

Метод k ближайших соседей (kNN): обучение



Дано: обучающая выборка $X = (x_i, y_i)_{i=1}^{\ell}$



Задача классификация

(ответы из множества $\mathbb{Y} = \{1, \dots, K\}$)



Обучение модели:

– Запоминаем обучающую выборку X

Метод k ближайших соседей: применение



Дано: новый объект x

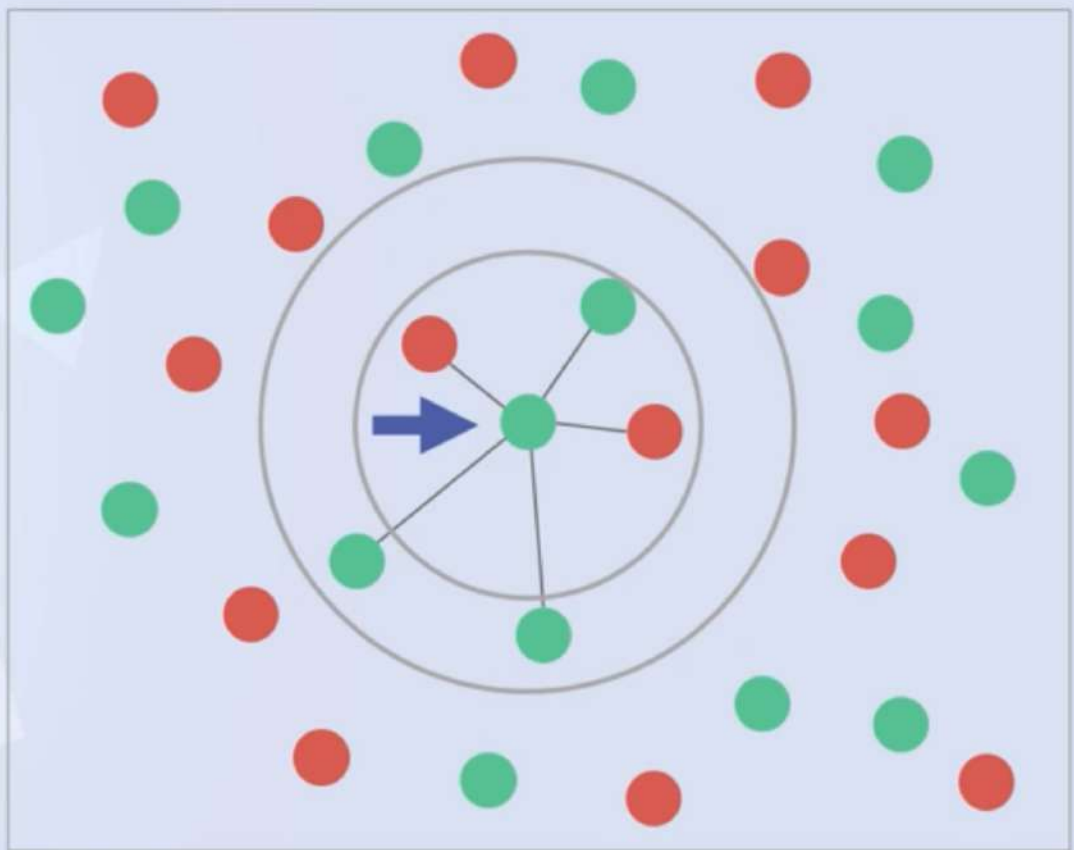


Применение модели:

- **Сортируем** объекты обучающей выборки по расстоянию до нового объекта: $\rho(x, x_{(1)}) \leq \rho(x, x_{(2)}) \leq \dots \leq \rho(x, x_{(\ell)})$
- **Выбираем** k ближайших объектов: $x_{(1)}, \dots, x_{(k)}$
- **Выдаём** наиболее популярный среди них класс:

$$a(x) = \arg \max_{y \in \mathbb{Y}} \sum_{i=1}^k [y_{(i)} = y]$$


Метод k ближайших соседей: применение




```
sklearn.neighbors.KNeighborsClassifier(  
n_neighbors=5, weights='uniform', algorithm='auto', le  
af_size=30, p=2, metric='minkowski', metric_params=No  
ne, n_jobs=None)
```

Простой и популярный случай: числовые признаки


Сколько раз в день вызывает такси	Средние расходы на такси в день	Как часто вызывал комфорт	Возраст	Согласился повысить категорию?
2	400	0.3	29	да
0.3	80	0	28	нет
...



Каждый объект описывается набором из d чисел – **вектором**



Если x – вектор, то x_i – его i -я координата

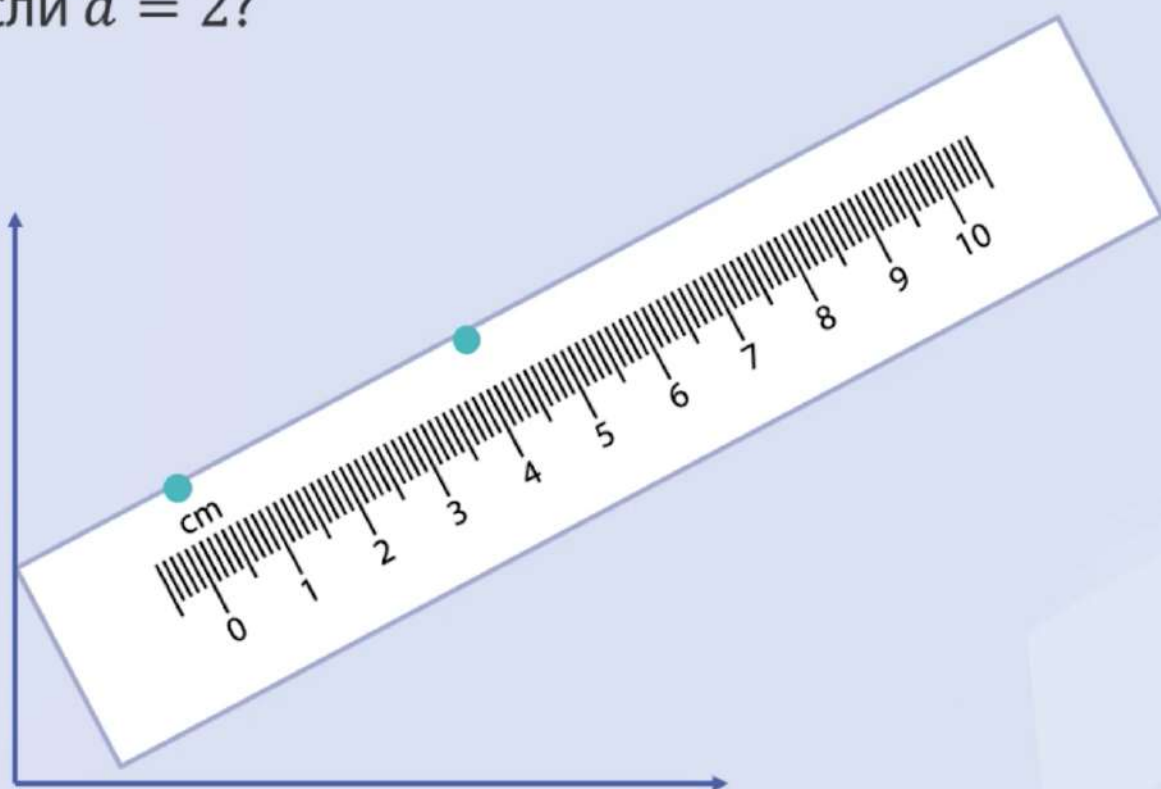



Если x_i – вектор, то x_{ij} – его j -я координата




Каждый объект описывается набором из d чисел – **вектором**

Что, если $d = 2$?





Метрика – обобщение расстояния на многомерные пространства

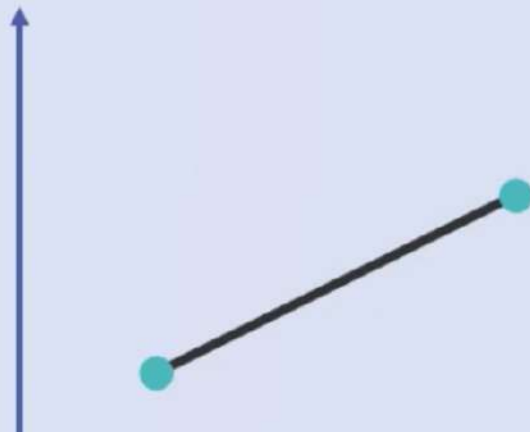


Метрика – это функция ρ с двумя аргументами, удовлетворяющая трём требованиям:

- $\rho(x, z) = 0$ тогда и только тогда, когда $x = z$
- $\rho(x, z) = \rho(z, x)$
- $\rho(x, z) \leq \rho(x, v) + \rho(v, z)$ –
неравенство треугольника

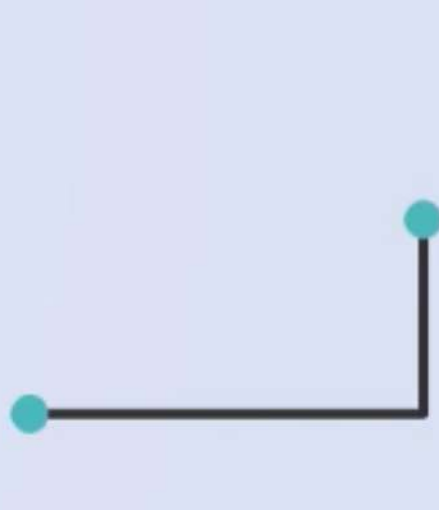
Евклидова метрика

$$\rho(x, z) = \sqrt{\sum_{j=1}^d (x_j - z_j)^2}$$



Манхэттенская метрика

$$\rho(x, z) = \sum_{j=1}^d |x_j - z_j|$$



```
sklearn.neighbors.KNeighborsClassifier(n_n  
eighbors=5, weights='uniform', algorithm=  
'auto', leaf_size=30, p=2, metric='minkowsk  
i', metric_params=None, n_jobs=None)
```

Более сложный случай: категориальные признаки

На каком классе чаще всего ездит	Ближайшее к дому метро	Способ оплаты	Согласился повысить категорию?
Эконом	Таганская	Карта	да
Комфорт	Юго-Западная	Наличные	нет
...

Простейшая метрика: подсчёт различий

$$\rho(x, z) = \sum_{j=1}^d [x_j \neq z_j]$$

Более сложная метрика: сравнение по частотам

j -й признак: на какой категории чаще всего ездит пассажир

Посчитаем для каждой категории, как часто пассажиры соглашаются повысить класс:

$$p_j(c) = \frac{\sum_{i=1}^{\ell} [x_{ij} = c][y_i = 1]}{\sum_{i=1}^{\ell} [x_{ij} = c]}$$

Эконом	Комфорт	Бизнес	Люкс
0.7	0.69	0.3	0

$$\rho(x, z) = \sum_{j=1}^d (p_j(x_j) - p_j(z_j))^2$$

Эконом	Комфорт	Бизнес	Люкс
0.7	0.69	0.3	0

Наличные	Карта
0.2	0.4

(эконом, наличные) и (комфорт, карта):

$$(0.7 - 0.69)^2 + (0.2 - 0.4)^2 = 0.0401$$

Частый выбор – бинарная функция потерь

$$L(y, a) = [a \neq y]$$

Функционал ошибки – доля ошибок (error rate)

$$Q(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} [a(x_i) \neq y_i]$$

Нередко измеряют долю верных ответов (accuracy):

$$Q(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} [a(x_i) = y_i]$$

Доля ошибок

$$Q(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} [a(x_i) \neq y_i]$$

Решаем задачу выявления редкого заболевания

950 здоровых ($y = +1$)

50 больных ($y = -1$)

Модель: $a(x) = +1$

Доля ошибок: 0.05

Всегда смотрите на баланс классов!

Резюме



Классификаторы часто оцениваются через долю ошибок



Доля ошибок может подвести при несбалансированных классах



Число соседей не получится подобрать по обучающей выборке

Отложенная выборка



Обучение



Слишком большое обучение – тестовая выборка нерепрезентативна

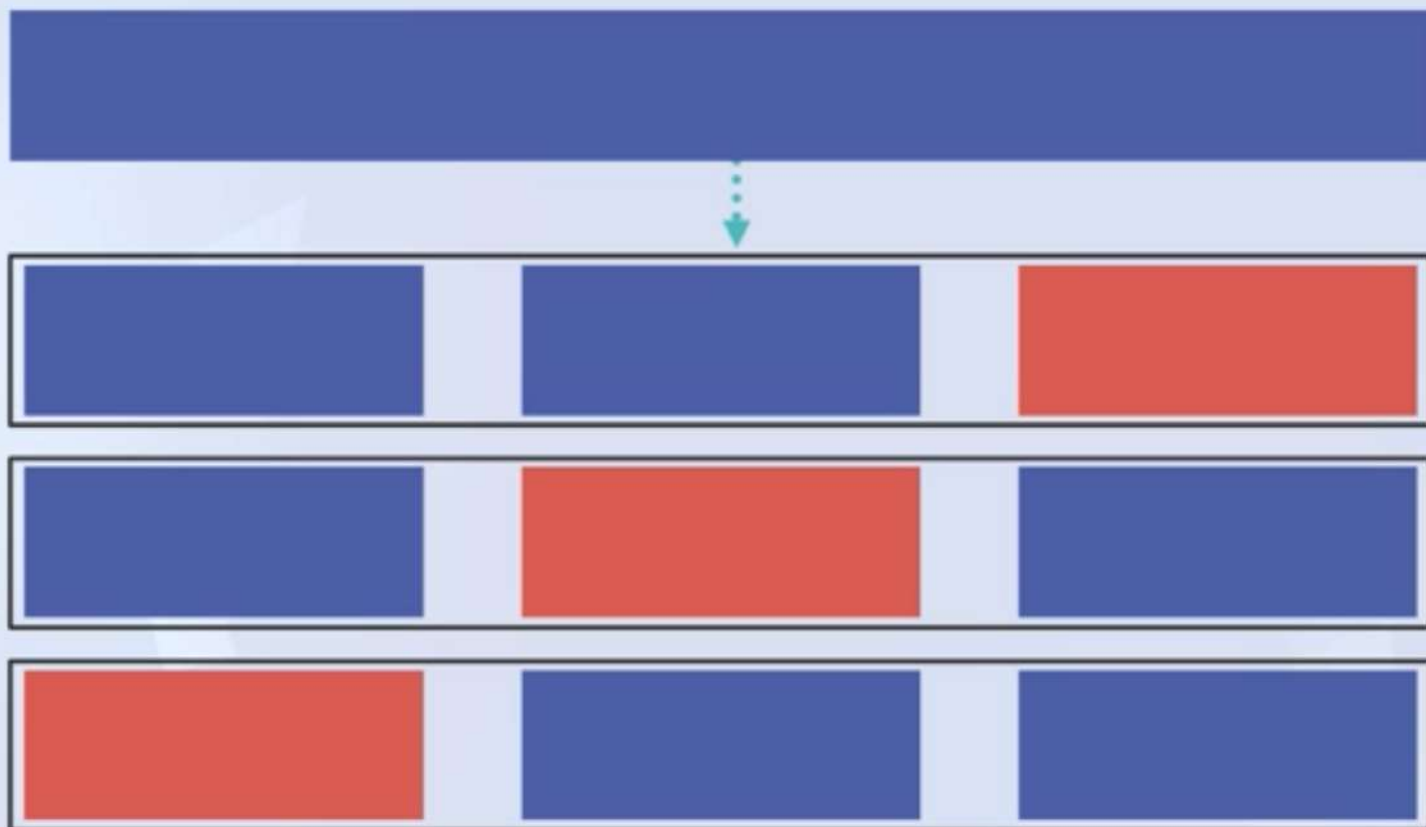
Тест



Слишком большой тест – модель не сможет обучиться

Обычно: 70/30, 80/20

Кросс-валидация



Кросс-валидация



Надёжнее отложенной выборки, но медленнее



Параметр – количество разбиений n (фолдов, folds)



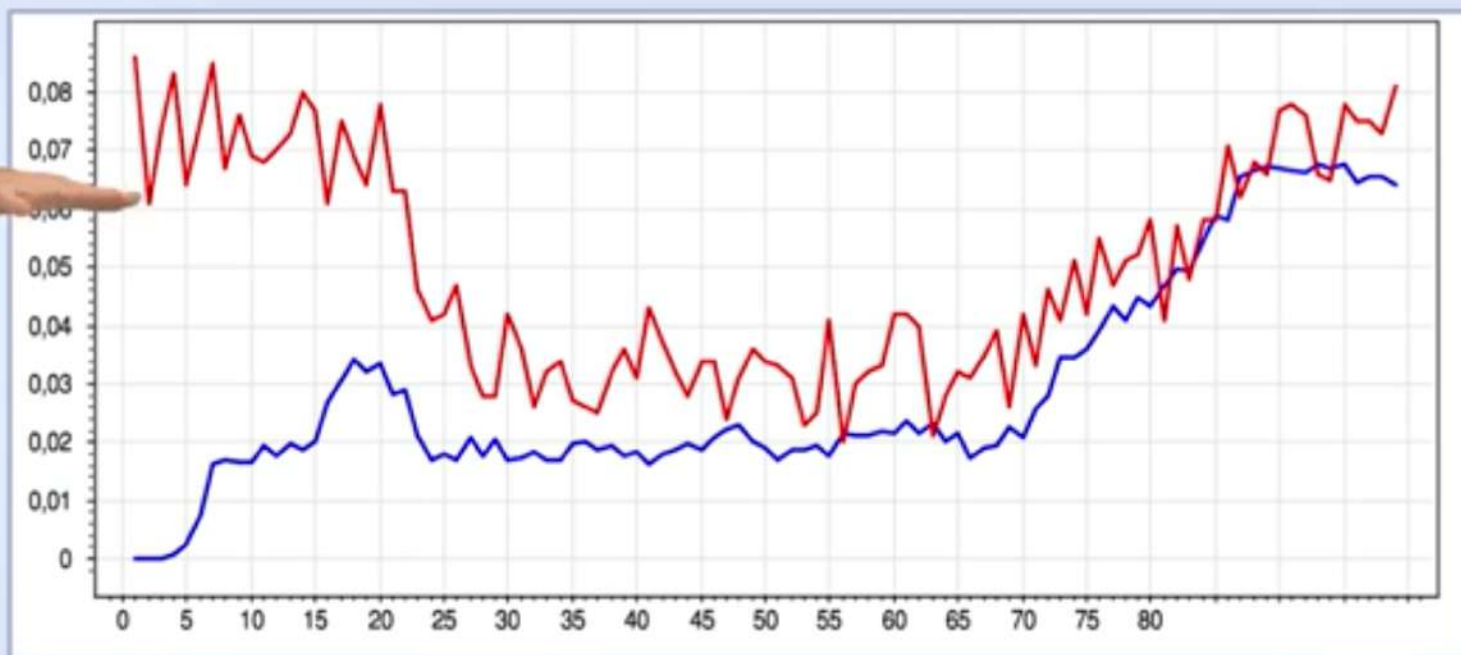
Хороший, но медленный вариант – $n = \ell$ (leave-one-out)



Обычно: $n = 3$ или $n = 5$ или $n = 10$

Подбор числа соседей

Частота ошибок



Число соседей q

Чуть больше терминов

После подбора всех гиперпараметров стоит проверить на совсем новых данных, что модель работает

Обучающая выборка – построение модели

Валидационная выборка – подбор гиперпараметров модели

Тестовая выборка – финальная оценка качества модели

Резюме



Чтобы понять, пригоден ли алгоритм, используют отложенную выборку или кросс-валидацию




Число соседей и метрику выбирают так, чтобы ошибка на внешних данных была как можно ниже



В конце построения модели имеет смысл проверить качество работы модели на полностью новых данных

Взвешенный kNN


$$a(x) = \arg \max_{y \in \mathbb{Y}} \sum_{i=1}^k w_i [y_{(i)} = y]$$

Варианты:

$$w_i = \frac{k + 1 - i}{k}$$

$$w_i = q^i$$

Не учитывают сами расстояния

Взвешенный kNN

$$a(x) = \arg \max_{y \in \mathbb{Y}} \sum_{i=1}^k w_i [y_{(i)} = y]$$

Парзеновское окно:

$$w_i = K \left(\frac{\rho(x, x_{(i)})}{h} \right)$$

K – ядро

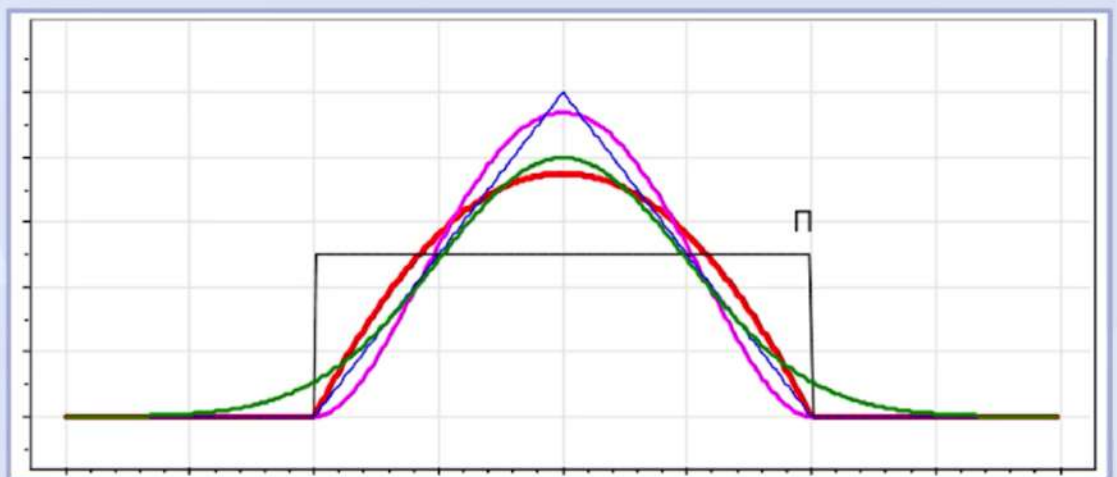
h – ширина окна

Ядра для весов

Гауссовское ядро:

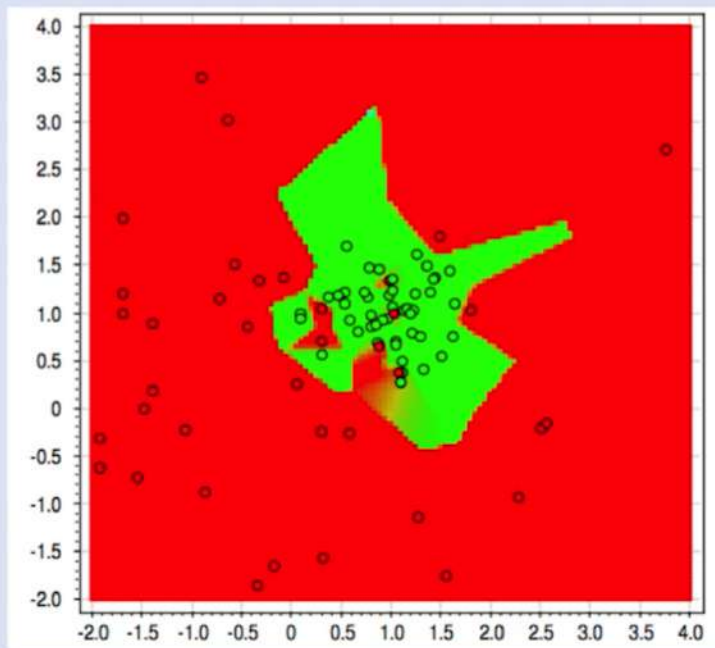
$$K(z) = (2\pi)^{-0.5} \exp\left(-\frac{1}{2}z^2\right)$$

И много других:



Ядра для весов

$$w_i = K\left(\frac{\rho(x, x_{(i)})}{h}\right)$$



$$h = 0.05$$

Метод k ближайших соседей в sklearn

```
sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None)
```

- `weights='uniform'` соответствует $w_i = 1$
- `weights='distance'` соответствует $w_i = 1/\rho(x, x_i)$

Резюме



В kNN может быть полезно учитывать, насколько близко расположен каждый сосед



Для этого можно добавить веса, которые тем больше, чем ближе сосед



В весах нужно подбирать ядро и ширину окна

Метод k ближайших соседей: применение



Дано: новый объект x



Применение модели:

- Сортируем объекты обучающей выборки по расстоянию до нового объекта: $\rho(x, x_{(1)}) \leq \rho(x, x_{(2)}) \leq \dots \leq \rho(x, x_{(\ell)})$
- Выбираем k ближайших объектов: $x_{(1)}, \dots, x_{(k)}$
- Усредняем ответы:

$$a(x) = \frac{1}{k} \sum_{i=1}^k y_{(i)}$$

Метод k ближайших соседей: применение

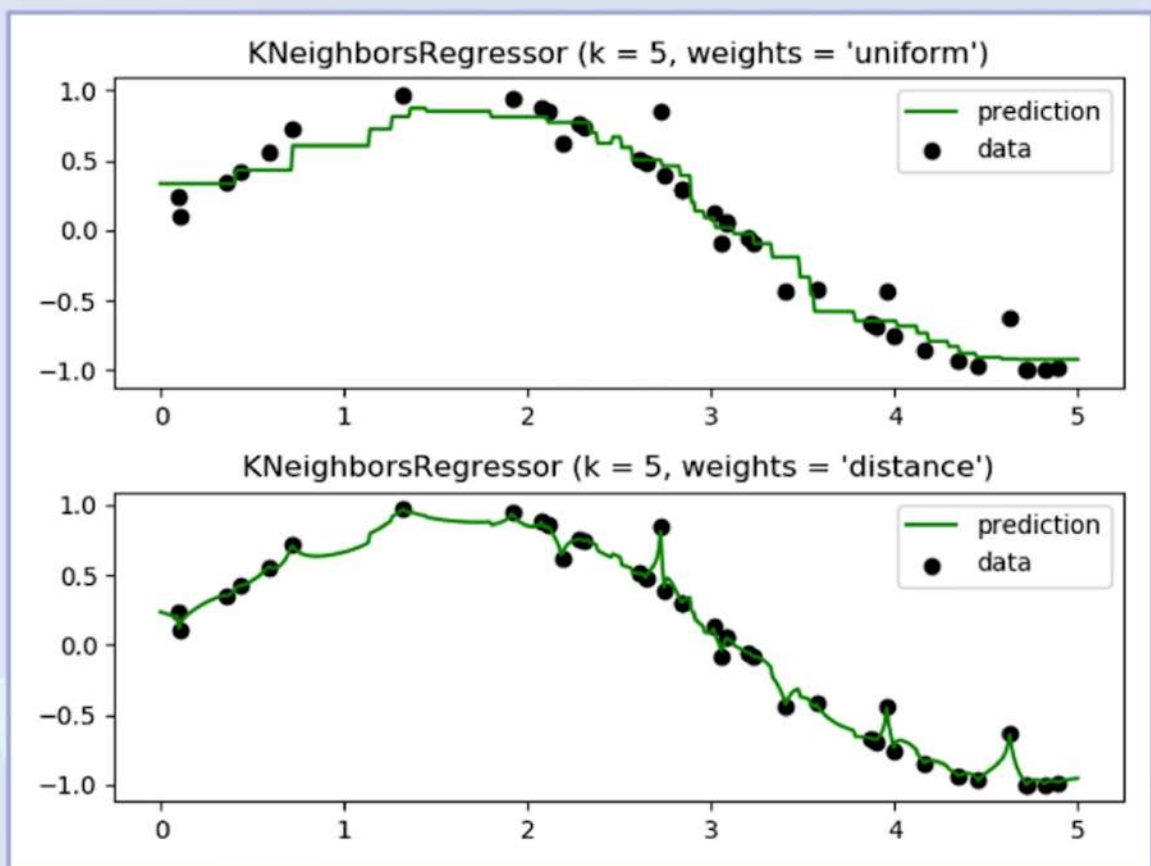
Можно добавить веса:

$$w_i = K \left(\frac{\rho(x, x_{(i)})}{h} \right)$$

$$a(x) = \frac{\sum_{i=1}^k w_i y_{(i)}}{\sum_{i=1}^k w_i}$$

Формула Надарая-Ватсона

Метод k ближайших соседей: применение



Функция потерь для регрессии

Частый выбор – **квадратичная функция потерь**

$$L(y, a) = (a - y)^2$$

Функционал ошибки – **среднеквадратичная ошибка (mean squared error, MSE)**

$$Q(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} (a(x_i) - y_i)^2$$

Функция потерь для регрессии

Ещё один вариант – **средняя абсолютная ошибка (mean absolute error, MAE)**

$$Q(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} |a(x_i) - y_i|$$

Слабее штрафует за серьёзные отклонения от правильного ответа

Резюме



Перейти в kNN от классификации к регрессии легко: заменяем выбор самого частого класса на усреднение ответов



Можно использовать веса



Частая функция потерь – квадратичная




Всё остальное аналогично – гиперпараметры подбираются через отложенную выборку и т.д.

Процесс построения модели



Плюсы kNN

- 
- Если данных много и для любого объекта найдётся похожий в обучающей выборке, то это лучшая модель
 - Очень простое обучение
 - Мало гиперпараметров
 - Бывают задачи, где гипотеза компактности уместна
 - Классификация изображений
 - Классификация текстов на много классов

Минусы kNN

- Часто другие модели оказываются лучше
- Надо хранить в памяти всю обучающую выборку
- Искать k ближайших соседей довольно долго
- Мало способов настроить модель