

Программирование и статистический анализ данных на языке R

Лекция 7 (Основы статистического
анализа на языке R)



Петровский Михаил (ВМК МГУ), michael@cs.msu.su

Что есть кластер?

- Кластер: группа «похожих» объектов
 - «похожих» между собой в группе (внутриклассовое расстояние)
 - «не похожих» на объекты других групп
 - Определение неформальное, формализация зависит от метода
- Кластерный анализ
 - Разбиение множества объектов на группы (кластеры)
- Тип моделей:
 - «описательный» (descriptive) Data mining => одна из задач - наглядное представление кластеров
 - «прогнозный» (predictive) Data mining => разбиение на кластеры, а затем «классификация» новых объектов
- Тип обучения:
 - всегда «без учителя» (unsupervised) => тренировочный набор не размечен
- Этапы кластерного анализа:
 - Подготовка данных
 - Применение алгоритма
 - Визуализация и интерпретация результатов

Качество кластеризации

- Хороший метод кластеризации находит кластеры
 - с высоким «внутриклассовым» сходством объектов
 - и низким «межклассовым» сходством объектов
- Оценка качества кластеризации (нет понятия «точность»)
 - необходима, так как влияет на выбор параметров метода
 - определяется либо экспертом – субъективная величина
 - либо «перекрестной» проверкой целевой функции кластеризации
- Качество кластеризации зависит:
 - от метода кластеризации
 - от меры сходства (или расстояния)
 - пространства признаков

Требования к методу кластеризации

- Масштабируемость
- Поддержка различных типов атрибутов и структур данных
- Возможность находить кластеры сложной формы
- Отсутствие обязательных требований к наличию априорных знаний о выборке (например, о распределениях)
- Устойчивость к «шуму» и выбросам
- Возможность работы с высокой размерностью и с большой выборкой
- Возможность включать пользовательские ограничения и зависимости
- Интерпретируемость и наглядность (прототипы, границы, правила, функции принадлежности и т.п.)
- Интуитивность параметров кластеризации

Этапы кластерного анализа



Процедуры кластерного анализа



Этапы кластерного анализа

Подготовка данных

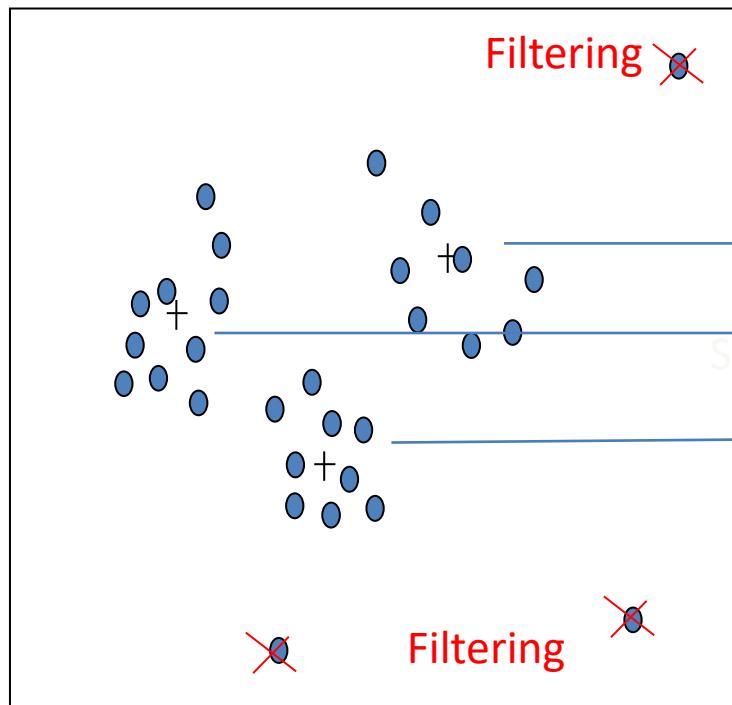


Подготовка данных

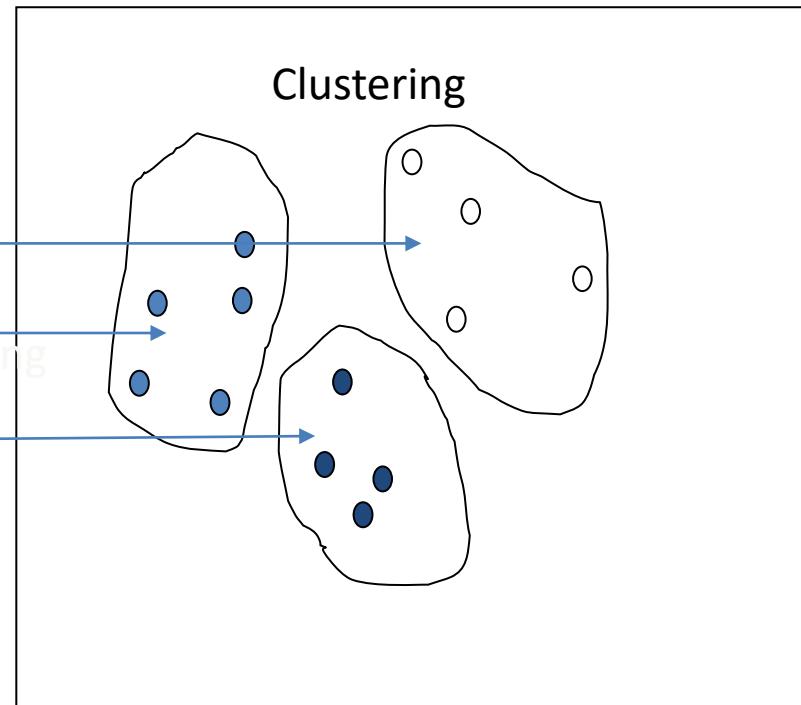
1. Отбор наблюдений
(Какие объекты делятся на группы?)
2. Отбор переменных
(Какие характеристики объектов важны?)
3. Визуальный анализ данных
(Какой формы кластеры и сколько их?)
4. Стандартизация переменных
(Сравнимы ли масштабы переменных?)
5. Трансформация переменных
(Переменные коррелируют? Кластеры не сферичны?)

Подготовка данных для кластеризации

«Сырые» данные



Кластерная/стратифицированная
случайная выборка



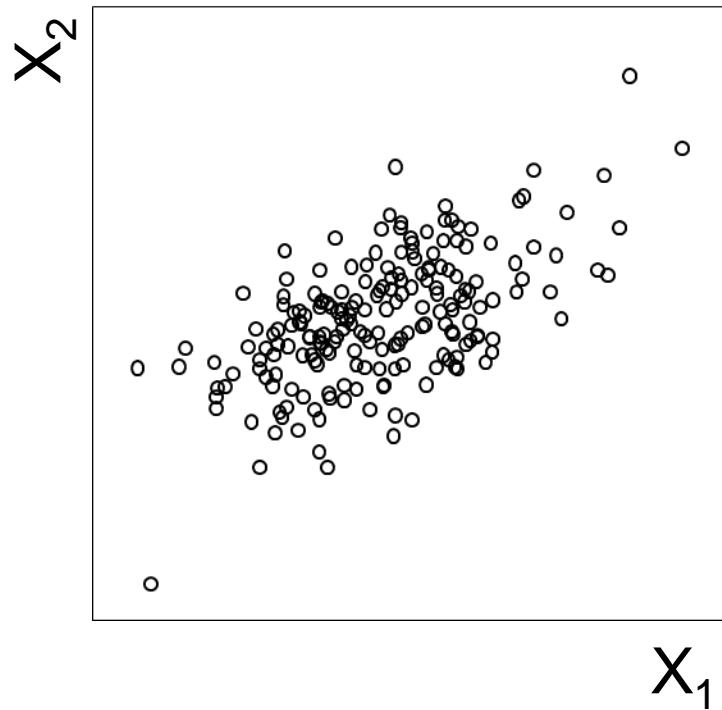
Подготовка и анализ данных

1. Отбор наблюдений
(Что я разбиваю на кластеры?)
2. Отбор переменных
(Какие характеристики объектов важны?)
3. Визуальный анализ данных
(Какой формы кластеры и сколько их?)
4. Стандартизация переменных
(Сравнимы ли масштабы переменных?)
5. Трансформация переменных
(Переменные коррелируют? Кластеры не сферичны?)

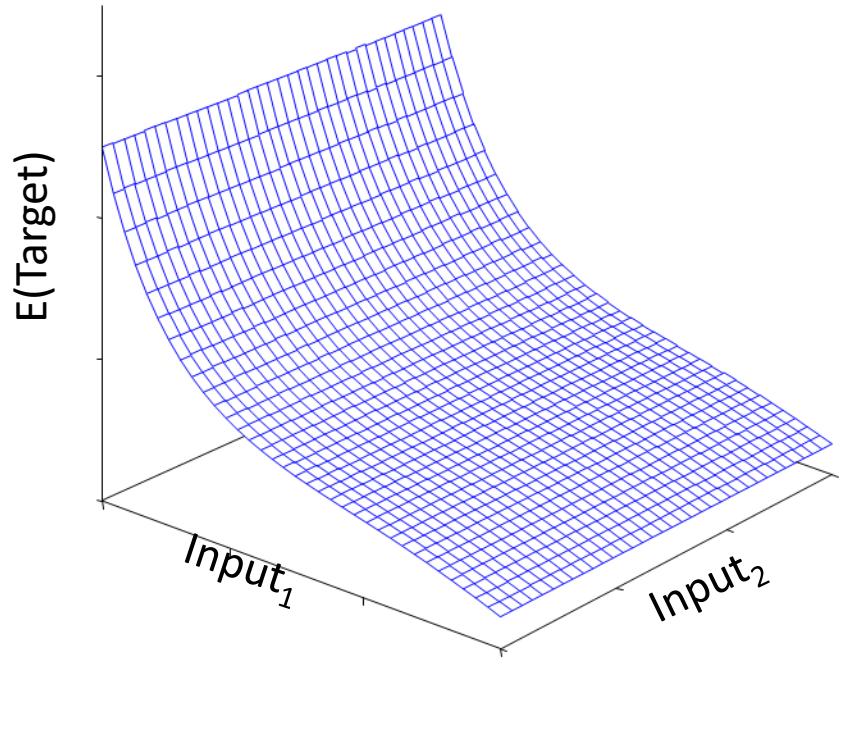
Снижение размерности

- Нет нужды включать в анализ все переменные

Избыточность



Незначимость

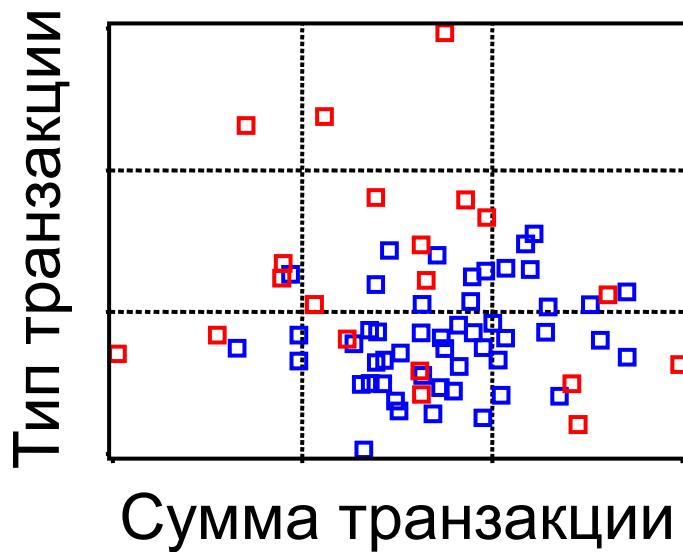


Отбор значимых переменных

- Регрессионные модели автоматически определяют значимость переменных на основе их влияния на целевую переменную
- Но в кластерном анализе целевой переменной НЕТ
- Поэтому все незначимые переменные должны быть удалены перед проведением кластеризации путем:
 - Анализа важности переменных на специально подготовленной выборке с целевой переменной
 - Подключения экспертных соображений

Секрет качественной кластеризации

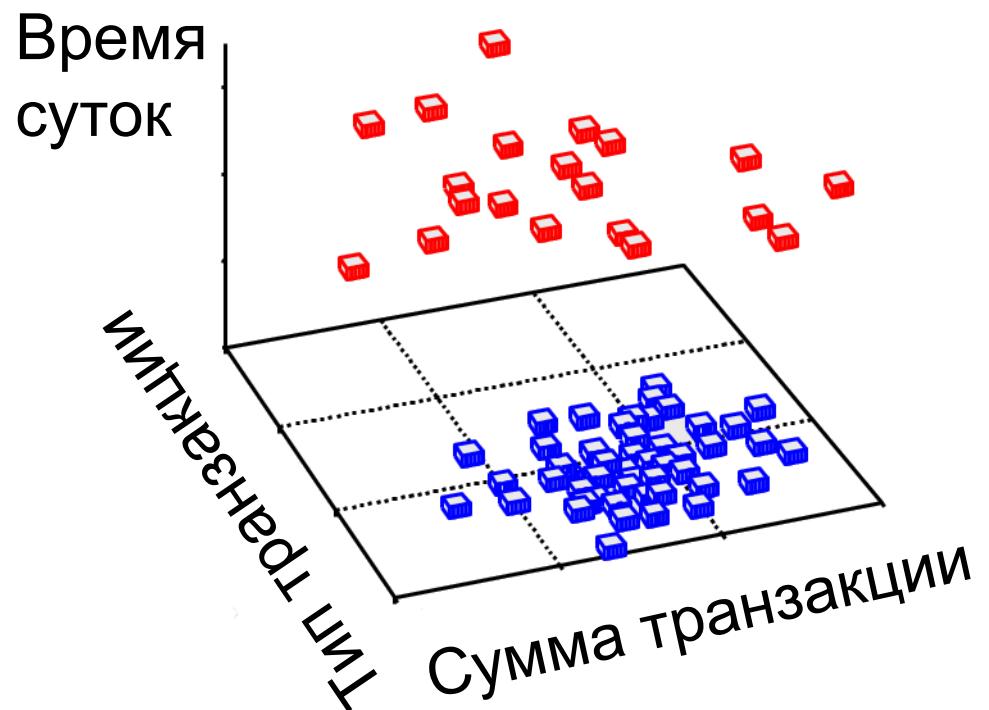
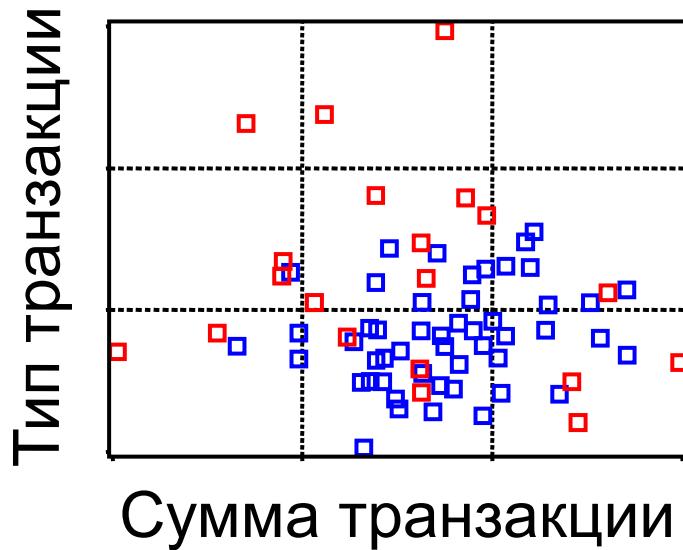
- Мошенник
- OK



Секрет качественной кластеризации

□ Мошенник

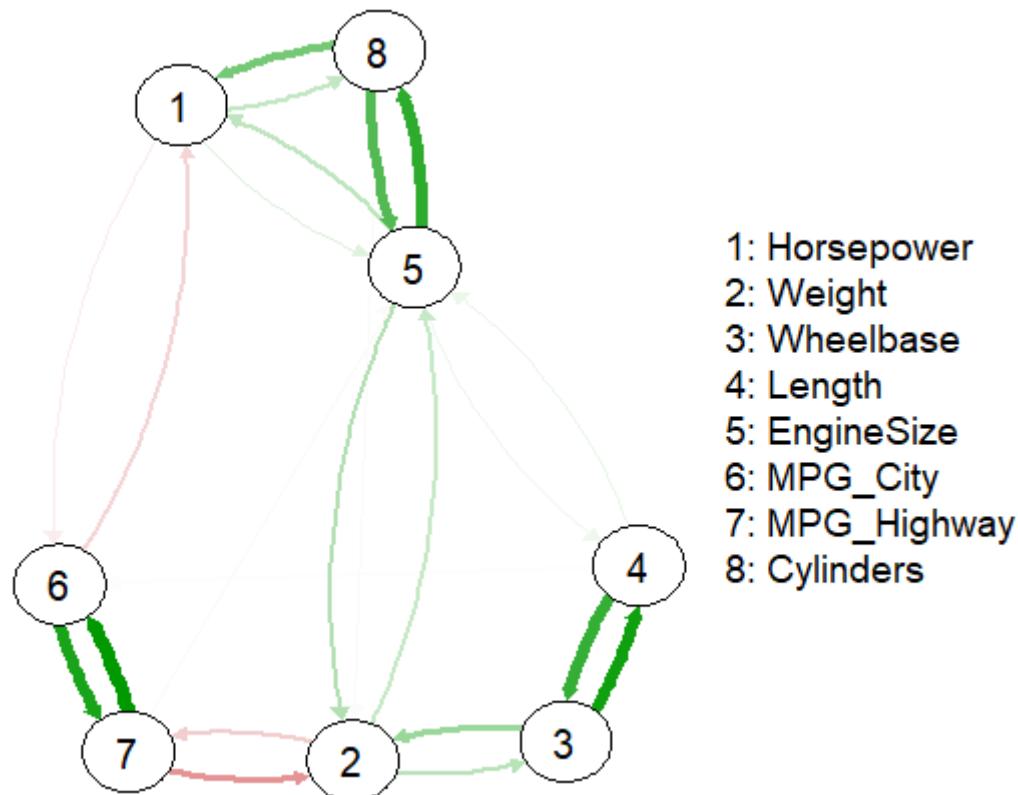
□ OK



Больше нескоррелированных
переменных = кластеры лучше!

Пример удаления избыточных переменных

```
> feats <- c("Horsepower", "Weight", "Wheelbase", "Length",
+           "EngineSize", "MPG_City", "MPG_Highway", "Cylinders")
> s<- var(scale(na.omit(cars[,feats])))
>
> a<-glasso(s, rho=0.1, approx = TRUE)
> qgraph(a$wi, nodeNames = feats)
```

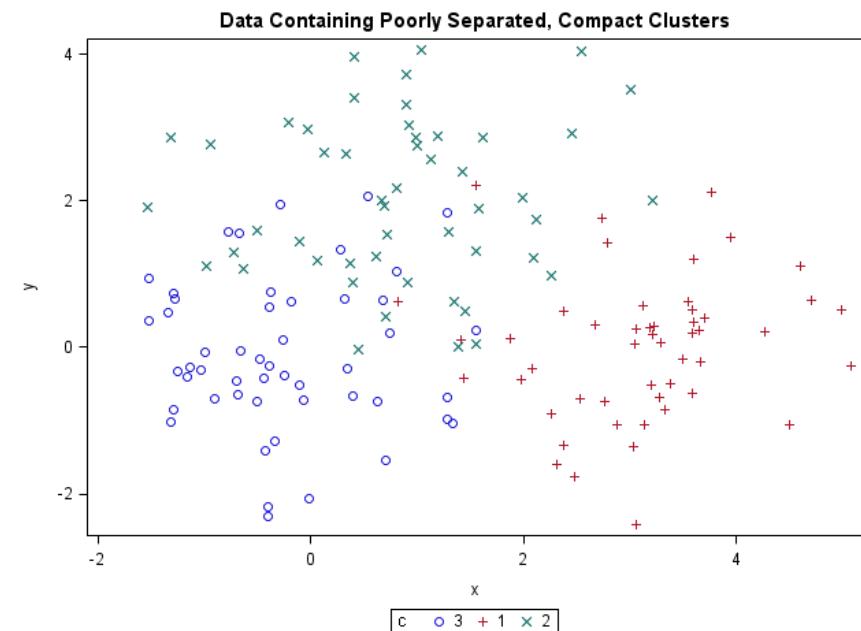
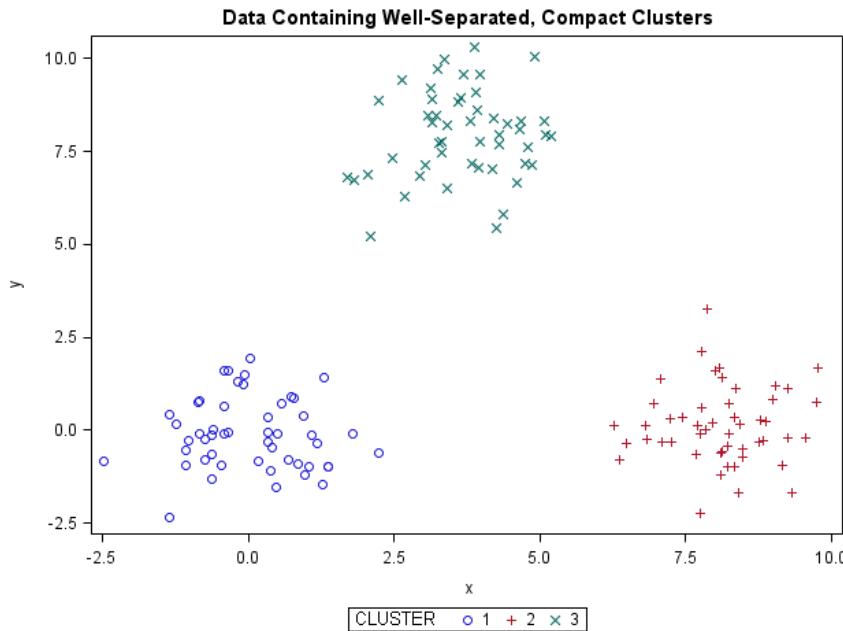


Подготовка и анализ данных

1. Отбор наблюдений
(Что я разбиваю на кластеры?)
2. Отбор переменных
(Какие характеристики объектов важны?)
3. Визуальный анализ данных
(Какой формы кластеры и сколько их?)
4. Стандартизация переменных
(Сравнимы ли масштабы переменных?)
5. Трансформация переменных
(Переменные коррелируют? Кластеры не сферичны?)

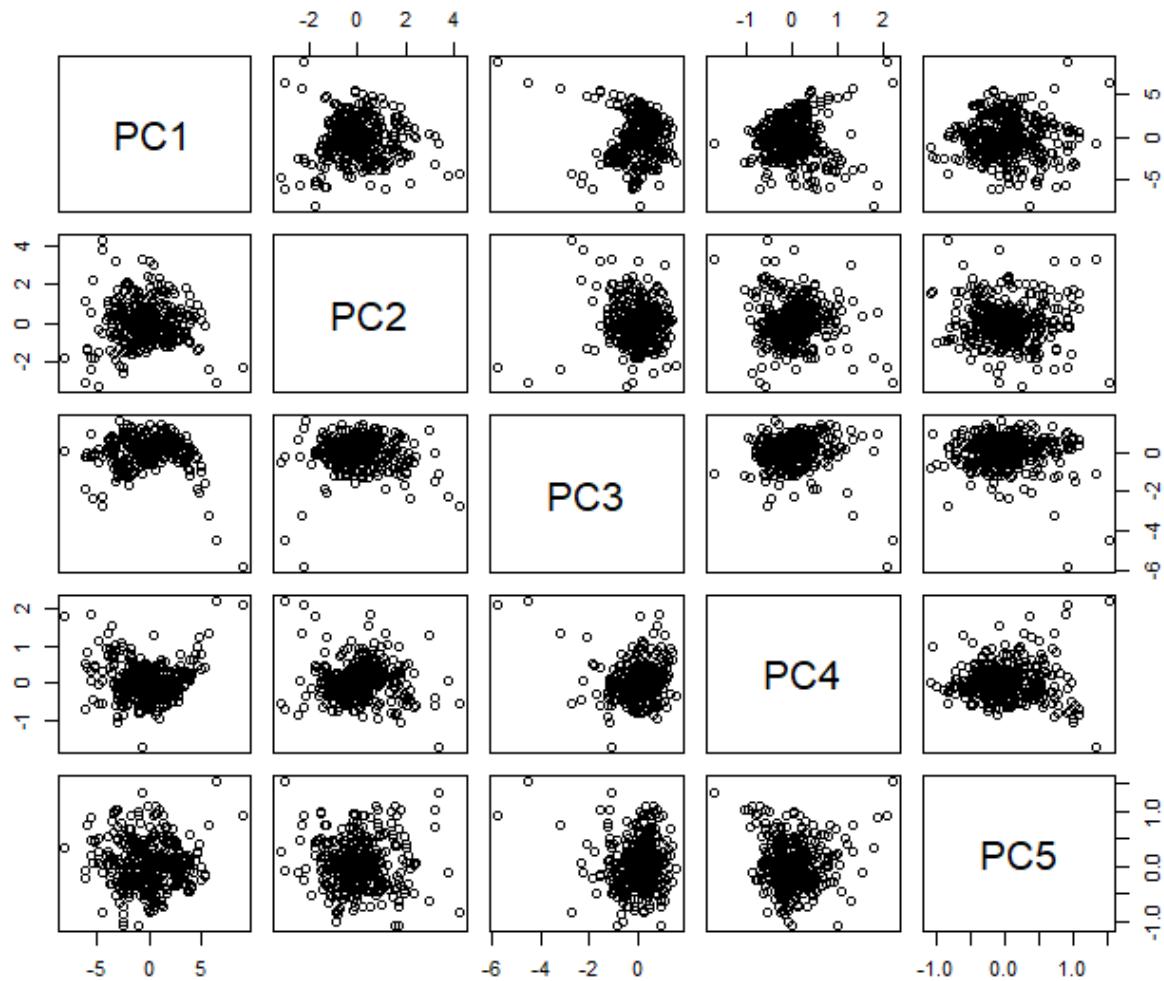
Визуальный анализ данных

- Визуализация помогает установить такие ключевые параметры задачи, как:
 - форму кластеров
 - дисперсию кластеров
 - примерное количество кластеров



Пример

```
> pca <- prcomp(~ EngineSize + Cylinders + Horsepower +
+                 MPG_City + MPG_Highway + Weight + Wheelbase + Length,
+                 data = cars, scale = TRUE)
> pairs(pca$x[,1:5])
```



Подготовка и анализ данных

1. Отбор наблюдений
(Что я разбиваю на кластеры?)

2. Отбор переменных
(Какие характеристики объектов важны?)

3. Визуальный анализ данных
(Какой формы кластеры и сколько их?)

4. Стандартизация переменных
(Сравнимы ли масштабы переменных?)

5. Трансформация переменных
(Переменные коррелируют? Кластеры не сферичны?)

Числовые значения – приведение к близким шкалам

- Нормализация на абсолютное отклонение более robustno (устойчиво к ошибкам), чем нормализация на стандартное отклонение:
 - Среднее абсолютное отклонение

где

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + \dots + |x_{nf} - m_f|)$$

– *z-score*

$$m_f = \frac{1}{n}(x_{1f} + x_{2f} + \dots + x_{nf})$$
$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

- Обычная нормализация:

$$y_{if} = \frac{x_{if} - m_f}{std_f}$$

$$std_f = \sqrt{\frac{1}{n-1}[(x_{1f} - m_f)^2 + (x_{2f} - m_f)^2 + \dots + (x_{nf} - m_f)^2]}$$

Стандартизация (scale)

scale(x, center = TRUE, scale = TRUE)

```
> cars_std<-scale(cars[8:15])
> cars_std
    EngineSize Cylinders Horsepower MPG_City MPG_Highway      Weight Wheelbase Length
[1,] 0.273563479 0.1235132 0.68370266 -0.58431090 -0.66945194 1.150284634 -0.25917397 0.18372002
[2,] -1.079500878 -1.1598192 -0.22113574 0.75202151 0.72398480 -1.053980188 -0.86072745 -1.00028961
[3,] -0.718683716 -1.1598192 -0.22113574 0.37021225 0.37562562 -0.458446596 -0.37948467 -0.23416573
[4,] 0.002950608 0.1235132 0.75330562 -0.01159701 0.20144602 -0.003891089 -0.01855258 -0.02522286
[5,] 0.273563479 0.1235132 0.12687903 -0.39340627 -0.49527235 0.397962331 0.82362228 0.74090103
[6,] 0.273563479 0.1235132 0.12687903 -0.39340627 -0.49527235 0.415090509 0.82362228 0.74090103
[7,] 0.002950608 0.1235132 1.03171743 -0.58431090 -0.49527235 -0.559898115 -0.98103815 -0.86099436
[8,] -1.259909459 -1.1598192 -0.63875346 0.37021225 0.72398480 -0.429460448 -0.49979536 -0.51275623
[426,] -0.267662264 0.1235132 0.72546443 -0.20250164 -0.14691316 0.098877982 0.22206881 0.25336765
[427,] -1.169705169 -1.1598192 -0.63875346 0.37021225 0.37562562 -0.996007891 -0.86072745 -0.44310861
[428,] -0.628479426 -0.5181530 -0.10977101 -0.01159701 0.02726643 0.322861856 0.10175811 -0.02522286
attr("scaled:center")
  EngineSize Cylinders Horsepower MPG_City MPG_Highway      Weight Wheelbase Length
  3.196729   5.807512 215.885514  20.060748   26.843458 3577.953271 108.154206 186.362150
attr("scaled:scale")
  EngineSize Cylinders Horsepower MPG_City MPG_Highway      Weight Wheelbase Length
  1.108595   1.558443 71.836032   5.238218   5.741201 758.983215  8.311813 14.357991

> cars_std<-scale(cars$Horsepower,center = (huber(cars$Horsepower)$mu), scale = (huber(cars$Horsepower)$s))
> cars_std
    [,1]
[1,] 0.801286070
[2,] -0.172978361
[3,] -0.172978361
[4,] 0.876229487
[5,] 0.201738728
[6,] 0.201738728
[427,] -0.622638867
[428,] -0.053068892
attr("scaled:center")
[1] 211.5406
attr("scaled:scale")
[1] 66.717
```

Методы стандартизации

```
> cars[c("MPG_City", "MPG_Highway", "Cylinders")]
# A tibble: 428 × 3
  MPG_City MPG_Highway Cylinders
    <dbl>      <dbl>     <dbl>
1     17         23       6
2     24         31       4
3     22         29       4
4     20         28       6
5     18         24       6
6     18         24       6
7     17         24       6
8     22         31       4
9     23         30       4
10    20         28       6
> scale(cars["MPG_City"])
          MPG_City
[1,] -0.58431090
[2,]  0.75202151
[3,]  0.37021225
[4,] -0.01159701
[5,] -0.39340627
[6,] -0.39340627
[7,] -0.58431090
[8,]  0.37021225
[9,]  0.56111688
[10,] -0.01159701
> cars[c("MPG_City", "MPG_Highway", "Cylinders")] %>%
+   mutate_all(~(scale(.) %>% as.vector))
# A tibble: 428 × 3
  MPG_City MPG_Highway Cylinders
    <dbl>      <dbl>     <dbl>
1 -0.584     -0.669     0.124
2  0.752      0.724     -1.16
3  0.370      0.376     -1.16
4 -0.0116     0.201     0.124
5 -0.393     -0.495     0.124
6 -0.393     -0.495     0.124
7 -0.584     -0.495     0.124
8  0.370      0.724     -1.16
9  0.561      0.550     -1.16
10 -0.0116    0.201     0.124
```

Подготовка и анализ данных

1. Отбор наблюдений
(Что я разбиваю на кластеры?)

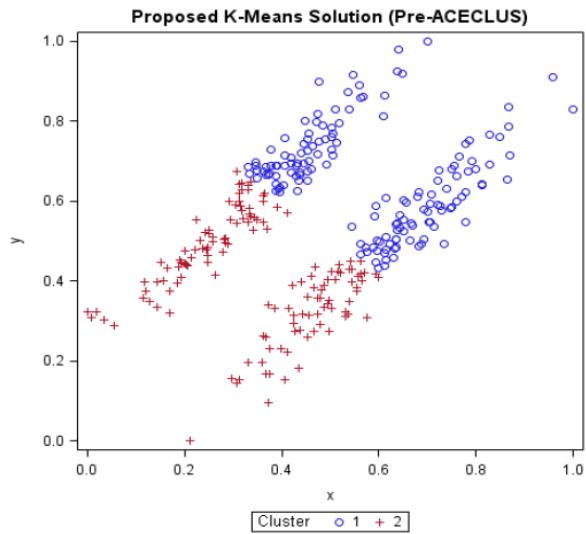
2. Отбор переменных
(Какие характеристики объектов важны?)

3. Визуальный анализ данных
(Какой формы кластеры и сколько их?)

4. Стандартизация переменных
(Сравнимы ли масштабы переменных?)

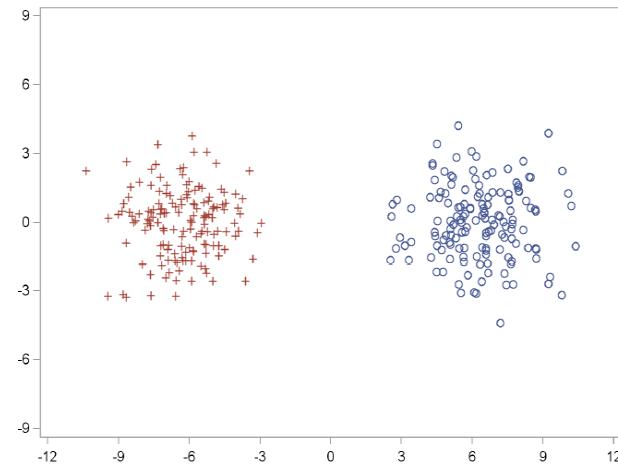
5. Трансформация переменных
(Переменные коррелируют? Кластеры не сферичны?)

Важность формы кластеров

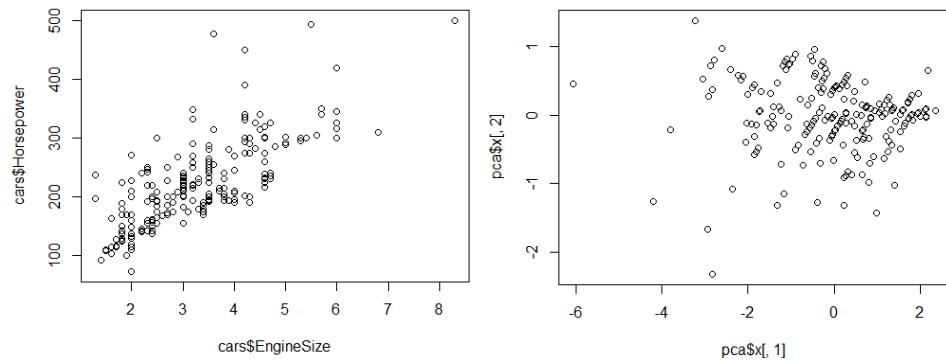


Все алгоритмы кластеризации
чувствительны к форме кластеров

```
> plot(cars$EngineSize, cars$Horsepower)
> pca <- prcomp(~ EngineSize + Horsepower,
+                 data = cars, scale = TRUE)
> plot(pca$x[,1], pca$x[,2])
```



Сферические группы объектов
оптимальны для обнаружения
большинством алгоритмов



Этапы кластерного анализа

Выбор и применение алгоритма



Что такое хорошее разбиение на группы?

- Поиск наилучшего разбиения множества объектов на кластеры всегда сводится к оптимизации т.н. **Natural Grouping Criterion**:
 - Максимизировать среднее межкластерное расстояние, или
 - Минимизировать среднее внутрикластерное расстояние между объектами.
- Большое межкластерное расстояние говорит о хорошей разделенности кластеров
- Малое внутрикластерное расстояние – признак однородности объектов внутри группы

Естественный критерий группировки (Natural Grouping Criterion)

Мера похожести объектов

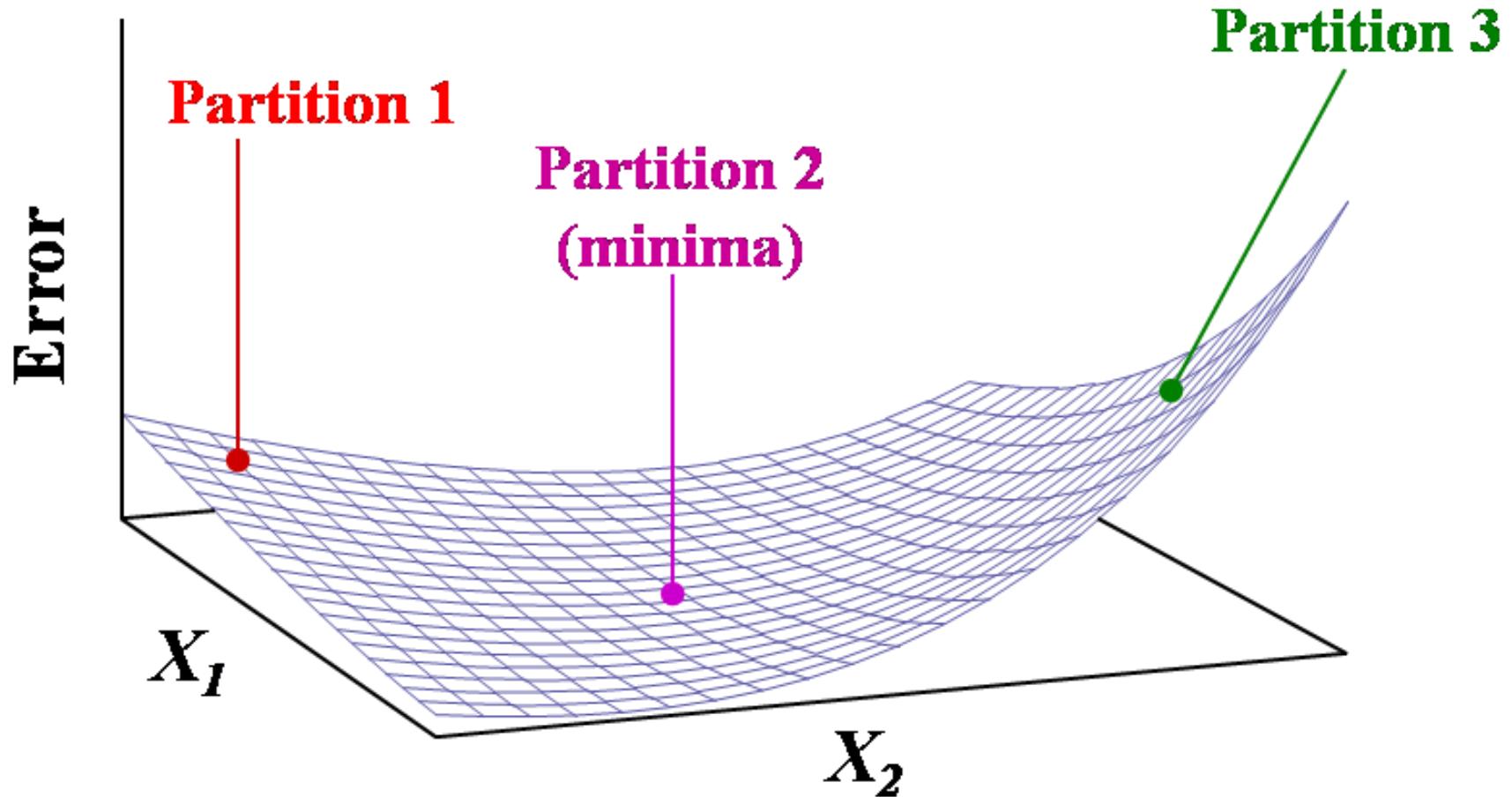
$$F = \frac{\sum_k \sum_{l,m} \Phi_1 \left(\begin{array}{c} \text{расстояние между} \\ \text{наблюдениями } x_l \text{ и } x_m \text{ в кластере } k \end{array} \right)}{\sum_{i,j} \Phi_2 (\text{расстояние между кластерами } i \text{ и } j)}$$



Мера близости кластеров

- Большое межкластерное расстояние → F уменьшается
- Малое внутрикластерное расстояние → F уменьшается

Оптимизация NGC



Исходные данные

- Матрица признаков:
 - Числовые
 - Бинарные
 - Номинальные (категориальные)
 - Упорядоченные шкалы
 - Нелинейные шкалы

$$\begin{bmatrix} x_{11} & \dots & x_{1f} & \dots & x_{1p} \\ \dots & \dots & \dots & \dots & \dots \\ x_{i1} & \dots & x_{if} & \dots & x_{ip} \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & \dots & x_{nf} & \dots & x_{np} \end{bmatrix}$$

- Матрица различия (или сходства):
 - «Естественные» расстояния предметной области
 - Экспертные оценки (противоречивы, нетранзитивны, недостоверны)

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix}$$

Меры сходства и различия для исходных данных с числовыми атрибутами

- Обычно строится на основе расстояния:

$$- d(i,j) \geq 0, d(i,i) = 0, d(i,j) = d(j,i), d(i,j) \leq d(i,k) + d(k,j)$$

- Наиболее популярно расстояние Минковского:

$$d(i,j) = \sqrt[q]{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \dots + |x_{ip} - x_{jp}|^q)}$$

где $i = (x_{i1}, x_{i2}, \dots, x_{ip})$ и $j = (x_{j1}, x_{j2}, \dots, x_{jp})$ - два объекта с p числовыми атрибутами, q - положительное целое число

- $q = 2$ - Евклидово (не фамилия, но имя) расстояние:

$$d(i,j) = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \dots + |x_{ip} - x_{jp}|^2)}$$

- $q = 1$, d – расстояние «Манхэтен»:

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ip} - x_{jp}|$$

Бинарные атрибуты

- Расстояние Хэмминга = сумма единиц после XOR $(M_{10} + M_{01})$
- Таблица «сопряженных признаков»
 - В ячейках – число совпадающих и несовпадающих значений из р бинарных атрибутов для объектов j и i

		Object j		<i>sum</i>
		1	0	
Object i	1	M_{11}	M_{10}	$M_{11} + M_{10}$
	0	M_{01}	M_{00}	$M_{01} + M_{00}$
<i>sum</i>		$M_{11} + M_{01}$	$M_{10} + M_{00}$	$M_{00} + M_{01} + M_{10} + M_{11}$

- На основе коэффициента совпадения
 - для симметричных атрибутов (значения равнозначны)
 - На основе коэффициента Jaccard
 - для асимметричных атрибутов (единица важнее)
- $$d(i, j) = \frac{M_{10} + M_{01}}{M_{11} + M_{01} + M_{10} + M_{00}}$$
- $$d(i, j) = \frac{M_{10} + M_{01}}{M_{11} + M_{01} + M_{10}}$$

Пример

Имя	Пол	Жар	Кашель	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

- пол - симметричный атрибут
- остальные ассиметричные
- пусть Y и P соответствует 1, а N соответствует 0

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Категориальные атрибуты и шкалы

- Категориальные атрибуты:
 - много значений, например, цвета: red, yellow, blue, green
- Подход 1: простое совпадение
 - M - число совпадений, p - число переменных (аналог нормированного расстояния Хэмминга)
- Подход 2: кодирование бинарными векторами
 - Для каждого значения категориального атрибута создается отдельная бинарная переменная: один категориальный атрибут с M возможными значениями => бинарный вектор длины M
- Категориальные упорядоченные шкалы:
 - Могут быть и дискретными и непрерывными
 - Порядок важен, «разница» - нет = ранги
 - сводятся к числовым: заменить x_{if} на его ранг, отобразить на $[0, 1]$ с нормировкой:
$$r_{if} \in \{1, \dots, M_f\}$$
 - затем использовать стандартные расстояния

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

Расчет расстояний

```
dist(x, method = "euclidean", diag = FALSE, upper = FALSE, p = 2)
```

```
> std_cars <- cars[c("EngineSize", "MPG_City", "Wheelbase")] %>%
+     mutate_all(~(scale(.) %>% as.vector))
> rownames(std_cars) <- paste(rownames(std_cars), cars$Model)

> d_manh <- dist(std_cars, method = "manhattan")
> as.matrix(d_manh)[1:5, 1:5]
      1 MDX 2 RSX Type S 2dr 3 TSX 4dr 4 TL 4dr 5 3.5 RL 4dr
1 MDX          0.000000          3.290950  2.067081 1.083948   1.273701
2 RSX Type S 2dr 3.290950          0.000000  1.223869 2.688245   4.182842
3 TSX 4dr      2.067081          1.223869  0.000000 1.464376   2.958973
4 TL 4dr       1.083948          2.688245  1.464376 0.000000   1.494597
5 3.5 RL 4dr   1.273701          4.182842  2.958973 1.494597   0.000000

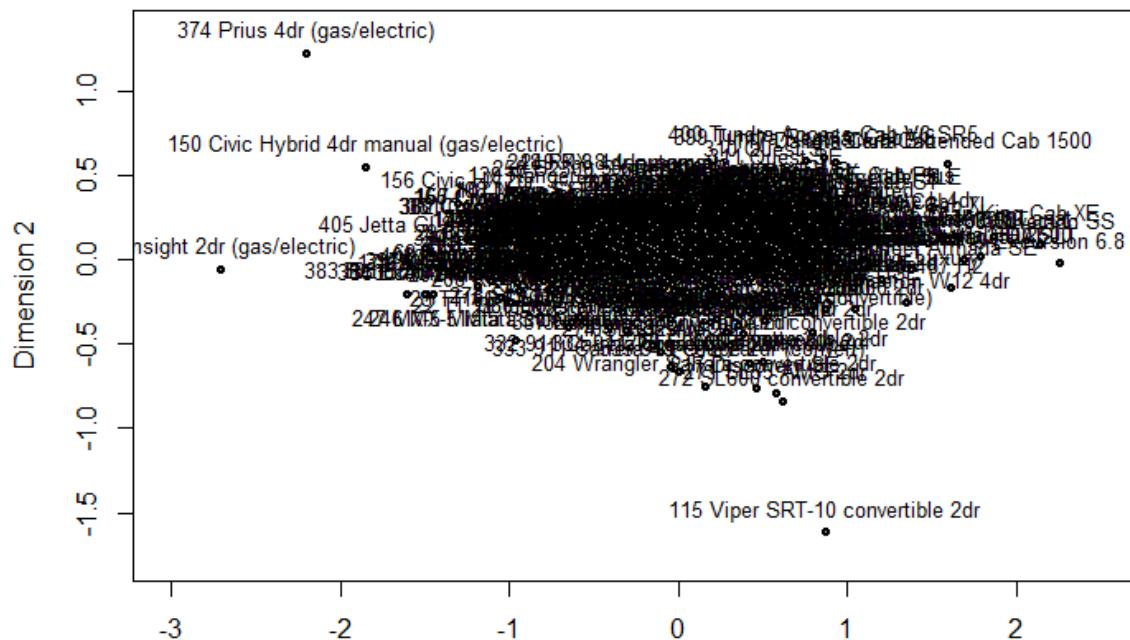
> cosineDist <- function(x) {
+   x%*%t(x)/(sqrt(rowSums(x^2) %*% t(rowSums(x^2))))
+ }
> d_cos <- cosineDist(as.matrix(std_cars))
> d_cos[1:5, 1:5]
      1 MDX 2 RSX Type S 2dr 3 TSX 4dr 4 TL 4dr 5 3.5 RL 4dr
1 MDX          1.0000000          -0.4680660 -0.50660523  0.80728647   0.1377274
2 RSX Type S 2dr -0.4680660          1.0000000  0.98348098  0.11704051  -0.8678345
3 TSX 4dr       -0.5066052          0.9834810  1.00000000  0.03177635  -0.7694697
4 TL 4dr        0.8072865          0.1170405  0.03177635  1.00000000  -0.4711240
5 3.5 RL 4dr    0.1377274          -0.8678345 -0.76946973 -0.47112404   1.0000000
```

Многомерные проекции с учетом расстояния

В общем случае ищутся координаты проекций объектов x_i в новом пространстве маленькой размерности, сохраняющие попарные расстояния d_{ij} как решение задачи оптимизации (для уменьшения случайности может использоваться бутстреппинг):

$$\operatorname{argmin}_{x_1, \dots, x_N} \sum_{i < j} (d_{ij} - \|x_i - x_j\|)^2$$

```
> d <- dist(std_cars, method = "manhattan")
> fit <- mds(d)
>
> resboot <- bootmds(fit, t(std_cars), nrep = 2)
> plot(resboot)
```



Типы алгоритмов

По оптимизируемой функции

- Оптимизирующие NGC
 - Группировка (разбиение) (*например K-means*)
 - Иерархические
- Оптимизирующие NGC в условиях ограничений
 - Параметрическое семейство алгоритмов (*Expectation-Maximization*)
 - Непараметрическое семейство алгоритмов (*Density / Kernel-based*)

Кластеризация на основе строгой группировки (partitioning)

- Основная задача:
 - Найти такое разбиение C исходного множества X из N объектов на K непересекающихся подмножеств C_k , покрывающих X , чтобы внутриклассовое расстояние было минимальным:

$$\min_{C_i \cap C_j = \emptyset, \cup C_i = X} \sum_{i=1}^K \sum_{x \in C_i} \sum_{x' \in C_i} d(x, x')$$

- Точное решение – перебор с отсечением
 - метод «ветвей и границ», но число комбинаций неприемлемо даже для 100 объектов:

$$S(N, K) = \frac{1}{K!} \sum_{i=1}^K (-1)^{K-i} \binom{K}{i} K^N$$

- Эвристические методы:
 - K-means (прототип кластера – мат. ожидание m), K-medoids (прототип кластера – средний элемент)

$$\min_{C_i \cap C_j = \emptyset, \cup C_i = X} \sum_{i=1}^K \sum_{x \in C_i} d(m_i, x)$$

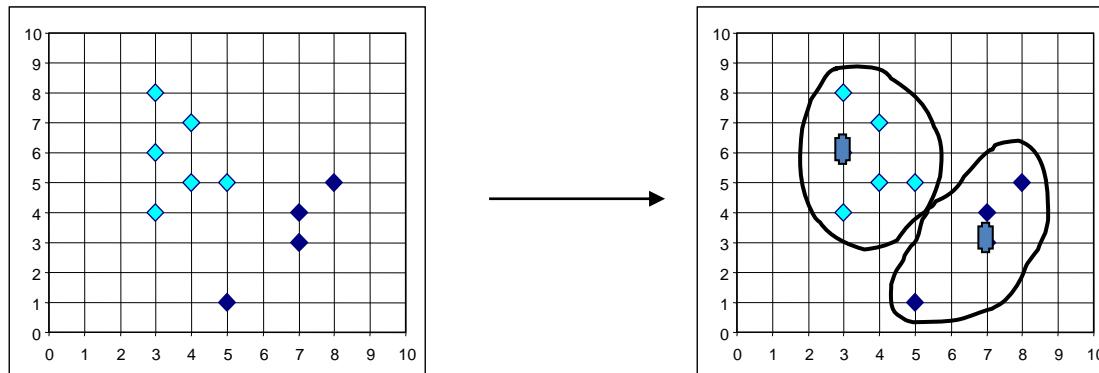
- ищется локальный минимум

K-Medoids

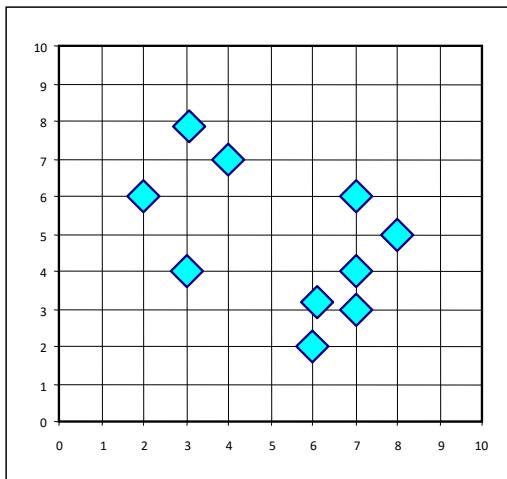
- K-Medoids:
 - Идея: вместо мат. ожидания кластера ищется представительный («наиболее центральный») объект – medoid
 - Процесс: случайная инициализация, переход к новому medoid, если это улучшает целевую функцию:

$$\min_{C_i \cap C_j = \emptyset, \cup C_i = X} \sum_{i=1}^K \sum_{x, m_i \in C_i} d(m_i, x)$$

- НО: не масштабируется и вычислительно неэффективный:
 - $O(K(N-K)^2)$ для каждой итерации



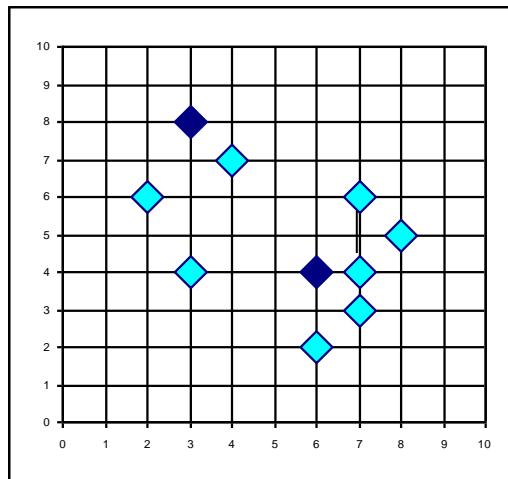
Алгоритм K-Medoids



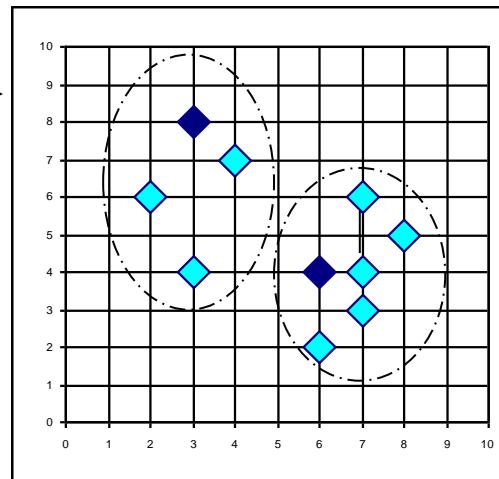
K=2

Пока есть
изменения

Случайн. K
medoids



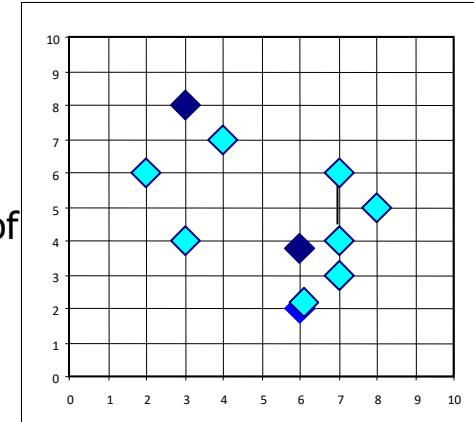
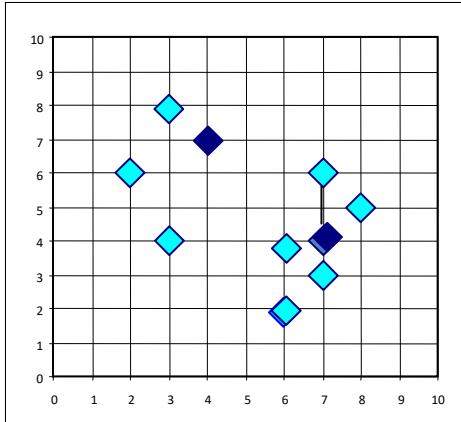
Каждый
объект к
ближ.
medoid



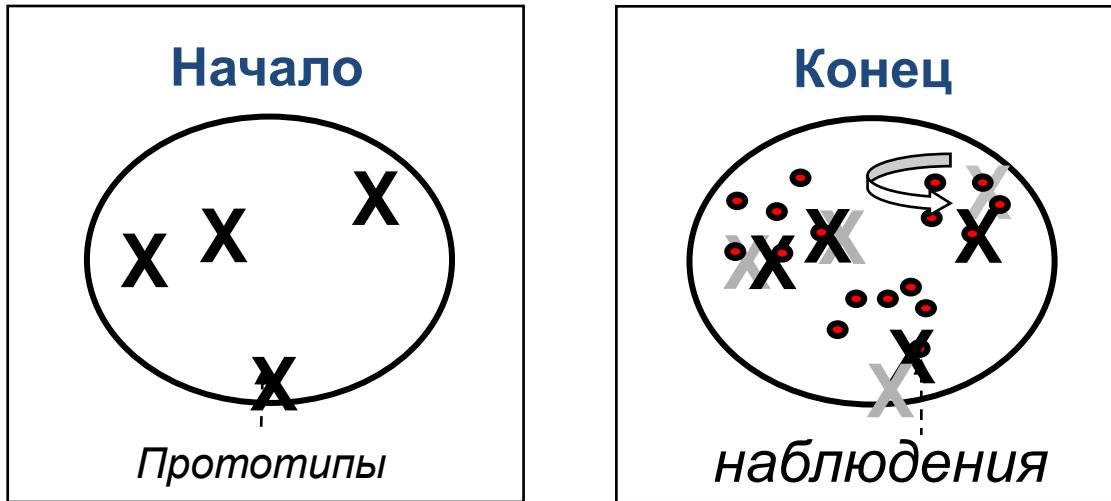
↑ Если качество кластеризац. улучш.,
то переход к нов. medoid

Перебор всех кандидатов
на замену medoid

Расчет
стоимости
перехода
(total cost of
swapping)

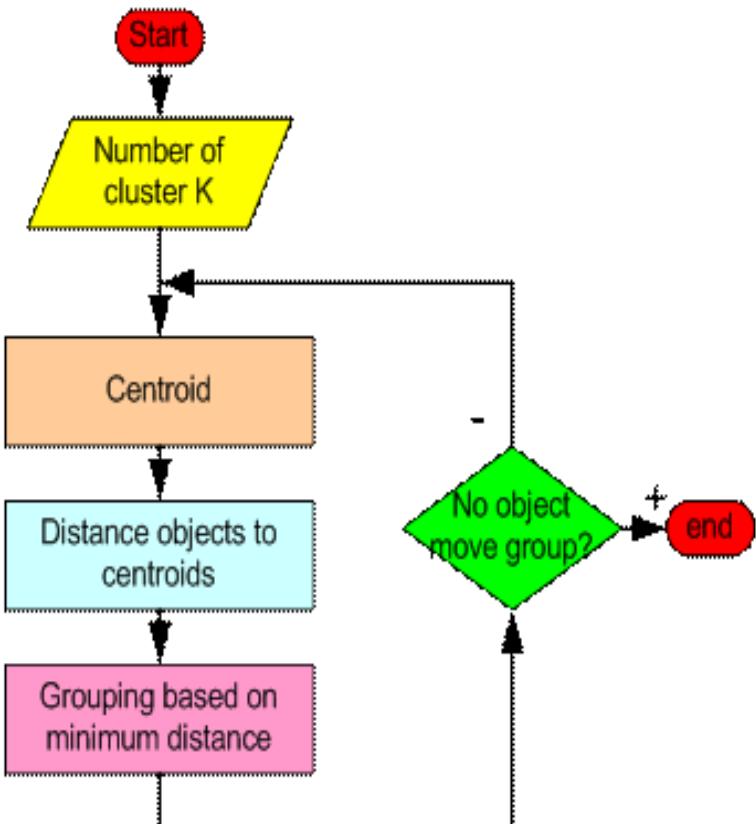


Основные проблемы алгоритмов группировки



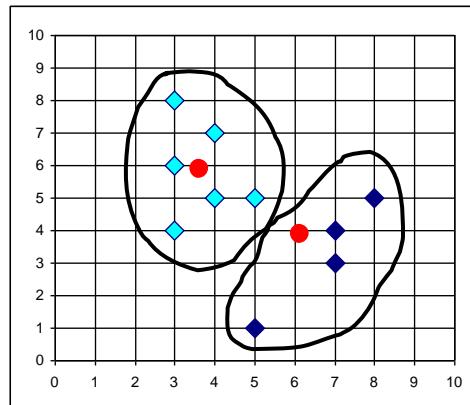
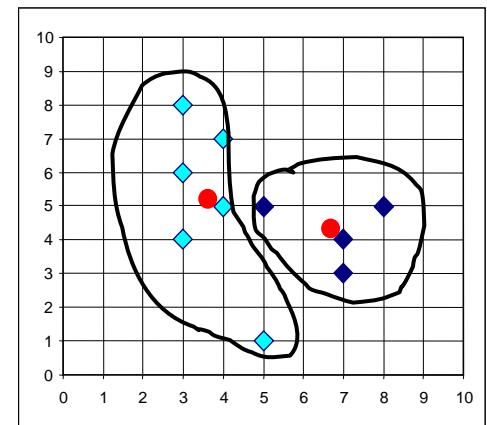
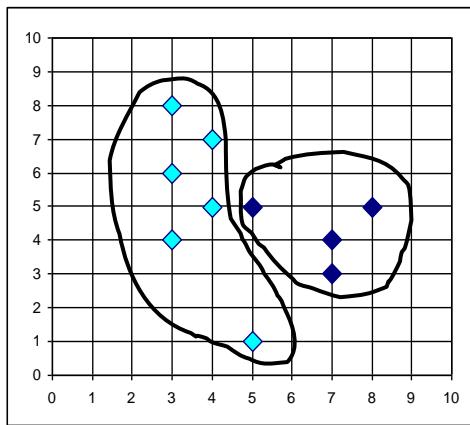
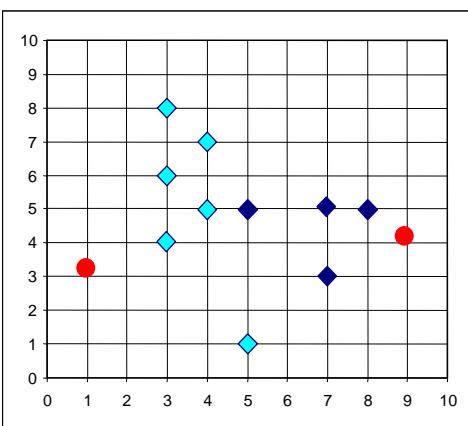
- Угадать число кластеров и хорошие начальные приближения
- Сферические кластеры только
- Влияют начальные приближения, выбросы, порядок обхода выборки
- Не находит глобально оптимального решения!!!!

Метод K-Means

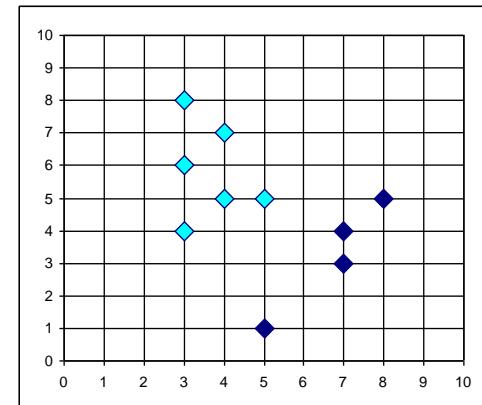
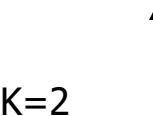


- Шаг 0. Инициализация:
 - произвольное разбиение на заданное число кластеров К (где значение К выбирается по ССС на основе иерархической кластеризации)
- Шаг 1. Поиск центров:
 - Для всех К кластеров $m_i = \sum_{x \in C_i} x / \|C_i\|$
- Шаг 2. Расчет расстояний до центров:
 - Для всех N объектов и К кластеров $d(m_i, x) = \sum_{x \in C_i} x / \|C_i\|$
- Шаг 3. Выбор ближайшего кластера:
$$x \in C_i \Leftrightarrow i = \min_j d(m_j, x)$$
- Если были перестановки, то Шаг 1.

Пример



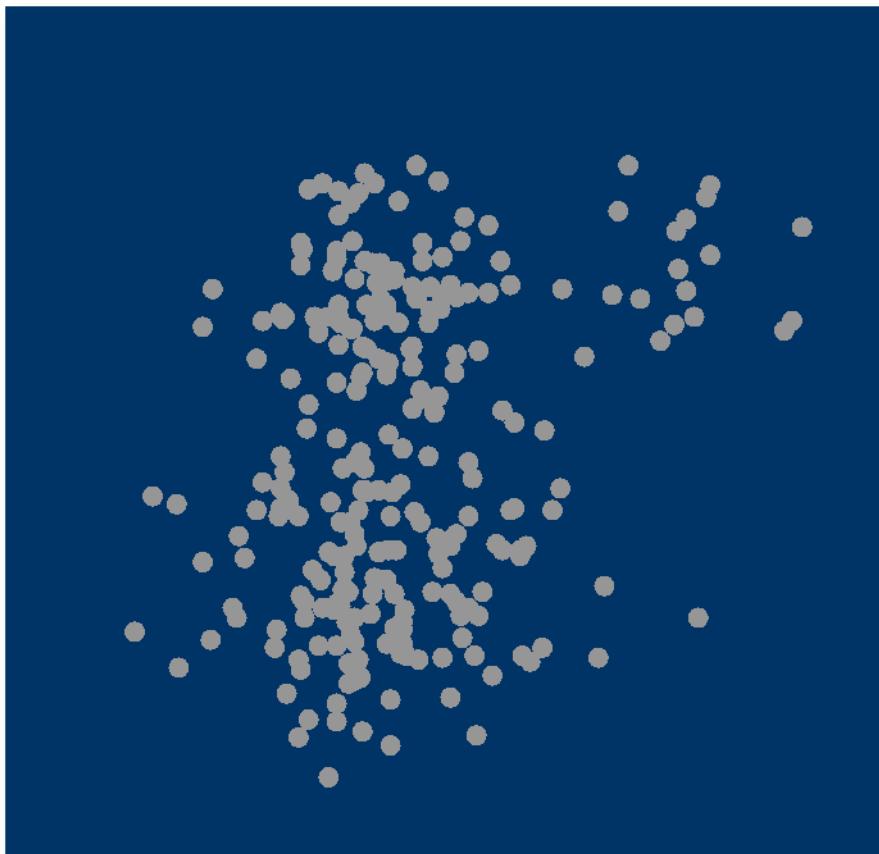
$K=2$



Особенности *K-Means*

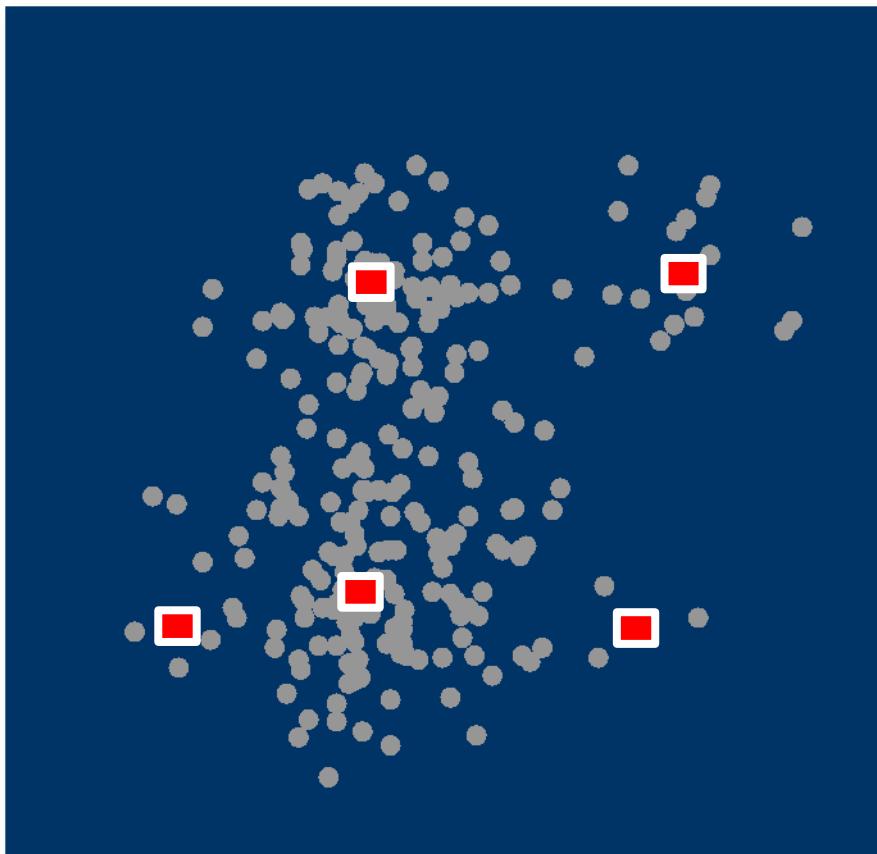
- Достаточно быстрый
- Локальный экстремум:
 - Глобальный можно искать «разумным» перебором: имитация отжига, генетические алгоритмы и т.д.
 - На основе нескольких инициализаций
- Недостатки
 - Числовые данные (иначе как найти центр?) – моды, центр кластера – число (для числовых атрибутов)
 - Необходимо задавать К заранее (есть методы «отбора» К)
 - Чувствительность к шуму и выбросам, кластеры сферической формы

k-Means алгоритм



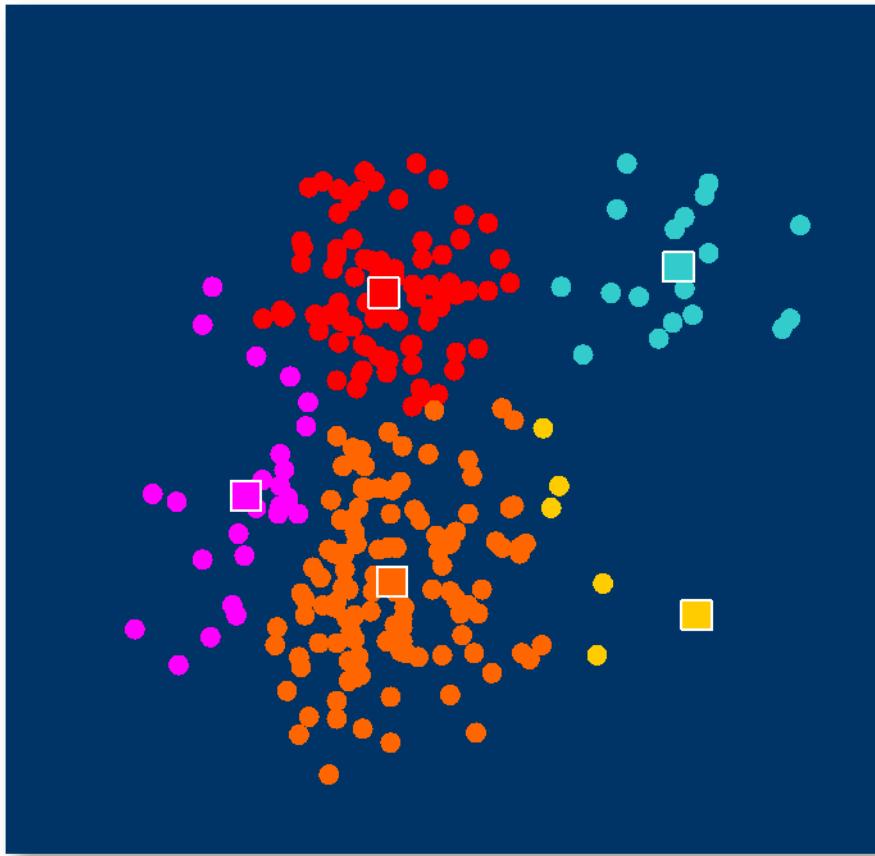
1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
4. Пересчитать центры.
5. Re-assign cases.
6. Повторять шаги 4 и 5 до сходимости

k-Means алгоритм



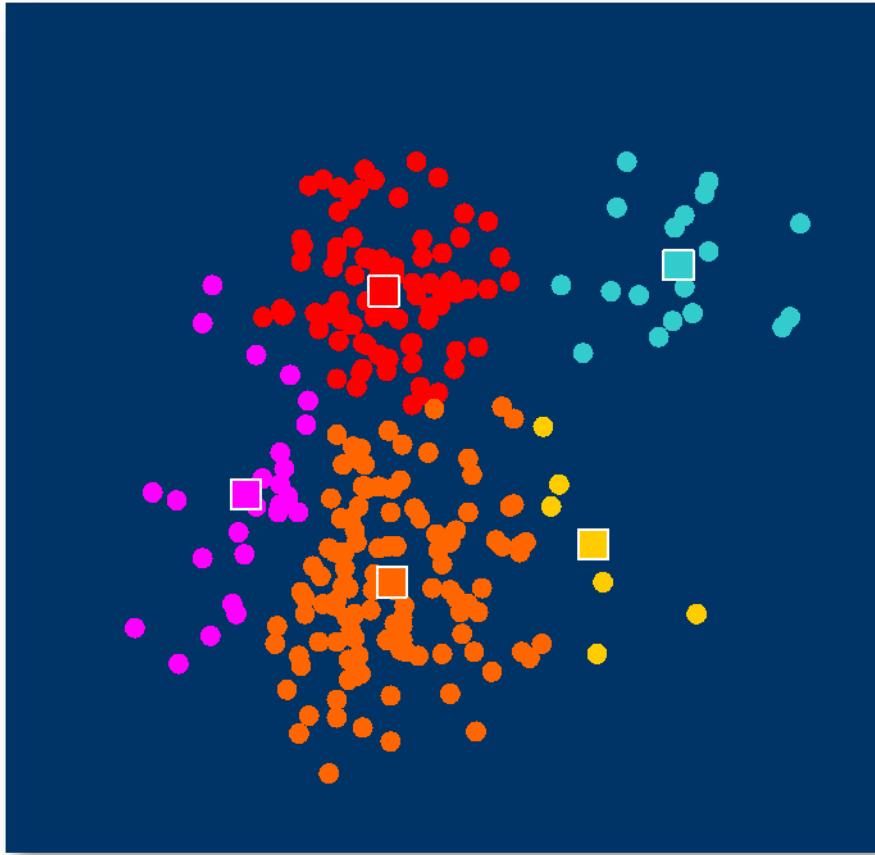
1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
4. Пересчитать центры.
5. Re-assign cases.
6. Повторять шаги 4 и 5 до сходимости

k-Means алгоритм



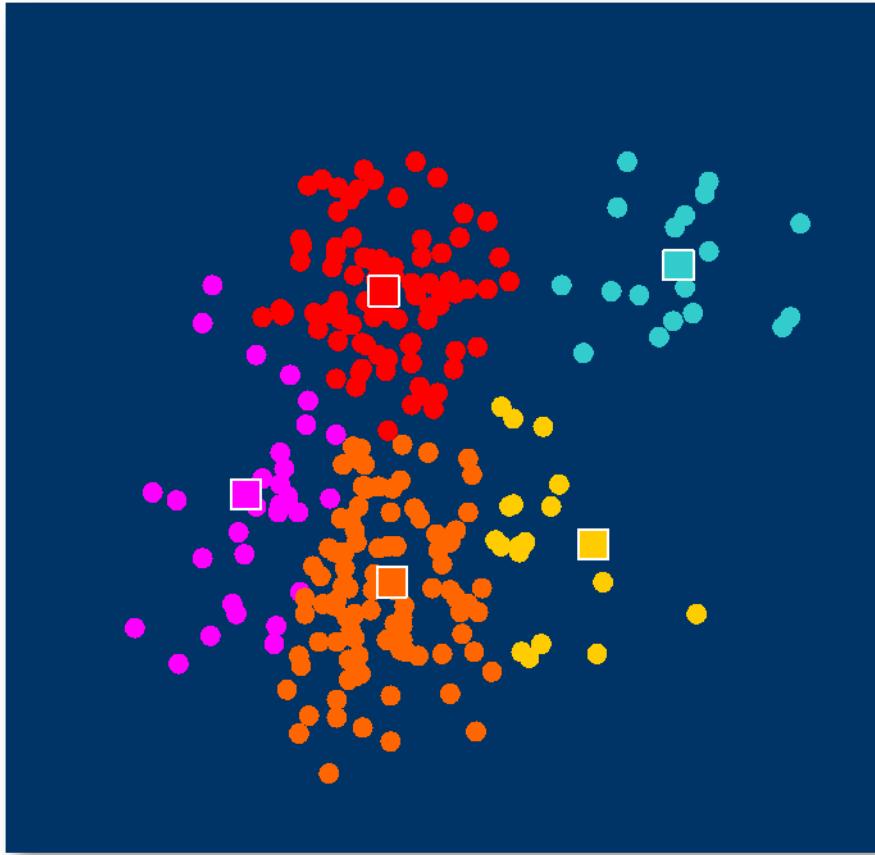
1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. **Отнести наблюдение к ближайшему кластеру.**
4. Пересчитать центры.
5. Пересчитать принадлежность наблюдений.
6. Повторять шаги 4 и 5 до сходимости

k-Means алгоритм



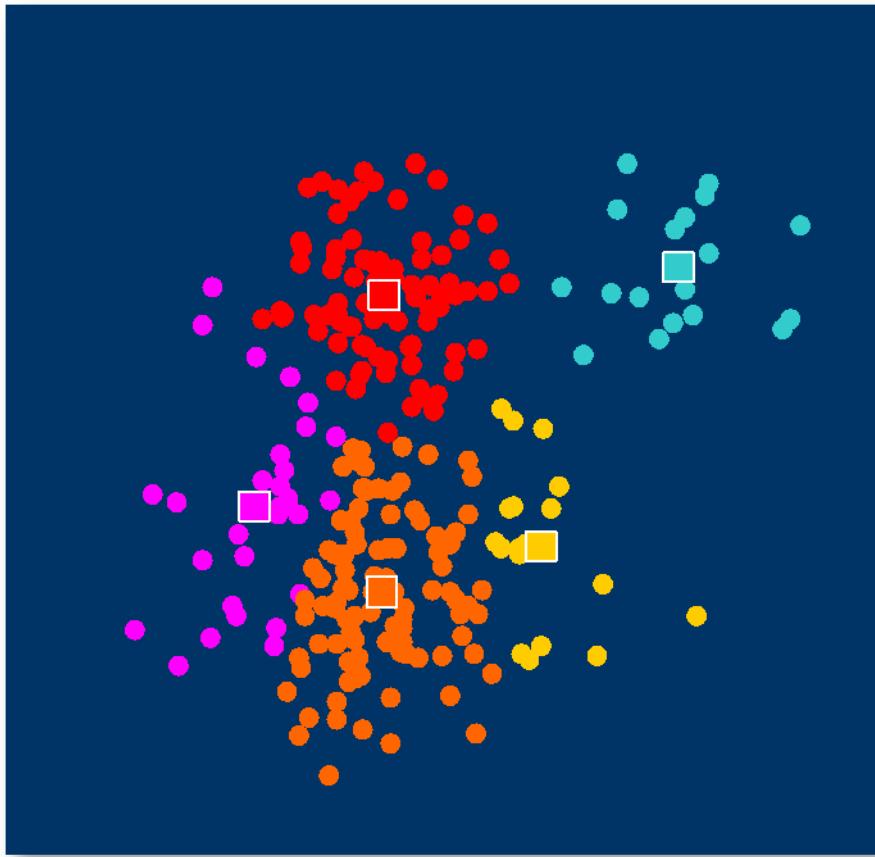
1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
- 4. Пересчитать центры.**
5. Пересчитать принадлежность наблюдений.
6. Повторять шаги 4 и 5 до сходимости

k-Means алгоритм



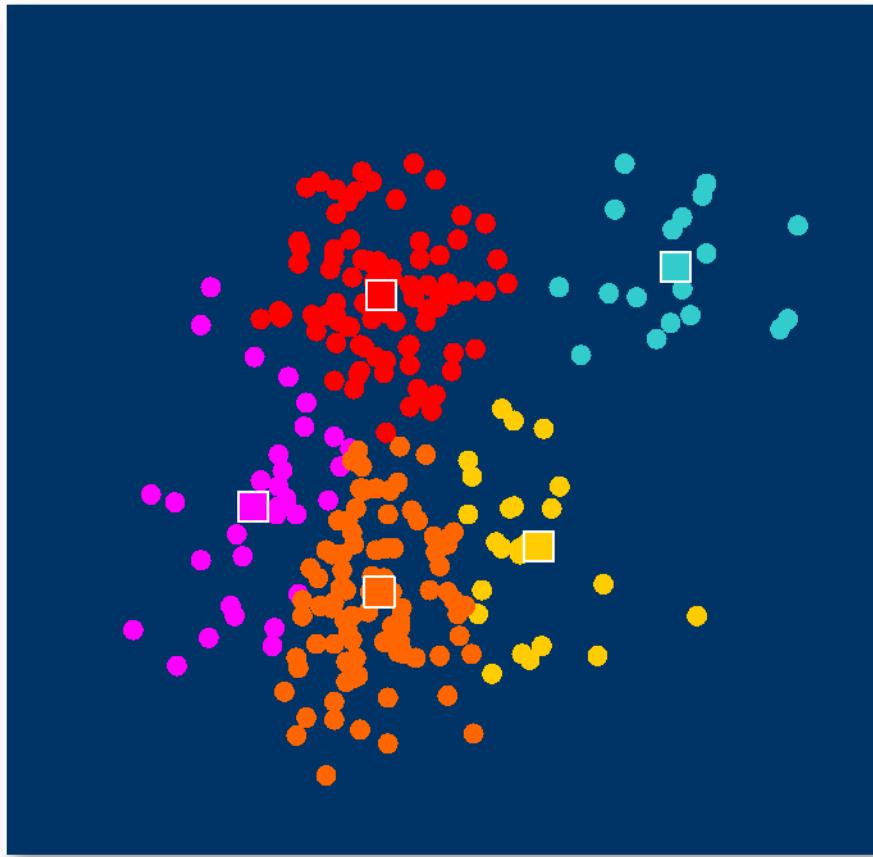
1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
4. Пересчитать центры.
5. Пересчитать принадлежность наблюдений.
6. Повторять шаги 4 и 5 до сходимости

k-Means алгоритм



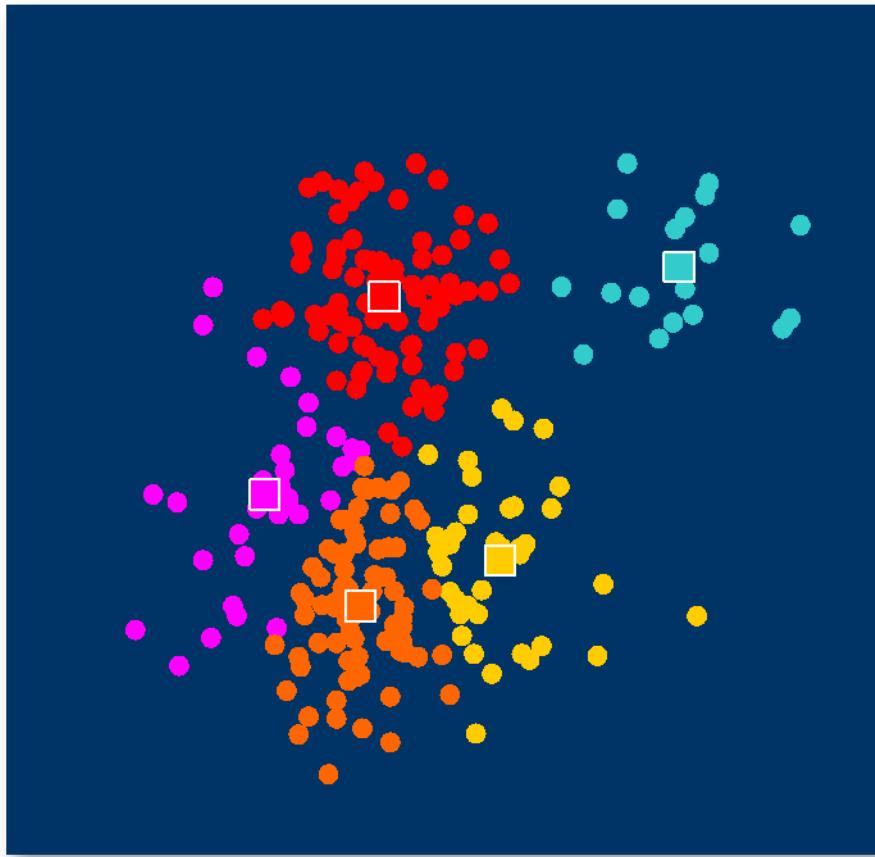
1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
- 4. Пересчитать центры.**
5. Пересчитать принадлежность наблюдений.
- 6. Повторять шаги 4 и 5 до сходимости**

k-Means алгоритм



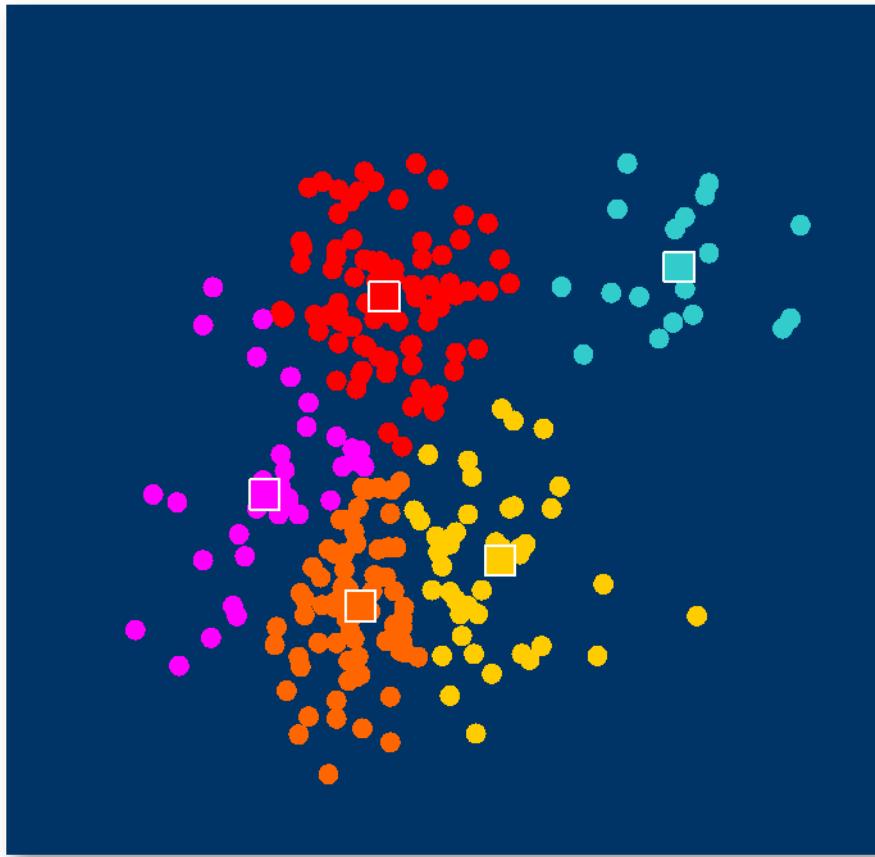
1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
4. Пересчитать центры.
5. Пересчитать принадлежность наблюдений.
6. Повторять шаги 4 и 5 до сходимости

k-Means алгоритм



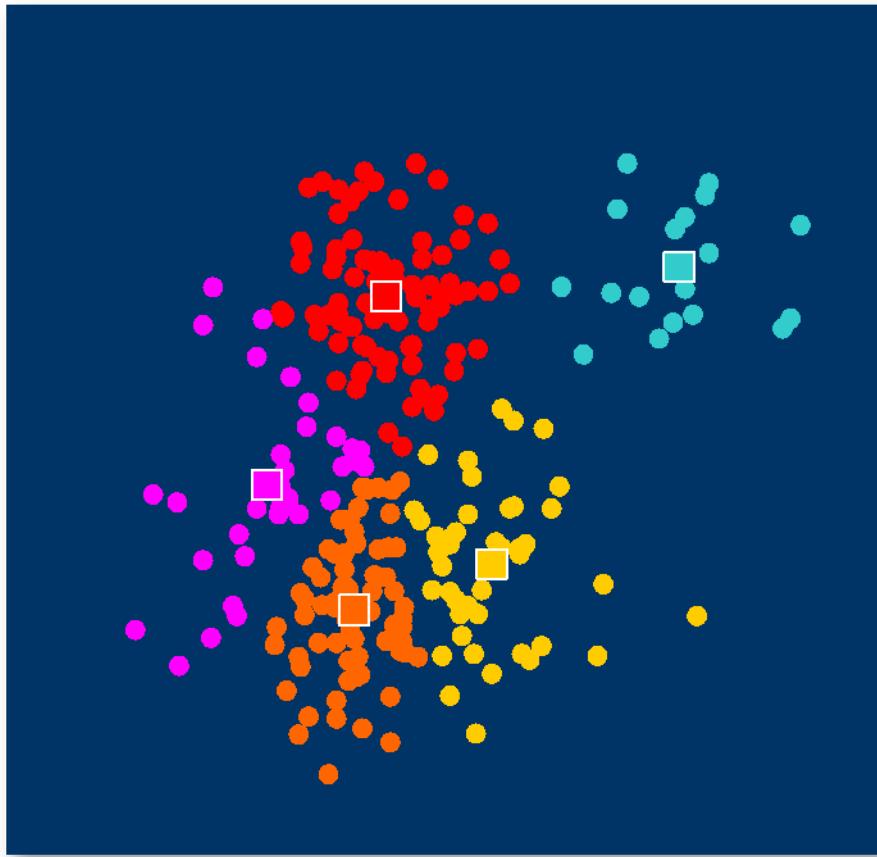
1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
- 4. Пересчитать центры.**
5. Пересчитать принадлежность наблюдений.
- 6. Повторять шаги 4 и 5 до сходимости**

k-Means алгоритм



1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
4. Пересчитать центры.
5. Пересчитать принадлежность наблюдений.
6. Повторять шаги 4 и 5 до сходимости.

k-Means алгоритм



1. Отобрать переменные.
2. Выбрать k центров кластеров.
3. Отнести наблюдение к ближайшему кластеру.
4. Пересчитать центры.
5. Пересчитать принадлежность наблюдений.
6. Повторять шаги 4 и 5 до сходимости

Процедура KMeans

```
kmeans(x, centers, iter.max = 10, nstart = 1, algorithm = c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"), trace=FALSE)
```

```
> k<-kmeans(na.omit(cars_std),center=5)
> k
K-means clustering with 5 clusters of sizes 102, 106, 98, 87, 33

Cluster means:
   EngineSize Cylinders Horsepower MPG_City MPG_Highway    Weight Wheelbase     Length
1  0.20723679  0.0794773 -0.02692985 -0.3578456 -0.2903552  0.3535014  0.6266430  0.6726190
2 -0.09831647  0.1477270  0.16732754 -0.2115066 -0.1288379 -0.1640980 -0.4668802 -0.4746472
3 -0.93867173 -1.1598192 -0.81531769  0.5942330  0.6120122 -0.7817578 -0.6495699 -0.5532656
4  1.48769249  1.4806004  1.27668783 -0.8783479 -0.9617533  1.2415592  0.9812708  0.9378357
5 -1.35558068 -1.1792637 -1.40016760  2.3602484  2.0540835 -1.4742793 -1.0794742 -1.3316435

Clustering vector:
 [1] 1 3 3 2 1 1 2 3 3 2 2 2 1 1 2 2 1 4 4 4 4 3 3 2 1 4 2 4 2 2 2 2 2 2 1 2 1 4 4 4 4 2 2 2 2 4 1 1 1 1 1 1 1 4 4 1 4 4 4 2 4 4
[79] 2 1 1 1 4 4 4 1 1 4 4 1 3 3 3 2 1 1 1 3 3 1 2 1 1 2 1 4 5 5 1 3 3 1 1 1 4 1 1 1 4 4 1 2 3 5 3 3 3 1 1 4 4 4 1 2 2 2 4 4 3 1 4
[157] 5 3 3 5 3 2 2 1 1 3 4 2 5 5 5 3 3 3 2 2 1 1 2 1 2 1 1 4 4 1 4 4 2 2 2 1 4 4 4 4 4 2 2 4 4 2 3 2 2 3 5 3 3 3 2 1 1 5 4 4 2 4 4
[235] 4 4 4 4 5 5 3 3 3 3 1 3 3 3 1 4 4 3 2 2 2 2 2 4 4 2 4 1 4 4 4 4 4 3 2 2 1 4 1 1 4 4 4 1 1 1 3 3 3 3 1 1 1 2 2 2 3 4
[313] 4 1 3 2 1 2 3 2 1 3 1 4 1 1 4 5 4 2 2 2 2 2 3 3 3 3 3 2 3 3 3 3 2 3 5 5 3 3 3 3 2 2 3 3 3 1 2 3 3 3 3 2 3 5 4 1 2 4
[391] 2 1 1 3 5 3 1 1 5 1 3 3 5 3 2 3 3 2 2 4 4 3 3 2 1 3 2 2 2 1 1 2 2 1 3 2

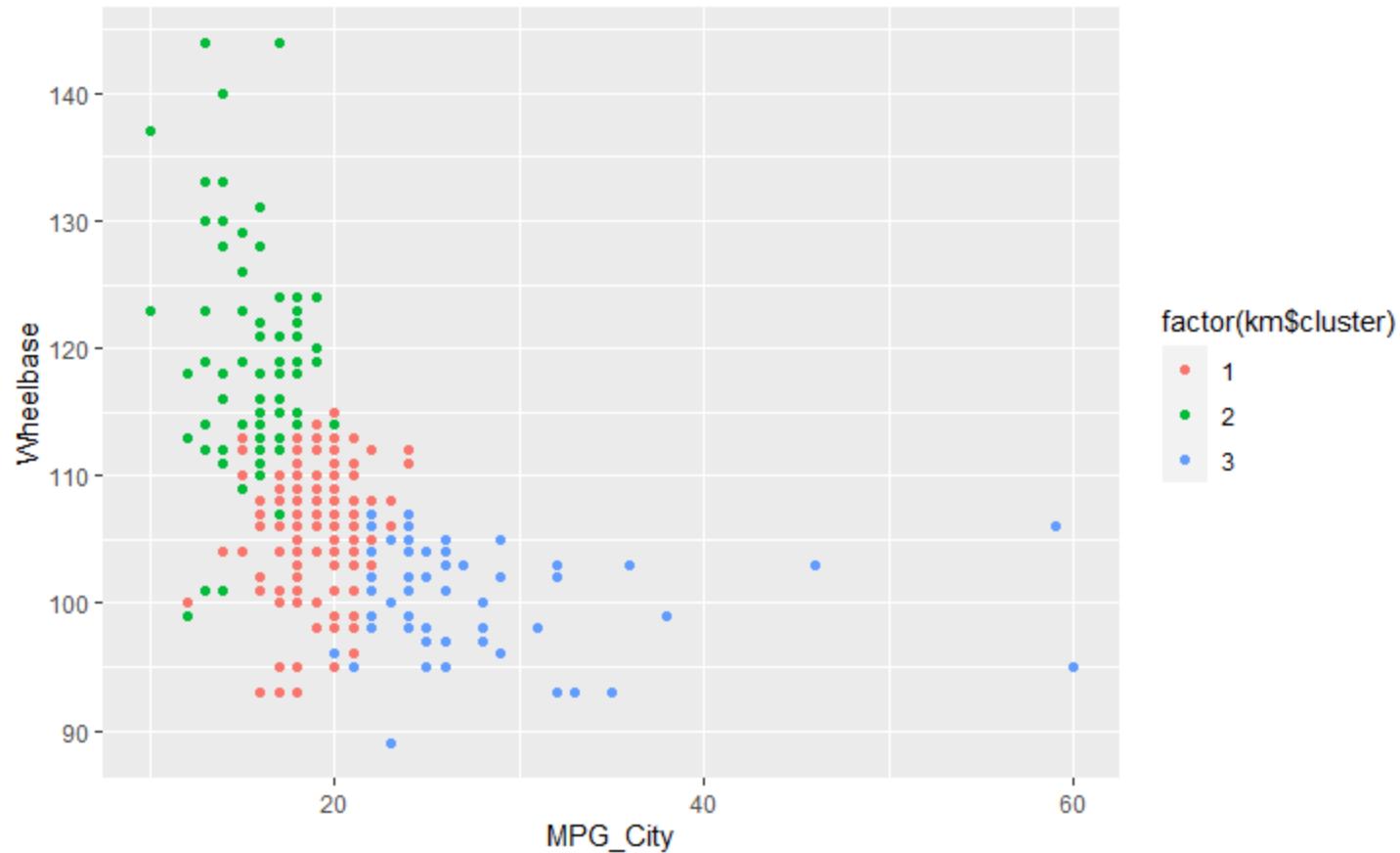
Within cluster sum of squares by cluster:
[1] 170.1216 205.0093 138.0920 435.6868 151.5063
(between_SS / total_SS =  67.7 %)

Available components:

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

Пример

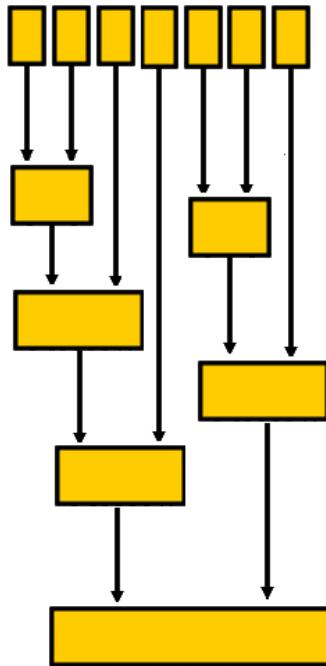
```
> km <- kmeans(std_cars, centers = 3)
>
> ggplot(cars, aes(x = MPG_City,
+                     y = Wheelbase,
+                     color = factor(km$cluster))) +
+   geom_point()
```



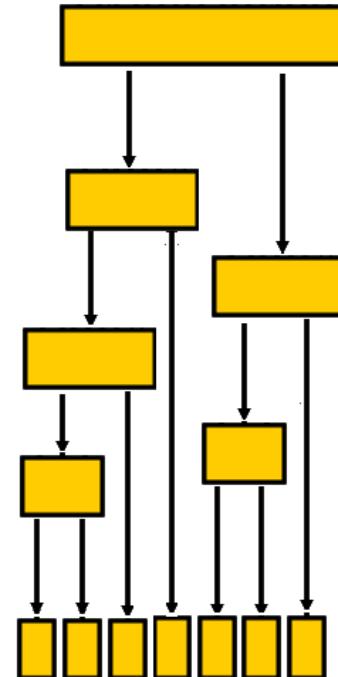
Иерархическая кластеризация

Способ добавления объектов в кластеры

Восходящая («склейка»)

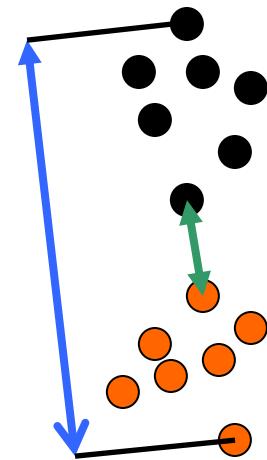


Нисходящая («разбиение»)



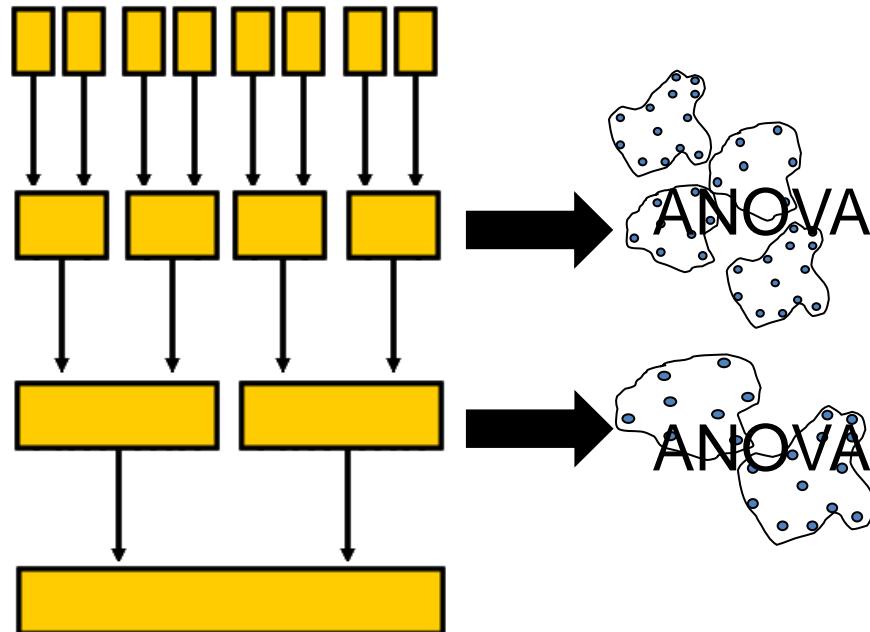
Оценка близости кластеров

- Расчет расстояния на основе попарных расстояний между элементами различных кластеров:
 - **Полное связывание:** наибольшее попарное расстояние. Дает компактные сферические кластеры.
 - **Среднее связывание:** усредненное попарное расстояние.
 - **Единственное связывание:** наименьшее попарное расстояние. Дает «растянутые» кластеры сложной формы.
 - **Центроидное связывание:** расстояние между центрами (мат. ожидание) кластеров.
 - Другие методы (например **метод Ward'a** – минимизирует внутрикластерные дисперсии или другую целевую функцию)



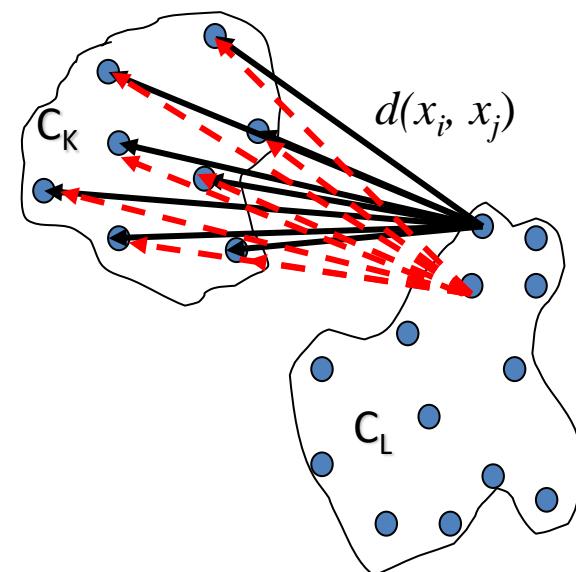
Стандартные меры связывания

- Ward использует ANOVA



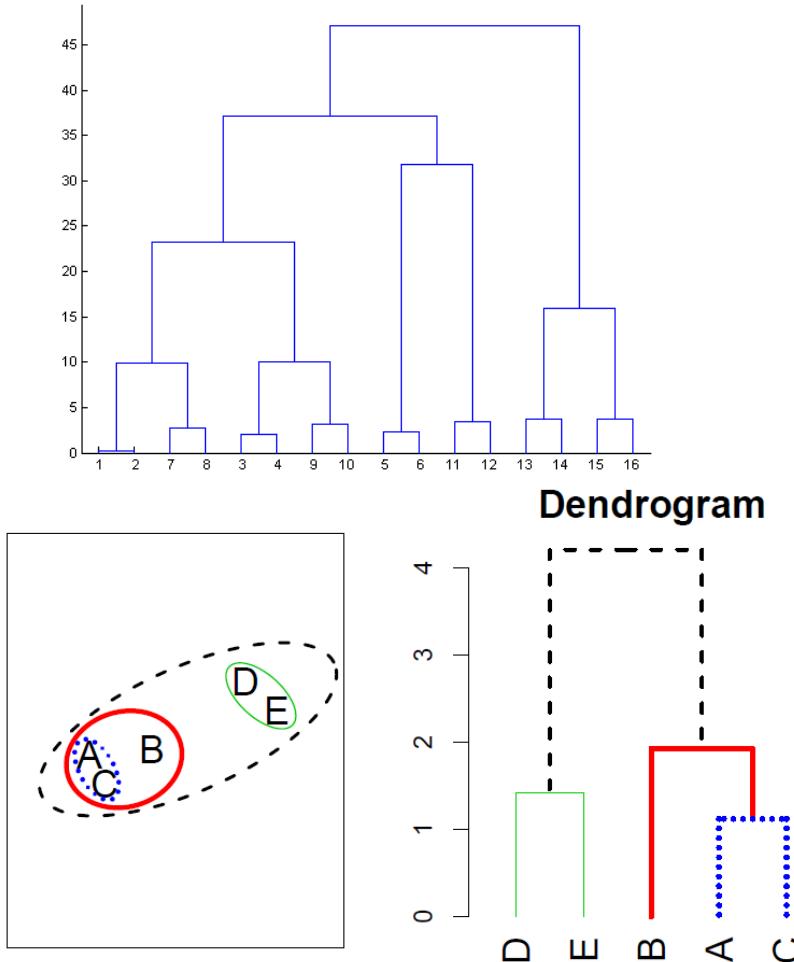
$$D_{KL} = \frac{\|\bar{x}_K - \bar{x}_L\|^2}{\left(\frac{1}{n_K} + \frac{1}{n_L}\right)}$$

- Среднее связывание



$$D_{KL} = \frac{1}{n_K n_L} \sum_{i \in C_K} \sum_{j \in C_L} d(x_i, x_j)$$

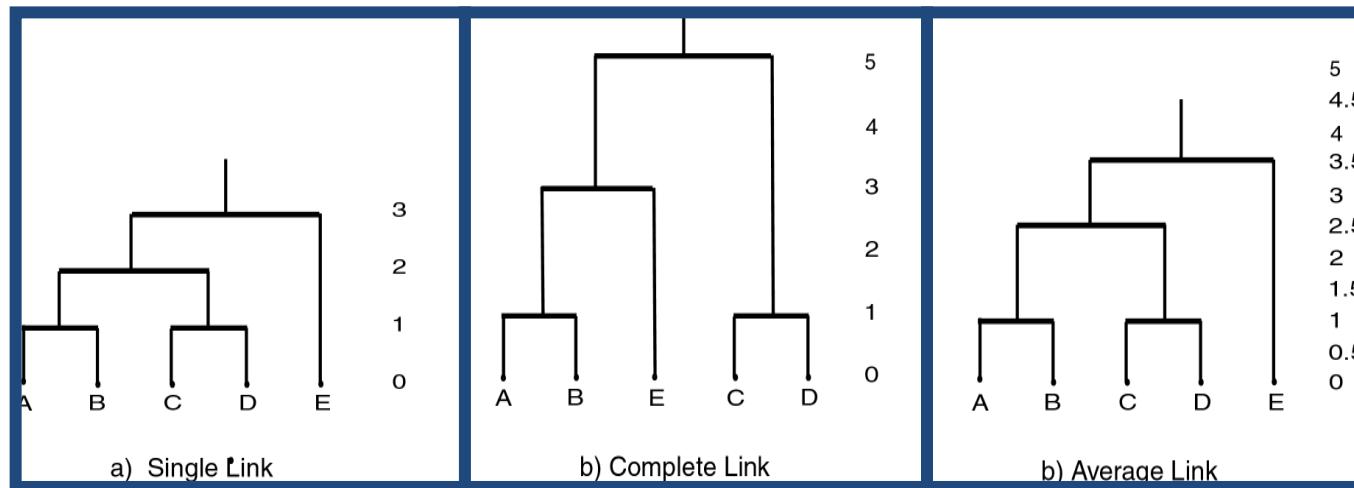
Представление иерархических кластеров - Дендрограмма



- бинарное дерево, описывающее все шаги разбиения
- Корень – общий кластер, листья - элементы
- «Высота» ветвей (до пересечения) – порог расстояния «склейки» («разделения»)
- Результат кластеризации – «срез» дендрограммы

Многое зависит от расстояния

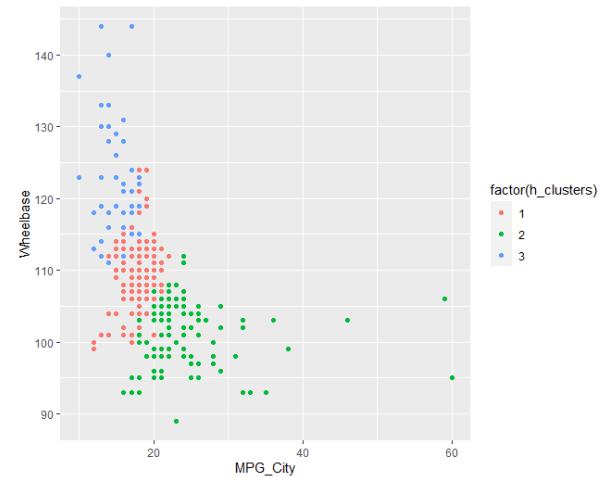
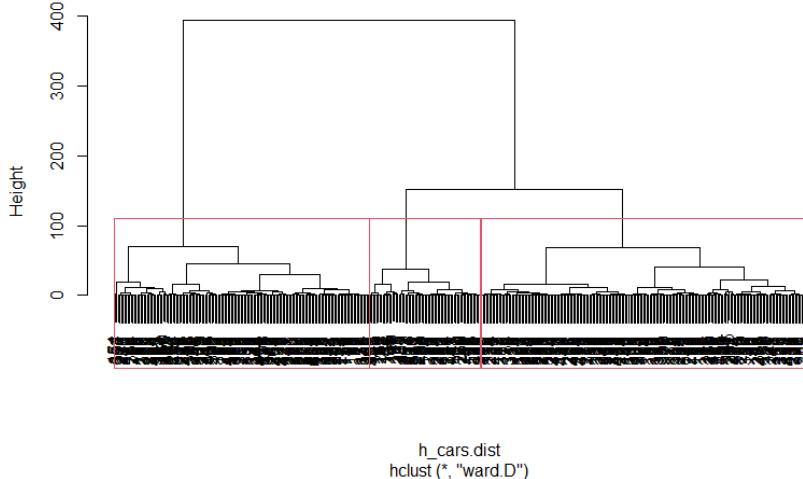
	A	B	C	D	E
A	0	1	2	2	3
B	1	0	2	4	3
C	2	2	0	1	5
D	2	4	1	0	3
E	3	3	5	3	0



Процедуры иерархической кластеризации

```
hclust(d, method = "complete", members = NULL)
rect.hclust(tree, ...)
cutree(tree, ...)
```

```
> h_cars <- cars[c("EngineSize", "MPG_City", "Wheelbase")] %>%
+     mutate_all(~(scale(.) %>% as.vector))
>
> h_cars.dist <- dist(h_cars, method = "manhattan")
> h_cars.clust <- hclust(h_cars.dist, method = "ward.D")
>
> plot(h_cars.clust)
> rect.hclust(h_cars.clust, k = 3)
```

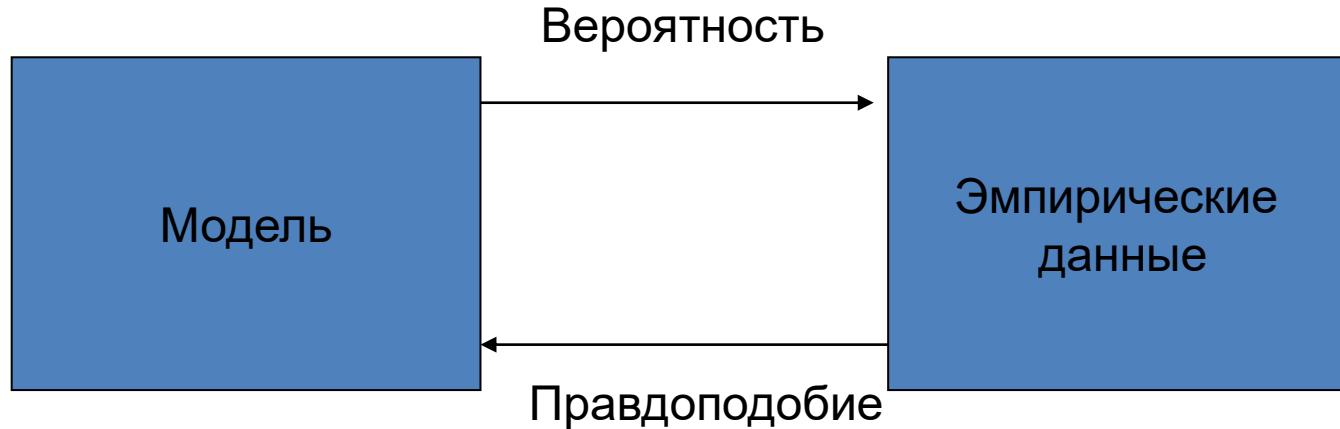


```
> h_clusters <- cutree(h_cars.clust, k = 3)
> ggplot(cars, aes(x = MPG_City, y = Wheelbase,
+                   color = factor(h_clusters))) +
+     geom_point()
```

Обсуждение иерархической кластеризации

- Достоинства:
 - Очень просто и понятно, легко реализовать
 - Наглядные дендрограммы
 - Нет метапараметров
- Недостатки:
 - не масштабируется: временная сложность $O(n^2)$, где n - число кластеризуемых объектов
 - жадный алгоритм - локальная оптимальность с точки зрения минимизации внутриклассовых расстояний и максимизации межклассовых
 - относительно слабая интерпретируемость

Общая идея параметрического подхода в кластеризации (и не только)



- Модель описывает процесс порождения эмпирических данных в терминах теории вероятности (например, плотность распределения для непрерывных данных)
- Наиболее общий подход оценки параметров модели через функцию правдоподобия

Функция правдоподобия

Формула Байеса

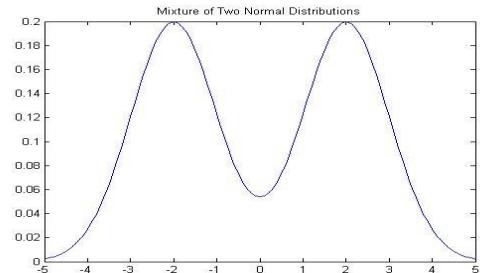
$$P(Model | Data) = \frac{P(Data | Model)P(Model)}{P(Data)}$$

Функция правдоподобия – совместное распределение выборки из параметрического распределения как функция параметра модели.

- Задача – найти «лучшую» модель, ту, которая наиболее точно описывает эмпирические данные.
- Функция правдоподобия – функция модели (параметров модели) при фиксированных данных.
- Спецификация модели – «искусство» (не процедура), фиксируется тип модели (например, распределение), а ищутся «лучшие» параметры.
- Для сложных задач модель = «смесь» (линейная комбинация) распределений.

Примеры

- Испытания Бернулли:
 - Пусть дана последовательность {0,1}: 0,1,1,0,0,1,1,0
 - Ищем модель в классе распределений $B(p)$: $P(X = x) = p^x(1 - p)^{1-x}$
 - Задача - найти наилучший параметр p : $\operatorname{argmax}_p P(Data|B(p))$
- Смесь нормальных распределений:
 - Пусть дана выборка, например, рост людей: 185, 140, 134, 150, 170 ...
 - Предполагаем нормальное распределение, но в среднем женщины ниже мужчин, а значит – «смесь» распределений:
$$\pi_1 N(\mu_1, \sigma_1) + \pi_2 N(\mu_2, \sigma_2)$$
 - где: $N(\mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$
 - Задача - найти параметры смеси:
$$\operatorname{argmax}_{\pi_1, \pi_2, \mu_1, \mu_2, \sigma_1, \sigma_2} P(Data | \pi_1, \pi_2, \mu_1, \mu_2, \sigma_1, \sigma_2)$$



Оценка максимального правдоподобия

- Точечная оценка параметров, которая максимизирует функцию правдоподобия при фиксированной реализации выборки.
- Если A_1, \dots, A_n независимые одинаково распределенные сл. вел.:
$$P(A_1, \dots, A_n) = \prod_{i=1}^n P(A_i)$$
- $l(\text{Data} | \text{Model}) = \log(P(\text{Data} | \text{Model}))$ - логарифмическая функция правдоподобия:
 - \log – монотонная функция, точки экстремумов совпадают
 - вместо произведения – сумма логарифмов
- Максимум функции правдоподобия в нуле производных:
 - Для всех параметров тета:

$$\frac{\partial l(X | \theta_1, \dots, \theta_n)}{\partial \theta_i} = 0$$

- Для нахождения максимума решаем полученную систему уравнений

Пример с распределением Бернулли

$$\begin{aligned}L(p) &= P(0, 1, 1, 0, 0, 1, 0, 1 | p) \\&= P(0|p)P(1|p)\dots P(1|p) \\&= (1-p)p\dots p \\&= p^4(1-p)^4\end{aligned}$$

- Надо найти p , максимизирующий логарифмическое правдоподобие
 $l(p) = \log P(Data/B(p))$

$$\begin{aligned}\ell(p) &= \log L(p) = 4\log(p) + 4\log(1-p) \\ \frac{d\ell(p)}{dp} &= \frac{4}{p} - \frac{4}{1-p} \equiv 0 \\ \rightarrow p &= \frac{1}{2}\end{aligned}$$

Скрытые (латентные) переменные и EM алгоритм

- Задача кластеризации как оценка скрытых (латентных) переменных – «меток» кластеров:
 - Пусть $Data = \{x(1), x(2), \dots, x(n)\}$ - набор сл. векторов «наблюдений»
 - Пусть $H = \{z(1), z(2), \dots, z(n)\}$ - множество соответствующих значений скрытой величины Z , где $z(i)$ соответствует $x(i)$
 - Считаем, что Z – дискретна.
- Оценка скрытых (ненаблюдаемых) величин – цель EM
- Приложения EM:
 - кластеризация
 - заполнение пропущенных значений в выборке
 - поиск скрытых состояний марковской модели

EM Алгоритм

- Логарифмическое правдоподобие наблюдаемых данных:

$$l(\theta) = \log p(D|\theta) = \log \sum_H p(D, H|\theta)$$

- Нужно оценить не только параметры θ , но и H
- Пусть $Q(H)$ есть распределение вероятности скрытых величин
- Неравенство Дженсона (для выпуклой функции):

$$\begin{aligned} \varphi\left(\frac{\sum a_i x_i}{\sum a_i}\right) &\leq \frac{\sum a_i \varphi(x_i)}{\sum a_i}; & \ell(\theta) &= \log \sum_H p(D, H|\theta) \\ && &= \log \sum_H Q(H) \frac{P(D, H|\theta)}{Q(H)} \\ \bullet F(Q, \theta) - \text{нижняя граница } l(\theta) && &\geq \sum_H Q(H) \log \frac{P(D, H|\theta)}{Q(H)} \\ && &= \sum_H Q(H) \log p(D, H|\theta) + \sum_H Q(H) \log \frac{1}{Q(H)} \\ && &= F(Q, \theta) \end{aligned}$$

EM Алгоритм

- Процедура EM (в цикле):

- максимизация F по Q (тета зафиксированы)
 - максимизация F по тета (Q фиксировано)

$$\text{E-step} \quad Q^{k+1} = \operatorname{argmax}_Q F(Q^k, \theta^k)$$

$$\text{M-step} \quad \theta^{k+1} = \operatorname{argmax}_{\theta} F(Q^{k+1}, \theta^k)$$

- E-step: $Q^{k+1} = P(H|D, \theta^k)$

- M-step: $\theta^{k+1} = \operatorname{argmax}_{\theta} \sum_H p(H|D, \theta^k) \log p(D, H|\theta^k)$

EM алгоритм для смеси нормальных распределений

$$f(x) = \sum_{k=1}^K \pi_k f_k(x, \mu_k, \sigma_k)$$

$$P(k|x) = \frac{\pi_k f_k(x, \mu_k, \sigma_k)}{f(x)}$$

$$\pi_k = \frac{1}{n} \sum_{i=1}^n P(k|x(i))$$

$$\mu_k = \frac{1}{n\pi_k} \sum_{i=1}^n P(k|x(i))x(i)$$

$$\sigma_k = \frac{1}{n\pi_k} \sum_{i=1}^n P(k|x(i))(x(i) - \mu_k)^2$$

Смесь нормальных распределений

E Step

M-Step

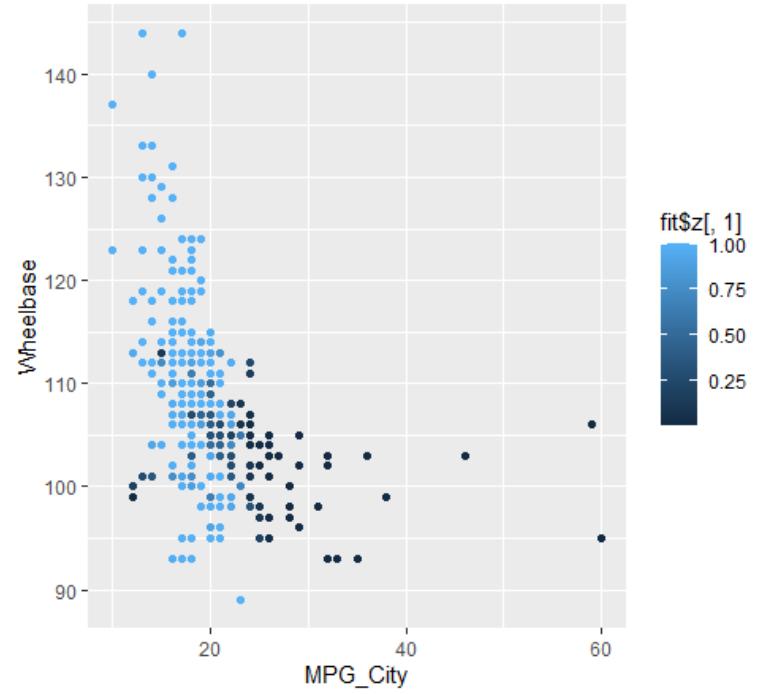
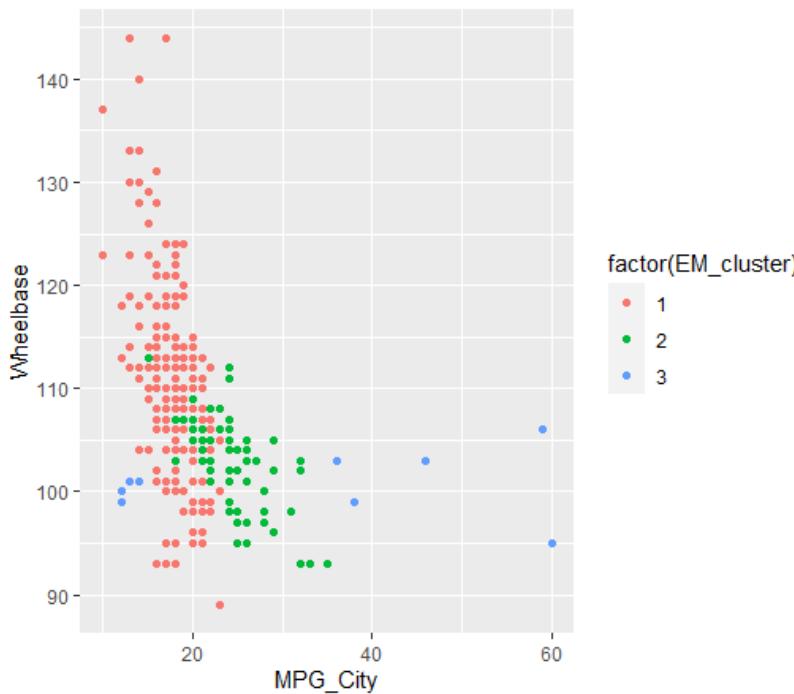
■ Похоже по сути на k-means, но:

- Оценка не только мат. ожидания, но и дисперсии (размер кластера)
- «перекрывающиеся» кластеры, лучше с выбросами

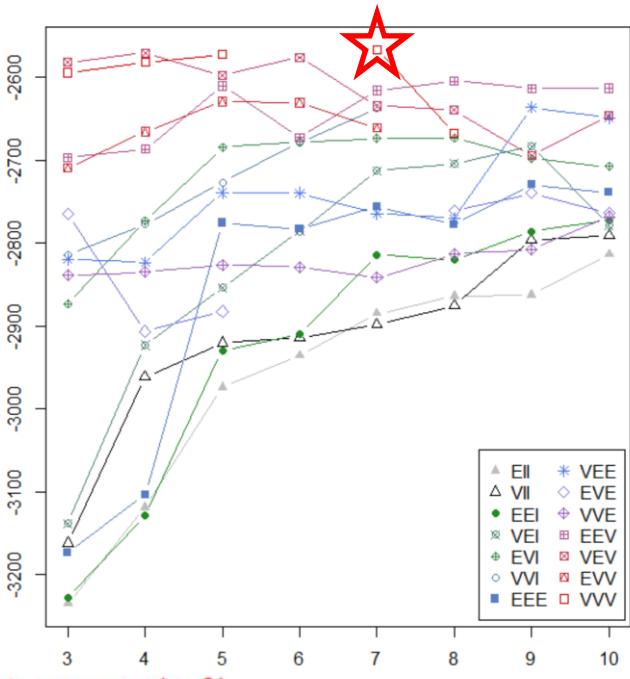
Процедура Mclus

```
Mclust(data, G = NULL, modelNames = NULL, prior = NULL,  
control = emControl(), initialization = NULL, warn =  
mclust.options("warn"), x = NULL, verbose = interactive(), ...)
```

```
> fit <- Mclust(std_cars, G=3)  
> EM_cluster <- apply(fit$z, 1, which.max)  
> ggplot(cars, aes(x = MPG_City, y = Wheelbase,  
+ color = factor(EM_cluster))) +  
+ geom_point()  
> ggplot(cars, aes(x = MPG_City, y = Wheelbase,  
+ color = fit$z[, 1])) +  
+ geom_point()
```



Выбор модели в Mclust



```
> summary(mc0)
```

```
Gaussian finite mixture model fitted by EM algorithm
```

```
Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 7 components:
```

log-likelihood	n	df	BIC	ICL
-1073.96	428	69	-2565.999	-2736.795

```
Clustering table:
```

1	2	3	4	5	6	7
161	14	34	7	17	106	89

```
> mc0<-Mclust(std_cars,G=3:10)
```

```
fitting ...
```

```
|=====
```

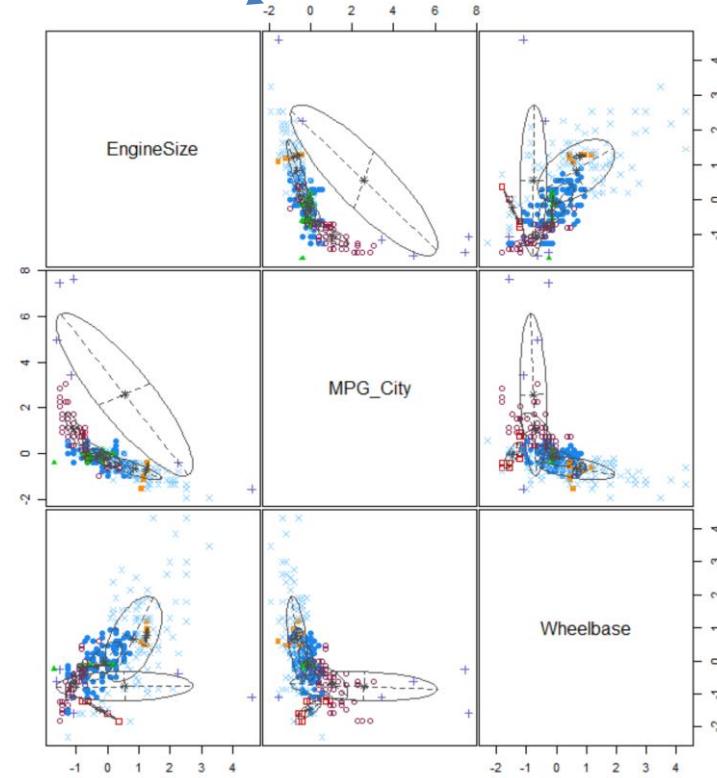
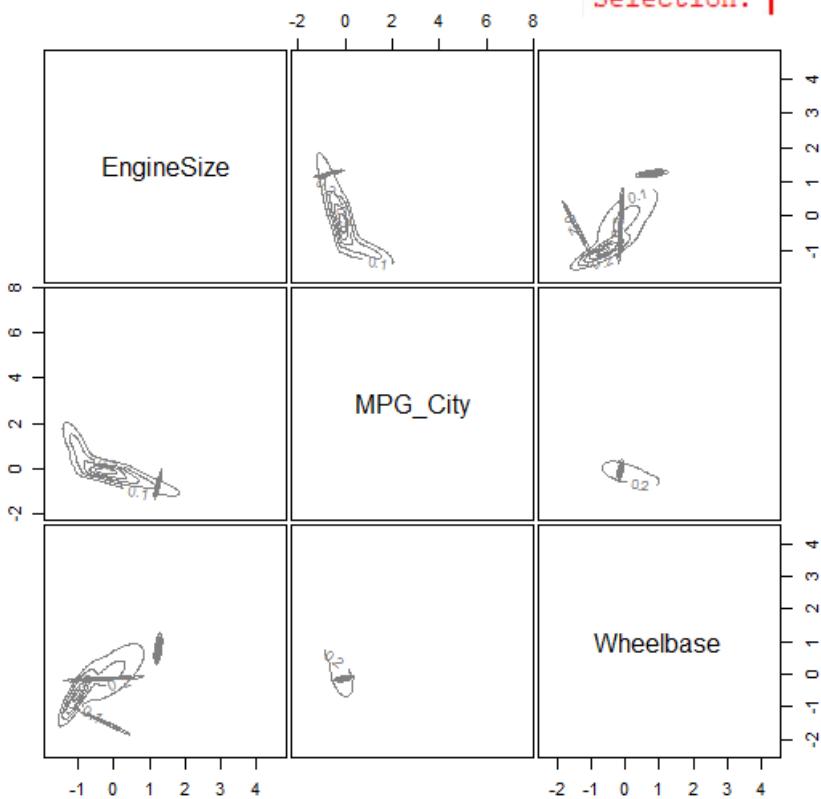
- Выбор лучшей модели на основе BIC - «восходящая» иерархическая кластеризация
- Тип моделей:
 - "E": equal variance (univariate)
 - "V": variable variance (univariate)
 - "EII": spherical, equal volume
 - "VII": spherical, unequal volume
 - "EEI": diagonal, equal volume and shape
 - "VEI": diagonal, varying volume, equal shape
 - "EVI": diagonal, equal volume, varying shape
 - "VVI": diagonal, varying volume and shape
 - "EEE": ellipsoidal, equal volume, shape, and orientation
 - "EEV": ellipsoidal, equal volume and equal shape
 - "VEV": ellipsoidal, equal shape
 - "VVV": ellipsoidal, varying volume, shape, and orientation.

Оценка плотности

```
> plot(mc0)
Model-based clustering plots:
```

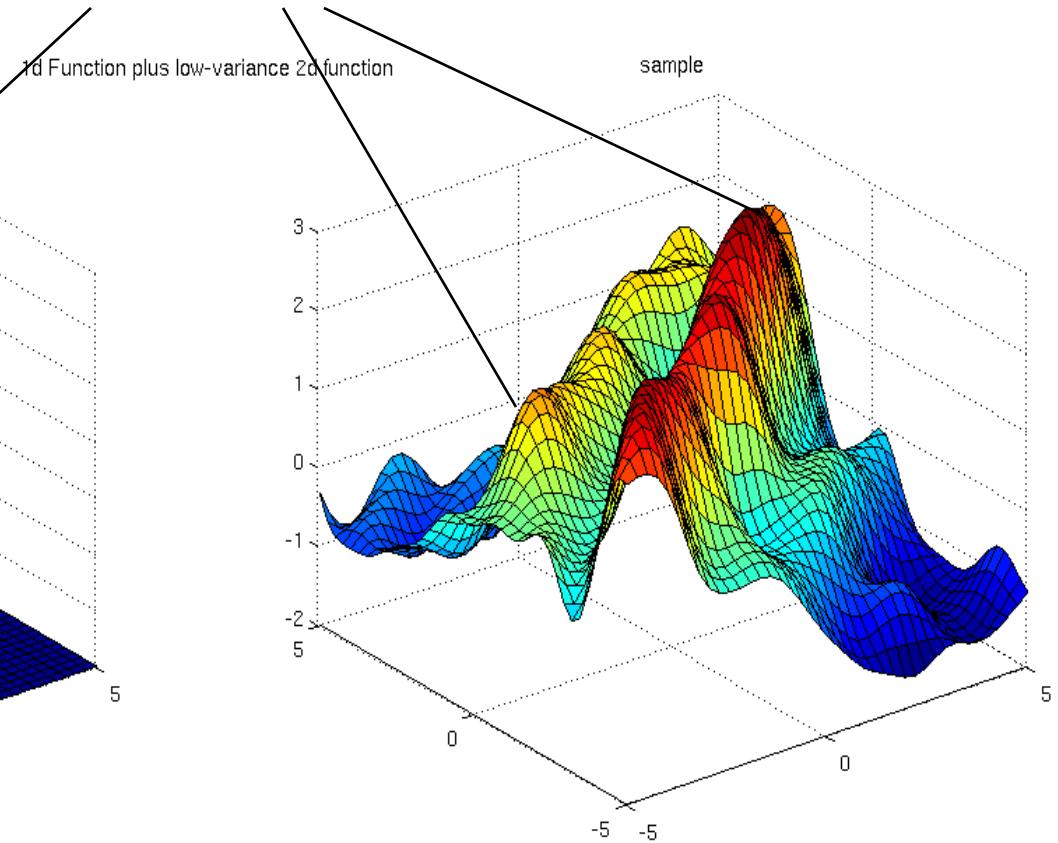
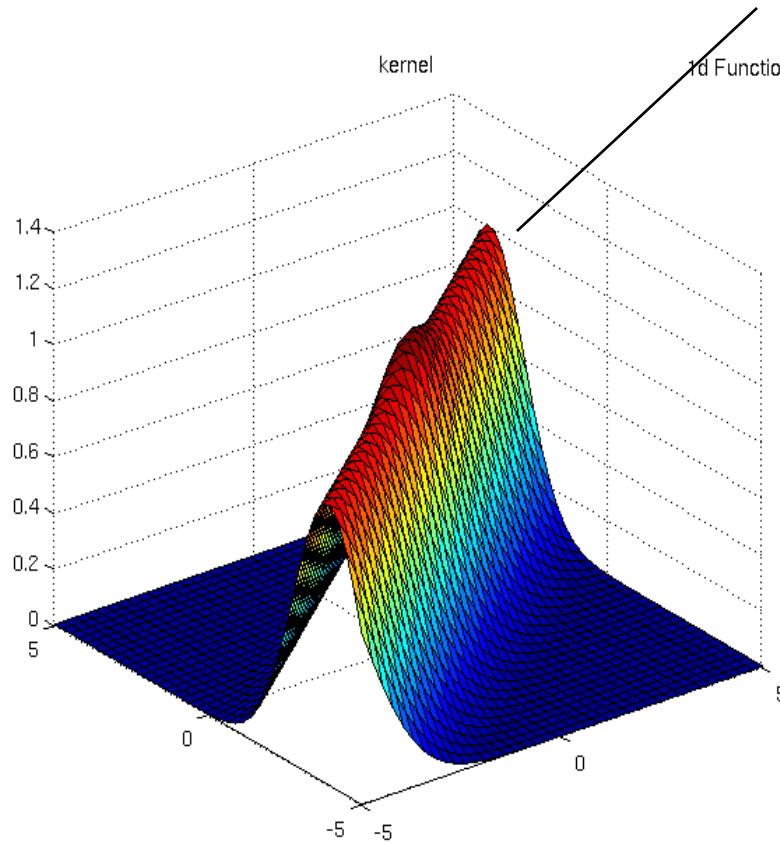
- 1: BIC
- 2: classification
- 3: uncertainty
- 4: density

Selection: |

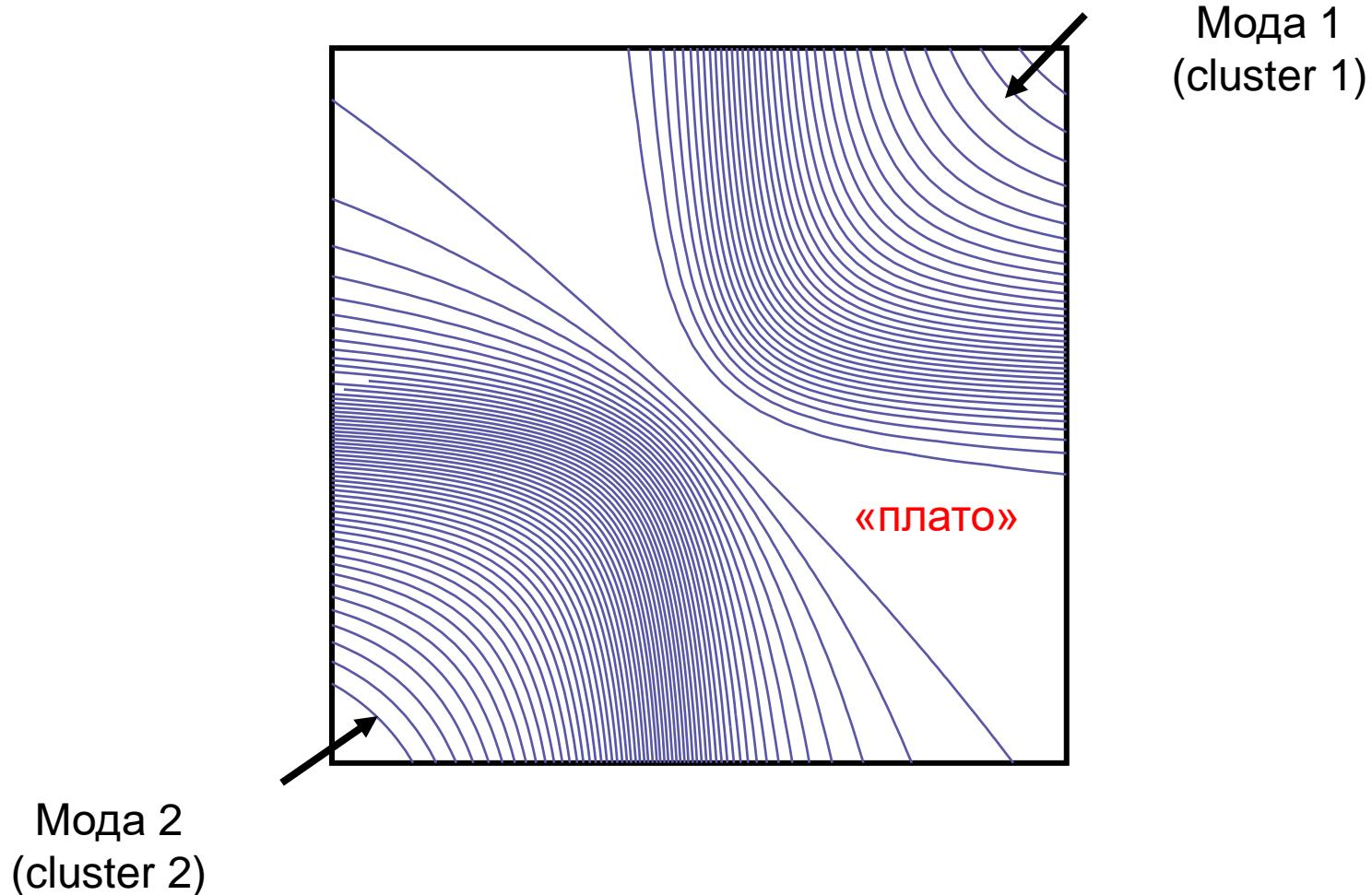


Kernel-приближение плотности распределения

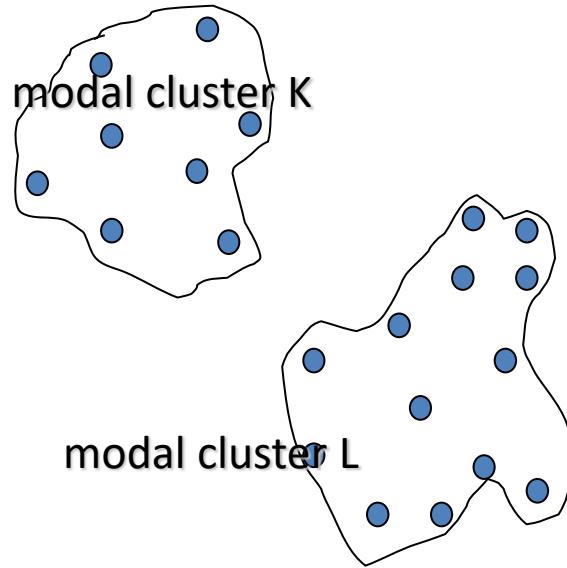
Моды



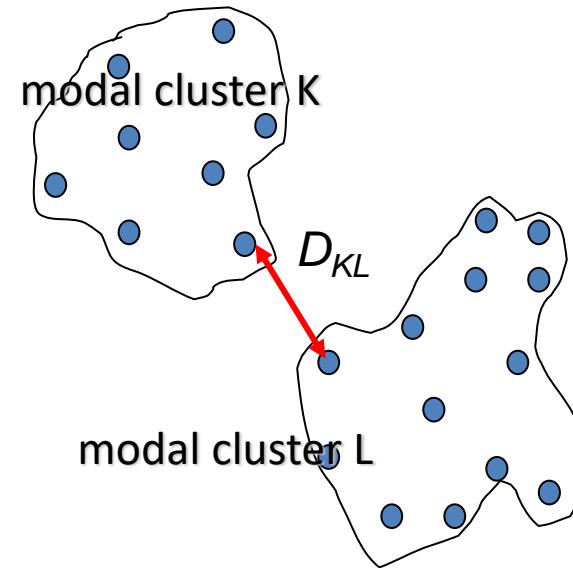
Поиск «плато» и «пиков» плотности распределения



Двух шаговое объединение кластеров



1. Поиск кластеров по
модам



2. Применение
single linkage

- На первом шаге строятся изолированные кластеры по оценке плотности распределения.
- На втором шаге проверяется, что кластер имеет размер больше или равный n . Если кластер – кандидат на «объединение», то используется *single linkage* для склейки.

Непараметрическая кластеризация

```
kepdf(x, eval.points = x, kernel = "gaussian", bwtype = "fixed", h =  
      h.norm(x), hx = NULL, alpha = 1/2)  
pdfCluster(x, graphtype, hmult, Q = "QJ", lambda = 0.1, grid.pairs =  
      10, n.grid = min(round((5 + sqrt(NROW(x))) * 4), NROW(x), ...))
```

- Хорошие результаты для компактных кластеров.
- Находит кластеры сложной формы.
- Менее чувствительна к выбросам, шкалам и зависимостям в пространстве признаков.
- Не требует указывать число кластеров заранее.

Оценка плотности керпдф

```
> kd<-kepdf(std_cars)
> summary(kd)
An S4 object of class "kepdf"

The highest density data point has position 37 in the sample data

Rows of 75 % top density data points: 1 2 3 4 5 6 8 9 10 11 12 13 14 15 16 17
Rows of 50 % top density data points: 1 2 3 4 5 6 8 9 10 11 12 13 14 15 16 17
Rows of 25 % top density data points: 3 4 10 12 13 14 15 16 17 25 29 30 31 32
> print(kd)
An S4 object of class "kepdf"

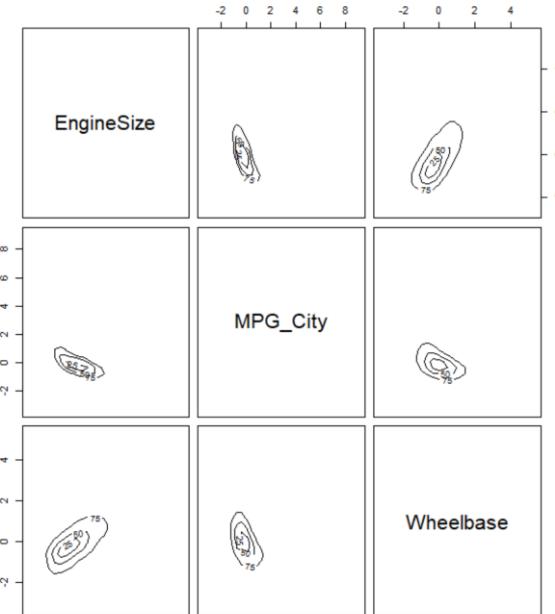
Call: kepdf(x = std_cars)

Kernel:
[1] "gaussian"

Estimator type: fixed bandwidth

Diagonal elements of the smoothing matrix: 0.4076009 0.4076009 0.4076009

Density estimate at evaluation points:
 [1] 0.085959534 0.085565799 0.115387757 0.141429837 0.077431922 0.077431922 0.028874944 0.070767143 0.069611716
[10] 0.115803426 0.075696539 0.099394980 0.142108466 0.130557394 0.132026059 0.114807594 0.108647912 0.061071328
[19] 0.039475305 0.017177609 0.044691114 0.015027190 0.018258095 0.016518632 0.130557394 0.024250591 0.066559202
[28] 0.060666360 0.132721463 0.132721463 0.122053310 0.122053310 0.150150592 0.150150592 0.150150592 0.037910698
[37] 0.151955619 0.065827960 0.066013222 0.051824517 0.023125287 0.077137855 0.077137855 0.044751762 0.032177179
[46] 0.122053310 0.057197041 0.111896822 0.141238753 0.076985897 0.080736244 0.094686668 0.076985897 0.063481736
```



Кластеризация pdfCluster

```
> dc<-pdfCluster(std_cars)
> summary(dc)
An S4 object of class "pdfCluster"

Call: pdfCluster(x = std_cars)

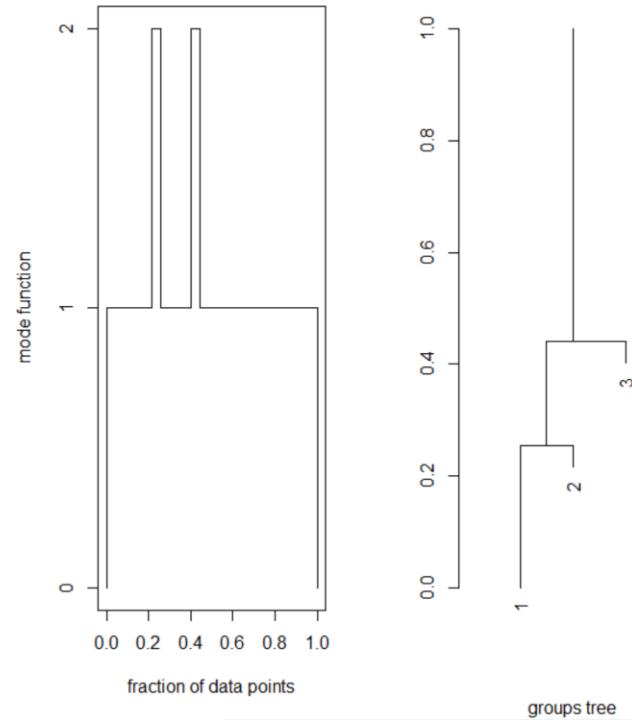
Initial groupings:
label 1 2 3 NA
count 101 10 8 309

Final groupings:
label 1 2 3
count 182 115 131

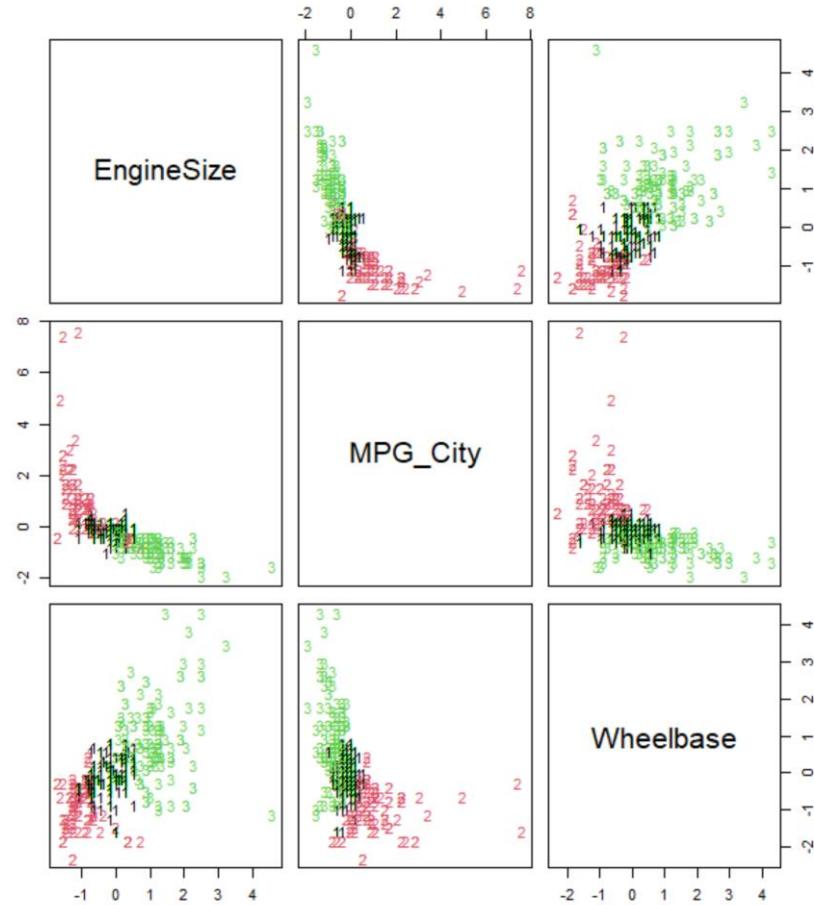
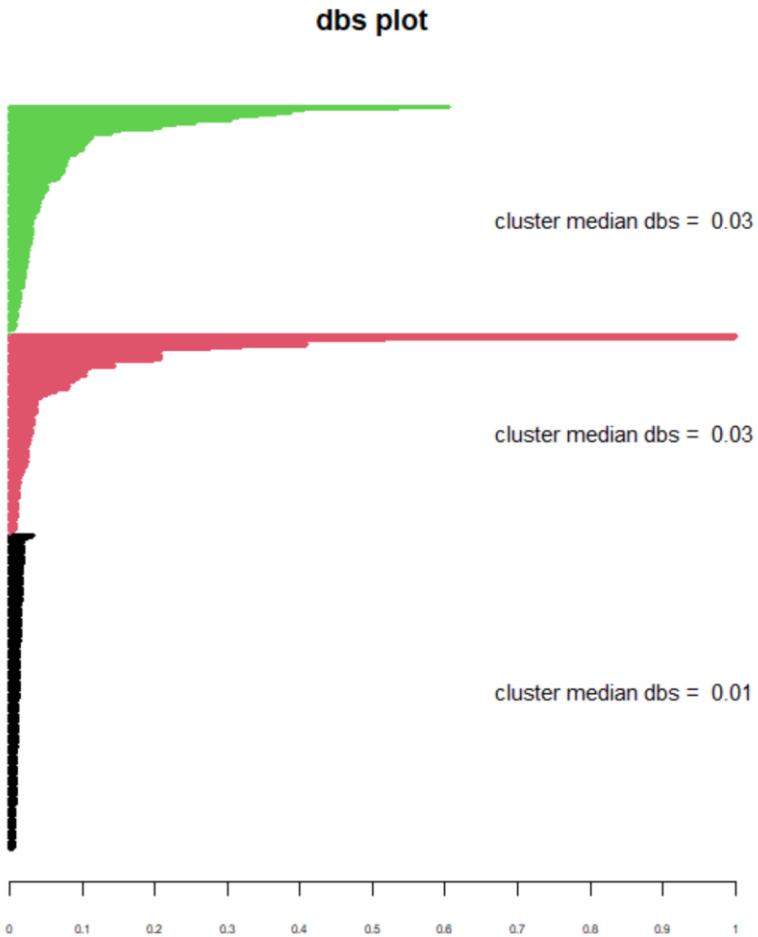
Groups tree (here 'h' denotes 'height'):
--[dendrogram w/ 1 branches and 3 members at h = 1]
`--[dendrogram w/ 2 branches and 3 members at h = 0.441]
 |--[dendrogram w/ 2 branches and 2 members at h = 0.255]
 | |--leaf "1"
 | '--leaf "2" (h= 0.216 )
`--leaf "3" (h= 0.402 )
```

Final groupings:

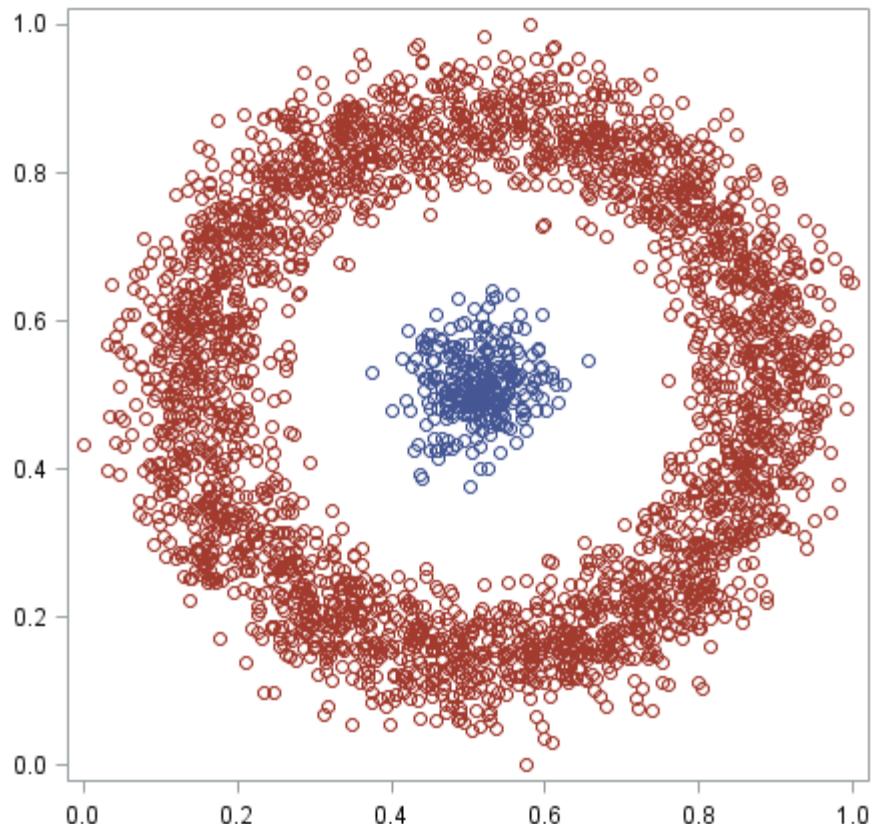
```
[1] 1 2 1 1 1 1 1 2 2 1 1 1 1 1 1 1 3 3 3 3 2 2 2 1 3 1 3 1 1 1 1 1 1 1 1
[37] 1 1 3 3 3 1 1 2 1 1 3 1 1 1 1 1 1 3 3 3 1 3 3 3 3 3 3 3 3 2 2 2 2 2 2
[73] 1 2 1 1 1 1 1 1 3 1 3 3 3 1 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 1 3 3 2 2
[109] 1 1 1 1 1 3 3 3 3 3 3 3 3 1 2 2 2 2 2 1 1 1 3 3 3 3 1 3 3 3 3 2 2 1 3 3 3
[145] 3 1 3 3 3 2 2 1 1 1 2 2 2 2 2 2 1 1 3 3 2 3 1 2 2 2 2 2 2 1 1 1 1 1 1 1 1
[181] 1 1 1 3 3 3 3 1 1 1 1 3 3 3 3 3 3 3 1 2 1 1 2 2 2 2 2 1 1 1 3 2 3
[217] 3 1 3 3 1 1 1 1 1 3 3 3 1 1 3 3 3 3 2 2 2 2 2 2 1 2 2 2 2 2 3 3
[253] 3 2 1 1 1 1 1 1 1 3 3 1 3 1 3 3 3 3 2 1 1 1 3 3 1 3 3 3 1 3 3 1 3 3
[289] 1 2 2 1 2 1 1 1 1 2 3 1 1 2 2 2 1 1 1 3 3 1 1 3 3 1 2 1 3 1 2 1 1 2
[325] 1 3 1 3 3 2 3 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2 2 1 2 2 2 2 1 1 1 1 1 1 1
[361] 2 1 1 2 1 1 2 2 2 2 2 1 2 2 3 3 1 3 2 2 2 2 2 2 1 2 1 1 1 1 1 1 3 3 2
[397] 2 1 3 3 2 3 2 2 2 2 2 1 3 3 3 2 2 3 1 2 1 1 1 1 1 1 1 1 2 1
```



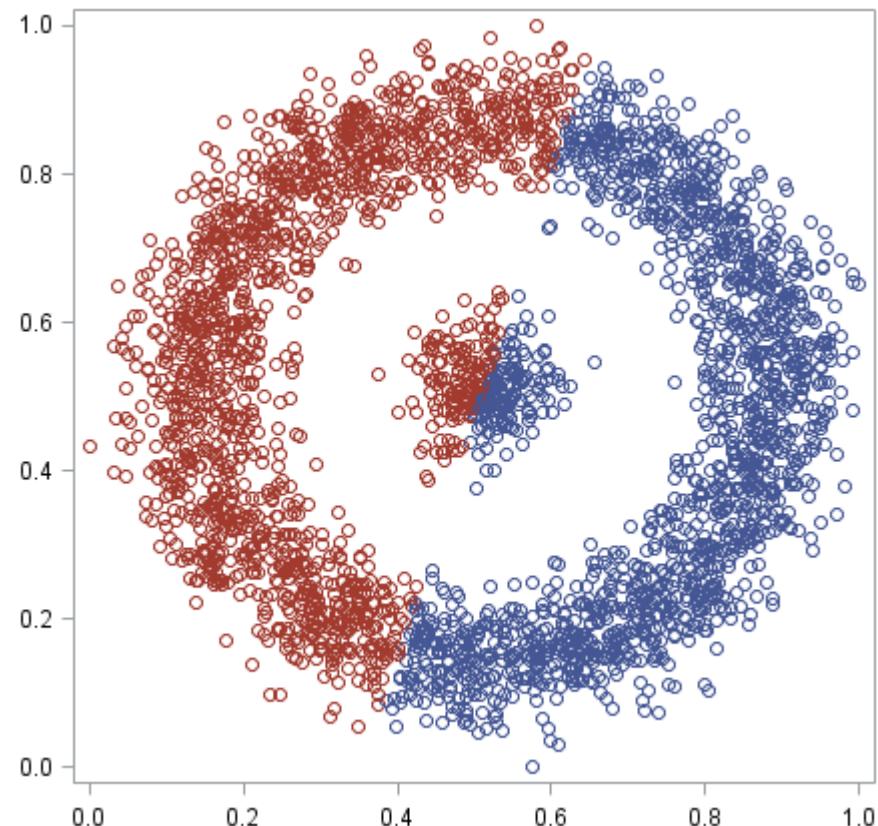
Кластеризация pdfCluster



pdfCluster



kmeans



Определение числа кластеров

SAS Cubic Clustering Criterion (CCC) (Sarle, 1983)

- Основная идея: сравнение R^2 (для отображения матрицы данных с помощью индикаторной матрицы в прототипы кластеров) для заданной модели кластеризации с $E(R^2)$ для равномерно распределенного множества прототипов кластеров (как наихудший возможный вариант)

$$CCC = \log \left[\frac{1 - E(R^2)}{1 - R^2} \right] \frac{\sqrt{np/2}}{(0.001 + E(R^2))^{1.2}}$$

- n = число наблюдений, p = число переменных, Y = $n \times p$ матрица данных (стандартизированных), M = $q \times p$ матрица центров кластеров, X = индикаторная матрица ($x_{ik}=1$ если i принадлежит k)
- $T = Y'Y$, $B = X'X$, $W = T - B$, $R^2 = 1 - \text{trace}(W)/\text{trace}(T)$
- Должно быть больше 2 и выбирается первый локальный максимум

Определение числа кластеров

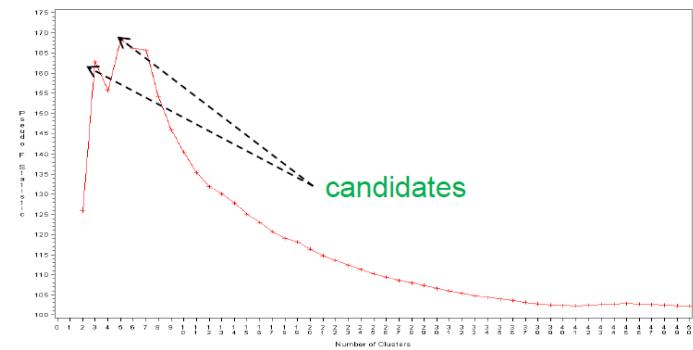
По сути – перебор моделей с разным числом кластеров и выбор по некоторому критерию, например Pseudo-F (Calinski and Habarasz, 1974)

$$W_j = \sum_{i \in C_j} \| \mathbf{T}(v_i) - \bar{\mathbf{T}}_j \|^2 \quad Q = \sum_{i=1}^V \| \mathbf{T}(v_i) - \bar{\mathbf{T}} \|^2$$

- N- число объектов,
- G – число кластеров,
- W-сумма внутри-кластерных квадратов расстояний,
- Q – сумма всех кв. расстояний до общего центра

$$pseudoF = \frac{(Q - \sum_{g=1}^G W_g) / (G - 1)}{\sum_{g=1}^G W_g / (N - G)}$$

- Выбирается вариант с максимальным *pseudoF*



Определение числа кластеров

```
NbClust(data, diss="NULL", distance = "euclidean", min.nc=2,
        max.nc=15, method = "ward", index = "all", alphaBeale = 0.1)
```

```
> nc<-NbClust(std_cars,method="complete")
*** : The Hubert index is a graphical method of determining the number of clusters.
      In the plot of Hubert index, we seek a significant knee that corresponds to a
      significant increase of the value of the measure i.e the significant peak in Hubert
      index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
      In the plot of D index, we seek a significant knee (the significant peak in Dindex
      second differences plot) that corresponds to a significant increase of the value of
      the measure.

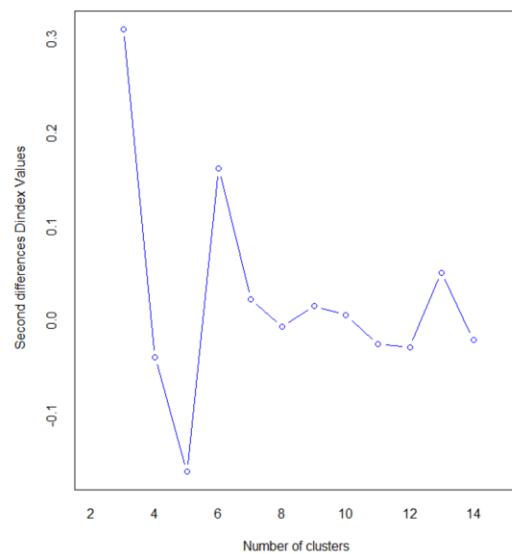
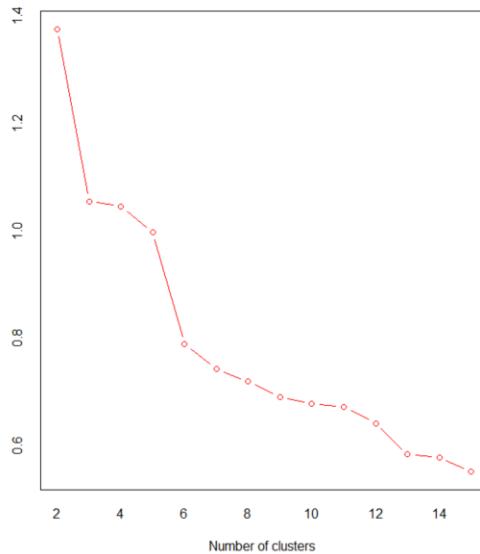
*****
* Among all indices:
* 7 proposed 2 as the best number of clusters
* 10 proposed 3 as the best number of clusters
* 1 proposed 5 as the best number of clusters
* 1 proposed 6 as the best number of clusters
* 1 proposed 7 as the best number of clusters
* 1 proposed 11 as the best number of clusters
* 1 proposed 13 as the best number of clusters
* 2 proposed 15 as the best number of clusters

***** Conclusion *****

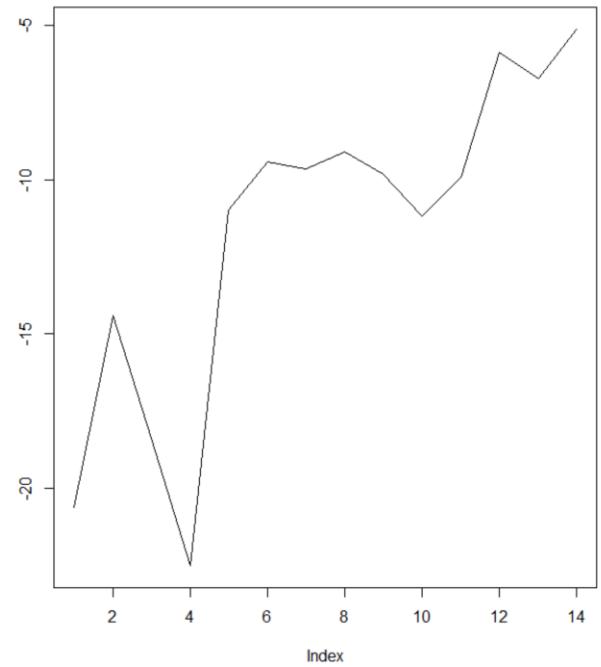
* According to the majority rule, the best number of clusters is 3
```

Определение числа кластеров

```
$Best.nc
      KL      CH Hartigan      CCC      Scott Marriot TrCovW
Number_clusters 6.0000    7.000   3.0000 15.0000    3.0000      3      3
Value_Index     9.1603 226.152 326.6398 -5.1246 424.9814 29497847 252347
                  TraceW Friedman Rubin Cindex      DB Silhouette Duda
Number_clusters 3.0000    3.0000 13.0000 2.0000 2.0000      2.0000 3.0000
Value_Index     487.2883 4.2714 -1.1044 0.2301 0.4328      0.7114 1.0347
                  PseudoT2 Beale Ratkowsky      Ball PtBiserial Frey McClain
Number_clusters 3.0000 2.0000    3.0000 3.0000 5.0000 2.0000      2.000
Value_Index     -3.8189 1.3586 0.4081 357.6025 0.5383 8.1571      0.004
                  Dunn Hubert SDindex Dindex      SDbw
Number_clusters 2.0000      0 11.0000      0 15.0000
Value_Index     0.1938      0 2.2327      0 0.1304
```



```
> plot(nc$All.index[, "CCC"], type="l")
~
```



Этапы кластерного анализа

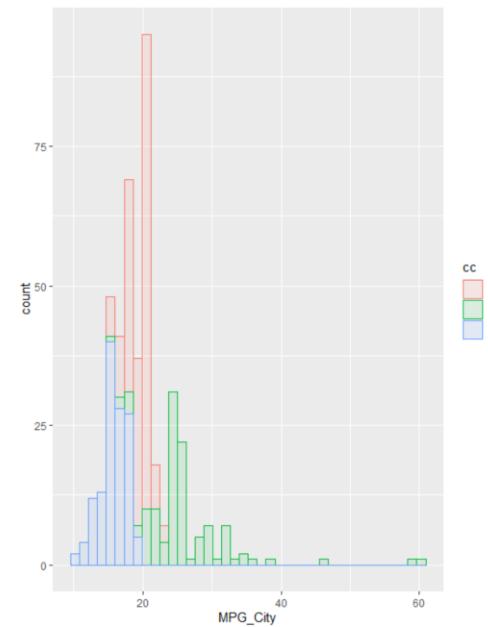
Интерпретация результатов



Подходы к профилированию

- Базовые подходы:
 - «Прототип»
 - Использование ANOVA
 - Проекции, графики и дендрограммы
- Продвинутые подходы:
 - Сравнить распределения переменных с другими кластерами или со всей выборкой
 - С помощью дискриминирующих моделей

```
> dc<-pdfCluster(std_cars)
> dfc<-cbind(df,dc@clusters)
> cc<-factor(dfc@clusters)
> ggplot(dfc, aes(x=MPG_City, color=cc, fill=cc)) +
+ geom_histogram(alpha=0.1, bins=40)
```



Этапы кластерного анализа

Применение результатов анализа к новым данным



Резюме курса (рассмотренные темы)

1. Основы программирования на R: объекты, функции, управляющие конструкции, структуры данных и управление ими.
2. Объектно-ориентированное программирование в R: S3 и S4.
3. Базовые графические возможности R: диаграммы для анализа данных.
4. Проверка гипотез и дисперсионный анализ, одномерная и многомерная ANOVA, ANCOVA, анализ контрастов
5. Построение и интерпретация простых регрессионных моделей, проверка условий применимости, оценки качества и сравнения моделей, мультиколлинеарность.
6. Регуляризация, алгоритмы пошагового отбора переменных и преобразование признакового пространства, факторный анализ, выбросы
7. Обобщенные линейные модели: логнормальная, гамма, пуассоновская регрессии, чрезмерная дисперсия, отрицательно-биномиальная регрессия
8. Нелинейные модели: нелинейные уравнения и проблема оптимизации, непараметрические регрессии, полиномиальная регрессия
9. Анализ категориальных данных: таблицы частот, логистическая регрессия, балансировка выборки
10. Кластеризация: на основе разбиения, иерархическая, на основе параметрических и непараметрических вероятностных моделей



https://docs.google.com/forms/d/e/1FAIpQLSfRgsr5I4KQopWogA2JgsDHTcR_43GdKCNFwg4NTBJFcAfDg/viewform

Резюме курса (чего не хватает?)

1. Смешанные модели
2. Анализ выживаемости
3. Байесовские методы
4. Планирование статистического эксперимента
5. Робастные методы
6. Взаимодействие с Big Data

Что дальше?

1. Сдача практической части
 - Согласно расписанию
 - На автомат нужно набрать 14+ на отл, 9+ на хоро
2. Защита работ
 - Готовить все, сдается одна («на выбор проверяющего»)
 - Если не сдали, то минус итоговый балл и следующая попытка и т.д.
 - У кого проблема с посещаемостью – сдает все пять работ
3. Кто не сдал практику ...