Лабораторная работа №5

(часть 1)

Задачи классификации и кластеризации

5.1. Цель работы:

Закрепить знания, об алгоритмах классификации и кластеризации данных, ознакомиться с некоторыми функциями языка R, осуществляющими этот вид анализа, принципами их работы. Научиться визуализировать результаты работы функций кластерного анализа и классификаторов, интерпретировать полученные результаты.

5.2. Общие сведения

5.2.1. Классификация

Под ***классификацией*** будем понимать отнесение объектов (наблюдений, событий) к одному из ***заранее известных классов***.

Классификация — это закономерность, позволяющая делать вывод относительно определения характеристик конкретной группы. Таким образом, для проведения классификации должны присутствовать признаки, характеризующие группу, к которой принадлежит то или иное событие или объект (обычно при этом на основании анализа уже классифицированных событий формулируются правила причисление объекта к группе).

Классификация относится ***к стратегии обучения с учителем*** (supervised learning), которое также именуют контролируемым или управляемым обучением.

Классификация может быть ***одномерной*** (по одному признаку) и ***многомерной*** (по двум и более признакам).

5.2.2. Кластеризация

Задача ***кластеризации*** сходна с задачей классификации, но ее отличие в том, что классы изучаемого набора данных заранее не предопределены и формируются автоматически, поэтому синонимами термина "кластеризация" являются "автоматическая классификация", "***обучение без учителя***" и "таксономия".

Кластеризация предназначена для разбиения совокупности объектов на однородные группы (кластеры или классы). Если данные выборки представить, как точки в признаковом пространстве, то задача кластеризации сводится к определению "сгустков точек".

Цель кластеризации - поиск таких сгустков.

Кластеризация является описательной процедурой, она не делает никаких статистических выводов, но дает возможность провести **разведочный анализ** и изучить "структуру данных".

Само понятие "кластер" определено неоднозначно: каждый алгоритм (или даже каждое исследование одного и того же алгоритма) формирует свои "кластеры". Переводится понятие кластер (cluster) как "скопление", "гроздь".

Таблица 5.1. Сравнение классификации и кластеризации

Характеристика	Классификация	Кластеризация
Контролируемость обучения	Контролируемое обучение	Неконтролируемое обучение
Стратегия	Обучение с учителем	Обучение без учителя

Наличие метки класса	Обучающее множество сопровождается меткой, указывающей класс, к которому относится наблюдение	Метки класса обучающего множества неизвестны
Основание для классификации	Новые данные классифицируются на основании обучающего множества	Дано множество данных с целью установления существования классов или кластеров данных

5.3. Некоторые функции для классификации в R

Шаг 1.

(Сначала проведите дескриптивный анализ и анализ распределения независимых параметров)

Шаг 2.

Поскольку в большинстве задач количество классов заранее неизвестно, то для выделения предполагаемого количества групп (классов), проведем иерархический кластерный анализ, результатом которого является дендрограмма (рис.5.1), позволяющая сформировать представление исследователя о возможном разделении выборки на классы.

Шаг 2.1. Чтение данных, отбросить столбец, классифицирующий ирисы

data(iris)

```
fix(iris)
labels_iris<-iris[,5]  #  Этот столбец позже используем для подписей на дендро-
грамме
iris_C<-iris[,-5]
```

Шаг 2.2. Удаление пропущенных значений

В данной задаче пропущенных значений нет.
Удалять нечего.

Шаг 2.3. Стандартизация переменных.

В данной задаче переменные существенно различны (рис 5.1.).
Стандартизировать надо.

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

Рис. 5.1. Фрагмен таблицы iris.

```
maxs <- apply(iris_C, 2, max)
mins <- apply(iris_C, 2, min)
iris_C <- scale(iris_C, center = mins, scale = maxs - mins)
```

Шаг 2.3. Создаем матрицу попарных расстояний (по умолчанию - Евклидово расстояние)

```
dist.iris <- dist(iris_C)
```

Шаг 2.4. Проводим кластерный анализ, результаты записываем в список clust.iris

```
# hclust ожидает матрицу расстояния, а не исходные данные.
clust.iris <- hclust(dist.iris, "ward.D")
```

Шаг 2.5. Прежде, чем построить дендрограмму, давайте оценим оптимальное число кластеров.

Чтобы найти оптимальное количество кластеров для k -средства, его рекомендуется выбирать исходя из:

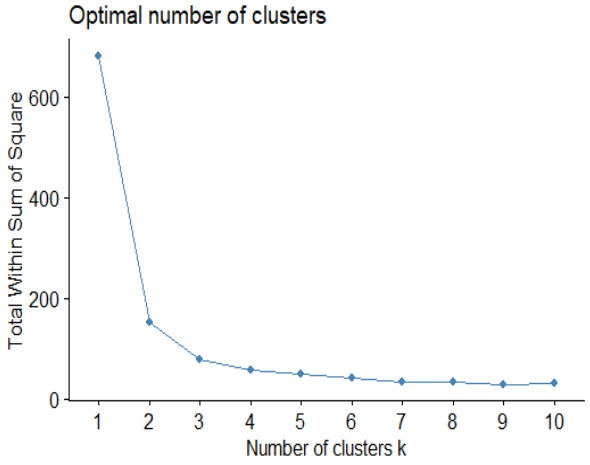
- контекст рассматриваемой проблемы, например, если вы знаете, что в ваших данных есть определенное количество групп (у вас есть сильные ожидания или гипотезы, однако этот вариант является субъективным), или
- следующие четыре подхода:
    1. Метод локтя (который использует суммы квадратов внутри кластера)
    2. Метод среднего силуэта
    3. Метод статистики разрывов
    4. Алгоритм на основе консенсуса

Ниже мы покажем R-код для этих 4 методов, дополнительную теоретическую информацию можно найти здесь .

2.5.1 Количество кластеров по сравнению с общей суммой квадратов (Метод локтя)

Метод Локтя рассматривает общую сумму квадратов внутри кластера (WSS) как функцию количества кластеров.

Для этого используем **функцию fviz_nbclust()** , чтобы построить график количества кластеров по сравнению с общей суммой квадратов. Нам понадобятся следующие пакет и библиотеки:




```
install.packages("factoextra")
library (factoextra)
library (cluster)
fviz_nbclust(iris_C, kmeans, method = "wss")
```

Метод основан на минимизации внутригруппового разброса W_{total}. Внутригрупповой разброс $W_{total}=\sum_k W(C_k)\rightarrow\min.$

Для этого графика кажется, что отрезки, символизирующие внутригрупповой разброс начинают уменьшаться при k = 3 кластера.

Рис. 5.2. Диаграмма "Метод локтя" по сумме квадратов

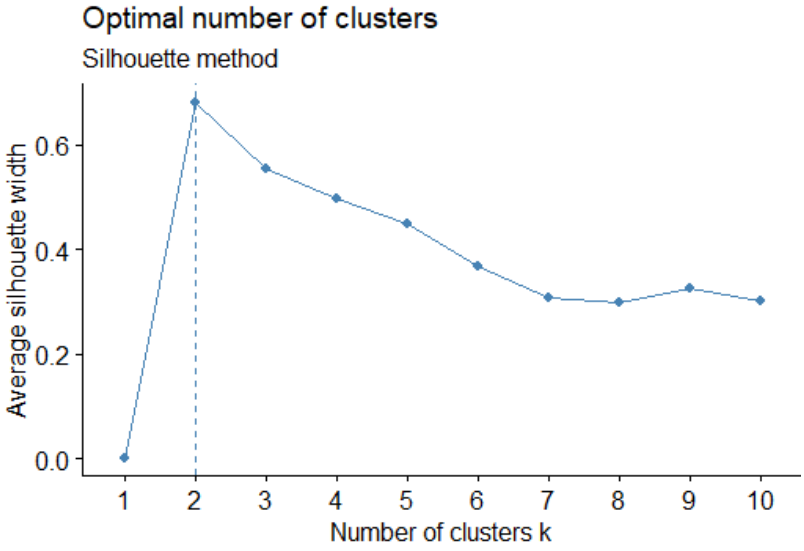
2.5.2 Количество кластеров и статистика пропусков (метод среднего силуэта)

Метод Silhouette измеряет качество кластеризации и определяет, насколько хорошо каждая точка лежит в пределах своего кластера. Число кластеров k в этом случае определяется по средней ширине силуэта каждого кластера (average silhouette width), а точнее по отношению:

$$s_i=(b(i)-a(i))/\max[b(i),a(i)],$$

где $a(i)$ - среднее расстояние между объектами i-го кластера, $b(i)$ - среднее расстояние от объектов i-го кластера до другого кластера, самого близкого к i-му. Диаграмму силуэтов можно построить с использованием функции `fviz_silhouette()` из пакета `factoextra`:

```
# Silhouette method
fviz_nbclust(ir, kmeans, method = "silhouette") +
  labs(subtitle = "Silhouette method")
```



Метод Силуэта предполагает 2 кластера – чем ближе к 1, тем лучше кластеризация.

Рис. 5.3. Диаграмма "Метод среднего силуэта"

Интерпретация силуэта:

- $s(i)\in[-1,1]$:
  - **$s(i)\approx1$** — объект хорошо кластеризован (находится далеко от соседних кластеров).
  - **$s(i)\approx0$** — объект находится на границе кластеров.
  - **$s(i)\approx-1$** — объект отнесён к неверному кластеру.

Общий силуэт (Silhouette Score):

Среднее значение $s(i)$ по всем объектам:

$$\text{Silhouette Score} = \frac{1}{N}\sum_{i=1}^{N} s(i)$$

Чем ближе к **1**, тем лучше кластеризация.

Плюсы и минусы метода силуэта:

+ Плюсы:

- Учитывает и компактность, и разделимость.
- Работает с любым алгоритмом кластеризации.
- Интерпретируемость (значения от -1 до 1).

- Минусы:

- Вычислительно затратен для больших данных ($O(N^2)$).
- Плохо работает с кластерами сложной формы (как и все метрики, основанные на расстояниях).

Вывод:

Метод силуэта — один из лучших способов оценки кластеризации, особенно если данные имеют сферическую или выпуклую форму. Используйте его вместе с визуализацией и другими метриками (например, Davies-Bouldin Index).

2.5.3 Количество кластеров и статистика пропусков (статистика разрыва)

Еще один способ определить оптимальное количество кластеров — использовать метрику, известную как статистика разрыва, которая сравнивает общую внутрикластерную дисперсию для разных значений k с их ожидаемыми значениями для распределения без кластеризации.

Этот метод генерируются на основе ресэмплинга и имитационных процедур Монте-Карло. Пусть $E^*_n\{\log(W^*_k)\}$ обозначает оценку средней дисперсии W^*_k, полученной бутстреп-методом, когда k кластеров образованы случайными наборами объектов из исходной выборки размером n.

Тогда статистика

$$Gap_n(k) = E^*_n\{\log(W^*_k)\} - \log(W_k)$$

определяет отклонение наблюдаемой дисперсии Wk

от ее ожидаемой величины при справедливости нулевой гипотезы о том, что исходные данные образуют только один кластер.

При сравнительном анализе последовательности значений $Gap_n(k), k=2,\ldots,K_{max}$

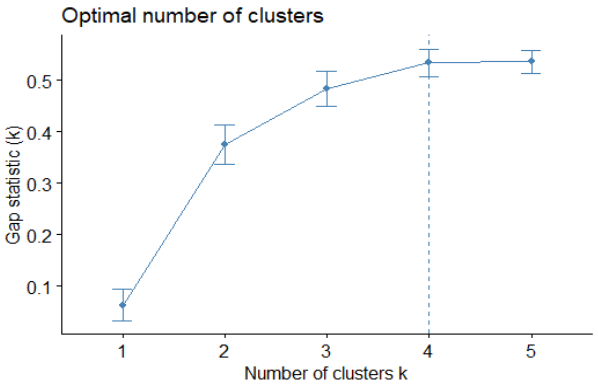
наибольшее значение статистики соответствует наиболее полезной группировке, дисперсия которой максимально меньше внутригрупповой дисперсии кластеров, собранных из случайных объектов исходной выборки:

Мы можем рассчитать статистику разрыва для каждого количества кластеров, используя **функцию clusGap()** из пакета *кластер*, а также график зависимости кластеров от статистики разрыва, используя **функцию fviz_gap_stat()** :

```
fviz_nbclust( iris, kmeans, method = "wss")
#посчитать статистику разрыва,  базирующуюся на числе кластеров K.max =5:
gap_stat <- clusGap(iris_C, FUN = kmeans, nstart = 5,K.max =5, B = 5)

#plot number of clusters vs. gap statistic
```

```
fviz_gap_stat(gap_stat)
```



Из графика видно, что статистика зазора максимальна при k = 4 кластера, что почти соответствует методу локтя, который мы использовали ранее.

Рис. 5.4. Диаграмма "Статистики разрыва"

Интерпретация графика:

- Ищем k_k, где **Gap(k)** достигает максимума или начинает снижаться.
- Учитываем стандартную ошибку (вертикальные линии).

+ Преимущества:

- Автоматически определяет оптимальное k_k.
- Учитывает структуру данных, сравнивая с "нулевой" моделью.
- Работает с любым алгоритмом кластеризации (K-Means, иерархическая и др.).

- Недостатки:

- Вычислительно сложный (особенно для больших данных).
- Требует выбора "нулевого" распределения (обычно равномерное, но может не подходить для всех случаев).
- Менее точен для кластеров сложной формы.

2.5.4 Алгоритм на основе консенсуса

Поскольку ни один из методов явно не является лучшим, четвертая альтернатива — запустить множество методов и выбрать то количество кластеров, которое является наиболее согласованным (т. е. найти консенсус).

Это легко сделать с помощью `n_clusters()` функции из `{parameters}` пакета:

```
#Алгоритм на основе консенсуса
install.packages('parameters')
library(parameters)

n_clust <- n_clusters(iris[, -5],
                      package = c("easystats", "NbClust", "mclust"),
                      standardize = FALSE)
n_clust
plot(n_clust)
```

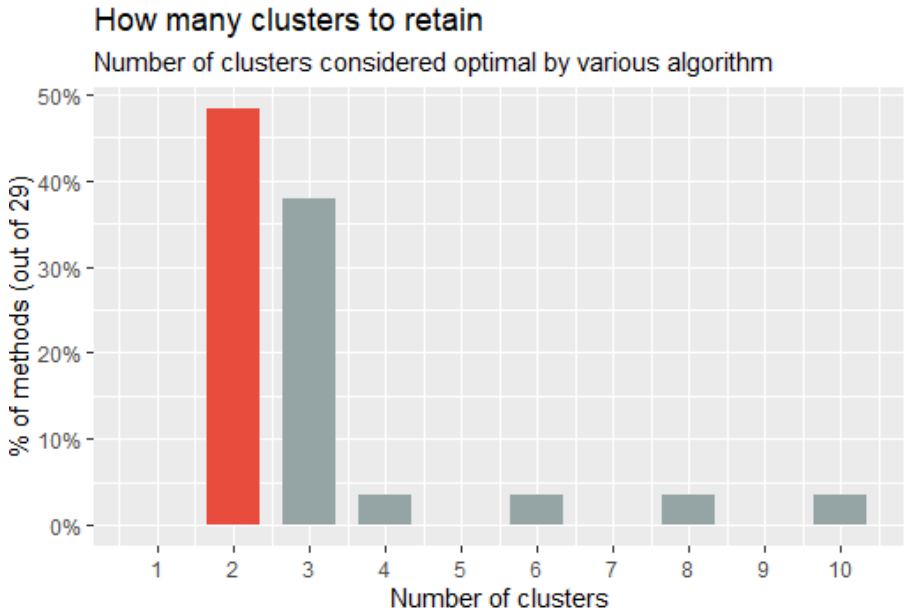


Рис. 5.5. Диаграмма на основе консенсуса

По итогу мы ориентируемся на 3 кластера. Тем более, что датасет Iris хорошо известен, в нем действительно 3 вида цветов ирисов.

+ Преимущества:

- Устойчивость к шуму и выбросам.
- Работает с любым базовым алгоритмом.
- Уменьшает зависимость от начальной инициализации.

- Недостатки:

- Вычислительно затратен ($O(M \cdot N^2)$).
- Требует настройки M (число итераций).

Сравнение методов определения кластеров

Метод	**Основан на**	**Плюсы**	**Минусы**
Elbow Method	Within-Cluster-Sum-of-Squares (WCSS)	Простота	Субъективность
Silhouette Score	Расстояния внутри/между кластерами	Интерпретируемость	Плохо для невыпуклых кластеров
Gap Statistic	Сравнение с нулевым распределением	Автоматический выбор k_k	Медленный
Consensus	Генерация множества кластеризаций	Устойчивость к шуму и выбросам. Работает с любым базовым алгоритмом.	Вычислительно затратен

Шаг 2.5. Построение дендрограммы

```
plot(clust.iris)
```

Увидев дендрограмму (рис.5.6), понимаем, что наиболее целесообразно разделить ее на 3 либо на 4 кластера. Попробуем разделить на 4 кластера.

```
plot(clust.iris, labels_iris, cex=0.5)
rect.hclust(clust.iris, k=4, border="red")
```

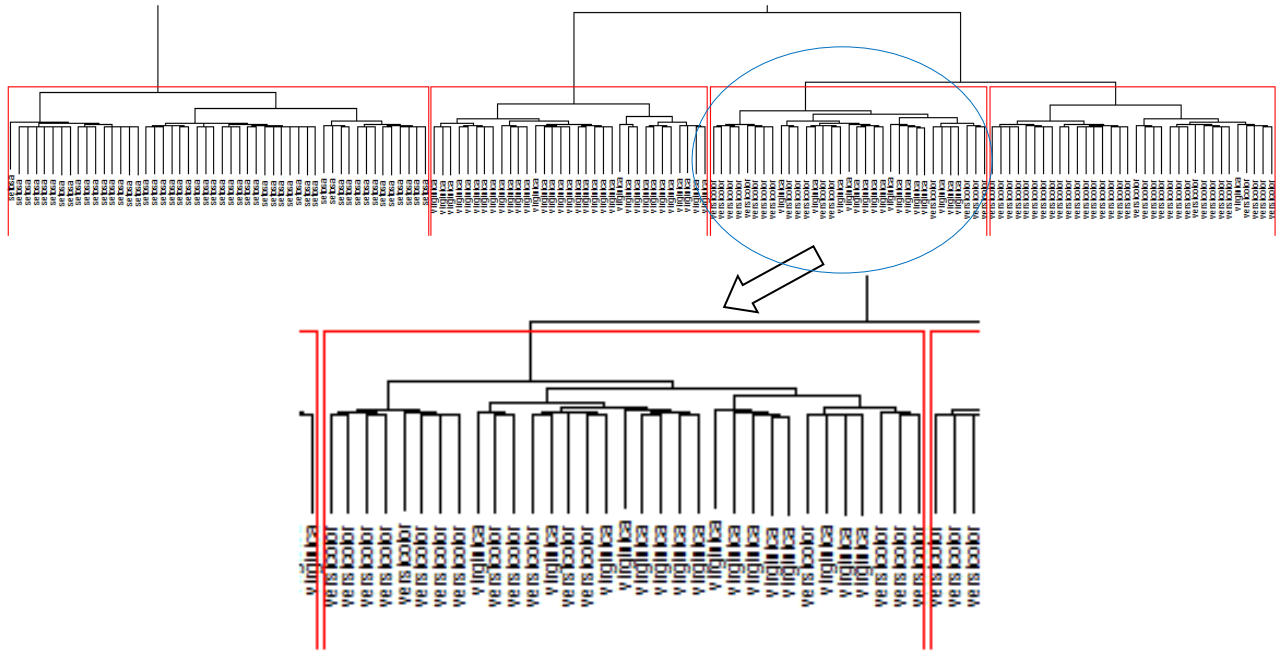


Рис. 5.5. Дендрограмма ирисов, деление на 4 кластера

Присмотревшись внимательнее ко второму справа кластеру, обнаруживаем, что он состоит из смеси двух сортов ирисов – versicolor и verginica, и, согласно дендрограмме, его лучше присоединить к первому справа кластеру. Поэтому дальнейший анализ проводим для 3-х кластеров.

Мы можем обрезать дендрограмму на нужной нам высоте и “приближать” нужную нам ветку.

```
hcd <- as.dendrogram(clust.iris)
hcd # дендрограмма, только пока не построенная
# далее будут графики подряд, в 3 строки, 1 столбец
par(mfrow = c(3, 1))
# верхняя часть при обрезке
plot(cut(hcd, h = 4)$upper, main = "Верхняя часть дендрограммы")
# первая ветка нижней части (Branch 1)
plot(cut(hcd, h = 4)$lower[[1]], main = "Branch 1")

# третья ветка нижней части (Branch 3)
plot(cut(hcd, h = 4)$lower[[3]],  main = "Branch 3")
dev.off()
```

Шаг 2.5. Разделим выборку на 3 кластера

```
#  Вектор groups содержит номер кластера, в который попал классифицируемый объект
groups <- cutree(clust.iris, k=3)

#  Для каждой группы определяем средние значения характеристик и строим датафрейм.

#  в 1-ом кластере
g1<-colMeans(iris[groups==1, 1:4])
#  во 2-ом кластере
g2<-colMeans(iris[groups==2, 1:4])
#  в 3-ем кластере
g3<-colMeans(iris[groups==3, 1:4])

df<-data.frame(g1,g2,g3)
df1<-t(df)
df<-t(df1)
barplot(df, col=c("red","green","blue","yellow")) #  построим график
```

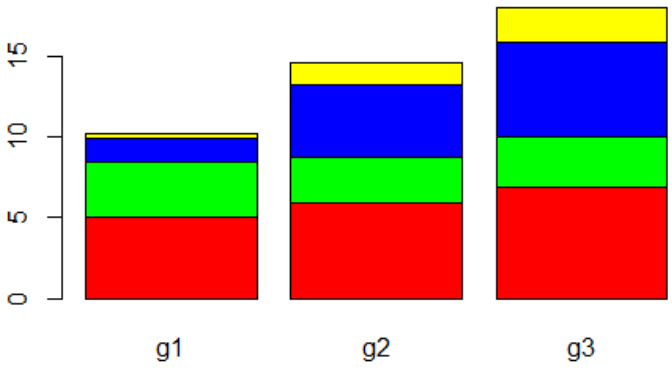


Рис. 5.7.Группы ирисов

Поработав немного над графиком, получим те же данные, несколько в другом виде (рис.5.7.)

```
barplot(df, ylim=c(0,12),
        main = "Groups of iris", axes = FALSE,
        col=c("red","green","blue","yellow"), beside=TRUE)
axis(2, at = 0:5, labels = 0:5)
legend("top", legend = rownames(df), col=c("red","green","blue","yellow"), lwd=
10, bty = "n")
```

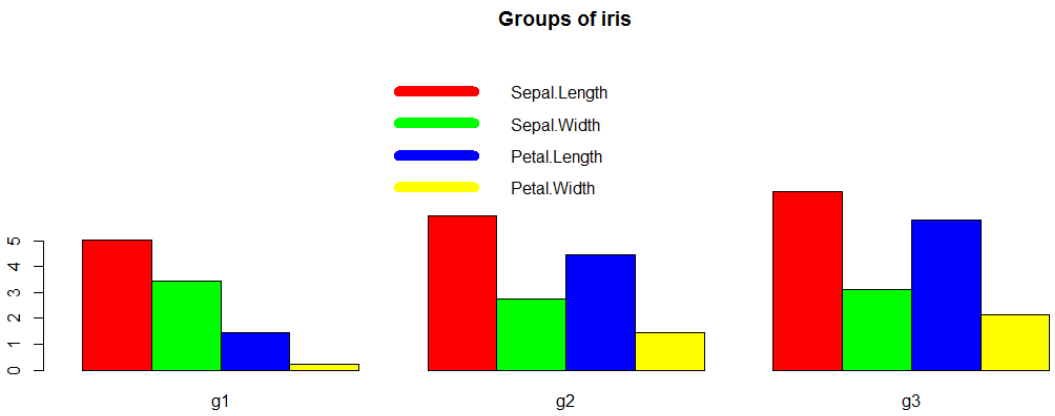


Рис. 5.8. Три группы ирисов с легендой

Шаг 2.7. **Кластеризация ирисов k-means:**

```
km.res <- kmeans(iris.scaled, 3, nstart = 10)
fviz_cluster(km.res, iris[, -5], ellipse.type = "norm")
```

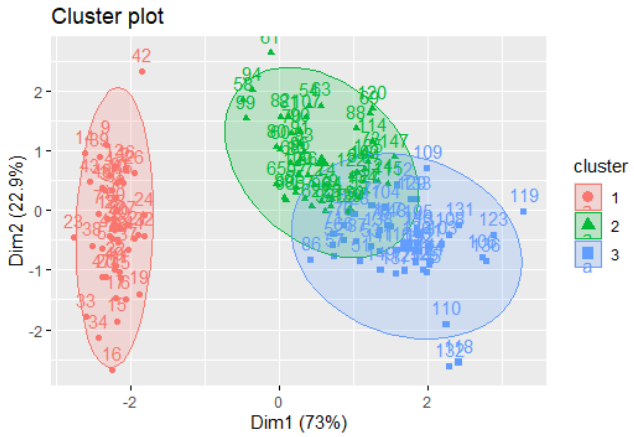


Рис. 5.9. Три группы ирисов с легендой

```
# Change the color palette and theme
fviz_cluster(km.res, iris[, -5],
             palette = "Set2", ggtheme = theme_minimal())
```

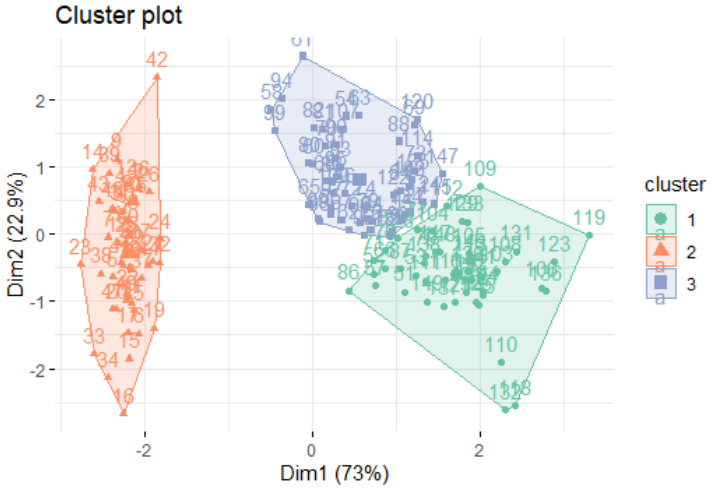


Рис. 5.10. Три группы ирисов с легендой, изменена палитра и эллипсоидное очерчивание

Плюсы и минусы кластеризации K-средних

Кластеризация K-средних предлагает следующие преимущества:

- Это быстрый алгоритм.
- Он может хорошо обрабатывать большие наборы данных.

Тем не менее, он имеет следующие потенциальные недостатки:

- Это требует от нас указать количество кластеров перед выполнением алгоритма.
- Он чувствителен к выбросам.

Двумя альтернативами кластеризации k-средних являются кластеризация k-medoids и иерархическая кластеризация.

Справка по функции fviz_cluste: https://search.r-project.org/CRAN/refmans/factoextra/html/fviz_nbclust.html

Первичное представление о многомерной классификации можно также получить с помощью библиотеки lattice (решетка).

Функция R **xyplot** () этой библиотеки используется для создания двумерных диаграмм рассеяния или графиков временных рядов. Упрощенный формат выглядит следующим образом:

```
library (lattice)
xyplot(y ~ x, data)
```

Итак, чтобы создать диаграмму рассеяния, используем функцию xyplot (формула, данные). Чтобы построить решетчатый график, нужно указать как минимум два аргумента:

- **формула:** как правило, представляется в виде y ~ x | Z. Это означает, что мы хотим создать график зависимости ***y*** от ***x*** с условием ***z*** . Другими словами, создаем график для каждого уникального значения ***z*** . Каждая из переменных в формуле должна быть столбцом во фрейме данных, который мы указываем в аргументе данных.
- **данные:** фрейм данных, который содержит все столбцы, указанные в аргументе формулы.

Снова возьмем для экспериментов датасет ИРИСЫ:

```
my_data <- iris
head(my_data)
```

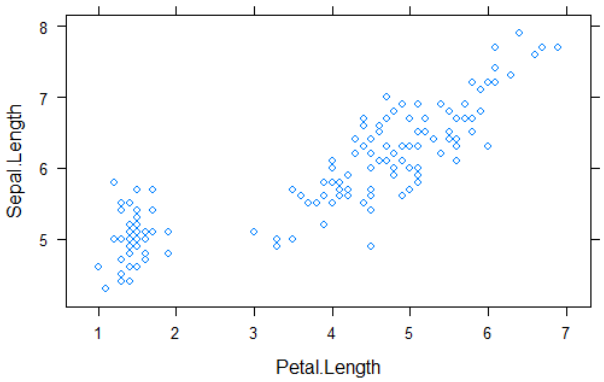
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

Наша классификация будет содержать 3 вида (кдасса) цветов: "setosa" "versicolor" "virginica"

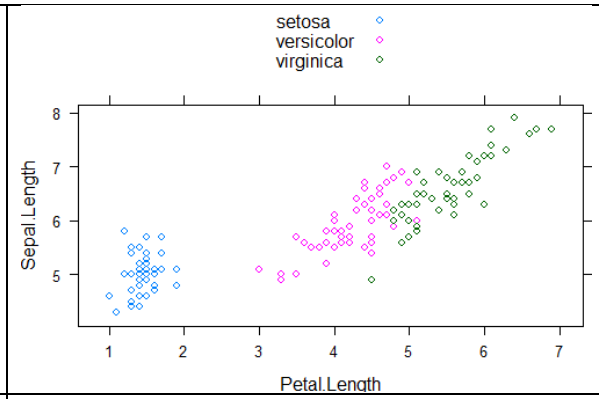
и выведем график рассеяния с минимальным количеством параметров:

```
xyplot(Sepal.Length ~ Petal.Length, data = my_data)
```

получим распределение элементов нашей выборки в зависимости: длина чашелистика от длины лепестка (рис. 5.10 (а))



а) без выделения сортов



б) с выделением сортов

Рис. 5.11. Зависимость длины чашелистика от длины лепестка в наборе данных "Ирисы"

Для варианта б) нужно добавить параметр `group = Species`:

```
xyplot(Sepal.Length ~ Petal.Length, group = Species, data = my_data, auto.key = TRUE)
```

очевидно, что этот параметр поможет нам визуально отличить группы друг от друга (параметр `auto.key = TRUE` добавит легенду цветов сверху от графика).

Как видим, здесь тоже уже понадобилось знание о возможных кластерах или классах.

Шаг 3. Дополнительная аналитика

Построим боксплот, который поможет нам убедиться, что мы действительно имеем 3 класса существенно отличных друг от друга:

```
boxplot(Sepal.Length ~ Species, data = iris, ylab = "Sepal.Length",
+         frame = FALSE, col = "lightgray")
```

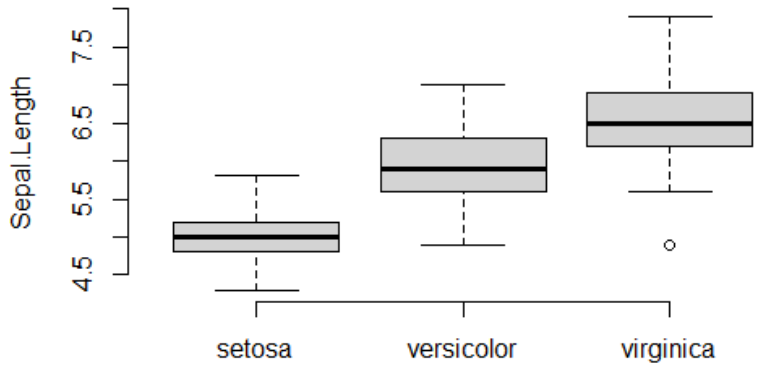


Рис 5.12. Боксплот, отражающий характеристики классов цветов

Построим график, который поможет нам увидеть, как разные предикторы вашего датасета зависят друг от друга:

```
pairs(iris[,-5])

pairs(iris[,-5],main= "Ирисы", col = c("red","green","blue"))

my_cols <- c("#00AFBB", "#E7B800", "#FC4E07")
pairs(iris[,-5],main= "Ирисы по сортам",pch = 19,  cex = 0.8,
      col = my_cols[iris$Species],
```

```
lower.panel=NULL)
```

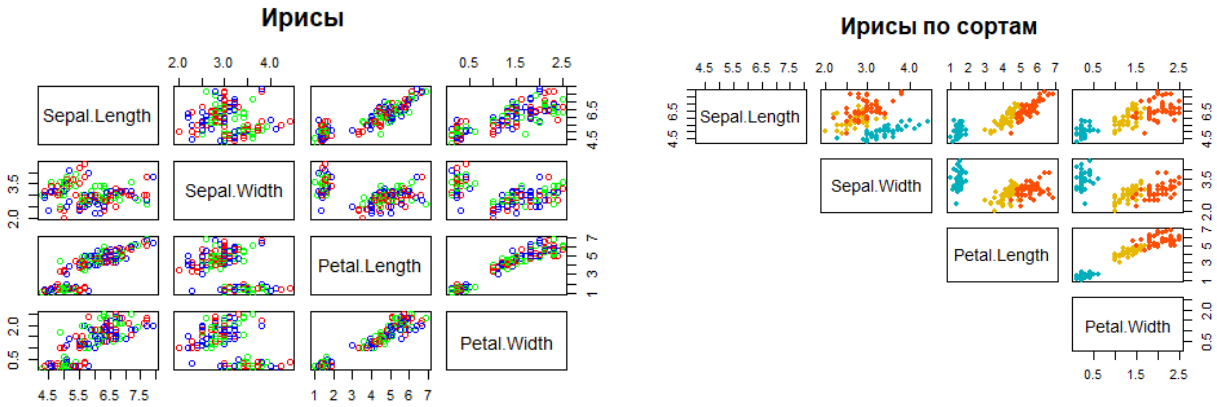


Рис 5.13.Скаттерплот, отражающий характеристики классов ирисов

Парный график, также известный как матрица диаграммы рассеяния, представляет собой сетку диаграмм рассеяния, которая отображает попарные связи между несколькими переменными в наборе данных. Каждая ячейка сетки представляет связь между двумя переменными, а диагональные ячейки отображают гистограммы или графики плотности ядра отдельных переменных. Парные графики невероятно универсальны и помогают нам выявлять закономерности, корреляции и потенциальные выбросы в наших данных.

Теперь, когда у нас есть график основных пар, давайте разберемся, как его интерпретировать:

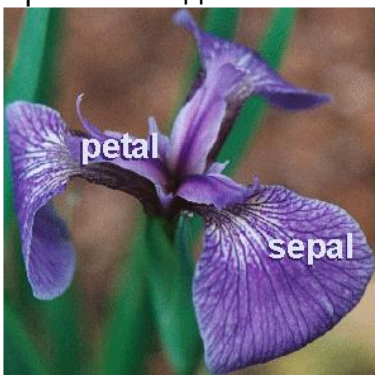
1. **Диагональные графики**: диагональные ячейки отображают гистограммы (или графики плотности) отдельных переменных. Эти графики показывают распределение каждой переменной в наборе данных.
2. **Внедиагональные графики**: Внедиагональные ячейки содержат диаграммы рассеяния, которые показывают взаимосвязь между парами переменных. Каждая точка на этих диаграммах рассеяния представляет собой точку данных в наборе данных.
    - **Закономерности**: ищите закономерности или тенденции на диаграммах рассеяния. Например, группируются ли точки вместе, что указывает на сильную корреляцию?
    - **Корреляции**: обратите внимание на общее направление точек. Они движутся вверх или вниз? Это может дать вам представление о силе и направлении корреляции.
    - **Выбросы**: определите любые выбросы или точки данных, которые значительно отклоняются от основного кластера. Выбросы могут указывать на ошибки или интересные случаи в ваших данных.

Подробности здесь: https://www.r-bloggers.com/2023/09/mastering-data-visualization-with-pairs-plots-in-base-r/

Шаг 4.

Задача:
Определить принадлежность цветка ирис к одному из трех видов (классов) по геометриче ским параметрам чашелистиков и лепестков:

Sepal—чашелистик,
Petal—лепесток,
Species—вид.



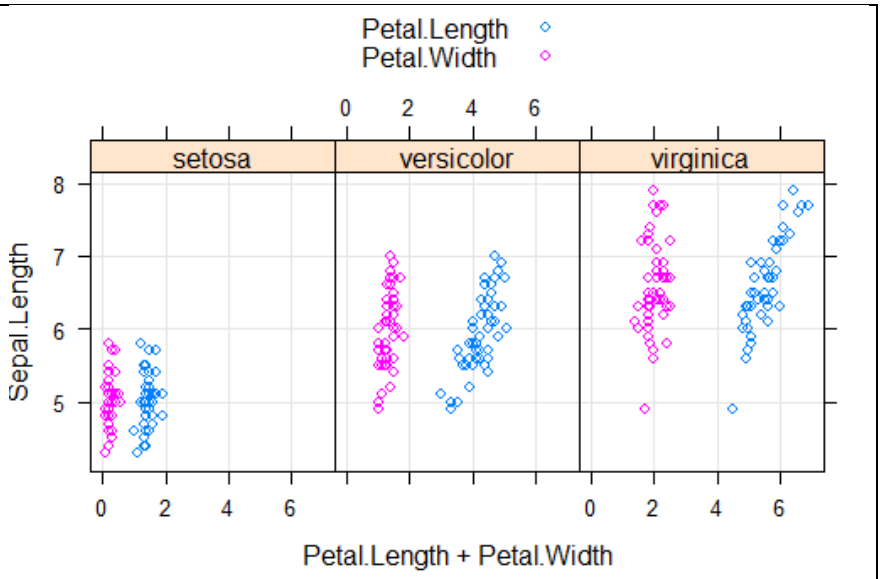


Рис 5.14. График, классифицирующий ирисы согласно их геометрическим параметрам.

Для получения такой классификации модифицируем параметры функции еще раз:

```
xyplot(Sepal.Length~Petal.Length+Petal.Width|Species,data=iris, grid = T, auto.key=TRUE)
```

Мы изменили формулу, указав, что длина лепестков, как мы считаем, зависит от длины и ширины чашелистиков и видов цветов.

***) Синтаксис функции для группировки данных:** y ~ x | group.

Добавим линию сглаживания:

```
xyplot(
  Sepal.Length ~ Petal.Length | Species,
  layout = c(3, 1),                 # panel with ncol = 3 and nrow = 1
  group = Species, data = iris,
  type = c("p", "smooth"),          # Show points and smoothed line
  scales = "free"                   # Make panels axis scales independent
  )
```

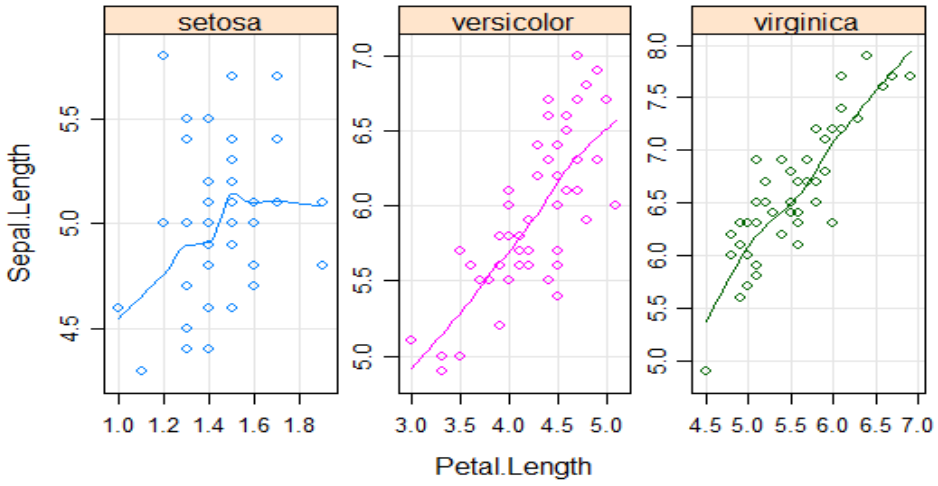


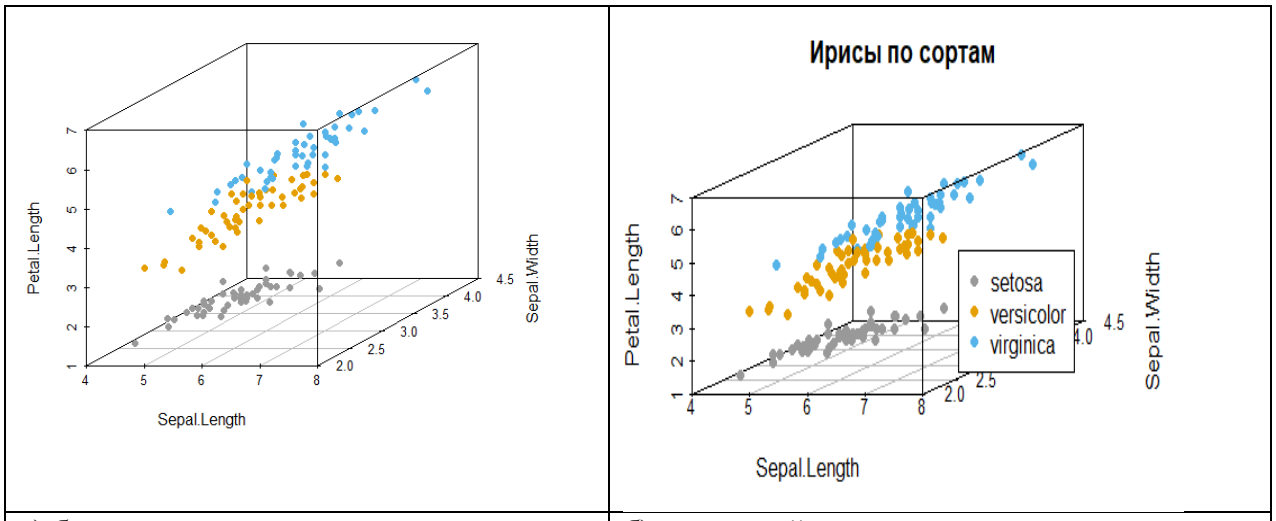
Рис 5.15. График, классифицирующий ирисы согласно их геометрическим параметрам с линией сглаживания

Шаг 5. Построим трехмерный график наших классов

```
>install.packages("scatterplot3d")
library("scatterplot3d")
```

Трехмерная классификация ирисов:

```
> colors <- c("#999999", "#E69F00", "#56B4E9")
> colors <- colors[as.numeric(iris$Species)]
> s3d <- scatterplot3d(iris[,1:3], main= "Ирисы по сортам", pch = 16, color=colo
rs)
legend(s3d$xyz.convert(7.5, 3, 4.5), legend = levels(iris$Species),
       col =  c("#999999", "#E69F00", "#56B4E9"), pch = 16)
```



а) без легенды | б) с легендой

Рис 5.15. Трехмерный график, классифицирующий ирисы согласно их геометрическим параметрам (scatterplot3d).

Информация по scatterplot3d: http://www.sthda.com/english/wiki/scatterplot3d-3d-graphics-r-software-and-data-visualization

Шаг 6. Кластеризация k-means (ggplot)

Для построения этого графика необходимо, чтобы в датасете были ясны зависимости между предикторами: в примере с Ирисами – это Petal.Length от Petal.Width.

```
#проанализируем датасет iris с помощью функций из прогрессивного пакета ggplot2

packages <- c('ggplot2', 'dplyr', 'tidyr', 'tibble')
# install.packages(packages)
library(ggplot2)
library(dplyr)
library(tidyr)
library(tibble)
iris %>%
  ggplot(aes(Petal.Length, Petal.Width, color = Species))+geom_point()
```

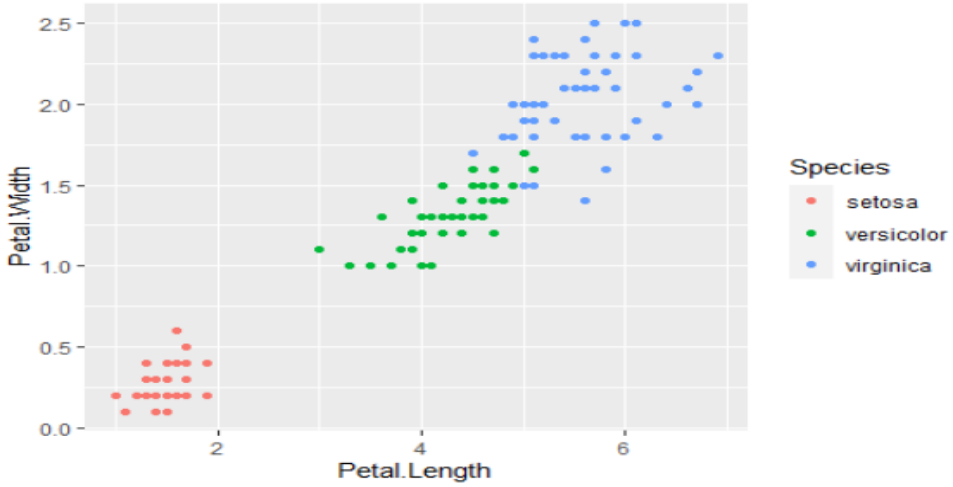


Рис. 5.17. Зависимость ширины чашелистика от длины чашелистика в наборе данных "Ирисы".

*) ***Функция aes().*** Эстетические сопоставления описывают, как переменные в данных сопоставляются с визуальными свойствами (эстетикой) геометрических объектов. Эстетические сопоставления могут быть установлены в ggplot() и в отдельных слоях.

*) ***Функция geom_point()*** используется для создания диаграмм рассеяния. Диаграмма рассеяния наиболее полезна для отображения взаимосвязи между двумя непрерывными переменными. Его можно использовать для сравнения одной непрерывной и одной категориальной переменных или двух категориальных переменных. Функция `geom_point()` накладывает следующий слой, который состоит из точек. Эта функция имеет обязательный аргумент `mapping`, в котором вы должны указать оси (переменные) для вашего графика. Существует целое семейство функций, которые имеют название `geom_xxx`.

Далее идет много функций, которые нужно глубоко изучать, поэтому дальнейший разбор показан только для примера.

```
#Теперь выполним кластеризацию с помощью алгоритма k-средних
# и функций из прогрессивного пакета ggplot2
#нужно использовать функцию kmeans(), указав количество кластеров в centers:

library(broom)
set.seed(42)
iris %>%
  select(Petal.Length, Petal.Width) %>%
  kmeans(centers = 3) ->  km
```

Посмотрим, насколько хорошо алгоритм k-средних справился с заданием. Воспользуем ся функцией augment() из пакета broom, чтобы добавить результаты модели к исходным данным (это работает и с регрессиями).

augment () - принимает объект модели и набор данных и добавляет информацию о каждом на блюдении в наборе данных. Чаще всего это включает прогнозируемые значения в столбце .fit ted, остатки в столбце .resid и стандартные ошибки для подогнанных значений в столбце .se.f it. Новые столбцы всегда начинаются с а . префикс, чтобы избежать перезаписи столбцов в и сходном наборе данных.

```
km %>%
  augment(iris) %>% count(Species, .cluster)
```

```
# A tibble: 5 x 3
  Species    .cluster     n
  <fct>      <fct>    <int>
1 setosa     1           50
2 versicolor 2           48
3 versicolor 3            2
4 virginica  2            4
5 virginica  3           46
```

Мы видим, что алгоритм все разбил на три кластера (1, 2, 3), 1 соответствует setosa, 2 соответствует versi-color, 3 соответсвтует virginica (смотрим с какой группой ассоциировано наибольшее n).

*) ***count()*** позволяет быстро подсчитать уникальные значения одной или нескольких переменных: df %>% count(a, b)

Воспользуемся функцией recode_factor() для того чтобы перекодировать переменную .cluster:

```
km %>%
  augment(iris) %>%
  mutate(.cluster = recode_factor(.cluster,
                                  `1` = "setosa",
                                  `2` = "versicolor",
                                  `3` = "virginica"),
         correct = Species == .cluster) %>%
  ggplot(aes(Petal.Length, Petal.Width))+
  geom_point(aes(color = correct, shape = Species))+
  geom_point(data = data.frame(km$centers)) # центроиды
```

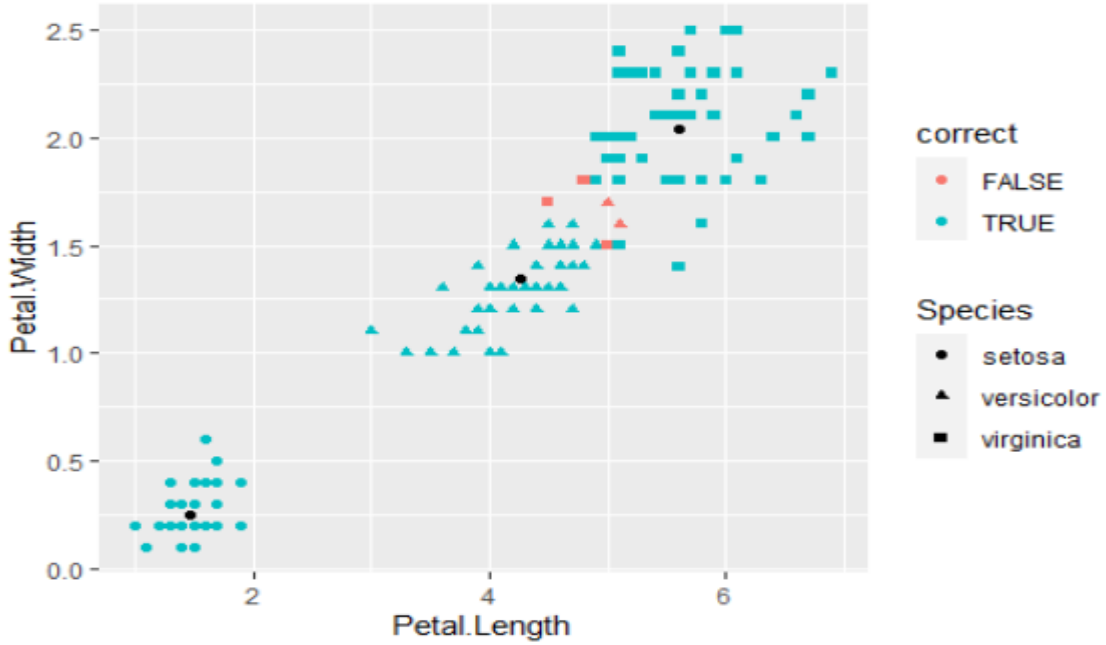


Рис. 5.18. Кластеризация в наборе данных "Ирисы" с построением центроидов кластеров и выделением ложных элементов.

Цветом выделены несовпадения с исходными данными, как видно, таких случаев всего 5: два цветка `virginica` были отнесены к классу `versicolor`, три цветка `virginica` были отнесены к `versicolor`. Так что в целом, можно сказать, что алгоритм хорошо справился. Черным обозначены центроиды получившихся кластеров.

5.4. Формулировка задания:

За наборами данных обращаться к преподавателю.

Общая схема выполнения ***ЛР 5:***

Часть первая - (ЛР 5.1) - предназначена для разведочного анализа данных и определения количества результирующих выходных групп.

1. В начале отчета необходимо разместить формулировку задания и фрагмент исходного датасета.
2. Выполнить дескриптивный анализ данных (здесь приветствуются дополнительные исследования).
3. Оценить оптимальное число кластеров, для этого построить диаграмму "Метод силуэта", “Метод локтя”, "Статистику разрыва" и Алгоритм консенсуса.
4. Выполнить иерархическую кластеризацию вашего набора данных, построив **дендрограмму**. Подробно обосновать Ваш выбор числа групп.
5. Построить диаграмму со столбчатыми диаграммами (рис. 5.8) и боксплотами групп (рис. 5.12). Провести сравнительный анализ полученных групп.
6. Выполнить кластеризацию своего датасета по k-means (рис.5.9, 5.10).
7. Выполнить построение scatterplot (рис. 5.13) с помощью функций plot или pairs.
8. Построить трехмерную кластеризацию по scatterplot3d (5.16)
9. В целом: выполнить шаги 1-3,5 анализа для своего набора данных (если какие-то из шагов нерелевантны вашему набору данных, объяснить почему).

Часть вторая - (ЛР ***5.2***) - использует предположение о количестве выходных классов Вашего набора данных, классификация строится из этого предположения – см следующую ЛР.

№	Задание		Параметры
1.	Экономика городов	15	Football
2.	Занятость в странах Европы	16	Крушение самолетов
3.	French_Food – расходы на питание во Франции	17	Школьники
4.	Исследование Пищевых Предпочтений 2019 Года	18	ЗП_в _Америке
5.	Страны_по_уровню жизни	19	Физическая активность в США
6.	Классификация филиалов сети магазинов	20	Покупатели магазина
7.	Туристические поездки	21	Фитнес-часы Индия
8.	Титаник	22	Кредиты
9.	Темпы роста стран	23	Интернет-компании
10.	Covid_Russia	24	Сердечные заболевания
11.	Шоколад	25	Резервирование отелей
12.	Меню ресторана	26	Заемы на недвижимость
13.	GooglePlay	27	Кандидаты на работу
14.	Выборы в США	28	Оценки учащихся

Варианты:

№	ФИО	Вар №
1.	Алмаева Анастасия Ильинична	11
2.	Блягоз Амаль Хазретович	20
3.	Вавакин Владислав Олегович	5
4.	Воробьев Артем Олегович	1
5.	Гаджиев Ахмед Русланович	3
6.	Задикян Аветис Арутюнович	19
7.	Иванченко Павла Андреевна	*17*
8.	Королев Сергей Юрьевич	22
9.	Лотарев Сергей Юрьевич	7
10.	Мазуренко Артём Алексеевич	4
11.	Матюха Филипп Андреевич	2
12.	Пономарь Дарья Сергеевна	21
13.	Сидоренко Александр Андр	26
14.	Смирнов Никита Олегович	6
15.	Стройный Александр Алекс	23
16.	Чеуж Асиет Асланбиевна	28
17.	Агаджанян Алёна Самвеловна	9
18.	Аникин Марк Андреевич	15
19.	Бачурин Иван Алексеевич	16
20.	Борисов Никита Алексеев	14
21.	Воробьев Игорь Александр	18
22.	Гаранина Людмила Виталь	10
23.	Григоренко Никита Алексе	27
24.	Искрич Александр Алексан	24
25.	Косенко Данил Сергеевич	13
26.	Масенко Мария Сергеевна	25
27.	Миков Никита Сергеевич	8
28.	Небывалов Максим Алекса	7
29.	Пикулев Андрей Сергеевич	5
30.	Решетка Даниил Владим	9
31.	Сгонник Николай Сергее	2
32.	Таран Владимир Сергеевич	3
33.		12

№	ФИО	Вар №
1.	Абраамян Кристина Гургеновна	25
2.	Барсука Жамал (бюджет)	4
3.	Дзамихов Залим Виктор	2
4.	Казарян Вероника Григор	18
5.	Кличенко Давид Артурович	21
6.	Козлов Эдуард Дмитриевич	20
7.	Кочнев Владимир Юрьевич	19
8.	Крапоткин Максим Василь	3
9.	Краснослободцева Екатерина Олеговна	6
10.	Синьков Дмитрий Вадим	7
11.	Сластенов Константин Вяче	8
12.	Усков Артём Алексеевич	22
13.	Цобехия Данил Темурович	5
14.	Шаблин Никита Дмитриев	12
15.	Арутюнян Артур Камоевич	9
16.	Арутюнян Рафаэль Гарегин	10
17.	Белокобыльский Богдан Витал	21
18.	Вебер Артем-Дариус Алексе	26
19.	Волков Андрей Александ	7
20.	Дерябин Андрей Виктор	13
21.	Дудо Сергей Николаевич	1
22.	Дука Виталий Андреевич	28
23.	Журавлёв Даниил Дмитриев	17
24.	Логвина Анна Владимир	27
25.	Маркелов Владислав Андреев	16
26.	Осипов Василий Романович	22
27.	Пшеничнов Андрей Алекс	23
28.	Чертоусов Владимир Сергеев	24
29.		
30.		
31.		