**Описательные статистики (по файлу «Descriptive statistics.R»)**

**#Предобработка данных**

Смотрим инфо по данным:

?mtcars

Сохраняем:

df <- mtcars

Смотрим типы:

str(df)

Переводим переменные «Тип двигателя (vs)» и «Коробка передач(am)» в более наглядный вид:

df$vs <- factor(df$vs , labels = c("V", "S"))

df$am <- factor(df$am , labels = c("Auto", "Manual"))

Проверяем, изменился ли тип:

str(df)

**#Описательные статистики**

Медиана:

median(df$mpg)

Среднее:

mean(df$disp)

Стандартное отклонение:

sd(df$hp)

Размах:

range(df$cyl)

Сохраняем среднее в переменную

mean\_disp <- mean(df$disp)

Считаем среднее по переменной mpg только для машин с 6 цилиндрами:

mean(df$mpg[df$cyl == 6])

С 6 цилиндрами и типом двигателя V

mean(df$mpg[df$cyl == 6 & df$vs == "V"])

Стандартное отклонение по переменной лошадиные силы для машин с количеством цилиндров отличным от трёх и автоматической коробкой передач

sd(df$hp[df$cyl != 3 & df$am == "Auto"])

**#Агрегация – рассчитывает суммарные описательные статистики некоторого подмножества**

Справка:

?aggregate

Среднее по переменной лошадиные силы, сгруппированной по переменной тип двигателя:

mean\_hp\_vs <- aggregate(x = df$hp, by = list(df$vs), FUN = mean)

Смотрим в редакторе:

View(mean\_hp\_vs)

Задаём понятные названия столбцов:

colnames(mean\_hp\_vs) <- c("VS", "Mean HP")

Смотрим в редакторе:

View(mean\_hp\_vs)

Тот же результат можно получить в более компактном виде: разбиваем hp по vs, указываем какие данные берём и что считаем:

aggregate(hp ~ vs, df, mean)

Группируем по двум переменным vs и am:

aggregate(hp ~ vs + am, df, mean)

Тоже самое:

aggregate(x = df$hp, by = list (df$vs, df$am), FUN = mean)

Считаем медиану по всем переменным кроме 8 и 9, группируя по типу коробки передач:

aggregate(x = df[,-c(8,9)], by = list(df$am), FUN = median)

Стандартное отклонение по двум переменным, группируя по типу коробки передач и типу двигателя:

aggregate(df[,c(1,3)], by = list(df$am, df$vs), FUN = sd)

Тоже самое, но более компактно в виде формулы:

aggregate(cbind(mpg, disp) ~ am + vs, df, sd)

Сохраняем результат:

my\_stats <- aggregate(cbind(mpg, disp) ~ am + vs, df, sd)

**#Библиотека "psych"**

Установим библиотеки psych (психометрические исследования) и ggplot2 (визуализация графиков), выберем пункт Tools-> Install Packages, далее вписываем названия пакетов psych, ggplot2 и жмём Install

Открываем библиотеку (появится на вкладке Packages):

library(psych)

Справка по базовым описательным статистикам:

?describe

Описательные статистики по всем переменным датафрейма:

describe(x = df)

Только по количественным переменным:

descr <- describe(x = df[,-c(8,9)])

Описательные статистики по группам (по типу двигателей, по количественным переменным):

descr2 <- describeBy(x = df[,-c(8,9)], group = df$vs)

Результаты по V-образному двигателю:

descr2$V

Результаты по S-образному двигателю:

descr2$S

Добавляем параметр матрица, чтобы получить данные в виде датафрейма, и ограничим количество выводимых знаков после запятой одним символом:

descr2 <- describeBy(x = df[,-c(8,9)], group = df$vs, mat = T, digits = 1)

Более компактный вариант статистик, добавим fast:

descr3 <- describeBy(x = df[,-c(8,9)], group = df$vs, mat = T, digits = 1, fast = T)

По двум группирующим переменным:

describeBy(df$qsec, group = list(df$vs, df$am), digits = 1, fast = T)

**#Пропущенные значения**

Считаем количество пропущенных значений во всём датафрейме, также можно посчитать по каждой переменной

sum(is.na(df))

Добавляем пропущенные значения в переменную mpg:

df$mpg[1:10] <- NA

Указываем при подсчете среднего, что нужно удалять пропущенные:

mean(df$mpg, na.rm = T)

Функция агрегирования по умолчанию игнорирует пропущенные:

aggregate(mpg ~am, df, sd)

По умолчанию также игнорируются пропущенные, но если установить True, то будут удалены все строки, где есть хоть одно пропущенное:

describe(na.rm = )

**Построение графиков (файл «Plots.R»)**

**#Основные графики**

Считываем датафрейм и делаем переменные категориальными

df <- mtcars

df$vs <- factor(df$vs , labels = c("V", "S"))

df$am <- factor(df$am , labels = c("Auto", "Manual"))

Строим гистограмму переменной с заданием количества столбцов гистограммы и подписью оси x

hist(df$mpg, breaks = 20, xlab = "MPG")

Ящик с усами по переменной mpg, сгруппированный по типу коробки передач, заданный через формулу (с подписью оси y):

boxplot(mpg ~ am, df, ylab = "MPG")

График взаимосвязи между двумя переменными:

plot(df$mpg, df$hp)

Построение графиков с помощью дополнительных характеристик:

hist(df$mpg, breaks = 20, xlab = "MPG", main ="Histogram of MPG",

col = "green", cex.lab = 1.3, cex.axis = 1.3)

plot(density(df$mpg), xlab = "MPG", main ="Density of MPG",

col = "green", cex.lab = 1.3, cex.axis = 1.3)

boxplot(mpg ~ am, df, ylab = "MPG", main ="MPG and AM",

col = "green", cex.lab = 1.3, cex.axis = 1.3)

boxplot(df$mpg[df$am == "Auto"], df$mpg[df$am == "Manual"], ylab = "MPG", main ="MPG and AM",

col = "green", cex.lab = 1.3, cex.axis = 1.3)

plot(df$mpg, df$hp, xlab = "MPG", ylab ="HP" , main ="MPG and HP", pch = 22)

plot(~ mpg + hp, df)

**#С помощью библиотеки ggplot2**

Подключаем библиотеку:

library(ggplot2)

Строим график, через aes всегда указываем переменную, а через geom\_histogram указываем тип отображения гистограмма (слои или уровни). Цвет заполнения столбцов гистограммы укажем белый, а цвет границ – черный, ширина столбцов 2 и выставим подписи:

ggplot(df, aes(x = mpg))+

geom\_histogram(fill = "white", col = "black", binwidth = 2)+

xlab("Miles/(US) gallon")+

ylab("Count")+

ggtitle("MPG histogram")

Зададим в качестве типа отображения точечный график распределения, также укажем разбиение наблюдений по группе тип коробки передач:

ggplot(df, aes(x = mpg, fill = am))+

geom\_dotplot()+

xlab("Miles/(US) gallon")+

ylab("Count")+

scale\_fill\_discrete(name="Transmission type")+

ggtitle("MPG dotplot")

График плотности:

ggplot(df, aes(x = mpg))+

geom\_density(fill = "red")

График плотности в зависимости от типа коробки передач, параметр alpha задает прозрачность:

ggplot(df, aes(x = mpg, fill = am))+

geom\_density(alpha = 0.5)+

xlab("Miles/(US) gallon")+

ylab("Count")+

scale\_fill\_discrete(name="Transmission type")+

ggtitle("MPG density plot")

Ящик с усами переменной лошадиные силы по типу коробки передач, с заполнением по типу двигателя:

ggplot(df, aes(x = am, y = hp, fill = vs))+

geom\_boxplot()+

xlab("Transmission type")+

ylab("Gross horsepower")+

scale\_fill\_discrete(name="Engine type")+

ggtitle("Gross horsepower and engine type")

Диаграмма рассеяния по двум переменным с размером точек qsec (время работы двигателя):

ggplot(df, aes(x = mpg, y = hp, size = qsec))+

geom\_point()+

xlab("Miles/(US) gallon")+

ylab("Gross horsepower")+

scale\_size\_continuous(name="1/4 mile time")+

ggtitle("Miles/(US) gallon and Gross horsepower")

Сохраняем график в переменную:

my\_plot <- ggplot(df, aes(x = mpg, y = hp, col = vs, size = qsec))+

geom\_point()

Сохраняем заготовку:

my\_plot2 <- ggplot(df, aes(x = am, y = hp, fill = vs))

Дальше указываем слой:

my\_plot2 + geom\_boxplot()

**Сохранение результатов (по файлу «save\_data.R»)**

Сохраняем датафрейм:

df <- mtcars

Считаем среднее:

mean\_mpg <- mean(df$mpg)

Описательные статистики по количественным переменным:

descr\_df <- describe(df[,-c(8,9)])

Строим ящик с усами:

my\_boxplot <- ggplot(df, aes(x = factor(am), y = disp))+

geom\_boxplot()+

xlab("Transmission")+

ylab("Displacement")+

ggtitle("My boxplot")

Функция getwd(), прописанная в консоли вернёт текущую директорию.

Для установки рабочей директории заходим в меню Session -> Set Working Directory и выбираем директорию, можно также в начале каждого скрипта на всякий случай задавать рабочую директорию с помощью setwd(путь).

Сохранение скрипта через File->Save as…

Сохранение графиков можно выполнять в окне Plots, где они и отображаются, нажав на кнопку Export->Save as Image, там же можно настроить размер графика.

Сохраняем встроенный датасет mtcars в формате .csv:

write.csv(df, "df.csv")

Сохранение результатов описательных статистик:

write.csv(descr\_df, "descr\_df.csv")

Сохранение переменной:

my\_mean <- mean(10^6 : 10^7)

Можем сохранить все переменные рабочей области на вкладке Environment, нажав на значок сохранения.

Можно сохранить в файл отдельную переменную прописав команду:

save(my\_mean, file = "my\_mean.RData")