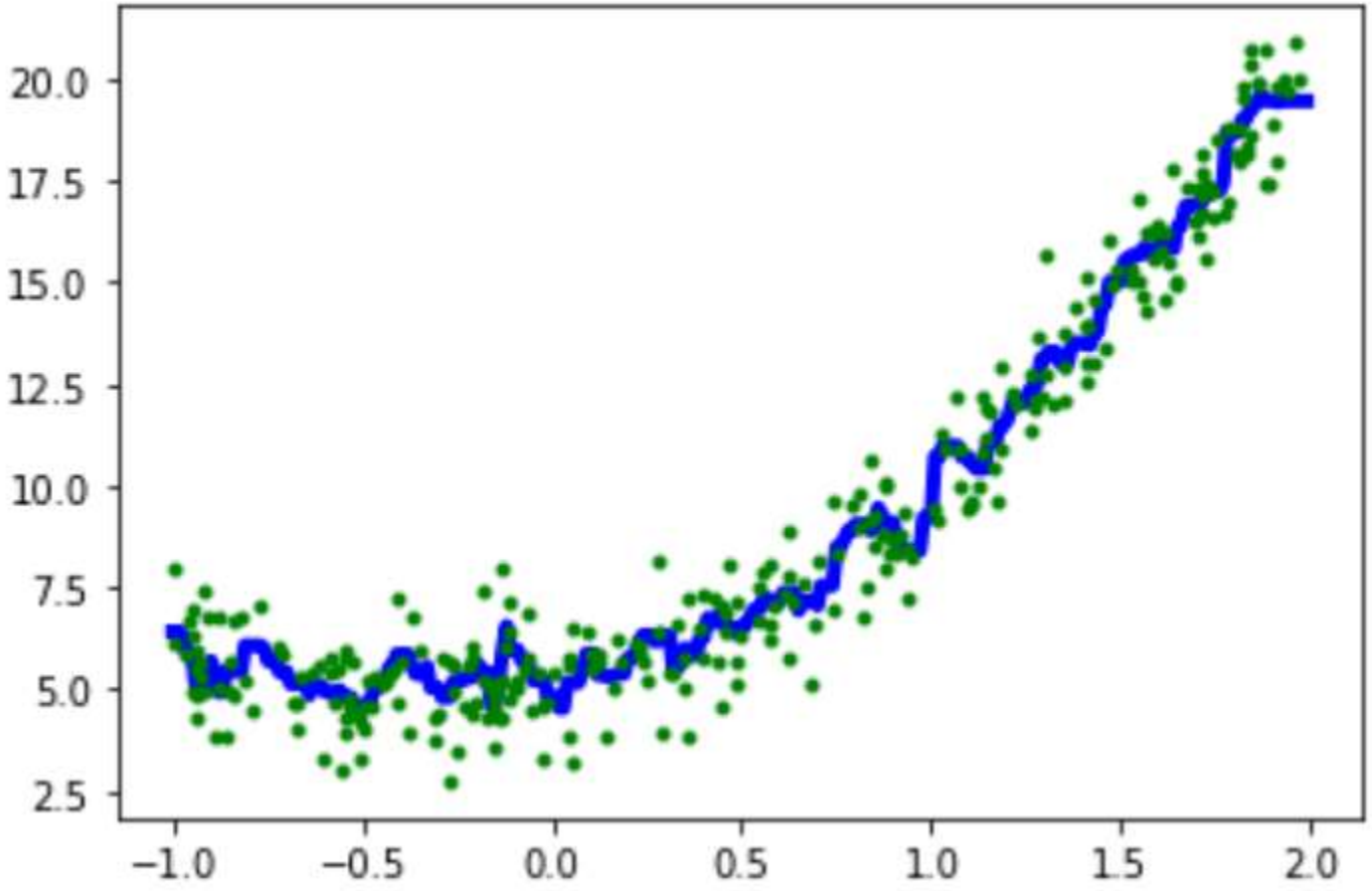


MSE=1.28 R2=0.95



Результат работы регрессора на базе kNN

Локальная интерполяция на основе ближайших соседей из тренировочного набора



n=5

MSE=1.039 R2=0.96