

Классификация

Метод k -го ближайшего соседа.

Примеры

Лекция 9

Классификация

Есть набор данных (X_i, y_i) , где $X_i = (x_{1i}, x_{2i}, \dots, x_{ni})$ — предикторы, y_i — отклик.

Мы настраиваем параметры некого алгоритма (модели) так, чтобы предикторам соответствовал отклик из обучающего набора данных.

Далее на вход алгоритма подаются новые данные — предикторы X_i^{new} , и алгоритм вычисляет соответствующее им значение отклика.

Похоже на регрессию? Да, но в задачах классификации множество y_i — дискретное и конечное (например, 0 и 1). Каждый отклик y_i указывает на группу (класс), к которому относятся наблюдения.

Другое название: **обучение с учителем** (supervised learning).

План

- ▶ метод k-го ближайшего соседа
- ▶ деревья классификации
- ▶ случайный лес
- ▶ gradient boosting machine

Метод k-го ближайшего соседа (K-Nearest Neighbors, kNN)

Еще недавно метод был очень популярен. Он очень прост, нечувствителен к выбросам и обладает полезными математическими свойствами.

Но

Он очень чувствителен к шуму в данных и к малоинформативным переменным.

“Скажи мне, кто твой друг, и я скажу кто ты” (Эврипид)

$k = 1$ — объект относится к тому же классу, что и его ближайший сосед.

Ближайший — значит максимально похожий. То есть снова возникает вопрос: как задать расстояние между объектами (наблюдениями)?

$$k > 1$$

Объект относится к тому классу, члены которого составляют большинство среди его k ближайших соседей.

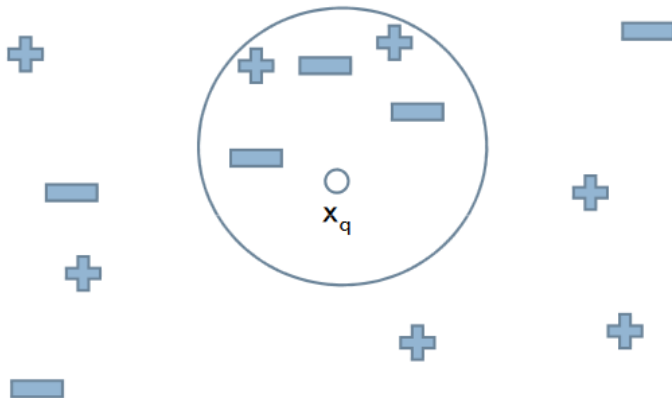


Figure 1: К какому классу относится x_q ? ($k=5$)

Сколько соседей выбрать?

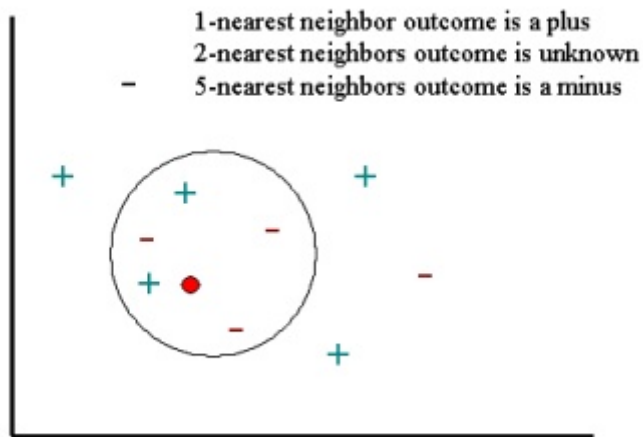


Figure 2:

Технические детали

- ▶ Если невозможно найти ровно k соседей, значение k увеличивают (вариант: выбирают случайным образом).
- ▶ Если невозможно определить, какой класс составляет большинство среди k соседей, значение k увеличивают (вариант: выбирают случайно).
- ▶ В случае задачи с двумя классами удобно использовать нечетные значения k .
- ▶ Правильно увеличивать k с ростом объема обучающей выборки n .
- ▶ k должно быть разным в зависимости от локальной плотности точек, но непонятно каким. . .

Модификация методов: взвешиваем влияние соседей

Каждому соседу приписывают вес, обратно пропорциональный расстоянию до него.

Нужны всего 2 управляющих параметра

1. k - число ближайших соседей.
2. Способ вычисления расстояния между объектами.

Математические подробности

Нет никаких предположений о распределении данных. При этом метод является состоятельным!

Т.е., не предполагая вообще ничего, мы, с ростом наших знаний (объема обучающей выборки), делаем при классификации все меньше и меньше ошибок.

Состоятельная оценка — это точечная оценка, сходящаяся по вероятности к оцениваемому параметру.

Одна из проблем классификации заключается в том, что теоретическое распределение элементов обучающей выборки неизвестно. Поэтому нельзя формально проверить, принадлежит ли распределение данной обучающей выборки к тому или иному классу.

Одним из вариантов решения этой проблемы являются методы классификации, состоятельные на любом распределении обучающих данных. Таким свойством обладает метод k -ближайших соседей.

Однако выяснилось, что универсально состоятельные методы могут сходитьсся (снижать риск ошибочной классификации с ростом обучающей выборки) как угодно плохо на некоторых распределениях обучающих данных.

Недостатки алгоритма

1. Необходимость хранить в памяти всю обучающую выборку.
2. Чувствительность к шуму и малоинформативным переменным.
3. Чувствительность к несбалансированности объемов классов.

Необходимость хранить в памяти всю обучающую выборку.

Ленивое обучение

Ленивое обучение (lazy learning) — это способ обучения, при которых обобщение данных обучающей выборки откладывается до тех пор, пока не будет сделан запрос к системе.

Это экономит силы, но: нужно хранить в памяти всю обучающую выборку.

Лекарство: хранить данные в виде R-tree или k-d tree.

Чувствительность к шуму и малоинформативным переменным

Лекарство: стандартизация и/или отбор признаков (feature selection).

Стандартизация предотвращает доминирование одной переменной над другими. **Не забывайте об этом простом средстве!**

Отбор признаков (feature selection, feature engineering):

- ▶ преобразования переменных (перенос начала координат в центр масс, переход к другим координатам).
- ▶ агрегация переменных (объединяем несколько переменных в одну).
- ▶ различные методы уменьшения размерности (например, метод главных компонент).

Чувствительность к несбалансированности объемов классов

Несбалансированность объемов классов приводит к тому, что доминирующий класс может доминировать и в большинстве окрестностей.

Лекарство: выравнивание количества наблюдений в разных классах на этапе обучения.

Как выбрать значение k ?

Этому вопросу посвящена следующая презентация. Сначала нужно познакомиться с понятием **кросс-валидация**.

Метод k-го ближайшего соседа в R

- ▶ **class**, **knn()**, $O(n)$ (время работы)
- ▶ **FNN** (fast k-nearest neighbor), **knn()**, $O(\log(n))$
- ▶ **kknn** (Weighted k-Nearest Neighbors), **kknn()**
- ▶ **RWeka** — интерфейс к библиотеке WEKA
- ▶ **caret** — почти все методы классификации и регрессии, объединенные под одним интерфейсом.

Представление наблюдений в виде точек с координатами не является необходимым

Если мы можем сосчитать расстояния между точками, то можем классифицировать методом kNN.

Классификация методом kNN в R

1. Создать обучающую и тестовую выборки.
2. Определить оптимальное число k .
3. Оценить качество прогноза.

Пример прогнозирование продаж нового сорта пива

Наблюдения об объемах продаж четырех известных сортов пива в 150 городах и мнение экспертов об уровне продаж в тех же городах нового сорта пива.

- ▶ продукт₁, ..., продукт₄ — объем продаж пива сортов 1, ..., 4
- ▶ уровень — уровень продаж нового сорта пива (по мнению экспертов): 1 - низкие, 2 - средние, 3 - высокие.

Нам нужно спрогнозировать уровень продаж нового сорта пива для городов, которых нет в наборе, по данным о продажах в этих городах сортов пива 1-4.

Чтение и проверка данных

```
sales <- read.table("discrim.txt", header=T,  
                    sep=";", row.names=1)
```

Что находится в наборе?

```
sales[1:5,] # вместо head(sales)
```

##	продукт1	продукт2	продукт3	продукт4	уровень
## 1	50	33	14	2	1
## 2	64	28	56	22	3
## 3	65	28	46	15	2
## 4	67	31	56	24	3
## 5	63	28	51	15	3

```
dim(sales)
```

```
## [1] 150 5
```

Формируем обучающую и тестовую выборки

```
# Чтобы результаты у меня и у вас совпадали
set.seed(1234)
# Формируем случайную подвыборку из  $150 * 1/3 = 50$  чисел
test.num <- sample(1:nrow(sales), 50, replace = FALSE)
# Тестовая выборка
test <- sales[test.num, 1:4]
# Обучающая выборка
train <- sales[-test.num, 1:4]
# Код класса для обучающей выборки
cl <- sales[-test.num, 5]
dim(test)
```

```
## [1] 50  4
```

```
dim(train)
```

```
## [1] 100  4
```

Классификация данных тестовой выборки

```
# Подключаем библиотеку class
library(class) # recommended
# Распознаем класс объектов из тестовой выборки
predicted.cl <- knn(train, test, cl, k = 3)
```

```
predicted.cl
```

```
## [1] 1 2 1 2 3 1 3 3 1 3 2 2 1 3 1 2 3 1 2 1 1 1 3 1 2 3 2 2
## [36] 1 3 2 1 2 2 1 1 1 3 1 2 1 3 1
## Levels: 1 2 3
```

```
sales[test.num, 5]
```

```
## [1] 1 2 1 2 3 1 3 3 1 3 2 2 1 3 1 2 3 1 3 1 1 1 3 1 2 3 2 2
## [36] 1 3 2 1 3 2 1 1 1 3 1 2 1 3 1
```

Проверяем соответствие распознанного класса
истинному классу

```
table(predicted.cl, sales[test.num, 5])
```

```
##
```

```
## predicted.cl  1  2  3
```

```
##           1 22  0  0
```

```
##           2  0 13  2
```

```
##           3  0  0 13
```

Определим значение k: простейшая кросс-валидация

```
err <- rep(0,15)
for (i in 1:15)
{
  pred.knn <- knn(train, test, cl, k = i)
  err[i] <- sum(pred.knn != sales[test.num, 5])
}
err
```

```
## [1] 2 2 2 2 1 1 1 1 1 1 1 1 1 1
```

```
table(knn(train, test, cl, k = 5),
      sales[test.num, 5])
```

```
##
```

```
##      1  2  3
```

```
##  1 22  0  0
```

```
##  2  0 13  1
```

```
##  3  0  0 14
```

Спрогнозируем уровень продаж в новом городе

```
new.town <- data.frame(66,25,40,17)  
knn(sales[,5], new.town, sales[,5], k = 5)
```

```
## [1] 2
```

```
## Levels: 1 2 3
```

Пример. Классификация вин

Данные в файле `wine.txt`.

Измеряются 13 характеристик химического состава вина для трех сортов вин, выращенных в одной и той же области Италии разными виноделами. Необходимо по значениям имеющихся переменных определить сорт вина.

Задача приобрела большую известность и активно обсуждалась. Как оказалось, статистические методы могут различать вина лучше профессиональных сомелье.

Чтение и проверка

```
wine <- read.table("wine.txt", header=T, sep="\t") wine[1:5,]
```

```
##      Input1 Input2 Input3 Input4 Input5 Input6 Input7 Input8 Input9  
  
## 1  14.23    1.71    2.43   15.6    127    2.80    3.06    0.28    2  
## 2  13.20    1.78    2.14   11.2    100    2.65    2.76    0.26    1  
## 3  13.16    2.36    2.67   18.6    101    2.80    3.24    0.30    2  
## 4  14.37    1.95    2.50   16.8    113    3.85    3.49    0.24    2  
## 5  13.24    2.59    2.87   21.0    118    2.80    2.69    0.39    1  
##      Input11 Input12 Input13 Desired1.3.  
## 1      1.04     3.92     1065           0  
## 2      1.05     3.40     1050           0  
## 3      1.03     3.17     1185           0  
## 4      0.86     3.45     1480           0  
## 5      1.04     2.93      735           0
```

```
dim(wine)
```

```
## [1] 178 14
```

Стандартизация

```
wine.st <- scale(wine[, -ncol(wine)], center = TRUE,  
                scale = TRUE)  
  
# или нормализация  
# normalize <- function(x) {  
#           ((x - min(x)) / (max(x) - min(x)))  
#           }
```

Попробуйте не сделать стандартизацию (`wine.st=wine`) и посмотрите на результаты классификации.

Формируем обучающую и тестовую выборки

```
set.seed(1234)

test.num <- sample(1:nrow(wine.st), 60, replace = FALSE)

test <- wine.st[test.num, -ncol(wine)]
train <- wine.st[-test.num, -ncol(wine)]

cl <- wine[-test.num, ncol(wine)]
```

Определим значение k

```
library(class)
err <- rep(0,15)
for (i in 1:15)
{
  pred.knn <- knn(train, test, cl, k = i)
  err[i] <- sum(pred.knn != wine[test.num, ncol(wine)])
}
err
```

```
## [1] 3 4 4 3 3 3 2 2 2 2 2 2 2 2 3
```

Распознаем класс и проверяем ошибки

```
predicted.cl <- knn(train, test, cl, k = 7)
table(predicted.cl, wine[test.num, ncol(wine)])
```

```
##
## predicted.cl  0  1  2
##              0 19  0  1
##              1  0 16  1
##              2  0  0 23
```

Переобучение

Overfitting — чрезмерная подгонка или переобучение.
Использование чрезмерно сложной модели для описания данных.

Подгонка данных полиномом степени M

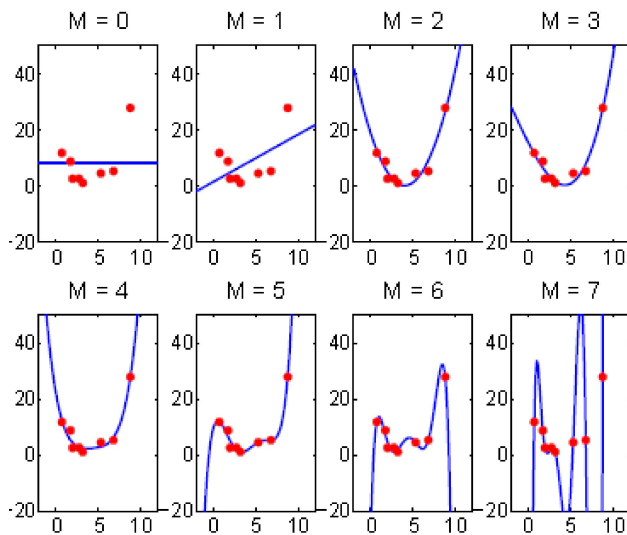


Figure 1: Какая подгонка лучше?

Какая модель лучше?

Надо выбрать ту из моделей, которая наилучшим образом подгоняет данные.

- ▶ То есть ту, у которой наименьшая средняя ошибка (или наименьшая средняя квадратичная ошибка).

Лирическое отступление

Портной научился хорошо шить костюмы для мистера Смита.

Пока он шьет для Смита — все идет хорошо.

Но если он будет шить для Джонса по мерке, снятой со Смита, результат может быть намного хуже.

Зачем нужна модель?

Чтобы успешно предсказывать будущие наблюдения.

- ▶ то есть для нового значения x предсказать значение Y с маленькой погрешностью.

Наилучшей будет та модель, которая лучше всех будет предсказывать, то есть описывать (подгонять) **новые** данные.

Аппроксимация подгоняет имеющиеся данные, классификация и регрессия — новые данные.

Вопрос для самопроверки

Почему линейная модель не может подгонять данные лучше, чем квадратичная?

Проблема

- ▶ У нас нет будущих значений. . .

Решение

- ▶ Так сделаем их из прошлых!
- ▶ Отберем часть наблюдений и объявим их “будущими”.

Следующий пример заимствован из “Cross-validation for detecting and preventing overfitting” Andrew W. Moore
(www.cs.cmu.edu/~awm)

Пример: выбор наилучшего полинома для регрессии

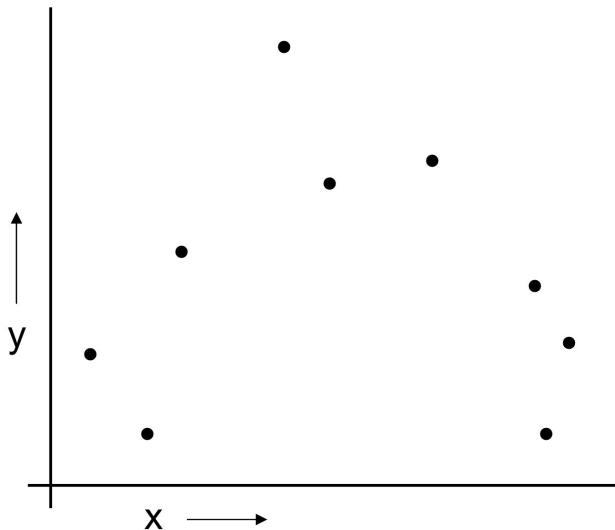


Figure 2: Данные

Три варианта

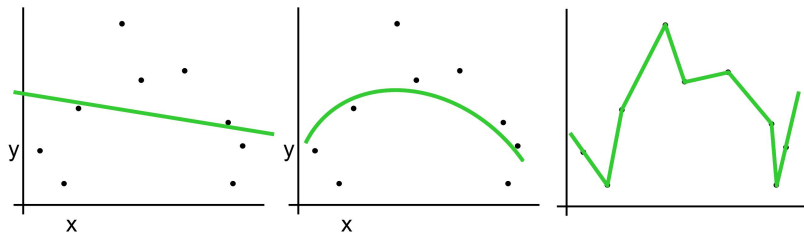


Figure 3: Какой из трех вариантов лучше?

Метод тестового множества

Случайным образом выберем 30% всех наблюдений и назовем их **тестовой выборкой**.

Остальные 70% наблюдений назовем **обучающей выборкой**.

- ▶ Обучающая выборка (training set) — имеющиеся наблюдения (образцы).
- ▶ Тестовая выборка (test set) — будущие наблюдения.

По наблюдениям из обучающей выборки построим модель.

Проверим модель на тестовой выборке.

Обучающая и тестовая выборки

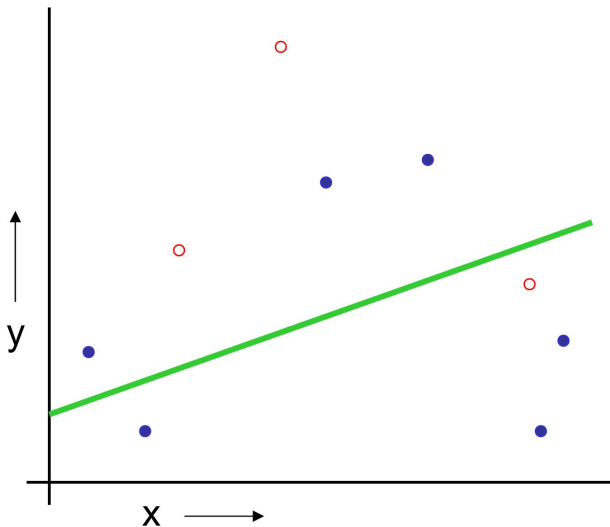


Figure 4: Линейная регрессия

Проверка подгонки на тестовой выборке

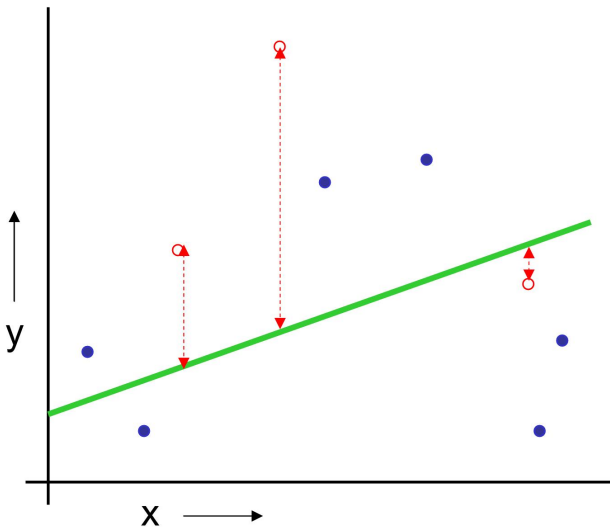


Figure 5: Линейная регрессия, Mean Squared Error = 2.4

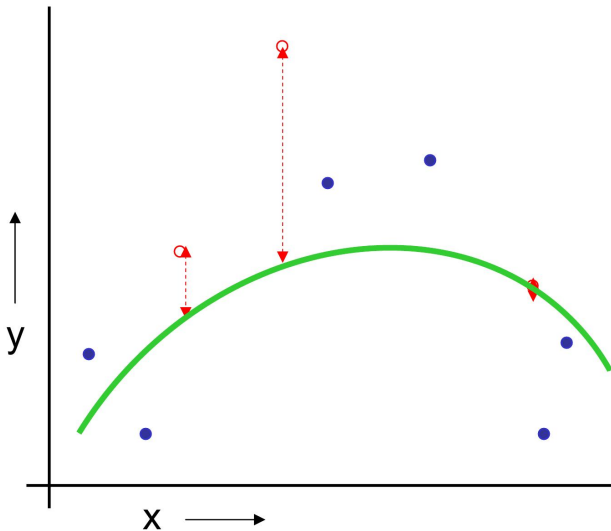


Figure 6: Квадратичная регрессия, $MSE = 0.9$

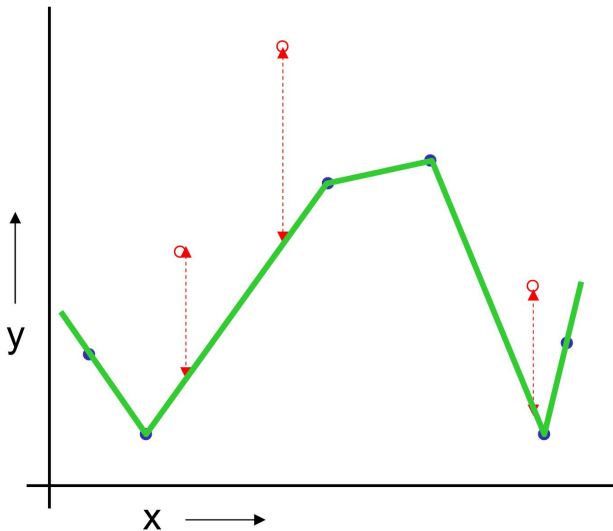


Figure 7: Линия по точкам, $MSE = 2.2$

Обсуждение метода тестового множества

Достоинства

- ▶ Понятен.
- ▶ Очень просто реализуется;

Недостатки

- ▶ Расточителен: при построении модели отбрасывается 30% данных
- ▶ Если данных мало, то как распределятся точки между обучающей и тестовой выборками? Дело случая. А это влияет на результат оценивания качества метода.

Другими словами: оценка качества модели с помощью тестового множества имеет большую дисперсию

Проверка посредством исключенных наблюдений (leave-one-out cross validation, LOOCV)

Пусть у нас имеется n точек (наблюдений).

- ▶ Предыдущую процедуру выполняем n раз.
- ▶ Но: тестовое множество теперь состоит из одной точки, каждый раз новой (обучающее множество состоит из оставшихся $n-1$ точек).
- ▶ За n шагов перебираем все точки множества. Каждый раз подсчитываем значение квадрата ошибки на тестовом наблюдении.
- ▶ Посчитаем среднее значение квадратов ошибок по всем n проверкам. Оно и даст нам оценку качества подгонки.

Линейная регрессия

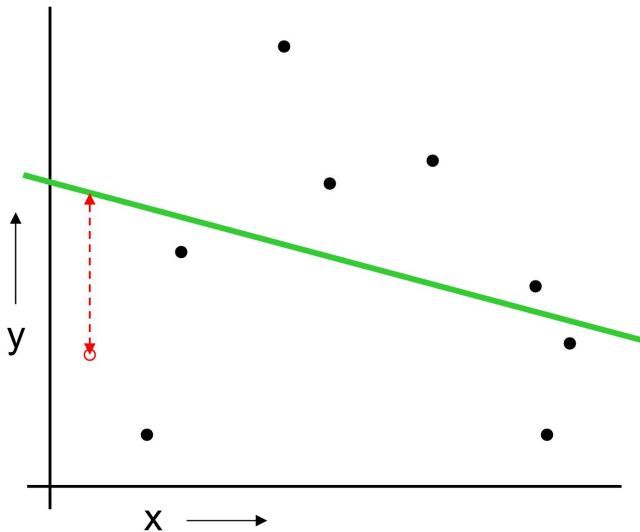


Figure 8: Проверка для первой точки

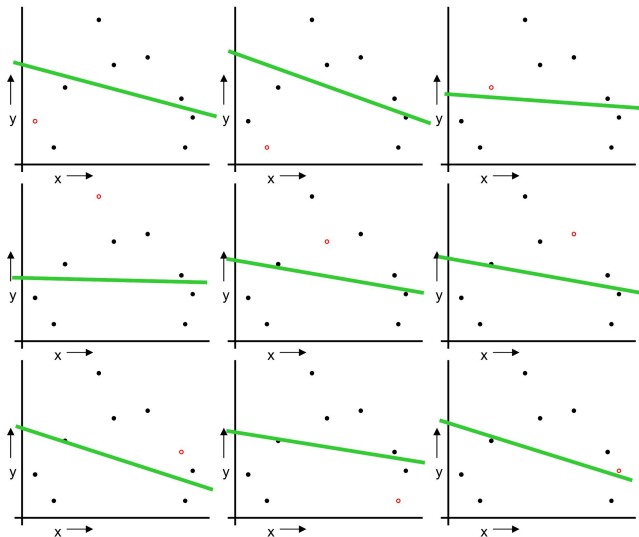


Figure 9: LOOCV, $MSE = 2.12$

Средние значения квадратов ошибок

- ▶ Для линейной регрессии: $MSE = 2.12$
- ▶ Для квадратичной регрессии: $MSE = 0.962$
- ▶ Для линии, проведенной по точкам: $MSE = 3.33$

Сравнение методов

Метод исключенного наблюдения не расходует много данных.

Но: он требует больше вычислений.

Можно ли как-то объединить достоинства методов, сократив расходы на вычисления и более экономно расходуя имеющиеся данные?

k-кратная кросс-валидация (k-fold cross-validation)

Случайным образом разобьем выборку на k одинаковых частей.
В рассматриваемом примере $k=3$.

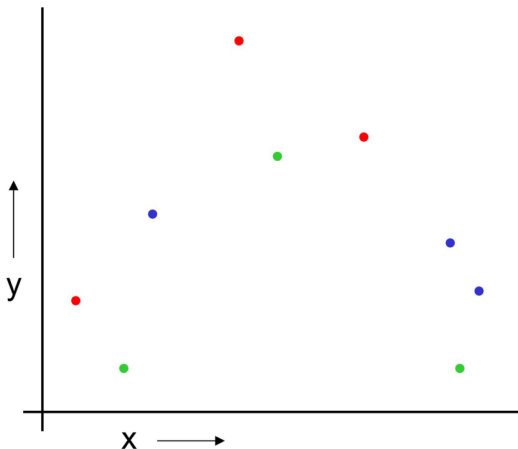


Figure 10: Точки из разных частей отмечены красным, синим и зеленым цветом.

Алгоритм

- ▶ Первая часть наблюдений (из k частей) — тестовая выборка.
- ▶ Все остальные наблюдения — обучающая выборка.
- ▶ Сосчитаем сумму квадратов ошибок для точек из тестового множества.

Линейная регрессия

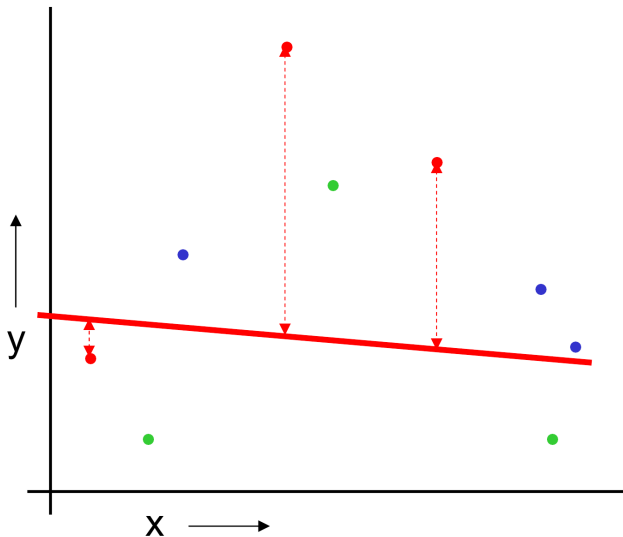


Figure 11: Для первой части выборки

Определяем обучающую выборку и тестовую выборку заново

- ▶ Вторая часть – тестовая выборка.
- ▶ Все остальные наблюдения – обучающая выборка.
- ▶ Сосчитаем сумму квадратов ошибок для точек из тестового множества.

и т. д.

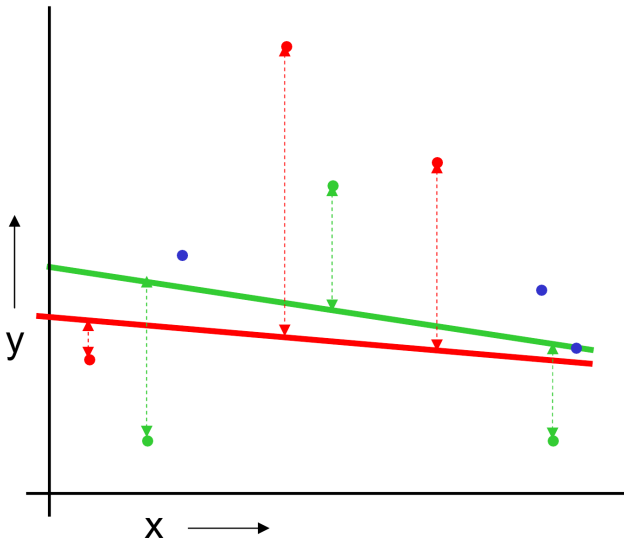


Figure 12: Для первой и второй частей выборки

Осталось сосчитать среднее значение квадратов ошибок

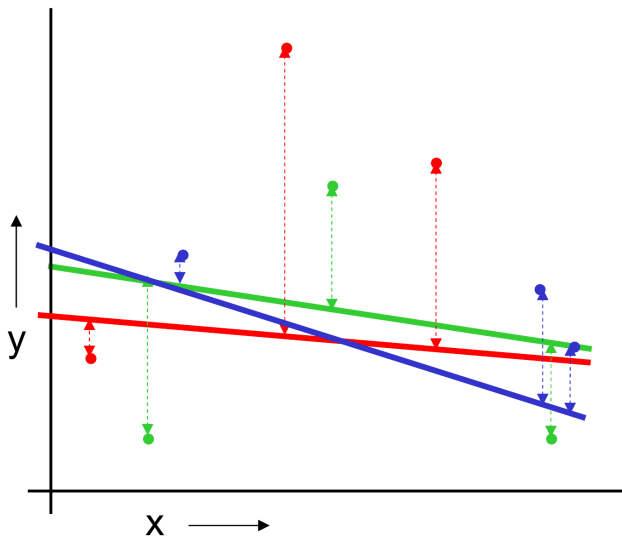


Figure 13: Линейная регрессия, $MSE(3fold) = 2.05$

Сравним результаты k-кратной кросс-валидации

- ▶ Для линейной регрессии: $MSE = 2.05$
- ▶ Для квадратичной регрессии: $MSE = 1.11$
- ▶ Для линии, проведенной по точкам: $MSE = 2.93$

Выбор метода кросс-валидации

- ▶ **3-кратная кросс-валидация:** чуть более экономна, чем метод тестового множества; требует немного больше вычислений.
- ▶ **10-кратная кросс-валидация:** исключает из обучения 10% данных, чем метод тестового множества; требует в 10 раз больше вычислений чем метод тестового множества, но меньше чем метод исключенного наблюдения.
- ▶ **n-кратная кросс-валидация:** совпадает с методом исключенного наблюдения (LOOCV).

Обычно выбираемые методы лежат в диапазоне от 10-кратной кросс-валидации до метода исключенного наблюдения.

Так как же выбрать значение k в методе kNN ?

1. Вычислим методом LOOCV среднюю ошибку для моделей с разным значением k
2. Выберем модель, давшую наименьшую ошибку, обучим ее на всем наборе данных и будем использовать для прогнозов.

<i>Algorithm</i>	TRAINERR	10-fold-CV-ERR	Choice
$K=1$			
$K=2$			
$K=3$			
$K=4$			
$K=5$			
$K=6$			

Figure 14:

ЧаВо по выбору k

Почему мы использовали LOOCV, а не, допустим, 10-кратную кросс-валидацию?

В методе kNN вычисления выполняются очень быстро, так что 10-кратная кросс-валидация не имеет вычислительных преимуществ перед LOOCV (и при этом расходует больше наблюдений).

Почему мы остановились на $k=6$?

Увидели, что после $k=3$ ошибка начала расти с ростом k .

Является ли значение k , полученное LOOCV, глобальным или локальным оптимумом?

Конечно это локальный оптимум.

Что делать, если придется искать k в широком диапазоне, например, от 1 до 1000?

1. Создать сетку и проверять значения k , например, через 50.
2. Перебирать по схеме $k=2^i$: $k=1$, $k=2$, $k=4$, $k=8$, $k=16$, $k=32$, $k=64$... $k=1024$.
3. Использовать какой-либо из методов поиска локального экстремума.

Где еще можно использовать кросс-валидацию?

- ▶ Для выбора наилучшего метода классификации.
- ▶ Для выбора наилучшего метода регрессии.
- ▶ Для определения степени полинома в линейной регрессионной модели.
- ▶ Для отбора признаков (feature selection).
- ▶ ...

С чем нужно разобраться

- ▶ Почему нельзя использовать ошибку, полученную на обучающем множестве, для выбора алгоритма обучения.
- ▶ Почему нельзя использовать ошибку, полученную на обучающем множестве, для выбора параметров алгоритма обучения (например, k в kNN).
- ▶ Кросс-валидация методом тестового множества (test-set cross-validation).
- ▶ Кросс-валидация методом исключенного наблюдения (leave-one-out cross-validation).
- ▶ k -кратная кросс-валидация (k -fold cross-validation)
- ▶ Использование кросс-валидации для выбора алгоритма классификации и параметров этого алгоритма.

Поиск компромисса между смещением и дисперсией

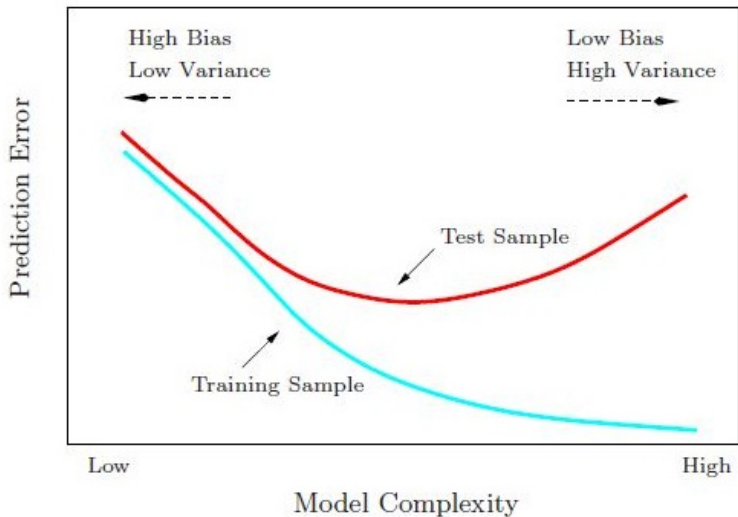


Figure 15: