

# Кластеризация

Анализ данных— осень 2016

# Цели кластеризации

Дано:

- $X = \{\vec{x}^{(i)}\}_{i=1}^m, \vec{x}^{(i)} \in \mathbb{R}^n$  -- выборка

Зачастую надо:

- Разбить выборку на группы схожих объектов, чтобы работать с ними по отдельности (классификация, регрессия, вот это все)

Иногда надо:

- Сократить объем хранимых данных
- Выделить нетипичные объекты
- Построить иерархию множества объектов

# Подводные камни

Решение задачи кластеризации неоднозначно:

- Точной постановки задачи не существует
- Непонятно (обычно), на сколько кластеров можно разбить данные – надо выбирать самому
- Непонятно, как оценивать качество кластеризации – существует множество метрик
- Результат кластеризации сильно зависит от выбора метрики

# Два основных подхода

## Статистические методы

- ЕМ-алгоритм
- Его частный случай – k-means

## Иерархические методы

- Агломеративная иерархическая кластеризация
- Еще есть сети Кохонена, нечеткие методы, что-то еще
- Мы разберем k-means и иерархическую кластеризацию
- За всем остальным – к Воронцову!

# Метод К-средних (K-means algorithm)

# Основная идея

Основная идея состоит в группировки немаркированных наблюдений в заданное количество кластеров (классов) путём минимизации расстояний до их центров

# Общая схема алгоритма

Задать начальные значения центроидов кластеров

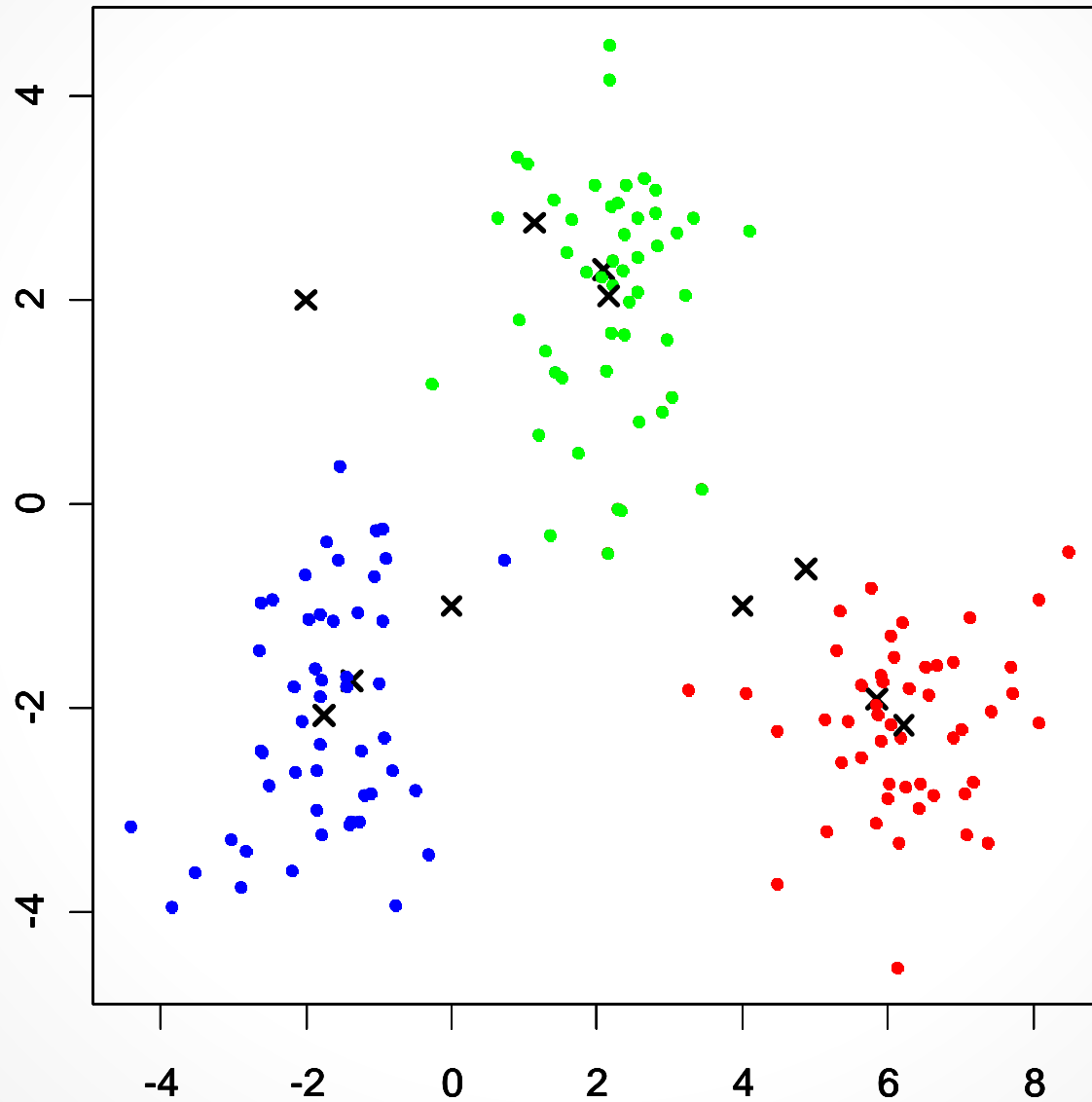
Повторять {

    присвоить наблюдениям номер кластера с ближайшим к  
    ним центром

    передвинуть центроиды кластеров к среднему значению  
    координат их членов

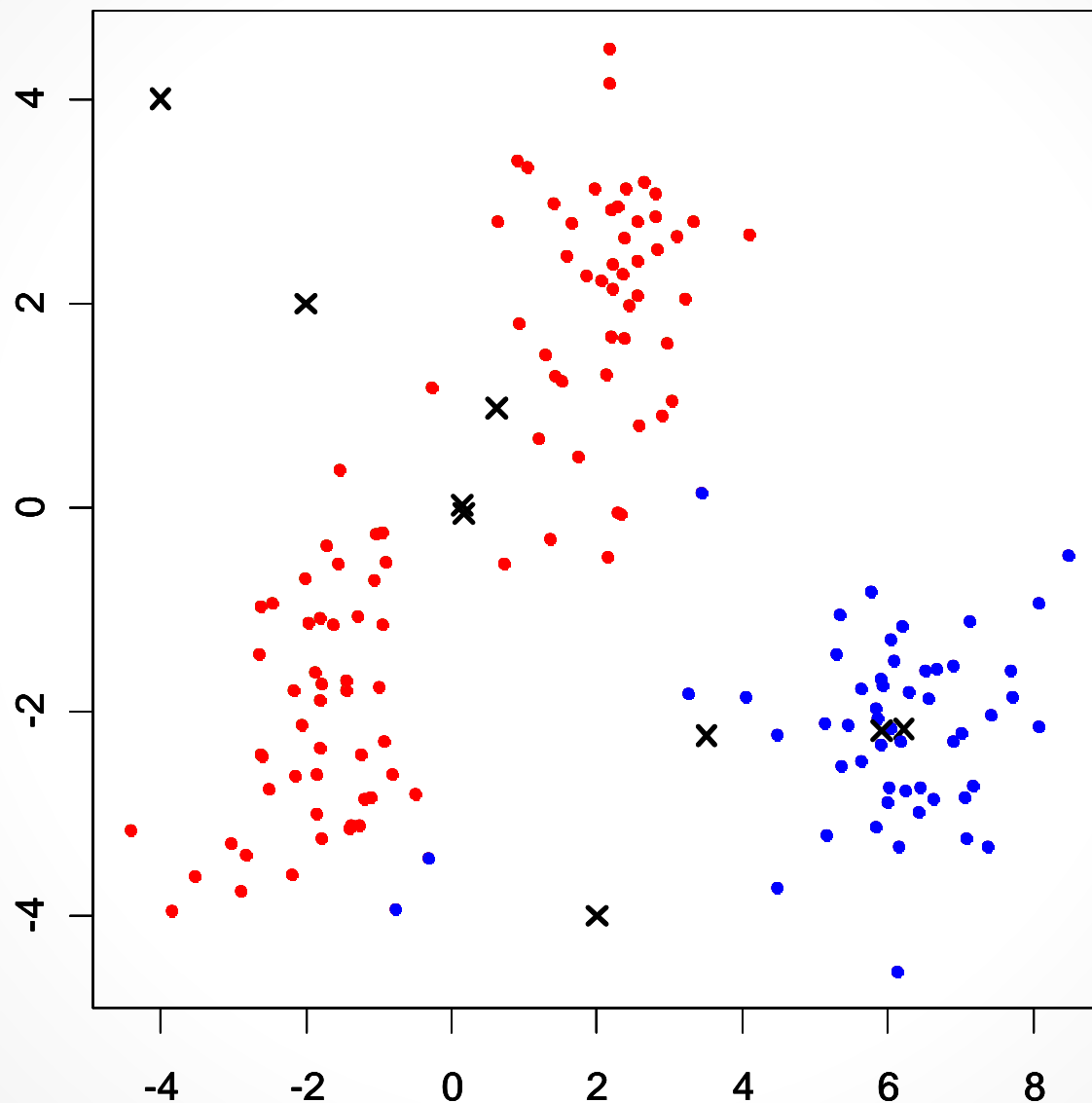
}

# Иллюстрация работы метода





# Влияние начальной инициализации



# Функция потерь

$K$  — количество классов,  $c^{(i)}$  — класс  $i$ -го наблюдения,  $i \in \{1; \dots; m\}$

$\vec{\mu}_k = [1 \times n]$  — центроид  $k$ -го класса,  $k \in \{1; \dots; K\}$

$$J(c^{(1)}, \dots, c^{(m)}, \vec{\mu}_1, \dots, \vec{\mu}_K) = \frac{1}{m} \sum_{i=1}^m \|\vec{x}^{(i)} - \vec{\mu}_{c^{(i)}}\|^2$$

Более формальный алгоритм:


Повторять {

для  $i = 1$  до  $m$   $c^{(i)} :=$  индекс ближнего центроида


для  $k = 1$  до  $K$   $\vec{\mu}_k := \text{mean}(\vec{x}^{(i)} \in \text{кластер } k)$

}

$\min_{c^{(1)}, \dots, c^{(m)}} J$



$\min_{\vec{\mu}_1, \dots, \vec{\mu}_K} J$



# Метод К-средних в R

Пусть  $X$  — матрица наблюдений

```
km <- kmeans(X, centers = K, nstart = 10, iter.max = 20)
```

```
K-means clustering with 2 clusters of sizes 50, 50
```

```
Cluster means:
```

```
      [,1]      [,2]  
1  0.98398589  1.03541527  
2 -0.03894685 -0.02637371
```

```
Clustering vector:
```

```
[1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
[38] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
[75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
Within cluster sum of squares by cluster:
```

```
[1] 8.535306 10.700694  
(between_SS / total_SS = 73.9 %)
```

```
Available components:
```

```
[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"  
[6] "betweenss"    "size"         "iter"         "ifault"
```

# Скользкие моменты

- Почему метод сходится за конечное время?

Потому что на самом деле центроиды хоть и находятся в  $\mathbb{R}^n$ , есть не более  $K^m$  способов их расположения

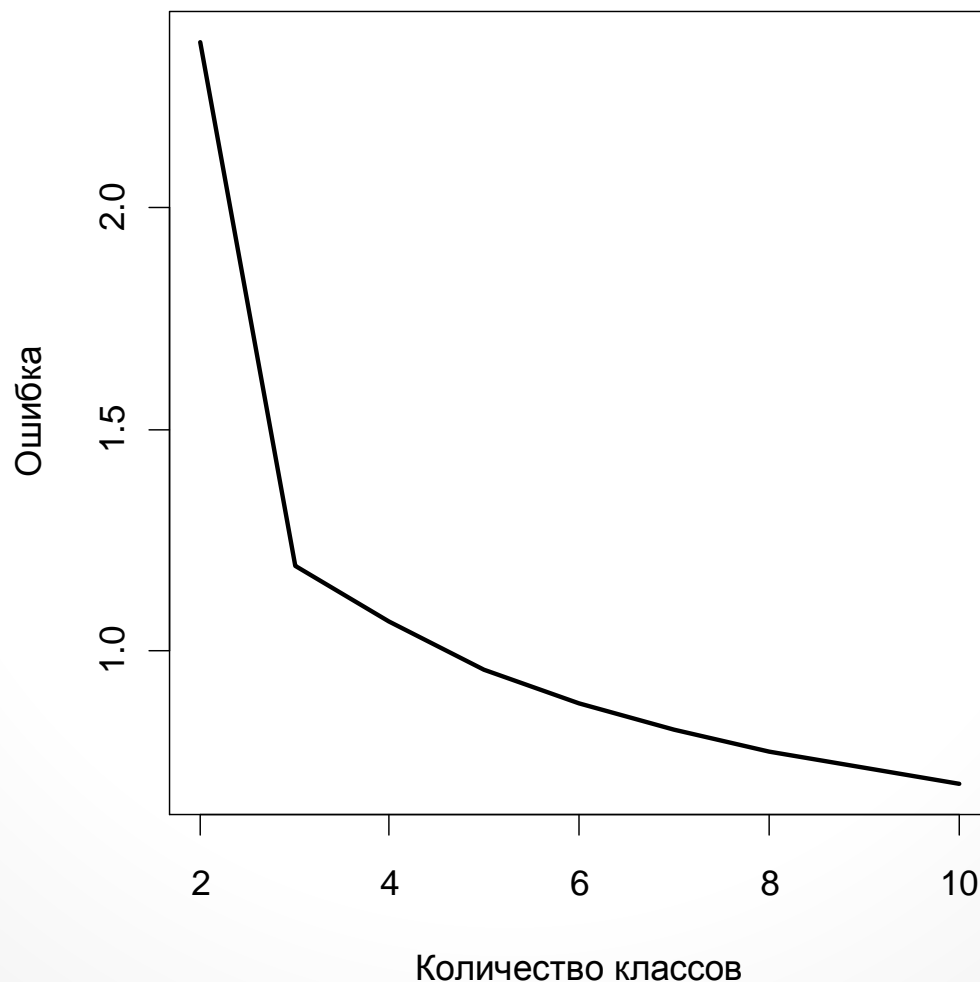
- Как выбрать начальные позиции центроидов?

Можно провести несколько кластеризаций со случайной инициализацией центроидов. Или выбрать один случайно из точек, а затем выбирать из оставшихся с вероятностью, пропорциональной их удаленности от ближайшего центроида.

- Как выбрать количество кластеров?

# Выбор количества классов

Количество классов рекомендуется увеличивать до тех пор, пока сохраняется быстрое снижение внутригрупповой ошибки



# Агломеративная иерархическая кластеризация

# Основная идея

Алгоритмы иерархической кластеризации можно разделить на два основных типа:

- Нисходящие алгоритмы, которые разбивают выборку на всё более и более мелкие кластеры.
- Агломеративные алгоритмы, в которых объекты объединяются во всё более и более крупные кластеры.

# Общая схема алгоритма

Поместить все элементы в отдельные кластеры

Повторять {

    выбрать два ближайши кластера

    объединить их элементы в один кластер

}



# Расстояние между кластерами

Как задать расстояние между кластерами? В общем случае – формула Ланса-Уильямса.

$$\begin{aligned} R(U \cup V, S) \\ = \alpha_U R(U, S) + \alpha_V R(V, S) + \beta R(U, V) + \gamma |R(U, S) - R(V, S)| \end{aligned}$$

Несколько частных случаев:

- $R(W, S) = \min \rho(w, s)$
- $R(W, S) = \max \rho(w, s)$
- $R(W, S) = \frac{|S||W|}{|S|+|W|} \rho^2(\sum_{w \in W} \frac{w}{|W|}, \sum_{s \in S} \frac{s}{|S|})$

Какая лучше?

Последняя (потому что обладает “хорошими” свойствами)

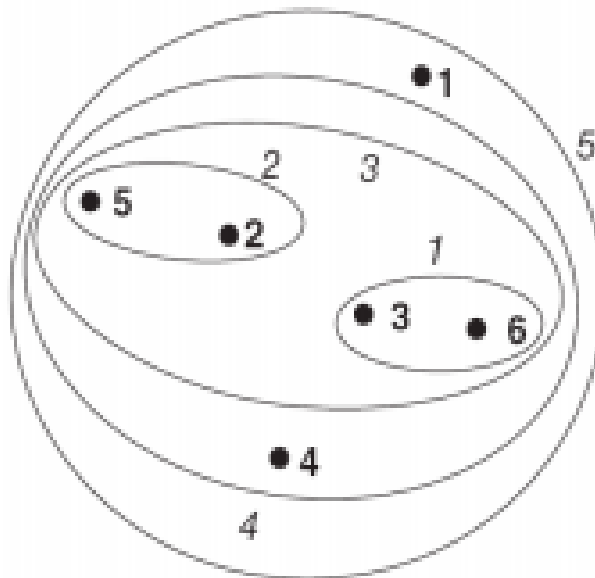
# Более формальное описание алгоритма

- 1) Задаем множество кластеров  $C_1: t=1$  ;  $C_t = \{x_1, \dots, x_l\}$
- 2) Для всех  $t = 2..l$  {
  - 3) В  $C_{t-1}$  найти два ближайших кластера  
 $(U, V) = \arg \min R(U, V); R_t = R(U, V)$
  - 4) убираем кластеры  $U, V$ ; добавляем  $W = U \cup V$
  - 5) для всех  $S \in C_t$  {
    - 6) вычисляем расстояние  $R(W, S)$}}

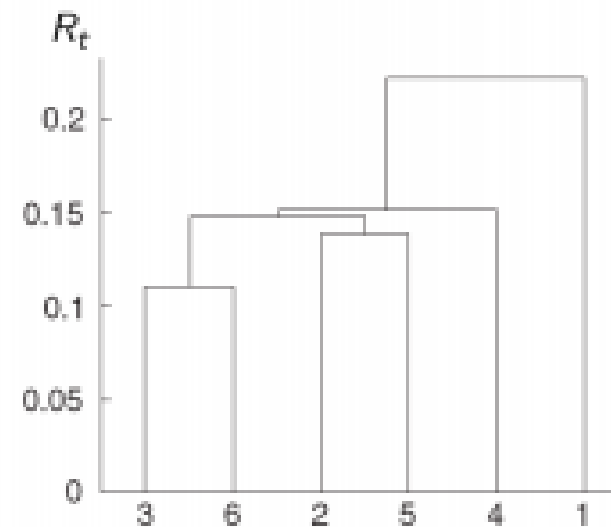
# Результат работы алгоритма

## 1. Расстояние ближнего соседа:

Диаграмма вложения



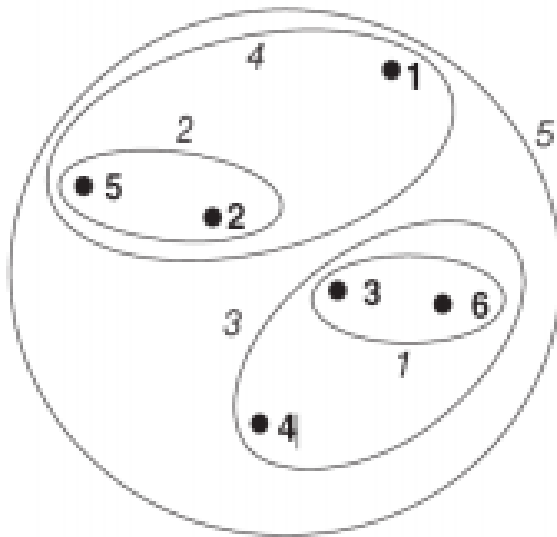
Дендрограмма



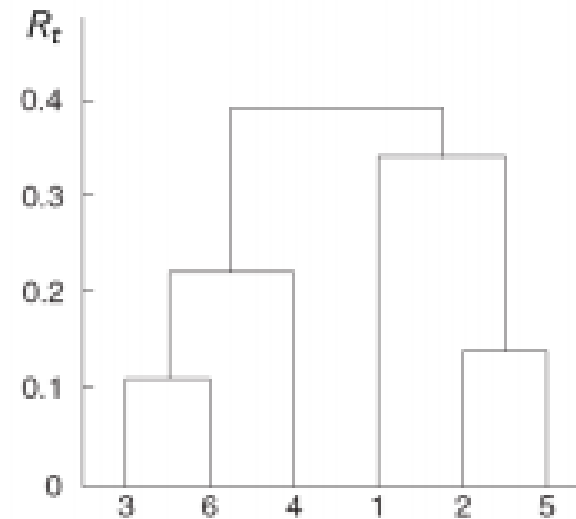
# Результат работы алгоритма

## 2. Расстояние дальнего соседа:

Диаграмма вложения

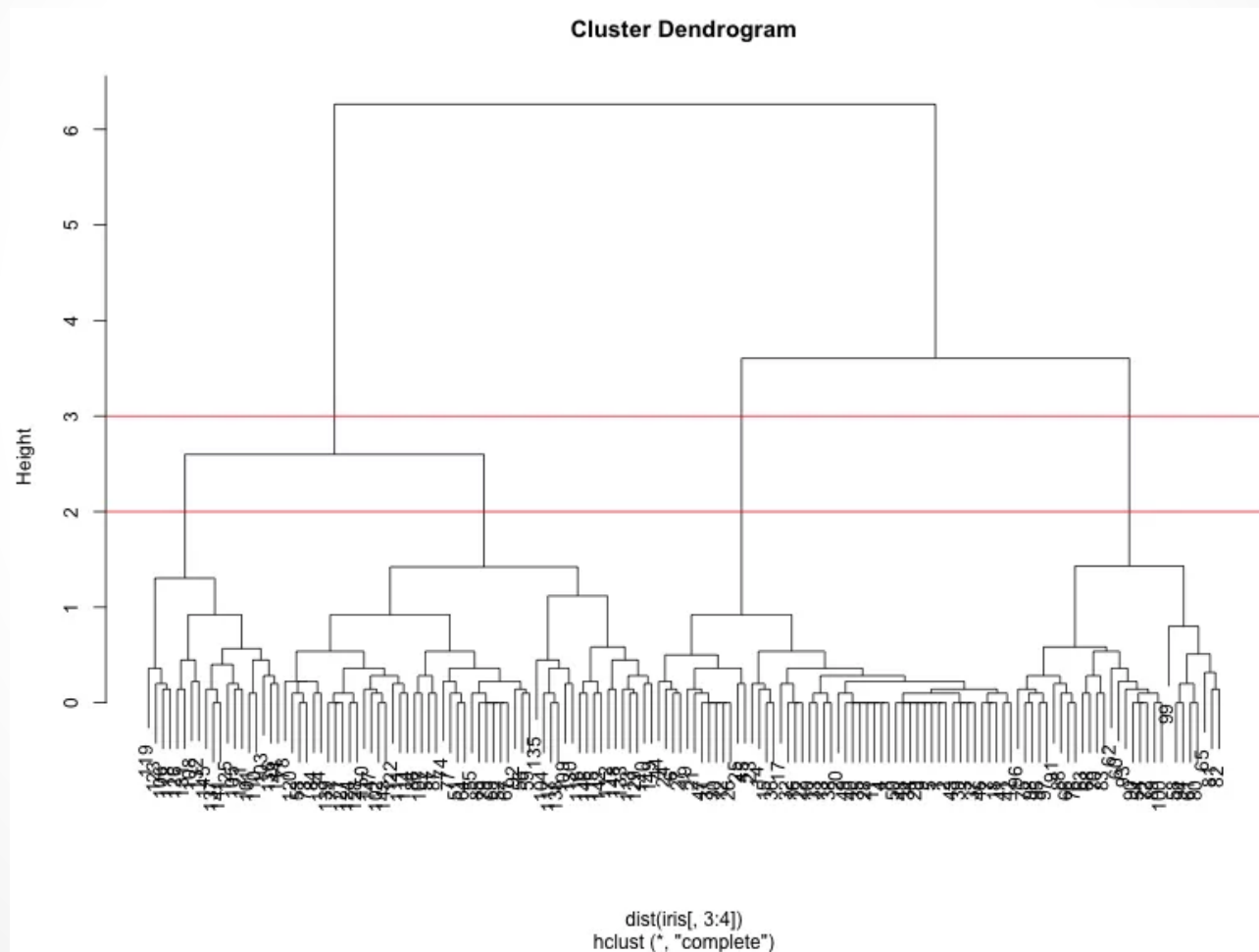


Дендрограмма



# Алгоритм в R

```
clusters = hclust(dist(iris[, 3:4]))  
plot(clusters)
```



# Домашнее задание

В файле «[grades.csv](#)» содержатся оценки 304-х студентов по 9-ти предметам

Вашей задачей является разделение этих студентов на академические группы, которое должно осуществляться, исходя из их успеваемости

Используйте оба метода, сравните результаты (количество кластеров, внутригрупповая дисперсия, что-нибудь еще)